Title
Skill Learning for Industrial Robot Manipulators

Author
Tang, Te

Publication Date
2018

Skill Learning for Industrial Robot Manipulators

by

Te Tang

A dissertation submitted in partial satisfaction of the
requirements for the degree of
Doctor of Philosophy

in

Engineering - Mechanical Engineering

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Masayoshi Tomizuka, Chair
Professor Roberto Horowitz
Professor Pieter Abbeel

Summer 2018

Skill Learning for Industrial Robot Manipulators

Abstract

Skill Learning for Industrial Robot Manipulators

by

Te Tang

Doctor of Philosophy in Engineering - Mechanical Engineering

University of California, Berkeley

Professor Masayoshi Tomizuka, Chair

Industrial robots have been kept upgraded for decades to achieve extraordinary accuracy, speed, and repeatability. However, even the most advanced manipulator today is functioning as a programmable machine, instead of an intelligent agent. This deficiency of intelligence restricts robots from broader applications. To meet the increasing demand for automation, it is essential to make industrial robots more skillful and intelligent. Under this background, the objective of this dissertation is to develop generic and efficient methodologies to teach robots novel skills. Three major approaches including model-based learning, model-free learning and analogy-based learning are discussed and explored. A series of skills such as assembly, grasping, tracking and motion planning, have been successfully taught to industrial manipulators and evaluated by experiments.

As the name implies, model-based learning tries to formulate skills analytically based on physical models. In Chapter 2, an auto-alignment skill is developed for robotic assembly by constructing a novel contact model. Robots are enabled to predict tilt angles between assembly parts from force/torque measurement and perform fine assembly from large misalignment conditions. With this skill model, traditional procedures such as the installation of positioning fixtures and manual alignment can be skipped, which saves tremendous preparation efforts for robotic assembly.

However, not always system models can be constructed, especially for those complicated scenarios. The model-free approach is then developed to learn control policies by regressing general parametric functions. In Chapter 3, a compliant robotic force controller is learned from human demonstration. A human operator holds a specially-designed handle and demonstrates to robots the compliant insertions. The Gaussian mixture regression is introduced to fit motion patterns from measured data. This approach enables to transfer the compliant assembly skill from humans to robots efficiently and intuitively.

Besides model-based and model-free learning, an analogy-based leaning approach is proposed in Chapter 4. The distinct idea is that instead of pursuing a control policy either constructed by models or regressed from data, we discover the correlation between scenarios, i.e., analogy. The past scenario that bears a strong similarity with the current one will be

identified, and a mapping function between these two scenes will be constructed. Applying the mapping function, the past problem-solving action can be transferred to a new one that is feasible for the current scenario. This analogy-based approach has been implemented in multiple industrial tasks, and taught robots various skills such as grasping versatile objects (Chapter 5), tracking and manipulating deformable objects (Chapter 6) and efficient motion re-planning for similar scenarios (Chapter 7).

To my family

Contents

List of Figures

List of Tables

Acknowledgments

The past fives years at UC Berkeley have been an extraordinary period in my life. I am excited and grateful for knowing so many friends and professionals who accompanied, assisted and advised me through this unforgettable journey.

First of all, my deep and sincere gratitude goes to my Ph.D. advisor, Prof. Masayoshi Tomizuka. The accomplishment of this dissertation would not be possible without his consistent support from the first day I came to Berkeley. I highly appreciate the freedom he gave me to explore the unknowns, and at the same time, his insightful visions to head me the right direction when I felt lost. Besides as a remarkable mentor, Prof. Tomizuka is also my role model for his diligence in work, kindness to people and enthusiasm on life. I sincerely wish that I could become a great person as him in the future.

I am also thankful to Prof. Pieter Abbeel and Prof. Roberto Horowitz for serving on my dissertation committee as well as qualifying exam committee. Their classes provided me a brand-new vision of controls and robotics, and inspired me profoundly in my research.

Moreover, I am grateful to Prof. J. Karl Hedrick and Prof. Andrew Packard. Prof. Hedrick was the chair of my qualifying exam committee and Prof. Packard was my supervisor when I was performing graduate student instructor. Their insightful instructions and suggestions have an extensive influence on my study and life at Berkeley.

Special thanks also go to FANUC Corporation, Japan, for their generous support throughout these years. In particular, I am sincerely grateful to Dr. Wenjie Chen, Mr. Tetsuaki Katou and Mr. Kaimeng Wang for their valuable discussions from the industrial perspective.

I am grateful for meeting so many friends at Mechanical Systems Control (MSC) laboratory. Without my lovely labmates, my life at Berkeley would not be so much fun. I would like to thank all the MSC members, especially Hsien-Chung Lin, Yu Zhao, Yongxiang Fan, Changliu Liu, Cong Wang, Chung-Yen Lin, Yujiao Cheng, Jessica Leu and Shiyu Jin for their valuable discussions and suggestion in my research. I also would like to thank Minghui Zheng, Dennis Wai, Shiying Zhou, Liting Sun, Wei Zhan, Xiaowen Yu, Chen-Yu Chan, Junkai Lu, Kevin Haniger, Shuyang Li, Jianyu Chen, Cheng Peng, Zining Wang, Daisuke Kaneishi, Chen Tang, Zhuo Xu, Yu-Chu Huang, Jiachen Li, Yeping Hu, Hengbo Ma, Kiwoo Shin and Taohan Wang, for their kind friendship and help throughout the years. I hope you all the best in your future endeavors!

Finally and foremost, I would express my deepest gratitude to my parents, my sister, and my wife, Leyi Pan. You are always there for me no matter for good or bad time. It is your unconditional love and support that make me achieve so far. I love you all and love you forever.

Chapter 1

Introduction

1.1 Background of Robotic Skill Learning

Industrial robots have been applied in manufacturing and continuously upgraded for decades. Over time, they are equipped with a variety of advanced hardware and software to enhance accuracy, speed and repeatability, which are far beyond humans' physiological limit. However, their application scenarios are still limited because of their high rigidity and low adaptivity [14]. As the demands for automation grow rapidly, it is essential to teach industrial robots more skills to meet increasing task requirements, such as fine assembly [108], robust tracking [110], versatile grasping [76] and dexterous manipulation [107], etc.

Under this background, the study of robot learning [2, 47, 87] upsurges in recent years, trying to develop techniques that allow robots to acquire novel skills efficiently and robustly. It is a research field at the intersection of robotics, controls and machine learning.

As shown in Fig. 1.1, the general objective of robot learning is to find a control policy which produces suitable actions u corresponding to the observed system states x. Note that states and actions here are macroscopic and task-dependent. For a grasping problem, states might be the object point clouds, and actions are the inferred optimal grasping pose [67]; while for an assembly problem, states could be the sensed resistance force, and actions are the compliant motion commands [111]. The control policy can be regarded as a quantitative construction of the cognition or skills that robots possess.

There are multiple approaches to construct control policies. They can be categorized as illustrated in Fig. 1.2. A straightforward approach is to model the policy mathematically based on physical principles and deductions [7]. Some policies can be completely determined offline without the need of online execution, such as classical control [29] and Markov decision process [43]. Some policies are only partially constructed and need online data for completion, for example, system parameters require online identification (adaptive control [4]), reward function needs online approximation (model-based reinforcement learning [30]) or the embedded optimization needs online formulation and calculation (optimal control [120], MPC [18]). Note that if the formulation has a closed-form solution, such as LQR,

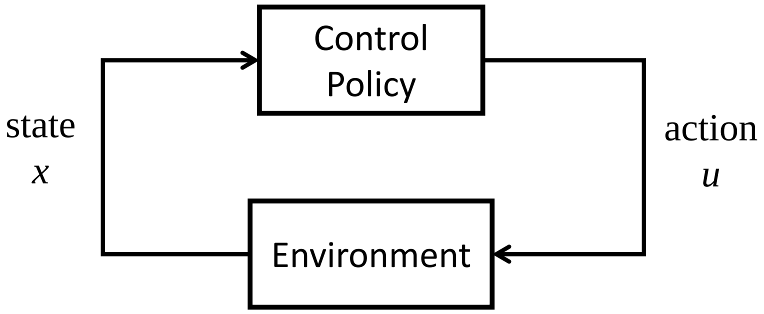


Figure 1.1: Learn a control policy to schedule actions corresponding to states.

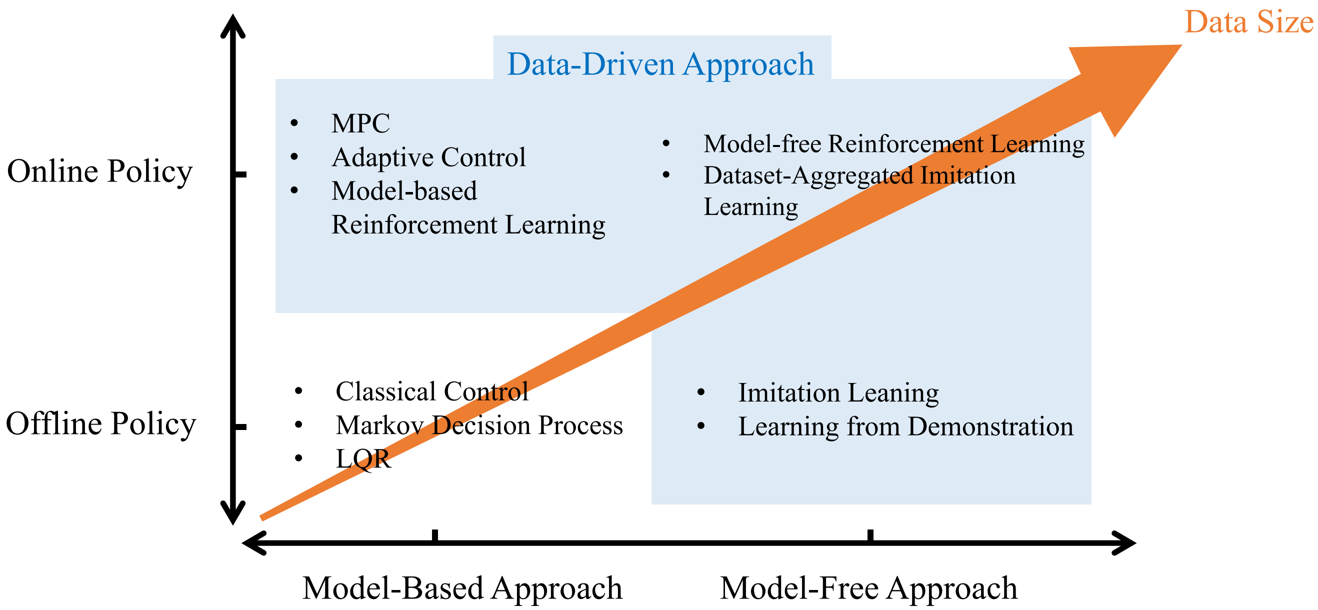
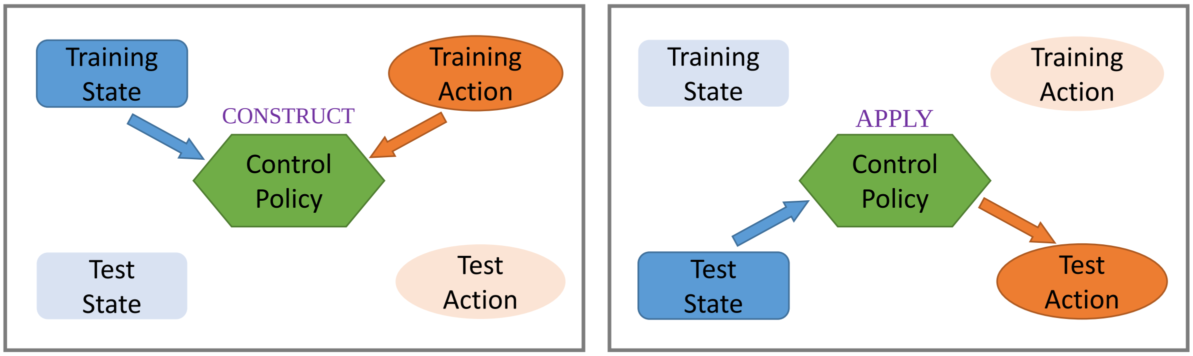


Figure 1.2: Classification of policy learning approaches.

then it still belongs to offline policy.

These model-based approaches build up the fundamentals of robotics. They have tremendous advantages in the system transparency and interpretability. Since each element is explicitly modelled, the stability and robustness of the system can be analyzed analytically, which is critical especially for industrial applications.

However, as robots are deployed to more sophisticated domains, such as grasping dexterous objects and manipulating deformable objects, a pure model-based approach runs into its bottleneck. In these scenarios, the system complexity might go beyond our capability to model, or the current constructed models cannot introduce satisfying results. On the other hand, along with the development of machine learning, especially the breakthrough of



(a) Training Stage

(b) Test Stage

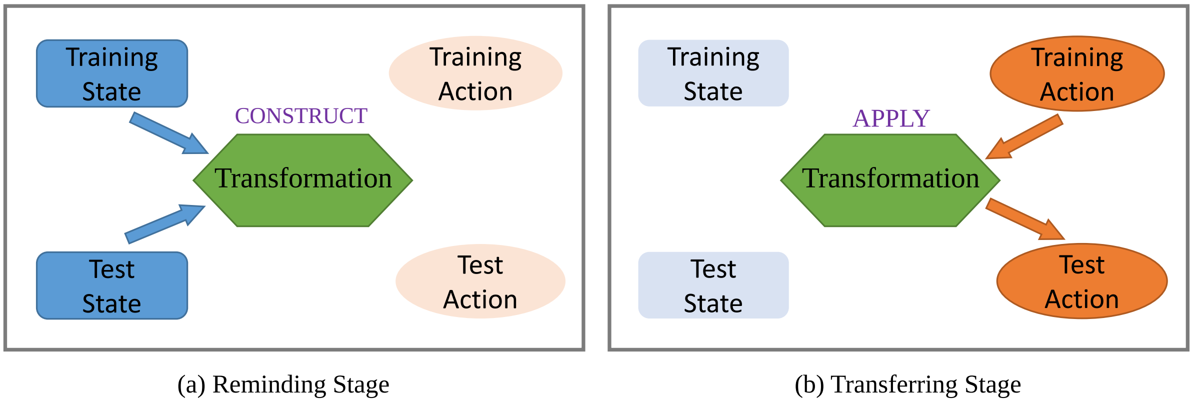
Figure 1.3: Pipeline of traditional learning methods. (a) During training stage, construct a control policy either by modelling or regressing from the training data. (b) During test stage, apply the control policy to the test states to achieve actions.

deep learning in recent years [59], people realize that though the human cognition process is complicated to model, it could be alternatively regressed by high dimensional parametric functions given a large amount of training data [80]. With this observation, many model-free approaches are proposed recently, trying to teach robots control policies which are learned from data, instead of modelling from principle. In the training stage, thousands or even millions of state-action pairs (x, u) are collected either by offline expert demonstration (imitation learning [9, 1], learning from demonstration [2, 6]), or by online trial and error (model-free reinforcement learning [77, 101]). The collected samples are then utilized to fit an optimal control policy $f : u = f(x)$. These model-free approaches feature in the universal structure and common training process, thus they can be quickly deployed to various scenarios. However, many of them are working in a black box manner, whose behavior is non-trivial to interpret and the system stability cannot be evaluated. Therefore, usually they are deployed for high-level tasks (e.g., vision detection) and the low-level implementation (e.g., motor servoing) is still based on model-based approaches. This hierarchical structure achieves a balance between performance and stability of the overall system. It is notable that for safety non-critical scenarios, there is also a trend of end-to-end learning [63, 13], where the hierarchical structures of systems are embedded in a single framework of deep neural networks, in which way the maximum potential of neural networks will be exploited.

The model-free approaches, together with the model-based approaches which need online system identification, are called data-driven approaches in general (Fig. 1.2).

Though there are numerous approaches for robot learning, most of them follow a similar pipeline as shown in Fig. 1.3. During training stage, the control policy is either constructed by modelling or regressed from training data. The trained control policy is assumed to be universal and applicable, therefore it is applied to test states to achieve corresponding test actions.

However, cognitive science finds that this pipeline is not necessarily the only way for



(a) Reminding Stage (b) Transferring Stage

Figure 1.4: Pipeline of analogy learning. (a) During reminding stage, construct a transformation between the correlated training states and test states. (b) During transferring stage, apply the transformation to the training actions to achieve the test actions.

human learning. Many studies showed that without understanding how the system progresses or the relation between states and actions (policy), humans can still make decisions through analogical reasoning [19, 88, 65]. Given a current scenario, human beings will try to find the correlation between the past and current, and transfer previous problem-solving actions to apply to the current situation. This analogy approach does not need to understand the universal policy or causality behind the system. It also enables humans to learn efficiently from small data samples.

Based on this observation, an analogy-based learning approach is proposed in this dissertation. Unlike those traditional model-based or model-free approaches, which try to formulate control policies, this new approach focuses on finding correlations between scenarios. As shown in Fig. 1.4, this approach has two general stages, the reminding stage and the transferring stage. During reminding stage, the similarity between the past scenarios and the current scene will be calculated. The most similar pair will be selected and a mapping function between the two scenarios will be constructed by non-rigid registration. During transferring stage, the past actions will be transformed by the mapping function so as to generate new actions which are feasible for the current situation. This approach is plausible for those applications where the model cannot be constructed while data is also non-trivial to collect. More properties and advantages of analogy-based learning will be introduced in detail in Chapter 4.

1.2 Dissertation Outline

In general, the objective of this dissertation is to develop generic methodologies to teach industrial robots novel skills. Three major approaches, including model-based learning, model-free learning and analogy-based learning, are discussed and explored. A series of skills, such as assembly, grasping, tracking and motion planning, have been successfully taught to industrial manipulators and evaluated by experiments. Experimental videos can be found at the supplementary website for this dissertation [104]. Figure 1.5 shows the structure overview of the dissertation. The chapter outline is as follows.

Robotics Assembly by Model-based Learning

In the past years, many methods have been developed for robotic peg-hole-insertion to automate the assembly process. However, most of them are based on the assumption that the peg and hole are well aligned before insertion starts. In practice, if there is a large pose (position/orientation) misalignment, the peg and hole may suffer from a three-point contact condition where traditional assembly methods cannot work. To deal with this problem, Chapter 2 proposes a novel three-point contact model to estimate the pose misalignment by force and geometric analysis. With the estimated values, the robot can autonomously correct the misalignment before applying traditional assembly methods to perform insertions. A series of experiments on a FANUC industrial robot and a H7h7 tolerance peg-hole testbed validate the effectiveness of the proposed method. Experimental results show that the robot is able to perform peg-hole insertion from unaligned conditions with 96% success rate. Part of this work was published in [108].

Robotics Assembly by Model-free Learning

For the insertion phase of robotic assembly, designing a satisfying force controller usually requires delicate parameter tuning, the procedure of which is unintuitive and time-consuming. In contrast, humans may accomplish assembly tasks with much less time and fewer trials. It will be a great benefit if robots can learn the humans' inherent skill of compliant motion and apply it to insertion. Chapter 3 proposes a model-free approach for learning a dynamic admittance controller from human demonstration. The basic idea is to collect the force and corrective velocity that humans apply during assembly, and then use Gaussian mixture regression (GMR) to fit a paramatric function to serve as the controller. The contribution of this work is that though as a model-free approach, a physical interpretation of GMR is found from a mechanics point of view. The system's closed-loop stability, therefore, can be evaluated and proved. A series of peg-hole insertion experiments on a FANUC manipulator validate the performance of the proposed learning method. Part of this work was published in [105] and [111].

Robotic Grasping by Analogy Learning

Grasp planning is essential for industrial robots to execute manipulation tasks. Solving the optimal grasps for various objects online, however, is challenging due to the heavy computation load during exhaustive sampling, and the difficulties to consider task requirements. Chapter 5 proposes an analogy-based approach to teach robots grasping by retrieving past experience. The example grasps are taught by human demonstration and mapped to similar objects by non-rigid registration. The mapped grasps are evaluated analytically and refined by an orientation search to improve the grasp robustness and robot reachability. The proposed approach is able to plan high-quality grasps, avoid collision, satisfy task requirements, and achieve efficient online planning. The effectiveness of the proposed method is verified by a series of grasping experiments. Part of this work was published in [67, 71].

Track and Manipulate Deformable Objects by Analogy Learning

Cable assembly is a labor-intensive task in the industry but has not been automated by robots for many years. The main challenge is that these deformable objects have infinite-dimensional configuration space and are computationally expensive to model, making it difficult for real-time tracking, planning and control. To deal with these challenges, a uniform framework which includes state estimation, task planning and trajectory planning is proposed in Chapter 6 based on the concept of analogy learning. A real-time observer is proposed to estimate the states of deformable objects by registering the last step estimation towards the current point clouds measurement. An online task planner is also developed to recognize the manipulation step according to the state estimation result. For trajectory planning, human operators first train example paths of robots given several object states. In the test stage, a new feasible path can be autonomously generated by a smooth transformation from training scenarios to test scenarios. A series of cable manipulation experiments on a dual-arm robotic platform are performed to validate the effectiveness of the proposed methods. Part of this work was published in [110] and [106].

Robotic Motion Re-planning by Analogy Learning

Motion planning is an important primitive for robotic applications. Many current motion planners regard each scenario as an isolated and independent problem, and solve it using the similar amount of computation power and time. This is plausible for unstructured and dynamic environment. However, for industrial applications, usually the environment is highly structured and the scenarios that the robot encounters are quite similar. It will be beneficial if we can transfer the planning solution from one scenario to other similar scenarios, so as to save computation and achieve consistent motion patterns. With this objective, a motion re-planning algorithm is proposed in Chapter 7 based on the concept of analogy learning. A mapping function between one scenario to another can be constructed by non-rigid registration. Then the previously planned motion can be transformed by the mapping function, and serve as an initial reference for motion planning of the new scenario. The proposed motion re-planning approach can be applied in configuration space, configuration-time space and tangent configuration space, so that it can perform various kinds of motion planning including path planning, trajectory planning and shape-conservative manipulation planning. A series of simulations on a 6-DOF manipulator are evaluated, which show the effectiveness of the proposed methods. Part of this work was published in [109] and [107].

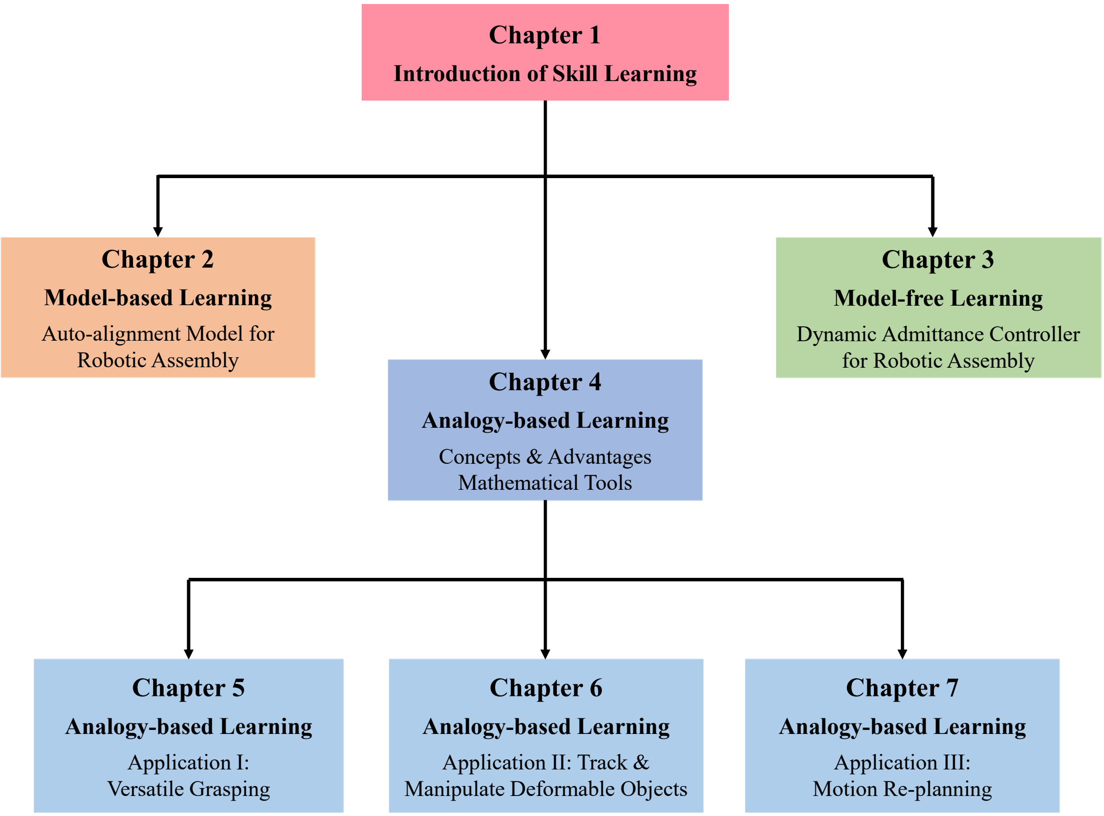


Figure 1.5: The structure overview of the dissertation.

Chapter 2

Robotic Assembly by Model-based Learning

As discussed in Chapter 1, some robotic skills can be constructed through the model-based approach, i.e., representing and predicting a system using physical principles and mathematical derivations. This chapter will show a novel auto-alignment skill learned by modelling to accelerate the robotic assembly process. Specifically, a contact model is constructed to estimate the tilt angles between assembly parts from force measurement. The major skill is modelled offline based on force and geometric analysis, while some model parameters, such as friction coefficient will be identified online through data measurement. Experiments on FANUC industrial manipulators show that with the learned skill, robots can perform assembly robustly starting from large misalignment status, which saves extensive efforts on installing positioning fixtures and manual alignment.

2.1 Introduction

Peg-hole-insertion, i.e., inserting a round peg into a round hole, is one of the most common tasks in industrial robotic assembly. In the past several years, many methods have been developed for robotic peg-hole-insertion. There are passive methods, which install passive components on end-effectors to provide compliance (remote center compliance [116]), as well as active methods, which actively adjust robotic motions to behave compliantly (impedance control [115], admittance control [24], hybrid force/velocity control [94]). The general objective is to make the robot system compliant to the environment, i.e., minimizing contact force during assembly, by modifying the nominal trajectory on-line.

Fig. 2.1 shows four possible alignment status before insertion begins. From left to right, they are 'line contact', 'one-point contact', 'two-point contact' and 'three-point contact', respectively. Many robotic peg-hole-insertion methods focus on the insertion phase and assume that the peg and hole are already well aligned. Those methods work well only if the initial configuration is one of the first three contact conditions [16]. Insertion with a three-

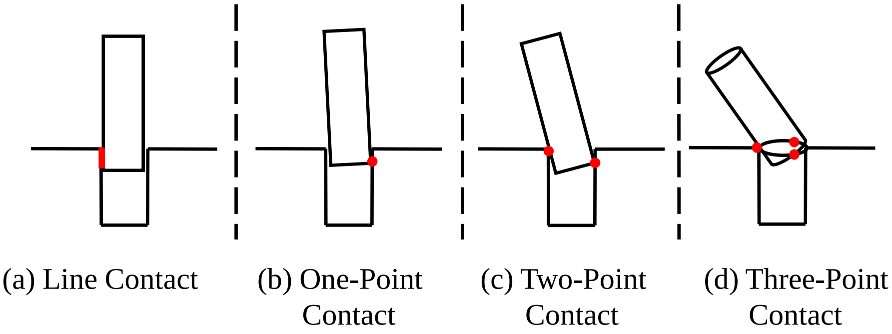


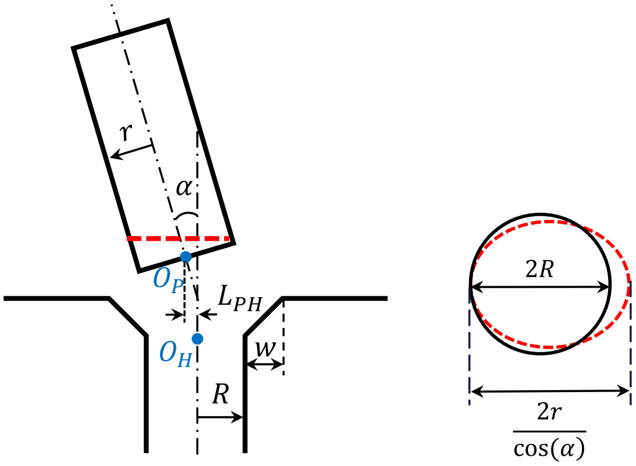
Figure 2.1: Four alignment conditions between peg and hole before insertion.

point contact is quite difficult because three degrees of freedom of the peg with respect to the hole are constrained. Therefore, there is little space left for the robot control algorithm to compliantly insert the peg into the hole. Note that in Fig. 2.1(d), the tilt angle is exaggerated for the ease of illustration. In practice, the three-point contact may appear more frequently compared to the other three conditions, especially in high precision industrial tasks where even a small tilt angle can result in a three-point contact.

If the assembly environment is well structured, i.e., the poses of the peg and hole are known precisely in advance, then it is not difficult to generate a suitable trajectory to align the two assembly parts accurately. However, it is usually required to install positioning fixtures to structure the environment, which is expensive and time-consuming. Moreover, in many cases, the environment cannot be well structured. Especially in the human-robot-collaboration (HRC) scenario [105], the human operator guides the robot arm via lead through teaching to align the peg coarsely towards the hole. Since human motion is inaccurate, it is highly possible that insertion has to start with the three-point contact condition. Therefore, an autonomous alignment skill is required to construct to assist assembly.

There are several works that address the peg hole alignment problem in different ways. In [44] and [45], the pose of the hole was estimated by vision feedback and the autonomous alignment was realized by visual servoing. However, these vision-based methods require additional vision sensors, which increase the system cost and complexity. Moreover, the hole's orientation is estimated by the normal vector of the surface around the hole. This involves making an implicit assumption that the hole is perpendicular to its surrounding surface, which is not always the case [3]. [21] and [42] performed the peg hole alignment by force control. They first adjusted the peg's orientation by pressing the peg's free end flush against the hole surface and then randomly slid the peg on the surface to eliminate position misalignment. However, the sliding process might leave scratches on the workpiece surface. Moreover, this method also requires the hole to be perpendicular to its surface and random search might take a long cycle time.

To deal with these problems, a new peg hole alignment method based on geometric and force analysis is proposed in this chapter. A physical model is constructed to relate the



(a) Cross section (b) Top view

Figure 2.2: Schematic diagram of peg hole insertion.

tilt angle and the external stress. The model parameters, especially the friction coefficient will be identified by regression methods. With the estimated angle values, a compensation movement can be generated to adjust the pose of the peg in order to eliminate the peg-hole misalignment. This auto-alignment skill does not require additional sensors (e.g., cameras), sliding on the surface, and the hole to be perpendicular to its surrounding surface. Therefore, the proposed method is non-costly, efficient and robust.

The remainder of this chapter is organized as follows: Section 2.2 discusses the three-point contact in details and the requirements to avoid three-point contact. Section 2.3 introduces the geometric and force analysis, as well as a three-point contact model, from which the closed-form equations for the relative pose between the assembly parts can be derived. An optimization formulation is also introduced to regress the friction coefficient. Section 2.4 presents a series of experiments performed on FANUC LR Mate 200*i*D/7L [32] to show the effectiveness of the auto-alignment skill. The robot demonstrated peg-hole-insertion from three-point contact conditions with 96% success rate. The supplementary videos can be found at [104]. Section 2.5 concludes this method and proposes future work.

2.2 Analysis of Initial Contact Conditions

Fig. 2.2(a) shows the cross section of the peg and hole, where R is the hole radius, r is the peg radius, and α is the tilt angle between the peg and hole's center axes. Denote the center point on the peg's end face as O_P, and the center point of the hole as O_H. If the hole is chamfered with width w, then O_H is not on the surface of the hole, but below the surface. The lateral distance between O_P and O_H along the hole's radius direction is denoted as L_{PH}.

The misalignment between the peg and hole is defined by the position misalignment

(L_{PH}) and the orientation misalignment (α). The peg would contact the chamfer instead of the surrounding surface if the following equation is satisfied,

$$L_{PH} < R - r\cos(\alpha) + w \tag{2.1}$$

With the compliance that the robot is provided (either by passive methods or active methods), if the peg contacts the chamfer first, it could easily slide along the chamfer and finally make contact with the hole. Equation (2.1) highlights the importance of chamfer. $R - r\cos(\alpha)$ on the right-hand side is typically a small number compared to chamfer width w. Therefore, the additive w term relaxes the difficulty in position alignment for assembly.

Suppose the peg has slid along the chamfer and finally contacts the hole, then the orientation misalignment (α) will determine which initial contact condition (see Fig. 2.1) will occur. As shown in Fig. 2.2, the peg's cross section is projected to the hole's surface plane (marked with the red dashed line). The length of the projected ellipse's major axis is $2r/\cos(\alpha)$. Three-point contact occurs if and only if

$$\frac{2r}{\cos(\alpha)} > 2R \tag{2.2}$$

Equation (2.2) can be rewritten as

$$\alpha > \cos^{-1}(\frac{r}{R}) \tag{2.3}$$

which indicates that if the tilt angle is smaller than a threshold, the three-point contact can be avoided. The smaller the clearance ($R - r$) between peg and hole, the lower the value of the threshold $\cos^{-1}(r/R)$.

The following sections will introduce models to compute the estimated lateral distance $\widehat{L}_{PH}$ and estimated tilt angle $\widehat{\alpha}$ from contact forces and torques. From (2.1) and (2.3), if the estimation errors are bounded by

$$\|\widehat{L}_{PH} - L_{PH}\| < R - r\cos(\alpha) + w \approx w \tag{2.4}$$

$$\|\widehat{\alpha} - \alpha\| < \cos^{-1}(\frac{r}{R}) \tag{2.5}$$

then the compensation motion based on $\widehat{L}_{PH}$ and $\widehat{\alpha}$ can successfully get assembly out of the three-point contact state. Experimental results in Section 2.4 will show that the estimation errors by the proposed method satisfy the above bounds.

2.3 The Model for Misalignment Estimation

Force Analysis

Assume that the peg and hole are in a three-point contact condition. As shown in Fig. 2.3, the Cartesian coordinate $x^P - y^P - z^P$ is attached to the peg center point O_P, where z^P is

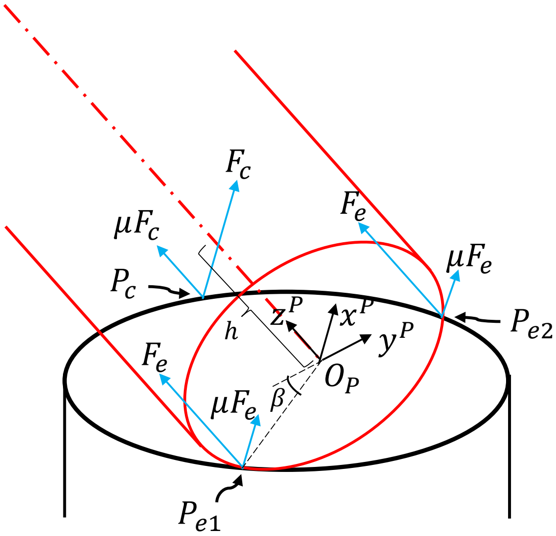


Figure 2.3: Force analysis of the three-point contact

along the peg axis, and x^P is selected such that the hole is symmetric about the $x^P - z^P$ plane. Among the three contact points, denote the one on the peg's cylindrical surface by P_c, and the other two symmetric points on the end face by P_{e1} and P_{e2}. On contact point P_c, the normal force applied to the peg is F_c along x^P axis, and the friction force is μF_c along z^P axis, where μ is the friction coefficient. On contact points P_{e1} and P_{e2}, the normal force is F_e along z^P axis, and the friction force is μF_e along x^P axis [41].

The forces on the three contact points generate resultant forces F_x, F_z and torque M_y on the center point O_P, with

$$F_x = F_c + 2\mu F_e \tag{2.6}$$

$$F_z = \mu F_c + 2F_e \tag{2.7}$$

$$M_y = F_c h + \mu F_c \cdot r + 2F_e \cdot r\sin(\beta) \tag{2.8}$$

where h denotes the distance between P_c and O_P along z^P, and β denotes the angle between $\overrightarrow{O_P P_{e1}}$ and y^P.

F_x, F_z and M_y can be measured by the force/torque sensor equipped at the robot end-effector. If μ is known, then F_e and F_c become

$$F_e = \frac{F_z - \mu F_x}{2(1-\mu^2)} \tag{2.9}$$

$$F_c = \frac{F_x - \mu F_z}{1-\mu^2} \tag{2.10}$$

Substitute (2.9) and (2.10) into (2.8) to get

$$\mu^2(rF_z - M_y) + \mu\{hF_z + [\sin(\beta) - 1]rF_x\} + [M_y - hF_x - \sin(\beta)rF_z] = 0 \tag{2.11}$$

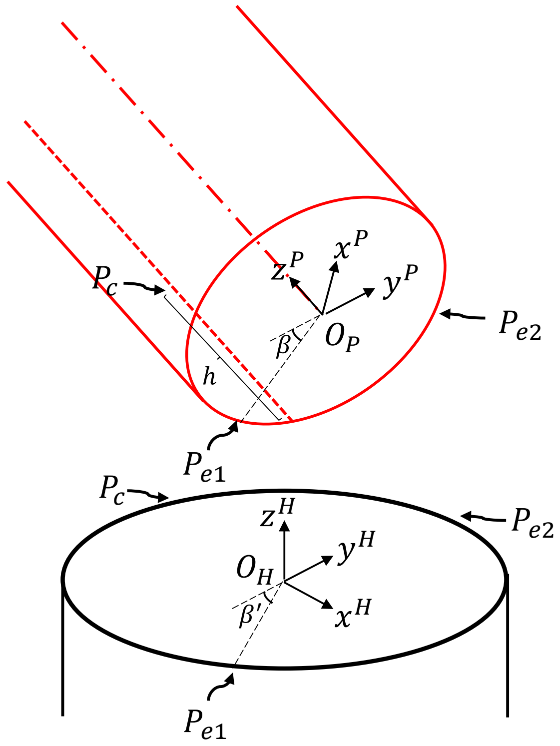


Figure 2.4: Geometric analysis of the three-point contact

Equation (2.11) can be furthur simplified to

$$M_y - hF_x - \sin(\beta) r F_z = 0 \tag{2.12}$$

if $\mu \approx 0$.

Equations (2.11) and (2.12) indicate that h and β form an equality constraint with the force/torque measurement values. The following geometric analysis will show that h and β are both functions of tilt angle α, which leads to the closed-form expression of α with respect to force and torque measurement.

Geometric Analysis

Similar to the above force analysis, another Cartesian coordinate $x^H - y^H - z^H$ is set up and attached to the hole center point O_H, with z^H along the hole axis and y^H parallel to the y^P. Note that the peg and hole in Fig. 2.4 are supposed to be in contact, but the peg in the figure is lifted virtually for the ease of coordinate illustration.

The transformation matrix from the hole coordinates to the peg coordinates is

$$T_P^H = \begin{bmatrix} \cos(\alpha) & 0 & -\sin(\alpha) & \Delta O_x^H \\ 0 & 1 & 0 & \Delta O_y^H \\ \sin(\alpha) & 0 & \cos(\alpha) & \Delta O_z^H \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.13}$$

where $[\Delta O_x^H, \Delta O_y^H, \Delta O_z^H]^T$ is the position of O_P in the hole coordinates.

The three contact points P_c, P_{e1} and P_{e2} can be described by the two coordinate frames [16]:

$$P_c^P = \begin{bmatrix} -r & 0 & h & 1 \end{bmatrix}^T \tag{2.14}$$

$$P_{e1}^P = \begin{bmatrix} -r\sin(\beta) & -r\cos(\beta) & 0 & 1 \end{bmatrix}^T \tag{2.15}$$

$$P_{e2}^P = \begin{bmatrix} -r\sin(\beta) & r\cos(\beta) & 0 & 1 \end{bmatrix}^T \tag{2.16}$$

$$P_c^H = \begin{bmatrix} -R & 0 & 0 & 1 \end{bmatrix}^T \tag{2.17}$$

$$P_{e1}^H = \begin{bmatrix} R\sin(\beta') & -R\cos(\beta') & 0 & 1 \end{bmatrix}^T \tag{2.18}$$

$$P_{e2}^H = \begin{bmatrix} R\sin(\beta') & R\cos(\beta') & 0 & 1 \end{bmatrix}^T \tag{2.19}$$

The above six vectors should satisfy the following transformations,

$$P_c^H = T_P^H \cdot P_c^P \tag{2.20}$$

$$P_{e1}^H = T_P^H \cdot P_{e1}^P \tag{2.21}$$

$$P_{e2}^H = T_P^H \cdot P_{e2}^P \tag{2.22}$$

Substitute (2.14)-(2.19) into (2.20)-(2.22) to get the expressions of h and β with respect to α,

$$h = \frac{2R - 2r\cos(\alpha)}{\sin(\alpha)} \tag{2.23}$$

$$\beta = \sin^{-1}\left(\frac{r\cos^2(\alpha) - 2R\cos(\alpha) + r}{r\sin^2(\alpha)}\right) \tag{2.24}$$

Furthermore, the position offset between the two origins O_P and O_H can be derived in the hole's coordinate frame,

$$\Delta O_x^H = R - r\cos(\alpha) \tag{2.25}$$

$$\Delta O_y^H = 0 \tag{2.26}$$

$$\Delta O_z^H = \frac{r\cos^2(\alpha) - 2R\cos(\alpha) + r}{\sin(\alpha)} \tag{2.27}$$

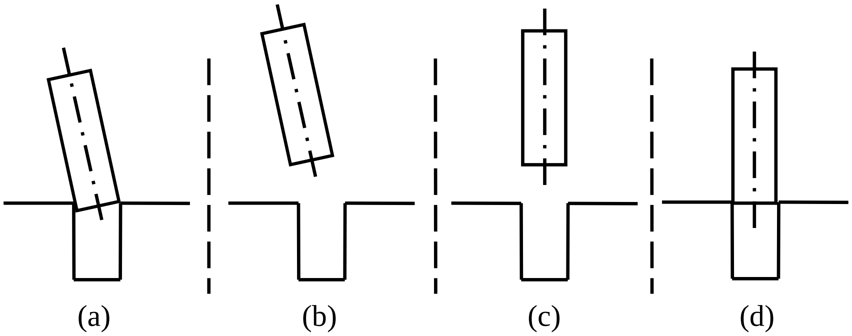


Figure 2.5: Trajectory to compensate the misalignment between peg and hole.

Autonomous Alignment by Compensation

Substituting (2.23) and (2.24) into (2.11), we arrive at an equation with α as the only unknown variable, while other variables come from force/torque sensor measurement.

$$\begin{aligned}&(F_z-\mu F_x)\left[r\cos^2(\alpha)-2R\cos(\alpha)+r\right]+2(F_x-\mu F_z)\sin(\alpha)(R-r\cos(\alpha))\\&=\left[(rF_z-M_y)\mu^2-rF_x\mu+M_y\right]\sin^2(\alpha)\end{aligned} \tag{2.28}$$

Assuming $R\approx r$, the following simplified closed-form expression for α is obtained,

$$\alpha=2\tan^{-1}\left(-\frac{F_x-\mu F_z}{F_z-\mu F_x}+\frac{\sqrt{r^2(F_x-\mu F_z)^2+r(F_z-\mu F_x)[(rF_z-M_y)\mu^2-rF_x\mu+M_y]}}{r(F_z-\mu F_x)}\right) \tag{2.29}$$

which describes the **orientation misalignment** between the peg and hole in the three-point contact condition.

Since it is easier to program the motion of robot end-effector in its tool frame, (2.25)-(2.27) are transformed to obtain the position of O_H in the peg coordinates,

$$\Delta O_x^P=R\cos(\alpha)-r \tag{2.30}$$

$$\Delta O_y^P=0 \tag{2.31}$$

$$\Delta O_z^P=\frac{R\cos^2(\alpha)-2r\cos(\alpha)+R}{\sin(\alpha)} \tag{2.32}$$

which represent the **position misalignment** between the peg and hole in the three-point contact condition.

With this misalignment estimation model, a compensation motion can be generated to eliminate the misalignment. Fig. 2.5 shows one possible motion pattern that was designed in this work. Starting from the three-point contact condition, the robot first backs the peg back along the peg axis (Fig. 2.5(b)), then adjust the peg's pose such that it is accurately aligned

Table 2.1: Dimension of the peg-hole insertion testbed

	inch (UI)	mm (SI)	Tolerance
Hole	$1.000^{+0.0001}_{-0.0000}$	$25.400^{+0.003}_{-0.000}$	H7
Peg	$0.999^{+0.0000}_{-0.0002}$	$25.370^{+0.001}_{-0.000}$	h7

above the hole (Fig. 2.5(c)). This adjustment takes place in the air in order to avoid collision. Finally, the robot feeds the peg straight down towards the hole until contact (Fig. 2.5(d)). With this compensation, the three-point contact would be effectively eliminated and the peg is aligned to the hole with line contact, one-point contact or two-point contact conditions (Fig. 2.1). Afterwards, those traditional force control methods can be utilized for performing insertion process.

Estimation of Friction Coefficient

Estimating the tilt angle α by (2.29) requires the friction coefficient μ between the surface of the peg and hole. The nominal value for friction coefficient between two materials can be easily achieved from the reference. However, in practice, the friction coefficient might deviate from the nominal value, influenced by the object shapes, tilt angles, and chamfers on top of the hole. This subsection introduces a data-driven method to estimate the friction coefficient if unknown.

From (2.29), the tilt angle is estimated by a nonlinear function

$$\widehat{\alpha} = f(\mu, m) \tag{2.33}$$

with μ and m as variables, where m are the measurement values from F/T sensor. At training stage, we can record several sets of true values by manual methods. By comparing the true α and the estimated $\widehat{\alpha}$, the friction coefficient can be regressed by solving the following optimization problem

$$\begin{aligned} \mu^* = \arg\min_{\mu} \quad & \sum_{i=1}^{n} \|\alpha_i - \widehat{\alpha}_i\|^2 \\ s.t. \quad & \widehat{\alpha}_i = f(\mu, m_i) \end{aligned} \tag{2.34}$$

At test stage, since the friction coefficient μ is known, the tilt angle can be estimated without manual measurement any more.

Note that since the constraint in (2.34) is non-convex, there is no guarantee for a global minimum solution. However, the nominal friction value from the reference can be utilized as the initialization for the optimization, by which way we can get an acceptable local minimum of μ.

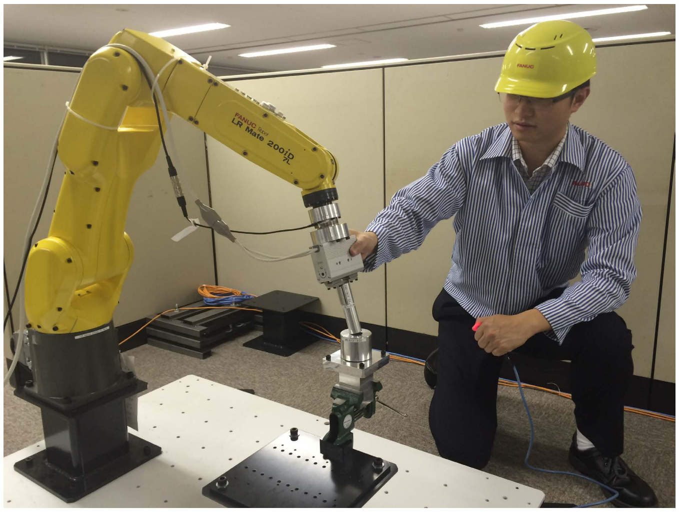


Figure 2.6: Testbed for peg hole alignment. The diameters of peg and hole are 25.370mm and 25.400mm respectively, with 0.030mm clearance (H7h7 tolerance). The model of industrial robot is FANUC LRMate-200iD/7L.

2.4 Experimental Study

To demonstrate the performance of the proposed alignment model, a series of experiments were performed on an industrial robot FANUC LR Mate 200iD/7L [32]. Experimental video can be found at [104].

The testbed is shown in Fig. 2.6. The peg and hole were both machined from Aluminium 6061-T6, with a peg diameter of 0.999in (25.370mm), a hole diameter of 1.000in (25.400mm) and 1.0mm chamfers. The assembly tolerance is industrial standard H7h7 (see Table 2.1). An ATI Mini45 F/T sensor [5] is embedded in the robot end-effector which can measure the force and torque during assembly.

The first part of the experiment is designed to estimate the friction coefficient μ. The hole is fixed with a known pose as well as having its axis aligned with the vertical line. The peg is grasped and pushed towards the hole with a three-point contact configuration by the robot. The pushing force is around 10N along the $-z^P$ direction and 30N along the $-x^P$ direction. The peg is tilted at four known angles, $10.36°, 15.75°, 20.38°, 26.47°$, and the corresponding force/torque measurements are recorded at these four configurations. The friction coefficient μ is estimated by performing the nonconvex optimization (2.34) with initial reference value $\mu^0 = 0.4$, which is a nominal friction value for aluminium [114]. The optimal solution is $\mu^* = 0.289$. As shown in Fig. 2.7, the estimated tilt angles (2.29) using μ^* are $10.86°, 15.02°, 20.63°, 26.28°$ respectively, which are all close to the real tilt angles.

After μ is estimated, the second part of the experiment is performed to estimate the pose misalignment between the peg and hole. During the experiment, the hole part is manually fixed on a table vice with random poses. The robot has no prior information on the pose of the hole. By lead through teaching mode, the human operator guides the robot to move

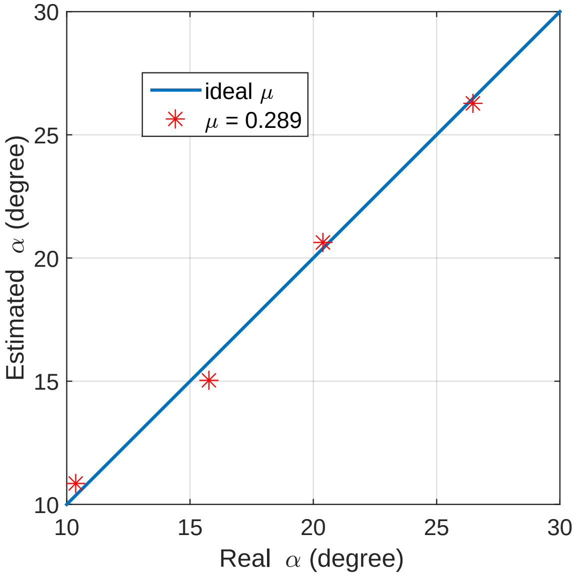


Figure 2.7: Estimated tilt angles of four calibration measurements.

until the grasped peg contacts the hole. The robot then switches to force control mode, and autonomously applies a constant force to push the peg to maintain close contact. The commanded pushing force is 10N along the $-z^P$ direction and 30N along the $-x^P$ direction. Actual forces/torques are measured and then the tilt angle (α) as well as the hole center position (O_H relative to O_P) are estimated respectively by the constructed model (2.29)-(2.32).

As discussed at the end of Section 2.2, there will always be errors in the misalignment estimations. However, if the estimation errors are bounded by (2.4) and (2.5), it can be guaranteed that the three-point contact status will be avoided after compensation. For this specific testbed, the error bound for orientation estimation is $\cos^{-1}(25.370/25.400) = 2.785°$, and the error bound for position estimation is $w = 1mm$.

Fig. 2.8 shows the orientation estimation results. The horizontal axis denotes the actual tilt angle, and the vertical axis represents the estimated tilt angle. In twenty-five experiments, most of the orientation estimations are inside the error bounds ($\pm 2.785°$), except for one case which has a deviation of $-5.1°$. The average of absolute estimation errors is $1.248°$. One interesting observation is that these estimations are not evenly distributed around the ideal estimation line (blue line in Fig. 2.8). Estimations tend to be larger when $\alpha \in [10°, 20°]$ and to be smaller when $\alpha \in [20°, 30°]$. This trend can possibly be attributed to the fact that our estimation equation for orientation (2.29) utilizes a constant friction coefficient μ. In practice, however, the friction coefficient might vary if the tilt angle between peg and hole changes.

Fig. 2.9 shows the estimation results of the position misalignment. The horizontal axis

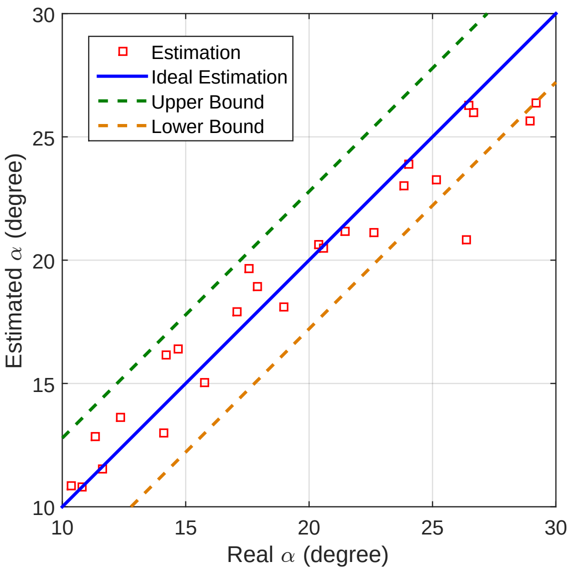


Figure 2.8: Estimated tilt angles from force/torque measurements.

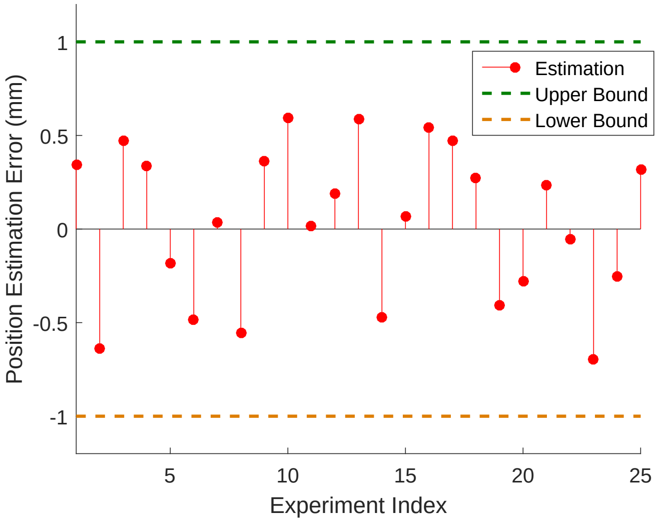


Figure 2.9: Estimated hole's position from force/torque measurements.

denotes the experimental index from 1 to 25. The vertical axis represents the position estimation error along the x^H direction in each experiment. The average of absolute estimation errors is $0.3546mm$. All of the position estimations are within the error bounds ($\pm 1mm$).

After the pose misalignment is estimated, a compensation trajectory is generated and performed by the industrial manipulator to eliminate the misalignment accurately (see Fig. 2.10(a)-(c)). It first pulls the peg back and then aligns the peg towards the hole in

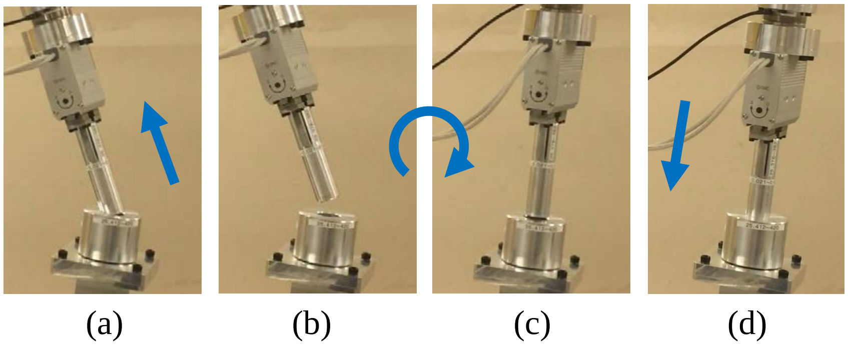


Figure 2.10: Autonomous alignment procedure for peg hole insertion.

the free space. In Fig. 2.10(d), since the three-point contact has been ruled out, traditional hybrid force/velocity control is utilized to insert the peg into the hole. In the twenty-five experiments, robot successfully inserts the peg into the hole twenty-four times (96% success rate). The unsuccessful experiment failed because the residue tilt angle between peg and hole was larger than the error bound (see Fig. 2.8).

To conclude, the experimental results show that the proposed alignment method can accurately estimate the pose misalignment between peg and hole. The misalignment can be effectively compensated by the designed compensation trajectory. After compensation, traditional assembly methods can be applied to insert the peg into the hole. This alignment method has many promising applications. For example, in the future assembly line, it is unnecessary for the operator to measure the hole's pose or program the robot an approaching trajectory. The human worker can roughly lead the robot to put assembly parts into contact with each other, then the robot can autonomously correct the misalignment and perform assembly.

2.5 Chapter Summary

The process of industrial robotic assembly easily suffers from a three-point contact status, where the peg has a large misalignment towards the hole, and the following insertion procedure cannot proceed. To deal with this problem, a model-based auto-alignment skill is developed for industrial manipulators to assist autonomous assembly. The model for peg-hole misalignment is derived based on force and geometric analysis. The model parameter (friction coefficient) is regressed by nonconvex optimization. With the estimated misalignment values, a compensation trajectory is designed to align the peg and hole to avoid three-point contact. A series of experiments are performed to evaluate the proposed method. Experimental results show that the misalignment between the peg and hole with H7h7 tolerance can be effectively eliminated and the robot can conduct peg-hole-insertion from three-point contact status with 96% success rate.

Chapter 3

Robotic Assembly by Model-free Learning

Model-based learning, as shown in Chapter 2, can construct new skills for robot manipulators by analytical system modelling. Since the model captures the physical principle in an intrinsic and succinct way, the whole approach is usually computationally efficient and operationally stable. However, in some complicated scenarios, constructing a solid model is non-trivial since the system's complexity is beyond human's cognitive capacity, or the in-hand model is insufficient to provide a satisfying system approximation. Under this background, model-free approaches are proposed which try to approximate the system or control policy by a high dimensional function, and the function is parametrized by learning from the training dataset. This chapter will introduce a model-free approach to teach a compliant motion skill to industrial robots. A dynamic admittance controller will be learned by Gaussian mixture regression from human demonstration. The highlight of this work is that, unlike the black-box behaviors in traditional approaches, this model-free method is transparent and interpretable. The stability of the closed-loop system can also be analyzed and proved.

3.1 Introduction

As shown in Fig. 2.5, with the peg and hole well aligned to each other, the insertion phase can be performed by industrial robots to insert the peg into the hole with force control. The objective of force control is to regulate both contact force and torque so as to make the robot compliant with respect to the surrounding environment.

In this chapter, we will use 'wrench' to denote both force and torque for description simplicity. Fig. 3.1 shows a classic structure for robotic force control [24]. The wrench error $\boldsymbol{w}_e \in \mathbb{R}^6$ generates a set-point for an inner velocity-control loop with admittance gain $K_A \in \mathbb{R}^{6\times 6}$. The system approaches to the steady state when the wrench is regulated to the desired value. Note that when the robot contacts the environment, the dynamics of the closed-loop system will change since the environment dynamics is involved. In practice,

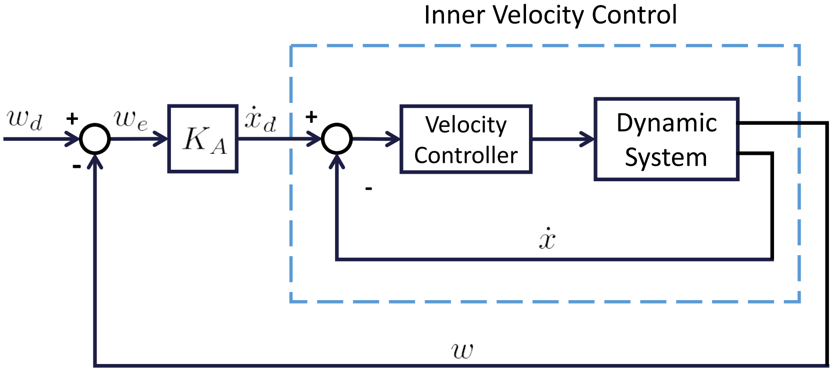
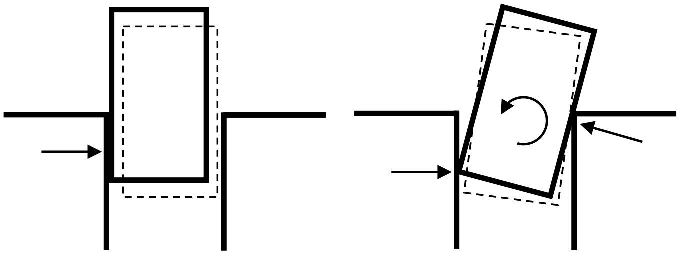


Figure 3.1: Force servo with inner velocity control loop

in order to achieve a good performance for robotic assembly, engineers have to tune the admittance gains case by case to adapt to different stiffness, damping or clearance of various assembly parts, the tuning procedure for which is non-intuitive, time-consuming and unsafe (damages to motor gearbox).

In contrast to the non-trivial tuning for the robot controller, humans can accomplish assembly tasks manually with much less time and fewer trials. If we regard the human's inherent assembly skill is due to internal control, the controller must have been well tuned through our daily operation and experience. Therefore, several researches have been conducted to try to teach robots the insertion skill from human demonstration [105, 62, 20, 55]. Instead of traditional model-based methods, the force controller is constructed by parametric functions and regressed by human demonstration data. In [105], the human assembly skill was modelled as a state-varying admittance and then utilized Gaussian mixture regression to predict the corrective velocity from wrench measurement. In [62], a neural network control policy was trained by trial and error and the robot learned to assemble a toy plane with unknown environment dynamics. In [20], the instructor demonstrated the manipulation task in a haptic rendered virtual environment using a haptic device and then used locally weighted projection regression to model the human corrective trajectory under jamming states.

These methods have shown to be effective on collaborative robots such as Baxter [8] and PR2 [93], with assembly tolerance H5h10 or larger. However, few of them are tested on industrial robots. Unlike collaborative robots, the traditional industrial robot is highly rigid because of large gear reduction ratio on the drivetrain. Basically, there is no compliance on the mechanism to assist assembly. Besides, the clearance requirement for industrial assembly is more strict. This chapter proposes a novel model-free framework for teaching industrial manipulators varying admittance from human demonstration. On the one hand, a remote demonstration tool is introduced in this work. The wrench data for peg-hole-insertion can be collected without physical contact between humans and robots, which guarantees the safety for data collection. On the other hand, we provide a clear physical interpretation of the proposed model-free structure, and its stability can be analyzed and proved by Lyapunov



(a) Position misalignment (b) Orientation misalignment

Figure 3.2: Cross section of peg-hole-insertion. (a) Position misalignment generates a contact force on the peg. Responding to this, the force controller generates a translational velocity command to avoid collision; (b) Orientation misalignment generates a contact torque on the peg. Responding to this, the force controller generates a rotational velocity command to adjust orientation.

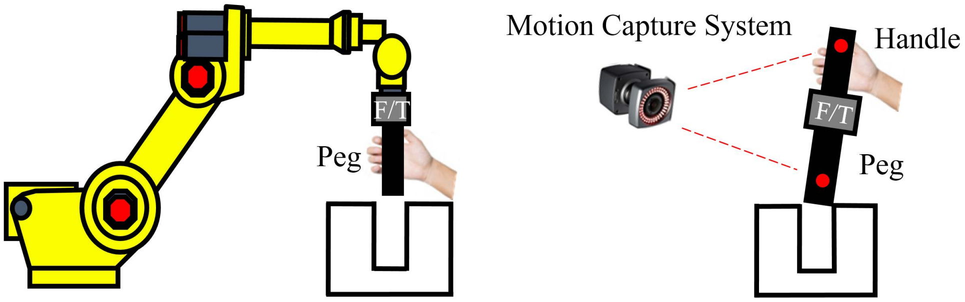
theorems.

This remainder of this chapter is organized as follows: Section 3.2 analyzes the insertion procedure for a typical peg-hole-insertion task. A demonstration tool is designed to acquire human demonstration data and the data processing procedures are introduced. Section 3.3 introduces a framework of learning dynamic admittance from human by Gaussian mixture regression (GMR). The physical interpretation of GMR is discussed and the closed-loop stability is analyzed. A series of experiments are performed on H7h7 peg-hole testbed with FANUC LR Mate 200*i*D/7L [32]. Experimental results are provided in Section 3.4, and experimental videos can be found in [104]. Section 3.5 is conclusion and discussion.

3.2 Data Acquisition and Processing

With the auto-alignment skill provided in Section 2, the major misalignment between the peg and hole can be compensated. However, a force controller is still needed to adjust the residue misalignment and fully insert the peg into the hole.

Fig. 3.2 shows the adjustment procedure of classic force controllers in the cross section. The Force/Torque (F/T) sensor on the end-effector detects the wrench applied on the peg. By implementing the control law, the wrench feedback generates a corresponding translational/rotational velocity command on the robot end-effector, so as to push the peg away from collision. The ratio of velocity over wrench is defined as admittance (the inverse of impedance). Like the robot force controller, human beings use the similar strategy during assembly tasks. Human brain perceives the hand's tactile feedback and then decides which direction to move in order to compliantly assemble the workpieces. However, instead of a constant and linear admittance gain, the human assembly skill is much more sophisticated. Humans would apply different admittance according to the material, tolerance of the work-



(a) Lead through teaching (b) Human demonstration device

Figure 3.3: Two different methods to acquire human demonstration data.(a) By lead through teaching. Contact wrench is detected by the end-effector F/T sensor, and velocity is calculated by robot forward kinematics; (b) By human demonstration device. Contact wrench is measured by the F/T sensor embedded in HDD, and velocity is measured by the motion capture system.

pieces, or even according to the different insertion phases. The core idea of this research is to learn a dynamic admittance from human demonstration, and then apply this trained admittance to robot force control.

Data Acquisition

The first step of learning from human is to acquire the human demonstration data. For example, how to measure the contact wrench that human feels during assembly? How to detect the corrective velocity that human applied on the peg? An intuitive idea is to utilize the lead through teaching mode (Fig. 3.3a): the human operator grasps and guides the manipulator to insert the peg into the hole. During assembly, the contact wrench is recorded by the F/T sensor on the end-effector, and the Cartesian space corrective velocity is calculated by the robot forward kinematics. In practice, however, we find that industrial robot's lead through teaching fails in this kind of contact tasks. Since there is usually only one F/T sensor on the industrial robot, it cannot distinguish the wrench applied by the human from that applied by the hole. Therefore the human operator loses control of the robot when it starts to contact the hole, and the following insertion procedure cannot be demonstrated.

To deal with this problem, a remote human demonstration device (HDD [66]) is proposed as shown in Fig. 3.3(b). Instead of using robot sensors to collect data, an independent handle is designed to collect the required wrench and velocity information. The HDD consists of three major components: from the bottom to the top, a round peg, a F/T sensor and a handle bar. There are also several vision markers attached for velocity tracking. During human demonstration, the human operator grasps the handle bar and performs peg-hole-insertion several trials. The contact wrench and corrective velocity are recorded by the F/T

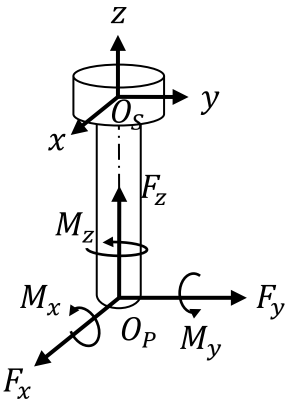


Figure 3.4: Wrench applied on the tool center point O_P, while the measurement value is relative to F/T sensor origin O_S.

sensor and the motion capture system simultaneously. Section 3.4 will introduce the design of HDD in details.

Compared to lead through teaching, the human operator grasps HDD instead of the robot arm to demonstrate the assembly tasks, which is more natural and intuitive. Moreover, the robot and human are separated during data acquisition, which ensures the safety of the human operator.

Data Processing

As shown in Fig. 3.4, a Cartesian coordinate is built up with the origin O_S at the center of the F/T sensor. O_P is the tool center point (TCP) at the end of the peg. The peg's velocity is described by $\dot{\boldsymbol{x}} = [v_x, v_y, v_z, \omega_x, \omega_y, \omega_z]^T \in \mathbb{R}^6$. The wrench data collected by F/T sensor is $\boldsymbol{w}^S = [F_x^S, F_y^S, F_z^S, M_x^S, M_y^S, M_z^S]^T \in \mathbb{R}^6$.

The collected data set $\dot{\boldsymbol{x}}$ and $\boldsymbol{w}^S$ cannot be directly utilized to train the dynamic admittance. Several data processing steps are required. During peg-hole-insertion, the contact wrench applies at O_P, but is measured at F/T sensor origin O_S. The measurement value is dependent on the offset between O_S and O_P. A general assembly skill should not be influenced by this specific offset value. Therefore $\boldsymbol{w}^S$ should be transformed with respect to O_P so as to eliminate the influence. Denote the offset as $[x_{sp}, y_{sp}, z_{sp}] \in \mathbb{R}^3$ in the sensor coordinate. The transformed wrench $\boldsymbol{w}^{TCP} = [F_x, F_y, F_z, M_x, M_y, M_z]^T \in \mathbb{R}^6$ at the TCP point can be calculated by:

$$\boldsymbol{w}^{TCP} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & +z_{sp} & -y_{sp} & 1 & 0 & 0 \\ -z_{sp} & 0 & +x_{sp} & 0 & 1 & 0 \\ +y_{sp} & -x_{sp} & 0 & 0 & 0 & 1 \end{bmatrix} \cdot \boldsymbol{w}^{S} \tag{3.1}$$

Besides the conventional transformation, if the human operator demonstrates insertions multiple times but with varying speeds, this velocity inconsistency might influence the quality of admittance training. In this situation, the dynamic time warping [97] from speech recognition area can be utilized to synchronize datasets. The process of synchronization is detailed in our previous work in [105].

3.3 Learn Dynamic Admittance by Gaussian Mixture Regression

Admittance describes the relation between wrench and velocity. For a force controller, the admittance gain determines how large the corrective velocity should be generated according to the measured contact wrench. As shown in Section 3.2, the wrench and velocity information are recorded simultaneously during human demonstration. In this section, we will train the admittance block such that given the same wrench input, a similar corrective velocity command will be generated as human does. This is a typical nonlinear regression problem in Machine Learning or Statistics. Since there are many nonlinear parametric functions developed, we set up three criteria to select the most appropriate one:

(1) Stability. Since this dynamic admittance block will be embedded in the feedback control loop, it will influence the stability of the whole system. The stability conditions of the nonlinear regressor should be explicitly formulated, so that the system stability can be analyzed.

(2) Efficiency. The computation power of industrial robot controller is limited due to cost concerns. It is preferred that the nonlinear regressor has a closed-form expression so that it could be implemented efficiently online.

(3) Interpretation. Many model-free approaches, such as neural networks, logistic regression, K-nearest neighbours, could generate proper output by learning from training data, but cannot provide transparent explanations on their internal dynamics. It is preferred that the nonlinear regressor in this work has a physical interpretation which explains why it can serve as a dynamic admittance.

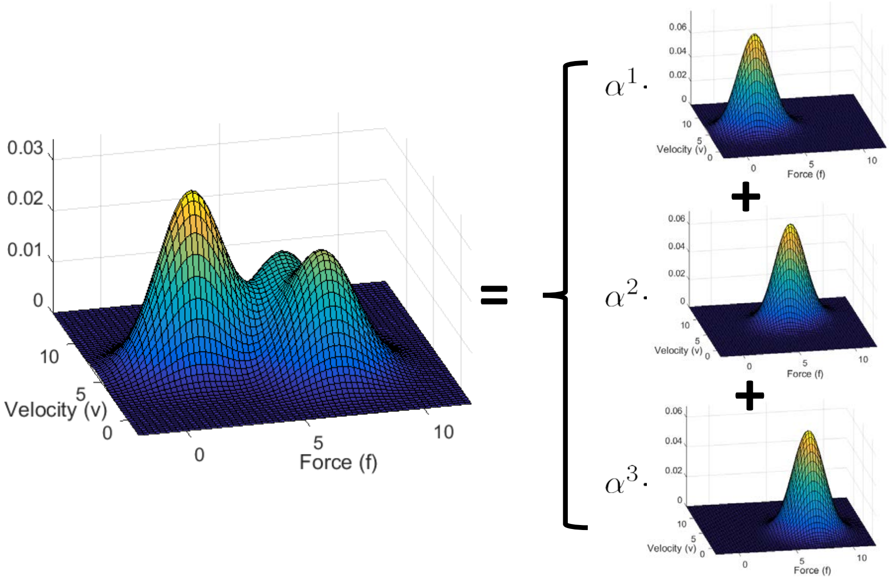


Figure 3.5: Estimate data distribution by Gaussian. For the ease of visualization, both velocity and wrench are one dimensional here. The distribution density of the velocity and wrench data collected from human demonstration is shown in the left. This density map is then fitted by superposition of several weighted Gaussian with different means and covariance.

To meet the three criteria, Gaussian mixture regression (GMR [11]) is introduced in this work to build the dynamic admittance block. It will be shown that GMR has explicit stability conditions, good efficiency and reasonable physical interpretations.

Introduction to GMR

The basic idea of GMR is to augment the human demonstration data (sensed wrench w and corrective velocity $\dot{x}_c$) into a high dimensional joint space. The joint probability distribution $p(w, \dot{x}_c)$ can be regressed from the demonstration data. Afterwards, the conditional probability $p(\dot{x}_c|w)$ can be constructed so that the suitable output $\dot{x}_c$ can be retrieved given any input w.

As shown in Fig. 3.5, the first step of GMR is to fit the joint probability distribution $p(w, \dot{x}_c)$ by the weighted sum of N Gaussian mixtures, each Gaussian with mean μ_i, covariance Σ_i and weight α_i

$$\begin{aligned} p(w, \dot{x}_c) &= \sum_{i=1}^{N} \alpha^i p^i(w, \dot{x}_c) \\ &= \sum_{i=1}^{N} \alpha^i \mathcal{N}(\begin{bmatrix} w \\ \dot{x}_c \end{bmatrix} \mid \begin{bmatrix} \mu_w^i \\ \mu_{\dot{x}_c}^i \end{bmatrix}, \begin{bmatrix} \Sigma_w^i & \Sigma_{w\dot{x}_c}^i \\ \Sigma_{\dot{x}_c w}^i & \Sigma_{\dot{x}_c}^i \end{bmatrix}) \end{aligned} \tag{3.2}$$

$$\text{with } \sum_{i=1}^{N} \alpha^i = 1 \tag{3.3}$$

Since each component $p^i(w, \dot{x}_c)$ is Gaussian, its conditional probability distribution $p^i(\dot{x}_c|w)$ is still Gaussian:

$$p^i(\dot{x}_c|w) = \mathcal{N}(\dot{x}_c|\mu_{\dot{x}_c|w}^i, \Sigma_{\dot{x}_c|w}^i) \tag{3.4}$$

The overall conditional probability $p(\dot{x}_c|w)$ can be constructed as the sum of each $p^i(\dot{x}_c|w)$, with a weight which indicates the probability of w belonging to each Gaussian component

$$p(\dot{x}_c|w) = \sum_{i=1}^{N} \frac{\alpha^i \mathcal{N}(w|\mu_w^i, \Sigma_w^i)}{\sum_{j=1}^{N} \alpha^j \mathcal{N}(w|\mu_w^j, \Sigma_w^j)} p^i(\dot{x}_c|w) \tag{3.5}$$

The admittance block is designed such that given an input wrench w, it will generate an output corrective velocity $\dot{x}_c^*$ that maximizes $p(\dot{x}_c|w)$. $\dot{x}_c^*$ could be calculated from (3.6) directly. Since it is an explicit function, the computation load is low and this control policy can be implemented in real time.

$$\begin{aligned} \dot{x}_c^* &= \arg\max_{\dot{x}_c} p(\dot{x}_c|w) \\ &= \sum_{i=1}^{N} \frac{\alpha^i \mathcal{N}(w|\mu_w^i, \Sigma_w^i)}{\sum_{j=1}^{N} \alpha^j \mathcal{N}(w|\mu_w^j, \Sigma_w^j)} \mu_{\dot{x}_c|w}^i \\ &= \sum_{i=1}^{N} \frac{\alpha^i \mathcal{N}(w|\mu_w^i, \Sigma_w^i)}{\sum_{j=1}^{N} \alpha^j \mathcal{N}(w|\mu_w^j, \Sigma_w^j)} \left[\mu_{\dot{x}_c}^i + \Sigma_{\dot{x}_c w}^i (\Sigma_w^i)^{-1} (w - \mu_w^i)\right] \end{aligned} \tag{3.6}$$

In GMR, The Gaussian parameters $(\mu^i, \Sigma^i, \alpha^i)$ can be estimated iteratively by E-M algorithm [10] from the human demonstration data $(w, \dot{x}_c)$. Note that since the performance of parameter identification is sensitive to the initialization, the K-means clustering [40] is applied first on the dataset to get a good initialization. Specifically, $(w, \dot{x}_c)$ is clustered into N groups according to the Euclidean distance, and then each group is treated as an initial Gaussian component and the respective parameters $(\mu^i, \Sigma^i, \alpha^i)$ are calculated.

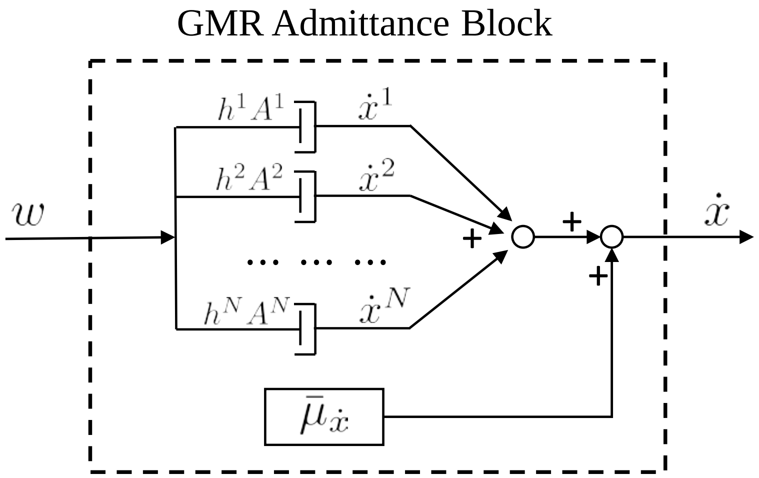


Figure 3.6: Interpretation of GMR in mechanics point of view. The admittance block formulated by GMR consist of N linear dampers with admittance A^i respectively. Each damper has a nonlinear weight $h^i(w)$ denoting its contribution to the whole block. The output of the admittance block is generated by the summation of the N dampers' outputs as well as a general preload velocity $\bar{\mu}_{\dot{x}}$.

Interpreting GMR with mechanics point of view

The probability form of GMR as (3.6) does not provide insight to the dynamic feature of the admittance system [50]. In this section, we rewrite (3.6) to formulate it like a dynamic system. To simplify the notation, we define:

$$h^i(w) = \frac{\alpha^i \mathcal{N}(w|\mu_w^i, \Sigma_w^i)}{\sum_{j=1}^{N} \alpha^j \mathcal{N}(w|\mu_w^j, \Sigma_w^j)} \tag{3.7}$$

$$A^i = \Sigma_{\dot{x}_c w}^i (\Sigma_w^i)^{-1} \tag{3.8}$$

where $h^i(w) \in [0,1]$ is a scalar function of w, and $A^i \in \mathbb{R}^{6\times 6}$ is a constant square matrix. Substituting (3.7) and (3.8) into (3.6), we have

$$\dot{x}_c = \sum_{i=1}^{N} h^i(w) \left[\mu_{\dot{x}_c}^i + A^i(w - \mu_w^i)\right] \tag{3.9}$$

Equation (3.9) has the following physical interpretation: the dynamic admittance block consists of the weighted sum of N linear dampers. Each damper has linear admittance A^i and a preloaded velocity $\mu_{\dot{x}_c}^i - A^i\mu_w^i$ (both are determined by human demonstration data). Equation (3.8) reveals the relation between the admittance A^i and the covariance of Gaussian distribution. The nonlinear weight $h^i(w)$ denotes the contribution of each damper to the

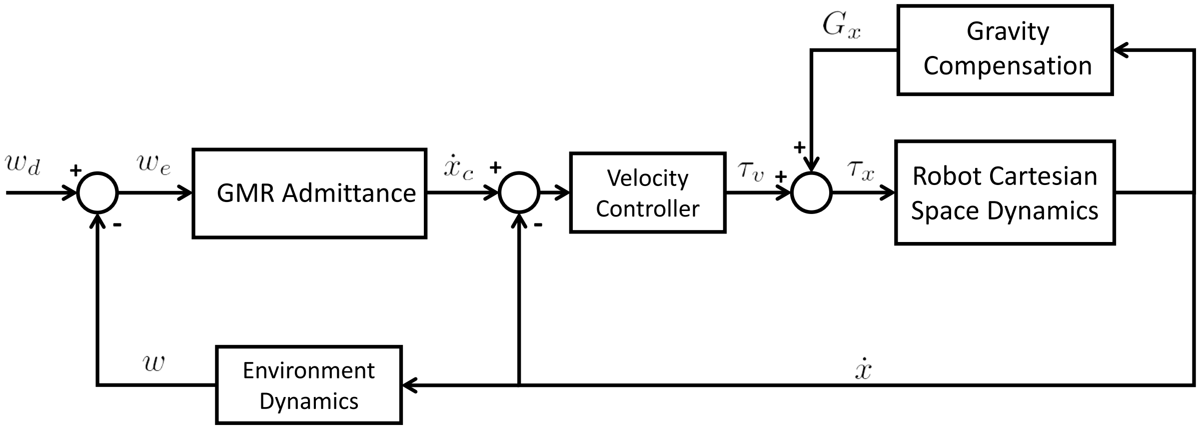


Figure 3.7: Force control block diagram with GMR admittance module.

whole block. Note that the weight is dynamic and dependent on the value of measured wrench during test. If combining multiple dampers into a single nonlinear one, then we can define

$$\bar{A} = \sum_{i=1}^{N} h^i(w)A^i \tag{3.10}$$

$$\bar{\mu}_{\dot{x}} = \sum_{i=1}^{N} h^i(w)(\mu^i_{\dot{x}_c} - A^i\mu^i_w) \tag{3.11}$$

and $\dot{x}_c$ can be described as

$$\dot{x}_c = \bar{A}w + \bar{\mu}_{\dot{x}} \tag{3.12}$$

where $\bar{A}$ is the general admittance of the block, and $\bar{\mu}_{\dot{x}}$ is the general preload corrective velocity.

To conclude, the structure of GMR has an inherent similarity with physical admittance systems. It can be interpreted as a nonlinear combination of multiple dampers with different linear admittances and nonlinear weights (see Fig. 3.6). This explicit interpretation of this model-free approach also provides insights to analyze the system's stability.

Stability condition of the closed-loop system

The learned dynamic admittance block is embedded in the robotic force controller as shown in Fig. 3.7. Since the feedback loop is modified, the system's closed-loop stability will be influenced. For industrial applications, it is a critical issue to guarantee the stability of the overall system. This subsection will analyze the stability conditions of the proposed learning method based on Lyapunov theories [103].

Theorem 1 *Consider a closed-loop robot control system (Fig. 3.7) consisting of a manipulator with classic dynamics, an inner-loop velocity controller* $\tau_v = K_P(\dot{x}_c - \dot{x}) - K_I x$*, a learned dynamic admittance block* $\dot{x}_c = \sum_{i=1}^{N} h^i(w_e)\left[\mu^i_{\dot{x}_c} + A^i(w_e - \mu^i_w)\right]$*, a gravity compensator, and damped environment* $w = K_d\dot{x}$*. The closed-loop system is asymptotically stable at* $(x, \dot{x}) = 0$ *if:*

$$K_P \succeq 0 \tag{3.13}$$

$$K_I \succ 0 \tag{3.14}$$

$$\bar{\mu}_{\dot{x}} = 0 \tag{3.15}$$

$$K_P A^i K_d \succeq 0 \qquad \forall i = 1, 2, \cdots, N \tag{3.16}$$

Proof 1 *First define a candidate Lyapunov function*

$$V = \frac{1}{2}x^T K_I x + \frac{1}{2}\dot{x}^T M_x(x)\dot{x} \tag{3.17}$$

where $\dot{x}$ *is the velocity of the robot end-effector and* $M_x(x)$ *is the robot inertia matrix in Cartesian space. Since* $K_I \succ 0$ *and* $M_x(x) \succ 0$*, thus* $V > 0$*. Take derivative of (3.17) and obtain*

$$\dot{V} = \dot{x}^T K_I x + \frac{1}{2}\dot{x}^T \dot{M}_x(x, \dot{x})\dot{x} + \dot{x}^T M_x(x)\ddot{x} \tag{3.18}$$

In Cartesian space, take the robot dynamic equation

$$M_x(x)\ddot{x} + C_x(x, \dot{x})\dot{x} + G_x(x) = \tau_x \tag{3.19}$$

where $C_x(x, \dot{x})$ *is the Coriolis matrix,* $G_x(x)$ *is the gravity term, and* τ_x *is the control input.* $\dot{M}_x(x, \dot{x}) - 2C_x(x, \dot{x})$ *keeps the property of skew-symmetry as in the joint space, thus*

$$\dot{x}^T \left[\dot{M}_x(x, \dot{x}) - 2C_x(x, \dot{x})\right]\dot{x} = 0 \quad \forall \dot{x} \tag{3.20}$$

Substitute (3.19) and (3.20) into (3.18) to get

$$\begin{aligned} \dot{V} &= \dot{x}^T K_I x + \frac{1}{2}\dot{x}^T \dot{M}_x(x, \dot{x})\dot{x} + \dot{x}^T\left[\tau_x - C_x(x, \dot{x})\dot{x} - G_x(x)\right] \\ &= \dot{x}^T K_I x + \frac{1}{2}\dot{x}^T\left[\dot{M}_x(x, \dot{x}) - 2C_x(x, \dot{x})\right]\dot{x} + \dot{x}^T[\tau_x - G_x(x)] \\ &= \dot{x}^T[\tau_x - G_x(x) + K_I x] \end{aligned} \tag{3.21}$$

For the Cartesian space control law,

$$\begin{aligned} \tau_x &= \tau_v + G_x(x) \\ &= K_P[\dot{x}_c - \dot{x}] - K_I x + G_x(x) \\ &= K_P\{\sum_{i=1}^{N} h^i(w_e)\left[\mu^i_{\dot{x}_c} + A^i(w_e - \mu^i_w)\right] - \dot{x}\} - K_I x + G_x(x) \\ &= K_P\left[\sum_{i=1}^{N} h^i(w_e)A^i w_e + \bar{\mu}_{\dot{x}} - \dot{x}\right] - K_I x + G_x(x) \end{aligned} \tag{3.22}$$

The desired contact wrench w_d *is zero, therefore* $w_e = w_d - K_d\dot{x} = -K_d\dot{x}$*. Also from the stability condition,* $\bar{\mu}_{\dot{x}} = 0$*. Substitute the relation into (3.22) to obtain*

$$\tau_x = -K_P\left[\sum_{i=1}^{N} h^i(w_e)A^iK_d + I\right]\dot{x} - K_I x + G_x(x) \tag{3.23}$$

Substituting (3.23) into (3.21), we obtain

$$\dot{V} = -\dot{x}^T\left[\sum_{i=1}^{N} h^i(w_e)K_PA^iK_d + K_P\right]\dot{x} \tag{3.24}$$

Since $K_P \succeq 0$*,* $K_PA^iK_d \succeq 0$ *and* $h^i(w_e) \in [0,1]$*, the superposition* $[\sum_{i=1}^{N} h^i(w_e)K_PA^iK_d + K_P]$ *should also be positive semi-definite. Therefore* $\dot{V} \leq 0$*, and the closed-loop system is Lyapunov stable at* $(x, \dot{x}) = 0$*. By further applying LaSalle's Invariance Principle on this autonomous system, it is found that the largest invariant set contains the only equilibrium point* $(x, \dot{x}) = 0$*. Finally, we can conclude that the closed-loop system is asymptotically stable at* $(x, \dot{x}) = 0$*.*

End of proof.

Note that the theorem only gives a sufficient condition for closed-loop stability, and the assumptions in the theorem are conservative. Assumption 3.16 is non-trivial to achieve. However, if K_d is known, we can manually choose $K_P = K_d^T$ to satisfy it, which can be proved by Cholesky decomposition. Another assumption is that the environment only contains damping terms. In the future work, it is desired to relax these assumptions so as to be feasible for more general scenarios.

3.4 Experimental Study

To demonstrate the performance of the learned skill for compliant assembly, a series of experiments were performed on industrial robot FANUC LR Mate 200*i*D/7L. The experimental video can be found in [104].

The testbed is shown in Fig. 3.8. The peg and hole are both machined from Aluminium 6061-T6, with a peg diameter of 0.999in (25.370mm), a hole diameter of 1.000in (25.400mm) and 1.0mm chamfers. The assembly tolerance is industrial standard H7h7. At the top of the peg, there are slots to fit the robot's parallel gripper, which ensures the peg to be grasped firmly by the robot.

To collect the human demonstration data, the HDD device (see Fig. 3.9) is designed. The ATI-Mini45 F/T sensor [5] is embedded between peg and handle bar to collect the wrench information that human perceives during demonstration. The PhaseSpace motion capture system [46] with five active makers on HDD records the corrective velocity that human applies on the peg.

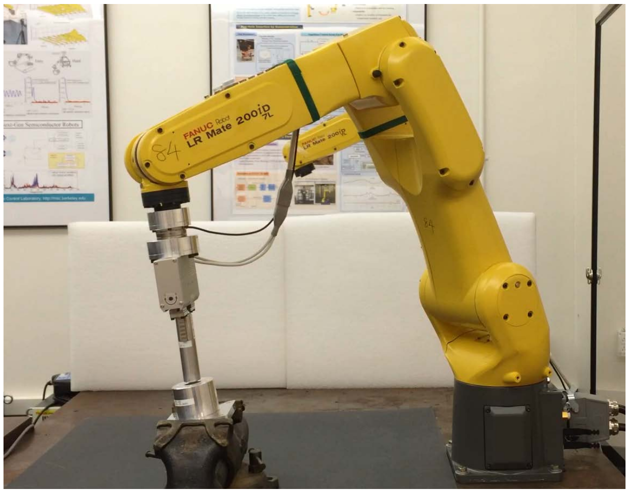


Figure 3.8: Peg-hole-insertion testbed. The diameters of peg and hole are 25.370mm and 25.400mm respectively, with 0.030mm clearance (H7h7 tolerance). The model of industrial robot is FANUC LRMate-200iD.

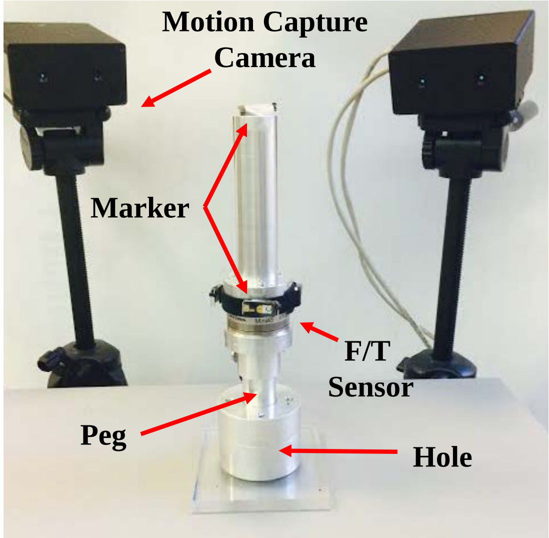


Figure 3.9: Human demonstration device (HDD) for data acquisition. ATI Mini45 F/T sensor collects the wrench information during assembly. PhaseSpace motion capture system records the corrective velocity that human applies on the peg.

As shown in Fig. 3.10, during the human demonstration phase, human demonstrates insertions 50 times from random initial poses. The wrench w and corrective velocity $\dot{x}_c$ are recorded with sampling frequency 1kHz and 960Hz respectively. The two data sets with different frequencies are then synchronized by linear interpolation and smoothed by a moving average filter.

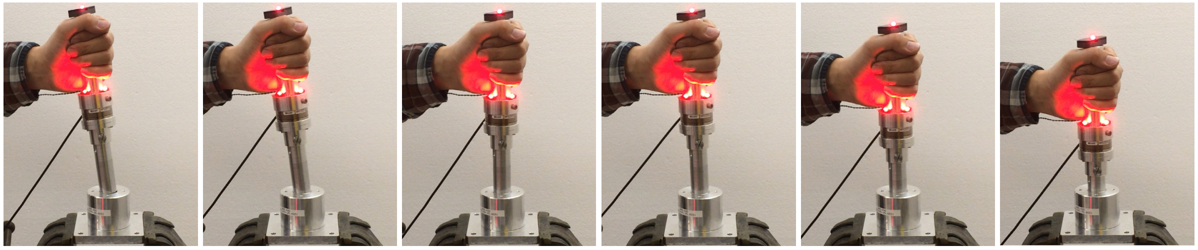


Figure 3.10: Human demonstrates the insertion procedure.

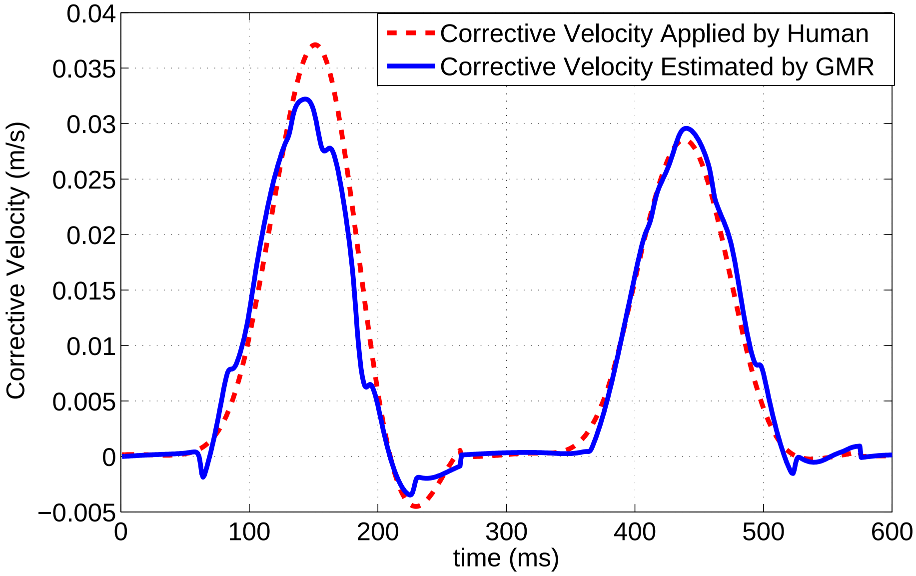


Figure 3.11: Prediction performance of GMR on test data.

In assembly tasks, the rotational velocity around peg axis is usually fixed to be zero ($\omega_z = 0$), and the translational velocity along peg axis is defined manually as task requirement. Here, $v_z = \max\{0, 0.01(1 - F_z/20)\}$. So insertion is at a 0.01m/s feeding speed along the peg axis if there is no resistance force F_z, and slows down linearly when F_z increases. If $F_z \geq 20N$, the robot will stop feeding. The velocity command for other four dimensions $[v_x, v_y, \omega_x, \omega_y]$ is calculated by the GMR admittance block.

For training the admittance block, fifteen Gaussian components ($N = 15$) are used in the GMR model. The Gaussian parameters are initialized by K-means clustering and iteratively optimized by E-M algorithm. Fig. 3.11 shows the GMR's performance on test data. For the same wrench input, the red dashed line is the corrective velocity applied by the human, and the blue solid line is the GMR output. The average estimation error is $0.213mm/s$, with standard variation $2.2mm/s$.

It is the residue misalignment between the peg and hole that makes the insertion task

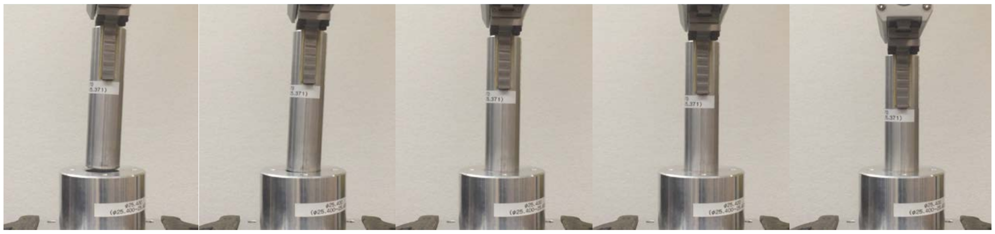


Figure 3.12: Process of robot autonomous insertion with initial orientation misalignment.

Table 3.1: Experimental results of insertion under four initial conditions. (a) Small position misalignment, and small orientation misalignment. (b) Large position misalignment, and small orientation misalignment. (c) Small position misalignment, and large orientation misalignment. (d) Large position misalignment, and large orientation misalignment.

	Success Rates (%)	Avg. Cycle Time (s)	Avg. Contact Force (N)	Avg. Contact Torque (N.m)
(a)	100.0	2.2	1.87	0.023
(b)	100.0	2.3	2.03	0.025
(c)	96.0	2.8	1.91	0.032
(d)	96.0	2.8	2.21	0.030

difficult. Therefore in the experiment, we introduce an initial misalignment on purpose to test the proposed learning method. The robot performs autonomous insertions in the following four groups with different misalignment conditions.

(a) Small position misalignment ($\pm 1mm$), and small orientation misalignment ($\pm 1^\circ$).

(b) Large position misalignment ($\pm 2mm$), and small orientation misalignment ($\pm 1^\circ$).

(c) Small position misalignment ($\pm 1mm$), and large orientation misalignment ($\pm 3^\circ$).

(d) Large position misalignment ($\pm 2mm$), and large orientation misalignment ($\pm 3^\circ$).

There are 25 trails in each group and the insertion depth is $20mm$ in each trial. The process of the robot autonomous insertion is illustrated in Fig. 3.12. This trial of insertion starts with a large orientation error. The robot succeeds to correct the orientation by the trained state-varying admittance block. Table 3.1 shows the experimental results under the four specified conditions. The success rates are all over 96%, which indicates the robot force control could generate proper commands to adjust the peg's pose during insertion. The best performance is achieved (100% success rate) when there is small orientation misalignment. The success rate drops to 96% for large orientation misalignment. This indicates that the

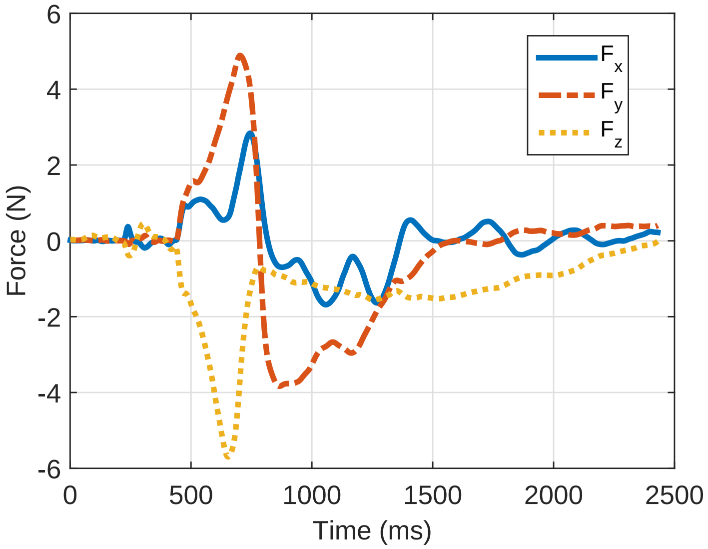


Figure 3.13: Force plot during one trial of peg-hole-insertion.

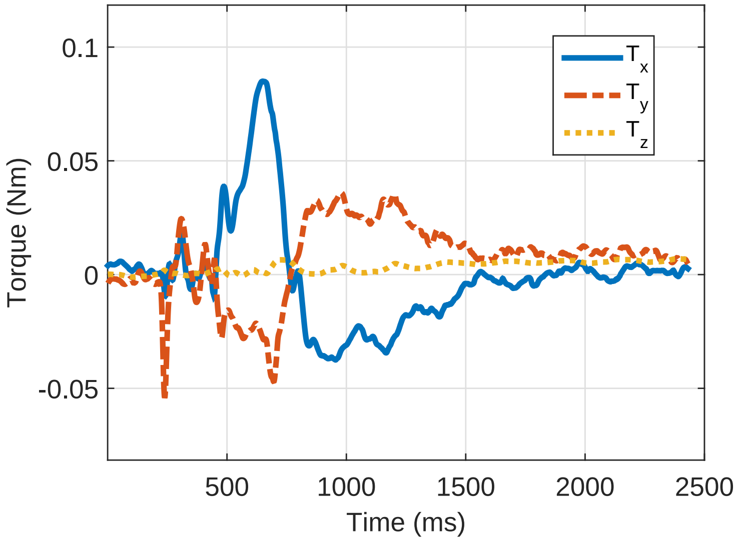


Figure 3.14: Torque plot during one trial of peg-hole-insertion.

proposed learning method is relatively non-sensitive to position misalignment, but more sensitive to orientation misalignment.

Fig. 3.13 and Fig. 3.14 show the force and torque plots during one trial of insertion. Contacting happens at 0.25s, and large force and torque occur during contact with peak force $5.71N$ and peak torque $0.085Nm$. The force controller then takes action and regulates both force and torque towards zero. The residual force error is less than $0.41N$ and residual torque is less than $0.006Nm$. Total insertion time is $2.43s$.

To conclude, the proposed data-driven method is suitable for non-backdrivable industrial robots and performs well on peg-hole-insertion tasks with industrial tolerance standard (H7h7). Traditional force control methods, if well tuned, can also regulate the contact wrench very well and achieve high success rate. However, the major advantage of our method is to eliminate the gain tuning process on the robot. The admittance gains in the force controller are directly learned from human demonstration data. The demonstration process is safe, efficient and intuitive. Moreover, since there is no tuning process, this method does not require high skills or expertise on the robot operator.

3.5 Chapter Summary

In this chapter, a model-free approach is introduced to learn a compliant motion skill for robotic assembly. Instead of manually tuning the linear admittance gain in model-based force controllers, this work utilizes Gaussian mixture regression (GMR) to learn the dynamic admittance directly from human demonstration data. A human demonstration device (HDD) is designed to collect the wrench and corrective velocity information during demonstration. To improve the data quality, which is critical for model-free approaches, the procedures for data acquisition and preprocessing are illustrated in detail. The efficiency, physical interpretation and stability conditions of the GMR admittance module are also analyzed. A series of assembly experiments performed on a FANUC industrial manipulator and a H7h7 tolerance testbed demonstrate the effectiveness of the proposed learning framework.

Chapter 4

A New Approach for Teaching Robots: Analogy Learning

4.1 Introduction

Model-based learning (Chapter 2) and model-free learning (Chapter 3) are both powerful approaches for teaching robots novel skills. Generally, they follow the same pipeline as shown in Fig. 1.3. During the training stage, the desired control policy is either constructed by modelling or regressed from training data, which tries to capture the system's causality and represent how the system progresses. The achieved policy is supposed to be universal and applicable. Therefore, it is applied during the test stage, i.e., new actions will be generated according to the test states and the trained control policy. This pipeline has been inherently or implicitly utilized in most of robot learning methods and achieved great success.

However, for some complicated tasks, neither sophisticated models nor a large amount of training data can be easily obtained for training the control policy. Under this background, this chapter will propose a novel pipeline for robot learning, named analogy learning, which is both model-free and data-efficient. This new pipeline focuses on finding correlation, instead of deriving causality. It focuses on finding a transformation between the training and test scenarios, instead of formulating a control policy to relate the training states and training actions. The concept of analogy learning will be introduced in detail in Section 4.2. Section 4.3 compares the analogy learning with traditional learning approaches to illustrate its advantages. A robust non-rigid registration method, structure preserved registration (SPR) is introduced in Section 4.4 for implementing analogy learning. Section 4.6 provides summaries and discussions. The applications of analogy learning on learning new robotic skills will also be introduced in the following chapters, including versatile grasping (Chapter 5), deformable object tracking (Chapter 6), deformable object manipulation (Chapter 6) and motion planning (Chapter 7).

4.2 Concept of Analogy Learning

Analogy is a cognitive process of transferring information from a particular subject (the training scenario) to another (the test scenario). It is a powerful mechanism for exploiting past experience in planning and problem solving [19].

In general, there are two major stages in a complete analogy pipeline (Fig. 1.4). First is the reminding stage, where the past scenarios are reminded and compared with the current problem, and the one bearing strong similarity will be identified. Second is the transferring stage, where a transformation process is deployed to retrieve actions that are appropriate for the previous similar scenarios to solve the current problem. Some adaptation is required to update the transformed action to meet the new demands for the current situation.

Analogy is a critical inference method in human cognition. Some robotic researches have implicitly utilized this idea for specific manipulation tasks, such as rope knotting [98]. Unfortunately, analogy learning as a methodology is still under-explored, and has not been broadly applied for learning various skills for robots.

Take robotic grasping as an example. The objective is to find a proper pose of the robot gripper to firmly grasp a specific object. There are already numerous methods developed for robotic grasping. Some methods developed grasping models based on quantitative analysis, such as force closure [86, 92] and form closure [73, 72]. Some methods follow a model-free approach assuming that a grasp policy can be learned from thousand or even million times of trial and error [61, 91, 76]. However, when human beings grasp objects, neither we need the complicated mathematical model for analysis, nor we require million times of learning iterations. There must be some mechanisms involved that enable humans to grasp objects robustly and intuitively with only limited training samples provided. Analogy leaning might be one of the hidden mechanisms. For example, when grasping a cup, humans might figure out that the handle is a good location for a robust grasp. In future scenarios where various kinds of cups are given, humans can transfer this experience and regard the handles as good candidates for grasping. There is no physical model involved in the procedure, and only one training sample is required.

Another example is path planning. The robot needs to plan a path connecting the start configuration towards the target configuration without colliding with obstacles. There are model-based approaches (optimization [95, 99], sampling [58, 49]) and model-free approaches (neural network [117, 118], reinforcement learning [54, 53]) for constructing a path planner. However, when human beings are planning paths, we do not need to initialize with an infeasible path and then use resampling or gradient-descent to gradually avoid collision. We neither have to collect much training data and run into obstacles thousands of times before achieving a satisfying policy. Analogy learning might play an important role for humans to plan paths efficiently and intuitively, without sophisticated models or large numbers of trails. As Fig. 4.1 shows, humans can transfer our previous successful experience across scenarios so as to plan new paths. In Fig. 4.1(a) and Fig. 4.1(b), the obstacle shapes are different. Instead of regarding the two scenarios independently and solve separately, we can find a spacial transformation from scenario A to scenario B, and transfer the path in A to

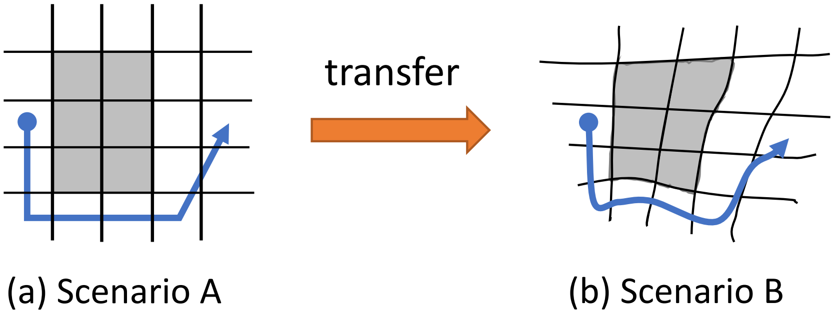


Figure 4.1: Path planning with analogy learning. Grey blocks are obstacles, and blue lines are planned paths. The obstacle shapes are different in two scenarios. A spatial transformation function between the two obstacles can be found and applied to transfer the path for A to get a feasible path for B.

immediately get a new feasible path in B. During this procedure, no sophisticated model is involved besides a simple spatial transformation. Moreover, only one training sample (scenario A) is required, and can be transferred to various scenarios which contain a single closed obstacle.

Fig. 4.2 shows the general procedures for applying analogy learning to robots. A training database (x_i, u_i) which includes pairs of training states and corresponding actions, is given in advance as past experience. At test stage, the system arrives at a new state x'. Analogy learning will remind the most similar scenario (x^*, u^*) in the database by calculating the similarity between the current scenario and past scenarios. A transformation function will be constructed to register x^* and x'. The same function is then utilized to transform the action u^* from the past scenario to get a new one u', which is feasible for the current scenario x'. If the similarity between x^* and x' is smaller than a threshold, which indicates that the most similar scenario in the past is still not similar enough with the current state, the transformation procedure will not be implemented since the base for analogy does not exist. Instead, human demonstration or traditional approaches will be utilized to provide the corresponding action u'. The demonstrated data sample (x', u') will then be augmented into the database. Since the database (experience) is growing, if the system runs into the similar scenario x' in the future, the analogy-based transformation can be executed without the need of additional demonstration.

4.3 Advantages of Analogy Learning

Analogy learning is an intuitive methodology since human beings utilize this mechanism inherently in our daily lives. Besides the intuitiveness, applying this approach to robots have many other benefits including efficiency in learning, robustness to noise and guaranteed collision-free property.

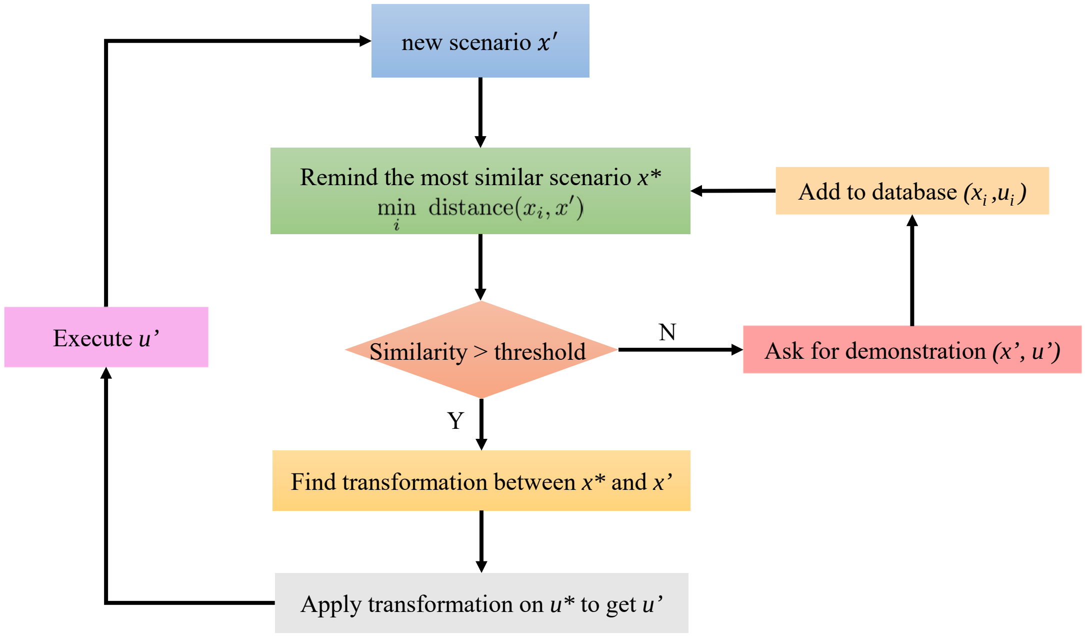


Figure 4.2: Procedures in analogy learning.

Learn from SMALL data

Because of the difficulty in modelling, many methods recently focus on data-driven approaches to regress a control policy for robots from data measurement, such as end-to-end learning [63]. Usually these methods require a tremendous amount of data for training before achieving a satisfying result at test. A new phrase, learning from Big Data, is created for describing this new trend [74]. However, accompanying with the rise of learning from Big Data, vast concerns also alarm that why so much data is required. On the one hand, data is not cheap to collect for physical robots, especially industrial manipulators. One may claim to use simulators to virtually simulate a robot. It is true that in a simulation platform, the virtual time can be accelerated arbitrarily to collect more data within the same period. Trial-and-error is also acceptable since simply restarting the program and the electronic world will recover from failures. Unfortunately, creating a sufficiently accurate simulator for physical robots is still a challenging and unsolved problem. As small errors due to the under-modelling accumulate, the simulated robot can quickly diverge from the real-world system. Therefore, reliable data for robot learning still comes from experiments instead of simulations at the current stage. As a result, the data collection process is very expensive. On the other hand, learning from big data is not the straightforward mechanism that human beings are taking. Just the opposite, humans learn from small data: only a few samples are enough for humans to learn a skill and to generalize to many other similar situations. It is noticeable that many state-of-the-art methods do succeed in reconstructing the human brain's structure, such as by deep neural networks, but one of the human brain's most remarkable features, learning from small data, is regrettably lost.

Why are the current learning methods data hungry? Why can human beings learn from limited data? How to let robots to learn from small data? These fundamental questions have not been answered in robotics society. In this section, we will try to tackle these problems from the perspective of analogy learning.

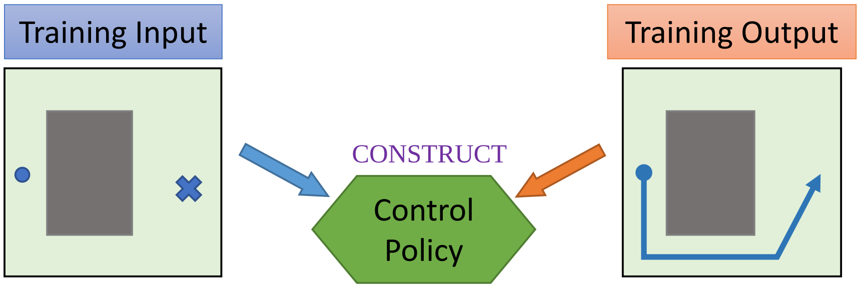
In the author's opinion, finding a control policy is trying to figure out a causality, which is universal and generalizable. Causality explains the nature of a system and relates the system states with proper actions. If the causality itself is simple, we can easily construct a model to describe it. If the causality is complex and beyond our ability of modelling, we use model-free approaches to approximate it by high dimensional functions. Whereas it is because the causality is complex and the parametric function is in high dimensional, this approximation inevitably requires big data. If we insist on finding a clear causality using the current approximation methodology, finally we will drop to the dilemma of requiring big data for training.

However, it is not always necessary to require a clear causality to solve a problem. Humans can grasp objects much better than robots without understanding force-closure. Humans can plan paths much robustly without understanding space sampling or optimization. Many times, we finish tasks without understanding the physical principles behind. We cannot explain the causality that why we take this specific action. Instead, we just intuitively transfer the previous successful experience and apply to the current scenarios, so called analogy learning. The core concept of analogy learning is to find the correlation between past scenarios and current situations, instead of the causality between states and actions. It is transferring knowledge, instead of explaining knowledge itself. Its objective is to finish a task by exploiting the past problem-solving actions, instead of explaining how the system works in a fundamental way.

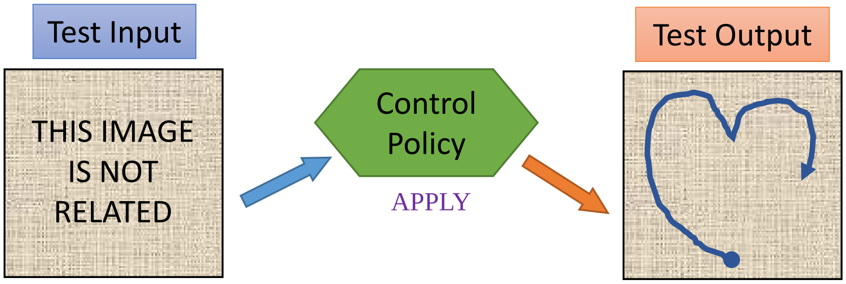
Many state-of-the-art robot learning methods are trying to find the causality, the procedure for which is data hungry. Our proposed analogy learning approach learns from small data, since it does not seek for the causality. In an extreme case, only one training sample is enough for knowledge transfer. Chapter 5, 6, 7 will show the implementations to teach robots complex skills with limited amount of training samples.

Robustness to input noise

For traditional learning methods, the control policy is learned during training stage and applied to test stage. However, many of them do not have an inherent functionality to distinguish the input relevance. Take path planning as an example. As shown in Fig. 4.3, a collision-free path planner is trained by learning methods with training dataset. At test stage, the test scenario might not be related to the training scenarios at all, but the trained control policy will still generate a corresponding output and execute it to the system, which is meaningless and might be dangerous to the robot operation.



(a) Training Stage



(b) Test Stage

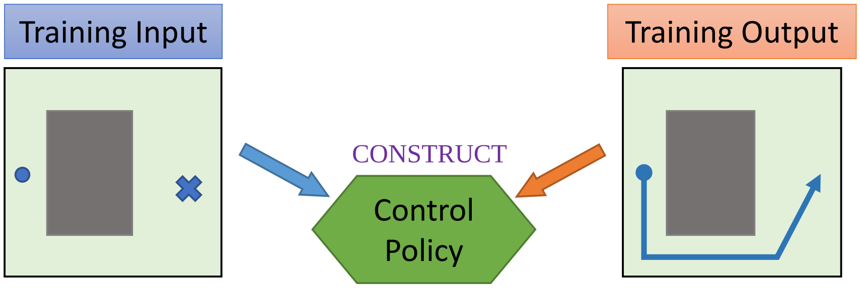
Figure 4.3: Many learning methods do not distinguish the input relevance. (a) At training stage, a collision-free path planner is trained by learning methods with training dataset. (b) At test stage, though the test input is not related to the problem at all, the trained control policy cannot distinguish this noise and will still generate a corresponding output to the system.

As a contrast, the analogy learning approach is robust to this input noise. As shown in Fig. 4.2, the reminding stage in analogy learning will first check the similarity between the current input and the training inputs. Following transferring operation will not be executed unless the scenario similarity is large enough. This self-awareness of what can be handled and what cannot brings additional robustness to the system.

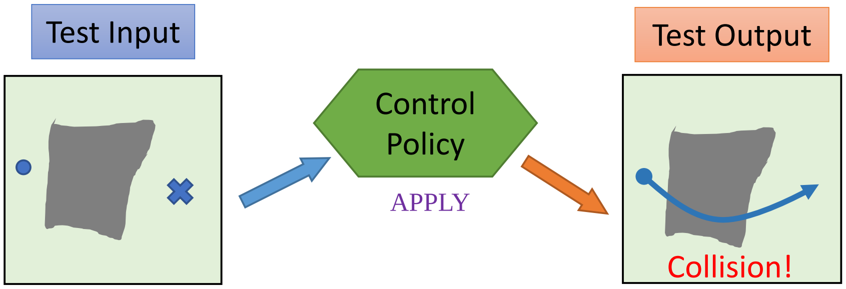
Guaranteed collision-free property

Because of the black-box nature in many model-free approaches, the stability and robustness of the system is non-trivial to analyze, which introduces potential risks on system safety. As shown in Fig. 4.4, the trained path planner might work well for training data, but could fail and generate infeasible paths for test scenarios.

In contrast, the proposed analogy learning approach can provide a guaranteed collision-free property for robotic motion planning tasks. As illustrated in Fig. 4.1, during the path transformation procedure, if the original path is collision-free, it can be proved that the trans-



(a) Training Stage



(b) Test Stage

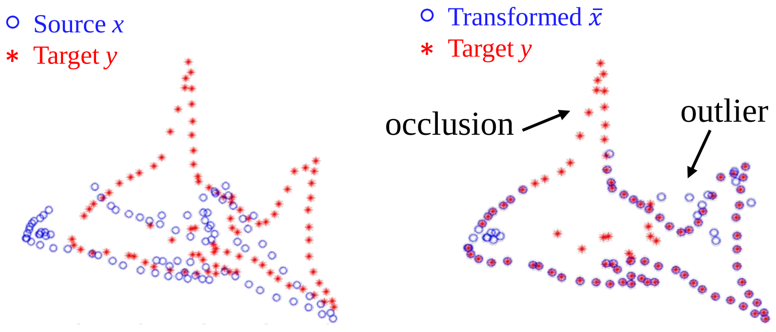
Figure 4.4: Many learned path planners do not guarantee collision-free property. (a) At training stage, a path planner is trained by learning methods with training dataset. (b) At test stage, a new path is generated given new obstacles. However, the collision avoidance is not guaranteed.

formed path is also collision-free. This property is desired especially in those safety-critical scenarios, such as autonomous driving and industrial manipulation. The mathematical proof is provided as follows.

Theorem 2 *Given an obstacle (closed set $O \in \mathbb{R}^n$), a collision-free path (closed set $P \in \mathbb{R}^n$), and a homeomorphism (bicontinuous) mapping function ($f : \mathbb{R}^n \to \mathbb{R}^n$), the transformed path $f(P)$ must be collision-free with respect to the transformed obstacle $f(O)$.*

Proof 2 *Since the source obstacle and source path are collision-free, we have $O \cap P = \varnothing$. Assume that the transformed path collides with the transformed obstacle, i.e., $f(O) \cap f(P) \neq \varnothing$, then there exists an intersection point $x \in f(P) \cap f(O) \neq \varnothing$. The inverse transform of the intersection point is $x' = f^{-1}(x) \neq \varnothing$. However, $x' \in P \cap O = \varnothing$. Contradiction. Therefore, we must have $f(O) \cap f(P) = \varnothing$, i.e., the transformed path does not collide with the transformed obstacle.*

End of proof.



(a) Before Registration (b) After Registration

Figure 4.5: Illustration of SPR registration. The source point set (blue circle) is registered towards the target point set (red start) by a smooth transformation.

4.4 Structure Preserved Registration for Analogy Learning

To implement analogy learning, we need to find a mapping between scenarios, such that: (1) the similarity between current scenario and training scenarios can be quantitatively calculated; (2) a transformation function can be constructed such that actions in the past scenario can be transferred.

To satisfy these requirements, structure preserved registration (SPR), a non-rigid registration method to register two point sets non-rigidly, is developed in this dissertation. The general idea is to regard the robot training and test scenarios as two point sets. The point correspondence between two sets will be calculated based on a Gaussian mixture probability model. A smooth transformation function will also be formulated to register the corresponding points. This SPR method is shown to be more effective and robustness than other state-of-the-art non-rigid registration methods, such as TPS-RPM [22], CPD [82] and GLTP [36]. SPR will be utilized in the following chapters to implement the analogy learning approach on robotic grasping, tracking, manipulation and planning.

Node Registration with Gaussian Mixture Model

Assume there are two sets of point clouds, source point set $X = \{x_1, x_2, \ldots, x_N\} \in \mathbb{R}^{N\times D}$ and target point set $Y = \{y_1, y_2, \ldots, y_M\} \in \mathbb{R}^{M\times D}$ to represent two environment. N and M are the point numbers in X and Y respectively. D is the point dimension. The objective of SPR is to find a smooth transformation function $\mathcal{T} : \mathbb{R}^D \to \mathbb{R}^D$ to map X to a new position $\bar{X} = \{\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_N\} \in \mathbb{R}^{N\times D}$, such that $\bar{X}$ is well aligned with Y.

Fig. 4.5 provides an example of SPR registration. The blue source point set is registered towards the red target point set by a smooth transformation. The point set might contain outliers or exclude partial points.

For measuring the alignment level between transformed source set $\bar{X}$ and target set Y, a Gaussian mixture model is constructed, where the transformed points in $\bar{X}$ are regarded as the centroids of multiple Gaussians, and points in Y are random samples from the Gaussian mixtures.

Assume that each Gaussian has equal membership probability $\frac{1}{N}$ and consistent isotropic covariance $\sigma^2\mathbf{I}$. The probability of point y_m sampled from the Gaussian mixtures can be calculated by:

$$\begin{aligned} p(y_m) &= \sum_{n=1}^{N} \frac{1}{N}\mathcal{N}(y_m; \bar{x}_n, \sigma^2\mathbf{I}) \\ &= \sum_{n=1}^{N} \frac{1}{N}\frac{1}{(2\pi\sigma^2)^{D/2}}\exp(-\frac{\|y_m - \bar{x}_n\|^2}{2\sigma^2}) \end{aligned} \tag{4.1}$$

In practice, however, the point set might be noisy and contain outliers. An additional uniform distribution is therefore added to the mixture model to account for noise and outliers. The complete mixture model takes the form:

$$p(y_m) = \sum_{n=1}^{N+1} p(n)p(y_m|n) \tag{4.2}$$

with

$$p(n) = \begin{cases} (1-\mu)\frac{1}{N}, & n = 1, \ldots, N \\ \mu, & n = N+1 \end{cases} \tag{4.3}$$

$$p(y_m|n) = \begin{cases} \mathcal{N}(y_m; \bar{x}_n, \sigma^2\mathbf{I}), & n = 1, \ldots, N \\ \frac{1}{M}, & n = N+1 \end{cases} \tag{4.4}$$

where μ denotes the weight of the uniform distribution.

$(\bar{x}_n, \sigma^2)$ parameterize the probability distribution and can be estimated by maximizing the log-likelihood $\mathcal{L}$ of the observation:

$$\begin{aligned} \mathcal{L}(\bar{x}_n, \sigma^2|Y) &= \log \prod_{m=1}^{M} p(y_m) \\ &= \sum_{m=1}^{M} \log\left(\sum_{n=1}^{N+1} p(n)p(y_m|n)\right) \end{aligned} \tag{4.5}$$

$$(\bar{x}_n^*, \sigma^{2*}) = \arg\max_{\bar{x}_n, \sigma^2} \mathcal{L}(\bar{x}_n, \sigma^2|Y) \tag{4.6}$$

However, it is non-trivial to solve for $(\bar{x}_n, \sigma^2)$ by directly optimizing over the log-likelihood function $\mathcal{L}$, since there is a summation inside $\log(\cdot)$ which makes convex optimization infeasible. A complete log-likelihood function Q is therefore constructed,

$$Q(\bar{x}_n, \sigma^2) = \sum_{m=1}^{M}\sum_{n=1}^{N+1} p(n|y_m)\log\left(p(n)p(y_m|n)\right) \tag{4.7}$$

It can be proved by Jensen's inequality [56] that function Q is the lower bound of function $\mathcal{L}$. Therefore, increasing the value of Q will necessarily increase the value of $\mathcal{L}$ unless it is already at the local optimum. Comparing the structure of Q to that of $\mathcal{L}$, the inside summation is moved to the front of $\log(\cdot)$, which provides much convenience for the following computation.

With the definition of complete log-likelihood function, the EM algorithm [27] which runs expectation step (E-step) and maximization step (M-step) can be utilized to iteratively estimate $(\bar{x}_n, \sigma^2)$ by maximizing Q:

E-step

The expectation step calculates the posteriori probability distribution $p(n|y_m)$ with "old" $(\bar{x}_n, \sigma^2)$ from the last M-step:

$$p(n|y_m) = \frac{\exp(-\frac{\|y_m - \bar{x}_n\|^2}{2\sigma^2})}{\sum_{n=1}^{N} \exp(-\frac{\|y_m - \bar{x}_n\|^2}{2\sigma^2}) + \frac{(2\pi\sigma^2)^{D/2}\mu N}{(1-\mu)M}} \tag{4.8}$$

M-step

The maximization step plugs $p(n|y_m)$ into the complete log-likelihood function. Ignoring the constants independent of $(\bar{x}_n, \sigma^2)$, we get

$$Q = -\sum_{m=1}^{M}\sum_{n=1}^{N} p(n|y_m)\frac{\|y_m - \bar{x}_n\|^2}{2\sigma^2} - \frac{N_p D}{2}\log(\sigma^2) \tag{4.9}$$

where $N_p = \sum_{m=1}^{M}\sum_{n=1}^{N} p(n|y_m)$.

Take partial derivative of Q and make $\frac{\partial Q}{\partial \bar{x}_n} = 0$, $\frac{\partial Q}{\partial \sigma^2} = 0$, then a new estimate of $(\bar{x}_n, \sigma^2)$ is achieved:

$$\bar{x}_n = \frac{\sum_{m=1}^{M} p(n|y_m) y_m}{\sum_{m=1}^{M} p(n|y_m)} \tag{4.10}$$

$$\sigma^2 = \frac{\sum_{m=1}^{M}\sum_{n=1}^{N} p(n|y_m)\|y_m - \bar{x}_n\|^2}{\sum_{m=1}^{M}\sum_{n=1}^{N} p(n|y_m) D} \tag{4.11}$$

The expectation step and maximization step will be taken alternately until the value of Q is converged.

Regularization on Local Structure

Under the aforementioned registration framework, there are no inner constraints between the Gaussian centroids. Each of the centroids will be inevitably registered to the point-rich area to pursue a higher likelihood value (Fig. 4.6b). With respect to a physical object, in contrast, there should exist an inherent topological structure that organizes all the nodes and constraints their motions in sequences.

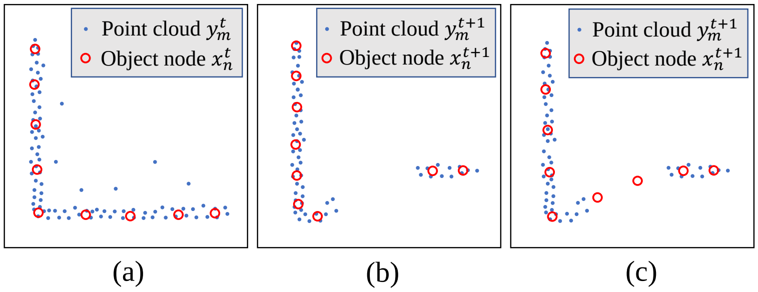


Figure 4.6: Comparison of GMM and SPR on point registration. Blue dot is the point cloud, and red circle is the estimated node position. (1) At time step t, GMM can align X towards Y regardless of noise and outliers. (2) At time step $t+1$, occlusion happens. GMM fails to register points in occlusion area, and the registration on other area are also seriously disturbed. (3) SPR still works under occlusion because the topological structure is preserved.

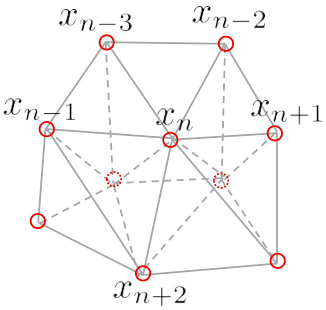


Figure 4.7: Local topology captured by surrounding points.

For this reason, we introduced topological regularization on the transformation, which considers both local and global topology maintenance during transferring X to $\bar{X}$.

Fig. 4.7 shows the point x_n and its neighbor points. The local structure around x_n can be characterized as the following weighted sum:

$$x_n = \sum_{i \in I_n} S_{ni} \cdot x_i \tag{4.12}$$

where I_n is the index of the K nearest points to x_n, which can be found efficiently by K-nearest neighbor (KNN, [90]) algorithm. Weight S_{ni} captures the local structure between x_n and its surrounding point x_i. When the point set X is deformed to the new set $\bar{X}$, the absolute point positions are changed. Their inner local structure S_{ni}, however, are expected to be maintained, i.e.:

$$\bar{x}_n \approx \sum_{i \in I_n} S_{ni} \cdot \bar{x}_i \tag{4.13}$$

The optimal combination weights S_{ni} can be achieved by solving the following constrained least square problem:

$$\begin{aligned} \min_{S_{ni}} \quad & \|x_n - \sum_{i \in I_n} S_{ni} \cdot x_i\|^2 \\ s.t. \quad & \sum_{i \in I_n} S_{ni} = 1 \end{aligned} \tag{4.14}$$

Define a difference matrix $D_n = [\cdots, x_i - x_n, \cdots]_{i \in I_n} \in \mathbb{R}^{D \times K}$, and denote the orthogonal projection of one vector $\mathbf{1}_K \in \mathbb{R}^K$ on the null space of D_n as y_0. The optimal solution S_n^* (vectorization of S_{ni}^*) of (4.14) is:

$$S_n^* = \begin{cases} \frac{y_0}{\mathbf{1}_K^T y_0} & \text{, if } \ y_0 \neq 0 \\ \frac{(D_n^T D_n)^\dagger \mathbf{1}_K}{\mathbf{1}_K^T (D_n^T D_n)^\dagger \mathbf{1}_K} & \text{, else} \end{cases} \tag{4.15}$$

Unfortunately, the problem of solving S_n^* is not stable if $D_n^T D_n$ is singular or nearly singular, i.e., a small perturbation on the values of x_i might result in large change in the solution of S_n^*. Therefore, the local structure is not captured well. From [119], if D_n has L small singular values, there are L suboptimal weight vectors that are linearly independent to each other, each of which partially describes the local structure around point x_n. We can integrate all the L suboptimal weights to characterize the local topology more completely. Denote

$$S_n^{(l)} = (1-\alpha) S_n^* + VH(:, l), \quad l = 1, \cdots, L \tag{4.16}$$

where V is the right singular vector matrix of D_n corresponding to the L smallest singular values, and $\alpha = \frac{1}{\sqrt{L}} \|V^T \mathbf{1}_K\|$. H is a Householder matrix given by $H = I - 2hh^T$ with $h \in \mathbb{R}^L$ defined as:

$$h = \begin{cases} \frac{\alpha \mathbf{1}_L - V^T \mathbf{1}_K}{\|\alpha \mathbf{1}_L - V^T \mathbf{1}_K\|} & \text{, if } \ \alpha \mathbf{1}_L - V^T \mathbf{1}_K \neq \mathbf{0} \\ 0 & \text{, else} \end{cases} \tag{4.17}$$

It can be proved that $\|D_n S_n^{(l)}\| \leq \|D_n S_n^*\| + \sigma_{K-L+1}(D_n)$ [119]. With these L suboptimal weights, a regularization term for maintaining local topology can be designed as

$$E_{\text{Local}} = \sum_{n=1}^{N} \sum_{l=1}^{L} \| \sum_{i=1}^{N} S_{ni}^{(l)} \bar{x}_i \|^2 \tag{4.18}$$

with $S_{ni}^{(l)} = -1$ if $i = n$, and $S_{ni}^{(l)} = 0$ if $i \notin I_n$. Note that the weight $S_{ni}^{(l)}$ is calculated in X, but the regularization is applied in $\bar{X}$. It is desired that E_{Local} is as small as possible, which indicates the local structure from X is maintained in $\bar{X}$.

Regularization on Global Topology

Besides the local structure which regularizes the relative motions between neighbor points, the global topology which regularizes the displacements of the source point set from X to $\bar{X}$ is also critical and needs consideration.

From the macroscopic view, $\bar{X}$ can be assumed to be generated by X by a coherent movement:

$$\bar{x}_n = \mathcal{T}(x_n) \tag{4.19}$$

where $\mathcal{T} : \mathbb{R}^D \rightarrow \mathbb{R}^D$ generates globally rigid transformation while also allows locally non-rigid deformation. We would like to find a $\mathcal{T}$ as smooth as possible so as to transform X to $\bar{X}$ coherently. According to the regularization theory [37], the function smoothness can be quantitatively measured by norm $\int_{\mathbb{R}^D} \frac{|T(s)|^2}{G(s)} ds$, where $T(s)$ is the Fourier transform of $\mathcal{T}$. $G(s)$ is symmetric and real with $G(s) \rightarrow 0$ as $s \rightarrow \infty$. This Fourier-domain norm basically passes $\mathcal{T}$ by a high-pass filter, then measures its remaining power at high frequency. Intuitively, the larger the norm, the more 'oscillation' $\mathcal{T}$ will exhibit, i.e., less smoothness.

Therefore, the regularization term for maintaining global topology can be defined as

$$E_{\text{Global}} = \int_{\mathbb{R}^D} \frac{|T(s)|^2}{G(s)} ds \tag{4.20}$$

A modified likelihood function $\tilde{Q}$ is achieved by involving the regularizations on both local structure and global topology:

$$\begin{aligned} \tilde{Q} &= Q(\bar{x}_n, \sigma^2) - \frac{\tau}{2} E_{\text{Local}} - \frac{\lambda}{2} E_{\text{Global}} \\ &= Q\left(\mathcal{T}(x_n), \sigma^2\right) - \frac{\tau}{2} \sum_{n=1}^{N} \sum_{l=1}^{L} \| \sum_{i=1}^{N} S_{ni}^{(l)} \mathcal{T}(x_i) \|^2 \\ &\quad - \frac{\lambda}{2} \int_{\mathbb{R}^D} \frac{|T(s)|^2}{G(s)} ds \end{aligned} \tag{4.21}$$

Compared to (4.9), the new likelihood function is parameterized by $(\mathcal{T}, \sigma^2)$, instead of $(\bar{x}_n, \sigma^2)$. $\tau \in \mathbb{R}^+$ and $\lambda \in \mathbb{R}^+$ are trade-off weights which balance the data fitting accuracy (from $\bar{X}$ to Y), the local structure maintenance and the global transformation smoothness (from X to $\bar{X}$). The negative signs before τ and λ indicate a similar local structure and a smoother global transformation is preferred.

Closed-form Solution for Transformation Function

The optimal transformation $\mathcal{T}$ can be found by maximizing the modified likelihood function $\tilde{Q}$. In order to implement analogy learning efficiently, it is critical to perform the aforementioned structure preserved registration as fast as possible. This subsection will prove the existence of the closed-form solution for SPR, which enables the registration running in a high frequency.

Theorem 3 *The maximizer $\mathcal{T}^*$ of the modified likelihood function* (4.21) *has the following Radial Basis Function (RGF) form:*

$$\mathcal{T}(z) = \sum_{n=1}^{N} w_n g(z - x_n) \tag{4.22}$$

where kernel $g(\cdot)$ is the inverse Fourier transform of $G(s)$, and $w_n \in \mathbb{R}^D$ is the kernel weight.

Proof 3 *The transformation function $\mathcal{T}(\cdot)$ can be represented by its inverse Fourier transform:*

$$\mathcal{T}(x) = \int_{\mathbb{R}^D} T(s) e^{2\pi i <x,s>} ds \tag{4.23}$$

Substitute (4.23) *to* (4.21)*, and take derivative of $\tilde{Q}$ over $T(s)$:*

$$\begin{aligned}
\frac{\delta \tilde{Q}}{\delta T(s)} &= \sum_{m=1}^{M} \sum_{n=1}^{N} p_{nm} \frac{1}{\sigma^2} [y_m - \mathcal{T}(x_n)] \frac{\delta \mathcal{T}(x_n)}{\delta \mathcal{T}(s)} \\
&\quad - \frac{\lambda}{2} \int_{\mathbb{R}^D} \frac{\delta}{\delta T(s)} \frac{|T(s)|^2}{G(s)} \\
&\quad - \frac{\tau}{2} \sum_{n=1}^{N} \sum_{l=1}^{L} \frac{\delta}{\delta T(s)} \| \sum_{i=1}^{N} S_{ni}^{(l)} \mathcal{T}(x_i) \|^2 \\
&= \sum_{m=1}^{M} \sum_{n=1}^{N} p_{nm} \frac{1}{\sigma^2} [y_m - \mathcal{T}(x_n)] e^{2\pi i <x_n,s>} \\
&\quad - \lambda \frac{T(-s)}{G(s)} \\
&\quad - \tau \sum_{n=1}^{N} \sum_{l=1}^{L} \sum_{i=1}^{N} \sum_{j=1}^{N} S_{ni}^{(l)} S_{nj}^{(l)} \mathcal{T}(x_i) e^{2\pi i <x_j,s>}
\end{aligned} \tag{4.24}$$

Set $\frac{\delta \tilde{Q}}{\delta T(s)} = 0$ and define two variables:

$$a_n = \sum_{m=1}^{M} p_{nm} \frac{1}{\sigma^2} [y_m - \mathcal{T}(x_n)] \tag{4.25}$$

$$b_n = -\tau \sum_{l=1}^{L} \sum_{j=1}^{N} \sum_{i=1}^{N} S_{ji}^{(l)} S_{jn}^{(l)} \mathcal{T}(x_i) \tag{4.26}$$

The optimal $T^*(s)$ can be formulated as

$$T^*(s) = \tilde{g}(-s) \sum_{n=1}^{N} \frac{a_n + b_n}{\lambda} e^{-2\pi i <x_n, s>} \tag{4.27}$$

Take inverse Fourier transform of $T^*(s)$ to get $\mathcal{T}^*(x)$:

$$\begin{aligned} \mathcal{T}^*(x) &= g(x) * \sum_{n=1}^{N} \frac{a_n + b_n}{\lambda} \delta(x - x_n) \\ &= \sum_{n=1}^{N} w_n g(x - x_n) \end{aligned} \tag{4.28}$$

End of proof.

In general, kernel $g(\cdot)$ can take any formulation as long as it is symmetric, positive definite, and $G(s)$ behaves like a low-pass filter. For simplicity, a Gaussian kernel $g(\cdot)$ is chosen, with $g(z - x_n) = \exp(-\frac{\|z - x_n\|^2}{2\beta^2})$. $\beta \in \mathbb{R}^+$ is a manually tuned parameter which controls the rigidity of function $\mathcal{T}$, where large β corresponds to rigid transformation, while small β produces more locally deformation.

Theorem 4 *The modified likelihood function* (4.21) *is equivalent to:*

$$\begin{aligned} \tilde{Q} = &- \frac{1}{2\sigma^2} Tr(Y^T d(P^T \mathbf{1}) Y) + \frac{1}{\sigma^2} Tr(\mathbf{W}^T \mathbf{G} P Y) \\ &- \frac{1}{2\sigma^2} Tr(\mathbf{W}^T \mathbf{G} d(P\mathbf{1}) \mathbf{G} \mathbf{W}) - \frac{N_p D}{2} \log(\sigma^2) \\ &- \frac{\lambda}{2} Tr(\mathbf{W}^T \mathbf{G} \mathbf{W}) - \frac{\tau}{2} Tr(\mathbf{W}^T \mathbf{G}^T \Phi \mathbf{G} \mathbf{W}) \end{aligned} \tag{4.29}$$

where $\mathbf{G} \in \mathbb{R}^{N \times N}$ is a symmetric positive Gramian matrix with element $\mathbf{G}_{ij} = g(x_i - x_j^{t-1})$ and $\mathbf{W} = [w_1, \ldots, w_N]^T \in \mathbb{R}^{N \times D}$ is the vectorization of kernel weights in (4.22). $P \in \mathbb{R}^{N \times M}$ is the correspondence matrix with $P_{nm} = p(n|y_m)$. $\mathbf{1}$ is a column vector with all ones. $\Phi = \sum_{l=1}^{L} [S^{(l)T} S^{(l)}]$. $d(\cdot)$ is the diagonalization operation.

Proof 4 *In the modified likelihood* (4.21)*, there are three major terms: the original likelihood term, the local topology regularization term and the global topology regularization term. We will transform each term to get the formulation in* (4.29)*.*

First, the original likelihood term. From (4.9),

$$
\begin{aligned}
&Q(\mathcal{T}(x_n),\sigma^2) + \frac{N_p D}{2}\log(\sigma^2) \\
&= -\sum_{m=1}^{M}\sum_{n=1}^{N} p(n|y_m)\frac{\|y_m - \mathcal{T}(x_n)\|^2}{2\sigma^2} \\
&= -\sum_{m=1}^{M}\sum_{n=1}^{N} p(n|y_m)\frac{\|y_m^T - G(n,\cdot)^T W\|^2}{2\sigma^2} \\
&= -\frac{1}{2\sigma^2}\sum_{m=1}^{M}\sum_{n=1}^{N} p(n|y_m)[y_m^T y_m + G(n,\cdot)^T W W^T G(n,\cdot) - 2G(n,\cdot)^T W y_m] \\
&= -\frac{1}{2\sigma^2}\{\mathit{Tr}[Y^T d(P^T\mathbf{1})Y] + \mathit{Tr}[W^T G d(P\mathbf{1})GW] - 2\,\mathit{Tr}(W^T GPY)\} \qquad (4.30)
\end{aligned}
$$

Second, the global topology term:

$$
\begin{aligned}
E_{Global} &= \int_{\mathbb{R}^D} \frac{|T(s)|^2}{G(s)} ds \\
&= \int_{\mathbb{R}^D} \frac{T^T(-s)T(s)}{G(s)} ds \\
&= \int_{\mathbb{R}^D} \frac{1}{G(s)} \cdot [G(-s)\sum_{n=1}^{N} w_n^T e^{+2\pi i<x_n,s>}] \cdot [G(s)\sum_{n=1}^{N} w_n e^{-2\pi i<x_n,s>}] ds \\
&= \int_{\mathbb{R}^D} G(s)\sum_{i=1}^{N}\sum_{j=1}^{N} w_i^T w_j e^{+2\pi i<x_j - x_i,s>} ds \\
&= \sum_{i=1}^{N}\sum_{j=1}^{N} g(x_i - x_j) w_i^T w_j \\
&= \mathit{Tr}\sum_{i=1}^{N}\sum_{j=1}^{N} w_j \mathbf{G}_{ij} w_i^T \\
&= \mathit{Tr}(W^T G W) \qquad (4.31)
\end{aligned}
$$

Third, the local topology term:

$$\begin{aligned} E_{Local} &= \sum_{n=1}^{N}\sum_{l=1}^{L} \| \sum_{i=1}^{N} S_{ni}^{(l)} x_i^t \|^2 \\ &= \sum_{n=1}^{N}\sum_{l=1}^{L} Tr[\sum_{i=1}^{N} S_{ni}^{(l)} x_i^T \sum_{j=1}^{N} S_{nj}^{(l)} x_j] \\ &= \sum_{n=1}^{N}\sum_{l=1}^{L} Tr[X^T S^{(l)}(n,\cdot)^T S^{(l)}(n,\cdot) X] \\ &= \sum_{l=1}^{L} Tr[X^T S^{(l)T} S^{(l)} X] \\ &= Tr\{X^T \sum_{l=1}^{L} [S^{(l)T} S^{(l)}] X\} \\ &= Tr\{W^T G^T \Phi G W\} \end{aligned} \tag{4.32}$$

with

$$\Phi = \sum_{l=1}^{L} [S^{(l)T} S^{(l)}] \tag{4.33}$$

Combine the above three derivations together, then we can get the formulation as listed in (4.29).

End of proof.

With the linear formulation of (4.29), the optimal $\mathbf{W}$ and σ^2 can be calculated by taking $\frac{\partial Q}{\partial \mathbf{W}} = 0$ and $\frac{\partial Q}{\partial \sigma^2} = 0$:

$$\mathbf{W} = [d(P\mathbf{1})\mathbf{G} + \lambda\sigma^2\mathbf{I} + \tau\sigma^2\Phi\mathbf{G})]^{-1} PY \tag{4.34}$$

$$\begin{aligned} \sigma^2 = \frac{1}{N_p D} \{ &\mathrm{Tr}(Y^T d(P^T\mathbf{1})Y) - 2\mathrm{Tr}(W^T \mathbf{G} PY) \\ &+ \mathrm{Tr}(\mathbf{W}^T \mathbf{G} d(P\mathbf{1}) \mathbf{G}\mathbf{W})\} \end{aligned} \tag{4.35}$$

EM algorithm can also be performed to estimate the parameters iteratively. In E-Step, the posteriori probability distribution P is calculated using the estimated $(\mathbf{W}, \sigma^2)$ from the last M-step. In M-Step, a new estimate of $(\mathbf{W}, \sigma^2)$ is updated by executing the closed-form solution (4.34) and (4.35).

After $\tilde{Q}$ is converged, the state estimation at time step t can be updated as:

$$\bar{X} = \mathcal{T}(X) = \mathbf{G}(X)\mathbf{W} \tag{4.36}$$

As shown in Fig. 4.6(c), because the topological structure is successfully preserved, the new transformation with SPR is able to produce a reasonable registration under occlusions.

Implementation Acceleration

The above registration process requires multiple matrix operations, which might be slow especially when the point number N increases and the state dimension D augments. To accelerate the registration, both fast Gaussian transform (FGT) and low rank approximation will be introduced to reduce the computational complexity.

In the M-step in the EM updates, (4.34) and (4.35) involve the matrix-vector products $P\mathbf{1}, P^T\mathbf{1}$ and matrix-matrix products P^TY, where P is a variation of Gaussian affinity matrix. Normal production approach takes $\mathcal{O}(MN)$ operations, while with FGT acceleration, the complexity drops to linear $\mathcal{O}(M+N)$. The basic idea is to expand the Gaussians in terms of truncated Hermit expansion for fast computation of the sum of exponentials. More details of the FGT implementation can be found in [38] and [82].

Another bottleneck in our algorithm is the operation of matrix inversion. In (4.34), the matrix $[d(P\mathbf{1})\mathbf{G}+\lambda\sigma^2\mathbf{I}+\tau\sigma^2\Phi\mathbf{G})]$ with dimension $N\times N$ has to be inverted with complexity $\mathcal{O}(N^3)$. A trick of low rank approximation can be applied here to decrease the complexity. First, calculate the eigenvalues and eigenvectors of $\mathbf{G}$ and approximate it by $\mathbf{G}\approx M\Lambda M^T$, with $\Lambda\in\mathbb{R}^{\sqrt[3]{N}\times\sqrt[3]{N}}$ the diagonal matrix with first $\sqrt[3]{N}$ largest eigenvalues, and $M\in\mathbb{R}^{N\times\sqrt[3]{N}}$ the corresponding eigenvectors. Using Woodbury identity, we get

$$\begin{aligned}&(AM\Lambda M^T+\lambda\sigma^2\mathbf{I})^{-1}\\ =&\frac{1}{\lambda\sigma^2}[\mathbf{I}-AM(\Lambda^{-1}+\frac{1}{\lambda\sigma^2}M^TAM)^{-1}M^T]\end{aligned} \tag{4.37}$$

where $A=d(P\mathbf{1})+\tau\sigma^2\Phi$.

In (4.37), the complexity of matrix inversion drops to be linear $\mathcal{O}(N)$.

4.5 Experimental Study

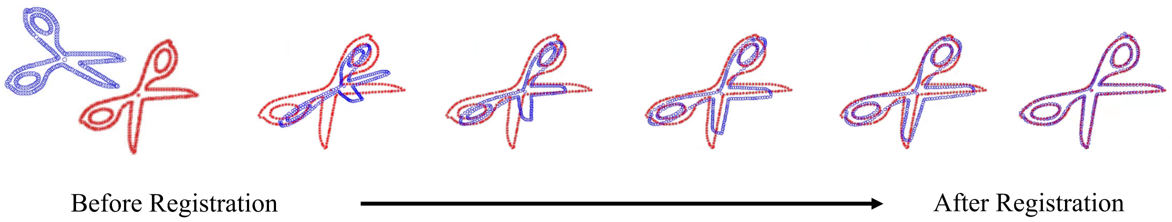
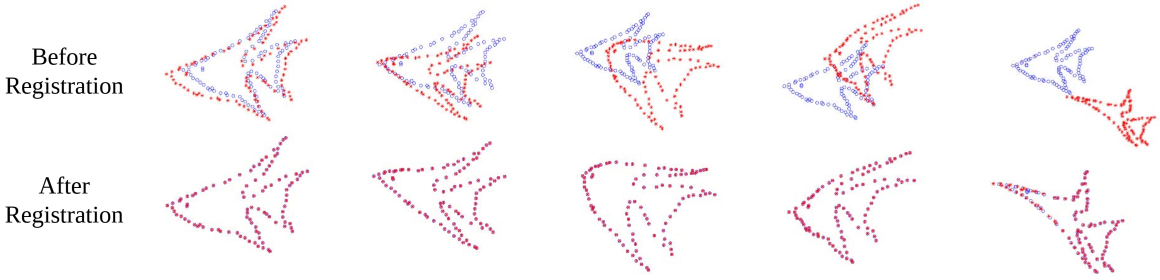
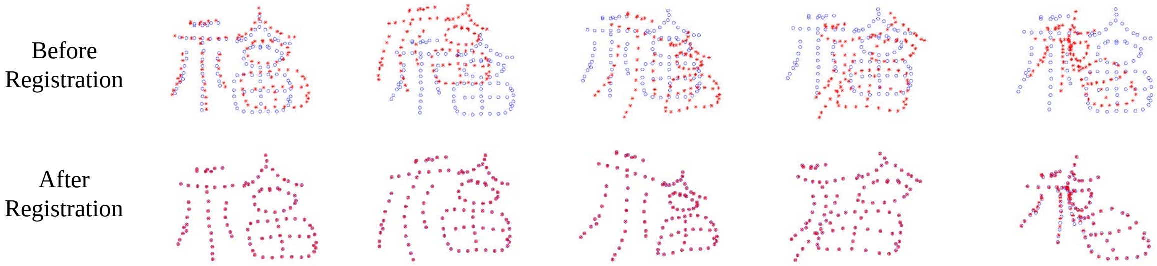


Figure 4.8: Procedure of SPR Registration. The blue scissors are registered towards the red one gradually by SPR.

Figure 4.8 shows the iterative procedure of registering the blue scissors to a red one by SPR. To have a quantitative comparison with other registration methods, we test the performance of SPR along with those state-of-the-art algorithms, including TPS-RPM [22], CPD [82], GLTP [36], on the commonly used CHUI dataset [23].



(a) Fish



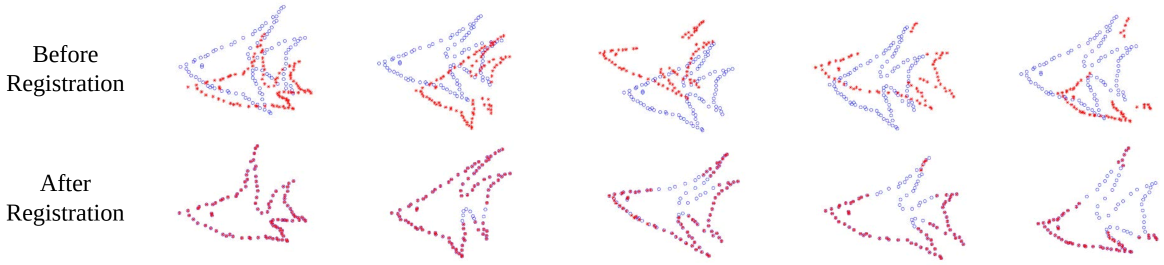
(b) Character

Figure 4.9: Registration result with different levels of deformations. From left to right, the deformation is increasing.

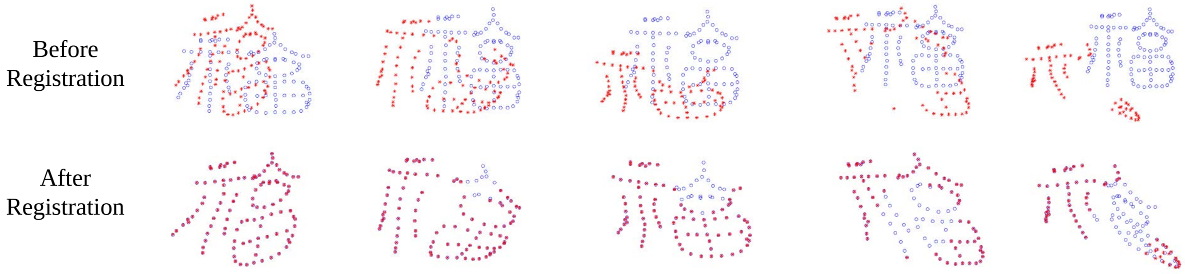
Three benchmark tests are performed, including registration with different levels of deformation, occlusions and outliers. For all the tests, the input point sets were first normalized to zero mean and unit variance before registration. The weight μ for uniform distribution was chosen to be 0.1. For global topology regularization, the regularization weight λ and Gaussian kernel's variance β were set as 3.0 and 2.0 respectively. For local structure regularization, the number of nearest neighbor is set as $K = 5$, and its regularization weight is $\tau = 10$. All the point sets were denormalized after registration. Regarding other methods, the default parameters in their papers are utilized, considering they used the same dataset as ours.

As shown in Fig. 4.9, the source point set (blue dot) is desired to be aligned towards the target point set (red dot). From left to right, the target fish and Chinese characters are twisted with larger deformation levels. The proposed SPR algorithm can generate appropriate transformation function and well align the source to the target at different deformation levels.

Fig. 4.10 shows the registration results under occlusions. Note that the target objects (red dots) are partially occluded on purpose, which results in missing points in the point sets. Because SPR maintains the topology of the object during registration, in the red point missing area, the blue points can still be reasonably registered. In the last column of Fig. 4.10, 50% of red points are removed, which shows the registration robustness under



(a) Fish



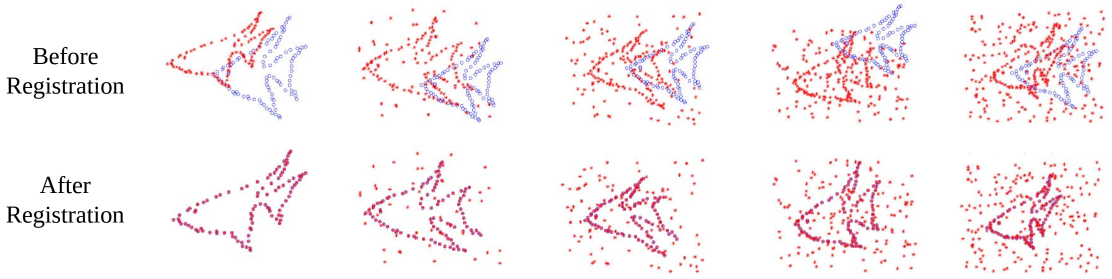
(b) Character

Figure 4.10: Registration result with different levels of occlusions. From left to right, more points are dropped in the source point set.

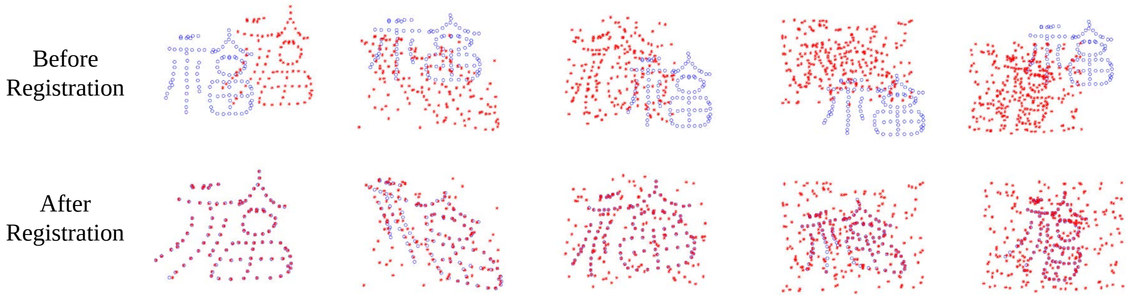
massive occlusions.

Similarly, registration is robust when outlier points exist. As shown in Fig. 4.11, dense outliers exist and contaminate the target point set. In the last column, the number of outliers is larger than that of objects, whereas an accurate alignment between the point sets can still be found.

Fig. 4.12, Fig. 4.13 and Fig. 4.14 show the quantitative comparison between SPR and other methods (TPS-RPM, CPD, GLTP) under different levels of deformation, occlusions and occlusions, respectively. In the CHUI dataset [23], there are 100 tests for each level of deformation, occlusions and outliers. The average registration error (Euclidean distance between the transformed source point set and the ground-truth target point set) of the 100 tests are calculated. Note that since TPS-RPM has a large error under occlusions, for the plotting convenience, TPS-RPM is not included in Fig. 4.13. It is shown that SPR has the best result in all the three benchmark tests. The outperformance is more obvious when there are larger levels of deformation, occlusions and outliers, which suggests that in those circumstances, the regularizations on local structure and global topology play a significant role and provide a more robust registration between point sets.



(a) Fish



(b) Character

Figure 4.11: Registration result with different levels of outliers. From left to right, more outlier points contaminate the target point set.

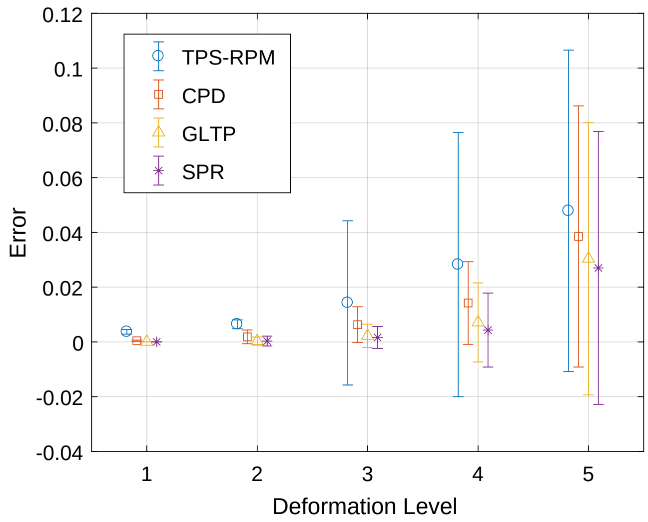


Figure 4.12: Registration error under deformation.

4.6 Chapter Summary

A new robot learning approach, named analogy learning is proposed in this chapter. Instead of finding a control policy to relate the system states and corresponding actions, analogy

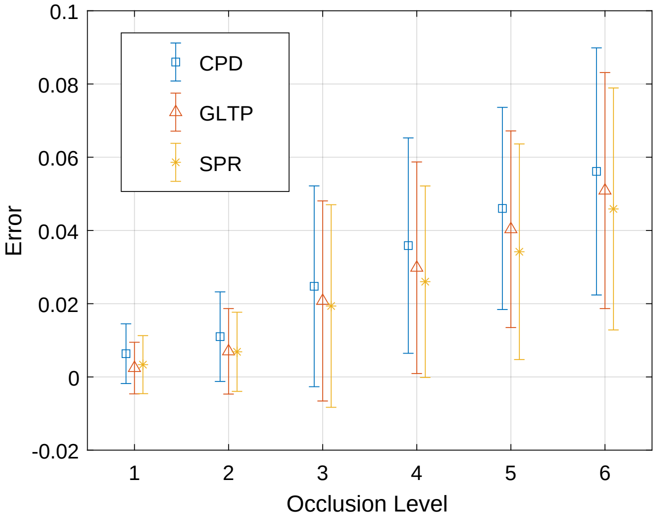


Figure 4.13: Registration error under occlusions.

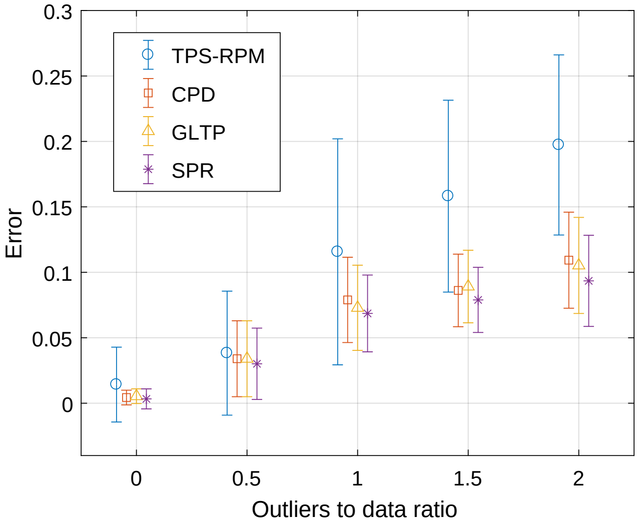


Figure 4.14: Registration error under outliers.

leaning tries to find correlation between current scenario with past training scenarios. A past scenario that bears a strong similarity with the current will be identified (reminding) and the past actions will be transferred to generate a new feasible action (transferring). The benefits of analogy learning, including efficiency in data learning, robustness to input noise and collision-free guarantee are discussed in detail.

To support the reminding and transferring procedures of analogy learning, a novel non-rigid registration method, named structure preserved registration, is developed. It regards the past and current scenarios as point sets, and find a smooth transformation function $\mathcal{T}$ to register them to each other. The similarity between two point sets can be quantitatively analyzed based on the likelihood value Q. The performance of registration is illustrated by a series of experiments to show its robustness under massive levels of deformation, outliers and occlusions.

Chapter 5

Robotic Grasping by Analogy Learning

With the concept of analogy learning proposed in Chapter 4, this chapter will apply this methodology to teach industrial robots grasping skills. In general, given the point cloud of a test object, the robot will first check its similarity with all the training objects. The most similar training object is then identified and a spatial transformation function will be constructed by structure preserved registration. With the function, the grasping pose on the training object can be transferred to achieve a new grasping pose that is suitable for the test object. Experimental results show that the proposed grasping method is efficient and robust with 94% success rate. Moreover, only a limited amount of training data is required for training the grasping skill.

5.1 Introduction

Grasping is an essential capability for robots to accomplish complex manipulation tasks. In recent years, more and more applications require robots to grasp various objects with general purpose grippers. For example, human-robot interaction requires collaboration and assistance between robots and humans, during which robots may pass different tools to humans or help to hold various workpieces for assembly tasks. The increasing demand for massive customization and warehouse automation also promotes the development of dexterous grasping.

However, the grasping planning for various objects is challenging to address due to heavy computational loads, large task variance and imperfect perceptions. Many model-based analytic planners, such as Ferrari-Canny metric [34] and grasping isotropy [51], require considerable time for searching and heavy computation for evaluation. Besides, these planners generally analyze contact quality based on local features such as contact position and contact normal, while the global task requirements such as robot reachability and collision avoidance are not under consideration. Moreover, analytic planners are usually sensitive to noises and distortions of point clouds caused by hardware limitations and calibration errors. Therefore, the grasping quality evaluated by analytic planners is sometimes inconsistent with the empirical success rate and cannot resemble reality effectively [12].

Another popular approach for grasping planning is model-free leaning, i.e., learning and predicting optimal grasping from previous grasping examples. For example, the Dex-Net [76, 75] trains a deep neural network from an enormous database which contains 2.8 million grasping examples built by analytic planners. The network is able to estimate optimal grasping for unseen objects after training. In [113], the grasping pose is calculated from heatmaps that generated by a trained neural network. These methods, however, usually require considerable data for the training process and the optimal grasping is planned without considering the task constraints, such as robot reachability.

Despite the variance of object shapes, we notice that objects to grasp can be classified into several categories. For example, in the tool picking scenario, objects can often be specified into categories such as wrenches, pliers, screwdrivers, etc. Objects in each category may have difference in shapes and sizes, but in general they share similar topological structures.

Some researches have been conducted based on this observation. In [15], the perceived cloud of the object is fitted to different objects templates in the database, and the grasping is estimated by superimposing all representations considering their confidence levels. A semantic grasping is proposed in [25] to consider task requirements. The task constraints are implicitly represented by a grasping example in each object category and the desired grasping pose on the novel object is retrieved by mapping the grasping example and refined by eigen-grasp planner. A dictionary of object parts is learned in [28] to generate grasping poses across partially similar objects. The dictionary assumes that the segments that shared by objects are rigid and have similar sizes. However, this assumption cannot hold in many scenarios.

In this chapter, we propose a novel framework [1] for efficient and effective grasping generation from previous grasping examples. First, a series of feasible grasping poses on training objects will be demonstrated by human experts. In the test stage, the category of the test object will be classified by its similarities towards the training objects. Then a grasping pose transferring is performed between similar objects based on the concept of analogy learning [4]. Moreover, the transformed poses will be rated by analyzing the grasping isotropy metric [51]. An orientation search method will also be introduced to improve the robot reachability and avoid collisions.

The remainder of this chapter is organized as follows: Section 5.2 introduces the normal formulation of grasping problems and the benefits of involving human demonstration. The application of analogy learning on grasping pose transferring is introduced in Section 5.3. The dissimilarity measure between objects and the refinement of poses after transferring are also presented. A series of experiments on grasping multiple categories of objects are shown in Section 5.4. Experimental videos can be found in [104]. Section 5.5 provides conclusions

---

[1] This work is published in [67] and equally contributed with Dr. Hsien-Chung Lin. Similar contents will also be included in his dissertation.

and proposes future works.

5.2 Grasp Planning with Human Demonstration

A general grasp planning problem with parallel grippers can be formulated as

$$\max_{\boldsymbol{c},\boldsymbol{n_c}} Q(\boldsymbol{c},\boldsymbol{n_c}) \tag{5.1a}$$

$$s.t. \quad c_i \in \partial O \quad i=1,2 \tag{5.1b}$$

$$\|c_1 - c_2\| \leq w_{\max}, \tag{5.1c}$$

where Q denotes the grasping quality to be maximized, $\boldsymbol{c} = \{c_1, c_2\}$ with $c_i \in \mathbb{R}^D$ denote the positions of the two contact points, and $\boldsymbol{n_c} = \{n_{c,1}, n_{c,2}\}$ denotes the normals of the contact pair with $n_{c,i} \in \mathbb{S}^{D-1}$. Constraint (5.1b) shows the contacts should lie on the surface of object ∂O, and (5.1c) shows that the distance of the contact pair should be less than the width of the gripper $w_{\max}$.

Equation (5.1) is challenging to solve by gradient based methods because of the high complexity of surface modelling, and the discontinuity of surface presentations as well as surface normals. Compared with gradient based searching, the sampling based method is able to adapt to discrete object representation and escape from local optimum. However, it requires considerable computation for sampling and quality evaluation to find a reasonable grasp due to the complicated structure of the object and the feasibility constraints such as gripper width, task requirements and collisions, thus the direct sampling method is generally not affordable for real-time implementation.

In this chapter, we assume that the objects to grasp can be clustered into various categories. The objects in the same category share similar topological structures but can have different shapes, sizes and configurations. The objective of this work is to provide an efficient framework to grasp objects in the same category without overwhelmed training, modelling and computation. To achieve this, we introduce human demonstration to accelerate grasp searching by providing heuristics to guide sampling. Instead of directly using human demonstration as the sampling pool for the target object to grasp, we use a mapping function to transfer the example grasps based on the topological similarity between the source object and the target object. Therefore, (5.1) becomes:

$$\max_{\boldsymbol{c},\boldsymbol{n_c}} Q(\boldsymbol{c},\boldsymbol{n_c}) \tag{5.2a}$$

$$s.t. \quad \{\boldsymbol{c},\boldsymbol{n_c}\} \in map(\mathcal{H}) \tag{5.2b}$$

$$\|c_1 - c_2\| \leq w_{\max}, \tag{5.2c}$$

where $\mathcal{H}$ denotes a human demonstration database containing example grasps on the source object, and the function $map(\cdot)$ represents a grasp transferring. Compared with (5.1), the introduction of human demonstration in (5.2) has the following advantages. First, incorporating human intelligence into the framework will improve the empirical success rate, since

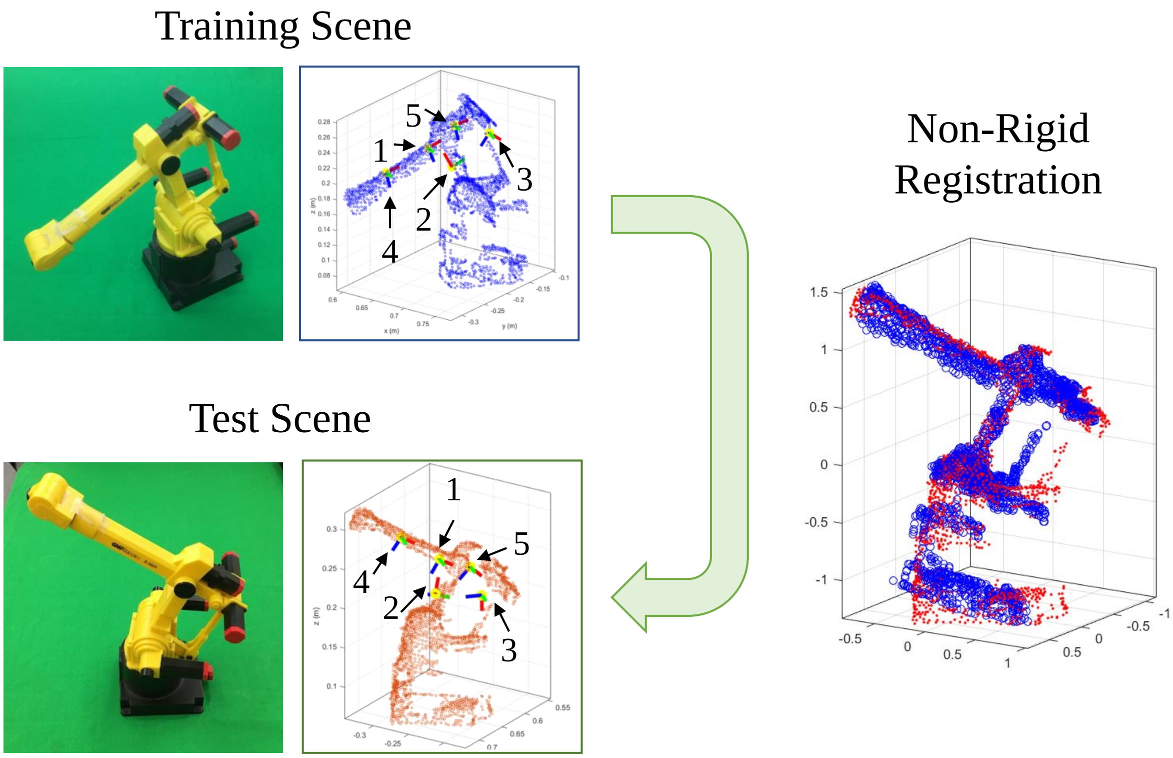


Figure 5.1: Grasping pose transferring. The point clouds of the training object is registered to the test object by non-rigid registration. A transformation function can be constructed and utilized to transfer the training grasping poses to get new ones which are feasible for the test object.

the human demo usually considers a variety of factors such as the local structure of the object and the global geometry for collision avoidance. Second, some tasks have special requirements. For example, some workpieces have fragile parts or polished surfaces which are not suitable for grasping. Some workpieces have some preferred grasping poses for the ease of following assembly procedures. Explicitly imposing such constraints to traditional approaches is nontrivial, while these requirements can be easily encoded by human demonstration. Moreover, by mapping the grasp examples to novel objects, the proposed method exploits much fewer but reasonable grasp samples compared to traditional exhaustive search methods. Therefore, the searching time is greatly reduced.

5.3 Grasping Pose Transferring by Analogy Learning

Assume a grasp template consists of a source object and multiple demonstrated grasping poses. A grasping pose is defined by the grasping position on the object and the orientation of the gripper. As Fig. 5.1 shows, the blue dots are the object point clouds and each coordinate represents a demonstrated grasping pose.

Denote the point clouds of the source object as $\mathbf{X} = (x_1, \cdots, x_N) \in \mathbb{R}^{N\times 3}$, where $x_n \in \mathbb{R}^3$ is the n-th point in the point set. The demonstrated grasping poses are denoted as $\boldsymbol{g}_i = (\boldsymbol{t}_i, \boldsymbol{R}_i) \in \mathbb{R}^3 \otimes \mathbf{SO}(3)$, $i = 1, 2, \cdots, I$, where $\boldsymbol{t}_i \in \mathbb{R}^3$ is the center of the grasping point, $\boldsymbol{R}_i \in \boldsymbol{SO}(3)$ represents the grasping orientation, and i is the index among the total I grasping poses. The target object is represented by another point set $\mathbf{Y} = (y_1, \cdots, y_M) \in \mathbb{R}^{M\times 3}$, where $y_m \in \mathbb{R}^3$ is the m-th point in the target point set. Our objective is to find a smooth transformation $\mathcal{T} : \mathbb{R}^3 \to \mathbb{R}^3$ that maps the source object to the target object as well as transferring the grasp examples to new grasping poses $\boldsymbol{g}'_i = (\boldsymbol{t}'_i, \boldsymbol{R}'_i)$ on the target object (Fig. 5.1).

Grasping Pose Transferring

With the structure preserved registration (SPR) algorithm proposed in Section 4.4, the desired transformation function $\mathcal{T}$ can be found by registering the two point sets. The topological structure of point sets is preserved during the alignment process so that the grasping poses can be transferred to reasonable locations.

As shown in Fig. 5.1, after finding the mapping $\mathcal{T}$, the demonstrated grasping poses on $\mathbf{X}$ will be transferred to achieve new grasping poses that are suitable for object $\mathbf{Y}$. The procedures for pose transferring can be decomposed to two parts. First, for grasping position transferring, the original grasping position can be directly mapped by function $\mathcal{T}$:

$$\boldsymbol{t}'_i \leftarrow \mathcal{T}(\boldsymbol{t}_i), \quad i = 1, 2, \cdots, I. \tag{5.3}$$

Second, for the grasping orientation transferring, we evaluate the Jacobian matrix of $\mathcal{T}$ (first-order partial derivatives) and utilize it to transform the original orientation to achieve a new one: $\nabla\mathcal{T}(\boldsymbol{t}_i)\boldsymbol{R}_i$. Singular value decomposition (SVD) is further applied to refine the orientation matrix to belong to $\mathbf{SO}(3)$ group. Specifically, the singular vectors U_i, V_i are extracted by

$$\mathbf{U}_i \Sigma \mathbf{V}_i^T = svd(\nabla\mathcal{T}(\boldsymbol{t}_i)\boldsymbol{R}_i) \tag{5.4}$$

The orientation matrix is recalculated as

$$\boldsymbol{R}'_i \leftarrow \mathbf{U}_i \mathbf{V}_i^T, \quad i = 1, 2, \cdots, I. \tag{5.5}$$

Dissimilarity Measure

During the training stage, multiple grasping poses for different categories of objects are demonstrated by human experts. Given a new object at test, it is necessary to first classify which category the object belongs to, then use structure preserved registration to transfer the corresponding grasping poses from the correct category to get a new feasible grasp. Therefore, an object classifier is essential for pose transferring.

There are some researches that apply the surface matching technique to rigidly fit the object template to the measured point clouds and calculate the dissimilarity [89, 52]. The source objects from different categories are exploited as the templates to match the target object. By measuring the dissimilarity between each of the source objects $\mathbf{X}$ and the target object $\mathbf{Y}$, the most similar pair will be selected to determine the category of the target object. In our work, since SPR can be applied to warp the template $\mathbf{X}$ to $\mathcal{T}(\mathbf{X})$ which is aligned with $\mathbf{Y}$, the residual dissimilarity between $\mathcal{T}(\mathbf{X})$ and $\mathbf{Y}$ instead of the dissimilarity between $\mathbf{X}$ and $\mathbf{Y}$ will be checked to provide a more robust category classification.

The average minimum distance between the two point sets can be designed as:

$$d(\mathcal{T}(\mathbf{X}), \mathbf{Y}) = \frac{1}{N} \sum_{n=1}^{N} \min_{m \in [1,M]} ||\mathcal{T}(x_n) - y_m||, \tag{5.6}$$

where $||\mathcal{T}(x_n) - y_m||$ is the Euclidean distance between point $\mathcal{T}(x_n)$ and y_m. However, (5.6) is asymmetric. We formuate the the dissimilarity between a source object and a target object to be

$$D(\mathbf{X}', \mathbf{Y}) = d(\mathbf{X}', \mathbf{Y}) + d(\mathbf{Y}, \mathbf{X}'), \tag{5.7}$$

where $\mathbf{X}' = \mathcal{T}(\mathbf{X})$ and $D(\cdot, \cdot)$ is symmetric to its input arguments.

Suppose there are K object categories, the most possible category that the target object belongs to can be estimated by

$$\mathbf{k}^* = \arg \min_{k \in [1,K]} D(\mathbf{X}'_k, \mathbf{Y}). \tag{5.8}$$

Grasping Pose Selection

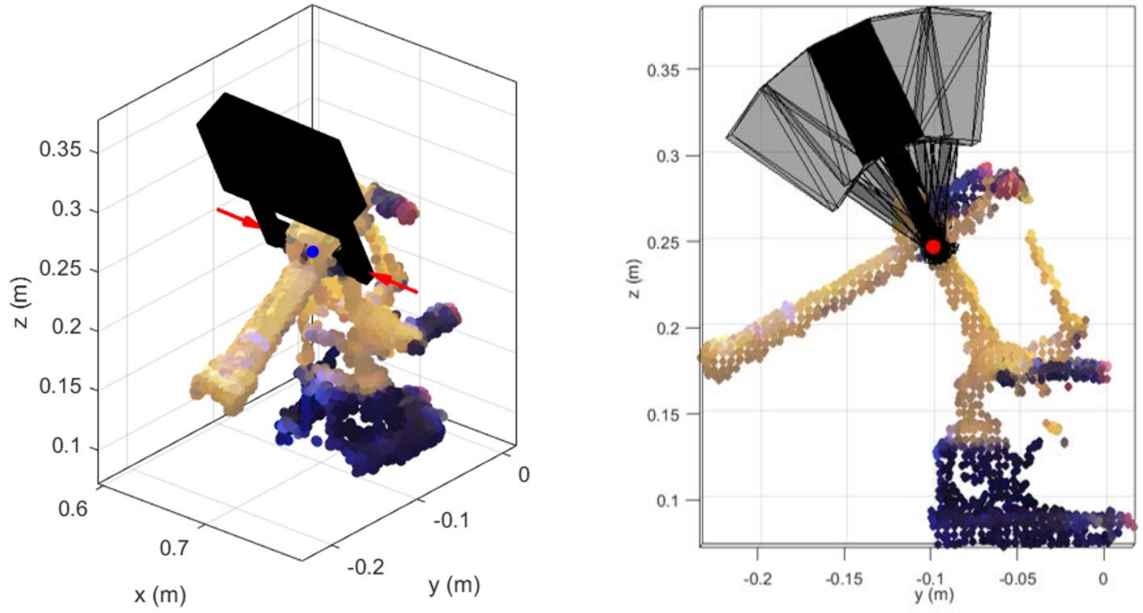
Once the object category is determined, we can map the example grasping poses from the corresponding training object to the target object.

The transformed poses serve as good candidates for grasping the test object. We will re-evaluate the quality of each transformed pose and select the best one for the robot to grasp.

The grasp quality of the transformed poses can be evaluated by analytic methods using the grasp isotropy index [51]. The grasp isotropy index measures the uniformness of different contact forces to the total wrench. More concretely, it can be written as

$$Q_i = \frac{\sigma_{\min} \mathcal{G}(\boldsymbol{g}'_i, \boldsymbol{g}_o)}{\sigma_{\max} \mathcal{G}(\boldsymbol{g}'_i, \boldsymbol{g}_o)}, \tag{5.9}$$

where $\boldsymbol{g}_o$ denotes the pose of the object, $\mathcal{G}(\boldsymbol{g}'_i, \boldsymbol{g}_o)$ represents the grasp map determined by the contacts and the object [81], and $\sigma_{\min}$ and $\sigma_{\max}$ respectively denote the minimum and maximum singular values of the grasp map. The contacts are inferred by the line search along the grasp axis. The line search tries to locate the nearest neighbor of the grasp center



(a) (b)

Figure 5.2: Grasping pose modification. (a) a grasp example, where the red arrows indicate the direction of gripper closing (which is also the grasp axis). (b) The side view of the grasp example, and the transparent grippers are shared the same grasp center and grasp axis but different orientations.

on the object's point cloud. The contacts are represented by the nearest neighbors search in the positive and negative directions of the grasp axis respectively. The transferred grasp would be treated as a bad pose if the contacts deviate from grasp axis too much, in which case a negative quality will be allocated.

Apart from the grasp quality, we also consider the feasibility constraints such as the reachability and the gripper-object collision.

A grasping example with parallel grippers is shown in Fig. 5.2a, where the blue dot is the center of grasping and the red arrows represent the grasp axis which is parallel to the operational direction of the gripper. Rotating along the grasp axis will not change the grasping quality analyzed by (5.9). However, new grasping poses can be generated, from which we can search for a valid one which is collision-free and reachable for the robots (Fig. 5.2b).

Suppose the initial orientation is denoted as $\boldsymbol{R}_0$, the sampled orientation is denoted as $\boldsymbol{R}_i$, and $\mathcal{R}$ is the set of all the sampled orientations. The orientation search can be formulated as

$$\min_{\boldsymbol{R}_i \in \mathcal{R}} \ \Delta\xi(\boldsymbol{R}_0, \boldsymbol{R}_i) + C\left[f_{IK}(\boldsymbol{t}, \boldsymbol{R}_i) + f_{col}(\boldsymbol{t}, \boldsymbol{R}_i, \mathbf{Y})\right], \tag{5.10}$$

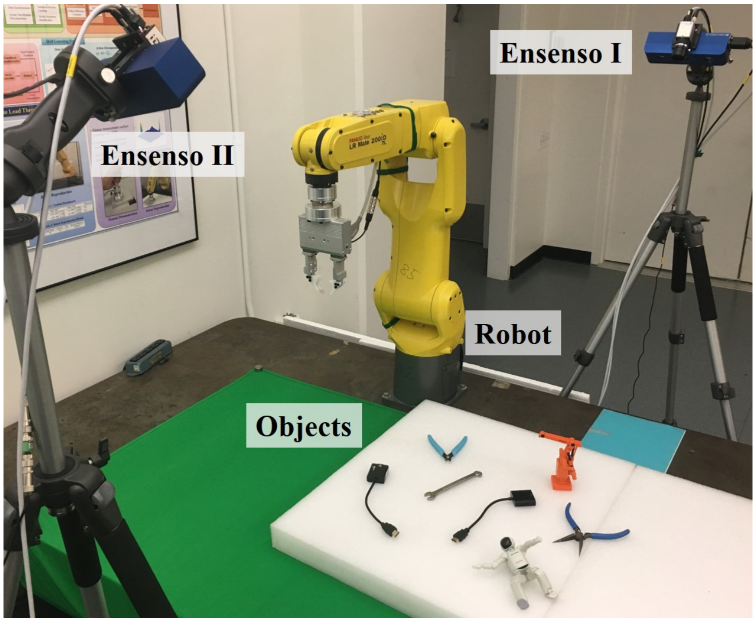


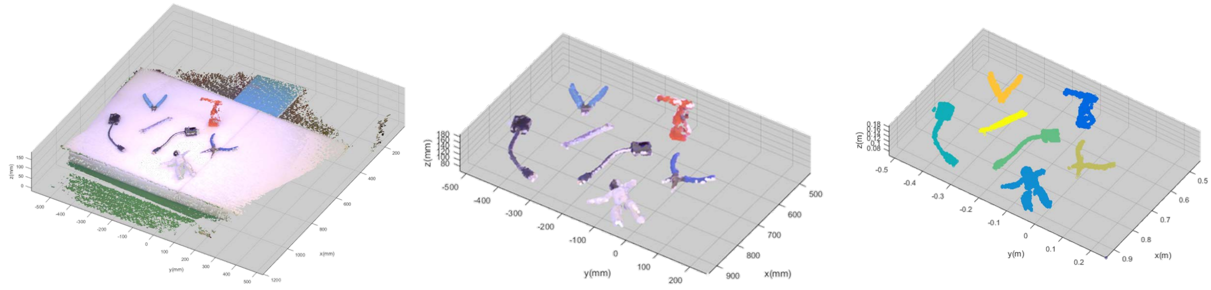
Figure 5.3: The experimental setup: a FANUC LR Mate 200iD/7L and dual Ensenso stereo cameras

where $\Delta\xi(\boldsymbol{R}_0, \boldsymbol{R}_i) = 1 - \xi(\boldsymbol{R}_0)^T \xi(\boldsymbol{R}_i) \in [0, 1]$ is the rotation deviation in quaternion between $\boldsymbol{R}_0$ and $\boldsymbol{R}_i$, $\xi(\cdot)$ converts a rotation matrix to a quaternion. We use quaternion rather than Euler angles to represent rotation difference to avoid singular representation in rotations. $f_{IK}(\boldsymbol{t}, \boldsymbol{R})$ is a boolean function that returns 1 when the inverse kinematics of $(\boldsymbol{t}, \boldsymbol{R})$ is invalid and returns 0 otherwise. $f_{col}(\boldsymbol{t}, \boldsymbol{R}, \mathbf{Y})$ is another boolean function which returns 1 when the gripper with pose $(\boldsymbol{t}, \boldsymbol{R}_i)$ is collided with $\mathbf{Y}$ while returns 0 otherwise. C is a large constant weight to penalize the conditions of both infeasible inverse kinematics and gripper-object collision. If all the sampled orientations are invalid, the value of (5.10) will be greater than or equal to C. Then the orientation search is applied to exploit the other candidates until it finds a feasible grasping pose to perform the task.

5.4 Experimental Study

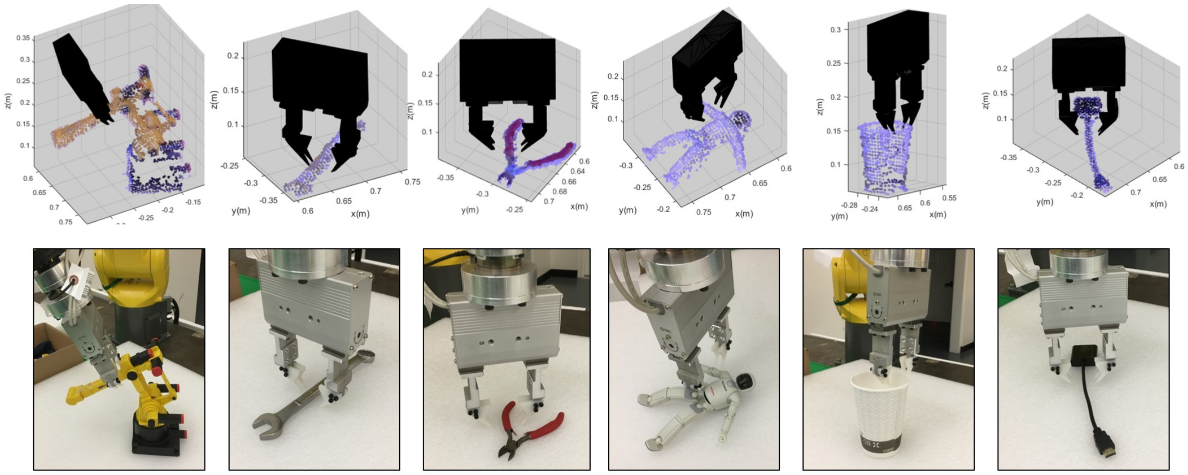
In order to verify the proposed grasping approach, a series of experiments were conducted to grasp various objects by an industrial robot manipulator. The experimental setup is shown in Fig. 5.3, where the robot was FANUC LR Mate 200iD/7L, and two stereo cameras (Ensenso) were calibrated and synchronized to capture the point clouds of objects in the workspace. All the programs were implemented in MATLAB on a Windows desktop with an Intel Core i5 CPU and 16GB RAM. The robot controller was deployed on a Simulink RealTime target.

The point clouds retrieved from the dual Ensenso stereo cameras were shown in Fig. 5.4a.



(a) (b) (c)

Figure 5.4: Process of point cloud. (a) The raw data was captured by the dual Ensenso stereo camera. (b) The objects were extracted from the background by predefined region. (c) The point cloud was clustered by DBSCAN.



(a) Manipulator (b) Wrench (c) Plier (d) Humanoid (e) Cup (f) Adapter

Figure 5.5: Grasp examples: the first row shows one of the grasping pose on each source object, and the second row provides the snapshots of the actual demonstrated grasping poses.

By applying the snapshot of the empty workspace as a filter mask, the point clouds of objects were extracted from the background as shown in Fig. 5.4b. By running the algorithm of ‘density-based spatial clustering application with noise’ (DBSCAN, [31]), the point clouds can be separated to several clusters to represent different objects (Fig. 5.4c). A voxel grid filter with step size 5mm was further implemented to uniformly downsample the point clouds.

Six categories of objects, including cups, pliers, wrenches, cable adapters, toy manipulator models and toy humanoid models, were tested in the experiment (Fig. 5.5). Note that neither CAD models nor mesh files were required in this work. For each category, a specific source object was selected, and the human operator taught multiple preferred grasping poses on it by lead through teaching. The point cloud of the object and the demonstrated grasping

Figure 5.6: Target objects, which are similar to the source objects but different in size, shape, and configuration.

poses were recorded as training database.

At the test stage, objects with different sizes and configurations across all the categories were randomly placed in the workspace. For example, multiple types of cups and wrenches were tested for grasping; the pliers were either open or closed; the cable adapter was twisted to various shapes; the joints of the two toy manipulator models and the toy humanoid model were rotated to random angles. All the target objects were shown in Fig. 5.6.

Before running the grasping experiment, an object classification test was performed by measuring the dissimilarity between the target object and all the source objects. The target objects in Fig. 5.6 were randomly placed, with each category of objects collecting 20 different configurations. The parameters of SPR registration were set as $\beta = 2$, $\lambda = 50$, $\mu = 0.1$, $K = 5$ and $\tau = 1e-4$.

As shown in Fig. 5.7, the performance of object classification was presented by a confusion matrix, where each column represented the predicted class and each row represented the actual class. The diagonal entries of the confusion matrix indicated the correct classification, whereas the off-diagonal entries were misclassification. The overall classification accuracy was 94.17% (113/120).

Each category of objects was tested 20 times for grasping with different orientations, shapes, sizes, and configurations. The parameters of SPR were the same as the ones in object classification.

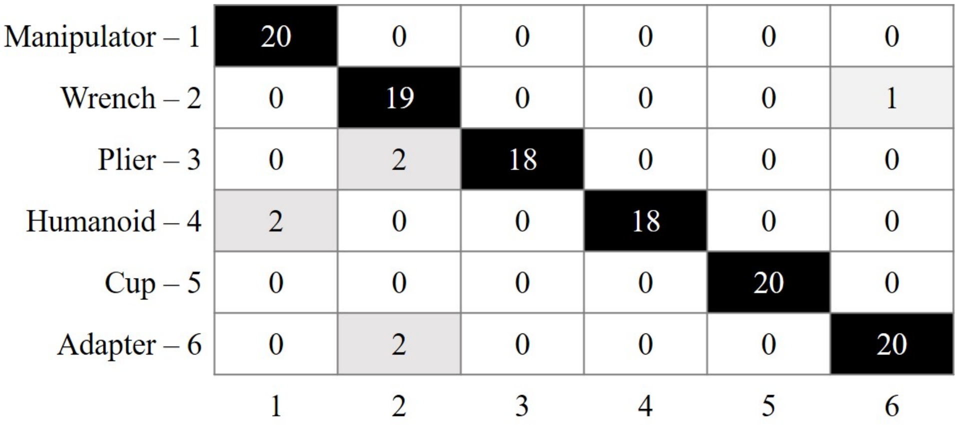
Take one target object (Fig. 5.1) as an example. The grasping qualities of the transferred grasps are provided in Table 5.1. Note that the qualities of the second and fifth transferred

Table 5.1: Grasping quality evaluation

Grasping Pose No.	1	2	3	4	5
Isotropy Index	0.0098	-1.000	0.0001	0.0089	-1.000

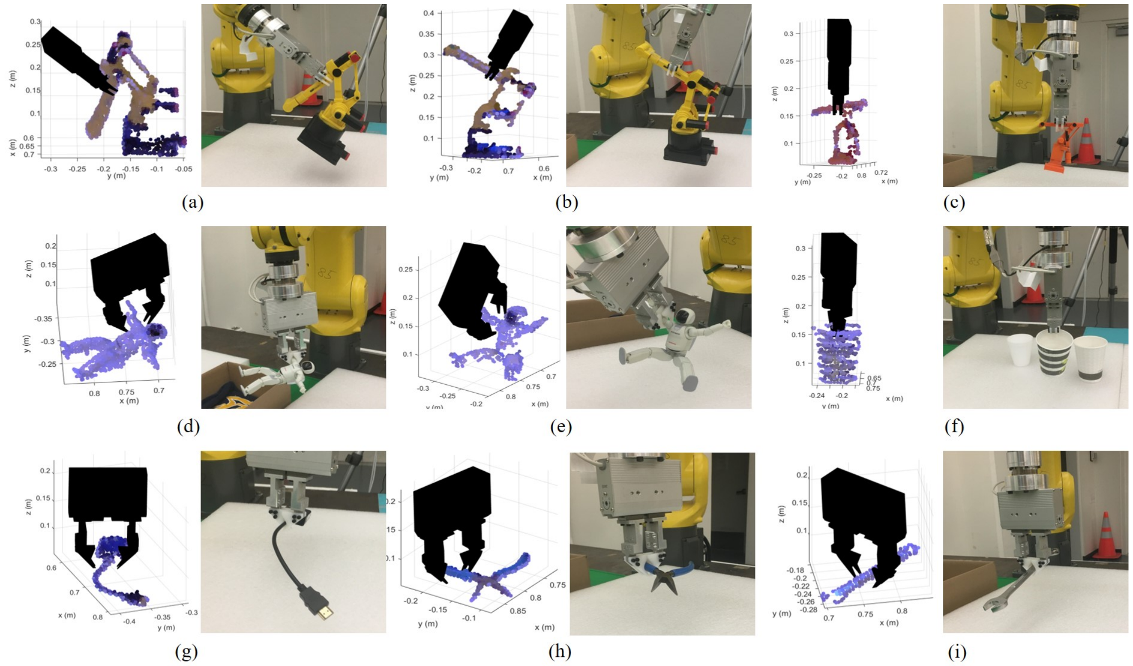
Table 5.2: Grasping results

class	success/trials	avg. SPR time (ms)	avg. numbers of points
manipulator	19/20	1276.4	1563.7
wrench	20/20	111.3	316.7
plier	18/20	706.1	1419.0
humanoid	17/20	369.5	773.3
cup	20/20	350.5	609.3
adapter	19/20	480.2	917.0
average	18.8/20	549.0	933.2



	1	2	3	4	5	6
Manipulator – 1	20	0	0	0	0	0
Wrench – 2	0	19	0	0	0	1
Plier – 3	0	2	18	0	0	0
Humanoid – 4	2	0	0	18	0	0
Cup – 5	0	0	0	0	20	0
Adapter – 6	0	2	0	0	0	20

Figure 5.7: The confusion matrix of object classification. Each column represents a predicted class, and each row represents a actual class.



(a) (b) (c) (d) (e) (f) (g) (h) (i)

Figure 5.8: The planned grasping poses and the corresponding snapshots of the grasping results.

poses are marked as negative based on the isotropy index analysis, since the second pose was mapped to a region with sparse points, and the contacts for the fifth grasp was wider than the width of the gripper. The remaining pose with the highest grasping quality, i.e., the first pose, was selected. The selected pose was then refined by the orientation search to improve the reachability and avoid collision. The final grasp performed in the experiment is shown in Fig. 5.8b. The grasp was regarded as success when the object could be robustly lifted up at least 10 cm without slipping. The success rate, average computation time and average point numbers for each category of objects are provided in Table 5.2. The experimental video can be found at [104]. The snapshots of grasping experiments are shown in Fig. 5.8.

Although the shapes and configurations of target objects were different to the ones of the source object, they shared the similar structures. Therefore, the grasping poses on the source object could be transferred to reasonable locations on the target objects. For instance, the grasping poses on the various toy manipulator models were invariant in terms of topological structures (see the first row of Fig. 5.8). The grasping poses taught by lead through teaching had the intuition from humans such as the task specific consideration and fairly good grasping quality, and SPR transferred the insight to the target objects. Therefore, the test can be successful in most of the cases.

The failure case happened when there was a very large distortion to transform the source object to the target, which degraded the accuracy of the transformation estimated by SPR. As a result, the grasping pose was not accurately transformed, which caused the grasp failed. Although SPR did not transfer grasping poses with high accuracy in this situation, it provided a relatively close one. In the future, we may include an adaptation on the warped grasping pose to avoid this failure.

5.5 Chapter Summary

This chapter proposed a framework for teaching robots grasping skills based on the analogy learning approach. A database containing multiple categories of source objects together with demonstrated grasping poses were first constructed by human experts. During the test scenario, a novel object was first classified into one of the example categories by measuring its dissimilarity to each source object. Then the grasping poses on the most similar source object were transferred to the novel object by the structure preserved registration (SPR) method. The qualities of all the transferred grasping poses were evaluated and sorted by the grasp isotropy metric. The selected pose was further refined by an orientation search mechanism, which improves the robot reachability and avoids collision. A series of experiments were performed to grasp six categories of objects with various shapes, sizes and configurations. The average success rate was 18.8 out of 20 grasp trials. The experimental video is available at [104].

Chapter 6

Track and Manipulate Deformable Objects by Analogy Learning

Manipulating rigid objects has been studied for a long period in robotic society. However, manipulating deformable objects by robots, such as cables and clothes, has not been extensively explored. Based on the concept of analogy leaning, this chapter will present a novel framework for deformable object manipulation, including state estimation, task planning and motion planning.

6.1 Introduction

Manipulating deformable objects by robots has a wide range of applications. For example, industrial robots can automate wire harness for electronic equipment; surgical robots assist surgeons to suture wounds; nursing robots help the elderly to wear or fold clothes; and so on. However, this manipulation problem remains challenging and under-explored. The major difficulty lies in that these deformable objects have high degrees of freedom which are expensive to model, track and control.

Take the rope knotting task as an example (Fig. 6.1). The objective is to manipulate the rope from a random initial state to a desired knotted state. Robots need to generate necessary motions to manipulate the rope based on the observation of current rope states. This task has many challenges in several aspects, especially in state estimation, task planning and motion planning.

First, for state estimation, the position of each rope segment needs to be identified from 3D camera measurements (point clouds). Usually the rope we are tracking is featureless. In other words, there are no distinguishable markers or features to recognize each segment, and it is unknown that which segment on the rope generates the measured points in the point clouds. This missing correspondence makes traditional visual tracking algorithms, for instance Kalman filter, unable to execute. Besides, since the rope is occluded by robot arms or self-occluded frequently during manipulation, the state estimator should be specially

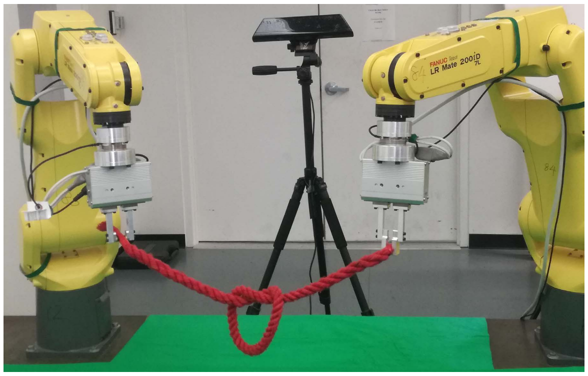


Figure 6.1: Two robots knotted a soft rope with real-time visual feedback.

designed to handle occlusion robustly. Moreover, considering the curse of dimensionality, running high dimensional state estimations in real time is also a challenging problem.

Second, regarding task planning, robots need to take several sequential steps to knot the rope gradually. Based on the state estimation result, a task planner needs to be developed to classify at which step the manipulation is, and determine what following actions each robot should take. Meanwhile, failure detection and recovering mechanism should also be included in the task planner in case a failure occurs.

Third, for motion planning, the difficulty lies in that the system is underactuated. Limited numbers of grippers (two in our case) are actuating the rope with high degrees of freedom. It is also observed that the rope always runs to non-repetitive shapes during manipulation, i.e., shape differences always exist between training and test scenarios. Therefore, simply replaying the predefined trajectory for training easily fails in the test stages. An online motion planner should be developed to refine the motion with high efficiency.

In this chapter, a uniform framework for manipulating deformable objects is proposed, which aims at addressing all the challenges discussed above. The core technique we are using is analogy learning proposed in Chapter 4. The structure preserved registration (SPR) algorithm is applied to map one point set to another non-rigidly. For state estimation, the position of each node on the object is acquired by registering the previous estimation results to the new point cloud measurements. The object states can be estimated robustly in real time under noise, outliers and occlusions. For task planning, SPR is introduced to check the similarity between current object states and pre-recorded training states, then the manipulation step can be determined by finding the maximum similarity. Operation failure can also be detected if the similarity value is below some threshold. For motion planning, human operators will first pre-program the manipulation trajectories for the deformable object starting from several specific shapes. During test, a mapping function that registers the training object shape towards the test object shape will be constructed by SPR. The training trajectory is then warped by the mapping function to obtain a new trajectory which is feasible for the test scenario.

The remainder of this chapter is organized as follows: Section 6.2 introduces related works on manipulating deformable objects. Section 6.3 explains the design of the framework in detail, which includes state estimation, task planning and motion planning modules. Section 6.4 tests the performance of the proposed framework by a series of experiments. Supplementary videos can be found in [104]. Section 6.5 concludes the study and proposes future work.

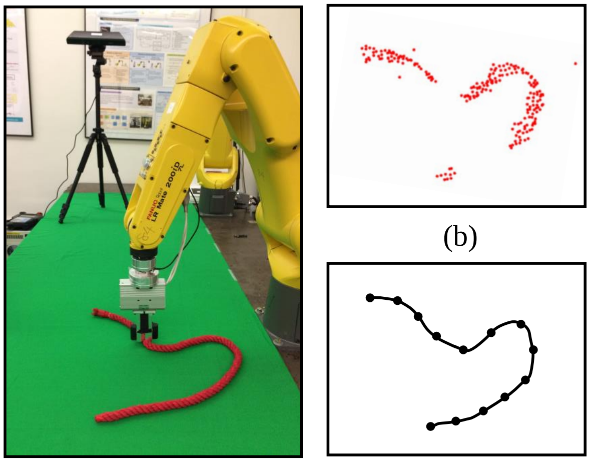
6.2 Related Work

Manipulation of deformable objects is gaining more attention recently because of its broad applications. Several researches have been proposed to tackle this challenging problem. Morita et al. [79] developed a 'knot planning from observation' (KPO) system which estimated the states of ropes, especially the overlap orders by knot theory. Moll et al. [78] constructed a minimal energy model to predict the movement of ropes and plan manipulation trajectories. Kudoh et al. [57] 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. Many of these methods, however, require empirical laws and are developed for a specific task, which is not easy to generalize to other tasks.

To generalize the manipulation skills, Navarro-Alarcon et al. [84] [85] developed a model-free method to automatically servo-control the soft object to a desired shape. A deformation Jacobian matrix which relates the motion of the robot end-effector and the deformation of the object was identified by an online adaptive controller. This matrix was then utilized in generating robot motions given shape errors of the object. Schulman et al. [98] proposed to teach robots to manipulate deformable objects from human demonstrations. They implemented the thin plate spline - robust point matching (TPS-RPM) algorithm [22] to warp the original trajectory taught by human demonstration to get a new trajectory which was suitable for the test scene. Several follow-up works further improved this demonstration-based method. Lee et al. [60] extended the Schulman's approach by jointly optimizing the registration and the trajectory optimization into a single optimization framework, such that the resulting trajectory is smoother. Tang et al. [109] implemented TPS-RPM in the object's tangent space to guarantee no over-stretching nor over-compression of the object during manipulation.

For state estimation, a modified expectation maximization (MEM) algorithm was proposed in [100] to track deformable objects from point clouds. They introduced a probabilistic generative model that incorporated observations of the point cloud and the physical properties of the tracked object and its environment. In [64], a simulation database of common deformable garments was proposed to facilitate recognition and manipulation. Mesh models of common deformable garments are simulated with the garments picked up in multiple different poses under gravity, and stored in a database for fast and efficient retrieval. Some neural network based methods [83] were also developed for planning manipulation trajectory through learning.

Most of the above works, however, are trying to deal with one aspect of the challenges



(b)

(a)

(c)

Figure 6.2: Illustration of state estimation for deformable objects. (a) A robot is manipulating a flexible rope, while a stereo camera is monitoring the object states. (b) The perceived object point cloud is of low quality. Partial points are missing because of the occlusion of robot arms. (c) A robust state estimator is designed to track the key nodes of the deformable object in real time.

(state estimation, task planning and motion planning) for manipulating deformable objects. Integrating all these works together to construct a complete framework is another challenging task. The contribution of this work is that we proposed a uniform framework which addresses all the three major problems, with only using a single technique, analogy learning. The simplicity and consistency of our framework bring great advantages to experimental implementation, parameter tuning, and long-term maintenance.

6.3 Framework for Deformable Object Manipulation

This section introduces the framework for robotic manipulation of deformable objects. Three major modules, state estimation, task planning and motion planning, are introduced in sequence. Each of them uses SPR as a primary tool. For the ease of illustration, an example of rope manipulation will be discussed in the following sections. However, the proposed framework is general for other types of deformable objects, which will be shown in the experimental section.

State Estimation of Deformable Objects

During the process of rope manipulation, since the soft rope easily deforms to unscheduled shapes, it is necessary to close the execution loop by monitoring the rope states in real time.

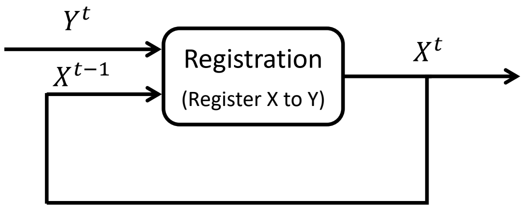


Figure 6.3: Framework of point set registration. Y^t is the perceived point cloud at time step t. X^{t-1} is the state estimation at previous step. A new estimation X^t is achieved by registering X^{t-1} to Y^t.

Tracking in the infinite-dimensional configuration space is impractical. Therefore, we first discretized the rope to a chain of connected nodes with uniform distance (Fig. 6.2c). Our objective is to estimate the position of each node at each time step from the dense, noisy and occluded point clouds (Fig. 6.2b) perceived by stereo cameras.

Suppose at the time step t, the rope state is noted as $X^t = \{x_1^t, x_2^t, \ldots, x_N^t\} \in \mathbb{R}^{N\times 3}$, where $x_n^t \in \mathbb{R}^3$ is the nth node's position in the three dimensional Cartesian space. N is the total number of nodes. At the next time step $t+1$, the rope is changed to a new state, and its point cloud $Y^{t+1} = \{y_1^{t+1}, y_2^{t+1}, \ldots, y_M^{t+1}\} \in \mathbb{R}^{M\times 3}$ is captured by cameras. $y_m^{t+1} \in \mathbb{R}^3$ denotes the position of a single point in the cloud. M is the total number of points and usually $M >> N$.

As shown in Fig. 4.9, considering the topology preserved property of SPR, we can register previous time step node positions X^t towards the next time step point cloud measurement Y^{t+1} by implementing SPR transformation. X^t can be smoothly transformed to new positions $\bar{X}$, which is well aligned with point cloud Y^{t+1}. Because of the well alignment, $\bar{X}$ can be utilized to serve as the state estimation of rope nodes at the time step $t+1$, i.e., $X^{t+1} \triangleq \bar{X}$.

Running the above procedure iteratively, the state estimation at the current time step can always be achieved by registering the previous step estimation towards the current point cloud measurement. Fig. 6.3 shows the closed-loop structure of this state estimator. Note that since tracking is performed in sequences, and the rope shapes between adjacent time steps should not deviate much, only a few iterations of EM updates will register X^{t-1} to Y^t. Therefore, the proposed state estimator can run efficiently in real time. Besides, the estimator is robust to occlusions. During robot manipulation, the view of the stereo cameras might be occluded by the robot arms, which results in missing points in the measured point cloud. However, since the transformation function $\mathcal{T}$ is applied on source points coherently, X^{t-1} can still be registered to the missing point area in Y^t, i.e., the node position in the occluded area is still able to be obtained (Fig. 6.2).

As a physical entity, the object needs to satisfy a series of physical laws, such as kinematics, dynamics, and penetration constraints. For example, the object might be deformable, but not elastic. Therefore, the distance between adjacent nodes should be constant: $\|x_n - x_{n-1}\|^2 \equiv c > 0$. Besides, because of the influence of inertia, the estimated

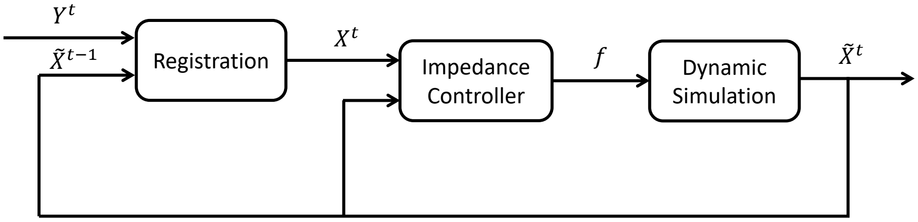


Figure 6.4: Framework of state estimator for tracking deformable objects. Point set registration and dynamic simulation run in a closed-loop to provide a robust estimation of the object states.

nodes should move smoothly even if the measured point cloud changes greatly (e.g., when object segmentation fails). Moreover, a physical body should not penetrate itself or the surrounding objects.

However, the estimator constructed above does not take these physical constraints into consideration. Hence, the estimated states might be only "statistically correct" but "physically wrong". One can try to incorporate these physical constraints by adding more regularizations to the likelihood function of SPR (4.21). However, the coupling of point registration and physical constraints might break the closed-form solution in the SPR's M-step calculation, and make the problem computationally intractable.

Alternatively, we take a decoupled approach. The object states are first estimated by SPR without considering physical constraints. Then the estimates are sent to dynamic simulation for further physical refinement.

As shown in Fig. 6.9 and Fig. 6.10, with the help of physics engine, the object we are tracking can be modelled and simulated in a virtual dynamic environment. The virtual object rendered by the physics engine satisfies physical constraints inherently.

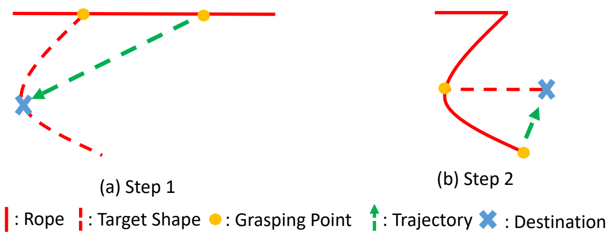
To interact point registration and dynamic simulation, an impedance controller is designed. Denote the states of the virtual object as $\tilde{X}^t = \{\tilde{x}_1^t, \tilde{x}_2^t, \ldots, \tilde{x}_N^t\} \in \mathbb{R}^{N\times 3}$, where $\tilde{x}_n^t \in \mathbb{R}^3$ is the position of the nth virtual node at time step t. If there is a deviation between $\tilde{x}_n^t$ (from simulation) and x_n^t (from registration), a tracking force will be generated and applied on the virtual nodes in the physics engine:

$$f_n^t = K_P(x_n^t - \tilde{x}_n^t) + K_D\frac{\tilde{x}_n^{t-1} - \tilde{x}_n^t}{\Delta t} \tag{6.1}$$

where $K_P \in \mathbb{R}^{3\times 3}$ is the stiffness gain to drive the virtual nodes to dynamically approach to the estimated positions, and $K_D \in \mathbb{R}^{3\times 3}$ is the damping gain to stabilize the dynamic system from excessive oscillations.

The states of virtual nodes in the physics engine, $\tilde{X}$, serve as the final state estimation for the deformable object. The estimation value is then sent back for the initialization of SPR at next time step.

Fig. 6.4 shows the overall closed-loop framework of our proposed state estimator. The combination of probability-based registration and dynamics-based simulation provides the



(a) Step 1

(b) Step 2

Figure 6.5: Two steps to move a straight line to a 'Z' shape.

estimator a strong robustness to sensing noise, outliers, and occlusions as well as satisfying physical constraints.

Task Planning

A complete manipulation task is usually composed of multiple sequential procedures. For example, as shown in Fig. 6.5, two major steps are required to move the rope from a straight line to a 'Z' shape. At each step, a corresponding trajectory can be programmed by human operators to guide the robot to successfully manipulate the rope.

For autonomous manipulation, it is necessary for the robot to recognize at which procedure the current state lies in, so that the most relevant trajectory can be selected for the following manipulation.

The SPR registration is utilized again to design this task planner. Suppose that during training, there are S procedures in total to manipulate the rope, and the initial shape (state) of the rope at each procedure is recorded as $X_s \in \mathbb{R}^{N\times 3}, s = 1, \cdots, S$. During test, the current state of the rope, X^t, is estimated by the proposed observer in the previous section. SPR is then applied to register each recorded state X_s to the current state X^t. The log-likelihood function Q_s^t can be calculated after each registration by (4.21). Note that Q_s^t is negative, and the less negative Q_s^t is, the more similar between X_s and X^t. To normalize the similarity within the 0-100% range, a similarity metric η_s^t is defined as follows:

$$\eta_s^t = \frac{Q_t^t}{Q_s^t} \tag{6.2}$$

where Q_t^t is the log-likelihood calculated by registering X_t towards itself by SPR. η_s^t approaching to 100% indicates stronger similarity between X_s and X^t. The most possible step that the current manipulation lies in can be determined by finding maximum similarity:

$$s^* = \arg\max_s \ \eta_s^t, \quad s = 1, \cdots, S \tag{6.3}$$

Moreover, the task planner can be applied to detect failures during manipulation. If the maximum similarity $\eta_{s^*}^t$ is smaller than a pre-defined threshold, η_{thre}, it indicates that the current rope state differs from all the scheduled steps. Rope manipulation runs into some

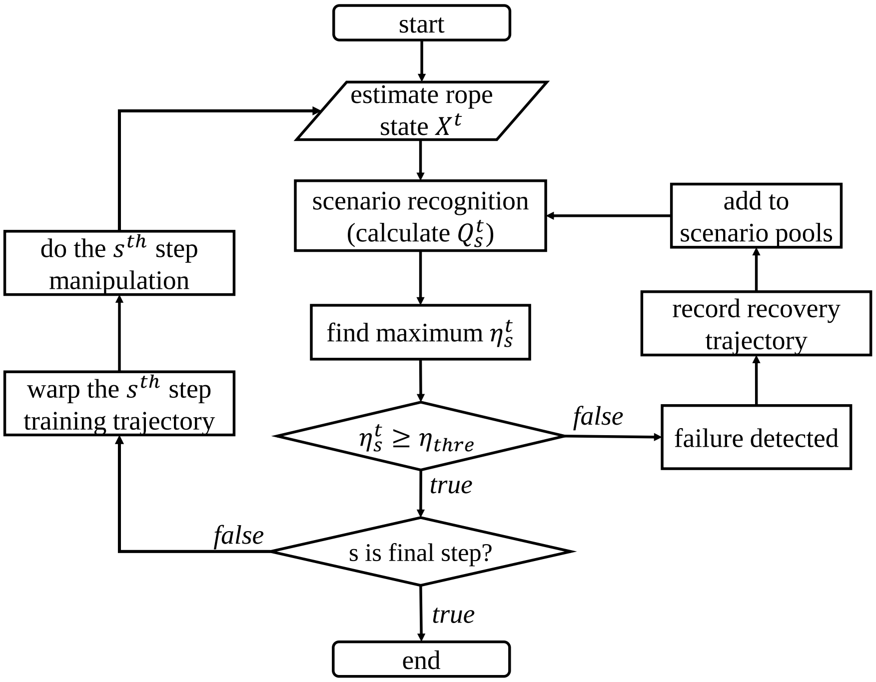


Figure 6.6: The framework of task planning. η_s^t represents the similarity between current state X^t and recorded steps. If the maximum η_s^t is smaller than a lower-bound η_{thre}, an unknown failure occurs. Human is asked to help robots recover the rope. The scenarios pools will be augmented so that when running into the similar scenario next time, this failure will be recognized and the taught recovering trajectory will be traced back.

unknown failures. Human operators need to interfere and teach robots recovering trajectories to move the rope back to one of the recorded states. The failure state will also be augmented to the scenario pools X_s. If this similar failure occurs again in the future, no human interference is required, instead this failure will be recognized, and the planner will use the taught trajectory for recovering as last time. The framework of the proposed task planning module is shown in Fig. 6.6.

Motion Planning

During training, for each of the S manipulation procedures, human operators will program a corresponding trajectory T^s_{train}, $s = 1, \cdots, S$ for the robot end-effector. At test, robots succeed to recognize that the current rope is at sth step by the task planning module in the previous section. However, the sth step's corresponding trajectory T^s_{train} cannot be directly applied to the test scenario, since no matter how similar, there is always some minor shape difference between the rope at training and that at test. This minor difference makes the exact replay of the training trajectory fail at test, such as failing to grasp the rope. Therefore,

T^s_{train} only serves as an approximate reference, while some trajectory refinement based on T^s_{train} is required to achieve a feasible manipulation at test time.

Note that when registering X_s to the current rope shape X^t during task planning, besides the likelihood function Q^t_s, the transformation function $\mathcal{T} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ is also constructed by (4.22). As shown in Fig. 6.14a and Fig. 6.14b, $\mathcal{T}$ transforms X_s to align to X^t by twisting the overall Cartesian space. Similarly, the trajectory T^s_{train} corresponding to X_s can be twisted to get a new trajectory T^s_{test} that is feasible for the test scenario.

The trajectory T of a robot end-effector can be regarded as a sequence of poses $\{p, R\}$, where $p \in \mathbb{R}^3$ is the position vector of the end-effector, and $R \in \text{SO}(3)$ is the orientation matrix. With this observation, the feasible trajectory T^s_{test} can be achieved by applying the following transformation on T^s_{train}:

$$p_{test} = \mathcal{T}(p_{train}) \tag{6.4}$$

$$R_{test} = orth(J_{\mathcal{T}}(p_{train}) \cdot R_{train}) \tag{6.5}$$

$J_{\mathcal{T}}(p)$ is the Jacobian matrix of $\mathcal{T}$ evaluated at position p, and $orth(\cdot)$ is matrix orthogonalization, which can be achieved by singular value decomposition (see (5.4) and (5.5)).

6.4 Experimental Study

A series of experiments were performed to test the proposed state estimation, task planning and motion planning algorithms for manipulating soft ropes. The testbed set-up is first introduced, then followed by experimental results and analysis. Experimental videos can be found at [104].

State Estimation

The Microsoft Kinect was utilized to get the point cloud of a 1-meter-long rope or a middle size T-shirt. 640 × 480 RGB and depth images were captured synchronously at 10Hz. Since the point cloud separation is not our focus in this work, we simply placed the object above a green background, and utilized a color-based filter to segment out the object's point cloud.

For the point set registration, the 1-meter rope was discretized and represented by 50 linked capsules. SPR algorithm implemented with C++ was utilized to register the 50 nodes' positions towards the rope's point cloud. The point sets were first normalized to zero mean and unit variance before registration. The weight μ for uniform distribution was chosen to be 0.1. Smoothness regularization parameter λ and Gaussian kernel's variance β were set as 3.0 and 2.0 respectively. The local neighbour number is selected as $K = 5$ and weight $\tau = 1e-3$. All the data points were denormalized after registration. The registration code was running in C++ on a Ubuntu PC with Intel i7@3.60 GHz and RAM 16GB.

For dynamic simulation, the virtual object was modeled in [17]. The rope was modeled by fifty linked capsules with density $1.5g/cm^3$. Joint stiffness (0.5 Nm/rad) was added to simulate the real rope's bending behavior. For the clothes, it was simulated as mass spring systems on triangular meshes. The stiffness gain K_P and damping gain K_D were set as 10 N/m and 0.5 Ns/m respectively in the impedance controller. [112] was utilized to serve as the interface to communicate between Kinect and Bullet. The code for Bullet graphics rendering was from [100].

Algorithm 1 shows the pseudocode for implementation, which follows the inner and outer loop structure in Fig. 6.4.

```
Output: virtual node position $\tilde{X}$
while point cloud received from Kinect do
    $Y \leftarrow$ segment and downsample object's point cloud
    $\tilde{X} \leftarrow$ get virtual node position from Bullet
    $X \leftarrow$ register $\tilde{X}$ to $Y$ by SPR (4.36)
    while next point cloud not received yet do
        $\tilde{X} \leftarrow$ get virtual node position from Bullet
        $f \leftarrow$ calculate tracking force from $X$ and $\tilde{X}$ (6.1)
        apply force $f$ in Bullet, step simulation
    end
end
```

Algorithm 1: Implementation of State Estimator

During the experiment, we arbitrarily manipulated the object at a moderate speed. The speed of point set registration was tested first. Since the registration time primarily depends on the size of data sets, the point cloud was downsampled to different numbers for testing. As shown in Table 6.1, SPR is able to perform registration efficiently. One reason is that SPR is accelerated by the low rank approximation tricks as stated in Section 4.4. Another reason is that the estimator is running in a closed-loop manner. The estimation result at the last time step serves as a good initialization for the next step. Therefore, only a few

Table 6.1: Registration time of SPR.

Virtual Node #	Point Cloud #	Registration Time
50	100	5ms
	200	11ms
	500	20ms
	1000	35ms

Table 6.2: Execution time of State Estimator

Segmentation	7 ms
Downsampling	2 ms
SPR Registration	11 ms
Total	~20 ms

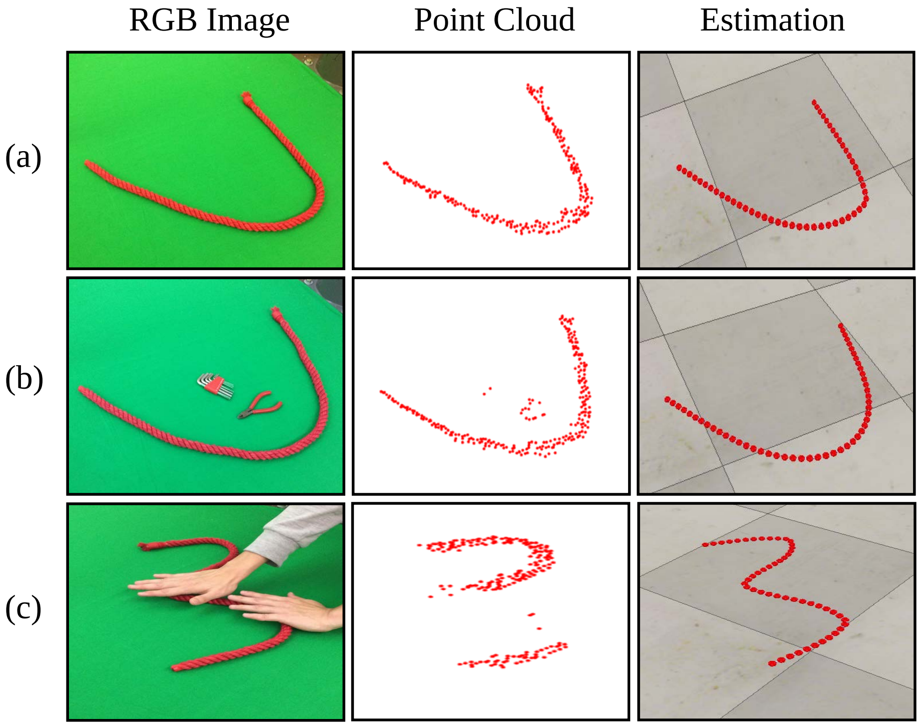


Figure 6.7: Real-time state estimation for a flexible rope. (a) Estimation with sensor noise. (b) Estimation with outliers. (c) Estimation with occlusions.

iterations of EM updates inside SPR will drive the likelihood function to converge. For the following tests, all the point cloud was downsampled to 200 points, which were dense enough to represent the object shape. Table 6.2 lists the execution time of major components in the estimator.

Fig. 6.7 and Fig. 6.8 show the robustness of the proposed estimator towards sensor noise, outliers and occlusions when tracking a flexible rope, where the first column is image or point clouds of the whole environment, the second column is the filtered point cloud for ropes, and the third column is the state estimation result.

A sequence of snapshots during real-time tracking experiments is shown in Fig. 6.9. The rope was manipulated manually from a line to a complicated shape with multiple knots. It is notable that achieving a correct estimation of the knot is a challenging task. At the knot area, point missing is inevitable since the top layer occludes the bottom layer completely. Moreover, the two layers are touching each other closely, i.e., the in-between distance is negligible and beyond the Kinect's resolution. However, our proposed estimator is robust under this extreme condition and can track the knot topology correctly.

In Fig. 6.10, a red T-shirt was folded by the human operator by eight steps. As shown in the second and fourth rows, the tracking algorithm was executable even when the point cloud of the T-shirt was occluded by human arms or itself during the folding process.

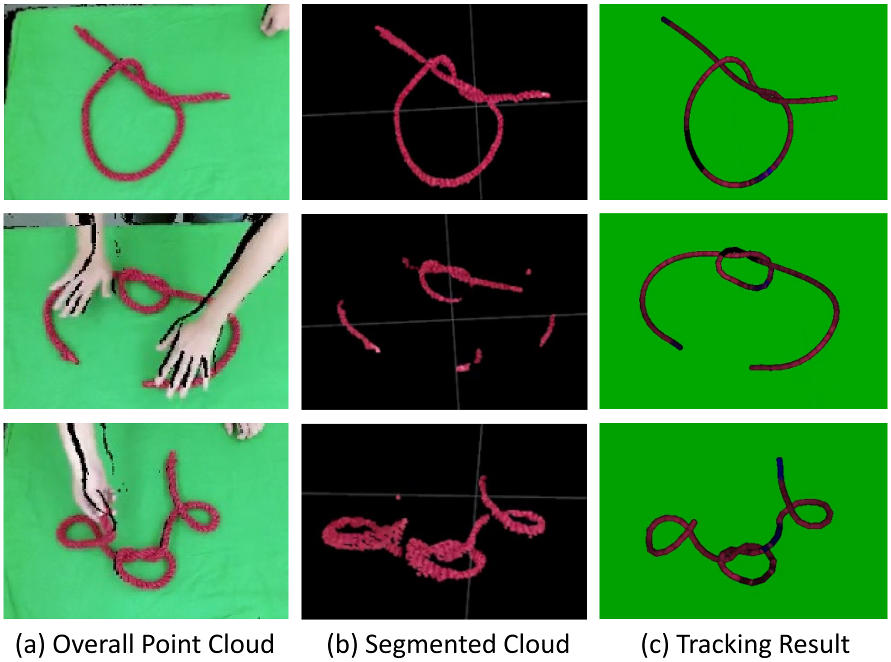


Figure 6.8: Snapshots during the real-time tracking experiments.

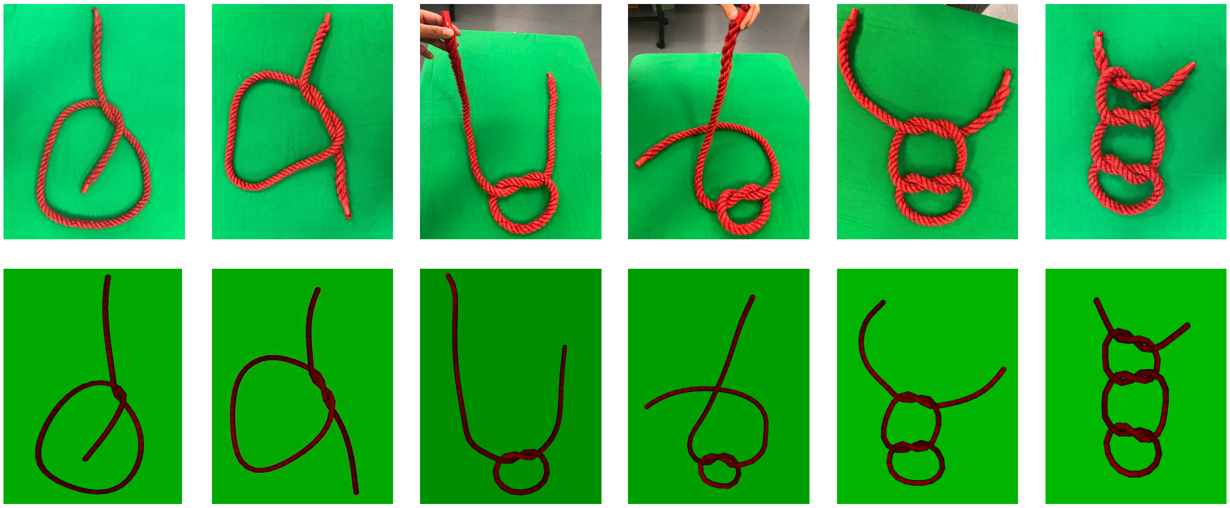


Figure 6.9: Track the deformation of a flexible rope in sequence. The rope was manipulated manually to deform to a complicated shape with multiple knots. The first row shows the RGB image. The second row shows the state estimation result.

As shown in Fig. 6.11, to analyze the tracking accuracy quantitatively, 11 markers with distinct colors were attached on the rope with 10cm interval. These markers were distinguished by a color-based filter and their ground-truth positions were measured directly from

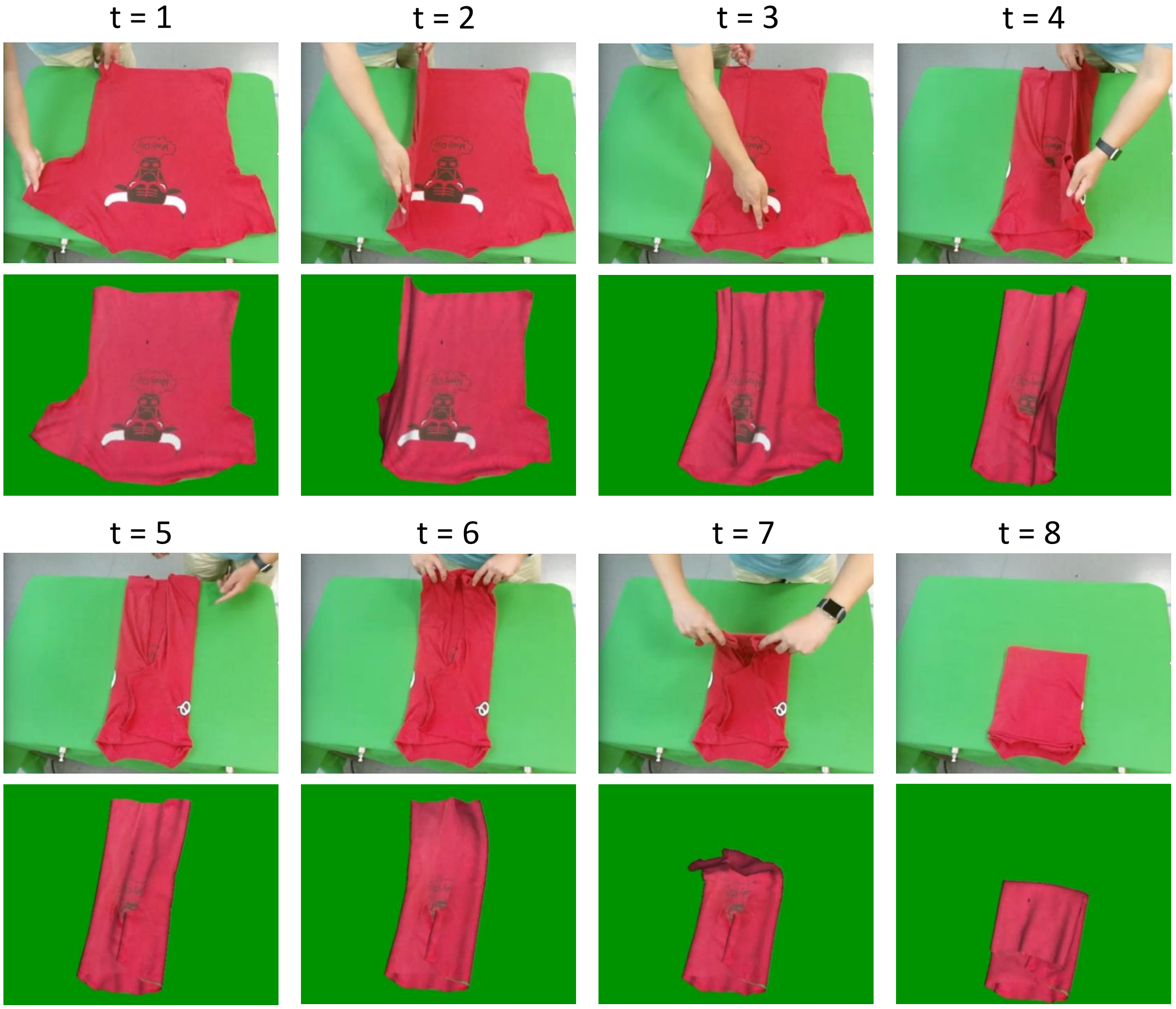


Figure 6.10: Track the deformation of a flexible T-shirt in sequence.

Kinect. Note that these markers were only used for the purpose of ground truth. They were not utilized in our state estimation algorithm. The estimation results from our proposed method were compared with these ground-truth values. Fig. 6.12 shows the average tracking error and standard deviation at six different rope configurations. In general, the tracking error is less than 2.2cm, smaller than the jaw width (6cm) of the robot gripper. Therefore, even if the gripper went to an inaccurate grasp pose because of the tracking error, the rope still located between the gripper's fingers and could be successfully grasped. We also compared our tracking algorithm with the MEM method proposed in [100]. Fig. 6.12 shows that their tracking performance is similar. However, our proposed framework advances on the extendibility since its application is not limited in state estimation, but can also be applied on task planning and trajectory planning for deformable object manipulation.

(a) V shape (b) N shape (c) M shape

(d) Circle (e) Half knot (f) Knot

Figure 6.11: Six configuration shapes with markers attached.

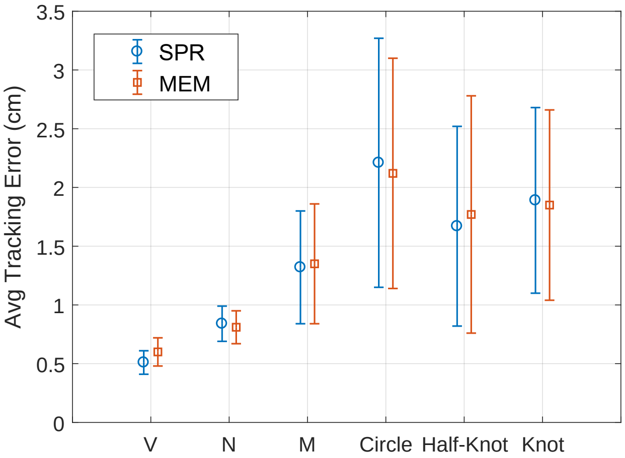


Figure 6.12: Average tracking errors and standard deviation at the marker positions.

Task Planning

Following the pipeline in Fig. 6.13, the state estimation result was then sent to a Windows 10 desktop (Intel i7@3.60 GHz + RAM 8GB) which ran the task planning and trajectory planning algorithms. ROS [112] served as the interface to communicate between the Kinect, the Ubuntu PC and the Windows PC.

As shown in Fig. 6.14(a), four major steps were predefined by human operators for the task of rope knotting. At each step, the initial shape of the rope was recorded by the state estimator. The corresponding manipulation trajectory was then demonstrated by human by lead through teaching. To be specific, operators guide the two robots' end-effectors to go through some key poses, then the training trajectory was obtained by linear interpolation

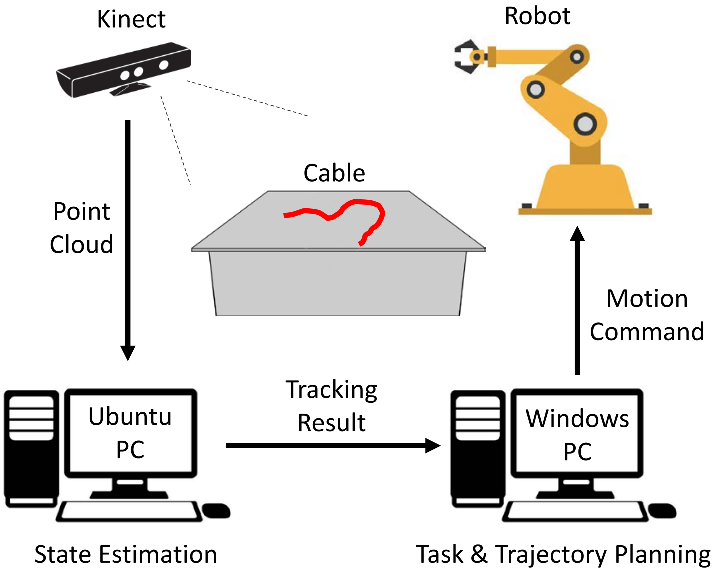
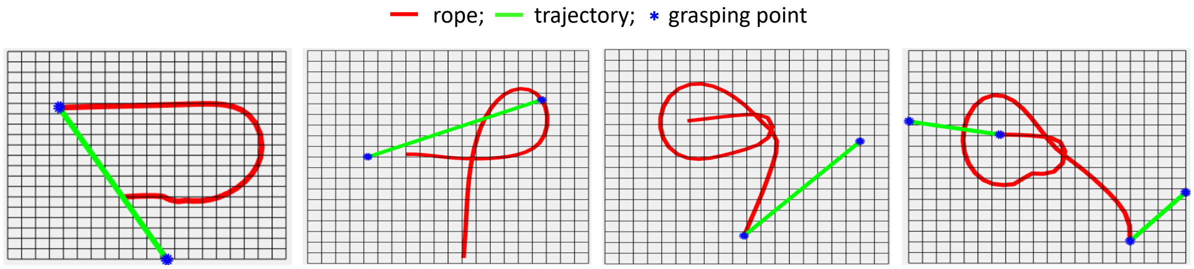
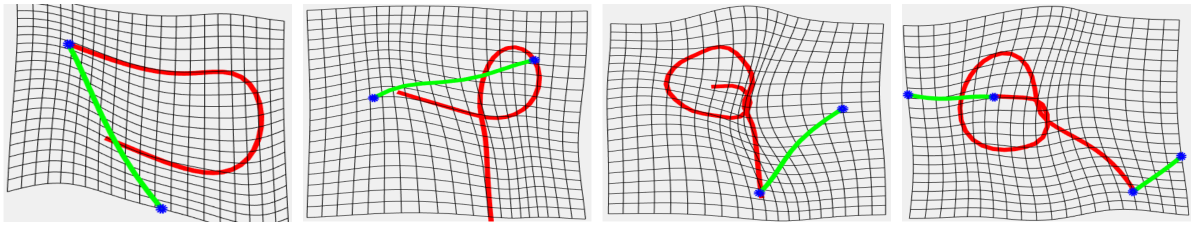


Figure 6.13: The testbed setup.



(a) Training Scenarios

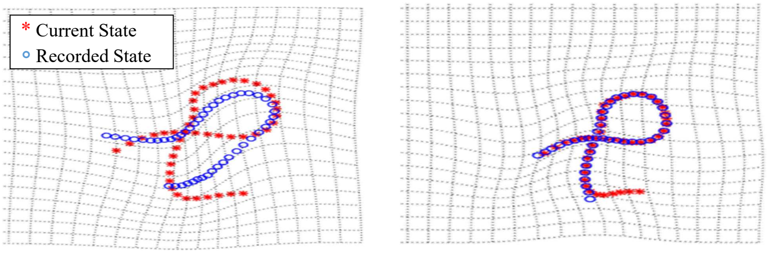


(b) Test Scenarios

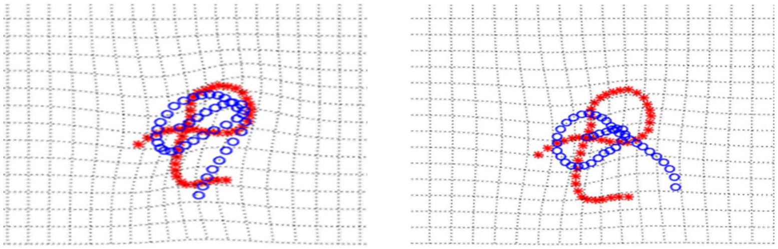
Figure 6.14: Four major steps for rope knotting manipulation. The red line is the rope state, and the green line is the trajectory of robot end-effector. Blue dots are the grasping/releasing positions. Black grids show that the original Cartesian space is twisted so as to map the training scenarios towards test scenarios.

between neighbour poses.

At test, the current rope states were estimated and compared by SPR to each of the four recorded templates. The similarity level was calculated by (6.2). As shown in Fig. 6.15,



(a) Comparison with Step 1，$\eta_1 = 23\%$ (b) Comparison with Step 2，$\eta_2 = 90\%$



(c) Comparison with Step 3，$\eta_3 = 37\%$ (d) Comparison with Step 4，$\eta_4 = 13\%$

Figure 6.15: Similarity check between the current rope state (red dots) and the four recorded training states (blue dots). The second scenario is most similar to the current state with a 90% similarity.

the red point set (current state) and the blue point set (recorded state) in the second image had the largest similarity (90%), which indicated that the manipulation process was at the second step at that moment. The similarity lower-bond η_{thre} was set as 80%. If all the similarity check is below 80%, a warning message will be shown to ask the human operator to demonstrate a recovering trajectory. The failure states and recovering trajectories were then augmented into the task planning sample pools. The robot's ability of detecting and recovering from failures is shown in the attached videos.

Trajectory Planning

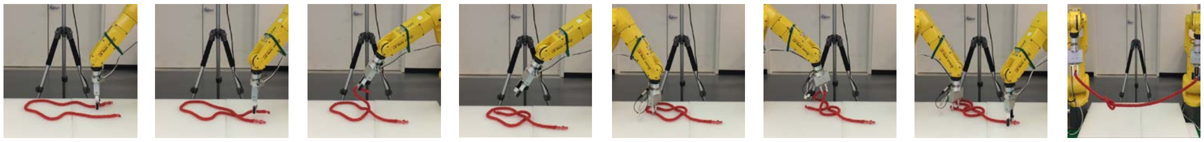
After identifying the task step, the manipulation trajectory was generated by the proposed trajectory planning algorithm.

To represent the rope positions and the manipulation trajectory under the same coordinate system, the relative translation (extrinsic parameter) between the Kinect and the robot world frame is calibrated first. The rope states were all translated to the robot world frame afterwards.

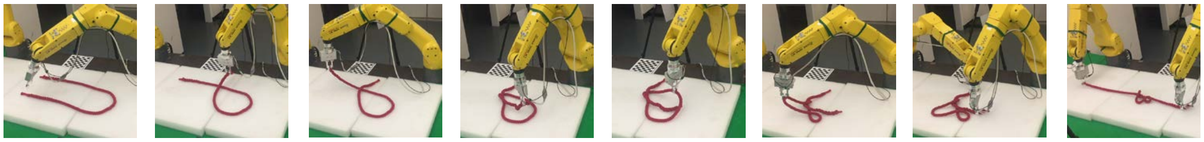
As shown in Fig. 6.14, the rope state during test was different from that in training. With SPR registration, the training trajectory was transformed by (6.4) and (6.5) at each step to achieve the test trajectory. The end-effector poses were then transformed to robot joint command by robotic inverse kinematics. The joint command was finally sent to the robot controller for execution.

Fig. 6.16 shows the snapshots of autonomous rope knotting by two robot arms. Three types of knotting were designed, with each type tested 15 times. The overall success rate was 40/45. Most of the failure was miss-grasping, which might result from the relatively low accuracy of Kinect and calibration error between the camera frame and robot base frame.

(a) Knot Type 1



(b) Knot Type 2



(c) Knot Type 3

Figure 6.16: Snapshots of the rope knotting experiments. Two robot arms were collaborating to knot the rope based on the transformed trajectory. (a)(b)(c) show three different types for knotting.

More Applications on Manipulating Deformable Objects

More manipulation scenarios are designed to show the application potentials of our framework. One is rope manipulation to any sketched shapes. Another is failure detection during wire harness assembly.

For rope manipulation, the human operator first drew the desired shape manually through MATLAB GUI. As shown in Fig. 6.17, the blue curve is the sketched shape. The objective of robots is to manipulate a rope from a random initial shape to this desired shape. By calculating curvatures, the inflection points (red dots) on the drawing can be detected. With the feedback of the proposed state estimator, robots could track the states of the inflection points and manipulate them to the desired positions. As shown in Fig. 6.18, the rope was gradually manipulated to the desired shape by the dual-arm robot. Note that during manipulation, though the robot arms occluded the rope, the proposed estimator could track it robustly.

Fig. 6.19 shows the process of robotic wire harness assembly and the real-time tracking results. The dual-arm robot was expected to assemble the soft blue wire into four specified fixtures. With the proposed estimator, the shape of the wire was tracked and the assembly

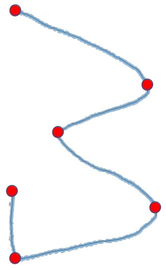


Figure 6.17: Sketch the desired shape manually.

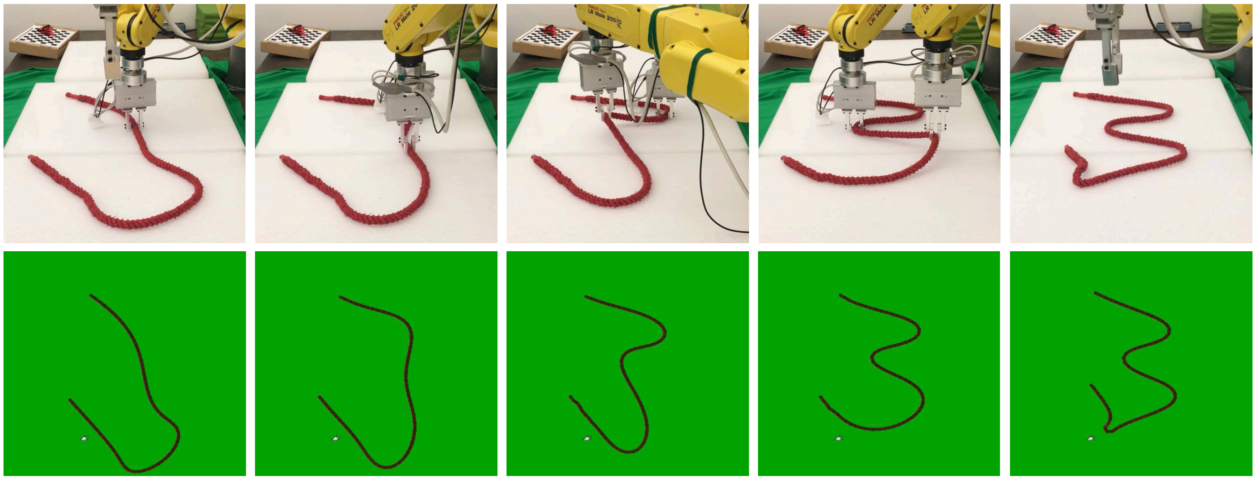


Figure 6.18: Rope manipulation with real-time visual feedback.

quality could be evaluated online. As the picture shows (second and fourth row), if the wire was successfully installed, a green circle would be marked above the corresponding fixture. Otherwise, a red cross showed up and the robot would try to redo the assembly. The assembly success was determined by checking the distance between the fixture and the corresponding node on the wire. The length of the wire between neighbor fixtures could also be calculated accumulatively. Therefore, if the wire was detected to be too loose, the robot could also straighten the wire with a suitable displacement.

6.5 Chapter Summary

Based on the concept of analogy learning and structure preserver registration (SPR), a skill for tracking and manipulating deformable objects is developed for industrial robots. The proposed framework includes functions such as state estimation, task planning and motion planning. A real-time observer is developed to estimate the node position of the object by registering the last step estimation towards the current point cloud measurement. A task planner is then developed to let robots recognize at which procedure the current ma-

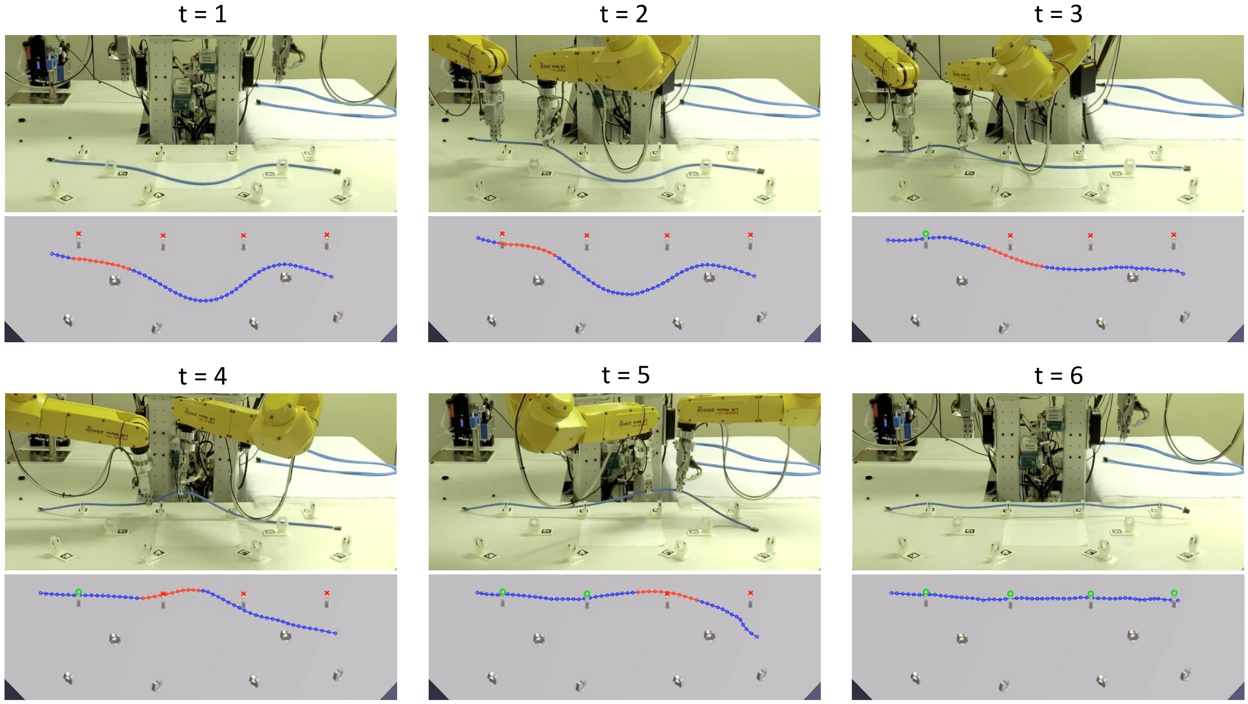


Figure 6.19: Failure detection of robotic wire harness assembly.

nipulation is by registering training scenarios towards the test scenario. Finally, utilizing the transformation function constructed during scenario registration, the training trajectory can be warped to achieve a new trajectory which is feasible for the test. A series of experiments are implemented, including rope manipulation, clothes manipulation and wire harness assembly, which indicate the effectiveness of the proposed framework.

Chapter 7

Robotic Motion Re-planning by Analogy Learning

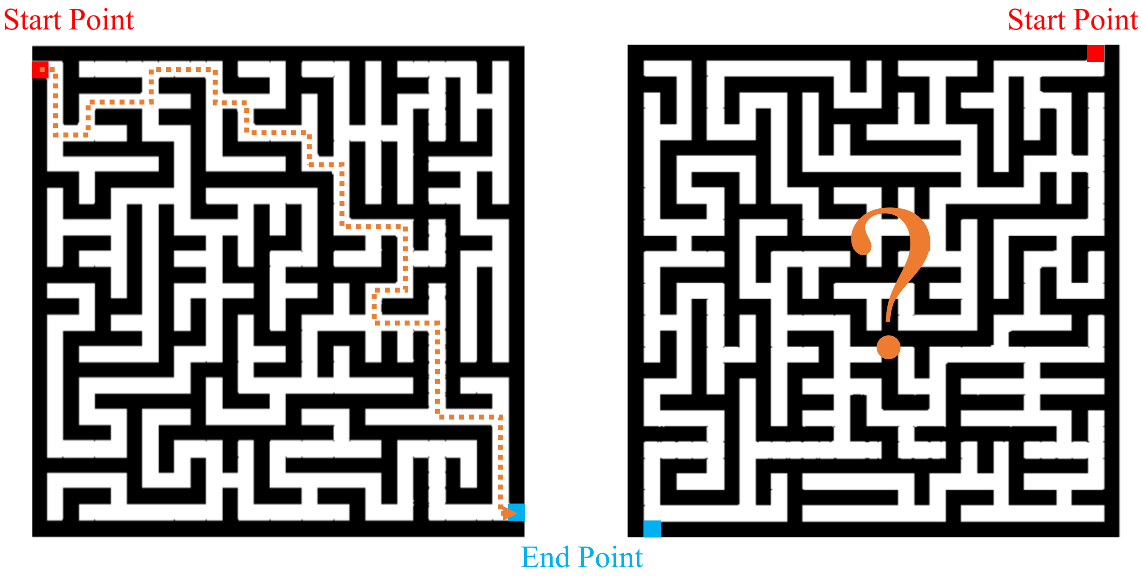
7.1 Introduction

Motion planning is one of the essential primitives for robotics. It is the foundation for manipulation, navigation and many other applications. A basic motion planning problem is to schedule a continuous motion that connects the start configuration x_S and the goal configuration x_G, while avoiding collision with obstacles, satisfying constraints (e.g., max speed, max acceleration) as well as maximizing values of rewards (e.g., smoothness, distance, control cost).

As a well-explored field in the past decades, there are already plenty of approaches developed for robotic motion planning, including grid-based [102, 39, 33, 26], sampling-based [58, 49, 48, 35] and optimization-based methods [69, 70, 95, 99].

Grid-based approaches, such as Dijkstra's algorithm [102] and A* [39], overlay a grid on the robot's configuration space, and assume each configuration is identified with a grid point. At each grid point, the robot is allowed to move to adjacent grid points as long as it is collision-free. Heuristic functions can also be designed to accelerate the searching speed by guiding the exploration direction instead of exploring in all directions. These methods are usually complete algorithms for the path planning problem, i.e. they will find a solution eventually if it exists. However, the number of points on the grid grows exponentially in the configuration space dimension, which makes them inappropriate for high-dimensional problems, such as industrial manipulators with multiple degrees of freedom.

Instead of evaluating all possible solutions on grids, sampling-based approaches, such as probabilistic roadmap (PRM [49]) and rapidly-exploring random tree (RRT [58]), represent the configuration space with a roadmap of sampled configurations. For PRM, random samples are taken from the configuration space and connected to construct a sparse graph. The start and goal configurations are added into the graph, and a graph search algorithm is applied to determine the shortest path between the start and goal configurations. RRT creates



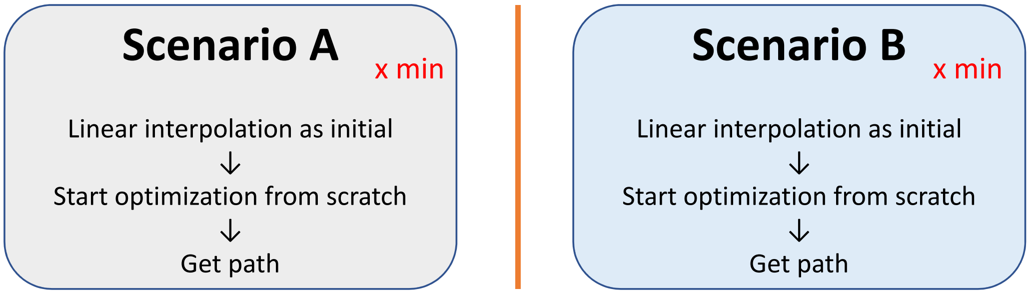
(a) Scenario A (b) Scenario B

Figure 7.1: Find collision-free paths for robots in two mazes. Many motion planners treat the two scenarios independently and solve separately. However, scenario B is nothing but rotating scenario A with 90 degrees clockwise. Utilizing the similarity between scenarios can achieve a feasible solution for scenario B immediately without heavy computation.

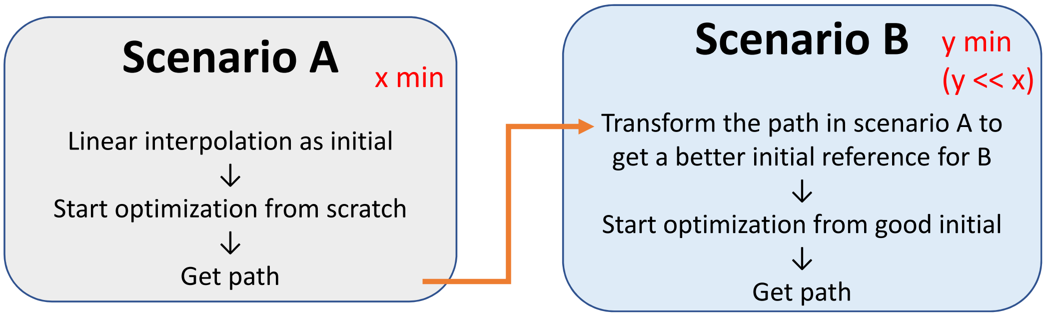
possible paths by randomly adding points to a tree until some solution is found or time expires. These algorithms work well for high-dimensional configuration space, because their running time is not exponentially dependent on the dimension. They are probabilistically complete, meaning that the probability that they will produce a solution approaches 1 as more time is spent.

However, the paths found by grid-based or sampling-based approaches are usually zigzag and non-smooth. The dynamic constraints of robots are also not considered during the path search. To address these challenges, optimization-based approaches [95, 99] are proposed to formulate the motion planning task as a non-convex optimization problem, where the objective is to maximize a constructed reward function, and the constraints are initial/end conditions, system dynamics as well as configuration boundaries. The bottleneck of optimization-based planning is feasibility. Considering the non-linear robot dynamics and non-convex obstacle shapes, usually the formulated optimization for motion planning is highly non-convex. The optimization procedure might get stuck in local optimum, and sometimes even no feasible solution can be found. The selection of good initial reference becomes critical to alleviate these problems.

The aforementioned approaches achieve great success in robot motion planning. However, we observe that these approaches usually regard each motion planning scenario as an independent problem, while seldom search for the correlation between scenarios so as to



(a) Traditional planner: process each scenario independently



(b) Analogy Learning: find correlation between scenarios

Figure 7.2: Comparison between analogy learning and other approaches for path planning.

shorten the re-planning time for similar scenes.

Take the 2D motion planning problem as an example. Figure 7.1 shows two mazes with different configurations. An autonomous robot needs to plan collision-free paths under this two scenarios from the start point to the end point. Most motion planners will solve these two problems separately, i.e., planning the motion from scratch under each scenario. However, scenario B is actually a rotation of scenario A. If we simply rotate the path planned in scenario A with the same way, then we can get a feasible path immediately for B without the need of heavy computation.

With this observation, this chapter will discuss how to apply the concept of analogy learning to accelerate robot re-planning under similar scenarios. The remainder is organized as follows: Section 7.2 introduce the concept and benefits of path replanning, with a case study on a 6-DOF industrial manipulator. Section 7.3 augments the configuration space to configuration-time space, and apply analogy learning for trajectory replanning. To manipulate objects without violating their physical constraints, a tangent space path replanning algorithm is proposed in Section 7.4. Section 7.5 concludes the work and discusses remaining problems. Simulation videos can be found at [104].

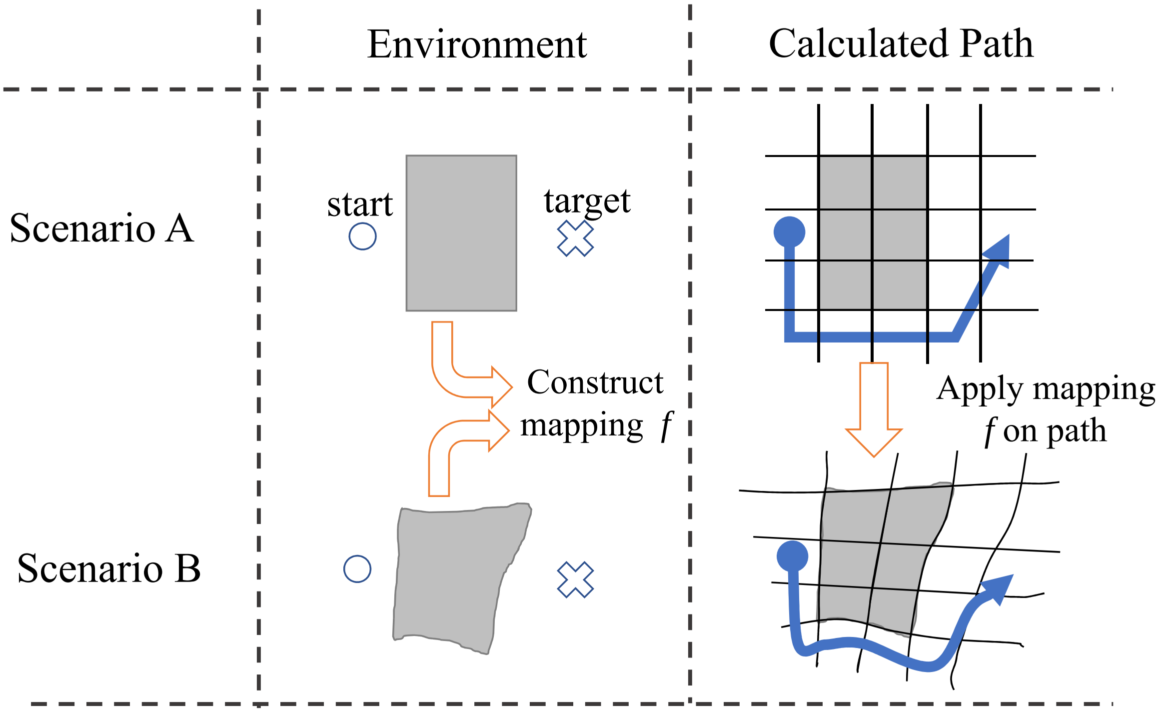


Figure 7.3: Comparison between analogy learning and other approaches for path planning.

7.2 Path Re-planning by Analogy Learning

In some robotic applications, the surrounding environment might be highly dynamic, unstructured and stochastic. It is necessary to deploy large computation power for a safe motion planning, and regard each scenario as a new one because of the low configuration repeatability. However, in many other applications, especially in industrial automation, the surrounding environment usually can be structured in advance. The configuration of the assembly line might be adjusted from time to time, so there is still need for path re-planning. However, since the configuration modification is minor, it is unnecessary to regard these configurations independently and solve separately. Instead, considering the high similarities between scenarios, we can exploit this affinity to accelerate the path planning for future similar scenarios.

Take the optimization-based path planning as an example. As shown in Fig. 7.2, scenario A, we need to plan a 2D path for the mobile car to avoid collision with the grey obstacle. A linear interpolation between the start and end point is utilized to initialize the optimization. A collision-free path (blue line) is finally obtained by gradient descent. However, not every time a feasible solution can be achieved from the optimization solver. As discussed in Section 7.1, because of the high non-convexity, the optimization process might get stuck in bad local optimum, which results in long computation time, sub-optimum result or even infeasible

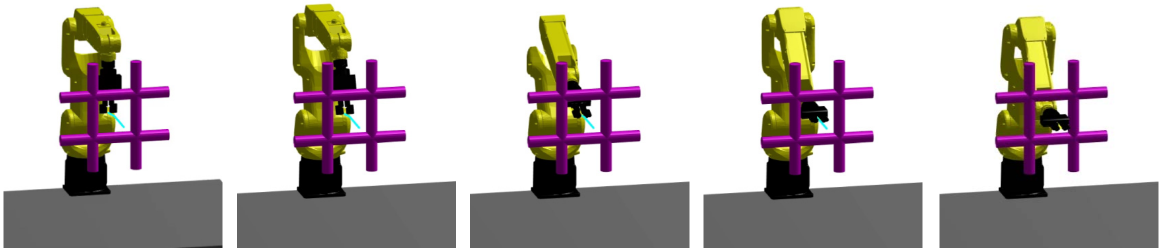
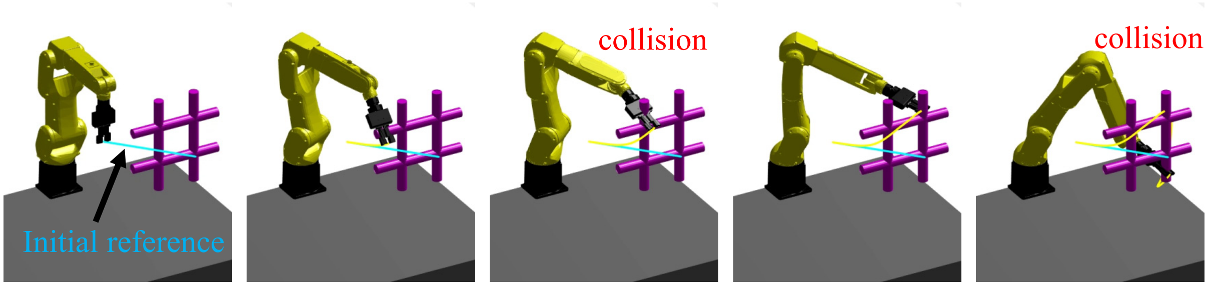
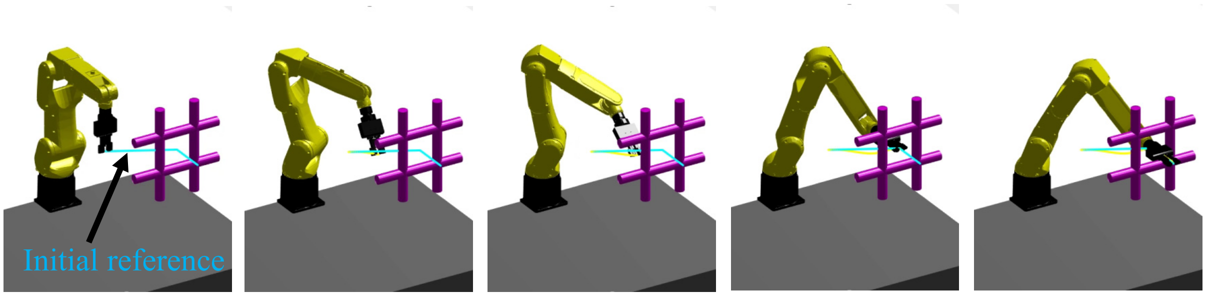


Figure 7.4: Path planning of Scenario A.



(a) Scenario B, with linear interpolation as initial reference

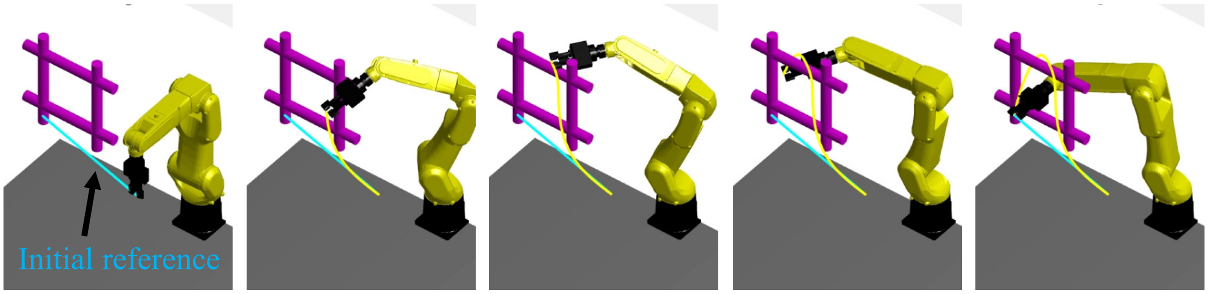


(b) Scenario B, with transformed path from Scenario A as initial reference

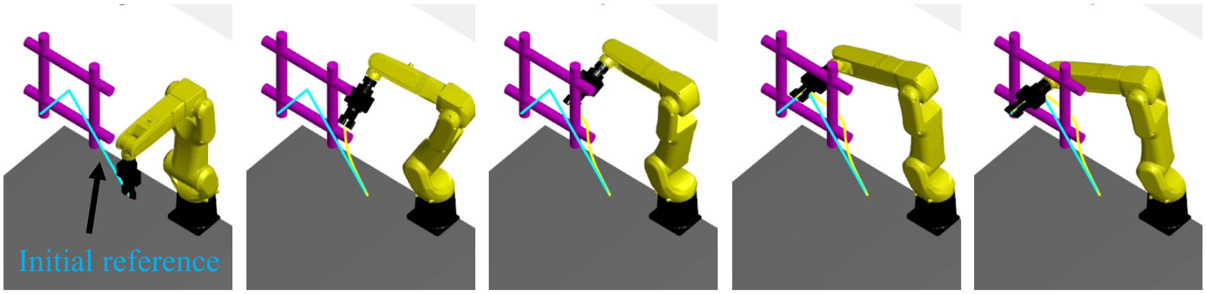
Figure 7.5: Path planning of Scenario B.

result. One promising method to solve this dilemma is to provide the optimization solver a ‘good’ initial reference, i.e., a reference that is as close to the global optimum as possible. Therefore, when it comes to scenario B in Fig. 7.2, a rational solver should not initialize with linear interpolation again. Instead, a spacial mapping between the two scenarios can be found, and utilized to transform the optimized path for A to get a new path for B. The transformed path then serves as the initial reference for the solver so as to boost optimization.

The detailed procedure of path transformation has been presented in Section 6.3. Assume the environment in scenario A and B can be formulated as two point sets X and Y in the Cartesian space. The structure preserved registration (SPR, Section 4.4) algorithm can be utilized to construct a non-rigid transformation $\mathcal{T} : \mathbb{R}^D \to \mathbb{R}^D$ to map X towards Y. As shown in Fig. 7.3, $\mathcal{T}$ transforms X to align to Y by warping the overall configuration space.



(a) Scenario C, with linear interpolation as initial reference



(b) Scenario C, with transformed path from Scenario A as initial reference

Figure 7.6: Path planning of Scenario C.

Similarly, the path P_A corresponding to scenario A can be twisted to get a new path P_B that is suitable for scenario B.

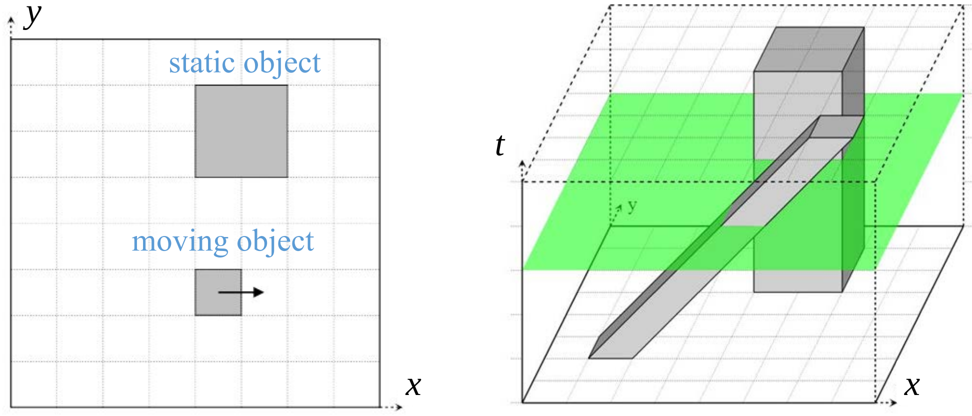
The path P of a robot can be regarded as a sequence of poses $\{p, R\}$, where p is the position, and R is the orientation. The path P_B can be achieved by applying the following transformation on P_A:

$$p_B = \mathcal{T}(p_A) \tag{7.1}$$

$$R_B = orth(J_{\mathcal{T}}(p_A) \cdot R_A) \tag{7.2}$$

$J_{\mathcal{T}}(p)$ is the Jacobian matrix of $\mathcal{T}$ evaluated at position p, and $orth(\cdot)$ is a function that orthogonalizes matrices.

A case study of path re-planning on a 6-DOF manipulator is shown in Fig. 7.4. The four purple bars are obstacles and the robot's objective is to plan a collision-free path such that its end-effector passes through the hole to reach a target point. In scenario A (Fig. 7.4), a collision-free path is planned by an optimization solver, convex feasible set (CFS, [68]). In Scenario B (Fig. 7.4a), the obstacles together with the target point shift to the left side of the robot. We still use linear interpolation between the start point and end point as the initial reference. However, CFS fails to find a collision-free path because of getting stuck into bad local optimum. In contrast, the analogy learning concept is applied to register the two obstacles and transform the path found in scenario A to get a new path for scenario B.



(a) Configuration space (b) Configuration-time space

Figure 7.7: Illustration of configuration-time space.

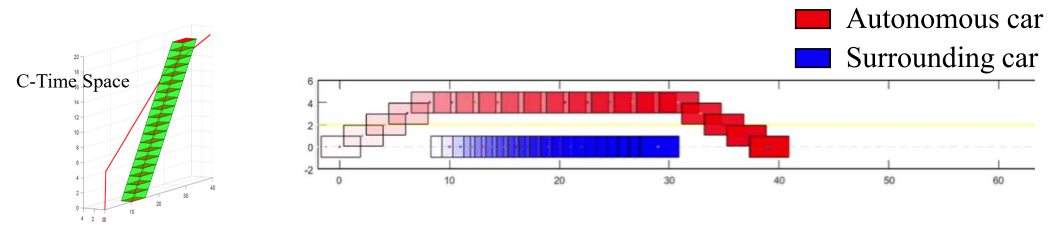
This new path is utilized as the initial reference for CFS optimization (Fig. 7.4b). With this better initialization, an optimal and collision-free path for scenario B is found finally.

Moreover, the obstacles and target point are moved to the right side of the robot to form scenario C (Fig. 7.6a). With linear interpolation as an initial reference, CFS does succeed in finding a collision-free path for this scenario. However, the total calculation time is 3.2 sec. As a comparison, with the transformed path as initialization for optimization, CFS spends only 0.7 sec to find a collision-free solution. Note that the procedure of path transformation is also efficient, which takes 0.12 sec. The total computation speed increases 3.9 times by the proposed approach.

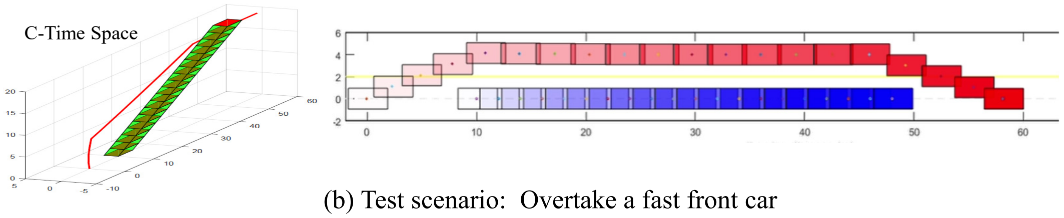
To conclude, path re-planning with analogy learning utilizes previous experience in past scenarios and transforms the previously solved paths to get a new one as the initial for optimization. Since this initial reference should be closer to global optimum compared to naive initialization, the optimization can benefit by avoiding bad local optimum and converging at a faster speed.

7.3 Trajectory Re-planning by Analogy Learning

The idea of applying analogy learning for path planning can also be extended for trajectory planning. The only difference between the path and the trajectory is that the trajectory has an additional time dimension, i.e., each waypoint in the path has a corresponding requirement for pass-through time. To incorporate the dimension of time for analogy learning, the original configuration space can be augmented and the configuration-time space is constructed as shown in Fig. 7.7. Non-rigid registration is applicable in this augmented space, and the trajectory can be transformed as before.



(a) Training scenario: Overtake a slow front car



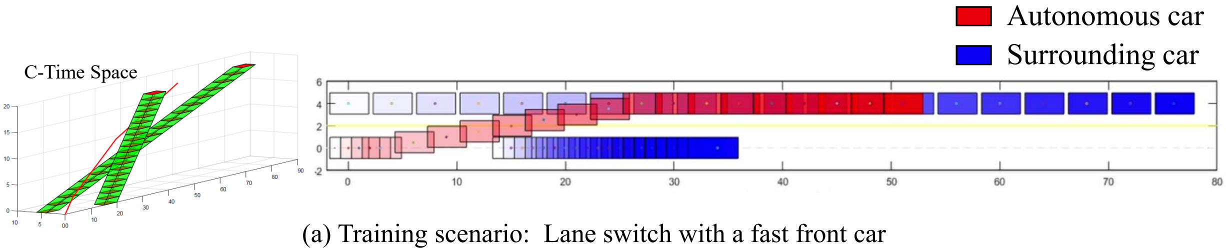
(b) Test scenario: Overtake a fast front car

Figure 7.8: Trajectory replanning for test scenarios: overtaking

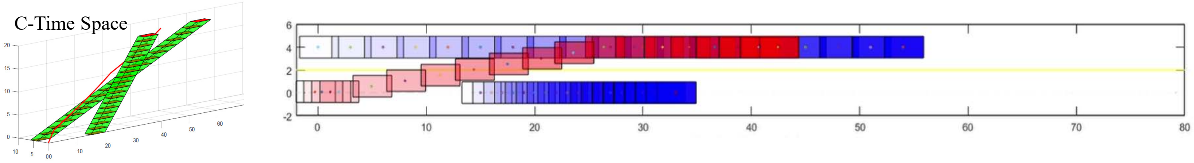
A case study of trajectory re-planning for highway autonomous driving is shown in Fig. 7.8, Fig. 7.9 and Fig. 7.10. Three types of scenes are designed: (1) overtaking, (2)lane switching, (3) lane switch and returning back. In each case, there is a training scenario and a test scenario. The trajectory path for training scenario is given from human demonstration. In the test scenario, since the surrounding car's speed is different, a new collision-free trajectory needs to be planned. As illustrated in the left side of Fig. 7.8, the configuration-time space is constructed for both training and test scenarios. A transformation function is found by SPR to register the two obstacles in the augmented space. The function is then utilized to transform the demonstrated trajectory (red line) to achieve a new trajectory for the test scenario. Simulation results show that the proposed trajectory replanning approach is robust to the speed change of surrounding cars and all the transformed trajectories all collision-free and suitable for the test scenarios.

7.4 Tangent Space Path Re-planning by Analogy Learning

In Chapter 6, we showed that the concept of analogy learning could be utilized for teaching robots to manipulate deformable objects, such as cables. However, this current approach regards the physical object as a bunch of discrete and independent points, while dismissing the object's local physical properties, such as curvatures and distances between points. As a result, the transformed path might not manipulate the object into a similar shape as shown at training. Overstretch and over-compression might also occur because the path is generated without considering the object's physical limitations. In practice, these accidents

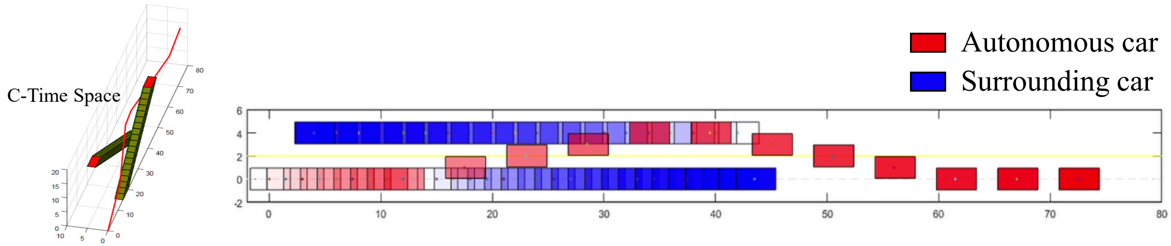


(a) Training scenario: Lane switch with a fast front car

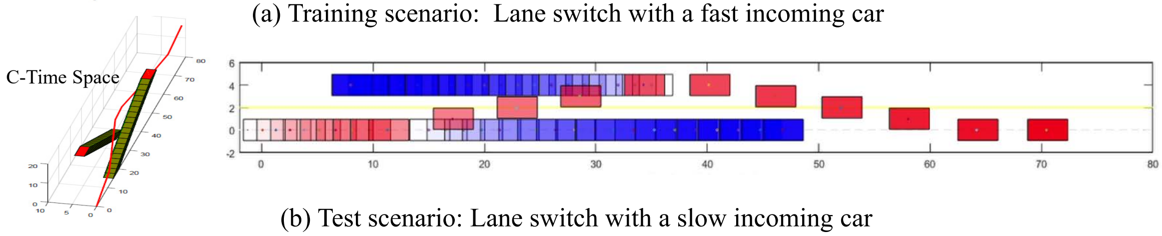


(b) Test scenario: Lane switch with a slow front car

Figure 7.9: Trajectory replanning for test scenarios: lane switch



(a) Training scenario: Lane switch with a fast incoming car



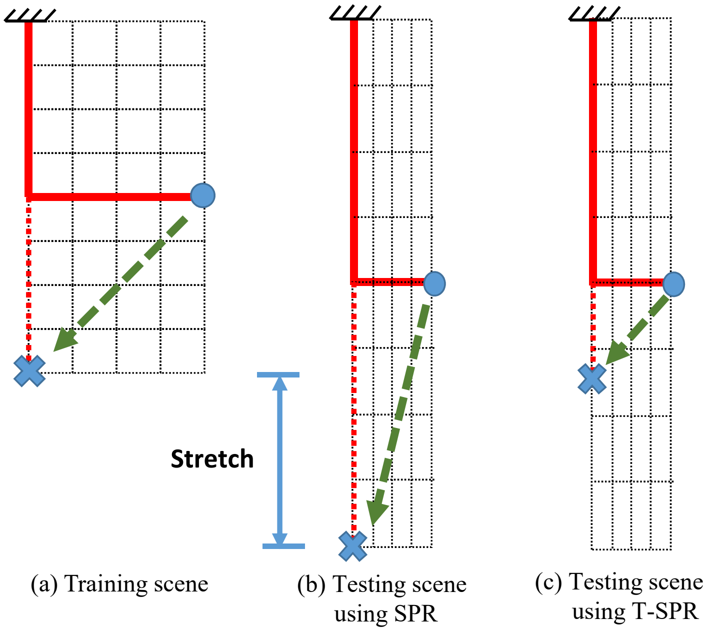
(b) Test scenario: Lane switch with a slow incoming car

Figure 7.10: Trajectory replanning for test scenarios: lane switch and return back

could cause damage to the object or to the robot.

Take rope manipulation (Fig. 7.11) as an example. The initial shape of the rope is slightly changed from training scene to test scene. The SPR algorithm tries to register the two scenes by shrinking the Cartesian space in the horizontal directional while expanding in the vertical direction. As a result, it shrinks/expands the training path in the same way to get the test path (green line in Fig. 7.11b). This warped path, however, violates the rope's length constraint and overstretches the rope during manipulation.

To deal with this problem, this section proposes a tangent space path transformation method, named tangent space structure preserved registration (T-SPR). Instead of mapping



(a) Training scene

(b) Testing scene using SPR

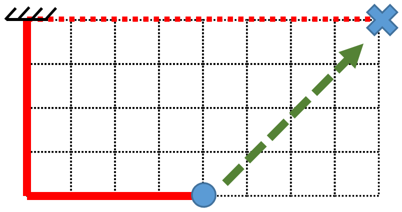
(c) Testing scene using T-SPR

Figure 7.11: Rope manipulation by SPR and T-SPR. The red solid line shows the deformable rope, and the green line is the manipulation path. One end of the rope is fixed at the origin. (a) At training scene, the robot is taught to manipulate the L-shaped rope into a vertical straight line. (b) At test scene, SPR shrinks/extends the space in the horizontal/vertical direction, so as to register the training scene and the test scene. The training path is warped in the same way to get the test path, which overstretches the rope during manipulation. (c) The proposed T-SPR algorithm maps the scenes in the tangent space. The Cartesian space length is maintained during transformation. It manipulates the rope into a straight line without overstretching.

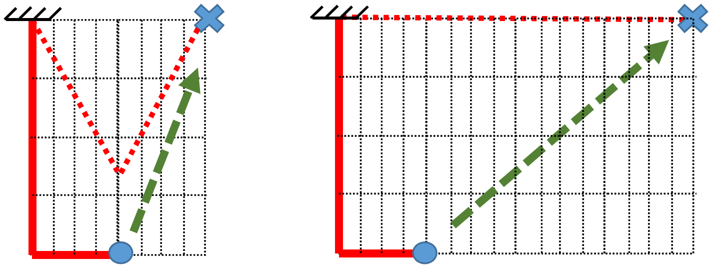
the training scene and the test scene in the Cartesian space, the registration is performed in the tangent space to maintain the structural information of the object. It is able to manipulate the object into a similar shape as training (Fig. 7.12c), and is guaranteed not to overstretch the object during manipulation (Fig. 7.11c).

Tangent Space Mapping

For the ease of explanation, here we take rope manipulation on a plane as an example. As shown in Fig. 7.13, the rope can be equivalently represented in the Cartesian space and in the tangent space. Fig. 7.13a and Fig. 7.13c show the rope at training scene and at test scene, both in the Cartesian space. Fig. 7.13b and Fig. 7.13d present the corresponding tangent graphs of the rope, where the horizontal axis is the arc length along the rope, and the vertical axis is the direction of the unit tangent vector. In this case, the rope's unit tangent vector is one dimensional.



(a) Training scene



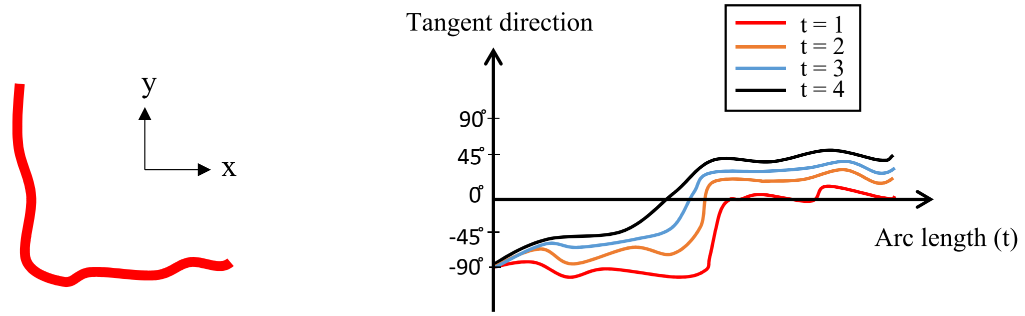
(b) Testing scene
using SPR

(c) Testing scene
using T-SPR

Figure 7.12: Example of Rope manipulation by SPR and T-SPR. The red solid line shows the deformable rope, and the green line is the manipulation path. One end of the rope is fixed at the origin. (a) At training scene, the robot is taught to manipulate the L-shaped rope into a horizontal straight line. (b) At test scene, SPR shrinks/extends the space in the horizontal/vertical direction, so as to register the training scene and the test scene. The training path is warped in the same way to get the test path, which manipulates the rope into V-shape, not similar as training. (c) The proposed T-SPR algorithm maps the scenes in the tangent space. The curvature information is maintained during transformation. It manipulates the rope into a straight line similar as training.

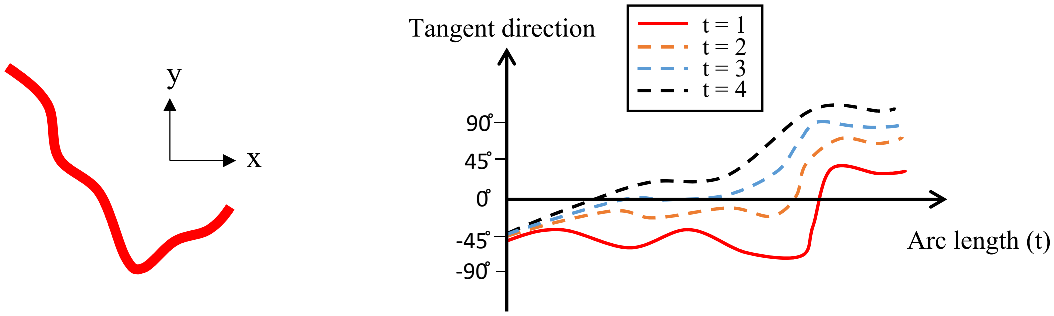
During training, the rope is manipulated from the initial shape to the final shape. This training procedure can be decomposed into several snapshots at different time frames. At each time frame, a tangent graph of the rope can be constructed (see Fig. 7.13b). At test time, the rope starts with a different initial shape and consequently, a new initial tangent graph. A transformation function $\mathcal{T}$ could be found in the tangent space that maps the initial tangent graph at training to the initial tangent graph at test. That same function $\mathcal{T}$ can be utilized to warp the tangent graphs at training to get the corresponding tangent graphs at test in subsequent time frames.

After getting the tangent information of the rope at test, the tangents are integrated along the arc length to convert the tangent information into position information in order to get the manipulation path that robot should follow at test time.



(a) Initial Shape of the Training Rope in Cartesian space

(b) Tangent Graphs of the Training Rope



(c) Initial Shape of the Test Rope in Cartesian space

(d) Tangent Graphs of the Test Rope

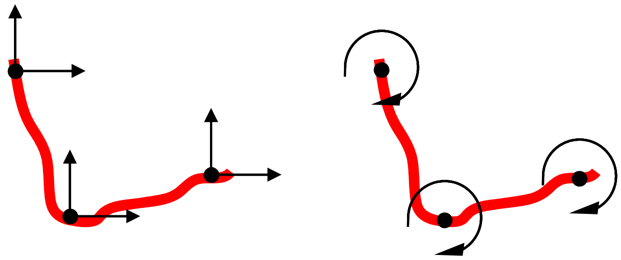
Figure 7.13: Rope in the Cartesian space and in the tangent space. $t = 1, 2, 3, 4$ represents four time frames of rope manipulation. $t = 1$ is the initial time, $t = 4$ is the final time.

T-SRP Algorithm

This subsection introduces the mathematical formulation of the T-SRP algorithm. Some notations of the tangent vectors are built up first. For a d-dimensional object living in n-dimensional space, $\phi_i^t \in \mathbb{R}^{n-d}$ denotes the training object's tangent vector at point i and at time frame t. Similarly, the test object's tangent vector can be represented by $\psi_j^t \in \mathbb{R}^{n-d}$, where j is the point index of the test object. The T-SRP algorithm is formulated as follows.
Step 1: Run SRP algorithm in the tangent space

(1) During training, extract the object's tangent vectors ϕ_i^t at each point i and each time frame t. At test, extract the test object's initial tangent vector $\psi_j^{t=1}$ at each point j.

(2) Calculate the tangent space correspondence matrix P and non-rigid transformation function $\mathcal{T}$ that register $\{\phi_i^{t=1}, i = 1, 2, \cdots, N\}$ to $\{\psi_j^{t=1}, j = 1, 2, \cdots, N'\}$ by implementing SPR.



(a) SRP (b) T-SRP

Figure 7.14: Illustration of the difference between SRP and T-SRP. SRP tries to reallocate the positions of each point on the rope, which can easily lead to overstretching and thus cause damage. T-SRP tries to reassign the joint angles of the rope. No matter how much each joint is twisted, the rope's original length is maintained.

Step 2: Run tangent space transferring

(1) Calculate the tangent vectors at test scene by non-rigid transformation from the second time frame $t = 2$ to the last time frame $t = t^{final}$

$$\psi_j^t = \sum_{i=1}^{N} P_{ij} \cdot \mathcal{T}(\phi_i^t) \qquad \forall t = 2, \cdots, t^{final} \tag{7.3}$$

where $P_{ij} = p(i|j)$ is the correspondence matrix between two sets found by registration (4.8).

(2) Calculate the grasping point at test. Suppose the robot grasps the point i_G of the object at training, then the grasping point at training can be calculated by the correspondence matrix

$$j_G = \arg\max_j P_{(i_G, j)} \tag{7.4}$$

(3) Integrate the test tangents to get the Cartesian space position of grasping point at each time frame

$$p_{j_G}^t = \sum_{j=1}^{j_G} \Psi_j^t \cdot \delta_j \tag{7.5}$$

where δ_j is the distance between neighbouring points. Note that $\Psi \in \mathbb{R}^n$ is an unit vector, equivalent to expressing ψ in n-dimensional space. Ψ is utilized here so that the dimension is consistent to the Cartesian space points. The new manipulation path is given by $\{p_{j_G}^t, t = 1, 2, \cdots, t^{final}\}$

In practice, it is found that the correspondence matrix P' that register the two point clouds in the Cartesian space can directly serve as a good tangent space transformation. This makes sense because the corresponding points of two objects in the Cartesian space usually share the similar tangent features. This special transformation is formulated as

$$\psi_j^t = \sum_{i=1}^{N} P'_{ij} \cdot \phi_i^t \qquad \forall t = 2, \cdots, t^{final} \tag{7.6}$$

Invariance Theorem of the T-SRP algorithm

Objects like ropes can be considered as curves in differential geometry. This subsection will show that transformations under T-SRP maintain the curve's structural information and thus keep the curve's length invariant during manipulation.

Theorem 5 *The length of the curve is invariant under the manipulation generated by T-SRP.*

Proof 5 *Assume the initial length of the curve at test is* $L_{test}^{t=1}$*. The length becomes* L_{test}^{t} *at time frame* t*. For any tangent vector* $\Psi^t \in \mathbb{R}^n$ *at time frame* t*, we have*

$$|\Psi^t| = 1 \tag{7.7}$$

Therefore,

$$\begin{aligned} L_{test}^{t} &= \int_0^{L_{test}^{t=1}} |\Psi^t| d\tau \\ &= \int_0^{L_{test}^{t=1}} 1 \cdot d\tau \\ &= L_{test}^{t=1} \end{aligned} \tag{7.8}$$

where $d\tau$ *is the differential arc length of the curve.*

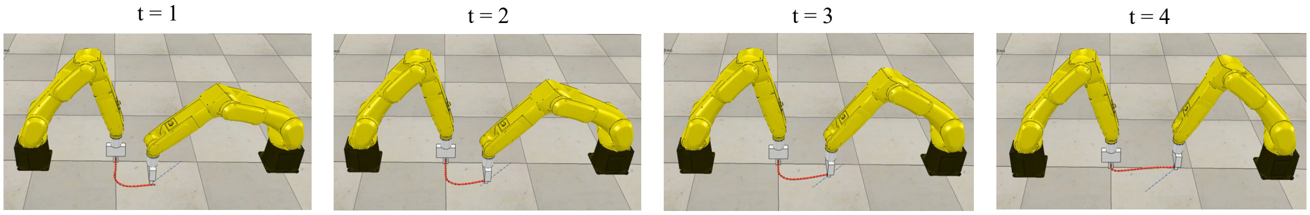
Equation (7.8) indicates the length of the test curve always keeps the initial value $L_{\text{test}}^{t=1}$, which is independent of the training curve's length L_{train}^{t}. This is a desired property since even if the curve length changes from training to test (for example, a long rope at training and a short rope at test), the algorithm will manipulate the test curve under the test length limitation despite of the training curve's length.

This invariance theorem can also be intuitively understood through Fig. 7.14. SRP regards the rope as a bunch of discrete points and tries to relocate the position of each point directly in the Cartesian space, which can lead to overstretching the rope during manipulation. In contrast, the T-SRP algorithm regards the rope as a long chain and instead rotates the joint angle of each chain in the tangent space. No matter how the angles are twisted, the rope's length does not change, i.e., length invariance. This is an important feature because otherwise safety cannot be guaranteed during robot manipulation.

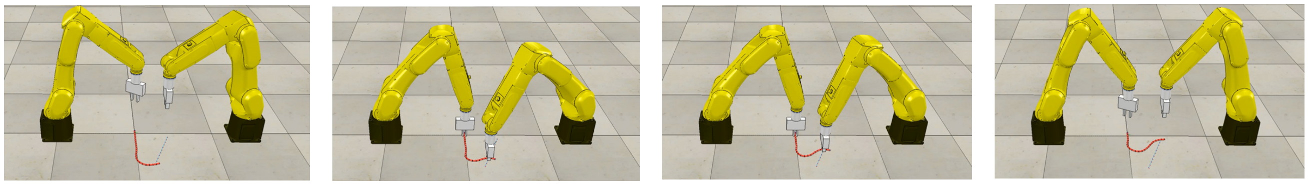
To conclude, a new path transformation method T-SRP is introduced. It is inspired by SRP, but different in nature. The key difference is that the transformation takes place in the tangent space instead of Cartesian space in order to maintain the object's structural information.

Simulation Study

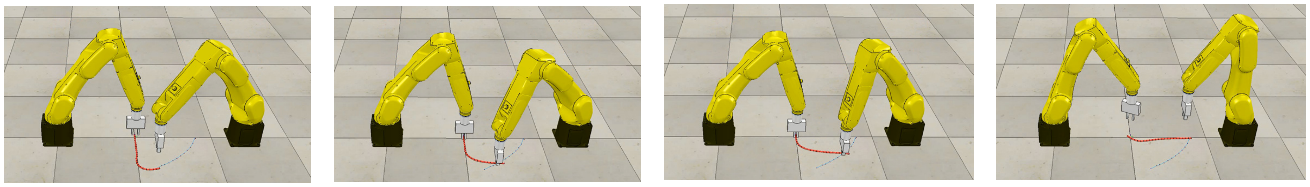
To test the performance of the proposed T-SRP algorithm, several rope manipulation tests are performed and the results are analyzed in this section. Supplementary videos can be found at [104].



(a) Train Scene



(b) Test Scene by SPR



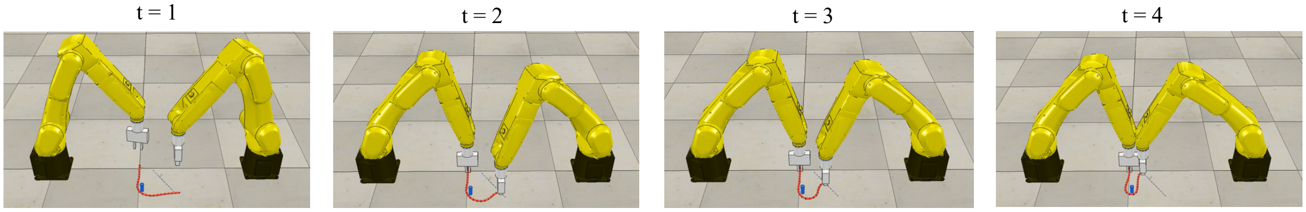
(c) Test Scene by T-SPR

Figure 7.15: Manipulating the rope into a straight line (horizontal direction) over four time frames, $t = 1, 2, 3, 4$. The initial shape of the rope is changed at test. SRP manipulates the test rope into a strange shape. T-SRP performs as well as training.

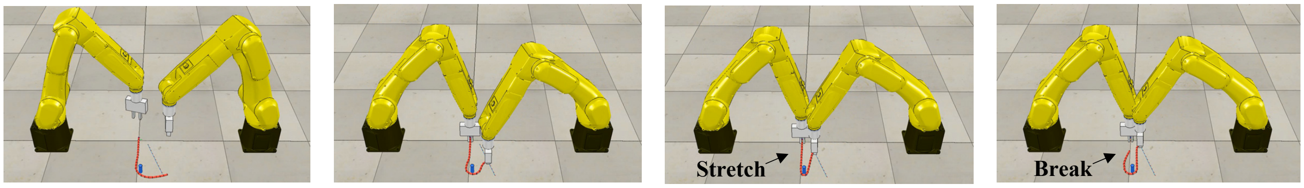
The manipulation tasks are simulated in V-REP [96] and the Bullet Physical Library [17] is selected as the physics engine. In these tests, two robot arms (FANUC LR Mate 200iD/7L) collaborate with each other to manipulate a single red rope of length 40cm. The flexible rope is modelled as twenty 2cm long cylinders connected sequentially by spherical joints. At training, one robot arm fixes one end of the rope, while the other arm is taught by human to move the free end to manipulate the rope into a desired shape. At test time, the initial pose and shape of the rope are altered on purpose. The T-SRP and the SRP algorithms generate new manipulation paths given the test rope's initial states. The correspondence matrix P and transformation function $\mathcal{T}$ are calculated by solving (4.8) and (4.36).

As shown in Fig. 7.15a, the first test is to manipulate a curved rope into a straight line. The initial shape of the rope at training is like a 'L', of which the two sides have equal lengths. The test rope is also in L-shape, but one side is longer than the other. During the test scene, the manipulation path (Fig. 7.15b) calculated by SRP moves the rope to the correct direction, but the final shape is curved and not similar to the one in training. The path calculated by T-SRP (Fig. 7.15c) not only moves the rope to the right direction, but also manipulates to accurately reflect what was demonstrated in training.

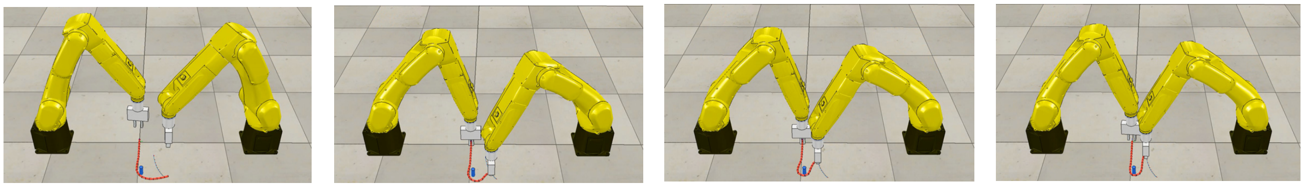
The second test is to wind the red rope around a blue shaft. It is desired to wind the rope tightly around the shaft without overstretching the rope during manipulation. Fig. 7.16 shows a segment of the winding task. The robot is taught to bend the rope ninety degrees at training phase (Fig. 7.16a). At test scene, the initial shape of the rope is changed. SRP generates a path to bend the new rope (Fig. 7.16b). However, at time frame $t = 3$, overstretch occurs and the rope breaks at time frame $t = 4$. In contrast, the T-SRP bends the test rope tightly around the shaft without overstretching the rope (Fig. 7.16c).



(a) Train Scene



(b) Test Scene by SPR



(c) Test Scene by T-SPR

Figure 7.16: Bending the rope ninety degrees around the shaft. The initial shape of the rope is changed at test. SRP overstretches the test rope at time frame $t = 3$ and the rope severs at $t = 4$. In contrast, T-SRP succeeds and does as well as training without incurring overstretching.

To conclude, these tests show that mapping in the tangent space make T-SRP have the ability to ‘memorize’ the shape of the object during training. At test, it will try to generate a path which recovers the object to the configuration demonstrated at training. Therefore the final shape is usually as good as training. Moreover, as proven in the invariance theorem, the manipulation path generated by T-SRP will not exceed the length limitation of the object and therefore guarantee safety during manipulation.

7.5 Chapter Summary

Chapter 7 discusses the application of analogy learning to teach robots the skill of efficient motion re-planning. Instead of regarding each scenario as an independent problem and solving separately, we try to find the correlation between current scenario and past solved scenarios. In this way, the past experience can be utilized and transformed to serve as a better initialization to solve the current motion planning problem. The bad local optimum for optimization can be avoided and the convergence rate can be improved. The proposed motion re-planning approach can be applied in configuration space, configuration-time space and tangent configuration space, so that it can perform various kinds of motion planning, including path planning, trajectory planning and shape-conservative manipulation planning. A series of simulations on a 6-DOF manipulator and a 2-DOF autonomous car are implemented, which prove the effectiveness of the proposed methods.

Chapter 8

Conclusions and Future Work

This dissertation introduced several methodologies for robot skill learning under three fundamental approaches: model-based approach, model-free approach and analogy-based approach. A series of applications including robotic assembly (Chapter 2, 3), grasping (Chapter 5), tracking (Chapter 6), manipulation (Chapter 6) and motion planning (Chapter 7) have been discussed and tested on industrial robots.

Chapter 2 introduces a novel contact model to predict the misalignment of mating parts from force measurement. Robotic assembly easily suffers from a three-point contact initial condition, where the peg has a large misalignment towards the hole, and the following insertion procedure cannot proceed. To deal with this problem, a model for peg-hole misalignment are derived by force and geometric analysis. The model parameter (friction coefficient) is regressed by nonlinear optimization. With the estimated misalignment values, a compensation trajectory is designed to realign the peg and hole to avoid three-point contact. Experimental results show that the misalignment between the peg and hole with H7h7 tolerance can be effectively eliminated and the robot can conduct peg-hole-insertion from three-point contact condition with 96% success rate. Future work can be conducted on the model extension. The current contact model is only designed for round pegs and holes. Though round mating parts are most common in industrial assembly, there are some scenarios such as corner assembly and dual-peg assembly that the current model cannot deal with. One possible solution is to construct different models and introduce a hyperparameter to switch between models for robotic autonomous assembly.

Chapter 3 presents a model-free approach to learn a force controller for peg-hole-insertion from human demonstration. Instead of designing or tuning a static admittance controller, Gaussian mixture regression (GMR) is utilized to learn the state-varying admittance directly from human demonstration data. A human demonstration device (HDD) is designed to collect the wrench and corrective velocity information during demonstration. The efficiency, physical interpretation and stability of the GMR admittance module are analyzed. A series of experiments performed on FANUC industrial robot and H7h7 tolerance testbed demonstrate the effectiveness of the proposed learning framework. However, as many other model-free methods, the performance of trained admittance controller is highly sensitive to the quality of training data. Future exploration may focus on how to quantitatively define the data quality so as to identify bad training data autonomously. Besides, how much amount of data is enough for training a satisfying controller is also an open question and needs to be explored.

Different from model-based or model-free approaches, a new robot learning approach, named analogy learning is proposed in Chapter 4. Instead of finding a control policy to relate the system states and corresponding actions, analogy leaning tries to find correlation between current scenario with past training scenarios. A past scenario that bears a strong similarity with the current situation will be identified and the past actions will be transferred and updated to apply on the current scenario. The benefits of analogy learning, including efficiency in data learning, robustness to input noise and guaranteed no collision are discussed as well. To support the implementation of analogy learning, a new non-rigid registration method, called structure preserved registration (SPR) is developed. SPR regards the past and current scenarios as point sets, and find a smooth non-rigid transformation function to register them to each other. The similarity value between two point sets can be quantitatively calculated.

Chapter 5 proposes a framework for efficient grasping pose generation by robotic analogy learning. A database containing multiple categories of source objects with demonstrated grasping poses were constructed by human experts. During the test scenario, a novel object was first classified into one of the example categories by measuring its dissimilarity to each source object. Then the grasping poses on the most similar source object were transferred to the novel object by the structure preserved registration (SPR) method. All the transferred grasping poses were evaluated and sorted by the grasp isotropy metric. The selected pose was further refined by an orientation search mechanism, which improves the robot reachability and avoids collision. A series of experiments were performed to grasp six categories of objects with various shapes, sizes and configurations. The average success rate was 18.8 out of 20 grasp trials. To improve the object registration speed, the point clouds observed from stereo cameras were downsampled to a sparser set. However, the downsampling may negatively influence the registration accuracy. At current stage, the downsampling ratio was manually tuned. Future work will focus on how to select an appropriate value to achieve a balance between speed and accuracy.

Based on the concept of analogy learning and SPR, a uniform framework, which includes state estimation, task planning and trajectory planning, is proposed in Chapter 6 for manipulating deformable objects. A real-time observer is developed to estimate the node position of the rope by registering the last step estimation towards the current point cloud measurement. A task planner is then developed to let robots recognize at which procedure the current manipulation is by registering training scenarios towards the test scenario. Finally, utilizing the transformation function constructed during scenario registration, the training trajectory can be warped to achieve a new trajectory which is feasible for the test. A series of experiments on rope knotting, cable assembly and clothes manipulation are implemented, which indicate the effectiveness of the proposed methods. Future work will focus on tracking multiple deformable objects at the same time, so as to make the proposed framework more practical in real scenarios.

Chapter 7 discusses the application of analogy learning on motion re-planning. Instead of regarding each scenario as an independent problem and solving separately, we try to find the relation between current scenario and past solved scenarios. In this way, the past experience can be utilized and transformed to serve as a better initialization to solve the current motion planning problem. The bad local optimum for optimization can be avoided and the convergence rate can be improved. The proposed motion planning approach can be applied in configuration space, configuration-time space and tangent configuration space, so that it can perform various kinds of motion planning, including path planning, trajectory planning and shape-conservative manipulation planning. A series of simulations on 6-DOF manipulator and 2-DOF autonomous car are conducted, the results of which show the effectiveness of the proposed methods. The current method has been tested to improve the efficiency of optimization-based motion planning. In the future, more study will be performed to combine the approach with grid-based and sampling-based motion planning algorithms, e.g., using the transformed planning results to guide the sampling of next planning problems.

Bibliography

[1] Pieter Abbeel and Andrew Y Ng. “Apprenticeship learning via inverse reinforcement learning”. In: *Proceedings of the twenty-first international conference on Machine learning*. ACM. 2004, p. 1.

[2] Brenna D Argall et al. “A survey of robot learning from demonstration”. In: *Robotics and autonomous systems* 57.5 (2009), pp. 469–483.

[3] Naoki Asakawa, Kenji Toda, and Yoshimi Takeuchi. “Automation of chamfering by an industrial robot; for the case of hole on free-curved surface”. In: *Robotics and Computer-Integrated Manufacturing* 18.5 (2002), pp. 379–385.

[4] Karl J Åström and Björn Wittenmark. *Adaptive control*. Courier Corporation, 2013.

[5] ATI Mini45 F/T sensor. `http://www.ati-ia.com/Products/ft`.

[6] Christopher G Atkeson and Stefan Schaal. “Robot learning from demonstration”. In: *ICML*. Vol. 97. 1997, pp. 12–20.

[7] Christopher G Atkeson et al. “Model-based robot learning”. In: (1988).

[8] Baxter from Rethink Robotics. *http://www.rethinkrobotics.com*.

[9] Darrin C Bentivegna, Christopher G Atkeson, and Gordon Cheng. “Learning tasks from observation and practice”. In: *Robotics and Autonomous Systems* 47.2-3 (2004), pp. 163–169.

[10] Jeff A Bilmes et al. “A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models”. In: *International Computer Science Institute* 4.510 (1998), p. 126.

[11] Christopher M Bishop. “Pattern Recognition”. In: *Machine Learning* (2006).

[12] Jeannette Bohg et al. “Data-driven grasp synthesisa survey”. In: *IEEE Transactions on Robotics* 30.2 (2014), pp. 289–309.

[13] Mariusz Bojarski et al. “End to end learning for self-driving cars”. In: *arXiv preprint arXiv:1604.07316* (2016).

[14] Torgny Brogårdh. “Present and future robot control developmentAn industrial perspective”. In: *Annual Reviews in Control* 31.1 (2007), pp. 69–79.

[15] Peter Brook, Matei Ciocarlie, and Kaijen Hsiao. "Collaborative grasp planning with multiple object representations". In: *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE. 2011, pp. 2851–2858.

[16] Herman Bruyninckx, Stefan Dutre, and Joris De Schutter. "Peg-on-hole: a model based solution to peg and hole alignment". In: *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on*. Vol. 2. IEEE. 1995, pp. 1919–1924.

[17] Bullet Physics Engine. `http://bulletphysics.org/wordpress`. 2018.

[18] Eduardo F Camacho and Carlos Bordons Alba. *Model predictive control*. Springer Science & Business Media, 2013.

[19] Jaime G Carbonell. "Learning by analogy: Formulating and generalizing plans from past experience". In: *Machine Learning, Volume I*. Elsevier, 1983, pp. 137–161.

[20] Yutuo Chen et al. "Intelligent robotic peg-in-hole insertion learning based on haptic virtual environment". In: *Computer-Aided Design and Computer Graphics, 2007 10th IEEE International Conference on*. IEEE. 2007, pp. 355–360.

[21] Siddharth R Chhatpar and Michael S Branicky. "Search strategies for peg-in-hole assemblies with position uncertainty". In: *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*. Vol. 3. IEEE. 2001, pp. 1465–1470.

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

[23] ChuiDataset. *Data set for testing non-rigid registration*. `http://legacydirs.umiacs.umd.edu/\%7Ezhengyf/PointMatching`. 2000.

[24] John J Craig. *Introduction to robotics: mechanics and control*. Vol. 3. Pearson Prentice Hall Upper Saddle River, 2005.

[25] Hao Dang and Peter K Allen. "Semantic grasping: Planning robotic grasps functionally suitable for an object manipulation task". In: *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE. 2012, pp. 1311–1317.

[26] Kenny Daniel et al. "Theta*: Any-angle path planning on grids". In: *Journal of Artificial Intelligence Research* 39 (2010), pp. 533–579.

[27] Arthur P Dempster, Nan M Laird, and Donald B Rubin. "Maximum likelihood from incomplete data via the EM algorithm". In: *Journal of the royal statistical society. Series B (methodological)* (1977), pp. 1–38.

[28] Renaud Detry et al. "Generalizing grasps across partly similar objects". In: *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE. 2012, pp. 3791–3797.

[29] Richard C Dorf and Robert H Bishop. *Modern control systems*. Pearson, 2011.

[30] Kenji Doya et al. "Multiple model-based reinforcement learning". In: *Neural computation* 14.6 (2002), pp. 1347–1369.

[31] Martin Ester et al. "A density-based algorithm for discovering clusters in large spatial databases with noise." In: *Kdd*. Vol. 96. 34. 1996, pp. 226–231.

[32] FANUC America, LR Mate 200*i*D/7L. `http://robot.fanucamerica.com/products/robots/productbyseries.aspx?seriesId=3\&robotseries=LR\%20-Mate\%20Series`.

[33] Dave Ferguson and Anthony Stentz. "Field D*: An interpolation-based path planner and replanner". In: *Robotics research*. Springer, 2007, pp. 239–253.

[34] Carlo Ferrari and John Canny. "Planning optimal grasps". In: *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*. IEEE. 1992, pp. 2290–2295.

[35] Jonathan D Gammell, Siddhartha S Srinivasa, and Timothy D Barfoot. "Informed RRT*: Optimal sampling-based path planning focused via direct sampling of an admissible ellipsoidal heuristic". In: *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*. IEEE. 2014, pp. 2997–3004.

[36] Song Ge, Guoliang Fan, and Meng Ding. "Non-rigid point set registration with global-local topology preservation". In: *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on*. IEEE. 2014, pp. 245–251.

[37] Federico Girosi, Michael Jones, and Tomaso Poggio. "Regularization theory and neural networks architectures". In: *Neural computation* 7.2 (1995), pp. 219–269.

[38] Leslie Greengard and John Strain. "The fast Gauss transform". In: *SIAM Journal on Scientific and Statistical Computing* 12.1 (1991), pp. 79–94.

[39] Peter E Hart, Nils J Nilsson, and Bertram Raphael. "A formal basis for the heuristic determination of minimum cost paths". In: *IEEE transactions on Systems Science and Cybernetics* 4.2 (1968), pp. 100–107.

[40] John A Hartigan and Manchek A Wong. "Algorithm AS 136: A k-means clustering algorithm". In: *Applied statistics* (1979), pp. 100–108.

[41] W Haskiya, K Maycock, and J Knight. "Robotic assembly: chamferless peg-hole assembly". In: *Robotica* 17.6 (1999), pp. 621–634.

[42] Max-Olivier Hongler et al. "A random exploration approach for automatic chamferless insertion". In: *The International journal of robotics research* 14.2 (1995), pp. 161–173.

[43] Ronald A Howard. *Dynamic programming and Markov processes*. Wiley for The Massachusetts Institute of Technology, 1964.

[44] Shouren Huang et al. "Fast peg-and-hole alignment using visual compliance". In: *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE. 2013, pp. 286–292.

[45] Shouren Huang et al. "Realizing peg-and-hole alignment with one eye-in-hand high-speed camera". In: *Advanced Intelligent Mechatronics (AIM), 2013 IEEE/ASME International Conference on.* IEEE. 2013, pp. 1127–1132.

[46] Impulse Motion Capture System from PhaseSpace Inc. `http://www.phasespace.com/index.html`.

[47] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. "Reinforcement learning: A survey". In: *Journal of artificial intelligence research* 4 (1996), pp. 237–285.

[48] Sertac Karaman and Emilio Frazzoli. "Incremental sampling-based algorithms for optimal motion planning". In: *Robotics Science and Systems VI* 104 (2010), p. 2.

[49] Lydia E Kavraki et al. "Probabilistic roadmaps for path planning in high-dimensional configuration spaces". In: *IEEE transactions on Robotics and Automation* 12.4 (1996), pp. 566–580.

[50] S Mohammad Khansari-Zadeh, Klas Kronander, and Aude Billard. "Modeling robot discrete movements with state-varying stiffness and damping: A framework for integrated motion generation and impedance control". In: *Proceedings of Robotics: Science and Systems X (RSS 2014)* (2014).

[51] Byoung-Ho Kim et al. "Optimal grasping based on non-dimensionalized performance indices". In: *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on.* Vol. 2. IEEE. 2001, pp. 949–956.

[52] Ulrich Klank et al. "Real-time cad model matching for mobile manipulation and grasping". In: *Humanoid Robots, 2009. Humanoids 2009. 9th IEEE-RAS International Conference on.* IEEE. 2009, pp. 290–296.

[53] Jens Kober and Jan Peters. "Reinforcement learning in robotics: A survey". In: *Reinforcement Learning.* Springer, 2012, pp. 579–610.

[54] Thomas Kollar and Nicholas Roy. "Trajectory optimization using reinforcement learning for map exploration". In: *The International Journal of Robotics Research* 27.2 (2008), pp. 175–196.

[55] Klas Kronander, Etienne Burdet, and Aude Billard. "Task transfer via collaborative manipulation for insertion assembly". In: *Workshop on Human-Robot Interaction for Industrial Manufacturing, Robotics, Science and Systems.* Citeseer. 2014.

[56] Marek Kuczma. *An introduction to the theory of functional equations and inequalities: Cauchy's equation and Jensen's inequality.* Springer Science & Business Media, 2009.

[57] Shunsuke Kudoh et al. "In-air Knotting of Rope by a Dual-arm Multi-finger Robot". In: *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on.* IEEE. 2015, pp. 6202–6207.

[58] James J Kuffner and Steven M LaValle. “RRT-connect: An efficient approach to single-query path planning”. In: *Robotics and Automation, 2000. Proceedings. ICRA’00. IEEE International Conference on*. Vol. 2. IEEE. 2000, pp. 995–1001.

[59] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep learning”. In: *nature* 521.7553 (2015), p. 436.

[60] Alex X Lee et al. “Unifying scene registration and trajectory optimization for learning from demonstrations with application to manipulation of deformable objects”. In: *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*. IEEE. 2014, pp. 4402–4407.

[61] Ian Lenz, Honglak Lee, and Ashutosh Saxena. “Deep learning for detecting robotic grasps”. In: *The International Journal of Robotics Research* 34.4-5 (2015), pp. 705–724.

[62] Sergey Levine, Nolan Wagener, and Pieter Abbeel. “Learning contact-rich manipulation skills with guided policy search”. In: *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE. 2015, pp. 156–163.

[63] Sergey Levine et al. “End-to-end training of deep visuomotor policies”. In: *The Journal of Machine Learning Research* 17.1 (2016), pp. 1334–1373.

[64] Yinxiao Li et al. “Model-Driven Feedforward Prediction for Manipulation of Deformable Objects”. In: *IEEE Transactions on Automation Science and Engineering* (2018).

[65] Chu-Min Liao and Richard SW Masters. “Analogy learning: A means to implicit motor learning”. In: *Journal of sports sciences* 19.5 (2001), pp. 307–319.

[66] Hsien-Chung Lin, Te Tang, et al. “Remote Lead Through Teaching by Human Demonstration Device”. In: *ASME 2015 Dynamic Systems and Control Conference*. American Society of Mechanical Engineers. 2015.

[67] Hsien-Chung Lin* et al. “A Framework for Robot Grasping Transferring with Non-rigid Transformation”. In: *Intelligent Robots and Systems (IROS), 2018 IEEE/RSJ International Conference on*. IEEE. in review.

[68] Changliu Liu, Chung-Yen Lin, and Masayoshi Tomizuka. “The convex feasible set algorithm for real time optimization in motion planning”. In: *SIAM Journal on Control and Optimization* (2016), in review.

[69] Changliu Liu, Chung-Yen Lin, and Masayoshi Tomizuka. “The convex feasible set algorithm for real time optimization in motion planning”. In: *arXiv preprint:1709.00627* (2017).

[70] Changliu Liu and Masayoshi Tomizuka. “Real time trajectory optimization for nonlinear robotic systems: Relaxation and convexification”. In: *Systems & Control Letters* 108 (2017), pp. 56–63.

[71] Changliu Liu* et al. “Introducing SERoCS: Safe and Efficient Robot Collaborative Systems for Next Generation Intelligent Industrial Co-Robots”. In: *Robotics and Computer-Integrated Manufacturing* (2018).

[72] Yun-Hui Liu. “Computing n-finger form-closure grasps on polygonal objects”. In: *The International journal of robotics research* 19.2 (2000), pp. 149–158.

[73] Yun-Hui Liu. “Qualitative test and force optimization of 3-D frictional form-closure grasps using linear programming”. In: *IEEE Transactions on Robotics and Automation* 15.1 (1999), pp. 163–173.

[74] Steve Lohr. “The age of big data”. In: *New York Times* 11.2012 (2012).

[75] Jeffrey Mahler et al. “Dex-net 1.0: A cloud-based network of 3d objects for robust grasp planning using a multi-armed bandit model with correlated rewards”. In: *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE. 2016, pp. 1957–1964.

[76] Jeffrey Mahler et al. “Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics”. In: *arXiv preprint arXiv:1703.09312* (2017).

[77] Volodymyr Mnih et al. “Playing atari with deep reinforcement learning”. In: *arXiv preprint arXiv:1312.5602* (2013).

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

[79] Takuma Morita et al. “Knot planning from observation”. In: *Robotics and Automation, 2003. Proceedings. ICRA’03. IEEE International Conference on*. Vol. 3. IEEE. 2003, pp. 3887–3892.

[80] Seyed Sajad Mousavi, Michael Schukat, and Enda Howley. “Deep reinforcement learning: an overview”. In: *Proceedings of SAI Intelligent Systems Conference*. Springer. 2016, pp. 426–440.

[81] Richard M Murray et al. *A mathematical introduction to robotic manipulation*. CRC press, 1994.

[82] Andriy Myronenko and Xubo Song. “Point set registration: Coherent point drift”. In: *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 32.12 (2010), pp. 2262–2275.

[83] Ashvin Nair et al. “Combining self-supervised learning and imitation for vision-based rope manipulation”. In: *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE. 2017, pp. 2146–2153.

[84] David Navarro-Alarcon et al. “Automatic 3-D manipulation of soft objects by robotic arms with an adaptive deformation model”. In: *IEEE Transactions on Robotics* 32.2 (2016), pp. 429–441.

[85] David Navarro-Alarcon et al. “On the visual deformation servoing of compliant objects: Uncalibrated control methods and experiments”. In: *The International Journal of Robotics Research* 33.11 (2014), pp. 1462–1480.

[86] Van-Duc Nguyen. “Constructing force-closure grasps”. In: *The International Journal of Robotics Research* 7.3 (1988), pp. 3–16.

[87] Duy Nguyen-Tuong and Jan Peters. “Model learning for robot control: a survey”. In: *Cognitive processing* 12.4 (2011), pp. 319–340.

[88] Jeanne Ellis Ormrod and Kevin M Davis. *Human learning*. Merrill London, 2004.

[89] Chavdar Papazov et al. “Rigid 3D geometry matching for grasping of known objects in cluttered scenes”. In: *The International Journal of Robotics Research* 31.4 (2012), pp. 538–553.

[90] Leif E Peterson. “K-nearest neighbor”. In: *Scholarpedia* 4.2 (2009), p. 1883.

[91] Lerrel Pinto and Abhinav Gupta. “Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours”. In: *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE. 2016, pp. 3406–3413.

[92] Jean Ponce and Bernard Faverjon. “On computing three-finger force-closure grasps of polygonal objects”. In: *IEEE Transactions on robotics and automation* 11.6 (1995), pp. 868–881.

[93] PR2 from Willow Garage. *https://www.willowgarage.com*.

[94] Marc H Raibert and John J Craig. “Hybrid position/force control of manipulators”. In: *Journal of Dynamic Systems, Measurement, and Control* 103.2 (1981), pp. 126–133.

[95] Nathan Ratliff et al. “CHOMP: Gradient optimization techniques for efficient motion planning”. In: *Robotics and Automation, 2009. ICRA’09. IEEE International Conference on*. IEEE. 2009, pp. 489–494.

[96] Eric Rohmer, Surya PN Singh, and Marc Freese. “V-REP: A versatile and scalable robot simulation framework”. In: *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE. 2013, pp. 1321–1326.

[97] Hiroaki Sakoe and Seibi Chiba. “Dynamic programming algorithm optimization for spoken word recognition”. In: *Acoustics, Speech and Signal Processing, IEEE Transactions on* 26.1 (1978), pp. 43–49.

[98] John Schulman et al. “Learning from Demonstrations through the Use of Non-Rigid Registration”. In: *in Proceedings of the 16th International Symposium on Robotics Research (ISRR)*. 2013.

[99] 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.

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

[101] David Silver et al. "Mastering the game of Go with deep neural networks and tree search". In: *nature* 529.7587 (2016), pp. 484–489.

[102] S Skiena. "Dijkstras algorithm". In: *Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica, Reading, MA: Addison-Wesley* (1990), pp. 225–227.

[103] Jean-Jacques E Slotine, Weiping Li, et al. *Applied nonlinear control*. Vol. 199. 1. Prentice-Hall Englewood Cliffs, NJ, 1991.

[104] Supplementary website for the dissertation. `http://me.berkeley.edu/~tetang/Dissertation`.

[105] Te Tang, Hsien-Chung Lin, and Masayoshi Tomizuka. "A Learning-Based Framework for Robot Peg-Hole-Insertion". In: *ASME 2015 Dynamic Systems and Control Conference*. American Society of Mechanical Engineers. 2015, V002T27A002–V002T27A002.

[106] Te Tang and Masayoshi Tomizuka. "Track Deformable Objects from Point Clouds with Structure Preserved Registration". In: *The International Journal of Robotics Research* (2018), in review.

[107] Te Tang*, Changhao Wang*, and Masayoshi Tomizuka. "A Framework for Manipulating Deformable Linear Objects by Coherent Point Drift". In: *IEEE Robotics and Automation Letters (RA-L)* (2018), to appear.

[108] Te Tang et al. "Autonomous alignment of peg and hole by force/torque measurement for robotic assembly". In: *Automation Science and Engineering (CASE), 2016 IEEE International Conference on*. IEEE. 2016, pp. 162–167.

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

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

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

[112] The Robot Operating System (ROS). http://www.ros.org/.

[113] Jacob Varley et al. "Generating multi-fingered robotic grasps via deep learning". In: *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE. 2015, pp. 4415–4420.

[114] Miomir Vukobratovic, Veljko Potkonjak, and Vladimir Matijevic. *Dynamics of robots with contact tasks*. Vol. 26. Springer Science & Business Media, 2013.

[115] Daniel E Whitney. “Historical perspective and state of the art in robot force control”. In: *The International Journal of Robotics Research* 6.1 (1987), pp. 3–14.

[116] DE Whitney and JL Nevins. *What is the remote center compliance (RCC) and what can it do?* Charles Stark Draper Laboratory, 1978.

[117] Ping Wu et al. “Vision-Based Robot Path Planning with Deep Learning”. In: *International Conference on Computer Vision Systems*. Springer. 2017, pp. 101–111.

[118] Shichao Yang et al. “Obstacle avoidance through deep networks based intermediate perception”. In: *arXiv preprint arXiv:1704.08759* (2017).

[119] Zhenyue Zhang and Jing Wang. “MLLE: Modified locally linear embedding using multiple weights”. In: *Advances in neural information processing systems*. 2007, pp. 1593–1600.

[120] Kemin Zhou, John Comstock Doyle, Keith Glover, et al. *Robust and optimal control*. Vol. 40. Prentice hall New Jersey, 1996.