Title
Learning and Planning for Industrial Robotic Manipulation

Author
Jin, Shiyu

Publication Date
2021

Peer reviewed|Thesis/dissertation

Learning and Planning for Industrial Robotic Manipulation

by

Shiyu Jin

A dissertation submitted in partial satisfaction of the

requirements for the degree of

in

Engineering - Mechanical Engineering

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Masayoshi Tomizuka, Chair
Professor Kameshwar Poolla
Professor Pieter Abbeel

Fall 2021

Learning and Planning for Industrial Robotic Manipulation

Abstract

Learning and Planning for Industrial Robotic Manipulation

by

Shiyu Jin

in Engineering - Mechanical Engineering

University of California, Berkeley

Professor Masayoshi Tomizuka, Chair

Industrial robot manipulators are widely deployed in various manufacturing tasks. Compared with human workers, industrial robot manipulators have advantages in terms of precision, efficiency, and repeatability. But it often requires tremendous engineering efforts to set up and program the manipulator for a specific task. The deficiency of intelligence restricts robots from broader applications. Therefore, it becomes more and more important to enable robots to acquire skills that can accomplish complex tasks and generalize across different scenarios. This dissertation aims to develop skill learning and planning methods for industrial robotic manipulation. We study 1) how to learn manipulation skills when there are uncertainties in the object state estimation, 2) how to generalize the manipulation skills across different scenarios, 3) how to achieve high-level task planning for long-horizon manipulation tasks.

Robotic manipulation of both rigid and deformable objects is studied in this dissertation. To manipulate rigid objects, a contact pose identification method is proposed to compensate for the pose uncertainties in the peg-in-hole assembly. In addition to rigid objects, the manipulation of deformable objects is also studied. A tracking and manipulation framework is proposed to robustly estimate the state of the cable and manipulate the cable to desired shapes. For more complex cable manipulation tasks, which often require long-horizon planning, a spatial representation is proposed to model the spatial relationship between the cable and environment fixtures. Multiple manipulation primitives are efficiently learned to configure the cable to desired states. For the task that combines both assembly and deformable object manipulation, a trajectory optimization with complementarity constraints is formulated to model the hybrid dynamics in belt drive units assembly. The problem is solved as a mathematical program with complementarity constraints to obtain feasible and efficient assembly trajectories.

To my parents

Contents

List of Figures

List of Tables

Acknowledgments

How time flies! I still remember the first day I came to Berkeley. I really don't want to leave, but it is time to say goodbye. The past five and half years at UC Berkeley have been an extraordinary period in my life. I am so grateful to meet so many friends and professionals during my Ph.D. study. I will never forget the great journey at Berkeley.

First, I would like to express my deepest appreciation to my advisor Prof. Masayoshi Tomizuka. Thanks for his invaluable advice, continuous support, and patience during my Ph.D. study. His immense knowledge and plentiful experience have encouraged me in all the time of my academic research. I could never finish this dissertation without the guidance and support from him. Beyond the guidance on research, he also gives me a lot of life advice. It is my great honor to have him as my advisor.

I would also like to thank Prof. Kameshwar Poolla and Prof. Pieter Abbeel for serving on my dissertation committee. Prof. Poolla is also on my qualifying exam committee. My major is not Control when I was admitted to Berkeley. In my first semester, I took two classes with Prof. Poolla. His lectures are so interesting. I think I found what I want to study since then. I also took one class with Prof. Abbeel and learned a lot in robotics. My first research project on deformable object tracking was motivated by Prof. Abbeel's work.

I also wish to thank Prof. Roberto Horowitz, Prof. Mark Mueller, and Prof. Jonathan Shewchuk for severing on my qualifying exam committee. The knowledge in Control and Machine Learning that I learned from their classes helped a lot in my research.

Special thanks to FANUC Corporation. Thanks for the generous support throughout these years. I especially want to thank Mr. Tetsuaki Katou, Mr. Kaimeng Wang, and Dr. Jason Tsai. The bi-weekly research discussions with them really give a lot of insight on my research. I also want to thank Mitsubishi Electric Research Laboratories (MERL) and Intrinsic, where I did internship. The results of belt drive units assembly and cable routing in this dissertation are based on the work I did at MERL and Intrinsic. In particular, I would like to thank my host at MERL, Dr. Diego Romeres and my hosts at Intrinsic, Dr. Wenzhao Lian and Dr. Stefan Schaal.

Many thanks to all the members that I have met at Mechanical Systems Control (MSC) Laboratory. Thank Te Tang, Yu Zhao, Hsien-Chung Lin, Yongxiang Fan, Yujiao Cheng, Changhao Wang, Xinghao Zhu, Ting Xu, and Xiang Zhang for the valuable discussion in my research. I also want to thank Linting Sun, Changliu Liu, Wei Zhan, Xiaowen Yu, Shuyang Li, Jianyu Chen, Cheng Peng, Zining Wang, Daisuke Kaneishi, Jiachen Li, Yeping Hu, Chen Tang, Zhuo Xu, Hengbo Ma, Jessica Leu, Kiwoo Shin, Taohan Wang, Lingfeng Sun, Yiyang Zhou, Huidong Gao, Ge Zhang, Zheng Wu, Wu-te Yang, Jinning Li, Catherine Weaver, Saman Fahandezhsaadi, Akio Kodaira, Chenfeng Xu, Ce Hao, Chenran Li, Ran Tian, Yichen Xie, Qiyang Qian, Keita Kobashi, Jen-Wei Wang, and all others.

I also want to thank all the friends I met at Berkeley. Thank you, Jingyu Qiao, Zhipeng Yu, Chen Bao, Zhengyu Zhang, Mo Yang, Nuocheng Xia, Xingle Wang, Haoming Sun, Dong Zhang, Gusty, Xinyan Huang, Xiangyu Wu, Xiaokun Pei, Zirui Zhong, Zhian Kuang, Jianlan Luo, Rui Wang, He Yin, Yu Zhang, Tianyi Li, and Xintian Liu. In addition, I want to thank

Recreational Sports Facility (RSF), I have spent a lot of time and enjoyed playing basketball there.

Finally, I would also like to extend my deepest gratitude to my parents. You are always there for me no matter for the good or bad times. Thank you for everything.

Chapter 1

Introduction

1.1 Background and Motivations

Industrial robots play an important role in the manufacturing production line. They are widely deployed in various manufacturing tasks. Compared with human workers, robots have advantages in terms of precision, efficiency, and repeatability. They can move fast and be precise at the same time. They can work 24 hours a day and do the same task repeatedly without making mistakes. With the help of robots, productivity in manufacturing has been increased significantly, which greatly improves the quality of human life. However, tremendous engineering efforts are often required to set up and program the robot for a specific task. The tasks that robots can accomplish are also limited due to the lack of intelligence. As a result, it is more and more important to enable robots to acquire skills that can accomplish complex tasks and generalize across different scenarios.

Consider a wire harness task in the assembly line, the goal is to manipulate the cables to the desired configurations constrained by several fixtures. This task is easy for human workers, but difficult for robot manipulators to accomplish. The difficulty mainly comes from the unexpected deformation of the flexible objects during the manipulation. This brings challenges in perception, planning, and control. 1) The perception includes the state estimation of the deformable cable in a complex and cluttered environment. 2) The planning includes high-level task planning on what to do in a logical way for the long-horizon wire harness. It also includes low-level reasoning about the kinematic and dynamic constraints of the task. 3) The control includes generating safe and efficient robot actions to grasp and manipulate the cables to the desired configurations. With the recent development in machine learning, optimization, and control theory, now we are able to combine those technologies and develop general methodologies to make the industrial robots deal with the above challenges by learning and acting as intelligent agents.

The objective of this dissertation is to develop effective methodologies to enable robots to learn manipulation skills and plan the optimal actions for high-demanding industrial manipulation tasks. We demonstrated the effectiveness of the proposed methods in various

tasks including peg-in-hole assembly (Chapter 2), cable manipulation (Chapter 3), belt drive units assembly (Chapter 4), and cable routing (Chapter 5).

1.2 Dissertation Outlines

In this dissertation, we develop skill learning methodologies for industrial robotic manipulation. We study 1) how to learn manipulation skills when there are uncertainties in the object state estimation, 2) how to generalize the manipulation skills to different even unseen scenarios, 3) how to learn the high-level task planning for long-horizon tasks. In particular, we study the manipulation of both rigid objects and deformable objects. For the manipulation of rigid objects, we study the typical peg-in-hole assembly task, where we utilize vision and force/torque feedback to compensate the pose uncertainties between the peg and the hole [24]. For the manipulation of deformable objects, which have infinite degrees of freedom, we study how to track the object states in real-time from vision feedback and then manipulate the cable to the desired configurations [23]. We also study the long-horizon deformable object manipulation tasks, belt-drive-unit assembly [27] and cable routing [26], from the assembly challenges competition [56, 20]. The structure of the dissertation is shown in Figure 1.1.

Peg-in-Hole Assembly under Uncertainties

In Chapter 2 , we first study the well-known peg-in-hole assembly task under small position uncertainties. Peg-in-hole assembly is a challenging contact-rich manipulation task. Most of the assembly tasks in the factory are pre-defined, and the robots just need to follow the programmed trajectory to finish the task. However, uncertainties in the pose of workpieces may prevent the robots from accomplishing the task robustly. There is no general solution to identify the relative position and orientation between the peg and the hole yet. We propose a novel method to classify the contact poses based on a sequence of contact measurements. When the peg contacts the hole with pose uncertainties, a tilt-then-rotate strategy is applied, and the contacts are measured as a group of patterns to encode the contact pose. A convolutional neural network (CNN) is trained to classify the contact poses according to the patterns. In the end, an admittance controller guides the peg towards the error direction and finishes the peg-in-hole assembly. Simulations and experiments are provided to show that the proposed method can be applied to the peg-in-hole assembly of different geometries. We also demonstrate the ability to alleviate the sim-to-real gap [24].

In the scenarios where both the peg and the hole have large pose uncertainties in 6 degrees of freedom, we propose a framework to deal with pose uncertainties with two modules, the alignment module and the insertion module. The alignment module utilizes a 3D pose estimation to reduce the pose uncertainties into a small region and provide a safe and efficient action space for the insertion module. The insertion module compensates for the remaining small uncertainties with an impedance controller by tracking a reference generated from a reinforcement learning (RL) policy. We have successfully validated the proposed method in

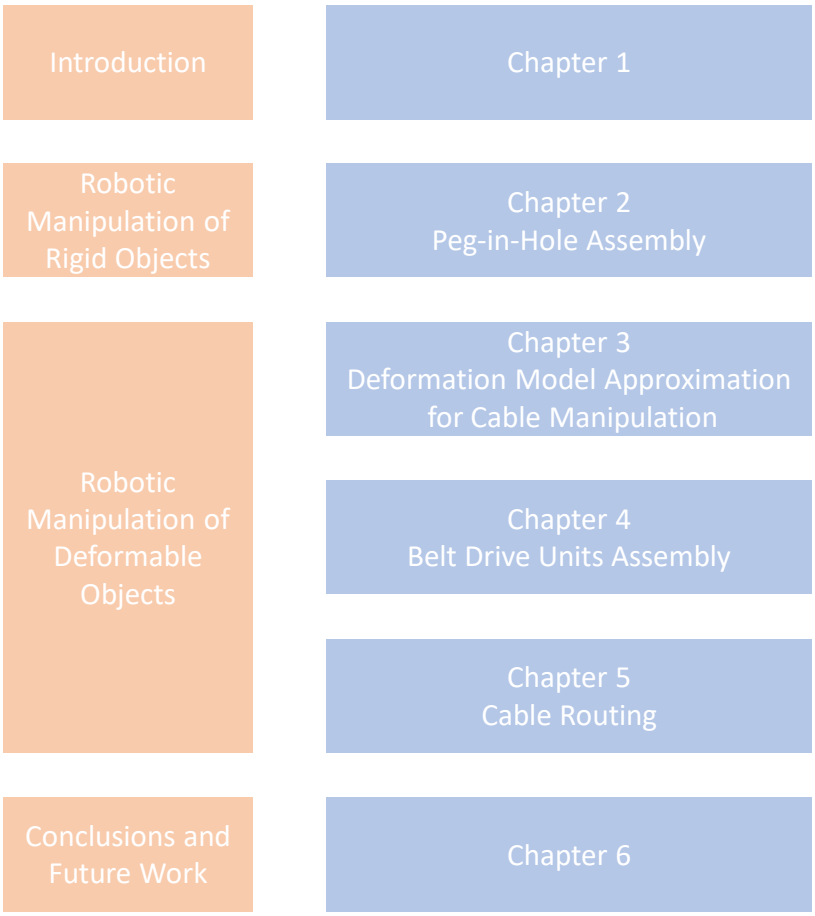


Figure 1.1: The structure of the dissertation.

simulations. We have also shown that the alignment module can work well in real-world peg-in-hole experiments.

Robust Deformation Model Approximation for Robotic Cable Manipulation

In Chapter 3, we study the tracking and manipulation of deformable objects. The flexible objects, such as wire and cable are also in high demand in manufacturing. The major challenge for the cable manipulation is that cables have high degrees of freedom and are easy to deform during manipulation. In this chapter, we propose a novel framework SPR-RWLS to manipulate cables, which includes real-time cable tracking and robust local deformation model approximation. For cable tracking, structure preserved registration (SPR) is utilized to robustly estimate the movement of selected points on a cable even in the presence of

sensor noise, outliers, and occlusions. Robust weighted least squares (RWLS) is then applied to calculate the local deformation model of the cable under uncertainties. We show that SPR-RWLS enables the dual-arm robots to manipulate cables with different thicknesses and lengths to different desired curvatures in multiple scenarios. We also show that real-time implementation of the proposed method can be simplified by parallel computation [23].

Trajectory Optimization for Manipulation of Deformable Objects: Assembly of Belt Drive Units

Chapter 4 presents a novel trajectory optimization formulation to solve the robotic assembly of the belt drive unit. Unlike previous tasks, belt drive unit assembly involving contact between deformable and rigid objects, which makes the overall problem even more challenging in terms of modeling and planning. For modeling, variations in the belt tension and contact forces between the belt and the pulley could dramatically change the system dynamics. For trajectory planning, it is computationally expensive to plan trajectories for such hybrid dynamical systems as it usually requires planning for discrete modes separately. In this chapter, we formulate the belt drive unit assembly task as a trajectory optimization problem with complementarity constraints to avoid explicitly imposing contact mode sequences. The problem is solved as a mathematical program with complementarity constraints (MPCC) to obtain feasible and efficient assembly trajectories. We validate the proposed method both in simulations with a physics engine and in real-world experiments with a robotic manipulator [27].

Robotic Cable Routing with Spatial Representation

Chapter 5 studies the robotic cable routing task. To accomplish the task, it requires a high-level path planner to generate a sequence of cable configurations from the initial state to the target state and a low-level manipulation planner to plan the robot motion commands to transit between adjacent states. However, there are yet no proper representations to model the cable with the environment objects, impeding the design of both high-level path planning and low-level manipulation planning. In chapter 5, we propose a framework for cable routing with spatial representation. For high-level planning, by considering the spatial relations between the cable and the environment objects such as fixtures, the proposed method is able to plan a path from the initial state to the goal state in a graph. For low-level manipulation, multiple manipulation primitives are efficiently learned from human demonstration, to configure the cable to planned intermediate states leveraging the same spatial representation. We also implement a cable state estimator that robustly extracts the spatial representation from raw RGB-D images, thus completing the cable routing framework. We evaluate the proposed framework with various cables and fixture settings, and demonstrate that it outperforms some baselines in terms of reliability and generalizability [26].

Chapter 2

Peg-in-Hole Assembly under Uncertainties

2.1 Introduction

As discussed in Chapter 1, industrial robots are widely used in manufacturing tasks, such as assembly. Most of the assembly tasks in the factory are pre-defined, and the robots just need to follow the programmed trajectory to finish the task. However, uncertainties in the pose of workpieces could prevent the robots from accomplishing the assembly robustly. In this chapter, we introduce the proposed state estimation and planning algorithms for peg-in-hole assembly under uncertainties.

In the first section of this chapter, we study the peg-in-hole assembly under small position uncertainties. We propose a tilt-then-rotate strategy to classify the contact poses. A sequence of contacts is measured and plotted as a group of patterns to encode the contact pose. A convolutional neural network is trained to classify the contact poses according to the patterns. In the second section, we consider the scenario where the peg and the hole have large pose uncertainties in 6 DoF. We propose an assembly framework with two modules, the alignment module and the insertion module. The alignment module reduces the pose uncertainties into a small region and provides a safe and efficient action space for the insertion module. The insertion module compensates for the remaining small uncertainties with an impedance controller by tracking a reference pose generated from an RL policy.

2.2 Contact Pose Identification for Peg-in-Hole Assembly under Position Uncertainties

Identifying the contact pose, the relative position and orientation between the peg and the hole, is required to align the peg and the hole before insertion. Visual feedback is the most common strategy to identify the pose [80, 70]. However, vision sensors suffer from high

precision requirements and occlusions during the assembly task. In order to avoid such problems, search-based algorithms such as random search or spiral search [6] have been proposed to compensate the uncertainties of contact pose. The search strategy generates a search path within the search area for hole localization, which is not efficient especially when the search area is large and the search dimension is high.

For insertion, the clearance between the peg and the hole is usually smaller than the precision of a robot. A tiny position and orientation error could cause workpieces to jam and wedge and may lead to failure or even damage to the workpieces. Compliance, either passive or active, has shown to be effective in handling the small uncertainties of position and orientation. Passive compliance utilizes passive compliance hardwares such as RCC [11, 78] to compensate uncertainties. In constrast, active compliance applies control strategies from software to let the robot mimic the spring-damping behavior [4, 48].

In contact-rich scenarios, force/torque-based method normally conveys more information than vision-based and search-based methods. Tang[65] analyzed a three-point contact model for round peg and hole. But the method lacked the ability to generalize to complex geometries. Kim proposed a peg shape recognition and hole detection algorithm using the force/torque sensor by inclining the peg in all directions, but their method suffered from the cumulative error [31]. In recent years, many learning-based methods have been proposed to solve the peg-in-hole assembly problem [68, 37, 13, 74, 36]. They treated the task as a Markov decision process, where the contact feedback at the current time step is used to determine the action of the next step. However, the mapping from the force/torque feedback to the contact pose is not injective as shown in Fig. 2.1. On one hand, the same contact forces can be measured at different contact poses. On the other hand, the same contact pose could generate different contact forces, i.e. all the possible forces within the Coulomb friction cone. To deal with the above problem, particle filter was applied to identify the location based on multiple observations in [7, 71]. However, it is time-consuming to generate the force-position mapping in the real world and hard to generalize.

We propose a novel method that can identify the contact poses based on a sequence of contact measurements. At initialization, the peg contacts the hole with pose uncertainties. The peg then follows a designed tilt-then-rotate motion to make contact with the hole. The contact measurements are plotted in polar coordinates to generate a group of patterns. An injective mapping between the patterns and contact poses is learned by a convolutional neural network (CNN), which classifies the contact poses based on the error directions. Finally, an admittance controller will guide the peg towards the error direction and finish insertion. There are two main contributions: 1) We construct the mapping using a sequence of measurements as input instead of feedback at one single time step. This makes the mapping become one-to-one. 2) We classify the contact pose based on patterns, which improves the generalization ability of the proposed method. It can even tackle the sim-to-real gap.

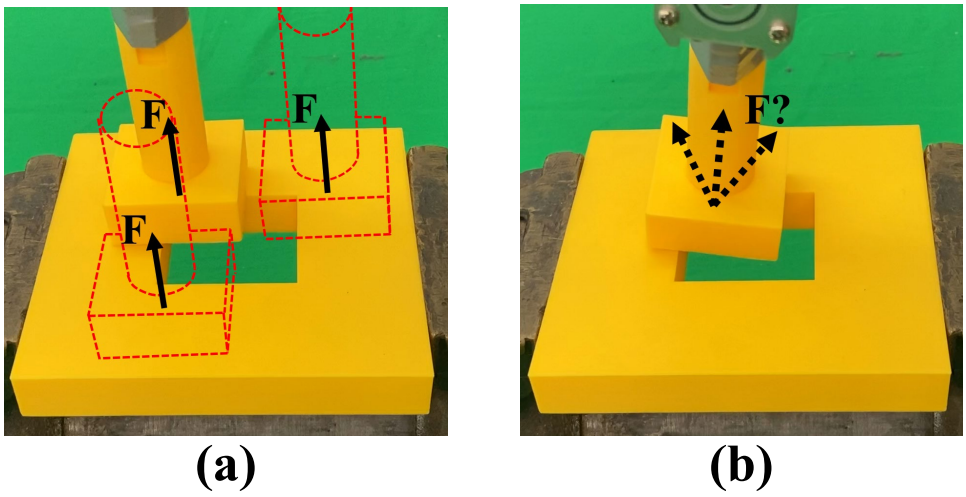


Figure 2.1: (a) The same upward contact force could come from many possible contact poses. (b) The same contact pose could generate many possible contact forces within the Coulomb friction cone.

Background

Task Description

We focus on the peg-in-hole assembly task under small pose uncertainties. Generally speaking, the pose of the peg and the hole might be noisy due to sensor inaccuracy. To simplify the problem, we assume the peg is fixed with the robot end-effector, and the pose can be obtained via forward kinematics. The hole is fixed on the table. After calibration, the pose can be estimated with uncertainties in 6 degrees of freedom (DOF). The magnitudes of the uncertainties are roughly $\pm 20mm$ and $\pm 3^\circ$ for position and orientation respectively, which are determined by the precision of the calibration. The clearance between the peg and the hole is $1mm$.

The goal of the task is to compensate the uncertainties of contact pose and achieve the peg-in-hole assembly. The contact surfaces of both the peg and the hole are assumed to be flat.

Admittance Control

Admittance control [4, 48] is widely used in robotic manipulation tasks to handle contact dynamics. By adding a virtual spring-damping system, the contact between the robot and the environment becomes soft, which improves the manipulation performance and prevents from damaging either the robot or the environment. We apply admittance control to the following assembly strategy to track the desired peg trajectory and compensate small uncertainties in assembly.

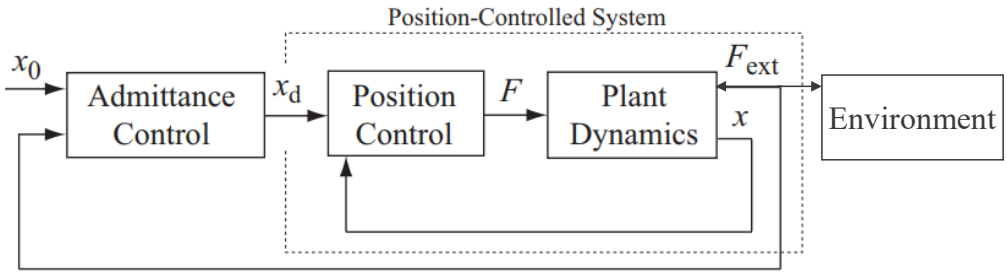


Figure 2.2: Admittance Control.

In admittance control, the desired pose x_0 and measured external force/torque F_{ext} are inputs to the admittance control block (Fig. 2.2), which generates the reference pose x_d for the PD position control.

$$F + F_{ext} = m\ddot{x} \tag{2.1}$$
$$F = k_p(x_d - x) - k_d\dot{x} \tag{2.2}$$
$$F_{ext} = M_d(\ddot{x}_d - \ddot{x}_0) + D_d(\dot{x}_d - \dot{x}_0) + K_d(x_d - x_0) \tag{2.3}$$

where M_d, D_d, and K_d represent the desired inertia , damping, and stiffness, respectively. k_p and k_d are PD position control gains.

Assembly Strategy

Peg-in-hole assembly has been studied for decades. An efficient and widely used assembly strategy divides the task into several stages [31, 28]: initialization, approaching, contact pose estimation, alignment, and insertion. At initialization, the peg and the hole are fixed on the robot manipulator and the table, respectively. The pose of the hole can be roughly estimated after calibration. At approaching, the peg approaches to the hole with an admittance controller. With well-tuned controller parameters, the plane contact between the flat surface of the peg and the hole could eliminate the pose uncertainties in 3 dimensions, roll axis, pitch axis, and z-axis. At contact pose estimation, the peg explores along the surface of the hole to estimate the relative position and orientation between the peg and the hole. This stage eliminates the uncertainties in x and y axes. Finally, based on the contact pose estimation, the peg can slide towards the hole and finish insertion with an admittance controller. Small oscillation is added to the yaw axis in this stage, together with admittance control, to compensate small uncertainties of yaw axis. In this work, we mainly focus on the contact pose estimation stage, which is introduced in section 2.2.

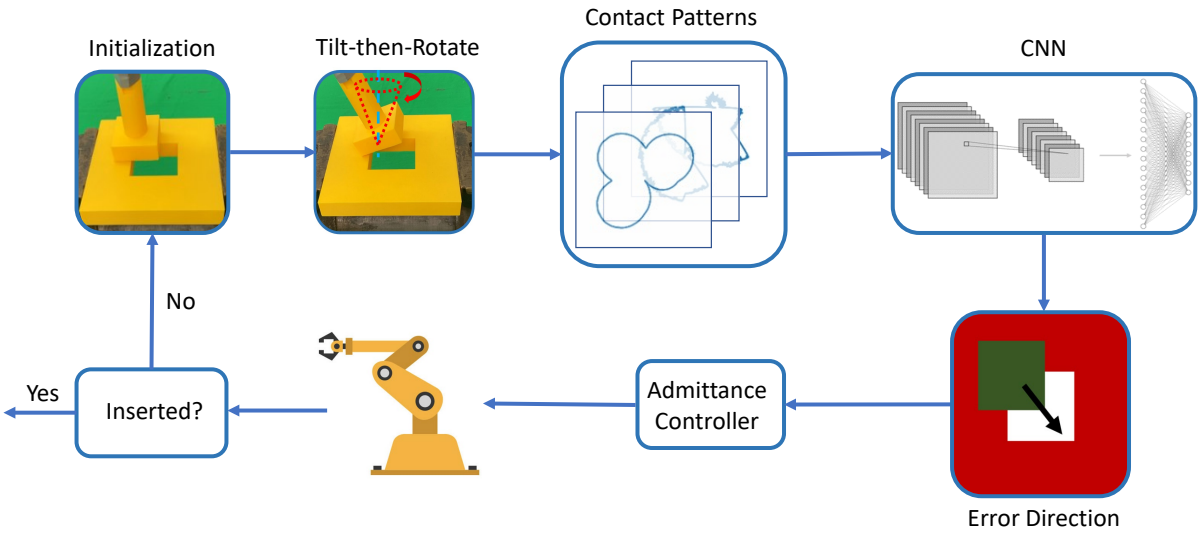


Figure 2.3: Framework of the proposed method.

Proposed Method

Peg-in-hole assembly can be accomplished easily by a human even with eyes closed. The human will first use the peg to make contact with the hole. Then he/she will locally move the peg to sense hole's location based on a sequence of contacts instead of just one single contact. If there is a hole in one direction, the tip of the peg could slide into the hole a little bit and the force/torque feedback also have an impulse in that direction. Based on the historical measurements in a sequence of contacts, the human keeps updating the knowledge of the contact pose and eliminating the hole uncertainties.

Inspired by the human strategy, we propose to use a sequence of contact feedback to identify the contact pose under uncertainties (Fig. 2.3).

Tilt-then-Rotate Strategy

The peg contacts the hole after the approaching stage (Fig. 2.4.1). The peg and the hole have some overlaps but are not aligned well due to the uncertainties of the contact pose. We propose a tilt-then-rotate strategy to identify the contact pose.

We tilt the peg for α degrees in all directions by rotating the peg for 2π (Fig. 2.4). The tilt-then-rotate trajectory can be described as continuously changing θ from 0 to 2π in order to change the roll and the pitch angle:

$$\{roll, pitch\} = \{\alpha sin(\theta), \alpha cos(\theta)\}, \quad \theta \in [0, 2\pi) \tag{2.4}$$

The desired tilt-then-rotate trajectory is tracked by an admittance controller. At the same time, a constant downward force is applied to the peg in order to maintain contact with the

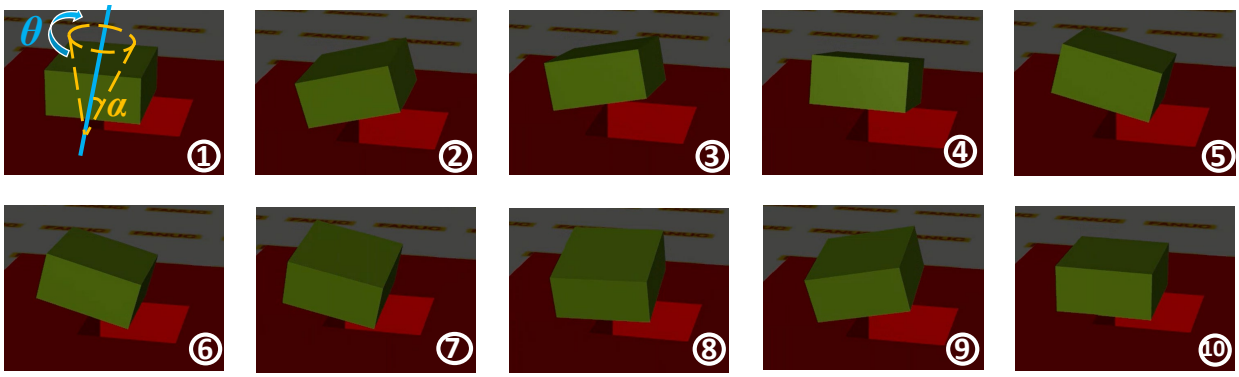


Figure 2.4: Snapshots of the tilt-then-rotate strategy. The blue line is z-axis. The yellow cone represents the designed trajectory for rotation. Tilt (2) then rotate (2-9) the peg for 2π. While the peg is being rotated, a constant downward force is applied to maintain a single point contact (3,5,9), line contact (2,4,6,8), or two points contact (7) between the peg and the hole.

hole. During the procedure, contact force and torque are measured by a force/torque sensor. As the peg is tilted in all directions, the contact keeps switching between one point contact, two points contact, and line contact (Fig. 2.4.2-2.4.9). The tip of the peg could go into the hole when the peg tilts towards the hole and the force/torque measurements would also have an impulse. Different contact poses will result in different sequences of measurements along the designed tilt-then-rotate trajectory. Comparing with one measurement at a single time step, the mapping from a sequence of measurements to contact poses becomes an injective mapping.

Contact Pattern Generation

The tilt-then-rotate strategy generates a sequence of measurements in 12 dimensions including force (R^3), torque (R^3), and peg pose (R^6). For different control forces or different sizes of the parts, those measurements can be different in the order of magnitude. Human can sense the contact pose in different scenarios by the same exploring strategy. There must be some high-level features we can extract from the measurements.

We propose to plot the measurements of each dimension in polar coordinate as one channel. The data in each channel is normalized, then smoothed by moving average. The normalization makes the data invariant to control forces and sizes of the parts. The moving average reduces the sensor noises. We utilize the plotted image with 12 channels as one contact pattern, which encodes high-level features about the contact pose. Fig. 2.5 shows z-axis channel of the contact pattern for different contact poses.

One contact pose corresponds to one contact pattern with 12 channels. In order to construct an informative mapping, we need to perform hundreds of tilt-then-rotate motions for

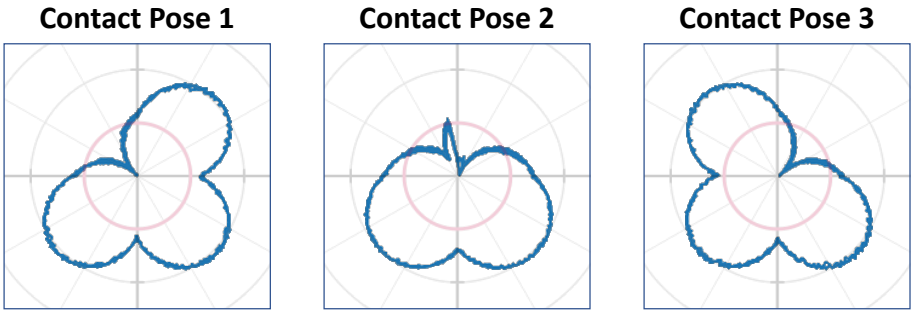


Figure 2.5: Contact patterns in polar coordinates for 3 different contact poses. Only z-axis channel is shown.

all contact poses. This is not only time-consuming but also inaccurate due to the limitation of pose sensing in the real world. We propose to generate the contact pattern in the MuJoCo physics engine. The simulated environment can perform hundreds of trails in a short time. In addition, ground truth contact pose can be obtained easily in simulation (Fig. 2.4).

Contact Pose Classification Neural Network

Contact poses of a square peg-hole can be classified into 9 classes according to which edge of the peg contacts the hole (Fig. 2.6). Each class of contact pose has a different error direction. Classifying the contact poses from the contact patterns is an image recognition problem. CNN has shown great success in image recognition in terms of efficiency and accuracy [33]. We train one simple CNN to classify the contact patterns. The CNN has two convolutional layers, two pooling layers, and one fully-connected layer. The input data are the 3 most informative channels out of the 12-channel pattern. The output is the class of contact pose, which has 9 error directions for a square peg-hole and 11 error directions for a pentagonal peg-hole. Once the contact pose is identified, the peg will be guided towards the error direction with admittance control and inserted into the hole.

Failure Recovery

From the experiments, we observe failure cases even with the method described above. The reason is either the contact pose classification model predicts a wrong error direction or the admittance control fails to compensate small uncertainties. To increase the robustness of the proposed method, we add a failure recovery module. If we fail to insert the peg into the hole, the peg will be initialized to a slightly different pose than the original one, and redo the tilt-then-rotate strategy again.

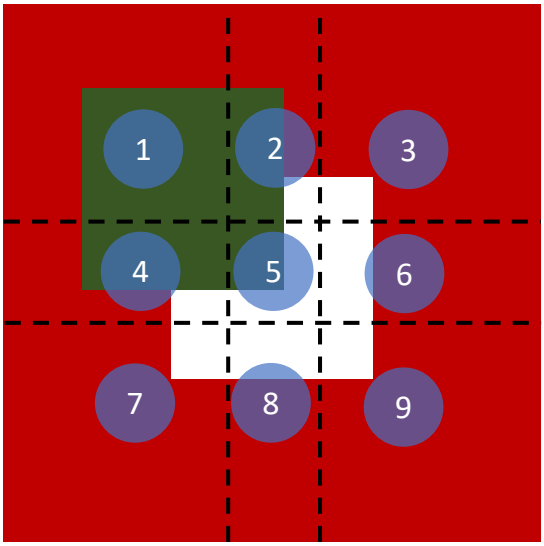


Figure 2.6: The contact poses are classified into 9 classes according to which edge of the hole contacts the peg.

Simulations and experiments

Simulations

Simulation Setup The simulated environment in MuJoCo is shown in Fig. 2.4. The environment includes a peg and a hole, where the hole is fixed on the ground, and the peg is controlled by a well-tuned admittance controller. The side length of the hole is $50mm$ and the side length of the peg is $49mm$ (clearance = $1mm$). Contact force/torque is measured at the peg's center of mass.

Data Collection A self-supervised scheme is applied to collect the data and build the contact pose mapping. As mentioned in 2.2, once the peg contacts the hole on the flat surface, the uncertainties in roll, pitch, and z-axis are eliminated. We only consider the remaining uncertainties in the x, y, and yaw axis. The contact poses are uniformly sampled from $x \in [-20, +20]mm$, $y \in [-20, +20]mm$, and $yaw \in [-3°, +3°]$. After the approaching stage in 2.2, the tilt-then-rotate strategy is applied, and α in equation (2.4) is set to 15°. The tilt-then-rotate motion is executed by the admittance controller in N time steps, where $N = 2000$. The 12-dimension peg pose and contact force/torque are recorded in a matrix $A \in R^{N\times 12}$. The data of each dimension is normalized then smoothed by moving average with a window length $n = 20$, and the contact pattern is recorded in polar coordinates as a $12 \times 200 \times 200$ binary image. We label the contact patterns of a square peg-hole with 9 classes according to the initial contact poses (Fig. 2.6). The uncertainty in the yaw axis is compensated by the admittance controller and small oscillations in the yaw axis. We also

Table 2.1: Peg-in-hole assembly in simulation

# of attempts	1	2	3	> 3	total	success rate
square ($50mm$)	96	3	1	0	100	100%
pentagon ($37mm$)	82	11	3	4	100	96%

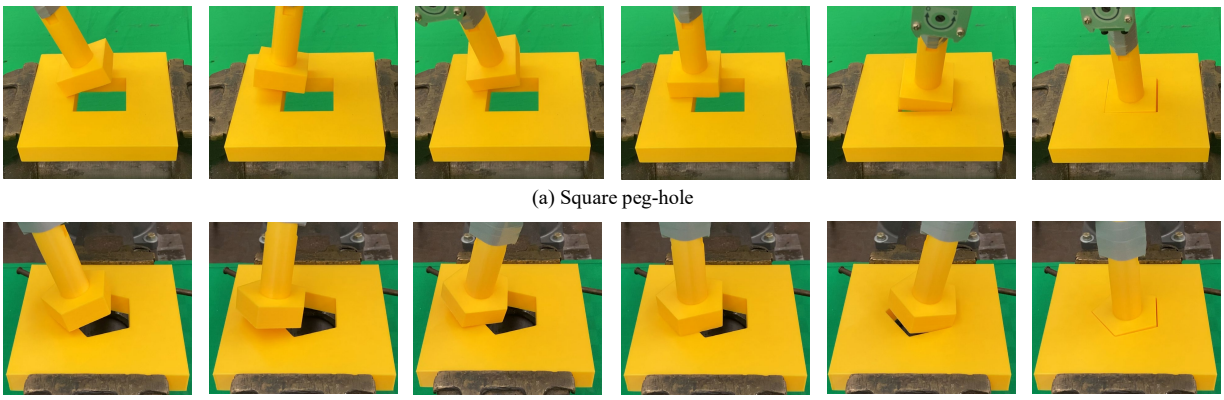
add 5% noise to the parameters of the admittance controller in order to introduce variance to the collected data. We perform the self-supervised data collection for 5000 trails. The computation time is around 10 minutes. We split 80% data as the training set and 20% data as the test set.

Model Training From the 12 channels contact patterns, we select 3 channels $A' \in R^{N\times 3}$ including the position in z axis X_z, the torque in roll axis M_x, and the torque in pitch axis M_y as the input to the CNN. The reason that we select these 3 channels is that we experimentally find that these channels contain more features than other channels. We downsample the contact patterns into $3 \times 20 \times 20$ images. We use an NVIDIA GeForce GTX 1080 Ti GPU for training. The training time is around 1 minute.

Results The test accuracy of the contact pose classification neural network is 97.4%. Most of the failure cases are the contact pose at the boundary between two classes. We test on a second data set by collecting 1000 data from a smaller square peg-hole, where the side length of the hole is $32mm$ (clearance = $1mm$). The test accuracy is 96.8%. This shows the generalization ability of the proposed method. Although the sizes of the parts, the contact measurements such as force, torque are different, the model still works very well. The reason is that we predict the contact pose according to the contact pattern, which is invariant to the size of the parts.

We perform another simulation experiment on a pentagonal peg-hole. The side length of the hole is $37mm$ (clearance = $1mm$). Because the contact pattern highly depends on the geometry of the peg-hole, we cannot apply the model learned from square peg-hole to pentagonal peg-hole. We redo the data collection and model training on the pentagonal pen-hole. Everything is the same as square peg-hole, except the number of contact pose classes becomes 11. The test accuracy is 91.0%.

We test the entire peg-in-hole assembly framework using the proposed method. We perform 100 trials on both square and pentagonal peg-hole. If the peg fails to be inserted into the hole, the failure recovery module will initialize the peg to a slightly different pose than the original one, and redo this trial again. If it requires more than 3 attempts to finish the task, we claim it fails. Table 5.1 shows the number of attempts needed to finish assembly in simulation. The high success rate shows that the proposed framework works well.



(a) Square peg-hole

(b) Pentagonal peg-hole

Figure 2.7: Snapshots of the experiments.

Experiments

Experimental Setup The experiment environment (Fig. 2.7) includes a 6 DOF FANUC LR-Mate 200iD, an ATI Mini45 F/T sensor, and 3D printed peg-holes. The F/T sensor is embedded in the robot end-effector to measure the force and torque during assembly. The force/torque measured at the robot wrist can be transfer to the force/torque at the peg's center of mass. The peg is fixed on the robot end-effector and the hole is fixed on a vise. The peg's pose can be controlled with an admittance controller at $125Hz$. The hole is randomly initialized with position and orientation uncertainties $\pm 20mm$ and $\pm 3°$, respectively. Three pairs of 3D printed peg-holes are tested, including a $50mm$ square hole (clearance = $1mm$), a $32mm$ square hole (clearance = $0.5mm$), and a $37mm$ pentagonal hole (clearance = $1mm$).

Results Fig. 2.8 shows the comparison of the contact patterns generated from tilt-then-rotate strategy in simulation and real-world experiments. They are generated from the same class of contact pose. The data collected from the real-world has much noise than from simulation. Although there is a huge sim-to-real gap[72] between the simulated environment and the real world in terms of friction coefficient, inertia, stiffness, damping ratio, etc., we observe that the contact patterns do share similar features.

The contact pattern classification model learned in the simulation are applied to real-world experiments. Fig. 2.7 shows the snapshots of the assembly experiments. We perform 20 experiments on 3 different pairs of peg-hole respectively. Table 5.2 shows the number of attempts needed to finish assembly in real-world experiments. The model learned in simulation ($50mm$ square, clearance = $1mm$) can be successfully applied to real-world peg-hole of different sizes ($32mm$) and smaller clearance ($0.5mm$). This shows that the proposed method is able to tackle the sim-to-real gap. Supplementary videos can be found in [1].

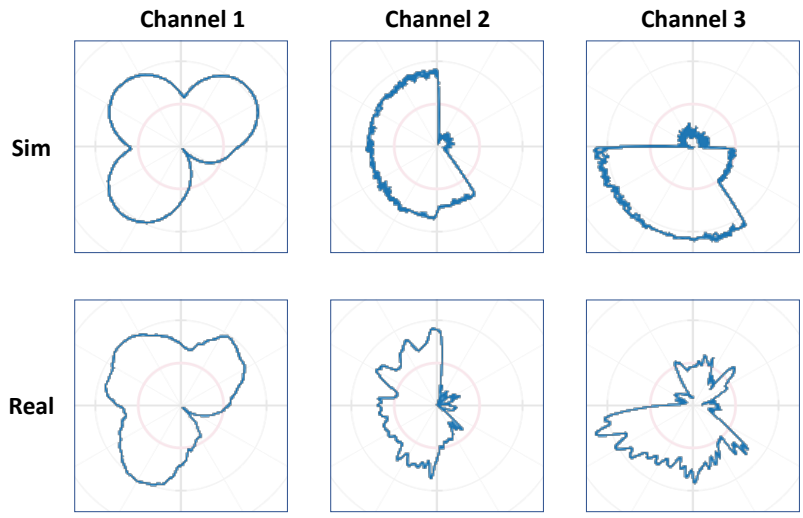


Figure 2.8: Comparison of the contact patterns in simulations and real-world experiments. They are generated from the same class of contact pose

Table 2.2: Peg-in-hole assembly in real-world experiments

# of attempts	clearance	1	2	3	> 3	total	success rate
square ($50mm$)	$1mm$	16	3	1	0	20	100%
square ($32mm$)	$0.5mm$	10	5	2	3	20	85%
pentagon ($37mm$)	$1mm$	15	3	1	1	20	95%

2.3 Peg-in-Hole Assembly under 6 DoF Pose Uncertainties

In this section, we consider a peg-in-hole assembly scenario where both the peg and the hole have large pose uncertainties in 6 degrees of freedom. We propose a framework to deal with pose uncertainties with two modules, the alignment module and the insertion module. In the alignment module, we utilize the Deep Neural Network (DNN) to segment the peg and the hole from depth images. With the knowledge of CAD model, we can recover the $3D$ point cloud of the workpieces's surface using Ransac. Then we use registration method to find the rotation and translation to align the peg and the hole. The alignment module is able to reduce the pose uncertainties into a small region and provides a safe and efficient action space for the insertion module. The insertion module compensates the remaining small uncertainties with an impedance controller by tracking a reference generated from Reinforcement Learning (RL).

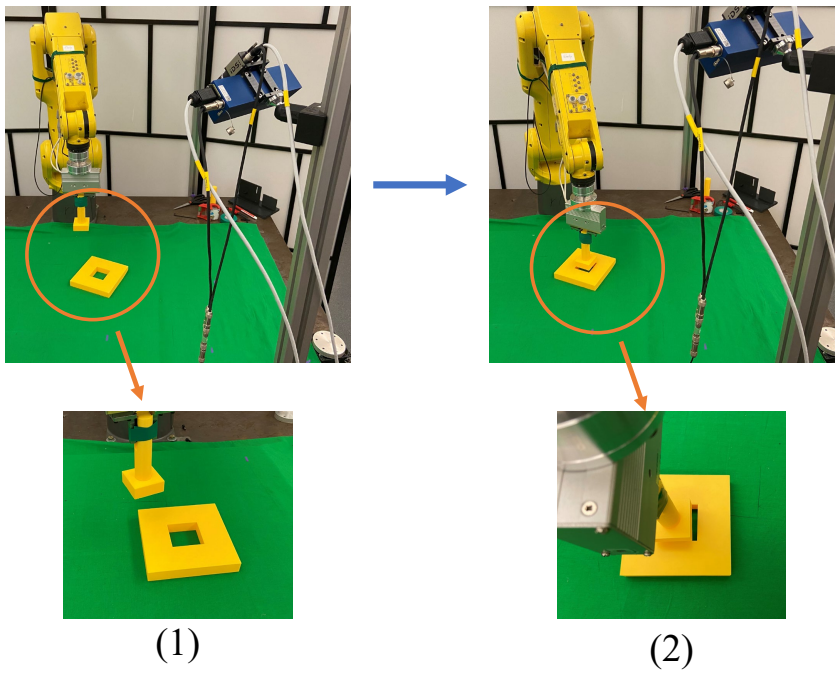


Figure 2.9: Peg-hole alignment

Background and Preliminary

Task Description

The previous chapter only considers small pose uncertainties which are roughly $\pm 20mm$ and $\pm 3°$ for position and orientation respectively. In this chapter, we consider a scenario where both the peg and the hole have 6 DoF large pose uncertainties. The magnitude of the uncertainties are within $\pm 200mm$ and $\pm 20°$ for position and orientation respectively. The clearance between the peg and the hole is $1mm$. The contact surfaces of both the peg and the hole are assumed to be flat. The goal of the task is to compensate the pose uncertainties using the proposed alignment and insertion modules and achieve the peg-in-hole assembly.

Image Segmentation using U-Net

Image segmentation has many applications in robotic perception. It involves partitioning images into multiple segments or objects. The segmentation can be formulated as a classification problem of pixel with individual objects. Over the past few years, deep learning models have shown remarkable performance and achieved very high accuracy rate on image segmentation. U-Net was proposed by Ronneberger et al. [55] for segmenting biological microscopy images. The architecture comprises two parts, a contracting path to capture context, and a symmetric expanding path that enables precise localization. Feature maps

from the down-sampling part of the network are copied to the up-sampling part to avoid losing pattern information. Finally, a 1×1 convolution processes the feature maps to generate a segmentation map that categorizes each pixel of the input images. In this work, we utilize the U-Net to segment the peg and the hole from a depth image input. The segmentation is then utilized to reconstruct the 3D point cloud of the peg and hole.

Reinforcement Learning for Robotic Manipulation

In recent years, we have seen a lot of applications of Reinforcement Learning on robotic manipulation tasks. RL has shown the ability to adapt to different scenarios. It utilizes a high-level reward function to indicate what to do and learn how the task should be performed. RL also performs well on peg-in-hole assembly task by learning a locally searched strategy to compensate the pose uncertainties. However, most of the works only consider the position uncertainties in a small region. If the orientation uncertainties are being considered, the action space becomes larger and the complexity increases exponentially, making the problem much harder to solve. In this work, we utilize the output from the alignment module as a constraint of the RL's action space. The constraint creates a safe and efficient region for the RL policy to explore even in high dimension. By adding an admittance controller, the RL policy is able to safely interact with the environment without damaging neither the robot nor the workpieces.

Proposed Method and Implementation Details

In this section, we describe the proposed alignment module and insertion module for the peg-in-hole assembly. We partition the space into two sets, the alignment set S_a where the alignment module is used, and the insertion set S_i where the insertion module is used. $S_i \subset S_a$. In S_a, the peg and the hole have large relative pose errors. We utilize 3D pose estimation to estimate the pose of both the peg and the hole. Then we align the peg and the hole without making contacts. The alignment module can only coarsely reduce their relative pose uncertainties due to the resolution and calibration error of the vision system. Once the pose uncertainties below a small threshold, we are in the insertion set S_i, where reinforcement learning is used to compensate the remaining small pose uncertainties.

Alignment Module

Vision systems are one of the most efficient tools for robot to perceive the environment and obtain the pose of an object. But the perception system may not have high accuracy due to the resolution of the camera and the calibration error. To avoid the extensive effort on camera calibration, we propose an alignment module for peg-in-hole assembly which aligns the pose of peg and hole based on their relative pose error in S_a. Since the pose alignment is done in the camera frame, the proposed alignment module does not require

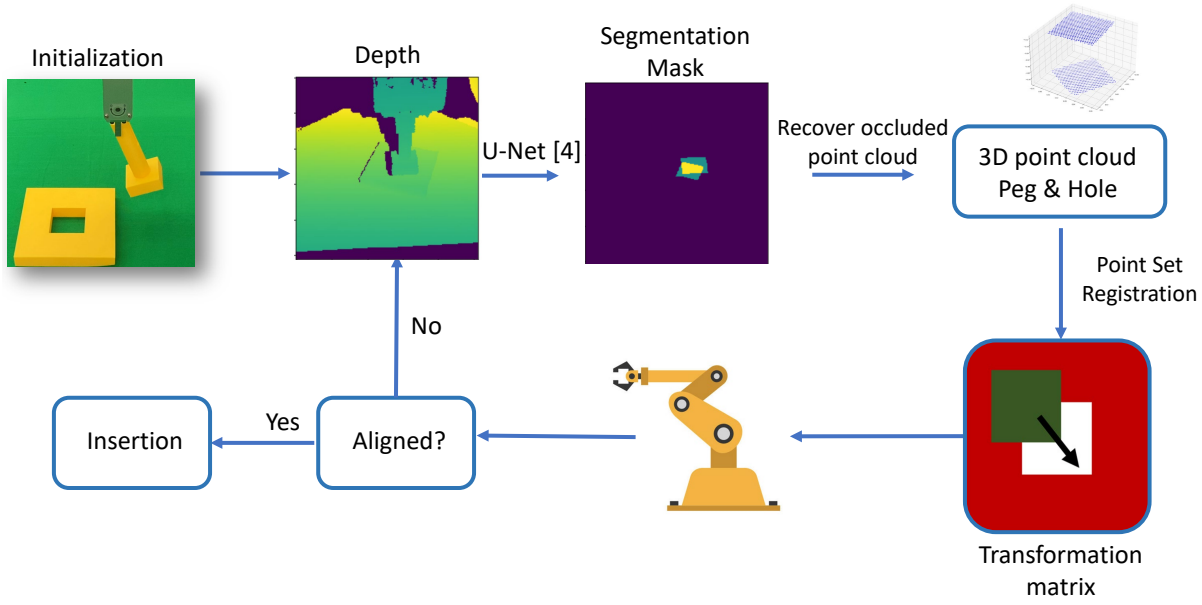


Figure 2.10: The alignment module mainly contains a segmentation neural network module, a point cloud recovery module, and a point set registration module.

precise camera calibration. The alignment module can be divided into three parts: 1) depth image segmentation. 2) occluded point cloud recovery. 3) peg-hole pose registration.

Depth Image Segmentation At the initial, the hole is fixed on the table and the peg is grasped by a robot end-effector. Both the peg and the hole have unknown poses in 6 DoF. We propose to segment the peg and the hole from an image using U-Net. For each time step, the only input is a depth image captured by depth camera. The outputs are binary segmentation masks categorize each pixel of the input depth images. In our setup, the masks are the upper flat surface of both the peg and the hole.

In order to generate the dataset D for training the U-Net. We utilize a well-calibrated depth camera. The pose of the hole is known exactly. The pose between the peg and the robot end-effector is also known. Then the segmentation masks for both the peg and the hole can be computed using the intrinsic and extrinsic matrix of camera calibration. By randomly moving the robot end-effector above the hole without making contacts, we can generate depth images and corresponding labels as many as we want to train U-Net. Note that we do need the camera calibration matrix to generate the labels in D. But once the U-Net is well trained, we do not require a precise camera calibration during the test time.

Occluded Point Cloud Recovery The segmentation masks provide us the pixel coordinates of the peg and the hole on the depth image at time step t. Due to the occlusions,

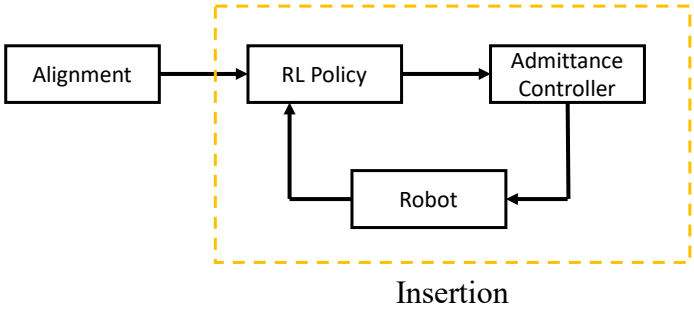


Figure 2.11: The insertion module.

reprojecting the depth information back to $3D$ space can only give us the occluded $3D$ point cloud of the peg P_o^t and the hole H_o^t. Taking the advantage of the CAD models, we can reconstruct the virtual $3D$ point cloud by detecting and refilling a flat surface using Ransac. We denote the recovered point cloud of the peg and the hole at time step t as P_r^t and H_r^t, respectively.

Peg-hole Pose Registration The relative pose error between the peg and the hole at time step t can be found by registering the recovered point cloud P_r^t and H_r^t. We utilize point set registration method to find the rotation $R^t \in R^{3\times3}$ and translation $T^t \in R^3$ between two point clouds, $H_r^t = R^t P_r^t + T^t$.

Insertion Module

After R^t and T^t are computed, we are able to align the peg and the hole using the robot. Due to the accuracy of U-Net prediction, resolution of the camera, and calibration, the peg and the hole cannot be perfectly aligned. The previous alignment module reduce the relative pose errors and brings peg-hole into the insertion set S_i, where only small pose uncertainties are remaining.

We propose to use the state of the art reinforcement learning, Soft Actor-Critic, to compensate this remaining pose uncertainties. The RL policy does not require global information such as the images input. Because the alignment module has already brought the state into S_i. The RL policy only needs to explore in S_i and focuses on locally searching the hole locations by making contacts. The S_i acts as a boundary for the RL policy, making the exploration more efficient by only learning a locally policy. Even if the state goes outside the S_i, the alignment module will bring it back. This guarantees the safety of the RL policy in the insertion module.

The input of the RL policy includes the positions and euler angles of the peg, their first derivatives, and the force/torque feedback. Instead of directly output a robot command, the output of the RL policy is a reference pose. We then utilize an admittance controller to track

the reference pose. Admittance control [4, 48] is widely used in robotic manipulation tasks to handle contact dynamics. By adding a virtual spring-damping system, the contact between the peg and the hole becomes soft, which improves the insertion performance and prevents from damaging either the robot or the environment. For the reward, we use a negative $L2$ norm to the estimated pose error in the alignment module. We will also assign a reward 1 when the insertion is finished.

Framework

The framework of combining the alignment module and the insertion module is shown in Algorithm 3. We denote the position and euler angle of the pose error at time step t as $s^t \in R^6$. The algorithm switches between alignment and insertion based on whether s^t is in S_a or S_i.

Algorithm 1: Peg-in-Hole Assembly under 6 DoF Pose Uncertainties

```
Initialize the peg and the hole, t = 0;
while not inserted and episode ≤ max_episode do
    depth image segmentation;
    occluded point cloud recovery;
    peg-hole pose registration;
    obtain s^t from R^t and T^t;
    if s^t not in S_a then
        reset robot;
    else if s^t not in S_i then
        align the peg with the hole based on s^t;
    else
        RL policy generates reference pose a^t;
        add (s^t, a^t, s^{t+1}, r^t) to replay buffer;
        admittance controller tracks a^t;
    end
    t = t + 1
end
```

Simulations and experiments

In this section, we explain the simulations and experiments to validate the proposed method.

Simulations

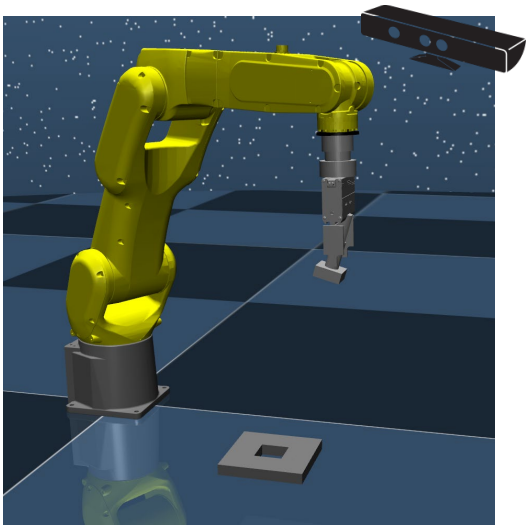


Figure 2.12: Simulation environment.

Figure 2.13: Peg and hole segmentation from a depth image input.

Simulation Setup The simulated environment in MuJoCo is shown in Fig. 2.12. The environment includes a square peg, a square hole, and a Faunc LR-Mate 200iD robot. The hole is fixed on the ground, and the peg is fixed to the robot end-effector. The robot end-effector can be controlled by a well-tuned admittance controller. The side length of the hole is $50mm$ and the side length of the peg is $49mm$ (clearance $= 1mm$). Contact force/torque is measured by the force/torque sensor mounted at the robot wrist. A depth camera is utilized to capture the depth images for the alignment module.

Alignment Module A self-supervised scheme is applied to collect the depth images and labels. We randomly initialize the pose of the peg and the hole. The poses of both the peg and the hole are uniformly sampled from $[-200, 200]mm$ in x, y, z positions and $[-20°, +20°]$

in $roll, pitch, yaw$ axes. The $3D$ poses of both the peg and the hole are known exactly. By utilizing the camera's intrinsic and extrinsic matrix, we are able to label the pixel of the perceived depth images. We use binary masks to represent the upper flat surface of the workpieces in the depth images. The input to the U-Net is a $800 \times 800 \times 1$ depth image. The output is a $800 \times 800 \times 2$ segmentation mask, where the masks of the peg and the hole are in two separate channels. 3000 depth images/labels are collected within 5 minutes for training the U-Net. We split 80% data as the training set and 20% data as the test set. The training takes about 12 hours on an NVIDIA GeForce GTX 1080 Ti GPU. The prediction results can reach above 99.5% pixel-wised success rate on the test set.

Insertion Module We use simplified environment to train the RL policy in the insertion module. The environment only contains a peg and a hole. The hole is randomly initialized with $\pm 5mm$ in positions and $\pm 5^\circ$ in orientations. We can directly control the pose of the peg using an admittance controller by tracking the output from Soft Actor-Critic policy.

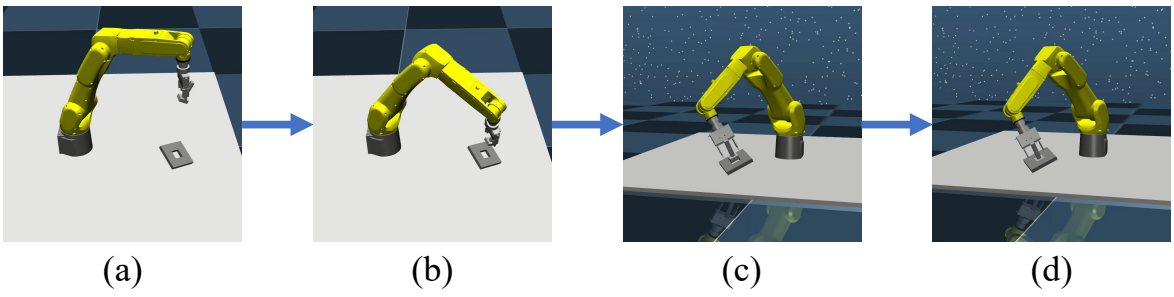


Figure 2.14: Snapshots of alignment and insertion in simulation.

Results We have conducted experiments on randomly initial pose of the hole. The relative pose between the peg and the robot end-effector is also unknown. The alignment module can successfully reduce to pose uncertainties under $2mm$ in positions and 1° in orientations. The insertion module can compensate the remaining pose uncertainties and achieve the insertion with 100% success rate on 20/20 trails. Even if we add 5% noises to both the depth image input and force/torque feedback, the proposed method can still achieve fully assembly on 19/20 trails.

Experiments

Experimental Setup The experiment environment (Fig. 2.7) includes a 6 DOF FANUC LR-Mate 200iD, an ATI Mini45 F/T sensor, an Ensenso camera, and 3D printed peg-holes. The F/T sensor is embedded in the robot end-effector to measure the force and torque during assembly. The force/torque measured at the robot wrist can be transfer to the force/torque

at the peg's center of mass. The peg can be fixed on the robot end-effector and the hole can be fixed on the table. We assume no movement between the peg and the robot end-effector, or between the hole and the table during assembly. The peg's pose can be controlled with an admittance controller at $125Hz$. The side length of the $3D$ printed peg and hole are $49mm$ and $50mm$, respectively.

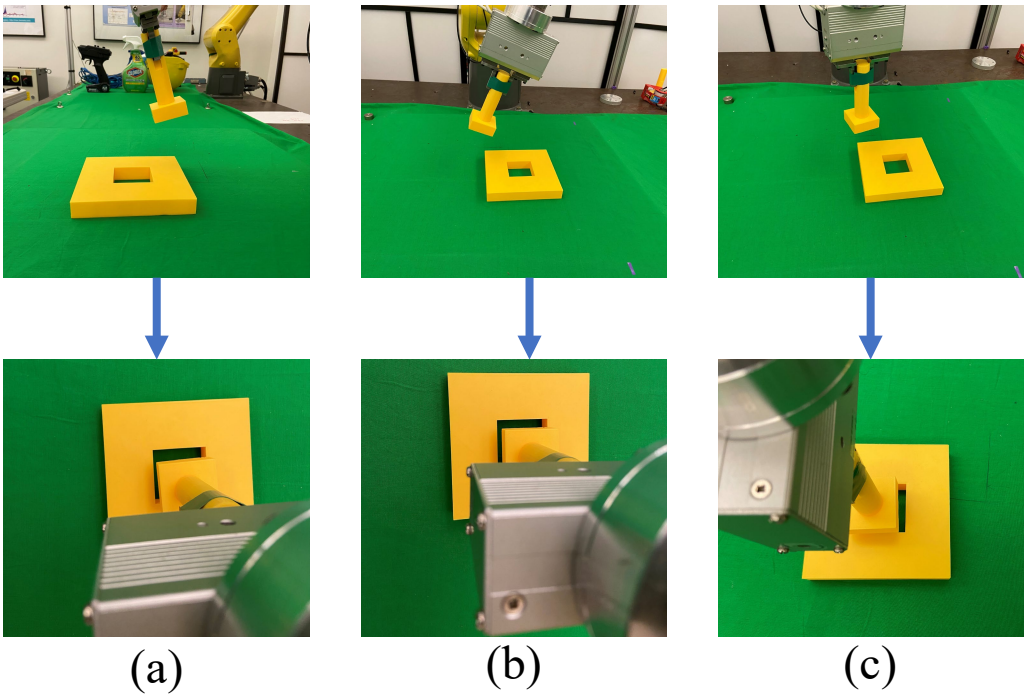


Figure 2.15: Snapshots of peg-hole alignments in real-world experiments.

Alignment Module and Results In order to collect the real-world data for training the U-Net. We need to well calibrate the Ensenso camera. We assume that the pose of the peg and the hole are known exactly after precise calibration. The segmentation mask can be obtained using the calibration matrix. By randomly moving the robot end-effector above the hole, we can collect the depth images and corresponding segmentation masks in a semi-self-supervised way. It takes about one hour to collect 400 data. We augment this dataset $10X$ using the affine and image-based transformation. The U-Net trained in simulation is then fine-tuned with the collected real-world data.

By using the proposed method, the alignment module can successfully reduce the relative pose uncertainties within $\pm 6mm$ in x, y positions, $\pm 1mm$ in z position, $\pm 2°$ in $pitch, roll$ axes, and $\pm 6°$ in yaw axis. The remaining pose uncertainties will be compensate with the insertion module.

2.4 Chapter Summary

In the first section of this chapter, we propose a novel framework to identify contact pose for peg-in-hole assembly under uncertainties. The proposed method utilizes a tilt-then-rotate strategy to generate contact patterns. A CNN is utilized to classify the contact poses and guide the robot to achieve the assembly task with admittance control. Simulation and experiment results are provided to demonstrate the effectiveness of the proposed method. The main advantages of the proposed method include:

- An injective mapping from the contact pattern to the contact pose.
- The contact pose classification model is easy to obtain. All the training data can be quickly generated in simulation with a self-supervised scheme.
- Good generalization ability and small sim-to-real gap. Since the contact data is normalized and recorded in a polar coordinate, the pattern is sensitive neither to the size of the object nor the parameters of the admittance controller. A model learned from a larger peg-hole can be successfully applied to smaller ones as long as the geometries are the same. Furthermore, the model learned in simulation can be adapted to the real world, despite the huge sim-to-real gap.

Here are the limitations of the proposed framework:

- The proposed method can only find the directions of the error, while it is unable to obtain the magnitude. In order to compensate for the error, the admittance controller needs to be well-tuned.
- The contact pose classification model can handle only position uncertainties, but it cannot classify the orientation uncertainties in the yaw axis.

In the second section, we consider a peg-in-hole assembly scenario where both the peg and the hole have large pose uncertainties in 6 degrees of freedom. We propose a framework with an alignment and an insertion module. In the alignment module, we utilize a Deep Neural Network (DNN) to segment the peg and the hole from depth images. We can recover the $3D$ point cloud of the workpieces by detecting the flat surfaces. Then we use point set registration to find the rotation and translation to align the peg and the hole. The alignment module is able to reduce the pose uncertainties into a small region and provides a safe and efficient action space for the insertion module. The insertion module compensates the remaining small uncertainties with an impedance controller by tracking a reference pose generated from a RL policy. We have successfully validate the proposed method in simulations. We also show that the alignment module can work well in real-world peg-in-hole experiments.

For future works, we plan to improve the tilt-then-rotate strategy so that it can handle large orientation uncertainties and test it with more challenging peg-hole shapes. We also intend to incorporate active and adaptive sensing strategies to our framework, so that we

don't have to use a predefined tilt-then-rotate strategy. For the alignment-insertion module, we will train and test the reinforcement learning and compliance controller in real-world. Considering the scenario where peg can slip in the robot end-effector during assembly is an interesting research direction.

Chapter 3

Robust Deformation Model Approximation for Robotic Cable Manipulation

In addition to the rigid objects, manipulating deformable objects are also in high demand in manufacturing, for example, wire harness and cable assembly. Unlike rigid objects, deformable objects, such as cables, wires, clothes, are soft and can deform to unexpected shapes during the manipulation. In chapter 3, 4, 5, we study the manipulation of deformable objects with applications on deformation model approximation (Chapter 3) [23], belt drive units assembly (Chapter 4) [27], and cable routing (Chapter 5) [26].

3.1 Introduction

Robotic cable manipulation has a wide range of applications, such as cable harnessing in factories, thread packing in production lines, and suturing in medical surgeries. However, these tasks are challenging for robots. Compared with rigid objects, models of cables are high dimensional and computationally expensive. Besides, such an object can easily deform to unexpected shapes during manipulation, which may make the manipulation process fail.

There are already some studies on robotic cable manipulation. Many of them are model-based methods. The deformation properties of the cable in terms of stiffness, Young's modulus, and/or FEM coefficients are required to build models for trajectory planning. However, such deformation parameters are difficult to estimate accurately and may even change during the manipulation process, especially for objects made by nonlinear elastic or plastic materials.

In this chapter, we introduce a robust online deformation model approximation method for cable manipulation planning. A deformation model is constructed to describe the relationship between the movement of the robot end-effectors and the displacement of the cable. The idea of the online deformation model approximation was first proposed by Navarro-

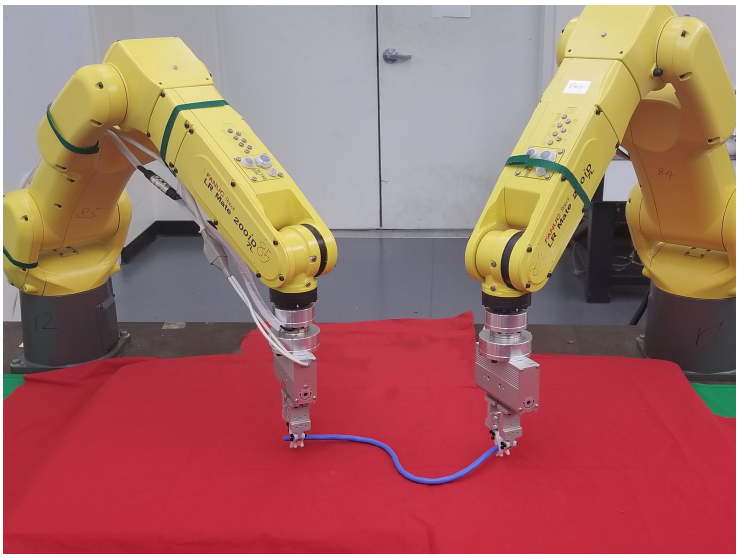


Figure 3.1: Two robots manipulating a cable to a desired shape

Alarcon [47]. Cables have infinite degrees of freedom and it is hard to find an explicit model. Instead of finding a global model, real-time data is utilized for local linear model approximation. After the model is obtained, we find the pseudo-inverse of the model, which takes the desired movement of the cable as the input and the velocity of the robot end-effectors as the output. As the robots manipulate the cable, the deformation model is updated online using real-time data.

In order to approximate the local deformation model of the cable, the motion data of the robot end-effectors and the cable are required. The motion of the end-effectors can be accurately obtained by solving forward kinematics. However, the motion of the cable is hard to obtain without the help of markers. Sensor noise and occlusions could introduce uncertainties when estimating the motion of the cable. Such uncertainties significantly affect the accuracy of the approximate deformation model.

To handle the above challenges, we propose a framework that is robust in both cable tracking and model approximation. For cable tracking, the core method we use is called structure preserved registration (SPR) [63], which is a robust non-rigid registration method for mapping one point set to another. Considering both global and local structure, SPR can robustly estimate the motion of the deformable object in real-time even in the presence of sensor noise, outliers, and occlusions. For model approximation, we take tracking uncertainties into account by solving a robust optimization problem. We formulate the problem as a 'Min-Max' problem, where the maximization takes the worst case of the measurement uncertainty into account, and the minimization penalizes the cost function to find an optimal deformation model. The deformation model is then utilized to plan a trajectory to manipulate the cable.

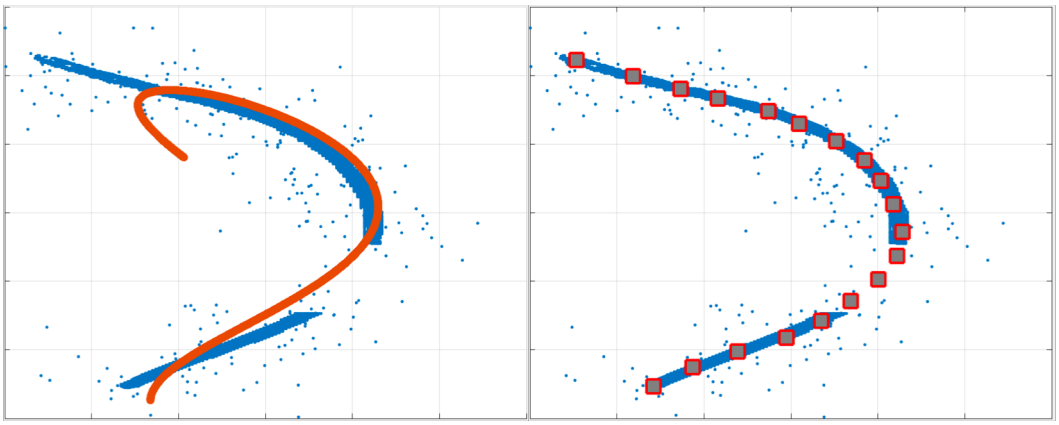
The remainder of this chapter is organized as follows. Section 3.2 introduces related works on cable tracking and manipulation. Section 3.3 describes the SPR method for cable representation and tracking. Section 3.5 explains the local model approximation method using robust optimization. Section 3.6 explains the design of the framework, which includes point tracking, local model approximation, and trajectory planning modules. Section 3.7 tests the performance of the proposed method by a series of experiments. Section 3.8 concludes the chapter and proposes future work.

3.2 Related works

Robotic cable manipulation is gaining more attentions recently for its broad applications. In order to accomplish this challenging task, a robust state estimator to track the configuration of the cable in real-time is vital. Metaxas and Terzopoulos [41] constructed a second-order dynamic model for multi-body objects, and recursively estimated the body motion from sequences of point clouds by an extended Kalman filter. Schulman et al. [**em**] proposed a modified expectation maximization (EM) algorithm for deformable object tracking,

Similarly, Petit et al. [49] introduced a finite element method for tracking elastic objects. Navarro-Alarcon et al. proposed a Fourier-based shape servoing method for deformable object representation [46]. Tang and Tomizuka used a non-rigid registration tracking method called structure preserved registration (SPR) [63]. SPR is able to estimate the positions of virtual tracking points on the deformable object in real-time robustly by considering both the local structure and the global topology of the deformable object (Fig. 3.2).

For manipulation planning, Morita et al. [43] developed a 'knot planning from observation' (KPO) system which estimated the states of ropes, especially the overlap orders by knot theory. Kudoh et al. [34] built a multi-finger hand and programmed skill motions by imitating human knotting procedures. They realized three dimensional in air knotting with diverse types of knots. However, many of these methods require empirical laws and are developed for a specific task, which is not easy to generalize for other tasks. To generalize the manipulation skills, Schulman et al. [57] proposed to teach robots to manipulate deformable objects from human demonstrations. They implemented the thin plate spline - robust point matching (TPS-RPM) algorithm [9] to warp the original trajectory taught by human demonstration to get a new trajectory which was suitable for the test scene. Tang et al. [66] proposed tangent space mapping method which guaranteed not to overstretch the cable during manipulation. Several follow-up works further improved this demonstration-based method. Tang et al. [64] proposed a uniform framework based on coherent point drift for robustly manipulating deformable linear objects. However, these methods lack the ability to achieve accurate position and can hardly apply to new scenarios which have significant differences with the training scene. Navarro-Alarcon et al. [47] proposed the idea of local deformation model approximation and then utilized the local model to automatically servo-control the soft object to desired shapes. Hu et al. [22] improved the performance by Gaussian process regression. Zhu et al. [87] extended their work by setting up a frame-



(a) Fourier-Based Estimation (b) SPR Estimation

Figure 3.2: Comparison of Fourier-based method and SPR on cable tracking. Blue dots (or the blue line) are the point cloud with outliers and occlusions, and red circles (or the red line) are the estimated position. Fourier-based method fails to estimate the state, while SPR still works well and can give the variance of estimation uncertainty.

work, Fourier-LS, which combines the truncated Fourier series visual servoing method and an efficient continuous local model approximation method. The framework proposed in this chapter has a similar structure to the Fourier-LS.

Compared with other methods, the proposed method SPR-RWLS is the first to take visual tracking uncertainties into consideration for robotic cable manipulation. As shown in Fig. 3.2, for cable tracking, compared with the Fourier-based visual servoing method, SPR works robustly in the presence of outliers and occlusions. For deformation model approximation, a novel algorithm for solving robust weighted least squares is introduced in this chapter. The robust local deformation model for trajectory planning can be obtained efficiently by solving several second-order cone program (SOCP) in parallel. We show that our method is able to obtain robust deformation models in different scenarios with modest computational cost by several experiments.

3.3 Structure preserved registration

In order to track the shape of cable, we select finite number of virtual tracking points along the cable to represent the state of the cable. In Fig. 3.3, the blue dots represent the tracking points, which we select to track the shape of the cable, and the red circles are the feature points, which we use to solve the deformation model in later section. Usually, feature points are a subset of tracking points. Because of the sparsity of feature points, we can assume

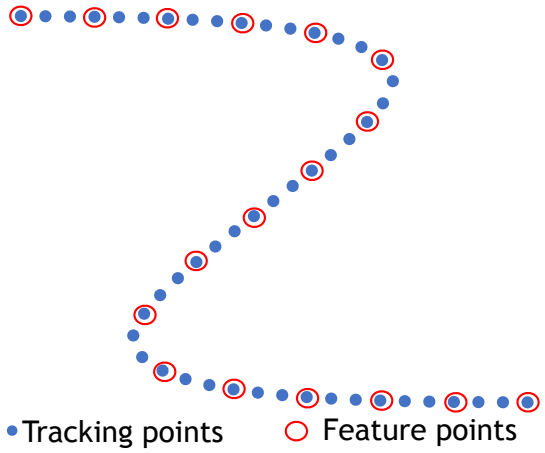


Figure 3.3: Illustration of tracking points and feature points.

that the position of every feature point is uncorrelated with other feature points.

Gaussian Mixture Model

We need to estimate the tracking points from the dense and noisy point cloud. Gaussian mixture model (GMM) is utilized here to estimate the positions of tracking points. In GMM, those tracking points are represented by Gaussian centroids and the point cloud is a set of randomly sampled points from this GMM. We denote the Gaussian centroids as $X^t = [x_1^t, x_2^t, ..., x_N^t] \in R^{N*D}$, where x_i^t is the position of *i-th* centroid at time step t, N is the number of tracking points and D is the dimension of the position for each tracking point. The point cloud of cable is denoted as $Y^t = [y_1^t, y_2^t, ..., y_M^t] \in R^{M*D}$, where y_i^t is the position of *i-th* point in the point cloud. M is the total number of points in point cloud and usually $M >> N$. Then the point cloud distribution can be expressed as

$$\begin{aligned} p(y_m^t) &= \sum_{n=1}^{N} \frac{1}{N} \mathcal{N}(y_m^t; x_n^t, \sigma^2 \mathbf{I}) \\ &= \sum_{n=1}^{N} \frac{1}{N} \frac{1}{(2\pi\sigma^2)^{D/2}} \exp(-\frac{\|y_m^t - x_n^t\|^2}{2\sigma^2}) \end{aligned} \tag{3.1}$$

All the Gaussians share the same weight. To account for the noise and the outliers of the point cloud, an additional uniform distribution is added to $p(y_m^t)$:

$$p(y_m^t) = \sum_{n=1}^{N+1} p(n) p(y_m^t | n) \tag{3.2}$$

with

$$p(n) = \begin{cases} (1-\mu)\frac{1}{N}, & n = 1, \ldots, N \\ \mu, & n = N+1 \end{cases} \tag{3.3}$$

$$p(y_m^t|n) = \begin{cases} \mathcal{N}(y_m^t; x_n^t, \sigma^2\mathbf{I}), & n = 1, \ldots, N \\ \frac{1}{M}, & n = N+1 \end{cases} \tag{3.4}$$

where μ denotes the weight of the uniform distribution.

The Gaussian centroid and corresponding covariance that best represent the point cloud of cable can be obtained by maximizing the following log-likelihood function.

$$L(x_n^t, \sigma^2|Y^t) = log \prod_{m=1}^{M} p(y_m^t) \tag{3.5}$$

$$= \sum_{m=1}^{M} log(\sum_{n=1}^{N+1} p(n)p(y_m^t|n)) \tag{3.6}$$

The above maximization problem is non-convex due to the summation inside $log(.)$ function. Hence, the problem is hard to solve efficiently. Instead of solving the above problem, we aim to maximize the lower bound for (3.6).

$$Q(x_n^t, \sigma^2) = \sum_{m=1}^{M} \sum_{n=1}^{N+1} p(n|y_m^t) log(p(n)p(y_m^t|n)) \tag{3.7}$$

It can be proved by Jensen's inequality [**lower˙bound**] that (3.7) is the lower bound of (3.6). It can be efficiently solved by using EM algorithm in which we iteratively update the GMM parameters and posteriori probability distribution. In the expectation(E) step, we compute the posteriori probability distribution using GMM parameters from the last maximization(M) step. In M step, we compute the new GMM parameters using posteriori probability distribution from the last E step. By iteratively executing E-M step, the log-likelihood function will converge and we can get a pair of local optimal parameters (x_n^{t*}, σ^{2*}) of GMM.

Structure Preserved Registration

Although the above GMM can register the point cloud of the cable to several tracking points, the estimation performance is poor especially when a part of the point cloud is missing, for example when occlusion happens during perception, which is very common in robot manipulation. The major problem is that there are no constraints on the positions of Gaussian centroids between different time steps. In reality, the cable cannot move arbitrarily and its deformation must follow some topological constraints. Globally those registered Gaussian centroids must form a smooth curvature, and locally those Gaussian centroids should maintain certain distance with their neighborhood.

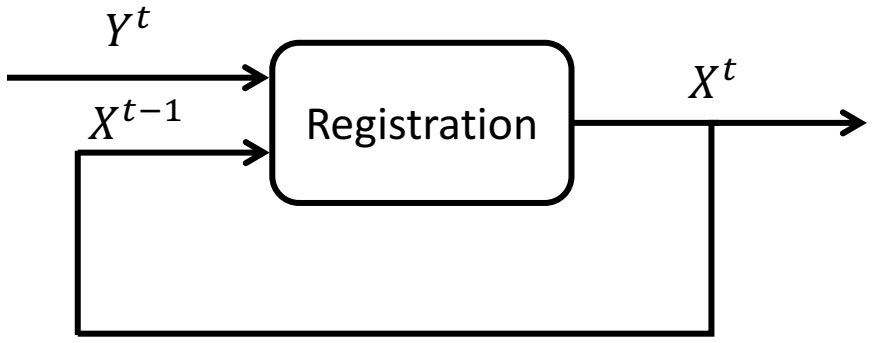


Figure 3.4: Framework of point set registration. Y^t is the perceived point cloud at time step t. X^{t-1} is the state estimated at the previous step. A new estimate X^t is achieved by registering X^{t-1} to Y^t .

To deal with this problem, we introduce both global and local structure regularization to (3.7) in GMM registration.

$$\tilde{Q} = Q(x_n^t, \sigma^2) - \frac{\tau}{2} E_{Local} - \frac{\lambda}{2} E_{Global} \tag{3.8}$$

where $\tau \in R^+$ and $\lambda \in R^+$ are trade-off weights that balance the regularization on local and global structure. E_{Local} and E_{Global} are regularization terms which will be explained in the following.

For local structure, any point at time step $t-1$ can be characterized as a weighted sum of its neighbor points. That is $x_n^{t-1} = \sum_{i \in I_n} S_{ni} \cdot x_i^{t-1}$, where S_{ni} is the weight matrix and I_n is the set for K nearest points to x_n^{t-1}. When the cable deforms to another shape at time step t, the position of tracking points might change, but their relative local structure is expected to be maintained, which means at time step t, $x_n^t \approx \sum_{i \in I_n} S_{ni} \cdot x_i^t$. The local structure weights S_{ni} can be obtained by solving a least squares problem. There could be many sub-optimal weights due to the singularity of matrix when solving least squares, so we integrate all L sub-optimal weights to characterize the local structure. More details can be found in [63].

$$E_{Local} = \sum_{n=1}^{N} \sum_{l=1}^{L} || \sum_{i=1}^{N} S_{ni}^{(l)} x_i^t ||^2 \tag{3.9}$$

Global structure should also be preserved during registration. Since cable in real world cannot move arbitrarily, the registered tracking points should also follow a smooth trajectory in neighboring time step. In order to preserve the global structure, we regularize the coherent movement $x_n^t = v(x_n^{t-1})$. $v : R^D \to R^D$ is a transformation function which should to be as

smooth as possible. The smoothness can be evaluated by $\int_{R^D} \frac{|V(s)|^2}{G(s)}$, where $V(s)$ is the Fourier transform of v and $G(s)$ is a low-pass filter.

$$E_{Global} = \int_{R^D} \frac{|V(s)|^2}{G(s)} ds \tag{3.10}$$

Substituting (3.9) and (3.10) into (3.8) we obtain the modified likelihood function $\tilde{Q}$.

$$\tilde{Q} = Q(x_n^t, \sigma^2) - \frac{\tau}{2} \sum_{n=1}^{N} \sum_{l=1}^{L} || \sum_{i=1}^{N} S_{ni}^{(l)} x_i^t ||^2 - \frac{\lambda}{2} \int_{R^D} \frac{|V(s)|^2}{G(s)} ds \tag{3.11}$$

Though the global and local structure regularization can be formulated in other different ways, the reason for the above regularization is to obtain closed-form solution for it. This is crucial if we want to solve the problem in real-time. More details about SPR and the proof of the existence of closed-form solution can be found in [63].

3.4 Point Cloud Recovery

The state estimator registers the old simulated rope to the point cloud in order to update the simulation. During tracking, human hands or robot arms may occlude the object resulting in a partially observed point cloud. From experiments, we found SPR [63] is robust when the occluded part is small. However, if a large portion of the point cloud is missing (Fig. 3.5) or the tip of the cable is occluded, SPR would fail to estimate the correct cable state.

Inspired by the background subtraction method in computer vision, we proposed to use a foreground mask to recover the occluded point cloud (Fig. 3.6). Fig. 3.6 (a) shows the environment background captured with a stationary RGB-depth camera. (b) and (c) are RGB images with their color-filtered point clouds in time step $t-1$ and t. (d) is the foreground mask constructed by subtracting background from frame t.

The goal of the point cloud recovery is to complete point cloud in frame t using the foreground mask and the recovered point cloud in frame $t-1$. Fig. 3.7 provides an example of the process. First, point clouds at frame $t-1$ and t are projected to the RGB image plane as (a) and (c) show. Second, we compute a foreground mask in frame t as shown in (b). Third, the projected image in frame $t-1$ is multiplied in pixel with the mask to obtain the complementary part. Finally, the complement is combined with origin point clouds in frame t to get (d). The advantage of this operation is that it could distinguish rope movement from occlusion. For the convenience of discussion, we divide rope's point clouds in each frame into four parts and labeled 1 to 4 in Fig. 3.7. The point cloud in the frame t (c) at part 3 is missing compared to the frame $t-1$ (a). However, the foreground mask (b) shows no occlusion at part 3, so the corresponding point cloud at frame $t-1$ will be discarded. The proposed algorithm is summarized in Algorithm 3.

Due to sensor noise and mismatch on RGB and depth cameras, there are alignment errors when constructing the foreground mask. This would result in a segmentation error of the

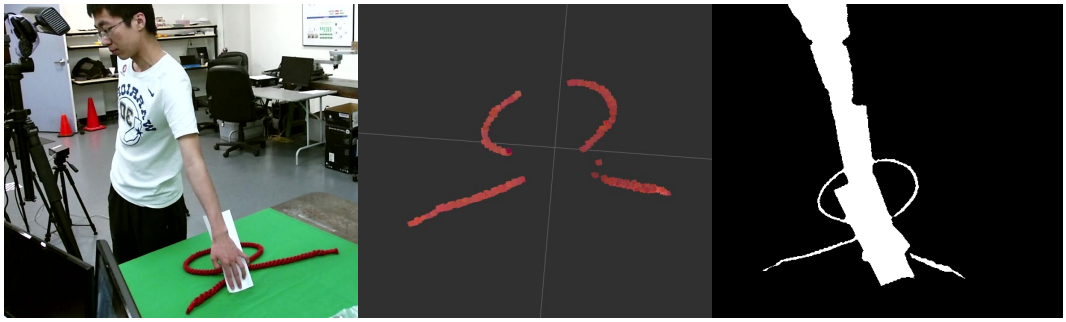


Figure 3.5: (left) Kinect is occluded by human arm. (middle) Partial observed point cloud. (right) Foreground mask.

point cloud. In our case, we prefer a false-negative mask rather than a false-positive. In other words, we could discard some occluded points, but not preserve inexistent points. Thus, an erosion operation is applied on the foreground mask to avoid false-positive masks.

Algorithm 2: Point Cloud Recovery

```
Initialize the environment, record the background;
Put the rope in the environment, obtain the point cloud of the rope using a color
  filter, t+ = 1;
while Tracking do
    Obtain the point cloud of the rope using a color filter;
    Obtain the foreground mask by subtracting the background from the current
      frame;
    for each pixel of the frame do
        if point cloud in frame t is True then
            preserve the point cloud ;
        else if point cloud in frame t − 1 is True and foreground mask is True then
            preserve the point cloud ;
        else
            discard the point cloud ;
        end
    end
    t = t + 1;
end
```

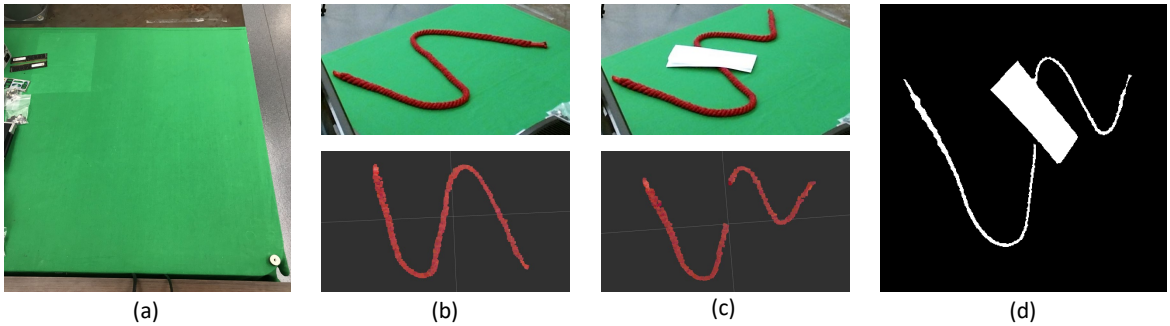


Figure 3.6: (a) Background. (b) $t-1$ frame. (c) t frame. (d) Foreground mask.

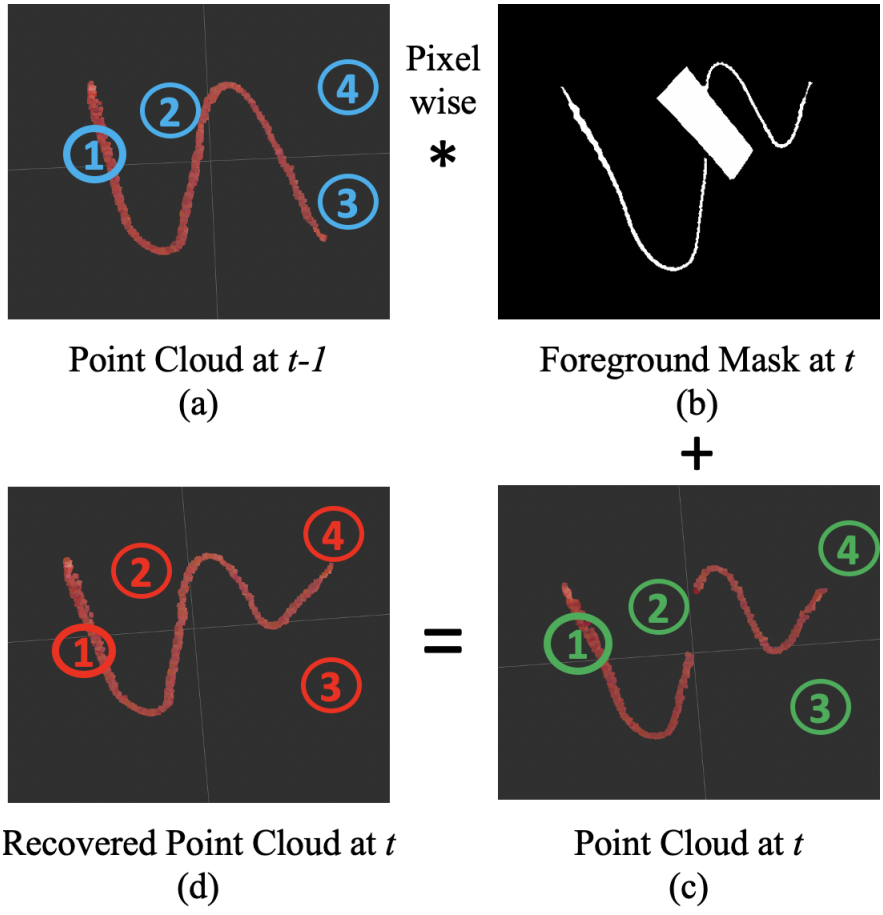


Figure 3.7: Point Cloud Recovery

3.5 Local Linear Deformation Model

Deformation Model

To approximate the deformation model, we uniformly select several feature points along the cable, which are a subset of the tracking points (Fig. 3.3).

By holding two tips of the cable, end-effectors of the robots are assumed to be fixed with cable tips. We can build a deformation model for describing the interaction between the end-effectors and the cable. Assuming that there are N_f selected feature points on the cable and the degrees of freedom for each point is D. The state of the cable is denoted as $c = [c_1, c_2, ..., c_{N_f}]^T \in R^{N_f \times D}$, where $c_i = [c_{i1}, c_{i2}, ..., c_{iD}] \in R^{1 \times D}$, c_{ij} denotes the coordinate of the $i\text{-}th$ point in the $j\text{-}th$ direction, for example in 2D space, $c_{5,2}$ denotes the second direction of the 5-th selected point. Assume that there are L manipulators and each end-effector has K degrees of freedom. The motion of the robots end-effectors is denoted as $r = [r_{11}, r_{12}, ..., r_{1K}, ..., r_{LK}] \in R^{L \times K}$. The local linear model we used is expressed in (3.12) [47]. As shown in (3.12) the desired local linear model is $\frac{\delta c}{\delta r}$, and each row of $\frac{\delta c}{\delta r}$ is decoupled with each other, therefore we can make use of parallel computation to find linear model $\frac{\delta c_i}{\delta r}$ for each direction simultaneously, which can greatly improve the efficiency.

$$\begin{aligned} \delta c(t) = \begin{bmatrix} \delta c_1(t) \\ \vdots \\ \delta c_{N_f}(t) \end{bmatrix} = \frac{\delta c}{\delta r}(t)\delta r(t) = \begin{bmatrix} \frac{\delta c_1}{\delta r}(t) \\ \vdots \\ \frac{\delta c_{N_f}}{\delta r}(t) \end{bmatrix} \delta r(t) \\ = A(t)\delta r(t) \end{aligned} \tag{3.12}$$

Local Linear Model

The time-varying deformation model is difficult to obtain due to its high dimensions, non-linear behaviors and configuration dependent properties. Actually, it is unnecessary to find a global deformation model which is suitable for every possible cable configuration. If the displacement of the robots end-effectors is small enough, the local deformation model can be approximated with linear functions.

The local linear deformation model is expressed in (3.12), where $A(t) \in R^{N_f \times LK}$ is a time-varying Jacobian Matrix, which represents the relation between the movement of the robots and the movement of the feature points. Remember that our goal is to plan trajectory for the robots end-effectors to manipulate the deformable cable to a desired shape. To achieve this, we try to find the motion of the robots $\delta r(t)$ given the desired displacement of the cable $\delta c(t)$. For convenience, instead of calculating the Jacobian matrix $A(t)$, we directly find the pseudo-inverse of Jacobian matrix $G(t) = A^{\dagger}(t)$. Since $A(t) \in R^{N_f \times LK}$, in practice, the number of feature points on the cable is always larger than the DOF of robots end-effectors $N_f >> LK$. Therefore, we can guarantee that $G(t)$ exists.

To estimate $G(t)$, we denote the current time as t_m. Using a constant sampling period δt, within the time period $(m-1)\delta t$, we collect m consecutive data of δc_i and δr_i while the robot is moving:

$$\delta C(t_m) = \begin{bmatrix} \delta c(t_1) & \delta c(t_2) & . & . & . & \delta c(t_m) \end{bmatrix} \in R^{N_f \times m}$$
$$\delta R(t_m) = \begin{bmatrix} \delta r(t_1) & \delta r(t_2) & . & . & . & \delta r(t_m) \end{bmatrix} \in R^{LK \times m}$$

The local linear model can be found by solving (3.13), which can be decomposed to a sum of several least squares. $G_n^T(t)$ represents the $n\text{-}th$ column of the matrix $G^T(t)$, and similarly $\delta R_n^T(t)$ represents the $n\text{-}th$ column of $\delta R_n^T(t)$.

$$\begin{aligned} G(t)^* &= \underset{G(t)}{arg\min} \quad \|\delta C^T(t) G^T(t) - \delta R^T(t)\|_F^2 \\ &= \sum_{n=1}^{LK} \underset{G_n(t)}{arg\min} \quad \|\delta C^T(t) G_n^T(t) - \delta R_n^T(t)\|_2^2 \end{aligned} \tag{3.13}$$

In the next subsection, we will introduce how to improve the robustness of the local model approximation by using robust optimization.

Robust weighted least squares

The displacement of the robots end-effectors $\delta R(t)$ can be calculated accurately using the robotic forward kinematics. However, $\delta C(t)$ is an estimation with lots of uncertainties from visual tracking. If uncertainties are not considered, we may fail to recover a suitable local deformation model. Uncertainty in $\delta C(t)$ can be approximated by Gaussian distribution, so we are able to bound it inside a certain area given a confidence probability.

In SPR, we regard the tracking points in the last time step as Gaussian centroids and each point in the new point cloud as a sample from the Gaussian mixture model. The objective is to maximize the log-likelihood of the point cloud sampled from the GMM. Thus it is reasonable to regard the variance of each Gaussian as the uncertainty of this movement. Taking the $i\text{-}th$ point as an example, $\mu(c_i(t))$ is the mean of the $i\text{-}th$ point at time step t, and σ_t is the uncertainty from time step $t-1$ to t. Besides, we assume that each Gaussian has an equal membership probability $1/N$ and a consistent isotropic covariance $\sigma^2 I$ in SPR registration. So all the selected tracking points on the cable have the same variance σ_t at time step t, and all the feature points are uncorrelated.

$$\delta C^T(t) = \mu(\delta C^T(t)) + \Delta \tag{3.14}$$

(3.14) shows the uncertainty in the matrix $\delta C^T(t)$ for robust optimization, where Δ describes the uncertainty. From the above analysis, the $j-th$ column of the matrix Δ can be regarded as a sample from a Gaussian Distribution $N(0, \Sigma)$, where $\Sigma = diag(\sigma_1^2, \sigma_2^2, ..., \sigma_m^2)$.

We rewrite (3.13) in the form of robust optimization in (3.15),

$$\sum_{n=1}^{LK} \min_{G_n(t)} \max_{\|\Delta\|_2 \leq s} \quad \|W[(\mu(\delta C^T(t)) + \Delta)G_n^T(t) - \delta R_n^T(t)]\|_2^2 \tag{3.15}$$

where s is the upper bound or equivalently the largest singular value of Δ, and $W = diag(w_1, w_2, ..., w_m)$ is a weight matrix for the data from different time steps.

When solving the robust optimization problem (3.15), a tight bound s is preferred. Each column of Δ can be regarded as a random sample from the Gaussian distribution, and there are some existing works in statistics to bound the largest singular value with a desired probability.

Theorem 1 *Let $\Delta \in R^{m \times n}$ be drawn according to the Σ-Gaussian ensemble. Then for all $\delta > 0$, the maximum singular value $\sigma_{max}(\Delta)$ satisfies the upper deviation inequality,*

$$P[\frac{\sigma_{max}(\Delta)}{\sqrt{n}} \leq \gamma_{max}(\sqrt{\Sigma})(1+\delta) + \sqrt{\frac{trace(\Sigma)}{n}}] \geq 1 - e^{-n\delta^2/2}$$

where $\gamma_{max}(\sqrt{\Sigma})$ denotes the largest eigenvalue of $\sqrt{\Sigma}$.

Theorem 1 provides a theoretically tight bound of the random matrix Δ which is proved in Chapter 6 of [**thm1**]. Using this theorem, we can find an upper bound of the largest singular value of uncertainty matrix Δ given a desired probability.

Theorem 2 *Any robust least squares in the form:*

$$\min_{x \in R^n} \max_{\|\Delta\|_2 \leq s} \quad \|(A + \Delta)x - b\|_2$$

is equivalent to a SOCP problem:

$$\min_{x \in R^n} \quad \|Ax - b\|_2 + s\|x\|_2$$

For fixed x, and using the fact that the Euclidean norm is convex, we have

$$\|(A + \Delta)x - b\|_2 \leq \|Ax - b\|_2 + \|\Delta x\|_2$$

By the definition of the largest singular value norm, and given our bound on the size of the uncertainty, we have

$$\|\Delta x\|_2 \leq \|\Delta\|_2 \|x\|_2 \leq s\|x\|_2$$

Thus, we have a bound on the objective value of the robust least squares problem:

$$\max_{\|\Delta\|_2 \leq s} \quad \|(A + \Delta)x - b\|_2 \leq \|Ax - b\|_2 + s\|x\|_2$$

The upper bound is actually attained by

$$\Delta = \frac{s}{\|Ax-b\|_2\|x\|_2}(Ax-b)x^T$$

Therefore, the robust weighted least squares (RWLS) in (3.15) can be written in the form of a summation over several SOCPs as shown in (3.16). For each SOCP, we find one column of the model matrix $G^T(t)$. The columns of $G^T(t)$ do not depend on each other, which means that we can make use of parallel computation to solve each SOCP and greatly increase the efficiency of the solution process.

$$\sum_{n=1}^{LK} \min_{G_n(t)} \quad \|W\mu(\delta C^T(t))G_n^T(t) - W\delta R_n^T(t)\|_2 + s\|WG_n^T(t)\|_2 \tag{3.16}$$

If the matrix $G(t)$ is obtained, it means that the local model at time step t is approximated. Given a desired movement of cable, we are able to obtain the trajectory of the robots end-effectors.

3.6 Framework Details

Algorithm Overview

Combining the above SPR estimation with RWLS for solving local deformation model, we propose our method 'SPR-RWLS' to manipulate soft cables to desired shapes. SPR is utilized to estimate the cable state in real-time. Robust weighted least squares (RWLS) is used to obtain a robust local deformation model considering tracking uncertainties. The proposed method is summarized in Algorithm 3.

Algorithm 3: SPR-RWLS

```
Initialize cable, and record desired cable shape;
Using SPR to get initial and desired tracking points along the cable;
Downsample tracking points with a fixed index to get feature points;
Initialize data set D(δR, δC) by randomly executing robot δR and collecting
  corresponding movement of cable δC for m_0 times;
while diff(c_current, c_desired) > ϵ do
    Compute weight matrix W;
    Solve Robust Weighted Optimization for local deformation Jacobian Matrix G(t);
    Compute δr = λG(t)δC_desired;
    Execute δr, collect new δc by SPR;
    Append δr and δc to dataset D;
end
```

Cable Tracking

We select $N_{tracking}$ tracking points uniformly distributed along the initial point cloud. At time step t, the tracking points are denoted as $X^t = \{x_1^t, x_2^t, ..., x_{N_{tracking}}^t\}$, where $x_n^t \in R^2$. At the next time step $t+1$, the cable deforms to a new shape, and its point cloud $Y^{t+1} = \{y_1^{t+1}, y_2^{t+1}, ..., y_M^{t+1}\} \in R^2$ is captured by the camera. By applying SPR registration as described in Section III, the node positions X^k can be smoothly registered towards the point cloud Y^{t+1}, and we can get the new estimation of tracking point positions X^{t+1}, and the variance σ_{t+1} of this step.

As shown in Fig. 3.4, running the above procedure iteratively, we can obtain the estimated position of each tracking point in real time.

Deformation Approximation

We select $N_{feature}$ feature points from the tracking points just as Fig. 3.3 shows. Also at time step t, the movement of each feature point $\delta c(t) = c(t) - c(t-1)$, and the tracking uncertainties of this step σ_k can be obtained from SPR registration. The movement of the robots end-effectors δr can also be calculated by forward kinematics. Then we append these new data $\delta c(t)$, $\delta r(t)$, and σ_t to data set $\delta C(t)$, $\delta R(t)$, and $\Sigma(t)$ respectively. Before we calculate the deformation model, we assign weights to different $(\delta r, \delta c)$ pairs in the data set. We rank the data pairs in data set based on their mean squared errors to the current cable shape. A discount factor γ is used to assign weights to different data pairs. The discount factor is a tuning parameter and we use 0.95 in our experiments. Besides, the bound of the uncertainties can be efficiently calculated by a given confidence probability. In practice, we set the confidence probability larger than 99%. Finally, the local deformation model $G(t)$ can be solved by the robust optimization problem (3.16).

By running this algorithm iteratively, we are able to get an approximation of the deformation model in real-time.

Trajectory Planning

After the deformation model $G(t)$ is obtained at time step t, we first need to compute the desired movement of the cable $\delta c_{desired}(t)$ in order to get the motion of the robots. For the scenarios which desired shape is far away from the initial shape, several intermediate desired shapes $c_{intermediate}$ are preferred to be generated and reached in sequence. In order to achieve such δC, the desired movement of end-effectors is computed using $\delta r(t) = \lambda G(t) \delta c_{desired}(t)$, where λ is a gain we need to tune. In order to make the cable moving in a low speed without vibration and make sure the deformation model is locally accurate, the gain λ is chosen to be small. In our experiment, λ is set to 0.1.

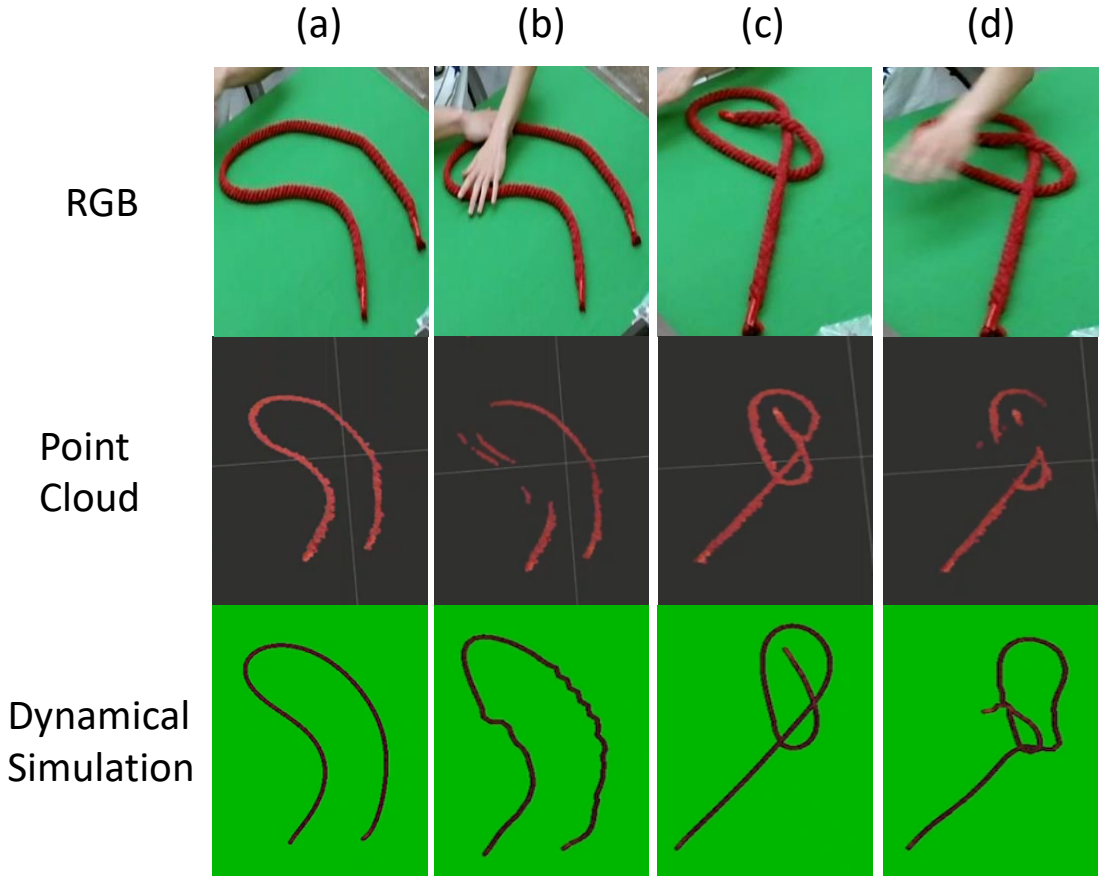


Figure 3.8: Without point cloud recovery. Waving hands above the rope without touching the rope. Registration fails and the simulated rope deforms to unexpected shapes due to point cloud missing.

3.7 Experiments and Results

To test the proposed point cloud recovery module, several representative rope tracking tasks were conducted.

Shown in Figs. 3.5, 3.8, and 3.9, a 1-meter-long red rope was placed on a green table. A Microsoft Kinect (version 2) was utilized to get point cloud of the rope. The simulated rope in the Bullet Physics Engine was modeled by fifty linked capsules with density $1.5g/cm3$. The stiffness gain K_p and damping gain K_d were set as $10N/m$ and $0.5Ns/m$ respectively in the feedback linearization controller.

During the experiment, we manipulated the rope at a moderate speed. Fig. 3.8 shows the tracking results using the previous work [63], which do not have point cloud recovery module

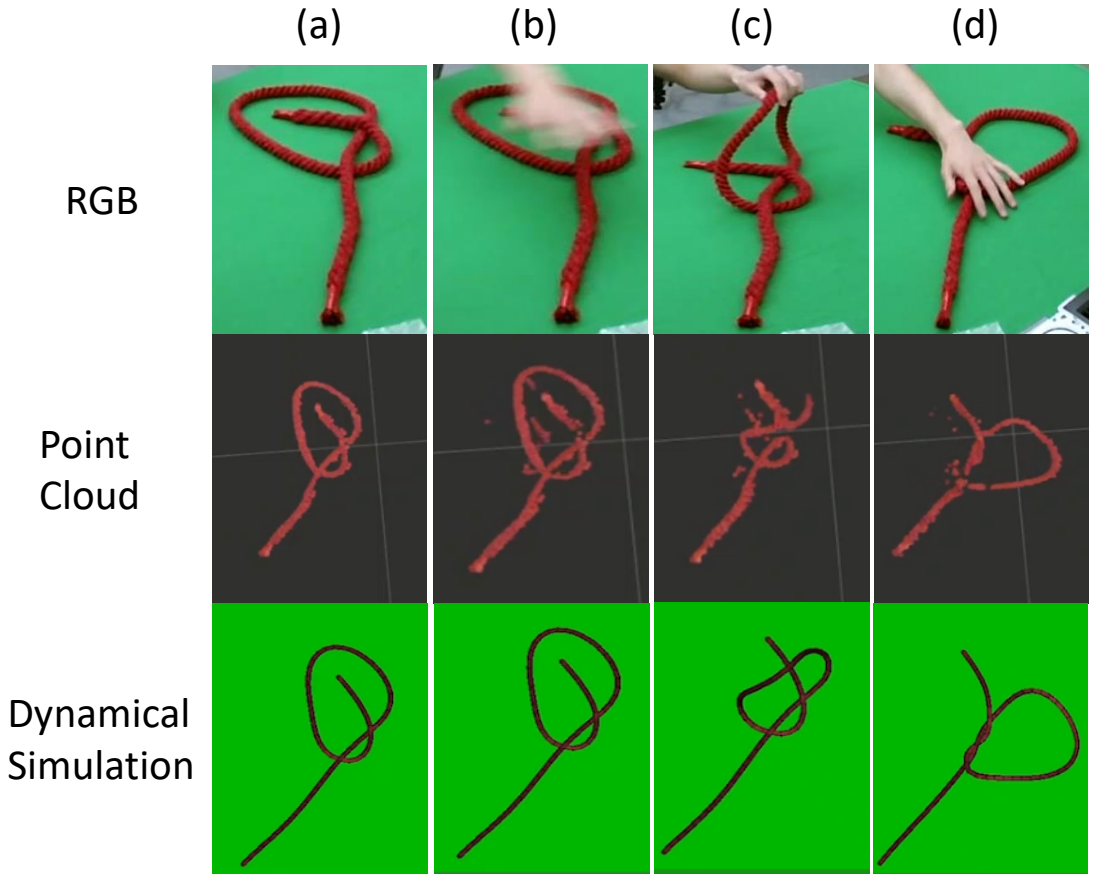


Figure 3.9: With point cloud recovery. Tracking is robust to large occlusion.

and feedback linearization controller. The tracking result is not stable due to missing point cloud. Fig. 3.9 shows the tracking result under occlusion with our proposed framework. A large portion of point cloud is missing due to occlusions. With the point cloud recovery module, we can recover most of the point cloud as the middle row images show, which improves the performance of node registration. Some point cloud, however, may still be missing when the rope is moving under occlusion even with the point recovery module. Such is a case when human holds the rope in hands as Fig. 3.9(c) shows. In this case, the foreground mask does not have historical information to recover the points until the occluded part is exposed to the Kinect again.

We conduct several experiments on two FANUC LR-Mate 200iD robots to show that SPR-RWLS is efficient and robust in the presence of outliers and occlusions for robotic cable manipulation.

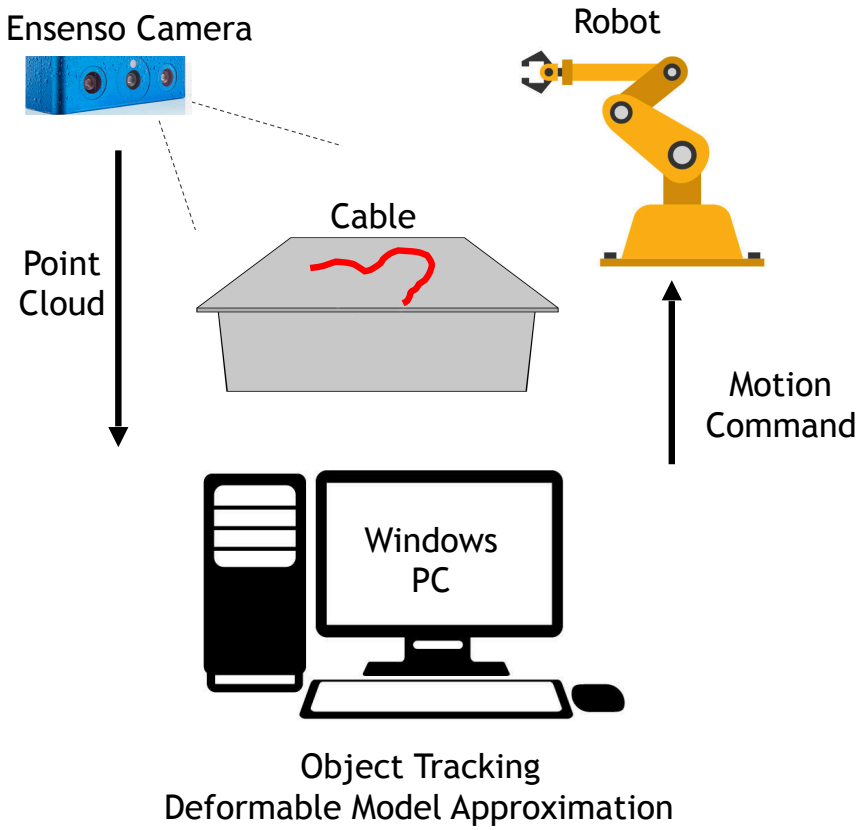


Figure 3.10: The testbed setup

Cable Tracking

As shown in Fig.3.10, an IDS Ensenso N35 stereo camera was utilized to monitor the environment. The captured point cloud was sent to a Windows 10 desktop (Intel i7@3.60 GHz + RAM 8GB), which ran SPR registration algorithms in real-time in MATLAB. Using a color filter, the point cloud of the cable was extracted from the red background. Since the cable was manipulated in a two dimensional plane, $3D$ point cloud was projected to the $2D$ plane. Given the initial point cloud of a straight cable, we manually selected 55 tracking points uniformly distributed along the cable. When the cable deformed to a different shape in the next step, we registered the newest point cloud to the point cloud from the last time step using SPR. Then the corresponding 55 tracking points which represent the current cable state were obtained.

Tracking results show that SPR cable tracking module is able to robustly track the movement of the cable in real-time. SPR outperforms Fourier-Based estimation when the point cloud is contaminated by outliers, noise, and occlusions. When manually adding white

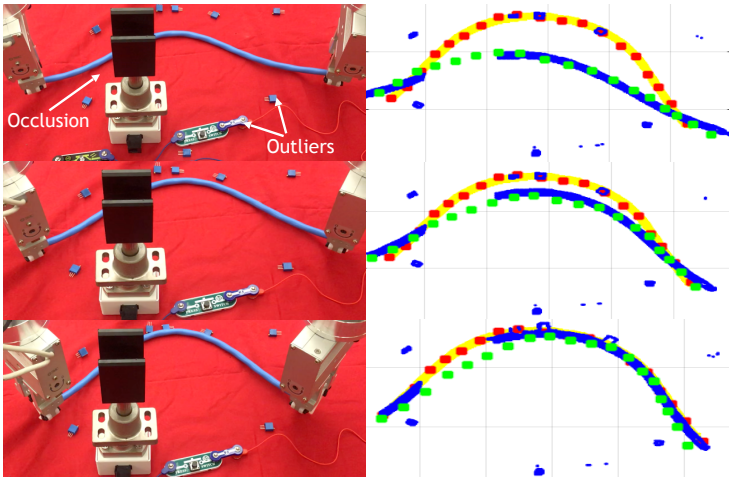


Figure 3.11: SPR cable tracking in the presence outliers and occlusions. In (b), blue dots represent the perceived point cloud; yellow dots represent the target shape; red squares represent the desired feature points; and green squares represent current feature points.

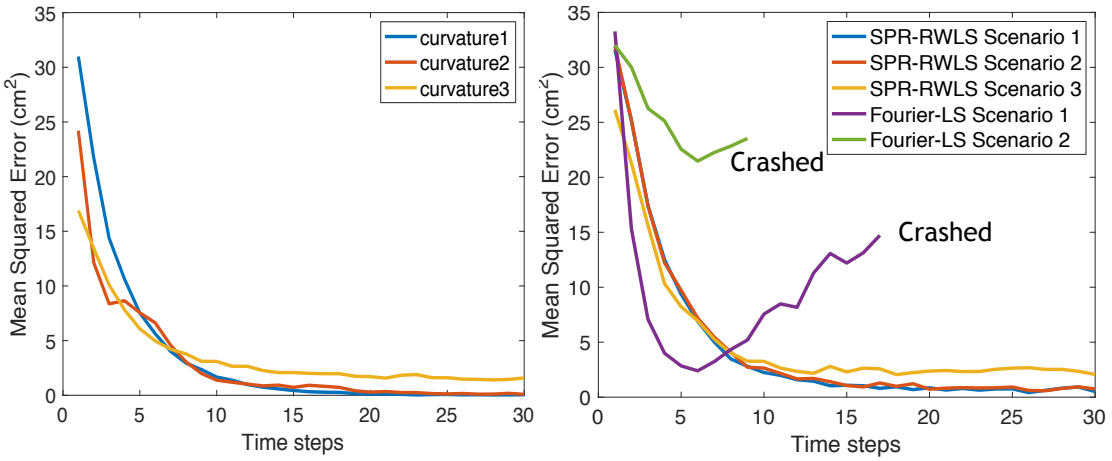
noise to the point cloud and removing 20% of the point cloud from the middle part to simulate occlusions (Fig. 3.2), Fourier-Based estimation fails tracking the cable, while SPR is still able to provide a robust estimation of the cable position as well as estimation of the uncertainties. Fig.3.11 shows SPR tracking performance in the presence of occlusions and outliers due to objects of similar colors to the cable appear in the work space.

Cable Manipulation

To evaluate the performance of the proposed framework, we conducted experiments to manipulate cables of different thickness under different scenarios. The goal is to manipulate the cable from straight lines to given desired shapes. Experimental videos can be found in [2].

The robots end-effectors are parallel grippers which can open and close. When closing, the gripper can clamp the cable firmly without any slipping. Since the experiment is conducted on a $2D$ plane, each end-effector has 3 degrees of freedom including two orthogonal displacement on horizontal plane and one rotation along axis that is perpendicular to the horizontal plane. For the model approximation, ECOS [10], an efficient SOCP solver, was utilized to solve the problem (3.16).

Given the positions and variances of 55 estimated tracking points from SPR, we select 19 feature points from the tracking points to estimate the deformation model. Each feature point has two degrees of freedom in $2D$ plane. Using Algorithm 3 introduced in section V, several experiments are conducted to test the proposed framework. We use the mean



(a) SPR-RWLS performance

(b) Comparison of Fourier-LS and SPR-RWLS under outliers and occlusions

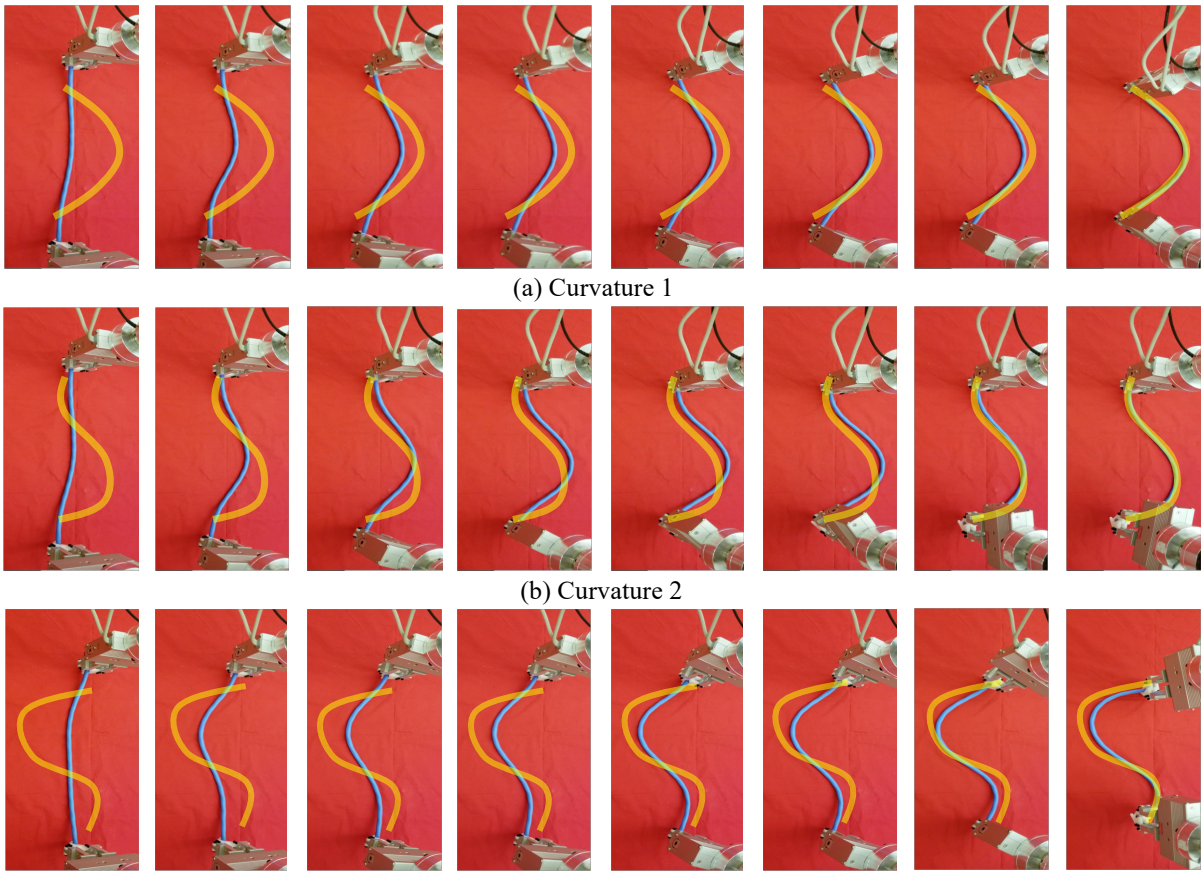
Figure 3.12: Mean Squared Error vs Timesteps. (a) shows the results of manipulation a cable to different curvatures, which are shown in Fig. 3.13, and (b) compares SPR-RWLS with Fourier-LS in different scenarios, which are introduced in Section VI.B

squared distance errors of 55 tracking points between the cable and the desired shape to evaluate the manipulation performance.

A sequence of snapshots is shown in Fig.3.13. The cable was successfully manipulated from a straight line to given desired curvatures. For a simple desired shapes (Fig.3.13 (a), (b)), the manipulated cable overlaps with the desired shape perfectly with the mean squared error (MSE) smaller than 0.08 cm^2. For complicated desired shapes with more curvatures (Fig.3.13 (c)), the MSE is about 1.5 cm^2. Fig. 3.12 (a) shows that the MSE converges over time steps.

The performance of the algorithm with different cables is analyzed in Table 5.1. We conducted experiments with two cables of different diameters. One Ethernet cable has a diameter of $4.04mm$, and the other cable has a diameter of $8.10mm$. We manipulate both cables to the same simple desired shape (curvature 1 in Fig.3.13) 10 times. Both experiments have very high success rates. The thinner cable has a little bit higher MSE. It is reasonable because the thinner cable is more likely to deform, which makes the deformation model not accurate.

We also performed several experiments to show that SPR-RWLS is able to perform robustly in the presence of outliers and occlusions. As shown in Table 5.2, we conducted experiments to manipulate the blue Ethernet cable to curvature 1 in 4 different scenarios. In the first scenario where there are no outliers and occlusions, both Fourier-LS and SPR-RWLS performe very well. Uncertainty scenario 1,2 and 3 are the scenarios in the presence of noise, outliers, and occlusions. In uncertainty scenario 1, we manually occluded 20% of the point



(a) Curvature 1

(b) Curvature 2

(c) Curvature 3

Figure 3.13: Snapshots of the cable manipulation experiments. Two robot arms were collaborating to manipulate a cable to desired shapes, which are shown by the yellow lines. (a), (b), and (c) show three different curvatures.

Table 3.1: Experiments with different cables

Cable diameters	Success rate	Mean squared error (cm^2)
$4.04mm$	9/10	0.242 ± 0.079
$8.10mm$	10/10	0.051 ± 0.014

cloud, and added 5% white noise with $\sigma = 10\%\delta c$. In uncertainty scenario 2, we occluded 25% point cloud, and added 10% white noise with $\sigma = 15\%\delta c$. Uncertainty scenario 3 is shown in Fig.3.11, where objects of color similar to the Ethernet cable are placed on the table and part of the Ethernet is occluded by other objects. In Table 5.2, we can clearly see that SPR-RWLS outperforms Fourier-LS [87] in the presence of noise, outliers, and occlusions.

Table 3.2: Comparison of mean squared error of our robust method and Fourier-LS Method

Feature	Fouriers-LS	SPR-RWLS
No outliers and occlusions	$0.045 \pm 0.026 cm^2$	$0.051 \pm 0.014 cm^2$
Uncertainty scenario 1	$14.712 \pm 1.5832 cm^2$	$0.411 \pm 0.093 cm^2$
Uncertainty scenario 2	$23.45 \pm 1.259 cm^2$	$0.629 \pm 0.1623 cm^2$
Uncertainty scenario 3	fail	$2.503 \pm 0.438 cm^2$

3.8 Chapter Summary

In this chapter, a novel framework SPR-RWLS for cable manipulation is proposed. In the framework, we combine real-time cable tracking and online deformation model approximation. For real-time tracking, a Gaussian mixture model based non-rigid registration is able to track deformable cable robustly in the presence of sensor noise, outliers, and occlusions. For deformation model approximation, the local deformation model can be approximated online by solving a robust optimization in parallel under uncertainty. Experiments showed that the proposed method is able to manipulate deformable cable to desired shape robustly.

For future work, we plan to test the performance on more complicated desired shapes. For a desired shape that is far from the initial shape or when the cable is too long, we plan to develop a method that can automatically and efficiently generate intermediate desired shapes. SPR is proved to be robust for deformable cable tracking in $3D$ space. More manipulation tasks with deformable cable and deformable $2D$ cloth will be tested in $3D$ space.

Chapter 4

Trajectory Optimization for Manipulation of Deformable Objects: Assembly of Belt Drive Units

4.1 Introduction [1]

As we studied in Chapter 3, we can robustly manipulate deformable cables via online approximating the deformation model. However, the scenarios may be more challenging in the factory. In the assembly challenge competition in World Robot Summit 2018[2], assembly of a polyurethane belt onto pulleys (see Figure 5.1) was one of the most challenging tasks [12]. While there have been attempts to solve manipulation problems involving deformable objects [86, 52, 88, 69, 64, 25], there is no general approach to it.

In this chapter, we consider the problem of wrapping a belt around a two pulleys system, considering as use case the challenge introduced in the World Robot Summit 2018. Working with a deformable object like the belt presents several challenges. These include: (i) infinite degrees of freedom for the belt; (ii) contact rich manipulation; and (iii) long-horizon planning problem.

Optimization-based planning and control may be applied to various problems in robotic manipulation. Given a controlled dynamical system, $\dot{x} = f(x, u)$, trajectory optimization aims to design a finite-time input trajectory, $u(t), \forall t \in [0, T]$, which minimizes some cost functions over the resulting input and state trajectories [50, 32, 85].

In the belt drive unit assembly, variations in the belt tensions and contact forces between the belt and the pulleys result in a hybrid dynamical system. Elastic force can exist or not, depending on whether the belt is slack or stretched. Contact forces can exist or not, depending on whether the belt contacts the pulley or not. Both elastic and contact forces

[1] The preliminary version of the results of this Chapter is obtained during the author's internship at Mitsubishi Electric Research Laboratories.

[2] `https://worldrobotsummit.org/en/about/`

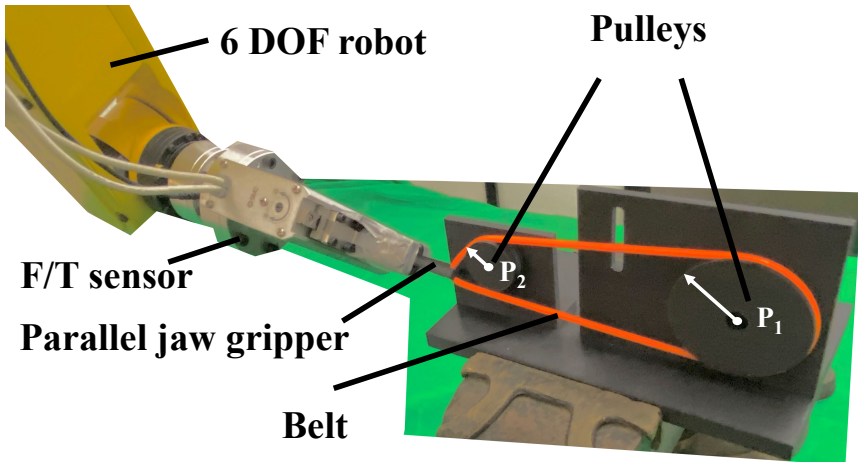


Figure 4.1: Belt drive unit assembly task. The robot grips a polyurethane belt and assembles it on two pulleys, P_1 and P_2.

might greatly impact the system dynamics. Planning for such a hybrid system usually requires planning for each dynamic system separately. There are many existing works on trajectory planning for hybrid systems [75, 14, 29]. But the drawback is that those methods require a task-specific mode schedule, which may bring extensive efforts in modeling and parameter tuning.

Inspired by the work on trajectory optimization of rigid bodies through contact [50, 85, 84, 83], we model the physics of contacts and the elastic properties through complementarity constraints. The elastic belt is modeled through a 3D keypoint representation. The hybrid behavior of the keypoints is captured by the complementarity constraints. We formulate the trajectory optimization problem as a Mathematical Program with Complementarity Constraints (MPCC) [39]. We successfully solve the MPCC to compute feasible and efficient trajectories to assemble the belt drive unit. Finally, we implement the solution into the real system with a controller to track the optimized trajectory.

The main contributions presented in this chapter are:

1. Trajectory optimization formulation for deformable objects manipulation with complementarity constraints. This provides a general-purpose, mathematical framework to tackle these problems.

2. Introduction of 3D keypoint representation for deformable objects.

3. Validation of the proposed approach through simulation as well as real experiments.

4.2 Related Work

Deformable linear (one-dimensional) object manipulation has been studied for decades. A randomized algorithm was proposed to plan a collision-free path for elastic objects [35]. Minimal-energy curves were applied to plan paths for deformable linear objects in stable configurations [42]. In [47], a local deformation model approximation method was proposed to control the soft objects to desired shapes. The authors of [87, 23] extended the local deformation model to the manipulation of cables. A deep neural network was trained to manipulate a rope to target shape based on a sequence of images [45]. However, those works do not consider the interaction between the deformable cables and the environment. In [86], the authors proposed a strategy to assemble a flexible beam into a rigid hole. An optimization-based trajectory planning was utilized to assembly ring-shaped elastic objects in [52], but the authors only validated their method in simulation. In [88], the authors took the advantage of environmental contacts to shape deformable linear objects by a vision-based contact detector. The authors of [69] considered a scenario to assemble the roller chain to sprockets. Their strategy successfully assemble the chain but lacks in generalization because each step is engineered for the specific system. To advance the research on robotic manipulation, the World Robot Summit 2018 proposed a competition on assembly challenges [12]. The challenge highlighted the complexity of solving manipulation tasks in a general manner, which still remains an open problem.

Optimization-based methods have been successfully implemented in many trajectory planning scenarios [53, 30, 58]. [50] proposed a trajectory optimization method for rigid bodies contacting the environment. They formulated an MPCC to eliminate the prior mode ordering in discontinuous dynamics due to inelastic impacts and Coulomb friction. The MPCC framework was extended to a quadrotor with a cable-suspended payload system in [15]. The complementarity constraint was utilized to model the limitation of a non-stretchable cable length. Inspired by [85, 84, 83, 50, 15], we introduce complementarity constraints to the belt drive unit assembly task to avoid the hybrid modes selection due to elastic force in the belt and contact force between the belt and the pulleys. We extend the keypoints representation introduced for rigid objects, [40], to model elastic objects like the belt and to formulate an MPCC to perform the assembly.

4.3 Problem Formulation

The belt drive unit consists of two pulleys attached to a base and of a deformable and stretchable belt as shown in Figure 5.1. The belt is assumed to be composed of a homogeneous isotropic linear elastic material which is a common assumption in mechanics. The pulleys have known geometries and can rotate freely around the shafts axis. The base of the belt drive unit is fixed to the workbench in a known pose. We assume that at the initial configuration, called ρ_0, the belt is grasped and lifted by a gripper held by a robotic manipulator, and the belt is freely hanging under the effect of gravity, see Figure 4.2. The task objective

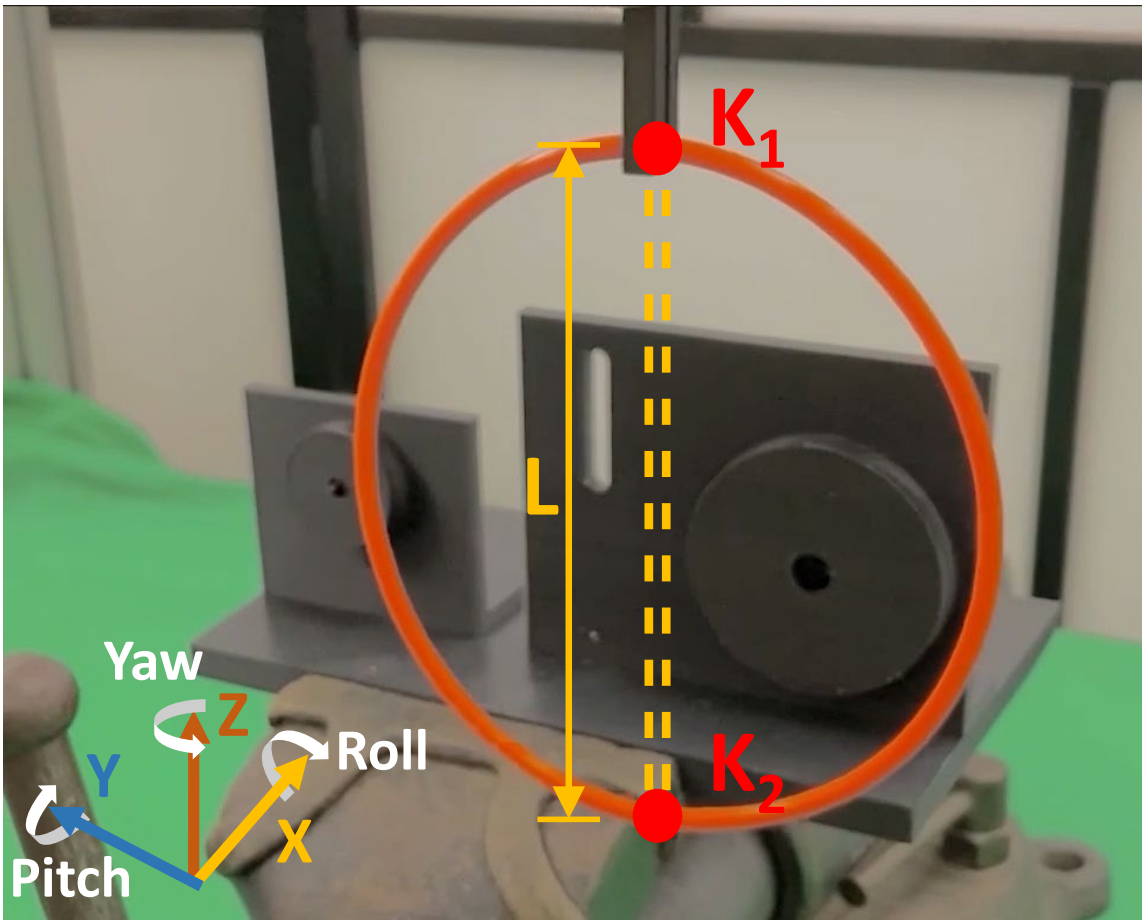


Figure 4.2: Initial belt configuration, ρ_0, with keypoints representation. The red dots represent the "grasped" keypoint K_1 and "opposite" keypoint K_2. The yellow dashed line shows the virtual cable, $\mathcal{C}$, of length L.

is to wrap the belt around the two pulleys as shown in Figure 5.1.

Inspired by recent work [40], we introduce a 3D keypoints representation for deformable objects. This representation consists of identifying points in the object that are representative of the whole object. With the proposed representation, the problem is mathematically tractable with a finite, low dimensional state space and interpretable constraints and cost function. In particular, we select two 3D keypoints for the belt as shown in Figure 4.2. The "grasped" keypoint, K_1, corresponds to the point-mass on the belt grasped by the robot gripper, and the "opposite" keypoint, K_2, which is the point on the belt that is the furthest away from K_1 when the belt is in configuration ρ_0. In the proposed keypoint representation, configuration ρ_0 can be represented by a virtual elastic cable, $\mathcal{C}$, that connects K_1 and K_2. The generalized coordinates of the system can now be described as

$q = [K_1^x, K_1^y, K_1^z, K_2^x, K_2^y, K_2^z, K_1^{roll}, K_1^{pitch}, K_1^{yaw}]^\top \in R^9$, where $K_1^x, K_1^y, K_1^z, K_2^x, K_2^y, K_2^z$ are the Cartesian coordinates of two keypoints and $K_1^{roll}, K_1^{pitch}, K_1^{yaw}$ are the orientation of K_1 with reference frame shown in Figure 4.2. We utilize the orientation of K_1 to express the rotation and the twist of the belt. The action space $u = [F_x, F_y, F_z, M_x, M_y, M_z]^\top \in R^6$ is the vector of forces and torques that are applied to K_1 through the gripper. This makes the belt drive unit an underactuated system as we cannot directly control K_2. Finally, we assume that in configuration ρ_0 the ellipsoidal shape of the belt is large enough to go around the first pulley P_1.

Subtasks Decomposition

Belt drive unit assembly is a complex task that requires a long-horizon planner. As often proposed in the literature, long-horizon planning tasks are decomposed into subtasks to reduce complexity. The belt drive unit assembly can have highly engineered solutions with a dense sequence of subtasks and simple planners whose subgoals are trivial to reach. However, this kind of approach requires extensive effort in parameter tuning and engineering work and lacks generality, since the goals of the subtasks need to be redefined as the scenario changes. We partially address this problem by reducing the number of subtasks to two. Following a logic similar to a human's approach, the first subtask, S_1, corresponds to wrap the belt around the first pulley, and the second subtask, S_2, corresponds to wrap the second pulley keeping the belt taut to maintain the wrap around the first pulley, see Figure 4.3. In a qualitative description, in S_1, the belt has to avoid the outer surface of the first pulley P_1 and K_2 has to get into the groove creating a contact force, while K_1 is stretched until the belt is taut. In S_2, the belt is assembled on the second pulley P_2 by rotation around the first pulley. During the rotation, the belt should remain taut in order to remain in the groove of the first pulley. Finally, the belt has to hook the internal groove of P_2 and K_1 has to reach the bottom of the second pulley to accomplish the task.

Given the proposed 3D keypoint representation of the belt drive unit, we can formulate a trajectory optimization problem, that uses complementarity constraints to model the contacts and the deformation of the belt, to solve each of the two subtasks. The two optimized trajectories are then executed sequentially in order to accomplish the task, the final condition of S_1 corresponds to the initial condition of S_2.

4.4 Trajectory Optimization for the Belt Drive Unit Assembly

We approach the belt drive unit assembly as a trajectory optimization problem formulated as an MPCC. The complexity of this problem is given by the presence of hybrid nonlinear dynamics due to contacts that may happen between the pulleys and the belt, the elastic properties of the belt, the obstacle avoidance constraints, and the long planning horizon. A trajectory optimization problem is solved for each of the two subtasks described in Sec. 4.3

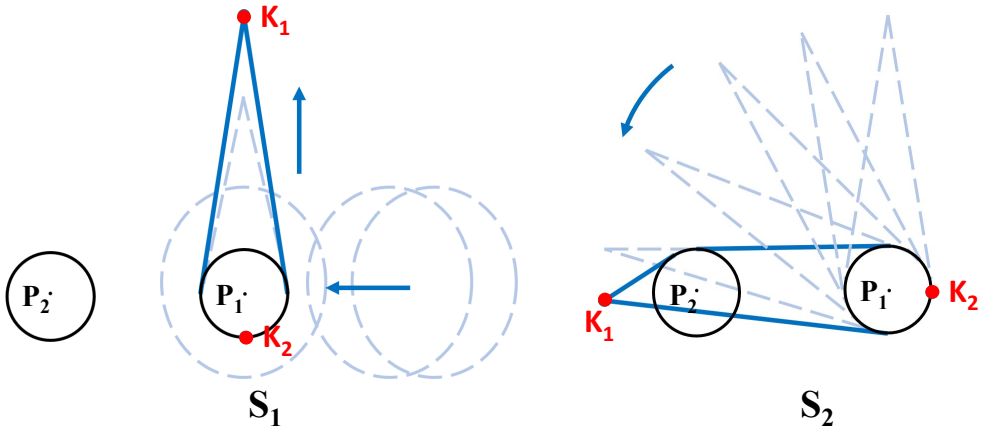


Figure 4.3: Two subtasks decomposition. P_1 and P_2 are two pulleys. The blue lines represent the belt gripped at keypoint K_1 by a robot. S_1: The belt wraps the first pulley P_1 and it stretched. S_2: The belt rotates around the first pulley and is assembled onto the second pulley P_2.

of the form

$$\min_{q,\dot{q},u,\lambda} \quad L(q,\dot{q},u,\lambda) \tag{4.1a}$$

$$\text{s.t.} \quad H(q)\ddot{q} + C(q,\dot{q}) + G(q) = B(q)u + \lambda \tag{4.1b}$$

$$g(q,\dot{q},u,\lambda) \leq 0 \tag{4.1c}$$

$$q \leq q \leq q,\ \dot{q} \leq \dot{q} \leq \dot{q},\ u \leq u \leq u,\ \lambda \leq \lambda \leq \lambda \tag{4.1d}$$

where $L(q,\dot{q},u,\lambda)$ is the cost function, $q \in R^9$ are the generalized coordinates described in Sec. 4.3, $\dot{q}$ and $\ddot{q}$ are its first and second order time derivatives, $u \in R^6$ is the control input and λ are the external forces acting on the belt. Eq. (4.1b) is the forward dynamics, where $H(q), C(q,\dot{q}), G(q)$ are the inertial matrix, the Coriolis terms, and the gravitational forces, respectively. $B(q)$ is input mapping. The general nonlinear constraints (4.1c) include the complementarity constraints and collision avoidance. Eq. (4.1d) represents the lower and upper bounds of the optimization variables.

We solve (4.1) as a nonlinear program and use a direct approach which in general has better numerical properties than shooting methods, and we can exploit the sparsity structure of the problem. We directly optimize the feasible general coordinates and its first-time derivative, the control inputs, and the external forces. The discretization of the forward dynamics is obtained by the trapezoidal rule. The formulation of the contact and elastic forces as complementarity constraints fits naturally well in this formulation. In practice, for numerical advantages, we use a relaxed version of the complementarity constraints as described in [51].

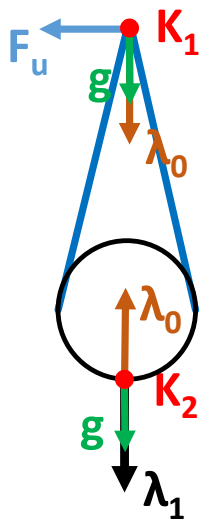


Figure 4.4: Force analysis at the two keypoints. F_u is the control input force. λ_0 is the elastic force. λ_1 is the contact normal force. g is the gravity force.

In the following, the dynamical constraints and the cost function for MPCC (4.1) are described for the two subtasks.

Dynamics Constraints

The system is composed of the two keypoints, K_1 and K_2, and the virtual elastic cable, $\mathcal{C}$. It is modeled similarly to a mechanical mass-spring-damper second-order system, with an actuator acting on K_1 and subject to the gravity and the external forces given by the elastic force λ_0 and the normal force λ_1 experienced during contacts. Figure 4.4 shows a schematic example of the forces that act on the system at the end of subtask S_1.

The system dynamics are defined as

$$\dot{x} = Ax + Bu + G + f(x, \lambda) \tag{4.2}$$

where $x = [q^\top, \dot{q}^\top]^\top \in R^{18}$ is the system state and $\lambda = [\lambda_0^\top, \lambda_1^\top]^\top$ is the vector of the external forces. The state transition matrix

$$A = \begin{bmatrix} 0_{9\times9} & & I_{9\times9} & \\ 0_{3\times9} & -\frac{k_d}{m_1} I_{3\times3} & \frac{k_d}{m_1} I_{3\times3} & 0_{3\times3} \\ 0_{3\times9} & \frac{k_d}{m_2} I_{3\times3} & -\frac{k_d}{m_2} I_{3\times3} & 0_{3\times3} \\ 0_{3\times9} & & 0_{3\times9} & \end{bmatrix} \in R^{18\times18}$$

represents the effect of the mass-spring-damper system with k_d the damping coefficient and

m_1, m_2 are the masses of the keypoints K_1 and K_2, respectively. The input matrix

$$B = \begin{bmatrix} 0_{9\times 6} \\ \frac{I_{3\times 3}}{m_1} \quad 0_{3\times 3} \\ 0_{3\times 6} \\ 0_{3\times 3} \quad \frac{I_{3\times 3}}{M_1} \end{bmatrix} \in R^{18\times 6}$$

maps the 6-dimensional end-effector force/torque input u to the linear and angular acceleration of the "grasped" keypoint K_1. M_1 is the moment of inertia of K_1. The gravitational acceleration is applied to the two keypoints through the vector $G = \begin{bmatrix} 0_{1\times 11}, & -g, & 0_{1\times 2}, & -g, & 0_{1\times 3} \end{bmatrix}^\top \in R^{18\times 1}$.

The contribution of the external forces is now given by the sum of the elastic and normal force $f(x, \lambda) = \lambda_0 + \lambda_1$. The elastic force is defined as

$$\lambda_0 = \begin{bmatrix} 0_{3\times 9}, & -\frac{I_{3\times 3}}{m_1}, & \frac{I_{3\times 3}}{m_2}, & 0_{3\times 3} \end{bmatrix}^\top \Pi_{K_1,p}\, \bar{\lambda}_0 \in R^{18\times 1}$$

where $\bar{\lambda}_0 \in R$ is the magnitude of the elastic force and is the variable optimized, $\Pi_{K_1,p} = \frac{[(K_1^x - p^x),(K_1^y - p^y),(K_1^z - p^z)]^\top}{||K_1 - p||}$ is the projection operator of the elastic force into the 3 axis. The point p is K_2 in S_1 and O_1 in S_2 for simplicity of computation. O_1 is the position of the first pulley's center. The normal force due to the contacts between the pulley and the keypoint K_2 is defined as

$$\lambda_1 = \begin{bmatrix} 0_{3\times 12}, & -\frac{I_{3\times 3}}{m_2}, & 0_{3\times 3} \end{bmatrix}^\top \Pi_{O_1,K_2} \bar{\lambda}_1 \in R^{18\times 1}$$

where $\bar{\lambda}_1 \in R$ is the magnitude of the normal force and is the variable optimized and $\Pi_{O_1,K_2} = \frac{[(o_1^x - K_2^x),(o_1^y - K_2^y),(o_1^z - K_2^z)]^\top}{||o_1 - K_2||}$ is the projection operator of the normal force into the 3-axis.

Complementarity Constraints

In order to model the hybrid dynamics due to elastic force and contact force, we use complementarity constraints

$$0 \leq g(\cdot) \quad \perp \quad h(\cdot) \geq 0 \tag{4.3}$$

Complementary constraints are a way to model constraints that are combinatorial in nature and impose the positivity and orthogonality of the variables.

Elastic force constraint. The first complementarity constraint is formulated as

$$\lambda_2 = \frac{\bar{\lambda}_0}{k_p} + L - l(x) \geq 0 \tag{4.4a}$$

$$\bar{\lambda}_0 \geq 0 \tag{4.4b}$$

$$\bar{\lambda}_0 \lambda_2 = 0 \tag{4.4c}$$

where L and k_p are respectively the length at configuration ρ_0 and the stiffness coefficient of the virtual elastic cable, $\mathcal{C}$. The length of $\mathcal{C}$ at each temporal instant is $l(x) = ||K_1 - K_2||$ in S_1, and $l(x) = ||K_1 - O_1|| + r_1$ in S_2, where r_1 denotes the radius of P_1. The pulley center O_1 is chosen because it is a fixed known point while the pulley is rotating. From eq. (4.4a) the elasticity of the belt is defined as proportional to the length $L - l(x)$ and depends on the stiffness coefficient k_p. λ_2 is an algebraic variable. If the cable is stretched, then $L < l(x)$, $\bar{\lambda}_0 > 0$, and $\lambda_2 = 0$. If the cable is slack, then $L > l(x)$, $\bar{\lambda}_0 = 0$, and $\lambda_2 > 0$.

Contact force constraint. The second complementarity constraint is formulated as

$$\lambda_3 = \sqrt{||K_2 - O_e||^2 + \epsilon^2} \geq \epsilon \tag{4.5a}$$

$$\bar{\lambda}_1 \geq 0 \tag{4.5b}$$

$$\bar{\lambda}_1 \lambda_3 = 0 \tag{4.5c}$$

where O_e is the contact point on the edge of P_1. ϵ denotes a small number to relax the complementarity constraint [51]. λ_3 is the algebraic variable describes whether the belt contacts the pulley. If the belt contacts the pulley, then $\lambda_3 = \epsilon$, and contact force $\bar{\lambda}_1 \geq 0$. If there is no contact, then $\lambda_3 > \epsilon$, and contact force $\bar{\lambda}_1 = 0$.

Obstacle avoidance

This constraint imposes that the keypoints cannot penetrate into the pulleys. Each pulley is approximated with an ellipsoid, since there is a known analytical expression of the distance function between a point and an ellipsoid. The obstacle avoidance constraints between a keypoint K_i and a pulley P_j can be denoted as $distance(K_i, P_j) = \sqrt{(K_i - O_j)^\top S (K_i - O_j)} - 1 \geq 0$, where $S = diag\{1/a^2, 1/b^2, 1/c^2\}$ is a diagonal matrix, a, b, c are half the length of the principal axes. O_j denotes the center of pulley P_j.

Physics Limitation

The belt might break if stretched over a certain limit, this condition is approximated by constraining the length of the virtual cable $\mathcal{C}$, $l(x) \leq L_{max}$. Moreover, L_{max} is assumed large enough for the loop to go around two pulleys.

Cost Function

We use a common quadratic cost function that penalizes the difference to the goal state x^{goal} and the control input $u(k)$:

$$J(x, u, \lambda) = \sum_{k=0}^{N} (x(k) - x^{goal})^\top Q (x(k) - x^{goal}) + u(k)^T R u(k) + w(\bar{\lambda}_0(k) - \bar{\lambda}_0^{desired})^2 \tag{4.6}$$

where the weights Q and R are diagonal matrices and w is a scalar. Moreover, the term $w(\bar{\lambda}_0(k) - \bar{\lambda}_0^{desired})^2$ adds a soft constraint in the elastic force. A positive $\bar{\lambda}_0^{desired}$ encourages

a solution with the belt in tension. This constraint is used in subtask S_2 to maintain the belt taut. Instead, in S_1 we set $w = 0$.

4.5 Experimental Results

In this section, we present the results of the proposed method both in simulation using the physic engine MuJoCo [73] and in a real system with a 6-DoF manipulator. We use the Ipopt [76] solver in a python wrapper.

Simulations

Simulation Setup

The belt drive unit is represented in a simulated environment in MuJoCo as shown in the top left corner of Fig. 4.5. The environment includes two pulleys and one belt. The radius of the pulleys are of $30[mm]$ for P_1 and $15[mm]$ for P_2. The belt is composed of 41 linked objects called capsules in MuJoCo. Any two adjacent capsules are connected by two hinge joints and one prismatic joint. The physical properties of the simulated belt are tuned to resemble the belt of the real belt drive unit. The belt is held by a parallel gripper attached to a 6 DOF Fanuc LR-Mate 200iD. The purpose of the manipulator is to actuate the end-effector in order to track the optimal trajectory, but in simulation could be removed.

Trajectory Optimization in Different Scenarios

In order to verify the generality of the approach to different known geometries, we consider 4 different scenarios, where the position of P_1 is fixed and the position of the smaller pulley P_2 varies, see Figure 4.5 and Table 5.1. Each pulley is modeled as three adjacent cylinders, and the two outer cylinders have larger radius than the inner one. The belts' lengths, P_{belt}, are chosen in each scenario based on the distance between two pulleys. The mass of the keypoints is $m_1 = m_2 = 0.042[kg]$. The moment of inertia of K_1 is $M_1 = 10^{-7}[kgm^2]$. The belt's stiffness and damping coefficient are $k_p = 63.34[N/m]$ and $k_d = 4.65[Ns/m]$, respectively.

In the trajectory optimization formulation described in Section 4.4, the goal for subtask S_1 is set vertically above the pulley P_1 for keypoint K_1, and right under the pulley P_1 for keypoint K_2, respectively, e.g., $q_1^{goal} = [0.10, 0.55, 0.53, 0.10, 0.23, 0.34, 0, 0, 0]^T$ and both with zero velocity. In this substask there is a change of mode in the dynamics from no contact to contact between the belt and the environment and the deformation of the belt for reaching the desired target.

In the second subtask S_2, the goal, q_2^{goal}, is set only for K_1 in both the Cartesian coordinates and angular orientation, according to the position of the pulley, P_2. A qualitative representation of the goal position is shown in Figure 4.5 for each of the scenarios. The desired $[K_1^{roll}, K_1^{pitch}, K_1^{yaw}]^\top$ is $[-\pi/2, 0, \pi/2]^\top$. The twist of the virtual cable $\mathcal{C}$ approximates

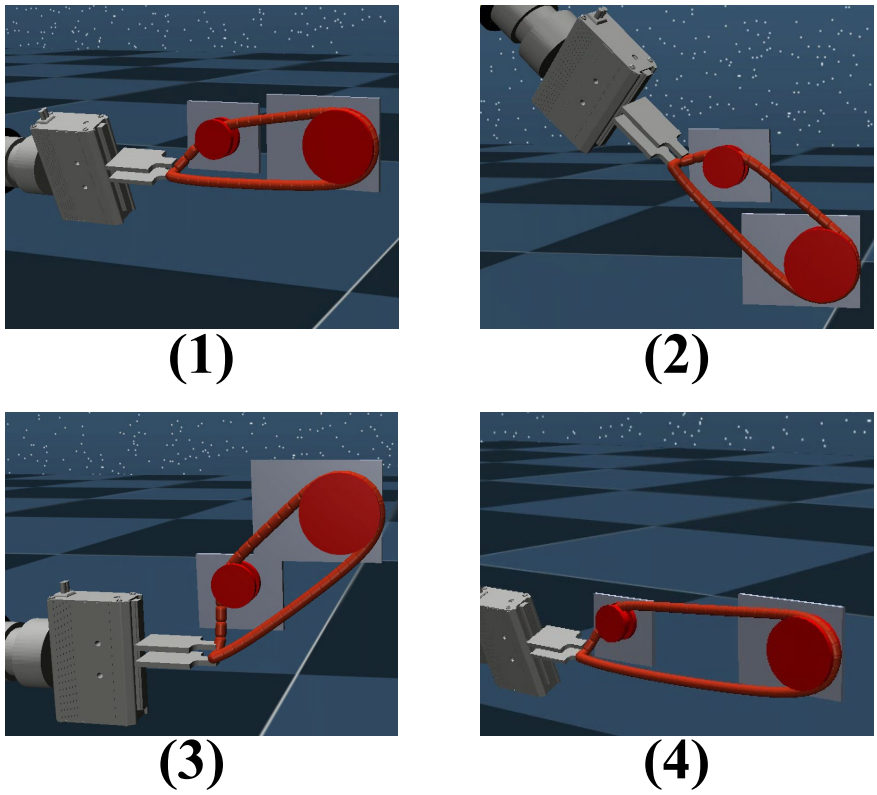


Figure 4.5: Snapshots of 4 different simulation scenarios at the goal position of subtask S_2. The relative positions of the two pulleys vary in each scenario.

Table 4.1: Simulation results in 4 scenarios. In each scenario the position of the pulley center O_2 varies.

	Scenario 1	Scenario 2	Scenario 3	Scenario 4
P_{belt}	$0.4m$	$0.4m$	$0.4m$	$0.4m$
O_1	[0.100, 0.550, 0.340]	[0.100, 0.550, 0.340]	[0.100, 0.550, 0.340]	[0.100, 0.550, 0.340]
O_2	[0.100, 0.680, 0.340]	[0.100, 0.680, 0.340]	[0.100, 0.680, 0.340]	[0.100, 0.680, 0.340]
Feasible trajectory	10/10	10/10	10/10	10/10
Successful assembly	10/10	8/10	7/10	9/10

the twist of the belt which leads to the assemble onto the groove of the pulley P_2. Based on q_2^{goal} and k_p it is possible to compute the target elastic force as $\bar{\lambda}_0^{desired}$, which encourages the belt to be stretched during rotation.

We perform 10 experiments for each scenario. In each experiment, the goal positions of the "grasped" keypoint K_1 are sampled from the normal distributions $\mathcal{N}(\mu_1, \Sigma)$ in S_1 and $\mathcal{N}(\mu_2, \Sigma)$ in S_2. Where, $\mu_1, \mu_2 \in R^3$ are the components $[K_1^x, K_1^y, K_1^z]^\top$ in a pre-selected successful q_1^{goal} and q_2^{goal} and $\Sigma = diag\{0.005, 0.005, 0.005\}$. The lower and upper constraints

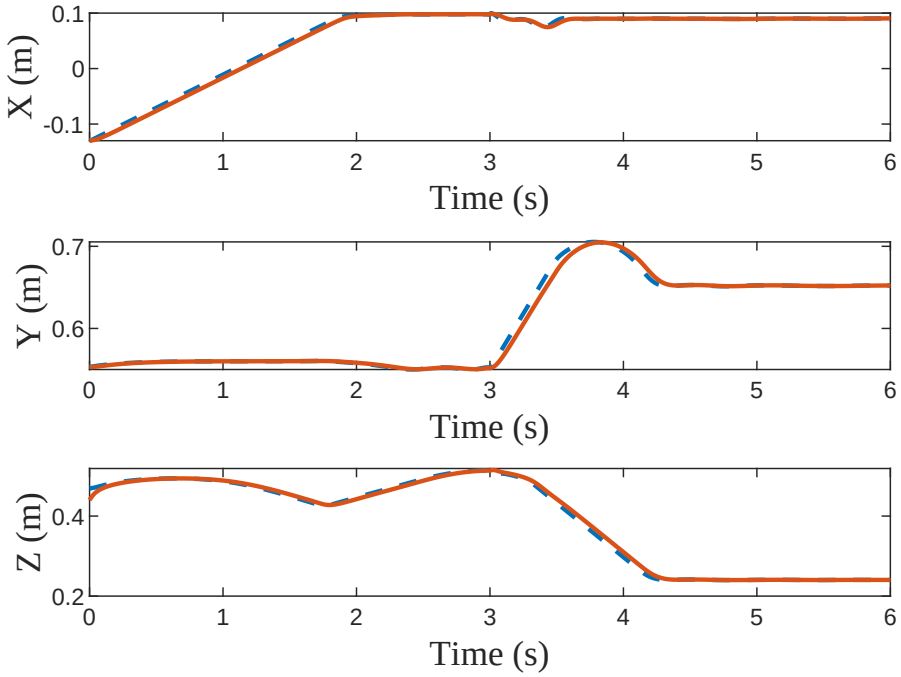


Figure 4.6: Trajectory of keypoint K_1 in a successful assembly for scenario Figure 4.5(3). The dashed line is the reference trajectory obtained from MPCC. The solid line is the measured trajectory.

of position, velocity, tilt angle, and force are $\pm 1m$, $\pm 0.5m/s$, $\pm\pi$, and $\pm 50N$, respectively.

Results

The simulation results are shown in Table 5.1. We initialize the trajectory with all states $x(k) = x(0)$, where $k = 0, 1, .., N$. The solver finds a feasible trajectory for both subtasks given any sampled goals. The optimal trajectory obtained for K_1 is then tracked by the end-effector, and the assembly is completed successfully in 34/40 experiments. The failure cases happen when the goal is sampled away from q_1^{goal} or q_2^{goal} because the belt fails to wrap around the pulley. The purpose of these experiments is to show that the engineering effort in finding the goal position for the subtask is reduced as it is not required to provide one specific point. But also the trade-off of having only two keypoints, more keypoints would make a more accurate model but also a more complex optimization problem. We use an Intel 12 Cores i7-9850H CPU @ 2.60GHz. The average computational time for one trajectory with 600 time steps is $36.138 \pm 5.747[s]$. The computational time highly depends on the number of time steps selected. Figure 4.6 shows one full successful assembly trajectory for scenario Figure 4.5(3).

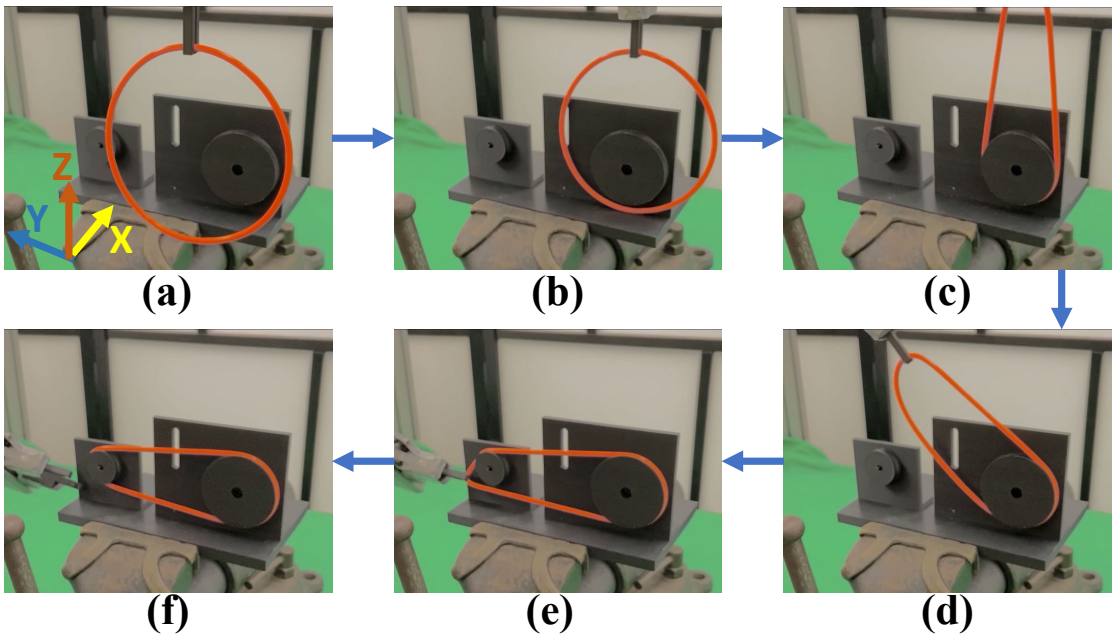


Figure 4.7: Snapshots of the experiment.

Real-World Experiments

Experimental Setup

As shown in Figure 5.1, the experiment environment includes a 6 DOF FANUC LR-Mate 200iD, an ATI Mini45 F/T sensor, and a 3D printed belt drive unit of the same dimensions in the assembly challenge [12] fixed on a vise. The belt is the same as in the challenge with length $0.40[m]$ and is gripped by a parallel jaw gripper. We assume no slip between the belt and the robot gripper. The pose of the pulleys is known exactly.

Results

Figure 4.7 provides the snapshots of the main phases during the execution of a successful experiment. Figure 4.8 shows the trajectory of the gripper tip that corresponds to K_1 and the measured forces at the robot's wrist along the trajectory. In the beginning, (Figure 4.7a), the belt approaches the pulley and position X increases and the forces are zero. The position Z goes down at $5.57[s]$ to avoid the outer cylinder of the first pulley. At $6.29[s]$, position X stops increasing because the pulley is reached (Figure 4.7b). Then the Z position increases as the belt is lifted and contacts the pulley at $7.82[s]$ with a corresponding increase in force along the negative direction in Z. At $10.50[s]$, the system accomplishes S_1 (Figure 4.7c). After that, the belt is rotated around O_1 (the Z position decreases, and Y position increases) while being stretched (Figure 4.7d). In this phase, the measured net force is closed to the desired

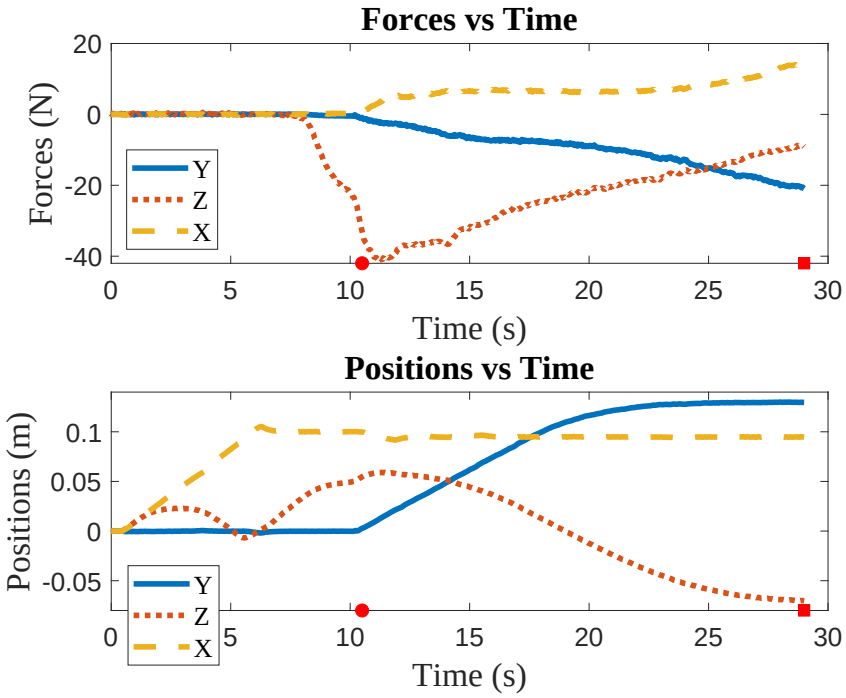


Figure 4.8: Forces and positions of the end-effector in a successful experiment. The red circle and square represent the end of subtask S_1, S_2, respectively.

elastic force $\bar{\lambda}_0^{desired}$. The target orientations are $[K_1^{roll}, K_1^{pitch}, K_1^{yaw}]^\top = [-\pi/2, 0, \pi/4]^\top$. The belt is twisted so that it hooks the second pulley without jamming. Finally, the goal of subtask S_2 is reached at 29.00$[s]$ (Figure 4.7e) and the gripper releases the belt (Figure 4.7f).

The experiment has been repeated multiple times but given the robot's accuracy the results were similar to each other.

4.6 Chatper Summary

In this chapter, we propose a trajectory optimization formulation to assemble the belt drive unit. We propose a 3D keypoints representation to model the elastic belt, which simplifies the complexity of the trajectory optimization problem. The problem is formulated as an MPCC with complementarity constraints to model the hybrid dynamics due to contact and elastic forces. Simulations results show that the proposed approach can find feasible trajectories for the belt drive unit assembly with known but variable geometry. To the best of our knowledge, this is the first work that formalizes the trajectory optimization problem for the belt drive unit assembly, and the solution works in the real system. Several future works are possible. The current method is based on the execution of an open-loop trajectory

which could fail under uncertainties in the position of the pulleys or of the belt. Adding a feedback controller is fundamental for a more robust and reliable solution. Moreover, in order to improve the generality of the problem, we are interested in an autonomous selection of the 3D keypoints for a given task. Our formulation of a trajectory optimization problem for deformable objects using complementarity constraints is not limited to belt drive unit assembly. The proposed method might be applied to a wider range of tasks such as cable routing and wire harness.

Chapter 5

Robotic Cable Routing with Spatial Representation

5.1 Introduction [1]

In this chapter, we consider a cable routing task (Fig. 5.1), as described in Chapter 1, where cables need to follow a designated path constrained by a set of fixtures on the table. Given randomly placed fixtures and a goal cable configuration, robots need to manipulate the cable from an initial configuration to the goal configuration. Sense-plan-act is an effective framework tackling the routing problem, which consists of 1) visual perception, 2) intermediate configuration planning, and 3) low-level manipulation planning and execution. However, there are multiple practical challenges preventing this method from being widely adopted. 1) For visual perception, a chain of connected nodes is commonly used to represent the cable state, where the node positions are estimated from the point cloud of the cable. The node estimation is significantly affected by the quality of the segmented cable point cloud and lacks robustness with conventionally used color filters [59, 67, 63, 8] requiring manual tuning and susceptible to environment lighting changes. 2) For configuration planning, human demonstrated sequences composed of intermediate cable states are often needed, and the robots can finish the task following the predefined sequence [64]. Demonstrating the full routing sequence requires extensive human efforts and does not generalize when the cable configuration changes. 3) During manipulation, the cable can easily deform to unexpected shapes. An over-stretched cable may break the cable or fixtures, while a slack cable may fail to reach the desired configuration due to the under-actuated dynamics.

To address the challenges in planning and manipulation, we propose a simple yet effective representation, called spatial representation, to model the spatial relations between the cable and environment objects (namely fixtures). The core insight of this representation comes from an empirical observation: the spatial relation between the cable and fixtures, instead

[1] The preliminary version of the results of this Chapter is obtained during the author's internship at Intrinsic.

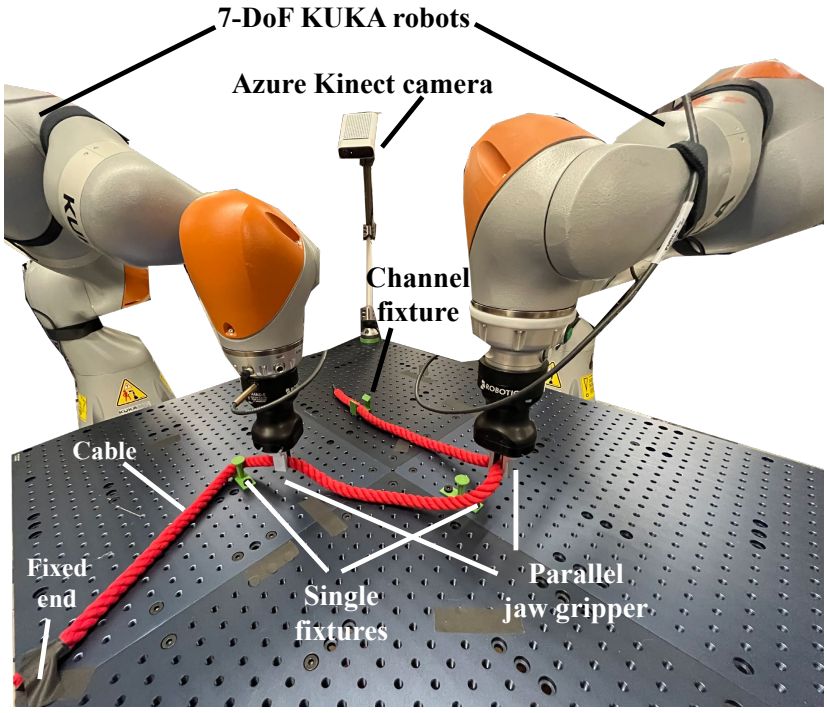


Figure 5.1: Cable routing task setup. Two robot manipulators attempt to manipulate the cable to the goal configuration, which is constrained by several fixtures, through a sequence of picking, placing, and holding actions.

of their accurate positions, contains the full information relevant to the routing task. With the proposed spatial representation, configuration planning can be efficiently achieved by searching a path from the initial to the goal state without the need for human demonstration. In addition, this representation enables efficient data collection and model learning for low-level manipulation. We design three low-level manipulation primitives, `stretch`, `cross`, and `insert`, to manipulate the cable from one spatial representation to another. A manipulation primitive takes the cable state, fixture positions, and the target spatial representation as input, and outputs picking and placing targets for robots. The `stretch` primitive stretches the cable to another configuration without changing the spatial representation. The `cross` and `insert` primitives change the cable configuration between two spatial representations. As will be shown in Sec. 5.5, cable routing with the learned primitives outperforms rule-based heuristics and achieves reliable performance with diverse cable and fixture settings.

Besides, to address the robustness and generalization challenge in visual perception, we propose a cable state estimator composed of a neural network and non-rigid registration, which is robust to different cable colors and backgrounds. Particularly, the cable segmentation neural network is trained with collected real images and data augmentation without human annotation. Experiments show the cable state estimation is robust to different cable

colors and backgrounds as long as the cable and the background have contrasting appearances.

The main contributions presented in this chapter are summarized as below:

- Spatial representation is proposed to model the topological relationship between the cable and fixtures. Based on this spatial representation, a high-level path planner is designed to remove the need for human demonstrated intermediate states.
- Three manipulation primitives are designed and learned to manipulate the cable from one spatial representation to another.
- A robust cable state estimator is implemented leveraging neural network based segmentation and non-rigid registration, thus attaining an end-to-end cable routing framework.

5.2 Related Work

State Estimation of Deformable Linear Objects

Detecting cables in a cluttered environment is a challenging problem as cables, unlike rigid bodies, have infinite degrees of freedom. Some previous works apply end-to-end methods without extracting the structure of the cable, but the representation is not informative for downstream planning [45, 79, 19, 60]. Another commonly used approach is color filtering, which relies on manual tuning and is sensitive to environmental changes such as lighting conditions [59, 67, 63, 8]. Though deep neural networks (DNNs) have been widely used in rigid object detection [55, 16, 54, 17], few works deploy DNNs to cable detection. One major reason is that, unlike rigid bodies, labeling deformable cables is time consuming. [61, 82] learn the cable detection using synthetic data generated in simulation. Although their methods do not require a manually tuned color filter to perceive the cable, there is a large sim-to-real gap when deployed to real-world tasks. Different from above, we train a cable segmentation neural network with collected real images adopting various data augmentation techniques while still avoiding the need of human annotation.

Deformable Linear Object Manipulation

Deformable linear object manipulation has been studied for decades. A randomized algorithm is proposed to plan a collision-free path for elastic objects [35]. Minimal-energy curves are applied to plan paths for deformable linear objects in stable configurations [42]. A deformation model is utilized to manipulate the cable to desired shapes [87, 23]. There are also a lot of works on knot planning [81, 43, 64] and untangling knots [38, 62].

Different from the above works that do not consider the environment constraints in the workspace, cable routing requires the cable to establish contact with environment objects such as fixtures. [27] formulates a trajectory optimization problem for belt drive unit assembly. [77] generates a visual plan for cable routing tasks via a casual InfoGAN, but the

method cannot generate discontinuous actions such as the cable crossing a fixture. [88] analyzes the contact mobility for cable routing around fixtures, but their method requires a customized end-effector that allows the cable to slip. [3] builds a state machine for cable routing, which requires extensive human efforts to choose the state machine parameters and lacks generalization to unseen scenarios. Inspired by [81, 62], which combine topological planning and manipulation primitives to solve long-horizon cable knotting tasks, we introduce spatial representation for cable routing, bridging high-level configuration planning and low-level primitive execution.

5.3 Problem Statement

Inspired by the cable routing/USB insertion task in the Robotic Grasping and Manipulation Challenge in IROS 2020 [21], we formulate a simplified and reconfigurable cable routing task as follows (Fig. 5.1). A cable is placed on a worktable with several fixtures. One end of the cable is rigidly attached to the table. Given a goal configuration, where the cable makes contact with the single fixtures or goes through the channel fixtures in a particular order (as illustrated in Fig. 5.4), two robotic manipulators attempt to manipulate the cable to the goal configuration through a sequence of picking, placing, and holding actions. We make the following assumptions on the task: 1) the number of fixtures and their positions are known in advance; 2) the parallel-jaw grippers can firmly grasp the cable during execution; 3) the cable is within the RGB-D camera's field-of-view throughout the task; 4) the cable is distinguishable by color contrast to its background.

Inspired by [81, 62], we decompose the cable routing problem into three subtasks. 1) Perception. A proper state representation of the cable needs to be extracted from the raw RGB-D image. The state extraction has to be robust to the cable color, background, environment lighting, and the cable being partially occluded. 2) Planning. Given the current cable state, positions of the fixtures, and the goal configuration, we need to generate the next intermediate state. This is expected to be generated autonomously without human demonstration. 3) Manipulation. Given the current and next intermediate states, robot commands such as picking and placing poses need to be inferred. To prevent the cable from moving to undesired states, a holding action is introduced after each cable placement as described in Sec. 5.4.

5.4 Approach

We propose a cable routing framework with three modules: 1) cable state estimation, 2) planning with spatial representation, and 3) learning manipulation primitives (Fig. 5.2). 1) The cable state estimation module takes an RGB-D image as the input and outputs a chain of nodes representing the cable state. 2) The planning module generates a sequence of intermediate states based on the spatial relation between the estimated cable state and the

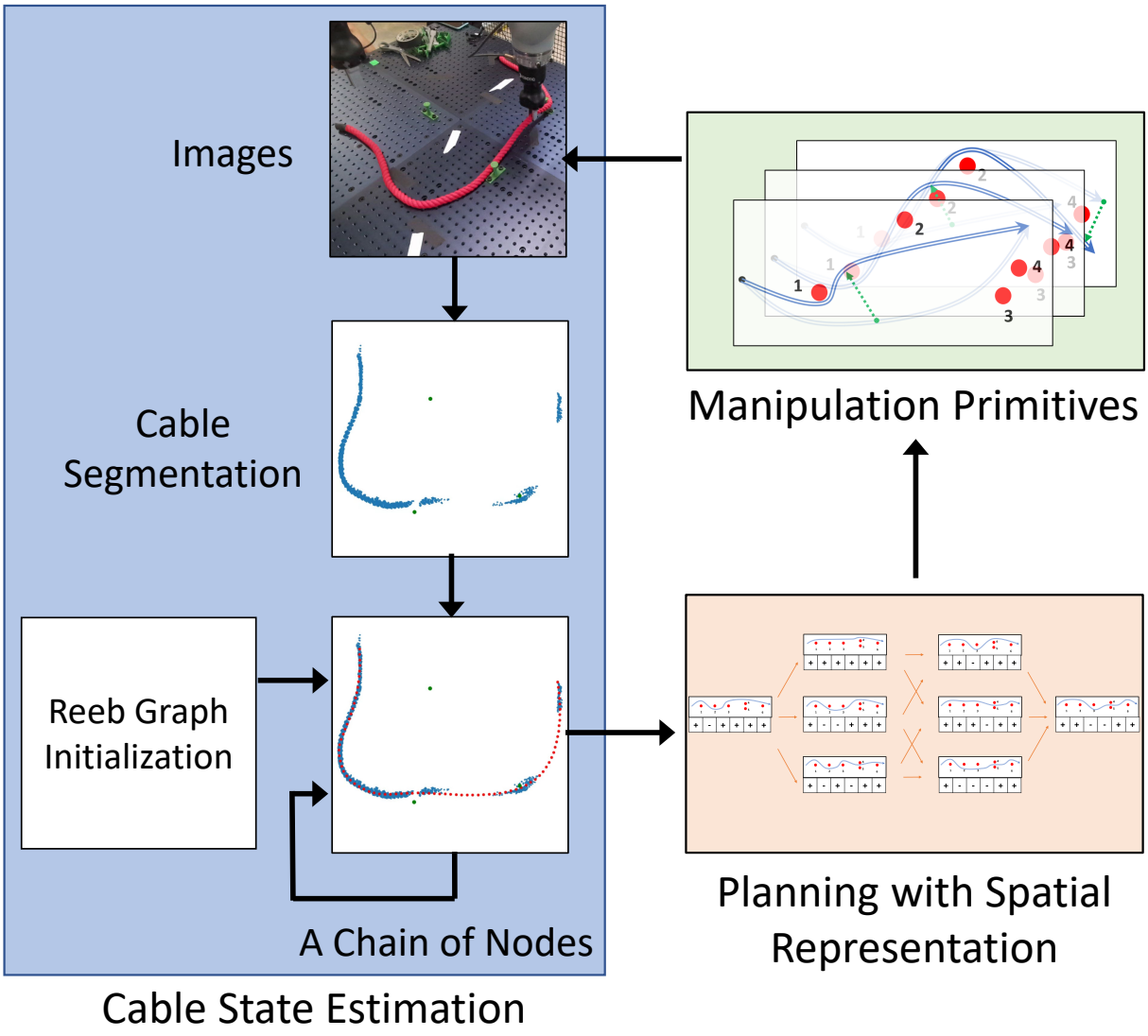


Figure 5.2: The proposed cable routing framework consists of three modules: 1) cable state estimation, 2) planning with spatial representation, and 3) learning manipulation primitives.

known fixtures. These intermediate states define a path for the cable to reach the goal from the initial configuration. 3) The manipulation primitives generate different robot actions that can manipulate the cable from the current states to the next intermediate states.

Cable State Estimation

Segmentation neural network

We propose a self-supervised data generation method that efficiently generates labeled images for cable mask segmentation without human annotation. The idea is to utilize a manually tuned color filter to automatically obtain the cable's segmentation mask from a pre-recorded video, then the images and the corresponding masks are augmented with different colors, backgrounds, occlusion, and noise.

We choose a cable with a contrasting color (e.g., a red cable) distinguished from the background and hand-design a color filter. We then record a video while an operator manipulates the cable to different configurations, changes the background, and adds occlusion. For each video frame, a cable segmentation mask can be generated automatically with the color filter. Afterwards we augment the data by randomly sampling different color distributions and impainting on the cable's mask. We apply standard augmentation techniques to increase robustness for the cable's mask and the remaining background, such as dropping out small patches and injecting pixel-wise noise [5]. Finally, the augmented data is utilized to train a U-Net [55].

Cable initialization with multi-resolution Reeb Graph

The segmentation neural network outputs the inferred point cloud of the cable. To facilitate the downstream planning and manipulation, a chain of nodes is preferred to represent the cable state, as shown in Fig. 5.2. We construct the cable nodes from the point cloud with multi-resolution Reeb Graph [59, 18]. Specifically, the segmented point clouds are grouped into different clusters, and the clusters are connected to form a smooth path while penalizing the path length and angle changes.

Non-rigid registration

Although cable initialization with Reeb Graph can handle minor occlusion, it would fail in practice because of heavy occlusion by the robot arms during task execution. To tackle this issue, we find the node positions at the current step $X^t = [x_1^t, x_2^t, ..., x_N^t] \in R^{N \times D}$ with non-rigid registration, given the node positions from the previous step X^{t-1} as a prior, where x_i^t is the position of *i-th* node at *t-th* frame, N is the number of nodes ($N = 50$ throughout our experiments), and D is the dimension of node positions ($D = 3$ in our case). We initialize the cable nodes X^0 from the result of Reeb Graph as described in Sec. 5.4. For each new frame, the cable nodes X^t can be obtained from the current point cloud $Y^t = [y_1^t, y_2^t, ..., y_M^t] \in R^{M \times D}$

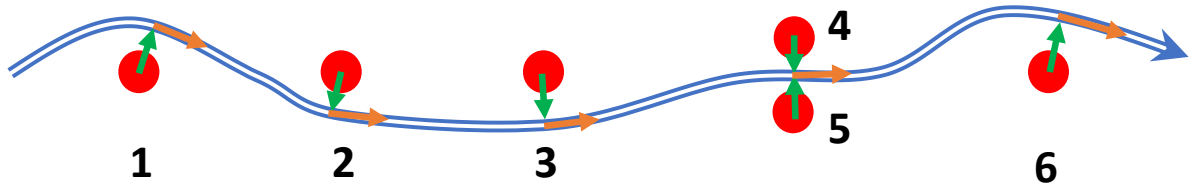


Figure 5.3: An example spatial representation. Red dots 1,2,3,6 are single fixtures. Red dots 4 and 5 form a channel fixture. The cable from head to tail is traced following the orange arrow. The green arrows are the projection of fixtures onto the cable. The spatial state for each fixture is determined by the sign of the cross product between the green and orange arrows. In this particular example, the spatial representation is (+,-,-,-,+,+). Intuitively, the spatial representation means that the six fixtures are on the (right, left, left, left, right, right) side of the cable.

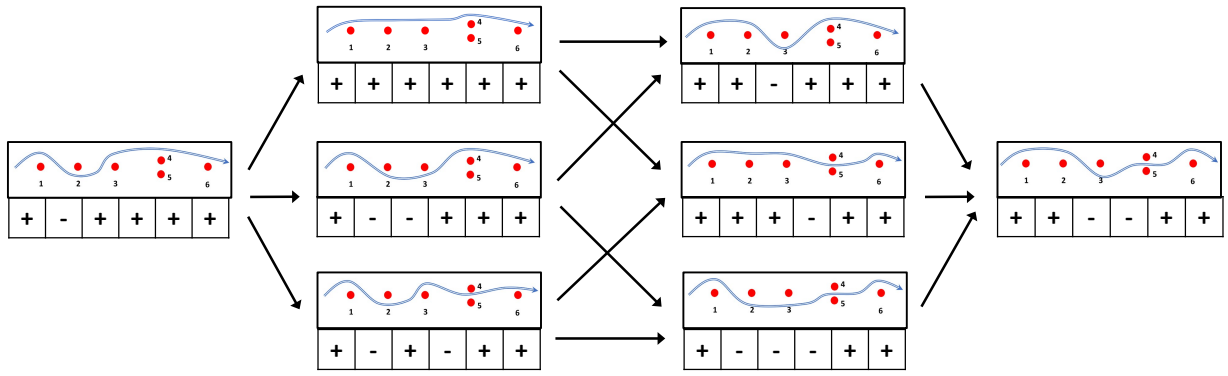


Figure 5.4: Planning from the initial spatial representation to the goal spatial representation. Intermediate states are generated along each path, where the spatial representation vector changes in one and only one dimension at every step.

and X^{t-1} via Coherent Point Drift (CPD) [44], where y_i^t is the position of $i\text{-}th$ point in the point cloud, M is the number of points in the point cloud, and usually $M >> N$.

One problem of registration cables using CPD is that the registered nodes do not have equal distances between neighboring nodes. [63] adds a regularization term in its optimization objective to maintain the local structure. Here we apply a simpler remedy by connecting the registered nodes and re-sampling along the connected path to obtain equally distributed nodes, which is found effective in our experiments.

Planning with Spatial Representation

Empirically, we observed that matching the cable nodes exactly with their correspondences on the goal configuration is unnecessary. Rather, it is sufficient that the spatial relation between the cable and the fixtures matches the goal configuration.

We define the spatial representation for cable routing in the $2D$ horizontal plane. The positions of cable nodes projected in the horizontal plane are denoted as $\tilde{X} = [\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_N] \in R^{N\times 2}$. The fixture positions are denoted as $P = [p_1, p_2, ..., p_K] \in R^{K\times 2}$, where K is the number of fixtures. We define two directional vectors $\vec{v}_{i1} = x_j - p_i$ (green arrows in Fig. 5.3) and $\vec{v}_{i2} = x_{j+1} - x_{j-1}$ (orange arrows in Fig. 5.3), where x_j is the closest nodes to the $i\text{-}th$ fixture p_i. The spatial representation for each fixture is then defined with a plus/minus sign, as illustrated in Fig. 5.3. Formally, the spatial states are determined as $\mathbf{s} = [s_1, s_2, \cdots, s_K] \in \{-1, +1\}^K$, where $s_i = \text{Sign}(\frac{\vec{v}_{i1}\times\vec{v}_{i2}}{|\vec{v}_{i1}\times\vec{v}_{i2}|} \cdot \vec{e}_z)$ and $\vec{e}_z$ is the unit normal of the horizontal plane. Intuitively, if we trace the cable from the fixed end to the other end, the spatial representation indicates whether each fixture is on the "left" or "right" side of the cable. The channel fixtures are treated as two single fixtures, e.g., fixture 4 and 5 in the example. Fig. 5.4 demonstrates a few example cable configurations along with their corresponding spatial states. Note that we do not consider the scenario where the cable circles around the fixtures.

Leveraging the proposed representation, a high-level planner for the long-horizon cable routing tasks can be easily implemented by only allowing single element changes in the spatial state $\mathbf{s}$ in each step. Fig. 5.4 illustrates example paths and intermediate states searched connecting the initial to the goal spatial state. In practice, we select the path along which the spatial state $\mathbf{s}$ changes sequentially from the cable's fixed end to the tail since it facilitates the downstream manipulation.

Learning Manipulation Primitives

In order to manipulate the cable to a desired spatial state, low-level robot commands such as picking and placing poses are needed. The low-level planner should be able to handle different cable configurations as well as diverse fixture locations.

Manipulation primitives

We propose a low-level cable routing planner with three manipulation primitives, `stretch`, `cross`, and `insert`, as demonstrated in Fig. 5.5. Each primitive consists of a pick-move-place action sequence where the picking and placing actions are constrained to be vertical. Specifically, in `stretch`, a robot picks a point on the cable and stretches the slack cable to establish contact with the fixtures. In `cross`, a robot selects one point on the cable to pick and transports the cable from one side of the fixture to the other. In `insert`, a robot picks the cable and inserts it between two fixtures of the channel.

As shown in Fig. 5.5, `cross` and `insert` change the spatial state, while `stretch` does not. The purpose of `stretch` is to reshape the cable to robustify `cross` and `insert`. In

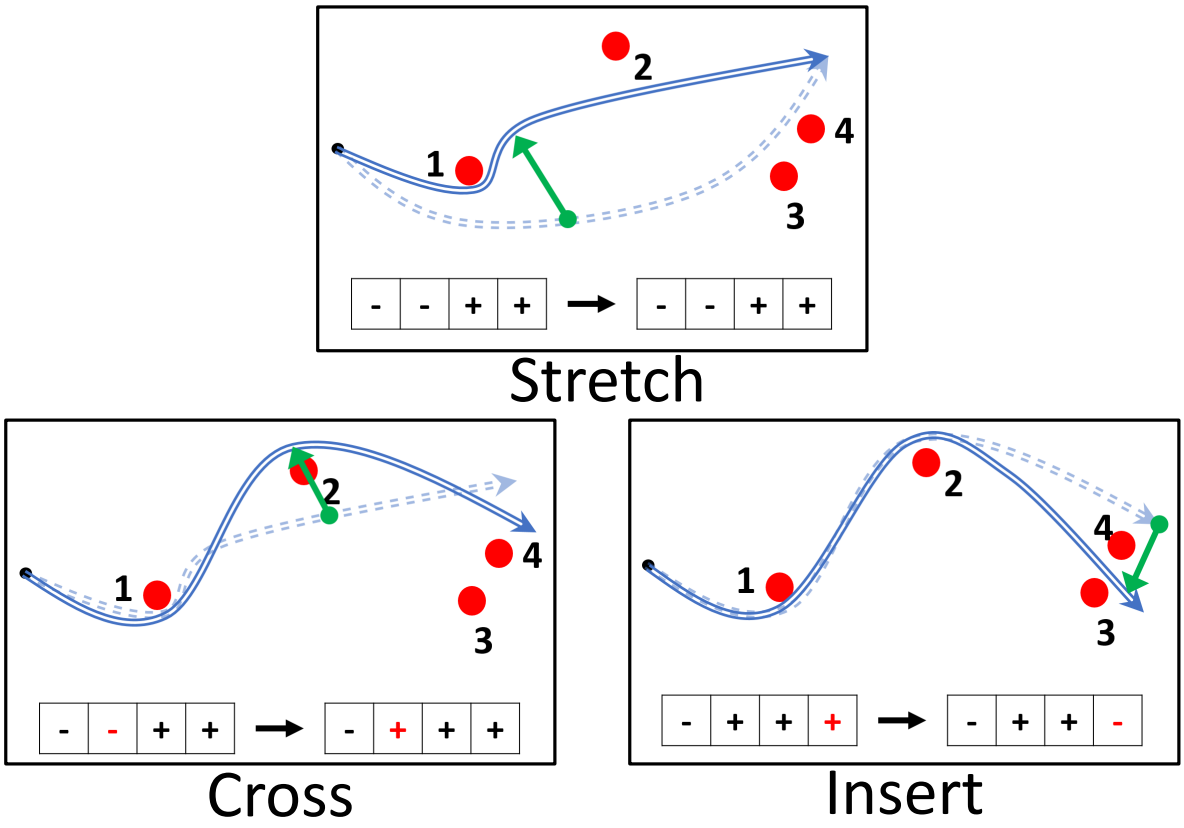


Figure 5.5: Manipulation primitives: (a) In `stretch`, a robot stretches the slack cable to establish contact with the fixtures. (b) In `cross`, a robot transports the cable from one side of the fixture to the other. (c) In `insert`, a robot inserts the cable between two fixtures of the channel. `cross` and `insert` change the spatial state, while `stretch` does not.

practice, after each `stretch`, one robot is holding the cable at the placing location, while the other robot performs `cross` or `insert`. The holding action is crucial since it prevents the cable from moving to undesired states during `cross` or `insert`. By iteratively executing `stretch-cross` or `stretch-insert`, robots are able to manipulate the cable to the desired configuration following the planned path in the spatial state space.

Learning manipulation primitives from labeled data

Each primitive consists of a picking point and a placing point. We propose to learn the target points from real data collected by randomly configuring the cable and fixtures in the workspace. The picking and placing target points for each primitive are then annotated based on the desired spatial state. Note that only the horizontal $2D$ positions of the target points are learned while the height is assumed to be known. The orientation of the picking

point is computed using the cable nodes' positions outputted from the cable state estimation, i.e., the yaw angle is the same as the estimated cable direction at the picking position. The orientation of the placing point is selected based on the desired spatial state. Specifically, the placing yaw angles are defined by the segment connecting the last and the next fixtures for `cross`, and by the segment connecting the placing position and the next fixture for `stretch` and `insert`. Learning the orientation and further, a 6-DoF pose is left to future work.

Two methods are implemented and compared to learn the target positions. In the first method, direct regression is applied using a Multilayer Perceptron (MLP) with two fully connected layers, whose inputs include cable nodes' positions, fixtures' positions, and the target spatial state. The output is the concatenated vector of the $2D$ picking and placing positions. In the second method, the cable nodes' and fixtures' positions are encoded in an image as the input to a U-Net[55], and similarly, the picking and placing target positions are encoded in a heatmap as the output. During training, the output heatmaps are constructed by the convolution of the point coordinates with a Gaussian kernel $\Phi = exp(-\frac{||p-p^*||^2}{2\sigma^2})$, where we select $\sigma = 4$ pixels. We hypothesize that outputting target point heatmaps allows for better spatial generalization than direct regression on point coordinates. A quantitative analysis on the two methods will be performed in Sec. 5.5.

5.5 Experiments

We aim to investigate three questions in our real-world experiments.

- First, we examine if the proposed framework based on spatial representations can solve cable routing tasks with various cables and fixture settings.
- Second, we evaluate whether the learned manipulation primitives outperform hand-designed policies and which learned model performs better.
- Third, we inspect whether the perception based cable state estimator provides reliable estimates for downstream planning and manipulation.

Experimental Setup

As shown in Fig. 5.1, our system includes two 7-DoF KuKa IIWA robot manipulators, a Kinect Azure RGB-D camera, two Robotiq Hand-E grippers, several single and channel fixtures, and 7 non-stretchable deformable cables (Fig. 5.6). We configured four routing scenarios with different goal configurations, as shown in Fig. 5.7. In each scenario, one end of the cable is rigidly attached to the table, and the fixtures are fixed on the table with known positions and orientations. We assume that the cable is within the robots' workspace throughout task execution, and collision-free robot motion sequences exist to manipulate the cable to desired configurations. Solving cable routing tasks with more dense fixtures and allowing 6-DoF picking/placing are left to future work.

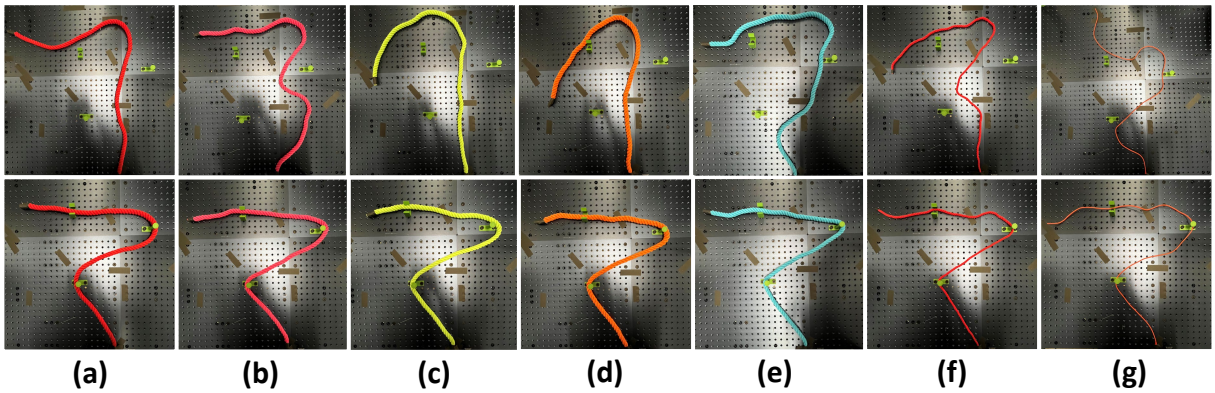


Figure 5.6: Cable routing with different cables for the same target spatial state (scenario 1). First row: initial state. Second row: final state. (a) red rope. (b) pink rope. (c) lime rope. (d) orange rope. (e) blue rope. (f) red thin rope. (g) red USB cable.

Implementation Details

Rule-based manipulation baseline

To evaluate the learned manipulation primitives, we implemented a rule-based policy that is composed of the same set of primitives as in Sec. 5.4, except that the picking and placing points are selected according to heuristics. Specifically, we compute the distance from all cable nodes to the relevant fixture and find the closest cable node $\tilde{x}_{closest}$ to the fixture. The node that is c nodes away from $\tilde{x}_{closest}$ is selected as the picking location, and the placing location is chosen as d distance away from the relevant fixture along the desired direction. By tuning c and d, the rule-based method achieves decent performance in each specific scenario, but a fixed value pair is found difficult to generalize. We experimentally set $c = 5$ and $d = 0.05m$ in all scenarios.

Model learning for manipulation primitives

For each manipulation primitive, 100 human demonstrations are collected, with varying fixture locations and different start and end cable configurations. We then annotate the picking and placing locations for each demonstration. The annotated dataset is augmented with flipping, rotating, injecting noise, and node resampling along the cable. Resampling nodes is crucial to data efficiency as the exact node locations are irrelevant to the task while the spatial relation between nodes and the fixtures matters.

Table 5.1: Comparison of methods with differently acquired primitives. Failure modes: A(over-stretching), B(slack and failed to cross), C(far-off prediction)

	Methods	Success rate	Failure modes
Scenario 1	Rule-based	2/5	A(2), B(1), C(0)
	Regression	4/5	A(1), B(0), C(0)
	Heatmap	**5/5**	A(0), B(0), C(0)
Scenario 2	Rule-based	1/5	A(1), B(3), C(0)
	Regression	**4/5**	A(1), B(0), C(0)
	Heatmap	**4/5**	A(0), B(0), C(1)
Scenario 3	Rule-based	1/5	A(4), B(0), C(0)
	Regression	3/5	A(0), B(2), C(0)
	Heatmap	**4/5**	A(1), B(0), C(0)
Scenario 4	Rule-based	0/5	A(3), B(2), C(0)
	Regression	0/5	A(0), B(5), C(0)
	Heatmap	**2/5**	A(0), B(3), C(0)
Overall	Rule-based	4/20	A(10), B(6), C(0)
	Regression	11/20	A(2), B(7), C(0)
	Heatmap	**15/20**	A(1), B(3), C(1)

Table 5.2: Cable routing with different cables. Failure modes: A(over-stretching), B(slack and failed to cross), C(far-off prediction), D(wrongly estimated cable state)

Cables	Success rate	Failure modes
Rope (red)	5/5	A(0), B(0), C(0), D(0)
Rope (pink)	5/5	A(0), B(0), C(0), D(0)
Rope (lime)	3/5	A(0), B(0), C(0), D(2)
Rope (orange)	4/5	A(0), B(0), C(1), D(0)
Rope (blue)	5/5	A(0), B(0), C(0), D(0)
Thin Rope (red)	3/5	A(0), B(1), C(1), D(0)
USB Cable (red)	3/5	A(0), B(2), C(0), D(0)

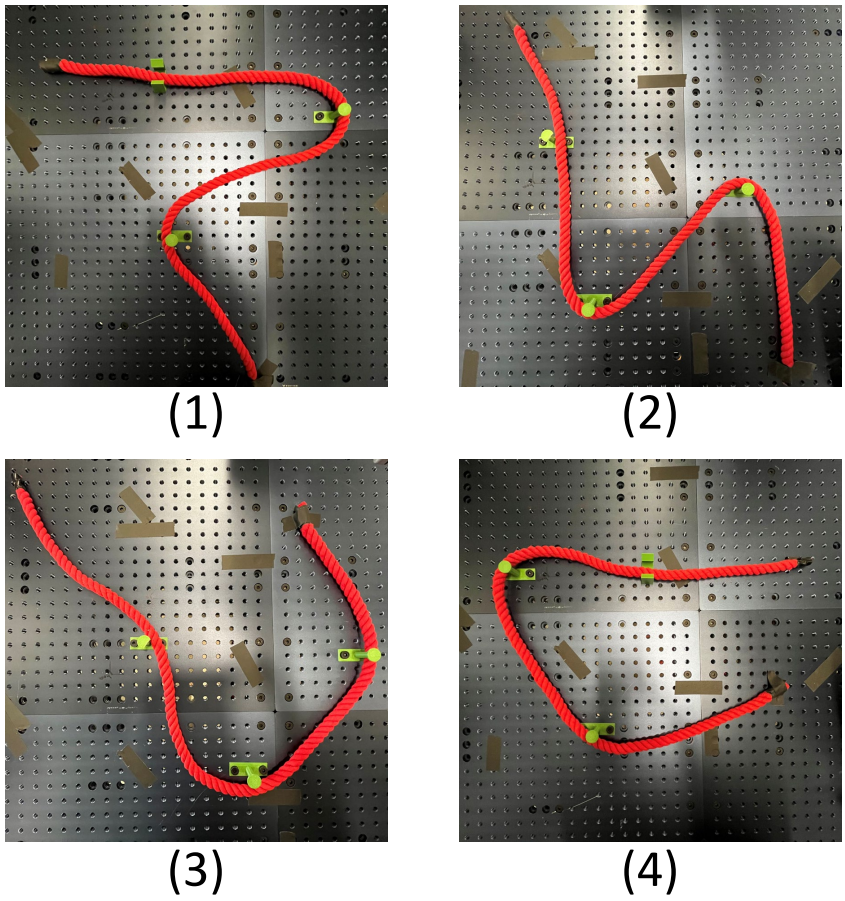


Figure 5.7: Four experiment scenarios where the fixture setting and target cable configurations are different.

Results

Table 5.1 compares cable routing success rates on the red rope with differently acquired primitives in all four scenarios (Fig. 5.7). In each scenario, there are three single/channel fixtures. As seen in the table, the rule-based policy does not perform reliably in different cable and fixture configurations. The regression method achieves decent performance, but its predicted target points have a large variance leading to over-stretching or slack cables. In additionn, there are occasions where the cable or the fixture might break if not interrupted. By contrast, learning with encoded heatmaps achieves a higher success rate with few over-stretching or slack cable cases. This suggests that outputting heatmaps allows for better spatial generalization than directly regressing on point coordinates. Fig. 5.7 shows some example trials where cables are manipulated to the desired goal spatial states.

Table 5.2 shows the experimental results on different cables (Fig. 5.6) with heatmap-

based learned primitives. As summarized in failure mode D, the cable state estimation succeeds in 33 out of 35 trials. This illustrates the proposed cable state estimator is able to detect cables of different colors and thicknesses in diverse configurations, despite being accessible to only limited data with the red rope during training. The inferred segmentation mask sometimes contains wrong predictions, especially when the robot partially occludes the cable. Even with a lower quality mask prediction, the cable state estimation remains accurate most times thanks to the cable initialization and node registration being robust to occlusion and outliers, as shown in the left plot of Fig. 5.2.

Table 5.2 also reports the performance of the proposed method with cables of different physical properties, such as thinner/softer ropes and a USB cable. Although the dynamics of the unseen cables are different from the trained thick rope, the `stretch` primitives and holding actions with reasonably predicted target points alleviate the effect of cable dynamics. Overall, a success rate of 28 out of 35 is achieved, showing promising performance of our cable routing method applied to various cables. To further improve the reliability, we suggest exploring real-time cable tracking and leveraging force/torque feedback. In addition, higher-bandwidth reactive planning and execution would help to detect failure modes early and recover from them.

5.6 Chapter Summary

This chapter proposes spatial representation for cable routing, which bridges the high-level long-horizon configuration planning and the low-level manipulation primitive execution. A simple configuration planner is implemented with the proposed representation to achieve the desired spatial relationship between the cable and fixtures. In addition, multiple execution primitives are designed and learned from the collected data, including stretching, crossing, and inserting. Real-world cable routing experiments are conducted with multiple cables, varying in visual appearances, physical properties, and fixture settings, demonstrating the method's effectiveness.

Chapter 6

Conclusions and Future Work

This dissertation introduced methodologies for enabling industrial robots to learn and plan for various factory manufacturing tasks. We study 1) how to learn manipulation skills when there are uncertainties in the object state estimation, 2) how to generalize the manipulation skills to different even unseen scenarios, 3) how to learn the high-level task planning for long-horizon tasks. We applied the learned skills to a series of applications including robotic assembly (Chapter 2), deformable objects tracking and manipulation (Chapter 3), belt drive units assembly (Chapter 4), and robotic cable routing (Chapter 5).

Chapter 2 proposes a novel framework to identify contact pose for peg-in-hole assembly under position uncertainties. The proposed method utilizes a tilt-then-rotate strategy to generate contact patterns. A CNN is utilized to classify the contact poses and guide the robot to achieve the assembly task with admittance control. Simulation and experiment results are provided to demonstrate the effectiveness of the proposed method. The proposed method has several advantages. 1) The mapping from the contact pattern to the contact pose is injective. 2) The method is robust to sensor noise. 3)The contact pose classification model is easy to obtain. All the training data can be quickly generated in simulation with a self-supervised scheme. 4) The method has good generalization ability and small sim-to-real gap. Since the contact data is normalized and recorded in a polar coordinate, the pattern is sensitive neither to the size of the object nor the parameters of the admittance controller. A model learned from a larger peg-hole can be successfully applied to smaller ones as long as the geometries are the same. Furthermore, the model learned in simulation can be adapted to the real world, despite the huge sim-to-real gap. For future works, we plan to improve the tilt-then-rotate strategy so that it can handle large orientation uncertainties and test it with more challenging peg-hole shapes. We also intend to incorporate active and adaptive sensing strategies to our framework, so that we don't have to use a predefined tilt-then-rotate strategy.

We also propose an alignment module and insertion module to deal with large pose uncertainties in 6 DoF. In the alignment module, we utilize the Deep Neural Network (DNN) to segment the peg and the hole from depth images. We can recover the $3D$ point cloud of the workpieces by detecting the flat surfaces. Then we use point set registration

to find the rotation and translation to align the peg and the hole. The alignment module is able to reduce the pose uncertainties into a small region and provides a safe and efficient action space for the insertion module. The insertion module compensates for the remaining small uncertainties with an impedance controller by tracking a reference generated from an RL policy. We have successfully validated the proposed method in simulations. We also show that the alignment module can work well in real-world peg-in-hole experiments. For future work, we will train and test the reinforcement learning and compliance controller in the real-world. Considering the scenario where peg can slip in the robot end-effector during assembly is an interesting research direction.

Chapter 3 proposes a novel framework SPR-RWLS for cable manipulation. In the framework, we combine real-time cable tracking and online deformation model approximation. For real-time tracking, a Gaussian mixture model based non-rigid registration is able to track deformable cable robustly in the presence of sensor noise, outliers, and occlusions. For deformation model approximation, the local deformation model can be approximated online by solving a robust optimization in parallel under uncertainty. Experiments showed that the proposed method is able to manipulate deformable cable to desired shape robustly. For future work, we plan to test the performance on more complicated desired shapes. For a desired shape that is far from the initial shape or when the cable is too long, we plan to develop a method that can automatically and efficiently generate intermediate desired shapes. SPR is proved to be robust for deformable cable tracking in $3D$ space. More manipulation tasks with deformable cable and deformable $2D$ cloth will be tested in $3D$ space.

Chapter 4 proposes a trajectory optimization formulation to assemble the belt drive unit. We propose a 3D keypoints representation to model the elastic belt, which simplifies the complexity of the trajectory optimization problem. The problem is formulated as an MPCC with complementarity constraints to model the hybrid dynamics due to contact and elastic forces. Simulations results show that the proposed approach can find feasible trajectories for the belt drive unit assembly with known but variable geometry. To the best of our knowledge, this is the first work that formalizes the trajectory optimization problem for the belt drive unit assembly, and the solution works in the real system. Several future works are possible. The current method is based on the execution of an open-loop trajectory which could fail under uncertainties in the position of the pulleys or of the belt. Adding a feedback controller is fundamental for a more robust and reliable solution. Moreover, in order to improve the generality of the problem, we are interested in an autonomous selection of the 3D keypoints for a given task. Our formulation of a trajectory optimization problem for deformable objects using complementarity constraints is not limited to belt drive unit assembly. The proposed method might be applied to a wider range of tasks such as cable routing and wire harness.

Chapter 5 proposes a spatial representation for cable routing, which bridges the high-level long-horizon configuration planning and the low-level manipulation primitive execution. A simple configuration planner is implemented with the proposed representation to achieve the desired spatial relationship between the cable and fixtures. In addition, multiple execution primitives are designed and learned from the collected data, including stretching, crossing,

and inserting. Real-world cable routing experiments are conducted with multiple cables, varying in visual appearances, physical properties, and fixture settings, demonstrating the method's effectiveness. For future work, the cable segmentation neural network can be trained with more data. It will be great if the network can robustly detect a thin cable of any color in an unseen clutter environment. The execution of manipulation primitive is open loop. Adding force/torque feedback on the end-effector and a real-time failure detection module could improve the robustness of the proposed method and detect the slack or overstretching problem.

Bibliography

[1] https://shiyujin0.github.io/TiltThenRotate/ACC2021.html.

[2] https://changhaowang.github.io/IROS2019/SPRRWLS.html.

[3] Gabriel Arslan Waltersson. "Planning and control for cable-routing with dual-arm robot". In: (2021).

[4] Siciliano Bruno and Oussama Khatib. "Springer Handbook of Robotics". In: 2008.

[5] Alexander Buslaev et al. "Albumentations: Fast and Flexible Image Augmentations". In: *Information* 11.2 (2020). ISSN: 2078-2489. DOI: 10.3390/info11020125. URL: https://www.mdpi.com/2078-2489/11/2/125.

[6] Siddharth Chhatpar and Michael Branicky. "Search Strategies for Peg-in-Hole Assemblies with Position Uncertainty". In: *2001 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Maui, Hawaii, USA, 2001.

[7] Siddharth R Chhatpar and Michael S Branicky. "Particle filtering for localization in robotic assemblies with position uncertainty". In: *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2005, pp. 3610–3617.

[8] Cheng Chi and Dmitry Berenson. "Occlusion-robust deformable object tracking without physics simulation". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2019, pp. 6443–6450.

[9] Haili Chui and Anand Rangarajan. "A new point matching algorithm for non-rigid registration". In: *Computer Vision and Image Understanding* 89.2-3 (2003), pp. 114–141.

[10] Alexander Domahidi, Eric Chu, and Stephen Boyd. "ECOS: An SOCP solver for embedded systems". In: *2013 European Control Conference (ECC)*. IEEE. 2013, pp. 3071–3076.

[11] Samuel Hunt Drake. "Using compliance in lieu of sensory feedback for automatic assembly." PhD thesis. Massachusetts Institute of Technology, 1978.

[12] Felix von Drigalski et al. *Robots Assembling Machines: Learning from the World Robot Summit 2018 Assembly Challenge*. 2019. arXiv: 1911.05884 [cs.RO].

[13] Yongxiang Fan, Jieliang Luo, and Masayoshi Tomizuka. "A learning framework for high precision industrial assembly". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 811–817.

[14] Rafael Fierro et al. "Hybrid control of formations of robots". In: *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)*. Vol. 1. IEEE. 2001, pp. 157–162.

[15] Philipp Foehn et al. "Fast Trajectory Optimization for Agile Quadrotor Maneuvers with a Cable-Suspended Payload." In: *RSS* (2017).

[16] Ross Girshick. "Fast r-cnn". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1440–1448.

[17] Kaiming He et al. "Mask r-cnn". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2961–2969.

[18] Masaki Hilaga et al. "Topology matching for fully automatic similarity estimation of 3D shapes". In: *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. 2001, pp. 203–212.

[19] Ryan Hoque et al. "VisuoSpatial Foresight for Physical Sequential Fabric Manipulation". In: *arXiv preprint arXiv:2102.09754* (2021).

[20] *https://worldrobotsummit.org/en/about/*.

[21] *https://www.nist.gov/el/intelligent-systems-division-73500/iros-2020-robotic-grasping-and-manipulation-competition*.

[22] Zhe Hu, Peigen Sun, and Jia Pan. "Three-dimensional deformable object manipulation using fast online gaussian process regression". In: *IEEE Robotics and Automation Letters* 3.2 (2018), pp. 979–986.

[23] Shiyu Jin, Changhao Wang, and Masayoshi Tomizuka. "Robust deformation model approximation for robotic cable manipulation". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2019, pp. 6586–6593.

[24] Shiyu Jin et al. "Contact Pose Identification for Peg-in-Hole Assembly under Uncertainties". In: *2021 American Control Conference (ACC)*. 2021, pp. 48–53. DOI: 10.23919/ACC50511.2021.9482981.

[25] Shiyu Jin et al. "Real-time State Estimation of Deformable Objects with Dynamical Simulation". In: *Workshop on Robotic Manipulation of Deformable Objects 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2020.

[26] Shiyu Jin et al. "Robotic Cable Routing with Spatial Representation". In: *IEEE Robotics and Automation Letters*. 2021 (in submission).

[27] Shiyu Jin et al. "Trajectory Optimization for Manipulation of Deformable Objects: Assembly of Belt Drive Units". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. 2021, pp. 10002–10008. DOI: 10.1109/ICRA48506.2021.9561556.

[28] Lars Johannsmeier, Malkin Gerchow, and Sami Haddadin. "A framework for robot manipulation: Skill formalism, meta learning and adaptive control". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 5844–5850.

[29] Leslie Pack Kaelbling and Tomás Lozano-Pérez. "Hierarchical planning in the now". In: *Workshops at the Twenty-Fourth AAAI Conference on Artificial Intelligence*. 2010.

[30] Mrinal Kalakrishnan et al. "STOMP: Stochastic trajectory optimization for motion planning". In: *2011 IEEE international conference on robotics and automation*. IEEE. 2011, pp. 4569–4574.

[31] Young-Loul Kim, Hee-Chan Song, and Jae-Bok Song. "Hole detection algorithm for chamferless square peg-in-hole based on shape recognition using F/T sensor". In: *International journal of precision engineering and manufacturing* 15.3 (2014), pp. 425–432.

[32] Patrik Kolaric et al. "Local policy optimization for trajectory-centric reinforcement learning". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 5094–5100.

[33] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems 25*. Curran Associates, Inc., 2012, pp. 1097–1105.

[34] Shunsuke Kudoh et al. "In-air knotting of rope by a dual-arm multi-finger robot". In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2015, pp. 6202–6207.

[35] Florent Lamiraux and Lydia E Kavraki. "Planning paths for elastic objects under manipulation constraints". In: *The International Journal of Robotics Research* 20.3 (2001), pp. 188–208.

[36] Michelle A Lee et al. "Making sense of vision and touch: Self-supervised learning of multimodal representations for contact-rich tasks". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 8943–8950.

[37] Sergey Levine et al. "End-to-End Training of Deep Visuomotor Policies". In: *The Journal of Machine Learning Research*. 2016.

[38] Wen Hao Lui and Ashutosh Saxena. "Tangled: Learning to untangle ropes with rgb-d perception". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2013, pp. 837–844.

[39] Zhi-Quan Luo, Jong-Shi Pang, and Daniel Ralph. *Mathematical programs with equilibrium constraints*. Cambridge University Press, 2008.

[40] Lucas Manuelli et al. "kPAM: KeyPoint Affordances for Category-Level Robotic Manipulation". In: (2019).

[41] Dimitris Metaxas and Demetri Terzopoulos. "Shape and nonrigid motion estimation through physics-based synthesis". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 15.6 (1993), pp. 580–591.

[42] Mark Moll and Lydia E Kavraki. "Path planning for deformable linear objects". In: *IEEE Transactions on Robotics* 22.4 (2006), pp. 625–636.

[43] Takuma Morita et al. "Knot planning from observation". In: *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*. Vol. 3. IEEE. 2003, pp. 3887–3892.

[44] Andriy Myronenko and Xubo Song. "Point set registration: Coherent point drift". In: *IEEE transactions on pattern analysis and machine intelligence* 32.12 (2010), pp. 2262–2275.

[45] Ashvin Nair et al. "Combining self-supervised learning and imitation for vision-based rope manipulation". In: *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2017, pp. 2146–2153.

[46] David Navarro-Alarcon and Yun-Hui Liu. "Fourier-based shape servoing: A new feedback method to actively deform soft objects into desired 2-D image contours". In: *IEEE Transactions on Robotics* 34.1 (2017), pp. 272–279.

[47] David Navarro-Alarcon et al. "Model-free visually servoed deformation control of elastic objects by robot manipulators". In: *IEEE Transactions on Robotics* 29.6 (2013), pp. 1457–1468.

[48] Christian Ott, Ranjan Mukherjee, and Yoshihiko Nakamura. "Unified impedance and admittance control". In: *2010 IEEE international conference on robotics and automation*. IEEE. 2010, pp. 554–561.

[49] Antoine Petit, Vincenzo Lippiello, and Bruno Siciliano. "Real-time tracking of 3D elastic objects with an RGB-D sensor". In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2015, pp. 3914–3921.

[50] Michael Posa, Cecilia Cantu, and Russ Tedrake. "A direct method for trajectory optimization of rigid bodies through contact". In: *The International Journal of Robotics Research* 33.1 (2014), pp. 69–81. DOI: 10.1177/0278364913506757.

[51] Arvind U Raghunathan and Lorenz T Biegler. "An interior point method for mathematical programs with complementarity constraints (MPCCs)". In: *SIAM Journal on Optimization* 15.3 (2005), pp. 720–750.

[52] Ixchel G Ramirez-Alpizar, Kensuke Harada, and Eiichi Yoshida. "Motion planning for dual-arm assembly of ring-shaped elastic objects". In: *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE. 2014, pp. 594–600.

[53] Nathan Ratliff et al. "CHOMP: Gradient optimization techniques for efficient motion planning". In: *2009 IEEE International Conference on Robotics and Automation*. IEEE. 2009, pp. 489–494.

[54] Joseph Redmon et al. "You only look once: Unified, real-time object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 779–788.

[55] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation". In: *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, pp. 234–241.

[56] Jose Sanchez et al. "Robotic manipulation and sensing of deformable objects in domestic and industrial applications: a survey". In: *The International Journal of Robotics Research* 37.7 (2018), pp. 688–716.

[57] John Schulman et al. "Learning from demonstrations through the use of non-rigid registration". In: *Robotics Research*. Springer, 2016, pp. 339–354.

[58] John Schulman et al. "Motion planning with sequential convex optimization and convex collision checking". In: *The International Journal of Robotics Research* 33.9 (2014), pp. 1251–1270.

[59] John Schulman et al. "Tracking deformable objects with point clouds". In: *2013 IEEE International Conference on Robotics and Automation*. IEEE. 2013, pp. 1130–1137.

[60] Daniel Seita et al. "Learning to Rearrange Deformable Cables, Fabrics, and Bags with Goal-Conditioned Transporter Networks". In: *arXiv preprint arXiv:2012.03385* (2020).

[61] Priya Sundaresan et al. "Learning rope manipulation policies using dense object descriptors trained on synthetic depth data". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 9411–9418.

[62] Priya Sundaresan et al. "Untangling Dense Non-Planar Knots by Learning Manipulation Features and Recovery Policies". In: *arXiv preprint arXiv:2107.08942* (2021).

[63] Te Tang and Masayoshi Tomizuka. "Track deformable objects from point clouds with structure preserved registration". In: *The International Journal of Robotics Research* (2018), p. 0278364919841431.

[64] Te Tang, Changhao Wang, and Masayoshi Tomizuka. "A framework for manipulating deformable linear objects by coherent point drift". In: *IEEE Robotics and Automation Letters* 3.4 (2018), pp. 3426–3433.

[65] Te Tang et al. "Autonomous alignment of peg and hole by force/torque measurement for robotic assembly". In: *2016 IEEE international conference on automation science and engineering (CASE)*. IEEE. 2016, pp. 162–167.

[66] Te Tang et al. "Robotic manipulation of deformable objects by tangent space mapping and non-rigid registration". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, pp. 2689–2696.

[67] Te Tang et al. "State estimation for deformable objects by point registration and dynamic simulation". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 2427–2433.

[68] Te Tang et al. "Teach industrial robots peg-hole-insertion by human demonstration". In: *2016 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*. IEEE. 2016, pp. 488–494.

[69] Keiki Tatemura and Hiroki Dobashi. "Strategy for roller chain assembly with parallel jaw gripper". In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 2435–2442.

[70] Bugra Tekin, Sudipta N. Sinha, and Pascal Fua. "Real-Time Seamless Single Shot 6D Object Pose Prediction". In: *CVPR*. 2018.

[71] Ulrike Thomas et al. "Multi sensor fusion in robot assembly using particle filters". In: *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE. 2007, pp. 3837–3843.

[72] Josh Tobin et al. "Domain randomization for transferring deep neural networks from simulation to the real world". In: *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE. 2017, pp. 23–30.

[73] 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.

[74] Joshua C. Triyonoputro, Weiwei Wan, and Kensuke Harada. "Quickly Inserting Pegs into Uncertain Holes using Multi-view Images and Deep Network Trained on Synthetic Data". In: *CoRR* abs/1902.09157 (2019). arXiv: `1902.09157`.

[75] Arjan J Van Der Schaft and Johannes Maria Schumacher. *An introduction to hybrid dynamical systems*. Vol. 251. Springer London, 2000.

[76] Andreas Wächter and Lorenz T. Biegler. "On the Implementation of an Interior-Point Filter Line-Search Algorithm for Large-Scale Nonlinear Programming". In: *Math. Program.* 106.1 (Mar. 2006), pp. 25–57.

[77] Angelina Wang et al. "Learning robotic manipulation through visual planning and acting". In: *arXiv preprint arXiv:1905.04411* (2019).

[78] Daniel E Whitney et al. "Quasi-static assembly of compliantly supported rigid parts". In: *Journal of Dynamic Systems, Measurement, and Control* 104.1 (1982), pp. 65–77.

[79] Yilin Wu et al. "Learning to manipulate deformable objects without demonstrations". In: *arXiv preprint arXiv:1910.13439* (2019).

[80] Yu Xiang et al. "PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes". In: *Robotics: Science and Systems (RSS)*. 2018.

[81] Mengyuan Yan et al. "Learning topological motion primitives for knot planning". In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2020, pp. 9457–9464.

[82] Mengyuan Yan et al. "Self-supervised learning of state estimation for manipulating deformable linear objects". In: *IEEE robotics and automation letters* 5.2 (2020), pp. 2372–2379.

[83] Kerim Yunt. "An augmented Lagrangian based shooting method for the optimal trajectory generation of switching Lagrangian systems". In: *Dynamics of Continuous, Discrete and Impulsive Systems Series B: Applications and Algorithms* 18.5 (2011), pp. 615–645.

[84] Kerim Yunt and Christoph Glocker. "A combined continuation and penalty method for the determination of optimal hybrid mechanical trajectories". In: *IUTAM Symposium on Dynamics and Control of Nonlinear Systems with Uncertainty*. Springer. 2007, pp. 187–196.

[85] Kerim Yunt and Christoph Glocker. "Trajectory optimization of mechanical hybrid systems using SUMT". In: *9th IEEE International Workshop on Advanced Motion Control, 2006.* IEEE. 2006, pp. 665–671.

[86] Yuan F Zheng, Run Pei, and Chichyang Chen. "Strategies for automatic assembly of deformable objects". In: *Proceedings. 1991 IEEE International Conference on Robotics and Automation*. IEEE Computer Society. 1991, pp. 2598–2599.

[87] Jihong Zhu et al. "Dual-arm robotic manipulation of flexible cables". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 479–484.

[88] Jihong Zhu et al. "Robotic manipulation planning for shaping deformable linear objects with environmental contacts". In: *IEEE Robotics and Automation Letters* 5.1 (2019), pp. 16–23.