Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Robotic In-hand Manipulation with Push and Support Method

Doctoral Thesis
in Group: Technical Aspects of Multimodal Systems, TAMS
Department of Informatik
MIN-Faculty
University of Hamburg

Submitted by
Junhu He

Hamburg, 2017

Reviewers: Prof. Dr. Jianwei Zhang
Department of Informatik
MIN-Faculty
University of Hamburg

Prof. Dr. Stefan Wermter
Department of Informatik
MIN-Faculty
University of Hamburg

Date of the oral defense: 28. 04. 2017

Abstract

With their ability to adapt to human tools, robots with anthropomorphic hands have great potential of being employed in domestic environments to facilitate various domestic tasks. In-hand manipulation is one of the distinctive skills in anthropomorphic hands. It is a process in which fingers push the object to generate expected manipulations. Although lots of research has been done on this topic, in many respects this is still a challenge in robotics. To begin with, the control of the robotic hand is a challenge. An anthropomorphic robotic hand generally has a large number of joints (more than 10 DOFs). The in-hand system which consists of the object and the fingers is redundant. Then the interaction state is complicated, which is hard to be modeled precisely. A small position error on the finger may change the contact force on the object dramatically. Therefore, a small error may cause the task to fail. The perception capability of the in-hand manipulation system is limited. The contact sensing on the fingertips is usually limited by sensors' restricted size, and the visual information of the object is usually occluded by fingers. As a result of these challenges, the in-hand object is assumed to be 'unknown' for the robotic system.

In this research, we focus on manipulating an unknown object with an anthropomorphic robotic hand. The target objects and the support fingers are treated as one black box system where action commands are sent as input and the observed visual-haptic feedback is output. In this black box, the process of the in-hand manipulation is a finger (push finger) pushing an object to rotate on other fingers (support fingers). To simplify this problem, the object and the support fingers are treated as one black box. In this black box, an unknown object is located on an elastic surface consisting of the support fingers. Hence, the in-hand manipulation is transferred into a process where the push finger pushes an unknown object to roll onto an elastic surface. In this process, visual-haptic feedback is observed to evaluate the push performance. To make the elasticity of the surface controllable, two kinds of support fingers are proposed in this thesis: a fixed support finger and a spring support finger. The fixed support finger is controlled by a position controller, and its position is fixed as a pivot around which the object rotates. The spring support finger performs as an elastic spring. It presses the object against other fingers and helps to rotate the object with proper contact force. More importantly, an error transfer matrix is given, which proves that positional errors in different fingers can be compensated by one spring support finger. Combining different kinds of support fingers, a single support model and a hybrid support model are proposed to carry out the robotic in-hand manipulations. As a result, the proposed models reduce the complexity of the in-hand manipulation system dramatically.

Without any knowledge of the black box, the robot cannot conduct successful in-hand manipulations. To acquire sufficient knowledge of the in-hand system, a process called haptic exploration is proposed in this thesis. In this process, the robot slightly pushes the object in different directions and estimates the interaction state from haptic feedback. Besides, in-hand manipulation experiments have been conducted successfully on a real anthropomorphic hand platform based on this concept. In these experiments, both visual and haptic feedback has been collected and researched. The experiments' result shows the feasibility of the proposed manipulation method and models. Specifically, in the multi-directional manipulation experiment, sufficient knowledge has been acquired; and the knowledge is used to guide the further in-hand manipulations in other tasks. Furthermore, from the results of these in-hand manipulation experiments, it can be concluded that haptic sensing is capable of taking the place of visual sensing. This result supports the idea of haptic exploration.

Reinforcement Learning (RL) has been adopted to improve the in-hand manipulation skill automatically. RL agents have been built to generate the in-hand manipulation push commands. Besides, in-hand manipulation simulators are constructed with Radial Basis Function Networks (RBFN) trained by real manipulation data for the purpose of taking the place of the real in-hand manipulation system. The agents improve step by step by interacting with the simulators. In this thesis, two stochastic RL algorithms are adopted: Williams' Episodic REINFORCE and Peters' Episodic Natural Actor-Critic. And finally, the learning experiments have been conducted based on different rewards: visual only rewards (unimodal) and visual-haptic rewards (multimodal). The experimental results demonstrate that our learning method is feasible; moreover, the use of multimodal rewards speeds up the learning process compared to the result from the use of unimodal rewards.

Zusammenfassung

Weil Roboter mit anthropomorphen Händen sich an menschliche Werkzeuge naturgemäß anpassen, eignen sie sich besonders für den Einsatz in Häusern, um Menschen bei der Hausarbeit zu unterstützen. In-Hand-Manipulation ist eine der charakteristischen Fähigkeiten anthropomorpher Hände. Es ist ein Prozess in dem Finger das Objekt drücken, um erwartete Manipulationen zu erzeugen. Obwohl hier umfassend geforschrt wurde, stellt dies immer noch eine Herausforderung in der Robotik dar.

In dieser Arbeit wird die in-Hand-Manipulation als Finger (Push-Finger) betrachtet, die ein Objekt auf andere Finger (Stützfinger) schieben, um es dort zu rotieren. Um dieses Problem zu vereinfachen, werden das Objekt und die Stützfinger als eine Blackbox behandelt. In dieser schwarzen Box befindet sich ein unbekannter Gegenstand auf einer elastischen Oberfläche, die aus den Stützfingern besteht. Folglich wird die In-Hand-Manipulation in einen Prozess übertragen, in dem der Druckfinger ein unbekanntes Objekt drückt, um es auf die elastische Oberfläche zu rollen. In diesem Prozess wird visuell-haptisches Feedback beobachtet, um die Push-Leistung zu bewerten. Zur Veranschaulichung der Feedbacks werden in dieser Arbeit zwei Stützfinger vorgeschlagen: fester Stützfinger und Federstützfinger. Der feste Stützfinger ist unkontrolliert und fixiert als Drehpunkt, um den sich das Objekt dreht. Der Federstützfinger dient als elastische Feder. Es drückt das Objekt gegen andere Finger und hilft, das Objekt mit der richtigen Kontaktkraft zu drehen. Noch wichtiger ist, dass eine Fehlertransfermatrix gegeben ist, die theoretisch beweist, dass Positionsfehler in verschiedenen Fingern durch einen Federstützfinger kompensiert werden können. Es werden verschiedene Stützfinger, ein einzelnes Stützmodell und ein hybrides Stützmodell kombiniert, um robotische In-Hand Manipulationen durchzufhren. Infolgedessen reduzieren die vorgeschlagenen Modelle die Komplexität des Manipulationssystems drastisch.

Ohne Kenntnis der Blackbox kann der Roboter keine erfolgreichen Manipulationen durchführen. Um genügend Wissen zu erwerben, wurde in dieser Arbeit ein Prozess namens haptische Erforschung vorgeschlagen. Dabei drückt der Roboter das Objekt leicht in verschiedene Richtungen und schätzt den Interaktionszustand des haptischen Feedback ein. Darüber hinaus wurden Hand-Manipulationsexperimente auf einer realen anthropomorphen Handplattform, die auf diesem Konzept basiert, erfolgreich durchgeführt. In diesen Experimenten wurde sowohl visuelles als auch haptisches Feedback gesammelt und erforscht. Das Ergebnis der Experimente zeigt, dass die vorgeschlagene Manipulationsmethode und die Modelle funktionieren. Bei diesen Experimenten, insbesondere beim multidirektionalen Manipulationsexperiment, wurde ausreichend Wissen erworben. Dieses wird verwendet, um weitere Manipulationen auch in anderen Aufgaben anzuleiten. Darüber hinaus kann gefolgert werden, dass die haptische Wahrnehmung in der Lage ist, die Stelle der visuellen Wahrnehmung in diesen Manipulationsexperimenten zu übernehmen, was die Idee der haptischen Erforschung unterstützt.

Weiterhin wurden Verfahren des Reinforcement Learning (RL) angewandt, um die Fähigkeiten der In-Hand-Manipulation automatisch zu verbessern. RL-

Agenten wurden aufgebaut, um die Manipulations-Push-Befehle zu erzeugen. Außerdem wird ein Manipulationssimulator mit Radial-Basis-Funktionsnetzwerken (RBFN) konstruiert, die durch reale Manipulationsdaten erzeugt werden, um an die Stelle des realen Manipulationssystems zu treten. Die Agenten verbessern sich Schritt für Schritt durch Interaktion mit dem Simulator. In dieser Arbeit werden zwei stochastische RL-Algorithmen angenommen: Williams ' Episodische REINFORCE and Peters' Episodischer Naturschauspieler-Kritiker. Schließlich wurden die Lernversuche auf der Grundlage verschiedener Belohnungen durchgeführt: visuelle alleinige Belohnungen und visuelle haptische Belohnungen. Die Forschungsergebnisse zeigen, dass unsere Lernmethode durchführbar ist. Darüber hinaus beschleunigt der Einsatz multimodaler (visueller und haptischer) Belohnungen den Lernprozess im Vergleich zu einzelnen modalen Belohnungen.

Contents

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Problem Statement . . . 4
  - 1.2.1 Large Joint Number . . . 4
  - 1.2.2 Complex Interaction Models . . . 5
  - 1.2.3 Limited Perception Capabilities . . . 5
- 1.3 Objective and Proposed Approach . . . 6
  - 1.3.1 Objective . . . 6
  - 1.3.2 Concept and Approaches . . . 7
  - 1.3.3 Push Manipulation Architecture . . . 10
- 1.4 Contributions . . . 12
- 1.5 Outline . . . 13

2 State of The Art **15**
- 2.1 Human Grasping and Manipulation . . . 16
  - 2.1.1 Grasping Classification . . . 16
  - 2.1.2 Manipulation Classification . . . 18
  - 2.1.3 Tactile perception in Human Manipulations . . . 18
- 2.2 Anthropomorphic Robot Hands . . . 20
  - 2.2.1 Anthropomorphic Robot Hand . . . 20
  - 2.2.2 Sensing Capabilities of Robotic Hand . . . 21
- 2.3 Physical Model-based Manipulation . . . 21
  - 2.3.1 Explicit Interaction Model . . . 21
  - 2.3.2 Virtual Object Models . . . 23
  - 2.3.3 Contact Models . . . 25
- 2.4 Manipulation with Synergies . . . 28
- 2.5 Multimodal in Robotic In-hand Manipulation . . . 30
- 2.6 Intelligent Manipulation . . . 32
  - 2.6.1 POMDP . . . 33
  - 2.6.2 Machine Learning . . . 33
  - 2.6.3 Learning for State Estimation . . . 36
  - 2.6.4 Learning for Action Optimal Control . . . 37

3 Manipulation Models **39**
3.1 Contact Models and Grasping Matrix 40
3.1.1 Contact Models 40
3.1.2 Grasping Matrix 42
3.2 Push and Support Manipulation Models 43
3.2.1 Elastic Surface Model 43
3.2.2 Discussion on Push and Support Manipulation Model 44
3.3 Single Support Model 47
3.3.1 Contact Force 47
3.3.2 Object's Movement 49
3.3.3 Object's Two Roll Cases 50
3.4 Hybrid Support Finger Model 51
3.4.1 Contact Force 51
3.4.2 Object's Movement 53
3.4.3 Control of Spring Support Finger 55
3.4.4 Error Compensated with One Spring Support Finger 56
3.5 Enhanced Manipulation Model 61
3.6 3D Manipulation Discussion 63

4 Manipulation Experiments **65**
4.1 Experiment Setups 66
4.1.1 Anthropomorphic robot hand 66
4.1.2 BioTac Tactile Sensor 67
4.1.3 Vision Tracking System 68
4.2 Experimental Procedure 70
4.2.1 Initial Grasping Configuration and Push Process 70
4.2.2 Automatic Manipulation 71
4.2.3 Object Frame 71
4.2.4 Haptic and Visual Features Extraction 74
4.3 Repeatability Experiment 76
4.4 Multi-Push Distance Experiment 78
4.5 Multi-Directional Manipulation Experiment 81
4.6 Hybrid Support Manipulation Experiments 89
4.6.1 Push Direction 91
4.6.2 Push Distance and Contact Force 91
4.7 Robustness Experiments 94
4.8 Enhanced Manipulation Experiments 98
4.8.1 Manipulating a Rigid Object 98
4.8.2 Manipulating Various Objects 99
4.9 Comparing to Virtual Frame Model 100
4.9.1 Manipulating with Virtual Frame Model 101
4.9.2 Comparative Results 103

5 Learn to Improve Manipulation Skills **107**
5.1 Learning Framework . . . 108
5.2 In-hand Manipulation Simulator . . . 111
5.2.1 Radial Basis Function Networks . . . 111
5.2.2 Density Push Experiment for Yaw Manipulation . . . 112
5.2.3 Manipulation Simulator based on RBFNs . . . 112
5.3 Policy Based Reinforcement Learning . . . 117
5.3.1 General Assumption and Problem Statement . . . 117
5.3.2 Reinforcement Learning Categories . . . 120
5.3.3 Stochastic Policy Gradient Learning . . . 122
5.4 Manipulation Learning Experiments . . . 127
5.4.1 Learning Rewards . . . 127
5.4.2 Learning with Visual-Only Rewards . . . 128
5.4.3 Learning with Visual-Haptic Rewards . . . 131
5.4.4 Discussion of the Learning Results . . . 133

6 Conclusion **137**
6.1 Summary . . . 137
6.2 Limitation and Discussion . . . 139
6.3 Future outlook . . . 140

A Nomenclature **143**

B Density Push Result **145**

C Approximate with MLP neural network **147**

D Acknowledgments **149**

E Publications **151**

Bibliography **168**

List of Figures

1.1 A robot chef cooks Michelin star food. 2
1.2 Nextage envisioned by Kwanda Industries. 3
1.3 A water tap. 6
1.4 Black box. This diagram shows the basic research concept of in-hand manipulations. In this black box, an unknown object is located on fingers other than the push finger. Therefore the manipulation is transferred into rolling the object on the surface of other fingers. . . 7
1.5 Haptic exploration process. 9
1.6 Push manipulation architecture. This diagram illustrates the push and support manipulation. The yellow dash rectangles denote the components in this system, and the red arrows refer to the sensing signals. A push manipulation process is described as follows. At the beginning, a planner configures the support fingers by selecting proper controllers on them. After that, push commands are given to the push finger. According to these commands, the finger pushes the unknown object to roll onto the support fingers. In this process, the object is tracked and recorded by a visual tracking system and the contact state is recorded with the tactile sensors. 11

2.1 Cutkosy's grasp taxonomy [50]. 17
2.2 An anthropomorphic robot hand: Shadow Hand. 20
2.3 An explicit model for high-speed manipulation [64]. 22
2.4 Object virtual frame. 23
2.5 Object virtual frame. 24
2.6 A kinematic structure of virtual linkages. 24
2.7 Friction cones. 25
2.8 Elastic hemispherical fingertip contact model [80]. 26
2.9 The rolling contact cases in [129]. $\mathbf{u}_1$ is the contact frame fixed on the object 1 (finger), and $\mathbf{u}_1$ is the contact frame fixed on the object 2 (manipulating object). After rolling forward, the new contact frames will be $\mathbf{u}'_1$ and $\mathbf{u}'_2$ respectively. 27
2.10 Contact spot model. 28
2.11 Three kinds of fingertip sensors. 31
2.12 The architecture of control policy in LfD (Learning from Demonstration). 34

2.13 Reinforcement learning where an RL agent interacts with its environment. 35
2.14 The architecture of the grasping neural model proposed by Gorce and Rezzoug. 37

3.1 A fundamental concept model. In this mode, the manipulation system is considered as a finger pushes a ball which is located on an elastic surface. 43
3.2 The evolution of the elastic surface model. 46
3.3 The contact force in single support model. In this model two overlapped support fingers are viewed as one fixed support finger. . . . 47
3.4 Push stiffness mode in dexterous manipulation tasks 49
3.5 Hybrid support finger model. 52
3.6 The schema of stiffness control. 56
3.7 Object's translation motion. 57
3.8 Object's rotational motion. 59
3.9 Object's rotational motion. 60
3.10 Object's translation and rotation respecting to the fixed support finger C_2. 61
3.11 An enhanced manipulation model. 62
3.12 Different perspectives of three fingers grasping. 63

4.1 In-hand manipulation experiment setup. 67
4.2 Shadow Dexterous Hand. 68
4.3 Shadow hand kinematics [12]. 68
4.4 An object with an AprilTag. 69
4.5 An in-hand object is tracked by the Apriltags system. 69
4.6 A box grasped by the shadow hand with three fingers. 70
4.7 Object frame. 72
4.8 Thumb push frame for pitch manipulation. 73
4.9 Index push frame for yaw manipulation. 73
4.10 Haptic feature extraction. 75
4.11 The raw data of 10 manipulations with $P_l = 4mm$ and $\mathbf{P}_d = (90, -15)$ 77
4.12 The raw data of 10 manipulations with $P_l = 8mm$ and $\mathbf{P}_d = (-15, 0)$. 78
4.13 Repeatability of haptic raw data in the different direction $\mathbf{P_d} = (90, 0)$ *and* $(90, -30)$ with support stiffness $\mathbf{K_s} = (0, 0)$. 79
4.14 Repeatability of haptic raw data in the different direction $\mathbf{P_d} = (-30, 0)$ *and* $(-30, -30)$ with support stiffness $\mathbf{K_s} = (0, 1)$. 79
4.15 Haptic and visual feature covariance diagram. 80
4.16 The relation between the finger's push distance and first haptic feature. 80
4.17 The relation between the finger's push distance and the object's rotated angle. 81
4.18 Haptic reward with different $\mathbf{pK}'$. 83
4.19 Visual reward. 84
4.20 Object's rotation axes. 84

4.21 Combined haptic and visual reward in pitch manipulations. 85
4.22 The snapshots of a pitch manipulation. 85
4.23 Haptic reward with different $\mathbf{pK}'$. 86
4.24 Visual reward. 87
4.25 Object's rotated axes. 87
4.26 The combined haptic and visual reward in yaw manipulations. . . . 88
4.27 The snapshots of the yaw manipulation. 88
4.28 Haptic and visual covariant diagram for the pitch manipulation. . . 89
4.29 Haptic and visual covariant diagram for the yaw manipulation. . . . 90
4.30 The manipulation results based on different push directions with $\mathbf{P}_d = (-30, \alpha)$. 91
4.31 The contact force on different fingers. 92
4.32 The stiffness performance of the ring finger. 92
4.33 Change in object' attitude. 93
4.34 Change of the object's position. 93
4.35 A disturbance during an object's grasp. 94
4.36 The haptic result of a robust grasp when a disturbance occurs. . . . 95
4.37 The visual result of a robust grasp when a disturbance occurs. (object's attitude) 96
4.38 The haptic result of a robust manipulation. 97
4.39 The visual result of a robust manipulation. (object's orientation) . . 98
4.40 The PR-2 robot with a shadow hand. 99
4.41 Snapshots of enhanced manipulation. 100
4.42 The rigid object with the same dimension as the foam box. 100
4.43 Various objects. 101
4.44 Snapshots of manipulating various objects. 102
4.45 The moving path of the fingers based on the virtual frame model. . 103
4.46 The snapshots of a manipulation based on virtual frame model. . . 103

5.1 A manipulation learning architecture. 108
5.2 A manipulation simulator. Its input is the push commands, and the output is the object's visual reward and haptic features. This simulator consists of four one-dimensional function approximators. . 110
5.3 A stochastic RL agent for manipulation. 111
5.4 The visual RBFN result in polar coordinates. In order to visualize this four-dimensional data, this result is sliced into 10 surfaces in sub-figures The red dots in Fig. 5.4(b), 5.4(c), 5.4(d),5.4(e), and 5.4(f) are training examples. 114
5.5 Transferring sub-figure 5.4(e) into Cartesian coordinate. 115
5.6 Visual RBFN result in a Cartesian coordinate. 115
5.7 Visual RBFN result shown with a two-dimensional plot. All experiment training examples are plotted with blue stars in this figure. The red solid lines refer to the visual RBFN result. 116
5.8 Haptic RBFN result shown with two-dimensional plot. 117
5.9 Haptic RBFN regression results in polar coordinate. 118

5.10 The third haptic RBFN result and the data distribution. 119
5.11 Actor-critic methods. 121
5.12 A three-layer ANN for Gaussian stochastic policy. 128
5.13 Visual rewards with Williams' REINFORCE algorithm. 129
5.14 Learning results with Williams' REINFORCE algorithm. 130
5.15 A learned path from REINFORCE algorithm. 130
5.16 Visual rewards in Peters' episodic natural actor-critic algorithm. . . 132
5.17 A learned path from Peters' episodic natural actor-critic algorithm. 132
5.18 Visual-haptic rewards with Williams' REINFORCE algorithm. . . . 133
5.19 Visual-haptic learning results with Williams' REINFORCE algorithm133
5.20 A learned path from Williams' REINFORCE algorithm with visual-haptic rewards. 134
5.21 Visual-haptic rewards with Peters' Episodic Natural Actor-Critic. . 134
5.22 A learned path from Peters' Episodic Natural Actor-Critic algorithm with visual-haptic rewards. 135
5.23 Comparison of the learning speed in different learning experiments. The 'REINFORCE' refers to the result from Williams' Episodic REINFORCE algorithm, and the 'NAC' refers to the result from Peters' episodic natural actor-critic algorithm. 135

B.1 Visual reward in the density push experiment. 145
B.2 Haptic results in the density push experiment. The first element of the haptic feature, pk_1, is shown in this figure. 146

C.1 Visual result approximated with an MLP neural network. 147

List of Tables

4.1 Three fingers cubic grasping configuration 71
4.2 Comparison of the enhanced and the virtual frame manipulations. . 104

5.1 Episodic REINFORCE algorithm parameters 129
5.2 Episodic Natural Actor-Critic parameters. 131

A.1 Important notations in this thesis 143

Chapter 1

Introduction

In modern society, robots are required to support humans from the elderly to the work force. With the progress of science and technology, human-assistance robots are no longer fiction. More and more commercially available robots are appearing around us, such as robotic toys, vacuum cleaners, robotic vehicles, surveillance robots, mental care robots, rescue robots, etc. [173], [37], [118].

With their ability to adapt to human tools, robots with anthropomorphic hands have great potential of being employed in domestic environments, where they could facilitate various domestic tasks like tidying up rooms, serving dinner, washing dishes, etc. For robots, grasping and in-hand manipulation are two fundamental skills required for these tasks. To date, a large amount of research has been carried out on grasping. By contrast, in-hand manipulation has been covered to a much lesser extent. Hence in-hand manipulation (also called regrasping or dexterous manipulation) is still a challenging research topic in the field of robotics. This skill not only enables robots to hold objects but also to move and position objects dexterously.

1.1 Motivation

The world over, robots performed superhuman feats of manipulation in factory environments. In recent years, robots in human environments have become the focus of attention. Unlike structured factory environments, human environments have a number of different challenges beyond the control of robots. As described in [95], among these challenges are the following:

- **"People are present".** Robots and the humans share the same environments. In some tasks, the robots have to adapt to and interact with humans who may have little knowledge of the robots.

- **"Environments are usually built-for-humans".** In human environments, objects are specially designed to adapt to humans. For example, door handles, cups, and computer mice are well designed according to the shape of human hands.

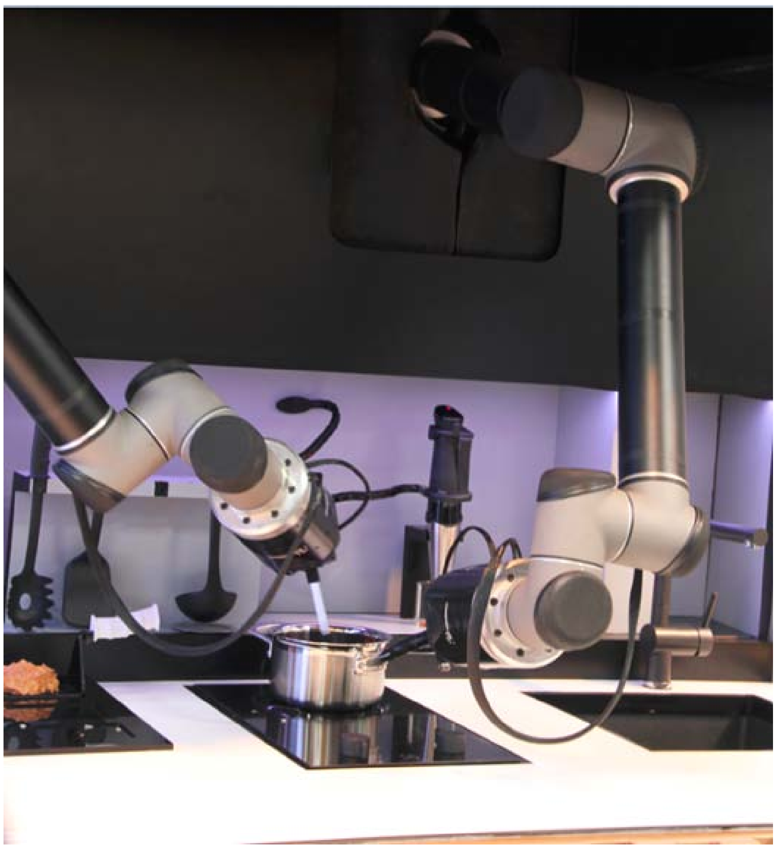


Figure 1.1: A robot chef cooks Michelin star food.

- **"Other autonomous actors are present".** In human environments, there may be pets and many other different robots, like cleaning robots, surveillant robots, etc.
- **"Dynamic variation".** Human environments change dynamically Sometimes, the change of the environment has nothing to do with the robots. For example, doors may open or close by humans,
- **"Sensory variation, noise, and clutter".** In human environment, robots' sensors are vulnerable to environmental interference, such as light changes, background sound, temperature changes, etc.

Although human environments constrain the application of robots, domestic robots are not just science fiction anymore due to the joint efforts made by engineers and scientists. On the HANNOVER MESSE 2015, the world's biggest industrial fair, several impressive collaborative robots were exhibited.

An excellent robot chef is shown in Fig. 1.1. As declared in [68], it will be on sale in 2017. A pair of robotic arms mounted with two dexterous shadow hands is mounted in a kitchen. This robot is able to cook world-class food with recipes downloaded from online stores. In the demonstrated scenario, this robot chef repeats the exact actions of Tim Anderson, the winner of 2011 BBC MasterChef,

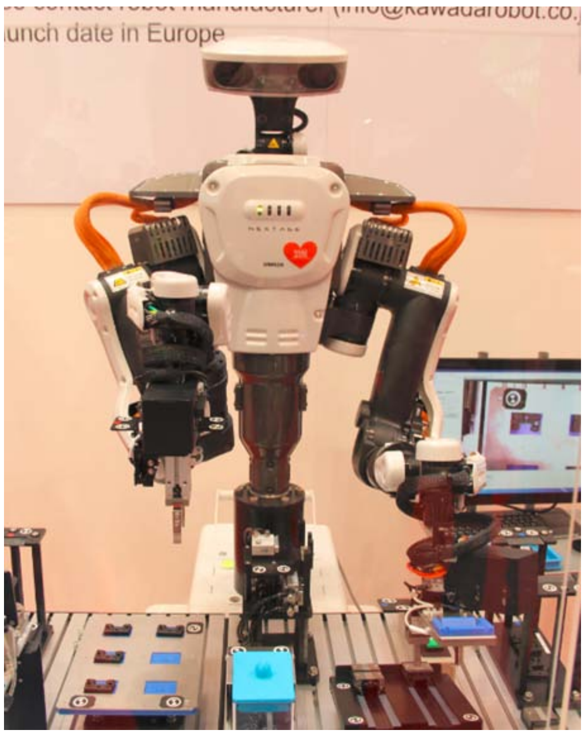


Figure 1.2: Nextage envisioned by Kwanda Industries.

such as scraping some butter off a knife, switching on the hob, measuring ingredients etc. This robot only takes 30 minutes to cook and serve a meal. In the future, with this robot, famous chefs can sell their recipes as an extension to their brands. Hence, from buying online recipes, robot's owners can enjoy the chef-cooked level meals every day.

Another example is Nextage: an autonomous dual-arm robot designed to work side by side with humans. With cameras in its head and hands, this robot is capable of recognizing 3D objects and making visual inspections. A demonstration was given in [94], in which it makes and serves cups of Espresso to its audience. In the future, this kind of robots will contribute to the flexible manufacture in high-mix and low-volume production. In other words, it will bring new opportunities by increasing the efficiency of both manufacturing plants and of domestic service. Other intelligent robots also have the potential to be applied in human environments, such as PR2 [49], ARMAR [19], PaPePo [132], HRP-2 [91], etc.

All these robots have mainly been developed to work as human assistants in human environments. Most of these assisting tasks require autonomous operations in human environments, which involve subtasks varying from localization, locomotion and grasping to in-hand manipulation. While robotic grasping has been well researched and implemented, robotic in-hand manipulation is still one of the weakest links in this task chain. As a result, today robotic in-hand manipulation is one of the key steps in robot assisting tasks. With skillful manipulating skills, robots can be remarkable assistants for us, doing labor-intensive or time-consuming jobs such as physically assembling something, taking care of an elderly person at home, and helping with household chores. In a word, in-hand manipulation is of critical importance to robotics.

In-hand manipulation covers many aspects. For humans, it is the ability to hold and move an object within their fingers. In in-hand manipulation tasks, the motion of the object can be divided into three major categories: translation, shift, and rotation [58]. **This thesis focuses on the in-hand task of an object's rotation.**

According to different working environments, the application requirements of robot hands in factories and human environments are different. In factories, robots are usually designed to grasp objects of a small number of similar types. By contrast, in human environments, robot hands are required to interact with a wide range of objects with different shapes; and the environments the robot hands interact with are specially designed for human hands. Thus, anthropomorphic robot hands are better equipped for tasks in human environments. This topic has been thoroughly researched. In the last decade, many anthropomorphic robot hands designed for flexible interaction tasks have become available off-the-shelf. **This research focuses on the in-hand manipulations carried out by anthropomorphic robot hands.** To facilitate writing, in the remainder of this thesis the term manipulation refers to general in-hand manipulation unless stated otherwise.

1.2 Problem Statement

Robotic manipulation is an interesting research topic. Good reviews can be found in [33] and [30]. A manipulation system consists of robotic fingers and an object. The fingers work cooperatively and attached to a base. The object is grasped or manipulated by changing the contact force applied by the fingers. Essentially, it is an interacting control problem where fingers apply a contact force to move an object within physical constraints. Three challenging problems of the robotic manipulation are a large number of joints, a complex interaction model, and limited perception capabilities.

1.2.1 Large Joint Number

In robotic manipulation, the fingers interact with the object directly. Generally, there are 5 fingers[1] on an anthropomorphic hand, and each of these fingers has at least 4 joints. The number of the joints is usually larger than 20 on an anthropomorphic hand. Therefore, the in-hand manipulation system is redundant. Moreover, this system is sensitive to contact force; if even one finger falls out of step with the others, the object could fall off the hand. Therefore, fingers have to cooperate synchronously and are capable of correcting occurred errors to ensure

[1]Most anthropomorphic robot hands have 5 fingers to emulate human hands as close as possible. However, some have only 4, like Shadow Hand-Lite [146].

successful manipulations. Besides, the size of an anthropomorphic hand is similar to that of a human hand, which means the whole hand control system has to be compressed into a very limited space. Some of the robot hands are controlled remotely via cables to save installation space; however, this mechanism results in low control accuracy and poor resolution of the finger joints. As a result, the performance of the hand system is not as good as that of general industrial robots. In short, this large number of joints and the limited joint space constrain the performance of the hand and challenge its control system. Hence a robust manipulation control method is required.

1.2.2 Complex Interaction Models

Besides the robotic control, planning the fingers' actions is also challenging. Most grasping and manipulation is planned based on synthesis methods, where a complete knowledge of robots, target objects, and physical laws is modeled explicitly [96], [64]. However, these synthesis methods are labor-intensive and sensitive to parameters. In addition, sometimes planning under contact conditions is impractical due to the computational complexity and the lack of precise and robust dynamic models. Due to the complexity of the models, these traditional manipulations are error-prone in the control of the interaction. In addition to precise models, adaptive force control methods offer another way to robotic manipulation. In robotics, contact force plays an important role in compensating for these errors from the 'poor' models. However, the contact force is sensitive to the contact state, such as the fingers' relative position, contact area, the object's attributes, etc. For force-based in-hand manipulation, all grasping fingers are equipped with force control algorithms. And the controller has to assign the contact force to the different fingers. Sometimes, this makes the system too complex to be implemented. Accordingly, a better manipulation planning approach is required.

1.2.3 Limited Perception Capabilities

Finally, robotic manipulation requires perception capabilities. Both finger control and action planning are based on sufficient information, including information on the object, the state of the fingers, and the contact state between them. Vision and tactile sensing are the two most used sensing channels in this area. In manipulation, visual sensing is often used to locate and track the in-hand object and tactile sensing is used to guide the fingers to adjust the contact state which is sensitive to the models' parameters and external disturbances. However, in manipulation tasks, the object is often occluded by the robot's fingers and the application of the tactile sensors is often limited by their size, density, resolution, etc. Thus an efficient visual-haptic[2] sensing method is required in robotic manipulation tasks.

[2]In this thesis, the term 'haptic' is used to denote tactile sensing. The 'haptic' refers more to the tactile information from the fingers' actions.

Figure 1.3: A water tap.

1.3 Objective and Proposed Approach

In this section, the objective of this thesis is presented; moreover, our manipulation approaches are proposed.

1.3.1 Objective

As discussed above, the principal objective of this thesis is to carry out robust in-hand manipulation efficiently using an anthropomorphic robot hand. In order to achieve this, the major objectives of this thesis are the following:

- Conducting various robust in-hand manipulations with an anthropomorphic robotic hand.
- Exploring the in-hand system with a given grasping configuration. Without possessing any information about the object in advance, the robot should have the ability to explore the in-hand object automatically to acquire the necessary manipulation knowledge. Then, the robot can conduct acceptable in-hand manipulations efficiently based on this knowledge.
- Seeking the relationship between the haptic and visual feedback by analyzing the results of the manipulation experiments conducted with our push and support manipulation method. Efficient methods are required to represent the manipulation state with the visual and haptic signals.
- Improving the robotic in-hand manipulation skill. The robot should have the ability to improve its manipulation skill through interacting with its environments. In other words, the robot should have the ability to learning its behaviors with the method of trial and error.

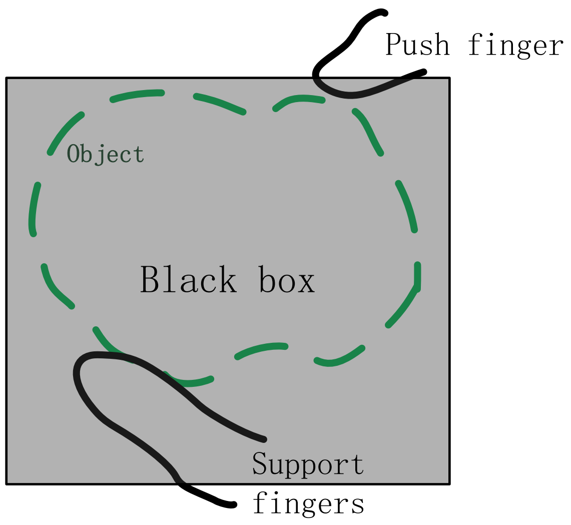


Figure 1.4: Black box. This diagram shows the basic research concept of in-hand manipulations. In this black box, an unknown object is located on fingers other than the push finger. Therefore the manipulation is transferred into rolling the object on the surface of other fingers.

1.3.2 Concept and Approaches

As presented in [29], two main abilities of a human hand are prehension and apprehension. The prehension is the ability to grasp and hold objects of different sizes and shapes, and the apprehension is the ability to understand through active touch. In this sense, the human hand interacts with the world not only by performing actions but also by perceiving external information on the world. One example to illustrate this is turning off a tap, as shown in Fig. 1.3. When a human hand manipulates a tap, the fingers perform push actions to rotate it. At the same time, the fingers' touch is analyzed to estimate the state of the tap. Specifically, the increase of rotational resistance felt by the fingers means the tap is closing; on the other hand, decreasing resistance denotes that the tap is opening. Hence, the state of the tap can be perceived not only from visual observation of the water flowing but also from the haptic feedback (push resistance during the manipulation). **Different from other research, this thesis strengthens the apprehension skill of the robot hands. Moreover, the visual and haptic information is used to estimate the state of the in-hand system.** To demonstrate the apprehension ability of the robotic hands, two basic concepts are proposed: the black box model and haptic exploration.

Manipulation with Black Box

Different from most current manipulation researchers who model the objects and the robots separately and combine the two with physical laws, **this research treats the target object and the support fingers as one black box system where control commands of a push finger are sent as input and the**

observed visual-haptic feedback is output. The fingers and the object (called in-hand system) are all contained in this black box. In the box, only one finger receives the input commands and moves to drive the manipulation. Without any knowledge of the in-hand system, the finger has to push into this black box to 'feel' the interaction state with the object. To explain the performance of the push actions in the black box, simple manipulation models are introduced in section 3.

As shown in Fig. 1.4, an object is grasped by two fingers. The finger at the top is called push finger, the one at the bottom is called support finger. Although only one support finger is plotted in Fig. 1.4, there can be more depending on the number of fingers involved in a grasp. The special surface consisting of support fingers have a decisive influence on the push properties of the black box. It is characterized by the relative position of the contact points and the elasticities of each support finger. By changing the elasticities of the support fingers via different control algorithms, the attributes of the special surface can be modified. More details will be discussed in section 3. Thus, the manipulation is transferred into rolling an object onto this special surface. To carry out these rolling actions, push commands are given to the push finger, which has no knowledge of the object and the special surface. With the push commands, the push finger pushes into this black box and the haptic information is collected; in addition, the performance of the push actions is evaluated with the visual rewards from the object's tracking system. Since the manipulation is mainly carried out through those push actions with the help of the support fingers, it is called push and support manipulation in this thesis.

Haptic Exploration

Before manipulating, the robot does not know anything about the black box except that the object has been grasped firmly. Without sufficient prior knowledge, the robot cannot conduct proper push actions to complete a successful manipulation. Therefore, a special perception strategy is required to help the robot acquire sufficient knowledge about the black box system. As a result, we developed a perception strategy named haptic exploration.

Haptic exploration is a process where the robot pushes the in-hand object slightly in different directions and estimates the interaction state by means of haptic feedback. The term haptic exploration is very similar to 'touch exploration' and 'active touch' which is widely used in robot recognition fields (like material and surface identification). However, most touch exploration research focuses on static object information (such as geometrical surface, texture, contact force, etc.) [145], [15], [98], [183], few researchers pay attention to dynamic information in the interaction process. Therefore, the term 'haptic exploration' is introduced in this research to explore the black box through dynamic interaction.

The haptic exploration process is shown in Fig. 1.5. As an interaction process, there are two important aspects worth mentioning in the haptic exploration: action and perception. The action refers to the push actions executed by the push finger. The term 'push' is used to denote these actions executed by the push finger which

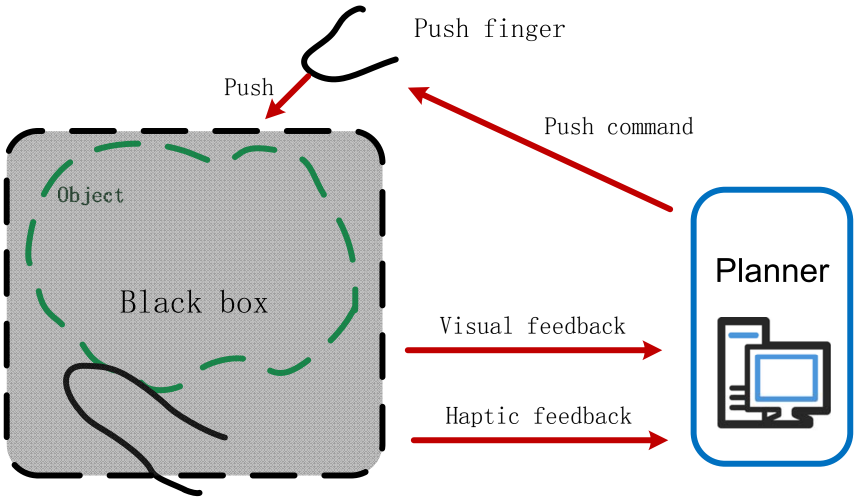


Figure 1.5: Haptic exploration process.

is actively controlled and plays a leading role in manipulations. Different from pushing a box on the ground, the purpose of these push actions is to collect feedback rather than move the object. **Consequently, if not stated otherwise, all the push actions presented in this thesis are limited to a very small range so that the push actions are comparable, since the object can still return to its original state after the push finger moves back to its initial position.** In addition to the push finger, the support fingers also play an important role in the haptic exploration. Different controllers are applied to the support fingers to acquire proper 'elasticity'. On the one hand, this elasticity protects the robot from damages while the finger pushes 'randomly'. On the other hand, it enables the system to go back to its initial state when the push action is completed. This makes sure the robot carries out different commands from the same initial state.

The perception refers to different feedback from multiple channels, specifically visual and haptic in this thesis. The visual feedback is provided by a visual tracking system which tracks the reaction of the object to the push actions, such as changes in the object's orientation, position, etc. The haptic feedback mainly refers to the contact force information provided by the tactile sensors mounted on fingertips. It is worth mentioning that though both visual and haptic feedback is considered, this process is called haptic exploration. This is because the visual feedback is mainly used to evaluate the manipulation (push actions) performance, whereas the haptic feedback is used to evaluate the interaction state. Furthermore, with the collected knowledge, the robot can select the 'right' push directions in its manipulation tasks. In this haptic exploration process, the robot pushes into this black box based on different commands and perceives the haptic feedback to evaluate the interaction state in this black box. These repeated actions and perception help the robot to collect sufficient knowledge on the black box. With this knowledge, the robot is able to conduct successful manipulations.

With the concept of trial and error, haptic exploration can be a general method to help the robot collect knowledge for its manipulation tasks. **More importantly, in this way, calculations based on complex physical laws are not required; and the interaction state of the robot system can be directly described with the push feedback.**

Learning to Improve Manipulation Skills

With the help of the black box and the haptic exploration, a general robotic manipulation can be acquired. The manipulation skills can then be improved with other optimization methods. In this thesis, Reinforcement Learning (RL) is applied to improve the robotic manipulation skills. The goal of the learning process is to create an optimized RL agent which can generate more appropriate finger paths in manipulation tasks. In this research, a stochastic policy is designed to map the current system state to finger actions. Besides, a manipulation simulator has been built based on real example data collected from real manipulation experiments. The RL agent generates actions for the manipulation simulator, and the simulator generates visual and haptic feedback. After calculating the feedback, action rewards are obtained. According to these rewards, the RL agent optimizes its parameters to improve its manipulation skills step by step.

1.3.3 Push Manipulation Architecture

In Fig. 1.6, a push manipulation diagram is illustrated. On the top of this system, there is a planner. It receives the visual-haptic feedback, configures the support fingers, and generates the push commands to the push finger. Once the object is grasped, the configuration of the support fingers is done according to manipulation tasks by this planner. Below the planner, there is an anthropomorphic hand which consists of two groups of fingers: the push finger and the support finger. The push finger receives and executes the push commands to push an unknown object. This finger is controlled via a position controller, which generates the motor commands to the joints without considering any external impact. It is an off-line controller. And the support fingers are configured respectively by choosing proper controllers before manipulations. There are two candidate controllers for the support fingers: the position controller and the stiffness controller. The position controller is the same as in the push finger, and the stiffness controller generates motor commands by considering the tactile feedback on the fingertips. With the stiffness controller, the finger behaves like a spring which is linearly compliant to the external force. Accordingly, this finger is called spring support finger in this thesis. With these fingers, the unknown object is grasped, shown below the anthropomorphic hand in this diagram. With the push commands and the configured controllers, the fingers interact with the object dynamically. In the manipulation, the object's movement is recorded by a visual tracking system, shown in the bottom of Fig. 1.6. This tracking system provides the object's pose to the planner to help estimate the performance of the current push command.

It is worth noting there are three kinds of feedback in this system: visual

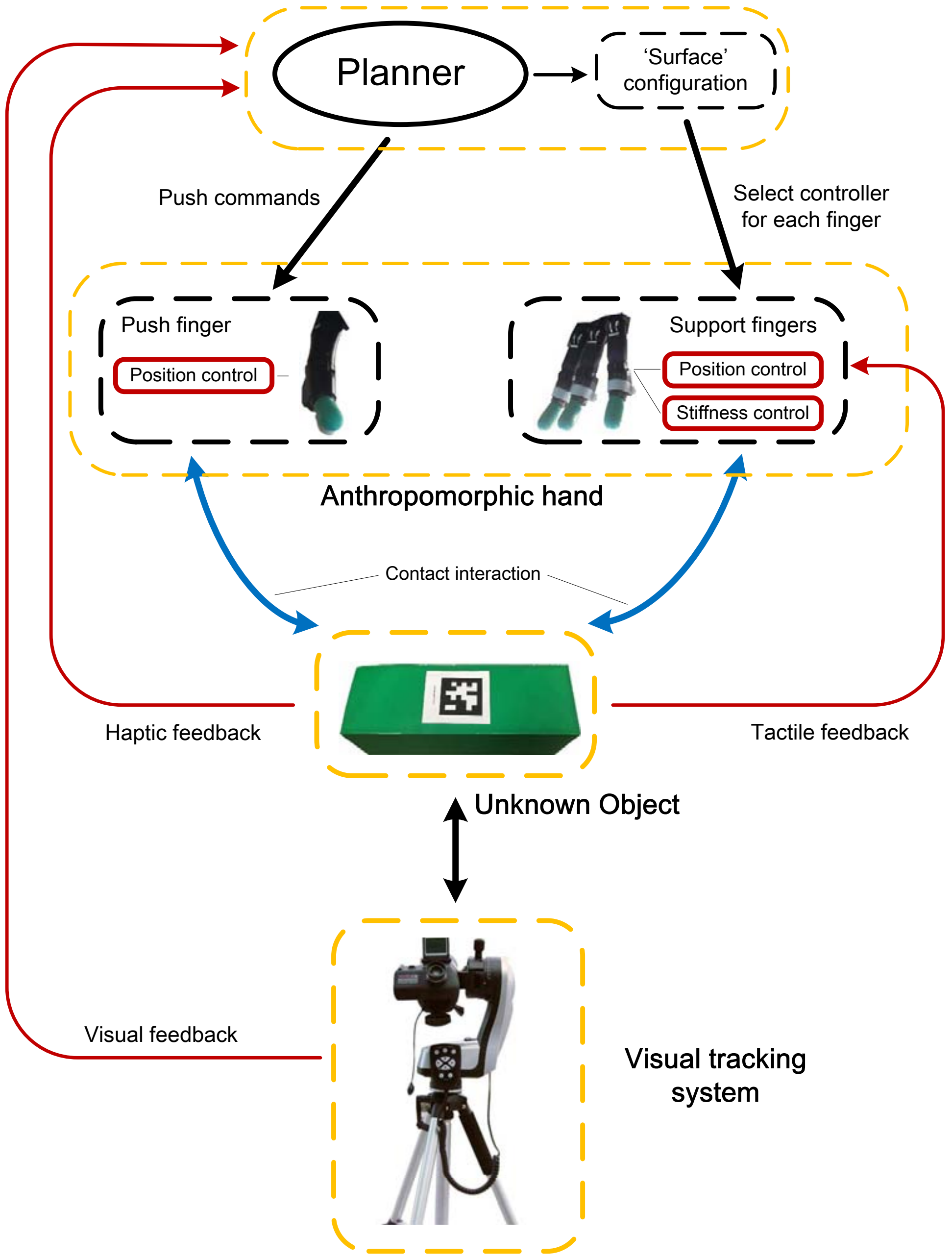


Figure 1.6: Push manipulation architecture. This diagram illustrates the push and support manipulation. The yellow dash rectangles denote the components in this system, and the red arrows refer to the sensing signals. A push manipulation process is described as follows. At the beginning, a planner configures the support fingers by selecting proper controllers on them. After that, push commands are given to the push finger. According to these commands, the finger pushes the unknown object to roll onto the support fingers. In this process, the object is tracked and recorded by a visual tracking system and the contact state is recorded with the tactile sensors.

feedback, haptic feedback, and tactile feedback. The visual feedback is provided by the visual tracking system and received by the planner. It contains the changes in the object's location and pose. In this planner, the visual feedback is processed to extract the visual features to estimate the performance of the push manipulation. The haptic feedback refers to the tactile force on the push fingertip when the finger pushes forward. It is a dynamic signal recorded during the whole process of the push action. This feedback is collected by the tactile sensor on the push finger and sent to the planner. In this planner, the haptic feedback is extracted to haptic features to evaluate the push performance. This tactile feedback denotes the instant contact force applied to the support fingers. It indicates the external force applied to the stiffness controllers.[3]

1.4 Contributions

The contributions of this thesis are the following:

- A novel robotic manipulation concept is proposed. In this thesis, grasping fingers and object are considered as a black box. In this black box, only one finger (named push finger) receives the manipulation commands and conducts active actions. Thus the manipulation is simplified to give the push finger commands and let it push into this black box.

- A process named haptic exploration is introduced to assist the robot in acquiring interaction knowledge of the in-hand system. Using the concept of the black box, the push finger pushes into the box slightly and repeatedly, and both visual and haptic feedback is collected in this process. The experimental result reveals that the haptic feedback is sufficient to indicate the right 'push' direction for manipulation tasks. In other words, the interaction knowledge of the system can be represented by the haptic feedback collected in the pushing process. Furthermore, our haptic method shows a novel way to extract the haptic features from the haptic signals in the pushing process.

- Pushing based manipulation models are developed to analyze our push and support manipulation method. At the beginning, an elastic surface model is put forward according to our black box concept. The surface consists of support fingers. The elastic properties of the surface are of critical importance in manipulations. In this thesis, to modify the elasticity of the surface two support fingers are proposed: fixed support finger and spring support finger. The fixed support finger acts as a pivot around which the object rotates. Its position is fixed with a position controller. The spring support finger performs as an elastic spring. It presses the object against the other fingers and helps to rotate the object with proper contact force. A single support model

[3]In this thesis, both the haptic and the tactile feedback refers to the contact force on the fingertips. However, the haptic feedback is of more concern to the force in a period of time (also called dynamic signal); whereas the tactile feedback refers to the contact in an instant of time.

and a hybrid support model are proposed to illustrate push manipulations based on combining different support fingers. With these two models, any rotational manipulation can be achieved by controlling the push finger.

- It is proved that only one spring support finger is sufficient to compensate for all the errors generated in 2D manipulation processes. More importantly, this conclusion guarantees the stability of the system in our push manipulation. This conclusion can be also extended to robotic grasping. Moreover, this spring support finger improves the robustness of this system against external disturbances.

- It is illustrated by the experiments that haptic sensing can be the domain sensing channel to indicate the interaction state in robotic manipulations. In this thesis, several manipulation experiments have been conducted, and both visual and haptic signals have been collected. Furthermore, the relation between visual and haptic feedback has been thoroughly researched. Compared to visual and haptic feedback, the haptic can not only replace the visual sensing but also provide more useful interaction information for our push manipulations.

- Reinforcement learning algorithms are applied to improve our in-hand manipulation skills. To our best knowledge, this is the first time that reinforcement learning algorithms are applied to learn the robotic in-hand manipulation through the manipulating experience of real robots. Our proposed learning frame is very close to real robotic learning, and the result shows the potential of reinforcement learning methods in robotic in-hand manipulations.

1.5 Outline

In this last section, the outline of the remaining chapters is presented. This thesis is organized as follows:

- In Chapter 2, "**State of The Art**", the state of the art of the robotic grasping and manipulation are presented. First, both human grasping and manipulation skills are introduced briefly. Then several well-known anthropomorphic robot hands are introduced. And then a presentation and discussion of the currently used **robotic grasping and manipulation methods** is introduced. Finally, other multimodal perceptions and intelligent planning methods used in manipulations are discussed.

- In Chapter 3, "**Manipulation Models**", our manipulation method is given, and related analysis models are built. First, the basic knowledge of contact models and grasping matrix is introduced. Then a **push and support model** is proposed to generalize our main idea to the manipulation. Then two models extracted from the push and support manipulation model are discussed specifically. They are the single support model and the hybrid support finger model. After his, an enhanced manipulation model is introduced to perform final manipulations. At last, a short discussion is given to illustrate the feasibility of 3D manipulations with our proposed manipulation methods.

- In Chapter 4, "**Manipulation Experiments**", manipulation experiments have been performed to verify the feasibility of the models put forward in Chapter 3. First the experiments' setup is introduced. Then the push procedure in each manipulation experiments is described. Then different robotic manipulation experiments are carried out, including repeatability experiments, a multi-push distance manipulation experiment, a multi-directional manipulation experiment, a hybrid support manipulation experiment, and a robustness experiment. At last enhanced manipulations are carried out to illustrate the final manipulation performance of our proposed method; furthermore, comparative manipulation results are given between our proposed method and the traditional virtual frame method.

- In Chapter 5, "**Learn to Improve Manipulation Skills**", learning methods are implemented to improve the robotic manipulation skills. First, a short introduction to machine learning is given. Then, a manipulation simulator is constructed to simulate the visual and haptic rewards in manipulations. Then stochastic policy reinforcement learning algorithms are applied to learn and optimize finger paths via interaction with the simulator. Finally, in learning experiments, learning with unimodal or multimodal rewards is compared. The result reveals that the use of multimodal rewards (visual-haptic) speeds up the learning process dramatically compared to using unimodal rewards (visual only).

- Finally, in Chapter 6, "**Conclusions**", the main contributions of this research are reviewed and our future work is presented.

Chapter 2

State of The Art

Human hands can not only grasp objects but also can move or position objects with their fingers. This skill of fingers' moving and position objects is usually known as "in-hand manipulation", "dexterous manipulation", or "re-grasping". In this chapter, the state of the art is presented from human to robotic manipulation problems.

In-hand manipulation is an active research topic. Good reviews can also be found in [33] and [129]. Although plenty of works have been done in decades, it is still a challenging topic in robotics [153], [31], [129]. This research involves multiple areas, from human hands to robotic grasping and manipulation skills, including capabilities of human hands, the design of robot hands, planning in manipulations, multimodal sensing, the optimization of robot skills, etc.

In order to present an overview of related research. This chapter is organized as follows:

- In section 2.1, human grasping and manipulation mechanisms have been introduced. Firstly human grasping and manipulation configurations are classified; then the human neural sensing mechanism is introduced briefly; finally, an example of human manipulation is presented.

- In section 2.2, well-known robotic hands are introduced, and their sensing capabilities are discussed.

- In section 2.3, state of the art models used in manipulations have been presented. There are three main streams in modeling robotic manipulation: synthetic explicit interaction models, virtual object frame models, and contact control models.

- In section 2.4, synergy-based manipulations have been introduced. Inspired by coupled behaviors of human fingers, synergies are brought into the control of anthropomorphic robot hands to reduce the number of the actively controlled joints.

- In section 2.5, different modalities used in manipulations have been reviewed. The two mostly used modalities in manipulations are visual and tactile sensing. Visual sensing is often used for the object's recognition and tracking; tactile sensing is used to estimate the stability of the system and to adjust the contact state of the fingers.

- In section 2.6, some other intelligent methods used in robotic manipulations have been reviewed. The intelligent methods, especially machine learning algorithms, offer another way to achieve and optimize the robotic manipulations.

2.1 Human Grasping and Manipulation

The anthropomorphic robot hand is derived from mimicking the kinematics of human hands and sensing capabilities. It is necessary to give a short introduction to human hands and their working mechanisms for better understanding. In this section, human grasping and manipulation are firstly classified according to human daily life tasks; then the neural mechanisms involved in manipulations are briefly introduced; at last, an example of human manipulation mechanism is offered.

2.1.1 Grasping Classification

As a fundamental step of manipulations, grasping plays a vital role in manipulation tasks. Humans usually unconsciously perform different grasps in different grasping tasks. Therefore, classifying the human grasps is an important step in the research of grasping and manipulations. Many grasp categorization methods have been proposed.

An early research on characterizing human grasps can be found in [125], where Napier gave one of the simplest ways to classify human grasps into power grasp and precision grasp. In a power grasp, both the fingers and the hand palm exert pressure on the object. In this grasp, the object is wrapped by the hand to gain as much contact area as possible. Typically, it is used in tasks of grasping large objects in which no further manipulation is required. Differently, in a precise grasp, only fingers make the contact and exert pressure on the object. Usually, fingertips are used for these contacts. In this kind of grasps, the fingers are flexible in moving the object, so that it is possible for fingers to manipulate the object dexterously. Therefore, precise grasps are dexterous and have the advantage in manipulations [163]. Moreover, Schlesinger proposed another categorization which divides the objects' shape into six geometry types: cylindrical, hook, tip, spherical, palmar, and lateral [117]. Bullock and Dollar also proposed a taxonomy for human and robotic manipulations based on whether there are slips at the contact or not [39].

Besides, a more practical grasp taxonomy was given by Cutkosky, on the basis of the machinists' grasps when they are working with hand tools and metal parts in single-handed operations [50]. The Cutkosky's grasp taxonomy is shown in

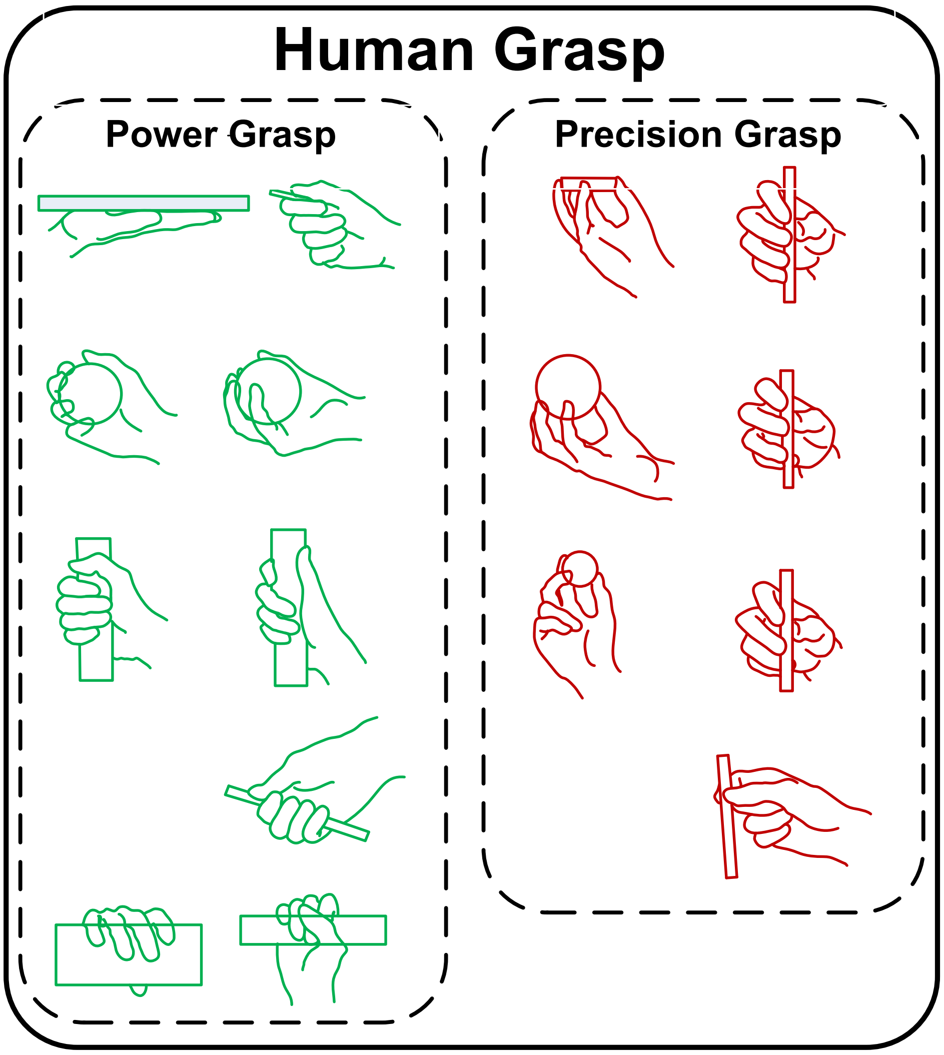


Figure 2.1: Cutkosy's grasp taxonomy [50].

Fig. 2.1, essentially, this taxonomy is an integration of the work of Napier and Schlesinger. In Curkosy's taxonomy, human grasps are firstly divided into power grasps and precision grasps; and then further divisions are made based on the shape of the object. Finally, 16 grasp types are mentioned in Curkosy's taxonomy, of which seven are precision grasps suitable for further manipulations.

Furthermore, based on Curkosy's taxonomy, Zheng et al. investigated one housemaid and machinist respectively for the using frequency of different grasp types in both mechanical shop tasks and daily household [186]. Their results denote that about only six kinds of grasps are used to undertake 80% household tasks. Their results show that precision grasp time for housemaid and machinist is 19%, 34% respectively. Furthermore, the prismatic grasp takes 42% of precision grasps for the housemaid but 38% for the machinist. **Therefore, in this thesis, we focus on precision grasps, especially the prismatic precision grasps, which is applied to the first grasping step in our manipulations.**

2.1.2 Manipulation Classification

In-hand manipulation is also called dexterous manipulation. The terminology 'dexterous' in manipulations is used to represent the capability of changing the object's pose from one configuration to another in the workspace of the object and hand. According to different assumptions about the contact model, there are two processes to achieve this dexterity: re-grasping/finger gaiting and sliding/rolling [31].

Re-grasping/finger gaiting is a practical method to achieve dexterous manipulations for simple hands (like grippers with two fingers or hands with a few DOFs). In this method, fingers may sometimes detach from the object during the manipulation [156]. After the detachment, some new contacts will be achieved in the next grasping steps. This method involves the sequent steps of grasping and releasing the object. However, re-grasping/finger gaiting has drawbacks. During this manipulation process, an additional support plane is required for holding the object in the releasing steps. Besides, it takes more time for the sequent steps of grasping and releasing compared to other methods.

In sliding/rolling, the contact between fingers and object is allowed to slide during manipulations [47]. In this method, contact sliding and rolling models are often used to control the contact change. That is quite a big challenge in robotics and elastic analysis, because the presence of rolling contacts causes many changes in the robotic system.

2.1.3 Tactile perception in Human Manipulations

According to Johanssons research [85], different types of tactile signals have been fully used by humans in manipulation tasks, and the information provided by the visual and proprioception afferents are less essential. Therefore, in this research, we mainly concern the tactile information in robotic manipulations.

Neural Mechanisms Involved in Human Manipulations

In humans' manipulations, the tactile information is mainly provided by four types of tactile afferents on their hands: FA-I (fastly-adapting type I), SA-I (slowly-adapting type I), FA-II (fastly-adapting type II) and SA-II (slowly-adapting type II). More details can be found in [170].

FA-I is sensitive to the dynamic skin deformations with high frequency ($5 - 50$ Hz). SA-I is sensitive to the skin deformations with lower frequency, hence it has responses to the sustained deformations. Both FA-I and SA-I afferents are mainly distributed in the skin of fingertips. Moreover, compared with SA-I, more FA-I afferents are found in the skin of fingertips. This reflects that the skin deformations with high frequency are more important than the skin deformations with lower frequency in the fingertips. The events detected by FA-I are skin forming/breaking contact with objects and scanning across a textured surface, etc. [86].

FA-II afferents can be easily excited by transient mechanical events, and they are to detect hand's holding and breaking of the contact with an object. Moreover, some FA-II-like afferents are in some fibrous tissues (such as muscle fascias and joint capsules and ligaments) in which they play the role of proprioceptors [52]. SA-II afferents can be excited to lateral stretching of the skin, and it is to detect the tangential shear strain in the skin during object manipulations [99], [116]. Different from FA-I and SA-I, FA-II and SA-II afferents terminate deeper in the hand's skin with a lower and roughly uniform density.

Human Grasp and Manipulation Example

Neural scientists have thoroughly researched the neural mechanisms of human grasps and manipulations. In [85], Johansson et al. researched human behaviors of object picking up. In his research, a human picking up behavior is divided into seven distinct phases: reach, load, lift, hold, replace, unload, and release. In the 'reach' phase, humans close their hands until their fingers reaching the object. The change between 'reach' and 'load' phase is trigged by the signals provided by the FA-I (Meissner) and FA-II (Pacinian) afferents in fingers. In the 'load' phase, humans close their hands to increase the grasp force to a target value. This target force value is estimated through the prior knowledge about the object and tactile information collected during the interaction. Hence, this process is mainly based on the information provided by the SA-I (Merkel) afferents. This 'load' phase ends after a stable hand posture has been achieved. After the object is grasped, the following phases are 'left', 'hold', and 'replace', where humans lift up the object with their arms, hold it in the air, and manipulate it to a new position. Certainly, in these phases, the human hands conduct corrective actions to adjust the grasp force for a stable grasp. Hence, in the 'lift' and 'hold' phases, detecting of slips is of critical importance for the success of the task. As researched by Srinivasan et al., the contact information provided by the FA-I and FA-II afferents is used to detect both fingertip slip and new object contact [155]. In the 'place' phase, the object is put back to the table. During this phase, contacts between the object and the table must be precisely detected. In the 'release' phase, the object is set down properly. Before the object is fully released, the contact information of the grasp is also provided by the SA-I afferents.

In summary, according to Johansson's experiment, it can be concluded that with tactile sensing capabilities and hand's corrective reactions human hands can adeptly hold and release a very wide range of objects without crushing or dropping them. Indeed, humans typically apply a grasp force which is only 10% to 40% more than the minimum amount needed to avoid slip, thereby achieving the dual goals of safety and efficiency [147], [85]. Inspired by human grasp mechanism, several research groups have developed autonomous robotic grasping and manipulation systems, where tactile sensing was adopted to guide the corrective actions of the robotic hand [147], [82], [56].

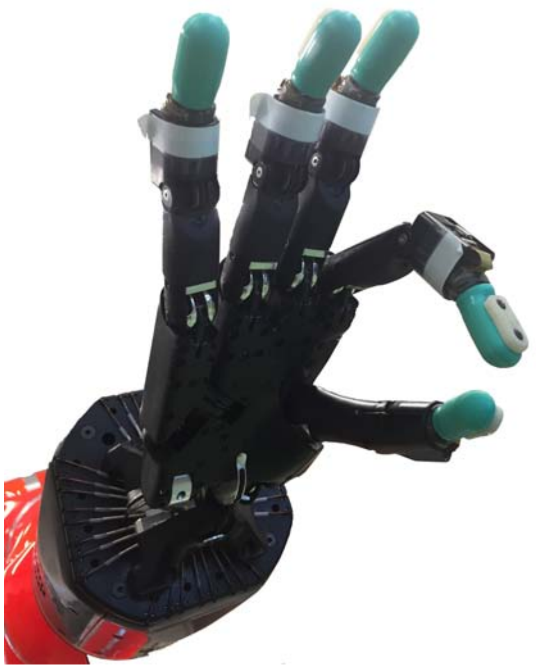


Figure 2.2: An anthropomorphic robot hand: Shadow Hand.

2.2 Anthropomorphic Robot Hands

For numerous and complicated manipulation tasks, powerful and flexible robotic hands are required. Operability is one of the most important keys to the success of intelligent grasp and manipulation tasks. The anthropomorphic hands are designed by mimicking human hands partly or totally [31]. They are similar to human hands in aspects of kinematics, contact attributes, etc. [29]. With these hand, it is easier for us to map human nature manipulation actions to robotic hands directly. In [29], Biagiotti estimated the dexterity of anthropomorphic hands from different elements, such as morphological features, task planning strategies, sensory equipment, control algorithms, etc. Many famous robot hands have been evaluated according to the proposed aspects. In this research, we define the anthropomorphic hands as hands whose kinematic and size are similar to the human hands (with 4 or 5 fingers).

2.2.1 Anthropomorphic Robot Hand

As introduced in [31], robot hands are systems with two or more fingers on a palm. The general robot hand can be divided into three types according to the number of fingers. First is the single-actuator parallel jaw, such as Kuka YouBot [7], [34], Willow Garage PR2 [8], [49], etc. Second is the simplified multi-finger hand, such as Barret Hand [3], [167], Robotiq Adaptive Gripper [1], Kinova KG-3

[6], Schunk Hand [10], iHY Hand [127], etc. Third is the anthropomorphic robot hand which is designed by truly mimicking a human hand, such as Utah/M.I.T. Hand [81], Shadow Hand [5], [148], Robonaut Hand [9], [115], UB Hand [114], [119], Karlsruhe Hand, [62], [61], DLR/HIT Hand [113], KITECH Hand [21] and Schunk SVH hand [11], etc. One of the most famous anthropomorphic robot hands is the Shadow Hand, as shown in Fig. 2.2.

2.2.2 Sensing Capabilities of Robotic Hand

Nowadays, sensor systems are usually equipped in robotic hands to capture multimodal information (for example, vision, sound, tactility, location) in tasks. To some extent, its sensing ability is a simulation of the human cognitive process.

Human grasp forces highly depend on the coefficient of the friction [41]. For viscoelastic materials like human skin, the main mechanisms of the friction are adhesion and hysteresis. Adhesion refers to the tendency of adhering at the asperities of the surfaces, and hysteresis denotes a material's delayed response to the forces applied on it. Therefore, increasing the contact area helps to increase the coefficient of friction during grasping [166]. That is why almost all robotic fingertips are made of or covered by soft materials like silicone rubber.

2.3 Physical Model-based Manipulation

In robotic manipulation systems, it is important to model the details of robot-object interaction properties which requires the specific knowledge of the system from precise object-hand interaction models [31]. Lots of manipulation models have been proposed for robotic manipulations.

In this section, manipulation models are introduced, according to how the interaction is modeled. First, explicit interaction is introduced, where the finger and the object are modeled respectively and their contact interaction constraints are considered. Then a virtual object frame is introduced, where the object is modeled on a geometry virtual frame but contact constraints are ignored. At last, special contact models are represented.

2.3.1 Explicit Interaction Model

Modeling the interaction explicitly is one of the most intuitive methods in robotic manipulations. In this method, object-hand interaction is analyzed explicitly. Nowadays, many impressive robotic manipulations have been achieved through building explicit models. One of the most impressive results is high-speed manipulations [64], [152], [112], where the manipulations were completed through controlling object's motions dynamically. This manipulation system tracked the state of the object accurately and calculated the hand's actions via an explicit model, as shown in Fig. 2.3. Besides, a dynamic single finger manipulation model is presented to roll a rigid dynamic circular object via regulating manipulation

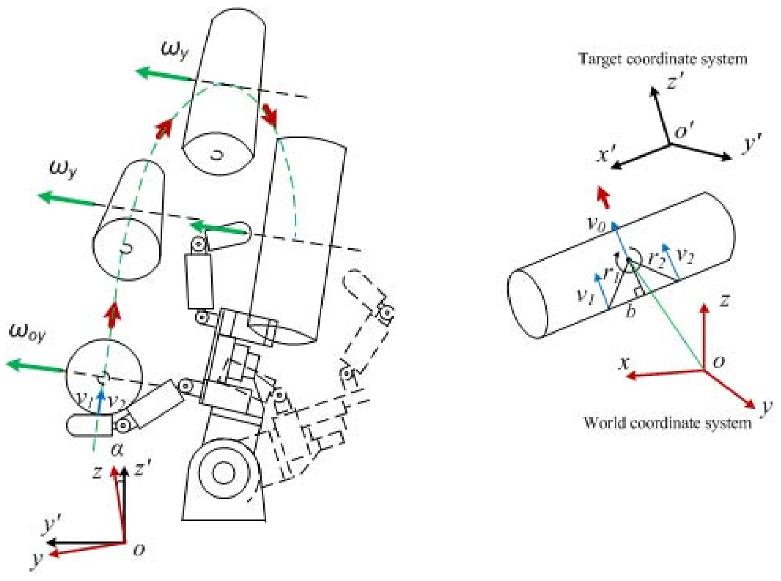


Figure 2.3: An explicit model for high-speed manipulation [64].

force and the object's position [67]. Ozawa et al. proposed a method for parallel surface object control without sensing the object and the tactile force [133].

The explicit models are based on the assumptions that the robot and its environments can be explicitly modeled. However, in most real manipulation scenarios, there are too many uncertainties in robotics. Generally, these uncertainties stem from several ways. One is because robotic hands are very complex and their control accuracy is not as high as traditional industrial robots. Most anthropomorphic hands have more than 10 DOFs. The grasping and manipulation systems are often redundant. This causes more uncertainties into this system. Moreover, modeling the physical contact is still a challenge, since both grasping and manipulations are sensitive to the contact state. Besides, in robotic manipulations, the interaction state is sometimes not easy to perceive. In object visual tracking, images are easy to be occluded by fingers so that the tactile state is hard to perceive in high density. In a word, explicit models are labor intensive and sensitive to model parameters. Sometimes planning under contact conditions is impractical due to the computational complexity and the lack of precise robust dynamic models.

Without precise models, adaptive force control methods show another way to robotic manipulations. In robotics, contact force plays an important role in compensating these errors in the 'poor' models. In [111], Li and Kao modeled dexterous manipulation with soft contacts and a stiffness controller. Biagiotti et al. designed a Cartesian impedance controller for in-hand manipulation [28]. Generally, the contact force is sensitive to the contact state, such as the fingers' relative position, contact area, the object's attributes, etc. In force based manipulations, all grasping fingers are equipped with force control algorithms, which assign proper contact force to fingers. However, in an anthropomorphic grasp, this method sometimes makes the system too complex to be implemented.

2.3.2 Virtual Object Models

To avoid building these precise explicit models, generally, two approximate methods were proposed: virtual frame and virtual linkage.

Virtual Frame

In [161], instead of using any information of the real object's position and orientation, the object is defined by a virtual frame. In this method, only internal sensors are adopted such as joints' angle, angular velocity, and torque. Obviously, a virtual object frame is defined to represent the position of the object. It is a centroid of a triangle consisting of each center of the fingertips. The concept of virtual object frame is suitable for senseless grasps and manipulations which aim to manipulate an object not precisely.

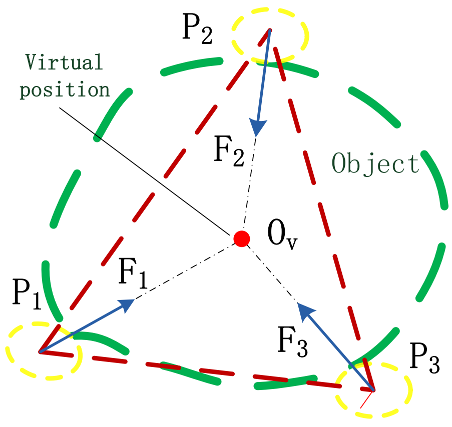


Figure 2.4: Object virtual frame.

In this method, the object position is replaced by a virtual frame $x_o \in \mathbb{R}^3$, which is assumed to be fixed at the center of each fingertip, as shown in Fig. 2.4. The virtual object position is:

$$x_o = \frac{1}{3}\Sigma_{i=1}^{3} x_i, \tag{2.1}$$

where x_i denotes the position of each fingertip; and $i = 1, 2, 3$ means there are three fingers in this grasp.

Furthermore, to improve the manipulation skills, an adequately modified virtual frame is introduced [162]. In this virtual frame, the position of the virtual frame changes according to the normal direction of desired grasp force f_{di} at each fingertip, as shown in Fig. 2.5. Therefore, the position of the virtual frame x_o is:

$$x_o \triangleq \frac{\Sigma_{i=1}^{3}(f_{di} x_i)}{\Sigma_{i=1}^{3} f_{di}}, \tag{2.2}$$

where term $f_{di} > 0$ refers to the normal desired grasp force.

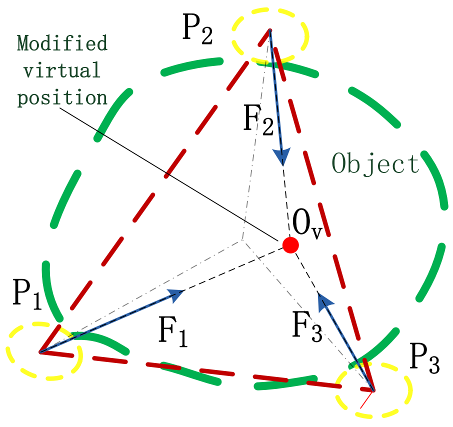


Figure 2.5: Object virtual frame.

Virtual Linkage

Apart from grasp posture, force distribution on each finger is another important aspect in grasp and manipulation tasks. Generally, internal force controller based on the grasping matrix is a widely used method in robotic manipulations [124], [35]. It requires the specific information of the object and the contact position from the object-hand models.

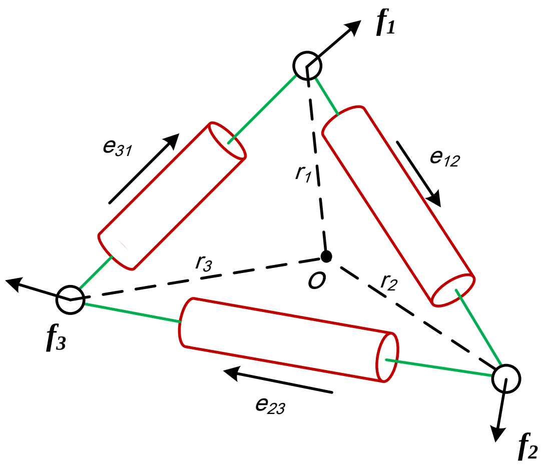


Figure 2.6: A kinematic structure of virtual linkages.

At the object level, the grasp force is assigned by the robot hand under the friction and stability constraints. To model the internal grasp force, a concept of virtual linkage is introduced in [177]. In this concept, a virtual linkage is a mechanism with 6 degrees of freedom in n-grasp manipulation tasks. It is used to model the internal force and the moment applied on the object. In manipulations, the internal grasp force is dependent on object's geometry and motion. Forces and moments applied at a virtual linkage generate joint force and torque at its actuators. A kinematic structure of the virtual linkages is shown in Fig. 2.6, in

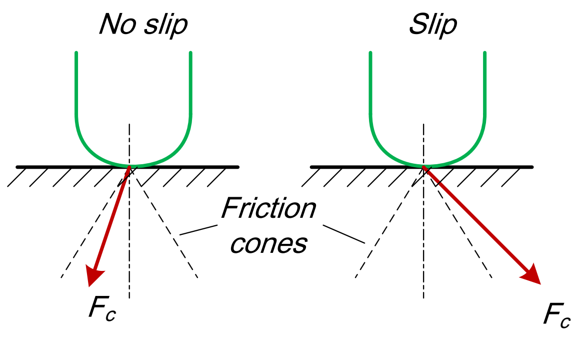


Figure 2.7: Friction cones.

which the object is grasped firmly and the actuated prismatic joints are connected by passive revolt joints.

Similarly, in [157] Stramigioli introduced a grasp model based on virtual linkages. In his method, there is a virtual object connected by grasping fingertips. The virtual object consists of several spatial springs which are determined by the virtual position of the fingers. The object pose is determined by considering the net forces and applying these to the virtual object. In [180], Wimböck et al. apply intrinsically passive control (IPC) into Stramigioli's virtual object model to achieve object's motions and specific grasp force. Its control law takes the desired object frame and desired grasping forces as input. It is passive, and the stability can be achieved even when a finger looses contact with the object. In [179], Wimböck et al. also specify the damping terms in the stiffness control of robot hands. With the model of the virtual object frame, Li et al. measured the manipulation impedance in a human grasp and applied the inspired impedance into conducting robotic grasping and manipulations [109].

However, the method of the virtual linkage does not take the contact constraints on the fingers into account. Furthermore, the object's manipulation can not be guaranteed because parts of dynamics are not considered, like contact deformations on each fingertip and the object surface [161].

2.3.3 Contact Models

Since the object is manipulated by fingers through the contact, contact models play an important role in both grasping and manipulation tasks.

Friction cone is one of the most used contact models in Classical Mechanics. In this model, the non-slip boundary of the friction is represented by a surface with the shape of a cone. In order to achieve a stable contact, the contact force $\mathbf{F}_c$ has to stay in the cone, as shown in Fig. 2.7. A detailed discussion of the friction cone model is in section 3.1.

The friction cone model is simple and efficient. Because it ignores the deformation on the contact surface. However, most grasp and manipulation tasks are carried out by soft fingertips whose surface is made from soft materials like silicon rubber, where large contact force may change the shape of the fingertips dramati-

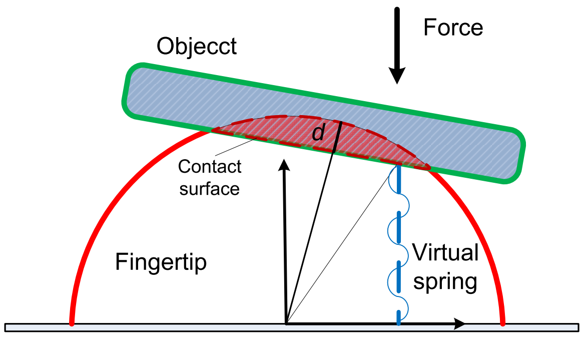


Figure 2.8: Elastic hemispherical fingertip contact model [80].

cally. Hence, the deformations on the fingertips can not be ignored. Elastic contact models consider the deformations on the contact surface. For non-adhesive elastic contact, one of the most famous theories is proposed by Hertzian [87]. Hertzian's contact model contains two elastic objects with arbitrary curved surfaces, where the normal contact force generated between an elastic sphere and a plane is expressed as:

$$F = \frac{4\sqrt{R}}{3}(\frac{E}{1-\lambda^2})d^{\frac{2}{3}}, \tag{2.3}$$

where term R denotes the radius of the elastic sphere; term E is Young Modulus; λ refers to Poisson ratio, and term d is the maximum displacement of the sphere. Moreover, Kao improved Hertzian's model with abstracting two parameters c_d and ς [92]. As a result, Eq. 2.3 changes to:

$$F = c_d d^{\varsigma}. \tag{2.4}$$

Furthermore, a straightforward contact model concerning hemispherical soft fingertips has been proposed by Inoue and Hirai [80]. They modeled the elastic force with potential energy equations. The deformation of the soft fingertip is modeled with an infinite number of virtual springs. The elastic hemispherical fingertip contact model is shown in Fig. 2.8. This model reveals that the relationship between the fingertip's deformation and the contact force concerns two variables: the maximum displacement of the fingertip and the orientation angle of a contacting object. Thus the final contact formula is:

$$F = \frac{\pi E d^2}{\cos\theta_p}, \tag{2.5}$$

where term E denotes the orientation angle of the contacted object. Besides, contact models can also be used to represent the constraints on rolling contact in theoretical models. Suguru et al. proposed a contact model, where the deformation of soft fingertips was solved from the angle of elastic potential energy by the Lagrangian function [18].

In most manipulations, contact points on the objects do not move. However, in some cases, the contact points may move on the surface of the object in the

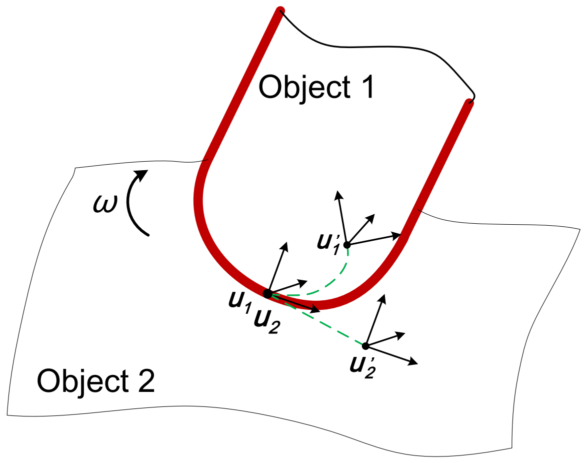


Figure 2.9: The rolling contact cases in [129]. $\mathbf{u}_1$ is the contact frame fixed on the object 1 (finger), and $\mathbf{u}_1$ is the contact frame fixed on the object 2 (manipulating object). After rolling forward, the new contact frames will be $\mathbf{u}_1'$ and $\mathbf{u}_2'$ respectively.

manipulation. According to the research in [129], the contact can be divided into two cases: rolling and sliding in manipulation problems. For a pure rolling in the contact plane, the constraint model is considered. The contact frames are built as shown in Fig. 2.9, where frame $\mathbf{u}_1$ is fixed on the object 1 (finger surface), whereas frame $\mathbf{u}_2$ is fixed on the object 2 (a plane). Each frame has velocity $\mathbf{v} = (v_x, v_y, v_z)^T$ and angular velocity $\omega = (\omega_x, \omega_y, \omega_z)^T$. As assumed in most manipulations, where thee is no relative velocity in the normal direction of the contact surface. When the finger rolls forward (along with the direction v_x), the constraint makes the linear velocity equal to zero, $v_x = 0, v_y = 0, v_z = 0$. On the other hand, if there is no spin on the object 1, there are two more constraints on angular velocity, $\omega_x = 0, \omega_z = 0$. Hence, the only degree of freedom for the finger surface frame ($\mathbf{u}_1$) is rolling in the angular velocity $\omega_y \neq 0$.

Furthermore, a specific contact spot model has been proposed in [20] to get the contact force in senseless grasping and manipulation. When a soft finger or a soft object contacted, the contact spot adjusts to the material and shape of fingers and object. Moreover, the stability of soft contacts has also been discussed in [20]. The contact spot model is shown in Fig. 2.10, and the contact spot center is expressed as:

$$P = \frac{1}{\int_S \|\delta f(s)\| \, \mathrm{d}s} \int_S \|\delta f(s)\| s \, \mathrm{d}s \tag{2.6}$$

where term $\delta f(s)$ refers to an infinitesimal force distributed over the contact area S.

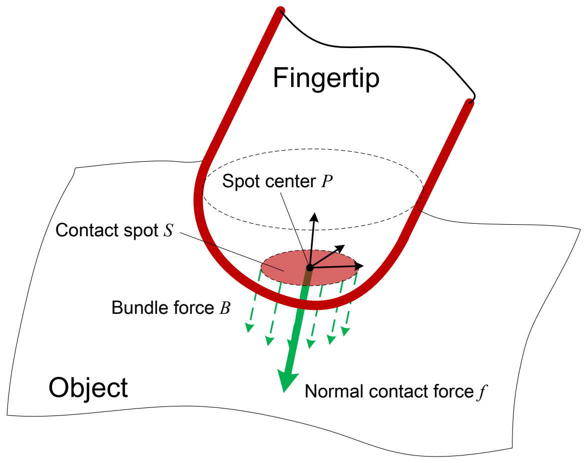


Figure 2.10: Contact spot model.

2.4 Manipulation with Synergies

There are more than 20 joints in a human hand. However, the joints' actions are not independent. On the one hand, it is from the mechanical coupling. Most tendons are internally connected in human hands. On the other hand, neural connections also play an important role in it. For example, Central Nervous System (CNS) controls dozens of muscles coordinately in achieving a grasping posture. This type of muscles' coordinate control is called synergy. Various hand's actions can be obtained through combining weighted postural synergies. In [32], a good overview about modeling natural and artificial hands with synergies is given. The explanations of human full-skilled grasp abilities are that humans acquire grasp and manipulation skills through practices day by day, as a result, their full-skilled abilities are improved little by little unconsciously. Accumulated day by day, humans learn to cooperate their joints to work coordinately. From the view of numerical, synergies simplify the large space of joints into a much smaller control space.

It is revealed that not all of the finger joints of the human hand are controlled independently. As it was tested by Santello et al. in [150], in human grasp tasks, the change of hand postures are not continued, but rather discrete. Thus according to Santello's experiments, there is a considerable reduction in the number of controlled DOFs. Flavigné and Perdereau consider the relationship between muscles and DOFs of the hand to model synergies mathematically from an existing taxonomy of grasps [60].

The synergies' research is not only limited to grasping research, but also to the research of human manipulation. Thakur et al. researched the synergies in different manipulation tasks (unconstrained haptic exploration) [164]. In their research, 9 synergies were identified and they defined a 9-dimensional space. All manipulation action can be considered as a trace or curve inside this 9-dimensional space.

Gradually, the research results of the human hands' control have been applied to design and analyze robotic hands. One of the first research applying synergies to robotics was done by Ciocarli et al. [46]. Ciocarli et al. proposed a general approach to reduce the dimensions for robotic hands with synergies [46]. In his work, each hand posture is combined by different 'eigengrasps' which are the principal components of grasp postures. One eigengrasp is a d dimensional vector and it can be described as:

$$\mathbf{e}_i = [e_{e,1} \quad e_{e,2} \quad \cdots \quad e_{e,d}]^T, \tag{2.7}$$

where term $e_{e,d}$ denotes one eigengrasp at the $i-$th degree of freedom. Therefore, a hand posture p is given by:

$$\mathbf{p} = \sum_{i=1}^{b} a_i \mathbf{e}_i, \tag{2.8}$$

where term a_i is the i-th element of amplitude vector, $a_i \in \mathbb{R}$. Furthermore, Bernardino et al. generated hand postural synergies from dexterous robot hands teleoperated by a human via a data glove in different precision grasping tasks [27]. Postural synergies were successfully applied in anthropomorphic hands' grasping, where twelve objects of different shapes and sizes were grasped based on eight different precision grasps [76].

Besides, in [65], Gabiccini et al. focused on the distribution of the optimized grasp force with the limit of synergies. He considered the elasticity in the grasping system and proposed a soft under-actuated model to optimize the grasp synergies. In their model, for a synergistic displacement, input command $\delta\sigma$ and joints reference position q_r have a linear relationship. The linear map $\mathbf{S} \in \mathbb{R}^{n \times s} \quad (1 \leqslant s \leqslant n)$:

$$\delta q_r = \mathbf{S}\delta\sigma. \tag{2.9}$$

Moreover, to evaluate the distribution of the grasp force, in [65] the cost function is defined by:

$$f = \mathbf{G}_K^R \omega_e + \delta f_{hr_s} + \delta f_{ho_s}, \tag{2.10}$$

$$\delta f_{hr_s} = E_s y, \tag{2.11}$$

$$\delta f_{ho_s} = P_s z, \tag{2.12}$$

where term $\mathbf{G}_K^R = \mathbf{K}\mathbf{G}^T(\mathbf{G}\mathbf{K}\mathbf{G}^T)^{-1} \in \mathbb{R}^{c \times 6}$ is $\mathbf{K}$-weighted inverse of grasp matrix $\mathbf{G}$. It minimizes potential energy $\frac{1}{2}\delta\xi_{0f}^T \mathbf{K}\xi_{0f}$ by fixing $\mathbf{G}_K^R \omega_w$ [75]. Matrix $\mathbf{E}_s \in \mathbb{R}^{c \times e_s}$ is a basis of active internal force δf_{hr_s} which can be controlled by the synergistic displacement $\delta\sigma$. The matrix $\mathbf{P}_s \in \mathbb{R}^{c \times p_s}$ is a basis of passive internal force δf_{fr_s}, which corresponds to the contact force reloaded at the beginning of the grasp operation. Therefore, the optimal distribution of the grasp force can be achieved by minimizing the cost function with respect to $y \in \mathbb{R}^{e_s}$. The corresponding variation of the joint torque is

$$\delta\tau = \mathbf{J}^T(\mathbf{I} - \mathbf{G}_K^R \mathbf{G})\mathbf{K}\mathbf{J}\mathbf{S}\delta\hat{\sigma}, \tag{2.13}$$

and the associated variation of synergistic force is

$$\delta\eta = \mathbf{S}^T \delta\tau. \tag{2.14}$$

Moreover, not only in robotic grasping research but also in manipulations synergies play an important role. Xu et al. applied the synergies to the robotic manipulation and the robotic hand's design. They designed an under-actuated anthropomorphic hand with two synergies synthesized from an in-hand manipulation task [182]. Planetary gear and flexible shafts are used to achieve the control of 19 hand joints. With these two synergies, the task of rotating two rehabilitation training balls has been performed successfully. Catalano et al. also developed a robotic hand (Pisa/IIT SoftHand) with adaptive soft synergies [42]. This hand consisted of 19 joints and one actuator, and the hand's soft synergies are achieved with innovative articulations and ligaments, which make the hand very soft and robust. Li et al. also used postural synergies to develop a myoelectric prosthetic hand, where linkages and pulleys are used to realize two postural synergies [110]. Furthermore, in their prosthetic hand, six channels of surface electromyography (SMEG) signals are used as the input of his system.

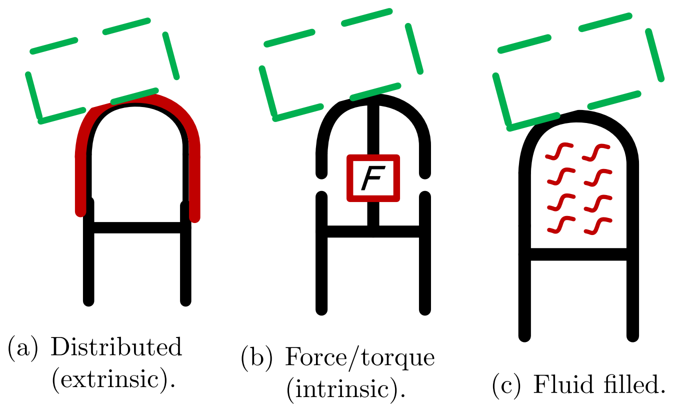
2.5 Multimodal in Robotic In-hand Manipulation

Tactile and visual perceptions are two most frequently used sensing channels in human manipulations. In robotic manipulations, they are of great importance as well. Visual sensing has been widely used to estimate the object's shape and pose. However, the use of the visual sensing is still challenging in robotic manipulations due to its poor performance in the case of occlusions by fingers. Even small errors in an object pose may cause failures in grasp and manipulation. Usually, these failures are almost unavoidable at the stage of grasping execution when the hand equips no other sensors. Therefore, tactile sensors play a significant role in a multi-finger robot hand system. Usually, tactile sensors are employed to measure the force exerted on the fingers.

Vision Based Manipulation

Visual sensing is an important sensing channel in robotics. It has been widely used to estimate the object's shape and pose. With cameras, well-known applications: visual servo and visual tracking are wildly applied in robotics [175], [43]. In robotic manipulation, the main way to apply the visual sensing is the recognition and tracking of objects.

In [106], Kragic et al. proposed a robust vision system for robotic manipulations in real domestic environments. In their system, both foveal and peripheral vision is used to provide depth visual information. The information of object's visual appearance and geometric is applied to object's recognition and localization in the realistic indoor tasks. In the task of catching a lightweight object, Murakami et al. developed a high-speed visual tracking system for robotic re-grasping. Muñoz used visual servo techniques in the task of 2D manipulation [123]. A novel framework has been proposed in [84] by using feedback from visual and tactile sensors. The



(a) Distributed (extrinsic).

(b) Force/torque (intrinsic).

(c) Fluid filled.

Figure 2.11: Three kinds of fingertip sensors.

essential component of this framework is a visual controller, which tracks the motion of objects by both considering the model of the robot hand and the grasping force on fingertips. Gao et al. proposed a pure visual haptic prediction model that enables robots to 'feel' the object without physical interaction. With a deep model, they fused the visual and haptic features generated from visual CNN and haptic CNN for object's classification [66].

Touch Based Manipulation

Tactile sensing is the other most important modalities in robotic manipulations, since the robotic manipulations are mainly based on the contact interaction between the robot hand and the object.

In [107], tactile sensors are defined as devices which are used to measure the object's property and contact events. In [163], authors divide tactile sensing into two type: passive and active sensing. Passive sensing concerns static tactile data. On the contrary, active sensing concerns tactile data during hand's actions. Usually, tactile sensors are mounted on fingertips. In Fig. 2.11 three basic principles of fingertip sensors are shown [163].

However, limited by sensors' sensibility and size, most current tactile sensors are not suitable for the applications of the robotic manipulation. Therefore, many researchers developed various advanced tactile sensors [181], [57], [122], [40], [14].

Besides tactile sensors, the processing of tactile data is another important part in tactile sensing as well. To deal with tactile data which is relatively high dimensional and redundant, image processing method was borrowed [26]. The two-dimensional tactile data is considered as an image and it is described by:

$$m_{p,q} = \sum_{z} \sum_{y} z^p y^q f(z, y) \tag{2.15}$$

where terms p and q denote the order of moment; z and y represent the horizontal and vertical position on the tactile patch; and term $f(z, y)$ is the contact force at position (z, y).

Generally, there are two general purposes for tactile sensing. One is to gain grasp state identification, the other is contact force control. For grasp state identification, the grasp or manipulation state is divided into several different classes. For example, object's dropped and non-dropped classes. In these applications, the tactile data is often combined with visual information and fingers' position, etc. In manipulation recognition research, the tactile data is a supplement to the visual system in the task of understanding and extracting the motion patterns of the hand [77], [174]. With training algorithms, the combined data is used to estimate the upcoming grasp state to keep the manipulation away from failing. In [26], the grasp stability was learned from tactile data acquired at the end of the grasp sequence after the final grasp is applied. In their recognition model, AdaBoost (Adaptive Boosting) and SVM (Support Vector Machine) are implemented to assess the stability of the grasping system. Furthermore, sequences of tactile data are also used in HMMs (Hidden Markov models) to perform a time-series grasp stability assessment in manipulations. In [102], Kojima et al. used both motor angles and contact state to predict the contact stability with an SVM algorithm while performing successfully and unsuccessfully manipulations. Afterward, the sequence of the contact state is used as input to a stability assessment model built with a neural network.

For contact force control, control algorithms are required to plan and control the motion of fingers to maintain the contact force at the desired value and to move the object to a goal state. Inspired by human control mechanism, Joseph et al. presented a novel grasp controller based on tactile sensing [147]. In this controller, both fingertip pressure arrays and hand-mounted accelerometer are used to mimic humans' SA-I, FA-I, and FA-II channels. In [108], based on the balance of the tactile force, Lei and Wisse employed an optimization method to select a suitable grasp region in fast unknown objects grasping tasks. In [179], [180], and [177], tactile force played a role in connecting the object and fingers together as special virtual linkages. With the concept of virtual linkages, both grasping and dexterous manipulation are transformed into distributed contact force control problems.

2.6 Intelligent Manipulation

Nowadays, the term Artificial intelligence (AI) inspires us with its countless extraordinary results in various areas. In the Encyclopedia Britannica, a definition of AI is given by *"Artificial intelligence (AI), the ability of a digital computer or computer-controlled robot to perform tasks commonly associated with intelligent beings "[48].* In a word, it aims to build an agent, which can generate proper actions according to its environments state.

AI has been used in a wide range of fields including robot control [38], machine vision [154] stock trading [168], E-mail processing [74] games [120], medical diagnosis [105], even music generator [169], [54], etc. In computer science, it is an useful tool for many difficult problems [143], [149].

2.6.1 POMDP

Grasping and manipulation tasks sometimes are difficult to be conducted reliably even in laboratory environments. The problems lie in the limited sensing abilities and the uncertainty of the control system. Therefore, the Partially Observable Markov Decision Process (POMDP) is introduced into the multi-finger robot hand systems. One of the first implements of POMDPs can be found in [79], in which the uncertainty state of robotic manipulation is modeled with POMDP. In the tasks of robotic pick-and-place operations, they assumed the robot's position is reasonably certain, but the object's information (pose and shape) are uncertain. Based on POMDPs an optimal control stochastic policy was proposed to solve a simple grasp problem in simulation. In [78], POMDPs were used in the planning of interactive non-prehensile manipulations. Pajarinen and Kyrki proposed an approach which planned over different possible object compositions to model a noisy partially viewed object [134].

2.6.2 Machine Learning

Machine learning is a research of methods aim to build an agent and improving it through interacting with its environment. Generally, there are two categories of learning methods: supervised learning and reinforcement learning. [1]

- **Supervised learning** means telling the agent how to do something. In this learning, the corresponding desired output is obtained in advance and learning parameters are modified by the deviation of the output.
- **Reinforcement learning** means the agent learns how to improve its act according to an observation of its environments. In this learning, every input has a corresponded critical reward, and the learning parameters are modified by the critical rewards.

Supervised Learning: learning from demonstration

Obviously, in supervised learning, additional training examples are required before learning. Hence it is suitable for those problems where training examples can be easily obtained.

One simple way to describe robot behaviors is based on 'IF-THEN' rules, which can be transferred to similar tasks with only slight local changes. The learning of a B-spline fuzzy controller was used in robotic manipulation tasks, from and to which the control rules were extracted and imported [184], [185]. In order to extract the 'IF-THEN' rules, samples are often collected from human's demonstrations. It is a practical method that has been wildly implemented in many human-robot interaction (HRI) tasks. Learning from demonstration (LfD) is an efficient method for transferring human knowledge to robots. After the transformation, with the acquired knowledge, the robots can complete tasks directly.

[1]In some books, the reinforcement learning is included in the unsupervised category.

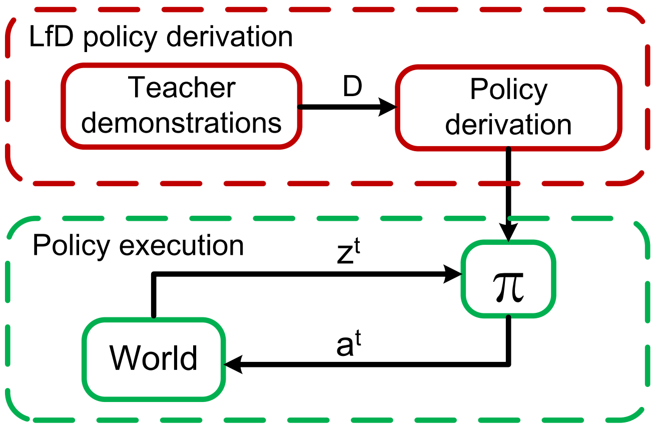


Figure 2.12: The architecture of control policy in LfD (Learning from Demonstration).

For the research of LfD, a comprehensive survey can be found in [17], where the LfD is considered as Supervised Learning problems. In LfD, the task is described by how to learn a function which transfers actions of teachers to robots. Especially, in LfD the training data is collected from tasks demonstrated by teachers. In [17], LfD is segmented into two fundamental parts: gathering examples and deriving a policy from them. On the one hand, a teacher conducts the desired behaviors as examples for the robot; on the other hand, with the examples, the robot derives a policy to reproduce the demonstrated actions. In gathering examples part, with sensors, many desired behaviors of the teacher are recorded as examples. These examples consist of state-action pairs, which can be used by the students to achieve successful executions on robots. In deriving a policy part, they want to get an approximation to the state-action mapping (mapping function), to learn a model of the world's dynamics or to derive a policy from the information to minimize the parameters tuning by a few training examples.

In LfD problems, the world is modeled by states S and actions $\mathbf{A}$ [17]. There is a probabilistic transition function $\mathbf{T}(s'|s, a : \mathbf{S} \times \mathbf{A} \times \mathbf{S} \longrightarrow [0, 1])$, which maps the state to actions. In some tasks the state is partial observable, thus observable state $\mathbf{Z}$ is used in mapping learning instead of state $\mathbf{S}$,. Therefore, the mapping is $\mathbf{M} : \mathbf{S} \longrightarrow \mathbf{Z}$, a policy $\pi : \mathbf{Z} \longrightarrow \mathbf{A}$ is introduced to generate actions based on observations of the environment. In the teacher's execution, both states z and selected actions a are recorded as one demonstration $d_i \in (z^i_j, a^i_j), z^i_j \in \mathbf{Z}, a^i_j \in \mathbf{A}, i = 0 \cdots k_j$. Apart from other learning approaches, the demonstrated data set D is made available to the learner. From the data set D, a policy is derived to enable the learner to select a proper action based on the current state. A typical architecture of control policy in LfD is shown in Fig. 2.12.

Vinayavekhin et al. built a task primitive model treating a human manipulation movement as a sequence of task primitives with the help of a recognition algorithm [172]. In [104], Kondo et al. conducted a manipulation through a sequence of manipulation primitives. They used dynamic programming to recognize

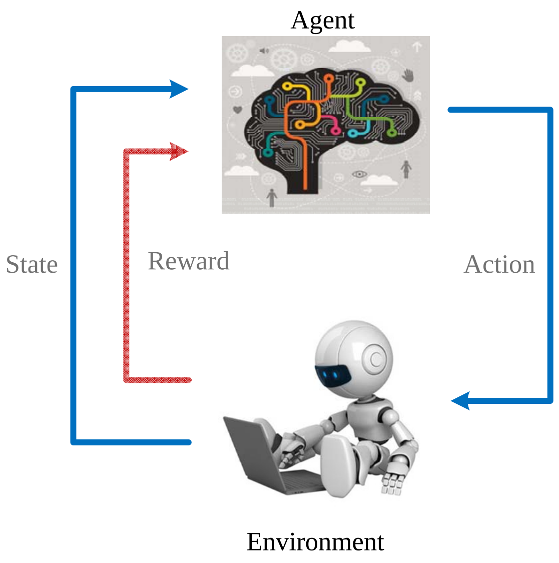


Figure 2.13: Reinforcement learning where an RL agent interacts with its environment.

the contact state between the object and the human hand in a manipulation task. With this system, the contact regions on the operator's hand are detected. Contact based manipulation primitive templates were built through the manipulation experiments; furthermore, Dynamic Programming was used to recognize the contact state out of the manipulation primitives by comparing the similarity between the contact input sequence and the primitive templates. Similarly, in [45] Cheng et al. suggested a kind of human-like action gist to represent the motions of human hands in in-hand manipulation tasks. In this method, a Gaussian Markov Random Field was used to processes the hand actions sensed with a data-glove.

Reinforcement Learning: learning from trial and errors

Rather than from being explicitly taught, reinforcement learning (RL) is a learning method which concerns problems that an RL agent interacts with its environment and its consequent actions and critic rewards are analyzed to improve the agent's performance [69], [88]. During these interactions, the agent learns to select proper actions according to its past experiences (exploitation) or with some new trials (exploration). In this process, only numerical rewards are used to critic the agent's actions with the purpose of maximizing the reward over time. A schematic reinforcement learning model is given in Fig. 2.13

Different from supervised learning where an agent learns from a teacher and the corresponding desired output is given, in reinforcement learning there is no teacher. In other words, the correct input-output pairs are not necessary in RL; and it focuses on online performance by trading off exploration and exploitation. Systematic introductions to reinforcement learning can be found in [88], [24], [159].

The dominant approach of this algorithm is the value-function based approach in the early years. Many planning, controls, scheduling, and game tasks were solved with this value-function based reinforcement learning methods. However, for the problems with infinite state space or the continuous action, this method may not work. Fortunately, combined with neural networks, policy-based reinforcement learning was proposed in the past decades. Several impressive RL implements have been obtained with policy-based RL algorithms. For example, hitting a baseball with an anthropomorphic robot arm was presented in [139], where the gradient policy method was applied into RL for motor primitive control. A successful autonomous aerobatic maneuvers control was achieved on a real helicopter [13], where RL was used to optimize the flying dynamic model and the reward function. A brain-machine interface (BMI) was designed in [55] based on RL, where the BMI control algorithm learned to complete tasks from interacting with the environment.

2.6.3 Learning for State Estimation

In real robotic applications, sometimes the state of the robotic system is hard to be observed directly.

In [102], Kojima et al. proposed a probabilistic learning framework to assess the grasp stability before and during executions with online sensory and proprioceptive data. They built 'one-shot' recognition models with AdaBoost (Adaptive Boosting) and SVM (Support Vector Machine) to demonstrate the ability to learn the grasping stability. Once the final grasp has been applied, the tactile data was acquired as the input of 'one-shot' recognition models to assess the stability. Besides they also constructed two HMMs (Hidden Markov models) to classify the stability of a grasp sequence for 'temporal recognition'. One HMM represented stable grasps and the other one represents unstable grasps. By evaluating their likelihood a 'temporal recognition' can be achieved. Similarly, in [26], another probabilistic learning framework was proposed, where AdaBoost, support vector machines (SVMs), and hidden Markov model (HMMs) were employed to assess grasp stability with tactile data. In [109], Li et al. used Gaussian Mixture Model (GMM) and Support Vector Machine (SVM) to learn a grasping stability estimator. With this estimator, objects' physical properties can be perceived, including the object's weight and the friction on contact areas. In [63], Funabashi et al. proposed a learning method to learn to manipulate various sized and shaped objects with a teleoperated anthropomorphic robot hand via a data glove. In their method, shallow artificial neural networks and deep learning were employed to produce the robot's next joints' state from current joint state and contact state. To implement the network, the learning process was achieved by applying a mini-batch stochastic gradient descent method with 300 successful trails. In [23], Baier-Lowenstein and Zhang presented an automatic value cut-off reinforcement learning algorithm to learn to grasp a wide set of everyday objects.

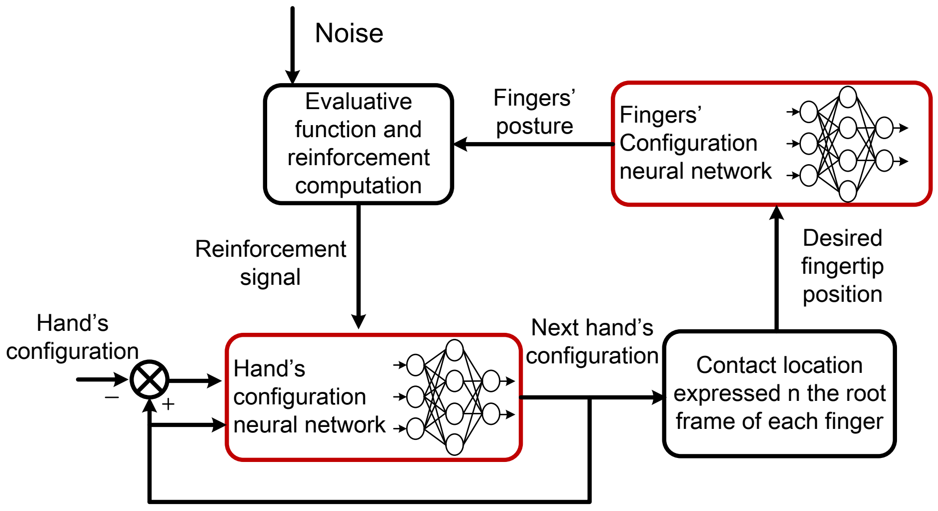

Noise
Evaluative function and reinforcement computation
Fingers' posture
Fingers' Configuration neural network
Reinforcement signal
Desired fingertip position
Hand's configuration
Next hand's configuration
Hand's configuration neural network
Contact location expressed n the root frame of each finger
−
+

Figure 2.14: The architecture of the grasping neural model proposed by Gorce and Rezzoug.

2.6.4 Learning for Action Optimal Control

It has been revealed that animals' control behaviors in nature are naturally stochastic [165]. Additionally, signals in the muscle, joint, tools, and even perception (perceived information) are noise. Moreover, the environments' changing over time is a further source of uncertainty. Inspired by the natural control mechanism, some learning researchers paid attention to statistical estimation theory. In [135], Parisi et al. compared the reinforcement learning with human programming in tetherball robot games. In his work, the robot motor skills were not only trained through learning methods (imitation learning and relative entropy policy search) but modeled with human hard coding as well. Its experimental result shows that the learning approach has a better performance compared with the high-quality hand-crafted system.

Learning methods also play an important role in grasping and manipulation research. In [165], a reinforcement learning algorithm called PI^2 was implemented in the robotic system to learn the shape parameters for a robust grasping to solve the objects uncertain problems (blurry shape and position). Additionally, in [158] authors extend the PI^2 algorithm to PI^2SEQ which optimizes the shape and goal parameters simultaneously to address the fundamental pick-and-place challenge. Moreover, in [137] authors applied PI^2 algorithm to accuracy pool stroke and box flipping tasks with $PR-2$, a dual-arm robot. As a result the robot learned to achieve the box flipping tasks with the result of 172 succeeded in 200 trails after 26 iterations.

In [71], Gorce and Rezzoug proposed a two-stage model to learn the grasping posture of an anthropomorphic hand with little knowledge about tasks and few sensing capabilities. In their model, a four layers neural network was first adopted to model the inverse kinematics of hand fingers; and another three layers neural network with SRV (Stochastic Real Valued) units was built to optimize the hand grasping configuration. The architecture of the hand posture definition model is shown in Fig. 2.14.

Some other research showed that humans use off-line simulation to plan their actions. In [102], the manipulation tasks consisted of three steps: a forward model for sensory prediction, a grasp stability assessment, and a sequence of motions' selection. The forward model for sensory prediction predicted the upcoming sensor state with current state and an action in a sequence of movements. The grasp stability assessment estimated the grasp state. It kept the in-hand object away from slipping and falling off. The sequence of motions' selection is to choose 'good' actions in a sequence to achieve a manipulation task. In [171], Hool et al. proposed a direct learning method to learn a control policy to manipulate an unknown object with tactile information. In their methods, relative entropy policy search was adopted, and the system state was represented by the joints' position and the tactile sensors. Their experiments reveal that the learned policies gain better rewards than a hand-coded feedback policy, although the learned policies are error-prone with large variance.

Chapter 3

Manipulation Models

With the concept of trial and error, robot fingers push in the black box in different directions and visual-haptic feedback is collected to evaluate the pushes' performance. In order to analyze this feedback, several manipulation models have been proposed to illustrate the push attributes of the black box. In this chapter, firstly, contact models between the fingers and the object are presented. Then a fundamental concept model (push and support model) is proposed. Furthermore, two models are proposed with specifying the elastic surface to different support fingers: single and hybrid support model. Both models are discussed based on the grasping matrix. The relation between object's motions and the push actions is deduced.

This chapter is organized as follows:

- In section 3.1, three basic contact models are introduced, and manipulation assumptions are given.
- In section 3.2, a fundamental concept model (named push and support model) is proposed. Derived from this model, four models are discussed based on different support fingers. As a result, two of them are used to guide the robotic manipulations in experiments presented in Chapter 4.
- In section 3.3, a single support model is proposed for two fingers manipulations. In this model, a fixed support finger is used in place of the elastic surface. Furthermore, two rolling cases are discussed based on whether the object rolls on the support finger or not.
- In section 3.4, a hybrid support model is proposed for three fingers manipulations, in which a fixed support finger and a spring support finger are employed instead of the elastic surface. In this model, the fixed support finger acts as a pivot about which the object rotates; and the spring support finger plays the role of a spring which helps rotate the object and moves the object to compensate the contact errors generated by other fingers.
- In section 3.5, enhanced manipulation model is proposed to conduct a rotational manipulation. This model is based on the hybrid support model,

and additional velocities are added to each finger to compensate the position error of the object.

- In section 3.6, a short discussion is given to summarize the work in this chapter.

3.1 Contact Models and Grasping Matrix

Robotic manipulation is an interaction problem where fingers apply contact force to move the object with physical constraints. The contact plays a critical role in manipulations. The force in the contact areas is the medium between fingers' actions and object's wrenches. Hence, contact models and grasping matrix are discussed firstly in this chapter.

3.1.1 Contact Models

Contact points are assumed fixed on an object. A coordinate frame C_i is given to represent the i^{th} contact location (position and orientation); and an object reference frame C_O is fixed on the object. Respecting to the reference frame, a contact location is represented as g_{OCi}. Typically, a wrench $\mathbf{F}_{Ci}$ is used to represent the force $\mathbf{f}_{Ci}$ and torque τ_{Ci} applied at the contact point C_i.

As described in [124], a set of feasible wrench applied by a finger at contact C_i is represented by the wrench base $\mathbf{B}_{Ci} \in \mathbb{R}^{p\times m_i}$:

$$\mathbf{F}_{Ci} = \mathbf{B}_{Ci} f_{Ci}, \qquad f_{Ci} \in FC_{Ci} \tag{3.1}$$

where $\mathbf{B}_{Ci} \in \mathbb{R}^{p\times m_i}$, and p depends on contact type, m_i refers to the dimension of the wrench. In more detail, in 3D problems, $p = 6$; and in 2D problems, $p = 3$. FC_{Ci} is a close subset of $\mathbb{R}^{m_i}$ which constrains the contact wrench.

In order to analyze the external wrench at contact points, three contact models were given by Murray in [124]: frictionless point contact model, point contact model with fiction, and soft-finger model.

Frictionless Point Contact Model

In Frictionless point contact model, it assumes there is no friction between the finger and the object in the contact area. In this case, the contact wrench contains only force components in their normal direction of the contact surface on the object. Therefore, the wrench is represented as:

$$\mathbf{F}_{Ci} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} f_{Ci}, \qquad f_{Ci} > 0, \tag{3.2}$$

where $f_{Ci} \in \mathbb{R}$ is the magnitude of the force applied in the normal direction. Obviously, f_{Ci} must be positive, since the contact force can only be applied by push not pull.

This contact model is simple. However, it is not practical. Most of the robot fingers are covered by materials with a large friction coefficient. Thus frictions in the contact areas can not be ignored. Actually, the friction plays a critical role in the success of grasping and manipulations. Therefore, the application of the frictionless point contact model is limited. A more frequently used model is point contact model with fiction.

Point Contact Model with Fiction

Point contact with fiction indicates the case where friction exists between the finger and the object in the contact area. In this case, the contact wrench contains the force both in normal and tangent directions to the surface of the object, and certainly, torque is not considered in this model.

According to the Coulomb Friction Model, a stable contact means the friction satisfies:

$$|f^t| \leq \mu f^n, \tag{3.3}$$

where μ is the coefficient of the friction, f^t refers to the force component in the tangent direction of the contact surface, and f^n refers to the force component in the normal direction of the contact surface. A geometrical representation of Eq. 3.3 is friction cone; the axis of the cone is to the normal direction of the contact surface, as illustrated in Fig. 2.7. Hence, the cone angle is given by:

$$\alpha = tan^{-1}\mu. \tag{3.4}$$

The wrench at the contact point is represented as:

$$\mathbf{F}_{Ci} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} f_{Ci}, \qquad f_{Ci} \in FC_{Ci}, \tag{3.5}$$

with the constraints:

$$FC_{Ci} = \{f \in \mathbb{R}^3 : \sqrt{f_1^2 + f_2^2} \leqslant \mu f_3, f_3 \geqslant 0\}. \tag{3.6}$$

where f_1, f_2, and f_3 refer to the first, second, and third element of f_{Ci}

Soft-finger Model

Another practical contact model is soft-finger model. It assumes that both force and torque (wrench) exists in the contact areas. Hence, the wrench in the contact area is represented by

$$\mathbf{F}_{Ci} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} f_{Ci}, \qquad f_{Ci} \in FC_{Ci}, \tag{3.7}$$

with the constraints:

$$FC_{Ci} = \{f \in \mathbb{R}^4 : \sqrt{f_1^2 + f_2^2} \leqslant \mu f_3, f_3 \geqslant 0, |f_4| \leqslant \gamma f_3\}, \tag{3.8}$$

where γ is the coefficient of the torsional friction, and f_4 refers to the fourth element of f_{Ci}.

3.1.2 Grasping Matrix

For further analysis, when a contact wrench is transformed from contact frame $C_C i$ to object reference frame C_O, a transformation matrix $Ad^T_{g^{-1}_{OCi}}$ is used. A grasp matrix is defined as [30]:

$$\mathbf{G}_i := \mathbf{A}d^T_{g^{-1}_{OCi}} \mathbf{B}_{Ci}, \tag{3.9}$$

$\mathbf{G}_i \in \mathbb{R}^{p \times m_i}$. The grasping matrix is a linear transformation of contact wrench. It transfers the representation of the contact wrench from the contact frame on the finger to the frame on the object. More generally, if there are k contact points, the sum of the wrench on the object applied by contacted fingers is:

$$\mathbf{F}_O = \mathbf{G} f_C = [\mathbf{G}_1 \cdots \mathbf{G}_k] \begin{bmatrix} f_{C1} \\ \vdots \\ f_{Ck} \end{bmatrix}, \tag{3.10}$$

with

$$\begin{aligned} f_C &= (f_{C1}, \cdots, f_{Ck}) \in \mathbb{R}^m, \\ FC &= FC_{C1} \times \cdots \times FC_{Ck} \subset \mathbb{R}^m, \\ m &= m_1 + \cdots + m_k. \end{aligned} \tag{3.11}$$

Particularly, in 2D problems and with soft-finger model, matrix $\mathbf{B}_{Ci} = \mathbf{I}$[1].

$$\begin{aligned} \mathbf{F}_{OCi} &= \mathbf{G}_i \mathbf{F}_{Ci}, \\ \xi_{Ci} &= \mathbf{G}_i^T \xi_O. \end{aligned} \tag{3.12}$$

[1]$\mathbf{B}$ is a filter that is represented by setting the torque of the twist to 0.

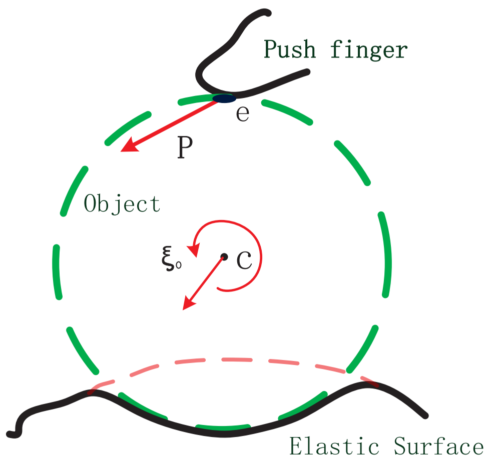


Figure 3.1: A fundamental concept model. In this mode, the manipulation system is considered as a finger pushes a ball which is located on an elastic surface.

where terms ξ_{Ci} and ξ_O refer to the twist regarding to frames C_i and C_O respectively.

3.2 Push and Support Manipulation Models

In this section, a fundamental concept model is proposed to model the robot manipulation. This model assumes the manipulation as one finger pushing an object to roll on an elastic surface. Therefore, this model is also named push and support manipulation model. Based on this model, four other models are deduced from specifying this 'elastic surface'.

In order to simplify the research problems, the following assumptions are considered:

- All actions of the robot take place within its workspace;
- The manipulating processes are kept quasi-static;
- The fingers and object are always in contact;
- Every contact finger has only one contact point locating at the center of fingertip;
- The torque of joints is sufficient for all push actions.

3.2.1 Elastic Surface Model

To illustrate this black box, a general elastic surface model is proposed (also called concept model), as shown in Fig. 3.1, where a ball is located on an unknown elastic surface. This surface consists of support fingers including the fixed support finger and the spring support finger. Hence, the aim of the manipulation is converted to move the ball on a surface. To move the ball, one finger is used to push it. After the push action, the ball moves forward on the surface. **Therefore, with this model, the process of the in-hand manipulation is simplified as a finger pushing a ball which is located on an unknown elastic surface.**

This model is reasonable, since in most human grasps the thumb is usually opposite to others fingers. According to the research in [186], humans' grasps for further manipulations are precise grasp, in which at least the thumb is always opposite to other fingers. According to the concept of the push and support model, **a in-hand manipulation could be a process where the thumb pushes the object and the other fingers act compliantly to adapt the object's motions. Therefore, this manipulation can be viewed as a thumb pushing task, where the thumb tries to push the object in different directions and the other fingers perform as a special surface which supports the object against the thumb's actions and helps the object's motions.** [2]

In Fig. 3.1, $\mathbf{P}$ represents a push action performed by a push finger. This push action consists of two elements: push direction $\mathbf{P}_d$ and push distance P_l. The push direction $\mathbf{P}_d$ is a unit vector. It starts at point e. In 2D tasks, it is represented by one angular parameter. In 3D tasks, $\mathbf{P}_d$ is represented by two angular parameters.

The push action not only rolls the object on the surface but also presses the object down to the surface. These 'roll' and 'down' actions constitute object's motions. Therefore, in this model, the purpose of the manipulation is to choose good push parameters (like push direction $\mathbf{P}_d$ and push distance $\mathbf{P}_l$) to generate the desired object motion by trading off the roll and down motions.

The object's motion is represented by a twist:

$$\xi_o = \begin{bmatrix} \mathbf{v}_o \\ \omega_o \end{bmatrix}, \tag{3.13}$$

where $\mathbf{v}_o$ is the object's velocity; and ω_o is the object's angular velocity. In 2D tasks, velocity $\mathbf{v}_o \in \mathbb{R}^2$ is a vector and angular velocity $\omega_o \in \mathbb{R}$ is a scalar. Whereas in 3D tasks, $\mathbf{v}_o \in \mathbb{R}^3$ and $\omega_o \in \mathbb{R}^3$ are two three-dimensional vectors.

Similarly, the force and torque applied to the object are represented by a wrench:

$$\mathbf{F}_o = \begin{bmatrix} \mathbf{f}_o \\ \tau_o \end{bmatrix}, \tag{3.14}$$

where $\mathbf{f}_o$ is the force; and τ_o is the torque. In 2D tasks, $\mathbf{f}_o \in \mathbb{R}^2$ and $\tau_o \in \mathbb{R}$. Whereas in 3D tasks, $\mathbf{f}_o \in \mathbb{R}^3$ and $\tau_o \in \mathbb{R}^3$.

3.2.2 Discussion on Push and Support Manipulation Model

Normally, it is hard to acquire the attributes of the elastic surface in unknown environments. Fortunately, in manipulation tasks, it is certain that the object is

[2]In some cases, other fingers can be the push finger, and the thumb could be a support finger.

supported by fingers. This makes it easy to analyze the surface explicitly. According to the type of the elastic surface, the elastic support model is specified with two different kinds of supports: fixed support model, spring support model [3], hybrid support model, and double spring support model, as illustrated in Fig. 3.2.

Fig. 3.2 illustrates the possible elastic surfaces and how they are derived. At the beginning, as discussed, the object is located on an elastic surface and one push finger pushes it. It is the elastic surface model: model (a), as shown in Fig. 3.2. Specifically, this 'elastic surface' can be one finger [4], two fingers or even more. In this thesis, only three fingers grasps are considered, hence the cases with one and two support fingers are discussed in this section. They are model (b) and model (c) in Fig. 3.2. Moreover, each support finger shows different properties according to the active controllers inside. If the finger is 'uncontrolled' (keep a certain position with a position controller), the support finger is rigid and performs like a pivot. In this case, the performance of the elastic surface relates more to the soft parts of the fingertips. If the finger is controlled by a stiffness controller. The support finger acts like a spring. In this case, the elastic surface relates more to the finger's compliance attributes. As a result, the elastic surface can be divided into four specific situations: models (d), (e), (f), and (g) in Fig. 3.2. Model (d) is the fixed support model, in which the support finger acts like a fixed pivot. Model (e) is the single spring support model, in which the support finger performs like a spring. Model (f) is the hybrid support model, in which there are two support fingers, one acts like a fixed pivot and the other acts like a spring. Model (g) is the double spring support model, in which both two support fingers perform like a spring. Although these four cases are achievable; they are not necessary in rotational manipulation tasks. To rotate an object, the fixed support model d and the hybrid support model f are simpler and more robust than the single spring support model e and the double spring support model. Without any fixed support finger, the object in models e and g will be pushed down easily which may cause unexpected translation in rotational manipulation tasks. In other words, models e and g are more suitable for translation manipulations; and models d and f are more suitable for rotation.

As a result, in this thesis, we focus on the single support model (model d or e) and the hybrid support model (model f). Actually in three fingers grasps, at most there are two support fingers. We can alway find a direction in which the two support fingertips overlap. In this direction, the manipulation system fits the fixed support model d. In other view directions, this system fits the hybrid support model f. **As a result, with these two models, a 3D manipulation can be divided into several 2D manipulations.** More details will be discussed in section 3.6.

[3]with one spring support

[4]This case also includes two support fingers overlapped. In this case, these two overlapped fingers perform exactly the same. Thus in 2D tasks, the two overlapped support fingers are considered as one.

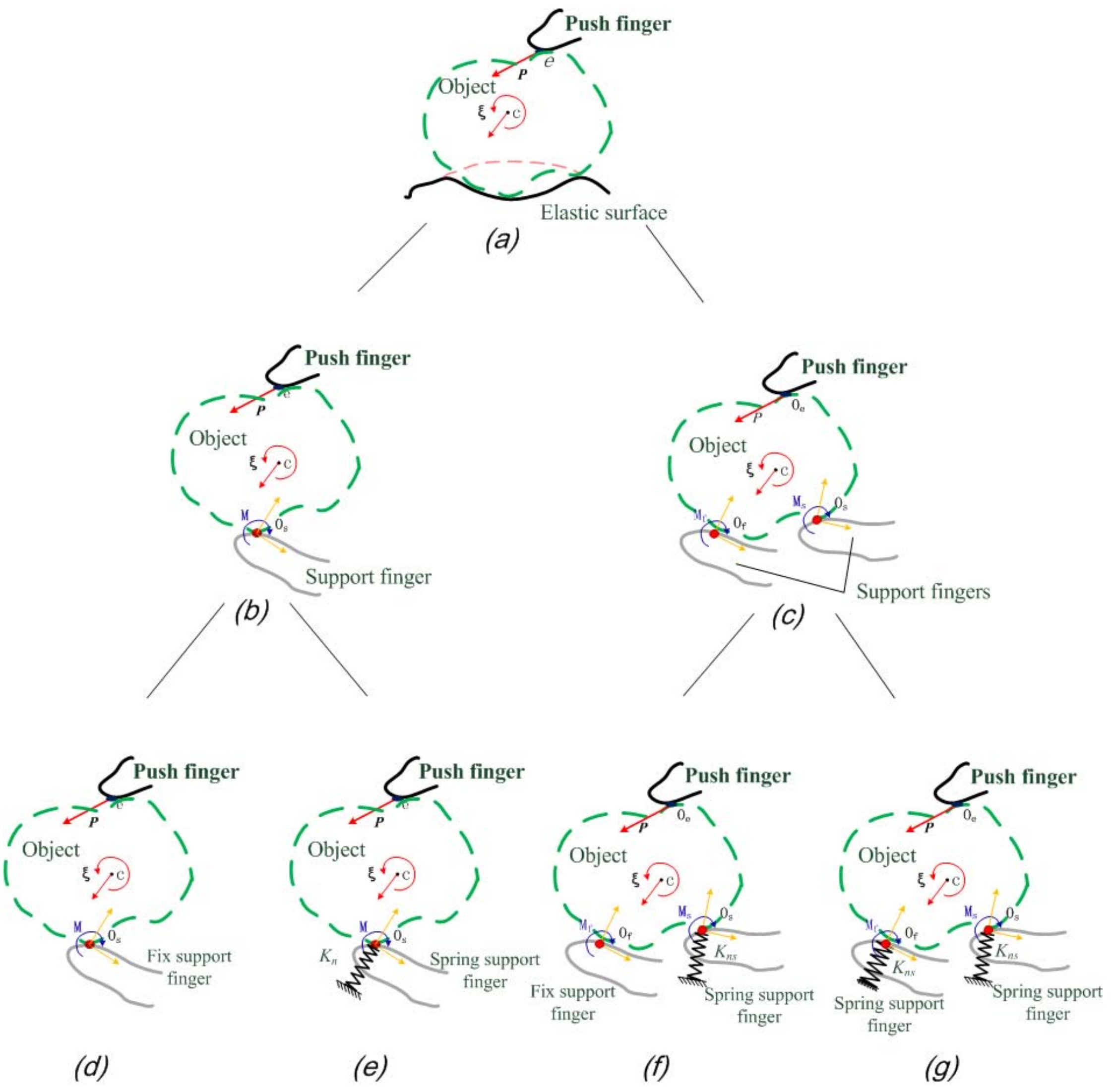


Figure 3.2: The evolution of the elastic surface model.
This diagram shows the derivative of the elastic surface model according to different supports. (a) is the conceptional elastic surface model. First, it is divided into two models based on how many grasping fingers involved in. (b) represents a model with one support finger in two fingers' grasps, and (c) represents a model with two support fingers in three fingers' grasps. Referring to different support finger' attributes (compliance performance), four more specific models are deduced. They are model (d), (e), (f), and (g). Model (d) is fixed support model, in which the support finger acts like a fixed pivot around which the object rotates. Model (e) is single spring support model, in which the support finger acts like a spring. Model (f) is hybrid support model, in which there are two support fingers, one acts like a fixed pivot and the other acts like a spring. Model (g) is double spring support model, in which both two support fingers act like a spring.

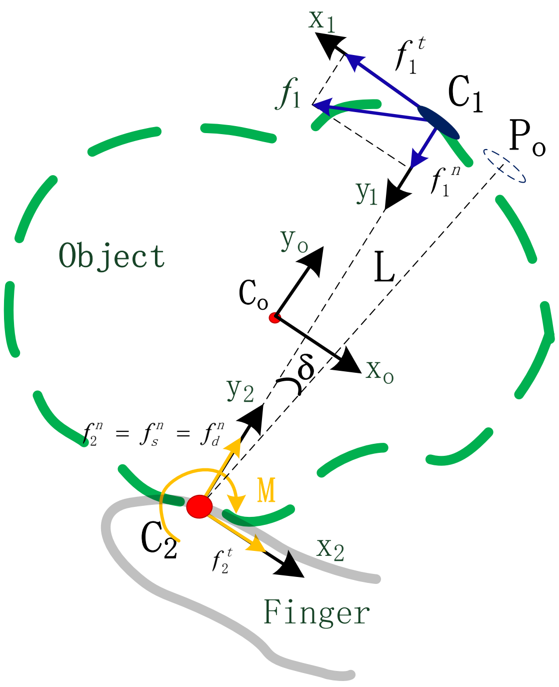


Figure 3.3: The contact force in single support model. In this model two overlapped support fingers are viewed as one fixed support finger.

3.3 Single Support Model

In tasks of manipulations, the elastic surface can be specified by different support fingers as discussed in the previous section 3.2.2. In this section, the 'elastic surface' is one support finger. The elasticity of the surface highly depends on the stiffness of support finger and its contact material. This proposed manipulation model is called single support model. Intuitively, in this model, the push finger can push in all directions; however, an inappropriate push action may cause the manipulations failed (like the thumb pushes in the direction away from the object). Therefore, the constraints of the friction and the stability of the system should always be considered.

3.3.1 Contact Force

A contact force diagram is shown in Fig. 3.3, where an object is grasped by two fingers. The finger on the top is the push finger, and the one on the bottom is the support finger.

According to Eq. 3.12, the wrench applied to the object is

$$\begin{aligned} \mathbf{F}_{OC1} &= \mathbf{G}_1 f_{C1} \\ \mathbf{F}_{OC2} &= \mathbf{G}_2 f_{C2}. \end{aligned} \tag{3.15}$$

The manipulation process consists of a series of grasping configurations. In order to successfully manipulate the object, all these grasping configurations satisfy the force-closure requirement to the applied wrench. Hence, with static equilibrium equation, we have:

$$\sum_{i=1}^{k} \mathbf{F}_{OCi} + \mathbf{F}_{Ogr} = 0, \tag{3.16}$$

where $\mathbf{F}_{Ogr}$ refers to the gravity wrench respecting to the object frame. In this thesis, the object is assumed to be light that the gravity wrench can be ignored, $\mathbf{F}_{Ogr} = \mathbf{0}$.

More specifically, this manipulation is carried out in 2D tasks. Hence the wrench has only three elements. Furthermore, a soft-finger model is adopted to represent the contact on the fixed support finger; and the point contact model with friction is used to model the contact on the push finger. Hence, the contact force is represented by

$$\begin{aligned} f_{C1} &= \begin{bmatrix} f_1^t \\ f_1^n \\ 0 \end{bmatrix}, \\ f_{C2} &= \begin{bmatrix} f_2^t \\ f_2^n \\ \tau_2 \end{bmatrix}, \end{aligned} \tag{3.17}$$

with

$$Ad^T_{g^{-1}_{OCi}} = \begin{bmatrix} \mathbf{R}_{OCi} & \mathbf{0} \\ \hat{\mathbf{p}}_{OCi}\mathbf{R}_{OCi} & 1 \end{bmatrix}. \tag{3.18}$$

Assuming the word frame C_w rotates a small angle respecting to the object frame C_O, and other contact frames are built as in Fig. 3.3, where y_O and y_1 have the same direction. Besides, y_O and y_2 have the opposite directions. Therefore, the transformation matrix has the parameters:

$$\begin{aligned} \mathbf{R}_{OC1} &= \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}, & \mathbf{R}_{OC2} &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \\ \mathbf{p}_{OC1} &= \begin{bmatrix} a_1 \\ b_1 \end{bmatrix}, & \mathbf{p}_{OC2} &= \begin{bmatrix} a_2 \\ b_2 \end{bmatrix}, \\ \mathbf{B}_{C1} &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, & \mathbf{B}_{C2} &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \end{aligned} \tag{3.19}$$

where terms (a_1, b_1) and (a_2, b_2) are the position of the contact frames C_1 and C_2 relating to the object frame C_O respectively. $(a_1 - a_2, b_1 - b_2)$ is exactly the position of the contact frame C_1 relating to the contact frame C_2. Assuming (a, b) is dimension parameters with $a = a_1 - a_2$ and $b = b_1 - b_2$.

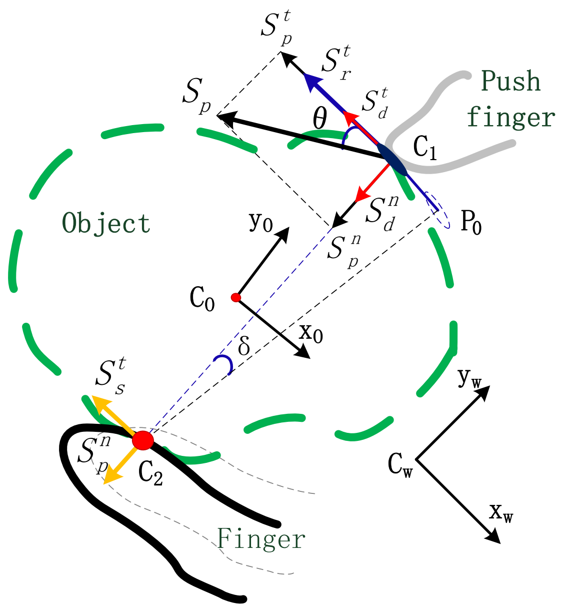


Figure 3.4: Push stiffness mode in dexterous manipulation tasks

From the definition of the grasping matrix in Eq.3.9 and Eq. 3.16, it can be obtained

$$\begin{cases} f_2^n = f_1^n, \\ f_2^t = f_1^t, \\ \tau = -bf_2^t + af_2^n \end{cases} \tag{3.20}$$

3.3.2 Object's Movement

The object's motion is shown in 3.4. The contact point C_1 moves along a vector $\mathbf{S}_p$ after executing the push action. A world frame is built to analyze the push effects, with its x-axis perpendicular to the line C_1C_2. The push action is $\mathbf{S}_p = u$ with an angle θ to x_W. According to the world frame, this push action is represented by

$$\mathbf{S}_p = \begin{bmatrix} cos\theta \\ sin\theta \\ 0 \end{bmatrix} u. \tag{3.21}$$

The vector $\mathbf{S}_p$ is decomposed orthogonally into S_p^t and S_p^n, where $S_p^t = S_p cos(\delta)$ and $S_p^n = S_p sin(\delta)$. Obviously, the motion $\mathbf{S}_p$ comes from the compliance of the deformed system. It mainly consists of three parts: the object's rolling on the support finger, the deformation on the contact surface, and the deformation on the support finger along line y_w. As a result, S_p^t and S_p^n can be expressed as:

$$\begin{aligned} S_p^t &= S_s^t + S_d^t + S_r^t, \\ S_p^n &= S_s^n + S_d^n, \end{aligned} \tag{3.22}$$

where S_d^t, S_d^n refer to the deformations on the contact surface (at both contact points C_1 and C_2); S_s^t, S_s^n refer to the position change in the support finger in orthogonal directions; and the position change S_r^t is due to the object's rotational motion relating to the contact point C_2 on the support finger. Compared to the changes in object's rotation S_r^t, the deformation S_d^t is much smaller, $S_d^t \ll S_r^t$. Therefore, S_d^t is ignored in the following sections.

As discussed in [80], the contact deformation relates to many factors such as materials of the contact surface, object's shape, applied force, etc. Thus the relationship between the contact deformation and contact force is complicated. Takahiro and Shinichi proposed a contact model for soft fingertips with $f = \frac{\pi E d^2}{cos\theta_p}$, where term E_d refers to Young's modulus of soft finger materials, and θ_p is the object's orientation relating to the fingertip. However, **When the contact force is small** (less than 5 N), **a linear model is sufficient to represent the relation between the contact force and the deformation on the fingertip** concluded from the experiment results in [80].[5] Therefore, we assume $f_d^n = -K_d^n S_d^n$, where K_d^n is a special elasticity coefficient.

Inspired from human arm stiffness, [70] the stiffness ellipse model is adopted to describe the stiffness performance on the support finger. Thus $f_s^t = -K_s^t S_s^t$, $f_s^n = -K_s^n S_s^n$. Terms K_s^n, K_s^t are the stiffness of fingers acquired from the stiffness ellipse model. The normal component of the push force f_p^n is described with S_p:

$$f_p^n = -\frac{K_s^n K_d^n}{K_s^n + K_d^n} S_p sin(\theta). \tag{3.23}$$

3.3.3 Object's Two Roll Cases

About the tangent components of the push force, two cases are worth mentioning according to whether the object rolls or not.

Rolled case

The rolled case denotes the object rolls on the support finger during the manipulation. In this case, the displacement S_r^t is much larger than the deformation term S_d^t, since the object rolls on the finger. Hence, in this case, assuming $S_d^t = 0$ and torque τ is rolling friction which can be calculated from:

$$\tau = C_{rr} f_p^n, \tag{3.24}$$

where term C_{rr} is the rolling friction coefficient.

From Eq. 3.23, 3.24 the tangential component of the push force f_p^t can be expressed by S_p:

$$f_p^t = -\frac{K_s^n K_d^n}{K_s^n + K_d^n} C_{rr} S_p sin(\theta). \tag{3.25}$$

[5] In more detail, the result can be concluded in 'Fig. 10' and 'Fig. 11' in [80]. Furthermore, within this small range, the deformation on the fingertip is not sensitive to the object's orientation θ_p.

Unrolled case

The unrolled case denotes the object does not roll on the support finger during the manipulation. In this case, the rolling torque τ is generated from the deformation of the contact area. Here the relation between the rolling torque and the deformed angle γ is simplified to a linear model:

$$\tau = C_d \gamma, \tag{3.26}$$

where term C_d refers to the linear coefficient of the deformation.

With Eq. 3.22 and Eq. 3.26, the tangential component of the push force f_p^t can be expressed by S_p:

$$f_{pt} = -\frac{K_s^t C_d}{L^2 K_s^t + C_d} S_p cos(\theta). \tag{3.27}$$

Critical conditions

Intuitively, the critical condition between these two cases is

$$\delta_c = arctan \frac{1}{K_d^n} \frac{K_s^t}{K_s^n} \frac{C_d}{C_{rr}} \frac{K_s^n + K_d^n}{L^2 K_s^t + C_d}. \tag{3.28}$$

However, δ_c is not a precise angle value, since in the rolled case both deformation and rolling occurs. The shifts between these two cases should be more continuous and smooth.

3.4 Hybrid Support Finger Model

Besides two fingers grasps, another widely used grasping configuration is three fingers grasp, as shown in Fig. 3.5. In this grasp, an unknown object is held by three fingers. The contact points are C_1, C_2, and C_3. An object frame is fixed on the center of the object named C_O.

3.4.1 Contact Force

As discussed in section 3.2.1, there are two support fingers which constitute the elastic surface. Of these support fingers, one is spring support finger; and the other is fixed support finger. In this model, the spring support finger performs as a spring to give an adapting contact force to the object. The fixed support finger acts as a pivot about which the object rotates. It is worth noting that the point contact with friction model is used in this model. In other words, the deformation on fingertips contributes no torque to the object in the manipulation. It is reasonable that the rotational torques generated by the push finger and the spring support finger are much larger than the torque generated by the contact deformation of the fingertips since they have much longer force arms.

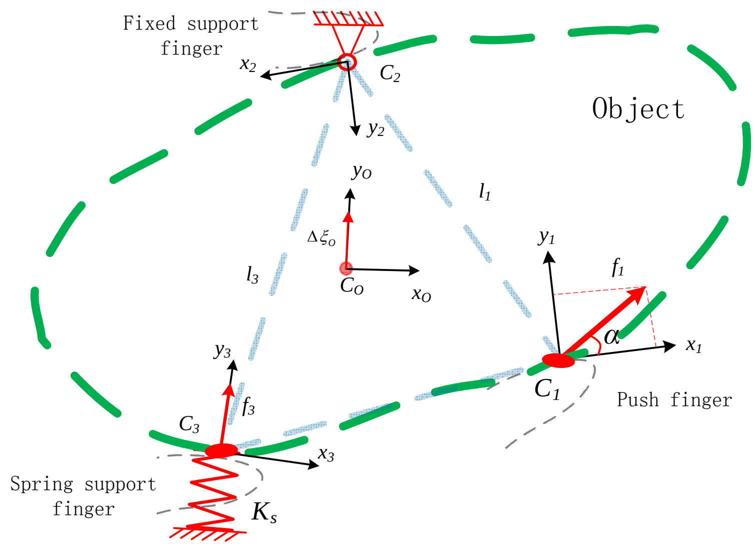


Figure 3.5: Hybrid support finger model.

When a push action $\mathbf{S}_p$ is given by the push finger at $\mathbf{C}_1$. The related push force is

$$\mathbf{F}_1 = \begin{bmatrix} f_1^x \\ f_1^y \\ 0 \end{bmatrix} = \begin{bmatrix} cos\alpha \\ sin\alpha \\ 0 \end{bmatrix} f_1. \tag{3.29}$$

Obviously, this push action is parameterized by an angle α to axis x_1.

The contact force applied by the spring support finger at point C_3 is

$$\mathbf{F}_3 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} f_3, \tag{3.30}$$

with f_3 is the force applied by the spring. Combining with the grasping matrix and static equilibrium equation 3.16, there is

$$\sum_{i=1}^{3} \mathbf{G}_i F_i = \mathbf{0}, \tag{3.31}$$

where $\mathbf{G}_i$ is the grasping matrix for finger i. And the gravity is also ignored here. According to the relation among the coordinate frames, the parameters of the transformation matrices are

$$\mathbf{R}_{OC1} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad \mathbf{p}_{OC1} = \begin{bmatrix} a_1 \\ b_1 \end{bmatrix},$$
$$\mathbf{R}_{OC2} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}, \quad \mathbf{p}_{OC2} = \begin{bmatrix} a_2 \\ b_2 \end{bmatrix}, \tag{3.32}$$
$$\mathbf{R}_{OC3} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad \mathbf{p}_{OC3} = \begin{bmatrix} a_3 \\ b_3 \end{bmatrix},$$

where $\mathbf{B}_{Ci} = \mathbf{I}$, which is ignored in Eq. 3.32. The points $(a_i,\ b_i)$ denote the origin position of the frame i relating to the object frame C_O. As a result, with considering the grasping static equilibrium equation (Eq. 3.31), we have the result:

$$\begin{cases} cos\alpha f_1 - f_2^x = 0 \\ sin\alpha f_1 - f_2^y + f_3 = 0 \\ -b_1 cos\alpha f_1 + a_1 sin\alpha f_1 + b_2 f_2^x - a_2 f_2^y + a_3 f_3 = 0 \end{cases} \tag{3.33}$$

The relation between f_1 and f_3 is:

$$f_1 = \frac{a_2 - a_3}{a_1 sin\alpha - b_1 cos\alpha - a_2 sin\alpha + b_2 cos\alpha} f_3. \tag{3.34}$$

3.4.2 Object's Movement

When the spring support finger performs as a spring, its behavior follows Hook's law. The centerline of the spring is $\mathbf{d}_{spring} = [0\ 1\ 0]$ in frame C_3. The support finger is modeled as

$$\mathbf{F}_3 = -\mathbf{K}_s \mathbf{d}_{spring} \Delta\mathbf{S}, \tag{3.35}$$

where $\mathbf{K}_s$ is the spring coefficient. Considering the elastic direction of the spring, term $\Delta\mathbf{S} = [\Delta S^x\ \Delta S^y\ 0]^T$ is the position change in frame's origin C_3. Writing Eq. 3.35 in a scalar form, $f_3 = -K_S \Delta S^y$. With Eq. 3.34 and 3.35, the push force is calculated by

$$f_1 = \frac{(a_3 - a_2)K_s}{a_1 sin\alpha - b_1 cos\alpha - a_2 sin\alpha + b_2 cos\alpha} \Delta S^y. \tag{3.36}$$

On the fixed support finger, the object rotates around its pivot C_2. The twist $\xi_3 = \mathbf{A}d_{g_{C2C3}} \xi_2$. The parameters are set by

$$\mathbf{R}_{C2C3} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}, \qquad \mathbf{p}_{C2C3} = \begin{bmatrix} a_2 - a_3 \\ b_2 - b_3 \end{bmatrix}. \tag{3.37}$$

Thus the twist of the contact frame C_3 is

$$\xi_3 = \begin{bmatrix} b_2 - b_3 \\ -(a_2 - a_3) \\ 1 \end{bmatrix} \dot{\delta} + \xi_2, \tag{3.38}$$

where δ is the object's rotated angle and ξ_2 is the twist of contact frame C_2. According to elastic fingertip and joints, there is also a small compliance on the fixed support finger. Since $\Delta S = \int \xi_3 dt$, the object's motion according to finger C_3 is

$$\Delta S = \begin{bmatrix} b_2 - b_3 \\ -(a_2 - a_3) \\ 1 \end{bmatrix} \delta + \mathbf{S}_2, \tag{3.39}$$

where $\mathbf{S}_2$ is the position change of contact frame C_2. $\mathbf{S}_2 = [S_2^x,\ S_2^y,\ 0]^T$. And

$$\Delta S^y = (a_3 - a_2)\delta + S_2^y. \tag{3.40}$$

As discussed in section 3.3.2, component of contact force f_2 is assumed by $f_2^y = -K_2^y S_2^y$ with linear elastic coefficient K_2^y of the fixed support finger. Combining with Eq. 3.33, with $f_2^y = sin\alpha + f_3$.

$$S_2^y = -\frac{sin\alpha f_1 + f_3}{K_2^y}. \tag{3.41}$$

With Eq. 3.34 and 3.41, the push force f_1 is represented as

$$f_1 = \frac{(K_2^y - K_s)f_3 + (a_3 - a_2)K_2^y K_s \delta}{K_s sin\alpha}. \tag{3.42}$$

Using ΔS^y instead of f_3, the relation among push force f_1, spring compliance ΔS^y, and object's rotated angle δ is

$$f_1 = \frac{(K_s - K_2^y)\Delta S^y + (a_3 - a_2)K_2^y \delta}{sin\alpha}. \tag{3.43}$$

with Eq. 3.36, a linear relation between object's rotated angle and spring support force is

$$\delta = \frac{sin\alpha}{(a_3 - a_2)K_2^y}[\frac{a_2 - a_3}{(a_1 - a_2)sin\alpha - (b_1 - b_2)cos\alpha} - \frac{K_2^y - K_s}{sin\alpha K_s}]f_3. \tag{3.44}$$

That is $\delta \propto f_3$.[6]

With Eq. 3.44 and 3.34, a linear relation between object's rotated angle and push force is

$$\delta = [\frac{sin\alpha}{(a_3 - a_2)K_2^y} + \frac{((a_1 - a_2)sin\alpha - (b_1 - b_2)cos\alpha)(K_2^y - K_s)}{(a_3 - a_2)^2 K_2^y K_s}]f_1. \tag{3.45}$$

That is $\delta \propto f_1$.

Similarly, the motion of contact frame C_1 is

$$\mathbf{S}_1 = \begin{bmatrix} b_2 - b_1 \\ -(a_2 - a_1) \\ 1 \end{bmatrix} \delta + \mathbf{S}_2. \tag{3.46}$$

[6]Contact force f_i, $(i = 1, 2, 3)$ only relates to force generated in manipulations. In our manipulation experiments, force signals are put on the ground first to reduce the grasping force.

As a result, Eq. 3.36 is written as

$$f_1 = \frac{(a_3 - a_2)K_s((a_3 - a_2)\delta + S_2^y)}{a_1 sin\alpha - b_1 cos\alpha - a_2 sin\alpha + b_2 cos\alpha}. \tag{3.47}$$

The contact point C_1 moves in $\mathbf{S}_1 = [S_1^x, S_1^y, \delta]^T$. It changes the push action in

$$\begin{aligned} S_1^x &= L_{C_1C_2}\sqrt{2(1 - cos\delta)}, \\ &\approx \sqrt{(a_2 - a_1)^2 + (b_2 - b_1)^2}\delta. \end{aligned} \tag{3.48}$$

where $S_1^x = cos\alpha S_1$.[7] Using S_p instead of S_1 to represent a general push distance:

$$\delta = \frac{cos\alpha}{\sqrt{(a_2 - a_1)^2 + (b_2 - b_1)^2}} S_p. \tag{3.49}$$

With Eq. 3.45, a linear relation between push force f_1 and push distance S_s is

$$f_1 = \frac{a_{32}^2 cos\alpha K_2^y K_s}{\sqrt{a_{12}^2 + b_{12}^2}(a_{32} sin\alpha K_s + (a_{12} cos\alpha - b_{12} sin\alpha)(K_2^y - K_s)} S_p. \tag{3.50}$$

To easily note, $a_{32} = a_3 - a_2$, $a_{12} = a_1 - a_2$, and $b_{12} = b_1 - b_2$. For the push finger, the push stiffness of this system is represented as:

$$f_1 = K_{sys} S_p \tag{3.51}$$

where term K_{sys} is the system stiffness coefficient of the push finger.

When word frame C_w is given with its y-axis along line C_1C_2, $a_1 = a_2$. According to Eq.3.49, we have

$$\delta = \frac{cos(\alpha + \gamma)}{\|b_2 - b_1\|} S_p. \tag{3.52}$$

where γ is used to describe the deviation of the ideal and real models. Eq. 3.45 changes to

$$f_1 = \frac{a_{32}^2 K_2^y K_s}{|b_{12}|(a_{32}K_s - b_{12}(K_2^y - K_s))tan\alpha} S_p. \tag{3.53}$$

3.4.3 Control of Spring Support Finger

To make the support finger perform as a spring, one of the most efficient ways is applying the stiffness control where the contact force is used to adjust the finger's behaviors. Therefore, a stiffness control algorithm is implemented to control the spring support finger. Based on the contact force, the finger stiffness is

$$\begin{aligned} f &= f_0 + \Delta f, \\ \Delta f &= -\mathbf{K}\Delta P, \end{aligned} \tag{3.54}$$

[7]The approximation is reasonable when δ is small.

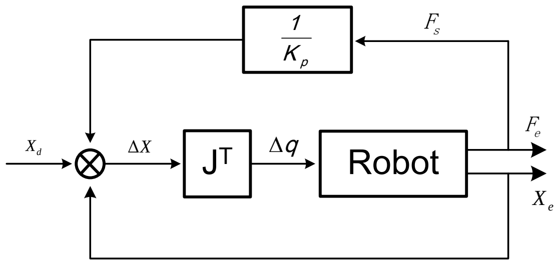


Figure 3.6: The schema of stiffness control.

where term f is the force applied to the finger; Δf is the changed force resulting from the changes in the position P, and term $\mathbf{K}$ is a stiffness matrix. As researched in [51], the stiffness matrix $\mathbf{K}$ is modeled by

$$\mathbf{K} = (\mathbf{C}_s + \mathbf{J}\mathbf{C}_q\mathbf{J}^{\mathbf{T}})^{-1}, \tag{3.55}$$

where $\mathbf{C}_s$ is structural compliance matrix which is from the flexible parts of the finger, like elasticity contact surfaces and other soft structural parts; $\mathbf{C}_q$ is joint compliance diagonal matrix, whose diagonal elements refer to the joints' stiffness; and $\mathbf{J}$ is finger's Jacobian matrix.

According to [65], $\mathbf{C}_s$ and $\mathbf{C}_q$ are set to

$$\begin{aligned} \mathbf{C}_s &= (1/k_{stru})\mathbf{I}, \\ \mathbf{C}_q &= (1/k_{ss})\mathbf{I}, \end{aligned} \tag{3.56}$$

where matrix $\mathbf{I}$ is an identity matrix; terms k_{stru} and k_{ss} are the structural and the joint stiffness respectively.

Referring to the control theory [128], the stiffness control scheme is illustrated in Fig. 3.6, where the 'robot' denotes the spring support finger; F_e refers to an external force; F_s is the contact force provided by contact sensors; and K_s is the stiffness of the finger.

3.4.4 Error Compensated with One Spring Support Finger

In these models, push actions are performed in a straight line. However, it is difficult for the push finger to move by following the line while keeping original contact point in manipulation. As a result, errors occur.

Practically, in manipulations, there are always errors in following the expected contact points on the object. In this research, these errors are called contact errors. In a manipulation system, the contact errors are inevitable. On the one hand, the contact points in models are assumed located at the center of the fingertips. However, their real position is on the surfaces of the fingertips and usually changes during the manipulation. Furthermore, the shape of the contact areas is almost unpredictable. On the other hand, fingers' control also generates errors due to the

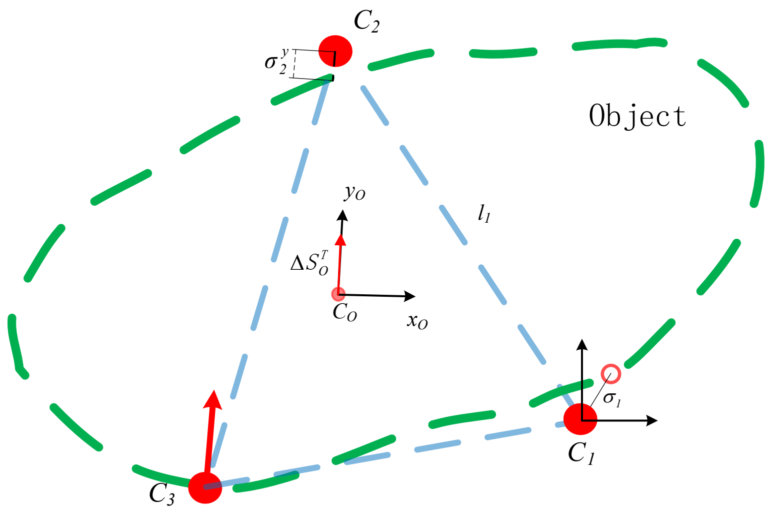


Figure 3.7: Object's translation motion.

low resolution of the fingers' joints. Besides, other factors could also affect the contact state, like object's shape, slips on contact area, et al.

The success of the manipulation is sensitive to the errors σ.[8] Many researchers apply the force control algorithms on each finger to let the fingers compensate their errors respectively. This makes the system very complicated and inefficient. Therefore, a better method is required to compensate these errors in an efficient way. **In this thesis, a novel way is proposed by combining all the errors together and using only one force-controlled finger to compensate them together.**

This thesis focus on a planar rotational manipulation task (2D task), the object's motion is noted as $\delta \mathbf{P}_O$, and the contact points are noted as in Fig. 3.5. Assuming both the push finger and the fixed support finger generates positional errors at their contact points. At the beginning, the spring support finger has a stable contact with the object.[9]

Assuming at an instant state, errors σ_1 and σ_2 are generated by fingertips C_1 and C_2 respectively, as shown in Fig. 3.7. At this state, the basic grasp force can not be maintained, hence the contact force at the point C_3 is reduced. Because

[8] σ is defined as the distance from the current finger position to the final contact position after a firm grasp is achieved. In this research, fingers' errors which are away from the contact surfaces on the object are considered, since the errors which are caused by the finger moving against the object can be controlled easily through contact detection.

[9] This assumption is reasonable. The object is firmly grasped and an initial grasp force is achieved. When the errors occur, the spring support finger moves forward until reaching the object again. Therefore, the object contacts at least one finger: the spring support finger.

of its spring-like attributes, the spring support finger moves forward to push the object generating more contact force. Due to this push action, a planar motion occurs to compensate those errors on other fingers. Therefore, this planar motion is decomposed into two parts: translation and rotation.

At the beginning, the only external force applied on the object is the push force at the contact C_3 (object's mass is ignored). There is no torque. Therefore, the object translates until reaching the finger at the point C_2. After that, with the push action, the object rotates around the pivot C_2 until reaching the finger at the point C_1.

The total object's motion $\Delta\mathbf{S}_O$ and the change in the finger's position C_3 are

$$\begin{aligned}\Delta\mathbf{S}_{C_3} &= \Delta\mathbf{S}^T_{C_3} + \Delta\mathbf{S}^{TR}_{C_3},\\ \Delta\mathbf{S}_O &= \Delta\mathbf{S}^T_O + \Delta\mathbf{S}^{TR}_O.\end{aligned} \tag{3.57}$$

As represented in Fig. 3.7, the object's translation motion is represented by $\Delta\mathbf{S}^T_O$. Obviously, this motion compensates the error σ_2, and simultaneously it generates an additional error σ_1^T to the finger C_1, $\sigma_1^T = \sigma_2$. In this translation motion, the object moves in a line and its contact point on the finger C_2 also moves in the direction parallel to object's current translation velocity. As a result, the compensated error in this step is $\sigma_2 = [0\ \sigma_2^y\ 0]^T$. The object's motion can be described as

$$\Delta\mathbf{S}^T_O = \begin{bmatrix} 0 \\ \sigma_2^y \\ 0 \end{bmatrix}. \tag{3.58}$$

The object's motion is the change in finger frame C_3, that is $\Delta\mathbf{S}^T_O = \Delta\mathbf{S}^T_{C_3}$.

After the object translates, it rotates around the contact C_2. At this stage, a spring force is applied on contact C_3'. C_3' moves to a new position C_3''. This is because a torque is generated by the spring support finger and the torque rotates the object. Assuming this torque is large enough to move the object to reach C_1'' from the position C_1'.[10] Obviously, this motion compensates the errors generated not only by the finger C_1 but also generated in the object's first translation motion.

$$\sigma_1' = \sigma_1^T + \sigma_1. \tag{3.59}$$

As shown in Fig. 3.8, the contact frames and the object frame are built with their $\mathbf{x}$-axes alone the spring direction of the spring support finger. The object's rotational motion is represented by $\Delta\mathbf{S}^{TR}_O$. The rotated angle is δ_O. The relation between frames C_2 and C_3 is $\xi_{C_3} = \mathbf{A}d_{g_{C_2C_3}}\xi_{C_2}$, where ξ_{C_2} is the twist of frame C_2, ξ_{C_3} is the twist of frame C_3. $\mathbf{A}d_{g_{C_2C_3}}$ is a transformation matrix which transfers a motion from frame C_2 to frame C_3. With the same parameters as in Eq. 3.37, [11]

[10]If this torque is not large enough, the system reaches a stable state. A two fingers grasp is obtained. That is the case discussed in section 3.3.

[11]Although there are new errors occurred in this motion, compared to the object's dimension (the distance between the fingers), the errors are too small to affect the transformation matrix. Therefore, this transformation matrix is mainly based on the geometry relative position of the fingers.

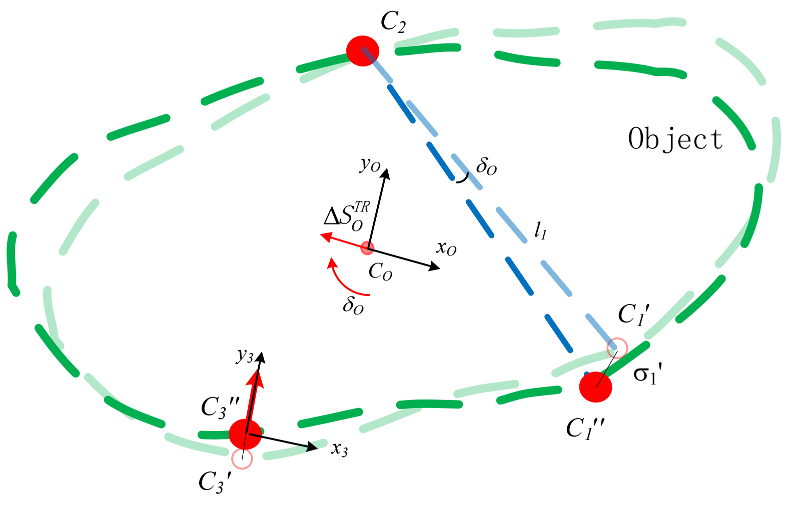


Figure 3.8: Object's rotational motion.

the change in finger's position C_3 is

$$\Delta \mathbf{S}_{C_3}^{TR} = \begin{bmatrix} b_2 - b_3 \\ -(a_2 - a_3) \\ 1 \end{bmatrix} \delta. \tag{3.60}$$

The object's motion is

$$\Delta \mathbf{S}_{O}^{TR} = \begin{bmatrix} b_2 \\ -a_2 \\ 1 \end{bmatrix} \delta. \tag{3.61}$$

The contact for finger C_1 moves from position C_1' to C_1''. Hence line $l_{C_2C_1'} = l_{C_2C_1''} = l_1$. According to frames' position, $l_1 = \|[a_2 - a_1 \; b_2 - b_1]^T\|$. When the rotated angle δ is small, $\sigma_1' = l_{C_1'C_1''}$ and $l_{C_1'C_1''} \approx l_1\delta$. Therefore, the relation between σ_1' and δ is

$$\sigma_1' \approx \sqrt{(a_2 - a_1)^2 + (b_2 - b_1)^2}\delta. \tag{3.62}$$

Hence, Eq. 3.60 and 3.61 can be written as

$$\Delta \mathbf{S}_{C_3}^{TR} = \begin{bmatrix} b_2 - b_3 \\ -(a_2 - a_3) \\ 1 \end{bmatrix} \frac{\sigma_1'}{\sqrt{(a_2 - a_1)^2 + (b_2 - b_1)^2}}, \tag{3.63}$$

and

$$\Delta \mathbf{S}_{O}^{TR} = \begin{bmatrix} b_2 \\ -a_2 \\ 1 \end{bmatrix} \frac{\sigma_1'}{\sqrt{(a_2 - a_1)^2 + (b_2 - b_1)^2}}. \tag{3.64}$$

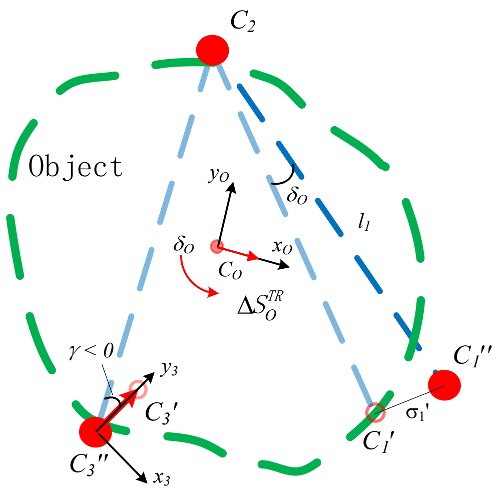


Figure 3.9: Object's rotational motion.

With Eq. 3.58, 3.59, and 3.63, the total motion of the finger C_3 and the object is

$$\begin{aligned}\Delta\mathbf{S}_{C_3} &= \begin{bmatrix} b_2 - b_3 \\ -(a_2 - a_3) \\ 1 \end{bmatrix} \sqrt{\frac{\sigma_1^{x2} + (\sigma_1^y + \sigma_2^y)^2}{(a_2 - a_1)^2 + (b_2 - b_1)^2}} \\ \Delta\mathbf{S}_O &= \begin{bmatrix} b_2 \\ -a_2 \\ 1 \end{bmatrix} \sqrt{\frac{\sigma_1^{x2} + (\sigma_1^y + \sigma_2^y)^2}{(a_2 - a_1)^2 + (b_2 - b_1)^2}}.\end{aligned} \tag{3.65}$$

From Eq. 3.65, it can be concluded that **though new errors are generated during fingers' motions; they all can be compensated by the spring-like attribute of one finger**.

Differently from the case shown in Fig. 3.8 where the object rotates clockwise until reaching the finger C_1'', there is another case. As shown in Fig. 3.9, the object has to rotate anticlockwise to reach the finger C_1''. In this case, the angle between the push direction and the line C_2C_3'' is negative, where $\gamma < 0$. The following process it exactly the same as the clockwise case discussed above.

Error Compensation Conditions

No matter the object rotates clockwise or anticlockwise. There is a necessary condition that the object should be big enough to reach the finger C_1. That is

$$\exists C_1' : l_{C_2C_1'max} > l_{C_2C_1}, \tag{3.66}$$

where term $l_{C_2C_1'max}$ is the maximum length from the finger C_2 to a point on object's contour, as shown in Fig. 3.9. There are two cases for this condition. If $\gamma > 0$, C_1' is located within $\angle C_1''C_3''C_2$; and if $\gamma < 0$, C_1' is located within $\angle C_3''C_2C_1''$. In other words, the condition is that on the object a point C_1' exists that the length

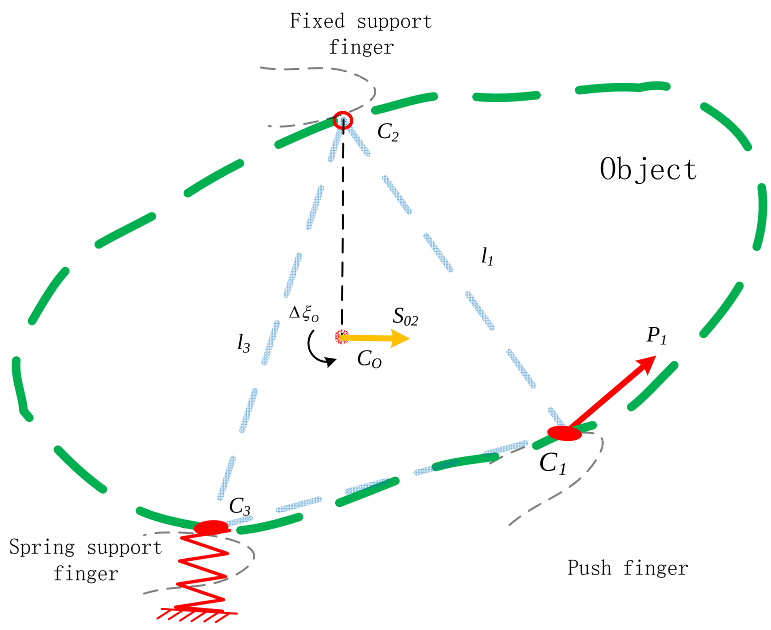


Figure 3.10: Object's translation and rotation respecting to the fixed support finger C_2.

$l_{C_2C_1'}$ is larger than the fingers' distance $l_{C_1''C_2}$ which is sufficient to stop the object from unconstrained rotating.

To summarize, Eq. 3.65 shows it is possible to transfer all fingers' errors into one with a transformation matrix. Importantly, this error transformation matrix proves that position errors in different fingers can be compensated by one spring support finger together. This conclusion is important that it inspires us to focus more on the planning of the fingers' action and to leave the errors to be compensated by the spring support finger. In force control aspect, **this conclusion shows that only one force-controlled finger is sufficient to guarantee a firm grasp in 2D manipulation tasks.**

3.5 Enhanced Manipulation Model

In previous models, the fixed support finger is not controlled according to the contact force. In hybrid support manipulations, the object's motion consists of a small additional translation and an expected rotation. However, in rotational manipulations, it is required to rotate the object without any translation. Therefore, an enhanced manipulation model is proposed in this section to reduce the object's translation motion in our manipulations.

As shown in Fig. 3.10, when the push finger is moving, a translation and an expected rotation occur at the centroid of the object, named $\delta\xi_O$ and $\mathbf{S}_{O2}$ relating to the fixed support finger C_2 respectively. In order to acquire a pure rotation,

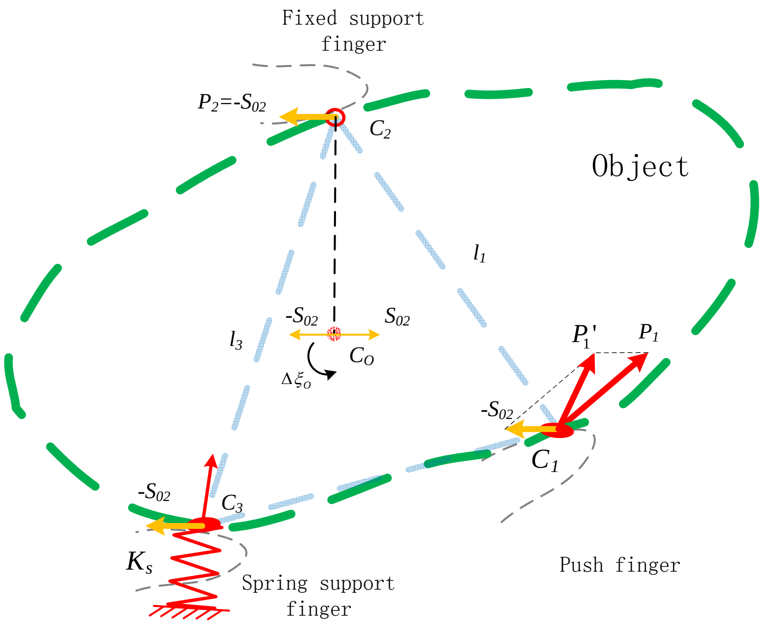


Figure 3.11: An enhanced manipulation model.
In this model, additional motions have been added to each finger to compensate the displacement of the object's position while rotating. The additional motion is $-V_{02}$, which is the opposite motion of the object on its centroid respecting to the fixed support finger C_2.

an additional opposite motion, $-\mathbf{S}_{O2}$, is added to each grasping finger. These opposite motions translate the object in the direction of $-\mathbf{S}_{O2}$, which compensates the object's translation, as represented in Fig. 3.11. $\mathbf{S}_O = \mathbf{S}_{O2} - \mathbf{S}_{O2}$. However, in real robotic manipulation tasks, it is difficult to set $\mathbf{S}_O = 0$. Therefore, a more practical additional oppositional motion is a motion which reduces object's total translation $\mathbf{S}_O$. The idea behind it is simple. It can be considered as adding an external velocity field into this system to make the object move against to its translation $\mathbf{S}_{O2}$. As a result, theoretically, a pure rotational manipulation can be achieved.

In Fig. 3.11, three additional motions are added to each finger to generate the 'velocity field'. For the fixed support finger, the motion of the fixed support finger[12] is $-\mathbf{S}_{O2}$. The fixed support finger moves according to

$$\mathbf{P}_2 = -\mathbf{S}_{O2}. \tag{3.67}$$

For the push finger, as the additional motion is added to the push action; the new push action is

$$\mathbf{P}'_1 = \mathbf{P}_1 - \mathbf{S}_{O2}. \tag{3.68}$$

[12]Here we still use the term 'fixed support finger' to identify the finger played the fixed role in our other models; although it moves in this model.

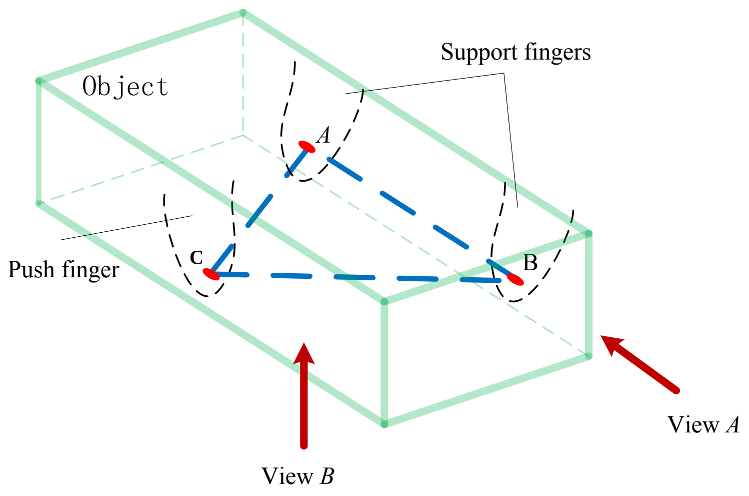


Figure 3.12: Different perspectives of three fingers grasping.

Similarly to the fixed support and push fingers, an additional motion should also be added to the spring support finger. Practically, it is not necessary. Since the additional motion is really small (about 10% of the push distance estimated from the results of hybrid support experiment in section 4.6); they can be considered as errors generated by the fingers according to the current object position. Thereby as illustrated in section 3.4.4, the errors generated by other fingers are able to be compensated by the spring support finger. As a result in this enhanced manipulation model, the spring support performs the same as in the hybrid support model.

Generally, this enhanced manipulation model has two main advantages in rotational manipulations. The first one is reducing the position errors of the object. Theoretically, this model makes the position change of the object to zero during the manipulation. It means a perfect pure rotational manipulation can be achieved with this model. The second advantage is that this model enhances the range of the object's rotation. Obviously, it can be seen in Fig. 3.11, the angle between the opposite motion $-\mathbf{S}_{O2}$ and the push action $\mathbf{P}_1$ is larger than 90°. $||\mathbf{P}_1 - \mathbf{S}_{O2}|| \leqslant ||\mathbf{P}_1||$. In order to achieve the same rotation performance, a smaller push distance is required. In other words, the workspace of the push finger is enhanced. Usually, in manipulation experiments, the object's rotational range is limited by the workspace of the fingers (especially the push finger). Therefore within the same workspace, this model is able to enhance the object's rotational range in robotic manipulations. That is why this model is named enhanced manipulation model in this research.

3.6 3D Manipulation Discussion

Until now, all the proposed models refer to 2D manipulation, in which the object rotates in a plane. However, it is also possible to achieve a 3D manipulation by combining our proposed models. Essentially, in a three fingers grasp, the single support model and hybrid support model can be found in all three fingers grasps. Different perspectives of a grasp with three fingers are represented in Fig. 3.12 where a box is grasped by three fingers: one push finger and two support fingers. From the view A, the two support fingers overlapped. When the push finger pushes down or up, this situation meets the single support model. This object's motion is named pitch rotation. From the view B (straight up), it is the hybrid support model when the push finger pushes in a horizontal plane. This object's motion is named yaw rotation. More details of the object's yaw, pitch, and roll rotation are discussed in the following chapter 4. With proper Euler Angles, any 3D rotation (in any direction) can be achieved by intrinsic rotations, $z - y' - z''$ or $y - z' - y''$. The intrinsic rotation sequence is 'Yaw-Pitch-Yaw'. Therefore, a 3D object rotation can be decomposed into yaw and pitch rotations, which can be achieved by manipulations with the hybrid support model and the single support model respectively. As a result, it can be concluded that the proposed two models are sufficient to achieve any 3D rotation manipulations.

Chapter 4

Manipulation Experiments

In the last chapter, manipulation models, which illustrate the relationship between the push actions and the object's motions, are proposed. This chapter focuses on conducting in-hand manipulation experiments to verify the feasibility of proposed manipulation method and models.

In these experiments, the robot keeps repeating manipulations automatically. In this repeating process, different push actions' parameters are tested and their manipulation results are recorded. Besides, the visual and haptic information is adopted to evaluate the performance of the manipulations. As a result, it is possible for the robot to collect sufficient knowledge to improve its manipulation skills.

This chapter is organized as follows:

- In section 4.1, a robotic manipulation system is built. It consists of an anthropomorphic robot hand, tactile sensors, and a visual tracking system.
- In section 4.2, basic experimental procedures are introduced, such as initial grasping configuration, visual and haptic features extraction, push steps, etc.
- In section 4.3, a repeatability experiment is conducted to verify the stability of our manipulation methods.
- In section 4.4, multi-push distance experiment is conducted to seek the relation between the push distance and the visual-haptic features, where the push actions are given with different push distances but in the same direction.
- In section 4.5, a multi-directional manipulation experiment is conducted to seek the relation between the push direction and the visual-haptic features, where the push actions are given in different push directions but with the same distance.
- In section 4.6, a hybrid support manipulation experiment is conducted to test the performance of the spring support finger and verify the proposed hybrid support manipulation model.

- In section 4.7, a robustness experiment is conducted to verify the error compensation performance of the spring support finger given in section 3.4.4. In this experiment, a disturbance is given by poking the object with a human finger. After the disturbance disappeared, the manipulation system is able to move back to a stable state.

- In section 4.8, enhanced manipulation experiments are conducted to verify the feasibility of the manipulation model proposed in section 3.5. In these experiments, large rotational manipulations are achieved; and various objects are tested.

- Finally, in section 4.9, a comparative manipulation experiment is carried out based on the virtual frame method. Its results show that our method has significant advantages on object's in-hand manipulation.

4.1 Experiment Setups

In order to perform manipulation experiments, a robotic manipulation system has been built, as shown in Fig. 4.1. In this system, an anthropomorphic robot hand is mounted on the end of a KUKA arm fixed on a table. An object is grasped by the hand, and a web-camera is located on the table with its face up to the in-hand object. Referring to its functions, the in-hand manipulation system can be divided into three parts: anthropomorphic robot hand, tactile sensors, and visual tracking system.

4.1.1 Anthropomorphic robot hand

The robot hand platform is the Shadow Dexterous Hand [5], [100], as shown in Fig. 4.2. The original hand has 20 actuated DOFs and further 4 under-actuated DOFs for a total of 24 joints (5 joints have been removed for 5 BioTac sensors in our platform). These joints are driven by remote motors through tendons. With hall effect sensors and strain gauges on each joint, the accuracy of the fingers joints is less than 1 degree. The moving range of its joints is close to that of a human hand.

In total, this hand contains 129 sensors, such as tactile sensors on fingertips, absolute position sensors for each joint, force sensors for each actuator, temperature sensors, and motor current/voltage sensors. All the sensor data is accessible for a user via an EtherCAT interface. Besides, with a control board in the hand's palm, this system can be extended via add-ons.

Fig. 4.3 shows the kinematics of this hand. It is worth noting that in the following experiments, 5 DPs (Distal Phalanxes) have been removed, instead, five BioTac tactile sensors are mounted on the fingertips. Hence, the value of the 5 DIP (Distal Interphalangeal Joints) joints (FFJ1, MFJ1, RFJ1, LFJ1 and THJ1) provided by their position sensors is fixed to a certain angle: 20°. Furthermore, the dimension of the DPs has been changed by replacing with BioTac sensors, which will be introduced in detail in the following section.

4.1.2 BioTac Tactile Sensor

Regarding human in-hand manipulations, fingertips and distal phalanges are most commonly used parts. Therefore, in this research BioTac sensors [176], [59] are selected as the tactile sensors in this research. The BioTac sensor is specially designed for the fingertips of the Shadow Hand.

The BioTac tactile sensor is designed to mimic not only the physical properties of human fingertips but also their sensory capabilities. Today, it is one of the leading sensors in the research of the machine touch. These sensors consist of a rigid core, which is surrounded by an elastic liquid filled-skin. Similar to human touch abilities, the BioTac is able to measure: vibration, pressure, and temperature. One advantage of this design is all electronics are protected inside the rigid core.

In this research, the direct pressure value (P_{DC} signals) is adopted to estimate the contact force with 90Hz updating rate. **Importantly, as mentioned in [126], the sensor's pressure value and the contact force is linearly dependent on each other. Hence, it is feasible to use the P_{DC} signal to represent the contact force directly.** In the following experiment, the pressure value, haptic, and contact force value all relate to the P_{DC} signals.

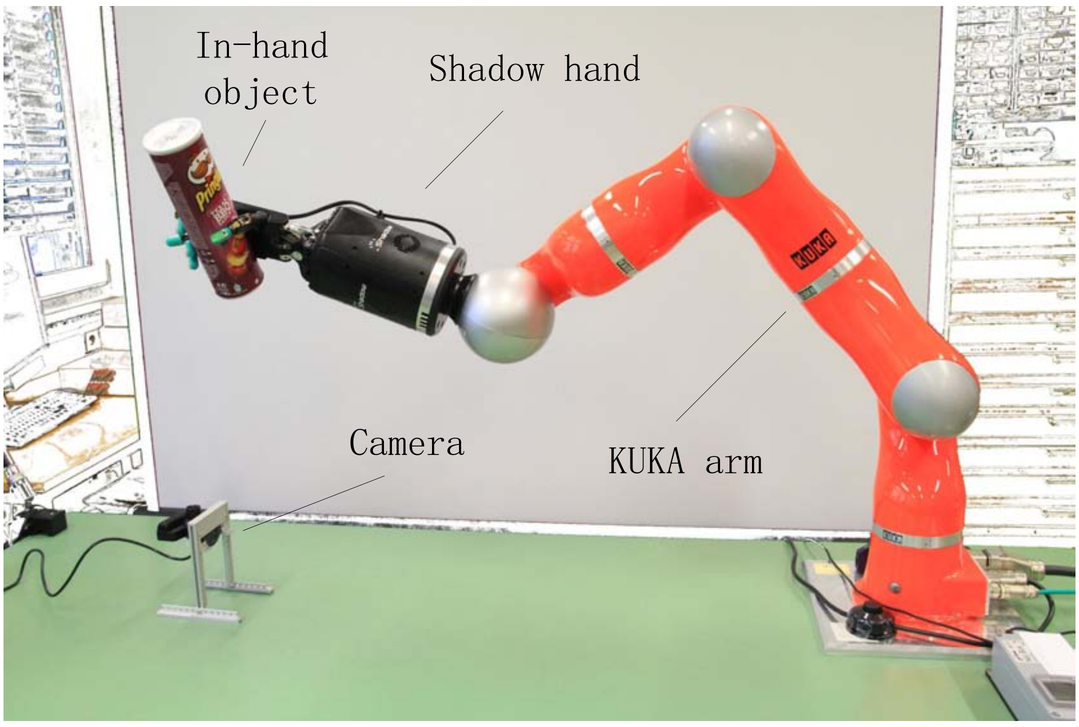


Figure 4.1: In-hand manipulation experiment setup.

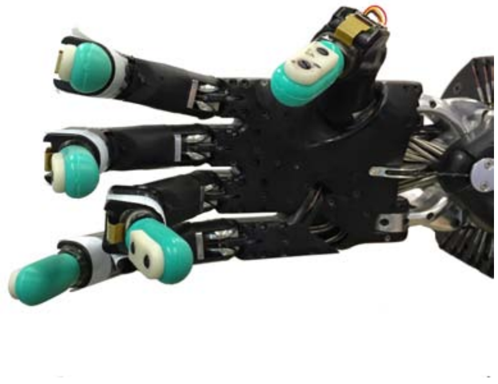


Figure 4.2: Shadow Dexterous Hand.

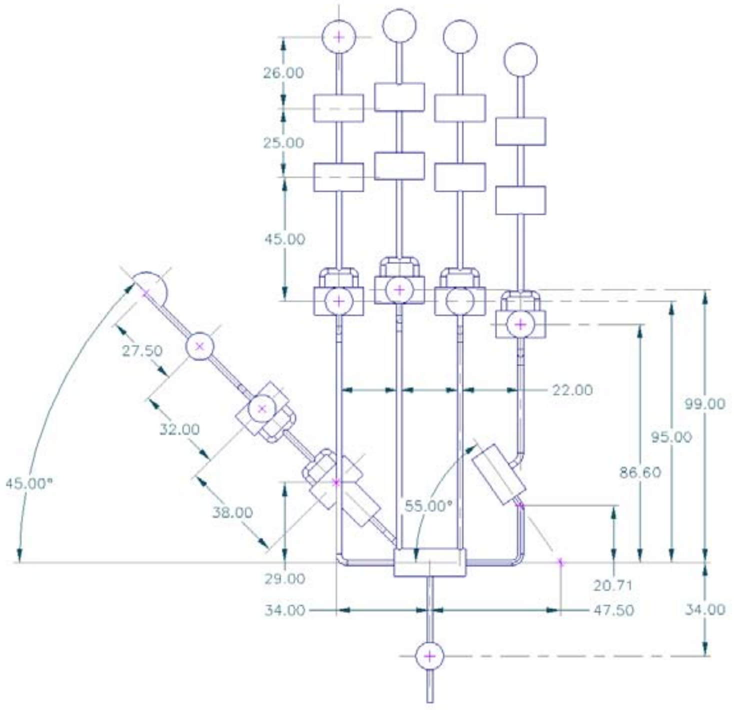


Figure 4.3: Shadow hand kinematics [12].

4.1.3 Vision Tracking System

In order to track the state of the object, the AprilTags system is used in our manipulation experiments. The AprilTags system is a visual fiducial system [89], which is an artificial landmark and designed to recognize and track tags. In this system, tags are conceptually similar to QR Codes. To be detected more robustly and from longer ranges with high localization accuracy, the tags are encoded with far smaller data payloads between 4 and 12 bits. This system is useful for a wide variety of

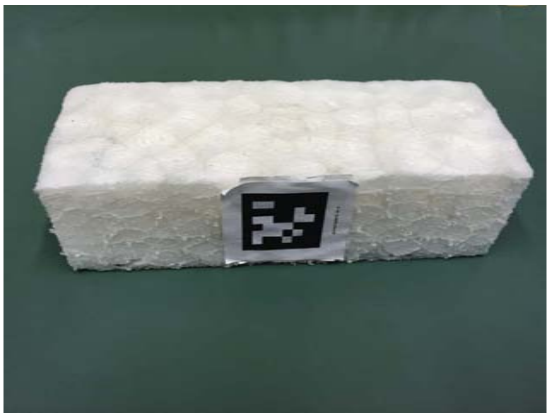


Figure 4.4: An object with an AprilTag.

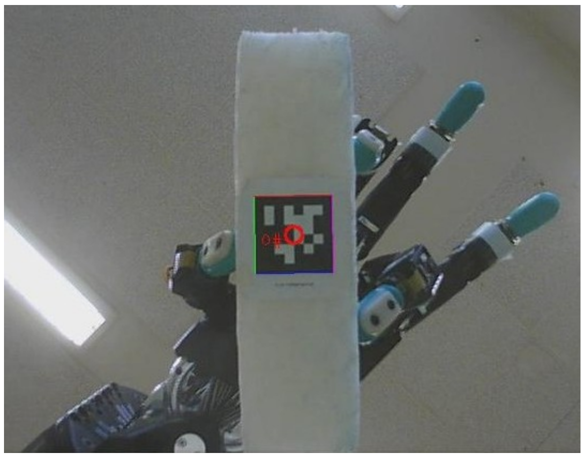


Figure 4.5: An in-hand object is tracked by the Apriltags system.

tasks including robotics, camera calibration, augmented reality, etc. Different from 2D bar-code systems in which the position of the 'code' is unimportant, the tag's relative position and orientation is provided in the AprilTags system. More details can be found in [130].

One advantage of using AprilTags is that it provides precise position and orientation of the target tags which can be created from an ordinary printer. The performance of AprilTags system has been tested in the applications for camera calibration [144].

In this research, AprilTags system is used to track the in-hand object in manipulation experiments. An AprilTag is attached to the bottom surface of the object, as shown in Fig. 4.5. In the experiments, the in-hand object is a foam box, with the dimension of $14cm \times 5cm \times 4cm$.

In the manipulation tasks, the tag always faces down; and a digital camera is mounted facing up to the palm of the hand. This keeps the tag in the visual field of the camera. As shown in Fig. 4.5, when the object is grasped, the tag is detected. This visual tracking system can track a series of tags; however, some characters in

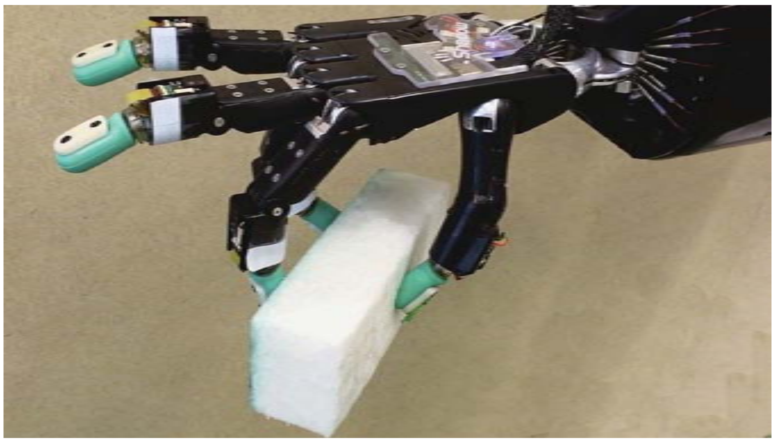


Figure 4.6: A box grasped by the shadow hand with three fingers.

the background or even some parts of the robot may cause visual error detections. Therefore, other tags except the one attached to the object are filtered out.

4.2 Experimental Procedure

In order to verify our 'push-based' manipulation method, experiments are conducted on the real manipulation robot platform introduced in section 4.1. In this thesis, we only focus on the object's manipulation but grasping; therefore the object is assumed already being grasped stably with a given configuration before manipulation experiments. Besides, in this research, a manipulation action mainly consists of two parts: push action carried out by the push finger and the adaptive behaviors from the support fingers. The adaptive behaviors of the support fingers will be discussed in following section 4.6. Hence, in the following experiments, we mainly focus on the push actions. If there is no special noting, the actions refer to the push actions in the following part of this thesis.

Generally, seven experiments are conducted in this section. They are repeatability, multi-push distance, multi-directional push, hybrid support manipulation, robustness, enhanced manipulation, and comparative experiments.

4.2.1 Initial Grasping Configuration and Push Process

Before the manipulations, a stable grasping has been achieved, as shown in Fig. 4.6, where a cubic object is grasped by 3 fingers: thumb, index finger, and ring finger. A corresponding illustrative diagram is shown in Fig. 4.7, where 3 red dots represent contact points (A,B,O) locating on the object surface. The joints' configuration of the initial grasping is shown in Table. 4.1, where the FF refers to

Table 4.1: Three fingers cubic grasping configuration

Joints	FF1	FF2	FF3	FF4	MF1	MF2	MF3	MF4	RF1	RF2	RF3	RF4
Angle [°]	20	50	23	0	20	0	0	-2	20	52	80	-18
Joints	LF1	LF2	LF3	LF4	LF5	TF1	TF2	TF3	TF4	TF5	WR1	WR2
Angle [°]	20	0	0	-25	0	20	20	0	63	-11	0	0

the index finger, MF refers to the middle finger, RF refers to the ring finger, LF refers to little finger, and TF refers to the thumb.

4.2.2 Automatic Manipulation

Due to little prior knowledge about the system, it is almost impossible for the robot to figure out the correct push parameters. Hence, a more practical way for successful manipulations is to let the robot try to push action with different parameters and evaluate the performance of these push actions through the collected visual and haptic information. As discussed in section 1.3.2, this trial process is called haptic exploration, in which the empirical knowledge about the push system is built. With this knowledge, the robot hand can conduct better push actions in the following tasks.

In order to carry out a series of push actions efficiently, an additional move back action is added after each push action. This move back action ensures that different push experiments start at the same initial state so that the comparison of different push actions is possible. As a result, in the manipulation experiments, each push action conducted by the push finger consists of two steps: a forward push and a backwards movement (named *an action pair*). In the forward push step, the push finger moves from its initial point C along a given push vector $\mathbf{P}$ to roll the object. In the backwards movement step, the push finger moves back to its origin C with pressing the object. After these two steps, the whole system returns to its initial state. Some extreme push parameters (such as the push actions with large push distance or in dangerous push directions) may cause the contact slips. It worths to note that sometimes the finger's backwards movement can not bring the system back to its initial state, due to the slips occurred on the contact areas. Therefore, in the following manipulation experiments, the push distance is limited to a small range, and the push directions are limited to a certain range, in order to make sure as few contact slips as possible. With these two push steps, all the manipulations can be carried out automatically, after the object grasped.

4.2.3 Object Frame

As discussed in section 4.1, a camera is installed on the table with its face up to the object. Usually, an additional visual calibration is required before every experiment. However, this makes the experiments inefficient, since we have to calibrate the visual system before every experiment. A more practical way to

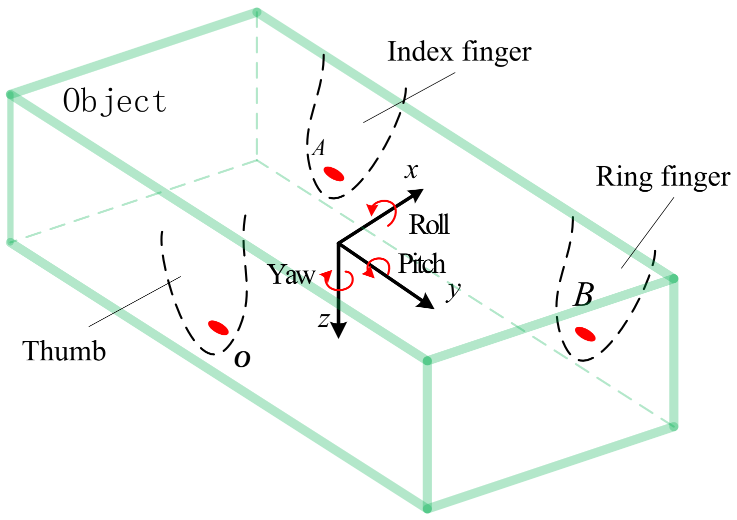


Figure 4.7: Object frame.

describe the object's movement is to fix an initial frame at the initial position of the object's center and to describe all object movements respecting to its 'initial frame'.

In this thesis, we focus on the rotation of the object. Moreover, Tait-Bryan angles are used to represent the object's rotation. They are widely used in the orientation systems of ships', aircrafts', and cars' orientation. **In this thesis, the rotation around axis x is called *Roll*; the rotation around axis y is called *Pitch*; the rotation around axis z is called *Yaw*.** Combining with the object frame, the yaw, pitch, and roll rotations are shown in Fig. 4.7. To easy note, the manipulations aiming to rotate the object in the yaw direction are called yaw manipulations. Equivalently, pitch manipulation aims to rotate the object in the pitch direction; and roll manipulation aims to rotate the object in the roll direction. As discussed in section 3.6, this division transfers a 3D manipulation into a 2D problem, which can be analyzed with the single support model and the hybrid support model proposed in sections 3.3 and 3.4.

In the following experiments, the pitch manipulations and yaw manipulations are discussed separately. However, the roll manipulation is not discussed separately in this thesis. That is because it can be achieved in yaw manipulations, more details can be found in the multi-directional manipulation experiment in section 4.5.

Pitch Manipulation

In a pitch manipulation, the single support model is adopted. Two fingers (index and ring fingers) behind the box are fixed support fingers which hold their position with position controllers. The finger (thumb) in front of the object is the push finger, which is controlled actively and pushes the object in given directions. Hence, a push frame is built with its origin locating at point O, as shown in Fig. 4.8. Its axis x is parallel to the support contact line AB and its axis y is perpendicular to

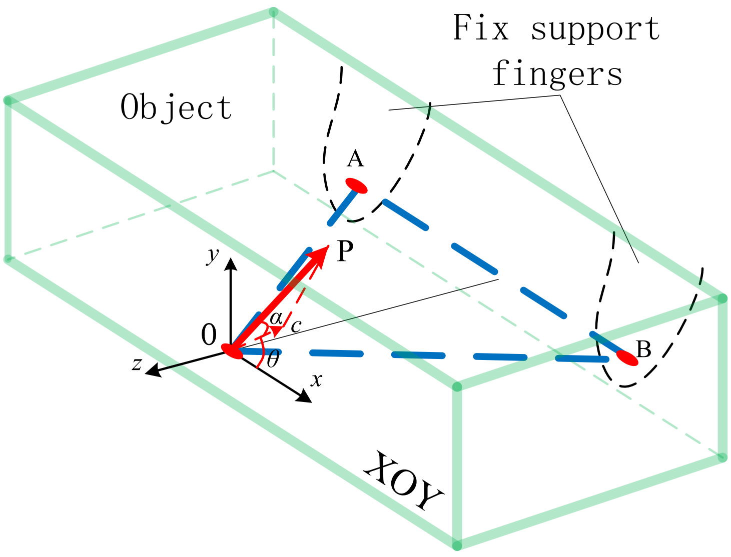


Figure 4.8: Thumb push frame for pitch manipulation.

the grasping plane ABO. A red vector **P** is used to denote a push action. Three parameters: θ, α, and P_l are used to describe this push vector. Parameter α refers to the angle between the push vector **P** and XOY plane; parameter θ refers to the angle between the axis x and vector **OC** which is the projecting of vector **P** in XOY plane. The parameter P_l refers to the length of the push vector **P**. For the purpose of convenient notation, a push vector **P** is written as $\mathbf{P} = [\theta \ \alpha \ P_l]^T$. Besides, a push direction vector $\mathbf{P}_d = (\theta, \ \alpha)$ is used to refer to the push direction.

Yaw Manipulation

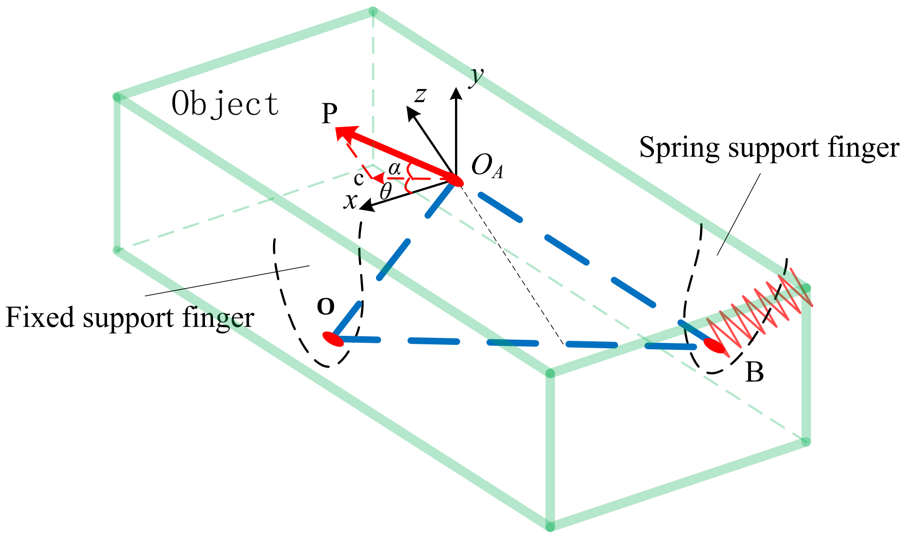


Figure 4.9: Index push frame for yaw manipulation.

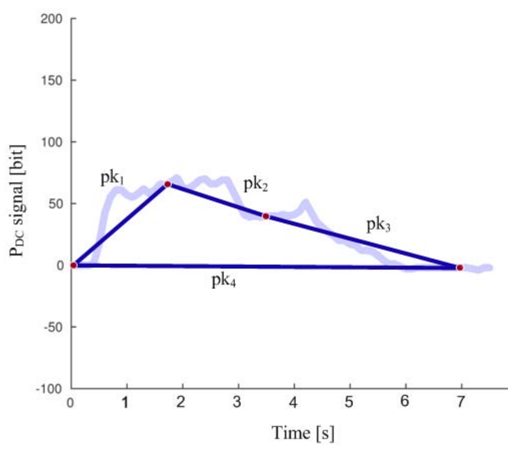
In a yaw manipulation, the hybrid support model is selected. In this model, the index finger acts like the push finger; the thumb acts like the fixed support finger, and the ring finger plays the role of the spring support finger. Hence, a push coordinate frame is built with its origin locating at point O_A, with its axis z perpendicular to line OB and its axis y perpendicular to grasping plane $O_ABO$as shown in Fig. 4.9. A push action is also represented by a red arrow $\mathbf{P}$. Similarly, this push vector $\mathbf{P}$ is also described with three parameters: θ, α, and P_l, with $\mathbf{P} = [\theta\ \alpha\ P_l]^T$. α refers to the angle between the push vector $\mathbf{P}$ and XO_AY plane, and θ denotes the angle between axis x and vector $\mathbf{O_A C}$ which is the projecting of push $\mathbf{P}$ in XO_AY plane. $\mathbf{P}_d = (\theta,\ \alpha)$. Furthermore, the direction of the spring support is selected perpendicular to line O_AB in plane OO_AB.

In the following sections, if it is not noted otherwise, the pitch manipulations are corresponding to the single support model. Its P_{DC} signals (haptic feedback) is collected by the sensor on the thumb. Equivalently, the yaw manipulations are corresponding to the hybrid support model. The P_{DC} signals (haptic feedback) is collected by the sensor on the index finger.

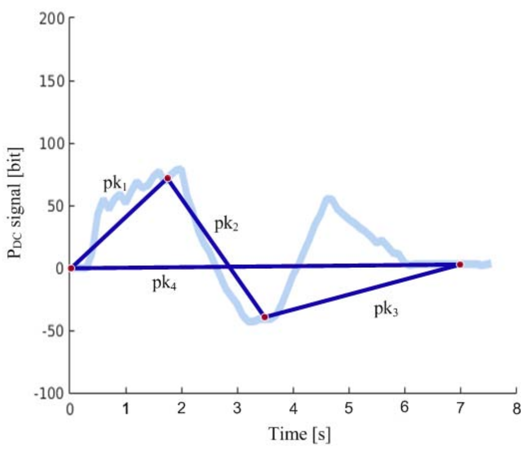
4.2.4 Haptic and Visual Features Extraction

In order to make the haptic signals comparable, all the haptic signals are put on the ground by setting their initial value to zero. In the manipulation experiments, haptic signals have 4 shapes, as shown in Fig. 4.10. To represent these shapes, four sampling points are adopted to generate a haptic feature. These points are sampled at the time step t_0, $t_{1.7}$ $t_{3.5}$, and t_7, where term t_i refers the haptic value at the i^{th} second. The point at the beginning is included, though it is always zero. Furthermore, four lines are plotted to connect the points: t_0 to $t_{1.7}$, $t_{1.7}$ to $t_{3.5}$, $t_{3.5}$ to t_7, and t_7 to t_0. Thereby the continuous haptic signals can be described by shapes represented by four lines whose slope coefficients are pk_1, pk_2, pk_3, and pk_4 respectively. More importantly, these four parameters also have practical meanings. The first slope coefficient pk_1 refers to the rising speed of haptic value. If this value is large, it means the object is difficult to be moved in this push direction. On the contrary, if this value is too small, it means that the contact force reduces too fast. In other words, the finger is likely pushing away from the contact area on the object. The second slope coefficient pk_2 denotes the stability of the second push step. The closer it is to zero, the smoother push action is in the second step. An ideal value of pk_2 is 0, like in Fig.4.10(c). It means there is no further constraint on the way to move the object in this push direction. The third slope coefficient pk_3 shows how fast the haptic feedback decrease in the backwards push process. The last coefficient pk_4 is the haptic value at the time step t_7 since the beginning point is always 0. It illustrates the stability of this push process. If any slip occurs in this manipulation process, the contact force changes dramatically for the tilted attitude of the object after fingers move back to their initial state. This results in a large value of pk_4.

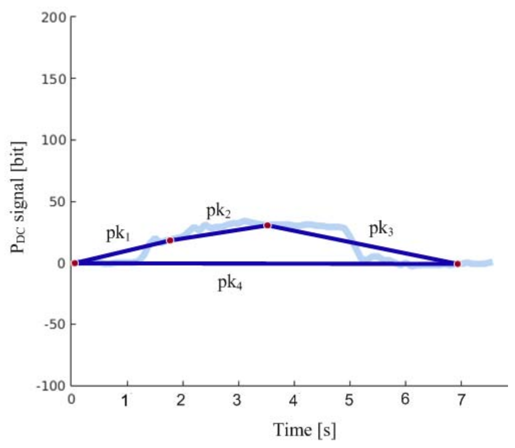
In more detail, in Fig. 4.10(b) and 4.10(d), $pk2$ is a very small negative value. More specifically, it happens in the repeatability experiment, such as in Fig. 4.13



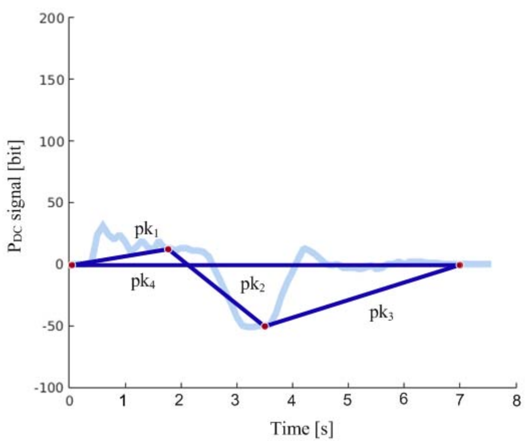
(a) $\mathbf{P_d} = (90, -30)$, $\mathbf{K_s} = (0, 0)$, $d = 4$



(b) $\mathbf{P_d} = (90, 0)$, $\mathbf{K_s} = (0, 0)$. $d = 6$



(c) $\mathbf{P_d} = (-30, -30)$, $\mathbf{K_s} = (0, 1)$. $d = 4$



(d) $\mathbf{P_d} = (-30, 0)$, $\mathbf{K_s} = (0, 1)$, $d = 4$

Figure 4.10: Haptic feature extraction.

in section 4.3 where the curve shows a valley in object's pitch manipulation. That is from the limitation of the robot joint (the thumb joint TF2). In that experiment, the thumb pushes on the object in given directions; however, sometimes the push actions are out of its workspace. In some manipulations, one thumb joint (TF2) stops after reaching its boundary; however other joints still move ahead. This unexpected constraint changes the push path away from the object. As a result, the valley appears due to the decline of the contact force. Fortunately, in these push directions, the manipulations are still completed successfully as long as the contact force is larger enough to keep stable contact. These valleys are caused by the limitation of robot joints; it can be detected by observing of the haptic features and be avoided via limiting these push directions. This is one of the advantages of our method that taking all object and fingers as one black box where we only focus on giving different commands and observing the visual and haptic feedback for successful manipulation tasks.

To summarize, the haptic feature is defined by [1]

$$\mathbf{pK} = \begin{bmatrix} pk_1 \\ pk_2 \\ pk_4 \end{bmatrix}. \tag{4.1}$$

The visual signals are provided by the visual tracking system that captures the current position and orientation of the object. The object's position is used to represent manipulation errors, which are not the expected motion in rotational manipulations. To denote the positional error, the relative translation of the object is recorded in manipulations. And the maximum translation distance is chosen as the positional error in this thesis. On the other hand, the object's attitude is used to evaluate the manipulation performance. According to Euler's rotation theorem, any displacement of a rigid body is equivalent to a single rotation. Hence the object's attitude is represented by a spatial vector, $\mathbf{V}_r \in \mathbb{R}^3$. Its length denotes object's rotated angle δ, and its direction refers to the axis of object's rotation. Similar to positional errors, the maximum rotational change is adopted as the visual feature $\mathbf{V}_r$ within the scope of this thesis.

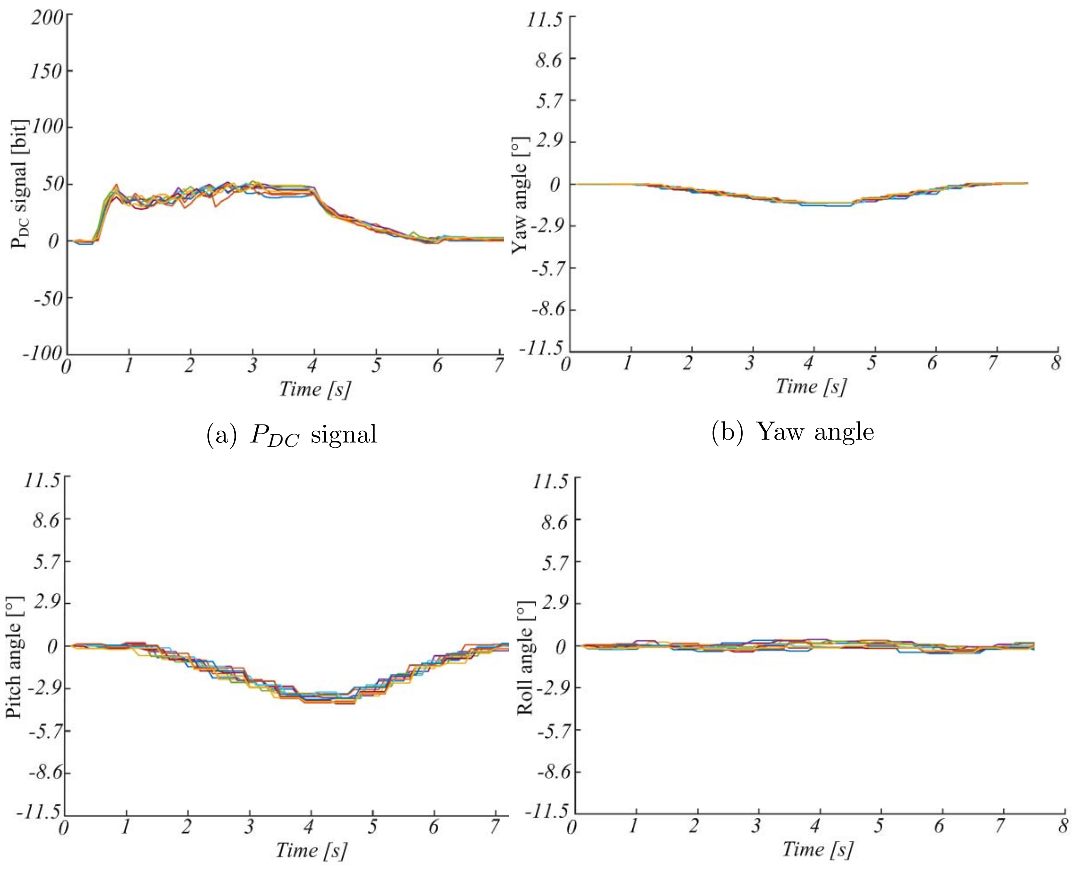
4.3 Repeatability Experiment

Before performing the 'haptic exploration', the first thing should be taken into account is the repeatability of the finger push actions. This experiment is to verify the questions: '*can the object move back to its initial state after each push?*' and '*can the feedback be repeated with the same push coefficient and grasping configuration?*'. Therefore, repeatability experiment is conducted in this section. In this experiment, we make the robot hand push 10 times in each given direction. After each push action, the push finger moves back automatically and waits for 2 seconds before the next push action. Each push and move back action takes 3 seconds and their speed is controlled with a polynomial motion planner. In this process, both visual and tactile information is collected to estimate the interaction state. In this section, pitch and yaw manipulations are carried out. As mentioned before, the roll manipulation is not discussed here, since it is inherently included in yaw manipulation experiments.

Fig. 4.11 and 4.12 represent the raw haptic and visual data. A pitch movement experiment is shown in Fig. 4.11, where push actions are given in direction $\mathbf{P}_d = (90,\ -15)$ [2] and with distance $P_l = 4mm$. The raw haptic feedback is represented by the P_{DC} signal from the BioTac sensor on the push finger. Fig. 4.11(a) shows the haptic feedback. The object's movement is recorded by the visual tracking system. Changes in object's yaw, pitch, and roll movements are shown in Fig. 4.11(b), 4.11(c), and 4.11(d) respectively. Similarly, a yaw manipulation experiment, where push actions are given in direction $\mathbf{P}_d = (-15,\ 0)$ and with distance $P_l = 8mm$, is shown in Fig. 4.12.

[1] In this research, in order to facilitate writing, pk_3 is used to represent pk_4 in $\mathbf{pK}$

[2] The default unit is degree[°] in $\mathbf{P}_d$.



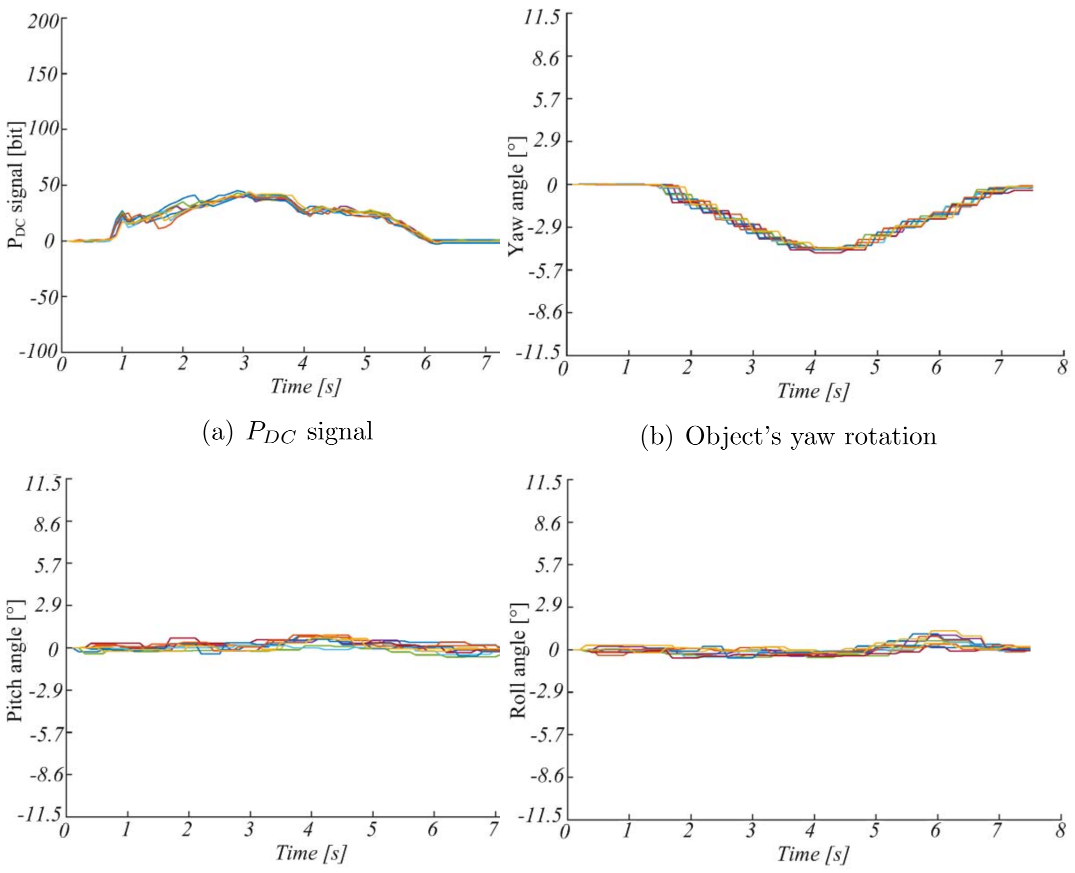
(a) P_{DC} signal (b) Yaw angle

(c) Pitch angle (d) Roll angle

Figure 4.11: The raw data of 10 manipulations with $P_l = 4mm$ and $\mathbf{P}_d = (90, -15)$

Besides, more experiments are carried out with four more different push directions. Due to the space constraints, only haptic P_{DC} signal result is given here, see Fig. 4.13 and 4.14. Manipulations represented in Fig. 4.13 concern object's pitch rotation; and the push directions are $\mathbf{P}_d = (90, 0)$ and $\mathbf{P}_d = (90, -30)$ with the distance $P_l = 8mm$. Equivalently, manipulations shown in Fig. 4.14 concern object's yaw rotation; and the push directions are $\mathbf{P}_d = (-30, 0)$ and $\mathbf{P}_d = (-30, -30)$ with the distance $P_l = 4mm$.

According to the results depicted in Fig. 4.11, 4.12, 4.13, and 4.14, it can be concluded: **in different push directions, shapes of visual and haptic signals are different; and when push actions are conducted with the same push parameters, the visual and haptic signals exhibit the same patten**. Therefore, this experiment supports the conclusion that our push manipulation method is feasible and repeatable.



(a) P_{DC} signal

(b) Object's yaw rotation

(c) Object's pitch rotation

(d) Object's roll rotation

Figure 4.12: The raw data of 10 manipulations with $P_l = 8mm$ and $\mathbf{P}_d = (-15, 0)$.

4.4 Multi-Push Distance Experiment

In order to investigate the relation between the push distance and results (haptic and visual features), multi-push distance experiment is conducted in this section.

For the yaw manipulation, the push finger pushes in one direction: $\mathbf{P}_d = (0,\ 0)$ but with 7 different push distances (P_l = 4 mm, 5 mm, 6 mm, 7 mm, 8 mm, 9 mm, 10 mm). In these manipulations, both haptic and visual signals are collected. After grasping, the object is pushed automatically with different distances (push once for each distance from P_l = 4 mm to P_l = 10 mm). After these push actions, the object is released from the hand. This sequence of actions, including grasping at the beginning, manipulating, and releasing at the end, is considered as one manipulation sequence. In this experiment, this manipulation sequence is repeated 3 times.

In order to visualize the relationship among the push distances and the visual and haptic features, a covariance diagram belongs to one manipulation sequence is shown in Fig. 4.15. Obviously, the push distance P_l has a strong linear relationship to the haptic feature pk_1 and the object rotation angle δ shown with red solid lines. pk_1 and P_l also have strongly linear relationship. Besides, as shown with a yellow

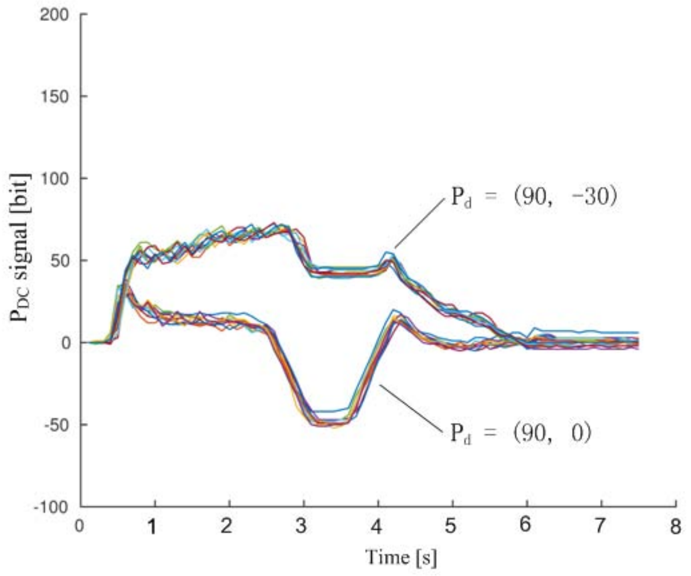


Figure 4.13: Repeatability of haptic raw data in the different direction $\mathbf{P_d} = (90, 0)$ *and* $(90, -30)$ with support stiffness $\mathbf{K_s} = (0, 0)$.

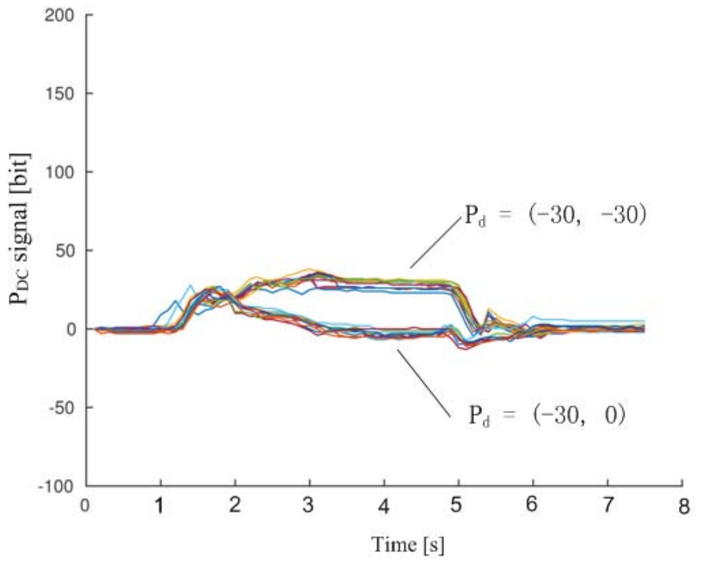


Figure 4.14: Repeatability of haptic raw data in the different direction $\mathbf{P_d} = (-30, 0)$ *and* $(-30, -30)$ with support stiffness $\mathbf{K_s} = (0, 1)$.

dash ellipse, the visual feature V_Z, the object rotated angle δ, and the push distance P_l have some weak linear relationship in a part of sections. As discussed in the visual feature section, this result reveals that the object's rotational axes are not strictly in axis V_z. They also have other components in axis V_x and V_y. It means that it is possible to take place of the roll manipulation with yaw manipulations. The object's pitch rotation in this yaw manipulation is considered as a noise since it

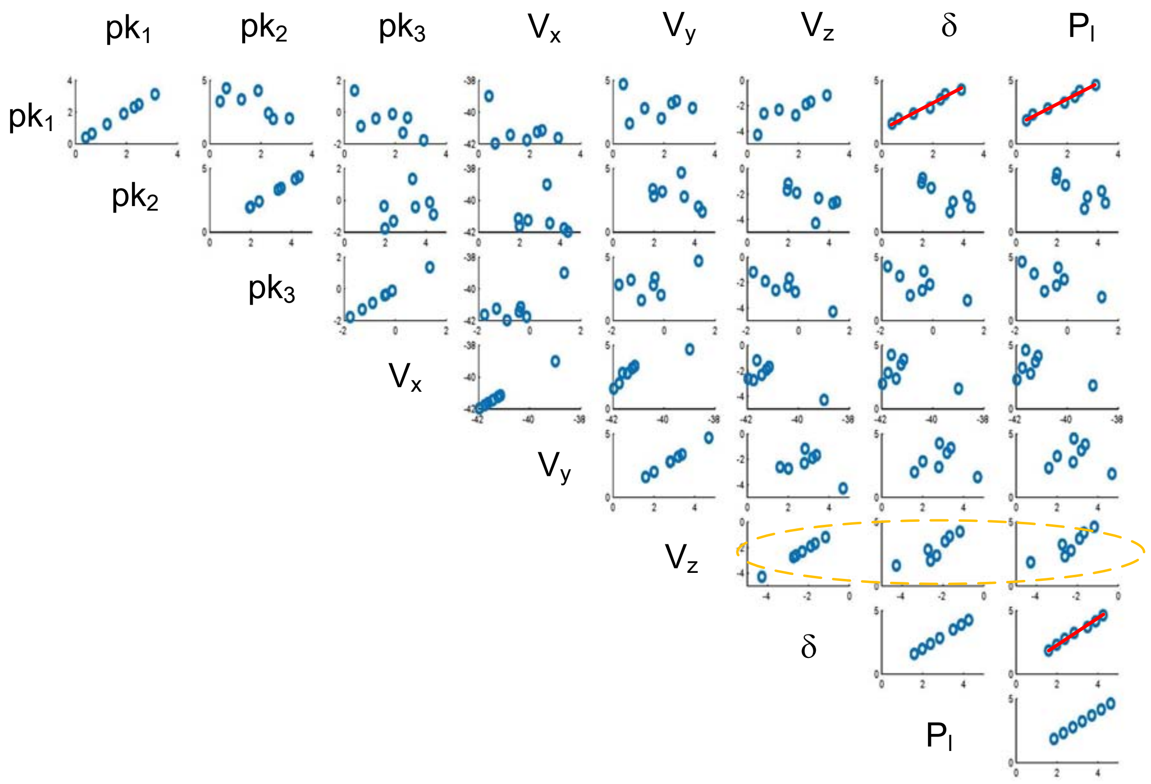


Figure 4.15: Haptic and visual feature covariance diagram.

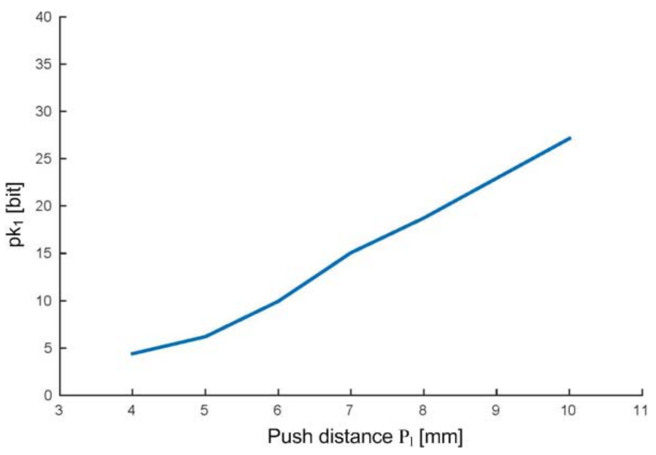


Figure 4.16: The relation between the finger's push distance and first haptic feature.

is unexpected movement. Results of the other two manipulation sequences exhibit similar pattern to the one presented in Fig. 4.15; therefore they are not presented. As a result, the push distance P_l can be represented linearly by haptic feature pk_1, and for the visual feature, it can be represented linearly by visual feature δ.

Fig. 4.16 and 4.17 represent the details of the linear performance, where the data is taken from the average of three manipulation groups. Fig. 4.16 shows

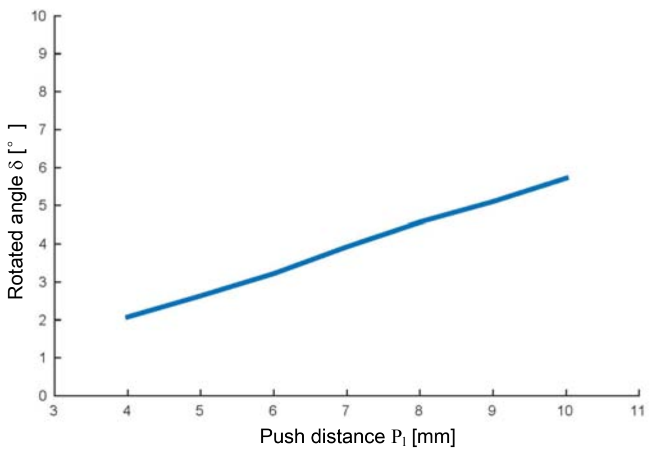


Figure 4.17: The relation between the finger's push distance and the object's rotated angle.

a linear relationship between the push distance P_l and the haptic feature pk_1. It starts at point (4, 4.4) and ends at point (10, 27.13) with a slope coefficient 3.79, resulting the line equation as $y = 3.79x - 10.75$. Fig. 4.17 shows a linear relationship between the push distance P_l and the visual feature δ. It starts at point (4, 2.08) and ends at point (10, 5.72) with slope coefficient 0.61. Hence, the line equation is $y = 0.61x - 0.36$. Furthermore, this linear relationship can be also extended to haptic and visual features. Because the features are a linear combination of their elements, and at least one of their elements linearly depends on the push distance. Therefore, there is a vector which maps the haptic and visual features to object's rotation angle in 2D movement. For example, for the haptic feature, this vector can be $\mathbf{Tr}_H = [3.79\ 0\ 0\ -10.75]$, which satisfies $\delta = \mathbf{Tr}_H[\mathbf{pk}^T,\ 1]^T$.

From the result of this manipulation experiment, two conclusions can be drawn. First, there are linear dependents between the push distance and at least one element of the haptic and visual features. In other words, it is possible to estimate part of the haptic and visual features in the manipulation with a large push distance, with the knowledge built up from manipulations with small push distances. Second, the haptic pk_1 and visual δ are also linearly dependent on each other with respect to the push distance. Although, they seem redundant; **the haptic features can be used to estimate the performance of a manipulation, especially in the manipulations with small push distance.**

4.5 Multi-Directional Manipulation Experiment

Besides push distance P_l, the manipulation performance also relates to the push direction $\mathbf{P}_d$. In this section, multi-directional manipulation experiments are carried out to illustrate the relation between the manipulation performance and the push directions. Equivalently, in this section, the experiments are presented in two aspects: pitch and yaw (or 'yaw and roll') manipulations.

In all manipulations, both haptic and visual signals are collected. For haptic reward R_H, the Euclidean distance is used to compare the haptic feature $\mathbf{pK}$ with an expected one $\mathbf{pK}'$ with

$$R_H = -d(\mathbf{pK}, \mathbf{pK}'), \tag{4.2}$$

where $d(\cdot)$ refers to the operator of the standard Euclidean distance, and term $\mathbf{pK}'$ denotes the expected haptic feature, which is given manually.

For visual signals, the projecting distance of the visual feature is used to represent the visual reward. In manipulations, the object's expected rotation direction is represented by a unit vector $\mathbf{V}_r'$. Hence, visual reward is from projecting the visual feature $\mathbf{V}_r$ to the expected rotational axis $\mathbf{V}_r'$

$$R_V = \mathbf{V}_r{}^T\mathbf{V}_r', \tag{4.3}$$

where the expected direction $\mathbf{V}_r'$ is a unit vector, and it is given according to the specific manipulation tasks. For example, if the manipulation is performed for a positive yaw rotation, $\mathbf{V}_r' = [0\ 0\ 1]^T$, and in our yaw manipulations, $\mathbf{V}_r' = [0\ 0\ -1]^T$.

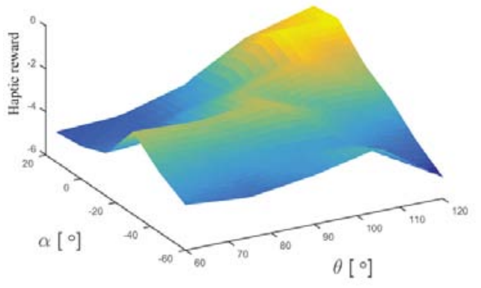
Pitch Manipulation

In pitch manipulations, the push directions are given by $\mathbf{P}_d = (\theta, \alpha)$, where θ changes from 60° to 120° with a step 15° and α changes from $-60°$ to 15° with a step 15°. In this experiment, the push distance $P_l = 4$ mm.

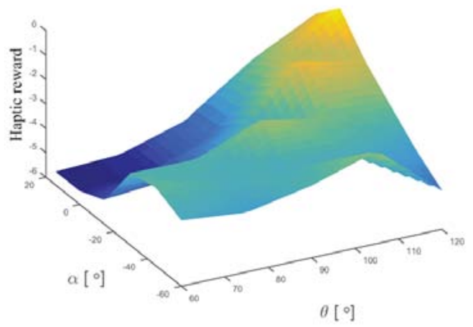
After a stable grasping, those push actions are carried out one by one automatically. The manipulations are also performed with push and back action pair: the push finger pushes forward with given direction and moves back to its initial state. This action pair repeats until all directions starting from $\mathbf{P}_d = (60, 15)$ and ending at $(120, -60)$ are completed. In total, 30 (5×6) push actions are carried out; and both haptic and visual features are extracted to estimate their performance.

Different haptic rewards obtained by choosing different expected haptic features $\mathbf{pK}'$ are shown in Fig.4.18. In this experiment, the haptic rewards in Fig. 4.18(c) is used where the expected haptic feature is given by $\mathbf{pK}' = [40\ \ -20\ 0]^T$ and the Euclidean weight is $\mathbf{w}_{Euc} = [40\ 20\ 10]^T$. Obviously, two bumps are clearly visible in Fig.4.18(c); one is located around position $(90, -30)$ and the other is located around the position $(105, 15)$. The local maximum position at $(90, -30)$ with a haptic reward -0.6. Although the result of haptic rewards depends on the expected feature $\mathbf{pK}'$ and the weights $\mathbf{w}_{Euc}$ which are chosen manually; the local maximum at position $(90, -30)$ is not sensitive to the chosen parameters.

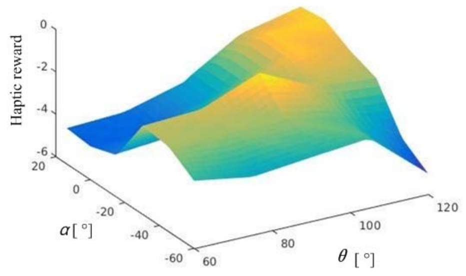
In this experiment, expected haptic feature $\mathbf{pK}' = [40\ \ -20\ 0]'$ is chosen with practical reasons based on the discussion in section 4.2.4. The first element of $\mathbf{pK}'$ indicates the initial rising slope in the early state of the push action; hence it is set as the average value of pk_1 in experiments. The second element of $\mathbf{pK}'$ denotes the change of push force in the latter stage of the push action. The contact force should be smaller after the object is rolled. Hence, a negative value is chosen.



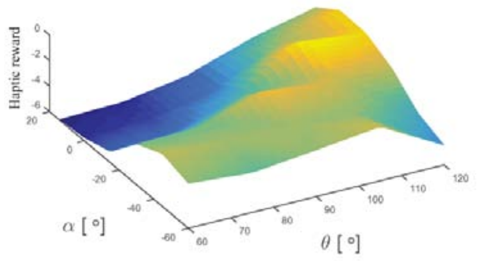
(a) $pK' = (20\ \ -20\ 0)^T$.



(b) $\mathbf{pK}' = (20\ 0\ 0)^T$.



(c) $\mathbf{pK}' = (40\ \ -20\ 0)^T$.



(d) $\mathbf{pK}' = (40\ 0\ 0)^T$.

Figure 4.18: Haptic reward with different $\mathbf{pK}'$.

It is a negative standard Euclidean distance to a desired haptic feature pK'. In these diagrams, the horizontal axes denote to different push directions where $\theta = 60°$ to $120°$ with the step of $15°$ and $\alpha = -60°$ to $15°$ with the step of $15°$. The different haptic rewards are listed in the sub figures according to the different expected haptic feature $\mathbf{pK}'$. Although $\mathbf{pK}'$ is given by different values, there is always a local maximum around the point $(90°, -30°)$ in all sub figures.

Besides, the last term $\mathbf{pK}'_3$ refers to whether any slip occurs in the manipulations. If it is large, it means a slip occurred and the object can not be moved back to its initial state completely. If it is small, it means there is no significant slip in this manipulation. As a result, it is set to 0.

Obviously, in Fig. 4.18(a), 4.18(b), 4.18(c), and 4.18(d), there are more than one bumps (local maximum) in each sub-figures; however at least a bump is located around position $(90, -30)$ in each sub-figure. This indicates that these haptic rewards are robust to the chosen parameters. The different expected features generate a similar result with a local maximum haptic reward around position $(90 - 30)$. Hence, any of the sub figures in Fig. 4.18 can be used as the haptic reward. As it can be seen from Fig. 4.18(c), this haptic reward can only show the best push actions locally. However, it seems not a good idea to simply identify the best action with a local maximum. Therefore, to obtain the best global action, visual rewards are taken into account.

Fig.4.19 illustrates the result of the visual reward. In this experiment, the

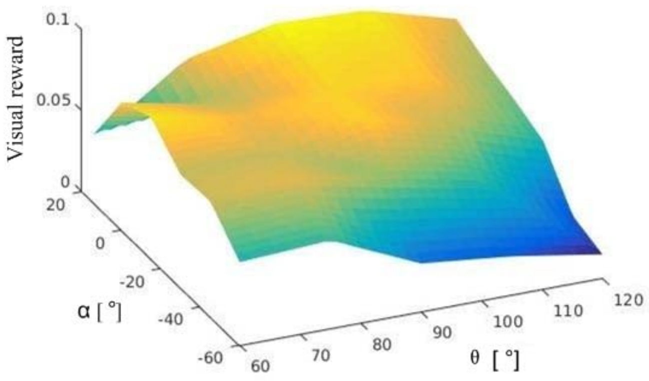


Figure 4.19: Visual reward.

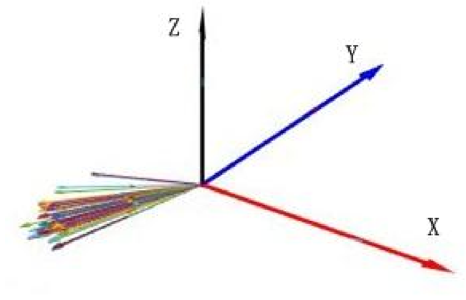


Figure 4.20: Object's rotation axes.

object's expected rotation direction is given by $\mathbf{V}'_r = [0\ \ -1\ \ 0]^T$. As shown in Fig.4.19, there is a smooth plateau around the point $(90, -30)$. In this figure, the maximum point is not clearly visible; and the good push direction is located in the yellow large area from the view of visual rewards. Besides, the object's rotated directions are plotted in Fig. 4.20 with arrows. In the figure, the rotated directions cluster in the direction $-Y$ within a small cone in a palm frame; although the push directions are given quite differently. **Hence, it can be concluded that the object's rotated directions are not sensitive to the push directions in the pitch manipulations.**

An intuitive way to choose the best push direction is combining the haptic and visual rewards together. Hence, we have

$$R_{total} = w_H R_H + w_V R_V, \tag{4.4}$$

where w_H and w_V denote the weights for haptic and visual rewards respectively. The combined haptic and visual reward is shown in Fig. 4.21 with the weights $w_H = 1$ and $w_V = 0.5$. These weights are selected because in this pitch manipulation the object' rotations are not sensitive to the push directions. Therefore the

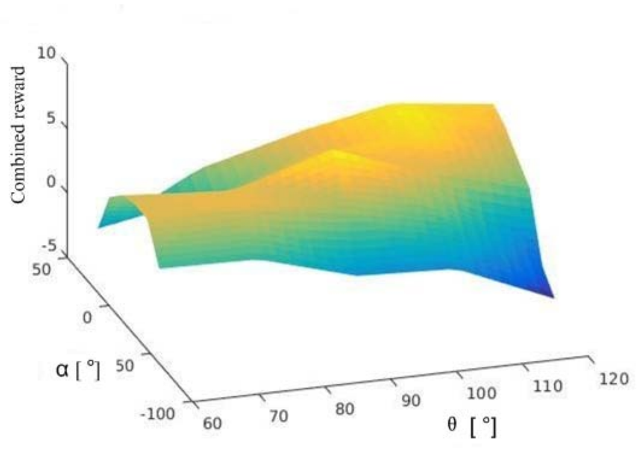


Figure 4.21: Combined haptic and visual reward in pitch manipulations.

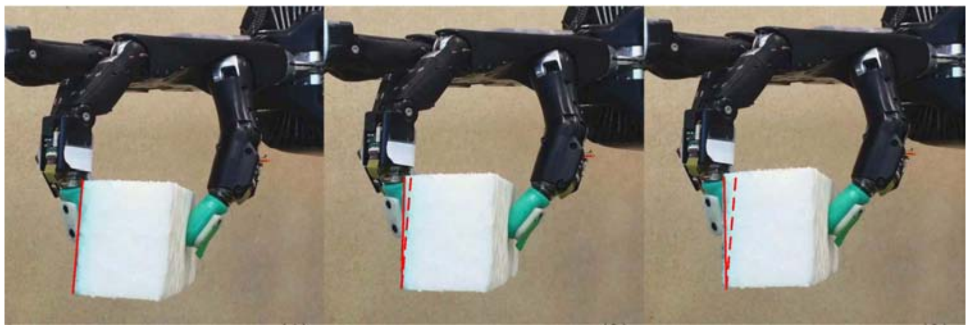


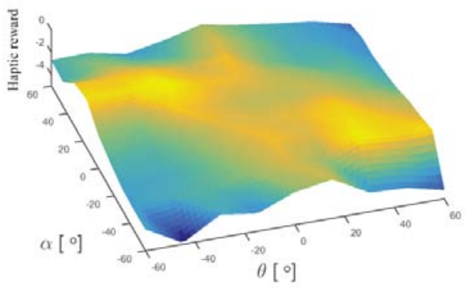
Figure 4.22: The snapshots of a pitch manipulation.

visual weight is given with a small value. The combined reward is shown in Fig. 4.21. In Fig. 4.21, there is a sharper bump compared to the visual reward represented in Fig. 4.19. However, no significant improvement is observed compared to the haptic reward given in Fig. 4.18. **As a result, in the pitch manipulations, it is better to focus on haptic (or haptic dominated) reward to evaluate the manipulation performance, since the pitch manipulation is robust to the push directional parameters.** The best push direction in this experiment is $\mathbf{P}_d = (90, -30)$. A pitch manipulation is carried out on the real robot platform with the best push direction, and the snapshots are shown in Fig. 4.22.

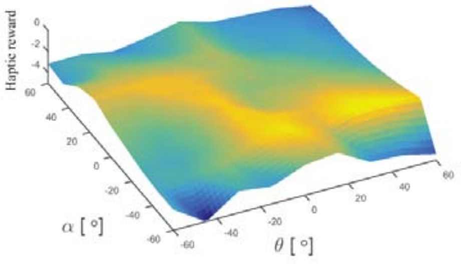
Yaw and Roll Manipulation

In the yaw manipulations, the push directions are given by $\mathbf{P}_d = (\theta, \alpha)$, where θ changes from $-60°$ to $60°$ with a step $15°$ and α changes from $-60°$ to $30°$ with a step $15°$. In this experiment, the push distance $P_l = 8$ mm.

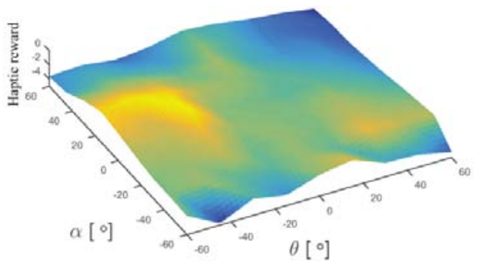
After the object is grasped, these push actions are performed one by one automatically. The manipulations are also conducted with push and back action pairs as in the experiments given before. It also repeats manipulating until all directions, starting from $\mathbf{P}_d = (-60, -60)$ and ending at $(60, 30)$, are completed. In



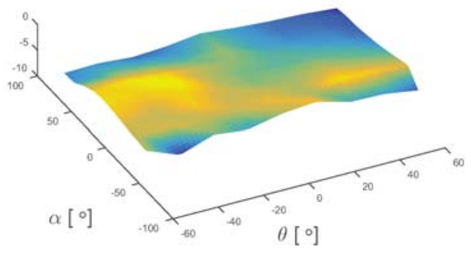
(a) $\mathbf{pK}' = (20\ 10\ 0)^T$



(b) $\mathbf{pK}' = (20\ 30\ 0)^T$



(c) $\mathbf{pK}' = (40\ 10\ 0)^T$



(d) $\mathbf{pK}' = (40\ 30\ 0)^T$

Figure 4.23: Haptic reward with different $\mathbf{pK}'$.

total, 63 (9×7) push actions are conducted, and both haptic and visual features are extracted to estimate the performance of the push actions.

Similar to 'Pitch Manipulation', different haptic rewards are obtained based on different expected haptic features $\mathbf{pK}'$ in yaw manipulations demonstrated in Fig. 4.23. In this experiment, the haptic reward in Fig.4.23(a) is adopted with the expected haptic feature $\mathbf{pK}' = [20\ 10\ 0]^T$ and the Euclidean weight $\mathbf{w}_{Euc} = [20\ 20\ 10]^T$. Two bumps can be seen in Fig.4.23(a); one is located around the point $(-30, 15)$ with the value of -0.51 and the other is located around the point $(30, 45)$ with the value of -0.62. Similarly, the expected haptic feature $\mathbf{pK}' = [20\ 10\ 0]^T$ is chosen due to practical reasons. The first element of $\mathbf{pK}'$ is chosen from the average of pk_1 value in this experiment. The second element of $\mathbf{pK}'$ is set to a small positive value. The push force increases since the contact force applied by the spring support finger increases when rotating. Hence, a small value 10 is chosen. And the last term of $\mathbf{pK}'$ is set to 0 to reduce the slip as in pitch manipulations.

Similarly, there are more than one bumps in each sub figures in Fig. 4.23. However, there is a local maximum point around the point $(-30, 15)$ in Fig. 4.23(a), 4.23(b), 4.23(c), and 4.23(d). This supports that these haptic rewards are not sensitive to the manually chosen parameters. It can be seen in Fig. 4.23(a): for the yaw manipulation, this haptic reward result can not show the best push action. Therefore, the visual rewards should be considered.

The visual reward is shown in Fig. 4.24. In this experiment, the object's expected rotation is given by $\mathbf{V}_r' = [0\ 0\ -1]^T$. There is one smooth bump around the position $(-30, 15)$ with the value of 0.086. Obviously, it is the global maximum

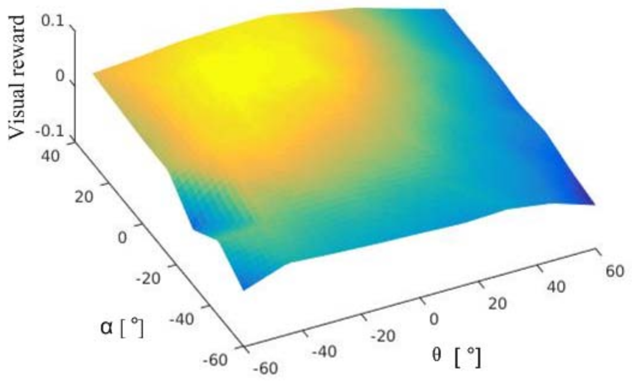


Figure 4.24: Visual reward.

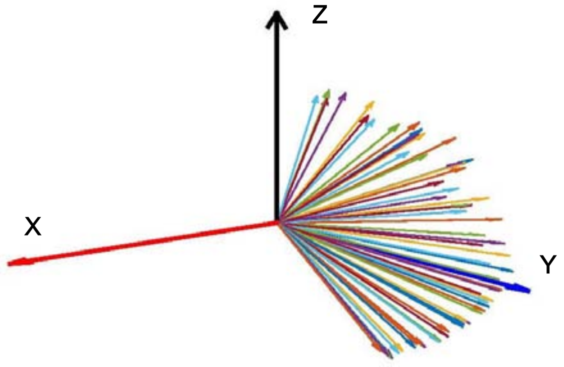


Figure 4.25: Object's rotated axes.

value. Therefore, the best visual-based push direction can be considered as at the top position of the bump $(-30°, 15°)$. Unlike the pitch manipulation, the object's rotation directions are sensitive to the push directions in yaw manipulations. All the object's rotated directions are plotted in Fig. 4.25, where the short arrows refer to the object's rotation axes. Obviously, these arrows are evenly distributed in a plane in the palm frame.

In this task, both haptic and visual rewards reflex the manipulation performance; and they have their own strengths. The haptic reward shows the best point with small bumps; however, there are two bumps in Fig. 4.23. Furthermore, the visual reward shows a single bump in Fig. 4.24, but it is much smoother than the bumps in the haptic reward.

Similarly, referring to Eq. 4.4, the haptic and visual combined reward is plotted in Fig. 4.26 with the weights $w_H = 1$ and $w_V = 20$. In Fig. 4.26, there is a single sharper bump compared to the visual reward presented in Fig. 4.24. As a result, the best push direction is $\mathbf{P}_d = (-30, 15)$. A manipulation with the best push direction is carried out on the real robot platform. The snapshots of its results are

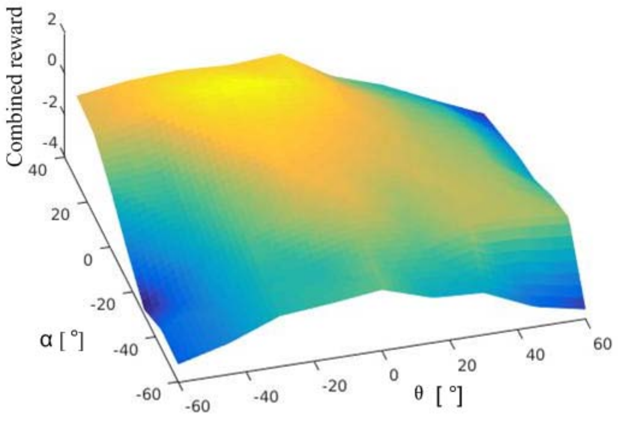


Figure 4.26: The combined haptic and visual reward in yaw manipulations.

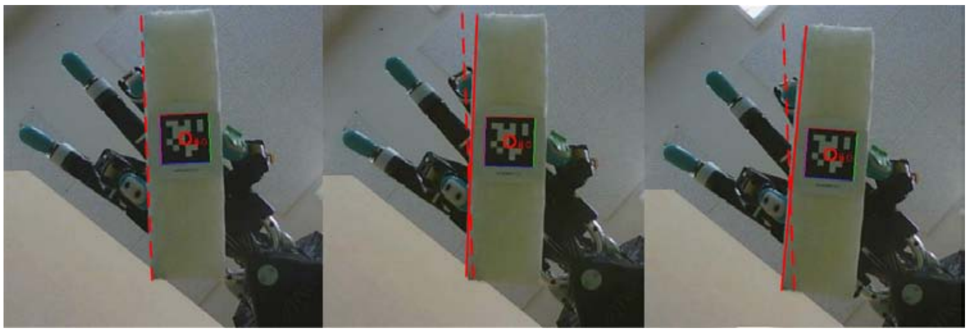


Figure 4.27: The snapshots of the yaw manipulation.

given in Fig. 4.27, where the object rotates in 6.8°.

Relation between Haptic and Visual Features

The relation among haptic, visual features and the push directions are shown in Fig. 4.28 and 4.29. Additionally, one more element, which is the sum of the first and second elements of the haptic feature: $pk_1 + pk_2$, is added in Fig. 4.28 and 4.29.

For the pitch manipulation, there are three linear relations worth noting in Fig. 4.28. They are $(pk_1 + pk_2)$-V_Z, $(pk_1 + pk_2)$-δ, and $(pk_1 + pk_2)$-α demonstrated with three red lines in Fig. 4.28. Although their linear dependencies are not strong; some useful conclusions can be drawn from them. For $(pk_1 + pk_2)$-α, the sum of the first and second elements of the haptic feature can be controlled easily with the second element of the push direction α. Therefore, a linear relation is assumed $(pk_1 + pk_2) \propto \alpha + b$, where b is a constant baseline. For $(pk_1 + pk_2)$-δ, the expected object rotation decreases when the haptic value $pk_1 + pk_2$ increases. This suggests that the finger should push in the directions with small haptic value $pk_1 + pk_2$. Besides, this figure also highlights other interesting relations, i.e. the

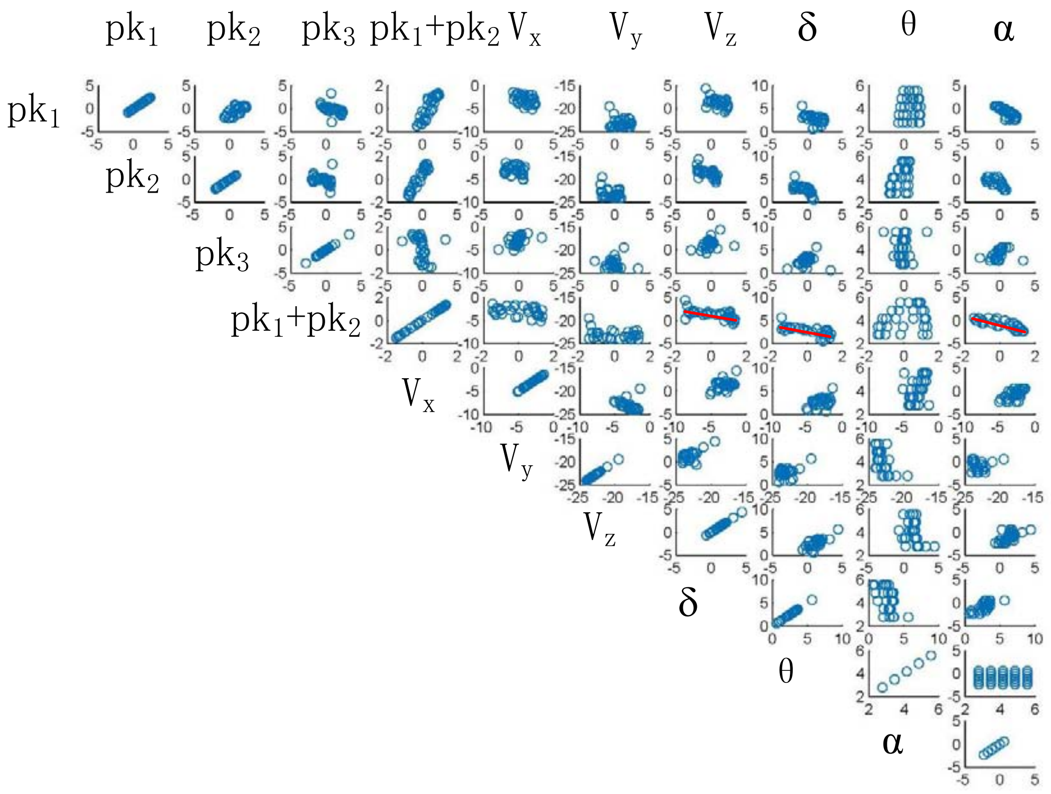


Figure 4.28: Haptic and visual covariant diagram for the pitch manipulation.

sub-diagrams in the columns θ and α. The points in this sub-diagrams relating to θ are much scattered; on the contrary, the points relating to α are concentrated on one particular area. It means that the push direction parameter α has a much stronger effect on the manipulation performance (visual and haptic features) than θ.

For the yaw manipulation, the relationship among the parameters is more regular, as can be seen in Fig. 4.29. One of the most important strong linear relation is V_Z-θ. **As V_Z is the expected rotation in yaw manipulations, it denotes the object's expected rotational direction (V_Z) is linearly proportional to one of the push parameter (θ).** Therefore, the best push direction can be found efficiently by focusing more on θ. Besides, there are also some other interesting relations can be also seen in Fig. 4.29. The sub-diagram $(pk_1 + pk_2)$-pk_1, where the data shape is very similar to a line. It denotes that its second element of the haptic features is less important compared to the first element.

4.6 Hybrid Support Manipulation Experiments

In order to verify the hybrid support model, a hybrid support manipulation experiment is carried out. In this experiment, only one push direction parameter α is considered, due to it is a 2D manipulation. The object is assumed to rotate in a

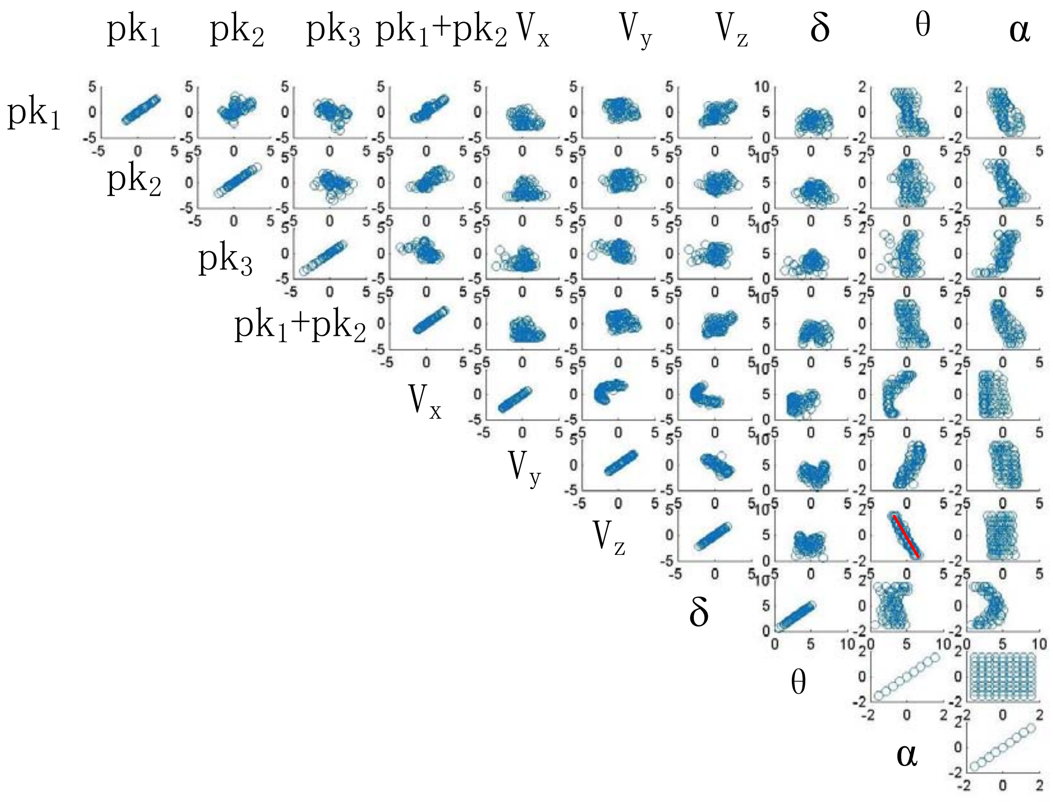


Figure 4.29: Haptic and visual covariant diagram for the yaw manipulation.

horizontal plane. This can be achievable by fixing the other push direction parameter, θ, to $-30°$ according to the experimental results (the best push direction) in section 4.5.

This experiment consists of two sub yaw manipulation experiments. The first one is the push direction experiment, designed to verify the relation between the push direction and the object's rotation. The second one is the push distance experiment. It is conducted to verify the relationship between the push distance and object's rotation. Besides, the stiffness control performance on the spring support finger is also investigated in the push distance experiment.

In this experiment, the push direction is given by $\mathbf{P}_d = (-30,\ 15)$; and the push length P_l changes in the range of $4mm$ to $10mm$ with the step of $1mm$. And each push action is performed three times to reach a stable average value. For the spring support finger, a stiffness control is applied with $50Hz$ frequency and the finger stiffness is set to 6.76.[3] The contact force is represented by the maximum value of the haptic feature ($max(pk_1, pk_2, pk_14)$), which is the P_{DC} signal read from the sensors.

[3]The stiffness is set to this value in all manipulation experiments in this thesis.

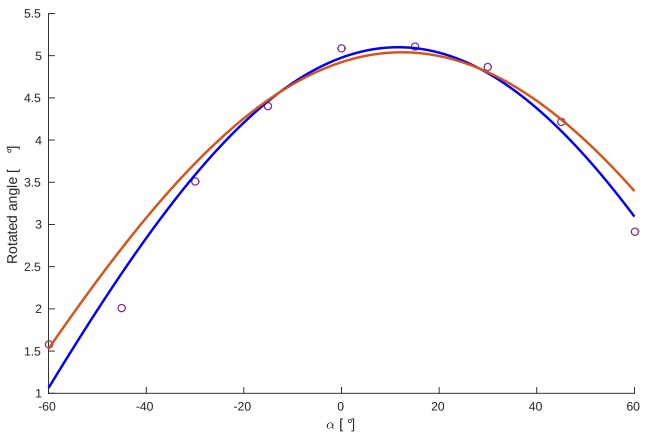


Figure 4.30: The manipulation results based on different push directions with $\mathbf{P}_d = (-30, \alpha)$.

4.6.1 Push Direction

In Push Direction Experiment, the push direction α changes from $-60°$ to $60°$, and the push distance is set to $P_d = 8mm$. In order to facilitate the experiments, the robot moves back to its initial grasping state after each push action. Hence, all the nine manipulations are finished automatically. After that, the object is released from the hand. Besides, these manipulation processes are conducted three times and their average value is represented in Fig. 4.30, where the violet circles are experimental data points. The red line is a regression result with a function of $y = 5.1cos(x - 12.6)$, where $\varphi = -12.6°$. This result fits well with the Eq. 3.49. Differently from Eq. 3.49, there is a phase angle $-12.6°$ in this experiment result. That is because on the real robot contact positions on the fingertips are different from the assumed position in the theoretical model. This indicates the feasibility of our proposed hybrid support model.

Therefore, it can be concluded that the results of the Push Direction Experiment verify our proposed hybrid support finger model, proposed in section 3.4. Furthermore, according to the visual result, the best push direction is 12.6°. This result provides us an optimized push direction for the further manipulations.

4.6.2 Push Distance and Contact Force

Fig. 4.31 illustrates the contact force applied by the different fingers. In this figure, the blue line denotes the contact force on the thumb; the green line refers to the contact force on the index finger, and the red line represents the contact force applied by the ring finger. Obviously, the contact force on the thumb approximates to the sum of the contact force on the index and ring fingers. That is because, in this grasping configuration, the thumb is on the opposite side of other two fingers. According to the force-closure condition, the sum of the contact force should be zero. However, the BioTac sensors can not identify the force direction. Thus the

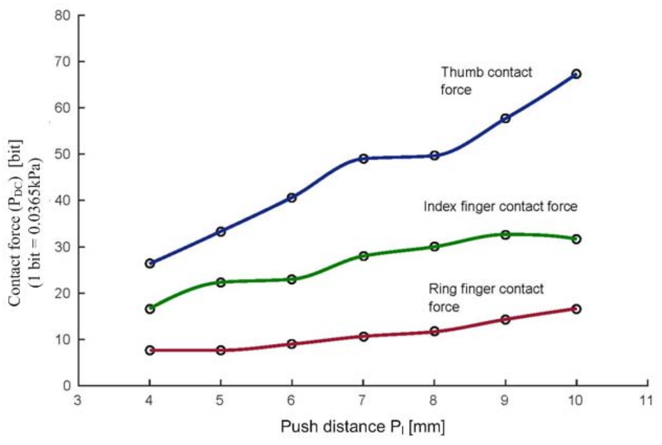


Figure 4.31: The contact force on different fingers.

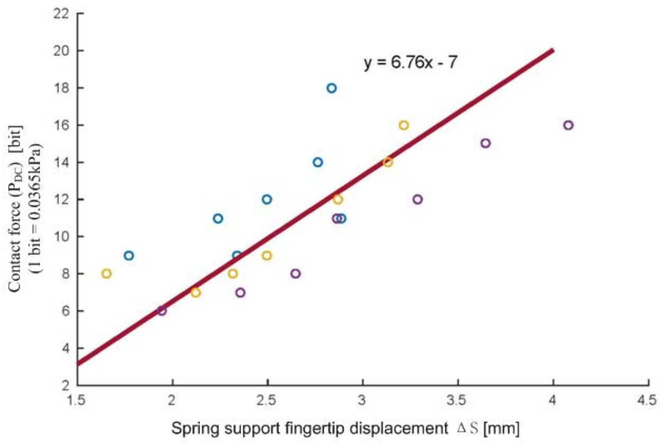


Figure 4.32: The stiffness performance of the ring finger.

contact force in Fig. 4.31 is all positive scalar, and the contact force on the thumb approximates to the sum of the other fingers.

The stiffness control performance of the spring support finger is shown in Fig. 4.32. The dots in this figure represent the relation between the contact force (represented by P_{DC} value) and the finger's displacement; and the red solid line denotes the expected stiffness with the equation of $y = 6.76x - 7$, where 6.76 is the finger stiffness and 7 is a baseline for compensating the initial grasping force and fingertip position. It is worth noting that raw data is a little away from the expected stiffness line. It mainly lies in two aspects. The first aspect is about the low control resolution of the robot hand which is 1° for each joint. The finger does not move when the control commands are given to change the joints in a small angle (less than the control resolution). The second aspect is about the structure compliance. The shadow hand is driven with tendons, even without the stiffness controller the joints still have some compliance attributes that make them deformable to the external force. Therefore, the fingertip moves when either the

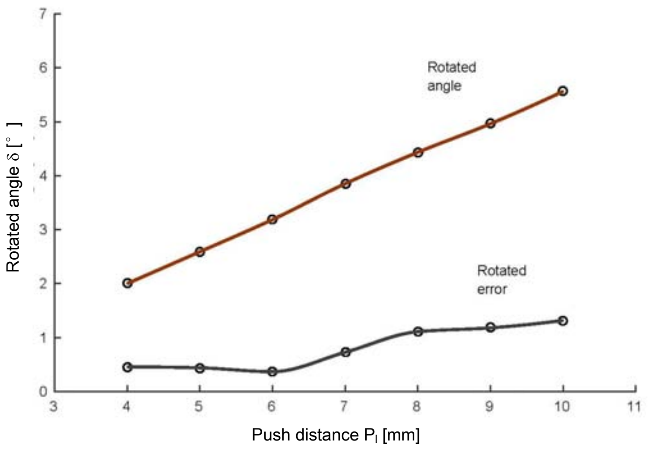


Figure 4.33: Change in object' attitude.

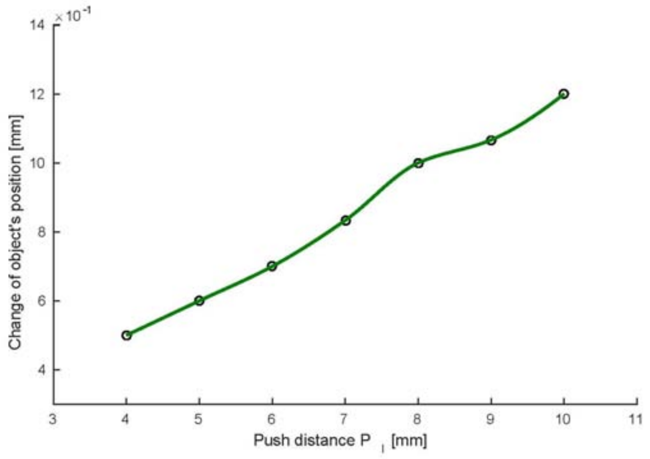


Figure 4.34: Change of the object's position.

direction or the contact force changes.

Manipulation Errors

The object's attitude provided by the visual tracking system is plotted in Fig. 4.33. In this figure, the red line denotes the rotation angle in a horizontal plane; and the gray line corresponds to the rotational errors. It is easy to find that the rotation angle is almost a line that refers to the push distance, which shows a linear relationship between the push distance and the object's rotation angle. On the other side, the object's rotational error is much smaller compared to the rotation angle. Unlike the rotation angle, the error shows no signification linear relation. Actually, within a small range ($P_l \leqslant 6mm$), it changes quite slowly.

Besides, the change in object's attitude, this manipulation method also generates positional errors. In Fig. 4.34, the change in the object's position is plotted. Obviously, there is also a linear relationship between the push distance and the

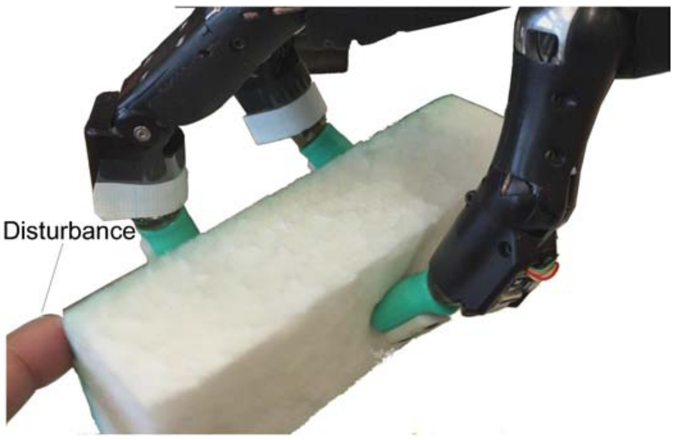


Figure 4.35: A disturbance during an object's grasp.

change in object's position. However, its maximum value of the change in position is 1.2 mm. Comparing this value to the change in object's rotation, it is a very small value. And due to the low control resolution of finger joints, it is almost impossible for our robot to move the object to compensate such small positional errors. Therefore, it can be concluded that it is practical to ignore the position change in this manipulation method. Besides the purpose of this manipulation is not to rotate the object but to perceive the interaction state of the in-hand system. The errors will be compensated in the enhanced manipulation experiments which are performed in section 4.8.

Short Discussion

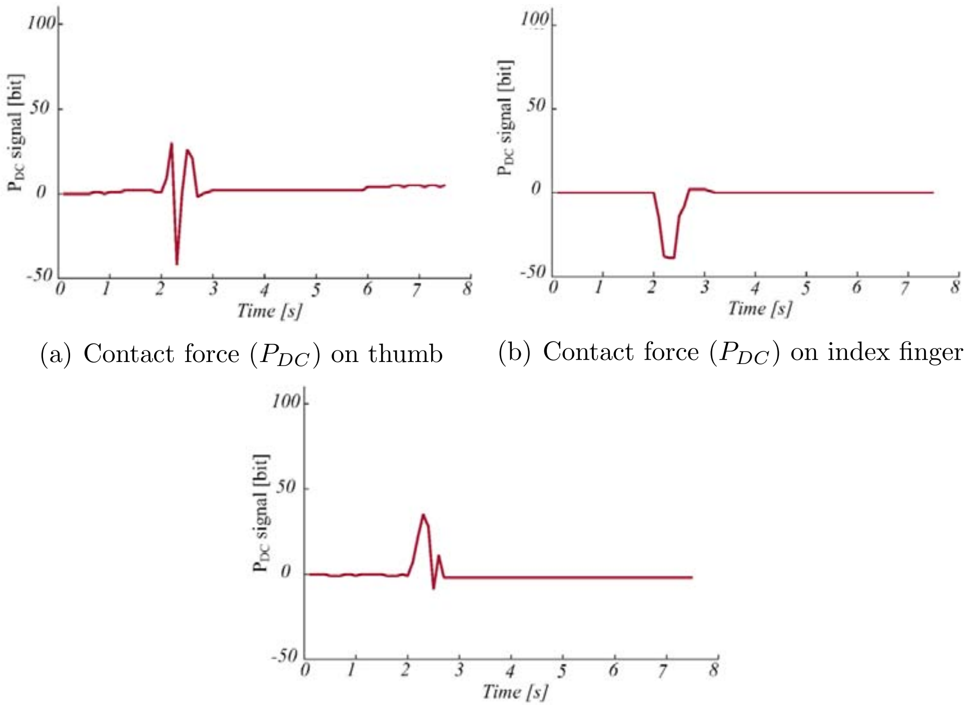
Although the low-level controller on the shadow hand is PD controller, the push actions are still limited by robot's structure. In these experiments, all these push actions are controlled off-line and the resolution of each joint is set to 1°. However, in haptic exploration, both the limitation of the push finger's structure and the control resolution is a part of the environment for the robot.

4.7 Robustness Experiments

Compared to the other manipulation methods, our method can not only reduce the complexity of the system but also enhance the robustness of grasping and manipulation abilities dramatically.

The stiffness-control strategy on the spring support finger, discussed in section 3.4.3, benefits not only the stability of the grasp but also the stability of the manipulations. Hence, in this section two robustness experiments are presented. The first one is grasping robustness experiment, where a disturbance occurs when the object is stably grasped. In this grasping state, only the spring support finger (ring finger) is controlled passively to compensate the errors generated by an external disturbance. The second experiment is manipulation robustness experiment, where the disturbance occurs in the process of manipulation. This manipulation is the same as in section 4.6. In both experiments, the disturbance is a quick poke on one side of the object given by a human finger, as shown in Fig. 4.35.

Robustness grasping experiment

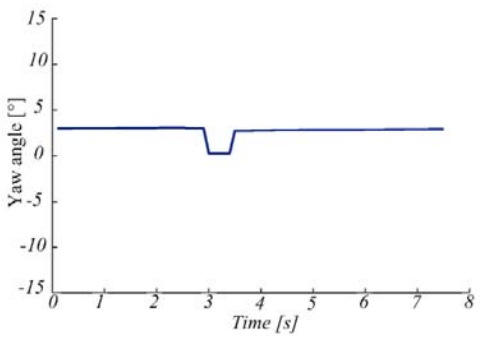


(a) Contact force (P_{DC}) on thumb (b) Contact force (P_{DC}) on index finger

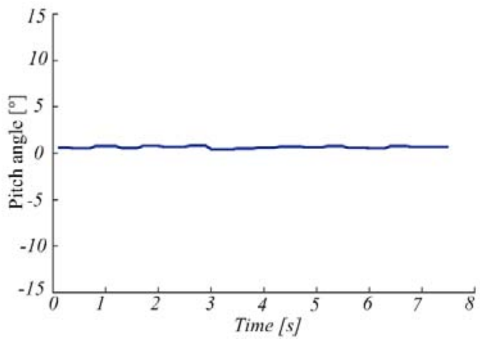
(c) Contact force (P_{DC}) on ring finger

Figure 4.36: The haptic result of a robust grasp when a disturbance occurs.

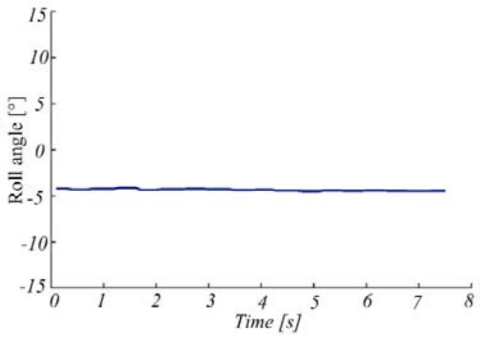
The result of grasping robustness experiment is shown in Fig. 4.36 and 4.37. Fig. 4.36 illustrates the haptic result; and Fig. 4.37 shows the visual result in this experiment. The disturbance is given at the $2s$. In Fig. 4.36(a), first the thumb P_{DC} signal increases and then it drops immediately. That is due to the action delay of the spring support finger. After the disturbance is given, the contact force on the spring support finger increases. However, because of the low resolution of the joints, the spring support finger does not move until its positional error is larger than 1°. Therefore, the contact force on thumb increases first for the disturbance, and then drops for the compliance actions from the spring support finger. This is also demonstrated in Fig. 4.36(c), where contact force on the ring finger increases for the disturbance at the beginning, and then drops back immediately for its compliance actions. Fig. 4.36(b) shows the P_{DC} signal from the index finger. In this stable grasp, the index finger is not controlled, and the disturbance makes the object away from the index fingertip. Therefore, the contact force drops dramatically until the object departs from the fingertip (no contact anymore). And after the disturbance disappeared, the haptic signals return back.



(a) Yaw angle



(b) Pitch angle



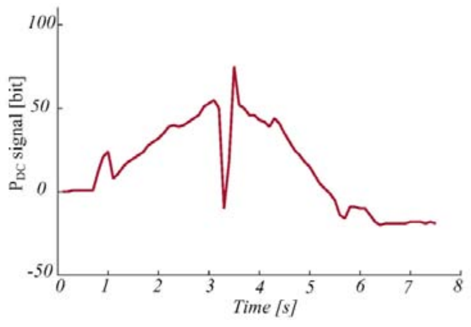
(c) Roll angle

Figure 4.37: The visual result of a robust grasp when a disturbance occurs. (object's attitude)

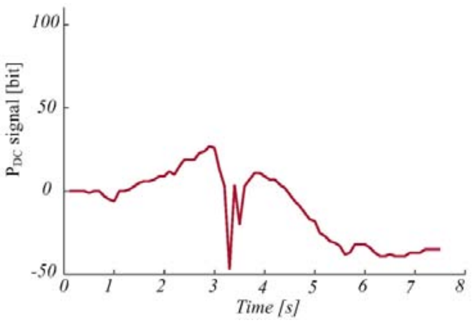
As shown in the visual result, the disturbance generates a negative yaw rotation on the object, as shown in Fig. 4.37(a); and after it disappears, the object moves back to its initial state. Two things should be noted here. The first one is that the change in the object's position is ignored here since it is really small according to the result in Fig. 4.34. The other thing is that the update frequency of the visual signals is not as high as haptic signals, since detecting and tracking tags takes time for the visual system. Hence, the disturbance signal has a flat bottom in Fig. 4.37(a); and there is a little delay compared to the haptic signals.

Robustness manipulation experiment

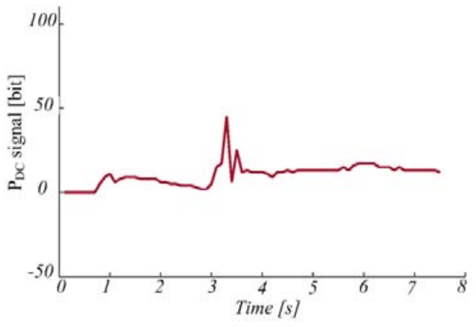
Similarly, Fig. 4.38 and 4.39 present the result of the robustness manipulation experiment, where a disturbance is given in the process of manipulation. Similar to the first robustness grasping experiment, the disturbance is given by poking at the object with a human finger. Fig. 4.38 and Fig. 4.39 show the haptic result and the visual result respectively. The manipulation is conducted with same push parameters ($\mathbf{P_d} = (-30,\ 15)$ and $P_l = 8mm$) as explained in section 4.6. Fig. 4.38(a), 4.38(b), and 4.38(c) show the P_{DC} signals from the push finger (thumb), the fix support finger (index finger), and the spring support finger (ring finger) respectively. Obviously, at the beginning, the signal increases normally due to



(a) Contact force (P_{DC}) on thumb



(b) Contact force (P_{DC}) on index finger



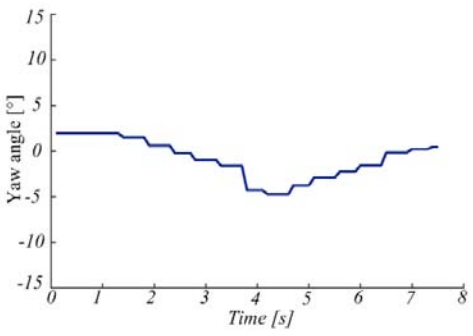
(c) Contact force (P_{DC}) on ring finger

Figure 4.38: The haptic result of a robust manipulation.

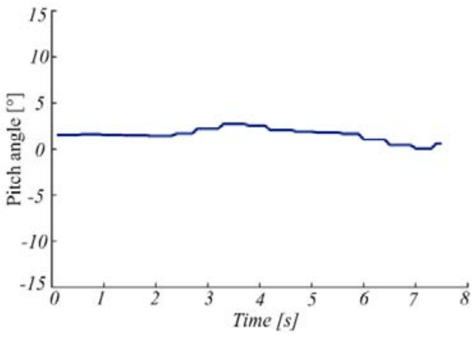
the push action; and then spikes occur at the $3s$ due to the disturbance; finally, after the disturbance disappears the system goes back to a stable state and the contact signals decrease with respect to the rest manipulation actions. However, the contact force on push and fix support fingers can not move back to their initial state. This is because the external disturbance generates slips on the push and fixed support fingers, which cause the system can not move back to its initial state. On the contrary, there are few slips on the spring support finger for its spring-like attributes. For the visual result, the disturbance generates a fast decrease on the yaw rotation at the time 3.8s; and there is no distinct effect on other rotation directions. According to the visual result, the disturbance generates a negative yaw rotation on the object, as shown in Fig. 4.39. After the disturbance disappears, the object moves back to its initial state.

Additionally, hundreds of manipulations have been performed, after the manipulation algorithm is completed. Except few experiments conducted in dangerous push directions, almost all manipulations are completed successfully.[4] In other words, if a proper push direction is given, the successful ratio of our manipulation methods is almost 100%.

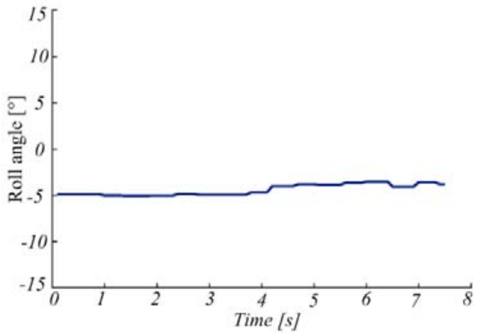
[4]Here the success of the manipulation denotes that the object does not fall off the robot hand.



(a) Yaw angle



(b) Pitch angle



(c) Roll angle

Figure 4.39: The visual result of a robust manipulation. (object's orientation)

4.8 Enhanced Manipulation Experiments

In above experiments, the best push direction is found in the haptic exploration. With the push direction, the object is rotated by a small angle (about 6.8°). To apply this 'best push direction' and improve the manipulation performance, enhanced manipulation experiments are carried out. In this section conducted manipulations are based on the enhanced manipulation model introduced in section 3.5. In following experiments, the manipulations are performed on a new platform where the shadow hand is mounted on the PR-2 robot as its right hand, as illustrated in Fig. 4.40.

The snapshots of an enhanced manipulation are shown in Fig. 4.41. In these snapshots, the object rotates about 18°. Compared with the push manipulation in Fig. 4.27 (rotated by 6.8°), the yaw rotation is significantly improved by the enhanced manipulation method. On the other hand, the enhanced performance not only increases the rotational range of the object but also enable the robot to manipulate various objects.

4.8.1 Manipulating a Rigid Object

In above sections, the manipulated object is the foam box which is soft and light. In fact, our manipulation method is able to manipulate rigid objects. As shown in Fig. 4.42, a rigid box is used in this manipulation experiment. This object has the

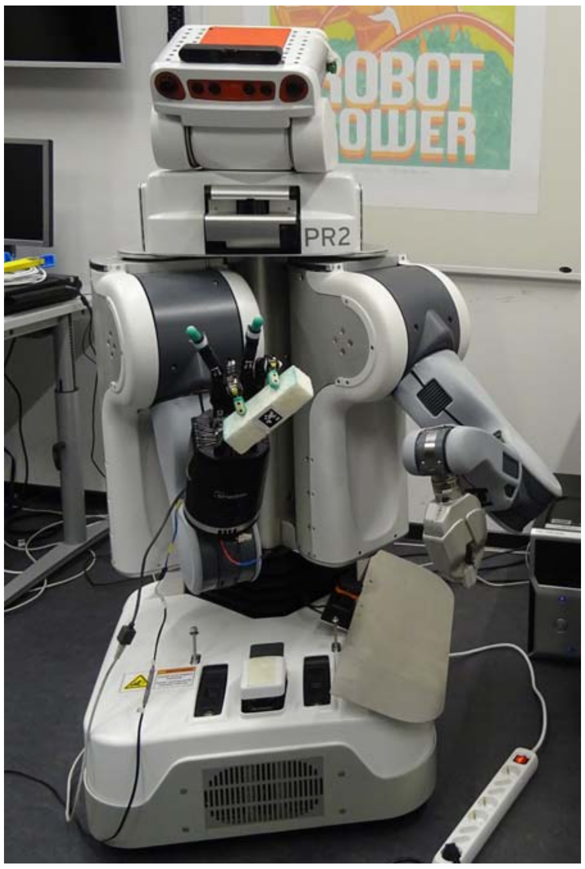


Figure 4.40: The PR-2 robot with a shadow hand.

same dimension as the foam box in Fig. 4.4, but it is made from rigid plastic via a 3D printer. With the same grasp configuration, the rigid object is manipulated as the foam box explained in the last enhanced manipulation experiment. The snapshots are shown in Fig. 4.44(a).

4.8.2 Manipulating Various Objects

Apart from the material, of which the object is made, our method is also able to manipulate objects with different shapes. In this section, the results about the manipulation of the objects with different shapes and materials are presented. Besides the boxes, different objects, like a remote control, a coffee capsule, and a square plate, are used as shown in Fig. 4.43. Since these objects have different sizes, shapes, and materials; their grasping configurations have to be different. The same as the manipulations conducted before, stable grasps have been achieved (from human knowledge). Enhanced manipulations have been implemented to rotate these objects. Finally, successful manipulations are carried out on them. The manipulation result is presented in Fig. 4.44. In Fig. 4.44(b), snapshots of manipulating the coffee capsule are shown. In Fig. 4.44(c), snapshots of manipulating the square plate are shown. In Fig. 4.44(d) snapshots of manipulating the remote

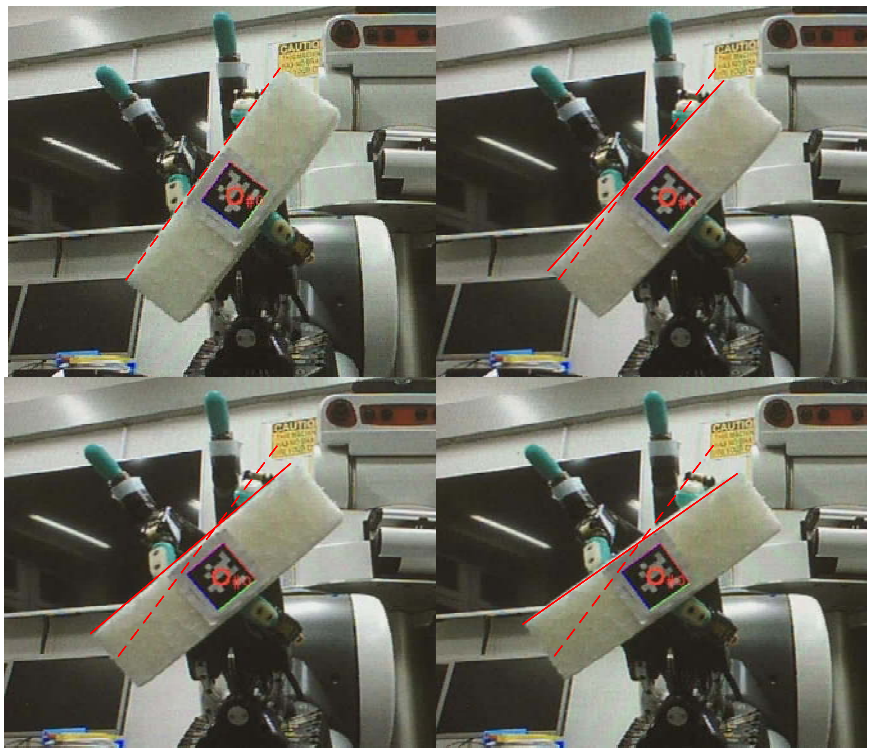


Figure 4.41: Snapshots of enhanced manipulation.

Figure 4.42: The rigid object with the same dimension as the foam box.

control are shown.

4.9 Comparing to Virtual Frame Model

In order to evaluate our manipulation method comprehensively, a comparative manipulation is performed based on the virtual frame model proposed in [161] which is briefly introduced in section 2.3.2. In this experiment, the manipulated object is the foam box.

(a) A remote control.

(b) A caffee capsule.

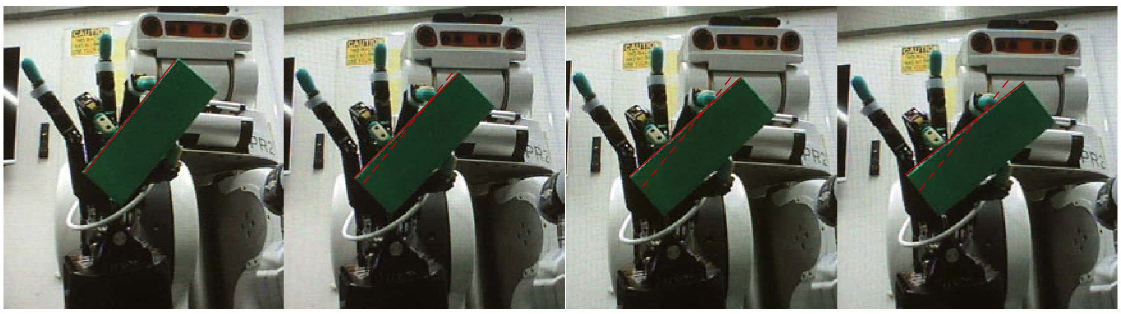
(c) A square plate.

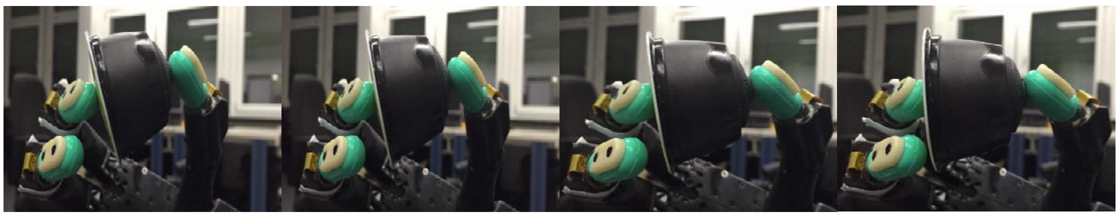
Figure 4.43: Various objects.

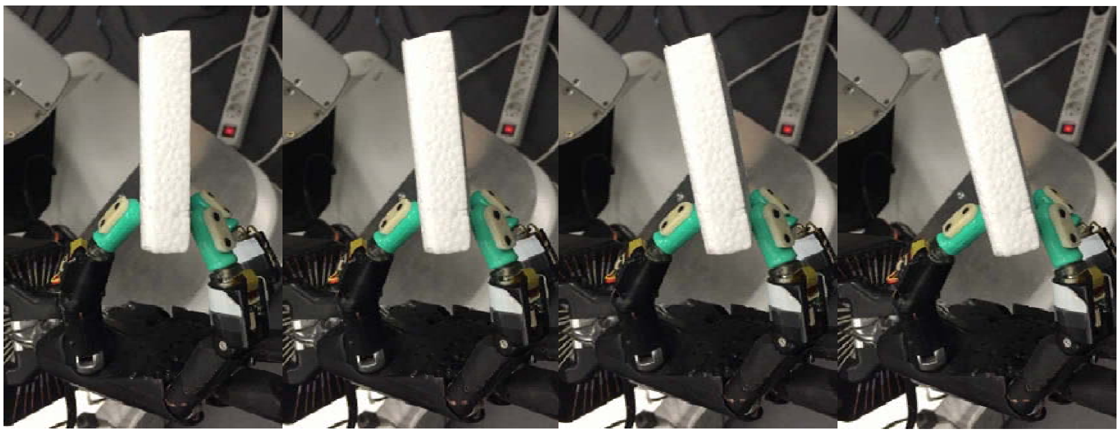
4.9.1 Manipulating with Virtual Frame Model

In this manipulation, a three fingers grasp is used. This grasp has the same initial grasping configuration as in the experiments conducted before. The contact position is assumed to be located at the center of the fingertips. Therefore, a virtual frame in a shape of a triangle is built with dash lines and its motion path is shown in Fig. 4.45. In this process, the rotation center of the frame is calculated according to Eq. 2.2. And the bunches of red arrows denote the velocity of the fingers with respect to the virtual frames.

The snapshots of the manipulation are shown in Fig. 4.46. In this manipulation, the object rotates as in yaw manipulations. And the robot tries to rotate the object as much as it can. As a result, the object rotates in 12° until one of the


(a) Snapshots of manipulating the rigid box.


(b) Snapshots of manipulating the coffee capsule.


(c) Snapshots of manipulating the square plate.

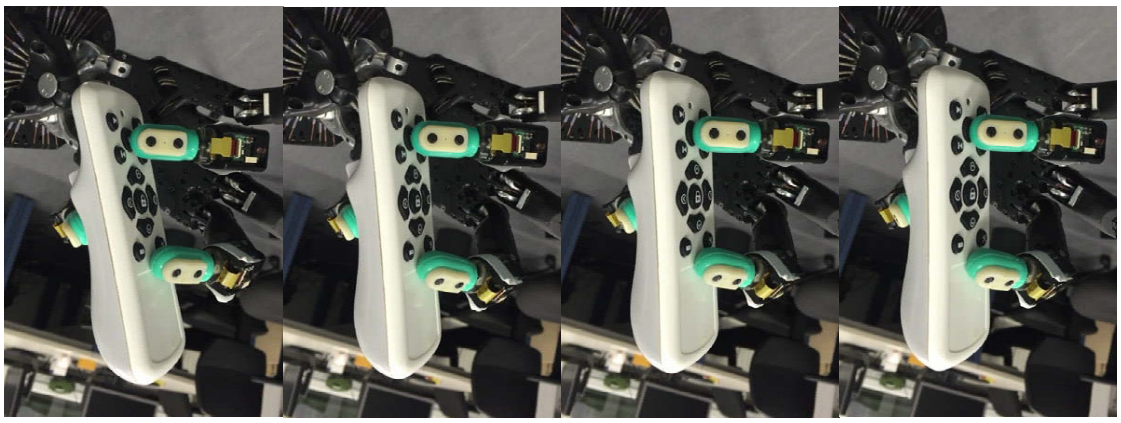

(d) Snapshots of manipulating the remote control.

Figure 4.44: Snapshots of manipulating various objects.

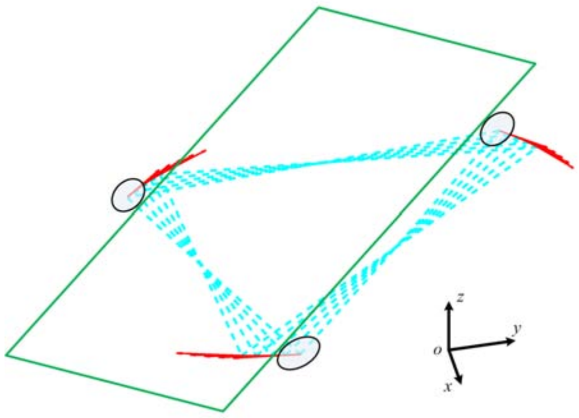


Figure 4.45: The moving path of the fingers based on the virtual frame model.

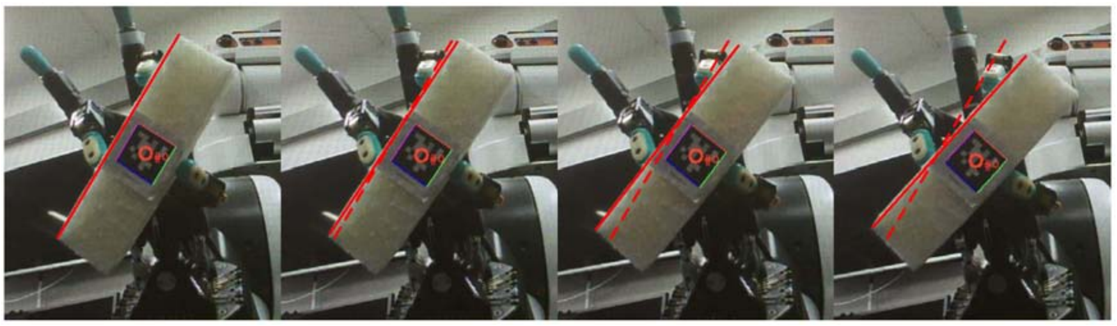


Figure 4.46: The snapshots of a manipulation based on virtual frame model.

hand's joints reached its boundary. In the end, its position changes about $3.7mm$ which is considered as the position error in rotation manipulations.

4.9.2 Comparative Results

Tab. 4.2 presents the comparative results between the enhanced and the virtual frame manipulations.

In Tab. 4.2, several comparative aspects are considered. In this table, the 'Manipulate various objects' refers to whether the method can manipulate various objects with different shapes, materials, weight, etc. Obviously, since the object is modeled with a triangle frame based on the fingers' position, both of these methods are able to manipulate different objects. The 'Prepare in advance' means whether the robot has to do something before manipulating. Only in enhanced manipulations, 'exploration' is required to collect proper information for the 'right' push direction. The 'visual and tactile information' refers to the necessity of sensing channels in these methods. The virtual frame method requires no sensing information, but the enhanced manipulation requires the visual and haptic feedback. The 'Two finger Manipulation' denotes to whether these methods are available to manipulate under two fingers grasps. In section 4.4, the pitch

Table 4.2: Comparison of the enhanced and the virtual frame manipulations.

	Virtual Frame	Enhanced Manipulation
Manipulate various objects	yes	yes
Prepare in advance	no	yes
Visual and haptic information	no	yes
Two fingers manipulation	bad	good
Max rotation [$\circ$]	small (12°)	large (18°)
Position error [$mm/^\circ$]	$0.31 mm/^\circ$	$0.26 mm/^\circ$
3D manipulation	yes	yes
Robustness	bad	good
Safe to the robot	bad	yes

manipulation is a special case of the virtual frame manipulation when the push direction is given by $P_d = (0, 0)$. In that experiment, there are slips in contact areas. In other words, the object may fall off when the finger pushes as in the virtual frame manipulation. Hence, we conclude that the virtual frame is not available to two fingers manipulations. The 'Max rotation' is the largest rotational angle of the object within the workspace of the hand. In experiments of section 4.9.1 and 4.8, with the same grasp configuration, the object is rotated until any joint reaches its boundaries. As a result, the object's largest rotated angle achieved by them is 12° and 18° respectively. The 'position error' refers to position changing of the object when it reaches the largest rotational angle. According to the experiments, their position change is $3.7mm$ and $4.6mm$ respectively. And this research uses the average position change in every degree as their position errors, that is $3.7/12 = 0.31mm/^\circ$ and $4.6/18 = 0.26mm/^\circ$. The '3D manipulation' refers to the ability to rotate the object in the roll and pitch directions. The 'Robustness' means whether these methods can deal with poor parameters of their models. The virtual frame method requires no information about the objects, but a precise robot model is required. Our enhanced manipulation requires neither object's information nor robot's precise model. The 'Safe to the robot' refers to whether there is any mechanism to protect the robot from unexpected behaviors or disturbances from the environments during the manipulations. In our enhanced manipulation, the spring support finger protects the hand from damages. When weird actions happen, hazard contacts are generated on fingers. By using a spring support finger, the compliance properties make the finger step away in order to reduce the hazard contact force to protect the robot.

In summary, our enhanced manipulation method is better than the virtual frame method in terms of 'Two fingers manipulation', 'Two fingers manipulation', 'Max rotation', 'Position error', 'Robustness', and 'Safe to the robot'. Certainly, our manipulation method also has some drawbacks, such as it requires more time to collect some information before manipulations, more sensors are required in the process of manipulations, more computing power is required, etc. To conclude, according to the comparative result, the enhanced manipulation is a precise, efficient, flexible, and robustness method.

Chapter 5

Learn to Improve Manipulation Skills

Learning is one of the most important aspects of artificial intelligence. Its goal is to build an agent which can adapt to its environments by learning from its past experience. Machine learning is an active research topic, and it attracts many researchers from various research fields [159].

In robotics, many tasks are complicated, like a sequence of actions are required for future successes. Generally speaking, robot learning is an optimal process, in which an agent is trained to generate proper actions to reach maximum (or minimum) rewards in the future[93]. In our robotic manipulation tasks, sequence actions of finger joints are carried out by the robot. After fingers' actions, the in-hand object moves to a new state. The performance of the sequence actions is evaluated from rewards including visual and haptic information.

Although, good in-hand manipulations have been achieved through linear push actions in section 4; an alternative approach to the robot manipulation is reinforcement learning. In this chapter, we would like to let the robot learn the manipulation skills itself through interacting with its environments.

This chapter is organized as follows:

- In section 5.1, the architecture of the manipulation learning system is introduced.
- In section 5.2, a manipulation simulator is built. It consists of 4 RBFNs: visual, first haptic, second haptic, and third haptic RBFNs. The visual RBFN maps the push commands to visual reward, and the haptic RBFNs map the push commands to the elements of the haptic features. These RBFNs are trained by the data collected from real manipulation experiments.
- In section 5.3, stochastic policy based reinforcement learning is introduced.
- In section 5.4, learning experiments are performed. In these experiments, two reinforcement learning (RL) algorithms are adopted: Williams' Episodic RE-

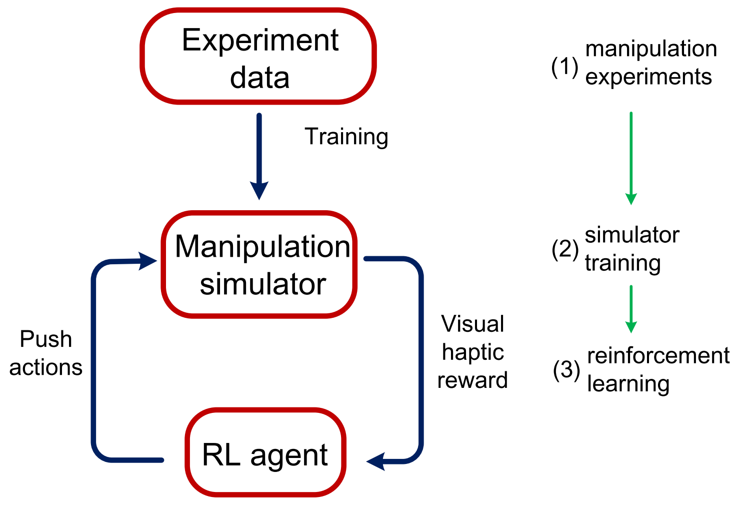


Figure 5.1: A manipulation learning architecture.

INFORCE and Peters' Episodic Natural Actor-Critic. In the manipulation learning task, multimodal rewards have been used in the learning process.

5.1 Learning Framework

Different from most classic RL benchmark problems, robotic RL problems are often represented by high dimensional continuous state and actions. Moreover, with little knowledge of the environment, robots have to discover their optimal actions automatically, which is called exploration. In robot control problems, improper exploration is very dangerous, since it may cause hazard damages to the robot and its environments. Furthermore, more than thousands of episodes are often required in training process of reinforcement learning. However, conducting thousands of uncertain actions on real robots is impractical in most robotic applications.

To apply learning algorithms and protect the robot, a robotic in-hand manipulation simulator is built. This simulator simulates the output close to the real system. After that, an RL agent is built to interact with the simulator. Through interacting with this simulator, the learning agent improves little by little with learning algorithms.

Architecture of Manipulation Learning

A manipulation learning architecture is presented in Fig. 5.1. It consists of two parts: a manipulation simulator and an RL agent. The manipulation simulator receives the manipulation actions from the RL agent and generates the output close to the real manipulation system. The RL agent is built to receive the system state and generate the manipulation actions. Besides, a density manipulation experiment is performed on our real robot platform to generate the training examples (manipulation data) to train the manipulation simulator. More experimental details are presented in section 5.2.2.

With the manipulation simulator and the RL agent, the learning process is performed in three steps. Firstly, a density push manipulation experiment is conducted to collect visual and haptic data. In this experiment, the push parameters and the visual and haptic feedback is recorded respectively as training examples. Secondly, a manipulation simulator is built with neural networks which are trained with the manipulation examples collected in the first step. At last, a policy based RL agent is built and RL algorithms are implemented. This agent generates manipulation commands according to the current system state. Through interacting with the manipulation simulator, the agent improves step by step with reinforcement algorithms.

Manipulation Simulator

A manipulation simulator is built to take the place of the real manipulation system with which the RL agent interacts. To build this simulator, usually there are two approaches.

One approach is physics engine based dynamic simulation. It models the time-varying behaviors of the system based physical theories. Usually, to describe the model's behaviors, ordinary differential equations or partial differential equations are used [53]. In this simulation, mathematical models are used to model the real-world constraints and the models solved by iterating over the state. This physics engine based dynamic simulation is widely used in robot controls, nuclear power, vehicle modeling, etc. Physics engine based dynamic simulation is very useful in the tuning of the mechatronic systems, since it can run in real time but in a virtual space, and it gives a result close to the real system. There are many famous software based on this approach, such as Gazebo[1], Adams[2], V-REP[3], etc. However, in contact problems, this approach fails in most cases, especially referring to the contact with soft materials. In our manipulation, the object is driven by the contact force applied by fingers, and the contact areas (fingertips) are often covered by soft materials. Therefore, this approach is not suitable to simulate the robotic manipulation.

Another approach is using function simulator in which a general function approximator is used to approx the input-output map. The function approximator is often constructed with neural networks. However, in this approach, there is an additional training process in which training examples are required. The performance of the approximator highly depends on how it is structured and trained. If good training examples are available, it is a much generalized and efficient approach. Fortunately, in this research, these visual and haptic training examples are attainable in real robot manipulations, like the manipulations performed in the experiments in section 4.5. As a result, this approach is used to simulate the

[1]http://gazebosim.org/

[2]http://www.mscsoftware.com/zh-hans/product/adams

[3]http://www.coppeliarobotics.com/

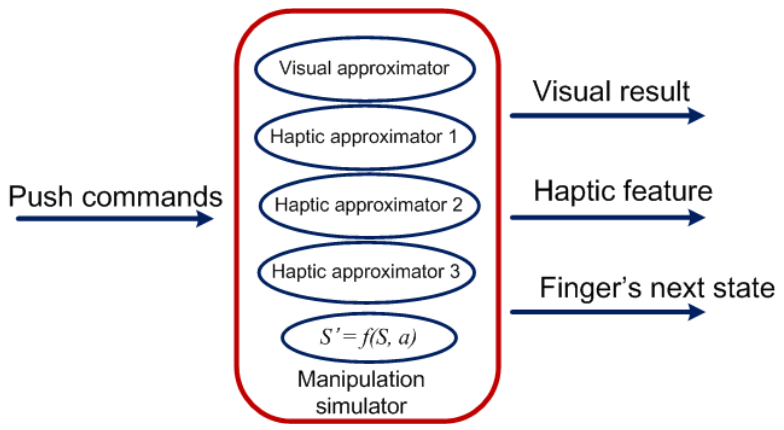


Figure 5.2: A manipulation simulator. Its input is the push commands, and the output is the object's visual reward and haptic features. This simulator consists of four one-dimensional function approximators.

manipulation system. Essentially, the function approximator is a function which maps the manipulation commands to the visual and haptic rewards.

Of this manipulation simulator, the input is the push commands; and output is the visual reward and the haptic features, as shown in Fig. 5.2. This simulator consists of four one-dimensional function approximators: one visual approximator and three haptic approximators. The visual approximator maps the push commands (the push distance and push direction) to the visual reward (degree change of the object's orientation). The three haptic approximators map the push commands to elements of the haptic features (pk_1, pk_2, and pk_3) respectively. All these approximators are constructed with Radial Basis Function Neural Networks (RBFNs). These networks are trained with manipulation examples collected in real robot experiments. More details are introduced in the following sections.

Reinforcement Learning Agent

Policy based method is one of the most used RL methods in robot control problems [22], [101], [139], [141]. In this thesis, a policy-based learning agent is used to explore the space of the possible policies according to the received feedback.

As shown in Fig. 5.3, this RL agent is a stochastic policy which is represented by π_θ. It takes the finger's state (position in Cartesian space) as input and produces a modified push command. With this push command, visual and haptic rewards are generated by the simulator. After processing these rewards, an optimized gradient descent direction $\nabla\mathbf{W}$ is obtained, which denotes the expected changing direction of the policy parameters θ. Policy π_θ updates in direction $\nabla\mathbf{W}$. After repeating this process, the changes in the policy parameters accumulate. Therefore, the RL agent improves little by little. After the rewards are converged, this learning process is completed successfully.

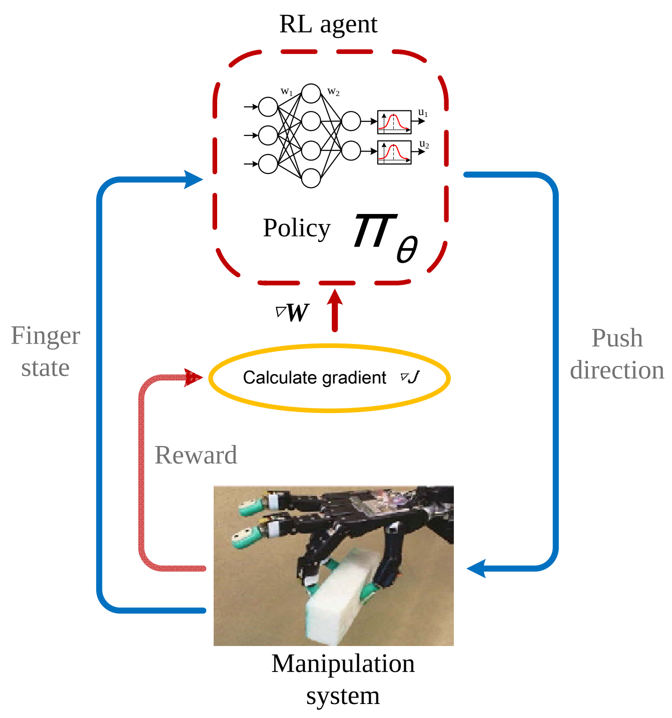


Figure 5.3: A stochastic RL agent for manipulation.

5.2 In-hand Manipulation Simulator

In this section, a manipulation simulator has been built based on the real experimental data. Moreover, to construct the simulator, Radial Basis Function Networks (RBFNs) are used.

5.2.1 Radial Basis Function Networks

A Radial Basis Function Network (RBFN) is a kind of neural networks. After first proposed by Park and Sendberg in 1991, [136] it is very popular in data modeling, like function approximation.

RBFN is a feed-forward neural network consists of three layers: an input layer, a hidden layer, and an output layer. The input layer is a 'fan-out' without any data processing in this layer. The second layer consists of kernels which weight the distance between the input and the kernels, called radial basis function. The final layer adds its input linearly together as its output. Obviously, the kernel is the most important part of the RBFN, which stores data (examples) from the training set.

Various radial basis functions have been tested as the kernels of the RBFN. One of the most frequently used kernels is the Gaussian kernel. The i^{th} Gaussian kernel with an input vector $\mathbf{x}$ is

$$\phi_i(\mathbf{x}) = exp(-\sum_{j=0}^{n} \frac{(x_j - u_{ij})^2}{2\sigma_{ij}^2}), \tag{5.1}$$

where u_{ij} refers to the center location of the i^{th} kernel for the j^{th} variable. It is also called prototypes. σ is the width of the kernel.

Obviously, the training of an RBFN refers to three parameters: prototypes μ, the width of kernel σ, and output weights $\mathbf{w}$. Many methods have been proposed to select the prototypes and train the RBFN [44], [136], [131], [151]. According to the research in [151], in this thesis, the algorithm k-Means clustering is used to select prototypes intelligently. In this algorithm, the training data set is clustered, and their cluster centers are chosen as the prototypes. In more detail, first the training data is separated into n classes; and then the average of all the points in each cluster is computed as the cluster center. In this algorithm, the number of the clusters n is determined heuristically. A large number of n denotes more prototypes. This means a more complex decision boundary can be received, however, this also causes the problem that the network' training and evaluation is more difficult.

5.2.2 Density Push Experiment for Yaw Manipulation

In order to collect sufficient training examples, a density push experiment has been carried out in this section. This experiment is very similar to the multi-directional manipulation experiment introduced in section 4.6. Differently, the push actions are conducted with different distance in this experiment. The push distance changes from $P_l = 2mm$ to $10mm$ by step $1mm$, and the push direction changes from $\mathbf{P}_d = (-60, -60)$ to $(60, 60)$ by step 15°. In summary, the push sequence is listed in Algorithm 1. Totally, 380 pushes are performed in this experiment.

Algorithm 1 The push sequence in the density push experiment.

```
for $P_l = 2$ to 10 mm do
    for $\theta = -60°$ to $60°$ by $15°$ do
        for $\alpha = -60°$ to $60°$ by $15°$ do
            Push execution with $\mathbf{P} = [\theta, \alpha, P_l]^T$;
        end for
    end for
end for
```

In these manipulations, all the visual and haptic data are collected; moreover, the visual reward and the haptic features are extracted as training data set. Their visual reward and the haptic features are shown in Appendix B.

5.2.3 Manipulation Simulator based on RBFNs

In the manipulation simulator, there are 4 RBFNs. They are visual RBFN, first haptic RBFN, second haptic RBFN, and third haptic RBFN.

The visual RBFN maps the push parameters to the visual reward introduced in 4.2.4 in Eq. 4.3. More specifically, the input of this RBFN is the push direction

$\mathbf{P}_d$ and the push distance P_l. Its output is the degree change in object's rotations in the expected manipulation direction $\mathbf{V}'_r$.

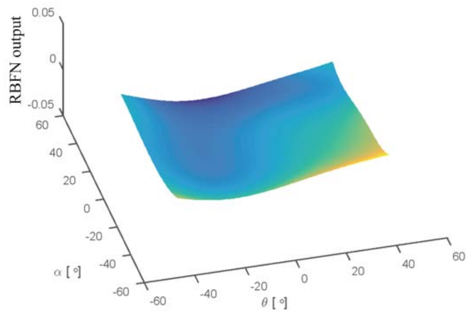
The first haptic RBFN maps the push parameters to the first element of the haptic feature introduced in 4.2.4. The input of this RBFN is the push direction $\mathbf{P}_d$ and the push distance P_l. Its output is the first element of the haptic feature pk_1. Similarly, the second and third haptic RBFNs map the push parameters to the second and the third element of the haptic feature pk_2 and pk_3 respectively.

Visual RBFN

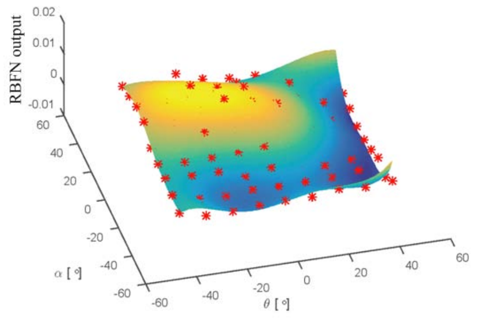
The visual RBFN result is shown in Fig. 5.4 in polar coordinates, where the training parameters are $\sigma = 1$, $n = 45$. As a result, after training the mean squared error (MSE) $e = 1.8 \times 10^{-5}$. To visualize the four-dimensional data, the result of the visual RBFN is sliced into 10 surfaces regarding to the push distance P_l, as shown in sub-figures 5.4(a), 5.4(b), 5.4(c), 5.4(d), 5.4(e), 5.4(f), 5.4(g), and 5.4(h). Each sub-figure shows the visual RBFN result with a fixed push distance P_l. In Fig. 5.4, the red dots are the real example data. Specially, $P_l = 0mm$ means that there is no push action. Hence there is no data point in Fig. 5.4(a), and the output of the visual RBFN approximates to 0. Equivalently, in this experiment, no manipulation is performed when the push distance $P_l > 10mm$; so that there is no data point neither in Fig. 5.4(g) nor 5.4(h). It can be found that all the sub-figures show a surface with single bump except the sub-figures when push distance $P_l = 1mm$. However the center position of the bumps changes as the push distance P_l increases. Besides, the surfaces in Fig. 5.4(b) and 5.4(a) are not as regular as others. That is because when the push distance is small, $P_l \leqslant 2mm$, the fingers are hard to be precisely controlled in our robotic platform for their limited control resolution discussed in the previous section. Therefore, there is no training example data within this small push distance. The RBFN result is poor in the area of small push distance. The surface in Fig. 5.4(a) is very close to a horizontal whose value is zero. In the RBFN, the output will be small when the input points are far away from the kernels. As a result, the irregular result of the RBFN has less impact on the performance of the manipulation simulator. That is one of the reasons why we use RBFN in this thesis.

The result of Fig. 5.4 is also shown in polar space. For better understanding, one of the sub-figures (Fig. 5.4(e)) is transferred and represented in a Cartesian space, as shown in Fig. 5.5. The origin of this coordinate frame is at the contact point of the push finger at its initial state. The direction of the radius is the push direction, and the length of the radius is the visual rewards.[4] A new ellipsoid surface is obtained in Fig. 5.5. Intuitively, the longer radius length, the better manipulation is. Hence, in Fig. 5.5, the convex direction of the ellipsoid surface indexes the direction in which the best visual reward can be obtained. Considering all the push distance, the total results of visual RBFN is shown in Fig. 5.6 in a

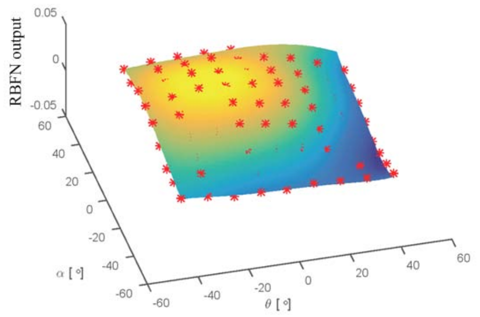
[4]A baseline $b = 0.04mm$ is added to the radius length, since there are negative visual rewards in some push directions.



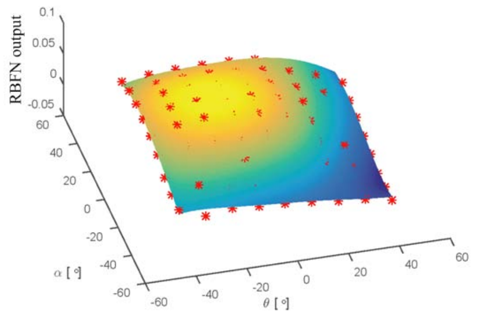
(a) $d = 0\ mm$



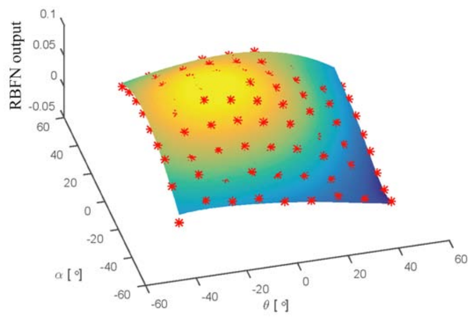
(b) $d = 2\ mm$



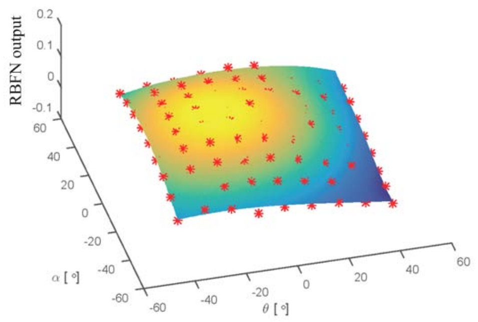
(c) $d = 4\ mm$



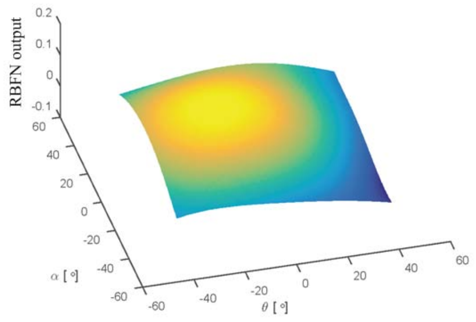
(d) $d = 6\ mm$



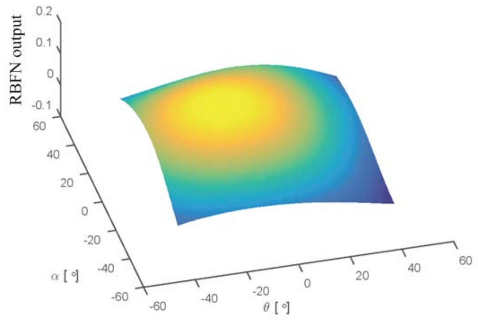
(e) $d = 8\ mm$



(f) $d = 10\ mm$



(g) $d = 12\ mm$



(h) $d = 14\ mm$

Figure 5.4: The visual RBFN result in polar coordinates. In order to visualize this four-dimensional data, this result is sliced into 10 surfaces in sub-figures The red dots in Fig. 5.4(b), 5.4(c), 5.4(d),5.4(e), and 5.4(f) are training examples.

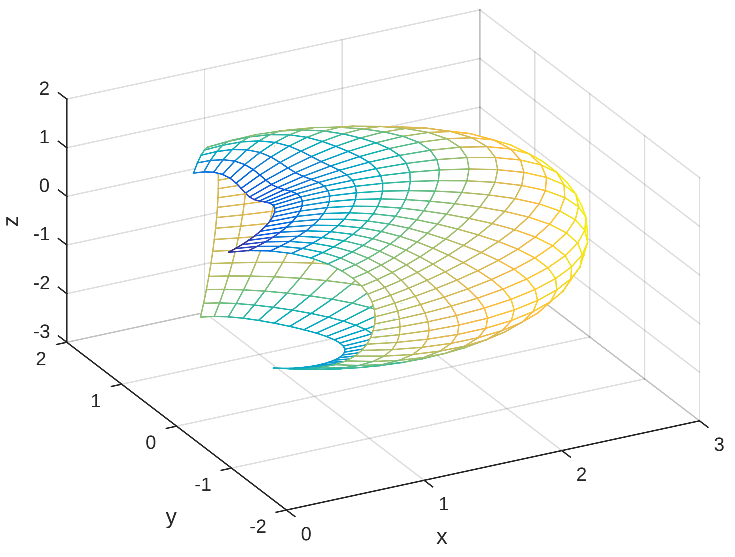


Figure 5.5: Transferring sub-figure 5.4(e) into Cartesian coordinate.

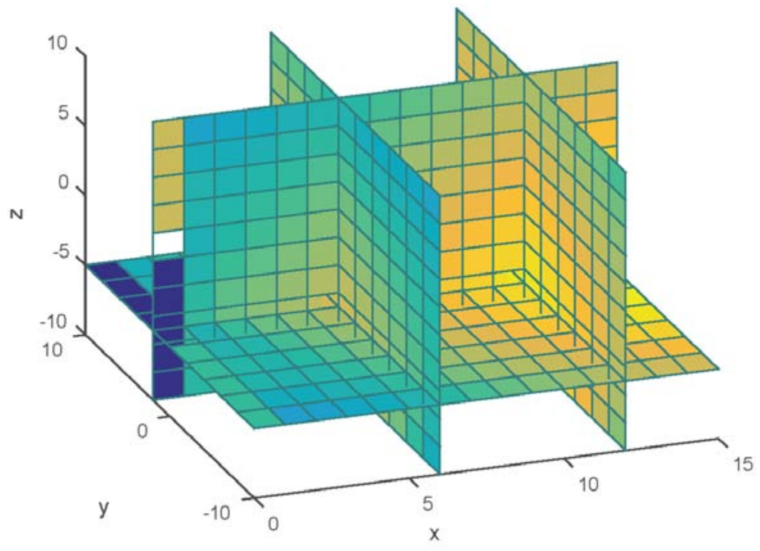


Figure 5.6: Visual RBFN result in a Cartesian coordinate.

Cartesian coordinate, where the yellow area refers to a large visual reward, and the blue part refers to a small value.

Another way to represent the result of the visual RBFN is Fig. 5.7, where the horizontal axis refers to a data sequence arranged by the order of Algorithm 1, and the vertical axis refers to the output of the visual RBFN. In this figure, the blue stars refer to the training example data and the red lines denote the result from the visual RBFN. This figure gives an intuitive impression of the result of the visual RBFN in a 2D plot. Obviously, there are 5 segments in Fig. 5.7. Each of them refers to different push distances from $P_l = 2mm$ to $P_l = 10mm$. Taking

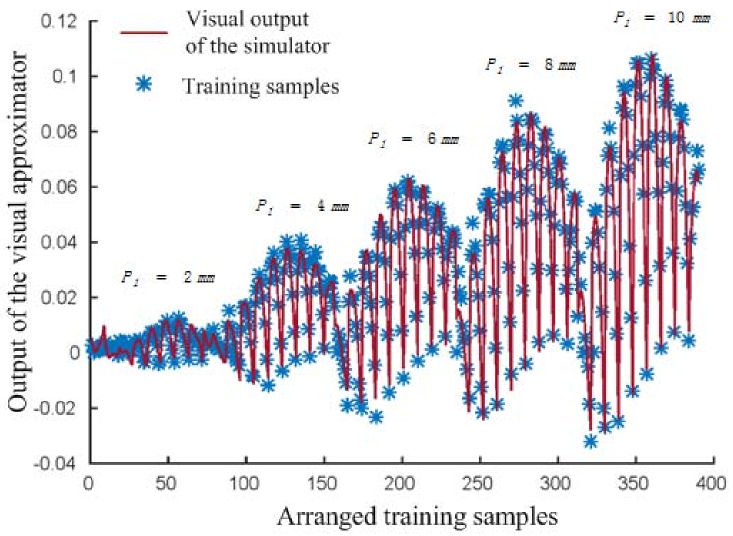


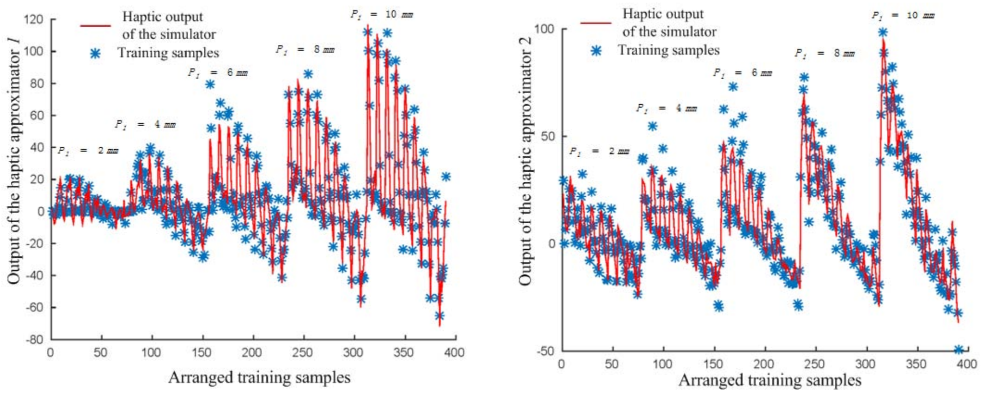
Figure 5.7: Visual RBFN result shown with a two-dimensional plot. All experiment training examples are plotted with blue stars in this figure. The red solid lines refer to the visual RBFN result.

into account the layout of this thesis is limited, results of the rest haptic RBFNs are represented as in Fig. 5.7 in the following section.

Haptic RBFNs

Besides visual RBFN, results of haptic RBFNs are presented in this section. The parameters of the first haptic RBFN are given with $\sigma = 1$, $n = 50$. After training, its MSE $e = 38.6$.[5] The second haptic RBFN parameters are given with $\sigma = 1$, $n = 50$. After training its MSE $e = 46$. Their result is plotted in Fig. 5.8. Fig. 5.8(a) illustrates the result of the first haptic RBFN; and Fig. 5.8(b) shows the result of the second haptic RBFN. Obviously, it can be seen that the performance of the second haptic RBFN is not as good as the first haptic RBFN. To explain that the result of the second haptic RBFN is represented in Fig. 5.9 in polar coordinates. Points with large errors are in the area with the push distance $P_l = 4mm$ and $6mm$ and in the direction $\mathbf{P}_d = (60, 60)$, represented in Fig 5.9(c) and 5.9(d). This is because these small slips occur in these push directions. These slips can be confirmed by the third element of the haptic feature pk_4 in Fig. 5.10(a) marked with two black circles. As discussed in section 4.3, a large value of pk_3 denotes that the object slips on the fingertips during the manipulation. These small slips change the haptic data slightly. Since the output errors of the haptic RBFN are not small, the performance of the haptic RBFN is still acceptable. That is because the points with large errors gather in one corner of the surfaces. They can be

[5]This MSE is much larger than the visual RBFN result. That is because they have different units. The unit of the visual reward is rad. For a 10° rotation, the visual reward changes only 0.17. However, the haptic is bit (here 1bit = 0.00365kPa). Generally, it changes from 10 to 200. Hence, the MSE of this RBFN result is larger compared to the result of the visual RBFN.



(a) First haptic RBFN result.

(b) Second haptic RBFN result.

Figure 5.8: Haptic RBFN result shown with two-dimensional plot.

ruled out easily in Cartesian coordinates. Actually, this corner is rarely reached by the robot in the learning process. As a result, these unstable push directions are ignored, and the simulated system is assumed to be stable in manipulations.

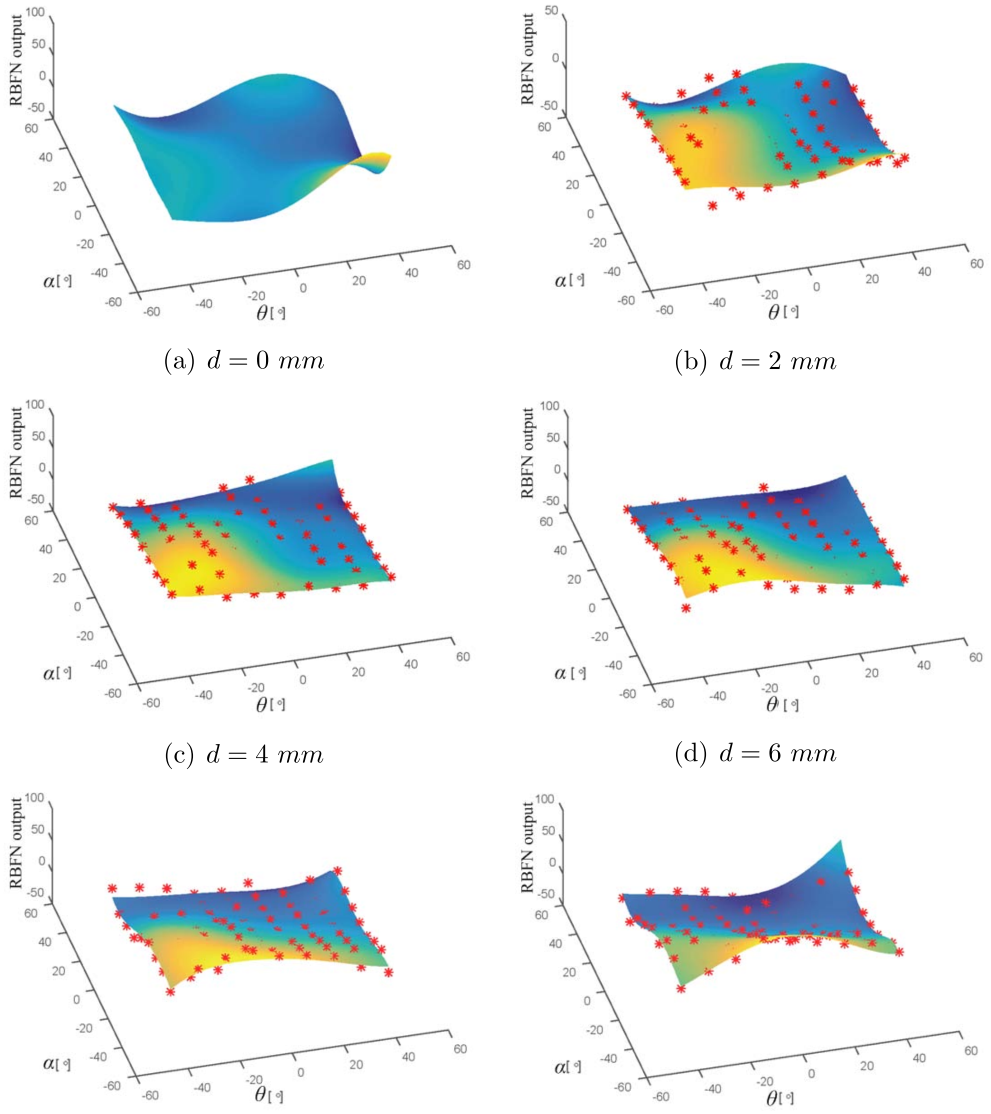
The third haptic RBFN parameters are given with $\sigma = 1$, $n = 50$. After training, its MSE $e = 35$. Its result is plotted in Fig. 5.10(a). Unfortunately, the performance of the third haptic RBFN is bad. That is because the value of pk_4 is smaller compared to other elements. Hence, they are easily impacted by noise. Thereby we focus on its distribution as represented in Fig. 5.10(b), where the solid red curve refers to experiment data and the blue dash line refers to a Gaussian distribution $\mathcal{N}(0, 5.95)$. Obviously, these two curves exhibit the similar pattern. As a result, the result of the third haptic RBFN is not used in the following learning experiments.

5.3 Policy Based Reinforcement Learning

As described in section 5.1, besides the simulator, RL agent is the other important part of this learning system. In this section, we focus on building an RL agent and implementing it to learn the manipulation skills through interacting with the manipulation simulator built in section 5.2.

5.3.1 General Assumption and Problem Statement

In reinforcement learning, the environment is typically built with a Markov Decision Process (MDP). An MDP is defined by state-action sets and an one-step dynamics of the environment [159]. Given state s, action u, and possible next state s', the environment dynamics are modeled by a transition probabilities $P^u_{ss'}$ and a reward $R^u_{ss'}$. In this thesis, the transition probabilities $P^u_{ss'}$ and the reward $R^u_{ss'}$ is replaced by functions f and ρ. As a result, an MDP is represented by a tuple $\langle X, U, f, \rho\rangle$. In this tuple, X is the state; U is the action; f is the state

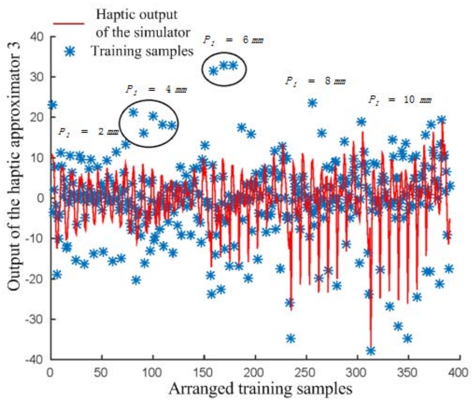


(a) $d = 0\ mm$ (b) $d = 2\ mm$

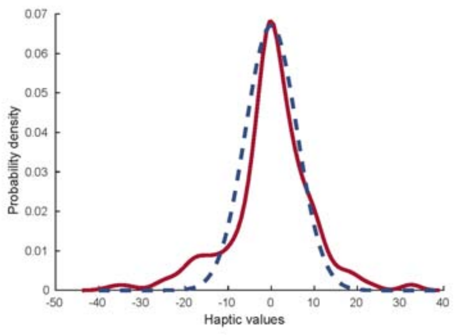
(c) $d = 4\ mm$ (d) $d = 6\ mm$

(e) $d = 8\ mm$ (f) $d = 10\ mm$

Figure 5.9: Haptic RBFN regression results in polar coordinate.
The data sets are four-dimensional. In order to visualize the RBFs result, it is sliced into 10. The red dots in Fig. 5.9(b), 5.9(c), 5.9(d),5.9(e), and 5.9(f) are real measured data. Obviously, the slice shape in Fig. 5.9(a) and 5.9(b) are not as regular as others. It is because when the push distance is small, the robot fingers are hard to be controlled precisely for its poor control resolution discussed in the previous section.



(a) The third haptic RBFN result.



(b) Data distribution.

Figure 5.10: The third haptic RBFN result and the data distribution.

transition probability density function with $f : X \times U \times X \longmapsto [0,\ \infty)$; and ρ is a reward function with $\rho : X \times U \times X \longmapsto \mathbb{R}$.

It is important to note that in this manipulation problems the state and actions are continuous. Therefore, it is impractical to use a probability function which maps a current state s_k and an action u_k to a certain next state. Therefore, in the probability function, reaching a certain state region is used instead of a certain next state [72]:

$$P(x_{k+1} \in \mathbf{X}_{k+1}|x_k, u_k) = \int_{\mathbf{X}_{k+1}} f(x_k, u_k, x')dx'. \tag{5.2}$$

The action u_k at the state x_k is drawn from a policy defined by $\pi : X \times U \longmapsto [0, \inf)$. After reaching the state region $\mathbf{X}_{k+1}$, the reward is

$$r_{k+1} = \rho(x_k, u_k, x_{k+1}). \tag{5.3}$$

In learning, the agent estimates a cost function $J(\pi)$, or named 'cost-to-go' function, which is the expected value of a certain function g of the received rewards with a policy π. The cost function is

$$J^\pi = E\{g(r_1, r_2, \dots)\}. \tag{5.4}$$

In some algorithms, cost-to-go value relates only to state; therefore a **state value function** is defined:

$$V^\pi(x) = E\{\sum_{k=0}^{\infty} \gamma^k r_{k+1}|x_0 = x, \pi\}. \tag{5.5}$$

When the chosen action u_k is considered, a **state-action function** is defined:

$$Q^\pi(x) = E\{\sum_{k=0}^{\infty} \gamma^k r_{k+1}|x_0 = x, u_0 = u, \pi\}. \tag{5.6}$$

In [72], the functions J^π, $V^\pi(x)$, and $Q^\pi(x)$ have different definitions depending on discounted reward or average reward.[6]

5.3.2 Reinforcement Learning Categories

Generally, there are three main categories in reinforcement learning: critic-only, actor-only, and actor-critic methods.

Critic-only Methods

The critic-only methods use only state value function or state-action function. The well-known algorithms Q-learning and SARSA(λ) fall into this category. It learns the optimal value function $V^\pi(x)$ or value-action functions $Q^\pi(x)$ with a determined policy π:

$$\pi(x) = \arg\max_u Q(x, u). \tag{5.7}$$

One of the most wildly used updating methods for the value functions is Bellman optimality equation [159]:

$$\begin{aligned} V^*(x) &= \max_u E\{\rho(x, u, x') + \gamma V^*(x')\}, \\ Q^*(x) &= \max_u E\{\rho(x, u, x') + \gamma \max_{u'} Q^*(x', u')\}. \end{aligned} \tag{5.8}$$

Unfortunately, as described in [72], there is no reliable guarantee on the policy for any approximated value function in online learning. In POMDPs, a small change or error in estimated value function may cause large changes. It results in continuous changes in the policy. In real robot applications, with this major drawback, learning agent may generate hazard changed actions which may endanger robots and its environments. Although the critic only methods are useful in many completed observable problems, it is inefficient in continuous state or action applications. On the one hand, the space of the value function increases exponentially as the number of actions increase. Besides, it is almost impossible for a determined policy to generate continuous actions. On the other hand, for multi-dimensional or continuous state, the true value function is hard to be estimated with a determined policy. Therefore, the actor-only methods are used in some reinforcement learning algorithms by focusing on optimizing the policy.

Actor-only Methods

The actor-only methods are policy gradient methods, which parameterize the policy without restoring any value function. Usually, the policies are stochastic, named stochastic policy. The well-known Williams' REINFORCE algorithm [178] and Baxter's GPOMDP algorithm [25] belong to this category. In these algorithms,

[6]In Peters' reinforcement learning [139], these different forms are generalized by a coefficient of the rewards a_l.

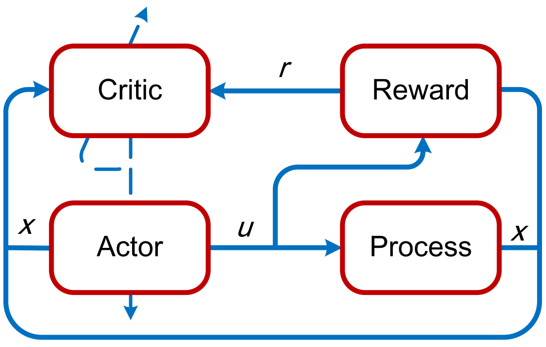


Figure 5.11: Actor-critic methods.

the policy is parameterized with θ; and their general goal is to maximize the expected return $J(\theta)$ by optimizing the policy parameter θ.

Assuming the parameterized expected return is differentiable with respect to θ, the gradient of the cost function is

$$\nabla_\theta J = \frac{\partial J}{\partial \pi_\theta} \frac{\partial \pi_\theta}{\partial \theta}. \tag{5.9}$$

With optimization technology, a simple updating rule of the parameterized policy is gradient decent:

$$\theta_{k+1} = \theta_k + \alpha_k \nabla_\theta J, \tag{5.10}$$

where α_k refers to learning rate.

There is a guarantee for the policy to converge at a local minimal if the estimated gradient is unbiased and the learning rate α_k fulfills [159]

$$\sum_{k=0}^{\infty} \alpha_k = \infty, \quad \sum_{k=0}^{\infty} \alpha_k^2 < \infty. \tag{5.11}$$

Usually, the policy is represented by an independent function approximator (usually neural networks with parameterized weights and some distribution density functions), whose input is the environment state and output is the distribution of selected actions.

Actor-critic Methods

The actor-critic method is a combination of the actor-only and critic-only methods [159]. In Fig. 5.11, a schematic overview of the actor-critic algorithms is shown [72]. In this method, the actor generates actions based on a policy; and the critic evaluates the current policy. Usually, the evaluation is given by a value function method such as TD, LSTD, etc. In the learning process, the critic is first approximated and updates with samples, and then it is used to evaluate the policy. After the policy's evaluation, the actor is updated by using the information from the critic. For this methods, more details will be discussed in the following section.

5.3.3 Stochastic Policy Gradient Learning

In autonomous robotic applications, usually there is no sufficient knowledge about the dynamics of the environment. Since the policy gradient can not be obtained with model-based methods directly, the stochastic policy gradient method is used. A stochastic policy is a map, which maps the state x to action u with a probability distribution over the action space A [103], [160].

Early works on the stochastic policy gradient methods are SRV algorithm [73] and REINFORCE algorithm [178]. In [97], Kimura et al. used function approximations to represent a stochastic policy in POMDPs.

The stochastic policy is achieved by selecting the actions in a given stationary distribution. This distribution can be Gibbs distribution or Gaussian distribution.

Gibbs policy:

$$\pi_\theta(u|x) = \frac{e^{\theta_{xu}}}{\sum_b e^{\theta_{xb}}}. \tag{5.12}$$

Gaussian policy:

$$\pi_\theta(u|x) = \mathcal{N}(u|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(u-u_{x\theta})^2}{2\sigma^2}}. \tag{5.13}$$

One important result of the policy gradient is from the trick of likelihood ratio [178], [139]. A trajectory τ is generated by roll-outs with the distribution:

$$p_\theta(\tau) = p(x_0) \prod_{k=0}^{n} p(x_{k+1}|x_k, u_k)\pi_\theta(u_k|x_k). \tag{5.14}$$

Therefore, the gradient of the cost function is [139]

$$\begin{aligned} \nabla_\theta J(\theta) &= \int\limits_{\mathbb{T}} \nabla_\theta p_\theta(\tau) r(\tau) d\tau \\ &= \int\limits_{\mathbb{T}} p_\theta(\tau) \nabla_\theta log p_\theta(\tau) r(\tau) d\tau \\ &= E\{\nabla_\theta log p_\theta(\tau) r(\tau)\} \end{aligned} \tag{5.15}$$

The distribution of the trajectory gradient is exactly zero:

$$\int\limits_{\mathbb{T}} p_\theta(\tau) \nabla_\theta log p_\theta(\tau) d\tau = \int\limits_{\mathbb{T}} \nabla_\theta p_\theta(\tau) d\tau = \nabla_\theta 1 = 0, \tag{5.16}$$

A constant baseline can be added to Eq. 5.15. According to the research in [178], the baseline can be chosen arbitrarily, but a carefully chosen baseline can reduce the learning variance. Therefore, a suggested baseline b is:

$$b = b + (r - b)/j, \tag{5.17}$$

where j is the number of the episode, and r is the reward acquired from one episode. The baseline in Eq. 5.17 is a scalar based on the time and accumulated rewards. Some other optimal baselines are discussed in [139], [178] to decrease the learning variance without biasing the policy gradient in episodes. The baseline also can be given according to the current state and the policy: $b^\pi(x)$ as discussed in [139].

Eq. 5.14 and 5.15 are very important. They show that the derivative of the state distribution can be obtained from samples without a model of the system.

Based on this stochastic policy method, Williams proposed an algorithm named REINFORCE[7] [178]. It is shown in Algorithm 2 with cost function gradient:

$$\nabla_\theta J(\theta) = \langle (\sum_{k=0}^{n} \nabla_\theta \pi_\theta(u_k|x_k))(\sum_{k=0}^{n} a_k r_k - b)\rangle. \tag{5.18}$$

Algorithm 2 Williams' Episodic REINFORCE algorithm

```
**Input:** policy parameter $\theta$, learning rate $\alpha$, policy standard deviation $\sigma$, and baseline $b$;
**for** episode j **do**
    **Initialization**: $\pi_\theta \leftarrow \theta$, get initial state $X_0$, $r = 0$;
    **for** each step $i$ **do**
        $u_k \leftarrow \pi(\theta)$ and do action $u_k$;
        get next state $X_{k+1}$ and reward $r_k$;
        $r = r + r_k$;
        $\mathbf{e} = \mathbf{e} + \frac{\partial ln(\pi_\theta(X_k))}{\partial \theta}$;
    **end for**
    b = b + (r - b) / j;
    $\theta_{k+1} = \theta_k + \alpha_k (r - b)\mathbf{e}$;
**end for**
```

With policy gradient theory, an advanced way to express Eq. 5.18 by replacing the baseline with $b^\pi(x)$ is [160]:

$$\nabla_\theta J(\theta) = \int_X d^\pi(x) \int_U \nabla_\theta \pi(x,u)(Q^\pi(x,u) - b^\pi(x)) du dx. \tag{5.19}$$

where $d^\pi(x)$ is stationary distribution of the state x under policy π. $d^\pi(x) = lim_{t\to\infty} P\{x_t = x | x_0, \pi\}$ [160].

Particularly, when the state based baseline is the value function $V(x)$, an **advantage function** is defined: $A^\pi(x,u) = Q^\pi(x,u) - V^\pi(x)$. As a result, Eq. 5.19 is derived to

$$\nabla_\theta J(\theta) = \int_X d^\pi(x) \int_U \nabla_\theta \pi(x,u) A^\pi(x,u) du dx. \tag{5.20}$$

[7]In this thesis, this algorithm is called Williams' Episodic REINFORCE algorithm.

According to the research in [160], [103], term $(Q^{\pi}(x,u) - b^{\pi}(x))$ [8] can be approximated with a compatible function $f_w^{\pi}(x,u)$ with parameter $\mathbf{w}$ without affecting the unbiasedness of the gradient:

$$f_w^{\pi}(x,u) = (\nabla_{\theta} log\pi(u|x))^T \mathbf{w} \equiv Q^{\pi}(x,u) - b^{\pi}(x). \tag{5.21}$$

Therefore, Eq. 5.18 is deduced with the parameterized compatible function:

$$\begin{aligned} \nabla_{\theta} J(\theta) &= \int_X d^{\pi}(x) \int_U \nabla_{\theta}\pi(x,u)(\nabla_{\theta} log\pi(u|x))^T dudx\mathbf{w} \\ &= F_{\theta}\mathbf{w}, \end{aligned} \tag{5.22}$$

where

$$F(\theta) = \int_X d^{\pi}(x) \int_U \nabla_{\theta}\pi(x,u)(\nabla_{\theta} log\pi(u|x))^T dudx. \tag{5.23}$$

For the definition of the policy π, the probability of selecting an action is determinate, $\nabla \int_U \pi(u,x)du = \nabla 1 = 0$. An important attribute of the compatible function $f_w^{\pi}(x,u)$ is

$$\int_U \pi(u|x) h_w^{\pi}(x,u)du = \int_U \nabla_{\theta}\pi(x,u)du\mathbf{w} = 0. \tag{5.24}$$

The gradient used in Eq. 5.19 is the standard gradient, which is useful in the space with single minimum and has isotropic magnitude with respect to any directions away from its minimum, like Euclidean space. Obviously, the performance of the standard gradient highly depends on the structure of the cost function. In robot applications, robot joints are usually represented by rotational angles and task rewards are usually nonlinear. As a result, the parameter space of the cost function usually is not in Euclidean coordinate but with Riemannian metric structure, where the standard gradient does not indicate the steepest direction of a target function. Therefore, one of the efficient ways to this problem would be applying natural gradients suggested by Amari [16].

Kakade [90] used natural gradient in policy gradient methods to make the agent choose greedy optimal actions rather than just better actions. Peter introduced Natural Actor-Critic (NAC) algorithms by applying natural gradients [140], [138]. Their results show the natural gradient improves their learning methods significantly compared to the standard gradient.

A natural gradient is the steepest direction with respect to Fisher matrix. More details can be found in [16]. Suppose a cost function is $J(\theta)$, with the form $J(\theta) = \int_X p(x)l(x,\theta)dx$. The steepest ascent direction in the Riemannian metric structure is not in the direction $\nabla_{\theta} L(\theta)$ but in the direction:

$$\widetilde{\nabla}_{\theta} L(\theta) = G^{-1}(\theta)\nabla_{\theta} L(\theta), \tag{5.25}$$

[8] In other research, like in [72], there is no baseline $b^{\pi}(x)$ in this equation. It is from the different definition of Q function.

where $G(\theta)$ is the Fisher information matrix. It is defined by

$$G(\theta) = \int_X p(x)\nabla_\theta logp(x)(\nabla_\theta logp(x))^T dx. \tag{5.26}$$

From Eq. 5.25, the natural gradient is a linear transformation of the standard gradient with the inverse of the Finisher matrix. It is proved that the angle between the standard gradient and the natural gradient is never more than ninety degree. This makes sure the natural gradient converge to a local optimum [140].

One important conclusion in policy gradient learning methods is that the Fisher information matrix is exactly the matrix $F(\theta)$ defined in Eq. 5.23, that is $G(\theta) = F(\theta)$. With Eq. 5.22 and 5.25, the natural gradient of the cost function $\nabla_\theta J(\theta)$ is

$$\widetilde{\nabla}_\theta J(\theta) = G^{-1}(\theta)\nabla_\theta J(\theta) = G^{-1}(\theta)F(\theta)\mathbf{w} = \mathbf{w}. \tag{5.27}$$

This result denotes that for the natural gradient the matrix $F(\theta)\mathbf{w}$ is not necessary and only $\mathbf{w}$ is required. The policy parameter θ is updated by

$$\theta = \theta + \alpha\mathbf{w}, \tag{5.28}$$

where term α denotes the learning rate.

As a result, in terms of the advantage function, the Bellman equation [159] is written as:

$$\begin{aligned} Q^\pi(\mathbf{x},\mathbf{u}) &= A^\pi(\mathbf{x},\mathbf{u}) + V^\pi(\mathbf{x}) \\ &= r(\mathbf{x},\mathbf{u}) + \gamma\int_{\mathbb{X}} p(\mathbf{x}'|\mathbf{x},\mathbf{u})V^\pi(\mathbf{x}')d\mathbf{x}' \end{aligned} \tag{5.29}$$

Peters and Vijayakumar et al. applied natural actor-critic algorithms to LSTD(λ) introduced in [36], [141], [140], [138]. The advance function $A^\pi(\mathbf{x},\mathbf{u}) = f_w^\pi(\mathbf{x},\mathbf{u})$ is used to critic the action $\mathbf{u}$ in the state $\mathbf{x}$; and the value function is estimated by appropriating the basis $V^\pi(\mathbf{x}) = \phi(\mathbf{x})^T\mathbf{v}$. Therefore, Eq. 5.29 is

$$\nabla_\theta log\pi(\mathbf{u}_t|\mathbf{x}_t)^T\mathbf{w} + \phi(\mathbf{x}_t)^T\mathbf{v} = r(\mathbf{x}_t,\mathbf{u}_t) + \gamma\phi(\mathbf{x}_{t+1})^T\mathbf{v} + \epsilon(\mathbf{x}_t,\mathbf{u}_t,\mathbf{x}_{t+1}), \tag{5.30}$$

where $\epsilon(\mathbf{x}_t,\mathbf{u}_t,\mathbf{x}_{t+1})$ is an error term which is mean-zero referring to Eq. 5.29. Obviously, the basis of value function determines the quality of gradient.

In episode tasks, a simple path is given according to Eq. 5.30 with

$$\sum_{t=0}^{H-1}\gamma^k A^\pi(\mathbf{x}_k,\mathbf{u}_k) = V^\pi(\mathbf{x}_0) + \sum_{k=0}^{H-1}\gamma^k r(\mathbf{x}_k,\mathbf{u}_k) - \gamma^H V^\pi(\mathbf{x}_H). \tag{5.31}$$

When $\gamma < 1$ and the step number is large enough, there are final rewards in each episode $r(\mathbf{x}_{H-1},\mathbf{u}_{H-1})$, and the last term $\gamma^H V^\pi(\mathbf{x}_H)$ disappears. Furthermore, when the initial state is fixed, the value function of the initial state is a constant scale value J. Hence Eq. 5.31 is

$$\sum_{k=0}^{H-1}\gamma^k\nabla_\theta log\pi(\mathbf{u}_t|\mathbf{x}_t)^T\mathbf{w} + J = \sum_{k=0}^{H-1}\gamma^k r(\mathbf{u}_t|\mathbf{x}_t). \tag{5.32}$$

Eq. 5.32 denotes a linear regression with focusing on the parameterized vector $\mathbf{w}$ by mapping the advance function basis to trial-out rewards. A simple mathematic trick used here is extending the dimension of vector $\mathbf{w}$ to $[\mathbf{w}^T, J]^T$ [141], [139]. As a result in a non-stochastic task, based on least squares regression the natural gradient is

$$\begin{bmatrix} \mathbf{w} \\ J \end{bmatrix} = (\mathbf{\Psi}^T\mathbf{\Psi})^{-1}\mathbf{\Psi}^T\mathbf{R}, \tag{5.33}$$

where

$$\begin{aligned} \mathbf{\Psi}_i &= \left[\sum^H a_t \nabla log\pi(\mathbf{x}_t, \mathbf{u}_t)^T, 1\right] \\ \mathbf{R}_i &= \sum^H a_t r(\mathbf{x}_t, \mathbf{u}_t). \end{aligned} \tag{5.34}$$

As a result, this regression learning method is known as Peters' Episodic Natural Actor-Critic (eNAC) algorithm, shown in Algorithm 3 [141]. To speed up the eNAC, an alternative matrix inversion method is used instead of Eq. 5.33, more details can be found in [83].

Algorithm 3 Peters' Episodic Actor-Critic algorithm

```
Input: policy parameters $\theta$, learning rate $\alpha$, policy standard deviation $\sigma$;
repeat
    for $m$ episodes do
        Initialization: $\pi_\theta \leftarrow \theta$, get initial state $\mathbf{x}_0$;
        Calculate:
        policy derivatives: $\psi_k = \nabla_\theta \log \pi_\theta(\mathbf{u}_k|\mathbf{x}_k)$
        fisher matrix $\mathbf{F}_\theta = \langle(\sum_{k=0}^{H} \psi_k)(\sum_{l=0}^{H} \psi_l)^T\rangle$.
        vanilla gradient $\mathbf{g} = \langle(\sum_{k=0}^{H} \psi_\mathbf{k})(\gamma^{(H-k)}r)\rangle$.
        average reward $\overline{r} = \langle\sum_{k=0}^{H} \gamma^{H-k}r\rangle$.
        eligibility $\phi = \langle\sum_{k=0}^{H} \psi\rangle$.
        natural gradient:
        baseline $b = \mathbf{Q}(\overline{r} - \phi\mathbf{F}_\theta^{-1}\mathbf{g})$
        where $\mathbf{Q} = \frac{1}{m}(1 + \phi^T(m\mathbf{F}_\theta - \phi\phi^T)^{-1}\phi)$
        natural gradient $g_n = \mathbf{F}_\theta^{-1}(\mathbf{g} - \phi b)$
    end for
    policy update $\theta$
until $\theta$ converged
```

5.4 Manipulation Learning Experiments

In this thesis, reinforcement learning algorithms are used to learn the path of the push action discussed in Chapter 3. **The purpose of this learning is to find an optimized path for a better in-hand manipulation.** To generate push paths, RL agents are built referring to different learning algorithms. The agents are built based on stochastic policies which map current state of the push finger to push actions. In manipulation learning tasks, the state is the position of the push fingertip in Cartesian space $\mathbf{x}$, and the action is the push direction $\mathbf{P}_d$. As a result, with reinforcement learning, agents generate better push paths via interacting with its environment.

5.4.1 Learning Rewards

In manipulation learning experiments, reinforcement learning algorithms are carried out to train the agent through interacting with the manipulation simulator built in section 5.2.3. In each interaction, once an action is generated and applied to the simulator, immediate rewards are estimated. With these rewards, the agent modifies its parameters with reinforcement learning algorithms. Since the critic rewards can only be made after a full path generated, this manipulation learning problem belongs to the episodic category. The episodic reward is

$$r = \int \mathbf{k}^T \mathbf{r}_{step}(\mathbf{x}_t) dt + r_{end}, \tag{5.35}$$

where term $\mathbf{r}_{step}(\mathbf{x}_t)$ refers to a reward vector at the state $\mathbf{x}_t$; term $\mathbf{k}$ denotes a weight vector, and term r_{end} is the final reward which only can be obtained at the end of episodes. The reward vector $\mathbf{r}_{step}(\mathbf{x}_t)$ consists of two parts: visual reward r_{stepV} and haptic reward r_{stepH} with $\mathbf{r}_{step}(\mathbf{x}_t) = [r_{stepV}(\mathbf{x}_t)\ r_{stepH}(\mathbf{x}_t)]^T$. The weight vector $\mathbf{k}$ assigns the weights between visual and haptic rewards. Therefore, in one episode, the integral reward $\int \mathbf{k}^T \mathbf{r}_{step}(\mathbf{x}_t)$ is a path reward which defines a cost to criticize the trajectory. In other words, it makes the push action reach the target as efficiently as possible.[9] In manipulation tasks, one critic aspect is the end position of the path. It highly relates to whether this push task meets the expected manipulation target. Hence, a final reward is set as 10 times of final visual reward with $r_{end} = 10 r_{stepV}(\mathbf{x}_{end})$.

More practically, in this experiment, a discrete sum is used instead of the continuous integral term in Eq. 5.35, and a discount factor is used to keep the path reward away from infinity. As a result, the episode reward is

$$r = \sum_{k=1}^{H-1} \gamma^{H-k} \mathbf{k}^T \mathbf{r}_{step}(\mathbf{x}_t) + 10 r_{stepV}(\mathbf{x}_H), \tag{5.36}$$

where H is the number of steps in one episode. In this learning experiment, the step number is set to 12, the decay rate is 0.8. Term $\mathbf{k}$ gives us a convenient

[9]Here the 'efficiently' is defined by the user. In this chapter, the efficiently refers to the large push force during the manipulation.

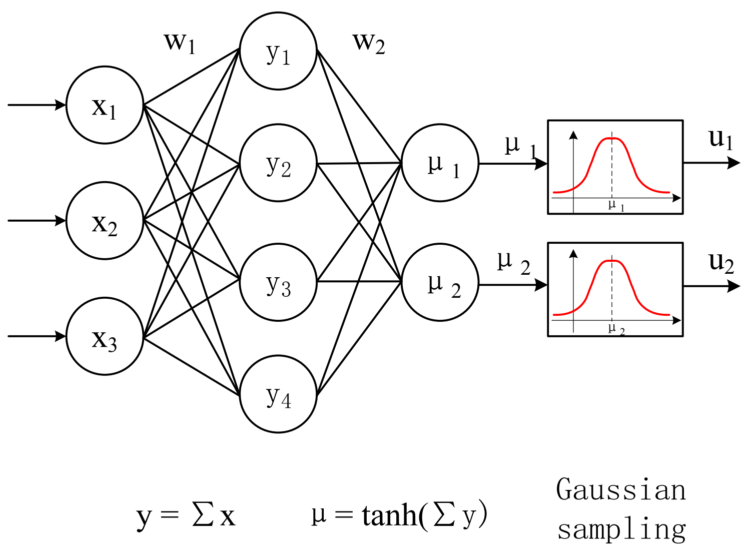


Figure 5.12: A three-layer ANN for Gaussian stochastic policy.

way to switch the learning critic between visual-only and visual-haptic rewards. For example, in visual only learning, $\mathbf{k} = [1\ 0]^T$. In visual-haptic learning, $\mathbf{k} = [1\ 0.01]^T$. Therefore, in the following sections, experiments have been performed to compare the learning performance between visual-only rewards and visual-haptic rewards with different stochastic learning algorithms.

5.4.2 Learning with Visual-Only Rewards

In this section, learning experiments are carried out based on visual-only rewards. Thus term $\mathbf{k}$ in Eq. 5.36 is set to $[1,\ 0]^T$. Williams' Episodic REINFORCE algorithm and Peters' episodic natural actor-critic algorithm are applied to learn the push paths.

Williams' Episodic REINFORCE with Visual-Only Rewards

First, Williams' Episodic REINFORCE algorithm is implemented. With this algorithm, we trained the RL agent to optimize the push path. The stochastic policy is built with a three-layer ANN, with 3 input nodes, 5 hidden neurons, and 2 output. In the input layer, the input is the position of the push fingertip in Cartesian space. In the hidden layer, the hyperbolic function is used. With these two layers, this policy is deterministic but stochastic. As introduced in [178], in the output layer, an additional layer with Gaussian units is used to generate the stochastic output. In other words, the real output is sampled from the Gaussian distributions:

$$\mathbf{u} = \mathcal{N}(\mathbf{u}_{neuron}, \sigma^2), \tag{5.37}$$

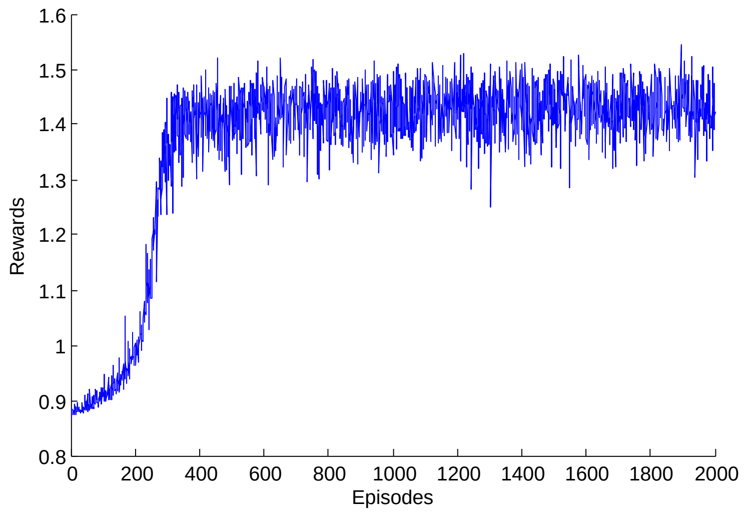


Figure 5.13: Visual rewards with Williams' REINFORCE algorithm.

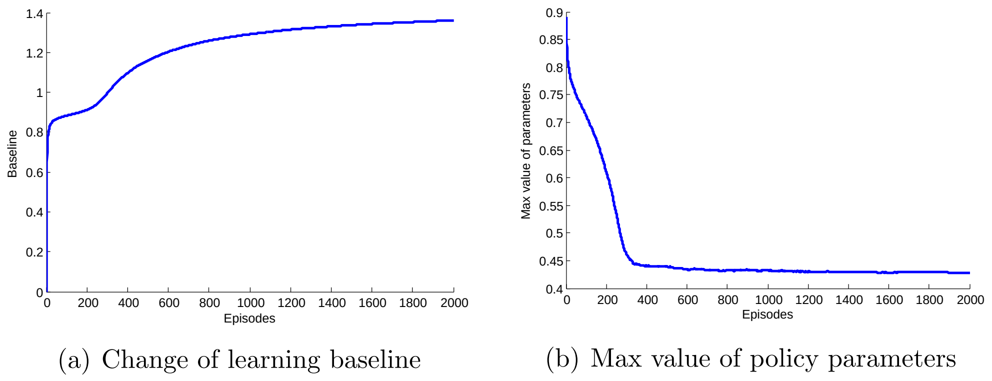
where $\mathbf{u}$ is sampled actions; $\mathbf{u}_{neuron}$ is the output of neural networks, and σ is the standard deviation. They all have dimension n_u in this experiment. A full stochastic ANN diagram is shown in Fig. 5.12. The usage of the Gaussian sampling layer is mainly based on two considerations. On the one hand, the deterministic part of the ANN generates expected value for actions; on the other hand, the stochastic layer of the ANN generates stochastic actions which contribute the exploration in the learning process with regard to standard deviation σ.

This stochastic ANN in Fig. 5.12 is the core part of Williams' Episodic REINFORCE algorithm shown in Algorithm 2. It is the stochastic policy $\pi_\theta(\mathbf{u}|\mathbf{x})$ which generates paths by mapping the state to push actions. The details of learning parameters are shown in Table 5.1.

Table 5.1: Episodic REINFORCE algorithm parameters

Parameters	Notation	value
State dimensions	n_s	3
Action dimensions	n_u	2
Hidden layers units	n_h	8
Learning rate	α	0.005
Discount factor	γ	0.8
Policy standard deviation	σ	0.4
Steps in one episode	H	12
Maximum episode number	n_{ep}	2000

After training, the learning rewards are shown in Fig. 5.13. At the beginning, the initial reward is about 0.87. After around 310 episodes, the reward increases fast to 1.42; after that, it oscillates around 1.43. These oscillations are from the



(a) Change of learning baseline (b) Max value of policy parameters

Figure 5.14: Learning results with Williams' REINFORCE algorithm.

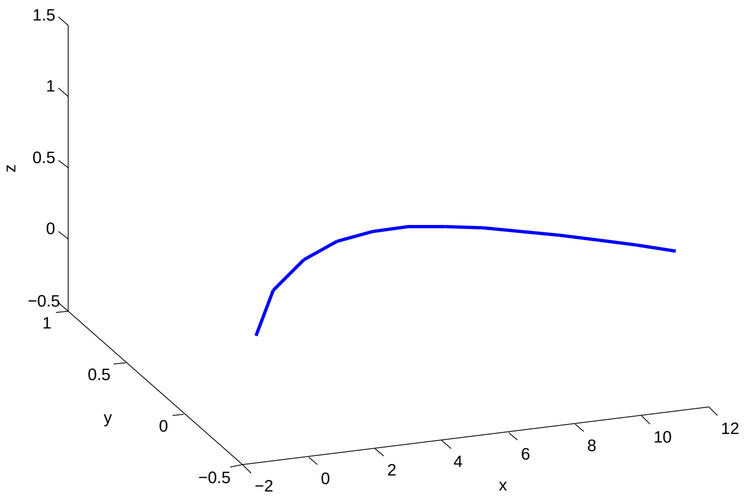


Figure 5.15: A learned path from REINFORCE algorithm.

constant deviation of Gaussian sampling. In this experiment, the Gaussian deviation is chosen to control the distribution of the generated actions from stochastic policy $\pi_\theta(\mathbf{u}|\mathbf{x})$. In other words, a constant deviation means the learning agent keeps exploring from start to the end.

In Fig. 5.14, other learning results are illustrated. Fig. 5.14(a) represents the change in the baseline b during the learning process. At the beginning, it starts at 0; then it increases up to 0.8 dramatically within first several episodes; at last, it converges to 1.4 in the later episodes. In Fig. 5.14(b) the max value of policy parameters is shown. At the beginning, policy parameters are given randomly sampled between -1 to 1; and then in first 300 steps, the max value decreases to 0.45.

In Fig. 5.15, an optimized path is given after training. This path starts around point (0, 0, 0), and ends at point (11.92, -0.24, 0.41) with a curved path. It is worth noting that the start points are given randomly around point (0, 0, 0) in each episode; and the policy parameters are also initialized randomly. Therefore, sometimes the learning results are given slightly different.

Peters' Episodic Natural Actor-Critic with Visual-Only Rewards

Peters' Episodic Natural Actor-Critic algorithm is also implemented to learning the push path. As shown in Algorithm 3, the feature state used here is $\mathbf{x}_f = [x_1^2,\ x_2^2,\ x_3^2,\ x_1x_2,\ x_1x_3,\ x_2x_3,\ x_1,\ x_2,\ x_3,\ 1]^T$, where $\mathbf{x} = [x_1,\ x_2, x_3]^T$ is the state. A Gaussian policy is used:

$$\pi(\mathbf{u}|\mathbf{x}) = \mathcal{N}(\theta^T\mathbf{x}, \sigma^2). \tag{5.38}$$

In this algorithm, it should be known previously that how many episodes are required to accumulate the 'sufficient' roll-outs before updating policy parameters. On the one hand, this number should be large enough to indicate the unbiased estimation of the state distribution under the current policy; on the other hand, a large number of the roll-outs slows down the learning process. Therefore, in this experiment, the number of roll-outs is set to 50 episodes. Other learning parameters are shown in Tab. 5.2.

Table 5.2: Episodic Natural Actor-Critic parameters.

Parameters	Notation	value
State dimensions	n_s	3
State feature dimensions	n_{sf}	10
Action dimensions	n_u	2
Learning rate	α	0.5
Discount factor	γ	0.8
Policy standard deviation	σ	0.2
Steps in one episode	n_{step}	12
Maximum episode number	n_{ep}	1000

The learning reward is shown in Fig. 5.16. At the beginning, the initial reward is about 0.7. After 301 episodes, the reward increases very fast up to 1.24; and then the reward slowly converges to 1.4 in the later episodes. In this learning process, the reward increases with few fluctuations. These fluctuations come from the biased estimation of the cost function. Although 50 paths are sampled, it is still not large enough for a precise unbiased estimation. However, as the result shows, the learning rewards can be considered as converged. Hence, 50 roll-outs are used in this experiment.

In Fig. 5.17, a learned path is given after training. This path starts around the point $(0,\ 0,\ 0)$, and end at the point $(11.84,\ 1.17,\ -1.19)$ with a curved path.

5.4.3 Learning with Visual-Haptic Rewards

More than visual information, in this section haptic information is involved in the learning process. The visual-haptic rewards are adopted. As described in Eq. 5.36, term $\mathbf{k}$ is set to $[1\ 0.005]^T$.

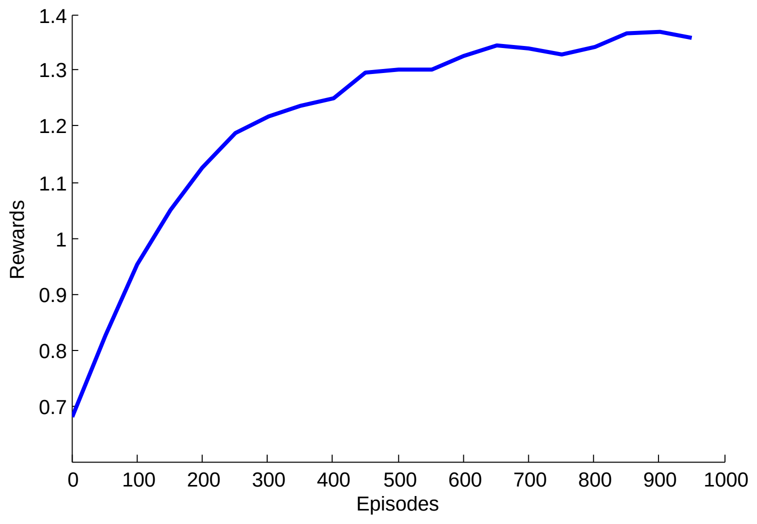


Figure 5.16: Visual rewards in Peters' episodic natural actor-critic algorithm.

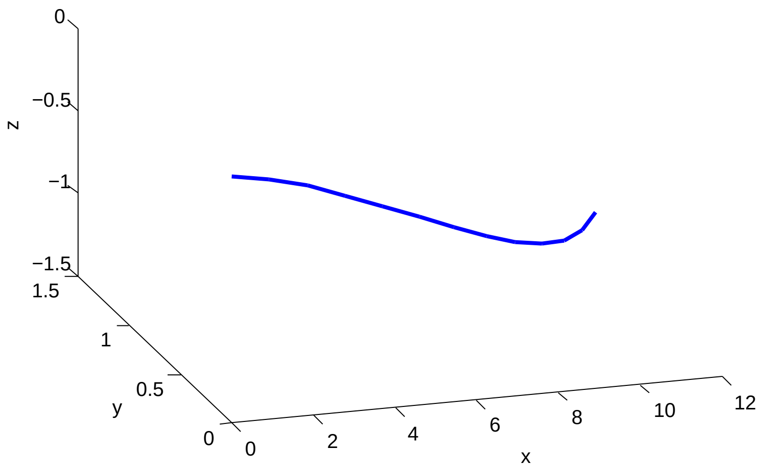


Figure 5.17: A learned path from Peters' episodic natural actor-critic algorithm.

Williams' Episodic REINFORCE with Visual-Haptic Rewards

Based on visual-haptic rewards, the learning result is shown in Fig. 5.18 with the same learning algorithm and parameters as in Table 5.1. At the beginning, the episode reward is about 0.25. After 170 episodes, the reward increases fast to 2.11; and in the later episodes, the reward fluctuates around 2.1.

Like visual only learning reward, other learning results are shown in Fig. 5.19. Fig. 5.19(a) shows the change in baseline b. It starts from 0, increases up to 1.1 within first several episodes, and converges to 2.5 in the following episodes. Fig. 5.19(b) shows the max value of policy parameters.

In Fig. 5.20, the path is given after training. This path starts around the point (0, 0, 0) and ends at the point (11.99, −0.51, 0.54) with a curved path.

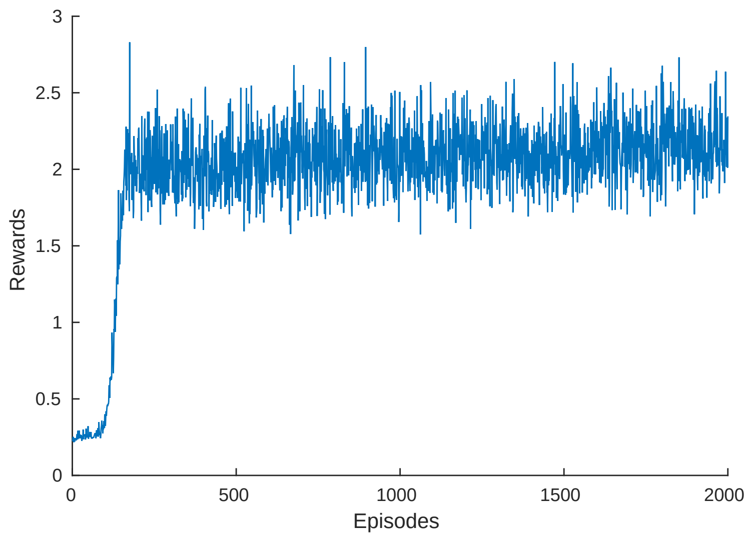
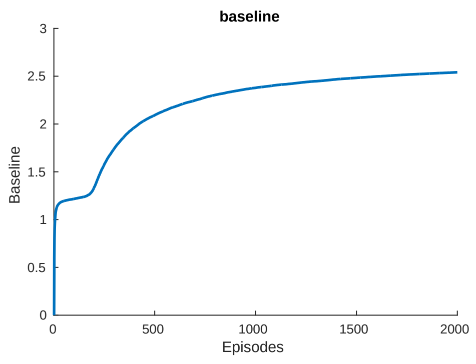
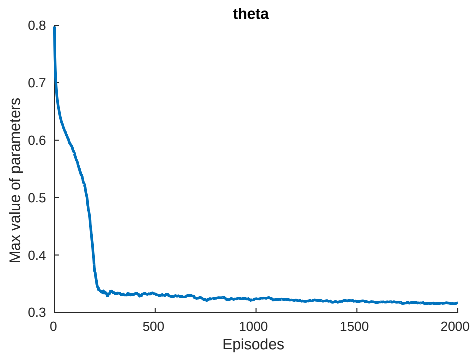


Figure 5.18: Visual-haptic rewards with Williams' REINFORCE algorithm.



(a) Change of learning baseline



(b) Max value of policy parameters

Figure 5.19: Visual-haptic learning results with Williams' REINFORCE algorithm

Peters' Episodic Natural Actor-Critic Based on Visual Haptic Rewards

Based on the visual-haptic rewards, learning result is shown in Fig. 5.21, with the same learning algorithm and parameters as in Table 5.2. At the beginning, the episode reward is about 1.6. After 151 episodes, the reward increases fast to 2.17; and after 451 episodes, rewards fluctuate around 2.2.

In Fig. 5.22, a learned path is given after training. The path also starts around the point $(0,\ 0,\ 0)$ and ends at the point $(11.97,\ 0.97,\ -0.31)$ with a curved path.

5.4.4 Discussion of the Learning Results

In these learning experiments, different rewards have been implemented in manipulation learning process. Comparative results of the learning speed are shown in Fig. 5.23. In Fig. 5.23, the plotted learning speed is from different learning algorithms. Obviously, Peters' Episodic Natural Actor-Critic algorithm is a

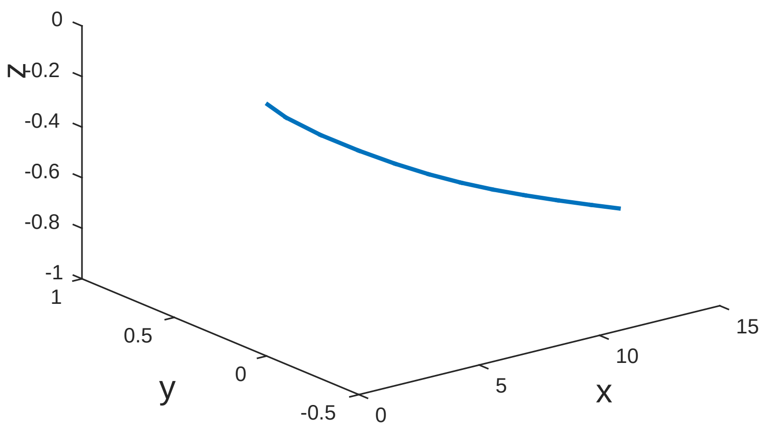


Figure 5.20: A learned path from Williams' REINFORCE algorithm with visual-haptic rewards.

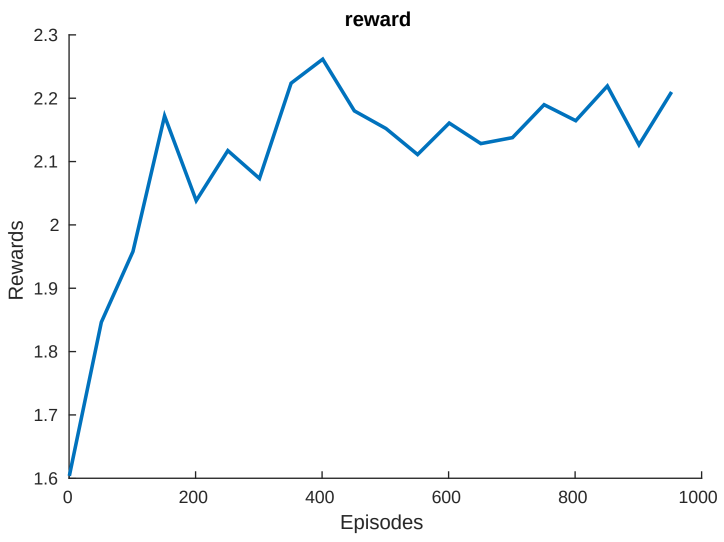


Figure 5.21: Visual-haptic rewards with Peters' Episodic Natural Actor-Critic.

little faster than Williams' Episodic REINFORCE algorithm in both learning experiments with visual-only rewards and visual-haptic rewards, but not too much. Besides, a more distinct difference is the use of different rewards. The learning with the visual-haptic rewards is much faster than the learning with the visual-only rewards, both in Williams' and Peters' algorithms.

In their learning results, Peters' algorithm does not show a distinct advantage over the learning speed comparing to Williams' algorithm. It comes from the two aspects. The first one is Peters' algorithm supposes to use the natural gradient to speed up the learning process instead of the standard gradient. However, in this manipulation tasks, the state is in Cartesian space, which makes no difference between the natural gradient and the standard gradient. As a result, there is no advantage in the aspect of the natural gradient for Peters' algorithm. The other aspect is Peters' algorithm does not update the parameters of its policy

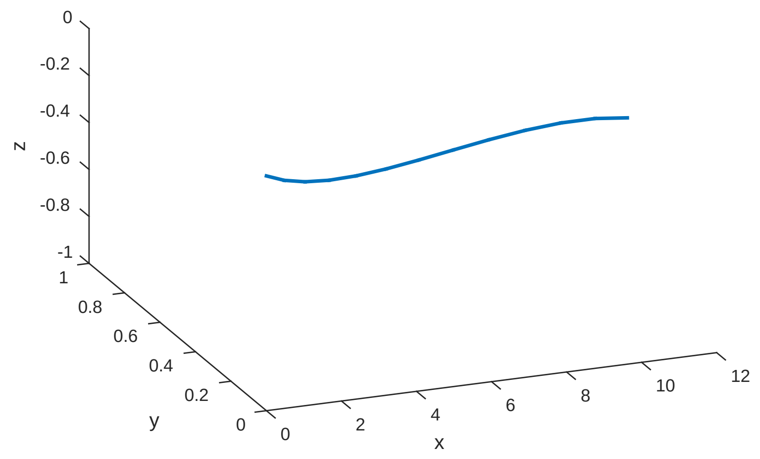


Figure 5.22: A learned path from Peters' Episodic Natural Actor-Critic algorithm with visual-haptic rewards.

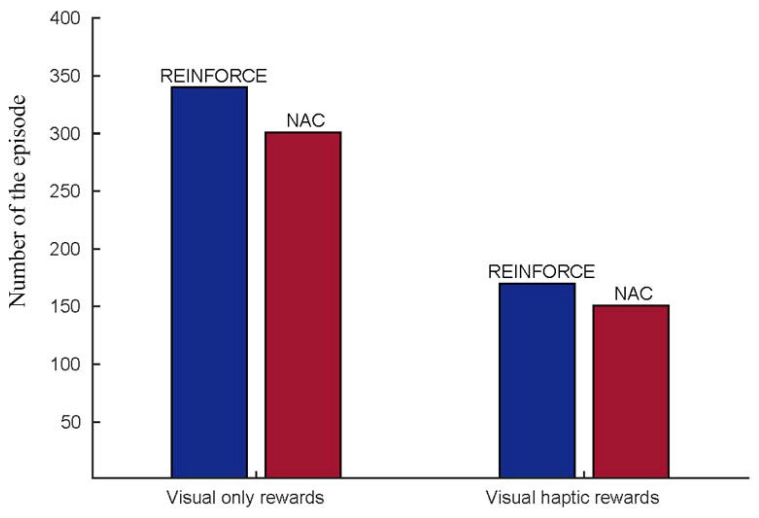


Figure 5.23: Comparison of the learning speed in different learning experiments. The 'REINFORCE' refers to the result from Williams' Episodic REINFORCE algorithm, and the 'NAC' refers to the result from Peters' episodic natural actor-critic algorithm.

until accumulating enough roll-outs but Williams' algorithm updates its policy parameters in the end of each roll-out. In other words, in Peter's algorithm 150 episodes are performed; however only 3 updates are performed in this learning process.

On the other hand, compared with the results based visual only rewards and visual-haptic rewards, the using haptic involved rewards speeds up the learning speed in both learning algorithms. Therefore it can be concluded that the use of multimodal rewards shows advantages in manipulation learning process compared to the use of the unimodal reward.

Chapter 6

Conclusion

In this chapter, firstly, the main work of this research is summarized. Besides, the limitation of this research is presented and discussed. Finally, the future outlook is given in the end of this chapter

6.1 Summary

With their ability to adapt to human tools, robots with anthropomorphic hands possess great potential as employees in domestic environments, where they could facilitate various domestic tasks like tidying up rooms, serving dinner, washing dishes, etc. In-hand manipulation is a distinctive skill in anthropomorphic hands. Although a lot of research has been done on in-hand manipulation, it still poses a challenge to robotics. As discussed in chapter 1, the three main problems to be solved in robotic manipulation are a large number of joints, complex interaction models, and limited capabilities of perception.

In an anthropomorphic hand, there are more than ten joints. Control of this large number of joints is not easy. Most robotic in-hand manipulations are carried out with synthesis models; however, the interaction constraints on the contact are hard to model in such a redundant system. Moreover, existing models are labor intensive and error-prone. The two most widely used sensing channels in robotics are visual and haptic sensing. However, the use of them is limited in robotic in-hand manipulations. The visual sensing is often disturbed by a noisy background and occluded by fingers; tactile sensing is often limited by the sensors' size, density, resolution, etc.

To deal with these challenges, the black box manipulation concept has been proposed in section 1.3.2. This black box consists of one object and support fingers. In this black box, an unknown object is assumed to be located on an elastic surface consisting of the support fingers. In this system, only the push finger is controlled actively. It receives the push commands and pushes the object against the other fingers. Since the push finger has little prior knowledge of the system, the object and support fingers are considered as a black box for the push finger. What the push finger does is carry out given push commands and the robot observes the visual and haptic feedback from the black box to evaluate its push actions. Without any knowledge of this black box, the robot cannot manipulate objects successfully. Hence to acquire sufficient knowledge, a process called haptic exploration has been introduced in section 1.3.2. In this haptic exploration, the robot slightly pushes the object in different directions and estimates the interaction state by means of the haptic feedback.

With the black box, the process of in-hand manipulation is transferred into rolling the object onto an elastic surface. In chapter 3, in-hand manipulation analytic models have been proposed. According to different elasticity of the surface, we use two support fingers: a fixed support finger and a spring support finger. The fixed support finger, whose position is fixed with a position controller, acts as a pivot around which the object rotates. The spring support finger performs as an elastic spring. It is controlled passively by a stiffness controller. This finger presses the object against the other fingers to ensure the stability of the system and helps to rotate the object with proper contact force. More importantly, an error transfer matrix is given, which proves that position errors on different fingers can be compensated by one spring support finger. With the combination of different support fingers, different manipulation models are obtained, as discussed in section 3.2. Two of them have been further discussed. With one fixed support finger, the single support model has been proposed in section 3.3 for two fingers manipulation. With the combination of one fixed support finger and one spring support finger, the hybrid support model is proposed in section 3.4 for three fingers manipulation. Finally, we developed an enhanced manipulation model to perform final manipulations in section 3.5.

In order to verify the proposed method and models, in-hand manipulation experiments have been conducted in chapter 4. They have been carried out on a robot in-hand manipulation platform consisting of a Shadow Hand, a Kuka arm, BioTac sensors, and a digital camera. The in-hand manipulation task was to rotate a rectangular box. First, a repeatability experiment has been conducted to support the idea of automatic repeating push. Then we have carried out a multi push distance experiment to discover the relation between the push distance and the haptic and visual features. This has been followed by a multi-directional manipulation experiment to illustrate the relation between the manipulation results and push directions. Then a hybrid support manipulation experiment has been conducted to verify the proposed hybrid support model. And then a robustness experiment followed to illustrate the robustness of the proposed manipulation method and to support the conclusion about the error transfer matrix. Finally, enhanced manipulation and comparative experiments have been carried out to illustrate the great merits of our method. In these experiments both visual and haptic feedback has been collected; moreover, their relation has been thoroughly examined. Based on the experimental results, we conclude that haptic sensing has the potential to take the place of visual sensing in these manipulation experiments. Moreover, it offers more information on the interaction than visual sensing. Besides, in these experiments, especially in the multi-directional manipulation experiment, enough knowledge has been acquired to use for further in-hand manipulations in other tasks. This supports the idea of haptic exploration.

To automatically learn and improve the in-hand manipulation skills, reinforcement learning (RL) methods have been adopted in chapter 5. Before learning, in section 5.2 a manipulation simulator was constructed from Radial Basis Function Networks (RBFNs) for the purpose of taking the place of the real manipulation system. The RBFNs are trained by the real data collected from the density push experiment in section 5.2.2. Two stochastic RL algorithms, Williams' Episodic REINFORCE and Peters' Episodic Natural Actor-Critic, have been adopted in this thesis. Therefore, stochastic RL agents are built to map the current state to push commands. Through interacting with the manipulation simulator, the RL agent improves step by step. Besides, in section 5.4.1, the learning experiments were based on two different rewards: visual only rewards and visual-haptic rewards. The learning results prove that our proposed learning architecture is feasible; moreover, using the multimodal (visual and haptic) rewards can speed up the learning process dramatically compared to using unimodal (visual only)rewards.

To summarize, the problems presented in section 1.3.1, "Objectives", have been solved in this thesis. Specifically,

- Robust in-hand manipulations have been achieved by an anthropomorphic robotic hand, as shown in the experiments in chapter 4; in particular, the robustness attribute has been verified in the experiment in section 4.7.
- In the multi-directional manipulation experiment in section 4.5, the robot automatically explores the in-hand object through haptic exploration to acquire sufficient manipulation knowledge without possessing any information about the object in advance. Moreover, further manipulations were carried out based on the acquired knowledge, such as those in sections 4.6 and 4.7.
- Robust and efficient in-hand manipulation models have been proposed in chapter 3. These models reduce the complexity of the in-hand manipulation dramatically. They explain the in-hand manipulation results and the visual and haptic relations in our experiments.
- The robot has learned and improved its in-hand manipulation skill correctly through interacting with its environment in chapter 5.

6.2 Limitation and Discussion

Even though robust in-hand manipulation has been achieved in this thesis, there are still some problems to be solved.

The first one is the manipulation efficiency. In the process of haptic exploration, more than 50 manipulations have been carried out to collect the relevant knowledge. This takes too much time and makes the system inefficient in achieving proper manipulations.

The second problem is that the object's rotated angle is small in the experiments. It lies within the limitation of our robotic hand. The thumb plays an important role in in-hand manipulations, especially its joints THJ1 and THJ2. However, in our robotic hand, the thumb joint TH1 has been removed for the tactile sensor. And the joints THJ2 and THJ3 only have a range of $-15°$ to $15°$. It is very small compared to that of humans (about $90°$). Moreover, other fingers' first joints have been removed for the tactile sensors. This reduces the dexterity of the hand.

The third problem is object's recognition. Although AprilTag system is used to track the object's position and orientation, as an application-oriented method, the object should be more generalized. The real application scenario requires an object to be recognized and tracked in any environment without special marks on it. Therefore, a robust vision tracking system is required to meet the strict application requirements.

6.3 Future outlook

Due to time constraints, not all ideas and in-hand manipulation experiments were able to be carried out within this Ph.D. research. The future work will involve the following problems.

3D manipulation

For robotic manipulation skills, one important improvement would be extending our method to 3D in-hand manipulations. In 3D tasks, the manipulation tasks will not be limited to rotational manipulation but will cover translational manipulation as well. Any object's motion could be achieved by one rotational manipulation and one translational manipulation. The rotational manipulation can be completed by the push and support method proposed in this thesis, and the translational manipulation can be easily achieved with the virtual frame method. To verify the performance of 3D in-hand manipulations, more real 3D in-hand manipulation experiments should be performed.

More fingers in a grasp

More robotic grasp configurations should be tried. In this thesis, only two-finger grasps and three-finger grasps are adopted in the manipulation experiments. However, in human's manipulations, five fingers are often used in a grasp. In the proposed manipulation method, there can be four support fingers in a five-finger grasp. With well-defined stiffness of these support fingers, the push and support method is still feasible for the success of the in-hand manipulations. Therefore, the push manipulation with four support fingers is one of the future research topics.

Haptic efficient

The process of the haptic exploration proposed in this thesis is inefficient since the robot explores all the possible push directions exhaustively. A better exploration method can be using the haptic and visual cues to find the best push direction efficiently. In [142], Pinto and Gupta modeled the grasping problem as an 18-way binary classification over 18 angle bins. Similarly, in robotic in-hand manipulation problems, the push directions of the haptic exploration also can be a problem of a binary classification. According to the results of this thesis, the haptic data is suggested to be used to select the most efficient push direction in the haptic exploration.

Learning with deep learning methods

Our learned in-hand manipulation skill could be improved with better learning algorithms. Some modern learning methods are worth a try, like Deep Reinforcement Learning. In [121], a Deep Q-Network (DQN) algorithm was used in the learning of playing classic Atari games. A deep learning model was proposed to learn the control policies, which were directly learned from the high-dimensional sensory input (pixels of the images). In robotic in-hand manipulations, the in-hand system is usually represented by a high dimensional state vector which includes high dimensional haptic data, continuous visual tracking data, high dimensional joints' state, etc. In dealing with this high dimensional state, this deep learning method has great potential in learning the skill of the robotic in-hand manipulation.

Besides, the interaction models proposed in this thesis can also benefit robot-human cooperation tasks, and the proposed haptic sensing method offers some insight into other contact-related research.

Appendix A

Nomenclature

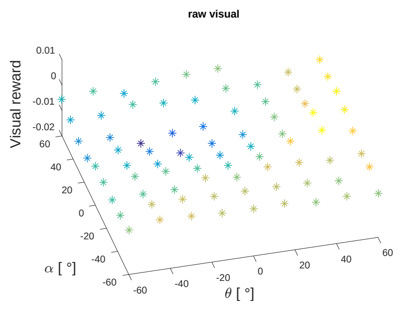
Table A.1: Important notations in this thesis

Notation	Definition
$\mathbf{P_d}$	Push direction
$\mathbf{P_l}$	Push distance
ξ_o	Twist of the object
$\mathbf{v}_o$	Velocity of the object
ω_o	Angular velocity of the object
$\mathbf{K_s}$	Support stiffness vector
$\mathbf{pk}$	Haptic feature
pk_i	The ith element of haptic feature
P_{DC}	Static pressure on the fingertip
δ	Object's rotated angle in the expected direction
R_H	Haptic result for multi-directional manipulations
R_V	Visual result for multi-directional manipulations

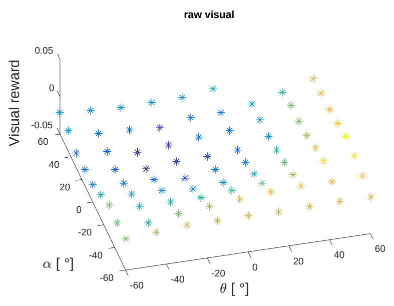
Appendix B

Density Push Result

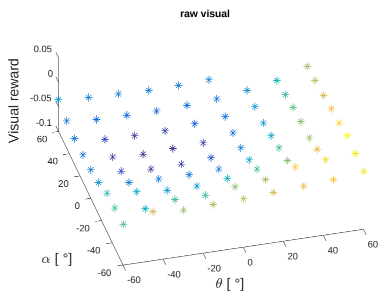
According to section 5.2.2, the training data in density push manipulation is plotted in Fig. B.1 and B.2 respectively.



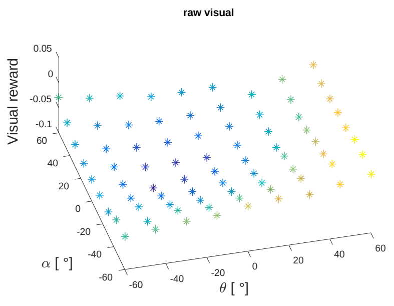
(a) $d = 2\ mm$



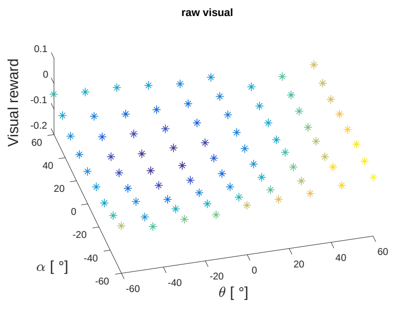
(b) $d = 4\ mm$



(c) $d = 6\ mm$

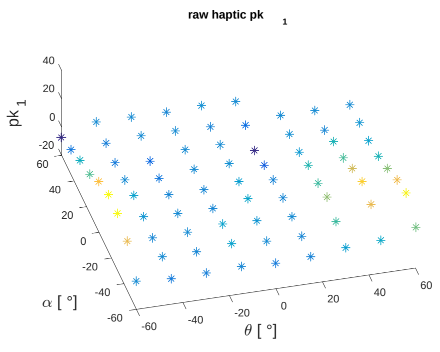


(d) $d = 8\ mm$

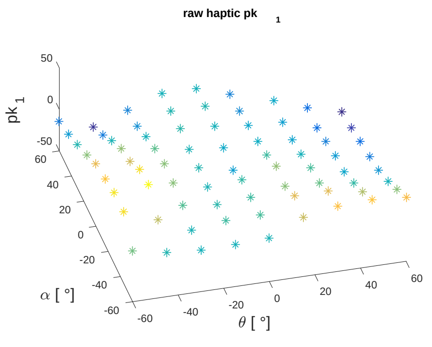


(e) $d = 10\ mm$

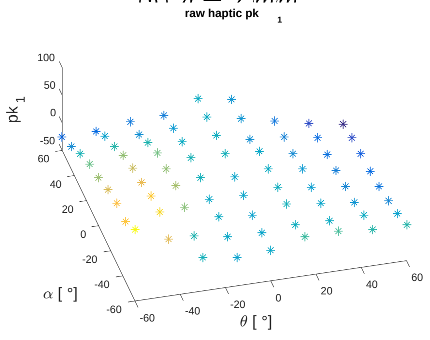
Figure B.1: Visual reward in the density push experiment.



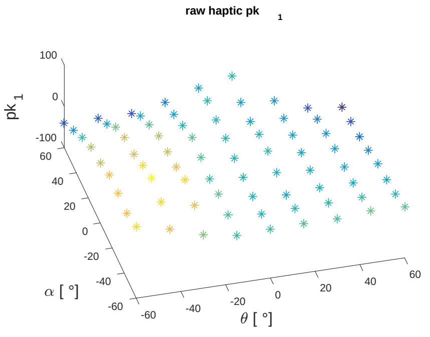
(a) $d = 2\ mm$



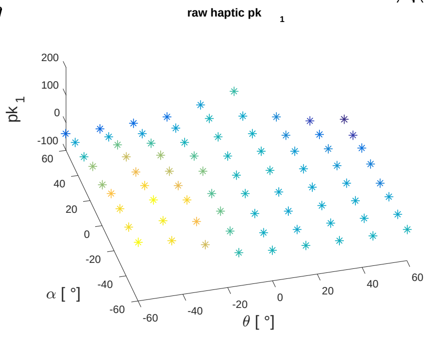
(b) $d = 4\ mm$



(c) $d = 6$



$= 8\ mm$



(e) $d = 10\ mm$

Figure B.2: Haptic results in the density push experiment. The first element of the haptic feature, pk_1, is shown in this figure.

Appendix C

Approximate with MLP neural network

Beside the RBFN, a multilayer perceptron (MLP) neural network is also used to approximate the manipulation simulator. The result is shown in Fig. C.1. In Fig. C.1, the general shape is similar to the RBFN's result in Fig. 5.4(f); however, the output of network raises in the marginal area. This raise is from the slipping during the manipulation. When the finger pushes in the directions of this area, the object slips on the fingertips. This causes this irregular result on the edge of the surface in Fig. 5.4(f). Compared to the MLP result, the RBFN show advantages over dealing with this kind of noise. The output of the neural networks decreases when its inputs are away from the kernels. This attributes improve the performance of the simulator by reducing the output in marginal area. Therefore, in this thesis the RBFN is chosen to approximate visual and haptic results.

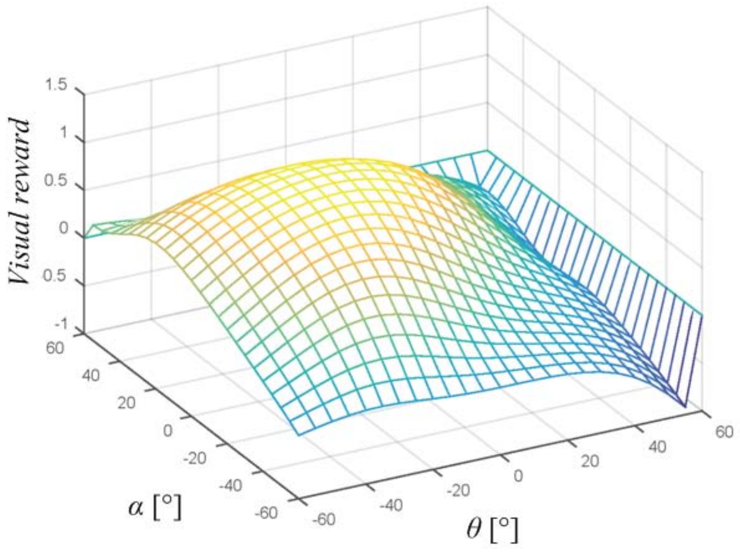


Figure C.1: Visual result approximated with an MLP neural network.

Appendix D

Acknowledgments

The work presented in this thesis was carried out in the institute of Technical Aspects of Multimodal Systems (TAMS), Department of Computer science, University of Hamburg. It was mainly supported by Cross-Modal Interaction in Natural and Artificial Cognitive Systems (CINACS DFG IGK 1247).

Along with this research road, I have met wonderful people who played a very important role in achieving this research. First of all, I would like to thank Prof. Dr. Jianwei Zhang, my principal adviser, for his belief in me and for giving me the best possible start into advanced robotics. I would not finish this research without his initiation, continuing encouragement, coordination, supervision and understanding help. It is my honor to be a part of his group. Besides, I would like to thank my another advisor Prof. Dr. Stefan Wermter for reviewing my thesis. His professional suggestions are really helpful. Jianwei and Stefan are wonderful people, I learned from you not only aspects of the research on Robotics, but also life lessons, which helped me to become a better researcher and man.

Secondly, I would like to thanks my parents, Chenlong, Aihua and my brother Jianjun for their understanding and encouraging in this research journey. And I would like to think my friends, Lucia Chen, Dr. Xin Dong, Dr. Özge Alacam, Haojun Guan, Xiaoxiang Zheng, Zheng Deng, Tingyu Li for encouraging me and checking the English of this thesis.

I did enjoy and love working in German especially in Hamburg. It was my pleasure to work in TAMS, where I met really nice, unforgettable and endearing people. Special thanks to Tatjana Tetsis and Wiebke Eggers for their friendly support. Thanks to Florens Wasserfall for printing object models. Thanks to Dr. Meng Chen, Dr. Andreas Mäder, and Dr. Norman Hendrich for their technical support. And other thanks goes to my colleagues and friends: Dr. Hannes Bistry, Dr. Gang Chen, Dr. Bo Sun, Dr. Sebastian Rockel, Dr. Zhiyi Fang, Dr. Tao Li, Dr. Xiaofeng Xiong, Weipeng He, Jinpeng Mi, Hongzhuo Liang, et al.

Appendix E

Publications

Junhu He and Jianwei Zhang. In-hand haptic perception in dexterous manipulations, Science China Information Sciences, 57(12): 1-11, 2014.

Junhu He and Jianwei Zhang. Push resistance in in-hand manipulation, in Proceedings of the International Conference on IEEE International Conference on Intelligent Robots and Systems (IROS), pages 2488-2493. IEEEE, 2014.

Junhu He, Sicong Pu, and Jianwei Zhang. Haptic and visual perception in In-hand Manipulation systems, in Proceedings of the International Conference on Intelligent Robotics and Biomimetics (ROBIO), pages: 303-308. IEEE, 2015.

Junhu He, Sicong Pu, Jianwei Zhang. In-hand Manipulation with Fixed and Spring Support Fingers. in Proceedings of the third International Conference on Cognitive Systems and Information Processing (ICCSIP), 2016.

Bibliography

[1] Adaptive robot grippers. http://robotiq.com/products/. Accessed: 2015-03-26.

[2] Apriltags. https://april.eecs.umich.edu/wiki/AprilTags. Accessed: 2016-03-07.

[3] Barrett Technology, Inc. - Products - BarrettHand. http://www.barrett.com/robot/products-hand.htm. Accessed: 2015-03-26.

[4] Biotac sensor technology. http://www.syntouchllc.com/Products/BioTac/. Accessed: 2016-03-07.

[5] Dexterous Hand. http://www.shadowrobot.com/products/dexterous-hand/. Accessed: 2015-03-26.

[6] Kinova | Reach your potential. http://kinovarobotics.com/. Accessed: 2015-03-26.

[7] Kuka youbot. http://www.kuka-labs.com/de/service_robotics/research_education/youbot/. Accessed: 2015-03-26.

[8] Overview | Willow Garage. https://www.willowgarage.com/pages/pr2/overview. Accessed: 2015-03-26.

[9] Robonaut. http://robonaut.jsc.nasa.gov/R1/sub/hands.asp. Accessed: 2015-03-26.

[10] SDH servo-electric 3-Finger Gripping Hand: SCHUNK Mobile Greifsysteme. http://mobile.schunk-microsite.com/en/produkte/produkte/sdh-servo-electric-3-finger-gripping-hand.html. Accessed: 2015-03-26.

[11] Servo-electric 5-Finger Gripping Hand SVH: SCHUNK Mobile Greifsysteme. http://mobile.schunk-microsite.com/en/produkte/produkte/servo-electric-5-finger-gripping-hand-svh.html. Accessed: 2015-03-26.

[12] Shadow hand. https://www.shadowrobot.com/products/dexterous-hand. Accessed: 2016-03-07.

[13] Pieter Abbeel, Adam Coates, Morgan Quigley, and Andrew Ng. An application of reinforcement learning to aerobatic helicopter flight. *Advances in Neural Information Processing Systems*, 19:1, 2007.

[14] Edward Adelson. Tactile sensor using elastomeric imaging, 2015. US Patent RE45,578.

[15] Peter Allen and Paul Michelman. Acquisition and interpretation of 3-d sensor data from touch. In *Proceedings of the Workshop on Interpretation of 3D Scenes*, pages 33–40. IEEE, 1989.

[16] Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10(2):251–276, 1998.

[17] Brenna Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469–483, 2009.

[18] Suguru Arimoto, Pham Thuc Anh Nguyen, Hyun-Yong Han, and Zoe Doulgeri. Dynamics and control of a set of dual fingers with soft tips. *Robotica*, 18(01):71–80, 2000.

[19] Tamim Asfour, Kristian Regenstein, Pedram Azad, Joachim Schröder, Alexander Bierbaum, Nikolaus Vahrenkamp, and Rüdiger Dillmann. Armar-iii: An integrated humanoid platform for sensory-motor control. In *Proceedings of 6th IEEE International Conference on Humanoid Robots*, pages 169–175. IEEE, 2006.

[20] Ji-Hun Bae, Sung-Woo Park, Doik Kim, Moon-Hong Baeg, and Sang-Rok Oh. A grasp strategy with the geometric centroid of a groped object shape derived from contact spots. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3798–3804. IEEE, 2012.

[21] Ji-Hun Bae, Sung-Woo Park, Jae-Han Park, Moon-Hong Baeg, Doik Kim, and Sang-Rok Oh. Development of a low cost anthropomorphic robot hand with high capability. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 4776–4782. IEEE, 2012.

[22] Andrew Bagnell and Jeff Hneider. Autonomous helicopter control using reinforcement learning policy search methods. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 1615–1620. IEEE, 2001.

[23] Tim Baier-Lowenstein and Jianwei Zhang. Learning to grasp everyday objects using reinforcement-learning with automatic value cut-off. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 1551–1556. IEEE, 2007.

[24] Andrew Barto. *Reinforcement learning: An introduction.* MIT press, 1998.

[25] Jonathan Baxter and Peter Bartlett. Direct gradient-based reinforcement learning. In *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*, volume 3, pages 271–274. IEEE, 2000.

[26] Yasemin Bekiroglu, Janne Laaksonen, Jimmy Alison Jorgensen, Ville Kyrki, and Danica Kragic. Assessing grasp stability based on learning and haptic data. *IEEE Transactions on Robotics*, 27(3):616–629, 2011.

[27] Alexandre Bernardino, Marco Henriques, Norman Hendrich, and Jianwei Zhang. Precision grasp synergies for dexterous robotic hands. In *Proceedings of the IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 62–67. IEEE, 2013.

[28] Luigi Biagiotti, Hong Liu, Gerd Hirzinger, and Claudio Melchiorri. Cartesian impedance control for dexterous manipulation. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, volume 4, pages 3270–3275. IEEE, 2003.

[29] Luigi Biagiotti, Fabrizio Lotti, Claudio Melchiorri, and Gabriele Vassura. How far is the human hand? a review on anthropomorphic robotic endeffectors. *University of Bologna*, 2008.

[30] Antonio Bicchi. On the problem of decomposing grasp and manipulation forces in multiple whole-limb manipulation. *Robotics and Autonomous Systems*, 13(2):127–147, 1994.

[31] Antonio Bicchi. Hands for dexterous manipulation and robust grasping: A difficult road toward simplicity. *Transactions on Robotics and Automation*, 16(6):652–662, 2000.

[32] Antonio Bicchi, Marco Gabiccini, and Marco Santello. Modelling natural and artificial hands with synergies. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 366(1581):3153–3161, 2011.

[33] Antonio Bicchi and Vijay Kumar. Robotic grasping and contact: A review. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 348–353. IEEE, 2000.

[34] Rainer Bischoff, Ulrich Huggenberger, and Erwin Prassler. Kuka youbot-a mobile manipulator for research and education. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–4. IEEE, 2011.

[35] Robert Bonitz and Tien Hsia. Internal force-based impedance control for cooperating manipulators. *IEEE Transactions on Robotics and Automation*, 12(1):78–89, 1996.

[36] Justin Boyan. Technical update: Least-squares temporal difference learning. *Machine Learning*, 49(2-3):233–246, 2002.

[37] Joost Broekens, Marcel Heerink, and Henk Rosendal. Assistive social robots in elderly care: a review. *Gerontechnology*, 8(2):94–103, 2009.

[38] Rodney Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, 2(1):14–23, 1986.

[39] Ian Bullock and Aaron Dollar. Classifying human manipulation behavior. In *Proceedings of the IEEE International Conference on Rehabilitation Robotics (ICORR)*, pages 1–6. IEEE, 2011.

[40] Gereon Büscher, Risto Kõiva, Carsten Schürmann, Robert Haschke, and Helge Ritter. Flexible and stretchable fabric-based tactile sensor. *Robotics and Autonomous Systems*, 63:244–252, 2015.

[41] Geneviève Cadoret and Allan Smith. Friction, not texture, dictates grip forces used during object manipulation. *Journal of Neurophysiology*, 75(5):1963–1969, 1996.

[42] Manuel Catalano, Giorgio Grioli, Edoardo Farnioli, Alessandro Serio, Cristina Piazza, and Antonio Bicchi. Adaptive synergies for the design and control of the pisa/iit softhand. *The International Journal of Robotics Research*, 33(5):768–782, 2014.

[43] François Chaumette and Seth Hutchinson. Visual servo control. I. basic approaches. *Robotics and Automation Magazine*, 13(4):82–90, 2006.

[44] Sheng Chen, Colin Cowan, and Peter Grant. Orthogonal least squares learning algorithm for radial basis function networks. *IEEE Transactions on Neural Networks*, 2(2):302–309, 1991.

[45] Gang Cheng, Norman Hendrich, and Jianwei Zhang. In-hand manipulation action gist extraction from a data-glove. In *Foundations and Practical Applications of Cognitive Systems and Information Processing*, pages 773–782. Springer, 2014.

[46] Matei Ciocarlie, Corey Goldfeder, and Peter Allen. Dimensionality reduction for hand-independent dexterous robotic grasping. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 3270–3275. IEEE, 2007.

[47] Arlene Cole, Ping Hsu, and Shankar Sastry. Dynamic control of sliding by robot hands for regrasping. *IEEE Transactions on Robotics and Automation*, 8(1):42–52, 1992.

[48] Jack Copeland. *artificial intelligence (AI)*. Encyclopedia Britannica Online, 2015.

[49] Steve Cousins. Ros on the pr2. *Robotics and Automation Magazine*, 17(3):23–25, 2010.

[50] Mark Cutkosky. On grasp choice, grasp models, and the design of hands for manufacturing tasks. *IEEE Transactions on Robotics and Automation*, 5(3):269–279, 1989.

[51] Mark R Cutkosky and Imin Kao. Computing and controlling compliance of a robotic hand. *IEEE Transactions on Robotics and Automation*, 5(2):151–165, 1989.

[52] Ian Darian-Smith. The sense of touch: performance and peripheral neural processes. *Comprehensive Physiology*, 2011.

[53] Javier Garcia De Jalon and Eduardo Bayo. *Kinematic and dynamic simulation of multibody systems: the real-time challenge.* Springer Science & Business Media, 2012.

[54] Ramon Lopez De Mantaras and Josep Lluis Arcos. AI and music: From composition to expressive performance. *AI Magazine*, 23(3):43, 2002.

[55] Jack DiGiovanna, Babak Mahmoudi, Jose Fortes, Jose Principe, and Justin Sanchez. Coadaptive brain–machine interface via reinforcement learning. *IEEE Transactions on Biomedical Engineering*, 56(1):54–64, 2009.

[56] Aaron Dollar, Leif Jentoft, Jason Gao, and Robert Howe. Contact sensing and grasping performance of compliant hands. *Autonomous Robots*, 28(1):65–75, 2010.

[57] Jonathan Engel, Jack Chen, and Chang Liu. Development of polyimide flexible tactile sensor skin. *Journal of Micromechanics and Microengineering*, 13(3):359, 2003.

[58] Charlotte Exner. In-hand manipulation skills. *Development of Hand Skills in the Child*, pages 35–45, 1992.

[59] Jeremy Fishel, Gary Lin, Blaine Matulevich, and Gerald Loeb. Syntouch LLC biotac product manual, v. 16. `http://www.syntouchllc.com/Products/BioTac/_media/BioTac_Product_Manual.pdf`. Accessed: 2016-03-07.

[60] David Flavigné and Véronique Perdereau. A learning-free method for anthropomorphic grasping. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 2985–2990. IEEE, 2013.

[61] Naoki Fukaya, Tamim Asfour, Rüdiger Dillmann, and Shigeki Toyama. Development of a five-finger dexterous hand without feedback control: The tuat/karlsruhe humanoid hand. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 4533–4540. IEEE, 2013.

[62] Naoki Fukaya, Shigeki Toyama, Tamim Asfour, and Rüdiger Dillmann. Design of the tuat/karlsruhe humanoid hand. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, volume 3, pages 1754–1759. IEEE, 2000.

[63] Satoshi Funabashi, Alexander Schmitz, Takashi Sato, Sophon Somlor, and Shigeki Sugano. Robust in-hand manipulation of variously sized and shaped objects. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 257–263. IEEE, 2015.

[64] Noriatsu Furukawa, Akio Namiki, Senoo Taku, and Masatoshi Ishikawa. Dynamic regrasping using a high-speed multifingered hand and a high-speed vision system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 181–187. IEEE, 2006.

[65] Marco Gabiccini, Antonio Bicchi, Domenico Prattichizzo, and Monica Malvezzi. On the role of hand synergies in the optimal choice of grasping forces. *Autonomous Robots*, 31(2-3):235–252, 2011.

[66] Yang Gao, Lisa Anne Hendricks, Katherine Kuchenbecker, and Trevor Darrell. Deep learning for tactile understanding from visual and haptic data. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 536–543. IEEE, 2015.

[67] Rodolfo García-Rodríguez and Vicente Parra-Vega. Rolling a dynamic object with a planar soft-fingertip robot arm. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 2472–2478. IEEE, 2013.

[68] Megan Gibson. Meet the robot chef that can prepare your dinner. `http://time.com/3819525/robot-chef-moley-robotics/`. Accessed: 2016-10-09.

[69] Pierre Yves Glorennec. Reinforcement learning: An overview. In *European Symposium on Intelligent Techniques (ESIT-00)*, pages 14–15, 2000.

[70] Hiroaki Gomi and Mitsuo Kawato. Human arm stiffness and equilibrium-point trajectory during multi-joint movement. *Biological cybernetics*, 76(3):163–171, 1997.

[71] Philippe Gorce and Nasser Rezzoug. A method to learn hand grasping posture from noisy sensing information. *Robotica*, 22(03):309–318, 2004.

[72] Ivo Grondman, Lucian Buşoniu, Gabriel Lopes, and Robert Babuška. A survey of actor-critic reinforcement learning: Standard and natural policy gradients. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews,*, 42(6):1291–1307, 2012.

[73] Vijaykumar Gullapalli. A stochastic reinforcement learning algorithm for learning real-valued functions. *Neural Networks*, 3(6):671–692, 1990.

[74] Thiago Guzella and Walmir Caminhas. A review of machine learning approaches to spam filtering. *Expert Systems with Applications*, 36(7):10206–10222, 2009.

[75] Hideo Hanafusa and Haruhiko Asada. A robot hand with elastic fingers and its application to assembly process. In *Proceeding of the IFAC First Symposium on Information Control Problems in Manufacturing Technology*, pages 127–138, 1977.

[76] Norman Hendrich and Alexandre Bernardino. Affordance-based grasp planning for anthropomorphic hands from human demonstration. In *Proceedings of the IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 687–701. Springer, 2014.

[77] Norman Hendrich, Denis Kliment Jew, and Jianwei Zhang. Multi-sensor based segmentation of human manipulation tasks. In *Proceedings of the IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pages 223–229. IEEE, 2010.

[78] Matanya Horowitz and Joel Burdick. Interactive non-prehensile manipulation for grasping via pomdps. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3257–3264. IEEE, 2013.

[79] Kaijen Hsiao, Leslie Pack Kaelbling, and Tomas Lozano-Perez. Grasping pomdps. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4685–4692. IEEE, 2007.

[80] Takahiro Inoue and Shinichi Hirai. Elastic model of deformable fingertip for soft-fingered manipulation. *IEEE Transactions on Robotics*, 22(6):1273–1279, 2006.

[81] Stephen Jacobsen, Edwin Iversen, D Knutti, R Johnson, and K Biggers. Design of the utah/mit dextrous hand. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 3, pages 1520–1532. IEEE, 1986.

[82] Advait Jain and Charles Kemp. El-e: an assistive mobile manipulator that autonomously fetches objects from flat surfaces. *Autonomous Robots*, 28(1):45–64, 2010.

[83] Peters Jan. *Machine learning of motor skills for robotics.* PhD thesis, University of Southern California, 2007.

[84] Carlos Jara, Jorge Pomares, Francisco Candelas, and Fernando Torres. Control framework for dexterous manipulation using dynamic visual servoing and tactile sensors feedback. *Sensors*, 14(1):1787–1804, 2014.

[85] Roland Johansson and Randall Flanagan. Coding and use of tactile signals from the fingertips in object manipulation tasks. *Nature Reviews Neuroscience*, 10(5):345–359, 2009.

[86] Roland Johansson, Ulf Landstro, and Ronnie Lundstro. Responses of mechanoreceptive afferent units in the glabrous skin of the human hand to sinusoidal skin displacements. *Brain Research*, 244(1):17–25, 1982.

[87] Kenneth Langstreth Johnson and Kenneth Langstreth Johnson. *Contact mechanics*. Cambridge University Press, 1987.

[88] Leslie Pack Kaelbling, Michael Littman, and Andrew Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, pages 237–285, 1996.

[89] Michael Kaess. Apriltags c++ library, 2013. `http://people.csail.mit.edu/kaess/apriltags`. Accessed: 2016-03-07.

[90] Sham Kakade. A natural policy gradient. In *Advances in Neural Information Processing Systems (NIPS)*, volume 14, pages 1531–1538, 2001.

[91] Kenji Kaneko, Kensuke Harada, Fumio Kanehiro, Go Miyamori, and Kazuhiko Akachi. Humanoid robot hrp-3. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 2471–2478. IEEE, 2008.

[92] Imin Kao and Fuqian Yang. Stiffness and contact mechanics for soft fingers in grasping and manipulation. *IEEE Transactions on Robotics and Automation*, 20(1):132–135, 2004.

[93] Hilbert Kappen. An introduction to stochastic control theory, path integrals and reinforcement learning. *Cooperative Behavior in Neural Systems*, 887:149–181, 2007.

[94] Nagata Kazuaki. At Japan Robot Week, mechanical barista treats visitors to coffee. `http://www.businessinsider.com/japanese-coffee-making-robot-2014-10?IR=T`. Accessed: 2016-03-07.

[95] Charles Kemp, Aaron Edsinger, and Eduardo Torres-Jara. Challenges for robot manipulation in human environments. *Robotics and Automation Magazine*, 14(1):20–29, 2007.

[96] Fouad Khalil and Pierre Payeur. *Dexterous robotic manipulation of deformable objects with multi-sensory feedback-a review.* INTECH Open Access Publisher, 2010.

[97] Hajime Kimura, Kazuteru Miyazaki, and Shigenobu Kobayashi. Reinforcement learning in pomdps with function approximation. In *Proceedings of the Fourteenth International Conference on Machine Learning (ICML)*, volume 97, pages 152–160, 1997.

[98] Robert Klatzky, Susan Lederman, Chery Hamilton, and Gillian Ramsay. Perceiving roughness via a rigid probe: Effects of exploration speed. In *Proceedings of the ASME Dynamic Systems and Control Division*, volume 67, pages 27–33, 1999.

[99] Martin Knibestöl. Stimulus-response functions of slowly adapting mechanoreceptors in the human glabrous skin area. *The Journal of Physiology*, 245(1):63, 1975.

[100] Anna Kochan. Shadow delivers first hand. *Industrial robot: An International Journal*, 32(1):15–16, 2005.

[101] Nate Kohl and Peter Stone. Policy gradient reinforcement learning for fast quadrupedal locomotion. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 3, pages 2619–2624. IEEE, 2004.

[102] Keisuke Kojima, Takao Sato, Alexander Schmitz, Hiroaki Arie, Hiroshi Iwata, and Shigeki Sugano. Sensor prediction and grasp stability evaluation for in-hand manipulation. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 2479–2484. IEEE, 2013.

[103] Vijay Konda and John Tsitsiklis. Onactor-critic algorithms. *SIAM Journal on Control and Optimization*, 42(4):1143–1166, 2003.

[104] Masahiro Kondo, Jun Ueda, and Tsukasa Ogasawara. Recognition of in-hand manipulation using contact state transition for multifingered robot hand control. *Robotics and Autonomous Systems*, 56(1):66–81, 2008.

[105] Igor Kononenko. Machine learning for medical diagnosis: history, state of the art and perspective. *Artificial Intelligence in medicine*, 23(1):89–109, 2001.

[106] Danica Kragic, Mårten Björkman, Henrik Christensen, and Jan-Olof Eklundh. Vision for robotic object manipulation in domestic settings. *Robotics and Autonomous Systems*, 52(1):85–100, 2005.

[107] Mark Lee and Howard Nicholls. Review article tactile sensing for mechatronicsa state of the art survey. *Mechatronics*, 9(1):1–31, 1999.

[108] Qujiang Lei and Martijn Wisse. Fast grasping of unknown objects using force balance optimization. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 2454–2460. IEEE, 2014.

[109] Miao Li, Hang Yin, Kenji Tahara, and Aude Billard. Learning object-level impedance control for robust grasping and dexterous manipulation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 6784–6791. IEEE, 2014.

[110] Shunchong Li, Xinjun Sheng, Honghai Liu, and Xiangyang Zhu. Design of a myoelectric prosthetic hand implementing postural synergy mechanically. *Industrial Robot: An International Journal*, 41(5):447–455, 2014.

[111] Yanmei Li and Imin Kao. A review of modeling of soft-contact fingers and stiffness control for dextrous manipulation in robotics. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 3, pages 3055–3060. IEEE, 2001.

[112] Zexiang Li, Ping Hsu, and Shankar Sastry. Grasping and coordinated manipulation by a multifingered robot hand. *The International Journal of Robotics Research*, 8(4):33–50, 1989.

[113] Hong Liu, Peter Meusel, Gerd Hirzinger, Minghe Jin, Yiwei Liu, and Zongwu Xie. The modular multisensory dlr-hit-hand: hardware and software architecture. *IEEE Transactions on Mechatronics*, 13(4):461–469, 2008.

[114] Fabrizio Lotti, Paolo Tiezzi, Gabriele Vassura, Luigi Biagiotti, Gianluca Palli, and Claudio Melchiorri. Development of ub hand 3: Early results. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4488–4493. IEEE, 2005.

[115] Chris Lovchik and Myron Diftler. The robonaut hand: A dexterous robot hand for space. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 907–912. IEEE, 1999.

[116] Vaughan Macefield, Charlotte Häger-Ross, and Roland Johansson. Control of grip force during restraint of an object held between finger and thumb: responses of cutaneous afferents from the digits. *Experimental Brain Research*, 108(1):155–171, 1996.

[117] Christine MacKenzie and Thea Iberall. *The grasping hand*, volume 104. Elsevier, 1994.

[118] Fumitoshi Matsuno and Satoshi Tadokoro. Rescue robots and systems in japan. In *Proceedings of IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 12–20. IEEE, 2004.

[119] Claudio Melchiorri, Gianluca Palli, Giovanni Berselli, and Gabriele Vassura. Development of the ub hand iv: Overview of design solutions and enabling technologies. *Robotics and Automation Magazine*, 20(3):72–81, 2013.

[120] Ian Millington and John Funge. *Artificial intelligence for games*. CRC Press, 2012.

[121] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei Rusu, Joel Veness, Marc Bellemare, Alex Graves, Martin Riedmiller, Andreas Fidjeland, and Georg Ostrovski. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[122] Haseena Muhammad, Calogero Oddo, Lucia Beccai, Carmine Recchiuto, Carl Anthony, Mike Adams, Maria Carrozza, David Hukins, and Mike Ward. Development of a bioinspired mems based capacitive tactile sensor for a robotic finger. *Sensors and Actuators A: Physical*, 165(2):221–229, 2011.

[123] Laura Alcaide Muñoz. Robust dexterous manipulation: a methodology using visual servoing. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, volume 1, pages 292–297. IEEE, 1998.

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

[125] John Napier. The prehensile movements of the human hand. *Journal of Bone and Joint Surgery*, 38(4):902–913, 1956.

[126] Benjamin Navarro, Prajval Kumar, Aicha Fonte, Philippe Fraisse, Gérard Poisson, and Andrea Cherubini. Active calibration of tactile sensors mounted on a robotic hand. In *IEEE/RSJ IROS Workshop on Multimodal sensor-based robot control for HRI and soft manipulation*. IEEE, 2015.

[127] Lael Odhner, Leif Jentoft, Mark Claffee, Nicholas Corson, Yaroslav Tenzer, Raymond Ma, Martin Buehler, Robert Kohout, Robert Howe, and Aaron Dollar. A compliant, underactuated hand for robust manipulation. *The International Journal of Robotics Research*, 33(5):736–752, 2014.

[128] Katsuhiko Ogata. *Modern control engineering*. Prentice Hall PTR, 2001.

[129] Allison Okamura, Niels Smaby, and Mark Cutkosky. An overview of dexterous manipulation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 1, pages 255–262. IEEE, 2000.

[130] Edwin Olson. Apriltag: A robust and flexible visual fiducial system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3400–3407. IEEE, 2011.

[131] Mark Orr. Introduction to radial basis function networks. 1996.

[132] Junichi Osada, Shinichi Ohnaka, and Miki Sato. The scenario and design process of childcare robot, papero. In *Proceedings of the 2006 ACM SIGCHI international conference on Advances in computer entertainment technology*. ACM, 2006.

[133] Ryuta Ozawa, Suguru Arimoto, Shinsuke Nakamura, and Ji-Hun Bae. Control of an object with parallel surfaces by a pair of finger robots without object sensing. *IEEE Transactions on Robotics*, 21(5):965–976, 2005.

[134] Joni Pajarinen and Ville Kyrki. Robotic manipulation in object composition space. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 1–6. IEEE, 2014.

[135] Simone Parisi, Hany Abdulsamad, Alexandros Paraschos, Christian Daniel, and Jan Peters. Reinforcement learning vs human programming in tetherball robot games. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 6428–6434. IEEE, 2015.

[136] Jooyoung Park and Irwin Sandberg. Universal approximation using radial-basis-function networks. *Neural Computation*, 3(2):246–257, 1991.

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

[138] Jan Peters and Stefan Schaal. Natural actor-critic. *Neurocomputing*, 71(7):1180–1190, 2008.

[139] Jan Peters and Stefan Schaal. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21(4):682–697, 2008.

[140] Jan Peters, Sethu Vijayakumar, and Stefan Schaal. Reinforcement learning for humanoid robotics. In *Proceedings of the 3rd IEEE-RAS international conference on humanoid robots*, pages 1–20, 2003.

[141] Jan Peters, Sethu Vijayakumar, and Stefan Schaal. Natural actor-critic. In *European Conference on Machine Learning (ECML)*, pages 280–291. Springer, 2005.

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

[143] Ali Punjani and Pieter Abbeel. Deep learning helicopter dynamics models. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3223–3230. IEEE, 2015.

[144] Ariella Richardson, Juha-P Strom, and Edwin Olson. Aprilcal: Assisted and repeatable camera calibration. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 1814–1821. IEEE, 2013.

[145] Kenneth Roberts. Robot active touch exploration: Constraints and strategies. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 980–985. IEEE, 1990.

[146] Shadow robot company. Hand-Lite almost all the capability of a dexterous hand in a smaller, lighter, cheaper package. In *https://www.shadowrobot.com/hand-lite*, Accessed: 15, Jan. 2016.

[147] Joseph Romano, Kaijen Hsiao, Günter Niemeyer, Sachin Chitta, and Katherine Kuchenbecker. Human-inspired robotic grasp control with tactile sensing. *IEEE Transactions on Robotics*, 27(6):1067–1079, 2011.

[148] Frank Rothling, Robert Haschke, Jochen Steil, and Helge Ritter. Platform portable anthropomorphic grasping with the bielefeld 20-dof shadow and 9-dof tum hand. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 2951–2956. IEEE, 2007.

[149] Frederik Ruelens, Bert Claessens, Salman Quaiyum, Bart De Schutter, Robert Babuska, and Ronnie Belmans. Reinforcement learning applied to an electric water heater: From theory to practice. *arXiv:1512.00408*, 2015.

[150] Marco Santello, Martha Flanders, and John Soechting. Postural hand synergies for tool use. *The Journal of Neuroscience*, 18(23):10105–10115, 1998.

[151] Friedhelm Schwenker, Hans Kestler, and Günther Palm. Three learning phases for radial-basis-function networks. *Neural Networks*, 14(4):439–458, 2001.

[152] Taku Senoo, Yuji Yamakawa, Satoru Mizusawa, Akio Namiki, Masatoshi Ishikawa, and Makoto Shimojo. Skillful manipulation based on high-speed sensory-motor fusion. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1611–1612. IEEE, 2009.

[153] Karun Shimoga. Robot grasp synthesis algorithms: A survey. *The International Journal of Robotics Research*, 15(3):230–266, 1996.

[154] Milan Sonka, Vaclav Hlavac, and Roger Boyle. *Image processing, analysis, and machine vision.* CL Engineering, 2014.

[155] Mandayam Srinivasan, JM Whitehouse, and Robert LaMotte. Tactile detection of slip: surface microgeometry and peripheral neural codes. *Journal of Neurophysiology*, 63(6):1323–1332, 1990.

[156] Sascha Stoeter, Stephan Voss, Nikolaos Papanikolopoulos, and Heiko Mosemann. Planning of regrasp operations. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 1, pages 245–250. IEEE, 1999.

[157] Stefano Stramigioli. *Modeling and IPC control of interactive mechanical systems: a coordinate-free approach*. Springer US, 2001.

[158] Freek Stulp, Evangelos Theodorou, and Stefan Schaal. Reinforcement learning with sequences of motion primitives for robust manipulation. *IEEE Transactions on Robotics*, 28(6):1360–1370, 2012.

[159] Richard Sutton and Andrew Barto. *Reinforcement learning: An introduction*, volume 1. MIT Press, 1998.

[160] Richard Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems (NIPS)*, volume 12, pages 1057–1063. MIT Press, 2000.

[161] Kenji Tahara, Suguru Arimoto, and Morio Yoshida. Dynamic object manipulation using a virtual frame by a triple soft-fingered robotic hand. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4322–4327. IEEE, 2010.

[162] Kenji Tahara, Keigo Maruta, Akihiro Kawamura, and Motoji Yamamoto. Externally sensorless dynamic regrasping and manipulation by a triple-fingered robotic hand with torsional fingertip joints. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3252–3257. IEEE, 2012.

[163] Johan Tegin and Jan Wikander. Tactile sensing in intelligent robotic manipulation-a review. *Industrial Robot: An International Journal*, 32(1):64–70, 2005.

[164] Pramodsingh Thakur, Amy Bastian, and Steven Hsiao. Multidigit movement synergies of the human hand in an unconstrained haptic exploration task. *The Journal of Neuroscience*, 28(6):1271–1281, 2008.

[165] Evangelos Theodorou, Jonas Buchli, and Stefan Schaal. A generalized path integral control approach to reinforcement learning. *The Journal of Machine Learning Research*, 11:3137–3181, 2010.

[166] Sarah Tomlinson, Roger Lewis, and Matt Carré. Review of the frictional properties of finger-object contact when gripping. *Proceedings of the Institution of Mechanical Engineers, Part J: Journal of Engineering Tribology*, 221(8):841–850, 2007.

[167] William Townsend. The barretthand grasper-programmably flexible part handling and assembly. *Industrial Robot: An International Journal*, 27(3):181–188, 2000.

[168] Robert Trippi and Efraim Turban. *Neural networks in finance and investing: Using artificial intelligence to improve real world performance*. McGraw-Hill, Inc., 1992.

[169] Konstantinos Trohidis, Grigorios Tsoumakas, George Kalliris, and Ioannis Vlahavas. Multi-label classification of music into emotions. In *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, volume 8, pages 325–330, 2008.

[170] Åke Vallbo and Roland Johansson. Properties of cutaneous mechanoreceptors in the human hand related to touch sensation. *Human Neurobiology*, 3(1):3–14, 1984.

[171] Herke Van Hoof, Tucker Hermans, Gerhard Neumann, and Jan Peters. Learning robot in-hand manipulation with tactile features. In *Proceedings of the IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 121–127. IEEE, 2015.

[172] Phongtharin Vinayavekhin, Shunsuke Kudoh, Jun Takamatsu, Yuuki Sato, and Katsushi Ikeuchi. Representation and mapping of dexterous manipulation through task primitives. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3722–3729. IEEE, 2013.

[173] Kazuyoshi Wada and Takanori Shibata. Living with seal robotsits sociopsychological and physiological influences on the elderly at a care house. *IEEE Transactions on Robotics*, 23(5):972–980, 2007.

[174] Qian Wan, Ryan Adams, and Robert Howe. Variability and predictability in tactile sensing during grasping. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 158–164. IEEE, 2016.

[175] Ying Wang, Haoxiang Lang, and Clarence W De Silva. A hybrid visual servo controller for robust grasping by wheeled mobile robots. *Transactions on Mechatronics*, 15(5):757–769, 2010.

[176] Nicholas Wettels, Veronica Santos, Roland Johansson, and Gerald Loeb. Biomimetic tactile sensor array. *Advanced Robotics*, 22(8):829–849, 2008.

[177] David Williams and Oussama Khatib. The virtual linkage: A model for internal forces in multi-grasp manipulation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1025–1030. IEEE, 1993.

[178] Ronald Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, 1992.

[179] Thomas Wimböck, Christian Ott, Alin Albu-Schäffer, and Gerd Hirzinger. Comparison of object-level grasp controllers for dynamic dexterous manipulation. *The International Journal of Robotics Research*, 31(1):3–23, 2012.

[180] Thomas Wimboeck, Christian Ott, and Gerhard Hirzinger. Passivity-based object-level impedance control for a multifingered hand. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 4621–4627. IEEE, 2006.

[181] Anurat Wisitsoraat, Viyapol Patthanasetakul, Tanom Lomas, and Adisorn Tuantranont. Low cost thin film based piezoresistive mems tactile sensor. *Sensors and Actuators A: Physical*, 139(1):17–22, 2007.

[182] Kai Xu, Huan Liu, Yuheng Du, and Xiangyang Zhu. Design of an underactuated anthropomorphic hand with mechanically implemented postural synergies. *Advanced Robotics*, 28(21):1459–1474, 2014.

[183] Takashi Yoshioka, Sliman Bensmaia, James Craig, and Steven Hsiao. Texture perception through direct and indirect touch: An analysis of perceptual space for tactile textures in two modes of exploration. *Somatosensory and Motor Research*, 24(1-2):53–70, 2007.

[184] Jianwei Zhang and Markus Ferch. Extraction and transfer of fuzzy control rules for sensor-based robotic operations. *Fuzzy Sets and Systems*, 134(1):147–167, 2003.

[185] Jianwei Zhang and Alois Knoll. Designing fuzzy controllers by rapid learning. *Fuzzy Sets and Systems*, 101(2):287–301, 1999.

[186] Joshua Zheng, Sara De La Rosa, and Aaron Dollar. An investigation of grasp type and frequency in daily household and machine shop tasks. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4169–4175. IEEE, 2011.

Erklärung der Urheberschaft

Ich versichere an Eides statt, dass ich die vorliegende Dissertation selbstständig und ohne unerlaubte Hilfe Dritter angefertigt habe. Alle Stellen, die inhaltlich oder wörtlich aus anderen Veröffentlichungen stammen, sind kenntlich gemacht. Diese Arbeit lag in gleicher oder ähnlicher Weise noch keiner Prüfungsbehörde vor und wurde bisher noch nicht veröffentlicht.

Ort, Datum	Unterschrift
Hamburg,25.07.2017	Junhu He