DATA-DRIVEN FINE MANIPULATION

LIYIMING KE

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington
2024

Reading Committee:
Siddhartha S. Srinivas, Chair
Abhishek Gupta
Sanjiban Choudhury

Program Authorized to Offer Degree:
Paul G. Allen School of Computer Science & Engineering

University of Washington

ABSTRACT

DATA-DRIVEN FINE MANIPULATION

Liyiming Ke

Chair of the Supervisory Committee:
Professor Siddhartha S. Srinivas
Paul G. Allen School of Computer Science & Engineering

We are driven to build robots that operate *outside specialized factories* and automate *precision-critical* tasks with sub-millimeter accuracy, such as assistive homecare, surgery, harvesting, and agile manufacturing. We ask, how to enable general-purpose robots to *learn, adapt, and improve fine motor skills in uncontrolled environments*?

Current robotic systems tend to be either precise or general, but rarely both. Robots designed for specific use cases rely on controller environments and specific tooling to offer high precision, but are costly to deploy to novel tasks in less structured settings. Conversely, the learning methods for developing general systems have not demonstrated necessary granularity for sub-millimeter accuracy. Challenges for robots to generalize to high precision tasks can come from both the robot and the data: inaccuracy in execution, scarcity of data and suboptimality of demonstrations. We emphasize the need for general-purpose robots capable of fine manipulation in diverse and uncontrolled environments.

This dissertation describes three steps that sought to make more practical and generalizable robotic fine manipulator by developing data-driven algorithms and systems. The first part introduces a general-purpose robot platform for fine manipulation and uses it as a testbed to characterize the challenge. The second project learns from expert data via imitation learning but concerns the compounding errors problem - small execution or prediction mistakes accumulate along trajectory, leading to increasingly significant deviations from expert data. The third project explores learning beyond expert data in a data-efficient manner to address difficult tasks whose complexity limits the data quantity and quality.

An emergent theme through these works is that, while we aim to develop data-driven methods to address fine manipulation challenges in a general way, leveraging structures and priors, such as dynamics and constraints, significantly enhances the efficiency and robustness of the learning process. We conclude with a conceptual analysis of moving data-driven robotics forward - as models and data keep improving, what key challenges would likely persists and what structures can be useful.

天高地迥，觉宇宙之无穷；

兴尽悲来，识盈虚之有数。

王勃《滕王阁序》

The sky soars high, the earth stretches wide,
revealing the endless expanse of the universe;
Through joy and sorrow,
one comprehends the cyclical nature of gain and loss.

Bo Wang, 649 - 676 A.D.

ACKNOWLEDGMENTS

I am immensely thankful to everyone who has supported me and shaped my perspective, on this long and winding journey. I apologize to those I missed, as I know I inevitably have, but I cherish the footprints each and every encounter has left on my life.

I would like to begin by thanking my advisors: Siddhartha Srinivasa, for guiding me into robotics, providing enthusiastic support for my personal and mental growth, sharing elegant teaching, presentations, theories, and visualizations, enlightening me about the beauty of simplicity and pushing me to seek a higher standard in research. Thanks to Abhishek Gupta, for being a relentless advocate for my success, encouraging me to step out of my comfort zone and seek fundamental insights, and inspiring me to better integrate my research with the community.

I thank all the members of my doctoral supervisory committee: Sanjiban Choudary, for showing me the way to research and shaping the foundation of my work; Byron Boots, for thoughtful discussions and motivating me to think outside the box. Thanks to Ashis Banerjee and Karen Leung for their helpful advice on my proposal and defense presentations.

I am grateful to receive additional mentorship from Tapomayukh Bhattacharjee, whose kindness motivates me to be a better mentor; Yonatan Bisk, who helped me navigate the research landscape; Vikash, whose research insights enlighten my path. I thank the support and guidance from Lilian Ratliff, Aravind Rajeswaran, Christoforos Mavrogiannis, Xiujun Li, Yejin Choi, Jianfeng Gao, Kendall Lowrey, Jesse Thomason and Matt Barnes.

To my undergraduate mentors who introduced me to research: Bo Li and Yevgeniy Vorobeychik, I would not be here today without your supervision. I appreciate the precious opportunity you give me to explore research for the joy of it. I am committed to extend such invitations of research to others.

To Pedro Domingos, thank you for admitting me to UW and allowing me to find this lifelong passion.

To my collaborators: Andrew, Colin, Ajinkya and Kevin, thank you for teaching me and inspiring my research. Also, I am grateful for the awesome products and support from Dave and Matthew from HEBI Robotics which constitute critical pieces of my research.

To Ari, for being a master of words and sharing your wisdom, you are always an inspiration in my work and language.

To the students I've helped mentor: Jingqiang, Abhay, Gaoyue, Yunchu, Quinn and Octi. It was such a joy to see you grow! Thank you for giving me the opportunity to grow as a mentor; I'm proud to see you thrive.

To all the folks from the Personal Robotics Lab, WEIRD Lab and the UW Robotics groups, my time has been incredibly special: Gilwoo, Brian, AVK, Sherdil, Ethan, Schmittle, Amal, Taylor, Talia, Helen, Bernie, Yunchu, Marius, Sriyash, Mateo, Chuning, Daphne, Patrick, Mohit, Boling, Motoya, Jiafei, Wenhao, Li, Nick, Naveena, Jacob, Sasha, Adam and Selest. You all helped make my grad school experience more fun and changed my perspective in more ways than I can count.

To all the friendships I formed along this journey: Sam, Willie, Rosario, Xuan, Lianhui, Zoey, Peter, Tim, Anqi, Aaron, Mohak, Rowan, Zhutian, and Tyler. The moments we laughed and cried together defined me, and I was lucky to have them.

Finally, none of this would have been possible without the infinite support of my family—not all of whom are bound by blood: my parents, Lixia Zhu and Qiubi Ke; my childhood friend, Liang; and my American family, Stephanie, Tim, and Parker. Thank you for your patience and infinite support while I struggled and learned.

CONTENTS

1 Introduction 1

I Hardware Test Bed and Human Study

2 Hardware Test Bed 4
- 2.1 Actuator and Control . . . 4
- 2.2 Chopsticks and Simple Tools . . . 5
- 2.3 Modeling and Calibration . . . 5
- 2.4 Teleoperation . . . 7

3 Human Study 9
- 3.1 Experiments . . . 10
- 3.2 Analysis . . . 12
- 3.3 Discussion . . . 18

II Learning from Expert Data

4 Imitation Learning for Fine Manipulation 21
- 4.1 Methods . . . 22
- 4.2 Experiments . . . 25
- 4.3 Results . . . 27
- 4.4 Discussion . . . 29

5 Model-based Corrective Imitation Learning 31
- 5.1 Background . . . 32
- 5.2 Preliminaries . . . 34
- 5.3 Continuity-based Corrective Labels . . . 35
- 5.4 Warm-up: Simulation Experiments . . . 39
- 5.5 Real-World Experiment Design . . . 42
- 5.6 Results . . . 45
- 5.7 Discussion . . . 49
- 5.8 Supplementary . . . 52

III Learning beyond Expert Data

6 Reinforcement Learning for Dynamic Fine Tasks 61
- 6.1 Related Works . . . 63
- 6.2 Method . . . 64
- 6.3 Complete System Design for CherryBot . . . 71
- 6.4 Evaluation . . . 73
- 6.5 Discussion . . . 76
- 6.6 Supplementary . . . 77

7 Conclusion 81
- 7.1 Future Direction . . . 81

Bibliography 83

1

INTRODUCTION

Robots can alleviate human burdens by performing intricate tasks that can be difficult even for humans. We focus on the challenge of fine manipulation, which involves meticulous movements to address precision-critical tasks like handling delicate or small objects. Fine manipulation tasks are prevalent in modern life—clasping necklace chains, cutting fingernails, assembling miniature puzzles, and conducting surgery on patients' bodies. Robotic fine manipulation can have broad applications in diverse fields such as healthcare, scientific development, manufacturing, and service industries. This dissertation focuses on pushing the boundary of robotic fine manipulation via data-driven approaches.

Existing approaches to fine manipulation work well in controlled settings but can be costly or infeasible to generalize to less structured environments. Traditional industrial robots are designed for repetitive tasks, demanding specific tooling and structures for alignment and precision, and lack the flexibility to handle tasks outside their dedicated environments. Conversely, tasks such as assembling small electronic components, performing surgical procedures, or assisting individuals with disabilities require *adaptability*. Despite the possibility of adding an incredible amount of structures to control the environment, some features could be impossible to control. Achieving generalizable fine manipulation in robotics necessitates dexterous adaptation strategies to work amidst unstructured environments.

Conversely, general-purpose learning methods have not demonstrated fine manipulation capability at sub-millimeter accuracy with reliability on par with industrial solutions. Benefiting from the recent progress in large language and visual models, robots could use language instructions to ground their actions in real-world tasks or a relatively small amount of data to achieve success on tasks. However, it remains a question how to build general-purpose robots to achieve the necessary granularity and robustness for sub-millimeter accuracy required in fine manipulation tasks.

We preview the challenge of building generalizable systems for robotic fine manipulation that stem from hardware and data. Low-cost hardware often comes with limitations such as actuator backlash and sensor noise, which can significantly hinder their capability for precise control. The challenge of fine manipulation tasks leads to a scarcity of high-quality data. Scaling data-driven methods can be resource-intensive and may not always capture the full range of variability seen in real-world tasks. Robots need to generalize from limited training data to a wide range of tasks and environments, making it difficult to achieve the desired level of precision and reliability.

Addressing these challenges requires a holistic approach that integrates hardware platforms, sophisticated decision-making algorithms, and robust and sample-efficient machine learning techniques.

This dissertation makes the following contributions:

- Chapter 2 presents our **Hardware Platform**, an accessible platform built from off-the-shelf actuators to use as a test bed for data-driven algorithms and robotic fine manipulation tasks.
- Chapter 3 presents a **Human Study** that characterizes human manipulation strategies and evaluates data collection methods.
- Chapter 4 proposes **Practical modification to behavior cloning** to learn fine manipulation skills from only expert demonstrations.
- Chapter 5 proposes the **Continuity-based Corrective Imitation Learning Framework** to address the compounding errors challenge in imitation learning.
- Chapter 6 presents the **CherryBot Reinforcement Learner Framework** to efficiently learn beyond a given subset of data with limited coverage and quality of the task.

Part I

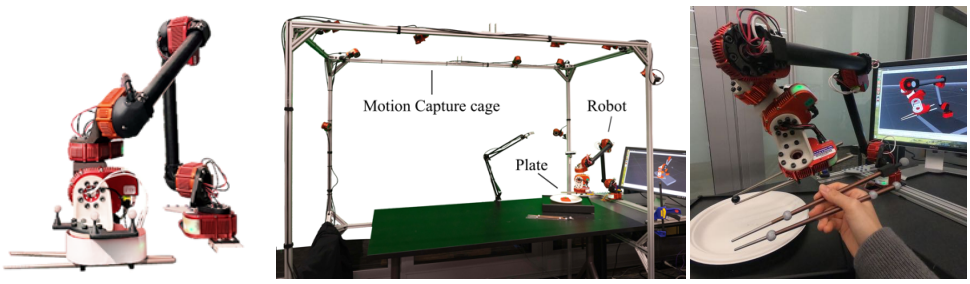
HARDWARE TEST BED AND HUMAN STUDY

2

HARDWARE TEST BED

Robotics is fundamentally grounded in real-world experiments. For our research in fine manipulation, we first built a hardware platform to serve as a test bed for developing algorithms and evaluating autonomous systems. This platform integrates off-the-shelf actuators and components, creating an accessible and reproducible setup. We choose chopsticks as the robot end effector, leveraging their simplicity and natural use in fine manipulation tasks. To leverage human familiarity, we designed a teleoperation interface to enable data collection.

An overview of our hardware platform is shown in Fig. 2.1.



(a) Robot (b) Motion capture (c) Teleoperation

Figure 2.1: Overview of our system

2.1 ACTUATOR AND CONTROL

We build a hardware platform which includes a 6-DOF robot arm equipped with a pair of chopsticks as its end effector (Fig. 2.1a). The robot arm is assembled from HEBI Robotics components (*HEBI Robotics* n.d.). Each joint contains built-in controllers for position, velocity, and torque control that operate at a loop rate of 1KHz.

We observe the state of the robot arm and command its motion via the end-effector(EE) poses. They are 8D vectors containing (1) the x-y-z position of the bottom chopstick tip, (2) a quarternion representation of the rotation of the chopsticks, and (3) the opening angle of the last joint. Each application might request additional information (e.g., regarding the pose of an object to interact with, or an RGB image of the scene). To apply end-effector pose command, we first use an Inverse Kinematics solver to translate the EE pose to joint positions and send the positional command to the PID controller on board of each joint.

2.2 CHOPSTICKS AND SIMPLE TOOLS

Fine manipulation necessities a fine end effector. Roboticists have used both complex tools (such as universal grippers (Brown et al., 2010), vacuum suction tools (Robertson and Paik, 2017), and anthropomorphic hands (Kontoudis et al., 2015; Zhe Xu and Todorov, 2016)) and simple ones (such as parallel-jaw grippers and forks (Bhattacharjee et al., 2019; Mason, Srinivasa, and Vazquez, 2011; Melchiorri and Kaneko, 2016; Monkman et al., 2007; Odhner et al., 2014; Rhodes and Veloso, 2018; Rodriguez, Mason, and Srinivasa, 2014; Yamazaki et al., 2010; Zisimatos et al., 2014)) for manipulation. Complex tools can be customized, while simple tools impose less hardware burden and scale easily, but require adaptive manipulation strategies for different usage scenarios.

We focus on chopsticks, a simple tool that many humans are already familiar with. Chopsticks have widespread use in fine manipulation. Billions of people use chopsticks everyday, demonstrating their versatility and effectiveness in manipulating a variety of objects, from food items to small, rigid objects. One wily and dexterous human even used chopsticks to pick-pocket cellphones (*Picking Cellphones with Chopsticks* n.d.). This makes chopsticks an ideal tool for studying fine manipulation in robotics, providing a simple yet challenging benchmark for developing and testing new techniques. Researchers have adopted the general design of chopsticks for various applications, such as meal assistance (Chang et al., 2007; Yamazaki and Masuda, 2012), surgery (Sakurai, Kanno, and Kawashima, 2016), micro-manipulation (Ramadan et al., 2009), repetitive pick-and-place (Pastor et al., 2011; Vassileva and Boiadjiev, 2009) and articulated-hand design (Chepisheva, Culha, and Iida, 2016; Sugiuchi, Morino, and Terauchi, 2002; Wang and Stephanou, 1988). A chopsticks-wielding robot has a potentially versatile use case to pick up objects of diverse shape, size, weight distribution, and deformation. The design of chopsticks can be easily extended and its versatility may apply to other robots, e.g., a surgical robot with a chopsticks-shaped end effector may effortlessly switch between grasping, moving and rotating various shapes of tissues with different deformability without switching hardware (Sakurai, Kanno, and Kawashima, 2016).

Further, studying the dexterity humans demonstrate in using chopsticks could help us extract insights for developing autonomous manipulators with comparable flexibility to humans.

2.3 MODELING AND CALIBRATION

We document how we build a model for the robot and improves the modeling. Initially, the average EE position error was 10 mm. Controllers had low success rates for reproducing their success of picking up small items placed at fixed position, even just replaying successful trajectories. We highlight one example of inaccuracies in general-purpose gear-based actuator: joint backlash. Since the robot is assembled from parts with joints that are not strictly rigid, inaccuracies can accumulate along robot links. The kinematic model for our joint is not highly accurate and the position errors range from 1 mm to 6 mm meausred at the robot's end effector.

We built a simulation model for the robot via kinematic modeling and system identification procedure. We explain how we used a neural network to predict the residual backlash to achieve position errors < 3 mm at the robot's end effector.

MODELLING THE KINEMATICS Performing such fine-motor skills requires a carefully calibrated kinematics model that reflects the hardware setup in the real world. The robot manufacturer provides a kinematics model, but this (a) does not account for the chopsticks end-effector, and (b) is not tuned to the degree of accuracy that we require. Therefore, we employ data-driven calibration pipelines that allow us to achieve highly accurate position estimates from joint angles.

The Optitrack cage provides precise and accurate pose information, which we can leverage as ground truth data. When we mount a pair of chopsticks to the robot end-effector, one of them is fixed ("primary" chopsticks) and the other one can be rotated by the end effector joint ("moving" chopsticks). By placing a tracker on the tip of the primary chopstick, we can measure the end-effector position in the 3D space.

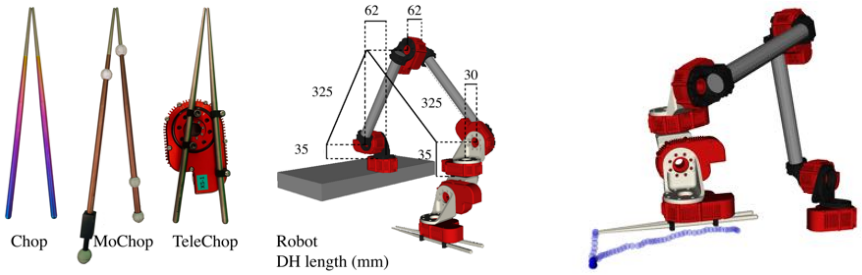
First, we need to measure the robot's base position and orientation on the table. We spin the base joint of the robot for several rotations in both directions. This causes the tracker to trace out a circle several times. By measuring the position and inclination of the circle, we can determine the tilt and position of the robot on the table.

We then, using teleoperation, control the robot to move inside the workspace, recording the end-effector position and the robot joint angles. Later we can then use this data with either a black-box or gradient-based optimizer (using the factory defaults as the initial guess) to solve for a set of accurate parameters for the kinematics model by minimizing the FK loss. To regularize the optimization, we find that it is important to penalize deviating too far from the manufacturer-given kinematics model. We observed the kinematic error being $1.2 \sim 3$ mm at the end effector tip in the task space we tested.

RESIDUAL ESTIMATION TO IMPROVE KINEMATICS To combat inaccuracies in our kinematics, particularly due to backlash in the arm joints, we train a model to predict the backlash in each joint of the arm as a function of the current joint angles. The intuition behind this is that in certain orientations the backlash of the joints may induce errors that are predictable. These dynamics wouldn't be able to be captured by the DH parameters that parameterize the arm kinematics, but a nonlinear function approximator like a neural network would be able to predict this.

The residual estimator network is trained on a dataset of joint angles and ground truth positions (as determined by the mocap cage) collected in and around the workspace. Additionally, the predicted backlash is constrained to be at most 10% more than the figure listed by the manufacturer, to prevent deviating too far from the kinematic model. In practice, we found that this approach reduced the average position error from 1.8mm to 1.5mm, about a 17% reduction, which is significant given the high-precision nature of the task.

SYSTEM IDENTIFICATION To build a simulator with dynamics as close to real as possible, we also employ an optimization-based to fit the dynamic transition in our



(a) Teleoperation (b) Example Demonstration

simulator to the real world Using recorded trajectories of the robot in and around the workspace, we optimize the environment's parameters (centers of mass of joint bodies, friction coefficients, contact solver parameters, etc.) in order to minimize the divergence of simulated rollouts from the ground truth recording. Again, we regularize this optimization by penalizing the divergence of the simulator parameters from the initial guess.

To run the optimization program efficiently, the code leverages the fast computational speed of Julia and the Lyceum MuJoCo wrapper. We do not expect that this optimization pipeline will result in a simulator that matches reality exactly because of the limitation of the simulator: e.g., the shape of the chopsticks and contact parameters are simplified. But it does yield good enough parameters so as to enable simulator pre-training.

2.4 TELEOPERATION

We designed a pair of chopsticks for motion-capture ("MoChop") and an interface for users to teleoperate a robot holding chopsticks ("TeleChop"). Both were adapted from normal chopsticks ("Chop"). See Fig. 2.2a. All methods used the same consumer-grade titanium chopsticks that differ only in colors.

2.4.1 *MoChop*

We 3D printed five light-weight, ball-shaped markers, wrapped them in reflective material and mounted them onto MoChop. To ensure users could still hold the chopsticks, we placed the markers near the tips and tails of chopsticks at different positions on each stick. Note the markers changed the weight distribution and added constraints to finger positions, e.g. users cannot hold the chopsticks at the tail.

To track the motion of MoChop held by users, we used the OptiTrack motion capture system (*OptiTrack* n.d.) with 11 cameras (Fig. 2.1b). The system uses optical reflection to track the position of markers. Its tracking error is up to 0.4mm, and the tracking updates at 100Hz. From the tracked markers' positions, we extracted MoChop's pose.

2.4.2 *TeleChop*

We custom built a 6-DOF robot manipulator, assembled from components provided by HEBI Robotics (*HEBI Robotics* n.d.). We attached the TeleChop to this robot's end-effector. TeleChop has an actuated pair of chopsticks. The two chopsticks operated on the same plane: one was fixed at the actuator's body, and the other is attached at the actuator's output shaft. We used HEBI's X-Series actuators (*HEBI Robotics* n.d.) since they provide built-in controllers for position, velocity, or torque control, running at a loop rate of 1KHz.

2.4.3 *Teleoperation Interface*

To initiate teleoperation, users held MoChop to match TeleChop's orientation. During teleoperation, users moved MoChop (*leader*) to teleoperate the robot mounted with TeleChop (*follower*). This further constrains chopsticks' movement because of the motion-retargeting problem induced by the difference between the human arm and the robot arm. We designed a controller that guide TeleChop to smoothly mimic MoChop's pose. Depending on how the user held MoChop, the two chopsticks may be on different planes. We projected the two chopsticks in MoChop onto the same plane to ensure TeleChop could follow the projected pose. This became the target pose that our controller tracked.

Our controller translated MoChop poses to joint commands for the robot. Upon receiving the desired pose, the controller first computed an Inverse Kinematics solution. However, since smooth chopstick trajectories for humans could require jumps in the robot's joint space, we applied a convolution smoother to get the joint position command, effectively enabling tremor cancelling. We used a position PID controller to follow the joint position command. To add smoothness to the robot's movement, we also supplied a torque command based on gravity compensation and passed it through a torque PID controller. [1] We added both PID controllers' PWM outputs to compose the final command.

1 The amount of torque commanded was based on the mass of the robot as specified on the manufacturer's datasheet (*HEBI Robotics* n.d.)

3

HUMAN STUDY

Human familiarity with fine manipulation opens up the possibility of *easily collecting expert demonstrations*. Despite the wide range of applications involving chopsticks, we are not aware of any prior efforts to learn from human demonstrations for chopstick-based robot manipulation tasks. To this end, we analyze how different interfaces and user expertise levels affect the quality of demonstrations by comparing three data-collection methods (Fig. 2.2a): normal chopsticks ("Chop"), motion captured chopsticks ("MoChop"), and a teleoperation interface ("TeleChop"). We conduct a within-subjects user study with 25 participants and examine how human factors affect the success rate of picking up everyday-life objects.

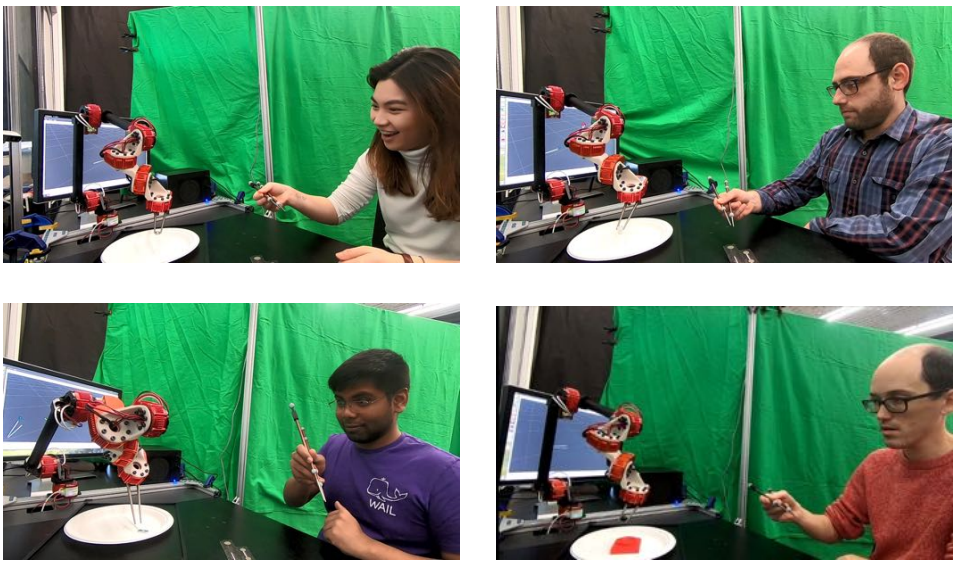


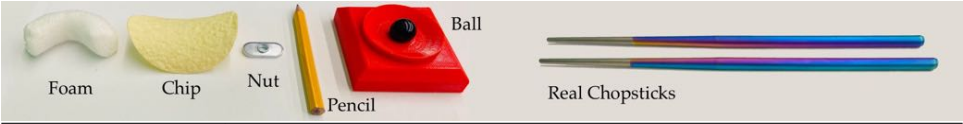
Figure 3.1: Subjects teleoperating a robot holding chopsticks using a motion-capture-marker instrumented chopstick to pick up different objects.

Overall, we make the following contributions:

1. An analysis of human factors underlying user demonstrations in chopstick-based telemanipulation.
2. A comparative evaluation of three different technologies for chopstick-based manipulation across a series of grasping tasks of varying difficulty: normal chopsticks, motion-captured chopsticks, and teleoperated chopsticks.
3. A novel teleoperation interface featuring a custom-built, inexpensive 6-DOF robot manipulator, equipped with a pair of chopsticks attached to its end-effector

and the ability to partially filter vibrations arising from users' grips or motion tracking.

Humans successfully teleoperated our robot to complete a challenging manipulation task: picking up a slippery glass ball with slippery metal chopsticks, without haptic feedback. We believe that teleoperation could be the preferred interface to yield on-hardware demonstrations for robots to learn from (Billard, Calinon, and Dillmann, 2016).



Name	Foam	Chip	Nut	Pencil	Ball
L × W × H (mm)	50×25×20	65×45×15	25×10×3	86×7×6	14×14×14
Texture	rough	rough	slippery	slippery	slippery
Geometry	curved	curved	flat	long	spherical
Deformation	compliant	brittle	hard	hard	hard

Table 3.1: Different objects to pick up.

3.1 EXPERIMENTS

We conducted an in-lab user study with human subjects in which participants performed a series of pick-and-place manipulation tasks using chopsticks on a variety of objects. These experiments aimed to compare human performance on pick-and-place manipulation tasks using TeleChop, MoChop and Chop. We also wanted to explore how humans adapted to different interfaces during this process. Our object set (foam, chip, nut, pencil and ball) for the manipulation tasks included objects with varying levels of chopstick grasping difficulty. See Table 3.1. The user study was conducted in accordance with our University's Institutional Review Board (IRB) review.

3.1.1 *Participants*

We recruited 25 human participants (13 male, 11 female, 1 non-binary of age $M = 27.28$, $SD = 7.89$) for our human-subject studies. The participants had various amount of experience using chopsticks ($M = 14.44$, $SD = 8.79$ years of experience). Fourteen out of 25 subjects reported having experience with teleoperation, but none were previously exposed to our system. To offset individual differences among subjects, we chose a within-subjects design, where all subjects performed the same set of grasping tasks under the same conditions with different orders of tasks.

3.1.2 *Experiment Procedure*

Before beginning the experiments, participant signed a consent form and reported demographic information (procedure approved by the IRB review of the University of Washington). They were informed that the experiment was intended to evaluate their interactions with all three grasping methods. Prior to the recorded trial, participants went through a training procedure. First they held the Chop and then the MoChop, trying to open and close them. They watched the researcher demonstrate how to use TeleChop and then tried to initiate the teleoperation by matching the orientation of MoChop and TeleChop. During training, subjects interacted with Chop, MoChop and Teleop but not with any object in the recorded trials. The subjects finished training by picking up a piece of broken foam *just once* using TeleChop.

Subjects then proceeded to the formal trials to be recorded. All subjects tried to pick up *five* different items using all *three* methods for grasping. Participants manipulated Chop and MoChop to directly pick up items; for TeleChop, they held MoChop to teleoperate the robot, who was holding TeleChop, to pick up items. For each combination of item and method, they had *three* trials. Each subject therefore contributed 45 trials in total (5 objects × 3 methods × 3 trials).

During each trial, participants were asked to pick up a specified object and hold it statically in the air for 1 second to show the grasp was firm. We defined success strictly as procuring the object *in the first attempt.* If the chopsticks moved the object without procuring it, or the object immediately slipped away from chopsticks after being picked up, the trial failed. But the subjects were allowed to re-try the task until procuring the object or 20 seconds elapsed. We chose 20 seconds as the maximum trial length because we wanted to (1) give subjects an opportunity to learn from interacting with the object, and (2) to control the total trial time so the subject would be less likely to feel tired or frustrated.

Each subject worked with a randomized order of objects. For each object, the subject used all three methods. They completed all methods for one object before moving on to another. The subject might have gained more experience dealing with the object by the final method. We therefore randomized the order of methods for each object per person to ensure that each method's data is not skewed. Upon finishing all tasks for one object, the subject rated the difficulty and comfort of each method on a 5-point Likert scale.

Upon completing all trials, participants responded to an open-ended post-task questionnaire. Samples of all questionnaires (pre-task, during-task, post-task) are available in (*Questionnaire* n.d.).

3.1.3 *Data Acquisition*

We recorded all trials using two RGB cameras and collected written questionnaires from subjects. We tracked and recorded MoChop movements during both MoChop's and

TeleChop's trials. We recorded joint commands and robot states during teleoperation trials.

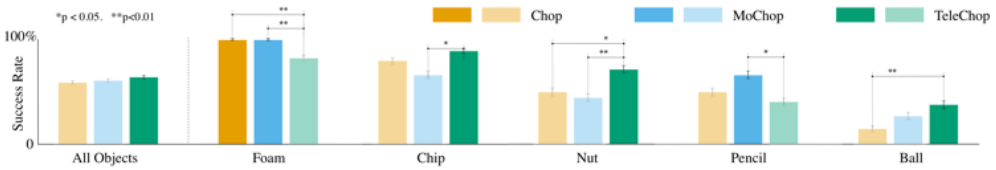


Figure 3.2: Success rate for each method. Error bars indicate 95% confidence interval. No significant variance was observed across the three methods for all objects ($p = 0.44$). However, for each individual object, we observed significant variance across methods ($p < 0.05$ in ANOVA). The most successful method for each object is highlighted with high saturation color.

3.2 ANALYSIS

We evaluated how participants' performance varied by 1) object, 2) control interface and 3) chopstick expertise. We studied how participants adjusted manipulation strategies for different objects, methods and after failures. We also compared the subjective ratings with the objective success rate for each method and object.

3.2.1 *What factors affected the success of grasping?*

SUBJECTIVE RATING OF OBJECT DIFFICULTY MATCHED OBJECTIVE SUCCESS RATE In Fig. 3.3, the variation in participants' success rates shows that the selected set of tasks presents a full spectrum variance of difficulty (ANOVA F=38.49, $p < 0.001$), ranging from very easy (foam) to very difficult (ball). The ranking derived from subjective ratings of difficulty roughly correlates with the corresponding ranking of performance, with the only exception being the order between the nut and the pencil. Paired-T tests on subjective ratings across objects were all significant ($p < 0.0001$). Paired-T test on success rates across objects were all significant ($p < 0.01$) except between nut and pencil ($p = 0.65$).

DIFFERENT GRASPING METHODS AFFECTED THE SUCCESS RATE FOR EACH OBJECT Fig. 3.2 depicts the success rate per method and how performance varied per method for each object. Overall performance was similar (F=0.8256, p=0.44) between the grasping methods, although TeleChop performed slightly better.

However, for each individual object, we observed statistically significant differences across methods ($F > 3$, $p < 0.04$). Chop and MoChop were significantly better ($p < 0.005$) at picking up foam, while TeleChop was significantly better at picking up a nut ($p < 0.05$). Foam was the simplest task and participants had 100% success rate when directly using Chop and MoChop. They were probably familiar with the environment,

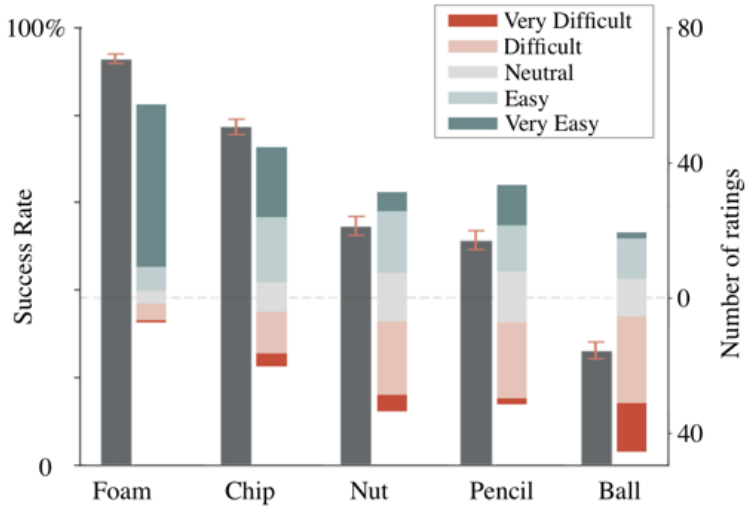


Figure 3.3: Success rate (gray bar) and subjective rating (multicolor bars) of difficulty for each object. From left to right, objects were easy to difficult.

and more specifically, the interaction between chopsticks and objects; unfamiliarity with teleoperation might explain the drop in success rate for this object. A nut, on the other hand, is a flat and slippery item that required a firm grasp after being picked up. Failures occurred mostly because the nut slipped from chopsticks after being lifted up, possibly because users were tired of supplying a concentrated force on their fingers. Teleoperation could alleviate this problem by offering a firm grasp without participants supplying force. More analysis on TeleChop's force output is in Sec. 3.2.4.

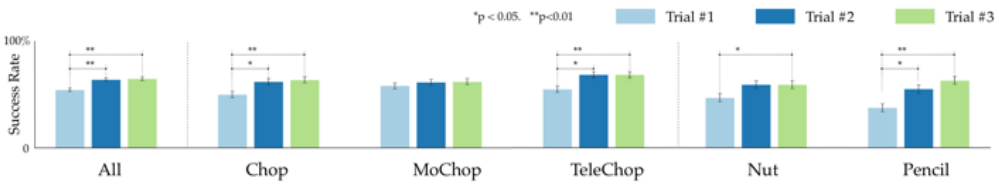


Figure 3.4: Success rate over trials

PARTICIPANTS IMPROVED PERFORMANCE WITHIN 3 TRIALS How quickly did subjects learn to adapt to the use of chopsticks for these trials? We looked at the change in success rate over the three trials as a proxy for estimating the effect of learning on participants' performance. For each task (5 in total) and each method (3 in total), each subject attempted 3 trials. Fig. 3.4 depicts the success rate per trial number. As the trial count increased, performance increased, suggesting the existence of a learning effect (F=5.47, p< 0.001). The variance in performance across trials was significant for Chop and TeleChop methods ($F = 3.79$, $p = 0.027$, and $F = 3.6$, $p = 0.032$ respectively), suggesting that subjects had an easier time adjusting to these two methods through trials. Additionally, subjects significantly increased performance over trials when picking up a nut and pencil. For the nut, they might have realized the need to

supply a firm grasp after failed trials. For the pencil, subjects might have mastered a more effective grasping point that was closer to CoM after experiencing failures. Further qualitative examination of how participants adjust their manipulation strategies is in Sec. 3.2.2.

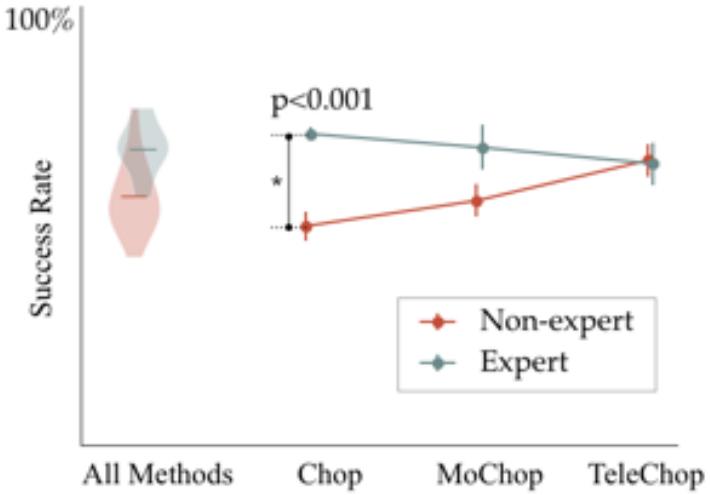


Figure 3.5: Success rate and chopstick expertise.

CHOP EXPERTISE DOESN'T IMPLY TELECHOP EXPERTISE We identified expert subjects by choosing individuals that achieved higher than median success rates when using Chop to pick up items ($\mathcal{SR}_{\text{expert}} > 60\%$). Based on this standard, our sample consisted of 11 expert and 14 non-expert subjects. Fig. 3.5 depicts the performance of expert and non-expert chopstick subjects. Both cohorts had statistically significantly different success rates when using *Chop* ($p < 0.001$) but not when using MoChop or TeleChop ($p = 0.06, 0.89$ respectively). Non-experts had a statistically significant improvement using TeleChop relative to using Chop ($p < 0.01$). We suspect that non-expert participants may have had less stable and precise control of chopsticks, which would be critical in directly picking up a nut. However, teleoperation removed this requirement and therefore enabled non-experts to perform better. This suggests that teleoperation can be a desirable interface for future data collection, especially for collecting data on certain *challenging tasks for humans*, such as prolonged grasping, as researchers can improve both hardware and controllers to alleviate the burden of controls from users.

GIVEN CHANCES TO RE-TRY AFTER A FAILED GRASP, PARTICIPANT SUCCESS RATES CAN INCREASED TO ABOVE 75% EVEN FOR THE MOST DIFFICULT TASK Many robotics grasping tasks evaluate success based on one-shot grasping, i.e., whether the robot successfully grasps the object in one try. However, humans learn from and adapt to failures (Bhattacharjee et al., 2019). In our experiment, we let subjects re-try a task after a failed trial, even though the first failed attempt might

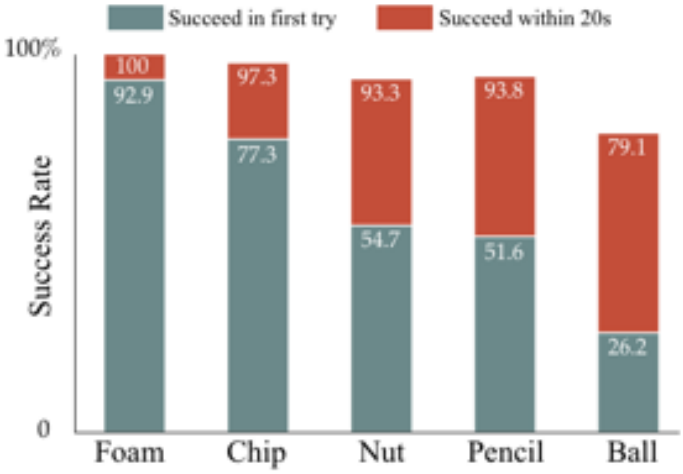
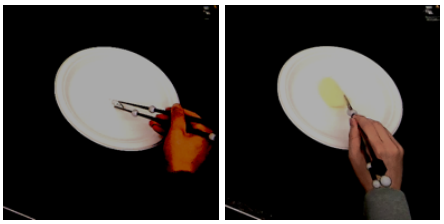
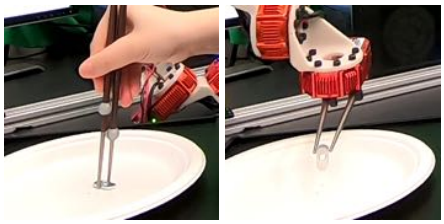


Figure 3.6: Given chances to re-try, subjects achieved greater success within 20 seconds.



(a) Nut versus Chip



(b) MoChop versus TeleChop

(c) Adapting to failures

Figure 3.7: Subjects adjusted their manipulation strategies conditioned on (a) objects, (b) methods, and (c) adapting to failures. Each set of pictures shows the same user.

have changed the object's configuration. We evaluated re-try success rates based on whether subjects could pick up the same object within 20 seconds (see Fig. 3.6.) We see a significant boost in the success rate where, within 20 seconds, subjects managed to pick up the most difficult item (glass ball) with a 79.1% success rate even though their initial success rate at first try was only 26.2%. Humans demonstrated an impressive robustness, which suggest that alternative metrics for evaluating manipulation performance might consider allowing successive tries.

3.2.2 *How did participants adjust their manipulation strategies?*

SUBJECTS APPLIED DIFFERENT STRATEGIES FOR DIFFERENT SHAPES OF OBJECTS For a flat object like nut, almost all subjects held the chopsticks parallel to the nut from the top view, as shown in Fig. 3.7a. To adapt to the curved surface of chips, 21 of 25 subjects rotated the chopsticks to procure the object. Some subjects changed the rotation of the chopsticks several times *without touching the object*, as if trying to find the optimal approach angle through visualizing the alignment. Visual information could play an important role in this adaptation.

FOR DIFFERENT METHODS, PARTICIPANTS USED DIFFERENT STRATEGIES FOR THE SAME OBJECT Fig. 3.7b shows a subject using MoChop to pick up the nut from a different angle than was used via TeleChop. Intuitively, holding chopsticks from the vertical angle could allow subjects to supply the strong force needed to pick up the Nut, whereas TeleChop could output the required force from any angle. We observed similar phenomena for chips: many participants used TeleChop to grip the chip *on one pressure point* and used MoChop and Chop to *lock* the chips between chopsticks. And for the glass ball: subjects needed to first grip the ball and then lift it up while retaining grasp strength. However, the glass ball frequently slipped during lifting. When using TeleChop, subjects seemed to pay more attention to gripping the ball and less to keeping the grip firm. These different strategies might also contribute to the performance difference.

PARTICIPANTS CHANGED STRATEGIES AFTER FAILURE As shown in Fig. 3.7c, the long pencil shape made it tricky to pick up. Most subjects tried to grip its center. Because some did not estimate the CoM accurately, the pencil rotated and dropped. After failure, many subjects adjusted the gripping point on the pencil and the contact point on the chopsticks. In this process, subjects also slowed down the manipulation motion, perhaps having realized that a rushed pick-up attempt made the pencil prone to rotate and drop. Interestingly, some had a different adaptation strategy when using TeleChop: 23 out of 26 subjects gripped the pencil at its geometry center, adjusted how much to close the leader chopsticks (and how much force TeleChop to output) until the robot could grasp the pencil firmly.

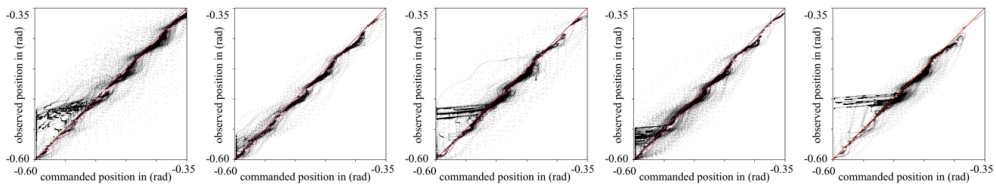
The round slippery glass ball is among the most challenging object for chopsticks to pick up. Conceptually, picking up the ball is straight-forward: one need to close the chopsticks around the diameter of the ball. But placing the chopsticks exactly across the diameter of the ball was challenging. Many subjects went through a trial-and-error process, slightly lifting up the chopsticks and closing them around the ball, observing whether the ball was being picked up and adjusting the next grip accordingly. It allowed subjects to succeed at picking up the ball eventually, achieving an astonishing success rate of 79% when allowed re-try.

3.2.3 *Subjective Ratings*

Participants rated teleoperation as the least comfortable and the most difficult to use ($p < 0.0001$), except when to pick up the ball. Participants explained that the negative ratings came from the sense of indirection added by the teleoperation interface, lack of haptic feedback and the misalignment between the robot arm and human arm. We also observed that subjects took significantly longer time on TeleChop's trial than Chop and MoChop.

However, participants' performance using teleoperation was empirically better than using other methods to pick up 3 out of the 5 objects chosen (chips, nut and ball shown in Fig. 3.2). Possible reasons of such a contradiction include (1) teleoperation alleviated the burden of firm and stable control from the subject for certain tasks, (2) subjects found a strategy for teleoperation method that could achieve higher success rate but required more effort, and (3) subjects, *feeling* that the teleoperation interface was more difficult and less comfortable, *spend more effort and concentration* using this method. We found qualitative evidence supporting these hypotheses: (1) 15 of 25 subjects in the post-task questionnaire reported that TeleChop simplified grasping because it "(was) easier to maintain a constant grip," "(required) less effort on my hand to grip the object between the chopsticks," and "(users) only need to care about the motion of chopsticks without, considering how much force to exert" , (2) 19 of 25 subjects developed a different strategy for TeleChop compared to the other two methods for grasping the same object, and (3) some subjects commented on how *they* adapted to work with the TeleChop: "I started focusing on the robot joints and how to move my arm in a way that translated to the robot joint, the placement tasks became easier." and "It takes a little time to learn and be familiar with the movement (of TeleChop)". Therefore, the subjects adaptation to the robot and TeleChop's stable force output enable the teleoperation to achieve higher success rate on certain tasks but made it uncomfortable regardless.

3.2.4 *Position as a proxy for force*



(a) Foam (b) Chips (c) Nut (d) Pencil (e) Ball

Figure 3.8: Displacement between commanded position and observed position. The X-axis shows commanded position. The Y-axis shows observed position. (unit rads). A smaller number means closing chopsticks. The displacement is proportional to the torque output between chospticks.

One main benefit of our teleoperation interface is its ability to collect force commands without needing specialized force measurement devices. We achieved this by using position as a proxy for force. Subjects could close the leader chopsticks to direct TeleChop to close. However, TeleChop could hold a ball between its tips rather than closing the chopsticks as instructed. The error between the commanded and the actual position incurred a proportional torque output from our PID controller. To pick up any object that requires gripping, this mechanism is necessary. Fig. 3.8 shows the recordings of displacements, which indicate the generation of forces. Note that the clear displacement gaps for the Nut and Ball correspond to subjects gradually closing leader chopsticks to add force to firmly grip those objects. On post-task questionnaires, subjects commented on how firm the TeleChop grasp and that "(they) didn't have to squeeze that hard to increase the tension." This might explain why TeleChop achieved highest success rate on the Nut and Ball. It also suggested that teleopeartion could be more helpful in picking up hard, small, and heavy pieces.

3.3 DISCUSSION

3.3.1 *What interface is the best for collecting demonstrations?*

MoChop and Chop were better at picking up the foam and the pencil whereas TeleChop achieved the highest success rate picking up the chip, nut, and ball. Although TeleChop might be the most unintuitive and uncomfortable to use, subjects in our study demonstrated impressive flexibility and used teleoperation to achieve, overall, a comparable success rate to the other methods.

TeleChop has proved to be more helpful in picking up small, hard and slippery items that demanded precision and stability in control than soft, compliant, lightweight objects. This might be related to motor noise in human movement control. (Jones, Hamilton, and Wolpert, 2002) proposed that higher muscle force generally comes with higher force variability. For tasks requiring strong and stable force output, e.g., surgical operation, teleoperation might allow for lower muscle forces, avoid muscle tremors, and could be the preferred method.

The subjective feedback suggested that the teleoperation interface tire out users at a faster rate than other methods, raising the question of whether the teleoperation interface can collect large amounts of demonstrations. Collecting data via motion-tracking could be less tiring and can perhaps capture more a realistic reflection of how humans use chopsticks. One possible remedy for this is to collect data in short-burst batches.

However, an advantage of using teleoperation is that recorded trajectories can be replayed on the robot to accomplish the task. Transforming and replaying motion-captured data is similar to open-loop control, whereas replaying teleoperation data is closer to closed-loop control as the human demonstrator serves as the closed-loop controller for the robot during recording. This feature can be critical for approaches like learning from demonstrations.

All things considered, we would recommend using a teleoperation interface to collect demonstrations for robotics manipulation with chopsticks.Teleoperation superiority on certain tasks—even though it takes away haptic feedback and introduces additional delays—implies that teleoperation could boost human performance, e.g., offering tremor canceling in robotic surgery, supplying stronger force output in exoskeletons, or aiding people with disabilities.

3.3.2 *Can we learn from human adaptation to a robotic arm?*

Human subjects have demonstrated an impressive capability to serve as the controller of imperfect hardware that is not precise to control, counters their intuition during usage, and lacks the haptic feedback that they are familiar with in challenging manipulation tasks.

During human study, we observed multiple learning effects happening. The subject learned about different chopsticks methods, the robot interface, the object dynamics, and the manipulation strategies they used. We highlight how non-expert subjects using TeleChop could achieve comparable success rates to expert subjects using Chop. We quantify the subjects' improvement by studying the change in success rate over three trials, noting a marked increase in success. More importantly, humans recognize their mistakes immediately and alter their manipulation strategies during the course of a single task, achieving a median success rate of over 93% but not on the first try. Therefore, recognizing the short-term learning factors in humans and formalizing them in algorithms may boost robotics manipulator's robustness.

3.3.3 *Limitations*

Our teleoperation interface allows successful demonstrations of chopsticks manipulation, but it can be counter-intuitive and uncomfortable to use. It would be more beneficial to design the teleoperation robot to use a parallel configuration to human arms, have more stable movement, and enable more advanced tremor canceling. We used a simple controller and haven't tuned our system to completely eliminate the control noise. A smoother and more precise teleoperation system could improve the human experience and reduce the cognitive load.

Nevertheless, our work brings out the potential of involving chopsticks in robotic manipulation and demonstrates how a teleoperated robot with chopsticks can pick-up challenging items without relying on a complicated end-effector. We intend to extend this work by learning from the demonstrations we collected and building a skillful robot at using chopsticks autonomously.

Part II

LEARNING FROM EXPERT DATA

4

IMITATION LEARNING FOR FINE MANIPULATION

We hope to derive an autonomous policy from expert demonstrations and study imitation learning (Dyrstad et al., 2018; Ho, Gupta, and Ermon, 2016; Pastor et al., 2009; Zhang et al., 2018).

We have access to demonstration data but not to the expert's policy function or the environment's transition model. Under these conditions, supervised learning methods like *behavior cloning* (Pomerleau, 1989) learn a policy function by matching the expert's action distribution. Minimizing action distribution divergence, however, does not necessarily guarantee the recovery of parsimonious states that lead to task success (Osa et al., 2018). A learned agent can suffer from *covariate shift* (Quionero-Candela et al., 2009), i.e., compounding errors in the action space that lead the agent to unseen states during test time. This problem can be especially detrimental for fine manipulation, the success of which critically depends on a few steps that usually occur near the end of a trajectory.

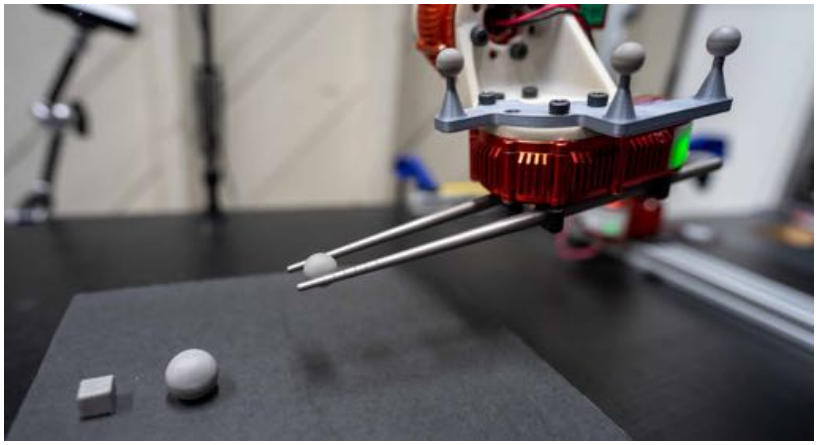


Figure 4.1: Fine manipulation using chopsticks.

To remedy covariate shift, researchers have proposed *interactive* imitation learning methods, such as DAgger (Ross, Gordon, and Bagnell, 2011) and DART (Laskey et al., 2017), to query an expert *online* for corrective labels. DAgger rolls out a learned agent and asks the expert for labels on learner visited states, which can be computationally expensive and unnatural on a teleoperation interface (). DART injects noise during data collection, disturbs expert teleoperation, and forces the expert to provide corrective labels. However, injecting noise during data collection can burden the expert: adding

a small amount of random noise for our fine manipulation task, as DART suggests, would require the expert to spend 43% more time on collecting data.[1]

These challenges prompt us to address covariate-shift in model-free imitation learning in a *non-interactive* setting, where we have access to demonstration data but not to an interactive expert. Since covariate shift results from the interplay of single-step errors and their accumulation over time, our key ideas are to (1) increase data support to address single-step errors, and (2) provide corrective labels to address the accumulation of errors. Specifically, we provide:

- *Enhanced data support* by transforming the data to an object-centric frame that preserves the relative transformation between the end effector and object, while making training data denser around the *critical* region for grasp success.
- *Corrective labels by injecting noise* into the collected state, assuming the same action may serve as the *corrective label* for the deviated state. Thus, we implicitly enforce smoothness to the learned policy and tell the agent how to recover from deviated states.
- *Corrective labels by choosing a combination of parametric and non-parametric methods* that improve matching of the action distribution at unseen states. Because of our problem structure, a better match in action distribution leads to a higher likelihood of matching the state distribution, preventing error accumulation.

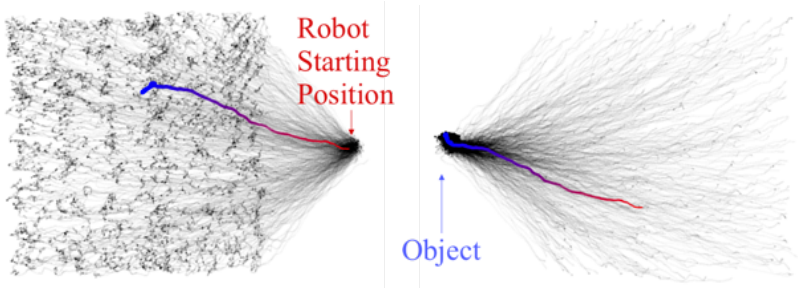
4.1 METHODS

4.1.1 *Transform: Increasing Data Support*

Our goal is to develop an agent that can generalize from demonstration data to predict an action for any query state. However, we lack data support for some states (e.g., the "unseen state" during rollout). We propose to apply an invariant operator to transform the data, making it denser around the region of interest and thus increasing the data support.

In manipulation, changing the frame of reference can significantly change the distribution of trajectories (Fig. 4.2). We could choose a *robot-centric* frame, where the robot base is the origin, or an *object-centric* frame (Mason, Srinivasa, and Vazquez, 2011), where the object location is the origin. The change of frame preserves the relative transformation between the end-effector and the object and is therefore an invariant operator. We propose that using an object-centric frame can reduce the covariate shift and improve the policy generalization, especially for fine manipulation. The transformation to an object-centric frame would result in a denser distribution of trajectories near the origin where the object is located, increasing data support for this critical region that determines grasping success. Using an object-centric frame also allows

1 We injected an independent Gaussian noise to each joint. Though 95% of the noise resulted in at most 0.35° deviation per joint, it lowered the expert success rate by 18% and forced the expert to spend more time completing each trajectory.

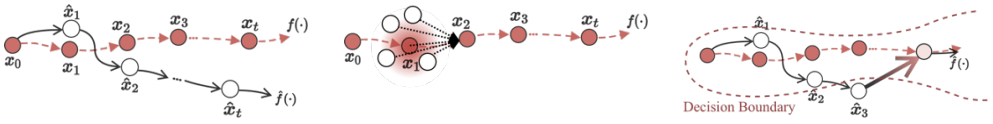


(a) Robot-centric frame. (b) Object-centric frame.

Figure 4.2: Visualizing the end-effector positions for all demonstrations under different coordinate frames. Each black dot is an xyz-position of the end effector in one step. We highlight one trajectory, which starts with red dots and ends in blue.

the policy learned to be invariant to the translation of object location. This makes the learned policy more sample efficient when generalizing to novel object locations.

4.1.2 *Noise: Generating Synthetic Labels*



(a) Covariate shift: A learner roll out (black) deviates from the demonstration (red) and error accumulates.

(b) Inject noise into the collected states and reuse the collected action as synthetic corrective labels.

(c) Use non-parametric methods (like a k-NN) to return the agent to proper region when deviations occur.

Figure 4.3: Prevent error escalation in imitation learning.

Although the transformation technique we use improves the agent's success rate, we still observe significant deviations during test time that result in task failure (Fig. 4.3a). This is understandable because machine learning algorithms generally need exponentially more data for progressive improvement (Laskey et al., 2017). Instead of naively collecting more data, we introduce corrective action labels that can help the agent recover from deviations. For example, Venkatraman *et al.* (Venkatraman, Hebert, and Bagnell, 2015) rolled out trained agents, collected their deviation states and used model-predictive control to generate corrective labels to go back to the demonstrated trajectory. Unfortunately, models sufficiently accurate for fine manipulation can be challenging to build.

We propose to generate *synthetic* corrective labels by injecting noise into the collected demonstration *states* ("deviated state") and reusing the collected action ("corrective labels"), thus not requiring access to an expert or a model. Unlike DART and DAgger, which emphasize collecting corrective labels for the states that the agent will visit during rollout (test state distribution), we hypothesize that we do not need to match the deviated states' distribution accurately. Instead, we need to collect enough corrective labels to

cover the deviated states' distribution. Since we can generate labels for free without burdening an expert, we choose to generate labels for randomly sampled deviated states, thus simplifying the selection of states for which to generate synthetic corrective labels. Fig. 4.3b shows an example where we sample states around a demonstrated state and reuse the demonstrated action as synthetic corrective labels.

Researchers have injected noise (Bishop, 1995) into problems that reduce a high-dimensional input to a low-dimensional output, e.g., for classification (Sietsma and Dow, 1991) and object recognition in visual and language domains (Shorten and Khoshgoftaar, 2019). In these works, such tasks are invariant under a wide variety of transformations (Goodfellow et al., 2016). However, our robotic manipulation task has *low-dimensional* states and actions, where the mapping learned may not be invariant to the noise. We provide two insights to justify why injecting noise can still be desirable.

First, we apply a *small* amount of additive Gaussian noise to the demonstration state instead of a large amount that could pollute the data by mapping a state to a detrimental action. Inspired by (Poole, Sohl-Dickstein, and Ganguli, 2014), which showed the effectiveness of noise injection for autoencoders by carefully tuning the magnitude of the noise, we generate Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma)$ to add to the collected states, where σ is the covariance of the noise. For simplicity, we correlate the noise size σ with the variance of the data.

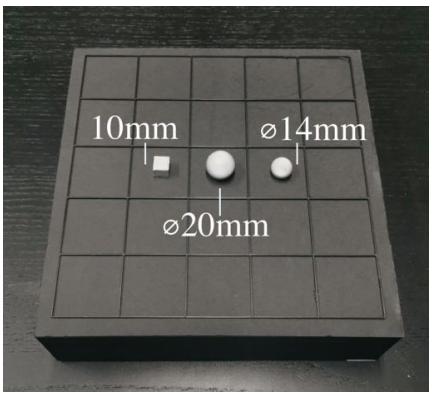
Second, because of the structure of our problem, the collected action can serve as the corrective label for the noise-injected deviated state. Our state and action representations both include the end-effector pose. Therefore, when an agent starts drifting from a demonstrated trajectory and enters a deviated state, our algorithm can teach it to return to the original trajectory by reusing the same action label. Injecting noise can also ensure the learned policy is smooth, which is desired since we assume the actions are Lipschitz continuous w.r.t the states.

4.1.3 *Ensemble: Following the Expert Advice*

We can reduce error accumulation at unseen states by choosing methods that more effectively recover the action distribution independent of the states. A neural network's optimization objective is limited to its training data and will not necessarily generalize well to unseen inputs (Russell and Norvig, 2009). In contrast, non-parametric methods generate test outputs by combining the training data, their predictions must come from the training data and are therefore constrained (Altman, 1992). e.g., a k-nearest neighbor (`k-NN`) agent will not cause the robot to move its joint positions beyond the interpolation of its training data.

As an example, we use `k-NN` in conjunction with behavior cloning (`BC`). Specifically, our agent follows the `k-NN` predicted action if the query state deviates from the training data (Fig. 4.3c).

By using the `k-NN` method, we are *forcing a known action* to a new unseen state during test time to ensure the action distribution during training and testing will match. For our manipulation task, the state and action both include the robot's end-effector



(a) Training Objects and Evaluation Coverage.

Figure 4.4: Experiment setup.

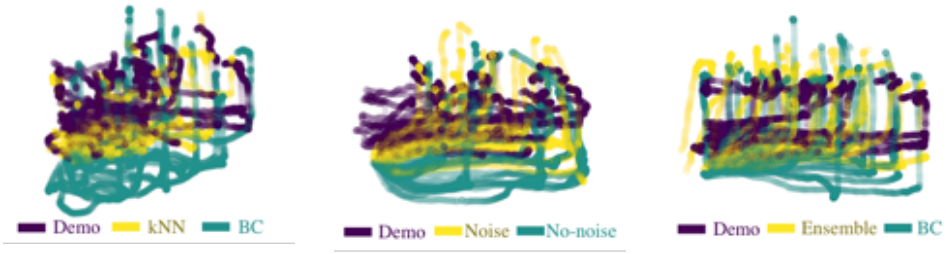
pose. Sending a *known action* is equivalent to sending the agent to *a known state*, implicitly reducing the agent's deviation from training data, thus reducing covariate shift. However, nonparametric method's performance is subject to its distance function, which can be difficult to design for high-dimensional data.

The distance function for non-parametric methods serves two purposes: (1) to evaluate the proximity of a query to the stored data points; and (2) to weight and combine the expert labels. Our key observation is that (1) requires only a rough estimate of the distance to decide whether a query state is far from the training data, and (2) needs a carefully tuned distance function to assign weights to expert labels. Therefore, we propose to use a simple decision tree to invoke a `k-NN` agent *only* when the distance of the query state is far from its nearest neighbors and invoke a behavior cloning neural network agent otherwise. By invoking `k-NN` only when the agent is far away, we bypass the need to carefully design a distance function for it, favor `BC`'s scalability with data when we are inside the training data distribution, and rely on `k-NN` to correct the agent's deviation when we are outside the training data distribution. We only explore `k-NN` to serve as an example and believe that other non-parametric methods that select actions from data-supported states could work similarly well.

4.2 EXPERIMENTS

4.2.1 *Experimental Setup*

Our agent assumes access to the tracked location of the objects and the robot's end-effector pose. We defined success as *grasping* the objects using chopsticks, *lifting* them above the workstation, and *holding* them in the air for 1 s. We evaluated the performance of each method on each object by computing the success rate over 25 trials. During evaluation, we divided the square workstation plate into a 5×5 grid (Fig. 4.4a) and placed the object in the center of each grid cell to ensure effective coverage over the entire workspace.



(a) `k-NN+ObjC` versus `BC+ObjC`.

(b) `BC+ObjC` versus `BC+ObjC+Noise`.

(c) `Ensemble` versus `BC+ObjC+Noise`.

Figure 4.5: Comparing the state distributions as a proxy of covariate shift for trained agents. Each dot is a state in the agent's roll out. Green states are farther away from the demonstrations (purple), indicating that their corresponding agent suffers from more covariate shift than the yellow agent.

DEMONSTRATIONS We collect the demonstration at 100 Hz to match the test scenario. Each trajectory contains an average of 600 (state, action) pairs. The state is a 11-D vector containing the robot's state and the object's tracked x-y-z position. The action is the target end-effector pose. During each trajectory, we initiate the robot around a fixed home configuration and place the object at a random location across the workstation. One expert user collect all trajectories to reduce multi-modal behavior that might interfere with learning (e.g., picking up object using different strategies). We remove failed trajectories and keep only the 500 successful ones.

4.2.2 *Experimental Procedure*

We compared our methods in Section 5.3 with human demonstrations during teleoperation (`Expert`) and a replay of the successful demonstrations (`Replay`). `Replay` tests the *repeatability* of our hardware. We chose successful demonstrations, placed objects at *exactly the same locations* used during data collection, and replayed the demonstrations to see if the robot could pick up the objects.

We used two baselines. The first is a parametric method, `BC+RobotC`, a neural-network based behavior cloning agent that uses the default robot-centric frame. The second is a non-parametric method, `k-NN+RobotC`, which is a k-nearest-neighbor agent that also uses the robot-centric frame.

We evaluated three methods as described in Section 5.3: (1) using the object-centric frame to train behavior cloning and the k-nearest neighbors agents, `BC+ObjC` and `k-NN+ObjC`, respectively, (2) injecting a small amount of Gaussian noise into the behavior cloning agent, `BC+ObjC+Noise`, and (3) combining the parametric method `BC+ObjC+Noise` and non-parametric method `k-NN+ObjC` via a decision tree model, denoted as `Ensemble`.

Method	Cube	Ball⌀20 mm	Ball⌀14 mm	All
`Expert`	100	80	68	82.7
`Replay`	100	80	80	86.7
`BC +RobotC`	84	16	12	37.3
`BC +ObjC`	92	16	24	44.0
`BC +ObjC+Noise`	92	76	48	72.0
`k-NN +RobotC`	64	28	8	33.3
`k-NN +ObjC`	84	64	12	53.3
`Ensemble`	**96**	**84**	**60**	**80.0**

Table 4.1: Percentage success rates evaluated over 25 trials.

4.3 RESULTS

4.3.1 *Success Rates for Fine Manipulation*

The experimental results are shown in Table. 4.1, and the best performers in each column are highlighted. Our parametric method baseline, `BC+RobotC`, and nonparametric method baseline, `k-NN+RobotC`, had relatively low success rates. However, the causes of their failures differ. `BC+RobotC` has difficulty picking up objects that are placed farther away from the robot. The agent tends to reach towards the wrong location after moving over a long distance to approach the object, highlighting the covariate shift's impact. In contrast, the `k-NN+RobotC` agent's poses look more similar to expert demonstrations. However, its trajectories are not smooth and sometimes end abruptly on top of the object without picking it up. This occurs because `k-NN` does not guarantee a smooth policy function; even after careful tuning of the distance function, it was challenging to eliminate the jerky motions. `k-NN`'s sudden stops are due to direct imitation of the training data. During demonstration, the human expert often slowed or even paused their movements around the object, adjusting the approaching pose before closing the chopsticks and lifting the object. The distance function we chose fails to select and mix the more relevant action labels. This confirms the sensitivity of `k-NN` to its distance function.

Transforming to the `ObjC` frame improved the success rates for `k-NN` and `BC` by 20% and 6.7%, respectively. `k-NN` becomes less likely to generate jerky motions or stop since it benefits from the increased data support. `BC` still suffers from covariate shift, but the agent has a higher likelihood of reaching towards the object due to denser data distribution near it.

Injecting noise to `BC+ObjC` during training increases its success rate by 28%. When the items are *close* to the robot, the agent has an almost 100% success rate picking up even the most challenging item. For objects that are far away, the robot sometimes

picks up the object by successfully reaching the location; at other time, it ends up merely rotating the chopsticks.

Using a decision tree to combine our best parametric method (`BC+ObjC+Noise`) and non-parametric method (`k-NN+ObjC`) yields the highest performing agent that achieves near-expert performance. During test time, if a state's distance to its nearest neighbors exceeds a threshold, the agent triggers the non-parametric method to bring the state back. We observe that almost all rollouts trigger the non-parametric method at least once. We observed that the `Ensemble` agent can reach an object in a way similar to poses demonstrated by the expert, no matter how far it is placed. Failures occasionally occur as the agent misses the grasping point by some sub-mm error.

4.3.2 *Covariate Shift Across Methods*

To gauge the covariate shift for different agents, we visualize the distributions of their test states. We collect 25 rollouts from each agent, record the robot-visited states, and plot the state distribution after dimensionality reduction using Principal Component Analysis (PCA), as shown in Fig. 4.5. First, we observe that `BC` encounters more covariate shift than `k-NN`, i.e., that states visited by `k-NN` are closer to the demonstrated states, confirming that a better matching of action distribution will lead to a better match of state distribution. Second, injecting noise into `BC` results in less covariate shift than no noise, verifying that noise injection can provide effective correcting labels. Third, the `Ensemble` model that combines `BC` with `k-NN` has less covariate shift than using `BC` alone.

4.3.3 *Noise Injection: Validation through MuJoCo Environments*

We apply the noise injection method to MuJoCo simulated environments (Todorov, Erez, and Tassa, 2012b) to test the method's generality. We use demonstration data from (*University of California Berkeley CS 285: Deep Reinforcement Learning* n.d.), train 5 behavior cloning agents under consecutive random seeds as baselines, and train another 5 agents with noise injection for comparison. Figure 4.6 compares performances before and after noise injection. A paired T-test shows that $p < 0.05$ for all environments. There is strong evidence that, on average, noise injection improves the imitation learner.

Though the performance gains for some simulated environments are not as significant as those we see for our real robot, we think the difference may be due to the demonstration source. We use "real" human data for our robot experiment versus the "synthetic expert demonstration" generated by a reinforcement learning (RL) agent for the MuJoCo tasks. Human experts are known to exhibit multi-modal behaviors during demonstrations, whereas trained RL agents tend to have single modes in their reaction (Ke et al., 2020a). Given that noise injection improves the success rate for our physical robot task by a considerable 28%, further inquiry is needed to determine if noise injection is better at enhancing learning from multi-modal demonstration data.

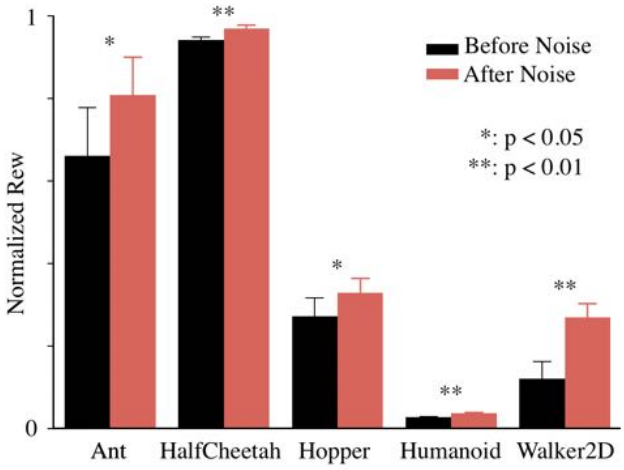


Figure 4.6: Performance comparison before and after noise injection. For each MuJoCo environment and each condition, (before or after noise), we trained 5 agents using consecutive random seeds. The performance difference is statistically significant under paired T-tests.

4.4 DISCUSSION

We leave some topics for future work. During noise injection, for simplicity, we experiment only with independent multivariate Gaussian noise with a fixed size of covariance. It is worth exploring how to formalize the bounded noise and analyzing how different task domains may benefit from different noise shapes. For the ensemble model, future work could explore an alternative way to switch between `k-NN` and `BC` agents in the `Ensemble` model, perhaps by learning a threshold condition from the data.

Our work critically depend on two key assumptions. First, to increase data support by applying an invariant operator, we assume the existence of a *critical* region that demands more data support. Second, to reuse collected action labels and leverage a nonparametric method to generate corrective labels, a more accurate match of action distribution should lead to a more accurate match of the states. The assumption holds if a part of the state and action representation is directly connected, e.g., the robot state contains its joint position, and the robot command accepts the target joint position. The assumption does not hold, for example, if the robot is torque-controlled; in these cases, further exploration on how a learner can generate synthetic corrective labels is needed.

Nevertheless, our proposals do not assume access to a model or an interactive expert and are therefore more easily applicable to fine manipulation tasks. Compared to DAgger and DART, which collect corrective labels from experts, we can generate synthetic corrective labels for free. Because of the relatively lower cost of doing so, we generate labels for randomly sampled state distributions that *cover* the deviated state distribution without accurately *matching* it. Though our proposals focus on a non-interactive setting, they can directly transfer to an interactive one.

We choose model-free imitation learning because an *accurate* model for fine manipulation is rare. However, it remains to be seen how to leverage an *inaccurate* model in imitation learning. This is but a first step towards exploring general-purpose autonomous fine manipulation using simple tools.

5

MODEL-BASED CORRECTIVE IMITATION LEARNING

Previously we motivate the choice of *model-free* imitation learning due to the difficulty of building precise models. It can take tremendous efforts to create simulator models to achieve the level of precision required in fine manipulation tasks (Billard and Kragic, 2019; Cutkosky, 2012) because of repeatability, actuator backlash and hard-to-model and hard-to-control dynamics. However, here we explore how to construct and leverage a dynamics model that is not necessarily precise across the global state and action space.

For general applicability, we still focus on imitation learning (IL) that rely *solely on expert demonstrations*. We propose a method to enhance robustness of IL by generating corrective labels for data augmentation. We recognize an under-exploited feature of dynamic systems: local continuity. Despite the complex transitions and representations in system dynamics, they need to adhere to the laws of physics and exhibit some level of local continuity, where small changes to actions or states result in small changes in transitions. While realistic systems may contain discontinuities in certain state space portions, the subset exhibiting local continuity proves to be a valuable asset.

Armed with this insight, our goal is to synthesize *corrective* labels that guide an agent encountering unfamiliar states back to the distribution of expert states. Leveraging the presence of local continuity makes the synthesis of these corrective labels more tractable. A learned dynamics model with local Lipschitz continuity can navigate an agent from unfamiliar out-of-distribution states to in-distribution expert trajectories, even extending beyond the expert data support. The local Lipschitz continuity allows the model to have bounded error in regions *outside* the expert data support, providing the ability to generate appropriately corrective labels.

We propose a mechanism for learning dynamics models with local continuity from expert data and generating corrective labels. Our practical algorithm, **CCIL**, leverages local **C**ontinuity in dynamics to generate **C**orrective labels for **I**mitation **L**earning. We validate CCIL empirically on a variety of robotic domains in simulation. In summary, our contributions are:

- **Problem Formulation:** We present a formal definition of *corrective labels* to enhance robustness for imitation learning. (Sec. 5.3.1).
- **Practical Algorithm:** We introduce **CCIL**, a framework for learning dynamics functions, and leveraging local continuity to generate corrective labels. Our method is lean on assumptions, primarily relying on availability of expert demonstrations and the presence of local continuity in the dynamics.
- **Theoretical Guarantees:** We showcase how local continuity in dynamic systems would allow extending a learned model's capabilities beyond the expert data. We present practical methods to enforce desired local smoothness while fitting a dynamics function to the data and accommodating discontinuity (Sec. 5.3.2). We provide a

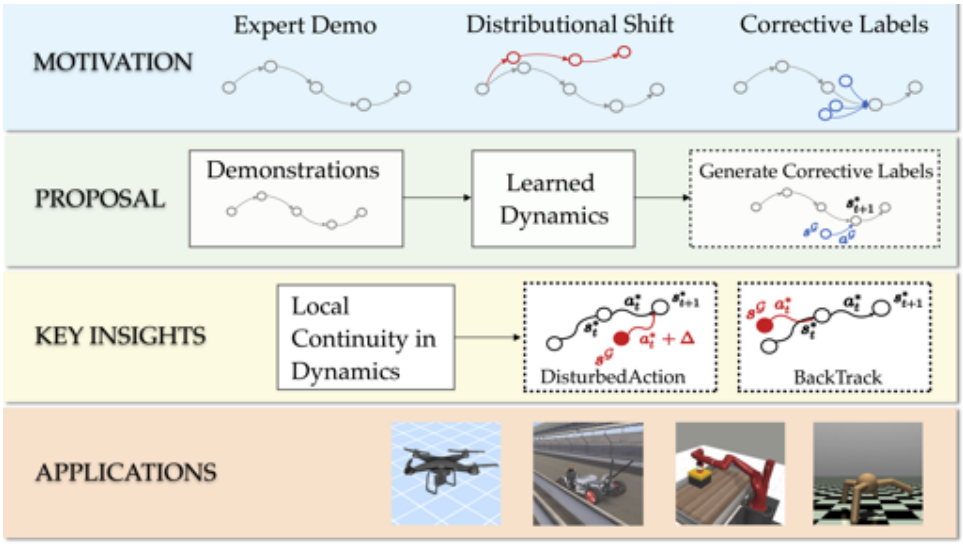


Figure 5.1: **Our proposed framework, CCIL.** To enhance the robustness of imitation learning, we propose to augment the dataset with synthetic corrective labels. We leverage the local continuity in the dynamics, learn a regularized dynamics function and generate corrective labels *near* the expert data support. We provide theoretical guarantees on the quality of the generated labels. We present empirical evaluations CCIL over 14 robotic tasks in 4 domains to showcase CCIL improving imitation learning agents' robustness to disturbances.

theoretical bound on the quality of the model in this area and the generated labels (Sec. 5.3.3).

- **Extensive Empirical Validation:** We conduct experiments over 4 distinct robotic domains across 14 tasks in simulation, ranging from classic control, drone flying, high-dimensional car navigation, legged locomotion and tabletop manipulation. We showcase our proposal's ability to enhance the performance and robustness of imitation learning agents (Sec. 5.4).
- **In-depth Study of Real-World Applications:** We perform extensive evaluation on a real robotic platform for fine manipulation, demonstrating **CCIL** 's capability to improve the performance of imitation learning with limited data availability. Through comprehensive ablations, we also analyze how design choices and hyperparameters impact **CCIL**, providing practical guidance on choosing appropriate parameters for maximum efficacy.

5.1 BACKGROUND

Imitation Learning (IL) and Data Augmentation. Given only the expert demonstrations, Behavior Cloning (BC) remains a strong empirical baseline for imitation learning (Pomerleau, 1988a). It formulates IL as a supervised learning problem and has a plethora of data augmentation methods. However, previous augmentation methods mostly leverage interactive expert (Ross, Gordon, and Bagnell, 2011) or some form of invariance in the action space (Bojarski et al., 2016; Florence, Manuelli, and Tedrake, 2019; Ke et al., 2021b; Spencer et al., 2021; Venkatraman, Hebert, and Bagnell, 2015;

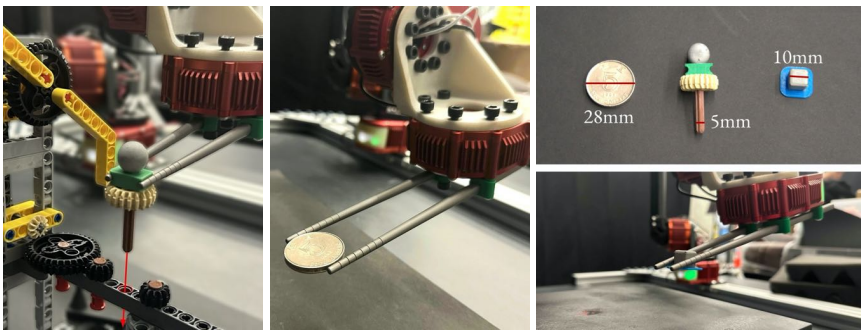


Figure 5.2: The three tasks we consider for validating CCIL in real-world fine manipulation, *GearInsertion*, *GraspCoin*, and *GraspCube*, along with our three task objects: a coin, a Lego gear, and a small cube.

Zhou et al., 2023a), which is a domain-specific property. Specifically, Ke et al. (2021) explores noise-based data augmentation for position-control, but lacks principle for choosing appropriate noise parameters. Block et al. (2024) constructs a stabilizing controller around expert demonstrations, conceptually similar to our own approach, but relies on suitably parameterizable low-level stabilizing controllers, which may be difficult to formulate. Park and Wong (2022) learns a dynamics model for data augmentation, similar to our approach, but learns an *inverse* dynamics model and lacks theoretical insights. In contrast, our proposal leverages local continuity in the dynamics, is agnostic to domain knowledge, and provides theoretical guarantees on the quality of generated data.

Mitigating Covariate Shift in Imitation Learning. Compounding errors push the agent astray from expert demonstrations. Prior works addressing the covariate shift often request additional information. Methods like DAGGER (Ross, Gordon, and Bagnell, 2011), LOLS (Chang et al., 2015), DART (Laskey et al., 2017) and AggrevateD (Sun et al., 2017) use interactive experts, while GAIL (Ho and Ermon, 2016), SQIL (Reddy, Dragan, and Levine, 2020) and AIRL (Fu, Luo, and Levine, 2018) sample more transitions in the environment to minimize the divergence of states distribution between the learner and expert (Ke et al., 2021a; Swamy et al., 2021). Offline Reinforcement Learning methods like MOREL (Kidambi et al., 2020) are optimized to mitigate covariate shift but demand access to a ground truth reward function. Reichlin et al. (2022) trains a recovery policy to move the agent back to data manifold by assuming that the dynamics is known. MILO (Chang et al., 2021) learns a dynamics function to mitigate covariate shift but requires access to abundant offline data to learn a high-fidelity dynamics function. In contrast, our proposal is designed for imitation learning without requiring additional data or feedback, complementing existing IL methods. We include MILO and MOREL as baselines in experiments.

Local Lipschitz Continuity in Dynamics. Classical control methods often assume local Lipschitz continuity in the dynamics to guarantee the existence and uniqueness of solutions to differential equations. For example, the widely adopted C^2 assumption in optimal control theory (Bonnard, Caillau, and Trélat, 2007) and the popular control framework iLQR (Li and Todorov, 2004). This assumption is particularly useful in the

context of nonlinear systems and is prevalent in modern robot applications (Kahveci, 2007; Sarangapani, 2018; Seto, Annaswamy, and Baillieul, 1994). However, most of these methods leveraging dynamics continuity are in the optimal control setting, requiring a pre-specified dynamics model and cost function. In contrast, this work focuses on robustifying imitation learning agents by learning a locally continuous dynamics model from data, without requiring a human-specified model or cost function.

Learning Dynamics using Neural Networks. Fitting a dynamics function from data is an active area of research (Hafner et al., 2020; Kaufmann et al., 2023; Wang et al., 2022). For continuous states and transitions, Asadi, Misra, and Littman (2018) proved that enforcing Lipschitz continuity in training dynamics model could lower the multi-step prediction errors. Ensuring *local* continuity in the trained dynamics, however, can be challenging. Previous works enforced Lipschitz continuity in training neural networks (Arjovsky, Chintala, and Bottou, 2017; Bartlett, Foster, and Telgarsky, 2017; Miyato et al., 2018), but did not apply it to dynamics modeling or generating corrective labels. Shi et al. (2019) learned smooth dynamics functions via enforcing *global* Lipschitz bounds and demonstrates on the problem of drone landing. Pfrommer, Halm, and Posa (2021) learned a smooth model for fictional contacts for manipulation. While this work is not focusing on learning from visual inputs, works such as Khetarpal et al. (2020); Zhang et al. (2021); Zhu et al. (2023) are actively building techniques for learning dynamics models from high-dimensional inputs that could be leveraged in conjunction with the insights from our proposal.

5.2 PRELIMINARIES

We consider a finite-horizon Markov Decision Process (MDP), $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, f, P_0\}$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, f is the ground truth dynamics function and P_0 is the initial state distribution. A *policy* maps a state to a distribution of actions $\pi \in \Pi : s \to a$. The dynamics function f can be written as a mapping from a state s_t and an action a_t at time t to the change to a new state s_{t+1}: $f(s_t, a_t) = s_{t+1} - s_t$. Following the imitation learning setting, the true dynamics function f and reward are unknown, and we only have transitions drawn from the system dynamics.

We are given a set of demonstrations $\mathcal{D}^*$ as a collection of transition triplets: $\mathcal{D}^* = \{(s_j^*, a_j^*, s_{j+1}^*)\}_j$, where $s_j^*, s_{j+1}^* \in \mathcal{S}, a_j^* \in \mathcal{A}$. We can learn a policy from these traces via supervised learning (behavior cloning) by maximizing the likelihood of the expert actions being produced at the training states: $\arg\max_{\hat{\pi}} \mathbb{E}_{s_j^*, a_j^*, s_{j+1}^* \sim \mathcal{D}^*} \log(\hat{\pi}(a_j^* \mid s_j^*))$. In practice, we can optimize this objective by minimizing the mean-squared error regression using standard stochastic optimization.

In this paper, we consider the local continuity of the dynamics function of the environment, which we formally define as the local Lipschitz continuity.

Definition 2.1 (Local Lipschitz Continuity)**.** A function f is *locally Lipschitz continuous* around (s, a) with coefficient K if there exists a neighborhood of (s, a) of size δ such that for every $(\tilde{s}, \tilde{a})$ where $\|(s, a) - (\tilde{s}, \tilde{a})\| \leq \delta$:

$$\|f(s, a) - f(\tilde{s}, \tilde{a})\| \leq K\|(s, a) - (\tilde{s}, \tilde{a})\|. \tag{5.1}$$

5.3 CONTINUITY-BASED CORRECTIVE LABELS

Our objective is to enhance the robustness of imitation learning methods by generating corrective labels that bring an agent from out-of-distribution states back to in-distribution states. We first define the desired *high quality corrective* labels to make imitation learning more robust in Sec. 5.3.1. Without the ground truth dynamics, we discuss how this can be done using a *learned* dynamics model. Importantly, to generate high-quality corrective labels using learned dynamics functions, the dynamics need to exhibit local continuity. We present a method to train a locally Lipschitz-bounded dynamics model in Sec. 5.3.2, and then show how to use such a learned model to generate corrective labels (Sec. 5.3.3) with high confidence, beyond the support of the expert training data. Finally, in Section 5.3.4, we instantiate these insights into a practical algorithm - **CCIL** that improves the robustness of imitation learning methods with function approximation.

5.3.1 *Corrective Labels Formulation*

To robustify imitation learning, we aim to provide corrections to disturbances by bringing the agent back to the "known" expert data distribution, where the policy has been trained and is likely to be successful. We generate state-action-state triplets $(s^{\mathcal{G}}, a^{\mathcal{G}}, s^*)$ that are *corrective*: executing action $a^{\mathcal{G}}$ in state $s^{\mathcal{G}}$ can bring the agent back to a state s^* in support of the expert data distribution.

[Corrective Labels]. $(s^{\mathcal{G}}, a^{\mathcal{G}}, s^*)$ is a corrective label triplet with starting state $s^{\mathcal{G}}$, corrective action label $a^{\mathcal{G}}$ and target state (in the expert data) $s^* \in \mathcal{D}^*$ if, with respect to dynamics function f,

$$\|[s^{\mathcal{G}} + f(s^{\mathcal{G}}, a^{\mathcal{G}})] - s^*\| \leq \epsilon_c. \tag{5.2}$$

This definition is trivially satisfied by the given expert demonstration $(s^*_j, a^*_j, s^*_{j+1}) \in \mathcal{D}^*$. We aim to search for a larger set of corrective labels for out-of-distribution states. If the policy has bounded error on not just the distribution of expert states but on a distribution with larger support, $\text{Support}(d^{\pi^*}) < \text{Support}(d^{\pi_{\text{aug}}})$, it is robust against a larger set of states that it might encounter during execution. However, without knowledge of the true system dynamics f, we have only an approximation $\hat{f}$ of the dynamics function derived from finite samples.

Definition 3.1. [High Quality Corrective Labels under Approximate Dynamic Models]. $(s^{\mathcal{G}}, a^{\mathcal{G}}, s^*)$ is a corrective label if $\|[s^{\mathcal{G}} + \hat{f}(s^{\mathcal{G}}, a^{\mathcal{G}})] - s^*\| \leq \epsilon_c$, w.r.t. an approximate dynamics $\hat{f}$ for a target state $s^* \in \mathcal{D}^*$. Such a label is "high-quality" if the approximate dynamics function also has bounded error w.r.t. the ground truth dynamics function evaluated at the given state-action pair $\|f(s^{\mathcal{G}}, a^{\mathcal{G}}) - \hat{f}(s^{\mathcal{G}}, a^{\mathcal{G}})\| \leq \epsilon_{co}$.

The high-quality corrective labels represent the learned dynamics model's best guess at bringing the agent back into the support of expert data. However, an approximate dynamics model is only reliable in a certain region of the state space, i.e., where the predictions of $\hat{f}$ closely align with the true dynamics of the system, f. When the

dynamics model is trained without constraints, such as in a maximum likelihood framework, its performance might significantly degrade when extrapolating beyond the distribution of the training data. To address this challenge, we will first describe how to learn locally continuous dynamics models from data. Subsequently, we will outline how to utilize these models for high-quality corrective label generation beyond the expert data.

5.3.2 *Learning Locally Continuous Dynamics Models from Data*

Our core insight for learning dynamics models and using them beyond the training data is to leverage the local continuity in dynamic systems. When the dynamics are locally Lipschitz bounded, small changes in state and/or action yield small changes in the transitions. A dynamics function that exhibit locally continuity would allow us to extrapolate beyond the training data to the states and actions that are in proximity to the expert demonstration - a region in which we can trust the learned model.

We follow the model-based reinforcement learning framework to learn dynamics models from data (Wang et al., 2019). Essentially, we learn a residual dynamics model $\hat{f}(s_t^*, a_t^*) \rightarrow s_{t+1}^* - s_t^*$ via mean-squared error regression (MSE). We can use any function approximator (e.g., neural network) and write down the learning loss:

$$\arg\min_{\hat{f}} \mathbb{E}_{s_j^*, a_j^*, s_{j+1}^* \sim \mathcal{D}^*} \left[\left\| \hat{f}(s_j^*, a_j^*) + s_j^* - s_{j+1}^* \right\| \right] \tag{5.3}$$

A model trained with the MSE loss alone is not guaranteed to have good extrapolation beyond the training data. Critically, we modify the above learning objective to ensure the learned dynamics model to contain regions that are locally continuous. Previously Shi et al. (2019) used spectral normalization to fit dynamics functions that are *globally* Lipschitz bounded. However, most robotics problems involve dynamics models that are hybrids of local continuity and discontinuity: for instance, making and breaking contacts. In face of discontinuity, we present a few practical methods to fit a dynamics model that (1) enforces as much local continuity as permitted by the data and (2) discards the discontinuous regions when subsequently generating corrective labels. Due to space limit, we defer the reader to App. 5.8.4 for a complete list of candidate loss functions considered. Here, we show an example of a sampling-based penalty of violation of local Lipschitz constraint.

Local Lipschitz Continuity via Sampling-based Penalty. Following (Gulrajani et al., 2017), we relax the global Lipschitz constraint by penalizing any violation of the local Lipschitz constraint.

$$\arg\min_{\hat{f}} E_{s_j^*, a_j^*, s_{j+1}^* \sim \mathcal{D}^*} \left[\text{MSE} + \lambda \cdot \mathbb{1}\left(\hat{f}'(s_j^*, a_j^*) > K \right) \right] \tag{5.4}$$

$$\hat{f}'(s_j^*, a_j^*) \approx \mathbb{E}_{\Delta_s \sim \mathcal{N}} \left[\frac{\| \hat{f}(s_j^* + \Delta_s, a_j^*) - \hat{f}(s_j^*, a_j^*) \|}{\| \Delta_s \|} \right] \tag{5.5}$$

Figure 5.3: **Generating corrective labels from demonstration.** Given expert demonstration state s^* and action a^* that arrives at s^*_{next}. BackTrack Labels: search for an alternative starting state $s^{\mathcal{G}}$ that can arrive at expert state s^* if it executes expert action a^*. DisturbedAction Labels: search for an alternative starting state $s^{\mathcal{G}}$ that would arrive at s^*_{next} if it executes a slightly disturbed expert action $a^{\mathcal{G}} = a^*_t + \Delta$, where Δ is a sampled noise.

We use samples to estimate the local Lipschtiz continuity $\hat{f}'(s^*_j, a^*_j)$, by perturbing the data points with small sampled Gaussian noises $\Delta_s \sim \mathcal{N}$. Alternatively, one can compute the Jacobian matrix to estimate the local Lipschitz continuity. The penalty term weighted by λ enforces the local Lipschitz constraint at the specific expert datapoint. Doing so ensures that the approximate model is mostly K Lipschitz-bounded while being predictive of the transitions in the expert data. This approximate dynamics model can then be used to generate corrective labels.

5.3.3 *Generating High-Quality Corrective Labels*

We employ learned dynamics models to generate corrective labels, leveraging local continuity to ensure the generated labels have bounded error in proximity to the expert data's support. When the local dynamics function is bounded by a Lipschitz constant w.r.t. the state-action space, we can (1) perturb the dynamics function by introducing slight variations in either state or action and (2) quantify the prediction error from the dynamics model based on the Lipschitz constant. We outline two techniques, BackTrack label and DisturbedAction label, to generate corrective labels (Fig. 5.3).

TECHNIQUE 1: BACKTRACK. Assuming local Lipschitz continuity w.r.t. states, given expert state-action pair s^*_t, a^*_t, we propose to find a different state $s^{\mathcal{G}}_{t-1}$ that can arrive at s^*_t using action a^*_t. To do so, we optimize for a state $s^{\mathcal{G}}_{t-1}$ such that:

$$s^{\mathcal{G}}_{t-1} + \hat{f}(s^{\mathcal{G}}_{t-1}, a^*_t) = s^*_t \tag{5.6}$$

The quality of the generated label $(s^{\mathcal{G}}, a^*_t, s^*_t)$ is bounded and we present the proof in Appendix 5.8.1.1.

Theorem 3.2. *When the dynamics model has a training error of ϵ on the specific data point, under the assumption that the dynamics models f and $\hat{f}$ are locally Lipschitz continuous w.r.t. state with Lipschitz constant K_1 and K_2 respectively, if $s^{\mathcal{G}}_t$ is in the neighborhood of local continuity, then*

$$\begin{aligned} \left\| f\left(s^{\mathcal{G}}_t, a^*_{t+1}\right) - \hat{f}\left(s^{\mathcal{G}}_t, a^*_{t+1}\right) \right\| \leq \\ \epsilon + (K_1 + K_2) \left\| s^{\mathcal{G}}_t - s^*_{t+1} \right\|. \end{aligned} \tag{5.7}$$

TECHNIQUE 2: DISTURBEDACTION. Assuming local Lipschitz continuity w.r.t., state-action, given an expert tuple $(s_t^*, a_t^*, s_{t+1}^*)$, we ask: Is there an alternative action $a^{\mathcal{G}}$ that slightly differs from the demonstrated action a_t^*, i.e., $a^{\mathcal{G}} = a_t^* + \Delta$, that can bring an imaginary state $s^{\mathcal{G}}$ to the expert state s_{t+1}^*? We randomly sample a small action noise Δ and solve for an imaginary previous state $s_t^{\mathcal{G}}$:

$$s_t^{\mathcal{G}} + \hat{f}(s_t^{\mathcal{G}}, a_t^* + \Delta) - s_{t+1}^* = 0. \tag{5.8}$$

For every data point, we can obtain a set of labels $(s_t^{\mathcal{G}}, a_t^* + \Delta)$ by randomly sampling the noise vector Δ. We show that the quality of the labels is bounded (proof in App. 5.8.2).

Theorem 3.3. *Given $s_{t+1}^* - \hat{f}(s_t^{\mathcal{G}}, a_t^* + \Delta) - s_t^{\mathcal{G}} = 0$ and that the dynamics function f is locally Lipschitz continuous w.r.t. states and actions, with Lipschitz constants K_A and K_S respectively, then*

$$\begin{aligned} \|f(s_t^{\mathcal{G}}, a_t^* + \Delta) - \hat{f}(s_t^{\mathcal{G}}, a_t^* + \Delta)\| \leq \\ K_A\|\Delta\| + (1 + K_S)\|\epsilon\|. \end{aligned} \tag{5.9}$$

SOLVING THE ROOT-FINDING EQUATION IN GENERATING LABELS Both our techniques need to solve root-finding equations (Eq. 5.6 and 5.8). This suggests an optimization problem: $\arg\min_{s^{\mathcal{G}}} \|s^{\mathcal{G}} + \hat{f}(s^{\mathcal{G}}, a^{\mathcal{G}}) - s^*\|$ given $\hat{f}$ and $a^{\mathcal{G}}, s^*$. There are multiple ways to solve this optimization problem (e.g., gradient descent or Newton's method). For simplicity, we choose a fast-to-compute and conceptually simple solver, Backward Euler, widely adopted in modern simulators (Todorov, Erez, and Tassa, 2012b). It lets us use the gradient of the next state to recover the earlier state. To generate data given $a^{\mathcal{G}}, s^*$, Backward Euler solves a surrogate equation iteratively: $s^{\mathcal{G}} \leftarrow s^* - \hat{f}(s^*, a^{\mathcal{G}})$.

LABEL ERROR. Theorems 3.2 and 3.3 show that the error of the labels generated with these techniques are bounded. If we can compute this bound, we can individually determine the quality of each generated label. To do so, we first consider how to compute the local Lipschitz coefficient of the learned dynamics model.

Remark 3.4. Consider a function $f(s, a)$ that is locally Lipschitz-continuous around (s, a) with coefficient K. Letting J_f notate the jacobian of f, then:

$$K \approx \|J_f(s, a)\|_2 \tag{5.10}$$

Note that we can easily compute the jacobian of our learned dynamics using auto-differentiation libraries. Then, with some suitable approximations, we can use our derived error bounds to calculate the error of each generated label.

Definition 3.5 (BackTrack Label Error)**.** Consider Theorem 3.2. We can assume $\epsilon \approx 0$ by only generating labels for data points with low model error, and also that $K_2 \propto K_1$. We define the BackTrack label error to be

$$\mathcal{E}_{\text{BT}} = \|J_{\hat{f}}(s_t^*, a_t^*)\|_2 \cdot \|s_t^{\mathcal{G}} - s_t^*\| \tag{5.11}$$

Definition 3.6 (DisturbedAction Label Error)**.** Consider Theorem 3.3. We can assume $K_A \propto \|J^A_{\hat{f}}(s^*_t, a^*_t)\|_2$ and $K_S \propto \|J^S_{\hat{f}}(s^*_t, a^*_t)\|_2$, where $J^A_{\hat{f}}$ and $J^S_{\hat{f}}$ are the Jacobians of $\hat{f}$ with respect to actions and states, respectively. We define the DisturbedAction label error to be

$$\mathcal{E}_{\text{DA}} = \|J^A_{\hat{f}}(s^*_t, a^*_t)\|_2 \cdot \|\Delta\| + (1 + \|J^S_{\hat{f}}(s^*_t, a^*_t)\|_2) \cdot \|s^{\mathcal{G}}_t - s^*_t\| \tag{5.12}$$

LABEL FILTERING. Now that we can calculate the quality of each generated label, we can reject labels that result in a large error bound, as defined by Definitions 3.5 and 3.6. In practice, we calculate the label error for all generated labels, and filter out some fraction of the labels with highest associated error. By setting the label error filtering threshold as some quantile, we can directly control the size of the error bound.

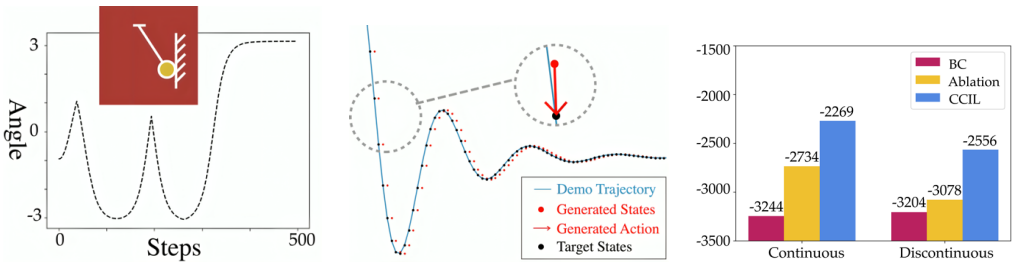
5.3.4 *CCIL: Continuity-based Corrective Labels for Imitation Learning*

We instantiate a practical version of our proposal, **CCIL** (**C**ontinuity-based data augmentation to generate **C**orrective labels for **I**mitation **L**earning)with three steps: (1) fit an approximate dynamics function that exhibit local continuity, (2) generate corrective labels using the learned dynamics, and (3) use rejection sampling to select labels with the desired error bounds. To use the generated data $D^{\mathcal{G}}$, we simply combine it with the expert data and train policies using standard behavior cloning on the augmented dataset. We evaluate the augmented agent to empirically test whether training with synthetic corrective labels would help imitation learning.

5.4 WARM-UP: SIMULATION EXPERIMENTS

As a warm-up to real-world robot experiments, we evaluate CCIL over 4 simulated robotic domains of 14 tasks to answer the following questions:
Q1. Can we empirically verify the theoretical contributions on the quality of the generated labels;
Q2. How well does CCIL handle discontinuities in environmental dynamics;
Q3. How would training with CCIL-generated labels affect imitation learning agents' performance.

We compare CCIL to standard Behavior Cloning (BC) (Pomerleau, 1988a), NoisyBC (Ke et al., 2021b), MILO (Chang et al., 2021) and MOREL (Kidambi et al., 2020) using the same model architecture across all methods, over 10 random seeds. Note that MOREL requests additional access to reward function. Appendix. 5.8.7 contains details to reproduce the experiments.



(a) The Discontinuous Pendulum (b) Example generated labels (c) Performance Comparison

Figure 5.4: Evaluation on the Pendulum and Discontinuous Pendulum Task.

5.4.1 *Analysis on Classic Control of Pendulum and Discontinuous Pendulum*

We consider the classic control task, the pendulum, and also consider a variant, The Discontinuous Pendulum, by bouncing the ball back at a few chosen angles, as shown in Fig. 5.4a.

Verifying the quality of the generated labels (Q1). We visualize a subset of generated labels in Fig. 5.4b. Note that the generated states (red) are outside the expert support (blue) and that the generated actions are torque control signals, which are challenging for invariance-based data augmentation methods to generate. To quantify the quality of the generated labels, we use the fact that ground truth dynamics can be analytically computed for this domain and measure how closely our labels can bring the agent to the expert. We computed the empirical error using the average L2 norm distance and obtained 0.02367, which validates our theoretical bound of 0.065: $K1|\Delta| + (1 + K2)|s_t^* - s_t^G| = 12 * 0.0001 + 13 * 0.005$ (Equation 5.7).

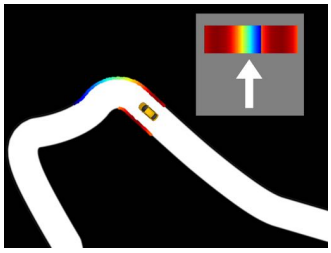
The Impact of Local Lipschitz Continuity Assumption (Q2) To highlight how local discontinuity in the dynamics affects CCIL, we compare CCIL performance in the continuous and discontinuous pendulum in Fig. 5.4c: CCIL improved behavior cloning performance even when discontinuity is present, albeit with a smaller margin for the discontinuous Pendulum. In an ablation study, we also tried generating labels using a naive dynamics model (without explicitly assuming Lipschitz continuity) which performed slightly better than vanilla behavior cloning but worse than CCIL, highlighting the importance of enforcing local Lipschitz continuity in training dynamics functions for generating corrective labels.

CCIL improved behavior cloning agent (Q3). For both the continuous and discontinuous Pendulum, CCIL outperformed behavior cloning given the same amount of data (Fig. 5.4c).

5.4.2 *Driving with Lidar: High-dimensional state input*

We compare all agents on the F1tenth racing car simulator (Fig. 5.5a), which employs a high-dimensional, LiDAR sensor as input (Fig. 5.5b). We evaluate each agent over 100 trajectories. At the beginning of each trajectory, the car is placed at a random location on the track. It needs to use the LiDAR input to decide on speed and steering, earning scores for driving faster or failing by crashing.

(a) F1tenth



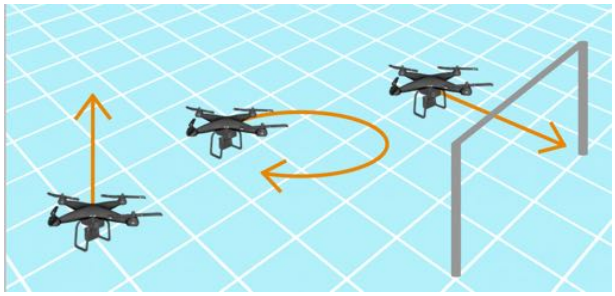
(b) LiDar POV

Method	Succ. Rate	Avg. Score
Morel	0.0%	0.001 ± 0.001
MILO	0.0%	0.21 ± 0.003
BC	31.9%	0.58 ± 0.25
NoiseBC	39.3%	0.62 ± 0.28
CCIL	**89.6%**	**0.95** ± 0.08

(c) Performance on Racing

CCIL can improve the performance of behavior cloning for high-dimensional state inputs (Q2, Q3). Table. 5.5c shows that CCIL demonstrated an empirical advantage over all other agents, achieving a higher success rate and better score while having fewer crashes. Noticeably, LiDar inputs can contain highly-complicated discontinuities and impose challenges to model-based methods (Morel and MILO), while CCIL could reliably generate corrective labels with confidence.

5.4.3 *Drone Navigation: High-Frequency Control Task and Sensitive to Noises*



(a) Drone Flying Tasks

Method	Hover	Circle	FlyThrough
BC	-1.08E8	-9.56E7	-1.06E8
NoisyBC	-1.13E8	-9.88E7	-1.07E8
Morel	-1.25E8	-1.24E8	-1.25E8
MILO	-1.26E8	-1.25E8	-1.25E8
CCIL	-0.77E8	-3.49E7	-0.41E8

(b) Performance on Drone

Drone navigation is a high-frequency control task and can be very sensitive to noise, making it an appropriate testbed for the robustness of imitation learning. We use an open-source quadcopter simulator, gym-pybullet-drone (Panerati et al., 2021) and consider three proposed tasks: hover, circle, fly-through-gate, as shown in Fig. 5.6a. Following Shi et al. (2019), we inject observation and action noises during evaluation to highlight the robustness of the learner agent.

CCIL improved performance for imitation learning agent and its robustness to noises (Q3). CCIL outperformed all baselines by a large margin.

5.4.4 *Locomotion and Manipulation: Diverse Scenes with Varying Discontinuity*

Manipulation and locomotion tasks commonly involve complex forms of contacts, raising considerable challenges for learning dynamics models and for our proposal. We evaluate the applicability of CCIL in such domains, considering 4 tasks from MuJoCo locomotion suite: Hopper, Walker2D, Ant, HalfCheetah and 4 tasks from the

MetaWorld manipulation suites: CoffeePull, ButtonPress, CoffeePush, DrawerClose. During evaluation, we add a small amount of randomly sampled Gaussian noise to the sensor (observation state) and the actuator (action) to simulate real-world conditions of robotics controllers and to test the robustness of the agents.

CCIL outperforms behavior cloning or at least achieves comparable performance even when varying form of discontinuity is present (Q2, Q3). On 4 out of 8 tasks considered, CCIL outperforms all baselines. Across all tasks, CCIL at least achieves comparable results to vanilla behavior cloning, shown in Table. 5.1, indicating that it's a effective alternative to behavior cloning without necessitating substantial extra assumptions.

Table 5.1: Evaluation results for Mujoco and Metaworld tasks with noise disturbances. We list the expert scores in a noise-free setting for reference. In the face of varying discontinuity from contacts, CCIL remains the leading agent on 4 out of 8 tasks (Hopper, Walker, HalfCheetah, CoffeePull). Comparing CCIL with BC: across all tasks, CCIL can outperform vanilla behavior cloning or at least achieve comparable performance.

	Mujoco				**Metaworld**			
	Hopper	Walker	Ant	Halfcheetah	CoffeePull	ButtonPress	CoffeePush	DrawerClose
Expert	3234.30	4592.30	3879.70	12135.00	4409.95	3895.82	4488.29	4329.34
BC	1983.98 ± 672.66	1922.55 ± 1410.09	2965.20 ± 202.71	8309.31 ± 795.30	3552.59 ±233.41	**3693.02** ± 104.99	1288.19± 746.37	3247.06 ± 468.73
Morel	152.19±34.12	70.27 ± 3.59	1000.77 ± 15.21	-2.24 ± 0.02	18.78±0.09	14.85±17.08	18.66± 0.02	1222.23± 1241.47
MILO	566.98±100.32	526.72±127.99	1006.53±160.43	151.08±117.06	232.49± 110.44	986.46±105.79	230.62±19.37	**4621.11**±39.68
NoiseBC	1563.56 ± 1012.02	2893.21 ± 1076.89	**3776.65** ± 442.13	8468.98 ±738.83	3072.86 ± 785.91	**3663.44**±63.10	**2551.11**± 857.79	4226.71± 18.90
CCIL	**2784.45** ± 188.56	**3724.89** ± 558.01	3546.32 ± 338.84	**9057.13** ± 425.47	**3949.54** ± 252.34	**3827.60**±24.37	**2551.18**± 770.02	4187.90± 44.40

5.4.5 *Summary*

We conducted extensive experiments to address three research queries. For **Q1**, we verified the proposed theoretical bound on the quality of the generated labels on the Pendulum task. For **Q2**, we tested that CCIL is robust to environmental discontinuities, improving behavior cloning in both continuous (Pendulum, Drone) and discontinuous scenarios (Driving, MuJoCo). For **Q3**, CCIL emerged as the best agent on 10 out of 14 tasks. In the rest of tasks, it achieved comparable performance to behavior cloning, demonstrating its cost-effectiveness and efficiency as an alternative approach without requiring significant additional assumptions.

5.5 REAL-WORLD EXPERIMENT DESIGN

5.5.1 *Motivation*

We see that the CCIL framework shows notable success in various simulation domains. However, its application to real-world scenarios hinges on several critical questions:

Local Continuity in Dynamics and Real-World Application CCIL relies on local Lipschitz continuity in system dynamics, yet real-world robotic tasks often involve discontinuities due to physical contact. Can this foundational assumption be validated

in realistic domains? Can CCIL still enhance imitation learning agent performance amidst such challenges?

Data Scarcity and Augmentation Impact Limited real-world robot demonstrations highlight the importance of efficient data augmentation. While abundant data typically guarantees better outcomes, the critical question lies in the low data regime: How effectively can CCIL address data scarcity, and what data volume is required to observe a tangible impact?

Sensitivity to Hyperparameters Tuning imitation learning algorithms on physical robots is logistically challenging. Direct execution on robots, while being the most reliable evaluation method, risks unpredictable behaviors and necessitates numerous real-world trials to achieve statistical significance. How sensitive is CCIL to hyperparameter variations? Can we offer guidelines for its real-world application to mitigate the high evaluation and tuning costs?

5.5.2 *Hypotheses*

We explore the following hypotheses:

- **H1:** CCIL can improve imitation learning policies in real-world fine-manipulation tasks. This improvement is statistically significant, especially in low-data regimes.
- **H2:** Real-world tasks with complex contacts and discontinuities in dynamics can still exhibit local Lipschitz continuity to justify CCIL's assumptions.
- **H3:** The performance of CCIL is highly sensitive to the label rejection hyperparameter, which determines the acceptable error bound for generated labels.
- **H4:** The performance of CCIL exhibits relative robustness to variations in the local Lipschitz constraint enforced during the training of the dynamics model.

For **H.1**, we compare the success rate of a BC agent trained with and without CCIL-generated labels. We vary the amount of demonstration data to investigate the impact of data quantity on these methods.

For **H.2**, we measure the local Lipschitz continuity of the trained dynamics model for tasks with complex discontinuity. We also vary the hyperparameter of the upper bound on the local Lipschitz constant.

For **H.3** and **H.4**, we run ablation studies varying the hyper-parameter of label rejection threshold and Lipschitz constraint to investigate (1) their impact on the success rate of CCIL and (2) the interplay between these hyper-parameters.

5.5.3 *Tasks and Data Collection*

We consider three fine manipulation tasks (Fig. 5.2) that require a millimeter level of precision.

- *GraspCube* - Grasp and lift a tiny 1cm cube above the table. Despite being the easiest of all three tasks, the task is non-trivial with scarce-data (e.g., 100 trajectories).

- *GearInsertion* - Inspired by industrial assembly (Gubbi, Kolathaya, and Amrutur, 2020), we design a mini-gear insertion task. The agent needs to insert a Lego gear into a hole on a board. Proper insertion leaves a gap less than 0.2 mm.
- *GraspCoin* - Using chopsticks to grasp and lift a metal coin lying flat on the table. The coin's round shape and slippery texture makes it hard even for human experts to pick up using chopsticks.

For all tasks, we vary the initial positions of the object. To collect demonstrations, we use a mix of heuristic controllers and human teleoperation. For *GraspCube* and *GearInsertion*, we designed heuristic controllers and collected 500 trajectories for *GraspCube* and 100 trajectories for *GearInsertion*. Note that the heuristic controllers did not have perfect success rates and we filtered out failed demonstrations. For *GraspCoin*, it was non-trivial to design a heuristic controller due to the difficulty of the task, we instead used successful demonstrations from an expert teleoperation. Due to the difficulty of the task, we only obtained 200 trajectories.

5.5.4 *Training*

For each task we train two behavior cloning agents, with and without CCIL-generated labels. Noticably, we formulate the action loss for our hardware and detail the parameter tuning procedure.

Loss Formulation To train a policy using behavior cloning, we need to design an action loss $L(a^*, \hat{a})$. We denote the action a for our hardware as (x, q, c) representing the end effector xyz location, orientation, and chopstick angle. Given the target action $a^* = (x, q, c)$ and the predicted action $\hat{a} = (\hat{x}, \hat{q}, \hat{c})$, the action loss is:

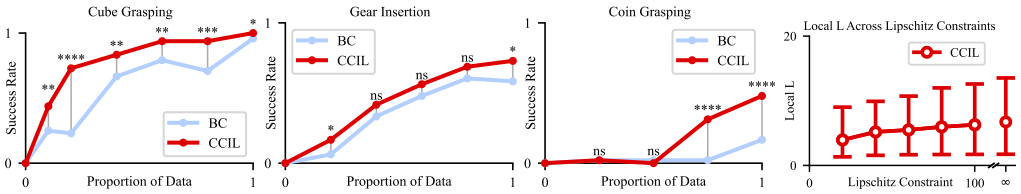
$$L(a, \hat{a}) = \alpha_1 \|x - \hat{x}\|^2 + \alpha_2 \theta^2 + \alpha_3 (c - \hat{c})^2 \tag{5.13}$$

Where θ is the angle between the orientations q and $\hat{q}$, and $\alpha_1, \alpha_2, \alpha_3$ weights each component of the loss function. In our experiments, we use $\alpha_1 = 10, \alpha_2 = 1, \alpha_3 = 10$.

Parameter Tuning Applying CCIL introduces two key parameters: the Lipschitz constraint for training the dynamics model and a threshold for filtering the generated labels. The Lipschitz constraint, K (Eq. 5.4), upper bounds the Lipschitz continuity of the learned dynamics model. A loose upper bound makes the training process easier and more likely to yield low training error. Conversely, a tighter bound is more likely to yield a dynamics model with lower Lipschitz coefficients, enabling CCIL to generate higher-quality corrective labels. The label error bound is defined in Definitions 3.5 and 3.6, and the threshold σ controls the acceptable bound.

To set the Lipschitz constraint K, we first train an unconstrained model and analyze the distribution of local Lipschitz coefficients on the learned model. If the distribution of coefficient suggests that the learned model has limited local Lipschitz continuity, we choose a tighter Lipschitz constraint.

To set threshold σ, we first generate corrective labels without filtering and analyze the distribution of label errors. Based on the distribution, we choose a label rejection quantile (between 0 to 1, where 0 means to filter out all generated labels) that filters out outliers or long tails.



(a) CCIL generally provides a performance boost over BC, especially in low-data regimes.

(b) CCIL is robust to Lipschitz.

Figure 5.7: (a) As the amount of data increases, CCIL provides a smaller boost over BC. We use asterisks to denote statistical significance: * p $<$ 0.1, ** $p < 0.05$, *** $p < 0.01$, and **** $p < 0.001$. *ns* denotes $p \geq 0.1$. (b) Mean and middle 95% of the local Lipschitz coefficients of the learned dynamics model across the demonstration dataset as the Lipschitz constraint increases. As the enforced constraint increases, the distribution of coefficients converges.

5.5.5 *Evaluation*

We test each agent's success rate by conducting 48 trials per agent to establish statistical significance. To ensure fairness, for each task we select 16 fixed initial conditions (i.e., for grasping agents, we place the object to grasp at 16 fixed positions across the workplace). For each initial condition, we test the learned policy for 3 trials. For each trial, we denote the success as a binary variable. To report the statistical significance of the empirical results, we compare the success rates between the two policies by performing two-proportion z-tests.

5.6 RESULTS

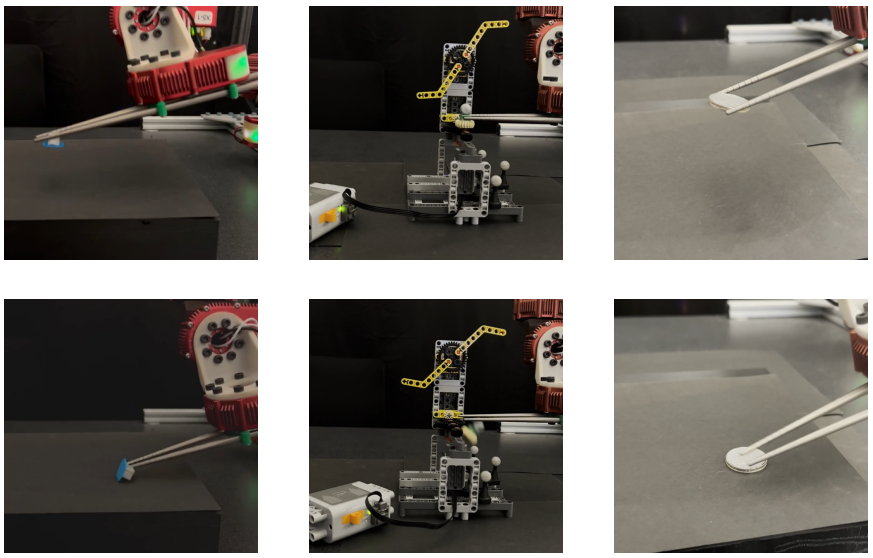
Across three tasks, we trained 36 agents and tested their success rate in the real world (Fig. 5.7a). For ablation, we further evaluated 33 agents on the GraspCube task.

5.6.1 *Corrective labels' improvement to imitation learning*

H.1 investigates whether CCIL can improve imitation learning performance in real-world fine-manipulation tasks. These tasks involve complex contact dynamics between objects, end-effectors, and the workspace, making it unclear if CCIL's assumptions hold.

CCIL yields an increase in performance for BC. Fig. 5.7a shows that training with labels generated by CCIL generally improves the performance of BC over all three tasks, as measured by success rate. We validated the statistical significant of the improvement. The performance boost is significant at the $p < 0.05$ level for the cube grasping and coin grasping tasks.

CCIL boost is significant in low-data regime. The empirical performance gain from applying CCIL is statistically significant in low-data regimes, as shown in Fig. 5.7a. CCIL could boost BC performance from 23% to 83% (using 100 GraspCube trajectories),



(a) CCIL (top) grasps the cube while BC (bottom) knocks it over.

(b) CCIL (top) inserts the gear while BC (bottom) misses the hole.

(c) CCIL (top) picks up the coin while BC (bottom) misses.

Figure 5.8: Comparison of CCIL (top) and BC (bottom) across fine-manipulation tasks.

from 6% to 17% (using 20 GearInsertion trajectories) or 58% to 72% (using 100 GearInsertion trajectories) and from 17% to 48% (using 200 GraspCoin trajectories). These limited-data regimes face significant challenges from covariate shift due to limited data support from expert demonstrations. CCIL demonstrated promising results to alleviate this problem.

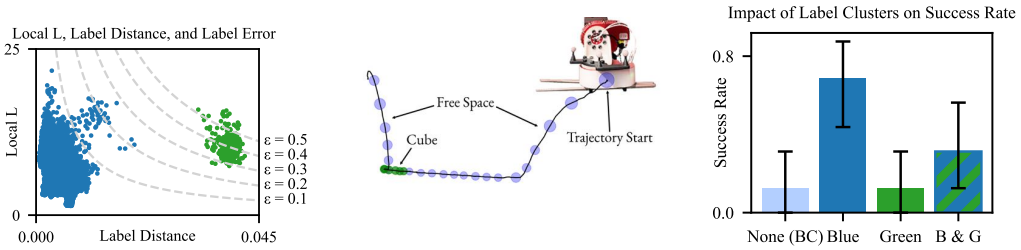
Remark Our experiments validate how CCIL can be applied to complex fine manipulation tasks in the real world, despite the presence of discontinuity in the contact dynamics.

5.6.2 *CCIL's Assumptions on Local Lipschitz Continuity*

H.2 investigates whether CCIL's local continuity assumptions are valid and realistic to be applied to real-world fine manipulation tasks.

Real-world tasks contain local Lipschitz continuity that can be realized by learning dynamics models. In training the dynamics model, we experiment with different Lipschitz constraints, measure the resulting models' local Lipschitz coefficient and plot the average of the coefficient in Fig. 5.7b. With a large upper bound on the Lipschitz constraint (loose constraint), the local coefficients increase but converge to that of an unconstrained model. These findings imply that our environments exhibit local continuity that the learned dynamics models are fitting to, even without explicitly enforcing the Lipschitz continuity coefficient.

The learned dynamics model and generated labels have varying continuity that correlates with the real world. For each generated label, we can measure the local Lipschitz coefficients using the learned dynamics model ("Local L"). We generate



(a) Clusters reveal dynamics regimes.

(b) High label rejection clusters around contact.

(c) Mix quality labels can hinder learning.

Figure 5.9: The 20% dataset for the *GraspCube* task reveals two distinct clusters of corrective labels (a). The green cluster mostly corresponds to labels where the cube is being manipulated, and the blue cluster mostly corresponds to arm free space motion (b). Policies trained using just the blue cluster (low label rejection threshold) are more successful when compared with those trained with the green cluster or both (high label rejection threshold (c).

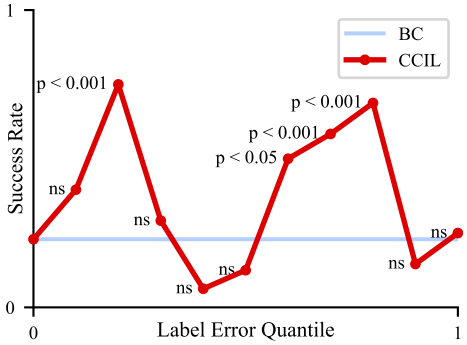


Figure 5.10: Policy performance when filtering out varying fractions of the generated labels by label error.

a scatterplot for the generated labels in Fig. 5.9a. We see two clusters (green and blue). We then sample some of the labels for visualization and plot them along an expert trajectory in Fig. 5.9b. We see the green cluster correspond to labels generated near the cube and the blue labels are mostly in free space when the robot moves without collision. Intuitively, the learned model and generated label have higher errors for the green group that contains discontinuity.

We test how the policy performance changes as we incorporate (1) blue labels, (2) green labels, and (3) both labels into the demonstration dataset, as shown in Fig. 5.9c. Training with blue labels improved the agent's success rate. However, training with the green labels associated with contact-rich states and higher errors is not as useful. Such labels actually hinder performance. This highlights how label quality can impact CCIL performance and the significance of filtering generated labels.

5.6.3 *Impact of Quality of Generated Labels*

The label filtering threshold controls what generated labels are used for when training a policy. It is crucial to understand its effects and how to tune it properly. We proposed a practical way to enforce filtering threshold via label rejection quantile, **H.3** explores how this design choice affects CCIL's performance.

We choose the GraspCube task with 100 trajectories. We evaluate running CCIL with different label rejection quantiles and report the success rate in Fig. 5.10. The rejection quantile controls how many generated labels are used (25%, 75%, etc) based on their computed error bound.

Low-quality labels can harm policy performance. Using all generated labels (choosing a quantile close or equal to 1) makes the performance of CCIL similar to BC or worse. With this quantile, the generated labels have high error bounds and could be incorrect under the true dynamics. If these labels are included when training the policy, they could potentially hinder policy performance.

Need to balance label error and usefulness for CCIL to succeed. Labels with lower error bounds are more trustworthy. However, these labels tend to be closer to the demonstration data (small label distance). Including labels with conservative error bounds may fail to expand the data support. Conversely, a label that is further from the demonstration data can expand the data support while potentially introducing a higher associated error. Fig. 5.10 shows intermediate values of the label error quantile achieving a higher success rate, indicating that one needs to balance between label accuracy and usefulness to maximize CCIL's performance.

5.6.4 *Impact of Continuity Constraint*

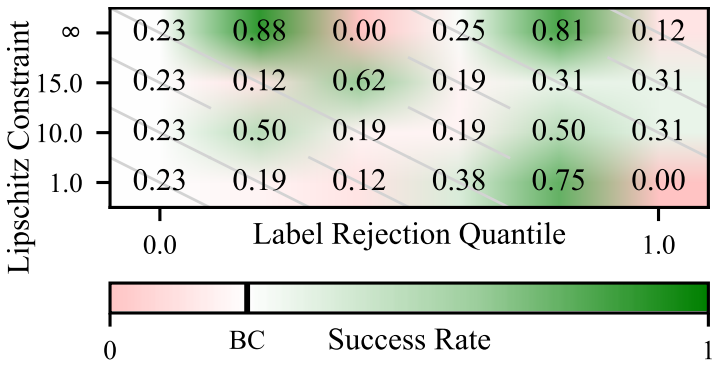


Figure 5.11: Hyperparameter ablation for CCIL, varying the Lipschitz constraint and the label rejection quantile. Green cells indicate a performance boost over BC, while red cells are worse. Crossed out cells are not significant at the $p < 0.05$ level, while the others are.

H.4 asks how enforcing Lipschitz continuity for training the dynamics model impacts CCIL's performance.

We train the dynamics model using different Lipschitz constraints; For each model, we generate labels and test the performance of CCIL with various label error quantiles. All hyperparameters other than those ablated upon are fixed. We illustrate the resulting success rate in Fig. 5.11 and cross the cells whose performance are not statistically significant compared to BC.

CCIL is robust to different Lipschitz constraints. Interestingly, we see that for each Lipschitz constraint, there is a label rejection quantile that achieves a significant boost in performance over BC. This indicates that CCIL could work well for a wide range of constraints, making it robust to this parameter.

CCIL can work with an unconstrained dynamics model. Surprisingly, even using an unconstrained dynamics model, indicated by the Lipschitz constraint of ∞, CCIL still achieves good performance with certain quantile. We investigate this phenomenon in Fig. 5.7b and observe that the unconstrained model still exhibits local continuity over the demonstration data, as shown by the relatively small local Lipschitz coefficients across the dataset. Interestingly, we observe that as the Lipschitz constraint increases, the distribution of local Lipschitz coefficients converges to that of the unconstrained model.

These findings together imply that in environments with sufficient local continuity, explicit Lipschitz constraint enforcement may be unnecessary and tuning label rejection quantile can be more critical. When applying CCIL to new environments, one can start with unconstrained dynamics model for their simplicity.

5.7 DISCUSSION

In this work, we introduce a novel framework that leverages local continuity in environmental dynamics to train dynamic functions from expert demonstrations and generate corrective labels. These generated labels encourage imitation learning agents to stay within the support of the demonstration data. Our key insight is to exploit local continuity which provide theoretical guarantees on where to trust the learned the dynamics models beyond the data support. We demonstrate the efficacy of our approach across diverse simulation tasks with varying state representations and action spaces, and conduct extensive evaluations and ablations on multiple fine-manipulation tasks on a real robotic platform. Our experiments empirically validate that teaching agent to correct their actions can have larger improvement than only teaching them to accomplish the tasks.

However, our method is not without limitations. One challenge is to train better dynamics models from expert demonstrations, which is an open research frontier. Another challenge is applying our core assumption of local Lipschitz continuity in the dynamics to domains with highly discontinuous representations, such as pixel-based image spaces. Finally, we have yet to explore whether our generated labels can augment imitation learning agents that use highly expressive policy classes like transformer or diffusion policies. Despite these limitations, our work paves the way for many exciting

avenues of future research in model-based imitation learning, aiming to overcome these challenges and expand the scope of CCIL.

5.7.1 *Learning Dynamics Models for Imitation Learning*

We are not the first to use learned dynamics models to combat covariate shift in offline imitation learning. Other works such as Chang et al. (2021) mark a family of methods that aim to leverage model learning to aid in the imitation learning setting. We, however, have the added complexities of learning a continuous model, while also relying solely on expert demonstrations, which have very limited coverage.

Whereas prior works have used additional offline demonstrations to increase data coverage and therefore accuracy of the learned model, we seek to exploit structure in the environment instead of requiring extra data. We reason that continuity in the environment can provide similar benefits as the supplemental offline dataset, due to our derived label error bounds. We have shown that a relatively simple method of model learning, namely a MLP architecture with Lipschitz regularization, works well and generates accurate labels near the expert data, but future work could increase the accuracy of learned models even further from the training data, which would allow CCIL to increase state coverage even more.

Beyond our usage of corrective labels, efficiently learning accurate dynamics models from offline data remains an opportunity for exciting new avenues of research. Learned models allow data-driven methods to build an understanding of the environment, which we can exploit to build powerful and robust robotic agents. Further development on offline model learning can unlock powerful methods with minimal requirements and assumptions, greatly increasing robotic capabilities in the real world.

5.7.2 *Highly Discontinuous State*

CCIL works well in real robotic tasks by exploiting local continuity in system dynamics. However, its key assumption relies on having access to a state representation with locally continuous properties. Images and other high-dimensional representations can exhibit varying discontinuities with respect to the dynamics, which can limit CCIL's applicability to these problem settings.

We posit that there can exist local continuity in the underlying system dynamics despite the representation being discontinuous. If we can extract a latent representation with sufficient local continuity from the raw observation, it would enable the application of CCIL on the latent space. For example, learning embeddings for image-based robotic tasks have been widely studied (Nair et al., 2022; Radford et al., 2021) and one can employ similar strategies for extracting latent representations. However, prior methods do not consider local continuity with respect to the system dynamics which would be critical for CCIL.

Learning a latent dynamics model while enforcing local continuity remains an open research question. Solutions can enable the application of CCIL and greatly increase its

applicability. Additionally, it would further our understanding of learning dynamics models for complex environments with highly discontinuous dynamics where low-dimensional states cannot be easily estimated, paving the way for other model-based methods (Chang et al., 2021; Kidambi et al., 2020) to make robotics learning algorithms more robust and efficient.

5.7.3 *Application to Other Policy Classes*

Recent works (Chi et al., 2023; Zhao et al., 2023) have explored imitation learning policies that use transformer or diffusion policies to output sequences of actions ("chunking") under the assumption of positional control. Applying CCIL to these novel policy architectures and output spaces could be valuable for understanding how CCIL can enhance data efficiency and robustness for these methods.

However, most of these works assume a specific action space that provides some form of invariance: position control. Under this action space, small perturbations to the input state can directly reuse the positional action label. These stronger assumption enable prior works to come up with alternative strategies to boost robustness. CCIL does not make this assumption and is compatible with this action space, so it remains a question whether its use could further improve the agent performance.

It is therefore a valuable avenue for future research to investigate how CCIL can be used in conjunction with observation and action chunking, and how much it is impacted by the use of a position-control action space. Answering these questions will further our understanding of CCIL's generalizability, and how it fits in with other state-of-the-art imitation learning methods and techniques.

5.8 SUPPLEMENTARY

5.8.1 *Proofs*

5.8.1.1 *Proof of Theorem 3.2*

Notation. Let f be the ground truth 1-step residual dynamics model, and let $\hat{f}$ be the learned approximation of f. Given a demonstration (s^*_{t+1}, a^*_{t+1}), we have generated a corrective label $(s^{\mathcal{G}}, a^*_{t+1})$.

Assumptions:

1. The estimation error of the learned dynamics model *at the training data* is bounded.
   $\|f(s^*_{t+1}, a^*_{t+1}) - \hat{f}(s^*_{t+1}, a^*_{t+1})\| \leq \epsilon$.
2. $\hat{f}$ is locally K_1-Lipschitz in a neighborhood $\overline{\mathcal{U}}$ around (s^*_{t+1}, a^*_{t+1}). i.e., for $s^{\mathcal{G}}_t \in \overline{\mathcal{U}}$:
   $\|\hat{f}(s^{\mathcal{G}}_t, a^*_{t+1}) - \hat{f}(s^*_{t+1}, a^*_{t+1})\| \leq K_1 \|s^{\mathcal{G}}_t - s^*_{t+1}\|$
3. f is locally K_2-Lipschitz in a neighborhood U around (s^*_{t+1}, a^*_{t+1}). i.e., for $s^{\mathcal{G}}_t \in U$:
   $\|f(s^{\mathcal{G}}_t, a^*_{t+1}) - f(s^*_{t+1}, a^*_{t+1})\| \leq K_2 \|s^{\mathcal{G}}_t - s^*_{t+1}\|$
4. $s^{\mathcal{G}}_t$ is within the region of local continuity around s^*_{t+1} for both f and $\hat{f}$:
   $s^{\mathcal{G}}_t \in U_{s^*_{t+1}} \cap \overline{\mathcal{U}}_{s^*_{t+1}}$.

Proof:

$$
\begin{aligned}
&\|f(s^{\mathcal{G}}_t, a^*_{t+1}) - \hat{f}(s^{\mathcal{G}}_t, a^*_{t+1})\| \\
&= \|f(s^{\mathcal{G}}_t, a^*_{t+1}) - f(s^*_{t+1}, a^*_{t+1}) + f(s^*_{t+1}, a^*_{t+1}) - \hat{f}(s^*_{t+1}, a^*_{t+1}) + \hat{f}(s^*_{t+1}, a^*_{t+1}) - \hat{f}(s^{\mathcal{G}}_t, a^*_{t+1})\| \\
&\leq \|f(s^{\mathcal{G}}_t, a^*_{t+1}) - f(s^*_{t+1}, a^*_{t+1})\| + \|f(s^*_{t+1}, a^*_{t+1}) - \hat{f}(s^*_{t+1}, a^*_{t+1})\| + \|\hat{f}(s^*_{t+1}, a^*_{t+1}) - \hat{f}(s^{\mathcal{G}}_t, a^*_{t+1})\| \\
&\leq \epsilon + (K_1 + K_2)\|s^{\mathcal{G}}_t - s^*_{t+1}\|
\end{aligned}
$$

Remark Our assumption (1) about the error of the learned dynamics model is not a global constraint but simply requires the model to have a small prediction error *at the specific data point*. Our assumptions (2) and (3) about the size of the local Lipschitz continuity is not a requirement on the global Lipschitz continuity in the dynamics, but simply requires space *near the expert support* to exhibit local continuity. Our proof leverages simple triangle inequality and is valid only near expert data support.

5.8.2 *Proof of Theorem 3.3*

Notation. Let f be the ground truth 1-step residual dynamics model, and let $\hat{f}$ be the learned approximation of f. Given a demonstration $(s^*_t, a^*_t, s^*_{t+1})$, we have generated a corrective label $(s^{\mathcal{G}}, a^*_t + \Delta, s^*_{t+1})$.

Assumptions

1. f is locally K_S-Lipschitz in *state* in a neighborhood U_S around (s_t^*, a_t^*).
   i.e., for $s_t^{\mathcal{G}} \in U_S$:
   $\|f(s_t^{\mathcal{G}}, a_t^*) - f(s_t^*, a_t^*)\| \leq K_S \|s_t^{\mathcal{G}} - s_t^*\|$.
2. f is locally K_A-Lipschitz in *action* in a neighborhood U_A around $(s_t^{\mathcal{G}}, a_t^*)$.
   i.e., for $a_t^* + \Delta \in U_A$:
   $\|f(s_t^{\mathcal{G}}, a_t^*) - f(s_t^{\mathcal{G}}, a_t^* + \Delta)\| \leq K_A \|\Delta\|$.
3. Using rejection sampling, we can enforce $\|s_t^{\mathcal{G}} - s_t^*\| \leq \epsilon_{rej}$.
4. Given that the generated labels come from a root solver:

$$
\begin{aligned}
s_{t+1}^* - \hat{f}(s_t^{\mathcal{G}}, a_t^* + \Delta) - s_t^{\mathcal{G}} &= \epsilon_{opt} \text{ where } \epsilon_{opt} \to 0 \\
s_t^* + f(s_t^*, a_t^*) - \hat{f}(s_t^{\mathcal{G}}, a_t^* + \Delta) - s_t^{\mathcal{G}} &= \epsilon_{opt} \\
f(s_t^*, a_t^*) - \hat{f}(s_t^{\mathcal{G}}, a_t^* + \Delta) &= s_t^{\mathcal{G}} - s_t^* + \epsilon_{opt}
\end{aligned}
$$

5. The corrective label is within the neighborhood of local Lipschitz continuity of f:
   $s^{\mathcal{G}} \in U_S$ and $a_t^* + \Delta \in U_A$

Proof

$$
\begin{aligned}
&\|f(s_t^{\mathcal{G}}, a_t^* + \Delta) - \hat{f}(s_t^{\mathcal{G}}, a_t^* + \Delta)\| \\
&= \|f(s_t^{\mathcal{G}}, a_t^* + \Delta) - f(s_t^{\mathcal{G}}, a_t^*) + f(s_t^{\mathcal{G}}, a_t^*) - f(s_t^*, a_t^*) + f(s_t^*, a_t^*) - \hat{f}(s_t^{\mathcal{G}}, a_t^* + \Delta)\| \\
&\leq \|f(s_t^{\mathcal{G}}, a_t^* + \Delta) - f(s_t^{\mathcal{G}}, a_t^*)\| + \|f(s_t^{\mathcal{G}}, a_t^*) - f(s_t^*, a_t^*)\| + \|f(s_t^*, a_t^*) - \hat{f}(s_t^{\mathcal{G}}, a_t^* + \Delta)\| \\
&\leq K_A \|\Delta\| + K_S \|s_t^{\mathcal{G}} - s_t^*\| + \|s_t^{\mathcal{G}} - s_t^*\| + \|\epsilon_{opt}\| \\
&\leq K_A \|\Delta\| + (1 + K_S) \cdot \|s_t^{\mathcal{G}} - s_t^*\| + \|\epsilon_{opt}\|
\end{aligned}
$$

When the root solver yields a solution with $\|\epsilon_{opt}\| \to 0$ and we apply rejection sampling to enforce $\|s_t^{\mathcal{G}} - s_t^*\| \leq \epsilon_{rej}$, we have the error bounded by $K_A \|\Delta\| + (1 + K_S) \cdot \epsilon_{rej}$

5.8.3 *Details for CCIL*

Our proposed framework for generating corrective labels, **CCIL**, takes three steps:

1. **Learn a dynamics model**: fit a dynamics model $\hat{f}$ that is locally Lipschitz continuous.
2. **Generate labels**: solve a root-finding equation in Sec. 5.3.3 to generate labels.
3. **Augment the dataset and train a policy**: We use behavior cloning for simplicity to train a policy.

5.8.4 *Learning a dynamics function while enforcing local Lipschitz continuity*

There are many function approximators to learn a model. For example, using Gaussian process can produce a smooth dynamics model but might have limited scalability when

dealing with large amount of data. In this paper we demonstrate examples of using a neural network to learn the dynamics model. There are multiple ways to enforce Lipschitz continuity on the learned dynamics function, with varying levels of strength that trade off theoretical guarantees and learning ability. In Sec. 5.3.2 we discuss a simple penalty to enforce local Lipschitz continuity, here we provide a few alternative methods, including the one used by (Shi et al., 2019) and the one inspired by (Gulrajani et al., 2017).

Global Lipschitz Continuity via Spectral Normalization. Using spectral norm with coefficient K provides the strongest guarantee that the dynamic model is global K-Lipschitz. Concretely, spectral normalization (Miyato et al., 2018) normalizes the weights of the neural network following each gradient update. (Shi et al., 2019) used this method to train a dynamics function for drone navigation. Parameterizing $\hat{f}$ as a n-layer neural network with weight matrices $W_1, \ldots, W_n$, the learning objective becomes:

$$\arg\min_{\hat{f}} E_{s_j^*, a_j^*, s_{j+1}^* \sim \mathcal{D}^*} [\text{MSE}] \quad \text{while} \quad W_i \leftarrow \frac{W_i}{\max(\|W_i\|_2, K^{-n})} \cdot K^{-n} \tag{5.14}$$

Local Lipschitz Continuity via Sampling-based Penalty. Following (Gulrajani et al., 2017), we can relax the global Lipschitz continuity constraint by penalizing any violation of the local Lipschitz constraint. This objective is stated in Eq. 5.4, which we reiterate here.

$$\begin{aligned} &\arg\min_{\hat{f}} E_{s_j^*, a_j^*, s_{j+1}^* \sim \mathcal{D}^*} \left[\text{MSE} + \lambda \cdot \mathbb{1}\left(\hat{f}'(s_j^*, a_j^*) > K \right) \right] \\ &\hat{f}'(s_j^*, a_j^*) \approx \mathbb{E}_{\Delta_s \sim \mathcal{N}} \left[\frac{\|\hat{f}(s_j^* + \Delta_s, a_j^*) - \hat{f}(s_j^*, a_j^*)\|}{\|\Delta_s\|} \right] \end{aligned} \tag{5.15}$$

To estimate the local Lipschitz continuity, we can use a sampling-based method, $\mathbb{E}_{\Delta_s \sim \mathcal{N}} \max\left(\hat{f}'(s_j^* + \Delta_s, a_j^*) - K, 0 \right)$, which is indicative of whether the local continuity constraint is violated for a given state-action pair. The sampling-based penalty perturbs the data points by some sampled noise and enforces the Lipschitz constraint between the perturbed data and the original using a penalty term in the loss function.

Doing so ensures that the approximate model is mostly K Lipschitz-bounded while being predictive of the transitions in the expert data. This approximate dynamics model can then be used to generate corrective labels.

Local Lipschitz Continuity via Slack-variable. We can allow for a small amount of discontinuity in the learned dynamics model in the same way that slack variables are modeled in optimization problems, e.g., SVM (Hearst et al., 1998) or max-margin planning with slack variables (Ratliff, Bagnell, and Zinkevich, 2006). With these insights in mind, we can reformulate the dynamics model learning objective as learning models that maximize likelihood (MSE) while minimizing the Lipschitz constant $L(s_j^*, a_j^*)$.

$$\begin{aligned}\arg\min_{\hat{f}} \mathbb{E}_{s_j^*, a_j^*, s_{j+1}^* \sim \mathcal{D}^*} \ & \text{MSE} + \lambda_j \cdot L(s_j^*, a_j^*) + \|\lambda_j - \bar{\lambda}\|_0 \\ \text{where } & \|\lambda_j - \bar{\lambda}\|_0 \approx 1 - \exp\left(-\beta|\lambda_j - \bar{\lambda}|\right) \\ \text{and } & L(s_j^*, a_j^*) = \mathbb{E}_{\Delta_s \sim \mathcal{N}} \max\left(\hat{f}'(s_j^* + \Delta_s, a_j^*) - K, 0\right).\end{aligned} \tag{5.16}$$

To account for small amounts of discontinuity, we introduce a state-dependent variable λ_j to allow violation of the Lipschitz continuity constraint. We minimize the number of non-zero entries in the slack variables (as noted by the L0 norm, $\|\lambda_j - \bar{\lambda}\|_0$) to ensure the model is otherwise as smooth as possible besides small amounts of local discontinuity. Building on Oliveira, Batista, and Seara (2021), we can use a practical, differentiable approximation for the L0 norm method, since the L0 norm itself is non-differentiable. Equipped with these locally continuous learned dynamics models, we can then generate corrective labels for robustifying imitation learning.

Local Lipschitz Continuity via Weighted Loss. Since we are optimizing a continuous function approximators, if the data contain subsets of discontinuity, such regions are likely to induce high training loss. We can create a slack variable for each data point, λ_j, to down-weight the on those regions. To do so, we reformulate Eq. 5.14 to add a slack penalty. Taking σ as the Sigmoid function:

$$\arg\min_{\hat{f}} \mathbb{E}_{s_j^*, a_j^*, s_{j+1}^* \sim \mathcal{D}^*} \left[\sigma(\lambda_j) \cdot \text{MSE} + \sum \sigma(\lambda_j)\right] \quad \text{while} \quad W_i \leftarrow \frac{W_i}{\max(\|W_i\|_2, K^{-n})} \cdot K^{-n} \tag{5.17}$$

Intuitively, we expect that spectral normalization tends to work better for simpler environments where the ground truth dynamics are global Lipschitz with some reasonable L, whereas the soft constraint should be better suited for data regimes with more complicated dynamics and discontinuity. It remains a research question how best to train dynamics function for each domain and representation with different physical properties.

5.8.5 *Generating corrective labels*

In Sec. 5.3.3 we discuss two techniques to generate corrective labels. Depending on the structure of the application domain, one can choose to generate labels either by backtrack or by disturbed actions. Both techniques require solving root-finding equations (Eq. 5.6 and Eq. 5.8). To solve them, we can transform the objective to an optimization problem and apply gradient descent.

Eq. 5.6 specifies the root finding problem for backtrack labels.

$$s_t^* - \hat{f}(s_{t-1}^{\mathcal{G}}, a_t^*) - s_{t-1}^{\mathcal{G}} = 0.$$

Given s_t^*, a_t^* and the learned dynamics function $\hat{f}$, we need to solve for $s_t^{\mathcal{G}}$ that satisfies the equation. We can instead optimize for

$$\arg\min_{s_{t-1}^{\mathcal{G}}} \|s_t^* - \hat{f}(s_{t-1}^{\mathcal{G}}, a_t^*) - s_{t-1}^{\mathcal{G}}\| \tag{5.18}$$

Similarly, we can transform Eq. 5.8 to become

$$\arg\min_{s_t^{\mathcal{G}}} \|s_t^{\mathcal{G}} + \hat{f}(s_t^{\mathcal{G}}, a_t^* + \Delta) - s_{t+1}^*\| \tag{5.19}$$

With access to the trained model $\hat{f}$ and its gradient $\frac{\partial \hat{f}}{\partial s_t^{\mathcal{G}}}$, one can use any optimizer. For simplicity, we use the Backward Euler solver that apply an iterative update $s_t^{\mathcal{G}} \leftarrow s_t^{\mathcal{G}} - s \cdot \frac{\partial \hat{f}}{\partial s_t^{\mathcal{G}}}$ where s is a step size. We repeat the update until the objective is within a threshold $\|s_t^{\mathcal{G}} + \hat{f}(s_t^{\mathcal{G}}, a_t^* + \Delta) - s_{t+1}^*\| \leq \epsilon_{opt}$.

5.8.6 *Using the generated labels*

There are multiple ways to use the generated corrective labels. We can augment the dataset with the generated labels and treat them as if they are expert demonstrations. For example, for all experiments conducted in this paper, we train behavior cloning agents using the augmented dataset. Optionally, one can favor the original expert demonstrations by assigning higher weights to their training loss. We omit this step for simplicity in this paper.

Alternatively, one can query a trained imitation learning policy with the generated labels and measure the difference between our generated action and the policy output. This difference can be used as an alternative metric to evaluate the robustness of imitation learning agents when encountering a subset of out-of-distribution states. However, for a query state, our proposal does not necessarily recover *all* possible corrective actions. We defer exploring alternative ways of leveraging the generated labels to future work.

5.8.7 *Simulation Warm-Up Experimental Details*

We provide details to reproduce our simulation warm-up experiments, including environment specification, expert data, parameter tuning for our proposal and details about the baselines. We will also open-source the code and the configuration we use for each experiment, once the proposal is published.

5.8.8 *Environment and Task Design*

We warmed up by conducting simulation experiments on 4 different domains and 8 robots, including the pendulum, a drone, a car, four robots for locomotion and one robot arm for manipulation. We consider 14 tasks: the pendulum, a modified pendulum swing task with discontinuity, three drone navigation tasks (fly-through, circle, hover), one LiDar racing task on F1tenth, four MuJoCo tasks (Hopper, HalfCheetah, Ant, Walker2D) and 4 MetaWorld tasks (coffee-pull, button-press-topdown, coffee-push, drawer-close). The F1tenth, MuJoCo and Metaworld environments are from open source implementations, and the drone environment is from a slightly modified open source

implementation, discussed below. We will also describe how we set up the Pendulum environment and how we modify it to test our method with discontinuity.

PENDULUM FORMULATION. The pendulum environment asks a policy to swing a pendulum up to the vertical position by applying torque. The properties of the system are controlled by the constants g, the gravitational acceleration, and l, the length of the pendulum. In all experiments, we take $g = 9.81$ and $l = 1$.

A pendulum's state is characterized by θ, the current angle, and $\dot{\theta}$, the current angular velocity. To avoid any issues regarding angle representation, we do not directly store θ in the state representation; instead, we parameterize the state as $s = \begin{bmatrix}\sin\theta & \cos\theta & \dot{\theta}\end{bmatrix}^T$. A policy can control the system by applying torque to the pendulum, which we represent as a scalar a, which is clamped to the range $[-3, 3]$.

The continuous-time dynamics function is given by:

$$\frac{ds}{dt} = f(s, a) = \begin{bmatrix} \dot{\theta}\cos\theta \\ -\dot{\theta}\sin\theta \\ -\frac{g}{l}\sin\theta + a \end{bmatrix}.$$

This continuous dynamics model is then discretized to a timestep of 0.02 seconds using RK4. Additionally, although not required by the algorithms we study, we create the following reward function. where θ is the normalized pendulum angle in the range $[0, 2\pi)$ to metricize policy performance:

$$r(s, a) = -\frac{1}{2}\left\|\begin{bmatrix}\theta - \pi \\ \dot{\theta}\end{bmatrix}\right\|_2^2 - \frac{1}{2}a^2.$$

EXPERT FORMULATION FOR PENDULUM. We formulate the expert policy using a combination of LQR and energy shaping control, where LQR is applied when the pendulum is near the top and energy-shaping is applied everywhere else. Note that the LQR gains were calculated by linearizing the dynamics around $\theta = \pi$, along with the cost function $c(s, a) = -r(s, a)$. So, the expert policy has the form:

$$\pi_e(s) = \begin{cases} -20.11(\theta - \pi) - 7.08\dot{\theta} & \text{if } |\theta - \pi| < 0.1 \\ -\dot{\theta}\left(\frac{1}{2}\dot{\theta}^2 - 9.81\cos\theta - 9.81\right) & \text{otherwise.} \end{cases}$$

DISCONTINUOUS PENDULUM. We create a wall in the Pendulum environment to create local discontinuity. When the ball hits the wall, we *negate* its velocity, creating a discontinuity in the dynamics.

DRONE. The drone environment used in our experiments is from a slightly modified fork of the original gym-pybullet-drones project. There are two notable modifications. Firstly, we added the circle task, where the agent aims to move in a circular trajectory. Second, we removed the floor, which better simulates a completely airborne drone and reduces training complexities caused by crashing into the floor.

Environment	Trajectories
Pendulum	50
Discontinuous Pendulum	500
F1tenth Racing	1
Drone hover	5
Drone circle	30
Drone fly-through	50
MuJoCo - Ant	10
MuJoCo - Walker2D	20
MuJoCo - Hopper	25
MuJoCo - HalfCheetah	50
MetaWorld, all tasks	50

Table 5.2: Number of expert demonstration trajectories used in our experiments. We limit the amount of expert data to avoid making the task trivially solvable by naive behavior cloning.

5.8.9 *Demonstration Data*

To feed expert data to train imitation learning agents, we design expert policies for the pendulum. For all other environments, we use the expert data from the D4RL dataset (Fu et al., 2020). For drone environments, we first generate a bunch of via points alongside the target trajectories and then use a low-level PID controller to hit the via points one by one.

We note that it is possible to solve most tasks with naive behavior cloning if we feed them with a sufficient number of demonstrations. We thus limit the number of demonstrations we use for all tasks, as shown in Table. 5.2.

5.8.10 *Baselines*

We evaluate the following algorithms to gain a thorough understanding of how our proposal compares to relevant baselines.

- EXPERT: For reference, we plot the theoretical upper bound of our performance, which is the score achieved by an expert during data collection.
- BC: a naive behavior cloning agent that minimizes the KL divergence on a given dataset.
- NOISEBC: a modification to naive behavior cloning that injects a small disturbance noise to the input state and reuses the action label, as described in (Ke et al., 2021b).

- MOREL: Morel (Kidambi et al., 2020) trains an ensemble of dynamics functions and uses the learned model for model-based reinforcement learning. It uses the variance between the model output as a proxy estimation of uncertainty to stay within high confidence region. We use the author implementation.
- MILO: MILO (Chang et al., 2021) learns a dynamic function from given offline dataset in an attempt to mitigate covariate shift. We use the author's implementation in our experiment. MILO traditionally requires a large batch of offline dataset to train a dynamics function, we use only the expert demo dataset to train a dynamics function.
- CCIL: our proposal to generate high-quality corrective labels.

Part III

LEARNING BEYOND EXPERT DATA

6

REINFORCEMENT LEARNING FOR DYNAMIC FINE TASKS

How can we automate the task of picking cherries from a tree branch that is blowing in the wind, causing the branch to shake and the cherries to tremble? This scenario is an example of fine grasping *without rigid-surface support*, and its challenges are two-fold. First, for fine manipulation of small objects, perception errors and sensor noise dominate, making it difficult to grasp the objects precisely (Cutkosky, 2012; Ke et al., 2021b). Second, the problem is inherently dynamic since any contact with the object might set the entire scene into motion, which is complicated to model (Billard and Kragic, 2019; Mason and Lynch, 1993). Similar challenges arise in our everyday interactions, from mundane tasks such as removing broken shells from gelatinous egg whites to surgical tasks that detach clots from deformable organs. Given the ubiquitous nature of these tasks, developing robotic solutions to automate them holds immense practical and economic value.

For a predetermined, specific task, it is possible to invest in dedicated hardware (Marohn and Hanly, 2004; Yuan, Dong, and Adelson, 2017), specialized tools (Bhattacharjee et al., 2019; Li et al., 2019; Zeng et al., 2022), and elaborately designed systems (Hwang et al., 2022; Lynch and Mason, 1999) to solve these challenges. However, this research investigates a more universal solution: assuming that fine manipulation is required, inaccuracy is unavoidable and real-time reaction is necessary, can we enable dynamic fine grasping without stable support? An ideal agent should be:

- **Precise** enough to increase the likelihood of task success.
- **Robust** to perception errors and sensor noises that are likely to arise in the fine manipulation domain.
- **Reactive** to hard-to-model dynamic scenarios, external perturbations, and changes caused by its own movements.
- **Generalizable** to objects with different sizes, shapes and textures.

Prior work constructed analytic models (Hogan and Rodriguez, 2020; Kumar, Todorov, and Levine, 2016; Mordatch, Todorov, and Popovi'c, 2012) or motion primitives (Schaal, 2006) for manipulation tasks with rich contacts. Instead, we choose reinforcement learning (RL) to circumvent some of the complexity of building accurate models for contacts, dynamics, different objects, or external disturbances. Despite their impressive potential for generalizability (Kalashnikov et al., 2018), applications of RL remain limited for real robots due to, for example, sample efficiency (Zhu et al., 2019) and the costs of resetting (2020). Though careful system design has enabled successfully deployed RL systems to learn on locomotion and dexterous manipulation tasks (Peng et al., 2020; Zhu et al., 2019), the characteristics of our problem, i.e., fine manipulation with precise

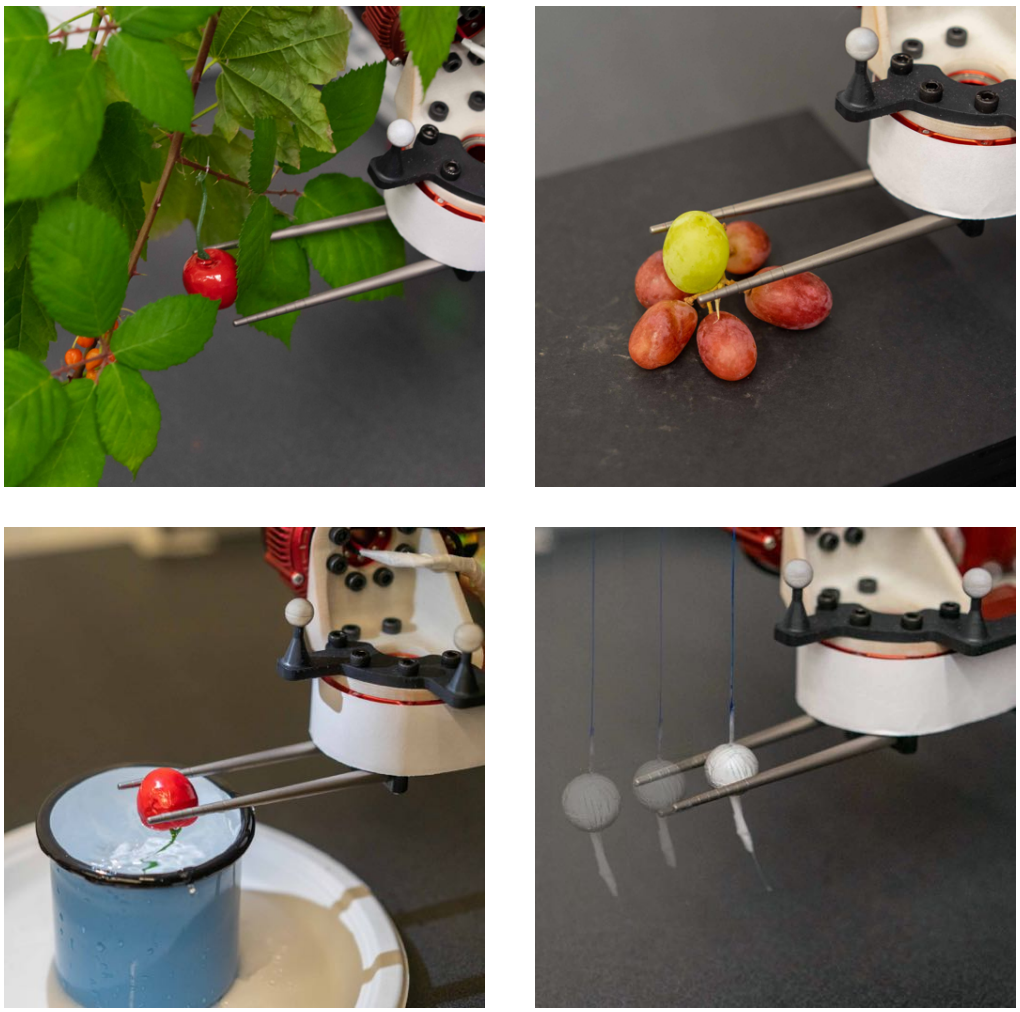


Figure 6.1: The CherryBot system generalizes to various scenarios of dynamic fine manipulation: blowing wind, sliding grape cluster, moving water, and swinging string.

Figure 6.2: The dynamic grasping task is challenging: any contact with an object might set it into motion, which is difficult to model. This challenge is exacerbated when fine manipulation is required, especially in the presence of sensor noise and perception errors.

contact and hard-to-model dynamics, raise additional challenges in robustness and reactivity.

It is tempting to bypass modeling entirely and directly deploy a model-free RL algorithm for training in the real world. While this approach may eventually learn, it often necessitates tremendous human effort for supervision and resets and is likely to be too inefficient. To make the training more practical, we propose CherryBot, an

RL system that combines pre-training in an imprecise simulation with fine-tuning in the real world. With an external state estimation module, CherryBot can be deployed across various scenarios and fulfills the aforementioned requirements:

- **Efficiency:** To enhance sample efficiency for real-world manipulation, we leverage imperfect information readily available to most robots, such as an inaccurate simulator and a heuristic-based baseline policy.
- **Precision and Robustness:** We introduce a challenging task that involves a single *hard-to-grasp* object with *diverse dynamics* for real-world fine-tuning. While this intensifies the difficulty of the task and training, it minimizes the need for human intervention and promotes the learning of robust policies that are resilient to disturbances.
- **Reactiveness:** We carefully design the action space to strike a balance between tractable learning (low-frequency control, slow response, short horizon) and responsiveness (high-frequency control, fast response, long-horizon reasoning). This design optimizes the system's ability to react swiftly while ensuring effective learning.
- **Generalizable**: Our system supports the plug-and-play integration of an external state estimation module even if it introduces certain inaccuracies, allowing deployment across various downstream tasks.

Our work contributes a system that, given only 30 minutes of interaction in the real world, achieves superhuman reactiveness on a dynamic, high-precision task: using chopsticks to grasp a slippery ball swinging in the air. We demonstrate the effectiveness of our system in a variety of evaluation conditions: operating under dynamic disturbances by human or environmental factors, varying perception noise, and changing object shapes and sizes, for which it outperforms a heuristic-based controller that requires hours of tuning. We conduct extensive ablations in a simulator and the real world to provide empirical evidence about how our design choices affect sample efficiency for deploying RL systems in the real world and to verify the robustness of our proposal.

6.1 RELATED WORKS

Dynamic fine grasping. From precondition grasping like DexNet (Mahler et al., 2016) and TransporterNet (Zeng et al., 2021) to a closed-loop, vision-based controller like Qt-Opt (Kalashnikov et al., 2018), most prior works grasped palm-sized objects in a quasi-static table-top setting (Calli et al., 2017). In our work, objects are much smaller. Even if the agent intends to grasp across the center of the cherry, for example, the execution alone demands a sub-millimeter precision. Several previous works addressed fine grasping (Hwang et al., 2022; Ke et al., 2021b) but are limited to static scenarios with rigid surface support. In contrast, we examine a dynamics scene in which failed grasps can potentially move the object. Researchers in the field of dynamic manipulation have developed highly capable systems (Hashimoto, Namiki, and Ishikawa, 2001), although

these systems come with the drawback of relying on expensive dedicated vision systems. In our approach, however, we prioritize the use of generic and accessible hardware, which is more susceptible to making mistakes. Consequently, it becomes crucial to develop a reactive agent that can effectively recover from such errors. We modify the common evaluation criteria for grasping to allow an agent that fails to get a firm grasp in one shot to continue the trial until it achieves a successful grasp, similar to how humans address dynamic grasping challenge (Ke et al., 2020b).

Reinforcement learning. Despite its wide successes in simulator and locomotion (Haarnoja et al., 2018a; b; Zhu et al., 2019), RL has limited real world applications in manipulation. The few successes of applying RL to dexterous manipulation came at the cost of sample inefficiency, which has not made it a preferred choice over model-based control or a hand-designed controller. (Kalashnikov et al., 2018) depends on the large scale of a robot arm farm; (W"uthrich et al., 2020) requires scalable and dedicated hardware. Previous works focused on model-free RL while our work leverages practical assumptions, including inaccurate simulation (Peng et al., 2020), imperfect demonstration (Rajeswaran et al., 2017), and normalization techniques (Chen et al., 2021; Smith, Kostrikov, and Levine, 2022) to boost sample efficiency for RL and show it is practical on real robots and can achieve superhuman reactiveness.

Offline reinforcement learning (ORL). The recently emerging field of ORL (Lange, Gabel, and Riedmiller, 2012; Levine et al., 2020) has shown some potential for leveraging offline datasets for RL training (Fujimoto, Meger, and Precup, 2018; Kidambi et al., 2020; Kostrikov, Nair, and Levine, 2021; Kumar et al., 2020; Peng et al., 2019; Zhou et al., 2023b). However, whether doing online fine-tuning on top of pre-trained ORL agents could keep improving the performance remains an open question (Hong, Kumar, and Levine, 2022).

Visual Servoing (VS). VS controls a robot using real-time visual feedback (Chaumette and Hutchinson, 2006; Hutchinson, Hager, and Corke, 1996). It usually requires estimating the pose of the object in the Cartesian space and deriving a control law, such as PID control, directly in the 3D space (Sanderson and Weiss, 1983). Due to the requirement of precision, it is natural to apply VS to fine manipulation tasks. We follow this protocol in the design of our heuristic controller. However, it would require a tremendous amount of expert knowledge and gain-tuning to make the controller generalize across different scenarios or be robust to disturbance and noise. Our baseline controller took hours of gain-tuning but still fails due to disturbances, exemplifying how VS can be sensitive to unexpected dynamics.

6.2 METHOD

To address fine manipulation problems in dynamic scenes, we propose a comprehensive framework for efficient training of real-world RL and build a high-performance robotic system. We train a policy to enable a robot equipped with chopsticks to conduct dynamic fine-grasping tasks with non-rigid support. Following established practices in visual servoing (Garrett et al., 2021; Zhang et al., 2022), we incorporate a separate perception

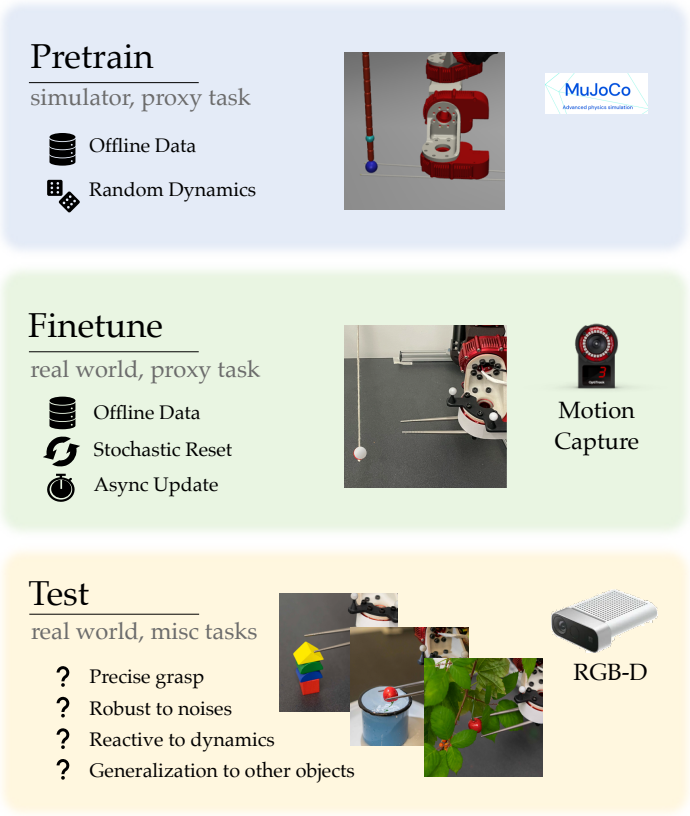


Figure 6.3: An overview of the learning framework. The system first pre-trains in simulation and then fine-tunes in the real world. We carefully design the training paradigm to ensure sample efficiency and learning robustness.

module, using estimated robot and environment states as input to the policy. Notably, our policy includes a *hierarchical controller* (`Hier`) to balance reactivity to dynamics and tractability of learning. At test time, we consider fine grasping of small objects (size $1 \sim 8$ cm) with varying shapes and textures in a dynamic scene: objects might have unstable support and can be moving (e.g., initial velocity, or wind disturbance).

We present a system overview in Fig 6.3. Initially, we employ pre-training in an approximate simulator with domain randomization (`SimR`), leveraging sub-optimal demonstrations (`Demo`). Subsequently, we fine-tune the system in the real world on a task with a single object (`Proxy`) that presents dynamic challenges but simplifies perception and allows for easy and stochastic reset (`StoRe`). To further enhance training efficiency, we incorporate asynchronous updating (`Async`) and appropriately regularized off-policy RL (`LN`). During testing, the policy integrates an external perception module (`Vis`) to achieve generalization across different objects and scenes.

Our contribution lies in instantiating distinct system components to address the combined challenges of real-world RL and dynamic fine manipulation. We outline these challenges in Table 6.1 and note how each system component addresses them. To ascertain the impact of each design choice, we conduct ablation studies to measure sample efficiency in both simulation and the real world, where all studies are conducted

using 5 consecutive random seeds (Refer to Appendix. 6.6.1 for details). While individual components may have been the subject of previous studies, the combination presented in our work is novel. Our system can grasp a diverse set of small objects in varying dynamic scenarios in the real world, demonstrating the robustness and generalizability to novel objects and disturbances, which could generalize to other tasks with dynamics or requiring fine manipulation.

In this section, we delve into the specifics of our design decisions, grouped into three categories: controller interface design, simulator pre-training and real-world fine-tuning.

	RL challenges		**Task challenges**			
	Efficient	Reset	Robust	Reactive	Precise	General
Hier	✓					
Demo	✓					
SimR	✓		✓			
Proxy	✓	✓	✓	✓	✓	
StoRe		✓	✓			
Async	✓					
LN	✓					
Vis	✓					✓

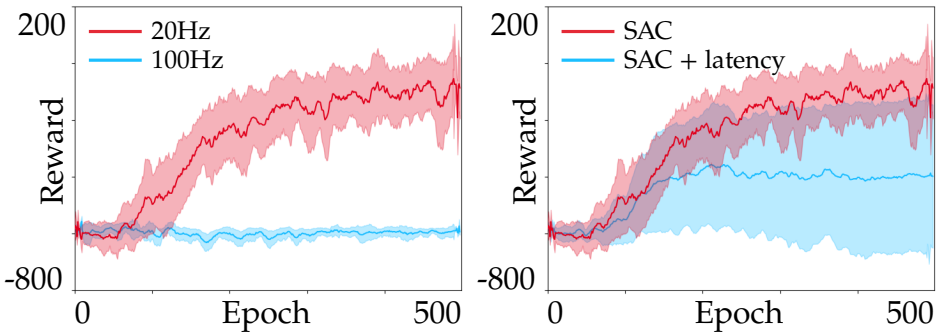
Table 6.1: Some challenges inherent to training RL algorithms for dynamic fine manipulation and the design decisions we make to address them.

6.2.1 *Hierarchical controller of mixed control frequency to balance tractability of learning and reactivity*

Previous research on dynamic tasks has favored the development of 1000Hz controllers (Chi et al., 2022; W"uthrich et al., 2020) and vision systems (Okumura, Oku, and Ishikawa, 2011). However, as demonstrated in Fig 6.4a, high-frequency control can negatively impact task horizon and sample efficiency for robotic learning. Real-life RL manipulations commonly employ controllers operating at 5 ~ 15 Hz for quasi-static tasks (Zhou et al., 2023b; Zhu et al., 2019).

To address this challenge, we adopt a hierarchical controller framework, enabling the RL policy to function at a higher level with a moderate control frequency of 20 Hz, while interpolating commands to a high-frequency (1000Hz) controller. Our system demonstrates that such a controller is capable of solving dynamic tasks and, combined with coherent exploration, makes the learning of a reactive policy tractable.

Furthermore, in the real world, sensor readings suffer from latency and latency can adversely affect reactivity. While (Peng et al., 2018) demonstrated the advantages



(a) Effect of control frequency (b) Effect of latency ($< 0.005s$)

Figure 6.4: Analysis of using hierarchical mixed-frequency controllers in simulation. **Left:** Impact of different control frequencies on learnability. A hierarchical hybrid-frequency strategy helps balance learnability and policy reactivity. **Right:** Impact of (simulated) latency on performance.

of latency randomization in simulation pre-training for a quasi-static task, our own experiments (Fig 6.4b) revealed that even a small simulated latency ($\leq$ 0.005 s) can harm our task's reactivity to dynamics. Consequently, we made the deliberate decision to exclude simulating latency or latency randomization in simulator training.

Insight: Learning medium-frequency hybrid controllers can effectively balance policy reactivity with the tractability of learning.

6.2.2 *Leveraging practical information: Approximate simulator* `SimR` *and sub-optimal offline data* `Demo`

An RL agent initialized from scratch in the real world makes random movements, frequently encounters safety constraints, and spends a significant portion of training time waiting to be reset. To counter these challenges, we turn to the paradigm of offline pre-training followed by real-world fine-tuning. Where should the data for pre-training actually come from? Since human data collection is expensive for our dynamic fine manipulation task, we instead rely on relatively cheap and abundant, but imperfect, sources of supervision, i.e., approximate simulators and data from heuristic controllers.

Choosing an appropriate off-policy algorithm. We first consider an appropriate RL algorithm that can efficiently learn from offline data, simulation samples and online fine-tuning. Although various offline RL methods (Kostrikov, Nair, and Levine, 2021; Kumar et al., 2020) have been proposed to conduct first pre-training and then fine-tuning, we found that standard *online* RL algorithms (such as soft actor critic (SAC) (Haarnoja et al., 2018b)) are far more effective for fine-tuning than targeted offline RL methods, as Fig 6.5 shows. In our work, we used variants of soft actor critic (b) for both pre-training and fine-tuning. Our modifications are described below.

Leveraging imperfect data from the simulation. Simulation can provide an appealing source of information since sampling is cheap. However, our task depends heavily on precise contacts and dynamics, but our hardware has varying degrees of

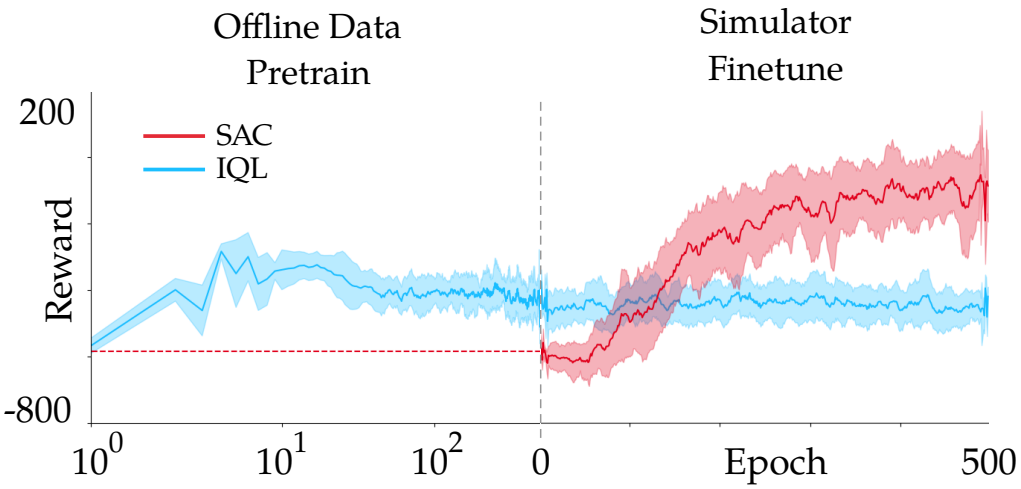
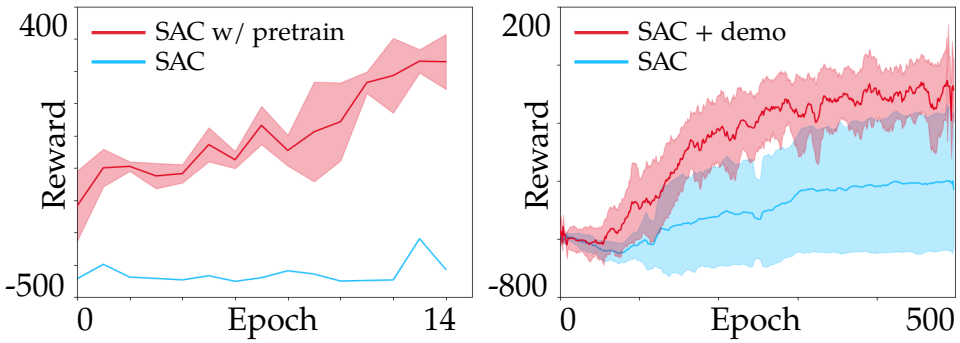


Figure 6.5: Analysis of the choice of an off-policy RL algorithm that can learn from offline data while continuing to improve efficiently during fine-tuning.

kinematic and dynamic errors, making it challenging to construct a precise simulator and making direct simulation-to-reality transfer difficult.

We constructed an approximate simulator, as described in Sec. 6.3. To best leverage such a simulation that is definitely mismatched with reality, we randomize the dynamics of the physical simulation (Peng et al., 2018), exposing the agent to a wide distribution of possible physics parameters. Unaware of the changing dynamics of the environment, the agent needs to develop a robust strategy that is conservative to variations in the world dynamics, making it more amenable to real-world fine-tuning. Specifically, we train a Q-function and policy networks with samples from our simulator and then use the networks to initialize the fine-tuning phase. Without pre-training, an RL agent could not improve its task performance even after hours of real-world training, as shown in Fig 6.6a.



(a) Pre-train from simulation

(b) Pre-train from heuristic controller

Figure 6.6: Analysis of the impact of pre-training for fine manipulation. **(Left)** Having pre-trained in simulation enabled real-world fine-tuning to make progress. **(Right)** Inclusion of offline data collected from a heuristic controller enabled simulator pre-training to succeed on the task in the simulator using many fewer samples. Both techniques significantly aid with learning efficiency.

Leveraging sub-optimal offline data. Beyond simulation data, it is useful to provide the agent access to real-world data with sufficient state-action coverage. While it may be expensive to solicit data from a human supervisor or to generate optimal demonstration, it is relatively easy to design a heuristic policy that is sub-optimal but can provide useful state action coverage. In this work, we design a simple heuristic controller that contains a state machine with closed-loop control (see Appendix 6.6.3). Though this controller has limited reactivity and is not robust to noise, we found that seeding the replay buffer of our RL agent with this sub-optimal data helps with both asymptotic performance and sample efficiency (Fig. 6.6b).

Insight: Pre-training using standard off-policy RL methods with imperfect prior data from heuristic controllers and simulation can significantly help with sample efficiency for real-world fine-tuning.

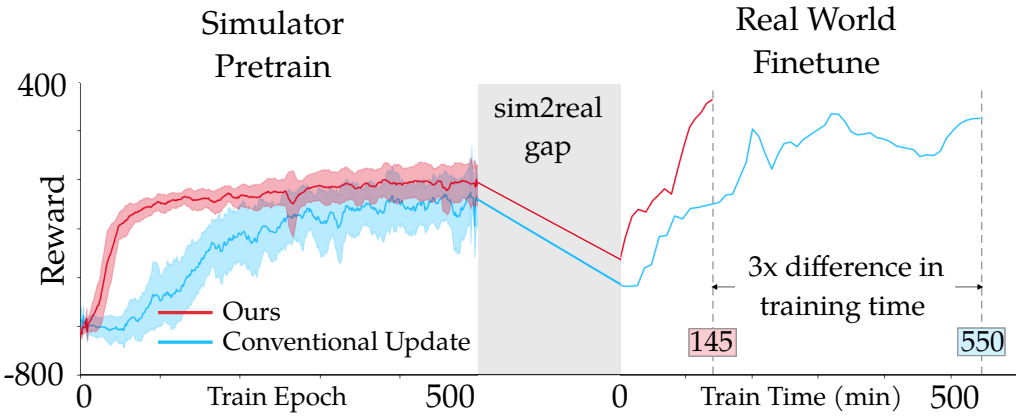
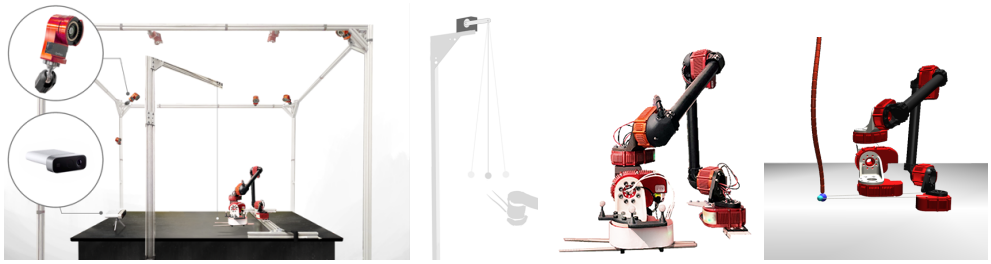


Figure 6.7: We measure sample efficiency in simulation by training epochs but in the real world by training time. Combining the asynchronous update, high Update-To-Data (UTD) ratio and LayerNorm regularizer greatly improves the speed of training during the real-world finetuning stage.

6.2.3 *Fine-tuning in the real world: Proxy task and efficient reinforcement learning*

Although pre-training can significantly aid with data efficiency, it is unlikely to achieve optimal performance without further training in the real world. To enable practical fine-tuning in the real world, we focus on two questions: (1) what real-world training setup allows for both ease of training and robust learning, and (2) how do we make this process more efficient?

Pragmatic fine-tuning via a single proxy task. Conventional wisdom might suggest training the RL agent directly on a large variety of test scenarios. However, a real-world training setup would require a prohibitively large number of objects and scenarios to be instrumented, which can be laborious and expensive. Instead, we posit (1) to isolate the perception challenge and control challenge, and (2) first address the control challenge via a `Proxy` task that is appropriately dynamic and difficult, such that



(a) Hardware overview. (b) Proxy task. (c) Assembled robot. (d) MuJoCo simulation.

Figure 6.8: The CherryBot hardware system.

it can yield a policy that generalizes to a variety of dynamic scenarios without actually being explicitly exposed to them during training. We identify the criteria needed to set up a proxy task as follows: (1) *representative* of the motion and strategies involved in a broader family of tasks, (2) *reset-friendly*, or having largely autonomous reset, (3) learnable but not trivial, and (4) yields a policy that is *robust*. For our dynamic fine grasping problem, we define the following proxy task: *firmly grip a ball swinging in the air*, as Fig 6.8b shows. We attach a ball to a thin fishing line to hang it in the air to allow a swinging motion. The task naturally provides a reset since the string resets the ball around its static position due to gravity, obviating the need for a human supervisor to reset the scene.

Exposing the agent to varying initial conditions enables developing a more robust policy: stochastic reset. While waiting for a long period to allow the object to come to rest would yield a *static reset*, we show, surprisingly, that resuming sampling with a randomly moving object enables learning of more dynamic and adaptive behaviors. By embracing this type of *stochastic* reset (`StoR`), we construct a harder-to-learn task, i.e., the agent must react to different initial positions and velocities, so it experiences a larger diversity of conditions during training. As a result, the learned policy becomes more robust to varying dynamic conditions, improving skill transfer to novel disturbances and objects (see Appendix. 6.6.1 for quantitative ablation).

Insight: Designing appropriate proxy tasks to train the agent can simplify the training infrastructure. Additionally, training with stochastic reset can improve the robustness and generalizability of learned policies.

Efficient fine-tuning: Improving gradient throughput with asynchronous updates and regularization

To improve training efficiency in the real world, we care about not only the sample efficiency but also the wall clock time efficiency. In the simulation, operations (e.g., sampling step, resetting hardware) are almost instant. In the real world, however, idle time is unavoidable and adversely affects the speed at which we collect samples: for every second of samples collected, our system spends 4 seconds waiting for a reset. We replace the common training paradigm in simulation, which takes one gradient update after collecting one environment step. We instead perform asynchronous RL updates

up to the limit of our computer alongside robot operation. This greatly increases our gradient throughput, effectively committing 10 to 20 updates per data point collected (Update-To-Data, UTD).

Most off-policy RL methods lower their UTD ratios because additional gradient updates lead to unbalanced training of policy/value functions (i.e., exploding actor / value losses). In this work we find that this problem can be significantly mitigated by using appropriate regularization. By using a standard technique such as layer normalization (Ba, Kiros, and Hinton, 2016), we are able to avoid overfitting and take significantly more gradient steps per data point. As shown in Fig 6.7, the combination of asynchronous updates and LayerNorm regularization achieve moderately faster training in simulation but, during fine-tuning on the real robot, significantly boosts the efficiency of learning in terms of wall-clock time.

Insights. Improving gradient throughput by leveraging asynchrony and more gradient steps per data point with regularization can make fine-tuning efficient enough for practical real-world use.

6.3 COMPLETE SYSTEM DESIGN FOR CHERRYBOT

We combine the preceding design decisions and insights into a single system, CherryBot (Fig 6.3), that can handle challenging dynamic fine manipulation tasks in the real world. Below, we describe the hardware infrastructure and concrete implementation details.

Reset mechanism. At the end of every sampled trajectory in the real world (average length is about 80 timesteps or 4 seconds), we conduct a fixed trajectory for reset. We (1) slowly raise the robot arm to a set position, (2) touch the string in the proxy task, expecting to constrain its motion and reduce system entropy, and (3) return the robot arm to a fixed start pose and restart the task. The reset process takes about 16 seconds, during which our policy and value networks keep sampling experiences from the replay buffer and conducting RL updates. Notably, the reset leaves the object in a dynamic condition, introducing the dynamism we need for learning robust policies.

Simulation. As described in Section 6.2.2, we leverage simulation for pre-training value functions and policies. We construct the simulation in MuJoCo (Todorov, Erez, and Tassa, 2012a) and perform our best possible system identification procedure, yet we acknowledge that inaccuracies and mismatch still persist.

Perception. We use different perception modules for training and testing. At training time, we use a motion capture system, Optitrack, to obviate many perception challenges and yield an accurate position for the single object to grasp used in the proxy task (error $\sim$ 0.01mm). At test time, accurate state estimation may not be available. Our system instead accepts any external perception module that can estimate the object's center of mass, following common practice in visual servoing (Garrett et al., 2021; Zhang et al., 2022). Noticeably, our system does not necessitate a highly precise estimation module w.r.t. the fine manipulation tasks.

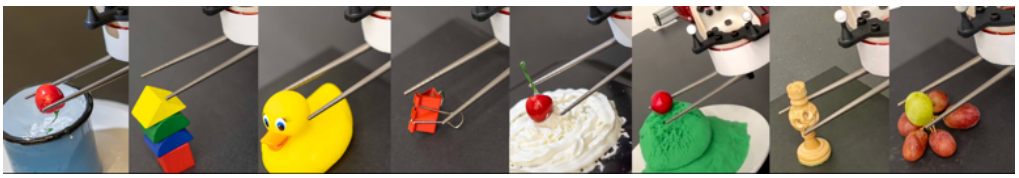
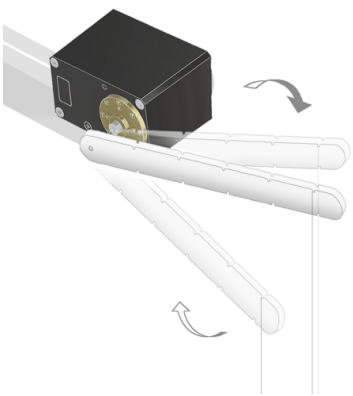


Figure 6.9: We deployed our system to generalize to a wide variety of objects with different shapes and textures.

Utilizing an external perception module allows our system to be deployed in various downstream tasks. For illustration, we demonstrate our system using (1) a simple segmentation method with smoothing and (2) an off-shelf detection system, YOLO (both documented in App. 6.6.1). When the object is static, easy to recognize and the scene is clean, our in-house module detects the object with little error. However, our scenes can have occlusions and can be changing (e.g., cherry shaking in the wind partially occluded by the leaves), and the perception error could reach ~5mm. In our experiment, we will also inject noise to the perception module's output to demonstrate the robustness of our proposal to perception noise.

Additional hardware for dynamic disturbance evaluation. The goal of our fine grasping policies is to firmly grasp the object, despite dynamic disturbances. It is challenging to test this type of robustness without additional machinery to introduce disturbances. To this end, we design a motor disturbance mechanism to systematically inject dynamic disturbance so we could evaluate the robustness of agents. As shown in Fig 6.10a, the motor perturbs the motion of the ball by rotating the motor's arm and dragging the string attached to it. By varying the string's hanging point, the object can be pulled with different sizes of disturbance (from smaller disturbances to bigger ones, denoted as 20 ~ 100).



(a) Motor for more dynamic disturbance.

(b) Human disturbance.

Figure 6.10: Dynamic disturbance evaluations we considered. These disturbances test the robustness of the learned CherryBot policies.

	Marble ball			Cherry			Gaussian noise			Static ball			Dynamic ball		
	Rew.	Succ.	TTS	Rew.	Succ.	TTS	Rew.	Succ.	TTS	Rew.	Succ.	TTS	Rew.	Succ.	TTS
Our RL	**242.3**	**100%**	**1.77**	**321.7**	**100%**	**0.92**	**130.4**	**90%**	**2.06**	**565.3**	**100%**	**0.28**	**512.3**	**100%**	**0.48**
20HZ VS	64.3	60%	1.12	113.1	70%	1.17	-30.9	50%	2.29	196.2	100%	1.49	183.5	90%	1.38
100HZ VS	48.4	50%	0.93	110.7	60%	0.32	109.3	60%	0.64	535.0	100%	0.49	255.0	90%	0.49
Human	-	-	-	-	-	-	-	-	-	-38.8	100%	2.20	-100.2	100%	2.86
Replay	-	-	-	-	-	-	-	-	-	-56.9	80%	1.45	-218.4	50%	2.29
BC	-	-	-	-	-	-	-	-	-	-354.4	30%	0.58	-487.0	10%	0.48

Table 6.2: Evaluation results, including average reward (Rew.), success rate (Succ.), and time-to-success for only **successful** trials (TTS in seconds). Note the RL agent is able to adapt to its mistakes and would retry after failure rapidly, therefore achieving a high success rate within the allocated 6 seconds.

6.4 EVALUATION

In our experiments, we evaluate our proposal on fine manipulation challenges (grasping slippery balls or various small items) and during dynamic disturbances (created using programmed motor movements or human interference). We intend to answer the following questions: (1) Can a policy learned by CherryBot be robust and reactive to dynamic scenarios? (2) Can our policy generalize to different objects? (3) Is our proposal robust across random seeds? (4) How do our design decisions impact the final results for CherryBot?

6.4.1 *Tasks*

We test the *reactivity*, *robustness*, and *generalization* of our agent quantitatively and qualitatively. View the video recordings on our website for visualization of our evaluations.

Reactivity to dynamics in the scene. In natural scenarios, a smBaall object might be moved by external disturbance, e.g., a leaf flying in the wind. We simulate this kind of disturbance by installing a motor that pulls the string that suspends the object (Fig. 6.10a). Additionally, we introduce a more challenging, more spontaneous source of external interference—humans. As shown in Fig. 6.10b, a person would drag the string while the agent tries to grasp and shake the ball attached to it. We provide a qualitative video on the website reporting the performance of our agent on the latter task.

Robust to perception noise. In the real world, perception errors are inevitable due to, e.g., lightning conditions or occlusions. We simulate such errors by injecting noise into the tracked positions of the object, i.e., adding a small Gaussian noise to the output of our Optitrack system and using this noisy position as the states (Tedrake, 2023).

Generalization. Different objects have varying shapes, sizes, and textures. We first conduct a quantitative evaluation of our system's generalization ability on grasping hanging cherries with varying shapes and sizes and a hanging marble ball with a slippery surface that made grasping with chopsticks taxing even for humans (Ke et al., 2020b). Further, we conduct qualitative evaluations on an assortment of objects and dynamic

scenarios (Fig. 6.9), grasping objects of varying shapes (clip, chess), sizes (duck, puzzles) and textures (chess, grapes) and with varying non-rigid support (wind disturbance, floating water, melting cream, falling sand). We further included experiments using an off-the-shelf detection system, YOLO, which empowered our system to grasp objects with more diverse shapes and colors.

Ablation. During training, we employ a high-accuracy tracking cage that eliminates perception errors, making it an ideal platform for ablation testing. We first place the tracker ball at a fixed position and ask the agent to grasp the **Static** ball, examining how precisely the agent can grasp. Second, we hang the ball and give it a random initial velocity (**Dynamic** ball), testing the agent's reactivity to dynamic scenes.

6.4.2 *Baselines*

We evaluate our agent on five candidate baselines: (1) 20HZ VS, our heuristic-based controller that we used to collect demonstrations, and (2) 100HZ VS: our heuristic controller with a different set of gains. Optionally, we test (3) Humans: a human expert teleoperation the robot, (4) Replay: replaying successful human demonstrations, and (5) Behavior Cloning (BC): a practical imitation learning algorithm (Pomerleau, 1988b) trained with our collected data, detailed in App. **??**.

For each task and each agent we selected, we test over 10 trials and summarize performance statistics. In total, we conducted more than 500 trials. For fair evaluation and to accommodate the slow motion of the human baseline, we use a fixed horizon (6 seconds) at test time.We determine the success of the trajectory by inspecting whether the robot could securely lift the object and stabilize it at a height of 0.1m in the world frame.

6.4.3 *Reactivity to dynamic disturbance*

Fig 6.11 shows the agents' performance under varying degrees of dynamic disturbance injected by the motor dragging. We observe that our RL agent outperforms the baselines by a large margin across all degrees of motor dragging disturbance. In the human disturbance task, our RL agent succeeded in 17 of 29 trials of human disturbance within the time limit.

6.4.4 *Robustness to noise*

As shown in Table 6.2, when Gaussian noise is applied to the sensor, the RL agent performed the best, achieving a 90% success rate, significantly outperforming the baseline VSs (20Hz achieved 50% and 100Hz achieved 60% success rates). The noisy state estimation is fatal to our vanilla VS baselines which took hours of gain tuning, and it will require further tuning or adding of adaptive modules to improve performance. In contrast, our agent was naturally robust to those noises due to our design choice of `SimR`, `Proxy`, and `StoRe` to train the agent on more challenging tasks before deployment.

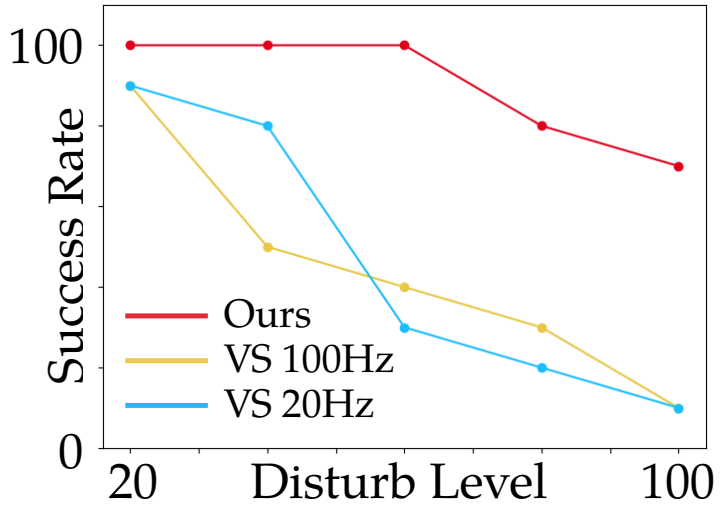


Figure 6.11: Success rate under different disturbance. The CherryBot controller's RL agent is robust to disturbances that significantly exceed those for baseline methods.

6.4.5 *Generalization*

Table 6.2 displays our quantitative results on hanging marble balls and cherries, measuring the rewards, success rate and time-to-success on picking up these objects. Our agent significantly outperforms the baselines; the baselines succeeded in some trials quickly, but they could not readjust to their failures to complete the task within the allocated time and instead kept batting the object around.

Task	**Succ. (%)**	**Task**	**Succ. (%)**	**Task**	**Succ. (%)**
Water Cherry	60 (9/15)	Puzzle	100 (10/10)	Duck	50 (5/10)
Clip	30 (3/10)	Cream Cherry	100 (10/10)	Sand Cherry	53.3 (8/15)
Chess	80 (8/10)	Grapes	70 (7/10)	Tree Cherry	40 (4/10)

Table 6.3: Success rates of generalization tasks. CherryBot learned policies and was able to generalize non-trivially across objects.

We further evaluate our agent on more diverse generalization tasks in Fig 6.9 and show the success rate in Table 6.3. Our agent's overall success rate was 64% (of 100 trials), even with our simple perception module. We summarize in four categories the factors explaining the low success rates on some generalization tasks: (1) the object has no firm contact point and keeps slipping out (clip and chess), (2) the object size is too big (duck), (3) the failure of grasping is fatal (cherry drops in the falling-sand task), and (4) perception errors due to occlusion (tree cherry-pick).

6.4.6 *Further analysis*

We examine the performance of all baselines on our proxy task, which removes perception noises but tests the agent's precision and dynamic reactivity. Table 6.2 presents results on proxy tasks in both static and dynamic modes. In general, our RL agent outperforms all baselines in success rate, time to succeed, and total rewards. Both the 20Hz and 100Hz VS baselines could also achieve 90 to 100% success rates, respectively, on the proxy tasks, albeit taking a longer time to succeed than our RL agent. It is worth noting that our RL agent, which runs at a hybrid control frequency, uses less time to succeed compared to 100HZ VS. This implies that our hierarchical framework with its low-frequency learned policy develops a more efficient strategy for grasping in the dynamic scene than our hand-designed controller.

Are we cherry-picking a cherry-picking agent? To clarify the reproducibility of our proposal: we use an open-sourced codebase d3rlpy (Seno and Imai, 2022), with default implementations and hyperparameters, to run our ablation studies in the simulation and our real-world robotic experiments. We conducted 5 random seed sweeping in simulator ablation studies and 3 random seed sweeping in real-world fine-tuning studies, verifying the the efficacy and reproducibility of our proposal.

6.5 DISCUSSION

We present a system, CherryBot, for learning robust dynamic fine grasping in unstable conditions. We provide empirical evidence demonstrating the effectiveness of our proposed system on a low-cost chopsticks-equipped robot and validate its efficiency, reactivity, robustness and potential to generalize, showing how reinforcement learning can be a practical and competitive tool to learn dynamic and precise behavior.

Despite successfully performing a diverse set of grasping tasks in the real world, there remain significant challenges to address in future work. Expanding our system to a wider range of tasks, including those with longer time horizons, could broaden its applicability. Upgrading the current perception module to include object pose and to have an end-to-end vision-based RL system could enhance its usability and allow further fine-tuning after being deployed. Exploring the theoretical reasons behind the importance of our design decisions, or providing insights into their applicability to different hardware or domains, would also open fascinating areas of future study.

6.6 SUPPLEMENTARY

6.6.1 *System Design Ablations*

All ablation experiments mentioned in 5.3 were ran with the same experimental design to make fair comparisons. Unless otherwise specified in the paper, we focus on the SAC implementation provided by d3rlpy library, using the default hyperparameters in d3rlpy library, sweeping across the seeds 120, 121, 122, 123, and 124. When run in simulation, we use the simulator whose tuning process is described in Sec. 2.3, and real-life experiments are run with the same hardware and perception as described in Sec. 6.3.

Aside from the ablations studied in depth in Section 5.3, we also performed a few more experiments validating other design choices.

UPDATE-TO-DATA RATIO The UTD (Update-to-Data ratio) controls the ratio of gradient steps per environment step. Our system achieves an effective UTD=10 20 through asynchronous updates. We illustrate in Fig. 6.12 that higher UTD values of 10 and 20 result in much faster convergence than the traditional choice of UTD=1. Running more updates for the same amount of data seems to drastically increase the sample efficiency and expedite learning. The key takeaway is that a large UTD ratio can be favored for its practicality in real world training with real time constraints and it showed empirically improvement to learning speed, making real-world learning more tractable.

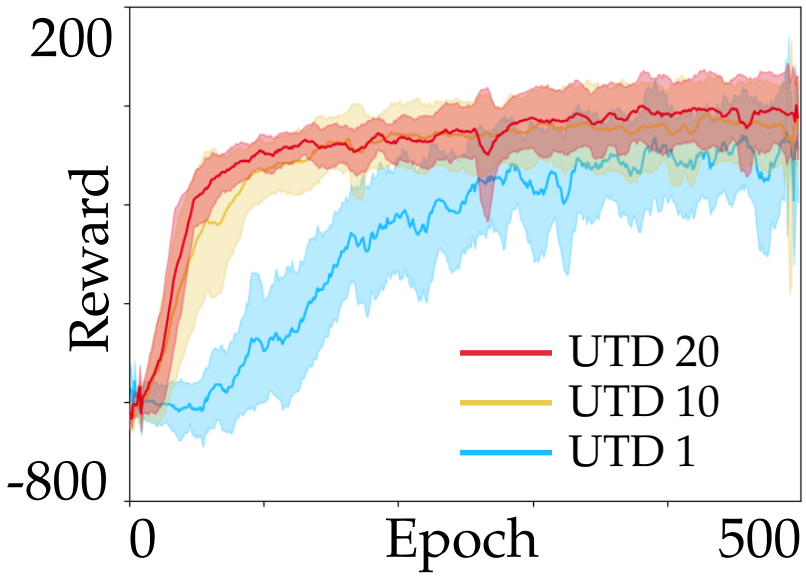


Figure 6.12: Training our RL agent in the simulator with varying values of UTD. Higher values of UTD converge much faster than lower ones.

STOCHASTIC AND STATIC RESET PERFORMANCE Our proxy task features a ball swinging from a string which, notably, has a stochastic reset, as described in Section 6.2.3.

We illustrate the impact of this design choice via an ablation study in the simulator, training two agents with static or stochastic reset for the same number of epochs. During the evaluation, we test the two agent's performance on the static-reset task and stochastic-reset task. The one trained with stochastic reset yielded an average score of 198.8 on 10 trials, significantly higher than the agent trained with static reset, which yielded only 127.8. Evidently, the use of stochastic reset, which yields a larger and more varied initial state distribution, promotes a better-performing policy.

6.6.2 *Task Design*

To explore the problem of dynamics fine manipulation, we use a proxy task (as described in the main paper) as a representative example during our agent's fine-tuning stage and then solve other generalization tasks in zero-shot at deployment time. Instead of "catching" the moving objects directly, we move the agent to be close to the target first and then manipulate it. Thus, the challenging task was divided divided into two stages: 1) the simple, slow reaching stage and 2) the reactive grasping stage. During the first stage, assuming the position of the moving objects is known, we use a heuristic policy to approach the object slowly to be within about $\sim$ 3 centimeters away (as the object moves, the distance is not guaranteed to be upper bounded). Then, we will use the learned RL policy or other baseline policies to solve the task in the reactive grasping stage.

Success criteria: The task is considered successful if the robot is able to firmly grasp and lift the correct target object. In the real world, we also used the force torque sensor on the robot joint (which yielded noisy readings) to judge the success of the grasp.

Reward design Prior RL works had reported using both sparse and dense rewards for manipulation tasks to encourage a desired motion (Rajeswaran et al., 2017; Riedmiller et al., 2018; Zhu et al., 2019). Our reward function also consists of two parts: dense rewards and sparse rewards. The various dense and sparse terms of the reward function are described in Table 6.4.

6.6.3 *Heuristic controller design*

We design our VS(visual servo) controller by following human's grasping philosophy in a state-machine manner. First, the robot will align the centroid of the object with the center of chopsticks tips using PD control in 3 axes. It adjusts its pose until the position offset is within a small threshold (1mm), and will then close the chopsticks to attempt the grasp. After that, it will move the object to the goal position with a proportional positional controller. If the robot cannot maintain the height offset within 1 mm during any state, we will return the policy back to the alignment state. The heuristic-based VS controllers are sensitive to gain-tuning and precision of the perception. We spent a reasonable amount of hours tuning the gains of our controller, but there perhaps still remains room for improvement.

Dense Rewards	
Weight	Description
-5	$\|\cdot\|_2$ between object and the chopsticks
-10	$\|\cdot\|_2$ between object and goal
+1	Correctness of distance between chopstick tips
+2	Squeezing torque of chopstick
Sparse Rewards	
-5	Object stayed around initial position
+5	Firmly grasp the object and lift to goal
-10	$\|\cdot\|_2$ between object and chopsticks >10cm

Table 6.4: Rewards design

GENERALIZATION TASKS AND THEIR EMPHASIS Our chosen real-world evaluation tasks effectively capture our agent's ability to generalize from the proxy task to more practical settings. Particularly, these tasks feature real-world complications that are not present at training time, allowing us to test the agent's ability to compensate for it at test time.

- Cherry-Picking: Our agent's eponymous cherry-picking task is complicated by noisy perception affected by occlusion. The branches and leaves get in the way of the camera, resulting in biased and noisy estimates of the cherry's position. In succeeding, the agent shows its ability to overcome these perception challenges.
- Water Cherry, Cream Cherry, Sand Cherry, and Grape: These tasks' main defining traits are the presence of new, unmodeled dynamics. In the case of the cherry floating in the water, if the cherry is disturbed it will float in the direction of the applied force. The cream cherry is relatively less mobile, but the base upon which it sits is deformable. Similarly, the cherry atop the sand is resting on a very unstable surface that is actively falling apart over time. Finally, the grape is on top of a pile of other grapes which tends to collapse when disturbed. In all four cases, there are new dynamics that are completely unrepresentative of conditions at train time. Evidently, the agent is able to generalize to these new dynamics and successfully solve the task.
- Clip, Chess, Puzzle, and Duck: These tasks showcase the agent's ability to grasp objects with completely different shapes. The agent was trained only to grasp small spheres, but in succeeding in these tasks, it shows that the agent still learns a grasping strategy that is widely applicable to other types of shapes.

ARE WE CHERRY-PICKING A CHERRY-PICKING AGENT? We used standard, open-source implementation of RL algorithms, d3rlpy, to run our ablation studies in the

simulation and our real-world robotic experiments. We use the default implementation of SAC and IQL in d3rlpy alongside the default hyperparameters. We did not change the hyperparameters or the algorithms when we ran the experiment. To enable d3rlpy to run on the real robot: we changed the location of the action normalization function in d3rlpy to ensure it will be called both for the offline dataset and online finetuning on a real robot.

Prior to running the experiments we fixed the random seed. To verify the significance of our findings in the ablation study, we conducted seed sweeping in simulator, with 5 consective random seed 120 ~ 124, as shown in 6.7. To verify the reproducibility of our proposed system in the real world, we trained our whole system including pretraining and fine-tuning with seed 121 ~ 124), the performance showed in 6.6a is summarized over all random seeds ran in the real world.

HUMAN PERFORMANCE AND IMITATION LEARNING In our early exploration, we experimented with human tele-operator and simple imitation learning. We noted that, replay of successful demonstration did not have 100% success rate, suggesting that fine manipulation requires a precise and reactive policy to be robust to potential sensor and actuator errors. Surprisingly, in both static and dynamic ball setups, BC's performance is even worse than replay, suggesting that the performance of imitation learning is heavily dependent on data support. It is hard to apply BC on these fine manipulation tasks because a tiny divergence between train and test will shift the agent far from success, motivating us to seed self-supervised learning method that can learn from trial and error.

7

CONCLUSION

This dissertation has explored the challenges and data-driven approaches to robotic fine manipulation. Our overarching goal was to develop data-driven methods that address the complexities of fine manipulation in a generalizable manner. While our initial approach focused heavily on leveraging data, we discovered that the integration of structural priors and leveraging existing knowledge about dynamics and constraints can make the learning process significantly more efficient and robust.

For instance, in our work on imitation learning, we found that leveraging structures within the dynamics of the system allowed us to better mimic expert behaviors. By understanding and incorporating the underlying physics of the tasks, we were able to synthesize data with quality guarantees and improve the learned agents. Similarly, in reinforcement learning, using prior information such as an approximate simulator and low-quality demonstrations helped bootstrap the learning process, allowing the robot to explore more effectively and learn from its environment more robustly.

A recurring question in data-driven robotics is whether data alone can solve all challenges. The question of "how much data is enough" is a complex one, and there is no one-size-fits-all answer for all tasks. In fields such as natural language processing, large datasets and powerful models have led to significant breakthroughs. While the structures of language are purely a matter of human convention, robotics must contend with the immutable laws of physics, which imposes additional constraints that data alone might miss. Our research suggests that while data is critical, it is not the sole solution.

The physical interactions that robots must navigate are governed by dynamics and kinematics that require precise modeling and understanding. Thus, purely data-driven approaches, while powerful, often fall short in scenarios requiring high precision and reliability. Our work highlights the importance of integrating structural and prior knowledge into data-driven methods. This integration not only enhances the efficiency of learning but also ensures that the learned models are more robust and generalizable to new tasks and environments.

7.1 FUTURE DIRECTION

LEVERAGING STRUCTURAL PRIORS One of the key takeaways from our research is the critical role of structural priors in robotic learning. To truly harness the potential of data-driven methods, it is essential to identify and integrate structures and constraints into the learning process.

Looking forward, there are several exciting directions. One promising avenue is the integration of data-driven methods with classical control approaches to provide robustness and safety guarantees while maintaining the flexibility to generalize to

novel tasks and environments. This hybrid approach could lead to more reliable and adaptable robotic systems capable of performing a wide range of fine manipulation tasks.

Another intriguing direction is the use of large language and vision models to extract common sense knowledge and reduce the human burden in specifying tasks for robots. By leveraging the vast knowledge embedded in these models, we could make data-driven robotics more efficient and scalable, enabling robots to understand and perform complex tasks with minimal human intervention.

DATA COLLECTION AND DEPLOYMENT PARADIGMS There is a paramount question for any data-driven method: where does the data come from? How could one characterize the data? For robotics, data is limited. Building a robust data collection pipeline is thus a critical area for future work. While teleoperation has proven effective for collecting high-quality demonstration data, it is not the end goal. Our vision is to design robotic systems that can proactively solicit data, continuously learn, and adapt by requesting data from users or collecting experiences from their interactions in real-world environments. This paradigm shift from pre-programmed task execution to autonomous data collection and learning would significantly enhance the scalability and applicability of robotic systems.

This approach aligns with the goal of creating robots that are not just capable of performing predefined tasks but are also able to learn new tasks after deployment. Such a paradigm would enable robots to be more versatile and useful in dynamic and unstructured environments, where manual programming for every possible task is impractical. This vision could be particularly beneficial for applications like assistive homecare for the aging population, where robots need to adapt to a wide variety of tasks in natural and uncontrolled settings.

FINE MANIPULATION IN UNSTRUCTURED, REAL-WORLD APPLICATIONS The drive to develop fine manipulation robots for real-world applications is particularly pertinent in the context of an aging global population. With over 400 million people aged 65 and older worldwide, and projections indicating that this demographic will surpass the number of children in the U.S. within a decade, there is an urgent need for robotic solutions that can assist with daily tasks, augmenting the dexterity of humans to enhance the quality of life for the elderly. We are therefore inspired to ground our research towards concrete applications, hoping to perform tasks such as cooking, personal care, and medical assistance.

This dissertation has demonstrated the potential of data-driven methods to enhance robotic fine manipulation. However, achieving human-like dexterity across all areas remains a long-term goal. The integration of structural priors, the development of advanced data collection methods, and the exploration of new learning paradigms will be crucial in advancing this field. Ultimately, the mission is to build fine manipulation robots that can operate effectively in the wild, accommodating the diverse and dynamic needs of the aging population and beyond.

BIBLIOGRAPHY

Altman, Naomi S (1992). "An introduction to kernel and nearest-neighbor nonparametric regression." In: *The American Statistician* 46.3, pp. 175–185.

Arjovsky, Martin, Soumith Chintala, and Léon Bottou (2017). "Wasserstein Generative Adversarial Networks." In: *International Conference on Machine Learning*. PMLR, pp. 214–223.

Asadi, Kavosh, Dipendra Misra, and Michael L. Littman (2018). "Lipschitz Continuity in Model-based Reinforcement Learning." In: *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*. Ed. by Jennifer G. Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, pp. 264–273. URL: http://proceedings.mlr.press/v80/asadi18a.html.

Ba, Jimmy Lei, Jamie Ryan Kiros, and Geoffrey E Hinton (2016). "Layer normalization." In: *arXiv preprint arXiv:1607.06450*.

Bartlett, Peter L, Dylan J Foster, and Matus J Telgarsky (2017). "Spectrally-Normalized Margin Bounds for Neural Networks." In: *NeurIPS* 30.

Bhattacharjee, Tapomayukh, Gilwoo Lee, Hanjun Song, and Siddhartha S Srinivasa (2019). "Towards robotic feeding: Role of haptics in fork-based food manipulation." In: *IEEE Robotics and Automation Letters* 4.2, pp. 1485–1492.

Billard, Aude G, Sylvain Calinon, and Rüdiger Dillmann (2016). "Learning from humans." In: *Springer handbook of robotics*. Springer, pp. 1995–2014.

Billard, Aude and Danica Kragic (2019). "Trends and challenges in robot manipulation." In: *Science* 364.6446.

Bishop, Chris M (1995). "Training with noise is equivalent to Tikhonov regularization." In: *Neural Computation* 7.1, pp. 108–116.

Block, Adam, Ali Jadbabaie, Daniel Pfrommer, Max Simchowitz, and Russ Tedrake (2024). "Provable Guarantees for Generative Behavior Cloning: Bridging Low-Level Stability and High-Level Behavior." In: *Advances in Neural Information Processing Systems* 36.

Bojarski, Mariusz, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. (2016). "End to End Learning for Self-Driving Cars." In: *arXiv preprint arXiv:1604.07316*.

Bonnard, Bernard, Jean-Baptiste Caillau, and Emmanuel Trélat (2007). "Second Order Optimality Conditions in The Smooth Case and Applications in Optimal Control." In: *ESAIM: Control, Optimisation and Calculus of Variations* 13.2, pp. 207–236.

Brown, Eric, Nicholas Rodenberg, John Amend, Annan Mozeika, Erik Steltz, Mitchell R. Zakin, Hod Lipson, and Heinrich M. Jaeger (2010). "Universal robotic gripper based on the jamming of granular material." In: *Proceedings of the National Academy of Sciences* 107.44, pp. 18809–18814.

Calli, Berk, Arjun Singh, James Bruce, Aaron Walsman, Kurt Konolige, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar (2017). "Yale-CMU-Berkeley dataset for robotic manipulation research." In: *The International Journal of Robotics Research* 36.3, pp. 261–268.

Chang, Bao-Chi, Biing-Shiun Huang, Ching-Kong Chen, and Shyh-Jen Wang (2007). "The pincer chopsticks: The investigation of a new utensil in pinching function." In: *Applied ergonomics* 38.3, pp. 385–390.

Chang, Jonathan, Masatoshi Uehara, Dhruv Sreenivas, Rahul Kidambi, and Wen Sun (2021). "Mitigating Covariate Shift in Imitation Learning via Offline Data with Partial Coverage." In: *2021 Advances in Neural Information Processing Systems* 34, pp. 965–979.

Chang, Kai-Wei, Akshay Krishnamurthy, Alekh Agarwal, Hal Daumé III, and John Langford (2015). "Learning to Search Better than Your Teacher." In: *International Conference on Machine Learning*.

Chaumette, François and Seth Hutchinson (2006). "Visual servo control. I. Basic approaches." In: *IEEE Robotics & Automation Magazine* 13.4, pp. 82–90.

Chen, Xinyue, Che Wang, Zijian Zhou, and Keith Ross (2021). "Randomized ensembled double q-learning: Learning fast without a model." In: *arXiv preprint arXiv:2101.05982*.

Chepisheva, Mariya, Utku Culha, and Fumiya Iida (2016). "A biologically inspired soft robotic hand using chopsticks for grasping tasks." In: *International Conference on Simulation of Adaptive Behavior*. Springer, pp. 195–206.

Chi, Cheng, Benjamin Burchfiel, Eric Cousineau, Siyuan Feng, and Shuran Song (2022). "Iterative residual policy: for goal-conditioned dynamic manipulation of deformable objects." In: *arXiv preprint arXiv:2203.00663*.

Chi, Cheng, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song (2023). "Diffusion Policy: Visuomotor Policy Learning via Action Diffusion." In: *2023 Robotics: Science and Systems*.

Cutkosky, Mark R (2012). *Robotic grasping and fine manipulation*. Vol. 6. Springer Science & Business Media.

Dyrstad, Jonatan S, Elling Ruud Øye, Annette Stahl, and John Reidar Mathiassen (2018). "Teaching a Robot to Grasp Real Fish by Imitation Learning from a Human Supervisor in Virtual Reality." In: *International Conference on Intelligent Robots and Systems*. IEEE, pp. 7185–7192.

Florence, Peter, Lucas Manuelli, and Russ Tedrake (2019). "Self-Supervised Correspondence in Visuomotor Policy Learning." In: *IEEE Robotics and Automation Letters* 5.2, pp. 492–499.

Fu, Justin, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine (2020). "D4rl: Datasets for Deep Data-Driven Reinforcement Learning." In: *arXiv preprint arXiv:2004.07219*.

Fu, Justin, Katie Luo, and Sergey Levine (2018). "Learning Robust Rewards with Adversarial Inverse Reinforcement Learning." In: *2018 International Conference on Learning Representations*.

Fujimoto, Scott, David Meger, and Doina Precup (2018). "Off-Policy Deep Reinforcement Learning without Exploration. CoRR abs/1812.02900 (2018)." In: *arXiv preprint arXiv:1812.02900*.

Garrett, Caelan Reed, Rohan Chitnis, Rachel Holladay, Beomjoon Kim, Tom Silver, Leslie Pack Kaelbling, and Tom'as Lozano-P'erez (2021). "Integrated task and motion planning." In: *Annual review of control, robotics, and autonomous systems* 4, pp. 265–293.

Goodfellow, Ian, Yoshua Bengio, Aaron Courville, and Yoshua Bengio (2016). *Deep Learning*. Vol. 1. MIT press Cambridge.

Gubbi, Sagar, Shishir Kolathaya, and Bharadwaj Amrutur (2020). "Imitation Learning for High Precision Peg-In-Hole Tasks." In: *ICCAR*.

Gulrajani, Ishaan, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville (2017). "Improved Training of Wasserstein Gans." In: *Advances in Neural Information Processing Systems* 30.

HEBI Robotics (n.d.). https://www.hebirobotics.com. Accessed: [02/29/2020].

Haarnoja, Tuomas, Sehoon Ha, Aurick Zhou, Jie Tan, George Tucker, and Sergey Levine (2018a). "Learning to walk via deep reinforcement learning." In: *arXiv preprint arXiv:1812.11103*.

Haarnoja, Tuomas, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. (2018b). "Soft actor-critic algorithms and applications." In: *arXiv preprint arXiv:1812.05905*.

Hafner, Danijar, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi (2020). "Dream to Control: Learning Behaviors by Latent Imagination." In: *2020 International Conference on Learning Representations*.

Hashimoto, Koichi, Akio Namiki, and Masatoshi Ishikawa (2001). "A visuomotor control architecture for high-speed grasping." In: *Proceedings of the 40th IEEE Conference on Decision and Control (Cat. No. 01CH37228)*. Vol. 1. IEEE, pp. 15–20.

Hearst, Marti A., Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf (1998). "Support Vector Machines." In: *IEEE Intelligent Systems and Their Applications* 13.4, pp. 18–28.

Ho, Jonathan and Stefano Ermon (2016). "Generative Adversarial Imitation Learning." In: *NeurIPS* 29.

Ho, Jonathan, Jayesh Gupta, and Stefano Ermon (2016). "Model-free imitation learning with policy optimization." In: *International Conference on Machine Learning*, pp. 2760–2769.

Hogan, Francois R and Alberto Rodriguez (2020). "Reactive planar non-prehensile manipulation with hybrid model predictive control." In: *The International Journal of Robotics Research* 39.7, pp. 755–773.

Hong, Joey, Aviral Kumar, and Sergey Levine (2022). "Confidence-Conditioned Value Functions for Offline Reinforcement Learning." In: *arXiv preprint arXiv:2212.04607*.

Hutchinson, Seth, Gregory D Hager, and Peter I Corke (1996). "A tutorial on visual servo control." In: *IEEE transactions on robotics and automation* 12.5, pp. 651–670.

Hwang, Minho, Jeffrey Ichnowski, Brijen Thananjeyan, Daniel Seita, Samuel Paradis, Danyal Fer, Thomas Low, and Ken Goldberg (2022). "Automating Surgical Peg Trans-

fer: Calibration With Deep Learning Can Exceed Speed, Accuracy, and Consistency of Humans." In: *IEEE Transactions on Automation Science and Engineering*.

Jones, Kelvin E, Antonia F de C Hamilton, and Daniel M Wolpert (2002). "Sources of signal-dependent noise during isometric force production." In: *Journal of neurophysiology* 88.3, pp. 1533–1544.

Kahveci, Nazli E (2007). "Robust Adaptive Control For Unmanned Aerial Vehicles." PhD thesis. University of Southern California.

Kalashnikov, Dmitry, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. (2018). "Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation." In: *arXiv preprint arXiv:1806.10293*.

Kaufmann, Elia, Leonard Bauersfeld, Antonio Loquercio, Matthias Müller, Vladlen Koltun, and Davide Scaramuzza (2023). "Champion-Level Drone Racing Using Deep Reinforcement Learning." In: *Nature* 620.7976, pp. 982–987.

Ke, Liyiming, Sanjiban Choudhury, Matt Barnes, Wen Sun, Gilwoo Lee, and Siddhartha Srinivasa (2020a). "Imitation Learning as f-Divergence Minimization." In: *Proceedings of the Workshop on Algorithmic Foundations of Robotics*.

– (2021a). "Imitation Learning as F-Divergence Minimization." In: *Algorithmic Foundations of Robotics XIV: Proceedings of the Fourteenth Workshop on the Algorithmic Foundations of Robotics 14*. Springer, pp. 313–329.

Ke, Liyiming, Ajinkya Kamat, Jingqiang Wang, Tapomayukh Bhattacharjee, Christoforos Mavrogiannis, and Siddhartha S. Srinivasa (2020b). "Telemanipulation with Chopsticks: Analyzing Human Factors in User Demonstrations." In: *International Conference on Intelligent Robots and Systems*. IEEE.

Ke, Liyiming, Jingqiang Wang, Tapomayukh Bhattacharjee, Byron Boots, and Siddhartha Srinivasa (2021b). "Grasping with chopsticks: Combating covariate shift in model-free imitation learning for fine manipulation." In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 6185–6191.

Khetarpal, Khimya, Zafarali Ahmed, Gheorghe Comanici, David Abel, and Doina Precup (2020). "What Can I Do Here? a Theory of Affordances in Reinforcement Learning." In: *International Conference on Machine Learning*. PMLR, pp. 5243–5253.

Kidambi, Rahul, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims (2020). "MOReL: Model-Based Offline Reinforcement Learning." In: *Advances in Neural Information Processing Systems*. Vol. 33, pp. 21810–21823.

Kontoudis, G. P., M. V. Liarokapis, A. G. Zisimatos, C. I. Mavrogiannis, and K. J. Kyriakopoulos (2015). "Open-source, anthropomorphic, underactuated robot hands with a selectively lockable differential mechanism: Towards affordable prostheses." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5857–5862.

Kostrikov, Ilya, Ashvin Nair, and Sergey Levine (2021). "Offline reinforcement learning with implicit q-learning." In: *arXiv preprint arXiv:2110.06169*.

Kumar, Aviral, Aurick Zhou, George Tucker, and Sergey Levine (2020). "Conservative q-learning for offline reinforcement learning." In: *Advances in Neural Information Processing Systems* 33, pp. 1179–1191.

Kumar, Vikash, Emanuel Todorov, and Sergey Levine (2016). "Optimal control with learned local models: Application to dexterous manipulation." In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 378–383.

Lange, Sascha, Thomas Gabel, and Martin Riedmiller (2012). "Batch reinforcement learning." In: *Reinforcement learning: State-of-the-art*, pp. 45–73.

Laskey, Michael, Jonathan Lee, Roy Fox, Anca Dragan, and Ken Goldberg (2017). "Dart: Noise injection for robust imitation learning." In: *arXiv preprint arXiv:1703.09327*.

Levine, Sergey, Aviral Kumar, George Tucker, and Justin Fu (2020). "Offline reinforcement learning: Tutorial, review, and perspectives on open problems." In: *arXiv preprint arXiv:2005.01643*.

Li, Shuguang, John J Stampfli, Helen J Xu, Elian Malkin, Evelin Villegas Diaz, Daniela Rus, and Robert J Wood (2019). "A vacuum-driven origami "magic-ball" soft gripper." In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 7401–7408.

Li, Weiwei and Emanuel Todorov (2004). "Iterative Linear Quadratic Regulator Design for Nonlinear Biological Movement Systems." In: *ICINCO (1)*. Citeseer, pp. 222–229.

Lynch, Kevin M and Matthew T Mason (1999). "Dynamic nonprehensile manipulation: Controllability, planning, and experiments." In: *The International Journal of Robotics Research* 18.1, pp. 64–92.

Mahler, Jeffrey, Florian T Pokorny, Brian Hou, Melrose Roderick, Michael Laskey, Mathieu Aubry, Kai Kohlhoff, Torsten Kr"oger, James Kuffner, and Ken Goldberg (2016). "Dex-net 1.0: A cloud-based network of 3d objects for robust grasp planning using a multi-armed bandit model with correlated rewards." In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 1957–1964.

Marohn, Col Michael R and Capt Eric J Hanly (2004). "Twenty-first century surgery using twenty-first century technology: Surgical robotics." In: *Current Surgery* 61.5, pp. 466–473.

Mason, Matthew T and Kevin M Lynch (1993). "Dynamic manipulation." In: *Proceedings of 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'93)*. Vol. 1. IEEE, pp. 152–159.

Mason, Matthew T., Siddhartha S. Srinivasa, and Andres S. Vazquez (2011). "Generality and Simple Hands." In: *Robotics Research*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 345–361.

Melchiorri, Claudio and Makoto Kaneko (2016). "Robot hands." In: *Springer Handbook of Robotics*. Springer, pp. 463–480.

Miyato, Takeru, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida (2018). "Spectral Normalization for Generative Adversarial Networks." In: *2018 International Conference on Learning Representations*.

Monkman, Gareth J, Stefan Hesse, Ralf Steinmann, and Henrik Schunk (2007). *Robot grippers*. John Wiley & Sons.

Mordatch, Igor, Emanuel Todorov, and Zoran Popovi'c (2012). "Discovery of complex behaviors through contact-invariant optimization." In: *ACM Transactions on Graphics (ToG)* 31.4, pp. 1–8.

Nair, Suraj, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta (2022). *R3M: A Universal Visual Representation for Robot Manipulation*. arXiv: 2203.12601 [cs.RO].

Odhner, Lael U., Leif P. Jentoft, Mark R. Claffee, Nicholas Corson, Yaroslav Tenzer, Raymond R. Ma, Martin Buehler, Robert Kohout, Robert D. Howe, and Aaron M. Dollar (2014). "A compliant, underactuated hand for robust manipulation." In: *The International Journal of Robotics Research* 33.5, pp. 736–752.

Okumura, Kohei, Hiromasa Oku, and Masatoshi Ishikawa (2011). "High-speed gaze controller for millisecond-order pan/tilt camera." In: *2011 IEEE International Conference on Robotics and Automation*. IEEE, pp. 6186–6191.

Oliveira, Felipe Dennis de Resende, Eduardo Luiz Ortiz Batista, and Rui Seara (2021). "On the Compression of Neural Networks Using 0-Norm Regularization and Weight Pruning." In: *Neural Networks* 171, pp. 343–352.

OptiTrack (n.d.). https://optitrack.com. Accessed: [02/29/2020].

Osa, Takayuki, Joni Pajarinen, Gerhard Neumann, J Andrew Bagnell, Pieter Abbeel, and Jan Peters (2018). "An algorithmic perspective on imitation learning." In: *arXiv preprint arXiv:1811.06711*.

Panerati, Jacopo, Hehui Zheng, SiQi Zhou, James Xu, Amanda Prorok, and Angela P Schoellig (2021). "Learning to Fly—a Gym Environment with Pybullet Physics for Reinforcement Learning of Multi-Agent Quadcopter Control." In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 7512–7519.

Park, Jung Yeon and Lawson Wong (2022). "Robust Imitation of a Few Demonstrations with a Backwards Model." In: *Advances in Neural Information Processing Systems* 35, pp. 19759–19772.

Pastor, Peter, Heiko Hoffmann, Tamim Asfour, and Stefan Schaal (2009). "Learning and generalization of motor skills by learning from demonstration." In: *International Conference on Robotics and Automation*. IEEE, pp. 763–768.

Pastor, Peter, Mrinal Kalakrishnan, Sachin Chitta, Evangelos Theodorou, and Stefan Schaal (2011). "Skill learning and task outcome prediction for manipulation." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3828–3834.

Peng, Xue Bin, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel (2018). "Sim-to-real transfer of robotic control with dynamics randomization." In: *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, pp. 3803–3810.

Peng, Xue Bin, Erwin Coumans, Tingnan Zhang, Tsang-Wei Lee, Jie Tan, and Sergey Levine (2020). "Learning agile robotic locomotion skills by imitating animals." In: *arXiv preprint arXiv:2004.00784*.

Peng, Xue Bin, Aviral Kumar, Grace Zhang, and Sergey Levine (2019). "Advantage-weighted regression: Simple and scalable off-policy reinforcement learning." In: *arXiv preprint arXiv:1910.00177*.

Pfrommer, Samuel, Mathew Halm, and Michael Posa (2021). "Contactnets: Learning Discontinuous Contact Dynamics with Smooth, Implicit Representations." In: *Conference on Robot Learning*. PMLR, pp. 2279–2291.

Picking Cellphones with Chopsticks (n.d.). https://newsfeed.time.com/2014/01/21/watch-these-thieves-use-chopsticks-to-pickpocket-without-leaving-fingerprints/. Accessed: [02/29/2020].

Pomerleau, Dean A (1988a). "ALVINN: an Autonomous Land Vehicle in A Neural Network." In: *Advances in Neural Information Processing Systems* 1.

– (1988b). "Alvinn: An autonomous land vehicle in a neural network." In: *Advances in neural information processing systems* 1.

– (1989). "Alvinn: An autonomous land vehicle in a neural network." In: *Advances in Neural Information Processing Systems*, pp. 305–313.

Poole, Ben, Jascha Sohl-Dickstein, and Surya Ganguli (2014). "Analyzing noise in autoencoders and deep networks." In: *arXiv preprint arXiv:1406.1831*.

Questionnaire (n.d.). https://drive.google.com/drive/folders/156RcpdsgiGzft6wqlJPczv1EHFzJEkbQ?usp=sharing. Accessed: [02/29/2020].

Quionero-Candela, Joaquin, Masashi Sugiyama, Anton Schwaighofer, and Neil D Lawrence (2009). *Dataset Shift in Machine Learning*. The MIT Press.

Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever (2021). *Learning Transferable Visual Models From Natural Language Supervision*. arXiv: 2103.00020 [cs.CV].

Rajeswaran, Aravind, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine (2017). "Learning complex dexterous manipulation with deep reinforcement learning and demonstrations." In: *arXiv preprint arXiv:1709.10087*.

Ramadan, Ahmed A, Tomohito Takubo, Yasushi Mae, Kenichi Oohara, and Tatsuo Arai (2009). "Developmental process of a chopstick-like hybrid-structure two-fingered micromanipulator hand for 3-D manipulation of microscopic objects." In: *IEEE Transactions on Industrial Electronics* 56.4, pp. 1121–1135.

Ratliff, Nathan D, J Andrew Bagnell, and Martin A Zinkevich (2006). "Maximum Margin Planning." In: *Proceedings of the 23rd international conference on Machine learning*, pp. 729–736.

Reddy, Siddharth, Anca D Dragan, and Sergey Levine (2020). "Sqil: Imitation Learning via Reinforcement Learning with Sparse Rewards." In: *2020 International Conference on Learning Representations*.

Reichlin, Alfredo, Giovanni Luca Marchetti, Hang Yin, Ali Ghadirzadeh, and Danica Kragic (2022). "Back to The Manifold: Recovering from Out-Of-Distribution States." In: *International Conference on Intelligent Robots and Systems (IROS)*. IEEE.

Rhodes, Travers and Manuela Veloso (2018). "Robot-driven trajectory improvement for feeding tasks." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2991–2996.

Riedmiller, Martin, Roland Hafner, Thomas Lampe, Michael Neunert, Jonas Degrave, Tom Wiele, Vlad Mnih, Nicolas Heess, and Jost Tobias Springenberg (2018). "Learning by playing solving sparse reward tasks from scratch." In: *International conference on machine learning*. PMLR, pp. 4344–4353.

Robertson, Matthew A. and Jamie Paik (2017). "New soft robots really suck: Vacuum-powered systems empower diverse capabilities." In: *Science Robotics* 2.9.

Rodriguez, Alberto, Matthew T. Mason, and Siddhartha S. Srinivasa (2014). "Manipulation Capabilities with Simple Hands." In: *Experimental Robotics: The 12th International Symposium on Experimental Robotics*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 285–299.

Ross, Stéphane, Geoffrey Gordon, and Drew Bagnell (2011). "A reduction of imitation learning and structured prediction to no-regret online learning." In: *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635.

Russell, Stuart and Peter Norvig (2009). *Artificial Intelligence: A Modern Approach*. 3rd. USA: Prentice Hall Press. ISBN: 0136042597.

Sakurai, Haruka, Takahiro Kanno, and Kenji Kawashima (2016). "Thin-diameter chopsticks robot for Laparoscopic Surgery." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4122–4127.

Sanderson, Arthur C and Lee E Weiss (1983). "Adaptive visual servo control of robots." In: *Robot vision*, pp. 107–116.

Sarangapani, Jagannathan (2018). *Neural Network Control of Nonlinear Discrete-Time Systems*. CRC press.

Schaal, Stefan (2006). "Dynamic movement primitives-a framework for motor control in humans and humanoid robotics." In: *Adaptive motion of animals and machines*, pp. 261–280.

Seno, Takuma and Michita Imai (2022). "d3rlpy: An Offline Deep Reinforcement Learning Library." In: *Journal of Machine Learning Research* 23.315, pp. 1–20. URL: http://jmlr.org/papers/v23/22-0017.html.

Seto, Danbing, Anuradha M Annaswamy, and John Baillieul (1994). "Adaptive Control of Nonlinear Systems with A Triangular Structure." In: *IEEE Transactions on Automatic Control* 39.7, pp. 1411–1428.

Shi, Guanya, Xichen Shi, Michael O'Connell, Rose Yu, Kamyar Azizzadenesheli, Animashree Anandkumar, Yisong Yue, and Soon-Jo Chung (2019). "Neural Lander: Stable Drone Landing Control Using Learned Dynamics." In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 9784–9790.

Shorten, Connor and Taghi M Khoshgoftaar (2019). "A survey on image data augmentation for deep learning." In: *Journal of Big Data* 6.1, p. 60.

Sietsma, Jocelyn and Robert JF Dow (1991). "Creating artificial neural networks that generalize." In: *Neural Networks* 4.1, pp. 67–79.

Smith, Laura, Ilya Kostrikov, and Sergey Levine (2022). "A walk in the park: Learning to walk in 20 minutes with model-free reinforcement learning." In: *arXiv preprint arXiv:2208.07860*.

Spencer, Jonathan C., Sanjiban Choudhury, Arun Venkatraman, Brian D. Ziebart, and J. Andrew Bagnell (2021). "Feedback in Imitation Learning: The Three Regimes of Covariate Shift." In: *CoRR* abs/2102.02872. arXiv: 2102.02872. URL: https://arxiv.org/abs/2102.02872.

Sugiuchi, Hajime, Tetsu Morino, and Mina Terauchi (2002). "Execution and description of dexterous hand task by using multi-finger dual robot hand system-realization

of Japanese sign language." In: *Proceedings of the IEEE International Symposium on Intelligent Control*, pp. 544–548.

Sun, Wen, Arun Venkatraman, Geoffrey J Gordon, Byron Boots, and J Andrew Bagnell (2017). "Deeply Aggrevated: Differentiable Imitation Learning for Sequential Prediction." In: *International Conference on Machine Learning*. PMLR, pp. 3309–3318.

Swamy, Gokul, Sanjiban Choudhury, J Andrew Bagnell, and Steven Wu (2021). "Of Moments and Matching: A Game-Theoretic Framework for Closing the Imitation Gap." In: *International Conference on Machine Learning*. PMLR, pp. 10022–10032.

Tedrake, Russ (2023). *Underactuated Robotics. Algorithms for Walking, Running, Swimming, Flying, and Manipulation*. URL: https://underactuated.csail.mit.edu.

Todorov, Emanuel, Tom Erez, and Yuval Tassa (2012a). "MuJoCo: A physics engine for model-based control." In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2012, Vilamoura, Algarve, Portugal, October 7-12, 2012*. IEEE, pp. 5026–5033. DOI: 10.1109/IROS.2012.6386109. URL: https://doi.org/10.1109/IROS.2012.6386109.

– (2012b). "Mujoco: A physics engine for model-based control." In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 5026–5033.

University of California Berkeley CS 285: Deep Reinforcement Learning (n.d.). http://rail.eecs.berkeley.edu/deeprlcourse/. Accessed: 2020-10-27.

Vassileva, D and G Boiadjiev (2009). "A learning control algorithm with experiments on a chopsticks robot." In: *IFAC Proceedings Volumes* 42.16, pp. 191–196.

Venkatraman, Arun, Martial Hebert, and J Andrew Bagnell (2015). "Improving multi-step prediction of learned time series models." In: *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

Wang, G and Harry E Stephanou (1988). "Chopstick manipulation with an articulated hand-a qualitative analysis." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 94–99.

Wang, Tingwu, Xuchan Bao, Ignasi Clavera, Jerrick Hoang, Yeming Wen, Eric Langlois, Shunshi Zhang, Guodong Zhang, Pieter Abbeel, and Jimmy Ba (2019). "Benchmarking Model-Based Reinforcement Learning." In: *arXiv preprint arXiv:1907.02057*.

Wang, Tongzhou, Simon S Du, Antonio Torralba, Phillip Isola, Amy Zhang, and Yuandong Tian (2022). "Denoised MDPs: Learning World Models Better than The World Itself." In: *2022 International Conference on Machine Learning*.

W"uthrich, Manuel, Felix Widmaier, Felix Grimminger, Joel Akpo, Shruti Joshi, Vaibhav Agrawal, Bilal Hammoud, Majid Khadiv, Miroslav Bogdanovic, Vincent Berenz, et al. (2020). "Trifinger: An open-source robot for learning dexterity." In: *arXiv preprint arXiv:2008.03596*.

Yamazaki, Akira and Ryosuke Masuda (2012). "Autonomous foods handling by chopsticks for meal assistant robot." In: *ROBOTIK 2012; 7th German Conference on Robotics*. VDE, pp. 1–6.

Yamazaki, Kimitoshi, Yoshiaki Watanabe, Kotaro Nagahama, Kei Okada, and Masayuki Inaba (2010). "Recognition and manipulation integration for a daily assistive robot working on kitchen environments." In: *Proceedings of the IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 196–201.

Yuan, Wenzhen, Siyuan Dong, and Edward H Adelson (2017). "Gelsight: High-resolution robot tactile sensors for estimating geometry and force." In: *Sensors* 17.12, p. 2762.

Zeng, Andy, Pete Florence, Jonathan Tompson, Stefan Welker, Jonathan Chien, Maria Attarian, Travis Armstrong, Ivan Krasin, Dan Duong, Vikas Sindhwani, et al. (2021). "Transporter networks: Rearranging the visual world for robotic manipulation." In: *Conference on Robot Learning*. PMLR, pp. 726–747.

Zeng, Andy, Shuran Song, Kuan-Ting Yu, Elliott Donlon, Francois R Hogan, Maria Bauza, Daolin Ma, Orion Taylor, Melody Liu, Eudald Romo, et al. (2022). "Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching." In: *The International Journal of Robotics Research* 41.7, pp. 690–705.

Zhang, Amy, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine (2021). "Learning Invariant Representations for Reinforcement Learning without Reconstruction." In: *2021 International Conference on Learning Representations*.

Zhang, Tianhao, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel (2018). "Deep imitation learning for complex manipulation tasks from virtual reality teleoperation." In: *International Conference on Robotics and Automation*. IEEE, pp. 1–8.

Zhang, Xiaohan, Yifeng Zhu, Yan Ding, Yuke Zhu, Peter Stone, and Shiqi Zhang (2022). "Visually grounded task and motion planning for mobile manipulation." In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 1925–1931.

Zhao, Tony Z., Vikash Kumar, Sergey Levine, and Chelsea Finn (2023). *Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware*. arXiv: 2304.13705 [cs.RO].

Zhe Xu and E. Todorov (2016). "Design of a highly biomimetic anthropomorphic robotic hand towards artificial limb regeneration." In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3485–3492.

Zhou, Allan, Moo Jin Kim, Lirui Wang, Pete Florence, and Chelsea Finn (2023a). "Nerf in The Palm of Your Hand: Corrective Augmentation for Robotics via Novel-View Synthesis." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17907–17917.

Zhou, Gaoyue, Liyiming Ke, Siddhartha Srinivasa, Abhinav Gupta, Aravind Rajeswaran, and Vikash Kumar (2023b). "Real World Offline Reinforcement Learning with Realistic Data Source." In: *2023 International Conference on Robotics and Automation*.

Zhu, Chuning, Max Simchowitz, Siri Gadipudi, and Abhishek Gupta (2023). "RePo: Resilient Model-Based Reinforcement Learning by Regularizing Posterior Predictability." In: *Advances in Neural Information Processing Systems*. Vol. 36. Curran Associates, Inc., pp. 32445–32467.

Zhu, Henry, Abhishek Gupta, Aravind Rajeswaran, Sergey Levine, and Vikash Kumar (2019). "Dexterous manipulation with deep reinforcement learning: Efficient, general, and low-cost." In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 3651–3657.

Zhu, Henry, Justin Yu, Abhishek Gupta, Dhruv Shah, Kristian Hartikainen, Avi Singh, Vikash Kumar, and Sergey Levine (2020). "The Ingredients of Real World Robotic

Reinforcement Learning." In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. URL: https://openreview.net/forum?id=rJe2syrtvS.

Zisimatos, A. G., M. V. Liarokapis, C. I. Mavrogiannis, and K. J. Kyriakopoulos (2014). "Open-source, affordable, modular, light-weight, underactuated robot hands." In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3207–3212.

COLOPHON

This dissertation was typeset using the typographical look-and-feel `classicthesis`, developed by André Miede and Ivo Pletikosić, and further refined by Amrita Mazumdar, with some imprvoements added by Maxwell Forbes.

Final Version as of June 20, 2024 (classicthesis v4.6).