Title

Model-Based Control for Complex Robotics Tasks

Author

Zheng, Tony

Publication Date

2024

Peer reviewed|Thesis/dissertation

Model-Based Control for Complex Robotics Tasks

By

Tony Zheng

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering - Mechanical Engineering

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Francesco Borrelli, Chair
Associate Professor Koushil Sreenath
Professor Tarek Zohdi

Fall 2024

Model-Based Control for Complex Robotics Tasks

Abstract

Model-Based Control for Complex Robotics Tasks

by

Tony Zheng

Doctor of Philosophy in Engineering - Mechanical Engineering

University of California, Berkeley

Professor Francesco Borrelli, Chair

In many industries, such as manufacturing and logistics, where the setting can be highly structured or simplified, robotic manipulators have been shown to improve safety, efficiency, and productivity through the automation of dangerous and repetitive tasks. As we continue to explore the use of robots in more dynamic scenarios with advanced tools or uncertain environments, reliance only on precise position control is no longer viable. Complex interactions require the robot to either plan in advance or adapt reactively to achieve success. Model-based approaches allow robots to be controlled in a constrained manner while predicting how they affect the world around them without requiring large datasets which may be difficult to obtain due to time or safety reasons.

This dissertation presents methods of modeling, planning and control for robotic manipulators to perform complex tasks while using limited data to improve performance. We examine three different applications which have their own unique set of challenges including hybrid dynamics, noisy measurements, human-robot interactions with partial knowledge on obstacles, and utilizing tools that rapidly degrade with usage. Our approaches are tested in simulation and validated in hardware experiments.

To my parents.

Contents

Contents **ii**

List of Figures **iv**

List of Tables **vi**

1 Introduction **1**
1.1 Background 1
1.2 Outline and Contributions 2
1.3 List of Publications 3

2 Learning to Play Cup-and-Ball **4**
2.1 Introduction 4
2.2 Related Work 6
2.3 Generating A Swing-up Trajectory 7
2.4 Designing Feedback Policy In Catch Phase 10
2.5 Experimental Results 12
2.6 Conclusions 15

3 Human-Robot Collaborative Transportation **17**
3.1 Introduction 18
3.2 Problem Formulation 20
3.3 Robot's Policy Design 22
3.4 Experimental Results 25
3.5 Conclusion 29

4 Deposition with Degradable Tools: Width Tracking **30**
4.1 Introduction 30
4.2 Related Work 31
4.3 Problem Formulation 33
4.4 Data-Driven Control Synthesis 37
4.5 Experimental Results 38
4.6 Conclusion 43

5 Deposition with Degradable Tools: Edge Planning **47**
5.1 Introduction 47
5.2 Problem Formulation 47
5.3 Edge Planning 50
5.4 Experimental Results 51
5.5 Conclusion 59

6 Conclusion and Future Directions **62**

Bibliography **64**

List of Figures

2.1 Manipulator with Kendama along with coordinate frame. 7
2.2 Start of catch phase (i.e., $i = N$) for 100 trajectories. Red line indicates the trajectory of the cup/end-effector during swing-up. Blue dots indicate ball positions during swing-up and pink dots indicate a position after catch phase is started. Closed-loop control begins when the relative position is in $\mathcal{E}_{\text{tr}}$. 9
2.3 Video feed from the depth camera during the catch phase of the experiment. . . 13
2.4 Percentage of times the ball hitting the cup center among all roll-outs vs sample size n. 14
2.5 One standard deviation interval around the mean (circle) of z-relative velocity at impact, i.e., $[u^{\star}_{T_{\text{im}}-1}]_z$ vs sample size n. 15
2.6 Percentage of successful catches vs sample size n. 16

3.1 The considered experiment setup. 19
3.2 Comparison of experimental results of robot with pure MPC policy vs. trust-driven policy. 27
3.3 Effect of the safe stop policy mode in avoiding collisions. 28

4.1 The considered experimental setup. 32
4.2 Stroke Analysis 34
4.3 Intersection of a cone and hyperplane. 35
4.4 Force vs penetration plot. 39
4.5 Surface plot of contact points on the drawing table. 40
4.6 Strokes performed with a marker. 41
4.7 Evolution of the error metric as more strokes are performed and data is collected. 42
4.8 Post-processed images of pencil strokes with $\gamma = 0°$ 43
4.9 Post-processed images of pencil strokes with $\gamma = 50°$ 44
4.10 Comparison of the error metric with standard ($\gamma = 0°$) vs our approach ($\gamma = 50°$) . 45
4.11 Close up of iteration 2 – Rotation of angled pencil allows for wider strokes. . . . 46
4.12 Close up of iteration 10 – Rotation of angled pencil allows for narrow strokes. . 46

5.1 Rendering the tool tip surface at different levels of d. 48
5.2 2D rendering of the tool tip. 49
5.3 Overhead view of Ur5e robot performing line strokes. 52

5.4 2D Rendering of the tip with new edge line. 52
5.5 Successive cuts at Iterations 2 and 3 53
5.6 Successive cuts at Iterations 5 and 10 54
5.7 Successive cuts at Iterations 20 and 25 55
5.8 Successive cuts at Iterations 50 and 100 56
5.9 Camera view from the robot performing line strokes with edge planning. 57
5.10 Camera view from the robot performing line strokes without edge planning. . . 57
5.11 Width analysis from robot performing pencil strokes. Trials A and B represent trials without edge planning. Trials C and D represent trials with edge planning. 58
5.12 Error analysis from robot performing pencil strokes. Trials A and B represent trials without edge planning. Trials C and D represent trials with edge planning. 59
5.13 Pencil strokes performed with constant force. 60
5.14 Pencil strokes performed with constant pressure. 60
5.15 Brightness analysis from robot performing pencil strokes with constant force or constant pressure. 61

List of Tables

3.1 Parameters used in control design. 25
3.2 The percentage and the average are computed numerically from 100 trials of the transportation task. 29

Acknowledgments

Here comes the best part. There are so many people in my life that I would like to thank. Everyone contributed in their own way in helping me become the person I am today.

First, I would like to thank my advisor, Francesco Borrelli. This would not have been possible without you and I truly feel so lucky to have you as a mentor. You have been patient with me and allowed me to explore my interests over the years without prejudice. I've honestly thought many times "Wow, I can't believe he's letting me do this wacky experiment." and also "Wow, I can't believe he hasn't kicked me out of the group yet." I appreciate that our bond goes beyond the classroom or laboratory (mostly into the kitchen). Your humor and personality really set the tone for the wonderful environment that you have created in the Model Predictive Control Lab. Thank you for taking a chance on me. To Professor Andy Packard, whom we all miss dearly, thank you for believing in me. I will always have your memorial frisbee hanging over my desk to remind myself that I should work hard but also play harder. I would like to give a massive thank you to my qualifying exam committee: Professor Mark Mueller, Professor Koushil Sreenath, Professor Roberto Horowitz, and Professor Anil Aswani. I also want to thank my dissertation committee: Professor Sreenath (again! big double thumbs up) and Professor Tarek Zohdi. Thank you to Madeline Augustine for mentoring me when I had no experience and guiding me to where I am today. Thank you to to the Mechanical Engineering staff, especially Yawo Akpawu and Isabel Blanco for all their help over the years. Thank you to Siemens, the National Science Foundation, and the AI Institute for Next Generation Food Systems for the funding support that made it possible for me to do research!

The people at Berkeley are truly what made it so enjoyable. I firmly believe there isn't a single lab in the entire world that could compare with the MPC lab. I cherish the free atmosphere, where we can all joke at ease and collaborate just by walking over to each other's desks to start a conversation. When any one of us was down, there was an amazing support network to lift each other back up. I would like to give a special mention to my lab mentors: Monimoy Bujarbaruah and Ugo Rosolia. Monimoy, thank you immensely for all of your guidance and wisdom. You never once looked down on me, even when I asked the dumbest questions. You have been a great collaborator and even better friend. I look forward to many more [redacted] conversations with you. Ugo, you welcomed me into the lab and treated me like family (maybe because you thought I was Italian). Thank you for helping me ease into the Ph.D. program and for all of your teachings. Thank you to Charlott Vallon, who has been here since even before day one. Our time battling through the trenches of teaching a hardware intensive lab together really solidified our friendship. Thank you to Edward Zhu for all the great times we had lounging around lab and watching more food videos while we ate. We have gone to another continent together but still no rave yet! I'm waiting... Thank you to each and every MPC member that I've had the privilege of interacting with through my very long Ph.D. journey: Xu Shen, Siddharth Nair, Tim Brudigam, Jon Gonzales, Greg Marcil, Grace Liao, Roya Firoozi, Yeojun Kim, Vijay Govindarajan, Jacopo Guanetti, George Zhang, Yvonne Stürz, Shima Nazari, Hotae Lee,

Eunhyek Joa, Thomas Fork, Hansung Kim, Fionna Kopp, Mark Pustilnik, Shengfan Cao, and Spencer Schutz. Thank you to Kate Schweidel for our walky-talkies for mental and physical health. I feel so appreciative of you and your family for basically adopting me. Thank you to Joseph Sahyoun, Jonathon Marr, Wendy Siu, and Etienne Marécal for the hilarious times. I probably would not have been able to get through the M.Eng. without you all. Thank you to Cyndia Cao, Michael Abbott, Sebastian Lee, and Oky for the amazing years living together which started with a very interesting lockdown. I think you all helped keep me sane. I love that any given moment, someone was always down to cook, play board games, get fried chicken, do some karaoke, go to a show, or many more activities. I will miss our house. Thank you to Brian Cera, Gloria Liang, Sean Anderson, Rebecca Hanna, Bike Zhang, Fernando Castaneda, Alan Zhang, Zhongyu Li, Vickie Ye, Monica Li, Laura Treers, Eric Thatcher, Yuri Murakami, Paula Chanfreut, Lace Cotingkeh, Markus Fossdal, Susan Le, Payton Goodrich, and many others for making Berkeley such a wonderful place to be. Maybe the real Ph.D. was the friends we made along the way.

Before Berkeley, there were many amazing friends that shaped who I am. Thank you to Jason Mazza and Billy Goodland who have always had my back since the days of fifth grade math club. Thank you to my best friend Erol Akkoc who I basically share one brain cell with. It has been nonstop laughter with you since that time we walked in sync at the SAC parking lot. Thank you to Joseph O'Hara, Anish Kanattu, Kevin Mulder, and Shannon Sunny for best times during undergrad. I probably could have been more productive back then but I wouldn't trade away a single moment we spent just goofing around. Thank you to Keiko Nagami, Jason Lumokso, and Eric Tola for being a part of home away from home as the California Stony Brook crew. Thank you to Dillon Martin, Mia Bodin, Natalie Twilley, Raul Riveros, Michael Biskach, Kim Allgood, and Ryan McClelland for the fun experiences at NASA which made me want to continue doing research.

Finally, the people I owe the most thanks to are my family. Thank you to my brother Jacky and sister Annie for a silly and chaotic childhood. Thank you to my bub Sharon Chen for pushing me through the finish line and being so supportive. Thank you to my grandparents who had a big hand in raising me and always protected me. Most of all, thank you to my parents who sacrificed their dreams so I could live mine. You both came to the United States, not even having finished highschool and worked 12 hours everyday at the restaurant with barely a single day off in the year. I definitely get my work ethic, sense of humor and personality from you two. I promise no more school!!! I can't wait to support you back and give you all the vacations you deserve. Everything I achieve is dedicated to you two.

Chapter 1

Introduction

1.1 Background

Robotic manipulators have significantly impacted society through the automation of repetitive and dangerous tasks in many industries such as manufacturing, agriculture, and logistics [1, 2, 3, 4]. Especially in sectors where there may be labor shortages [5, 6, 7], robotics can provide a safe and effective alternative. Expanding the capabilities of robots means increasing the complexity of their interactions with the surrounding environment which will require improvements in perception, manipulation, and planning.

As computing resources and robotic hardware continue to advance, much research has been focused on smarter planning algorithms and controllers. Model-based approaches have shown great success in their deployment on systems for tasks such as autonomous racing [8] or robotic locomotion and manipulation [9, 10]. Depending the type of task, there are a wide range of challenges arising from contact dynamics, safety concerns, hardware limitations, and more.

Take for example, the scenario of a robotic arm used for food preparation in an industrial kitchen [11]. If the goal is to automate the process of flipping burger patties on top of a griddle and then place them onto a bun when ready, the robot must first identify where the patty is relative to itself. This may require some sort of visual perception from sensors such as cameras or LIDAR. Even after the robot knows where the food is, it must plan how to move the spatula under the patty and flip without breaking anything. One solution may be to preprogram the movements of the end-effector such that it always goes to the same locations on the griddle and executes a fixed flipping maneuver that is designed through manual experiments similar to how manipulators are used for automotive manufacturing [12]. However, this means that the patty type, size, and placement should be close to what they were during the design process and any deviations may cause errors when handling. Closed-loop control could alleviate these errors but this is much more challenging control problem. When the robot is planning how to flip the burger, it must consider the hybrid dynamics of free movement in the air and the change arising due to contact with the griddle. The robot

must also be aware of surrounding obstacles to avoid external collisions. Depending on the size and weight, the robot may need to vary the forces applied to flip the patty.

One approach may be to use a purely data-driven method such as deep learning where a large number of experiments are conducted in order to obtain a dataset for training the patty flipping policy. However, a drawback is that it may be time-consuming or pose safety risks to obtain enough data through real trials. It could be trained in simulation first and then deployed in real hardware [13] but for tasks that require finer precision, there may end up being too large of a sim-to-real gap. Additionally, the lack of constraints can cause unsafe behavior when there are obstacles nearby. This is where model-based approaches can leverage their advantages. By utilizing expert knowledge to craft dynamics models, the robot can make predictions on how its behavior affects the environment. In the formulation of optimization problems, the user can specify a cost to minimize such as keeping the robot arm lower while conserving energy and also provide constraints such as avoiding the table. In this dissertation, we present model-based approaches for robot manipulators to perform various complex robotics tasks.

1.2 Outline and Contributions

This dissertation is organized as follows.

In Chapter 2, we present a model-based mixed open-loop and closed-loop control strategy for a robotic arm to play the cup-and-ball game. This is a challenging task since the robot must get the ball into the cup while dealing with the hybrid dynamics of the ball attached to the cup by a string and using noisy camera observations to estimate the ball position. First, we use a cart with inverted pendulum model to plan an open-loop trajectory which can launch the ball into the air above the cup. Then, we switch to an online closed-loop controller to catch the ball while iteratively collecting data to estimate the support of the camera noise and improve the controller performance.

In Chapter 3, we investigate the task of a robot transporting an object with a human to a desired location. The robot and human each have partial information of the environment and obstacles around but must collaborate on their shared goal. Safety is critical as the robot should not harm the human or itself during this task. We propose a Model Predictive Control (MPC) based strategy with a trust variable which allows the human and robot to dynamically switch between leader and follower roles. The robot will infer the location of obstacles known only to the human via force feedback and adjust its actions.

In Chapter 4, we propose a data-driven optimization approach for a robot to deposit material using a degradable tool. Since the tool wears away as it is used, the tip surface shape changes and this may lead to model mismatch. We present a model for the tip which accounts for degradation and uses visual feedback to iteratively update parameters. The new tip shape can be leveraged to achieve a desired deposition profile even with additional usage, thus prolonging the tool's lifespan and saving material costs. Chapter 5 explores the case where the tip wears away until the minimum deposition width for a fixed tilt angle exceeds

the desired width. We formulate a new model for the tool tip and present an algorithm which solves an optimization problem to determine how the robot should tilt the tool in order to decrease the deposition width again. We show an improvement of over 80% in width-tracking through experimental trials using a UR5e robot.

1.3 List of Publications

The results presented in this dissertation have appeared in a number of publications by the author. In particular:

- Chapter 2 is based on:
  - T. Zheng*, M Bujarbaruah*, A. Shetty, M. Sehr, and F. Borrelli, "Learning to Play Cup-and-Ball with Noisy Camera Observations". In: *IEEE International Conference on Automation Science and Engineering (CASE)*. Aug. 2020, pp. 372–377.
- Chapter 3 is based on:
  - T. Zheng*, M. Bujarbaruah*, Y. R. Stürz, and F. Borrelli. "Safe Human-Robot Collaborative Transportation via Trust-Driven Role Adaptation". In: *2023 American Control Conference (ACC)*. 2023, pp. 22–27.
- Chapter 4 is based on:
  - T. Zheng, M. Bujarbaruah, and F. Borrelli. "Data-Driven Optimization for Deposition with Degradable Tools". In: *22nd IFAC World Congress*. Vol. 56. 2. 2023, pp. 4375–4380.

Chapter 2

Learning to Play Cup-and-Ball

In this chapter, we present a learning model based control strategy for the cup-and-ball game, where a Universal Robots UR5e manipulator arm learns to catch a ball in one of the cups on a Kendama. Our control problem is divided into two sub-tasks, namely (i) swinging the ball up in a constrained motion, and (ii) catching the free-falling ball. The swing-up trajectory is computed offline, and applied in open-loop to the arm. Subsequently, a convex optimization problem is solved online during the ball's free-fall to control the manipulator and catch the ball. The controller utilizes noisy position feedback of the ball from an Intel RealSense D435 depth camera. We propose a novel iterative framework, where data is used to learn the support of the camera noise distribution iteratively in order to update the control policy. The probability of a catch with a fixed policy is computed empirically with a user specified number of roll-outs. Our design guarantees that probability of the catch increases in the limit, as the learned support nears the true support of the camera noise distribution. High-fidelity Mujoco simulations and preliminary experimental results support our theoretical analysis.

The results presented in this chapter have also appeared in:

- T. Zheng*, M Bujarbaruah*, A. Shetty, M. Sehr, and F. Borrelli, "Learning to Play Cup-and-Ball with Noisy Camera Observations". In: *IEEE International Conference on Automation Science and Engineering (CASE)*. Aug. 2020, pp. 372–377.

2.1 Introduction

Kendama is the Japanese version of the classic cup-and-ball game, which consists of a handle, a pair of cups, and a ball, which are all connected by a string. Playing the cup-and-ball game is a task commonly considered in robotics research [14, 15, 16, 17, 18, 19, 20, 21], where approaches ranging from classical PD control to reinforcement learning have been utilized to solve the task. The model-based approaches among the above typically decompose the task into two sub-tasks, namely (i) performing a swing-up of the ball when the string is taut, and (ii) catching the ball during its free-fall. The models of the joint system considered for both

sub-tasks are different, thus resulting in hybrid control design for the robotic manipulator. The key drawbacks in such existing approaches are namely the need for expert demonstrations, and the lack of guarantees of operating constraint satisfaction and obtaining catches under modeling uncertainty and sensing errors.

In this chapter, we propose a fully physics driven model-based hybrid approach for control design. The controller guarantees a constrained motion, while accounting for our best estimates of uncertainty in the system model and sensing errors. We use a mixed open-loop and closed-loop control design, motivated by works such as [22, 23, 24]. First, the swing-up phase is designed offline and then an open-loop policy is applied to the robotic manipulator. We use a cart with inverted pendulum model of the cup-and-ball joint system for swing-up policy design. For this phase, as we solve a constrained finite horizon non-convex optimization problem, we only consider a nominal disturbance-free model of the system. The swing-up trajectory is thus designed to ensure that the predicted difference in positions of the ball and the cup vanishes at a future time once the nominal terminal swing-up state is reached and the cup is held fixed.

After a swing-up, we switch to online closed-loop control synthesis once the ball starts its free-fall. We consider presence of only a camera that takes noisy measurements of the ball's position at every time step. We design the feedback controller in the manipulator's end-effector [25] space. This results in a Linear Time Invariant (LTI) model for the evolution of the difference between the cup and the ball's positions, thus allowing us to solve convex optimization problems online for control synthesis. In order to guarantee a catch by minimizing the position difference, it is also crucial to ensure that during the free-fall of the ball, the control actions to the manipulator do not yield a configuration where the string is taut, despite uncertainty in the model and noise in camera position measurements. Uncertainty in the LTI model primarily arises from low level controller mismatches in the manipulator hardware, and an upper bound of this uncertainty is assumed known. Bounds on the measurement noise induced by the camera are assumed unknown. This chapter presents a method to increase the probability of a catch, as the estimate of the support of camera measurement noise distribution is updated. Our contributions are summarized as:

- Offline, before the feedback control of the manipulator, we design a swing-up trajectory for the nominal cup-and-ball system that plans the motion of the ball to a state from which a catch control is initiated.
- Using the notion of *Confidence Support* from [26] which is guaranteed to contain the true support of the camera measurement noise with a specified probability, we use online robust feedback control for enforcing bounds on the probability of failed catches.
- With high-fidelity Mujoco simulations and preliminary physical experiments we demonstrate that the manipulator gets better at catching the ball as the support of the camera measurement noise is learned and as the Confidence Support and closed-loop policy are updated.

2.2 Related Work

In this section we review existing works in literature that solve the cup-and-ball problem. The literature can broadly be divided into two parts based on open-loop or closed-loop approach to controller design.

Offline Trajectory Planning

Miyamoto et al. [16] used successful expert demonstrations to extract way-points and design a trajectory for the end-effector. This trajectory is executed in open-loop and upon observing the performance, the way-points are accordingly updated offline to obtain a new trajectory. Sakeguchi and Miyazki [17] developed a method for parameterizing an elliptic trajectory for a two degree of freedom arm to play the Kendama game. The elliptic trajectory has spatial properties which dictate the catch position and temporal properties that influence how much dynamic energy is transferred to the ball. Once a task is executed, the parameters of the elliptic trajectory are updated based on the deviation of the ball's predicted and actual positions. Furthermore, Vollmer and Hemion [27] designed motion primitives for the manipulator from expert demonstrations which are then executed in open-loop. The performance of an executed trajectory is rated a-posteriori by an expert, and then the primitive is updated.

The major drawback in the aforementioned approaches is that they do not use any feedback of the position of the ball and the manipulator during task execution. Thus, presence of any uncertainty in the considered models is ignored.

Online Trajectory Planning

Nemec et al. [28] proposed an approach based on a combination of reinforcement learning and imitation learning. The swing-up trajectory is obtained using their State–Action–Reward–State–Action (SARSA) reinforcement learning algorithm with a goal of swinging the ball to a desired angle and angular velocity. The catch is then obtained with a closed-loop policy which is a function of the ball's measured position. This policy imitates an expert. Namiki and Itoi [18] designed only the catching controller assuming a free-falling ball. Given the position and velocity of the ball, the desired path of the manipulator position is planned at every time step using polynomial splines. Schwab et al. [19] presented a reinforcement learning method using purely vision (raw images, pixels, and features) information of the ball. Kober and Peters [15] also presented a reinforcement learning method which uses the ball's position information from a Vicon [29] system. Their algorithm is warm-started with expert demonstrations.

The key drawbacks in such approaches are (i) the need for expert demonstrations, and (ii) no guarantees of operating constraint satisfaction and obtaining catches under modeling uncertainty and sensing errors.

The novelties that our approach brings over the works in Section 2.2 and Section 2.2 are primarily:

- We propose a fully physics driven model-based hybrid approach for control design. The controller guarantees a constrained motion, while accounting for our best estimates of uncertainty in the system model and noise in the camera measurements.
- Our framework allows for failures of the catching task as we repeat. Using data, we update our estimate of the camera noise iteratively. The feedback control policy is thus updated, in order to improve the probability of catches during experiments.

2.3 Generating A Swing-up Trajectory

The swing-up phase begins with the arm in the home position such that the ball is hanging down at an angle of 0 radians from the vertical plumb line, as seen in Fig. 2.1.

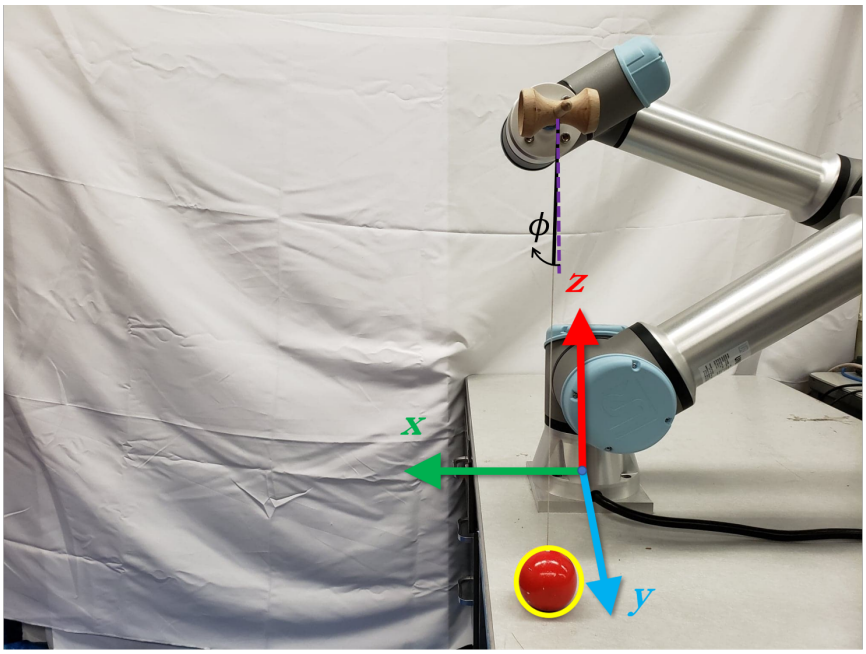


Figure 2.1: Manipulator with Kendama along with coordinate frame.

System Modeling

We model the system such that the cup is a planar cart with point-mass m_c and the ball acts as a rigid pendulum (mass m_b and radius r) attached to the cup. Assuming planar xz-motion of the ball, we derive the Lagrange equations of motion [25] with three generalized coordinates $\mathbf{q}(t) = (x^{\text{cup}}(t), z^{\text{cup}}(t), \phi(t))$, which denote the x position of the cup, z position of the cup, and swing angle of the ball with respect to the plumb line of the cup respectively

at any time $t \geq 0$. We reduce the equations to the general nominal form

$$M(\mathbf{q}(t))\ddot{\mathbf{q}}(t) + C(\mathbf{q}(t), \dot{\mathbf{q}}(t))\dot{\mathbf{q}}(t) + G(\mathbf{q}(t)) = F(t),\ \forall t \geq 0, \tag{2.1}$$

where $M(\mathbf{q}(t))$ is the inertia matrix, $C(\mathbf{q}(t), \dot{\mathbf{q}}(t))$ is the Coriolis matrix, $G(\mathbf{q}(t))$ is the gravity matrix, and $F(t)$ is the external input force at time t. Here $\dot{\mathbf{q}}(t)$ denotes the velocity of the cup and the angular velocity of the ball, and $\ddot{\mathbf{q}}(t)$ denotes the acceleration of the cup and the angular acceleration of the ball at any time $t \geq 0$. System (2.1) in state-space form is

$$\dot{\bar{x}}(t) = f(\bar{x}(t), F(t)), \tag{2.2}$$

where nominal state $\bar{x}(t) = [\mathbf{q}^\top(t), \dot{\mathbf{q}}^\top(t)]^\top \in \mathbb{R}^6$ for all time $t \geq 0$.

Optimization Problem

We discretize system (2.2) with one step Euler discretization and a sampling time of $T_s =$ 100Hz. The discrete time system can then be written as

$$\bar{x}_{i+1} = \bar{x}_i + T_s f(\bar{x}_i, F_i) = f_d(\bar{x}_i, F_i),\ \forall i \in \{0, 1, \dots\},$$

where a_i denotes the sampled time version of continuous variable $a(t)$. To generate a force input sequence for the swing-up, we solve a constrained optimal control problem over a finite planning horizon of length N, given by:

$$\begin{aligned} \min_{F_0,\dots,F_{N-1}} \quad & \sum_{i=0}^{N-1} \bar{x}_i^\top Q_s \bar{x}_i + F_i^\top R_s F_i \\ \text{s.t.,} \quad & \bar{x}_{i+1} = f_d(\bar{x}_i, F_i), \\ & \bar{x}_i \in \mathcal{X}, F_i \in \mathcal{F}, \\ & \bar{x}_0 = x_{\text{init}}, \\ & \bar{x}_N = x_{\text{f}},\ i = 0, 1, \dots, (N-1), \end{aligned} \tag{2.3}$$

where weight matrices $Q_s, R_s \succ 0$, and constraint set $\mathcal{X}$ is chosen such that the ball remains within the reach of the UR5e manipulator. Initial state x_{init} is known in the configuration as shown in Fig. 2.1. Due to the nonlinear dynamics $f_d(\cdot, \cdot)$, the optimization problem (2.3) is non-convex. Moreover, typically a long horizon length N is required. Hence, we solve (2.3) offline and apply the computed input sequence $\mathbf{F}^\star = [F_0^\star, F_1^\star, \dots, F_{N-1}^\star]$ in open-loop to the manipulator.

Terminal Conditions of the Swing-Up

Predicted Behaviour

The nominal terminal state x_{f} in (2.3) is selected such that the ball is swinging to $\phi = 2.44$ rad with an angular velocity of $\dot{\phi} = 4.18$ rad/s. At these values, the string is calculated to lose tension and the ball begins free-fall. The chosen value of x_{f} ensures that the predicted difference in positions of the ball and the cup (both modeled as point masses) vanishes at a future time, if the cup were held fixed and the ball's motion is predicted under free-fall.

Actual Behaviour

When considering the nominal system (2.1), we have ignored the presence of uncertainties. Such uncertainties may arise due to our simplifying assumptions such as: (i) the string is mass-less so the swing angle is only affected by the ball and cup masses, (ii) there are no frictional and aerodynamic drag forces to hinder the conservation of kinetic and potential energy of the system, (iii) the cup mass is decoupled from the mass of the manipulator, and (iv) there is no mismatch of control commands from the low level controller of the manipulator and F. Due to such uncertainties, realized states x_i for $i \in \{0, 1, \ldots, N\}$ do not exactly match their nominal counterparts.

A set of 100 measured roll-out trajectories of the ball after the swing-up are shown in Fig. 2.2 for a fixed open-loop input sequence $\mathbf{F}^\star$. We see from Fig. 2.2 that after N time

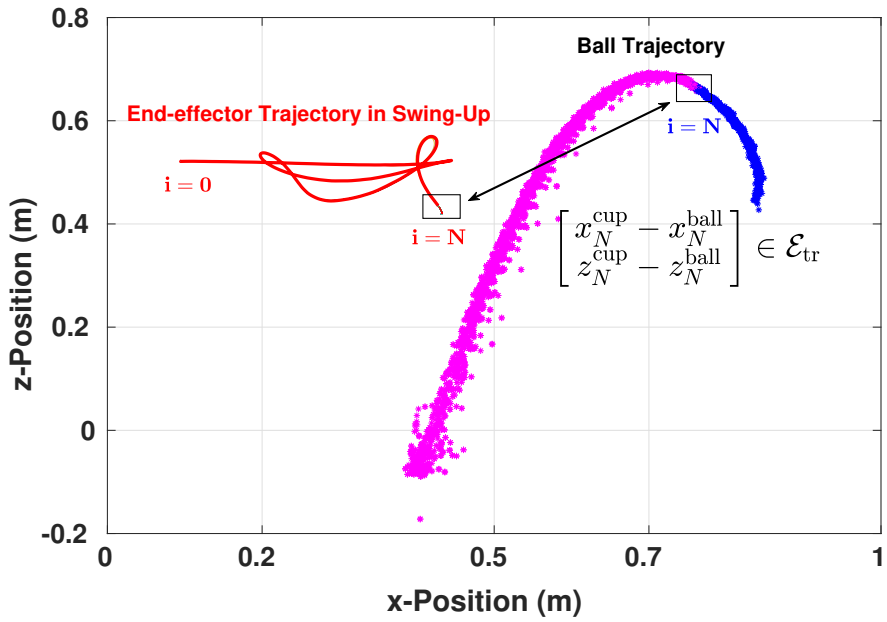


Figure 2.2: Start of catch phase (i.e., $i = N$) for 100 trajectories. Red line indicates the trajectory of the cup/end-effector during swing-up. Blue dots indicate ball positions during swing-up and pink dots indicate a position after catch phase is started. Closed-loop control begins when the relative position is in $\mathcal{E}_{\text{tr}}$.

steps of swing-up, the ball and the cup arrive at positions where their relative position is in a set $\mathcal{E}_{\text{tr}}$. A key assumption of well posedness will be imposed on this set in Section 2.4 in order for our subsequent feedback control policy to deliver a catch in experiments.

2.4 Designing Feedback Policy In Catch Phase

For the catch phase we start the time index $t = 0$ where the swing up ends, i.e., $i = N$. There are two main challenges during the design of the feedback controller, namely (i) position measurements of the ball from a noisy camera, and (ii) presence of mismatch between desired control actions and corresponding low level controller commands.

Assumption 2.1 *We assume that the UR5e end-effector gives an accurate estimate of its own position. The assumption is based on precision ranges provided in [30].*

Problem Formulation

During free-fall of the ball we design our feedback controller for the manipulator position only in end-effector space, with desired velocity of the end-effector as our control input. The joint ball and end-effector system in one trial can be modeled as a single integrator as:

$$e_{t+1} = Ae_t + Bu_t + w_t(e_t, u_t), \tag{2.4a}$$
$$y_t = e_t + v_t, \tag{2.4b}$$

with error states and inputs (i.e., relative position and velocity)

$$e_t = \begin{bmatrix} x_t^{\text{cup}} - x_t^{\text{ball}} \\ z_t^{\text{cup}} - z_t^{\text{ball}} \end{bmatrix}, \; u_t = \begin{bmatrix} v_{x,t}^{\text{cup}} - v_{x,t}^{\text{ball}} \\ v_{z,t}^{\text{cup}} - v_{z,t}^{\text{ball}} \end{bmatrix},$$

where $w_t(e_t, u_t) \in \mathbb{W}_m \subset \mathbb{R}^2$, is a bounded uncertainty which arises due to the discrepancy between (i) the predicted and the actual velocity of the ball at any given time step[1], and (ii) the commanded and the realized velocities of the end-effector, primarily due to the low level controller delays and limitations. System dynamics matrices $A = I_2$ and $B = dt \cdot I_2$ are known, where I_d denotes the identity matrix of size d, and sampling time $dt = 0.01$ second. We assume an outer approximation $\mathbb{W}$ to the set $\mathbb{W}_m$, i.e., $\mathbb{W}_m \subseteq \mathbb{W}$ is known, and is a polytope. We consider noisy measurements of states due to the noise in camera position measurements, corrupted by $v_t \overset{\text{i.i.d.}}{\sim} \mathcal{P}$, with $\text{Supp}(\mathcal{P}) = \mathbb{V}$, where $\text{Supp}(\cdot)$ denotes the support of a distribution. We assume $\mathbb{V}$ is not exactly known.

Using the set $\mathcal{E}_{\text{tr}}$ (see Fig. 2.2), a set $\mathcal{E}$ containing the origin where the string is not taut and (2.4) is valid can then be chosen. We choose:

$$\begin{aligned} &\gamma^{(i)} = \|\text{vert}^{(i)}(\mathcal{E}_{\text{tr}})\|_\infty, \; i \in \{1, 2\}, \\ &\mathcal{E} = \{x : -\gamma \leq x \leq \gamma\}, \; \gamma = [\gamma^{(1)}, \gamma^{(2)}]^\top, \end{aligned} \tag{2.5}$$

where $\text{vert}^{(i)}(\mathcal{A})$ denotes i^{th} row of all the vertices of the polytope $\mathcal{A}$, and $\|\cdot\|$ denotes the vector norm. This ensures

$$e_0 \in \mathcal{E}_{\text{tr}} \implies e_0 \in \mathcal{E}. \tag{2.6}$$

[1] we use the camera position information for ball's velocity estimation

As (2.6) holds true, we impose state and input constraints for all time steps $t \geq 0$ as given by:

$$e_t \in \mathcal{E},\ u_t \in \mathcal{U}, \tag{2.7}$$

where set $\mathcal{U}$ is a polytope. We formulate the following finite horizon robust optimal control problem for feedback control design:

$$\begin{aligned}
\min_{u_0, u_1(\cdot), \ldots} \quad & \sum_{t=0}^{T-1} \ell\left(\bar{e}_t, u_t\left(\bar{e}_t\right)\right) + Q(\bar{e}_T) \\
\text{s.t.,} \quad & e_{t+1} = Ae_t + Bu_t(e_t) + w_t(e_t, u_t), \\
& \bar{e}_{t+1} = A\bar{e}_t + Bu_t(\bar{e}_t), \\
& y_t = e_t + v_t, \\
& e_t \in \mathcal{E}, u_t(e_t) \in \mathcal{U}, \\
& \forall w_t(e_t, u_t) \in \mathbb{W},\ \forall v_t \in \mathbb{V}, \\
& e_0 \in \mathcal{E},\ t = 0, 1, \ldots, (T-1),
\end{aligned} \tag{2.8}$$

where e_t, u_t and $w_t(e_t, u_t)$ denote the realized system state, control input and model uncertainty at time step t respectively, and $(\bar{e}_t, u_t(\bar{e}_t))$ denote the nominal state and corresponding nominal input. Notice that (2.8) minimizes the nominal cost over a task duration of length T decided by the user, having considered the safety restrictions during an experiment. The cost comprises of the positive definite stage cost $\ell(\cdot, \cdot)$, and the terminal cost $Q(\cdot)$. We point out that, as system (2.4) is uncertain, the optimal control problem (2.8) consists of finding $[u_0, u_1(\cdot), u_2(\cdot), \ldots]$, where $u_t : \mathbb{R}^2 \ni x_t \mapsto u_t = u_t(e_t) \in \mathbb{R}^2$ are state feedback policies.

The main challenge in solving problem (2.8) is that it is difficult to obtain the camera measurement noise distribution support $\mathbb{V}$. Resorting to worst-case a-priori set estimates of $\mathbb{V}$ as in [31, 32] might result in loss of feasibility of (2.8). To avoid this, we use a data-driven estimate of $\mathbb{V}$ denoted by $\hat{\mathbb{V}}(n)$, where n is the number of samples of noise v_t used to construct the set. Details of how we generate $\hat{\mathbb{V}}(n)$ and deal with noisy output feedback can be found in [33]. We then solve the following tractable finite horizon constrained optimal control problem at any time step $t \geq 0$ as an approximation to (2.8):

$$\begin{aligned}
& V^{\star}_{t \to T}(\bar{\mathcal{E}}(n), \bar{\mathcal{U}}(n), \mathcal{R}^{\text{con}}(n), \hat{e}_t) := \\
& \min_{\bar{e}_t, \bar{u}_t, \ldots, \bar{u}_{T-1}} \quad \sum_{k=t}^{T-1} \ell(\bar{e}_k, \bar{u}_k) + Q(\bar{e}_T) \\
& \text{s.t.,} \qquad \bar{e}_{k+1} = A\bar{e}_k + B\bar{u}_k, \\
& \qquad\quad u_k = \bar{u}_k + K(\hat{e}_k - \bar{e}_k), \\
& \qquad\quad \bar{e}_k \in \bar{\mathcal{E}}(n), \bar{u}_k \in \bar{\mathcal{U}}(n), \\
& \qquad\quad \hat{e}_t - \bar{e}_t \in \mathcal{R}^{\text{con}}(n), \\
& \qquad\quad \bar{e}_T = 0, \\
& \qquad\quad \forall k \in \{t, t+1, \ldots, (T-1)\},
\end{aligned} \tag{2.9}$$

where $\hat{e}_t$ is the observed state at time step t, $\{\bar{e}_k, \bar{u}_k\}$ denote the nominal state and corresponding input respectively predicted at time step $k \geq t$, K is a state feedback policy gain matrix, and $\mathcal{R}^{\text{con}}(n)$ is our best estimate of the minimal Robust Positive Invariant set $\mathcal{R}^{\text{con}}$ for the control error.

Obtaining Catches

Constructing $\hat{\mathbb{V}}(n)$ to ensure the probability of violating the state constraints remains under a specific threshold is still not a sufficient condition to obtain a catch in an experiment with specified probability β, as our model (2.4) does not account for additional factors such as object dimensions, presence of contact forces, etc.

To that regard, we introduce the notion of a *successful catch*, which is defined as the ball successfully ending up inside the cup at the end of a roll-out. Thus, a successful catch accounts for the dimensions of the ball and the cup, and the presence of contact forces.

Assumption 2.2 (Existence of a Successful Catch) *We assume that given an initial state $e_0 \in \mathcal{E}_{\text{tr}}$, an input policy obtained by solving* (2.9) *can yield a successful catch, if true measurement noise support $\mathbb{V}$ were known exactly.*

Remark 2.1 *From [26] we know that as long as confidence intervals for parameters (μ, σ) in noise distribution converge, $\hat{\mathbb{V}}(n) \to \mathbb{V}$ as $n \to \infty$. So, if sample size n is increased iteratively approaching $n \to \infty$, obtaining a successful catch guaranteed owing to Assumption 2.2. However if a precise positioning system like Vicon is used to collect the noise samples, due to limited access to such environments, collecting more samples and increasing n could be expensive. We therefore stick to our method of constructing $\hat{\mathbb{V}}(n)$ for a fixed n, and we attempt successful catches with multiple roll-outs by solving* (2.9)*. For improving the empirical probability of successful catches in these roll-outs, one may then increase n and thus update the control policy. We demonstrate this in Section 2.5.*

2.5 Experimental Results

We present our preliminary experimental findings in this section. For our experiments, the original Kendama handle was modified to be attached to a 3D printed mount on the UR5e end-effector, as shown in Fig. 2.1. The cup used to catch the ball is unmodified which makes this task much more challenging as the ball can easily bounce out. A single Intel RealSense D435 depth camera running at 60 FPS was used to estimate the position and velocity of the ball as shown in 2.3. For these experiments, a red color filter was applied to find the ball within the camera image. The depth camera provides 3D coordinates of the ball which was calibrated with a sequence of movements of the end-effector while the ball was sitting in the cup. This was used to determine the ball's position in the UR5e's base frame. Between experiments, a resetting maneuver was executed in order to untangle the string and ensure that the ball was at rest in the air.

Figure 2.3: Video feed from the depth camera during the catch phase of the experiment.

Control Design in the Catch Phase

Once the swing-up controller is designed as per Section 2.3 and an open-loop swing-up control sequence is applied to the manipulator, we design the feedback controller by finding approximate solutions to the following problem:

$$\begin{aligned}
\min_{u_0,u_1(\cdot),\ldots} \; & \sum_{t=0}^{T-1} 500\,\|\bar{e}_t\|_2^2 + 0.4\,\|u_t(\bar{e}_t)\|_2^2 \\
\text{s.t.,} \quad & \\
& e_{t+1} = Ae_t + Bu_t(e_t), \\
& \bar{e}_{t+1} = A\bar{e}_t + Bu_t(\bar{e}_t), \\
& y_t = e_t + v_t, \\
& e_t \in \mathcal{E},\ \begin{bmatrix} -8\text{m/s} \\ -8\text{m/s} \end{bmatrix} \leq u_t(e_t) \leq \begin{bmatrix} 8\text{m/s} \\ 8\text{m/s} \end{bmatrix}, \\
& \forall v_t \in \mathbb{V}, \\
& t = 0, 1, \ldots, (T-1),
\end{aligned} \tag{2.10}$$

where set $\mathcal{E}_{\text{tr}} = [-0.316\text{m}, 0.349\text{m}] \times [-0.2095\text{m}, 0.2457\text{m}]$, shown in Fig. 2.2. Note that for this specific scenario the presence of model uncertainty can be ignored. Set $\mathbb{V}$ is unknown,

and we consider that the optimization problem is well posed. System matrices A, B are from Section 2.4. We find solutions to (2.10) for $T = 50$ steps, i.e., 0.5 seconds.

Learning to Catch

We conduct 50 roll-outs of the catching task by solving (2.9), having formed $\hat{\mathbb{V}}(n)$, with $n = 100$ and then iteratively increasing to $n = 2000$. Sets $\hat{\mathbb{V}}(n)$ are formed using [26]. Fig. 2.4 shows the percentage of roll-outs conducted for each iteration (i.e., for each value of n), that resulted in the ball successfully striking the center of the cup.

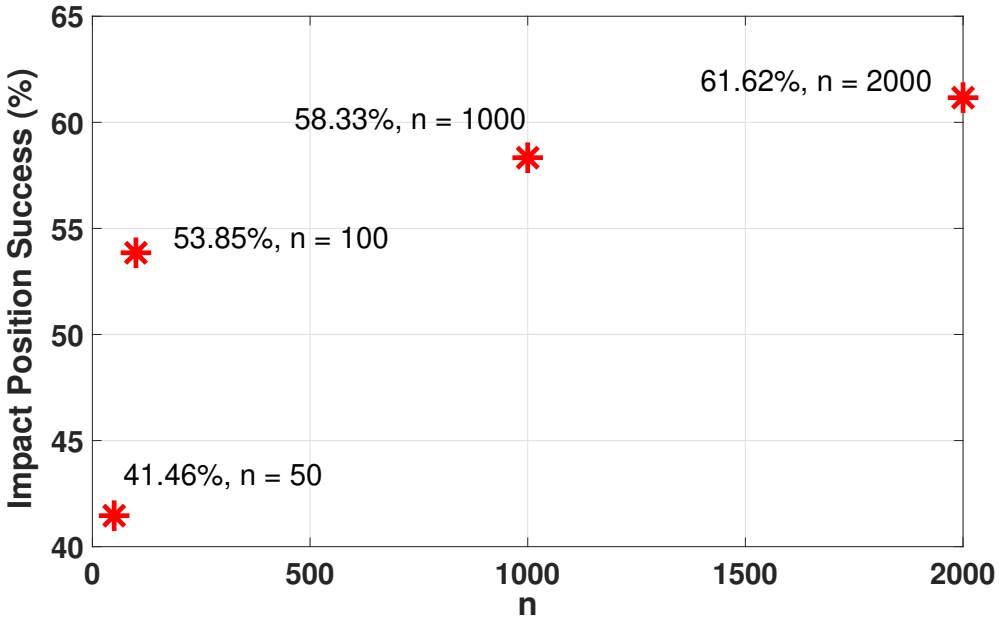


Figure 2.4: Percentage of times the ball hitting the cup center among all roll-outs vs sample size n.

The percentage increases from 41.46% to 61.62%. Furthermore, another crucial quantity at the time of impact is the commanded relative velocity in z-direction, a lower value of which indicates an increased likelihood of the ball not bouncing out. The average value and the standard deviation of of $(u^{\star}_{T_{\text{im}}-1})^{*\tilde{m}}_{z}$ for $\tilde{m} \in \{1, 2, \ldots, 50\}$ is shown in Fig. 2.5, where $(\cdot)^{*\tilde{m}}$ denotes the $\tilde{m}^{\text{th}}$ roll-out and $T_{\text{im}} \leq T$ denotes the time of impact. As seen in Fig. 2.5, the mean of the relative velocity at impact lowers from 0.38 m/s to -0.06 m/s. This together with Fig. 2.4 indicates a possibility of increasing successful catch counts as n is increased. Something we observed from unsuccessful trials is that the ball often bounced out due to the configuration of the cup itself being a shallow groove which only cradles the bottom of the ball. Contact dynamics were not modeled in our control formulation and is a likely source of error.

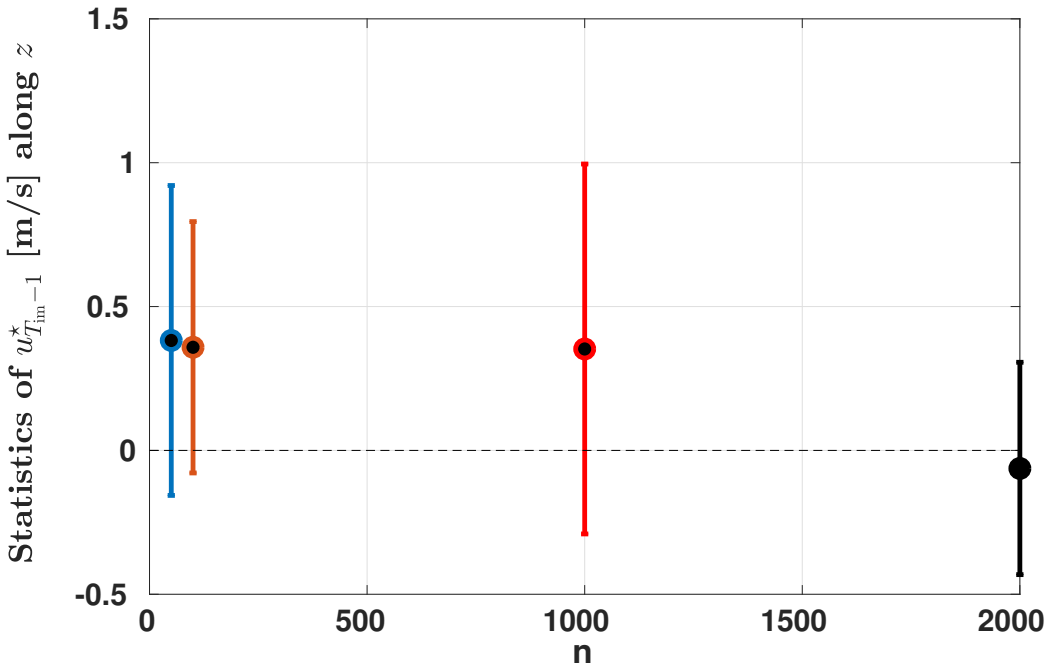


Figure 2.5: One standard deviation interval around the mean (circle) of z-relative velocity at impact, i.e., $[u^{\star}_{T_{\text{im}}-1}]_z$ vs sample size n.

Increasing Successful Catches

In order to prove that the trend shown in Fig. 2.4 and Fig. 2.5 results in an increasing number of successful catches, we resort to exhaustive Mujoco [34, 35] simulations. The task duration in this case is $T = 25$ steps. The trend in the percentage of successful catches with 1000 roll-outs corresponding to each n, varying from $n = 50$ to $n = 2000$, is shown in Fig. 2.6. For $n = 50$, 46.9% of the roll-outs result in a successful catch. The number increases to 68.3% for $n = 2000$. This verifies that the preliminary experimental results from Fig. 2.4 and Fig. 2.5 would very likely result in a similar trend as in Fig. 2.6. Thus we prove that our proposed approach enables successful learning of the kendama ball catching task.

2.6 Conclusions

We proposed a model based control strategy for the classic cup-and-ball game. The controller utilized noisy position measurements of the ball from a camera, and the support of this noise distribution was iteratively learned from data. Thus, the closed-loop control policy iteratively updates. We proved that the probability of a catch increases in the limit, as the learned support nears the true support of the camera noise distribution. Preliminary experimental results and high-fidelity simulations support our analysis.

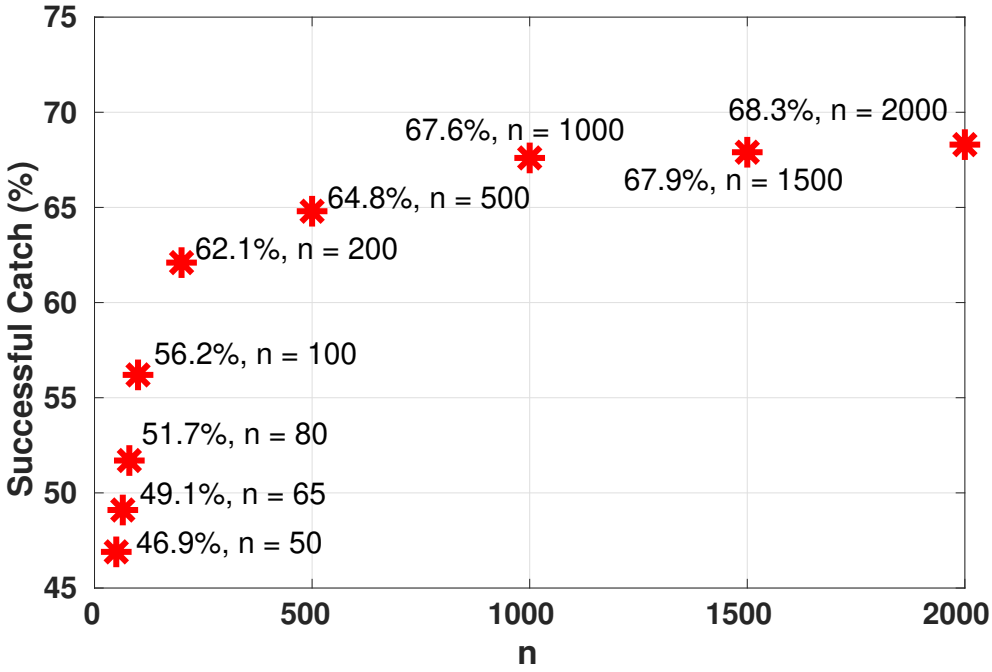


Figure 2.6: Percentage of successful catches vs sample size n.

Chapter 3

Human-Robot Collaborative Transportation

We study a human-robot collaborative transportation task in presence of obstacles. The task for each agent is to carry a rigid object to a common target position, while safely avoiding obstacles and satisfying the compliance and actuation constraints of the other agent. Human and robot do not share the local view of the environment. The human either assists the robot when they deem the robot actions safe based on their perception of the environment, or actively leads the task.

Using estimated human inputs, the robot plans a trajectory for the transported object by solving a constrained finite time optimal control problem. Sensors on the robot measure the inputs applied by the human. The robot then appropriately applies a weighted combination of the human's applied and its own planned inputs, where the weights are chosen based on the robot's *trust value* on its estimates of the human's inputs. This allows for a dynamic leader-follower role adaptation of the robot throughout the task. Furthermore, under a low value of trust, if the robot approaches any obstacle potentially unknown to the human, it triggers a safe stopping policy, maintaining safety of the system and signaling a required change in the human's intent. The robot also uses the sensor feedback to infer obstacles known only by the human and updates its planner to better align with the human's movements. With experimental results, we demonstrate that our proposed approach increases the success rate of collision-free trials while decreasing the effort required by the human to intervene.

The results presented in this chapter have also appeared in:

- T. Zheng*, M. Bujarbaruah*, Y. R. Stürz, and F. Borrelli. "Safe Human-Robot Collaborative Transportation via Trust-Driven Role Adaptation". In: *2023 American Control Conference (ACC)*. 2023, pp. 22–27.

3.1 Introduction

Human robot collaborative tasks have been a focus of major research work in robotics [36, 37]. For such tasks, roles of the agents are important, especially so in collaborative transportation. This is due to the fact that the transported object poses a compliance constraint that must be satisfied. The robot acts as a follower or helper to the human in [38, 39, 40]. In these works, the human knows the full environment and is the lead planner in the task. The robot follows the human by minimizing its felt forces and torques, and has no planning algorithms of its own. However, such fixed role assignment can be debilitating in situations when both agents have partial environment information, or if the human wants to lower their efforts in the task. In [41], they study how intelligent and safe human collaborators perceive a robot to be when the robot assumes a leader, follower, or non-collaborative role in a shared control ball-tilt maze game. Shared and/or switching roles can be used to produce better teamwork especially in situations where one agent may have a more advantageous position or greater access to local information. A shared role or blended policy often utilizes an input-scaling parameter that is based on a heuristic such as the robot's confidence in its prediction of the human partner's goals [42] or it could simply be time-varying [43]. Dynamic role switching using force sensor data in collaborative manipulation tasks between humans and robots have been studied in [44, 45, 46]. In such switching role assignments, it is essential for the robot to make predictions of the human's intent from the human's observed behavior and then adapt its policy accordingly during the task. Models for human intention can be estimated in a variety of ways including learning from motion data [47, 48] or force interactions [49, 50]. These models inherently contain some information about the human's reactions towards obstacles in the environment. Obstacle avoidance in human-robot collaborative tasks where the obstacle positions were estimated from depth camera images was studied in [51, 52]. However, to the best of our knowledge, inferring the positions of unknown obstacles in the environment from haptic feedback data and then explicitly incorporating the obstacle avoidance constraints in the collaborative robot's planning problem have not been addressed.

In this chapter, we propose a Model Predictive Control (MPC) based strategy for a human-robot joint transportation task, as shown in Fig. 4.1. The environment has obstacles partially known to each agent. The human's policy is allowed to be a combination of compliance and leadership, based on the human's intent during the task. The robot only estimates the compliant human behavior, and operates on a policy based on a computed *trust value* and also its proximity to obstacles. This allows for a dynamic leader-follower role of the robot throughout the task, depending on the learned value of trust from applied human inputs. The trust is low if the actual human inputs differ highly from the robot's estimates, and vice versa. The robot also infers the locations of obstacles only known prior by the human through force feedback to update its planner throughout the task. Our proposed framework can be summarized as:

- We design a two mode policy for the robot. The first mode is the nominal operation mode, where the robot solves an MPC problem for its control synthesis. The cost

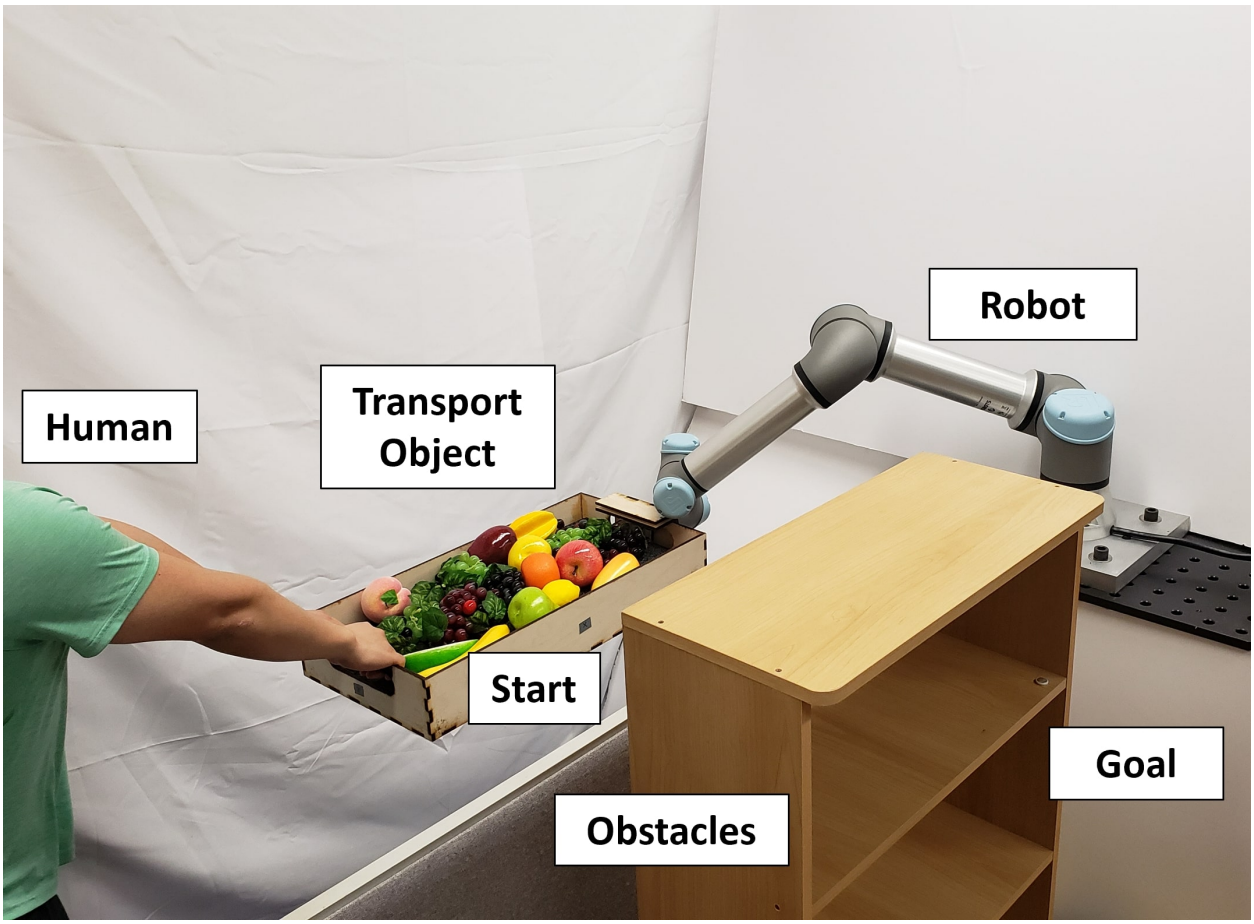


Figure 3.1: The considered experiment setup.

function in the MPC optimization problem adapts based on the corrective inputs of the human to the robot's inputs and inferred obstacles zones that the human may be avoiding. This enables the robot to plan trajectories that adapt with the human's behavior.

- The control applied by the robot in the first mode is a function of the *trust value*, similar to [42] and [45]. That is, after solving the MPC problem, the robot appropriately applies a weighted combination of the human's and its own planned actions, where the weights are adapted based on the deviation between robot's estimated and the actual human inputs.

- The second mode of the robot's policy is a safe stopping backup, which is triggered when the robot nears obstacles under a low value of trust on its estimated human's inputs. This safety mode enables the robot to decelerate the object, avoid collisions, and signal a required change in intent to the human via haptic feedback.

We highlight that the robot obtains a follower's role for low trust value, including safe stopping backup. On the other hand, it asserts a leader's role for high trust value, relying more on its MPC planned inputs. These leader-follower roles switch dynamically throughout the task as a function of the trust value. In Section 3.4, with experiments on a UR5e robot, we demonstrate the efficacy of our proposed approach. We present an experiment where with pre-assigned fixed roles the agents collide with obstacles, whereas a combination of trust-driven and safe stop policies completes the task safely.

3.2 Problem Formulation

In this section, we formulate the collaborative obstacle avoidance problem. We restrict ourselves to the case of two agents. The case of collaborative transportation with multiple agents is left as a subject of future research.

Environment Modeling

Let the environment be contained within a set $\mathcal{X}$. In this work, we assume that the obstacles in the environment are static, although the proposed framework can be extended to dynamic obstacles. At any time step t, let the set of obstacle constraints known to the human and the robot (detected at t and stored until t) be denoted by $\mathcal{C}_{h,t}$ and $\mathcal{C}_{r,t}$, respectively. We denote:

$$\mathcal{C}_{r,t} \cup \mathcal{C}_{h,t} = \mathcal{O}_t, \ \forall t \leq T,$$

where $T \gg 0$ is the task duration limit and $\mathcal{O}_t$ is the set of obstacle constraints to be avoided at t during the task. The approach proposed in this chapter focuses on the challenging situation where no agent has the full information of all the detected obstacles in $\mathcal{O}_t$, i.e., $\mathcal{C}_{h,t} \subset \mathcal{O}_t$ and $\mathcal{C}_{r,t} \subset \mathcal{O}_t$.

System Modeling

We model both the human and the robot transporting a three dimensional rigid object. Let $(\vec{I}_I, \vec{J}_I, \vec{K}_I)$ and $(\vec{I}_B, \vec{J}_B, \vec{K}_B)$ be the orthogonal unit bases vectors defining the inertial and the transported object fixed coordinate frames, respectively. Let (X, Y, Z) be the position of the center of mass of the transported object in the inertial frame, $\vec{v}$ be the velocity of the center of mass relative to the inertial frame, expressed in the body-frame as

$$\vec{v} = v_x \vec{I}_B + v_y \vec{J}_B + v_z \vec{K}_B. \tag{3.1}$$

Furthermore, let the Euler angles $E = \begin{bmatrix} \psi & \theta & \phi \end{bmatrix}^\top$ be the roll, pitch, yaw angles describing the orientation of the body w.r.t. the inertial frame, and $\vec{\omega}_{B/\mathbb{I}}$ be the angular velocity of the body-fixed frame w.r.t. the inertial frame, expressed in the body-fixed frame as

$$\vec{w}_{B/\mathbb{I}} = w_x \vec{I}_B + \omega_y \vec{J}_B + \omega_z \vec{K}_B. \tag{3.2}$$

We denote $\dot{E} = W^{-1}\begin{bmatrix}\omega_x & \omega_y & \omega_z\end{bmatrix}^\top$, with matrix

$$W^{-1} = \frac{1}{\cos\theta}\begin{bmatrix} 0 & \sin\phi & \cos\phi \\ 0 & \cos\phi\cos\theta & -\sin\phi\cos\theta \\ \cos\theta & \sin\phi\sin\theta & \cos\phi\sin\theta \end{bmatrix}.$$

Let (F_x, F_y, F_z) be the force components along the inertial axes applied at the body's center of mass, (τ_x, τ_y, τ_z) are the torques about the body fixed axes, and J be the moment of inertia of the body expressed in the body frame, given by $J = \text{diag}(J_x, J_y, J_z)$. Then the rigid body dynamics of the object transported are written as follows [53]:

$$\begin{aligned}
\begin{bmatrix}\dot{X} & \dot{Y} & \dot{Z}\end{bmatrix}^\top &= Q_{B/\mathbb{I}}\begin{bmatrix}v_x & v_y & v_z\end{bmatrix}^\top, \\
\begin{bmatrix}\dot{\psi} & \dot{\theta} & \dot{\phi}\end{bmatrix}^\top &= W^{-1}\begin{bmatrix}\omega_x & \omega_y & \omega_z\end{bmatrix}^\top, \\
\begin{bmatrix}\dot{v}_x & \dot{v}_y & \dot{v}_z\end{bmatrix}^\top &= \frac{1}{M}\begin{bmatrix}F_x & F_y & F_z\end{bmatrix}^\top - \Omega\begin{bmatrix}v_x & v_y & v_z\end{bmatrix}^\top, \\
\begin{bmatrix}\dot{\omega}_x & \dot{\omega}_y & \dot{\omega}_z\end{bmatrix}^\top &= J^{-1}\begin{bmatrix}\tau_x & \tau_y & \tau_z\end{bmatrix}^\top \\
&\quad - J^{-1}\Omega J\begin{bmatrix}\omega_x & \omega_y & \omega_z\end{bmatrix}^\top,
\end{aligned} \tag{3.3}$$

with M being the mass of the body and the angular velocity and rotation matrices given by

$$\Omega = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}, \text{ and}$$

$$Q_{B/\mathbb{I}} = \begin{bmatrix} c\theta c\phi & c\phi s\theta s\phi - c\phi s\psi & s\phi s\psi + c\phi c\psi s\theta \\ c\theta s\psi & c\phi c\psi + s\theta s\phi s\psi & c\phi s\theta s\psi - c\psi s\phi \\ s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix},$$

respectively, where sin and cos have been abbreviated. The corresponding state-space equation for the transported object is compactly written as:

$$\dot{S}(t) = f_c(S(t), u(t)), \tag{3.4}$$

with states and inputs at time t given by:

$$\begin{aligned}
S(t) &= [X(t), Y(t), Z(t), \psi(t), \theta(t), \phi(t), v_x(t), v_y(t), v_z(t), \\
&\qquad \omega_x(t), \omega_y(t), \omega_z(t)]^\top, \\
u(t) &= [F_x(t), F_y(t), F_z(t), \tau_x(t), \tau_y(t), \tau_z(t)]^\top.
\end{aligned}$$

We use the forward Euler method to discretize (3.4) with the sampling time of T_s of the robot to obtain its discrete time version:

$$S_{t+T_s} = f(S_t, u_t). \tag{3.5}$$

Given any input u_t to the center of mass of the object, we decouple it into the corresponding human inputs u_t^h and robot inputs u_t^r, such that $u_t = u_t^h + u_t^r$. We consider constraints on the inputs of the robot and the human given by $u_t^h \in \mathcal{U}^h$ and $u_t^r \in \mathcal{U}^r$ for all $t \geq 0$. The set $\mathcal{U}^h$ can be learned from human demonstrations' data.

3.3 Robot's Policy Design

We detail the steps involved in control synthesis by the robot in this section. The robot computes the net (i.e., from both the human and the robot) optimal forces and torques to be applied to the center of mass of the transported body by solving a constrained finite time optimal control problem in a receding horizon fashion. The robot's portion of those net optimal inputs are affected by its proximity to obstacles potentially unknown to the human and an estimate of the human's assisting input. We elaborate these steps next.

MPC Planner and Human's Inputs Estimation

The constrained finite time optimal control problem that the robot solves at time step t with a horizon of $N \ll T$ is given by:

$$\begin{aligned}
\min_{U_t} \sum_{k=1}^{N} [&(S_{t+kT_s|t} - S_{\text{tar}})^\top Q_s (S_{t+kT_s|t} - S_{\text{tar}}) + \cdots \\
&+ u_{t+(k-1)T_s|t}^\top Q_i u_{t+(k-1)T_s|t}] + \mathcal{I}_{\mathcal{O}}(S_t, u^h_{t-T_s}) \\
\text{s.t.,} \quad & S_{t+kT_s|t} = f(S_{t+(k-1)T_s|t}, u_{t+(k-1)T_s|t}), \\
& \mathcal{B}(S_{t+kT_s|t}) \in \mathcal{X} \setminus \mathcal{C}_{r,t}, \\
& u_{t+(k-1)T_s|t} \in \mathcal{U}^r \oplus \mathcal{U}^h, \\
& \forall k \in \{1, 2, \ldots, N\}, \\
& S_{t|t} = S_t,
\end{aligned} \tag{3.6}$$

where $\mathcal{B}(\cdot)$ is a set of positions defining the transported object, $U_t = \{u_{t|t}, \ldots, u_{t+(N-1)T_s|t}\}$, S_{tar} is the target state, $Q_s, Q_i \succcurlyeq 0$ are the weight matrices, and inferred obstacle zone penalty $\mathcal{I}_{\mathcal{O}}(S_t, u^h_{t-T_s})$ is defined in [54, Section III-D]. Once an optimal input $u_t^\star$ is computed, the robot utilizes the following assumption to estimate the human's inputs.

Assumption 3.1 *The human's compliant inputs at time step t are computed as*

$$\hat{u}^h_t = p u^\star_t, \tag{3.7}$$

where fraction $p \in (0, 1)$ remains constant throughout the task.

The fraction p can be roughly estimated from collected trial data where the human limits to playing a complying role in the task[1]. Thus, the robot's estimate of the human policy inherently considers that the human is trying to minimize their felt forces and torque in the task to assist the robot, while reacting to the surrounding obstacles in $\mathcal{C}_{r,t}$ in a way which

[1]If the human actively leads the task, potentially forcing/opposing robot's actions, human inputs may be drastically different from its approximate (3.7).

is consistent with the MPC planned trajectory by the robot. Utilizing Assumption 3.1, the robot computes its actions at t as:

$$u_t^{\star,r} = u_t^{\star} - \hat{u}_t^h. \tag{3.8}$$

Trust Value α_t via Difference in Estimated and Actual Human Behavior

Since the robot does not perfectly know the human's intentions and the configuration of obstacles in the vicinity of the human, it does not apply its computed MPC input $u_t^{\star,r}$ to system (3.4) directly. Instead, it checks the deviation of its estimated human inputs from the actual closed-loop inputs applied by the human. The latter can be measured using force and torque sensors on the robot. As the applied human inputs at the current time step are not available for this computation, the robot approximates[2] this deviation by:

$$\Delta u_t^h \approx \hat{u}_t^h - u_{t-T_s}^h.$$

The *trust* value α_t is then computed as:

$$\alpha_t = 1 - \min\{1, \frac{\|\Delta u_t^h\|}{\delta_{\text{thr}}}\}, \tag{3.9}$$

where δ_{thr} is a chosen threshold deviation. The robot uses this trust value to apply a weighted combination of its computed MPC inputs $u_t^{\star,r}$, and inputs proportional to $u_{t-T_s}^h$ as detailed later in equation (3.10). This trust-driven combination of inputs is motivated by the policy-blending approach for a shared control teleoperation task [42]. In our case, the human is directly transporting an object with the robot so we utilize the force feedback to predict the alignment of goals. Haptic feedback to signal intent of the robot to the human has been used in [55]. The robot additionally deploys a safe stopping policy, in case the computed trust value is below a chosen threshold, and it nears obstacles potentially unknown to the human. These two modes of the robot's policy are detailed in the next section.

Trust-Driven and Safe Stop Modes of the Robot Policy

At time step t, we denote the inertial position coordinates of the robot's seen point on the object closest to any obstacle in $\mathcal{C}_{r,t}$ as R_t. After finding a solution to (3.6) and computing $u_t^{\star,r}$ using (3.8), the robot utilizes (3.9) and applies its closed-loop input computed as follows:

$$u_t^r = \begin{cases} \text{proj}_{\mathcal{U}^r}(\alpha_t u_t^{\star,r} + K_1(1-\alpha_t)u_{t-T_s}^h), \text{ if (SS) not true,} \\ \text{proj}_{\mathcal{U}^r}(-K_2 \frac{\dot{R}_t}{Ts}), \text{ otherwise,} \end{cases} \tag{3.10}$$

[2]For sample period $T_s \ll 1$, this can constitute a reasonable approximation.

to system (3.5) in closed-loop with chosen gains $K_1, K_2 > 0$, where $\text{proj}_{\mathcal{A}}(x)$ denotes the Euclidean projection of x onto set $\mathcal{A}$, and the robot's safe stop policy triggering condition (SS) is given by:

$$(\text{SS}) : \alpha_t < \frac{1}{2},\ \min_{o \in \mathcal{C}_{r,t}} \|R_t - o\| \leq d_{\text{thr}}, \dot{R}_t \cdot (o - R_t) > v_{\text{thr}}, \tag{3.11}$$

with distance and velocity thresholds $d_{\text{thr}} > 0$ and $v_{\text{thr}} > 0$. That is, when point R_t approaches any obstacle o at a high velocity under a low trust $\alpha_t < \frac{1}{2}$, the robot actively tries to decelerate the the object and bring it to a halt. From policy (3.10), we make the following observations:

1. A large trust value (e.g., α_t closer to 1), corresponds to the case when the robot's estimates of the human's inputs align with the actual human's inputs. This means that the robot trusts the human to act with a follower role to assist it. The robot then utilizes more of its computed inputs $u_t^{\star,r}$ from the MPC problem (3.6) and takes the leader's role in the task.

2. A small trust value (e.g., α_t close to 0) corresponds to the case when the robot's predictions of the human's inputs do not align with the actual human's inputs. This means that the human is taking on the leader's role, either reacting to obstacles nearby or actively leading the task. The robot does not trust the computed inputs $u_t^{\star,r}$ from the MPC problem (3.6) and takes the follower's role (unless the safe stop policy condition is triggered).

Policy (3.10) is motivated by [46], and qualitatively has the properties of joint impedance and admittance. We see that satisfying condition 1 increases the efficacy of the robot's solution to (3.6), i.e., $u_t^{\star,r}$. To that end, we add the inferred obstacle zone penalty $\mathcal{I}_{\mathcal{O}}(S_t, u^h_{t-T_s})$ to (3.6), adapting the cost to be optimized by inferring information on potential obstacles at the human's vicinity.

Increasing Trust α_t via Inferred Obstacle Zone Penalty $\mathcal{I}_{\mathcal{O}}(S_t, u^h_{t-T_s})$

At time step t, we denote the inertial position coordinates of the human by H_t. We also denote the first three force components of the human input u_t^h by $u^h_{f,t}$. Motivated by the obstacle learning work of [56], we add the extra term $\mathcal{I}_{\mathcal{O}}(S_t, u^h_{t-T_s})$ to the cost in (3.6) at every time step. This term is to be chosen when $\alpha_t < \frac{1}{2}$, and the human applies forces along directions which are more than a user specified threshold ν_{thr} radians apart from its expected ones. We then choose the term $\mathcal{I}_{\mathcal{O}}(S_t, u^h_{t-T_s})$ as follows:

$$\mathcal{I}_{\mathcal{O}}(S_t, u^h_{t-T_s}) = \begin{cases} \sum_{i=1}^{n} \frac{1}{\|(S_t - H_t + K_3 u^h_{f,t-T_s} + o_i)\|}, & \text{if (IO)}, \\ 0, & \text{otherwise}, \end{cases} \tag{3.12}$$

with n choices of the random parameter $0 < o_i \ll 1$ (introduces noise in the direction vector), control gain $K_3 > 0$, and condition (IO) being

$$\text{(IO)} : \alpha_t < \frac{1}{2},\ |\arccos(\frac{\hat{u}^h_{f,t} \cdot u^h_{f,t-T_s}}{\|\hat{u}^h_{f,t}\|\|u^h_{f,t-T_s}\|})| > \nu_{\text{thr}}.$$

Intuitively, we assume that if the human unexpectedly pushes against the robot, they are attempting to avoid some obstacle unknown to the robot. The robot uses these force measurements and generates n virtual obstacle points that are placed relative to the human's location at a distance scaled by the negative force vector, plus some noise. These virtual obstacle points are the robot's estimates of potential obstacles in the human's vicinity, due to which the human's input $u^h_{t-T_s}$ is significantly different from the estimate $\hat{u}^h_t$. Introducing the penalty $\mathcal{I}_{\mathcal{O}}(S_t, u^h_{t-T_s})$ can improve the MPC planner (3.6) by causing it to adjust the path to avoid those obstacle points. This would likely increase the value of α_t as the robot begins to align itself with the human's movement to avoid that inferred obstacle and enables the robot to be a more effective partner.

3.4 Experimental Results

In this section, we present experimental validation results with our proposed approach. The experiments are conducted with a UR5e robot. The human and the robot start the joint transportation task with the center of mass of the transported rigid box at the start state S_0, as shown in Fig. 4.1. Goal state S_{tar} contains the target location, which is known to both agents. Since there is not an exact shared baseline for this problem formulation of a human-robot collaborative transportation task with partial obstacle information, we avoid directly comparing against controllers from other related work. We use the following set of parameters shown in Table 3.1 for the considered experimental scenario.

Table 3.1: Parameters used in control design.

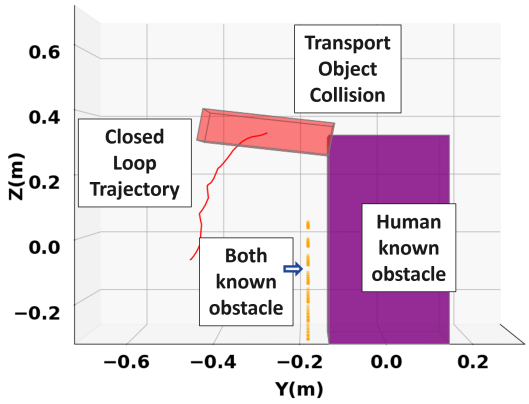
Parameter	Value
T, T_s	100s, 0.05s
N	20
p	0.5
$d_{\text{thr}}, v_{\text{thr}}, \nu_{\text{thr}}$	0.15m, 0.05m/s, $\frac{\pi}{6}$rad
K_1, K_2, K_3	1, 10, 0.005
Q_s	diag(20,20,20,1,1,1)
Q_i	diag(10,10,10,100,100,100)

Trust-Driven Policy vs Pure MPC Policy

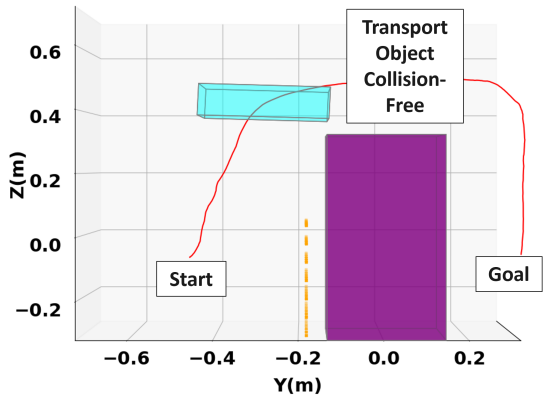
For this section, two obstacles are placed between the agents and the target, as shown in the rendered experiment space in Fig. 3.2b. We show the benefits of using the trust-driven policy mode, where the robot utilizes the trust value α_t to apply a weighted combination of its MPC inputs and the human's inputs to the system. The baseline for comparison is a pure MPC policy, with the robot solving MPC problem (3.6) and applying its optimal input (3.8), being agnostic to the responses of the human. These are computed at a frequency of approximately 2Hz. In the considered scenario in Fig. 3.2, the purple box obstacle located between $Y \in [-0.13m, 0.16m]$ is known only by the human. Both agents are aware of the dotted wall obstacle at $Y = -0.18$ m. In Fig. 3.2a, the robot is operating with the pure MPC baseline policy, agnostic to the human's actions. As a consequence, the planned trajectory by the robot results in the human colliding with this obstacle, as seen in Fig. 3.2a. Resisting force values by the human in Fig. 3.2d indicate the human's opposition to the robot's actions. On the other hand, with our proposed trust-driven policy mode, the robot is cognizant of the human's intentions. The evolution of α_t as the human navigates in the proximity of the box obstacle is shown in Fig. 3.2c. When the transport object nears the obstacle (around 10 sec), the robot completely distrusts its estimate of the human policy with a computed $\alpha_t \approx 0$ and applies the measured human input in (3.10). Collision is averted as a consequence, as seen in Fig. 3.2b. Lower force magnitudes in Fig. 3.2d further indicate that the human's resistance to robot's actions during this collision avoidance is lowered, as the robot lowers the contribution of its MPC inputs in (3.10) with a low value of α_t.

The Safe Stop Mode in Action

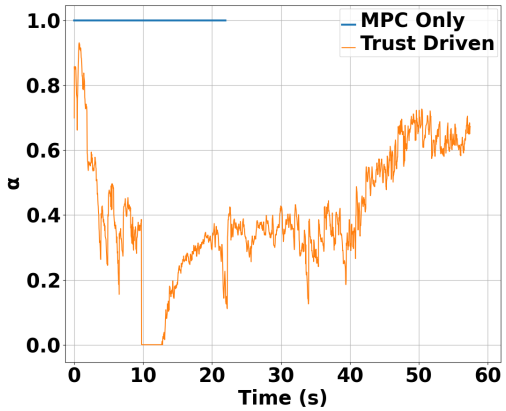
To highlight the safety benefits of adding the safe stop policy mode in (3.10), we consider the scenario shown in Fig. 3.3. For this scenario, only one simulated obstacle wall at $Y = -0.18$ m is in the experiment space which the human does not see. The human decides to drive the transport object towards the goal via the shortest path without being aware that it is leading towards the wall. Without activating the safe stop policy backup, the robot's inputs continue to comply with the inputs from the human, as shown in the force plots in Fig. 3.3c. As a result, the transported object collides with the obstacle wall, as seen in Fig. 3.3a. On the other hand, in Fig. 3.3b we see that utilizing the safe stop policy mode manages to prevent this collision and maintain safety in the transportation task. This safety retaining effect of the safe stop mode can be explained from Fig. 3.3d, where next to the obstacle wall when condition (SS) is triggered (around 10 sec), we no longer see the robot's applied forces complying with the human's forces. Instead, the robot applies a decelerating safe stop input, which results in the collision avoidance. There is some oscillation of the robot's input which shows it using the safe-stop policy when they are approaching a known obstacle and reverting to the human's input (due to low trust) when it appears safe again. Ultimately, the task is completed successfully.



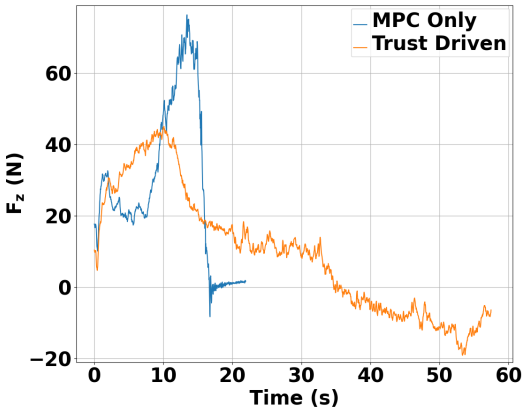
(a) Pure MPC policy ($\alpha = 1$) resulting in collision with obstacle only known by the human.



(b) Trust-Driven Policy (adaptive α) resulting in a collision-free trajectory.



(c) α vs Time. The trust-driven policy adapts the value of α_t for all $t \geq 0$ based on the human's responses in the task.

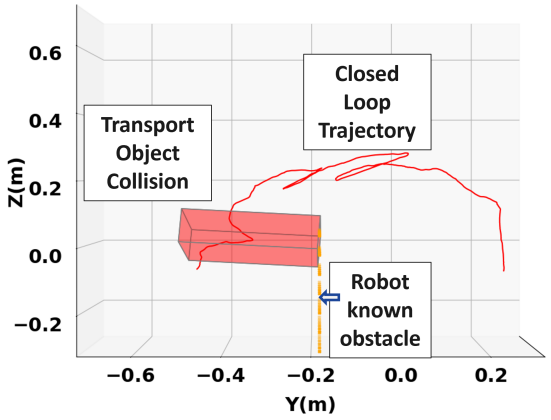


(d) Measurement of human force applied in Z direction vs Time. Using the trust-driven policy enables the human to lower resisting forces, while avoiding collision.

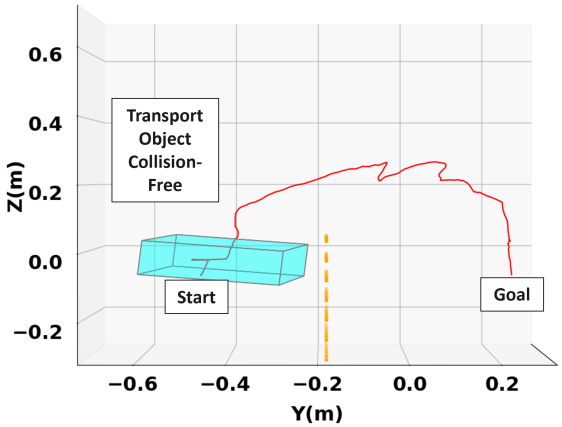
Figure 3.2: Comparison of experimental results of robot with pure MPC policy vs. trust-driven policy.

Randomized Analysis

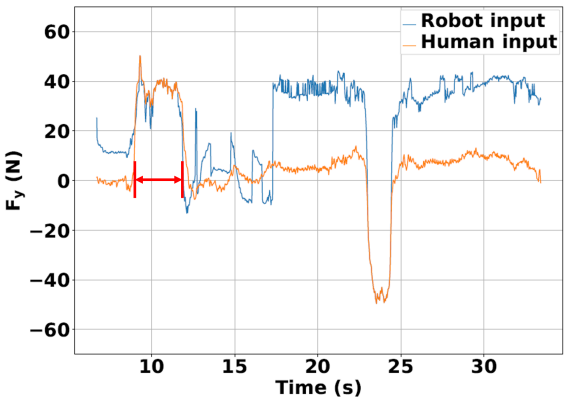
In order to generalize the validity of the above results beyond the considered example, we carried out the transportation task and analyzed the closed loop behaviors of the proposed controller with 100 configurations of randomized start, goal and obstacle positions. The shared transport object remained the same throughout all tasks. In some cases, the obstacles are purely simulated for faster testing purposes. The detailed results are shown in Table 3.2 where we use three metrics to compare the 100 trials. A *Collision-Free Success* is a trial



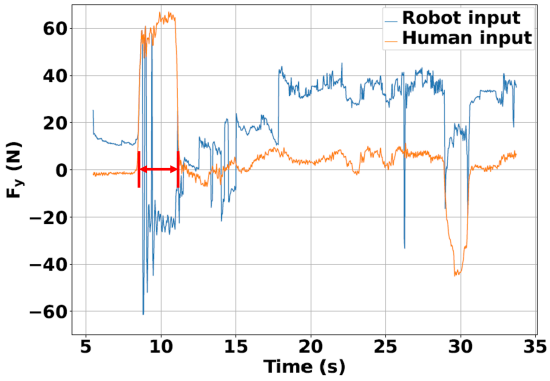
(a) No safe stop policy. The robot collides with the obstacle.



(b) With safe stop policy. Collision is avoided.



(c) Without the safe stop policy, the robot provides assisting force that matches the unexpected human inputs even if it leads towards a known obstacle (marked from 8.8s to 11.9s). The human behavior causes a collision with the obstacle wall and the robot helps them do so.



(d) With the safe stop policy, the robot applies decelerating safe stop input to cancel out the human inputs when it detects that a collision with an obstacle is imminent (marked from 8.5s to 11.2s). This prevents the human from leading the transport object into the obstacle wall

Figure 3.3: Effect of the safe stop policy mode in avoiding collisions.

where the transport object is brought to the target state without hitting obstacles. *Peak Human Force* is the largest magnitude of force applied by the human throughout a given trial. The *Duration of Intervening Forces* is the length of time in which the human has applied more than 30N in a given trial. Table 3.2 shows that the proposed approach results in a 37% increase in the number of Collision-Free Successes. Moreover, the average value of the Peak Human Force lowers by 14.9% with the proposed approach, indicating decreased opposition of the human during the task. The results show that the average Duration of Intervening Forces shortens by 61.8% with our approach. The robot cedes some of the

Table 3.2: The percentage and the average are computed numerically from 100 trials of the transportation task.

Feature	MPC Only	Trust-Driven w/ Safe Stop
Collision-Free Successes (%)	51	88
Avg. Peak Human Force (N)	63.276	53.835
Avg. Duration of Intervening Forces (s)	5.934	2.265

control authority to the human as the trust value decreases. This occurs when the human does something unexpected to the robot. On the other hand, with the pure MPC approach, the robot attempts to follow its optimal trajectory even in the case where a collision with an object known only by the human is imminent. Thus, the human needs to continuously apply the intervening force for longer periods of time when no trust value is used.

3.5 Conclusion

We proposed a framework for a human-robot collaborative transportation task in presence of obstacles in the environment. The robot plans a trajectory for the transported object by solving a constrained finite time optimal control problem and appropriately applies a weighted combination of the human's applied and its own planned inputs. The weights are chosen based on the robot's trust value on its estimates of the human's inputs. This allows for a dynamic leader-follower role adaptation of the robot throughout the task. The robot will also infer obstacles known only to the human when the trust is low so the planner doesn't continue to generate the same conflicting trajectory. With experimental results, we demonstrated the efficacy of the method.

Chapter 4

Deposition with Degradable Tools: Width Tracking

We present a data-driven optimization approach for robotic controlled deposition with a degradable tool. Existing methods make the assumption that the tool tip is not changing or is replaced frequently. Errors can accumulate over time as the tool wears away and this leads to poor performance in the case where the tool degradation is unaccounted for during deposition. In the proposed approach, we utilize visual and force feedback to update the unknown model parameters of our tool-tip. Subsequently, we solve a constrained finite time optimal control problem for tracking a reference deposition profile, where our robot plans with the learned tool degradation dynamics. We focus on a robotic drawing problem as an illustrative example. Using real-world experiments, we show that the error in target vs actual deposition decreases when learned degradation models are used in the control design.

The results presented in this chapter have also appeared in:

- T. Zheng, M. Bujarbaruah, and F. Borrelli. "Data-Driven Optimization for Deposition with Degradable Tools". In: *22nd IFAC World Congress*. Vol. 56. 2. 2023, pp. 4375–4380.

4.1 Introduction

Robotic manipulation in contact-rich tasks have seen great advancements in recent years [57]. There has been a push towards robots that can help in daily household chores such as folding clothes [58], wiping surfaces [59], or various kitchen tasks [60]. While these tasks are certainly challenging, there are still unaddressed problems in the field where the contact tool itself changes over time. Examples include cutting blades that decrease in sharpness through repeated use, sandpaper which wears away, or chalk for marking surfaces.

For this work, we consider the deposition with degradable tools in the application of robotic drawing. Artwork and videos generated by AI in recent works [61, 62, 63, 64, 65] have been able to produce complex creations that could easily be mistaken as drawn by

professional artists. Text-to-image generation has been able to produce some incredible results that allow the user to create highly specific combinations of subjects performing actions in locations, even in the artwork style that they desire [62, 64, 65]. Research in predictive language models like GPT-3 [66] linked with large image databases [67] have been a huge part of these advancements. However, one major hurdle has been translating these artworks into the real world. There are many complex physical interactions involved in various mediums of artwork such as oil painting, pencil sketching, water-colors, etc. Work has been done the decomposition of images into individual strokes to reproduce the image [68, 69]. The final drawn images can be highly accurate but they lack insight in actually making a robot hold a paintbrush or pencil to produce those strokes. Current state-of-the-art approaches for robotic drawing are predominately hand-tuned open loop sequences with custom end-effectors that make it easier to have a constant force output and frequent reset sequences to allow for consistency[70, 71]. A more accurate replication of human drawing would take the deformation of a tool-tip into account and change the policy accordingly. To the best of our knowledge, there has not been works that use visual feedback to update the model of a tip's degradation to produce more accurate strokes.

In this chapter, we formulate the deposition task as a model-based constrained finite time optimal control problem, where we model the deposition and the degradation of the tool tip. The parameters of these models are not known a-priori, and we learn these using collected data. We focus on the specific example of a robot sketching using a pencil. The unknown degradation and deposition models of the pencil are parameterized as a function of the applied force and the distance drawn. We present detailed experiments with a UR5-e robot where we show that accounting for the degradation of the tip and planning strokes accordingly improves the sketching quality measured in terms of the difference in stroke width error.

4.2 Related Work

Simulated stroke generation

Reinforcement learning approaches have been shown to work in generating realistic strokes. [72] formulates brush strokes as an Markov decision process and solves it using policy gradient methods. [68] takes target images and decomposes them into stroke sequences with deep reinforcement learning. [73] introduced a CNN-based auto-encoder to generate multi-class sketches. [69] uses the gradient of grayscale values to determine pencil stroke sequences used to recreate images. While these works produce strokes that appear similar to real ones, they are not tested on real robots where the actual deposition will likely differ from their expected.

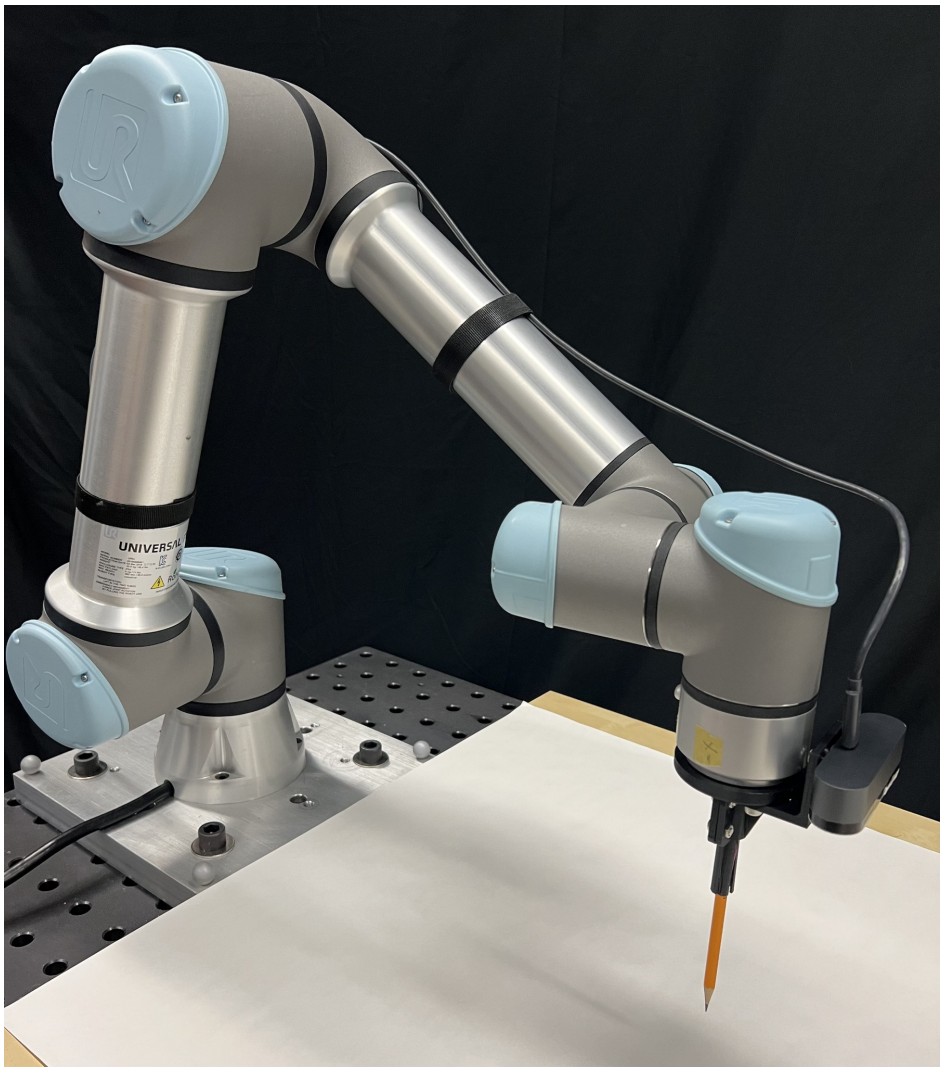


Figure 4.1: The considered experimental setup.

Real-world stroke generation

[74] formulate calligraphy writing as a trajectory optimization problem and developed a novel dynamic brush model. They produce open-loop trajectories for that a robot can follow but do not close the loop. [75] use linear regression to relate brush pressure with actual deposition width and generates trajectories to fill a desired calligraphy image. [76] trains a generative model on expert artist motion data in order to extract artistic style into robotic painting. They did not include closed-loop control and studied whether a playback of the artist motions with a robotic arm could produce brushstrokes similar to humans. [71] trains RNNs combined with LSTMs on images and produces commands for the robot that aim to draw strokes in continuous fluid motions. [77] uses CNNs and GANs to extract sketch

outlines from images and replicates the contours with a brush-pen. These actual robot demonstrations are done with open-loop sequences as well.

[78] explores pencil drawing and uses genetic algorithms to produce line segment sequences that can reproduce detailed drawings. Their robot uses a passive flexible tool to hold their custom graphite writing implement and compensates for drawing pressure changes over the surface. The tool-tip is designed in such a way that the graphite can be reset and calibrated frequently. [79] uses impedance control to draw on arbitrary surfaces using a pen. [80] corresponds force values with grayscale values of an image to shade using a pencil. These methods do not consider degradation of the tool-tip over time.

4.3 Problem Formulation

In this section, we describe the models used for our optimization based deposition problem.

Reference Stroke

When drawing a picture, there are a limitless number of ways to decompose a desired image into individual strokes. We start with the assumption that each reference stroke is already given using stroke generation methods such as [69, 68]. The output of these generators are the parametric curve equations, width along the stroke, and color values. We convert these image coordinates so that the reference states, $s_{\text{ref}}(\zeta(t))$, to be used in the cost function for our optimization problem are:

$$s_{\text{ref}}(\zeta(t)) = \begin{bmatrix} x_{\text{ref}}(\zeta(t)) \\ y_{\text{ref}}(\zeta(t)) \\ W_{\text{ref}}(\zeta(t)) \end{bmatrix}, \tag{4.1}$$

where t is the time step, $\zeta(t)$ is a parameter used to define the curve s_{ref}, $\{x_{\text{ref}}(\zeta(t)), y_{\text{ref}}(\zeta(t))\}$ are the x and y positions (m) respectively and $W_{\text{ref}}(\zeta(t))$ is the deposition width (m). Visually represented in Fig. 4.2, $p(\zeta(t))$ is the tangent vector to the stroke reference path $\{x_{\text{ref}}(\zeta(t)), y_{\text{ref}}(\zeta(t))\}$ sampled at any time step. Then, the deposition width $W_{\text{ref}}(\zeta(t))$ is defined as the thickness of the stroke cross section at that point measured in a direction perpendicular to $p(\zeta(t))$. Henceforth, we will replace $(\zeta(t))$ with (t) for the simplicity of notations.

End Effector Modeling

We start the modelling of the discretized system beginning with the robot arm. We treat the end-effector of the robot arm as a single integrator:

$$\bar{x}(t+1) = A\bar{x}(t) + Bu(t), \tag{4.2}$$

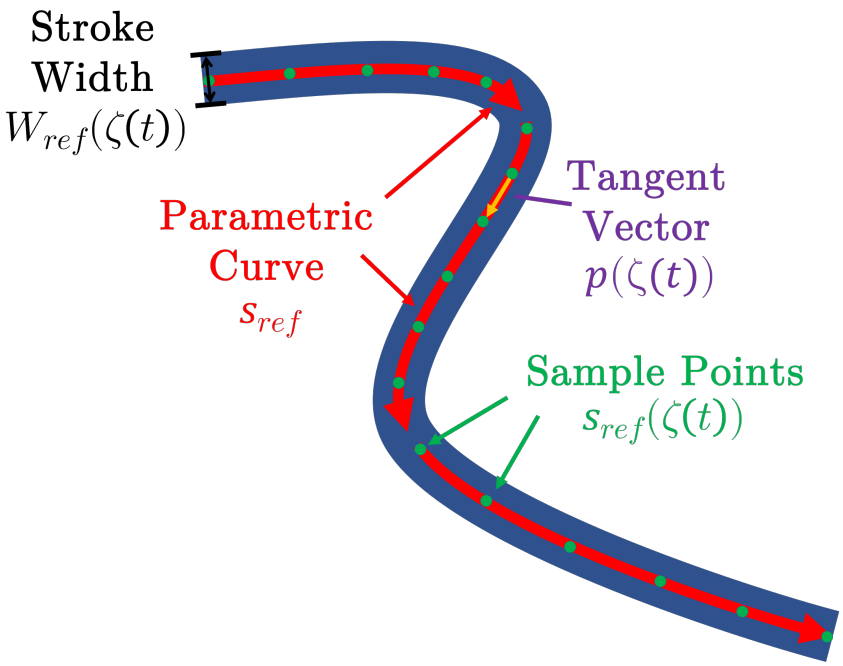


Figure 4.2: Stroke Analysis

with $A = I_4$, $B = dt \cdot I_4$ where I_n denotes the identity matrix of size n, and sampling time $dt = 0.008$s. The states and inputs are

$$\bar{x}(t) = \begin{bmatrix} x(t) \\ y(t) \\ z(t) \\ \psi(t) \end{bmatrix}, u(t) = \begin{bmatrix} v_x(t) \\ v_y(t) \\ v_z(t) \\ \omega_\psi(t) \end{bmatrix}, s(t) = \begin{bmatrix} x(t) \\ y(t) \\ W(t) \end{bmatrix}, \tag{4.3}$$

where $\{x(t), y(t), z(t)\}$ are end effector positions. The angle $\psi(t)$ is the angle between the pencil's ellipsoidal cross section's minor axis projected on the xy plane, and the tangent to the stroke at time t. The inputs are their respective velocities. During control, we impose state and input constraints for the end-effector at all time steps $t \geq 0$ as given by:

$$x(t) \in \mathcal{X},\ u(t) \in \mathcal{U}, \tag{4.4}$$

for all $t = 0, 1, \ldots, N$, where $N > 0$ is the task horizon, and the sets $\{\mathcal{X}, \mathcal{U}\}$ are polytopes.

As the tool is pushed deeper along the z-axis into the surface, a greater force is applied which we model in the following section.

Force Map Modeling

The force applied by the tool to the work surface can be modeled as a general nonlinear function

$$F(t) = f_F(z(t) - z_{\text{ref}}(t), \eta(t)), \tag{4.5}$$

where z_{ref} is the work surface height, $z(t) - z_{\text{ref}}(t) \leq 0$ ensures that the tool is penetrating into the surface (for a downward direction), and $\eta(t)$ is a parameter that describes other contact conditions such as soft or dissipative contact. To obtain this relationship, we control the robot such that the tool tip makes contact with the work surface and then applies various penetration depths while taking force measurements. Because the actual function $f_F(\cdot, \cdot)$ is unknown for our tool, we fit a simplified linear model of the form shown below, ignoring the dependence on $\eta(t)$:

$$F(t) = \theta(z(t) - z_{\text{ref}}(t)) + \theta_0, \tag{4.6}$$

where parameters θ, θ_0 are unknown and to be learned from collected data.

For drawing tools such as pencils, the contact force has a large effect on the rate at which the tip degrades. Our main contribution is that we utilize tip degradation of a rotated tip into the trajectory planning which we model in the next section.

Tool Tip Modeling

First, we model the pencil tip as a cone which will come into contact with a planar surface as shown in Fig. 4.3. This results in the general equation for the ellipse:

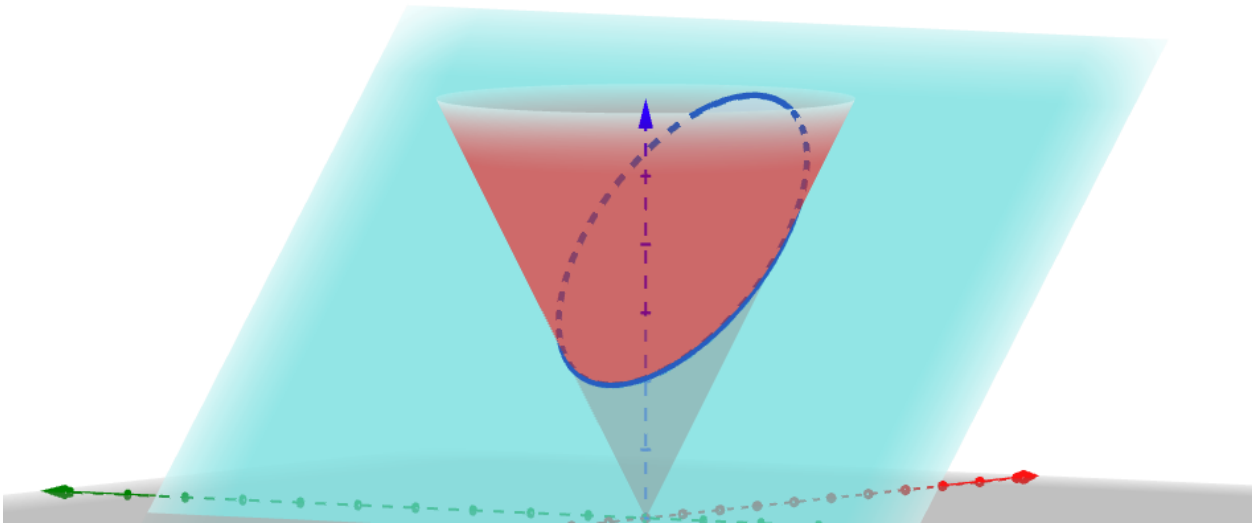


Figure 4.3: Intersection of a cone and hyperplane.

$$\frac{(m^2-a^2)^2(x+\frac{ad}{a^2-m^2})^2}{m^2d^2}+\frac{(m^2-a^2)y^2}{d^2}=1, \tag{4.7}$$

where m is the slope of the cone, a is the slope of the hyperplane, and d is the vertical offset of the hyperplane from the cone's tip.

Assumption 4.1 *The slope (a) remains the same throughout the task duration to ensure that the shape of the surface stays an ellipse. This also relies on an upper bound for d such that it does not make the hyperplane intersect past the top of the cone.*

Under Assumption 4.1, the intersection formed is an ellipse under the condition that the angle of the plane is shallower than the slope of the pencil's edge, i.e., $|a| < |m|$. We thus have:

$$\alpha(t)=\frac{2m\sqrt{1+a^2}d(t)}{m^2-a^2},\ \beta(t)=\frac{2d(t)}{\sqrt{m^2-a^2}}, \tag{4.8}$$

where $\{\alpha(t),\beta(t)\}$ are the major and minor axes of the elliptical cross section at t, respectively. We define the angle γ such that $a=\tan(\gamma)$ which physically describes the angle in which the pencil tip is oriented against the surface. Note that when $\gamma=0$, the pencil is pointed downward into a horizontal surface which is the perpendicular direction and $\alpha=\beta$.

Degradation Modeling

The degradation of the tip is modeled as function of the current state of the tip, force applied to the surface with the tip, and distance travelled while in contact. The evolution dynamics of the pencil tip can be described using d from (4.7):

$$d(t+1)=d(t)+K_dF(t)\begin{bmatrix}x(t+1)\\y(t+1)\end{bmatrix}-\begin{bmatrix}x(t)\\y(t)\end{bmatrix}_2 \tag{4.9}$$

where K_d is a scaling parameter. As the pencil is being used and wearing away, the hyperplane is pushed further into the cone.

Deposition Modeling

Using the geometry of the ellipse, we model this deposition width as:

$$W(t)=\max(\alpha(t)\cos(\psi(t)),\beta(t)\sin(\psi(t))), \tag{4.10}$$

thus the deposition width is the maximum between the major and minor axes projections measured perpendicular to the stroke at t.

4.4 Data-Driven Control Synthesis

In this section, we formulate our data-driven optimization approach for deposition with a degradable tool. Since the parameters from (4.6) in the degradation (4.9) models are unknown, we estimate them using data from repeated strokes, *before* attempting to track the reference stroke with the pencil. This is done during a training phase.

Learning Parameters during the Training Phase

We can rewrite (4.6) as:

$$F(t) = \mathbf{z}^\top(t)\Theta, \tag{4.11}$$

where $\mathbf{z}(t) = [z(t) - z_{\text{ref}}(t), 1]^\top$ and $\Theta = [\theta, \theta_0]^\top$. In this case, the estimated parameters $\hat{\Theta}$ can be obtained using ordinary least squares as:

$$\hat{\Theta} = (\mathbf{Z}^\top\mathbf{Z})^{-1}\mathbf{Z}^\top\mathbf{F}, \tag{4.12}$$

where

$$\mathbf{Z} = \begin{bmatrix} z(0) - z_{\text{ref}}(0) \\ z(1) - z_{\text{ref}}(1) \\ \vdots \\ z(T_{\text{off}}) - z_{\text{ref}}(T_{\text{off}}) \end{bmatrix}, \ \mathbf{F} = \begin{bmatrix} F(0) \\ F(1) \\ \vdots \\ F(T_{\text{off}}) \end{bmatrix},$$

where $T_{\text{off}} \gg 0$ is the offline time used to collect data and fit the models. We then use $\hat{\Theta}$ in our optimal stroke design. Note that after learning the parameters offline, we assume the pencil is sharpened to it's initial state again, so that the given reference stroke can now be tracked using our best parameter estimates $\hat{\Theta}$.

Optimal Stroke Tracking Synthesis

Let $\hat{\theta}$ and $\hat{\theta}_0$ be our estimates of unknown parameters θ and θ_0, respectively, obtained offline. The optimal control problem solved at $t = 0$ is given by:

$$\begin{aligned}
\min_{U_t} \quad & \sum_{t=0}^{N} (s(t) - s_{\text{ref}}(t))^\top Q (s(t) - s_{\text{ref}}(t)) \\
\text{s.t.,} \quad & \bar{x}(t+1) = A\bar{x}(t) + Bu(t), \\
& F(t) = \hat{\theta}(z(t) - z_{\text{ref}}(t)) + \hat{\theta}_0, \\
& d(t+1) = d(t) + K_d F(t) \begin{bmatrix} x(t+1) \\ y(t+1) \end{bmatrix} - \begin{bmatrix} x(t) \\ y(t) \end{bmatrix}_2 \\
& W(t) = \max(\alpha(t)\sin(\psi(t)), \beta(t)\cos(\psi(t))), \\
& \alpha(t) = \tfrac{2m\sqrt{1+a^2}d(t)}{m^2-a^2}, \beta(t) = \tfrac{2d(t)}{\sqrt{m^2-a^2}}, \\
& \bar{x}(t) \in \mathcal{X}, u(t) \in \mathcal{U}, \ t = 0, 1, \ldots, (N-1), \\
& s(0) = s_0, \ \bar{x}(0) = \bar{x}_0,
\end{aligned} \tag{4.13}$$

where $s(t) = [x(t), y(t), W(t)]^\top$, $Q \succ 0$, and $s_0, \bar{x}_0$ are known and fixed. After computing the optimal trajectory, we apply the whole batch solution, $U_t = [u(0), u(1), ..., u(N-1)]$, as an open loop rollout without re-solving (4.13) during the stroke. We then use an image of the drawn stroke to estimate the actual width $W_a(\zeta(t)) := W_a(t)$ along the generated stroke. The computation of this actual width is detailed in Section 4.5. The actual and the reference stroke widths are then compared to compute an *error metric* as:

$$V = \sum_{t=0}^{N} |W_a(t) - W_{\text{ref}}(t)|. \tag{4.14}$$

Note that the reference width $W_{\text{ref}}(t)$ is computed along the reference position trajectory for all t. Thus, the error metric in (4.14) is only concerned with quantifying the z-direction force tracking error and not x, y tracking. The x, y position tracking error is regluated by tuning matrix Q and the low level control parameters of the robot. We do not apply learning for such control loop.

4.5 Experimental Results

In this section we present detailed experimental results with our proposed theory. We first show the efficacy of the offline model fitting strategy presented in Section 4.4, but then also highlight the iterative lowering of error metric (4.14) if the model parameters in (4.6) are learned after iterative stroke attempts via (4.13). Such iterative learning attempts also offer an intuitive bound for T_{off}.

Hardware Setup

We perform the real-world experiments using a UR5e 6-DOF manipulator with a custom end-effector that is holding the drawing utensil. The tool is rigidly mounted, as shown in Fig. 4.1, which imposes a hard constraint on the maximum allowable force applied by the robot arm when drawing. We use a Logitech Brio 4K Webcam to capture images of the drawings.

Image Processing and Computing $W_a(t)$

In order to evaluate the closed-loop error metric of a stroke as defined in (4.14), we use images of the performed stroke and compare them to a desired one. In this case, we start with a desired (i.e., reference) stroke generated from a parametric curve (4.1). The first sequence is performed with a nominal open loop maneuver that is pre-selected. After the robot performs the stroke, the robot takes an image with the camera. The color image is converted to grayscale and then a threshold filter is applied based on the darkness of the actual stroke and the canvas. Finally, a contour filter is used to detect the stroke

outline. Starting at the known beginning of the stroke and following along the parametric curve, we compute the width at each sampled point by finding the closest points which are perpendicular to the tangent line. This lets us compute the error metric (4.14).

Offline Force Calibration and Model Fitting

To learn the parameters in model (4.6), we took measurements using the force sensor on the UR5e as the robot moved the tool downwards into the work surface. We ran sinusoidal sweeps of various penetration depths and used linear regression to determine the applied force as a function of penetration depth. Fig. 4.4 shows the line of best fit plotted on top

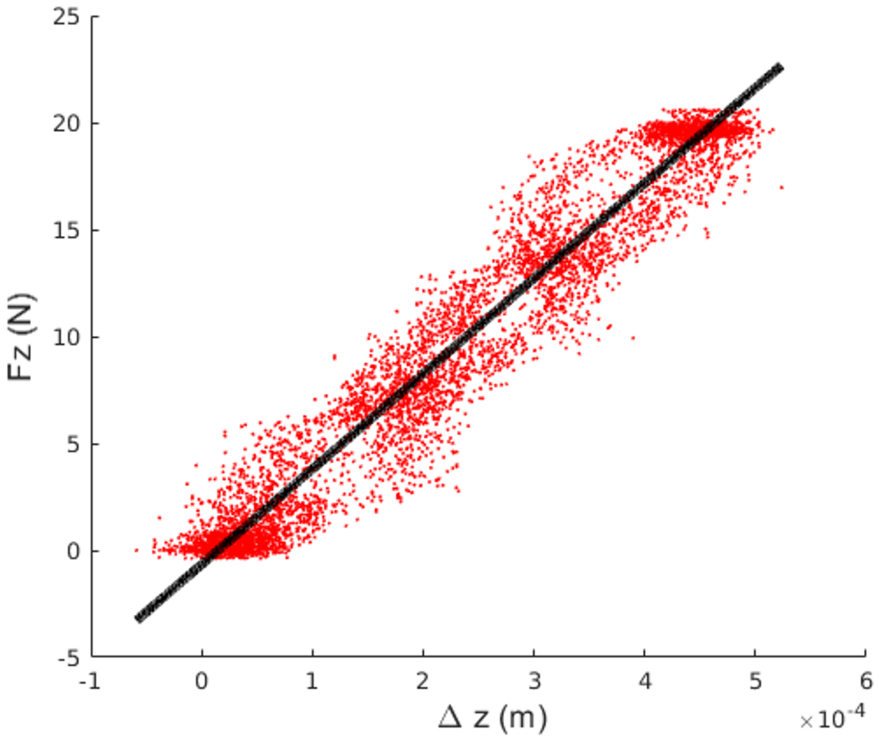


Figure 4.4: Force vs penetration plot.

of the force measurements vs penetration depths. Due to the work surface not being flat, we recorded the contact points along the x and y directions shown in Fig. 4.5. The height of the contact points are used as offsets to z_{ref} to maintain a consistent penetration depth along a stroke after contact is initially made during a trial.

Thus, parameters of (4.6) can be learned offline as shown in Fig. 4.5. However, in the next two sections, in order to demonstrate an iterative improvement of the stroke starting from the same initial condition, instead of offline fitting as shown in Section 4.4 and Fig. 4.5,

we learn the model parameters in (4.6) after every iterative attempt of the stroke obtained through a solution of (4.13). Note, each stroke is independent, and not drawn over the previous.

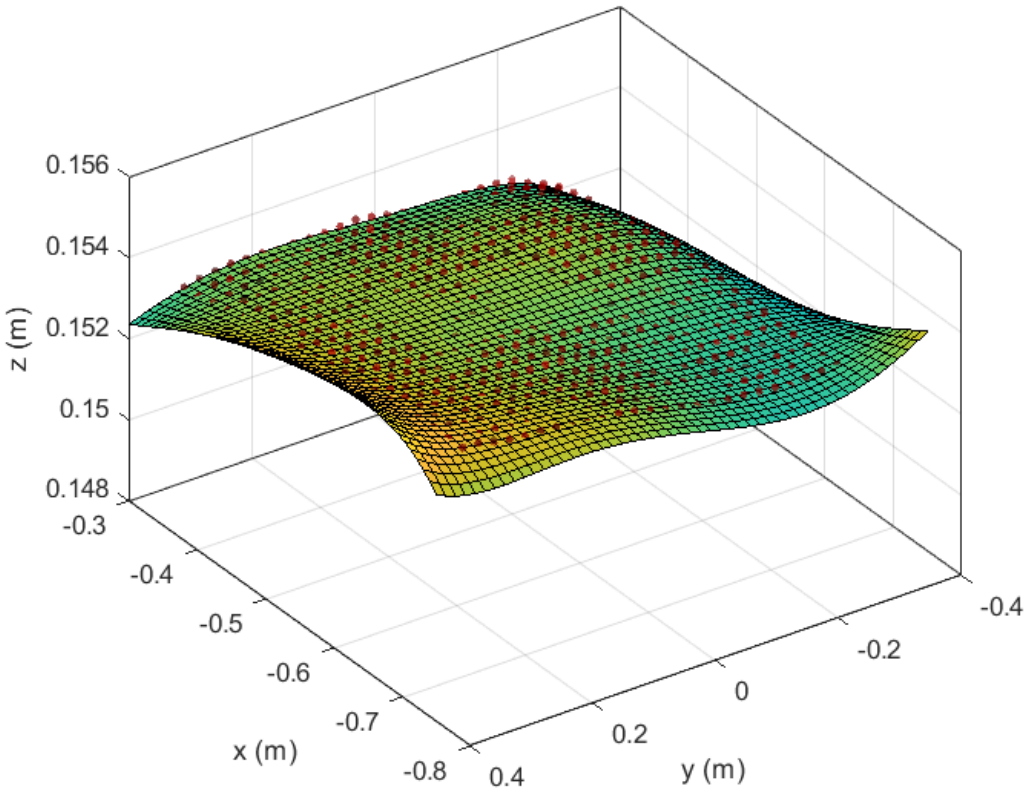
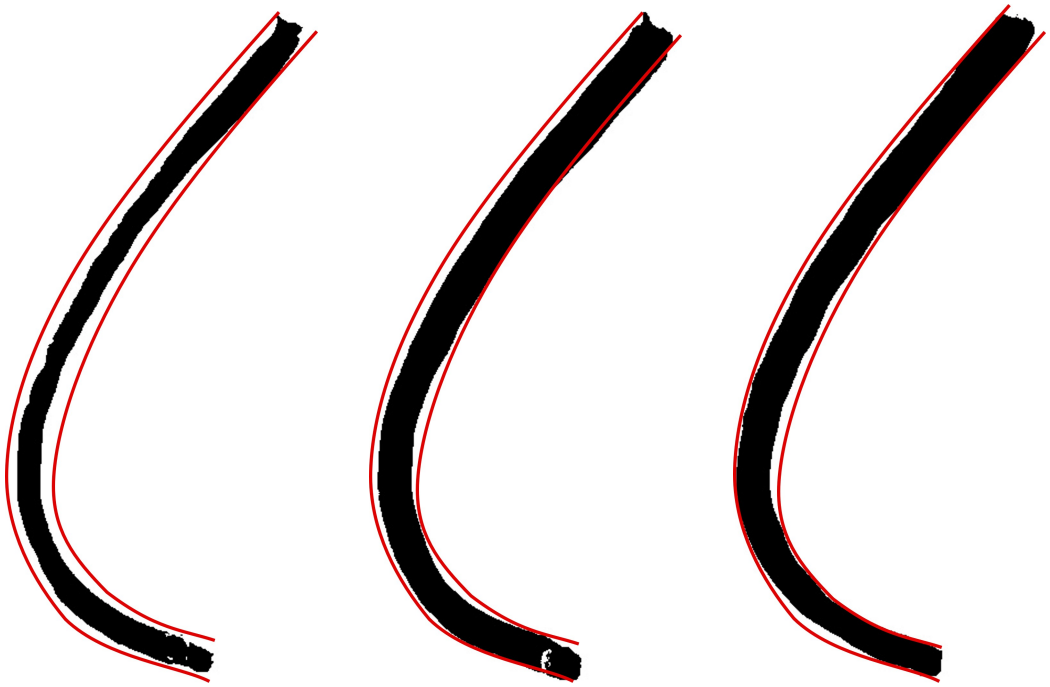


Figure 4.5: Surface plot of contact points on the drawing table.

Deposition without Degradation

To test our approach, we first consider the case where the deposition width is a function of the tip state and force applied but the tip does not degrade so $K_d = 0$ in (4.9). We equipped the robot with a marker perform reference strokes as shown in Fig. 4.6. We see



(a) Stroke 1 (b) Stroke 2 (c) Stroke 5

Figure 4.6: Strokes performed with a marker.

from Fig. 4.6 that the strokes get wider towards the target width as the iterations proceed. This is also reflected in the computed error metric in Fig. 4.7. Initially, the deposition model is unknown and starts out with a nominal value of 2N applied force required per millimeter of width. After each iteration is completed, the recorded actual width values along the stroke and forces applied are used to update the deposition model. Consequently, the error metric decreases by roughly 43% after 2 strokes and 66% after 5 strokes, as shown in Fig. 4.7.

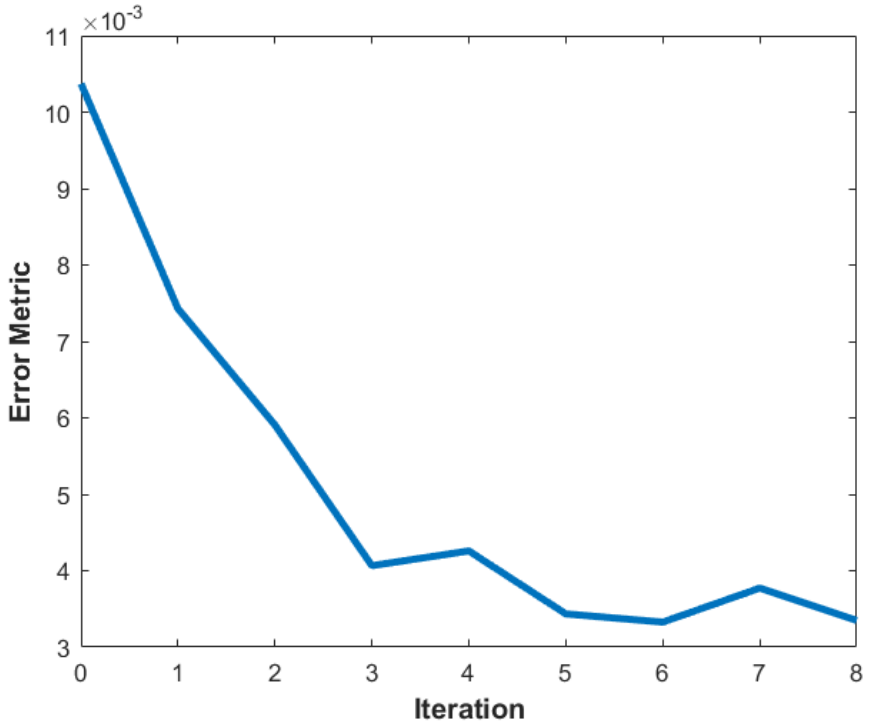
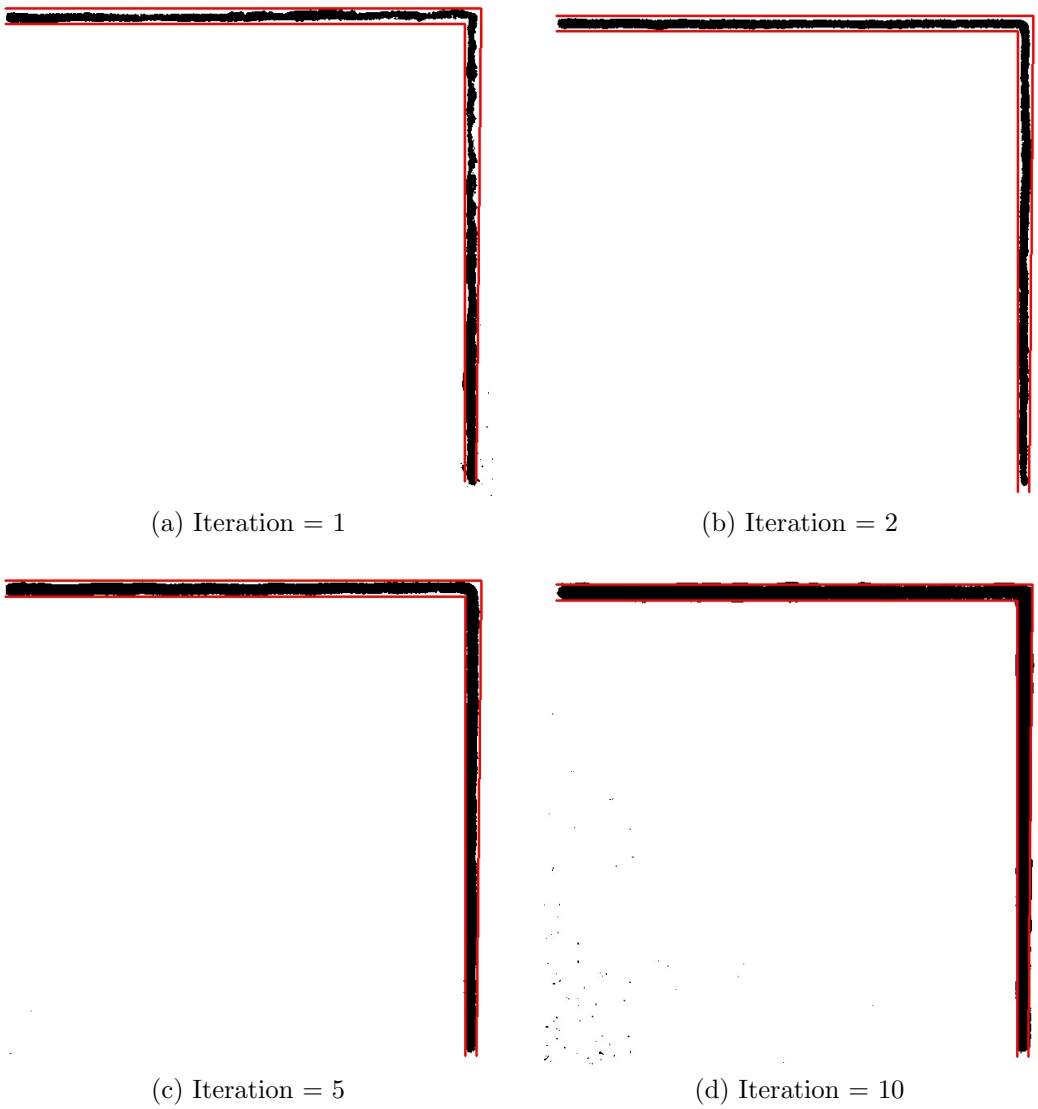


Figure 4.7: Evolution of the error metric as more strokes are performed and data is collected.

Deposition with Degradation

We set the baseline for comparison as a pencil that is mounted perpendicular to the work surface, similar to [78, 80], where $\gamma = 0°$ and initial tip width, $\alpha(0) = \beta(0)$, as 0.1mm. The robot performs a stroke where the desired width is 1.0mm as it moves right and 0.7mm as it moves down as seen in Fig. 4.8. We repeat these trials using an angled pencil where $\gamma = 50°, m = 5.45, d_0 = 0.1$mm. Fig. 4.10 shows our approach using an angled pencil outperforms the baseline throughout the iterations with error metric lowered by about 3.6% all the way to 65.5%.

A main reason for this is the ability to vary the deposition width by controlling the ψ of the pencil while it is tilted as seen in Fig. 4.11. Using the longer major axis diameter of the elliptical profile of the pencil tip, the angled pencil can get closer to the desired widths even when a low amount of wear has occured.A pencil with $\gamma = 0°$ can only have a single width since the surface of the tip that makes contact with the paper is a circle. This can be a problem as seen in iteration 10 where the tip becomes larger than desired and there is no way to correct it as shown in Fig. 4.12. Our approach can possibly decrease the number of strokes necessary to draw a picture that requires shading since it can leverage the longer diameter while still doing precise strokes with the shorter minor axis diameter.



(a) Iteration = 1

(b) Iteration = 2

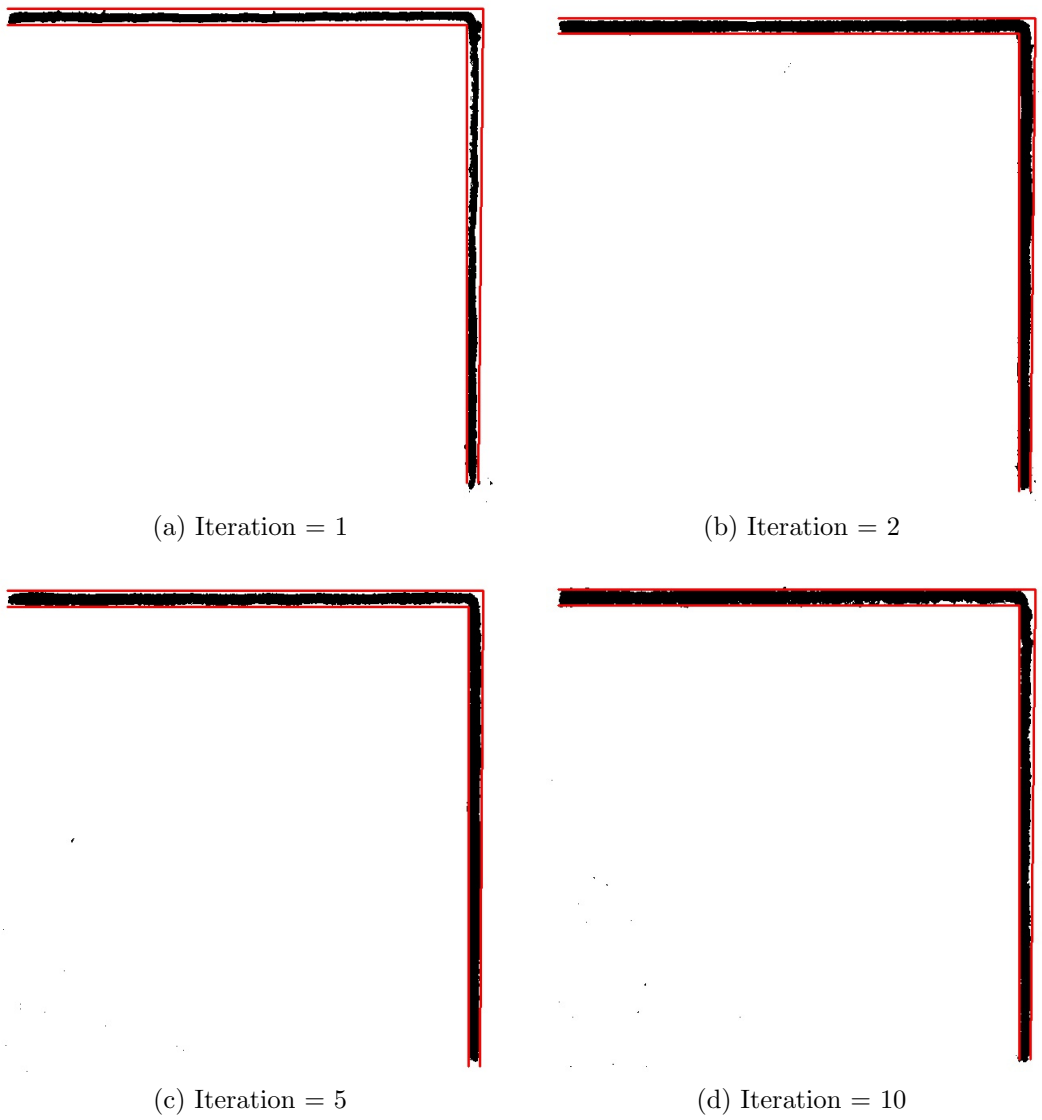
(c) Iteration = 5

(d) Iteration = 10

Figure 4.8: Post-processed images of pencil strokes with $\gamma = 0°$

4.6 Conclusion

We presented a data-driven optimization approach for robot controlled deposition with a degradable tool, specifically the robotic drawing problem. We utilized visual and force feedback to update the unknown model parameters of our tool-tip using least squares. We solved a constrained finite time optimal control problem for tracking the reference deposition profile, where our robot planned with the learned tool degradation dynamics. With real experiments on a UR5e robot, we showed that the error in target vs actual deposition



(a) Iteration = 1

(b) Iteration = 2

(c) Iteration = 5

(d) Iteration = 10

Figure 4.9: Post-processed images of pencil strokes with $\gamma = 50°$

decreased by up to 65% due to the incorporation of learned degradation models in our trials.

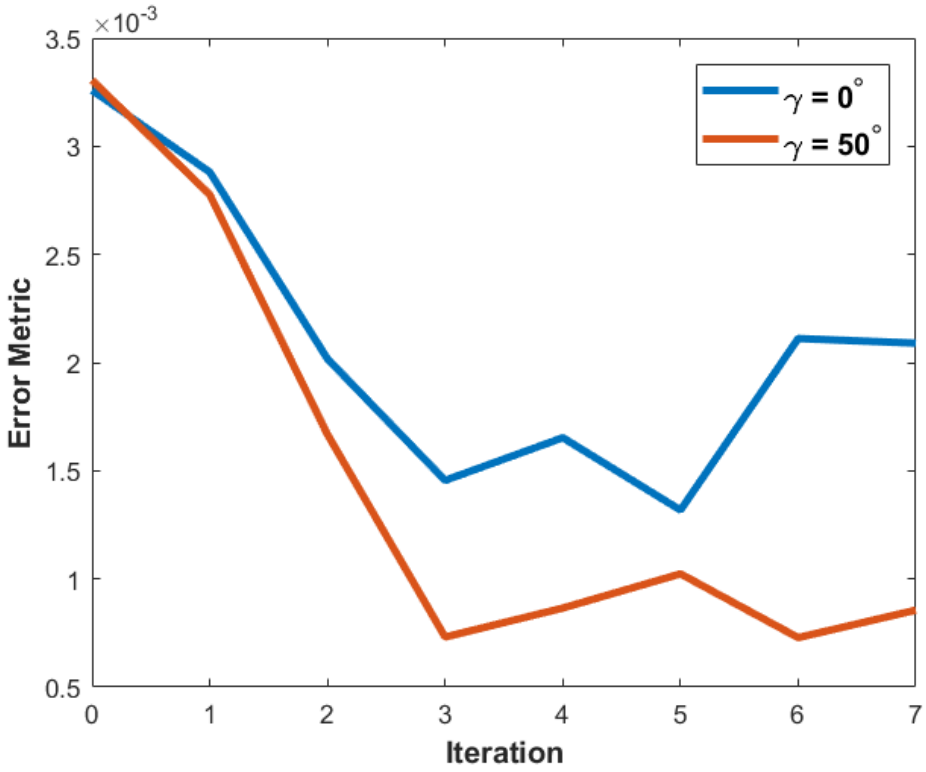
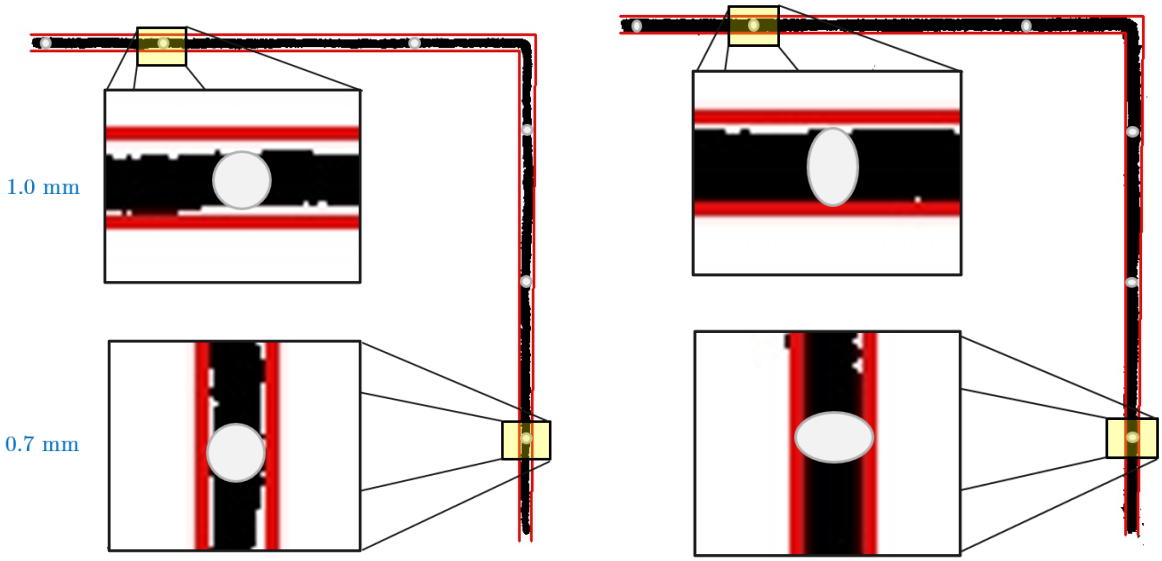
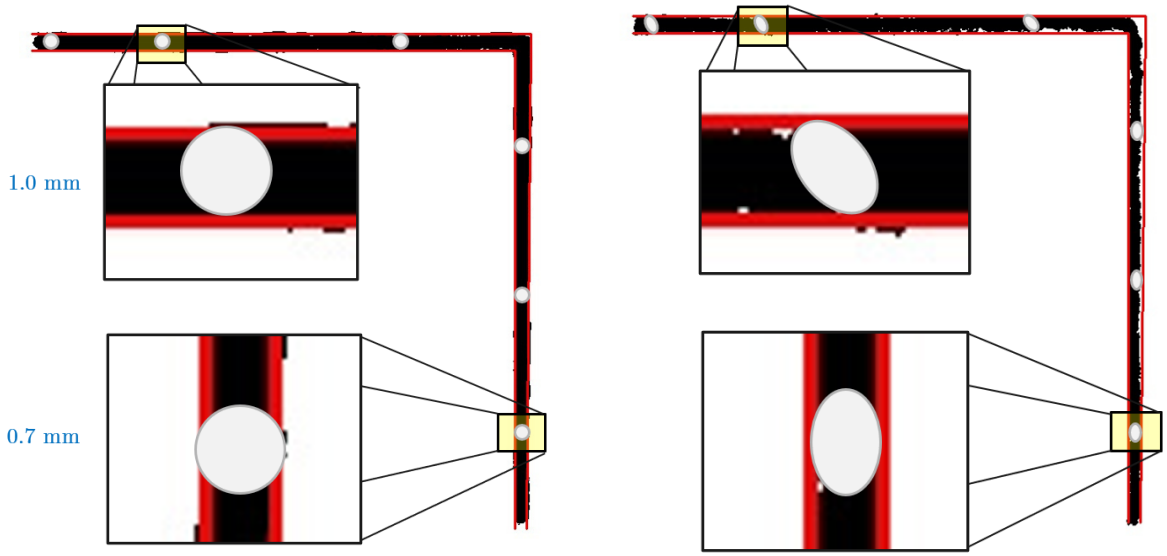


Figure 4.10: Comparison of the error metric with standard ($\gamma = 0°$) vs our approach ($\gamma = 50°$) .



(a) Perpendicular pencil $\gamma = 0°$ (b) Angled pencil $\gamma = 50°$

Figure 4.11: Close up of iteration 2 – Rotation of angled pencil allows for wider strokes.



(a) Perpendicular pencil $\gamma = 0°$ (b) Angled pencil $\gamma = 50°$

Figure 4.12: Close up of iteration 10 – Rotation of angled pencil allows for narrow strokes.

Chapter 5

Deposition with Degradable Tools: Edge Planning

5.1 Introduction

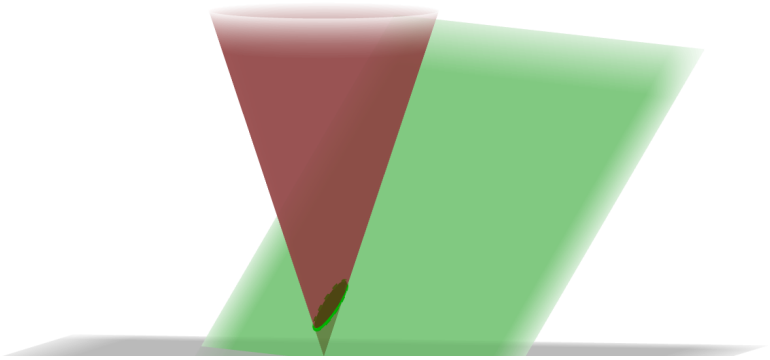
In Chapter 4, we proposed a novel tip model for a degradable tool. We solved an optimization problem using this model to track a desired deposition profile while iteratively learning model parameters for the tip. By being able to rotate the tool and leveraging the different widths of the elliptical surface, the robot was able to perform better than the current standard where the tool was always held perpendicular. One problem identified in further iterations was that eventually the tip wore away enough that the line widths would end up exceeding the desired width regardless of the orientation. In this chapter, we will present a new algorithm which allows the robot to decrease the tip width again and return to sharper line strokes.

5.2 Problem Formulation

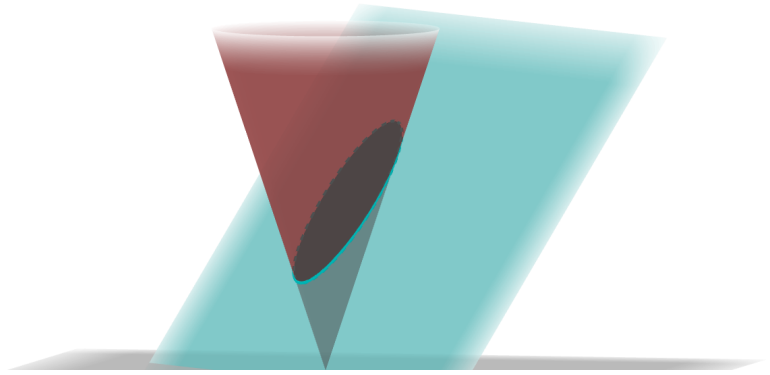
In this section, we discuss modifications to the tip model introduced in 4.3.

Tip Wear

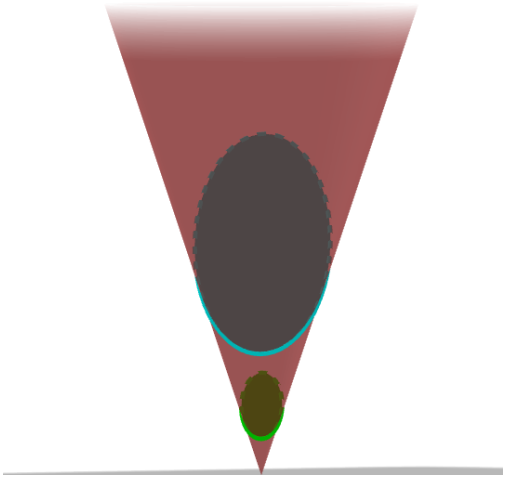
From (4.8), we can see that the major and minor axes lengths are linearly scaling with the degradation depth, d. When the tool is used at a fixed pitch angle, γ, it will continue to wear away which means that d is monotonically increasing. This is illustrated in Fig. 5.1. For some desired deposition widths, it means that even the minor axis length will grow too large and no amount of rotation ψ will produce a deposition profile that matches the desired width. One solution would be to resharpen the tool again but that would be inefficient and waste material. We propose a solution which allows the same tip to decrease the deposition width again, thus prolonging the tool's lifespan.



(a) $d = 0.001$m



(b) $d = 0.004$m



(c) Front view

Figure 5.1: Rendering the tool tip surface at different levels of d.

Adjusted Tip Model

We will now consider the tip in the 2D case where the two edges are represented by the line segments, s_0 and s_1, shown in Fig 5.2. The states used to describe each of these segments are:

$$s_i = \begin{bmatrix} a_i \\ d_i \\ x_{l,i} \\ x_{r,i} \end{bmatrix}, \tag{5.1}$$

where the a_i is the slope, d_i is the z-intercept, and $\{x_{l,i}, x_{r,i}\}$ are the x-coordinates of the boundaries for each line segment. Our goal is to find another segment with a desired length, l_{des}, which intersects the starting tip lines. This will model the surface of the tool tip which comes into contact with the workpiece and leaves a deposition profile when moving in the y-direction.

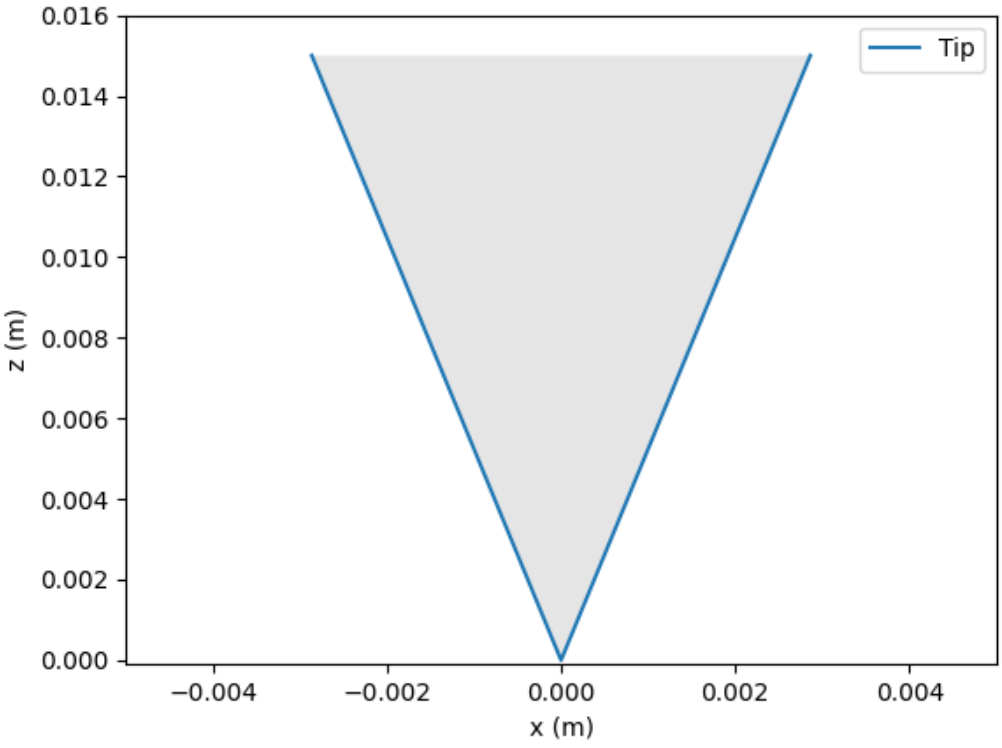


Figure 5.2: 2D rendering of the tool tip.

At any given time, we will store the line parameters in an array:

$$\mathcal{S}(t) = \left[s_0, s_1, ..., s_m\right] \tag{5.2}$$

where $m-2$ is the the total number of successive cuts into the tip.

5.3 Edge Planning

In this section, we will formulate an optimization problem to find the new line segment described in 5.2 and use it in an edge planning algorithm.

Optimization Problem Formulation

There are several constraints which must be considered when picking a new surface line. The slope of the new line is limited by the slopes of the tip edges or it can be further restricted depending on the shape of the robot's end effector to prevent collisions with the surface. The line segment must also never exceed a certain z-value, d_{max}, which represents the limit of the exposed tip surface (e.g. lead of a sharpened pencil). Using these constraints, we construct the following optimization problem:

$$\begin{aligned} \min_{a,d_m,x_m,z_m} \quad & d + (a - a_{ref})^2 \\ \text{s.t.,} \quad & z_{m,i} = a x_{m,i} + d \\ & z_{m,i} \leq d_{max} \\ & x_{l,i} \leq x_{m,i} \leq x_{r,i} \\ & 0.0001 \leq d \leq d_{max} \\ & |a| \leq m(1 - \tfrac{d}{d_{max}}) \\ & a_0 \leq a \leq a_1 \\ & (x_{m,j} - x_{m,k})^2 + (z_{m,j} - z_{m,2k})^2 = l_{des}^2 \\ & \forall i \in \{j, k\} \end{aligned} \tag{5.3}$$

where $\{j, k\}$ are any two lines selected from $\mathcal{S}(t)$, and a_{ref} is a reference slope which may be chosen such that the new line is further away from previous cuts. After solving (5.3), we obtain the parameters for a new line segment which matches the desired deposition width.

Edge Planning Algorithm

We start with the initial tip composed of two line segments. The parameters of the next edge with a desired deposition width is obtained by solving (5.3). The robot performs the strokes using the optimal tilt angle until the desired width is exceeded. The parameters, corrected by visual estimation, are then added to the collection $\mathcal{S}(t)$. Each line is a successive cut into the tip at varying angles and as this number increases, we solve (5.3) on a pair-wise basis. For each pair of lines in $\mathcal{S}(t)$, we compute the best new edge and compile the candidates into a list $\mathcal{C}(t)$. The best candidate which is furthest from the previous line segments is chosen.

To reduce the number of operations, we first prune the pairs of lines which are too close or too far from each other to have any connecting segments of the desired width. As new lines are cut deeper into the tip, they will also eventually remove older lines entirely which we take out of $\mathcal{S}(t)$.

Algorithm 1 Edge Planning Algorithm

Parameters: l_{des}, d_{max}
Input: $\mathcal{S}(t)$
Output: s_m

for each iteration t **do**
  $\mathcal{C}(t) = [\,]$
  for each pair $\{s_j, s_k\} \in \mathcal{S}(t), j \neq k$ **do**
    if valid pair **then**
      $s_{jk} = a_{jk}, d_{jk}, x_{l,jk}, x_{r,jk} \leftarrow$ solve from (5.3)
      append s_{jk} to $\mathcal{C}(t)$
    end if
  end for
  select best s_m from $\mathcal{C}(t)$
  append s_m to $\mathcal{S}(t)$
  prune $\mathcal{S}(t)$
end for

5.4 Experimental Results

To evaluate our approach in a similar manner to the previous Chapter 4 and use the same setup shown in 4.1. Robot will perform strokes with $l_{des} = 1mm$ as shown in Fig. 5.3. We use Algorithm 1 to find the parameters for the next edge. A rendering of the tip and new line s_m is shown in Fig. 5.4. Using a_m, the robot will adjust its tilt angle γ to perform a stroke. After each stroke, we can use the same vision-based approach to estimate the deposition width. Once that desired width has been exceeded, we update $\mathcal{S}(t)$ with a_m and the actual $d_m, x_{l,m}, x_{r,m}$ calculated from the actual width. This is when the next iteration begins. Figures 5.5-5.8 show the 2D renderings for iterations 2 through 100. As the iterations progress and the tip wears away more, we see that the previously stored lines also get pruned away which improves the efficiency of our algorithm.

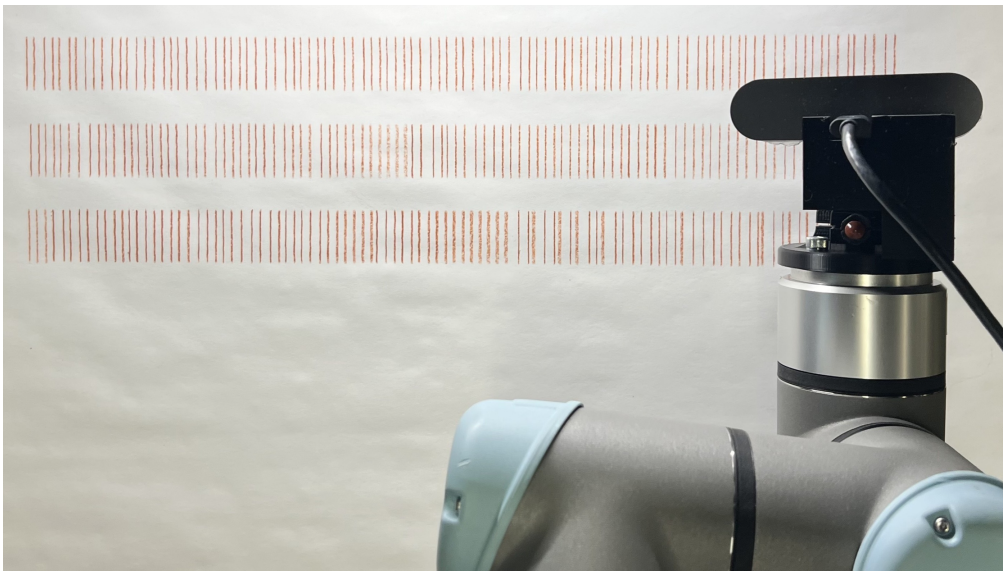


Figure 5.3: Overhead view of Ur5e robot performing line strokes.

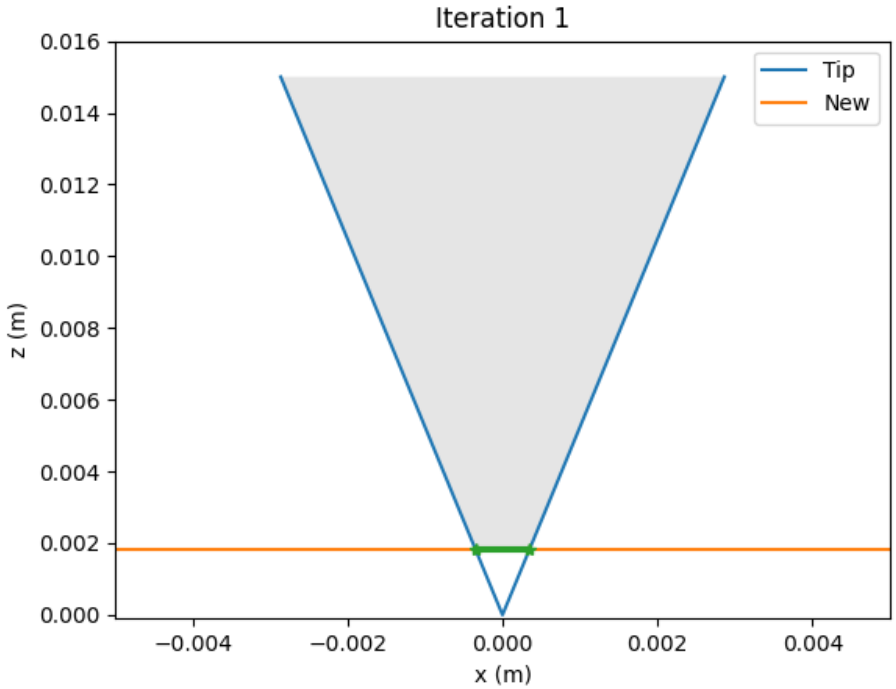
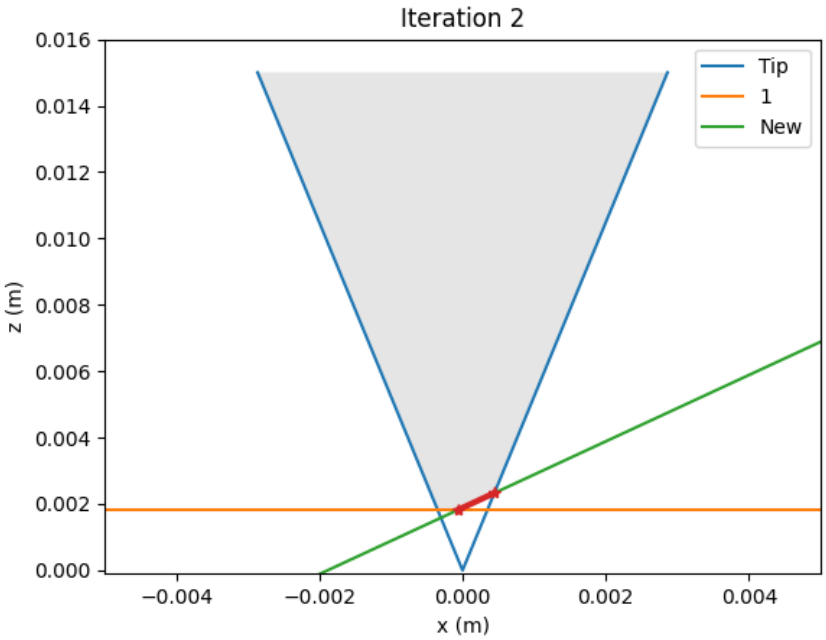
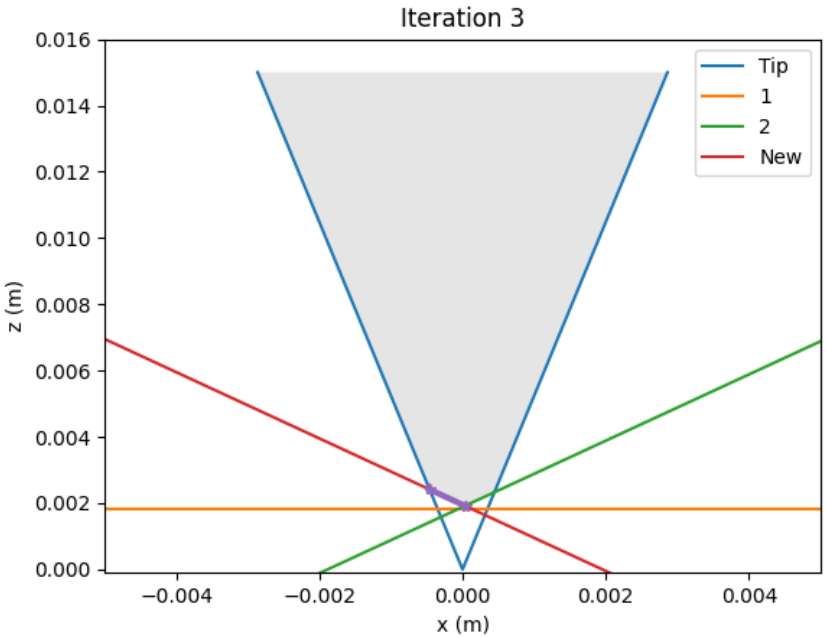


Figure 5.4: 2D Rendering of the tip with new edge line.

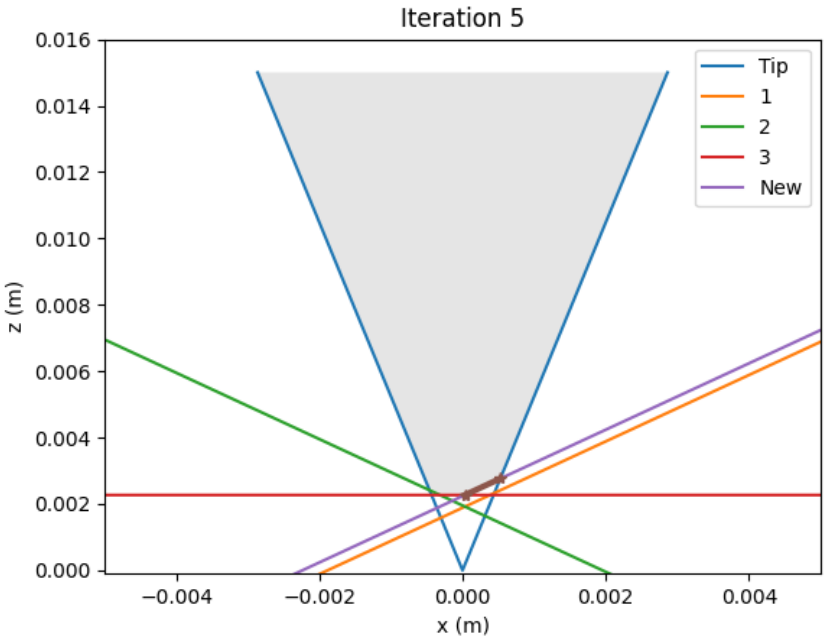


(a) Iteration = 2

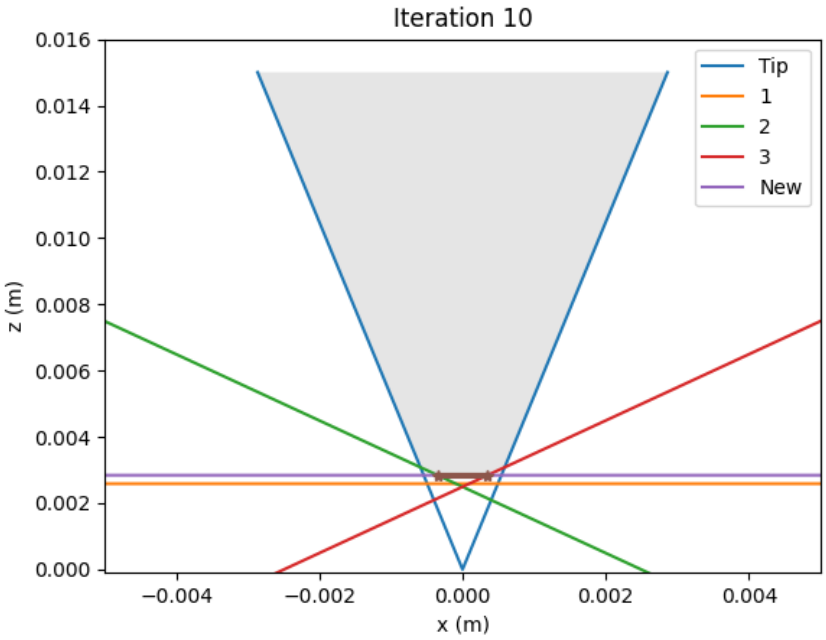


(b) Iteration = 3

Figure 5.5: Successive cuts at Iterations 2 and 3

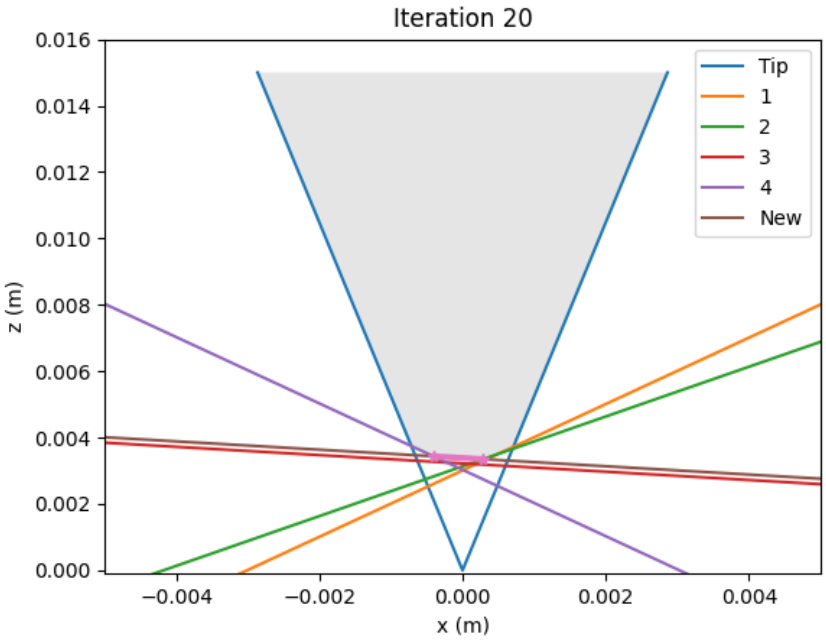


(a) Iteration = 5

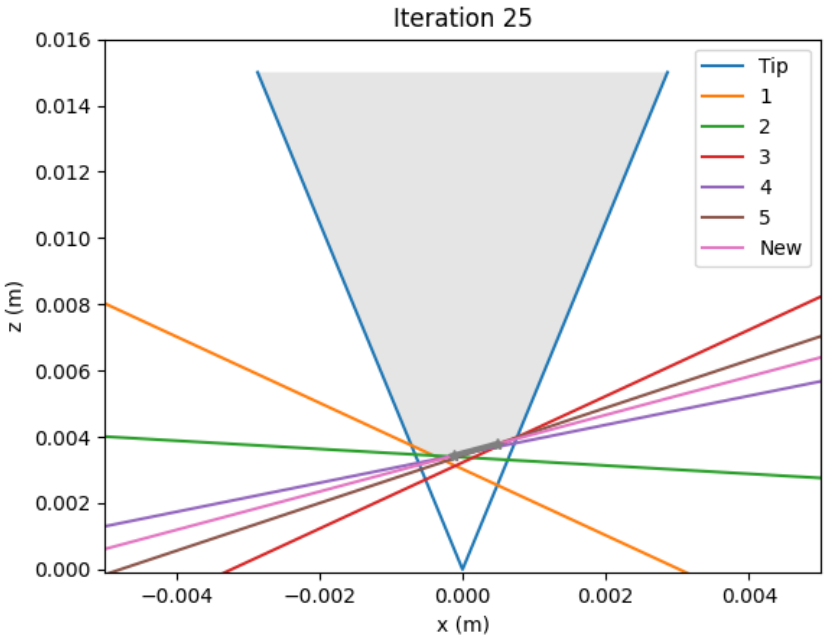


(b) Iteration = 10

Figure 5.6: Successive cuts at Iterations 5 and 10

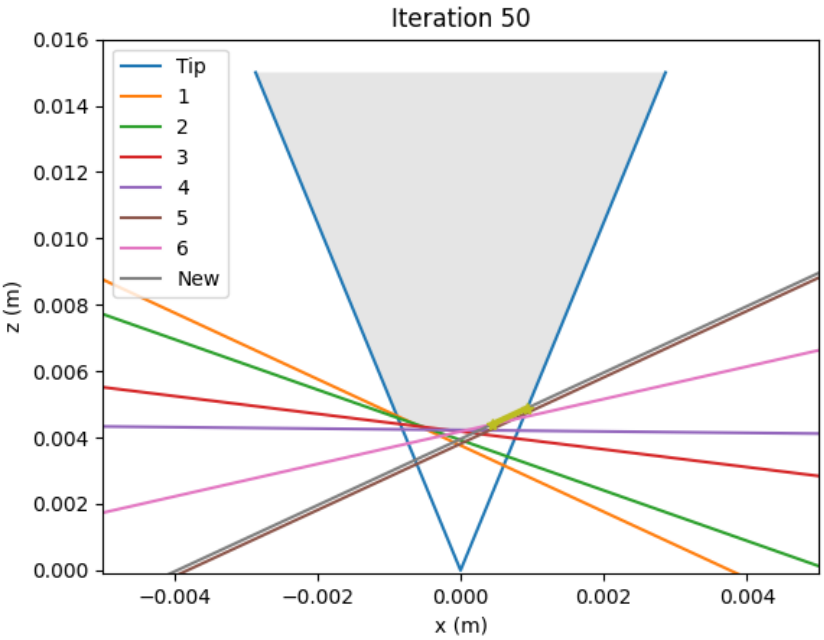


(a) Iteration = 20

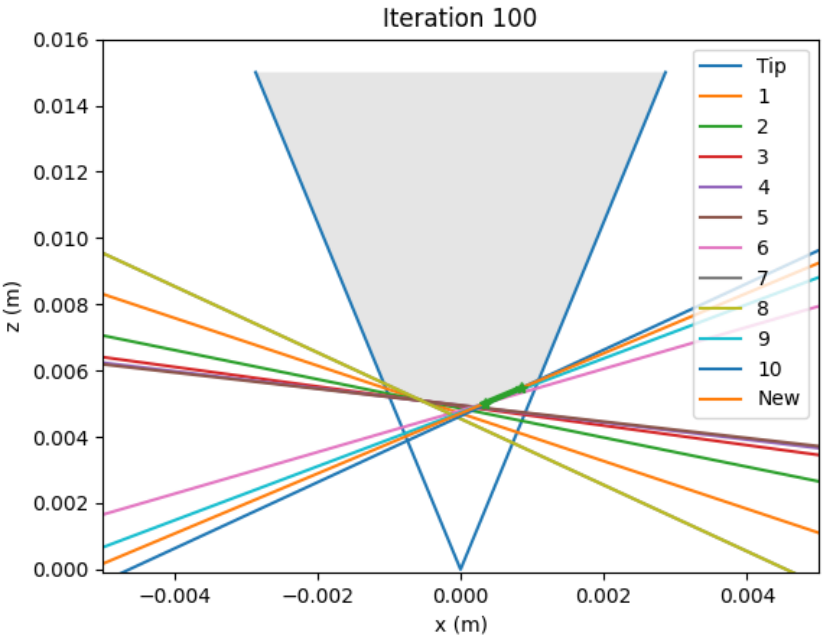


(b) Iteration = 25

Figure 5.7: Successive cuts at Iterations 20 and 25



(a) Iteration = 50



(b) Iteration = 100

Figure 5.8: Successive cuts at Iterations 50 and 100

Fig. 5.9 shows the view from the camera mounted on the robot after performing 250 strokes. We see that deposition width increases and decreases between successive strokes even though the tool tip has not been resharpened. In contrast, Fig. 5.10 shows the results when there isn't edge planning. The deposition width is only increasing.

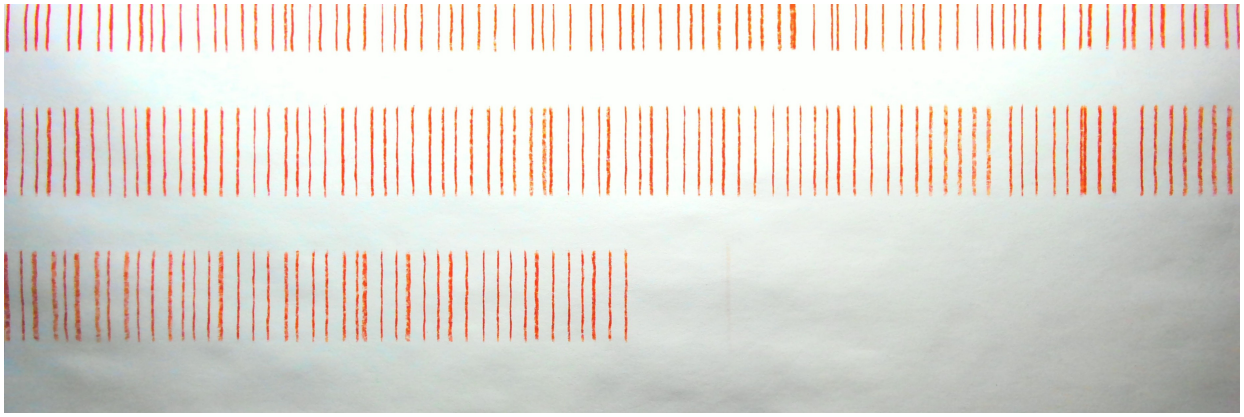


Figure 5.9: Camera view from the robot performing line strokes with edge planning.

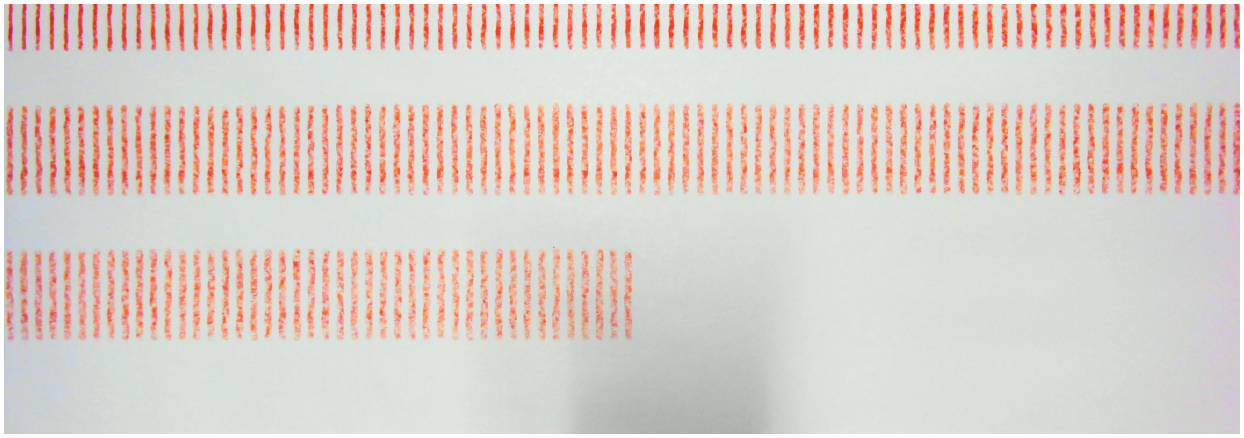


Figure 5.10: Camera view from the robot performing line strokes without edge planning.

These trends are confirmed by the width analysis shown in Fig. 5.11 which is obtained from width estimates using the camera mounted on the robot. In trials A and B, where the pencil was held at a fixed angle perpendicular to the surface, the width increases with each stroke. The values ranged from 1.25mm to 2.3mm. Some variation is due to noise from the camera estimation. In trials C and D, the edge planning algorithm allowed the robot to decrease the deposition width, hovering near the desired width of 1mm and never exceeding 2mm. Some model mismatch arising from the width estimation when updating $\mathcal{S}(t)$ may

have caused the new cut into the tip to be on a surface that was already wider. The error analysis computing from finding the difference between the actual and desired deposition widths is shown in Fig. 5.12. The average error in trial A was 1.41mm, trial B was 1.24mm, trial C was 0.200mm, and trial D was 0.188mm. Our approach was able to allow the robot to perform finer strokes without needing resharpen the tip, showing an improvement of over 80% in width-tracking.

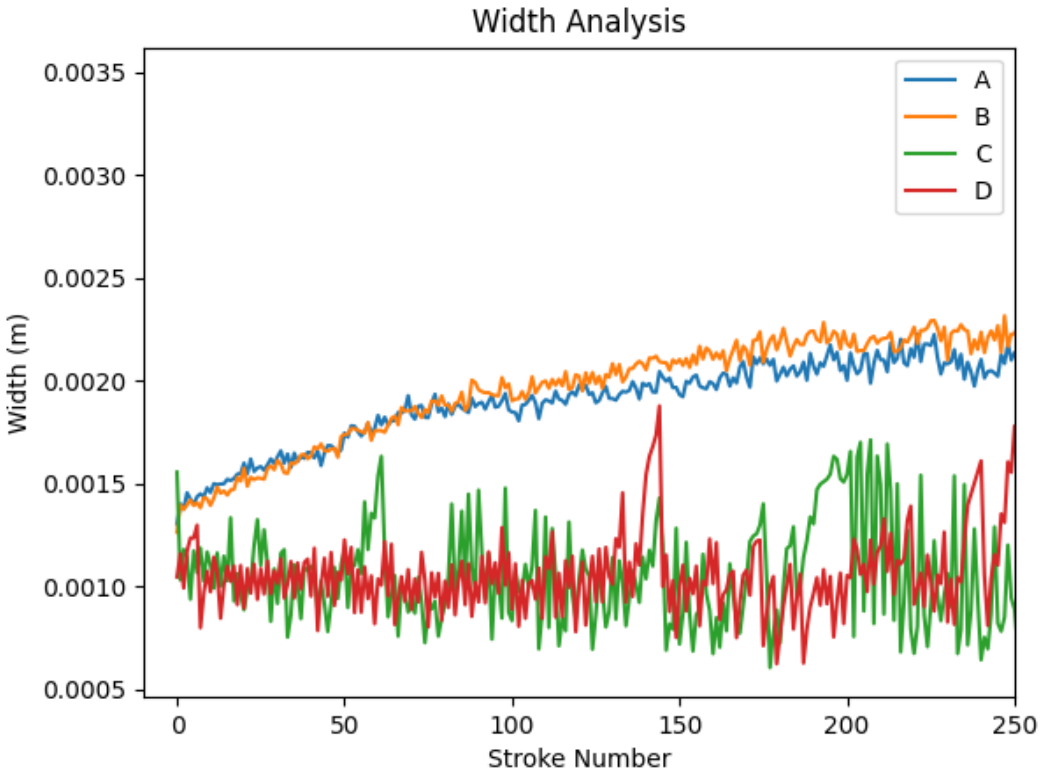


Figure 5.11: Width analysis from robot performing pencil strokes. Trials A and B represent trials without edge planning. Trials C and D represent trials with edge planning.

Pressure Control

Another important factor when using a tool is the pressure applied during the interaction. Constant force can lead to poor results when the tip surface gets larger since the pressure actually decreases. When using a pencil, this means that the deposition will be lighter with usage instead of a consistent dark profile. We test this by using the robot to perform 800 strokes with the pencil held perpendicular. We see from Fig 5.15 that after stroke number 500, the brightness starts diverge. By maintaining constant pressure, the robot is able to

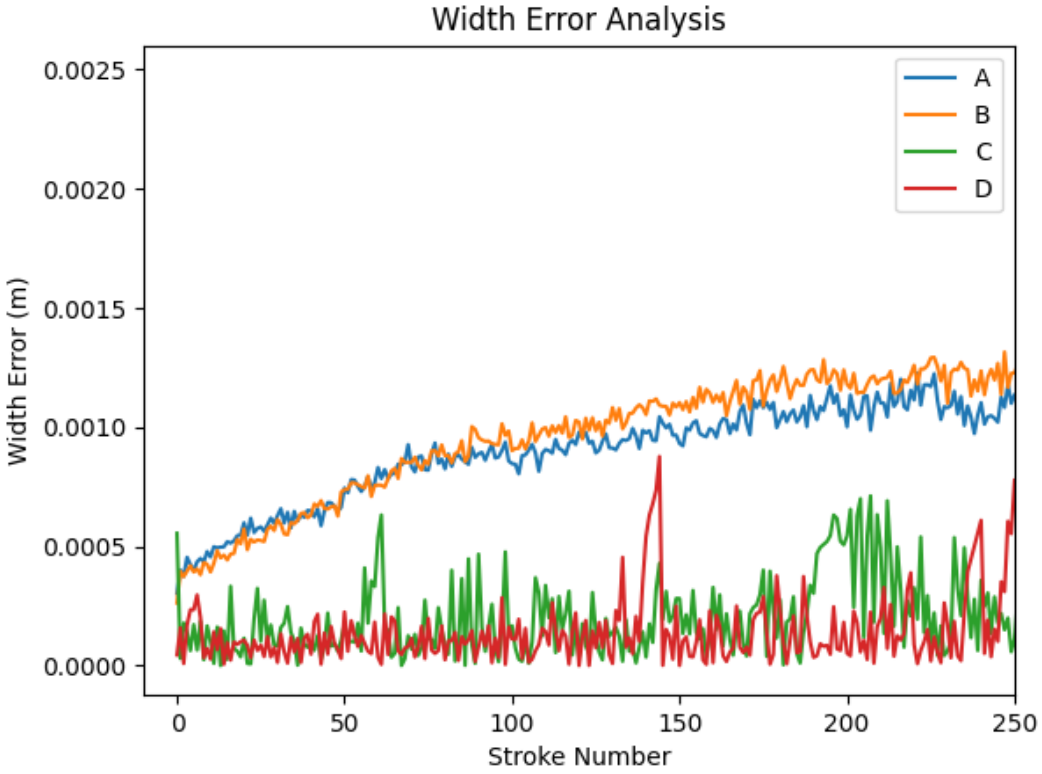


Figure 5.12: Error analysis from robot performing pencil strokes. Trials A and B represent trials without edge planning. Trials C and D represent trials with edge planning.

stabilize around a grayscale value of 180 whereas deposition from the constant force trials continued to get brighter. This highlights the importance of tracking the shape of the tip surface.

5.5 Conclusion

We presented an model-based approach for controlling the edge width for a degradable tool. We solve an optimization problem to find the best tilt angle to achieve a desired width. This was utilized in our edge planning algorithm which allowed the robot to perform finer strokes without needing to resharpen the tool tip. In hardware experiments conducted on a UR5e robot, we showed that our approach decreased the error by over 80%.

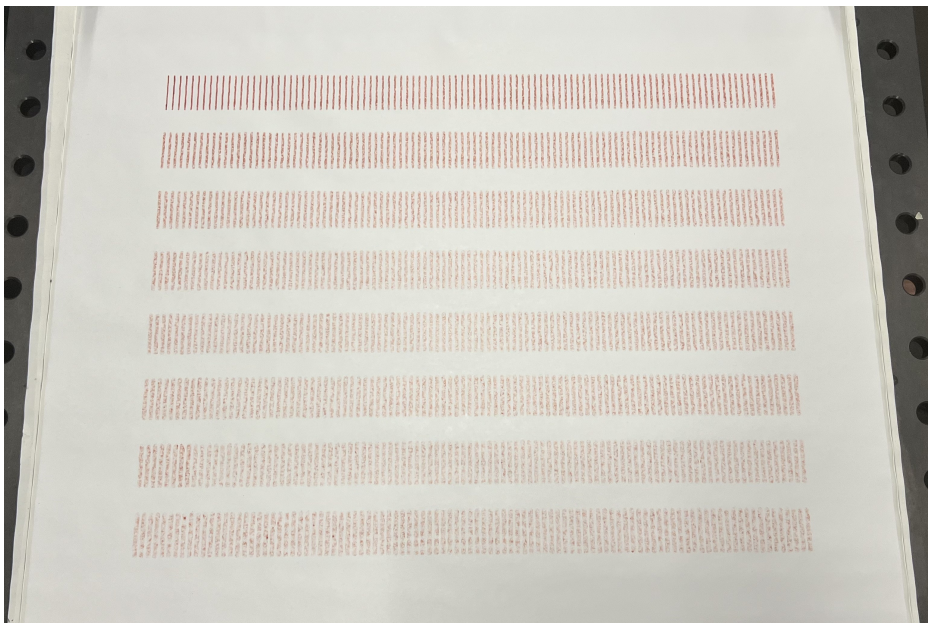


Figure 5.13: Pencil strokes performed with constant force.

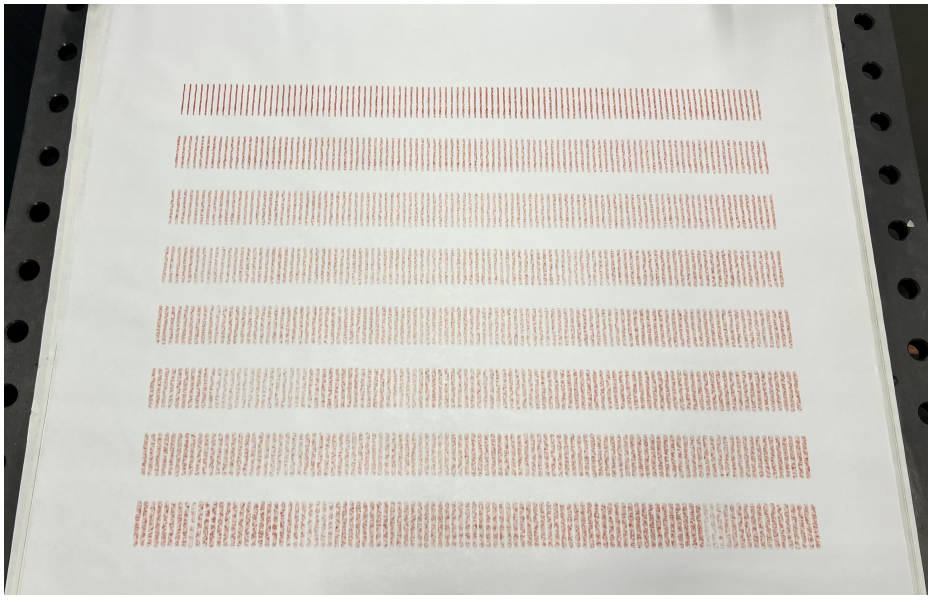


Figure 5.14: Pencil strokes performed with constant pressure.

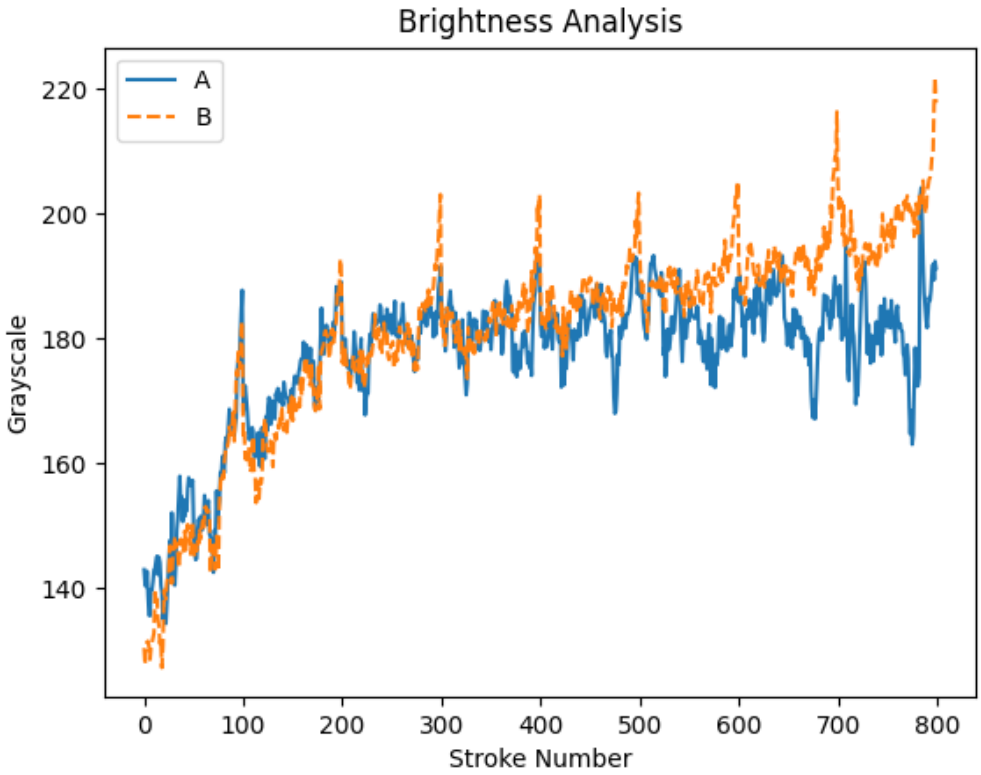


Figure 5.15: Brightness analysis from robot performing pencil strokes with constant force or constant pressure.

Chapter 6

Conclusion and Future Directions

As robotic hardware continues to improve and become more affordable, we will certainly see their integration into society further increase beyond commercial applications. Robots will be equipped with more advanced tools and interact with the world in a more complex manner. In this dissertation, we explored model-based approaches to tackle challenging scenarios for robotic manipulators. While the methods in this dissertation where tested on hardware experiments and showed promising results, there are always improvements that could be made due to underlying assumptions and simplifications. The following are some areas that could be explored:

- In Chapter 2, a mixed open-loop and closed-loop control strategy was used. As is the nature of executing open-loop sequences, it resulted in variation during the swing-up phase and sometimes the ball ended up in positions which could not be caught. Closing the loop on the swing-up could result in more consistent results. Furthermore, a fully hybrid control strategy that plans the entire swing and catch trajectory in one shot could produce more dynamic maneuvers. Another interesting application for learning-based controllers would be to switch the string material or ball weight between experiments and trying to adapt based on the deviation between expected and actual ball movement. Lastly, a straightforward extension would be to increase the dimension of the model outside of the planar case.

- In Chapter 3, we assumed the human's planned actions mirrored the robot's computed optimal actions and the robot would compensate when the human deviated from the expected movement. A better model for the human behavior could improve the performance of the robot and result in more fluid role interactions. Investigating the task in an environment with dynamic obstacles could bring robotics a step closer towards collaborative transportation in industrial settings.

- In Chapter 4, we developed a degradation model for a conical pencil tip and compared the performance using line strokes. It would interesting to use that model as an input for an image-to-stroke decomposition algorithm so that varied line width could be

utilized for shading or outline regions. Similar to changing the string or ball parameters for the cup-and-ball game, different pencil types with varying hardness and shapes could be utilized for a learning-based deposition controller.

- In Chapter 5, the tip model was in 2D and treated the surface as a single line instead of the ellipse in Chapter 4. As more cuts are added into the tip, the surface in reality may even be parabolic or rectangular if limited to rotations about the y-axis. An extension to this work could explore the different shapes that arise from planar cuts into the conical shape and optimize over 3D rotations.

Bibliography

[1] Zhihao Liu, Quan Liu, Wenjun Xu, Lihui Wang, and Zude Zhou. "Robot learning towards smart robotic manufacturing: A review". In: *Robotics and Computer-Integrated Manufacturing* 77 (2022), p. 102360. ISSN: 0736-5845. DOI: https://doi.org/10.1016/j.rcim.2022.102360. URL: https://www.sciencedirect.com/science/article/pii/S0736584522000485.

[2] Chris Lytridis, Vassilis G Kaburlasos, Theodore Pachidis, Michalis Manios, Eleni Vrochidou, Theofanis Kalampokas, and Stamatis Chatzistamatis. "An overview of cooperative robotics in agriculture". In: *Agronomy* 11.9 (2021), p. 1818.

[3] Manas Wakchaure, BK Patle, and AK Mahindrakar. "Application of AI techniques and robotics in agriculture: A review". In: *Artificial Intelligence in the Life Sciences* 3 (2023), p. 100057.

[4] I Karabegović, E Karabegović, M Mahmić, and EJAIPE Husak. "The application of service robots for logistics in manufacturing processes." In: *Advances in Production Engineering & Management* 10.4 (2015).

[5] Amy M. Schuster, Shubham Agrawal, Noah Britt, Danielle Sperry, Jenna A. Van Fossen, Sicheng Wang, Elizabeth A. Mack, Jessica Liberman, and Shelia R. Cotten. "Will automated vehicles solve the truck driver shortages? Perspectives from the trucking industry". In: *Technology in Society* 74 (2023), p. 102313. ISSN: 0160-791X. DOI: https://doi.org/10.1016/j.techsoc.2023.102313. URL: https://www.sciencedirect.com/science/article/pii/S0160791X23001185.

[6] Robert Bogue. "Robots addressing agricultural labour shortages and environmental issues". In: *Industrial Robot: the international journal of robotics research and application* 51.1 (2024), pp. 1–6.

[7] Erol Akkoc, Jan Fritz, and Hoi Cheung Zhang. "6-Minute turbo spin echo MRI of the elbow using combined simultaneous multi-slice and parallel imaging acceleration". In: *Proc Intl Soc Mag Reson Med*. Vol. 30. 2022.

[8] Edward L. Zhu and Francesco Borrelli. *A Sequential Quadratic Programming Approach to the Solution of Open-Loop Generalized Nash Equilibria for Autonomous Racing*. 2024. arXiv: 2404.00186 [cs.RO]. URL: https://arxiv.org/abs/2404.00186.

[9] Magnus Gaertner, Marko Bjelonic, Farbod Farshidian, and Marco Hutter. *Collision-Free MPC for Legged Robots in Static and Dynamic Scenes*. 2021. arXiv: 2103.13987 [cs.RO]. URL: https://arxiv.org/abs/2103.13987.

[10] Chenyu Yang, Bike Zhang, Jun Zeng, Ayush Agrawal, and Koushil Sreenath. "Dynamic Legged Manipulation of a Ball Through Multi-Contact Optimization". In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2020, pp. 7513–7520. DOI: 10.1109/IROS45743.2020.9341218.

[11] NPR. *Flippy, The Fast-Food Robot (Sort Of), Mans The Grill At CaliBurger*. Accessed: 2024-11-29. 2018. URL: https://www.npr.org/sections/thetwo-way/2018/03/05/590884388/flippy-the-fast-food-robot-sort-of-mans-the-grill-at-caliburger.

[12] Fanuc America. *Automotive Robots*. Accessed: 2024-11-29. 2024. URL: https://www.fanucamerica.com/solutions/industries/automotive-robots.

[13] Ilija Radosavovic, Tete Xiao, Bike Zhang, Trevor Darrell, Jitendra Malik, and Koushil Sreenath. "Real-world humanoid locomotion with reinforcement learning". In: *Science Robotics* 9.89 (2024), eadi9579. DOI: 10.1126/scirobotics.adi9579. eprint: https://www.science.org/doi/pdf/10.1126/scirobotics.adi9579. URL: https://www.science.org/doi/abs/10.1126/scirobotics.adi9579.

[14] B. Nemec and A. Ude. "Reinforcement learning of ball-in-a-cup playing robot". In: *2011 IEEE International Conference on Robotics and Biomimetics*. Dec. 2011, pp. 2682–2987.

[15] J. Kober and J. Peters. "Learning motor primitives for robotics". In: *2009 IEEE International Conference on Robotics and Automation*. May 2009, pp. 2112–2118.

[16] Hiroyuki Miyamoto, Stefan Schaal, Francesca Gandolfo, Hiroaki Gomi, Yasuharu Koike, Rieko Osu, Eri Nakano, Yasuhiro Wada, and Mitsuo Kawato. "A Kendama Learning Robot Based on Bi-directional Theory". In: *Neural Networks* 9.8 (1996), pp. 1281–1302.

[17] T. Sakaguchi and F. Miyazaki. "Dynamic manipulation of ball-in-cup game". In: *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*. May 1994, 2941–2948 vol.4.

[18] A. Namiki and N. Itoi. "Ball catching in kendama game by estimating grasp conditions based on a high-speed vision system and tactile sensors". In: *2014 IEEE-RAS International Conference on Humanoid Robots*. Nov. 2014, pp. 634–639.

[19] Devin Schwab et al. "Simultaneously learning vision and feature-based control policies for real-world ball-in-a-cup". In: *arXiv preprint arXiv:1902.04706* (2019).

[20] T. Senoo, A. Namiki, and M. Ishikawa. "Ball control in high-speed batting motion using hybrid trajectory generator". In: *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* May 2006, pp. 1762–1767.

[21] Shidi Li. "Robot Playing Kendama with Model-Based and Model-Free Reinforcement Learning". In: *arXiv preprint arXiv:2003.06751* (2020).

[22] Eric A. Hansen, Andrew G. Barto, and Shlomo Zilberstein. "Reinforcement Learning for Mixed Open-loop and Closed-loop Control". In: *NIPS*. 1996.

[23] C. G. Atkeson and S. Schaal. "Learning tasks from a single demonstration". In: *Proceedings of International Conference on Robotics and Automation*. Vol. 2. 1997, pp. 1706–1712.

[24] J. Z. Kolter, C. Plagemann, D. T. Jackson, A. Y. Ng, and S. Thrun. "A probabilistic approach to mixed open-loop and closed-loop control, with application to extreme autonomous driving". In: *2010 IEEE International Conference on Robotics and Automation*. 2010, pp. 839–845.

[25] Richard M Murray, Zexiang Li, and S Shankar Sastry. *A mathematical introduction to robotic manipulation*. CRC press, 1994.

[26] Monimoy Bujarbaruah, Akhil Shetty, Kameshwar Poolla, and Francesco Borrelli. "Learning Robustness with Bounded Failure: An Iterative MPC Approach". In: *arXiv preprint arXiv:1911.09910* (2019).

[27] Anna-Lisa Vollmer and Nikolas J Hemion. "A user study on robot skill learning without a cost function: Optimization of dynamic movement primitives via naive user feedback". In: *Frontiers in Robotics and AI* 5 (2018), p. 77.

[28] B. Nemec, M. Zorko, and L. Žlajpah. "Learning of a ball-in-a-cup playing robot". In: *19th International Workshop on Robotics in Alpe-Adria-Danube Region (RAAD 2010)*. June 2010, pp. 297–301.

[29] Pierre Merriaux, Yohan Dupuis, Rémi Boutteau, Pascal Vasseur, and Xavier Savatier. "A study of vicon system positioning performance". In: *Sensors* 17.7 (2017), p. 1591.

[30] Universal Robots. "e-Series from Universal Robots". In: *https://www.universal-robots.com/media/1802432/e-series-brochure.pdf* (2014).

[31] Marko Tanaskovic, Lorenzo Fagiano, Roy Smith, and Manfred Morari. "Adaptive receding horizon control for constrained MIMO systems". In: *Automatica* 50.12 (2014), pp. 3019–3029.

[32] X. Lu and M. Cannon. "Robust Adaptive Tube Model Predictive Control". In: *2019 IEEE American Control Conference (ACC)*. IEEE. July 2019, pp. 3695–3701.

[33] Tony Zheng*, Monimoy Bujarbaruah*, Akhil Shetty, Martin Sehr, and Francesco Borrelli. "Learning to Play Cup-and-Ball with Noisy Camera Observations". In: *IEEE International Conference on Automation Science and Engineering CASE*. Aug. 2020, pp. 372–377.

[34] Emanuel Todorov, Tom Erez, and Yuval Tassa. "Mujoco: A physics engine for model-based control". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2012, pp. 5026–5033.

[35] Yuval Tassa et al. "dm_control: Software and Tasks for Continuous Control". In: *arXiv preprint arXiv:2006.12983* (2020).

[36] Andrea Bauer, Dirk Wollherr, and Martin Buss. "Human–robot collaboration: a survey". In: *International Journal of Humanoid Robotics* 5.01 (2008), pp. 47–66.

[37] Nathanael Jarrasse, Vittorio Sanguineti, and Etienne Burdet. "Slaves no longer: review on role assignment for human–robot joint motor action". In: *Adaptive Behavior* 22.1 (2014), pp. 70–82.

[38] Oussama Khatib. "Mobile manipulation: The robotic assistant". In: *Robotics and Autonomous Systems* 26.2-3 (1999), pp. 175–183.

[39] Kazuhiro Kosuge and Yasuhisa Hirata. "Human-robot interaction". In: *2004 IEEE International Conference on Robotics and Biomimetics*. IEEE. 2004, pp. 8–11.

[40] Yusuke Maeda, Takayuki Hara, and Tamio Arai. "Human-robot cooperative manipulation with motion estimation". In: *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No. 01CH37180)*. Vol. 4. Ieee. 2001, pp. 2240–2245.

[41] L Beton, P Hughes, S Barker, M Pilling, L Fuente, and NT Crook. "Leader-follower strategies for robot-human collaboration". In: *A World with Robots*. Springer, 2017, pp. 145–158.

[42] Anca D Dragan and Siddhartha S Srinivasa. "A policy-blending formalism for shared control". In: *The International Journal of Robotics Research* 32.7 (2013), pp. 790–805.

[43] Paul Evrard and Abderrahmane Kheddar. "Homotopy switching model for dyad haptic interaction in physical collaborative tasks". In: *World Haptics 2009-Third Joint EuroHaptics conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*. IEEE. 2009, pp. 45–50.

[44] S Ozgur Oguz, Ayse Kucukyilmaz, Tevfik Metin Sezgin, and Cagatay Basdogan. "Haptic negotiation and role exchange for collaboration in virtual environments". In: *2010 IEEE haptics symposium*. IEEE. 2010, pp. 371–378.

[45] Alexander Mörtl, Martin Lawitzky, Ayse Kucukyilmaz, Metin Sezgin, Cagatay Basdogan, and Sandra Hirche. "The role of roles: Physical cooperation between humans and robots". In: *The International Journal of Robotics Research* 31.13 (2012), pp. 1656–1674.

[46] Behzad Sadrfaridpour, Maziar Fooladi Mahani, Zhanrui Liao, and Yue Wang. "Trust-based impedance control strategy for human-robot cooperative manipulation". In: *Dynamic Systems and Control Conference*. Vol. 51890. American Society of Mechanical Engineers. 2018, V001T04A015.

[47] Jim Mainprice and Dmitry Berenson. "Human-robot collaborative manipulation planning using early prediction of human motion". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2013, pp. 299–306.

[48] Jaime F Fisac, Andrea Bajcsy, Sylvia L Herbert, David Fridovich-Keil, Steven Wang, Claire J Tomlin, and Anca D Dragan. "Probabilistically safe robot planning with confidence-based human predictions". In: *arXiv preprint arXiv:1806.00109* (2018).

[49] Andrea Bajcsy, Dylan P Losey, Marcia K O'Malley, and Anca D Dragan. "Learning robot objectives from physical human interaction". In: *Conference on Robot Learning*. PMLR. 2017, pp. 217–226.

[50] Xinbo Yu, Yanan Li, Shuang Zhang, Chengqian Xue, and Yu Wang. "Estimation of human impedance and motion intention for constrained human–robot interaction". In: *Neurocomputing* 390 (2020), pp. 268–279.

[51] Fabrizio Flacco, Torsten Kröger, Alessandro De Luca, and Oussama Khatib. "A depth space approach to human-robot collision avoidance". In: *2012 IEEE International Conference on Robotics and Automation*. IEEE. 2012, pp. 338–345.

[52] Lihui Wang, Bernard Schmidt, and Andrew YC Nee. "Vision-guided active collision avoidance for human-robot collaborations". In: *Manufacturing Letters* 1.1 (2013), pp. 5–8.

[53] P. Michael Plesha, G. Gray, and P. Francesco Costanzo. *Engineering Mechanics: Statics and Dynamics*. McGraw-Hill Education, 2012. ISBN: 9780073380315. URL: https://books.google.com/books?id=o01QtQAACAAJ.

[54] Tony Zheng, Monimoy Bujarbaruah, Yvonne R Stürz, and Francesco Borrelli. "Safe Human-Robot Collaborative Transportation via Trust-Driven Role Adaptation". In: *arXiv preprint arXiv:2207.05896* (2022).

[55] Dexter RR Scobee, Vicenc Rubies Royo, Claire J Tomlin, and S Shankar Sastry. "Haptic assistance via inverse reinforcement learning". In: *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE. 2018, pp. 1510–1517.

[56] Monimoy Bujarbaruah, Yvonne R. Stürz, Conrad Holda, Karl H. Johansson, and Francesco Borrelli. "Learning Environment Constraints in Collaborative Robotics: A Decentralized Leader-Follower Approach". In: *International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2021, pp. 1636–1641.

[57] Markku Suomalainen, Yiannis Karayiannidis, and Ville Kyrki. "A survey of robot manipulation in contact". In: *Robotics and Autonomous Systems* 156 (2022), pp. 1–57. ISSN: 09218890. DOI: 10.1016/j.robot.2022.104224. arXiv: 2112.01942.

[58] Yahav Avigal, Lars Berscheid, Tamim Asfour, Torsten Kröger, and Ken Goldberg. *SpeedFolding: Learning Efficient Bimanual Folding of Garments*. 2022. arXiv: 2208.10552 [cs.RO].

[59] Daniel Leidner, Wissam Bejjani, Alin Albu-Schäffer, and Michael Beetz. "Robotic agents representing, reasoning, and executing wiping tasks for daily household chores". In: *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS* Aamas (2016), pp. 1006–1014. ISSN: 15582914.

[60] Frederik Ebert, Yanlai Yang, Karl Schmeckpeper, Bernadette Bucher, Georgios Georgakis, Kostas Daniilidis, Chelsea Finn, and Sergey Levine. "Bridge Data: Boosting Generalization of Robotic Skills with Cross-Domain Datasets". In: (2022), pp. 2–8. DOI: 10.15607/rss.2022.xviii.063. arXiv: 2109.13396.

[61] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. *Zero-Shot Text-to-Image Generation*. 2021. arXiv: 2102.12092 [cs.CV].

[62] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. *Hierarchical Text-Conditional Image Generation with CLIP Latents*. 2022. arXiv: 2204.06125 [cs.CV].

[63] Ruben Villegas, Mohammad Babaeizadeh, Pieter-Jan Kindermans, Hernan Moraldo, Han Zhang, Mohammad Taghi Saffar, Santiago Castro, Julius Kunze, and Dumitru Erhan. *Phenaki: Variable Length Video Generation From Open Domain Textual Description*. 2022. arXiv: 2210.02399 [cs.CV].

[64] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Bjorn Ommer. "High-Resolution Image Synthesis with Latent Diffusion Models". In: (2022), pp. 10674–10685. DOI: 10.1109/cvpr52688.2022.01042. arXiv: 2112.10752.

[65] Chitwan Saharia et al. *Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding*. 2022. arXiv: 2205.11487 [cs.CV].

[66] Tom B. Brown et al. "Language models are few-shot learners". In: *Advances in Neural Information Processing Systems* 2020-December (2020). ISSN: 10495258. arXiv: 2005.14165.

[67] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "ImageNet: A large-scale hierarchical image database". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848.

[68] Zhewei Huang, Shuchang Zhou, and Wen Heng. "Learning to paint with model-based deep reinforcement learning". In: *Proceedings of the IEEE International Conference on Computer Vision* 2019-October (2019), pp. 8708–8717. ISSN: 15505499. DOI: 10.1109/ICCV.2019.00880. arXiv: 1903.04411.

[69] Zhengyan Tong, Xuanhong Chen, Bingbing Ni, and Xiaohang Wang. "Sketch Generation with Drawing Process Guided by Vector Flow and Grayscale". In: *35th AAAI Conference on Artificial Intelligence, AAAI 2021* 1.Dodson (2021), pp. 609–616. ISSN: 2159-5399. DOI: 10.1609/aaai.v35i1.16140. arXiv: 2012.09004.

[70] Shubham Jain, Prashant Gupta, Vikash Kumar, and Kamal Sharma. "A force-controlled portrait drawing robot". In: *Proceedings of the IEEE International Conference on Industrial Technology* 2015-June.June (2015), pp. 3160–3165. DOI: 10.1109/ICIT.2015.7125564.

[71] Atsunobu Kotani and Stefanie Tellex. "Teaching robots to draw". In: *Proceedings - IEEE International Conference on Robotics and Automation* 2019-May (2019), pp. 4797–4803. ISSN: 10504729. DOI: 10.1109/ICRA.2019.8793484.

[72] Ning Xie, Hirotaka Hachiya, and Masashi Sugiyama. "Artist agent: A reinforcement learning approach to automatic stroke generation in oriental ink painting". In: *IEICE Transactions on Information and Systems* E96-D.5 (2013), pp. 1134–1144. ISSN: 17451361. DOI: 10.1587/transinf.E96.D.1134.

[73] Nan Cao, Xin Yan, Yang Shi, and Chaoran Chen. "AI-Sketcher: A deep generative model for producing high-quality sketches". In: *33rd AAAI Conference on Artificial Intelligence, AAAI 2019, 31st Innovative Applications of Artificial Intelligence Conference, IAAI 2019 and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019* (2019), pp. 2564–2571. ISSN: 2159-5399. DOI: 10.1609/aaai.v33i01.33012564.

[74] Sen Wang, Jiaqi Chen, Xuanliang Deng, Seth Hutchinson, and Frank Dellaert. "Robot calligraphy using pseudospectral optimal control in conjunction with a novel dynamic brush model". In: *IEEE International Conference on Intelligent Robots and Systems* (2020), pp. 6696–6703. ISSN: 21530866. DOI: 10.1109/IROS45743.2020.9341787. arXiv: 2003.01565.

[75] Josh H.M. Lam and Yeung Yam. "Stroke trajectory generation experiment for a robotic Chinese calligrapher using a geometric brush footprint model". In: *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009* 1 (2009), pp. 2315–2320. DOI: 10.1109/IROS.2009.5354709.

[76] Ardavan Bidgoli, Manuel Ladron De Guevara, Cinnie Hsiung, Jean Oh, and Eunsu Kang. "Artistic Style in Robotic Painting; A Machine Learning Approach to Learning Brushstroke from Human Artists". In: *29th IEEE International Conference on Robot and Human Interactive Communication, RO-MAN 2020* (2020), pp. 412–418. DOI: 10.1109/RO-MAN47096.2020.9223533. arXiv: 2007.03647.

[77] Fei Gao, Jingjie Zhu, Zeyuan Yu, Peng Li, and Tao Wang. "Making robots draw a vivid portrait in two minutes". In: *IEEE International Conference on Intelligent Robots and Systems* (2020), pp. 9585–9591. ISSN: 21530866. DOI: 10.1109/IROS45743.2020.9340940. arXiv: 2005.05526.

[78] Michal Adamik, Jozef Goga, Jarmila Pavlovicova, Andrej Babinec, and Ivan Sekaj. "Fast robotic pencil drawing based on image evolution by means of genetic algorithm". In: *Robotics and Autonomous Systems* 148 (2022), p. 103912. ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2021.103912.

[79] Daeun Song, Taekhee Lee, and Young J. Kim. "Artistic pen drawing on an arbitrary surface using an impedance-controlled robot". In: *Proceedings - IEEE International Conference on Robotics and Automation* (2018), pp. 4085–4090. ISSN: 10504729. DOI: 10.1109/ICRA.2018.8461084.

[80] Paul J. O'Dowd. "A Robot That Draws and Shades with Tactile Force Feedback Sensed Through a Pencil". In: *EVA*. 2019.