Vision-based Perception for Dexterous Hand-arm Teleoperation

Dissertation
zur Erlangung des akademischen Grades
Dr. rer. nat
an der Fakultät für Mathematik, Informatik und Naturwissenschaften
der Universität Hamburg

Beteuer: Prof. Dr. Jianwei Zhang

Eingereicht beim Fachbereich Informatik
von Shuang Li

Dezember 2021

Gutachter:
Prof. Dr. Jianwei Zhang
Prof. Dr. Stefan Wermter

Tag der Disputation: 22.03.2022

Abstract

Teleoperation empowers robotic systems with sophisticated reasoning skills, intuition, and creativity, offering the possibility of performing complex and dangerous tasks in remote and unconstructed environments. Leveraging the rapidly developing data-driven human hand pose estimation algorithms, markerless vision-based teleoperation offers the advantages of detecting natural human-limb motions and being less invasive. The main contribution of this thesis is employing markerless vision-based teleoperation using an end-to-end learning scheme for anthropomorphic hands and integrating it into a dexterous hand-arm teleoperation system.

The end-to-end learning scheme targets the robot space directly and generates robot joint angles from depth images of the human hand. Considering the domain difference between the human hand and the robot hand, two end-to-end neural networks, *TeachNet* and *Transteleop*, were designed to exploit the geometrical resemblance from human-robot hand pairs and to learn the kinematic mappings between them. *TeachNet* highlights a consistency loss function connecting the double-stream human-robot learning architecture, while *Transteleop* develops a multi-output learning structure based on the image-to-image translation method. Furthermore, two training sets, including 400K pairwise human-robot depth images and corresponding robot joint angles, were collected. The difference between them is whether the robot hand has the same viewpoint as the human hand in the depth images.

Then, we studied how to seamlessly integrate the markerless vision-based hand teleoperation into a dexterous hand-arm teleoperation system. The robotic arm control was implemented using an IMU suit and a motion tracking system. Meanwhile, a 3D printed camera holder and a controlled active vision system were successively built to perceive human finger motions from an optimal viewpoint and to decouple the human and the robot workspace.

Network evaluation of two proposed neural network models verifies that *Transteleop* achieves higher accuracy and lower error than other end-to-end baselines over three quantitative metrics. Robot experiments were conducted on an anthropomorphic hand installed on a robotic arm to demonstrate its applicability for different challenging tasks in robotics. These include a variety of complex manipulation tasks, including grasping, placing, inserting, pushing, sliding, pouring, handover, and bottle opening. In addition, the proposed markerless vision-based teleoperation models also accomplish compliant grasping and manipulation skills facilitated by an adaptive force controller. The experimental results show that the proposed teleoperation systems are accurate, efficient and robust, and can contribute to solving complex tasks in multiple applications, e.g., space station, medical surgery, industrial transportation, and daily life, which are of high need in robotics.

Kurzfassung

Durch Fernsteuerung von Robotik-Systemen können komplexe Aufgaben in unzugänglichen oder gefährlichen Umgebungen gelöst werden, die ein intelligentes, kreatives und problemorientiertes Vorgehen erfordern. Neue, lernende Algorithmen zur Erkennung der Hand-Pose sowie markerlose optische Tracking-Systeme ermöglichen dabei die Erkennung menschlicher Handhabungs- und Bewegungsabläufe in nicht invasiver Form. Der zentrale Beitrag dieser Arbeit ist ein derartiges markerloses optisches Teleoperations-System, das mithilfe von Ende-zu-Ende-Methoden trainiert und in ein Hand-Arm-System integriert wurde.

Die Lernmethode zielt dabei direkt auf den Bewegungssraum des Roboters und generiert passende Winkelstellungen für die Robotergelenke aus Tiefenbildern der menschlichen Hand. Um die spezifischen Unterschiede zwischen der menschlichen Hand und der Roboterhand zu berücksichtigen, wurden zwei neuronale Netze, *TeachNet* und *Transteleop*, entworfen. Diese lernen die notwendigen geometrischen Abbildungen vorzunehmen und dabei die Kinematik der Hände zu berücksichtigen. *TeachNet* nutzt eine „Consistency Loss“-Funktion, um die Abbildung vom Menschen auf den Roboter zu lernen, während *Transteleop* einen Bild-zu-Bild basierten Ansatz verfolgt. Zusammen mit der vorliegenden Arbeit wurden zwei Trainingsdatensätze erstellt, die über 400 000 paarweise Zuordnungen von Mensch und Roboter Tiefenbildern und den zugehörigen Winkelstellungen der Roboterkinematik enthalten. Diese Datensätze unterscheiden sich darin, dass der Beobachtungspunkt gleich bzw. unterschiedlich ist.

Anschließend untersuchten wir, wie sich die markerlose, optische Hand-Teleoperation nahtlos in ein Hand-Arm-Teleoperationssystem integrieren lässt. Zur Steuerung des kompletten Roboterarms wurden Motion-Tracking-Systeme sowie Beschleunigungssensoren genutzt. Für die Erfassung der Hand- und Fingerpositionen wurde der Sensor auf einem eigens entworfenen und 3D-gedruckten umschnallbaren Kamerahalter montiert und mithilfe eines „Active Vision“ Algorithmus angesteuert. Dadurch war es möglich, optimale Aufnahmen von Hand- und Greifbewegungen zu machen und diese vom Arbeitsraum des Roboters zu entkoppeln.

Ein Vergleich der beiden neuronalen Modelle mit anderen Ansätzen aus der Literatur hat gezeigt, dass speziell *Transteleop* eine höhere Genauigkeit und weniger Fehler hat als andere grundlegende Ende-zu-Ende-Ansätze. Dabei wurden drei unterschiedliche Metriken alternativ zugrunde gelegt. Um zu demonstrieren, dass damit verschiedene, sehr anspruchsvolle Aufgaben gelöst werden können, wurden verschiedene Experimente mit einer anthropomorphen Hand an einem Roboterarm durchgeführt. Unter anderem Greifen, Positionieren, Einführen und Verschieben von Objekten sowie Öffnen einer Flasche, Eingießen von Flüssigkeiten und Übergabe von Objekten. Die vorgestellten Teleoperationsmodelle ermöglichen auch nachgiebige Greif- und Manipulationsbewegungen, die durch eine adaptive Kraft-basierte Regelung unterstützt

werden. Die experimentellen Ergebnisse zeigen, dass dieses neuartige System zur Fernsteuerung anthropomorpher Robotermanipulatoren in Bezug auf Genauigkeit, Effizienz und Stabilität in der Lage ist, hochkomplexe Aufgaben zu lösen, wie sie beispielsweise in Weltraummissionen, bei medizinischen Eingriffen, industriellen Transport- und Montageaufgaben sowie beim Einsatz von Robotern in der Servicerobotik im täglichen Leben anfallen. All diese Szenarien stellen derzeit die Robotik noch vor große Herausforderungen.

Contents

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Aim of this Thesis . . . 3
- 1.3 Research Questions . . . 5
- 1.4 Thesis Structure . . . 5

2 State of the Art **7**
- 2.1 User-Level Interfaces at the Local Site . . . 8
  - 2.1.1 Contacting/Wearable Interfaces in Teleoperation . . . 8
  - 2.1.2 Markerless Interfaces in Teleoperation . . . 11
- 2.2 Human-Robot Motion Mapping . . . 13
  - 2.2.1 Human Pose Estimation . . . 13
  - 2.2.2 Human Pose Recognition . . . 14
- 2.3 Control Architectures in Teleoperation . . . 15
- 2.4 Data-driven 3D Human Hand Pose Estimation . . . 18
  - 2.4.1 Human Hand Pose Datasets . . . 20
  - 2.4.2 Input Modality in Hand Pose Estimation . . . 21
  - 2.4.3 Network Structure in 3D Hand Pose Estimation . . . 22
  - 2.4.4 Evaluation Metrics . . . 26
- 2.5 Discussion . . . 27

3 Pairwise Human-Robot Hand Dataset Generation **29**
- 3.1 Hand Kinematic Model . . . 30
  - 3.1.1 Human Hand Kinematic Model . . . 30
  - 3.1.2 Shadow Dexterous Hand . . . 30
  - 3.1.3 Comparison of Human Hand and Robot Hand . . . 30
- 3.2 Dataset Generation Pipeline . . . 33
- 3.3 Selection of Human Hand Pose Dataset . . . 34
- 3.4 Optimized Mapping Method . . . 34
- 3.5 Discussion . . . 38

4 TeachNet: Vision-based Teleoperation System for Anthropomorphic Robot Hand **41**
- 4.1 End-to-end Learning for Vision-based Teleoperation . . . 42
- 4.2 Teacher-Student Network: TeachNet . . . 42
  - 4.2.1 Network Structure of TeachNet . . . 43
  - 4.2.2 Loss Functions for Robot Joint Regression . . . 45
- 4.3 TeachNet Evaluation . . . 47

4.3.1 Training Details . . . 47
4.3.2 Baselines Comparison . . . 47
4.3.3 Loss Analysis . . . 49
4.4 Robotic Experiments . . . 50
4.4.1 Simulation Experiments of Gesture Imitation . . . 51
4.4.2 Grasping Experiments on Real Robot . . . 51
4.5 Discussion . . . 54

5 Transteleop: Image-to-image Translation Inspired Robot Hand Pose Learning 57
5.1 Why image-to-image translation . . . 57
5.2 Network Backbone Design Choices . . . 59
5.3 Transteleop . . . 60
5.3.1 Preprocessing Module . . . 60
5.3.2 Encoder-embedding-decoder Module . . . 63
5.3.3 Joint Module . . . 64
5.4 Network Experiments . . . 65
5.4.1 Training Details . . . 65
5.4.2 Baselines Comparison . . . 65
5.4.3 Reconstructed Images . . . 69
5.4.4 Analysis Based on Viewpoint . . . 69
5.5 Discussion . . . 71

6 Multimodal Hand-Arm Teleoperation System Based on Vision and IMU 73
6.1 System Description . . . 73
6.2 IMU-based Arm Teleoperation Method . . . 74
6.2.1 Axis Neuron Software . . . 75
6.2.2 Data Broadcast Through ROS . . . 75
6.2.3 Arm Tracking . . . 77
6.3 Camera Holder . . . 78
6.4 Manipulation Experiments . . . 78
6.5 Discussion . . . 82

7 Hand-Arm Teleoperation System Based on Active Vision 83
7.1 Active Vision for Teleoperation . . . 84
7.2 Design Choices for Hand Tracking . . . 85
7.3 System Infrastructure . . . 87
7.4 Controlled Active Vision System . . . 90
7.5 Slave Robot Motion Generation . . . 92
7.6 Robot Experiments . . . 92
7.6.1 Precision Analysis . . . 93
7.6.2 Manipulation Experiments . . . 96
7.7 Discussion . . . 99

8 Evaluation Experiments on Compliant Teleoperation by Adaptive Force Control 101
8.1 System Overview . . . 101
8.2 Adaptive Force Control Strategy . . . 102
8.2.1 Grasping and Manipulation based on Force/Torque Control . . . 102

8.2.2 Methodology . . . 103
8.3 Simulation Experiments . . . 106
8.4 Real Robot Validation . . . 110
8.5 Discussion . . . 113

9 Conclusions and Outlook **115**
9.1 Limitations . . . 117
9.2 Future Research . . . 117

A Basics of Convolution Neural Networks **119**
A.1 LeNet-5 . . . 120
A.1.1 Convolutional Layer . . . 121
A.1.2 Pooling Layer . . . 121
A.1.3 Fully-connected Layer . . . 122
A.1.4 Nonlinear Activation Function . . . 122
A.2 Residual Neural Network . . . 123
A.3 Convolution Neural Networks using Depth Images . . . 124
A.4 End-to-end Learning . . . 125

B List of Publications **127**

C Glossary **129**

References **131**

List of Web-Adresses **149**

List of Figures **153**

List of Tables **157**

Acknowledgements **159**

Chapter 1

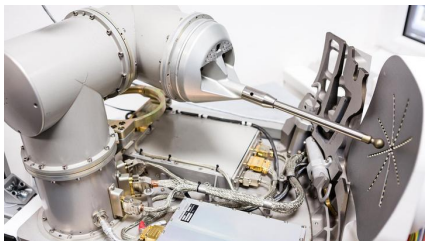
Introduction

1.1 Motivation

Throughout the history of robotics, *teleoperation* is perhaps one of the earliest applications. Integrating human intelligence, creativity, and cognition into the robotic systems, teleoperation is able to make fast decisions in remote and unconstructed environments. This characteristic of teleoperation prompts it to be used widely in hazardous environments threatening human safety, or remote places consuming a high cost to reaching, or micromanipulation and minimally invasive surgery requiring force amplification or position scaling, and other highly unconstructed environments. Typical applications are nuclear research, chemical industry, space station, disaster rescue, and the medical field [6, 48, 112].

Back to the 1950s, teleoperation research initially began with nuclear application and was used to handle radioactive materials behind remote shielded walls. In 1993, the first remotely controlled robot flew to space with the German Spacelab Mission D2. The six-axis robot arm was equipped with shared autonomy and teleoperated to perform assembling, floating object grasping, and "Orbital-Replaceable-Unit" connecting tasks with 5-7 s communication latency [W21]. The ROKVISS (Robotikkomponenten-Verifikation auf der ISS) project has completed over 500 repairing and maintaining tests on satellites by teleoperating a two-joint robot arm from 2005 to 2010 [W19]. The robot is controlled by a sophisticated joystick with force feedback without significant time delay, see the hardware in Fig. 1.1(a). Therefore, the operator on the ground feels the force exerted by the robot in space. In 2017, Shark Robotics announced Colossus, a remote-controlled mobile robot designed to support firefighters [W31]. Later in 2019, Colossus participated in the challenges for extinguishing the blaze, carrying heavy hoses, and clearing away debris at Notre Dame Cathedral (see Fig. 1.1(b)).

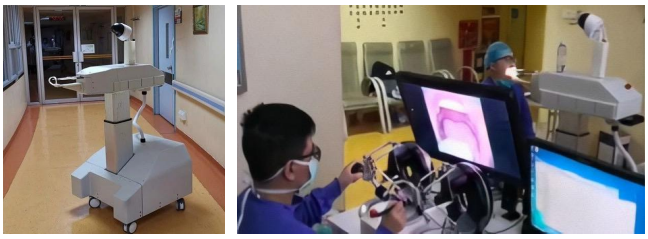
Moreover, medical surgery implemented by telerobotics and telesurgical systems has been studied since the 1980s. Fig. 1.1(c) shows one of the most advanced robotic surgical systems, the da Vinci Surgical System [W36]. The da Vinci surgical system transfers the surgeon's hand movements using the sophisticated hand controller to the robot arms. The hand controller can be moved, rotated while performing the procedure. More than seven million surgical procedures, e.g., cardiac surgery and colorectal surgery, have been completed by the da Vinci surgical system all over the world. Since the Coronavirus disease 2019 (COVID-19) shockingly spread worldwide, the robotic community has sped up to studying the use of



(a) The Hardware in ROKVISS project

(b) The Colossus robot at Notre Dame Cathedral

(c) The da Vinci Surgical System



(d) The throat-swabbing robot

Figure 1.1 – (a) The robotic arm and force-feedback joystick used in ROKVISS project. Reprinted left image: ©DLR, CC BY-NC-ND 3.0. Reprinted right image courtesy of DLR [W17]. (b) The Colossus robot teleoperated by the firemen helped fight a fire at Notre Dame Cathedral in Paris. Reprinted image courtesy of Shark robotics [W31]. (c) The main components of the da Vinci Surgical System: surgeon console, patient cart, and vision cart. The surgeon sits at the console, operating the teleoperation instruments while observing the patient's anatomy in a high-resolution 3D vision system. Reprinted image courtesy of Intuitive Surgical [W36]. (d) A throat swab sampling robot under shared control has been put into clinical trial to reduce disease transmission in 2019. Reprinted image courtesy of [W41].

robots in medical applications, e.g., supply chain automation, clinical care, and public services [89]. Mobile robots and drones have been used to deliver COVID-19 vaccines and medical samples in many countries. Beyond autonomous tasks in repetitive applications, the robots have also been teleoperated to perform care delivery either as a medium of communication or actual care. The robots serve as a mechanism to minimize direct contact between patients and medical personnel in the hospital or at home. To potentially reduce disease transmission, shared-control-based swab sampling robots were creatively researched as well (see Fig. 1.1(d)). Meanwhile, telemedicine or telehealth has quickly risen to evaluate, diagnose and even treat patients over the telecommunication infrastructure and using robots.

Shifting to industrial scenarios and daily life, telerobotics also shows enormous potentials with the development of cloud-based software and 5G networks. A teleoperation startup, Phantom Auto, develops remote operation software and dedicated equipment that integrates with unmanned vehicles, from robot-taxis and delivery robots to lift trucks and shunt trucks [W1]. Teleoperation is expanding into transportation and logistics, and some self-driving car companies rely on remote control technology as part of the transition to fully unmanned transportation. In addition, remotely-controlled assistive robots can help frail elderly people to perform specific tasks related to independent living, thereby improving their quality of life [124]. However, the

tasks mainly are remotely monitoring, care delivering, indoor navigation, but are still far from daily manipulation tasks, e.g., pouring water, picking clothes from the wardrobe, and putting on socks.

In general, most work still focuses on controlling mobile robots, robotic arms, and simple grippers rather than multi-fingered robotic hands. However, two-fingered grippers can only execute some simple object interactions and manipulation categories, e.g., pick and place, low-dimensional translational, and rotation. Looking into the future, dexterous robotic hands provide a prospective base for supplanting human hands in the execution of tedious and dangerous tasks [69]. Therefore, to endow robotic hands with similar dexterity to human hands, which can effectively grasp deformable, irregular objects and utilize various tools like wrenches, screwdrivers, anthropomorphic hands have become a promising solution and have gained much attention over the past years. However, the teleoperation of an anthropomorphic robotic hand performing dexterous manipulation faces many challenges, including inaccuracy, customization for each user, and efficiency issues because of high Degree of Freedom (DoF). Traditional teleoperation methods for dexterous hand teleoperation often rely on wearable datagloves. Nevertheless, wearable datagloves need to be customized for a certain size range of the human hand and may hinder natural human-finger motion. A promising alternative is vision-based teleoperation, which is cheap, less invasive, and allows to perform natural and comfortable gestures. Leveraging deep learning innovations in computer vision in recent ten years, this thesis studies on introducing visual perception into dexterous hand teleoperation for an accurate and efficient dexterous hand-arm teleoperation system.

1.2 Aim of this Thesis

In this thesis, visual perception for a dexterous hand-arm teleoperation system refers explicitly to controlling the anthropomorphic hand by vision-based teleoperation and combining this with an efficient arm teleoperation method for developing an accurate and efficient system.

Vision-based teleoperation methods for dexterous hand compute control commands for the robot from segmented images of the human hand based on a camera system. The typical vision-based teleoperation process estimates the 3D hand pose or recognizes the class of hand gestures using human hand pose estimation algorithms, followed by mapping the locations or the corresponding poses to the robot. Nevertheless, this method does not consider the physical constraints and joint limits of the robots in the estimation or recognize phase, so it easily generates poses that the robot cannot reach. Also, due to illumination changes, self-occlusion, and pose-ambiguity of the human hand images, the vision-based teleoperation methods face the challenge of inaccuracy.

Therefore, the first aim of this thesis is to develop an accurate end-to-end vision-based teleoperation method. Taking advantage of deep learning technologies in computer science, data-driven human hand pose estimation, which places more weight on feature extraction, object representation, and perceptual processing, is a valuable choice. In addition, an end-to-end learning scheme directly maps human hand poses to the robotic hand space instead of the human hand space, considering the robot model in the training process and bypassing the post-processing. Consequently, how to build a large number pairwise human-robot hand dataset,

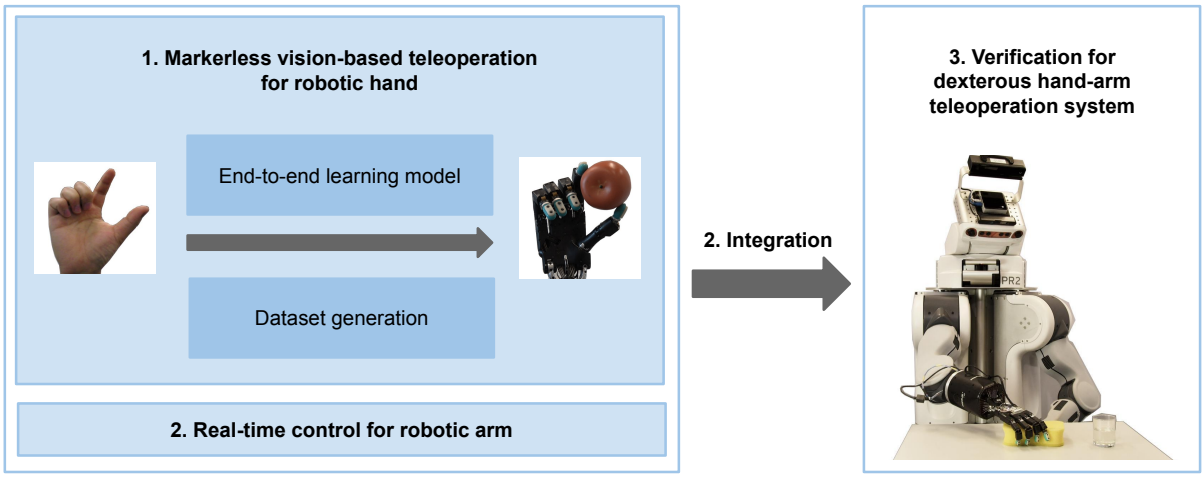


Figure 1.2 – Aims of this thesis.

that includes pairs of depth images, as well as corresponding joint angles of the robot hand in the same gesture, is an essential problem that should be solved in order to fulfill the aim.

The second aim is to seamlessly integrate vision-based dexterous hand teleoperation into a whole hand-arm system. Though a lot of work is investigating robot arm teleoperation, only a few work considers the connection between arm control and hand behavior [31]. From the software aspect, the robot arm should move smoothly following the trajectory of the human arm based on trajectory generation methods. From the hardware aspect on the operator side, the dexterous hand-arm hand system requires that:

1) the human hand is not covered or occluded by any devices;

2) the narrow field of view of the camera does not limit the workspace of the robot;

3) the camera attempts to capture the human hand from the best perspective.

All the requirements end up with appropriate arm teleoperation methods and a camera system to move along with the human hand.

Finally, the third goal is to ensure the robustness and stability of the developed teleoperation system from a practical perspective. Therefore, designing systematic verification is necessary for software and real-world experiments. In summary, a schematic diagram of the thesis aims is shown in Fig. 1.2.

Some existing teleoperation systems are customized for specific users or perform specific manipulation tasks [46]. The long-term vision of this thesis is an integrated robotic hand-arm teleoperation system, which is open for anyone and has a wide range of manipulation applications in daily life. Such a system could control the robot to assist daily routines, do chores in the hospital or at home by doctors or nurses for patients and elderly people, or perform complex assembly tasks in space exploration by astronauts or scientists.

1.3 Research Questions

As discussed in section 1.2, this thesis focuses on three main aims: accurate vision-based teleoperation for robotic hand, effective hand-arm system integration, and comprehensive system verification. Pivoting these aims, the following four main research questions were considered and studied in this thesis:

Q1: Teleoperation Methods for Anthropomorphic Hand: How to employ deep-learning-based human hand pose estimation methods into robotic hand teleoperation in such a way that different users can efficiently use the system and all human finger motions of are natural and unrestricted?

Q2: Teleoperation Methods for Robotic Arm: Which devices are applicable for teleoperating a robotic arm with multiple degrees of freedom? How to register the human arm motions to the robot system? How to ensure smooth and real-time arm trajectories?

Q3: Integration of Hand-Arm Teleoperation System: With regard to dexterous manipulation, the entire teleoperation system is supposed to be accurate, efficient, robust, and user-friendly. How to seamlessly combine the markerless vision-based robot hand control with the robotic arm teleoperation? How to extend the perception field of the sensors and perceive human motions better through improving the sensor installation?

Q4: System Verification: Which evaluation aspects, manipulation tasks, and metrics can be used to verify the stability of software and hardware in a robotic hand-arm teleoperation system? Controller wise, how to verify the proposed teleoperation method under both position control and force control?

1.4 Thesis Structure

The remaining parts of this thesis are organized as follows (illustrated in Fig. 1.3):

- Chapter 2 summarizes the state-of-the-art. Particularly, two topics highly related to this thesis are reviewed, i.e., the applications of markerless vision-based teleoperation and neural network design of data-driven 3D human hand pose estimation.
- Chapter 3 describes the collection pipeline of pairwise human-robot hand datasets. Comparison of human-robot kinematic structure, selection of the human hand pose dataset, generation of robotic ground truth by an efficient optimized mapping method are presented in succession. The pairwise datasets are used for training vision-based neural network models for robotic hand teleoperation. The related publication is [93].
- Chapter 4 develops an effective end-to-end vision-based teleoperation model, TeachNet (also published in [93]). The design concept, network structure, and loss functions are introduced. Then comprehensive network evaluation and imitation experiments are presented to verify the effectiveness and applicability of the end-to-end learning scheme in vision-based teleoperation. Finally, some remarks on network limitations and robot experiments conclude this chapter.

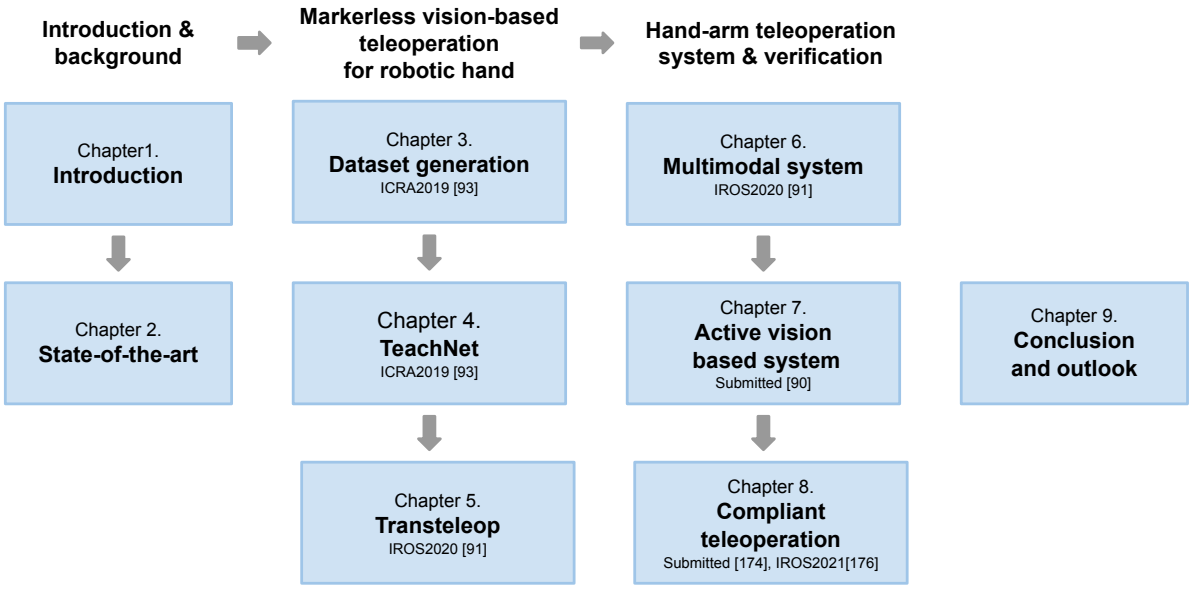


Figure 1.3 – The structure of this thesis.

- Chapter 5 continues with the other novel design of the end-to-end vision teleoperation model, Transteleop (published in [91]). The main focus of Transteleop is to adapt the idea of image-to-image translation for extracting latent pose representations. Except for baseline comparison, the visualization of synthesized hand images and viewpoint analysis are investigated as well.

- Chapter 6 elaborates on a hand-arm teleoperation system using Transteleop and a IMU-based arm control. Teleoperation for a robot arm and integrating the separate hand-arm control into an integral system are successfully achieved. Verification of this system is demonstrated in several complex manipulation tasks that go beyond simple pick-and-place operations (also published in [91]).

- Chapter 7 establishes a novel hand-arm teleoperation system with active vision, which intends to capture the human hand from the best viewpoint and at an optimal distance. Regarding the system integration, hardware preparation, software communication, and the control strategies of the robot arm performing active vision are analyzed. This chapter ends with various manipulation tasks on real robots, e.g., pouring and fader sliding (submitted [90]).

- Chapter 8 extends the proposed vision-based teleoperation models to compliant teleoperation using an adaptive control strategy. The performance of compliant teleoperation is validated through multiple simulation and real-world robot experiments (published in [176], and submitted [174]).

- Chapter 9 conveys the scientific contributions and the conclusion remarks of this thesis. It also presents the limitations of this thesis and gives an outlook on upcoming future research.

Chapter 2

State of the Art

The term *teleoperation* contains a broad field of techniques and hardware incorporated to receive and discern reliable decisions harnessing human cognition and creativity, and to achieve effective robotic manipulation. In contrast to autonomous control or intelligent programming, where a robot performs a task or motion without further guidance or instruction by the operator, teleoperation systems require commands from the operator then manipulate the robots to fulfill the commanded tasks. Robotic teleoperation systems conceptually consist of two sites: the local and remote sites. The local site, or the operator site, comprises the human teleoperator and multiple user-level interfaces used to measure or support the human's movements and display the real-time status at the remote site. The remote site comprises robots, supporting sensors, and manipulated objects. Receiving the perception information from sensors and the commands from the human, the robot then performs the manipulation tasks.

In this chapter, an overview of robotic teleoperation is studied from a user perspective. Section 2.1 presents user-level interfaces used at the local site and their applicability, including contacting/wearable interfaces and markerless interfaces. Then, section 2.2 depicts how to map human motions to the robot by two human-intention estimation methods and the corresponding mapping methods. Section 2.3 enumerates three control architectures of teleoperation systems based on how much the robot autonomy is involved in the teleoperation process.

Since this thesis highlights teleoperating an anthropomorphic robotic hand and markerless vision-based teleoperation is well tailored to this task, data-driven 3D hand pose estimation approaches used for the vision-based teleoperation are next discussed here. In section 2.4, after an analysis of the challenges in hand pose estimation, the state-of-the-art hand pose datasets, input modalities, network architectures, and evaluation metrics are presented.

In summary, this chapter focuses on the state-of-the-art of robotic teleoperation and data-driven 3D hand pose estimation. Fig. 2.1 gives an overview of this chapter.

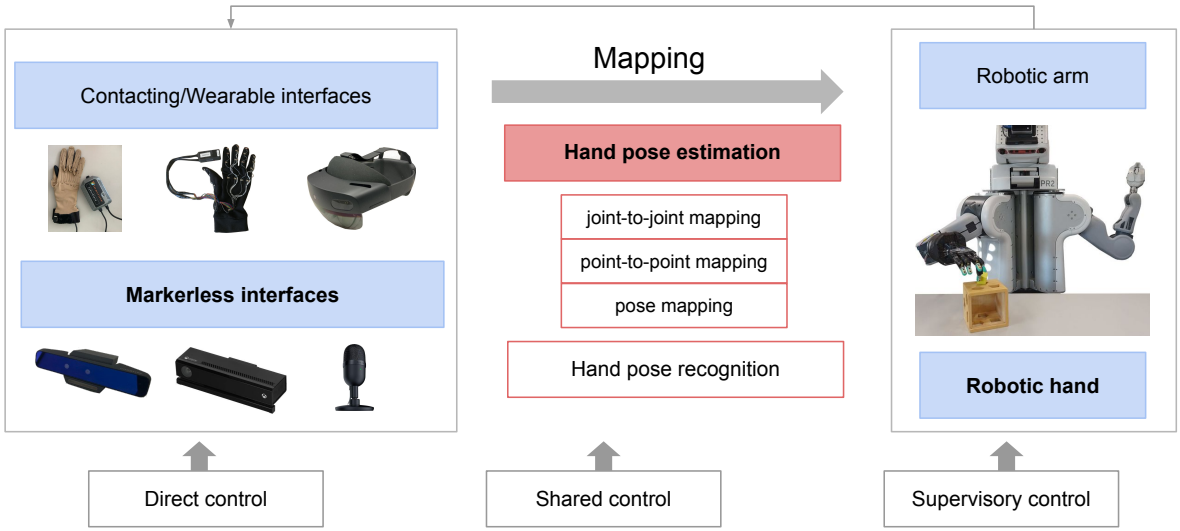


Figure 2.1 – Overview of state-of-the-art of robotic teleoperation and data-driven 3D hand pose estimation. The teleoperation system first obtains human intention using specific interfaces at the local site (contacting/wearable interfaces and markerless interfaces), then employs a suitable mapping method (hand pose estimation and hand pose recognition) to generate robot commands based on three control architectures (direct control, shared control and supervisory control). The data-driven hand pose estimation methods will be investigated extensively because of their teleoperation applications to dexterous robot hands.

2.1 User-Level Interfaces at the Local Site

In general, the teleoperation interfaces at the local site are used to measure human movements, generate control commands, or render remote feedback. Based on whether the human body contacts the interfaces, the local site's user-level interfaces fall into two main categories: contacting/wearable interfaces and markerless interfaces.

2.1.1 Contacting/Wearable Interfaces in Teleoperation

Robotic teleoperation has usually been implemented through different types of contacting/wearable user-level interfaces such as joystick, data glove, Inertial and Magnetic Measurement Unit (IMU), Electromyography (EMG) signal sensors, Electroencephalography (EEG) devices, immersive devices, and up-and-coming haptic devices. Unlike other interfaces used to sense human motions, the haptic devices assist the users in perceiving the tactile feedback from the robot side [14].

From simple to force-reflection joysticks, they are manipulated by humans and generate position, velocity, or acceleration commands for robots that have limited motion types, such as unmanned aerial vehicles and mobile bases [31]. Data gloves based on marker detection, bending sensors, or other technologies are the traditional input devices for dexterous hand teleoperation. Marker-based data gloves are assembled with several markers, such as LEDs (Light-emitting diodes), in a well-designed pattern on them, so that the human motion can be

recognized and captured by a sophisticated motion capture system [W29]. Motion capture systems provide accurate tracking data but can be expensive, and the correspondence problem between markers on the fingers and cameras still needs to be solved. Cerulo et al. [24] controlled an under-actuated anthropomorphic hand by marker-based hand tracking and motion mapping from the human hand to the robot hand. The marker-based hand tracking was established on OptiTrack motion tracking technology [W2] and a dynamic labeling algorithm that is robust to noise, outliers, and loss of markers. Another typical motion capture data glove is CyberGlove II [W8], which measures the 22 joint angles of the user's hand based on resistive bending sensors. Bernardino et al. [12] teleoperated the Shadow dexterous hand [W34] and the iCub hand [W38] by the CyberGlove II and recorded robot hand motions when the robots grasped a series of objects using different precision grasp types. However, the data gloves are customized for one human hand. Hence teleoperation based on data gloves cannot be easily generalized between users.

IMU-, EMG-based wearable interfaces are worn on human arms or human bodies and required for good calibration. IMU-based interfaces are electronic devices consisting of an accelerometer, gyroscope, and sometimes magnetometer, and these devices measure the angular velocity and acceleration of human motions [46]. The major disadvantage of IMUs is that the devices are subject to accumulated errors over time. EMG-based interfaces detect electrical activities generated by muscle cells, and the recorded signals can be used to analyze the human movements [135]. However, EMG signals are only from superficial muscles and can be influenced by electrode placements and the users' age, healthiness, willingness and body fat, etc. Due to the limitations of IMU and EMG technologies, these two interfaces are limited to generating accurate control commands for high-DoF robot hands. In practice, IMU- and EMG-based devices are convenient to set up and efficient in controlling the manipulators with multiple degrees of freedom. Fang et al. [46] established a multimodal fusion algorithm of a self-designed IMU device to deduce the orientations and positions of the human arm and hand. The robot experiments demonstrated an efficient arm control of the UR5 robot and the SCHUNK arm but a shallow grasping and releasing task of the Barrett robotic hand. Zhang et al. [178] presented an intuitive teleoperation system using an EMG armband for controlling a pre-prosthetic hand and using IMU sensors for operating a UR10 arm. However, the forearm's EMG signals are decoded to classify only two hand motions (open and grasp), so the multi-fingered hand is merely capable of achieving simple grasping tasks. Though some work employs EMG signals, IMU data, and marker tracking to control a robot hand or a hand-arm system, the expected dexterous manipulation has not been achieved yet [41]. In summary, regarding dexterous teleoperation, glove-based interfaces must be customized and easily obstruct natural joint motions, and IMU-based and EMG-based methods provide less versatility and dexterity. On the other hand, regarding the teleoperation of a multiple-DoF robotic arm and low-DoF, wearable interfaces are convenient to implement and efficient.

Teleoperation using Virtual Reality (VR), Mixed Reality (MR), or Augmented Reality (AR) interfaces has been gaining considerable attention in robotics [159] due to the benefits of immersive interactions and enhanced perceptual information. The holographic interfaces are usually head-mounted and equipped with multiple sensors, e.g., cameras, depth sensors, IMU, and microphones. For that reason, the holographic interfaces usually support hand tracking, eye tracking, and speech recognition except for 3D environment reconstruction. Zhang et al. [179]

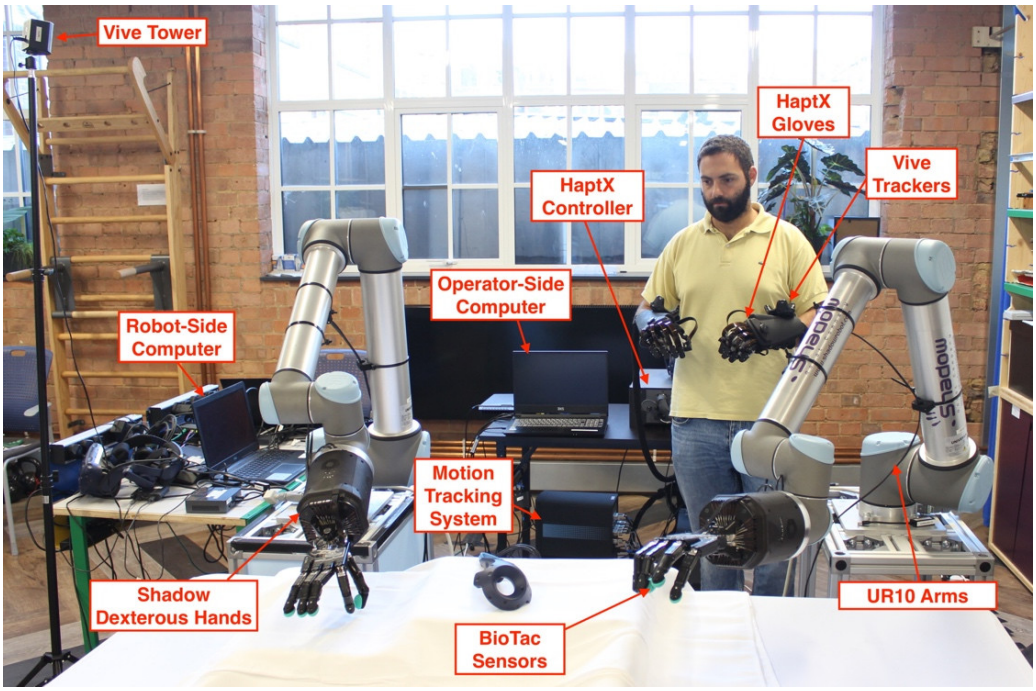


Figure 2.2 – Hardware components of the tactile telerobot. The human teleoperates two UR10 arms mounted with each Shadow hand. A BioTac sensor from SynTouch is installed on each finger of both Shadow hands. The HaptX gloves provide realistic force feedback mapped from the BioTac sensor. The Vive trackers are used to track human wrist poses [52]. Reprinted Image: ©2018 IEEE.

used a Vive VR headset and two hand controllers with 6D pose tracking to teleoperate a PR2 robot while recording the states or actions of the robot during the execution. Then the collected data is used to derive a learning policy that automatically reproduces or even generalizes the demonstrated behaviors. With a Microsoft HoloLens, Krupke et al. [83] introduced an MR-based human-robot collaboration system where the operator can intuitively see the co-located robot in its real physical surroundings and visual cues about planned trajectories. Once the human agrees with the planned trajectories observed from the HoloLens, the robot will autonomously complete the whole process of a pick-and-place task. Furthermore, there is some work using haptic interfaces and augmented reality simultaneously in teleoperation applications [60, 32], providing the user with an immersive perception when remotely controlling the robot system to carry out contact-rich, dexterous manipulation tasks [73, 72]. In 2019, one of the most advanced teleoperation systems, the bimanual tactile telerobot (see Fig. 2.2) built by the Shadow Robot Company [W35], HaptX inc [W7], and SynTouch Inc [W37], showed expressive performances on several in-hand manipulations tasks, e.g., Rubik cube rotation, and VR applications [52]. Through the use and tests of this telerobot system, the researchers concluded that tactile feedback or "getting a feel" for a task makes the task more accessible, intuitive, and fluid for the operators. However, the benefits of haptic feedback by the HaptX glove come with a heightened cost and additional bulk to the user.

2.1.2 Markerless Interfaces in Teleoperation

Compared to wearable interfaces, markerless interfaces have the advantages of allowing for natural, unrestricted body motions and of being less invasive with different teleoperators. Markerless vision-based teleoperation is suited to dexterous teleoperation, which requires capturing all the essence of finger motions [80]. Low-cost RGBD cameras are widely used in most markerless vision-based teleoperation methods.

Classic markerless vision-based teleoperation methods fall into two categories: model- and appearance-based approaches. Model-based approaches provide continuous object poses, but they are computationally costly and highly dependent on the availability of a multi-camera system. [39] computed continuous 3D positions and orientations of thumb and index finger from segmented images using a camera system. The end-effector position of the experimental robot arm was mapped by a differential positioning method, and its orientation was in accordance with the orientation formed by the thumb tip and index fingertip. Conversely, appearance-based approaches identify a discrete number of hand poses typically included in the training set, and these approaches are without high computational cost and hardware complexity. Romero [130] classified human grasp types from a single RGB image by finding similar hand gestures in a large database with grasp images and estimated the hand pose by the same grasp recognition module. Then they mapped the human grasps to a discrete set of corresponding robot grasp classes and implemented grasping experiments on a three-fingered Barrett hand and a five-fingered ARMAR-IIIb hand.

With the rapid expansion of deep learning methods, leveraging image processing algorithms like hand pose estimation or object segmentation is becoming a new trend in the robotic community. In vision-based teleoperation, a lot of research is focused on the visual perception of human bodies (e.g., human gesture classification or human hand pose estimation) by deep learning methods, then starts to consider the robot control (e.g., specific motions or kinematic retargeting). The kinematic retargeting takes the body detection results from visual perception algorithms and generates the robot commands in joint space [5]. Michel et al. [105] provided a teleoperation method for a NAO humanoid robot that tracked human body keypoints from markerless visual observations then calculated the joint angles of the humanoid robot through inverse kinematics. Antotsiou et al. [5] used a hierarchical hybrid hand pose estimator formulated on spatial attention deep network and partial Particle Swarm Optimization (PSO) and proposed a task-oriented retargeting method to achieve the teleoperation of an anthropomorphic hand. However, all robot experiments were carried out in simulation. Similar to [59], they designed the HandLocNet neural network to detect the hand in the RGB image and used a network called HandPoseNet to accurately infer the three-dimensional position of the joints retrieving the full hand pose. Recently, Handa et al. [63] collected a human hand pose dataset with hand poses required for dexterous manipulation and trained a PointNet++ inspired hand pose estimation model on a self-built dataset, which has rich hand postures relevant to dexterous manipulation. Combining with a fingertip-prioritized kinematic retargeting method, they built a 23-DoF hand-arm teleoperation system, DexPilot. This system achieved impressive results in dexterous manipulation, e.g., block stacking, cup insertion, and showed the effectiveness of the markerless vision-based teleoperation for highly-actuated hand-arm system (see Fig. 2.3). However, in this setup, the robot workspace was fully determined by the human workspace.

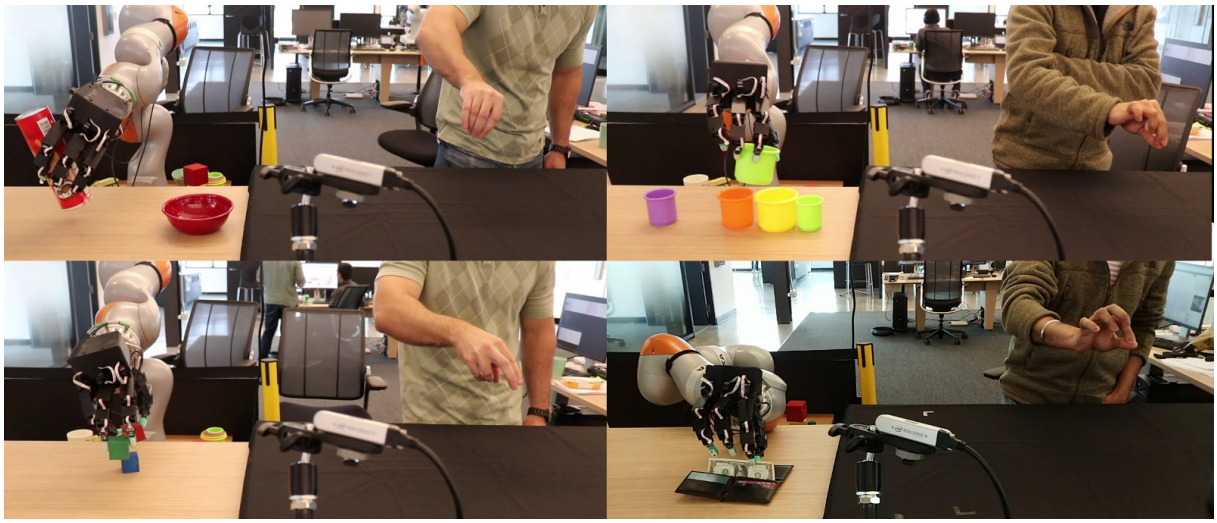


Figure 2.3 – The teleoperation examples of the DexPilot system. Six RGBD cameras observe the human hand, and the robot system consists of a KUKA LBR iiwa7 robot and a Wonik Robotics Allegro hand. This system enables teleoperation across a wide variety of tasks, e.g., pick and place, cup inserting, concurrent two-cubes-picking, and money extracting [63]. Reprinted Image: ©2020 IEEE.

The methods mentioned above do not consider the physical constraints and joint limits of the robots in the pose estimation stage, so they are inclined to generate poses that the robot cannot reach. In addition, these methods strongly depend on the accuracy of the hand pose estimation or the classification, with time-consuming post-processing. In contrast, an end-to-end regression model that takes human hand images as inputs and predicts the robot joint commands is an alternative choice. The end-to-end learning refers to training rather deep and complex neural networks by applying gradient descent to the system as a whole [15]. In our case, the end-to-end learning bypasses the intermediate retargeting process and directly targets the robot system. Therefore, the physical constraints of the robot are directly considered in the target space. Besides that, the end-to-end model is more intuitive for novice demonstrators and saves post-processing time in practice. Fang et al. [45] proposed a human-robot posture-consistent-based end-to-end neural network for teleoperating a 7-DoF Baxter arm. The network comprises three modules: skeleton point estimation, robot arm posture estimation, and robot joint angle generation. However, only simulated arm imitation experiments were demonstrated. Until this thesis, the end-to-end deep learning structure has not been utilized for a dexterous robot hand with multiple DoFs.

The other popular markerless interface is the microphone. Audio-based teleoperation is the most intuitive manner and is commonly used in human-robot interaction applications [1]. Natural language processing algorithms transfer human speeches to specific and discrete robot commands. Usually, these robot commands only cover a small variety of tasks and are employed on robots with high-level autonomy. Recently, the touchless haptic device STRATOS [W43], which simulates the sense of touch by turning ultrasound into mid-air haptic textures, has been released. The ultrasound waves are generated by multiple ultrasound speakers and coincide in 3D space. With hand pose tracking technologies, the tracked human hand will feel a force spot positioned by the coincided point. This combined touchless interface is envisioned to be

used in many applications, such as future kiosks, braille reading, video games, and markerless robotic teleoperation. Since current ultrasound speakers on STRATOS are displayed on one plane, employing force on the human hand when the fingers are folded is unsolved. Moreover, how to achieve point-to-point tactile mapping from the tactile sensors on the robot hand to the coincided points generated by ultrasound waves is to be studied.

2.2 Human-Robot Motion Mapping

How to map human motions to a robot depends on the user-level interfaces, robot platforms, and working scenarios. In this section, two issues are discussed: How to obtain human intentions from the sensory data? Which mapping methods are commonly used to map human intentions to the control commands? Human pose estimation and human pose recognition are two theoretical solutions for obtaining human intention. These two methods and the corresponding mapping methods are introduced in detail in the following two subsections.

2.2.1 Human Pose Estimation

Pose-estimation-based mapping methods receive the real-time human motion from the sensors and generate continuous kinematic parameters (i.e., position of body keypoints, link orientation, or joint angles) of the human body. Regarding wearable interfaces, such as data gloves, IMU-, EMG-based wearable suits, usually, the human body status can be directly read from the sensors. If users take a camera to capture their body or hand, pose estimation methods in computer vision are needed to predict real-time poses. The state-of-the-art of data-driven vision-based hand pose estimation methods are analyzed in section 2.4.

Conventional teleoperation mapping methods are divided into three main categories: joint-to-joint mapping, point-to-point mapping, and pose mapping. In most cases, however, only considering one type of mapping method is not enough [25]. For example, point-to-point mapping for a dexterous robot hand neglects the position and orientation of the phalanges and does not consider the special mechanical difference between the robot and the human.

Joint-to-joint mapping seeks to map the joint angles of a human body to those of a robot. The mapping method decreases the effect of the link difference between the human and the robot, and it is suitable for the cases where the human and the robot have similar kinematic mechanisms. In terms of teleoperating a robotic hand, joint-to-joint mapping hardly enables the robot to conduct in-hand manipulation and precision grasping. Kobayashi et al. [78] wore a CyberGlove II glove to control the thumb and first finger of an anthropomorphic robot hand [79]. Later, they developed a teleoperation method that a target joint of a robot hand was controlled by multiple human hand joints and corrected the sensor readings of the data glove by the generic algorithm to reduce the structural difference between the robot and the operator's finger [78]. Nevertheless, only grasping experiments by the thumb and first finger of an anthropomorphic robot were presented.

Point-to-point mapping prioritizes the robot positions in the task space and expects the robot to position the end-effectors at the same location that the human reaches. Regarding a robotic hand, the end-effectors are the fingertips, so the point-to-point mapping is also called fingertip

mapping. External devices can calculate the positions of human fingertips, then these positions are converted to joint angles of the robot by calibration between the human hand and the robot hand and inverse kinematics. For the multi-fingered hands which have less than five fingers, the extra human fingers will be directly ignored. The fingertip mapping is suitable for precision grasps [129]. However, when the length of the robot and the human fingers are not similar, fingertip mapping easily causes weird finger poses and unsmooth control. An alternative to fingertip mapping is virtual object mapping. The fingertips on the human hand define a virtual sphere, whose motion and strain are imposed on the virtual sphere relative to the robotic hand in the Cartesian space [57]. The virtual object mapping method can also be used to map synergy from human to robotic hands with dissimilar kinematics [12]. However, it is limited to manipulation applications using similar grasp types and regular-shaped objects.

Pose mapping tries to interpret the functions of human motion rather than to replicate end-effector positions or joint angles [161]. In pose mapping, the human poses are usually associated with predefined robot poses. Since the predefined robot poses are discrete, this mapping method is widely applied to a task scenario that only needs a few commands. For instance, the robots conduct repetitive tasks in constructed environments, or the humanoid robots are used for social human-robot interaction. For robotic hand teleoperation, some work projects human poses into a low-dimensional space based on human synergies to allow for continuous pose mapping. Meeker et al. [103] proposed decomposing a hand shape into three basic vectors. They are how wide the fingers are spread, how big the object can be grasped, and how curled the fingers are. Under this decomposition principle, they project the human hand poses into this 3D teleoperation subspace as an intermediary, then remap the teleoperation subsapce into the robot joint space. However, this method requires that robot and the human to start from the same initial state. In summary, the pose mapping is suitable for anthropomorphic and non-anthropomorphic robot hands but has difficulty affecting posture recognition and dexterous manipulation.

2.2.2 Human Pose Recognition

Pose recognition is a classification problem, while pose estimation is a regression problem. Pose recognition is somehow equivalent to pose mapping if only static and discrete poses are considered. In pose-recognition-based mapping methods, a classifier is required. The classifier gradually updates from single input to multiple sensing, from machine learning methods to Convolutional Neural Networks (CNNs) [38]. Wolf et al. [162] classified sixteen hand gestures by a multi-class support vector machine classifier. The input of the classifier is EMG and IMU data acquired from the user's hand and arm. Simao et al. [140] compared different commonly used classifiers with proper data dimensionality reduction to achieve accurate gesture recognition in real unstructured environments. Then the trained model was used to teleoperate a robot arm to prepare a breakfast meal.

In general, pose recognition is a high-level representation, thus leading to a high-level control strategy of the robots. On the one hand, pose-recognition-based mapping reduces the risk of damaging the robot hand and decreases the mapping difficulty. On the other hand, discrete commands imply that the robot cannot get continuous control. In dexterous robot hand teleoperation, some researchers design a useful gesture set based on grasp taxonomies or manipulation synergy to increase the manipulation capacity and generate high-quality robot

commands. Higashi et al. [65] proposed a functionally divided manipulation synergy method, which only applies synergy control to the fingers with the function of manipulation instead of to those that are locking or supporting. In this way, the synergy-based control could generate robot commands in low dimensions for achieving dexterous manipulation.

The human-robot motion mapping methods mentioned above do not include a robot teleoperated by interfaces that can directly generate position, velocity, or acceleration commands for the robots, e.g., control panels, joysticks, or keyboards. Hence, no specific mapping methods are required here.

2.3 Control Architectures in Teleoperation

Depending on how much robot autonomy is involved in the teleoperation process, the control architecture of teleoperation is divided into direct control, shared control, and supervisory control [112]. An illustration of these three control architectures is shown in Fig. 2.4.

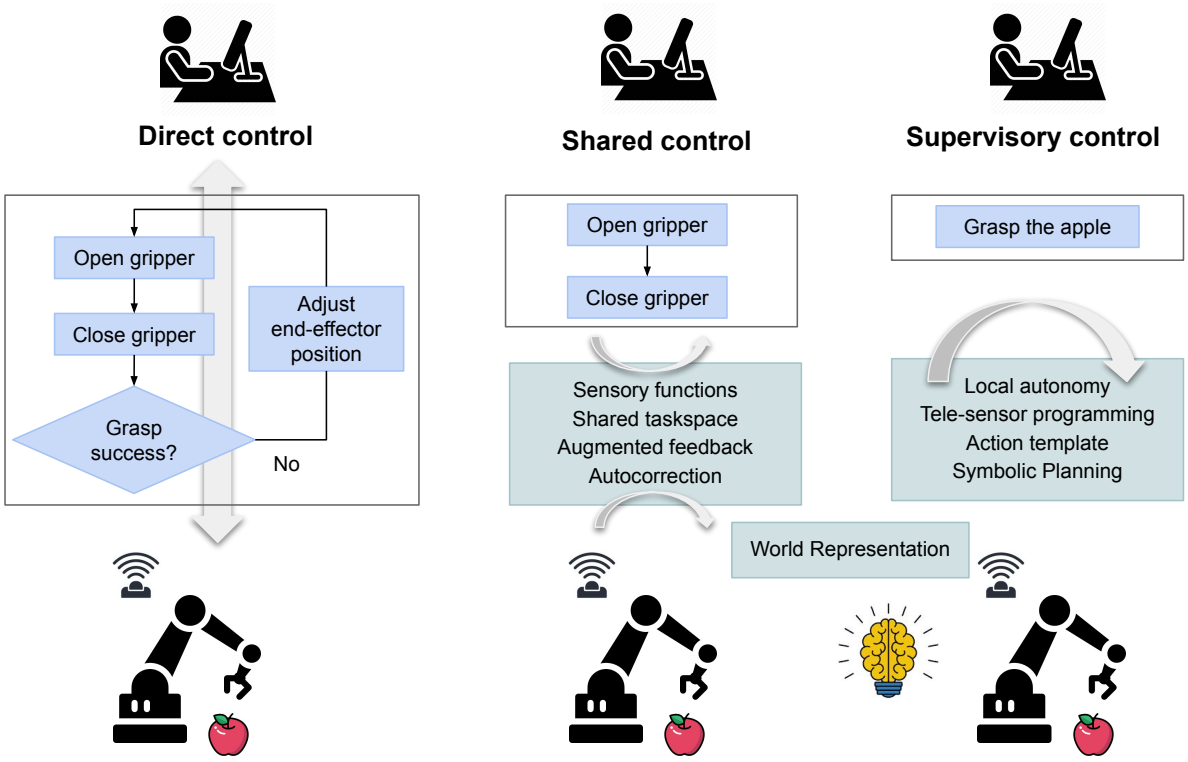


Figure 2.4 – Three control architectures in teleoperation. The texts in the blue rectangles are the typical examples of operator commands in an apple grasping task.

Most teleoperation systems employ direct control, where the operator directly controls the robot's motion without any local autonomy. The control commands usually are position, velocity, or acceleration, depending on the robot's control mode. The position commands for most robots could be generated by the mapping methods discussed in section 2.2. Velocity and acceleration control are attractive when the local master and the remote robot are fundamentally different kinematically or in their workspace. For example, the human

(a) EDAN (b) Justin (c) HUG

Figure 2.5 – Different teleoperation control architectures in three robotic systems from DLR. (a) an assistive robotic system EDAN; (b) a humanoid robot Justin used in household work and space station (c) a bimanual haptic device HUG for telemanipulating the humanoid robot Justin. Reprinted images: ©DLR, CC BY-NC-ND 3.0.

controls mobile robots, drones, or underwater robots. Acceleration control is less accurate than velocity control since regulating a second-order system is harder than a first-order system by humans. Usually, the velocity or acceleration commands may be proportional to the input device, e.g., a spring joystick. The direct control is intuitive and guarantees the safety of the robot system, but the continuous control requires a high workload of the teleoperator.

Unlike direct control, shared control inserts autonomous abilities, e.g., user intention detection, possible safety guarantees, joint regulation, commands overlay, or autonomous motion correction into the robot system. The slave can fine-tune its movements in the shared control framework when the control commands are inaccurate or with large communication delays and limited bandwidth. Dwivedi et al. [42] telemanipulated a robot arm to execute a whiteboard cleaning task by shared control framework. The human wrist controlled the robot end-effector's position, and a compliance control was used to guarantee that the desired contact force would always be maintained on the whiteboard surface. In [100], the autonomous obstacles avoidance of the controlled mobile robots was achieved by artificial potential field developed on EMG signals of the human arm and force feedback from the mobile platform. The shared control method enables the human operator to telecontrol the robot motion and enables the robot to avoid obstacles synchronously. With MR technologies, once the robot intended position is estimated, the users could visualize the robot trajectories, then the robots move along the accepted trajectory in an autonomous manner [173]. The robotic assistive system EDAN in Fig. 2.5(a) converts user commands in a target-oriented manner, effectively reducing the number of coordinates that users need to control in high-DoF tasks [W16]. A special implementation in shared control is virtual fixtures. Virtual fixtures are usually virtual fields, barriers, or guidance that are superimposed into the visual and haptic scene of the users. They provide the users with preknowledge of the

task system and guide the robot to move along desired paths or through restricted regions. For instance, an immersive telerobotics system with multimodal virtual fixtures was verified to improve operator performance in a pick-and-place task, in which have multiple obstacles [82].

Supervisory control requires only high-level commands from the human and allows the robot more autonomy and intelligence. The remote site continually sends summary information, working status, or planned motions to the users, and it may close the control loop through multiple sensors [138]. Supervisory control is quite useful in telerobotic systems with large time delays of a few seconds or more, e.g., in space, and undersea applications. In this case, a sophisticated predictive simulation system should be provided to emulate and predict the remote system, including all sensory perception. Supervisory control requires that the robot has the knowledge of the objects in the remote environment, the skills of symbolic and geometric planning, and the ability of self-localization and navigation. The humanoid robot Justin (see Fig. 2.5(b)) receives highly abstract commands and works under supervisory autonomy in elderly care and space station scenarios [W20]. To allow Justin to understand the working environment, researchers added numerous objects and action templates to the object database. Given the object information, a hybrid framework was utilized to solve the task symbolically and to find a suitable geometric solution using robot-specific planning modules. Then, an A-star path planning algorithm was executed based on a 3D map of the indoor scene and particle-filter-based indoor localization [156].

In addition, these three control strategies are often combined to get the most efficient and stable control. For example, Tanwani et al. [152] presented a probabilistic formulation to recognize the teleoperator intentions and subsequently assisted the teleoperator by time-independent shared control and/or time-dependent supervisory control of the model. In pursuit of telepresence and excellent task performance, bilateral control can be added into any of the control strategies mentioned above [112]. A typical bilateral control system additionally requires tactile sensors sensing the contacting in the remote environment and devices representing force feedback (e.g., haptic device, audio display) in the local site. Whilst the additional force feedback promotes operators to make more accurate decisions, the multiple feedback loop and potential communication delay make the control architecture particularly challenging [10]. For example, the haptic input system HUG [W18] is designed to teleoperate EDAN or Justin system and allows for different control strategies. HUG is composed of two DLR robot arms, an optical tracking system, a pair of data gloves, a head-mounted display, and a multi-layer security architecture (see Fig. 2.5(c)). A force-torque sensor is integrated at the end-effector of each robot arm and a haptic device is attached at the human hand. The setup aims to enable the user to perceive the remote environment with his/her own scenes (visual, audio, haptic) on an immersive and transparent level.

2.4 Data-driven 3D Human Hand Pose Estimation

Hand pose estimation is a typically essential research topic in vision-based teleoperation, which is also a well-studied topic in computer vision. Hand pose estimation is to estimate the full DoF hand poses that target the kinematic parameters (i.e., joint angles, hand position, or orientation) of the hand skeleton. The vision-based methods take a single image frame or multiple image frames as inputs, then estimate the high dimensional hand poses. Fig. 2.6 provides a graphical comparison among three similar fields to hand pose estimation. Hand pose recognition focus on classifying a set of discrete hand gestures; hand detection aims to locate the hand by a point on the hand or a bounding box area of the hand; hand segmentation algorithms are used to segment the human hand from the background, and it always serves as an initial step of hand pose estimation or recognition.

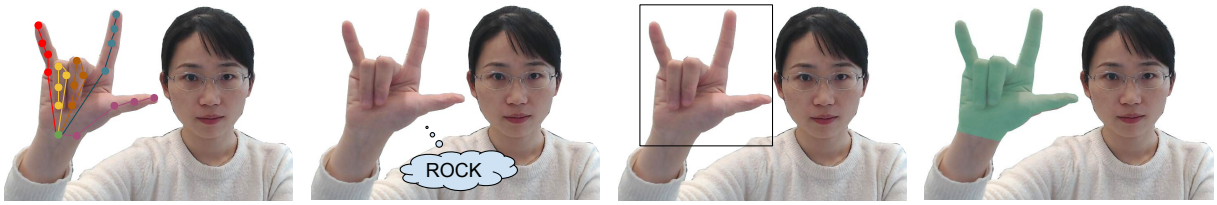
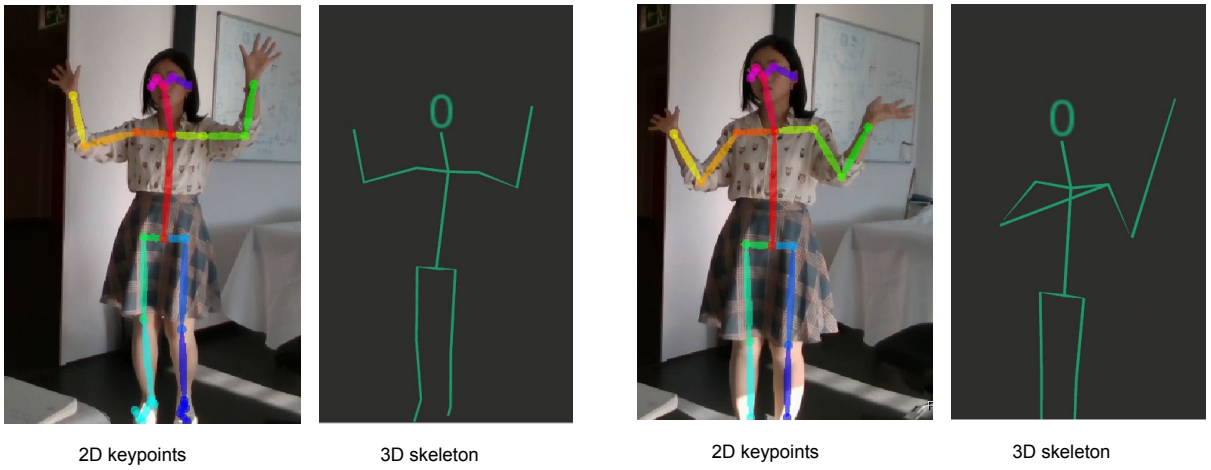


Figure 2.6 – Comparison of hand pose estimation and its similar fields. Adapted from [88].

Compared with human body pose estimation, hand pose estimation investigates human fingers, which are more likely to be occluded in a local area and are more flexible to construct various gestures than human bodies. Fig. 2.7 shows the 2D human body keypoints predicted by the state-of-the-art 2D human pose estimation model Openpose [21] and the 3D pose results calculated by the 2D keypoints prediction and the filtered depth information from an RGBD camera. The Openpose model performs well in 2D space, but simply using filtered depth information to construct 3D skeleton is not precise. Similarly, while 2D hand pose estimation methods estimate the poses of the hand joints relative to the image coordinate, 3D hand pose estimation is more challenging and predicts the hand keypoints in a 3D world coordinate. And 3D hand pose estimation is more piratical in interactive games, animation, VR, and robotics. There are some challenges that remain in 3D hand pose estimation [43]:

- Self-occlusion. As the human hand is quite flexible, its projection results in a variety of shapes with a lot of self-occlusions.
- Similar appearance. The five fingers have similar appearances, which causes regression difficulties in distinguishing them.
- Hand segmentation. When the hand is manipulating an object or multiple hands in the scene, the hand segmentation and hand pose estimation are intractable.
- Real-time performance. A human hand's translational and rotational speeds could be up to 5 m/s, and 300 $^\circ/s$. However, the frequency of a standard depth camera is 30-60 Hz. Apart from that, it is challenging for many algorithms to achieve a 30 FPS estimation speed.



(a) Example 1 (b) Example 2

Figure 2.7 – 2D human pose estimation by Openpose and 3D skeleton estimation calculated by the 2D keypoints and the filtered distance information from an RGBD camera.

Since one algorithm hardly satisfies all challenges simultaneously, researchers always focus on one particular aspect in hand pose estimation. In this thesis, all discussed methods work under some restrictive assumptions: only one hand is involved and does not interact with any objects; the hand is observed from one viewpoint.

Hand pose estimation methods can be categorized into model-based estimation and data-driven estimation [26]. Model-based methods formulate an optimization problem whose cost function measures the distance between the observed hands and the hands constructed by a generative hand model [117]. These methods start from a selected human hand model with initialized kinematic parameters. Then the hand features (e.g., edges, silhouettes, and optical flow) are extracted from actual human hand images and the generated hands by feature detection algorithms, e.g., scale-invariant feature transform [99]. The next step is to measure the similarity of these two hand features based on a well-defined objective function. At last, the optimal kinematic parameters of the hand model which fit the hand in the images are found by optimization algorithms, e.g., PSO [75] and Iterative Closest Point (ICP) [13]. In a word, model-based methods resolve an optimal problem by hand-crafted initialization and iterative search. The initialization parameters are typically the solution from the previous frame. Therefore, model-based methods easily cause pose-drifting issues owing to the accumulated estimation errors along the running process.

In recent years, data-driven estimation has become dominant with the development of computing techniques and the rise of deep learning. This thesis focuses on data-driven hand pose estimation methods using CNN. Data-driven methods learn a direct mapping from the hand images to the target parameter space by a discriminative classifier or regression model based on multiple annotated hand poses. Apart from the network architecture, the quantity and diversity of the training datasets also determine the quality of discriminative models. This is due to the fact of supervised learning. Even though data-driven methods avoid pose-drifting problems, a new challenge regarding annotation also arises in this research field. These annotation difficulties

cause research on synthesized hand datasets and weakly supervised learning in hand pose estimation [35].

- Annotation difficulties. Supervised learning relies on many labeled datasets, but annotating human hand poses is a tedious and time-consuming process. To get accurate positions of human hands, an expensive motion tracking system or a multi-camera system is required, and sophisticated computation operations are required to annotate the position of the hand keypoints.

In subsection 2.4.1, several state-of-the-art datasets are listed and compared. Then the following two subsections introduce data-driven hand pose estimation algorithms from the aspects of input, network structure, and output. In the end, the evaluation metrics of hand pose estimation are discussed in subsection 2.4.4.

2.4.1 Human Hand Pose Datasets

Many human hand pose datasets have been proposed for benchmarking hand pose estimation. The datasets usually provide the depth images or the RGB images of the human hand and the annotated 3D keypoints of hand joints by multiple subjects. They differ in scale, annotation methods, annotation accuracy, articulation, viewpoints, and occluded objects. The benchmark comparison of the existing datasets is listed in Table 2.1. With the development of technology and research in this field, the hand pose datasets tend to become larger, more precise, more complex (with different backgrounds, illumination, viewpoints), and generated by full automation. Apparently, the depth source is more popular than the RGB image as it has the additional dimension of distance information and has good resistance to color and illumination change in the scene. The commonly used commercial depth sensors are Microsoft Kinect [W10] and Intel RealSense camera [W9]. Meantime, the procedure of dataset collection and the corresponding automatic annotation methods are gradually open-sourced. Moreover, there has been increasing attention on synthetic datasets because of the application of generative models such as Generative Adversarial Network (GAN) [61] and the realistic rendering of the simulators.

Dataset	source	Annotation	No. frames	No. joints	No. subjects	Viewpoint	Year
Dexter 1 [146]	real RGBD	manual	2,137	5	1	3rd	2013
MSRA14 [126]	real depth	manual	2,400	21	6	3rd	2014
ICVL [151]	real depth	track + refine	17,604	16	10	3rd	2014
NYU [153]	real depth	track + refine	81,009	36	2	3rd	2014
MSRA15 [148]	real depth	track + refin	76,375	21	9	3rd	2015
HandNet [158]	real depth	automatic	212,928	6	10	3rd	2015
Graz16 [114]	real depth	semi-automatic	2,166	21	6	ego	2016
Simon et al. [141]	real RGB	manual	15K	21	10	full	2017
BigHand2.2M [172]	real depth	automatic	2.2M	21	10	full	2017
InterHand2.6M [109]	real RGB	semi-automatic	2.6M	21	27	3rd	2020
RHP [184]	Synth. RGBD	synth.	44K	21	20	3rd	2017
Mueller et al. [111]	Synth. depth	synth.	80K	16	5	full	2019

Table 2.1 – Comparison of existing 3D hand pose estimation datasets

2.4.2 Input Modality in Hand Pose Estimation

Traditionally, using a depth map to infer hand keypoint positions is the main approach in hand pose estimation, as a depth map represents a 2.5D surface information embedded in 3D space and is insensitive to shadow and illumination. The promotion of depth-map-based methods also benefits from the emergence of cheap commercial depth cameras, e.g., Kinect. A classical work using depth inputs is DeepPrior [116], which led to the trend of 2D CNN-based methods in hand pose estimation.

RGB images make the model easily generalized to many implementation scenarios, but extracting 3D information from a pure 2D input is a highly nonlinear mapping, which causes difficulty in the learning procedure. The first learning-based system to estimate pose from single RGB images was proposed by Zimmermann et al. [184]. This system first localized hand keypoints in the 2D images, then derived the 3D hand poses within a canonical coordinate frame, and additionally estimated the transformation into the canonical coordinate frame. Although the system shows almost competitive performance to methods using depth maps, its accuracy is limited by the lack of large-scale datasets.

Since depth images are intrinsically 3D data, some work converts 2D depth images into 3D voxel or point clouds in order to overcome the perspective distortion-invariant estimation and avoid nonlinear mapping. Moon et al. [108] firstly cast the hand pose estimation problem into a voxel-to-voxel prediction using a 3D CNN called V2V-PoseNet. Ge et al. [55] proposed Hand PointNet, which takes normalized point clouds as input and regresses the hand keypoints in a low dimension by the PointNet-based network model [125].

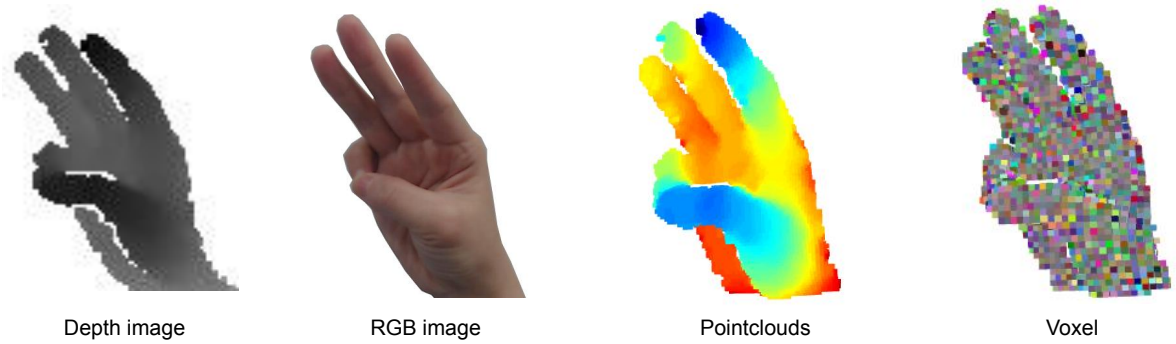


Figure 2.8 – Common four input modalities in 3D hand pose estimation.

The common four input modalities in hand pose estimation are exampled in Fig. 2.8. Generally, RGB-based methods perform worse than depth-based methods. Besides, the 3D volumetric representations outperform the depth map because they better exploit the 3D spatial information. However, models using 3D inputs take a longer inference time because the network structure is generally more complex. So far, only algorithms using one modality as the network input have been reported; ones with multiple modalities have not been discussed yet. Inspired by multimodal learning, Kazakos et al. [74] proposed a double-stream learning architecture for hand pose estimation. The two streams took a depth image and an RGB image, respectively, and are trained in parallel. The separate training features were fused after the last convolutional layer.

However, the results suggested that the double-stream network performs similarly with a network trained only with depth images. Later, other work explored the pose-related latent space from different modalities by Variational Autoencoder (VAE) networks and proved the performance improvement brought by multiple modalities, e.g., hand skeletons, heatmaps, and segmentation masks [145, 168]. For example, hand skeletons are an easy and intuitive way of manipulating data entries [9]. Heatmaps of the 2D hand key points on the RGB images are chosen as an additional modality to promote convergence of the RGB encoder, since the heatmaps are closely related to activation areas on the RGB images [168].

Hand Segmentation

The input of most hand pose estimation models is a cropped and clean image or a 3D representation containing a bare human hand with specific gestures. Therefore, hand segmentation is an indispensable preprocessing step before the hand pose estimation. To simplify the segmentation task, there is a lot of work extracting a fixed-size hand cube centered on the mass center in the depth image, assuming the hand is the closest object to the camera [116]. However, many research projects developed a hand detector or a per-pixel semantic segmentation model by machine learning or deep learning methods. Tompson et al. [153] classified each pixel in a depth image as belonging to a hand or background by training a randomized decision forest (RDF) model. As object detection and semantic segmentation are popularly implemented on RGB images, some work uses RGB images to capture the hand region then feeds the obtained depth images into the training model. Panteleris et al. [119] adapted a hand detector from the YOLO (You Only Look Once) model [127], which is a state-of-the-art real-time object detector. Derived from well-known semantic segmentation models such as SegNet [8], [37] and [184] trained pixel-wise hand segmentation networks on human hand datasets.

2.4.3 Network Structure in 3D Hand Pose Estimation

Regression-based Methods and Detection-based Methods

According to the representation of the output pose, the 3D hand pose estimation methods consist of regression- and detection-based methods. Regression-based methods directly map the depth image to the joint locations or the joint angles of a hand model, while detection-based methods give the probability density map for each joint.

The regression-based method DeepPrior [116] extracted a prior knowledge of hand models by designing a lower-dimensional bottleneck in the last layer, significantly improving the joint estimation accuracy. A follow-up work is DeepPrior++ [113], which introduced ResNet [64] layers into the model, conducted data augmentation and better initial hand localization. The DeepPrior and DeepPrior++ models are comparable baselines in the hand pose estimation field. The architecture of DeepPrior++ model is shown in Fig. 2.9. Region ensemble network (REN) [62] is a tree-structured model taking depth images as inputs and consisting of a CNN trunk and five ensemble branches. The features from five regional branches are concatenated and used to regress the 3D coordinates of each hand keypoint. In the spirit of ensemble learning to improve the generalization ability, an anchor-based regression network for hand pose estimation from a single depth image was proposed in [166]. The densely sampled anchor points were trained towards a certain joint with different weights from different viewpoints and distances by an anchor

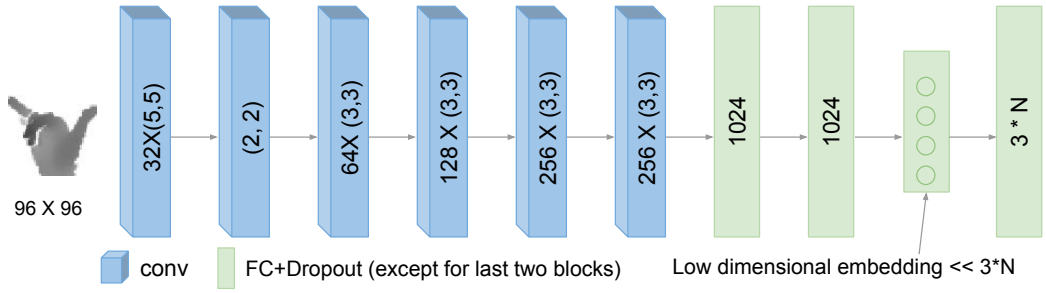


Figure 2.9 – Network architecture of the DeepPrior++ model. conv means a convolutional layer, FC denotes a fully-connected layer. N is the number of joints. The numbers in the convolutional layers are the number of filters and the filter size. The numbers in the FC layers are the number of neurons. The last layer computes the pose prior in a lower-dimensional space. Adapted from [113].

proposal network. As a result, the joint position is the aggregation of the outputs of all anchor points. Aside from the novel anchor-to-joint regression, the other highlight in this work is that the outputs are not the 3D positions of joints but a separate in-plain position and depth estimation.

Early work on detection-based methods firstly detected the joint positions in the 2D plane based on estimated 2D heatmaps then translated them into 3D space by complex optimization-based post-processing [153]. With the development of 3D CNNs, one-to-one mapping from a 3D input (e.g., point clouds) to a 3D output (e.g., 3D heatmap) is possible and allows to more accurately reconstruct the hand shape. In [56], point-to-point regression directly employed the 3D point clouds to a stacked hierarchical PointNet and produced point-wise estimations, i.e., heatmap, and unit vector fields on the 3D point clouds. The point-wise estimation manifests the proximity and direction of every point in the point clouds to the hand joint. An effective and powerful detection-based method is V2V-PoseNet [108]. It learns a one-to-one mapping that uses a 3D voxelized grid and estimates the per-voxel likelihood for each keypoint. Later, Malik et al. [102] integrated the V2V-PoseNet model into a novel 3D CNN architecture, which simultaneously estimates two different representations of 3D hand shape and 3D pose from a voxelized depth map. The two shape representations are voxelized grid and hand surface with hand mesh topology and number of vertices.

In general, detection-based methods outperform regression-based ones, but detection-based methods have a poor trade-off between accuracy and efficiency because the networks used for producing a probability density map for each joint are heavy-weighted. Therefore, most real-time hand pose estimation methods are regression-based.

Deep Generative Methods

Deep generative models (DGMs) are neural networks which are trained to represent complicated, high-dimensional probability distributions into low-dimensional hidden layers in a supervised manner or unsupervised manner [133]. The DGMs are often employed to estimate the likelihood of each observation and generate new samples from the underlying distribution. The two most commonly used models are VAE and GAN. Taking advantage of these features of DGMs, some hand pose estimation work explores latent hand poses through multiple modalities or synthesizes realistic hand images.

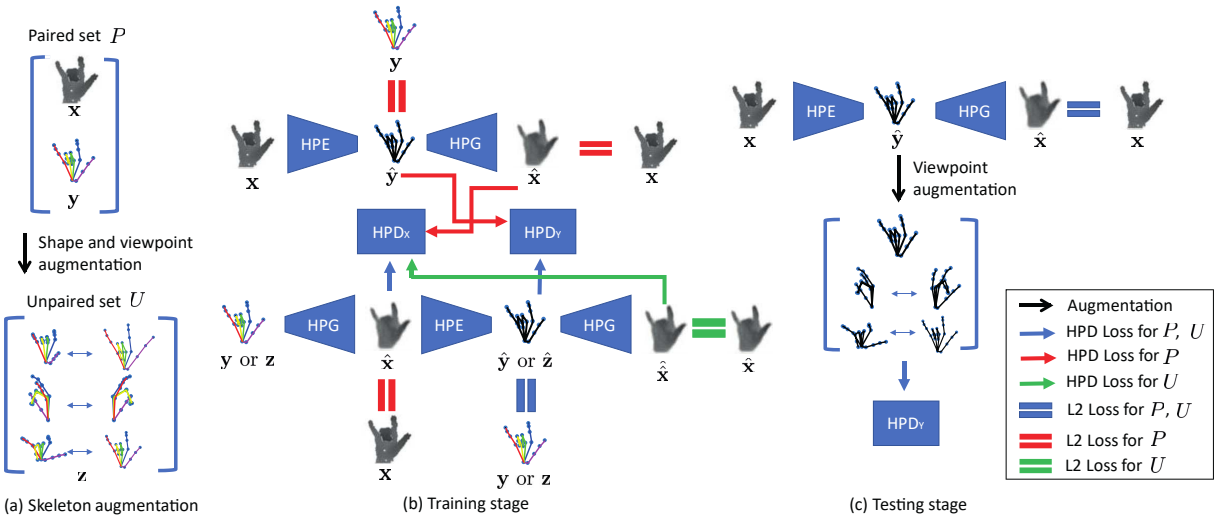


Figure 2.10 – Schematic diagrams of augmented skeleton space transfer for depth-based hand pose estimation. (a) indicates the paired training set P and unpaired training set U. (a) also shows that manipulating skeletons is easier than manipulating depth maps. (b) and (c) show the training and testing flowcharts. HPD_X and HPD_Y mean the depth hand pose discriminator and the skeleton hand pose discriminator, respectively [9]. Reprinted image: ©2018 IEEE.

The VAE-based framework developed on multiple modalities has been implemented in hand pose estimation. Spurr et al. [145] attempted to create a unified latent space for hand poses from multiple modalities, e.g., RGB images, 2D keypoint detections. They suggested a VAE-based deep neural network, which jointly optimizes the resulting crossmodal KL divergence (Kullback–Leibler divergence) and the posterior reconstruction objective across multiple modalities. However, its alternating training strategy from different modalities causes a slow convergence speed and tends to fluctuate as data is extracted from multiple modalities. Then, Yang et al. [168] developed a novel VAE-based framework, which aligns the pose latent space from individual modalities and leverages other modalities as weak labels to improve RGB-based hand pose estimation. Instead of training all modalities in one shared latent space, Yang et al.derived different loss functions for diverse modalities (point clouds and heatmaps) and presented two different ways to align their associated hand latent spaces. Experiments have shown that the resulting latent representation allows for high-quality hand pose estimation and realistic construction of hand point clouds from monocular RGB images.

The GAN-based models usually come together with synthesized datasets. To increase the realism of synthetic images, Mueller et al. [110] translated synthetic hand images to “real” images by a novel geometrically consistent GAN trained on unpaired hand images. Here, the GAN-based model does not participate in the pose estimation part. Baek et al. [9] built new database entries by synthesizing unseen human hands in the skeleton space because, unlike depth images, hand skeletons vary slightly with respect to mild variations in viewpoints and shapes. There are three network modules in this scheme, hand pose estimation (HPE), hand pose generator (HPG), and hand pose discriminator (HPD). Fig. 2.10 shows the schematic diagrams of this work. The HPG module reconstructs depth images from the skeleton information. The HPD module is a GAN-based framework that distinguishes real depth images from those synthesized by the HPG. The three modules were combined and jointly trained, which enabled the automatic transfer of

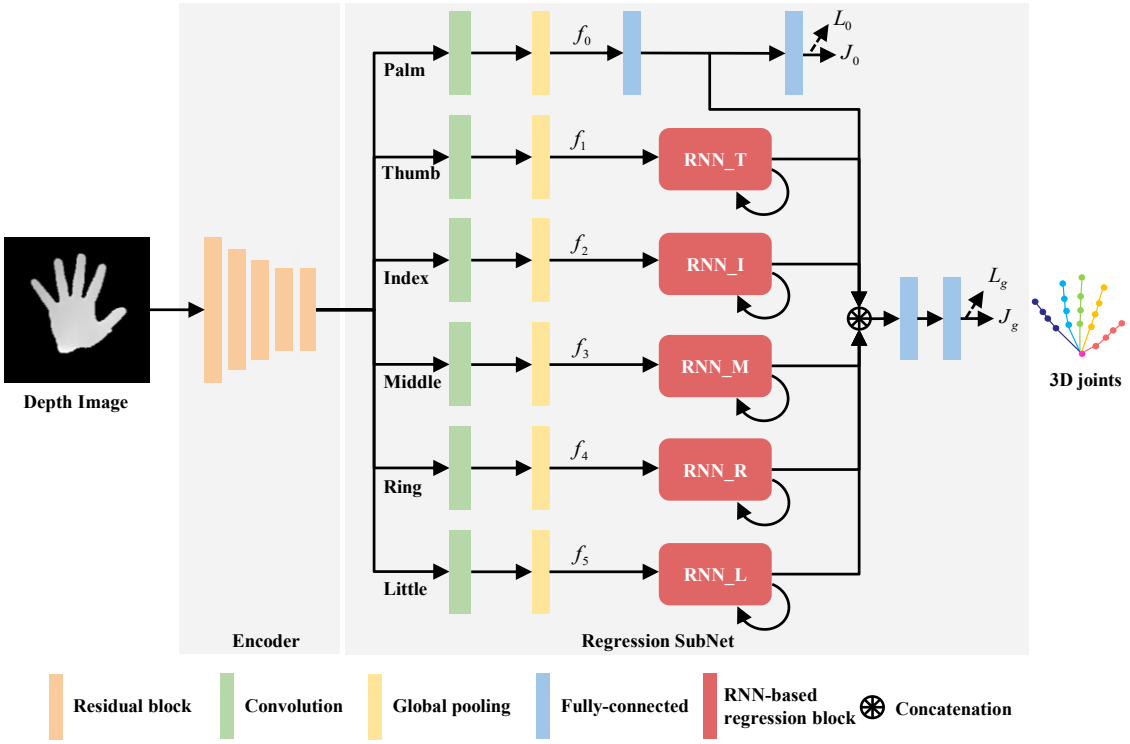


Figure 2.11 – The overall architecture of the HCRNN model, which is inspired by the spatial dependencies and sequential constraints between adjacent joints in one finger [170]. Reprinted image: ©2020 IEEE.

the augmented skeletons to depth images by imposing the consistency over existing datasets with pairwise skeleton and depth images and the self-consistency over unpaired augmented skeletons.

Hierarchical and Structured Methods

There are some network architectures that fully utilize the structural properties of hands, e.g., a hand has five fingers, the joints in one finger are correlated [62]. Some network models decomposed the whole hand pose estimation task into several hand-part-related sub-tasks. Therefore, these models tended to have a wider structure instead of a deeper structure. To understand the functional importance of different fingers, Zhou et al. [181] presented a hand branch ensemble (HBE) network, which handles the thumb, the index finger, and other fingers separately by three branches. The hierarchical CroosInfoNet model [40] made use of a multi-task information sharing mechanism and divided the hand pose estimation task into a palm pose estimation sub-task, and a finger pose estimation sub-task. Two sub-tasks were trained in two branches and a two-branch cross-connection structure was used to enhance the sub-task features effectively. Yoo et al. [170] proposed a hierarchically structured Recurrent Convolutional Neural Network (RCNN) (HCRNN) that estimates the 3D position of the palm and joints on five fingers by six parallel branches (see Fig. 2.11). The HCRNN model is inspired by the spatial dependencies and sequential constraints between adjacent joints in one finger. Therefore, a RCNN structure is chosen to handle the sequential features of the joints in a finger.

The above methods are still common 2D CNNs, which do not consider the input from a spatial viewpoint. In recent years, several methods have embedded kinematic correlations of human hands to ensure the spatial validity of the 3D structures based on graph convolution networks (GCNs) [77], which generalize CNNs to graph-structured data. Formulating the connections between hand joints as a 3D graph, graph convolution networks could learn the joint dependencies and further augment the local feature representations. In [18], a spatial-temporal graph on consecutive skeleton sequences of the human body or human hand was constructed and a hierarchical GCN-based method was designed to process and consolidate features across scales effectively. Later, Fang et al. [47] proposed a GCN-based joint graph reasoning module to incorporate geometric dependencies of hands explicitly. Three different graph structures (skeleton graph, feature similarity, and parameterized matrix) were used to construct the human hand, and all structures showed a similar estimation accuracy in tests.

2.4.4 Evaluation Metrics

This subsection describes how to assess whether a human pose estimation method is accurate or not. Almost all state-of-the-art human pose estimation algorithms are evaluated by two quantitative evaluation metrics:

1. the average Euclidean distance for all joints between the predicted 3D joint location and ground truth annotation,

2. the fraction of test samples whose predicted joint errors are below a given maximum distance from ground truth annotation.

The first metric reflects the average estimation error over all joints, and the second metric reflects the robustness of the model to outlier joints. The second metric is generally regarded as challenging, since a single outlier joint deteriorates the entire hand pose. Here, these two metrics assume that the outputs of the hand pose estimation algorithms are 3D positions of hand keypoints. If the outputs are the hand joint angles, the evaluation metrics should update to

3. the average joint angle error for all joints between the predicted joint angles and ground truth annotation,

4. the fraction of test samples whose predicted angle errors are below a given maximum angle threshold from ground truth annotation.

Further, in some survey works or the HANDS19 Challenge [W27], the models are evaluated by these two metrics (1 and 2) on several different well-designed test datasets. For instance, the samples in the test datasets are apportioned whether they have the same hand shape, viewpoints, or articulations presented in the training dataset. These new metrics provide comprehensive insights into the robustness of the models.

2.5 Discussion

This chapter presents the fundamental knowledge and principles required for designing a dexterous hand-arm teleoperation system through the survey of user-level interfaces, mapping methods, and control strategies in teleoperation. Teleoperation using wearable/contacting interfaces dominates in the research field, whereas markerless vision-based methods gradually show more potential for controlling dexterous robotic hands. With the premise of good real-time performance, human pose estimation is a good choice to obtain continuous control commands or human intentions from the sensory data. In terms of the multi-DoF robot arm, controlling its end-effector so that it moves to the correct position by point-to-point mapping is sufficient to fulfill most manipulation tasks. Regarding the multi-fingered robot hand, the joint-to-joint mapping would be a straightforward solution to power grasp, while point-to-point mapping would be suitable for dexterous manipulation and precision grasps. Therefore, we would expect the combination of markerless vision-based teleoperation, human pose estimation, and point-to-point mapping to yield a teleoperation framework that takes the human hand images as inputs and generates the joint angles commands for the robot. However, there is hardly any work investigating robot hand pose mapping directly from the images of human hands in an end-to-end manner. This thesis focuses on filling this gap and designing an end-to-end vision-based CNN that generates continuous robot poses and provides an efficient teleoperation experience.

The idea of end-to-end learning for robot teleoperation is highly relevant to human hand pose estimation, only the outputs are different. An algorithm with a good trade-off between accuracy and efficiency is appropriate in a teleoperation scenario. Data modality-wise, the depth map is the most popular modality in hand poses. Methodology-wise, the regression-based neural networks using 2D CNNs are usually more light-weighted than detection-based neural networks or 3D CNNs. While some point-wise detection methods are computationally efficient, the complex post-processing operations degrade the efficiency.

A remaining question of end-to-end learning for robot teleoperation is how to cross the domain gap between the human hand and the robot hand and how to compensate for the kinematic difference between them. Therefore, the robot hand pose estimation model should explore the shared pose features from paired human hands and robot hands. Deep generative methods using VAEs or GANs could be instructive to design the regression model. This thesis will leverage the essence of the deep generative methods and design novel human hand pose estimation methods to extract latent human-robot hand features based on depth inputs. Furthermore, this thesis will evaluate the proposed methods on multiple evaluation metrics and diverse robot experiments for more insights.

Chapter 3

Pairwise Human-Robot Hand Dataset Generation

In the previous chapter, the state-of-the-art teleoperation devices, mapping methods, and control strategies were analyzed by category. Compared to a robot arm, teleoperation of an anthropomorphic robotic hand to perform dexterous manipulation is more challenging. Fortunately, markerless vision-based teleoperation has been proved to offer advantages for dexterous teleoperation as it is versatile for different users and less invasive. Most existing markerless vision-based teleoperation methods contain two steps: human hand pose estimation followed by post-processing robot control. Instead, an end-to-end vision-based teleoperation method that takes human hand images as inputs and estimates robot joint commands aims directly at the robot system. In addition, the one-step end-to-end model is intuitive for novice demonstrators and saves post-processing time in practice. Therefore, one aim of this thesis is to design an efficient end-to-end vision-based robot hand pose estimation model, which can be exploited in dexterous teleoperation.

Training an end-to-end neural network for this task depends on massive human-robot pairings of images and ground truth. Consequently, this chapter explores an efficient mapping method that generates the corresponding robot poses from human hand poses, thus collecting synchronized hand pose data of the human and the robot. In general, the main challenges of hand-robot mapping stem from the structural discrepancies between the human hand and the robotic hand. Therefore, it is essential to firstly introduce and compare kinematic models of the human hand and the robot hand (see section 3.1).

Based upon the fundamental understanding of the human hand and the robot hand, section 3.2 describes the novel criterion of generating human-robot pairing by using labeled human hand depth images, obtaining the corresponding joint angles of the robot by an optimized kinematic retargeting method, then recording robot images in simulation. Accordingly, section 3.3 describes and visualizes the BigHand2.2M dataset, which is chosen from the existing human hand pose datasets. Then section 3.4 elaborates the retargeting methods and how to capture the depth images of the robot hand. In the end, a technical summary, illustration of angle distribution, and a few remaining issues about the self-built datasets are discussed in section 3.5.

3.1 Hand Kinematic Model

3.1.1 Human Hand Kinematic Model

There is numerous work studying human hand kinematics in terms of object grasping [12, 122]. However, the human hand models are not entirely consistent in anatomical research. A typical hand model defined in [28] is shown in Fig. 3.1(a). The Thumb (TH) comprises three joints Interphalangeal (IP), Metacarpophalangeal (MCP), and Trapeziometacarpal (TMC) and three phalanges (distal phalanges, proximal phalanges, and metacarpal). The other four fingers (First Finger (FF), Middle Finger (MF), Ring Finger (RF), Little Finger (LF)) are characterized by three joints (Distal Interphalangeal (DIP), Proximal Interphalangeal (PIP), MCP joints), and four phalanges (the phalanges of the thumb plus the middle ones). The joint movement is considered either flexion/extension motion or adduction/abduction motion. The TMC joint of the thumb and the MCP joints of the other four fingers are characterized by two DoFs (flexion/extension and adduction/abduction). A single DoF (flexion/extension) is designated to the MCP and IP joints of the thumb and the PIP and DIP joints of the other fingers (Fig. 3.1(b)). In addition, the Carpometacarpal (CMC) joint expresses the deformation of the palm. For example, when the hand is grasping an apple, the abduction angle for the MCP joint is defined before the flexion angle. Moreover, the wrist has two DoFs, pitch and yaw. In total, the hand model consists of 26 DoFs modeled by joints.

However, the CMC joints are usually ignored in the field of vision-based human hand pose estimation, and a simplified version of the human hand model (Fig. 3.1(c)) is commonly used. The thumb joints are modeled like those of the other fingers, which have PIP and DIP with a single DoF and MCP joint with two DoFs. The 22-DoF human model used in this thesis is based on this simplified version.

3.1.2 Shadow Dexterous Hand

The anthropomorphic robot hand used in this thesis is the motor-driven Shadow dexterous hand [W34], shown in Fig. 3.2. The Shadow hand is designed to match the mechanism of an adult hand, and the knuckles are staggered to provide comparable fingertip space to the human hand. Each finger is the same length and has four joints: the distal, middle, proximal, and metacarpal joints. The distal and middle joints of four fingers are designed to be coupled together to reduce the number of actuators. The little finger and the thumb are provided with an extra joint for modeling the CMC joint and holding objects. Moreover, the wrist has two DoFs, pitch and yaw. Summed up, the Shadow hand shown in Fig. 3.2(a) has 24 movements but 20 DoFs. In our setup, there are five Syntouch Biotac tactile sensors [W37] retrofitted at the fingertips of the Shadow hand. Therefore, the last phalanx of each finger was replaced, and the first joint of each finger is stiff. Consequently, the Shadow hand with BioTac sensors has 19 movements and 19 DoFs and its kinematic model is visualized in Fig. 3.2(c).

3.1.3 Comparison of Human Hand and Robot Hand

Even though the Shadow hand is the only robot hand on the market to have 24 movements and 20 DoFs and can grasp and manipulate a range of objects flexibly, there is still a big gap to achieving agility equal to the human hand. First, the skeletal shape and the whole silhouette

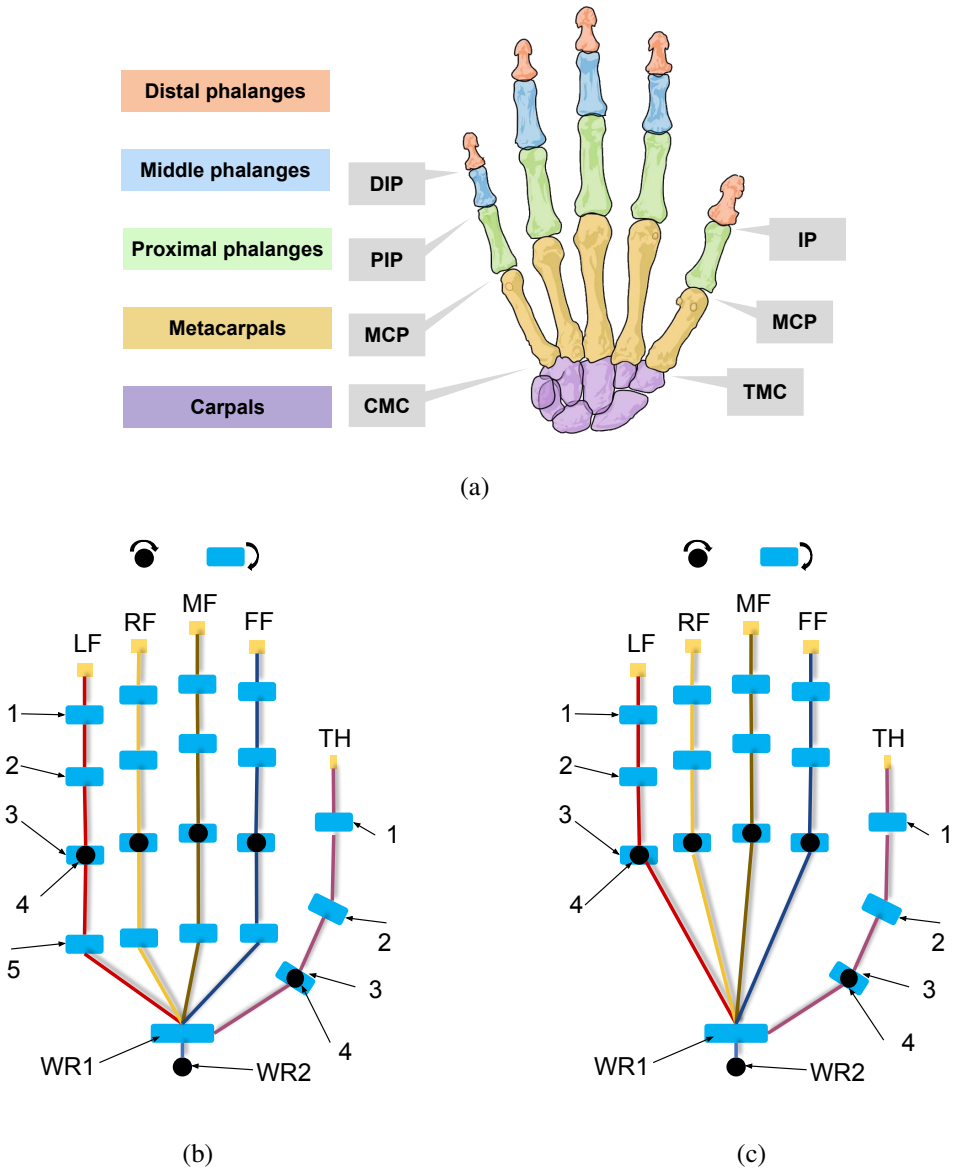
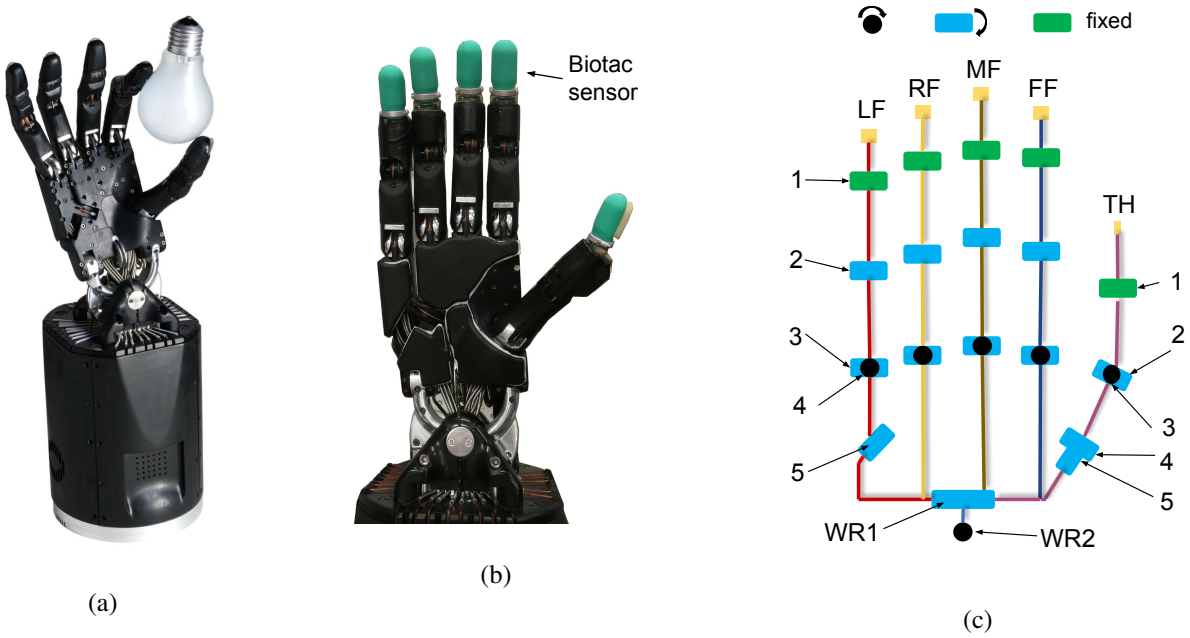


Figure 3.1 – (a) Human hand skeleton and (b) the kinematic chain of the 26-DoF human hand proposed by [28]. (c) The common 22-DoF human hand model in hand pose estimation.

of the robot hand and the human hand are similar, but the link length and the articulation are different. Second, the joint ranges also determine the anthropomorphism of the robot hand. The motion ranges of the human hand model proposed by [28] and the Shadow hand are shown in Table 3.1. The data are obtained from [W3] and [28]. As joints J1 through J4 of four Shadow fingers are exactly the same and those of human fingers are similar, only TH, FF, and LF fingers are investigated. Generally, the human hand joints have wider ranges than those of the robot. The distinct difference concerns the articulation of the thumb, which is the essential finger for dexterous manipulation. The first and second joints of the human thumb can still have small abduction/adduction motion. Apart from the kinematic model and the motion range, note that the Shadow hand used in this work has fixed J1 (cannot move between 0° and 90° any more) for all fingers due to the BioTac sensor. This installation increases the difficulty of accurately mapping the human hand to the robot hand.



(a) (b) (c)

Figure 3.2 – (a) The 20-DoF motor-driven Shadow dexterous hand with 24 movements. Reprinted image courtesy of Shadow Robot Company [W34]. (b) The 19-DoF motor-driven Shadow dexterous hand installed with BioTac sensors on all five fingers. (c) The kinematic chain of the Shadow hand with BioTac sensor. In this case, joint 1 of each finger is stiff.

Human hand			Shadow hand with BioTac		
Joint	Flexion/ Extension	Abduction/ adduction	Joint	Flexion/ Extension	Abduction/ adduction
THJ4		-30° – 30°	THJ5	-60° – 60°	
THJ3	15° – 90°		THJ4	0° – 70°	
THJ2	0° – 80°	-5°-5°	THJ3		-12° – 12°
			THJ2	-30° – 30°	
THJ1	-5° – 80°	-5°-5°	THJ1	20°	
FFJ5	0° – 5°				
FFJ4		-30° – 30°	FFJ4		-20° – 20°
FFJ3	-40° – 90°		FFJ3	-15° – 90°	
FFJ2	0° – 110°		FFJ2	0° – 90°	
FFJ1	-5° – 90°		FFJ1	20°	
LFJ5	0° – 15°		LFJ5	0° – 45°	
LFJ4		-25° – 25°	LFJ4		-20° – 20°
LFJ3	-40° – 90°		LFJ3	-15° – 90°	
LFJ2	0° – 135°		LFJ2	0° – 90°	
LFJ1	-5° – 90°		LFJ1	20°	

Table 3.1 – Joint ranges of human hand proposed by [28] and Shadow hand with BioTac sensor

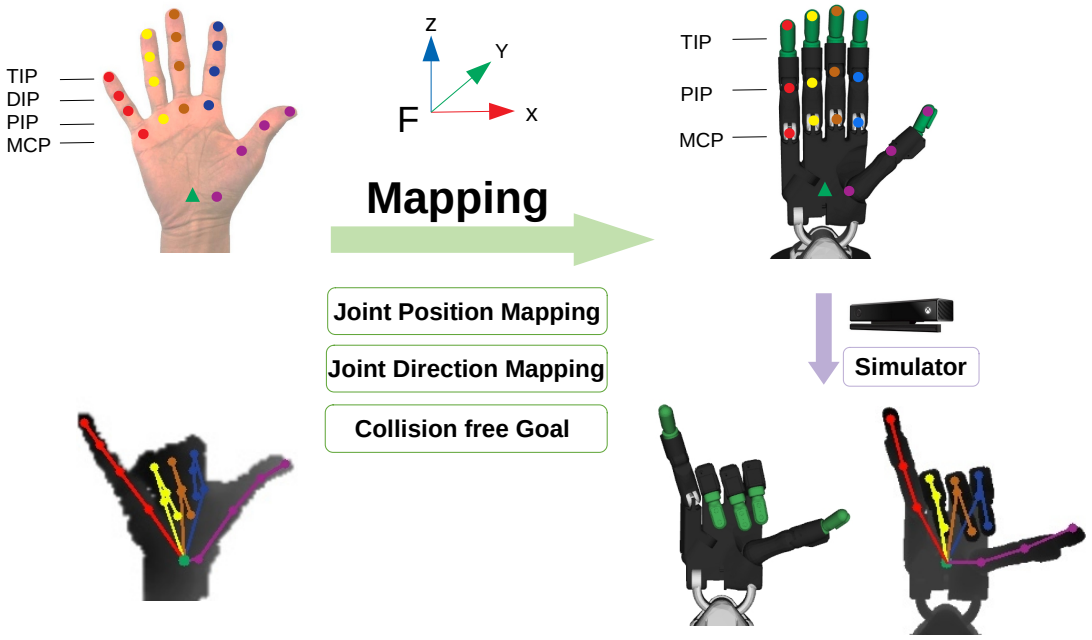


Figure 3.3 – Pipeline for dataset generation. (Top left) The human hand model and its keypoints. (Bottom left) A depth image of human hand. (Middle) Optimized mapping method from the human hand to the Shadow hand. (Top right) The Shadow hand with BioTac sensors has 24 joints and moves with 19 DoFs. (Bottom right) The corresponding RGB and depth images of Shadow gestures obtained from the simulator. The colored circles denote the joint keypoint positions on the hand, and the green triangles denote the origin of common reference frame F.

3.2 Dataset Generation Pipeline

End-to-end vision-based teleoperation requires an accurate mapping from the operators' hand to the robot. To learn the pose feature of the robot from images of the human hand, we have to consider how to get a number of human-robot pairings. The human-robot pairings should contain the pairwise human-robot images, where the robot hand acts the same as the corresponding human hand, and the robot joint angles. There are several off-the-shelf human hand pose datasets as summarized in Table 2.1, but there are no existing and open-sourced pairwise human-robot hand images. Prior work in [147, 139, 136] acquired the master-slave pairings by demanding a human operator to imitate the robot motion synchronously. The pairing data is costly to collect like this and typically comes with noisy correspondences. Also, there is no longer an exact correspondence between the human and the robot because physiological differences make the imitation non-trivial and subjective to the imitator. In fact, the robot state is more accessible and is relatively stable compared to the human hand, and there are many existing human hand datasets. However, operating on a real robot requires a significant time of data collection and hardware preparation, and has the risk of damaging the robot. An alternative approach is to record robotic data in a simulation then adapt the representation to a real robot [4].

With the above considerations in mind, this chapter proposes a novel approach of generating human-robot pairing by using an existing dataset of labeled human hand depth images, manipulating the robot and recording corresponding joint angles and images in simulation, and performing extensive evaluations on a physical robot. We achieve dataset generation by using

the off-the-shelf human hand dataset Big-Hand2.2M Dataset [172] and an optimized mapping method using the pipeline of Fig. 3.3. With this pipeline, two training datasets containing 400K pairs of human-robot depth images and corresponding robot joint angles were efficiently collected, respectively.

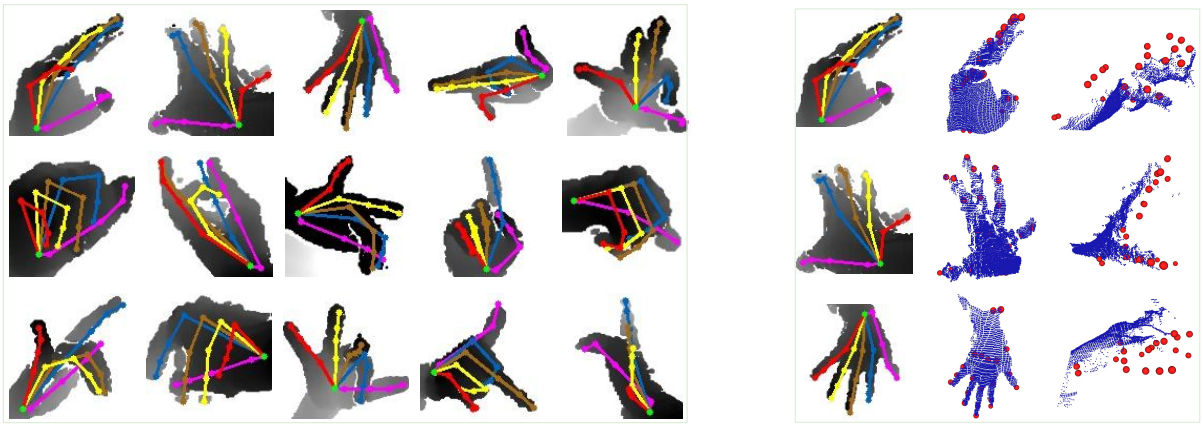
3.3 Selection of Human Hand Pose Dataset

The choice of the dataset selection touches on one of the aspects of neural network design, which is suitable input representation. As discussed in the subsection 2.4.2, depth images represent distance information of object points from the depth camera, but have lost color details of RGB images and do not contain full 3D information in point clouds or voxelized data. However, color is not a critical factor in hand pose estimation. Indeed, hand pose estimation methods trained on 3D inputs slightly outperform those depending on depth images, but 2D CNNs trained on depth inputs still show notable performance [113]. And depth images are more likely to result in light-weighted networks than 3D inputs, e.g., voxels and point clouds. Admittedly, the synthetic data tends to miss the random behaviors because it is purely generated based on the original dataset's properties, which are inferred from human assumptions during the data preparatory steps. To this end, we argued that the real-world depth map representation is more suitable for a vision-based teleoperation system.

Based on the discussion about different human hand pose datasets in subsection 2.4.1, the million-scale BigHand2.2M dataset is chosen as our human hand dataset because it contains 2.2 million depth maps and represents a significant advancement in the completeness of hand data variation and annotation quality. The depth maps in this dataset are accurately annotated with 3D joint locations with respect to the camera coordinate. The human hand model used in BigHand2.2M is shown in Fig. 3.1(c). This hand model has a total of 21 joints, which consist of four joints (TIP, MCP, DIP, and PIP joints) for each finger and a wrist joint. There are 21 keypoints for every hand, including one wrist position and four joint locations of each finger (see the top left image in Fig. 3.3). The keypoints' positions are inferred by the predefined physical constraints of the hand and six 6D magnetic sensors attached to the five fingertips and the back of the palm. A trakSTAK tracking system [W23] is used to track real-time poses of the 6D sensors, and an Intel RealSense SR300 depth sensor [W9] is used to record the depth images of the hands with a resolution of 640×480. The example images from the Bighand2.2M dataset are demonstrated in Fig. 3.4(a) and the joint annotations have high accuracy in 2D pixel coordinates. However, converting the depth images to point clouds the human hand, we observe that the 3D annotations are slightly drifting along with the camera viewpoint (see Fig. 3.4(b)). The possible reasons for this inaccuracy are that the joint positions, except for the fingertip, are determined by the defined sophisticated constraints and the distortion of the camera.

3.4 Optimized Mapping Method

After determining the human hand dataset, effectively mapping the human hand pose to the robot pose is the next step. To imitate the human hand pose better, an optimized mapping method is proposed to integrate position mapping, orientation mapping and properly take into account possible self-collisions. This method is extended from the bio-ik inverse kinematic



(a) 2D Depth images

(b) 3D annotations

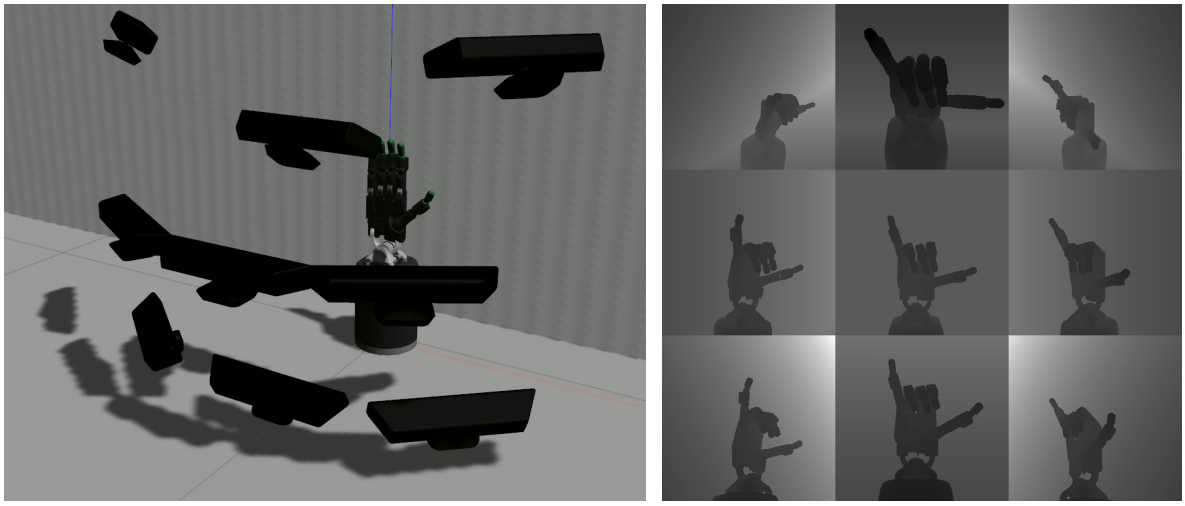
Figure 3.4 – Example data from the BigHand2.2M dataset in 2D depth images (a) and 3D point clouds (b). The joint annotations are very accurate in 2D pixel coordinates. However, the annotations in 3D space show some drift along with the camera viewpoint.

solver [131]. The bio-ik inverse kinematic solver is a memetic optimization algorithm that integrates the genetic algorithm, particle swarm optimization, and gradient methods. It solves inverse kinematics and motion tasks on robots with arbitrary kinematic trees and provides great flexibility and control of multiple objectives. Bio-ik takes a user-defined cost function that includes weighted combinations of several Cartesian- and joint-space goals, such as position, orientation, touch, and direction goals. The goals provided in bio-ik use a quadratic or almost-quadratic form rather than a linear form, resulting in better opportunities for gradient-based exploitation and faster distance computations. Given a weighted set of goals, the evolutionary solver then converges to an optimal solution. The flexible goal combination allows the user to specify secondary goals and to guide the search space exploration. For the dataset generation, the position goal, direction goal, and a self-designed collision goal are used.

The human hand poses are registered to the robot hand poses by a common reference F (see Fig. 3.3). The common reference frame F is located at the human wrist joint and 34 mm above the Z-axis of the robot wrist joint. Note that 34 mm is the height from the wrist joint to the base joint of the thumb. These locations are chosen because they lie at locations of high kinematic similarity. Since only finger motions are considered, two wrist joints of the Shadow are fixed and only 17 joint keypoints which are TIP, PIP, and MCP in each finger of the Shadow hand are calculated. Then position goals and direction goals are implemented by bio-ik. Position goals try to match the positions of five robot fingertips and five PIP links with the corresponding human hand positions. The cost function C_{pos} is defined as the square distance between robot link positions and goal positions. The scaling factors of C_{pos} are set to ω_{pf} for fingertips and a minor weight ω_{pp} for PIP joints.

$$C_{pos} = \|P_H - P_R\|^2, \tag{3.1}$$

The direction goals try to match a link axis with a goal direction. The axis is transformed by the link's current orientation and then the square distance between the axis and the goal direction is



(a) (b)

Figure 3.5 – (a) The data collection setup in Gazebo with nine simulated Kinect cameras. (b) The depth images of the Shadow hand from nine viewpoints corresponding to one human gesture in our dataset.

computed (see equation 3.2). The direction mapping with weight ω_d is applied to five proximal phalanges and the distal phalange of the thumb. In this dataset, $\omega_{pf}, \omega_{pp}, \omega_d$ are set as $1, 0.2, 0.2$.

$$C_{direc} = \|L_R - G\|^2), \tag{3.2}$$

where P_H, P_R respectively denote the positions of the human hand link, the robot hand link, and L_R, G are direction vectors of one axis of the robot hand link and the goal direction.

In case bio-ik calculates a self-collision output, a cost function F_{cost} in 3.3 which measures the distance between two links was defined.

$$C_{col} = \max(0, R\_col - \|P_{Ri} - P_{Rj}\|^2), \tag{3.3}$$

where P_{Ri}, P_{Rj} respectively denote the position of link i, link j of the robot hand, R_col is the minimum collision-free radius between two links.

Taking advantage of bio-ik to determine the robot joint angles $Q \in R^{17}$, the robot executes movements in Gazebo [W6] simulator and checks self-collision by MoveIt [30]. The virtual Shadow hand in the Gazebo is based on the Robot Operating System (ROS) package provided by the Shadow Robot Company. Considering that the BigHand2.2M dataset spans a wide range of observing viewpoints for the human hand, it is indispensable to increase the diversity of viewpoints of the robot data. Therefore, visual samples of the robot are collected through nine simulated depth cameras with different observing positions in Gazebo and record nine depth images for each pose simultaneously, as shown in Fig. 3.5(a). As an example, Fig. 3.5(b) presents the depth images of the robot from nine viewpoints. As is evident, the background darkness, especially some corners of the depth images, is different because in some viewpoints, the camera can possibly look into infinity and the depth values go to infinity. These infinity pixels will be filtered out before they are fed into the network training. The example human-robot

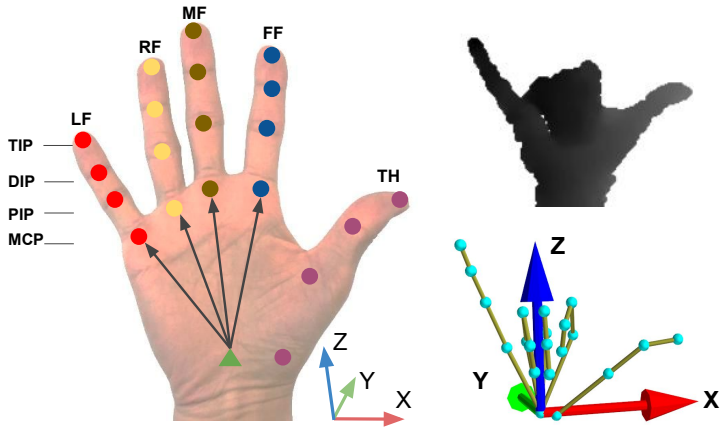
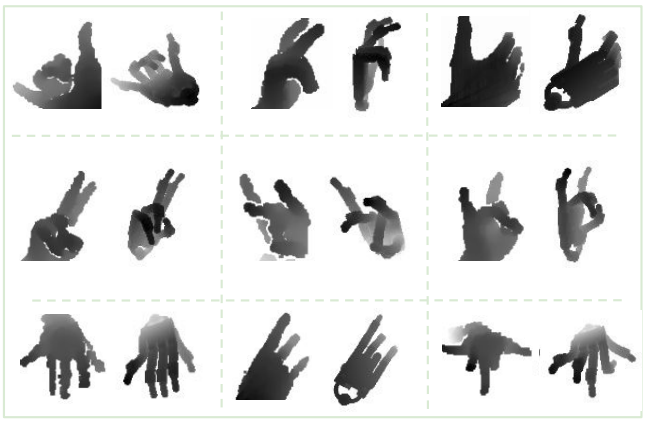


Figure 3.6 – The left image shows the keypoints of a human hand. The four lines with an arrow represent the vectors from the wrist pointing to the metacarpal joints of the first finger, middle finger, ring finger, and little finger. The right images illustrate the local hand frame of an example hand.

(a) Dataset1



(b) Dataset2

Figure 3.7 – Examples of paired human-robot depth images at a (a) different (Dataset1) and (b) the same wrist pose (Dataset2) in our datasets. The left and right images in each pair are the human hand and the robot hand, respectively.

pairings are shown in Fig. 3.7(a). In total, 400K pairs of simulated robot depth images and human hand depth images, with corresponding robot joint angles and poses, are efficiently collected. In this thesis, we refer to this dataset as "Dataset1".

Considering that the inconsistent orientation and position of the human-robot wrists admittedly yield more training challenges for the training process, it would be better to take the images of the human hand and the robot hand from the same viewpoint and at the same wrist poses. Therefore, a pairwise human-robot dataset from the same viewpoint is collected instead of keeping the cameras and the robot at a fixed global pose.

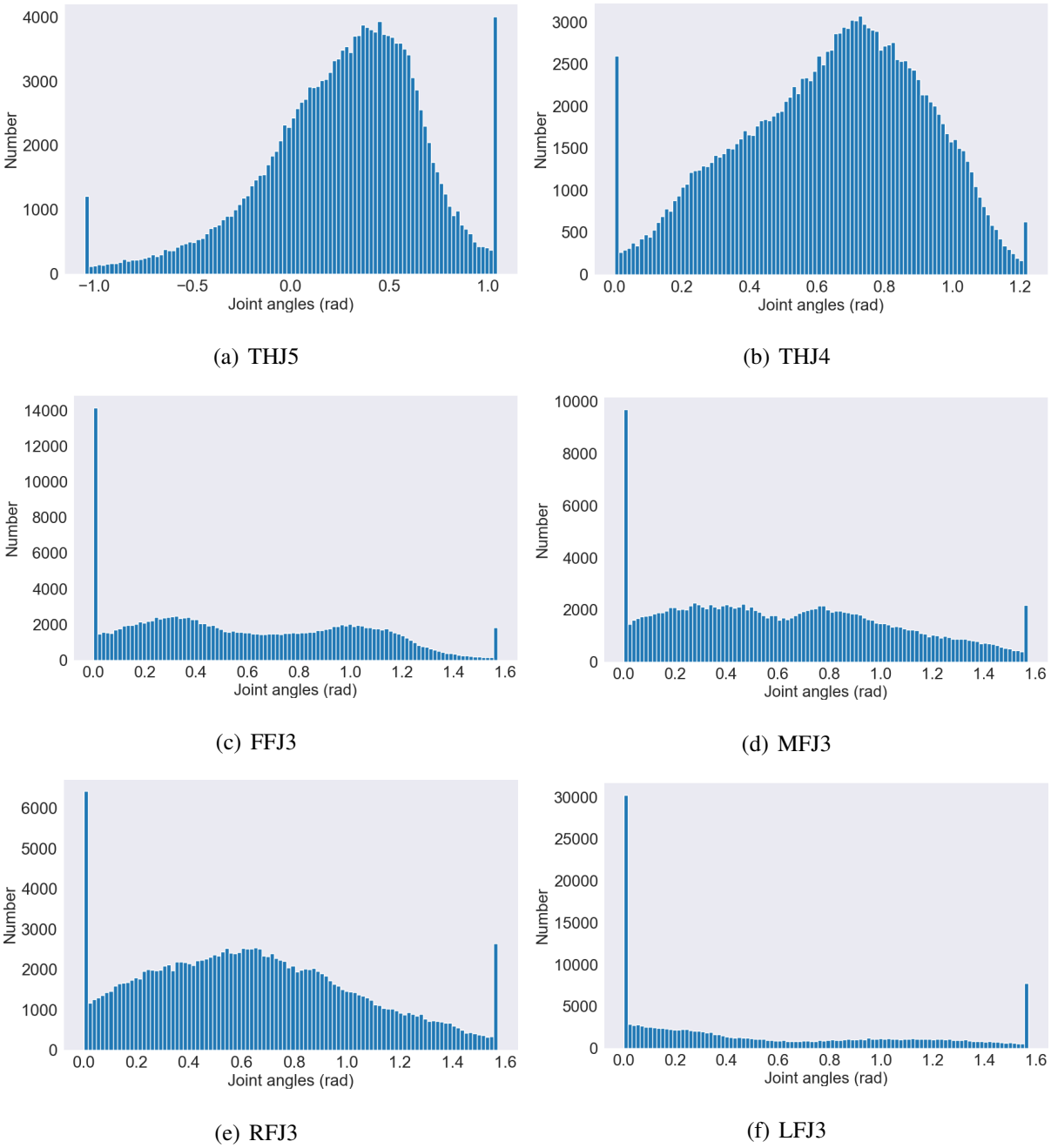
In order to obtain the wrist orientation, we build a local hand frame for each hand. The Z-axis is the mean of the vectors FF-palm, MF-palm, RF-palm, and LF-palm. The Y-axis is the cross product of MF-palm and RF-palm. Then we get the X-axis for a right-handed coordinate system. The vectors FF-palm, MF-palm, RF-palm, and MF-palm represent the vectors from the wrist pointing to the metacarpal joints of FF, MF, RF, and LF. The visualization of the human hand frame is shown in Fig. 3.6. Once we have the wrist orientation, we get the transformations from the camera to the wrist regarding the human hand dataset. Instead of using the planning method in MoveIt and moving the robot in Gazebo, a depth image generator through an OpenGL interface is designed. The platform change is caused by the instability and long execution time in Gazebo. Next, we set up a camera in OpenGL at the same orientation and position with respect to the robot wrist. Finally, given the transformation between the camera and the robot wrist, we render the robot model based on the calculated joint angles by bio-ik and capture the depth images of the robot hand in OpenGL. The pairwise dataset consists of 400K synchronized human-robot depth images and corresponding robot joint angles. As demonstrated in Fig. 3.7(b), the robot hands are imitating the human hand and are at similar wrist poses. This dataset is called "Dataset2" in this thesis.

3.5 Discussion

Beginning from the comparison of the human-robot hand kinematic models, this chapter highlights an efficient dataset generation pipeline. In this pipeline, the joint configurations of the robot hand are firstly generated from the keypoints of the human hands by an optimized mapping method, then the robot hand moves in simulation, finally the robot images and joint angles are recorded. Two datasets are collected and contain 400K pairs of the human hand depth images, simulated robot depth images in the same poses, and a corresponding robot joint angle, respectively. The difference between these two datasets is whether the robot wrist in the depth images has a similar pose to the human wrist.

Fig. 3.8 illustrates the joint angle distribution of the robot hand in Dataset2. Only six joints which are crucial to robot movements are analyzed, i.e., THJ5, THJ4, FFJ3, MFJ3, RFJ3, and LFJ3. We observe that THJ5 and THJ4 have more positions in the middle of their joint limits, while the other four joints have relatively even angle distribution. In addition, there are high peaks at the maximum joint limits or the minimum joint limits. The robot images when these six joints are at the boundary positions are shown in Fig. 3.9. When the joint values of THJ5 increase and the joint values of THJ4 decrease, the thumb tends to move closer to the other fingers or the hand center. We assume that the robot fingers tend

to fold into the center because the human hands are usually smaller than the robot hand. There are many human finger positions out of the range of the robot workspace. Therefore the IK solver generates many redundant solutions to compensate for the kinematic difference.



(a) THJ5

(b) THJ4

(c) FFJ3

(d) MFJ3

(e) RFJ3

(f) LFJ3

Figure 3.8 – The angle distribution of THJ5, THJ4, FFJ3, MFJ3, RFJ3, and LFJ3 in Dataset.

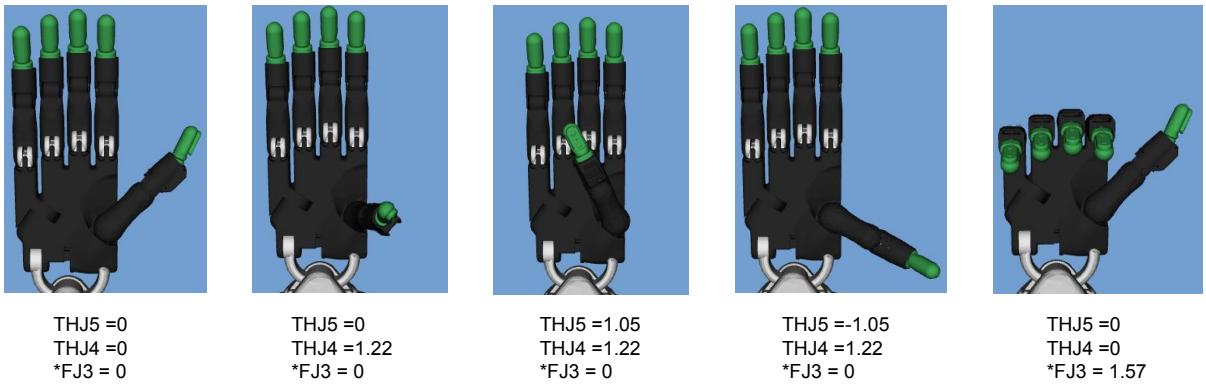


Figure 3.9 – Visualization of robot states when THJ5, THJ4, FFJ3, MFJ3, RFJ3, and LFJ3 are at the boundary positions. All joints which are not mentioned are at 0 rad, except for all J1 joints.

In the end, the question remains whether a network training on the datasets whose robot information is collected in simulation could functionally work on a real robot. On the one hand, even though the synthetic depth images are easier to adapt than the synthetic RGB images, there is no noise added into the synthetic depth images to enhance generalization. On the other hand, the robot model in simulation is perfectly calibrated, while the real robot is bad-calibrated because it has been used for years. The robot experiments in the following chapters (chapters 4, 6, 7, and 8) will reflect the practicality of the proposed datasets.

Chapter 4

TeachNet: Vision-based Teleoperation System for Anthropomorphic Robot Hand

Once a suitable human-robot hand dataset is available, it is essential to design an efficient network that could learn the corresponding robot pose feature in human pose space. This chapter presents Teacher-Student Network (TeachNet), a novel neural network architecture for intuitive and markerless vision-based teleoperation of dexterous robotic hands.

This chapter starts with a thorough discussion in section 4.1 which motivates the design needs for an end-to-end network model used in the vision-based teleoperation system. Then the proposed vision-based teleoperation framework is introduced by visualization.

Next, section 4.2 describes the motivation, detailed structure and loss functions of the end-to-end network, TeachNet. This particularly covers the discussion of two kinds of consistency losses and two different aligning layers, which all can be used for designing the aligning mechanism in TeachNet.

Later, section 4.3 explains the data preprocessing and training details, then provides comprehensive network evaluations through multiple baselines by three metrics. The network evaluation results verify the superiority of TeachNet, especially regarding the high-precision condition.

Finally, section 4.4 shows the robot experiments by the proposed teleoperation system in simulation and the real world. This section answers the question raised in chapter 3 of whether the network models learning from the robot data in simulation could work on a real robot system. Imitation experiments and grasp tasks conducted by novice users demonstrate that TeachNet trained on the self-collected dataset is more reliable and faster than the state-of-the-art vision-based teleoperation method [113].

4.1 End-to-end Learning for Vision-based Teleoperation

The markerless vision-based teleoperation system is designed to achieve a user-friendly and efficient control of the anthropomorphic robot hand. As discussed in section 2.1, unlike contacting or wearable device-based teleoperation, markerless vision-based teleoperation offers the advantages of capturing natural human-limb motions and of being less invasive. An increasing number of researchers have been focusing on data-driven vision-based teleoperation. This teleoperation method gets the 3D hand pose or recognizes the class of hand gestures using CNNs, and then maps the locations or the corresponding poses to the robot. The expected vision-based algorithm seeks to go beyond these methods by the end-to-end deep learning structure, taking a noisy image of the human hand as input and producing joint angles of the robot hand as output. The end-to-end vision-based teleoperation can be a natural and intuitive way to manipulate the remote robot and is user-friendly to the novice teleoperators. Therefore, it is essential to design an efficient network that could learn the corresponding robot pose feature in the human pose space.

Designing a supervised learning system in an end-to-end fashion for robot pose learning is essential to solve the domain gap between the human hand and the robot hand. The mechanical difference, mapping varied human hands to one specific robot hand, and the potential disturbance or noise in the human hand images are inevitable. These factors may cause regression challenges because the input features may not have encountered the target space. The key to bridging the gap is efficiently exploiting the common pose features between the human hand and the robot hand. Stanton et al. [147] allocated one feed-forward neural network for each actuator on the 25 DoF NAO humanoid robot [W32]. Each network only has one hidden layer with 20 neurons. They did not consider the human-robot kinematic relations for training at all. However, to get better teleoperation performance, the weights of each network then have to be optimized as closely as possible using particle swarm optimization. Later, with the improvement in representation learning, there are works focused on discovering the features from raw image data [11]. Sermanet et al. [136] achieved direct imitation of human pose based on the dedicated Time-contrastive Network (TCN) structure. The key idea of TCN is to extract the visual representations using a metric learning loss, where multiple simultaneous viewpoints of the same observation are attracted in the embedding space, while being repelled from temporal neighbors, which are often visually similar but functionally different. But the TCN structure was used to abstract useful representations on the human domain or robot domain individually. Finally, they only implemented body pose tests on a single-arm humanoid robot with a 7-DoF robot arm and a 1-DoF lift-and-down torso. We hypothesize that a neural network model that also considers abstracting kinematic relations between the human and robot domains would be effective.

4.2 Teacher-Student Network: TeachNet

After considering these design requirements, the end-to-end architecture for teleoperating the Shadow dexterous hand based on a single depth image is proposed, see Fig. 4.1. In this system, a camera is used to collect a depth image of the hand, and an end-to-end neural network modal estimates the desired robot joint angles based on the depth image, then joint commands are executed on the robot to manipulate the objects. However, solving joint regression problems

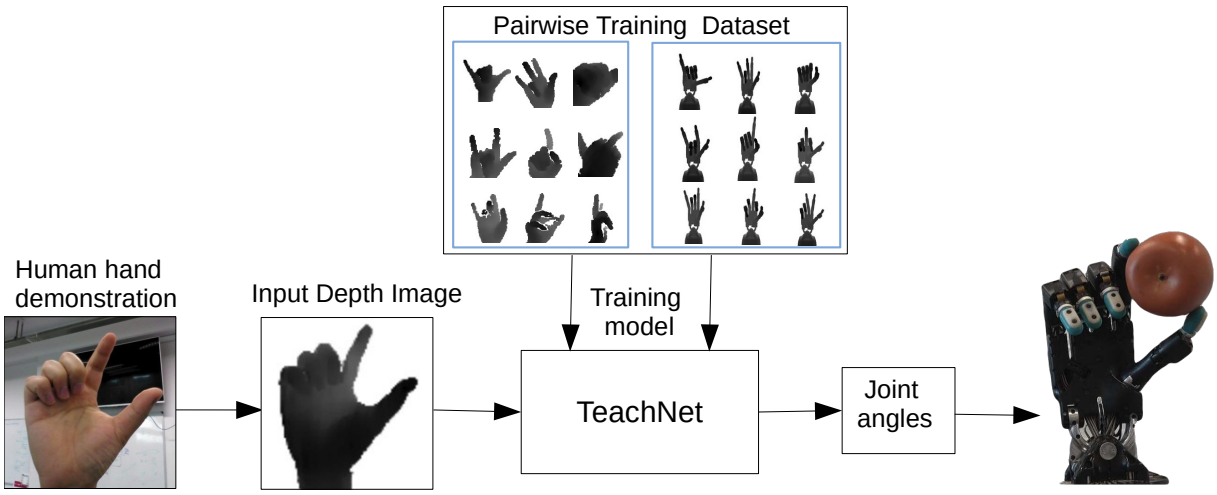


Figure 4.1 – The proposed vision-based teleoperation architecture. (Center) TeachNet is trained offline to predict robot joint angles from depth images of a human hand using our 400k pairwise human-robot hand dataset. (Left) Depth image of the operator's hand are captured by a depth camera then feed to TeachNet. (Right) The joint angles produced by TeachNet are executed on the robot to imitate the operator's hand pose.

directly from human images is quite challenging because the robot hand and the human hand occupy two different domains. Specifically, imagine that there are an image I_R of a robotic hand and an image I_H of a human hand, while the robotic hand in the image acts exactly the same as the human hand. The problem of mapping the human hand image to the corresponding robotic joint could be formulated as below:

$$\begin{aligned} f_{feat} &: I_H \in \mathbb{R}^2 \rightarrow Z_{pose} \\ f_{regress} &: Z_{pose} \rightarrow Q. \end{aligned} \tag{4.1}$$

4.2.1 Network Structure of TeachNet

To better process the geometric information in the input depth image and the complex constraints on joint regression, an encode-decode-style deep neural network was adopted. However, the human hand and Shadow hand are from different domains, thus it could be difficult for f_{feat} to learn an appropriate latent feature Z_{pose} in pose space. In contrast, the mapping from I_R to joint target Q will be more natural as it is exactly a well-defined hand pose estimation problem. Intuitively, for a paired human and robotic image, their latent pose features Z_{pose} should be encouraged to be consistent as they represent the same hand pose and will be finally mapped to the same joint target. Also, based on the observation that the mapping from I_R to Q performs better than I_H (these preliminary results can be found in Fig. 4.5(a)), the encoder f_{feat} of I_R could extract better pose features, which could significantly improve the regression results of the decoder.

With considerations above, the novel teacher-student network (TeachNet) is proposed to tackle the vision-based teleoperation problem (4.1) in an end-to-end fashion. The network architecture is illustrated in Fig. 4.2. TeachNet consists of two branches, the robot branch, which plays the role of a teacher, and the human branch as a student. TeachNet respectively accepts a

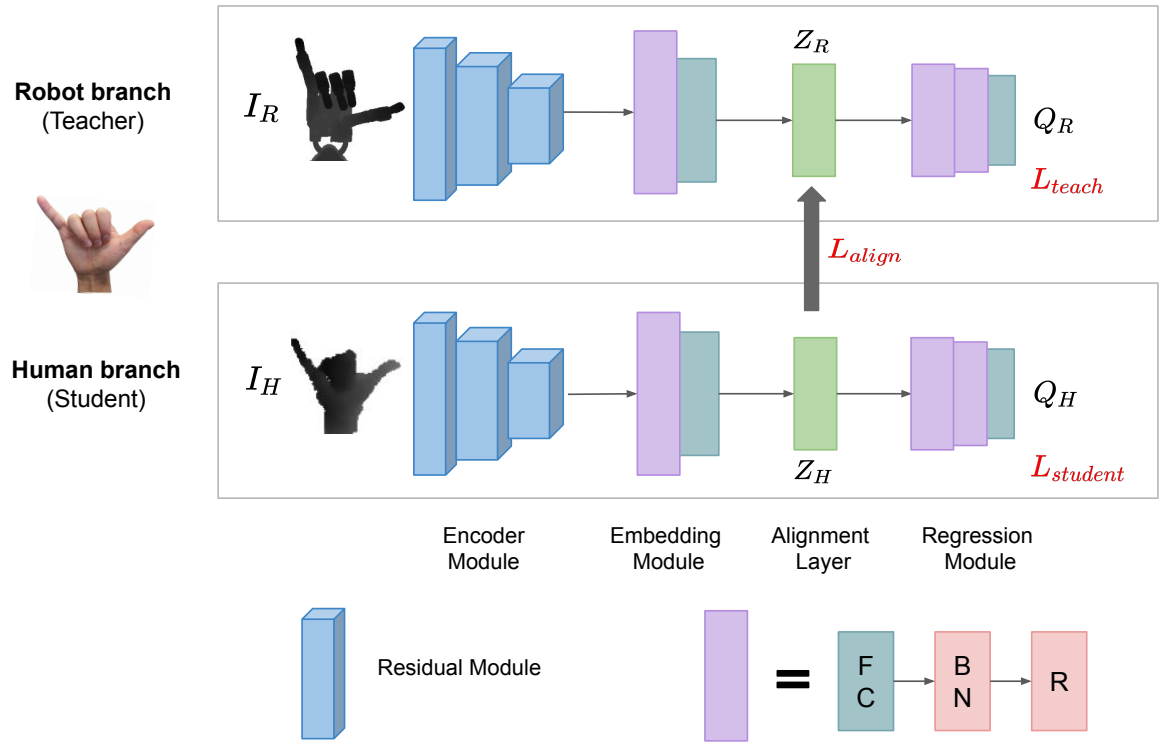


Figure 4.2 – Architecture of TeachNet neural network. Top: human branch, Bottom: robot branch. The input depth images I_H and I_R are fed to the corresponding branch that predicts the robot joint angels Q_H, Q_R. The residual module is a convolutional neural network with a similar architecture as ResNet [64]. FC denotes a fully-connected layer, BN denotes a batch normalization layer, R denotes a Rectified Linear Unit.

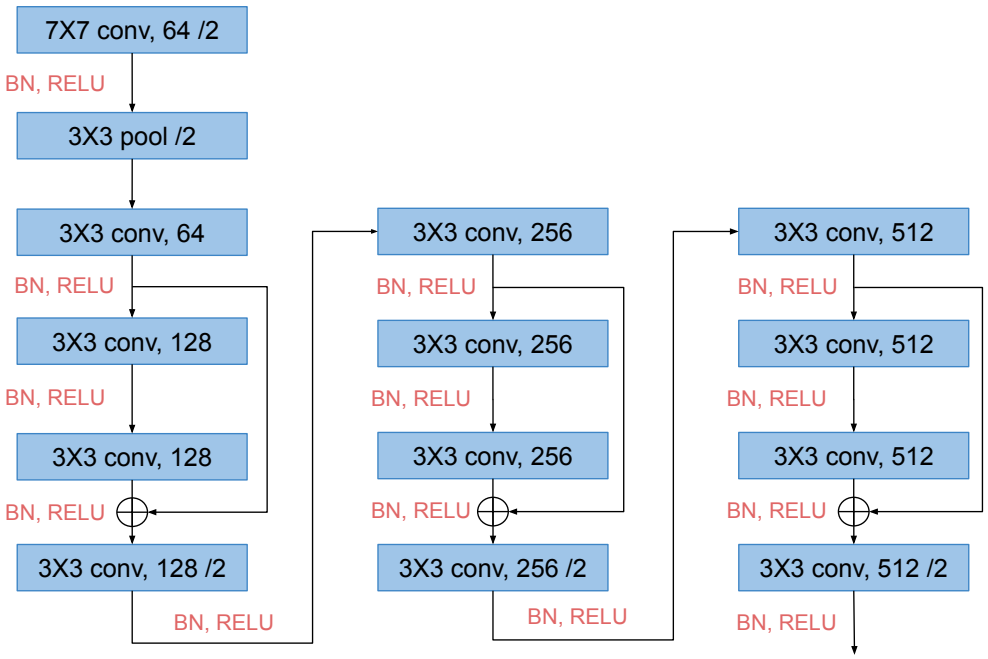


Figure 4.3 – Architecture of the encoder module in TeachNet model.

normalized depth image as the input for the corresponding branch in the training process. The architecture of each branch contains four components: the encoder module, the embedding module, the alignment layer, and the regression module. The encoder module contains an initial convolution layer with 64 filters, Batch Normalization (BN), and 3×3 max-pooling, following three residual modules, each with a stride of 3×3, and with $128, 256, 512$ filter (see Fig. 4.3). Then the embedding module flats the high-level features from the encoder module. As for the position of the alignment layer, *early teaching* and *late teaching* are put forward, respectively. For the former, the alignment layer is put after the embedding module and before the regression module, as shown in Fig. 4.2. In the *early teaching* model, the embedding module generates a 128 embedding feature by one Fully-connected Layer (FC) layer with batch normalization followed by a Rectified Linear Unit (ReLU) activation and an another FC layer. And the regression module contains two sets of FC-BN-BN combination and a final FC layer for the joint angle outputs Q. In the *late teaching* model, the embedding module deepens to four sets of FC-BN-BN combination, the alignment layer is positioned on the last but one layer of the whole model, and the regression module will only contain one FC layer.

4.2.2 Loss Functions for Robot Joint Regression

We formulate the robot joint learning problem as a supervised learning problem, in which the network takes a depth image of a random human hand as the input and outputs the corresponding joint angles of the robot. For each branch, the basic objectives are joint angle loss and an auxiliary loss.

Joint angle loss. Each branch is supervised with a Mean Squared Error (MSE) loss $\mathcal{L}_{ang}$:

$$\mathcal{L}_{ang} = \|Q - J\|^2, \tag{4.2}$$

where J is the ground truth joint angles.

Besides the encoder-decoder structure that maps the input depth image to joint prediction, two latent features Z_H and Z_R constitute the alignment layer. A consistency loss $\mathcal{L}_{align}$ between Z_H and Z_R is designed to exploit the geometrical resemblance between human hands and the robotic hand. Therefore, $\mathcal{L}_{align}$ forces the human branch to be supervised by a pose space shared with the robot branch. Even though the structure of each branch is similar to some regression-based 3D human hand estimation models, with the consistency loss, TeachNet presents the ability to learn the kinematic mappings between the slave and the master. To explore the most effective aligning mechanism, two kinds of consistency losses and two different aligning positions are formulated:

Hard consistency loss. The most intuitive mechanism for feature alignment would be providing an extra MSE loss over the latent features of these two branches:

$$\mathcal{L}_{align\_h} = \|Z_H - Z_R\|^2. \tag{4.3}$$

In this case, the complete training objective for each branch is:

$$\mathcal{L}_{teach} = \mathcal{L}_{ang}(Q_R) \tag{4.4}$$

$$\mathcal{L}_{student} = \mathcal{L}_{ang}(Q_H) + \alpha \cdot \mathcal{L}_{align\_h}. \tag{4.5}$$

where $\alpha = 1$ for hard consistency loss and $\alpha = 0.1$ for the soft consistency loss.

Soft consistency loss. Sometimes, (4.3) could distract the network from learning hand pose representations, especially in the early training stage. Inspired by [155], Z_H and Z_R are fed into a discriminator network D [61] to compute a *realism score* for *real* and *fake* pose features. The model structure of the D is two sets of one FC layer with batch normalization followed by a ReLU activation and the other FC layer followed by a sigmoid function (see Fig. 4.4). The feature size of the three FC layers is half the size of the input Z_H. The discriminator model D is trained by the Binary Cross Entropy (BCE) loss between the target and the input probabilities. We assume Z_R is the real pose features and Z_H is the fake pose features, therefore we have $\mathcal{L}_D$ in equation 4.9. Regarding the human branch, we expect the pose features Z_H is realistic enough to fool the discriminator model D, so the soft consistency loss is basically the negative of this score:

$$\mathcal{L}_{align\_s} = \log\left(1 - D(Z_H)\right). \tag{4.6}$$

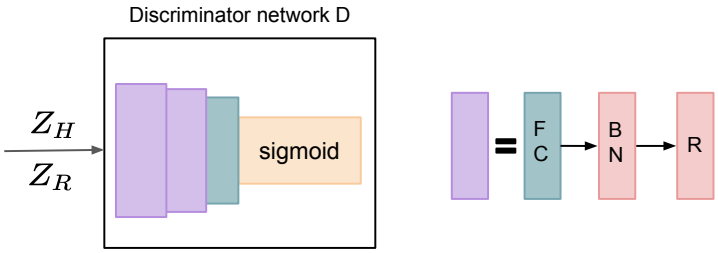


Figure 4.4 – Architecture of the discriminator network D for soft consistency loss.

Then, the complete training objectives are:

$$\mathcal{L}_{teach} = \mathcal{L}_{ang}(Q_R) \tag{4.7}$$

$$\mathcal{L}_{student} = \mathcal{L}_{ang}(Q_H) + \alpha \cdot \mathcal{L}_{align\_s} \tag{4.8}$$

$$\mathcal{L}_D = \log\left(D(Z_R)\right) + \log\left(1 - D(Z_H)\right). \tag{4.9}$$

where $\alpha = 1$ for hard consistency loss and $\alpha = 0.1$ for the soft consistency loss.

In the following, this chapter will refer to early teaching by $\mathcal{L}_{align\_s}$ as Teach Soft-Early, late teaching by $\mathcal{L}_{align\_s}$ as Teach Soft-Late, early teaching by $\mathcal{L}_{align\_h}$ as Teach Hard-Early, and late teaching by $\mathcal{L}_{align\_h}$ as Teach Hard-Late.

4.3 TeachNet Evaluation

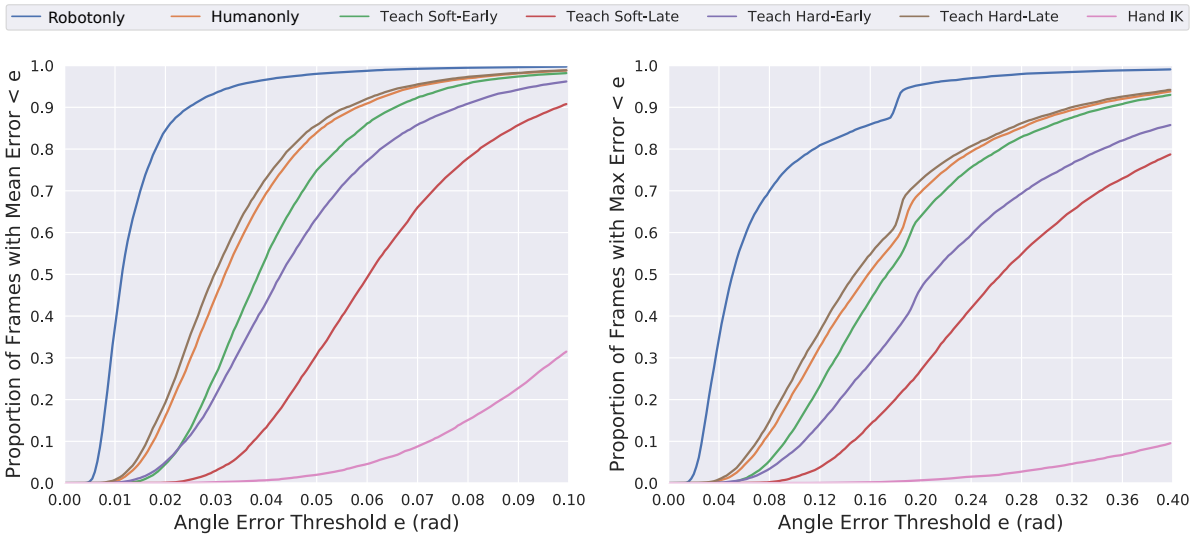
4.3.1 Training Details

The input depth images are first preprocessed using erosion followed by dilation to remove noise on the raw depth images. Then a 250×250 fixed-size cube around the hand is extracted from one depth image and projected to 2D space. Afterward, the images are resized to 100×100 and normalized to $[-1, 1]$. Random in-plane $[-180°, 180°]$ rotation and random jittering for all pixels are executed for data augmentation. Compared to keypoints regression, the output joint angles are invariant to the preprocessing of the input image. The outputs of the network are mapped to the joint angles by the upper and lower angle ranges, then are used to calculate joint angle loss. Therefore, the outputs from the network are limited in the range of $[0, 1]$ and the estimated joint angles are always kept in the joint ranges. The auxiliary physical loss [180], which enforces the physical constraints and joint limits, is not needed. At training time, the pairwise human-robot images are fed to the corresponding branch. The minibatch stochastic gradient descent and Adam optimizer with a learning rate $l_r = 0.002$ and momentum parameters $\beta_1 = 0.5, \beta_2 = 0.999$ was used. The learning rate was decayed by 0.5 every 80 epochs. Note that although there are nine images of Shadow hand (captured from nine viewpoints) corresponding to one human pose, one view of Shadow images is randomly chosen to feed into the robot branch during the training process of the TeachNet. At inference time, only the student branch is needed, namely, TeachNet takes an image of a human hand as input and then outputs the estimated joint angles Q of the robot hand.

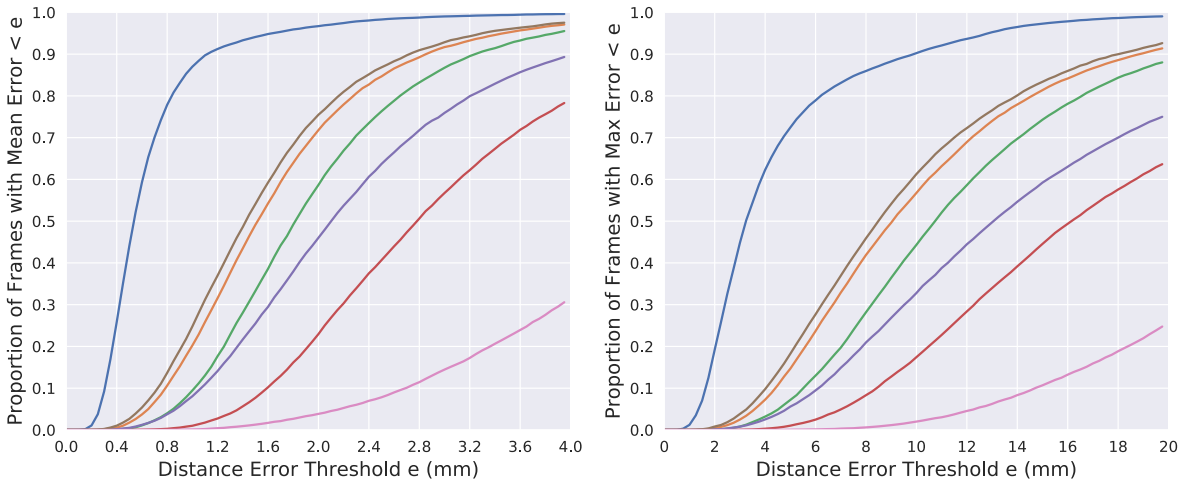
4.3.2 Baselines Comparison

This section examines whether the TeachNet could learn indicative visual representations that represent the human hand's kinematic structure. The proposed TeachNet was evaluated on the paired dataset with the following experiments: 1) To explore the appropriate position of the alignment layer and the proper align method, the proposed four network structures: Teach Soft-Early, Teach Soft-Late, Teach Hard-Early, and Teach Hard-Late were compared. 2) To validate the significance of the alignment layer, an ablation analysis by removing consistency loss $\mathcal{L}_{align}$ and separately training the single human branch and the single robot branch was designed. These two baselines respectively refer as "Humanonly" and "Robotonly". 3) The proposed end-to-end method was compared with the data-driven vision-based teleoperation method that estimates 3D keypoints of the human hand then maps the 3D keypoints to joint positions of the robot. This baseline refers to as "HandIK" solution. For the HandIK solution, the DeepPrior++ [113] network was trained on the same dataset. DeepPrior++ was chosen because its architecture is similar to the single branch of TeachNet, and it was the state-of-the-art hand pose estimation algorithm at that time. The outputs of DeepPrior++ are 21×3 human joint locations. Then these locations are used to acquire the joint angles of the Shadow hand by the mapping method explained in section 3.4.

There are three evaluation metrics used in this work: 1) the fraction of frames whose maximum/average joint angle errors are below a threshold; 2) the fraction of frames whose maximum/average joint distance errors are below a threshold; 3) the average angle error over all angles.



(a)



(b)

Figure 4.5 – The fraction of input frames whose maximum/average joint (a) angle and (b) distance error are below a threshold between Teach Hard-Late approach and different baselines on our test dataset. These show that Teach Hard-Late approach has the best accuracy over all evaluation metrics.

The comparative results, shown in Fig. 4.5 and Fig. 4.6, indicate that the Robotonly method is the best concerning all evaluation metrics and has the capability of the training "supervisor". Meanwhile, the Teach Hard-Late method outperforms the other baselines, which verifies that the single human branch is enhanced through an additional consistency loss. Especially regarding the high-precision condition, only the Teach Hard-Late approach shows an average 3.63% improvement of the accuracy below a maximum joint angle, which is higher than that of the Humanonly method (Table 4.1). It indicates that the later feature space f_{feat} of the depth images contains more useful information than the early feature space and the MSE method displays the stronger supervision in our case.

The regression-based HandIK method shows the worst performance among the three metrics. The unsatisfying outcome of the HandIK solution not only proves that TeachNet gives a better

Max Err.	Humanonly	Teach Soft-Early	Teach Soft-Late
0.1 rad	21.24%	12.31%	12.77%
0.15 rad	45.57%	38.06%	10.37%
0.2 rad	69.08%	63.18%	26.16%

Max Err.	Teach Hard-Early	Teach Hard-Late	Hand IK
0.1 rad	7.40%	**24.63%**	0.00%
0.15 rad	24.67%	**50.11%**	0.14%
0.2 rad	45.63%	**72.04%**	0.62%

Table 4.1 – Accuracy under high-precision conditions

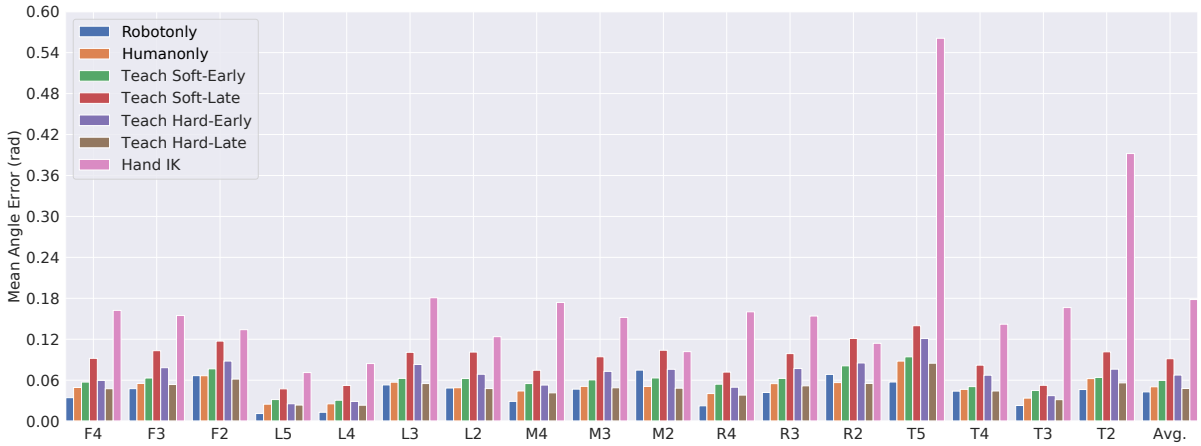
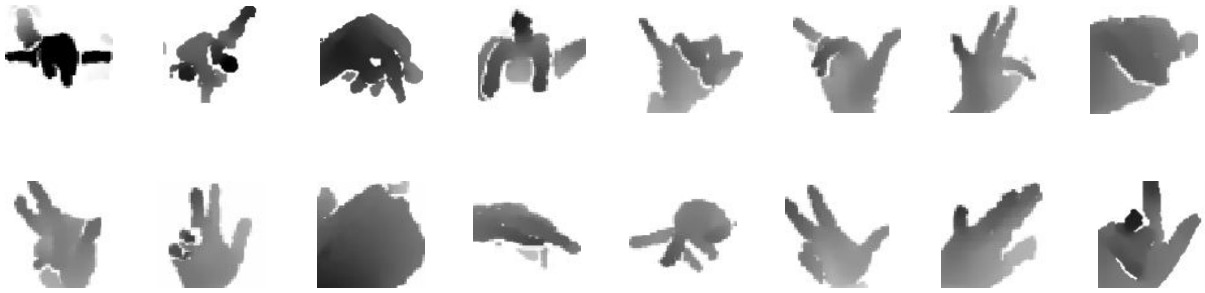


Figure 4.6 – Comparison of average angle error on the individual joint between the Teach Hard-Late approach and different baselines on our test dataset.

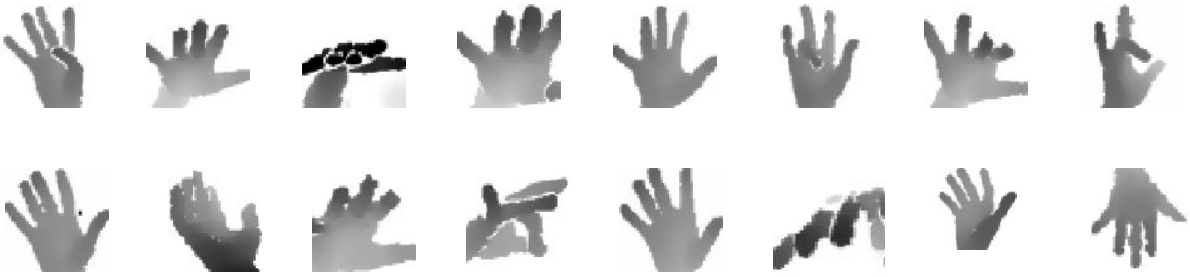
representation of the hand feature, but also suggests that the HandIK method does not consider the kinematic structure and the special limitation of the robot. Furthermore, direct joint angle regression should have decent accuracy on angles since that is the learning objective. The missing L_{phy} also gives rise to poor accuracy. Moreover, Fig. 4.6 demonstrates that the angles of THJ2, THJ4 and THJ5 are harder to be learned. These results are mainly because 1) the fixed distal joints of the robot in our work affect the accuracy of its second joint and third joint. 2) these types of joints have a bigger joint range than other joints, especially the base joint of the thumb. 3) there is a big discrepancy between the human thumb and the Shadow thumb.

4.3.3 Loss Analysis

In order to investigate which types of human hand images are more challenging for TeachNet, we sort out images in the test dataset based on their overall loss $L_{student}$. Fig. 4.7(a) reveals that the complexity of the finger poses and the orientation of the human hand both account for the low accuracy of hand pose estimation. Intriguingly, Fig. 4.7(b) suggests that the complexity of the finger poses affects the accuracy more severely than the orientation of the human hand.



(a) 16 human hand images with biggest losses



(b) 16 human hand images with smallest losses

Figure 4.7 – Visualization of the human hand images in the test dataset based on the loss L_{stud} sorting. From left to right, upper row to lower row, the losses of the images are gradually become smaller in (b), and versa vice in (a).

4.4 Robotic Experiments

To verify the reliability and intuitiveness of the proposed TeachNet in actual teleoperation tests, robotic experiments in simulation and real world were both performed by five adult subjects. The simulation experiments mainly aim to visualize the imitation ability of TeachNet. Furthermore, the real robot experiments highlight the convenience and reliability of the proposed vision-based teleoperation system. The Shadow dexterous hand was controlled under position control mode, so the position commands from the TeachNet model will directly apply to the robot joints. The robot was controlled through ROS platform.

For choosing a camera, any depth camera which creates a high-quality 3D depth stream at close range is suitable. Therefore, the Intel RealSense F200 depth sensor was used. Its horizontal and vertical depth fields of view are 73° and 59°, respectively. The frame rate of the RealSense F200 is 30 Hz. The effective range of the depth solution from the camera is optimized from 0.2 m to 1.2 m for indoor applications. Fig. 4.8 shows the depth image of the human hand captured from different distances. When the hand is getting too close to the camera, some stray points in the depth image, causing the strap phenomenon in the leftmost image in Fig. 4.8. When the distance between the hand and the camera is over 80 cm, the information at the finger edge starts losing details. During the experiments, the poses of the teleoperators' right hand are limited to the viewpoint range of [70°, 120°] and the distance range of [35 cm, 65 cm] from the camera (see Fig. 4.11(a)). Since the vision-based teleoperation is susceptible to ambient light, all the experiments were carried out under a uniform and bright light source as much as possible.

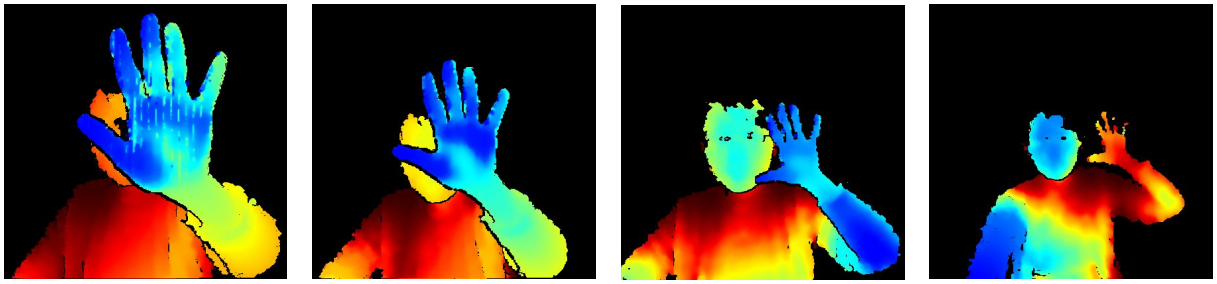


Figure 4.8 – The depth image of the human hand captured from different distances regarding the RealSense F200 camera.

The simulation experiments and grasping experiments run on the Ubuntu 16.04 system with the Intel Core i7-4720HQ CPU. The average computation time of the Teach Hard-Late method is 0.1051 s. Later, in-hand-grasp experiments were also tested on a Ubuntu 18.04 system computer with Intel i9-7900X CPU with 3.30 GHz and 128 G of RAM, and a GeForce GTX 1080 Ti GPU. Then, the average computation time of the Teach Hard-Late method is 0.015 s.

4.4.1 Simulation Experiments of Gesture Imitation

One user stood in front of the depth sensor and performed the gestures for the numbers 0-9 in American sign language and also random common gestures in a disordered way, then teleoperated the virtual Shadow robot in RViz [W33]. Qualitative results of teleoperation by the Teach Hard-Late method are illustrated in Fig. 4.9. On the one hand, the Shadow hand vividly imitates human gestures acted by hands of different sizes. These experiments demonstrate that the TeachNet enables a robot hand to perform continuous, online human hand imitation without explicitly specifying any joint-level correspondences. Owing to the fact that two wrist joints of the Shadow hand are fixed, whether or not the depth sensor captures the teleoperator's wrist will not affect the regression results. On the other hand, visible errors occurred mainly with the second joint, the third joint of the fingers, and the base joint of the thumb, probably caused by the special kinematic structure of the robot hand, occlusions of the human fingers, and the uncertain lighting conditions.

4.4.2 Grasping Experiments on Real Robot

In the real-robot experiments, the robot was controlled within a proper maximum force for each joint to ensure the safety of the robot. TeachNet was first validated in grasping tasks. As shown in Fig. 4.10, the robot hand succeeds in grasping objects with different shapes and different sizes, using both power grasp and precision grasp. Then the Teach Hard-Late method was compared with the DeepPrior++ HandIK method by an in-hand grasp and release [1]. Five objects (a water bottle, a small mug, a plastic banana, a cylinder can, and a plastic apple, see Fig. 4.11(b)) were used for this task. The objects were placed in the slave hand, which was in the open pose one at

[1]The Teach Hard-Late model was trained on the dataset which is generated based on the Shadow hand model with BioTac sensors, but during the grasp and release experiments, we implemented the model on a Shadow hand without BioTac sensors in another lab. In order to keep the same performance of different Shadow hands, the first joint of each finger is kept still.

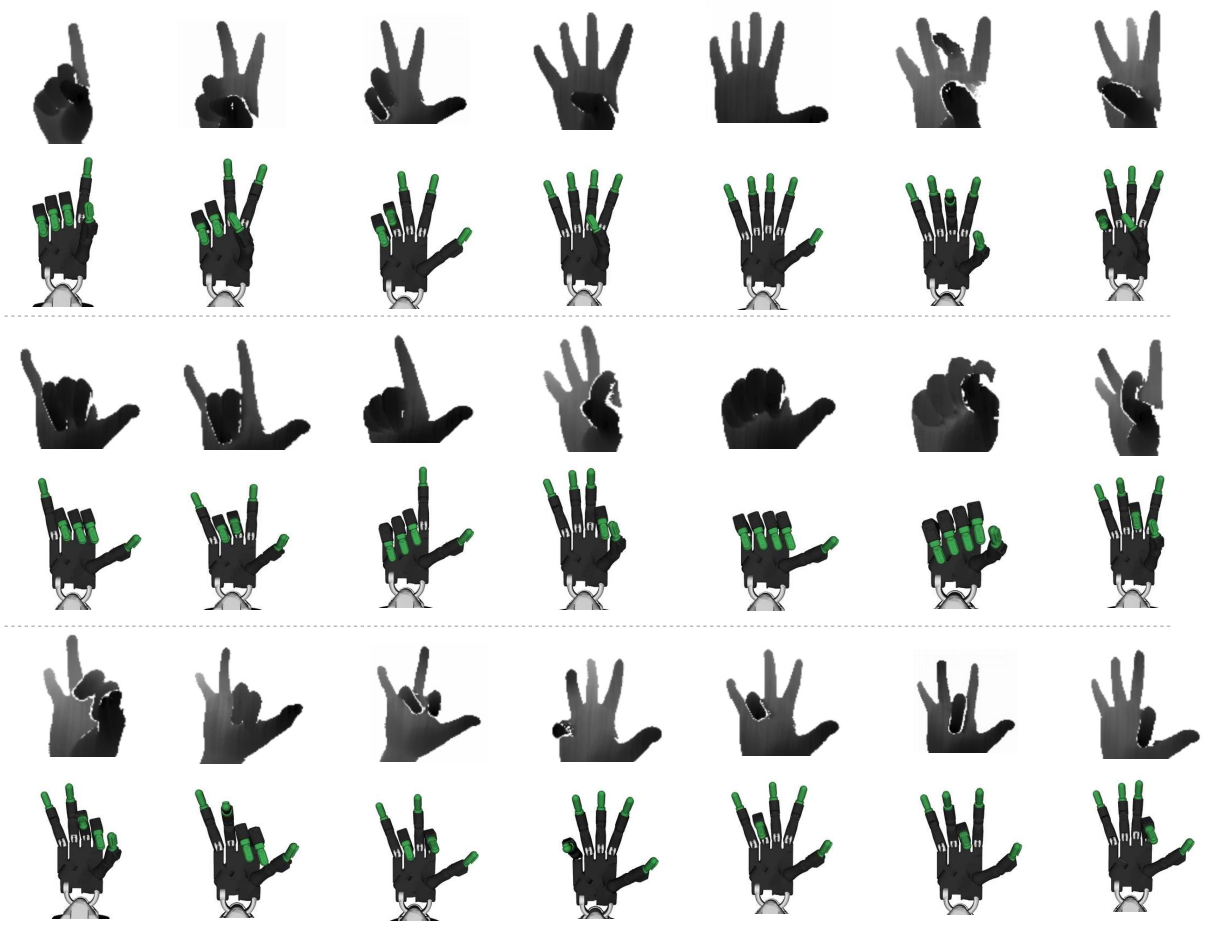


Figure 4.9 – Teleoperation results using the virtual Shadow hand with position control. The first, third, and fifth rows are depth images of the human hand captured by the Intel RealSense F200 depth sensor. The second, fourth, and sixth rows are the screenshots of the teleoperated Shadow hand in RViz.

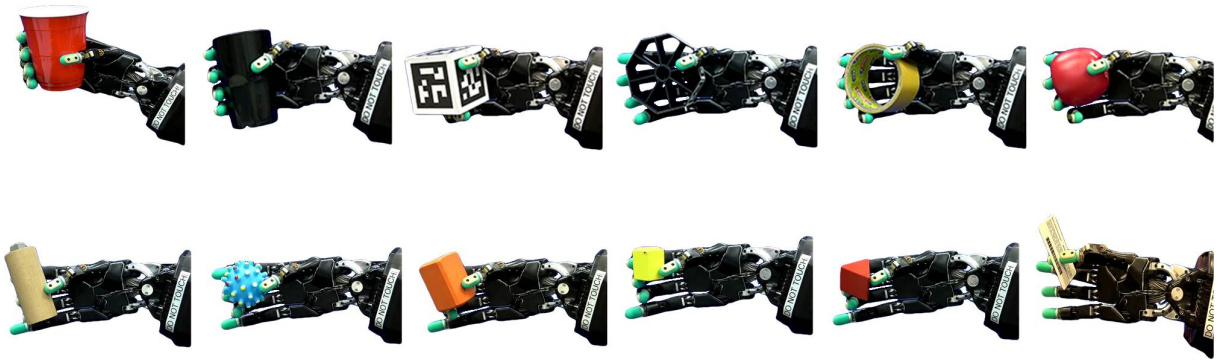
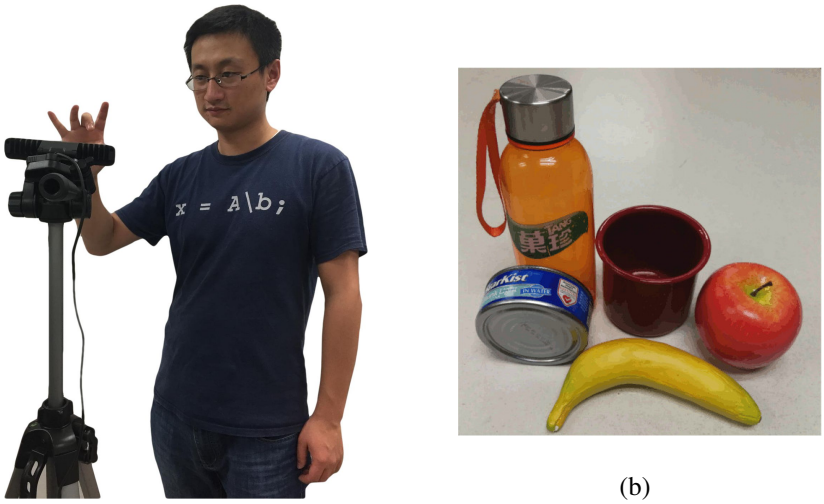


Figure 4.10 – Screenshots of grasping experiments using TeachNet. The upper and lower rows show examples of power grasp and precision grasp, respectively.



(a) (b)

Figure 4.11 – (a) The human status and (b) objects in the in-hand grasp and release task.

Methods	Bottle	Mug	Banana	Can	Apple	Average
Hand IK (CPU)	44.2	46.3	35.8	25.5	30.2	36.4
Ours (CPU)	23.7	18.8	25.8	19.8	15.6	20.728
Ours (GPU)	6.4	5.62	11.6	9.0	4.3	12.4

Table 4.2 – Average time (s) a novice took to grasp and release an object

a time to facilitate easier grasping with the robotic fingers. The operators are required to grasp the objects and to release them. The operators had to use power grasp for the water bottle and the mug, and to use precision grasp for the other three objects. If the user did not complete the task in four minutes, this trial was considered as failed.

The time to complete an in-hand grasp and release task was used as the metric for usability. Table 4.2 numerically shows the average time a novice took to grasp an object using each of the control methods. It clearly indicates that the low accuracy, especially for the thumb, and the post-processing of the HandIK solution results in a longer time to finish the task. The users needed to open the thumb first then perform proper grasp action, so the HandIK solution shows worse performance for the objects with a big diameter. Besides that, grasping the banana took the longest time on our method because the long and narrowly shaped object needed a more precise fingertip position. We also observed that the experiment time was significantly reduced when the tasks were tested on the computer with a better GPU and CPU. Fig 4.12 illustrate four examples for the real robot experiments.

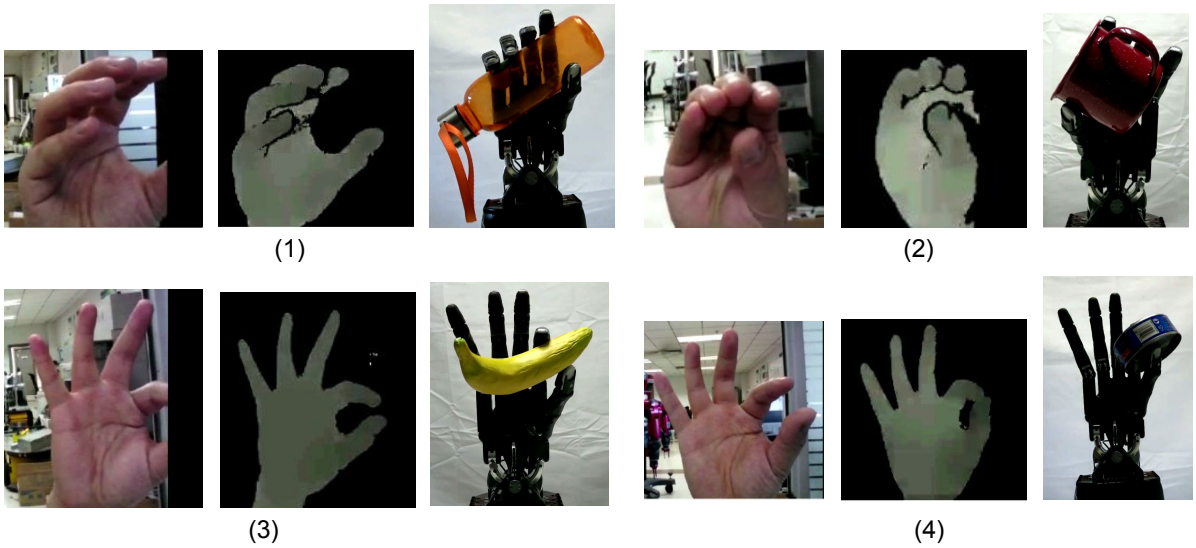


Figure 4.12 – Teleoperation results using the real-world Shadow hand. In (1)-(4), from left to right are the RGB images of the human hand, the depth images of the human hand and the robot status. In some RGB images, the hand is not fully captured because of the misalignment of the infrared camera and the color camera installed in the F200 depth sensor.

4.5 Discussion

This chapter presents an efficient vision-based teleoperation method for an anthropomorphic hand, which answers the first Research Question Q1 raised in section 1.3. This method develops an end-to-end teacher-student network (TeachNet), which finds kinematic mappings between the anthropomorphic robot hand and the human hand. The network evaluation and the robotic experiments verify that 1) TeachNet (the Teach Hard-Late method) is capable of modeling poses and the implicit correspondences between human demonstrators and robot imitators; 2) the end-to-end method allows novice teleoperators to grasp the in-hand objects faster and more accurately than the traditional two-step vision-based method; 3) the applicability of the proposed human-robot dataset to real-world experiments. Regarding the comparison in the fraction of frames whose maximum/average distance error below a threshold, TeachNet far outperforms the HandIK method or other stare-of-art hand pose estimation methods [55]. The best hypothesis for the significant improvement is the different exploration spaces. For human hand pose estimation methods, even though the keypoint positions are normalized to [0,1] for training, the data range is much more extensive than the joint angles of the robot hand. Therefore, the narrower exploration space promotes easier training and a higher accuracy. Code of the TeachNet model and video of the experiments are available at [W14].

Although the proposed TeachNet performs well in gesture imitation and grasping-and-releasing tasks, it has some limitations. First, TeachNet requires the position of the operator's hand relative to the camera within a proper range, otherwise it has a higher error of occluded joints. These could result from the simple preprocessing, the imperfect layer design of each branch in TeachNet, or the consistent supervision on one single layer being insufficient to exploit the geometrical resemblance. The wide variation in the global orientation of the human hand is always a training challenge in 3D hand pose estimation. We argue that a proper normalization

step of the input images will increase the robustness of the model to variations. Furthermore, designing a neural network model combining a higher-level representation and deeper regression would likely lead to more efficient training. Second, TeachNet only demonstrated simplistic grasping experiments on the real robot, so all experiments were employed by a robot hand with a fixed wrist and robot base. These limitations constrain the system from performing manipulation tasks that require arm motion, such as placing a block from one place to another. Accordingly, extending the teleoperation system to a hand-arm system is studied in the next chapters.

Chapter 5

Transteleop: Image-to-image Translation Inspired Robot Hand Pose Learning

The previous chapter presented TeachNet, a simple and effective robot hand pose estimation model, that finds kinematic mappings from a human hand to a robot hand and geometrical resemblance between them by a consistency loss. To overcome the remaining inaccuracy issues, this chapter aims to improve the end-to-end model from four aspects, network backbone, preprocessing module, layer assignment and loss functions. Inspired by latent pose hand poses extracted by deep generative models, a novel vision-based hand pose regression network (Transteleop) is developed using the image-to-image translation method. Transteleop observes the human hand through a low-cost depth camera and generates not only joint angles but also depth images of paired robot hand poses.

Section 5.1 presents the motivation of introducing image-to-image translation into the hand pose estimation task. Then section 5.2 discusses the two design choices of the network backbone. Consequently, the novel Transteleop model is described in section 5.3. Especially, two preprocessing modules are compared and discussed in order to learn invariance of the hand orientation and rotation. Later, section 5.4 first shows an accuracy comparison among several baselines from three aspects, preprocessing, network structure, and training dataset. Then, reconstructed hand images from two image-to-image translation inspired models are visualized. In the end, an analysis regarding the influence of camera viewpoint on the model accuracy is presented.

5.1 Why image-to-image translation

Recalling from previous chapters, we are dealing with the robot pose learning from human hand images. Despite the fact that the Teach Hard-Late method shows noticeable accuracy improvement over traditional methods, TeachNet still fails to find the correct pose of the robot when processing some complicated poses with self-occlusion. This could result from the consistent supervision on one single layer being insufficient to exploit the geometrical resemblance or the imperfect assignment of each branch in TeachNet. On the one hand, it may be helpful to deepen the convolutional layers or to extend the receptive field for perceiving the input. On the other hand, it would be great to find a method that can

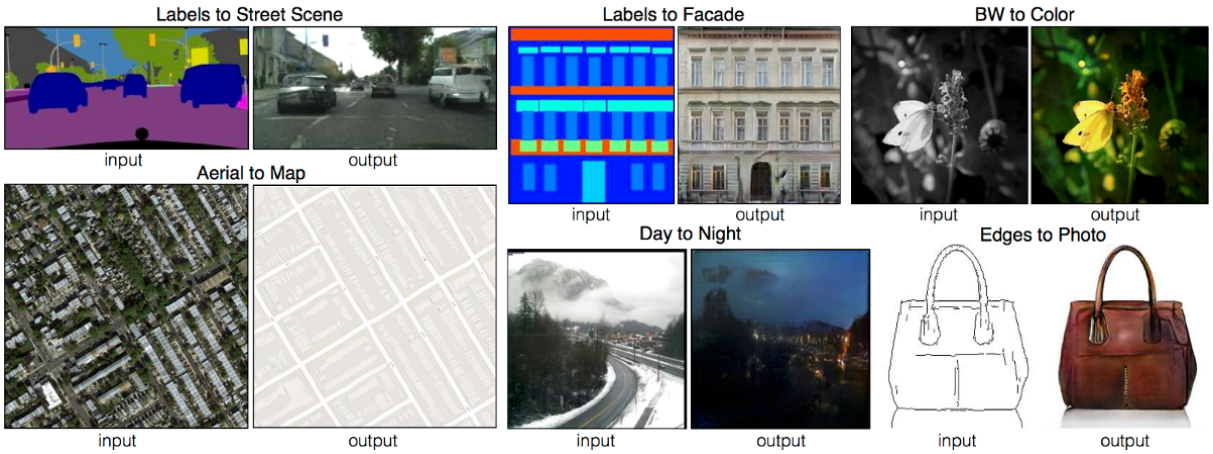
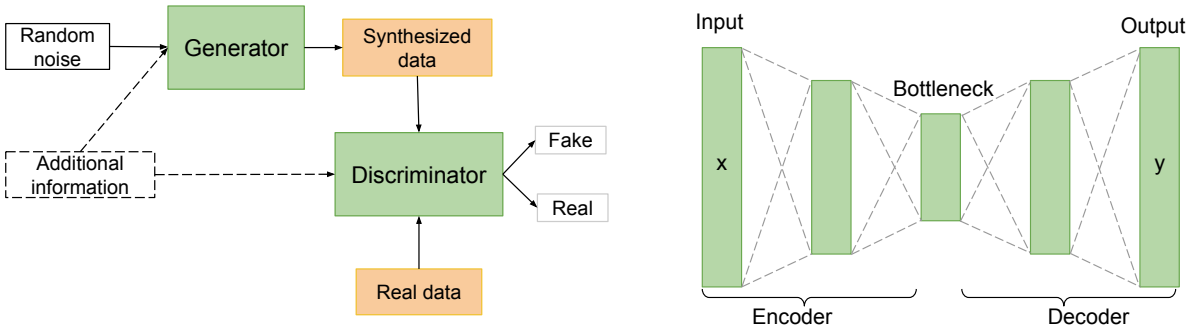


Figure 5.1 – Translation results of an image-to-image translation model [70]. Reprinted image: ©2017 IEEE.

thoroughly comprehend the kinematic similarity between the human hand and the robot hand than align the pose features on a single layer. Humans have a remarkable ability to fulfill complex tasks by observing others. How do we achieve this ability? Imitation requires inferring the goal/intention of the other person one is trying to imitate, translating these goals into one's own context, mapping the other's actions to first-person actions, and then finally using these translated goals and mapped actions to perform low-level control. In teleoperation, the humans dominate the inference part [137]. For the vision-based teleoperation method proposed in this thesis, the action mapping has been done in the dataset preparation part. Then, how does an end-to-end model translate the human goals into the robot's own context?

Assuming that if a robot could translate the observed scene (such as the human hand) to its scene (such as the robot hand), the robot would have perceived valuable hidden embeddings representing the resemblance of pose features between two image domains. Therefore, an image-to-image translation [70, 182, 121] concept could be used for vision-based teleoperation. Image-to-image translation, which aims to map a representation of a scene into another, is also a prevalent research topic widely used in collection style transfer, object transfiguration, and imitation learning. The key to image-to-image translation is to discover the hidden mapping features between two representations. Different generative models, such as restricted Boltzmann machines [23], generative adversarial networks (GANs), autoencoder models [76], and several variants of these models are widely used for image-to-image translation. Fig. 5.1 shows results of using conditional adversarial nets to translate an input image into a corresponding output image on a wide variety of applications [70]. In the robotic field, image-to-image translation methods have also been employed to map representations from humans to robots. Simith et al. [143] converted the human demonstration into a video of a robot and generated image instructions of each task stage by performing pixel-level image translation. They constructed a reward function for a reinforcement learning algorithm through translated instructions and evaluated the proposed method in a coffee machine operation task. Sharma et al. [137] decomposed third-person imitation learning into a goal generation module and an inverse control module. The goal generation module translated observed goal states from a third-person view to contexts of the robot by translating changes in the human demonstration images. All the above methods indicate that image-to-image translation methods are capable of learning latent features between



(a) GAN (b) Autoencoder

Figure 5.2 – Architecture of GAN and Autoencoder model. The block and the arrows with dash lines in (a) are required in cGAN, which has additional information for conditioning the generated images.

mapping pairings. Hence, it is worthwhile to study the image-to-image translation methods to extract common pose features between depth images of the human hand and the robot hand.

5.2 Network Backbone Design Choices

Imagine that we have a depth image I_R of a robotic hand from a fixed viewpoint and an image I_H of a human hand in random global pose, while the robotic hand in the image acts exactly the same as the human hand. Even though the bone length, the global pose, and the joint range of these paired hands are distinct, the pose feature Z_{pose} such as the skeletal shape and the whole silhouette will reveal the similarity between them. We believe that it would be very favorable to predict J_{hand} from the shared pose feature Z_{pose} rather than the bare I_H. In order to attain an instructive feature representation Z_{pose}, we adopt a generative structure that maps from image I_H to image I_R and retrieves the pose from the bottleneck layer of this structure as Z_{pose}. This learning scheme of mapping the human hand image to the corresponding robotic joint Q is formulated as

$$\begin{aligned} f_{trans} &: I_H \in \mathbb{R}^2 \rightarrow Z_{pose} \rightarrow I_R \in \mathbb{R}^2 \\ f_{joint} &: Z_{pose} \rightarrow Q. \end{aligned} \tag{5.1}$$

One typical generative structure is conditional GAN. GANs frame the unsupervised generative problem as a supervised learning problem with two sub-models: the generator model and the discriminator model. The input of the generator is a random vector from a Gaussian distribution, and the output is the generated new example. The discriminator model is a common classification model that tries to classify examples as either real (from the domain) or fake (synthesized). The two models are trained together in an adversarial way until the discriminator model is fooled about half the time, meaning the generator model generates plausible examples. GANs have led to a substantial boost in the quality of image generation due to an adversarial feedback loop. Conditional GANs (cGAN), in which both the generator and the discriminator are conditioned on additional inputs, are an important extension to GANs family [107]. The additional inputs could

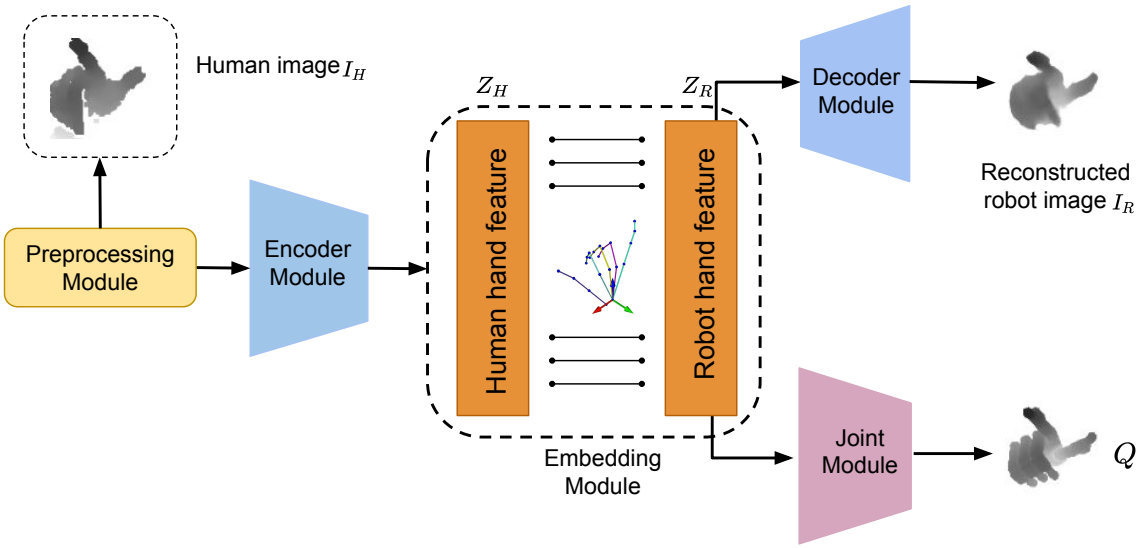
be aerial photos in the case of generating images of maps or hand-drawn cats in the generation of photographs of cats. Fig. 5.2(a) shows the architecture of the (conditional) GAN model. Alternatively, Autoencoders are known to learn efficient data codings in an unsupervised manner and is also widely used in image-to-image translation applications. The illustration of a typical Autoencoder model is shown in Fig. 5.2(b). The special bottleneck layer in an Autoencoder is imposed to maintain the compressed representation of the original input then fed to the decoder for reconstruction. Autoencoders are trained by minimizing the reconstruction error, which measures the input and output differences. Instead of the high realism and low blurriness of reconstructed images, Autoencoders mainly concentrate on efficiently memorizing the internal distribution of the input data in a compressed form. What we expect from a functional end-to-end neural network in a vision-based teleoperation application is learning hidden representations from the input data rather than from realistic robot images. To this end, we hypothesize that using an encoder-decoder style image-to-image translation method (Transteleop) for hand features H_{share} extraction is an efficient and suitable choice. The detailed accuracy comparison between these two backbones is quantified in section 5.4.

5.3 Transteleop

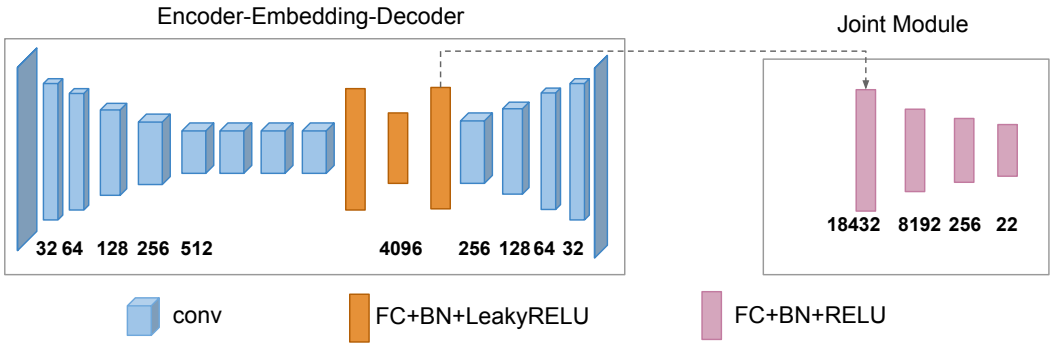
Transteleop is a novel end-to-end neural network, which extracts coherent pose features between the paired human and robot hand based on image-to-image translation methods, for vision-based teleoperation. Transteleop takes the depth image of the human hand as input, then estimates the joint angles of the robot hand, and also generates the reconstructed image of the robot hand. In the spirit of supervised learning, to enhance the richness of the features extracted from the image translation structure, a keypoint-based reconstruction loss is designed to focus on the local reconstruction quality around the keypoints of the hand. The structure of Transtelop is visualized in Fig. 5.3. Transteleop boils down to five modules: preprocessing module, encoder module, embedding module, decoder module, and joint module.

5.3.1 Preprocessing Module

One challenge of training Transteleop is that the poses of the human hand in the available datasets vary considerably in their global orientations. Spatial Transformer Network (STN) [71] has been applied to the input images for enhancing the geometric invariance of the models in human hand pose estimation tasks [115]. A spatial transformer is a differentiable module and can be included into a standard neural network model to provide spatial transformation capabilities, such as cropping, translation, rotation, scale, and skew. It contains a localization network, a grid generator, and a sampler, see Fig. 5.4. The localization network is a normal convolutional network and regresses the transformation parameters p. The dimension of the transformation parameters depends on the transformation type. For instance, an affine transformation p is 6 dimensional. The grid generator applies a sampling kernel to compute a grid of coordinates in the input feature map to each corresponding output pixel. Then, the sampler generates the output image given the grid coordinates and the input image. As the transformation is then performed on the entire feature map, this allows networks including spatial transformers to select regions of an image that are most relevant (attention) and transform those regions to a canonical, expected pose to simplify recognition in the following layers. In a common hand



(a) Architecture of the Transteleop model



(b) Sturcutres of the encoder-embedding-decoder module and the joint module

Figure 5.3 – Architecture of the Transteleop model. The encoder-embedding-decoder structure is an image-to-image translation branch, which is fed depth images of a human hand I_H and produces reconstructed depth images of the robot hand I_R. The joint module takes the pose embedding from the encoder-decoder structure and predicts the robot's joint angles J_{hand}. The preprocessing module is a spatial transformer network that explicitly permits the spatial manipulation of input images.

pose estimation application, the 3D positions of the hand keypoints are the regression goal. If an STN module transforms the input images, the output 3D positions depend on the newly transformed images. Some solutions such as adding a spatial de-transformation network before the output layer [27] or estimating the canonical hand pose and the hand orientation by two parallel networks [184] was proposed. However, these methods have more complex network structures and make the training more difficult.

Instead of an online training module in Transteleop, another method OBB (Orientated bounding box)-based point cloud normalization, was firstly utilized in 3D hand pose estimation using point clouds [55, 56]. Performing Principal Component Analysis (PCA) on the input points, this method builds a new bounding box frame aligning with the eigenvectors of input points'

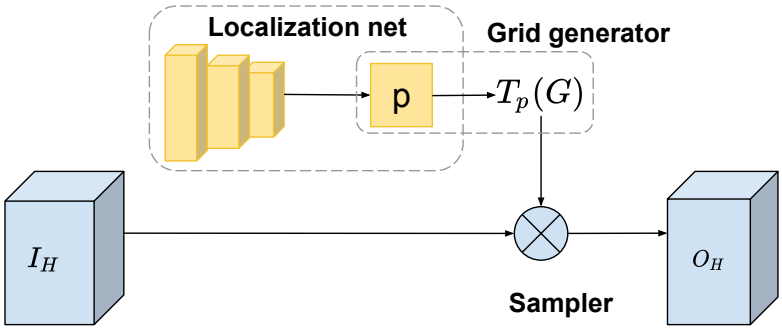


Figure 5.4 – Architecture of a spatial transformer module. The input human hand image I_H is passed to a localization network which regresses the transformation parameters p. The regular spatial grid G over O_H is transformed to the sampling grid $T_p(G)$, which is applied to the sampler S, producing the warped output feature map O_H. The combination of the localization network and sampling mechanism defines a spatial transformer. Adapted from [71].

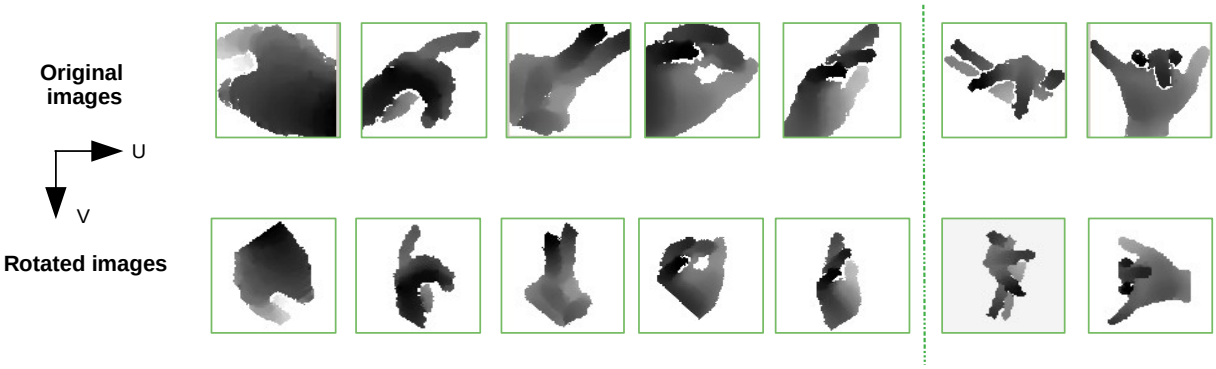


Figure 5.5 – Example images by an PCA-based rotation. The first row presents the depth images in the original UV coordinate with various global poses. The second row presents the sampled and rotated depth images in the new UV coordinate, of which the hand orientations are consistent. The last two columns show the special cases that the fingers spread widely in the original U-axis.

covariance matrix, which correspond to eigenvalues from largest to smallest. The input points are transformed to the new bounding box frame, then are shifted to zero mean and scaled to unit size. This OBB-based normalization has been proved that it can normalize point clouds with more consistent orientations and make the network easier to learn the hand articulations than the normalization without rotation. For a 2D depth map, the OBB-based normalization can be simplified to a PCA-based rotation, which rotates the 2D hand pixel in a new UV coordinate. The new V-axis is aligned with the eigenvector of the input pixels' covariance matrix, which corresponds to the largest eigenvalue. Then all pixels are transformed into the new UV coordinate. In case some pixels are rotated out of the image, proper padding is added before rotation. The rotated images are exampled in Fig. 5.5. The hands in most rotated images have consistent orientations, but some hands, where the fingers spread widely in the original U-axis, have a 90° orientation difference compared to other rotated images.

In this chapter, both the online STN module and the offline PCA-based rotation methods are studied (The newly generated images by the STN module are shown in Fig. 5.10 on page 67). As the outputs Q_R and I_R are invariant to the spatial transformation of the input image, the ground

truth does not need to be modified. Therefore, the STN module can be added before the encoder module without any other extra structures. The training results using these two preprocessing methods are discussed in section 5.4. Note that the depth images fed into these two preprocess modules have been extracted from the raw depth image as a fixed-size cube around the hand and are normalized to $[-1, 1]$. Same as the section 4.3.1, the input depth images are conducted noise removal as well.

5.3.2 Encoder-embedding-decoder Module

The encoder takes a depth image of a human hand I_H from various viewpoints and discovers the latent feature embedding Z_{pose} between the human hand and the robot hand. We use six convolutional layers containing four downsampling layers and two residual blocks with the same output dimension. Thus, given an input image of size 96×96, the encoder computes an abstract $6 \times 6 \times 512$ dimensional feature representation Z_H.

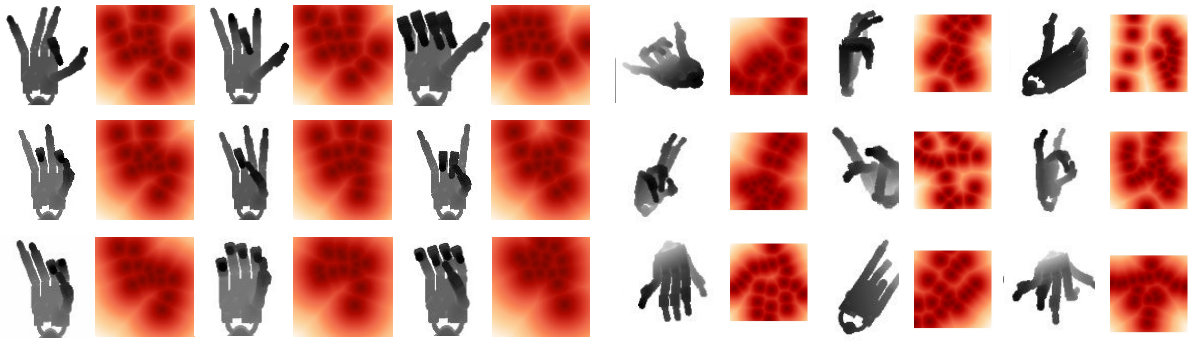
Similar to [121], we connect the encoder and the decoder through two fully-connected layers instead of convolutional layers because the pixel areas in I_H and I_R in our dataset are not matched. This design results from the fact that a fully-connected layer allows each unit in the decoder to reason on the entire image content. In contrast, a convolutional layer cannot directly connect all locations within a feature map. Through the embedding module, we extract the useful 4092-dimensional feature embedding Z_{pose} from Z_H.

The decoder aims to reconstruct a depth image of the robot hand $\hat{I}_R$ from a fixed viewpoint from the latent pose feature Z_{pose}. One fully-connected layer connects the feature from Z_{pose} to robot feature vector Z_R. Four up-convolutional layers with the learned filters and one convolutional layer for image generation follow.

Unlike common image-to-image translation tasks, the generated image I_R should care more about the accuracy of local features such as the position of fingertips instead of global features such as image style. This is because the pixels of the joint keypoints possess more information about the hand pose. Regarding the Shadow hand, as depicted in Fig. 3.3, each finger has three keypoints. Therefore, we designed a novel keypoint-based reconstruction loss to capture the overall structure of the hand, and to concentrate on the pixels around the 15 keypoints of the hand. We regard the eight neighboring pixels of each keypoint as important as these keypoints themselves. The scaling factor $\alpha \in [0, 1]$ (see equation 5.2) of each pixel error is determined by how close this pixel is to all keypoints, shown in Fig. 5.6.

$$\begin{aligned} D_i &= \min \|P_i - N\|^2 \\ \alpha_i &= 1 - \frac{D_i}{D_{max}}, \end{aligned} \tag{5.2}$$

where P_i is the position of i-th pixel in UV coordinate, N is the position array of all keypoints and their neighboring pixels, D is the minimum distance from each pixel to pixels in N, α_i is the scaling factor of the i-th pixel.



(a) Dataset1 (b) Dataset2

Figure 5.6 – The example heatmaps of scaling matrix α in dataset1 (the wrist of the robot is always fixed) and dataset2 (the wrist pose of the robot is as same as the human). The darker color illustrates how important these pixels are. Examples of paired human-robot depth images at the same wrist pose in our dataset.

The reconstruction loss $\mathcal{L}_{recon}$ is an L2 loss that prefers to minimize the mean pixel-wise error but does not encourage less blurring, defined as:

$$\mathcal{L}_{recon} = \frac{1}{M}\sum_{i=1}^{M} \alpha_i \cdot (I_{R,i} - \hat{I}_{R,i})^2, \tag{5.3}$$

where M is the number of pixels, $\hat{I}_R$ is the ground truth of the robot hand image.

5.3.3 Joint Module

The joint module focuses on deducing 19-dimensional joint angles Q from the latent feature embedding of the decoder. We choose Z_R instead of Z_{pose} because Z_R has richer features depicting the pose feature of the robot hand. Three fully-connected layers are employed in the joint module. The joint module is supervised with a MSE loss $\mathcal{L}_{joint}$

$$\mathcal{L}_{joint} = \frac{1}{N}\|Q - J\|^2, \tag{5.4}$$

where N is the number of joints and J denotes the ground truth joint angles.
Overall, the complete training objective:

$$\mathcal{L}_{hand} = \lambda_{recon} \cdot \mathcal{L}_{recon} + \lambda_{joint} \cdot \mathcal{L}_{joint}, \tag{5.5}$$

where λ_{recon} and λ_{joint} are the scaling factors.

5.4 Network Experiments

5.4.1 Training Details

To see whether the inconsistent orientation and position of the input and reconstructed images bring more training challenges, we trained Transteleop on Dataset1 and Dataset2, respectively. This paired dataset records nine depth images of the robot hand from different viewpoints simultaneously, corresponding to one human pose. Considering abstract an explicit kinematic configuration from the robot image, only the robot images taken in front of the robot are used. The image preprocessing are described in subsection 5.3.1.

The weights of the model are initialized by normal distribution $\mathcal{N}(0, 0.02)$. To optimize our networks, we use minibatch stochastic gradient descent and apply Adam optimizer with a learning rate $l_r = 0.002$ and momentum parameters $\beta_1 = 0.5, \beta_2 = 0.999$. The learning rate is decayed by 0.5 every 80 epochs. We use $\lambda_{recon} = 1, \lambda_{joint} = 10$. At inference time, only the encoder module and the joint module for joint angle regression are used. The average inference time is 0.027 s, tested on a computer with Intel i9-7900X CPU with 3.30 GHz and 128 G of RAM, and a GeForce GTX 1080 Ti GPU.

5.4.2 Baselines Comparison

First, to find the best preprocessing module which learns invariance to the hand orientation and rotation, the STN- and PCA-based preprocessing modules were compared. These two models are referred as "TranstelopSTN" and "TransteleopPCA".

Second, to evaluate whether Transtelop could learn more indicative visual representations, four baseline models were trained on the paired human-robot dataset, Dataset2.

1) GANteleop: a model that adds a PatchGAN discriminator in Transteleop and an adversarial loss based on the "pix2pix" framework [70], shown in Fig. 5.7);

2) TeachNet: an end-to-end robot hand pose estimation model with an auxiliary consistency loss [93];

3) Humanonly: a model that removes the decoder module in Transteleop;

4) Robotonly: a model that removes the decoder module in Transteleop and is fed by the depth images of the robot hand instead of the human hands.

Then, to find out whether Dataset2 improves posture learning, we also trained a model called "TransteleopFix" on Dataset1, where the robot wrist is fixed in the robot depth images.

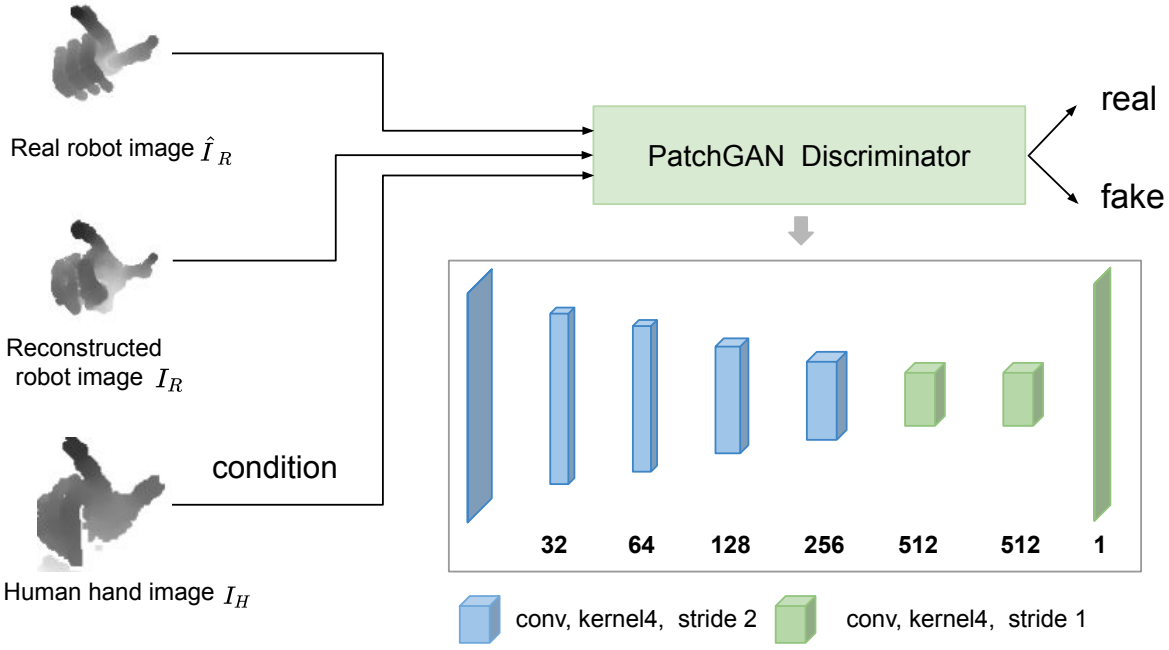
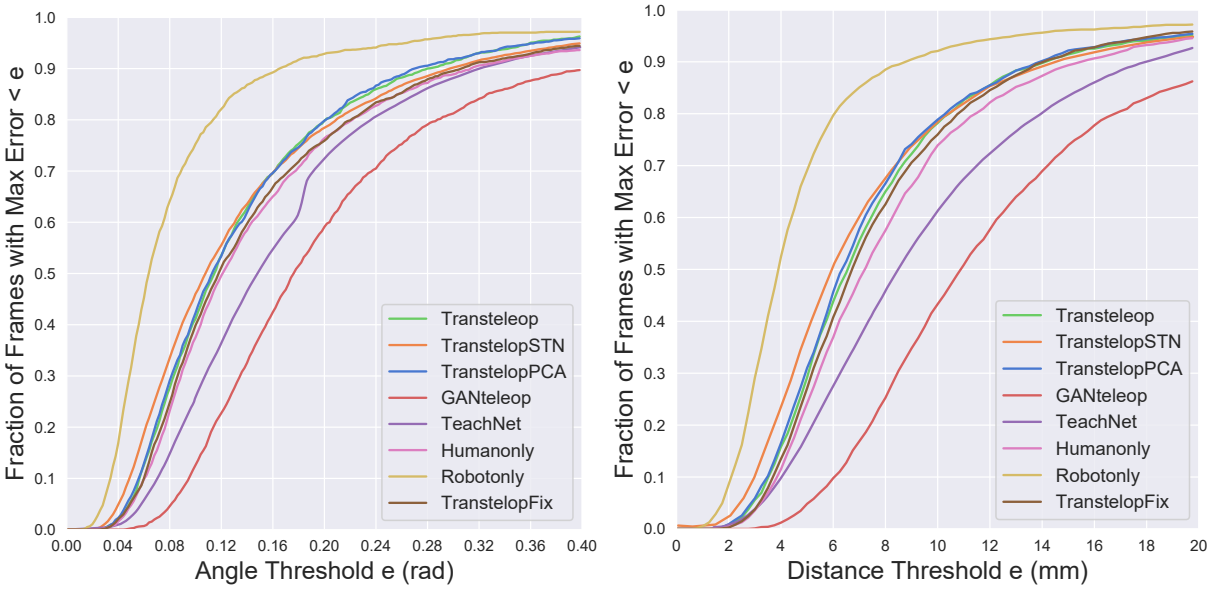


Figure 5.7 – Architecture of the GANteleop model. PatchGAN is a type of discriminator for GANs which only penalizes structure at the scale of local image patches instead of the whole image. The input of the human hand image is the additional information for training conditional GANs.



(a) (b)

Figure 5.8 – The fraction of frames whose absolute maximum joint angle/distance error is below a threshold between the Transteleop approach and different baselines on our test dataset. These results show that the Transteleop model has the best accuracy over all evaluation metrics except for the Robotonly model.

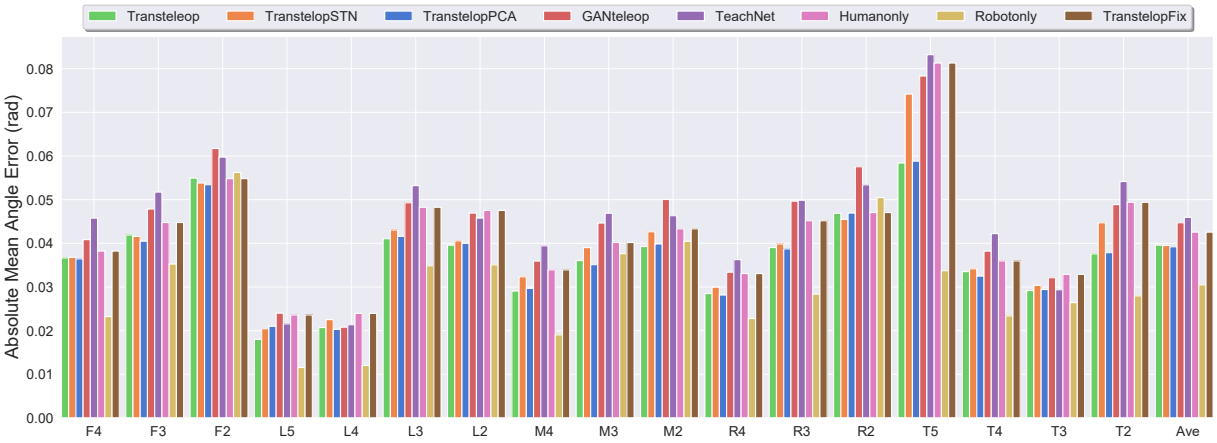


Figure 5.9 – Comparison of the absolute average angle error on the individual joint between the Transteleop approach and different baselines on our test dataset. F means the first finger, L means the little finger, M means the middle finger, R means the ring finger, T means the thumb. 2, 3, 4, 5 mean the n-th joint of the finger.

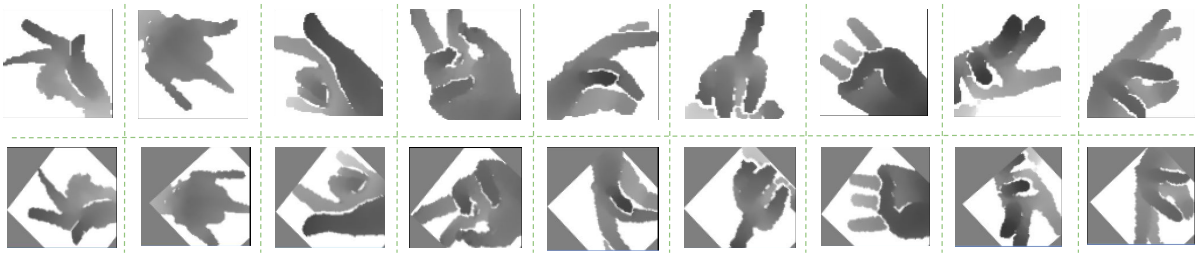


Figure 5.10 – Example images generated by the STN preprocessing module in Transteleop. The first row presents the original input of the depth images. The second row presents the corresponding generated images by the STN preprocessing module. The black borders of the second-row image outputs are generated after the image rotation.

We compared the fraction of frames whose maximum angle and distance error are below a threshold and the average angle error over all angles, as shown in Fig. 5.8 and Fig. 5.9. Comparing the Transtelop model with the TransteleopPCA model, there is no significant improvement resulting from the PCA-based preprocessing. This suggests that the OBB-based normalization brings a slight appearance normalizing effect to this task but does not significantly promote the hand pose transform to a canonical pose. To analyze the performance of the TranstelopSTN model, we also show the intermediate outputs from the STN module in Fig. 5.10. Note that the black borders of the STN generated outputs are generated after the image rotation. We surprisedly find that the STN module not only rotates or translates the input images but also flips all images. Even though the STN generated images have more consistent hand poses, the TranstelopSTN model only has higher accuracy regarding the high-precision thresholds, especially when the maximum angle error is lower than 0.16 rad and the maximum distance error lower than 8 mm. The TranstelopSTN model also performs worse than Transteleop comparing the average angle error on the individual joint.

Angle (rad)		Transteleop	GANteleop	TeachNet[93]	Humanonly	TransteleopFix
Max Err. <=	0.1	**41.15%**	12.60%	24.63%	37.51%	39.88%
	0.15	**66.77%**	37.82%	50.11%	61.87%	63.51%
	0.2	**79.82%**	59.10%	72.04%	76.40%	75.92%
	0.25	**86.98%**	73.42%	81.94%	84.06%	84.19%
Ave Err.		**0.030**	0.063	0.046	0.033	0.0343

Distance (mm)		Transteleop	GANteleop	TeachNet[93]	Humanonly	TransteleopFix
Max Dis. Err. <=	4	**15.83%**	1.21%	9.98%	11.50%	13.50%
	6	**43.82%**	9.78%	27.53%	36.91%	40.77%
	8	**65.00%**	25.26%	46.44%	57.52%	62.56%
	10	**78.21%**	43.31%	61.41%	73.91%	75.99%
Ave Dis. Err.		**1.14**	2.21	1.47	1.25	1.22

Table 5.1 – Angle/distance accuracy under high-precision conditions and average angle/distance error

Furthermore, the quantitative results of the angle/distance accuracy under high-precision thresholds are listed in Table 5.1. The Robotonly model significantly outperforms all other baselines over all evaluation metrics because of the matched domain. We also note that the performance of GANteleop is much worse than Transteleop because the discriminator in GANteleop focuses on pursuing realistic images and weakens the supervision of L_{joint}. Comparing two image-to-image translation based models (Transteleop, TransteleopFix) and TeachNet, the TeachNet model gets at least 15% lower accuracy below a maximum joint angle regarding the high-precision condition. We infer that the image translation structure seizes more valuable pose features than the alignment mechanism between two layers in TeachNet. Except for the GANteleop model, all other models show at least 10% improvement of the accuracy over TeachNet in the high-precision condition. It indicates that the deeper network layers play a more critical role than the supervision of the consistency loss in TeachNet.

Meanwhile, Transteleop shows an average 2.805% improvement of the accuracy compared to the Humanonly model. This result suggests that the additional reconstruction loss assists in gaining more indicative pose features. Also, Transteleop performs better than TransteleopFix, indicating that the new pairwise human-robot dataset with the same wrist orientation enables the model to learn more indicative shared pose features.

Moreover, the TransteleopPCA model has the lowest absolute average angle error, and all methods' absolute average angle error is lower than 0.09 rad. The highest error happens on THJ5 because there is a big discrepancy between the human thumb and the Shadow thumb. For four fingers, J2 and J3 have higher errors than J4 or J5. The reason is that these two joints are involved in the most complex finger motions and have wider joint angle ranges than J4. Last but not least, the fixed distal joints of the robot hand affect the overall accuracy of models as well.

5.4.3 Reconstructed Images

The examples of the reconstructed images by Transteleop and GANteleop are visualized in Fig. 5.11. Obviously, the images generated by GANteleop keep more details and are less blurry than by Transteleop. Nevertheless, the reconstructed images from Transteleop are still akin to the ground truth images and contain the rough silhouette.

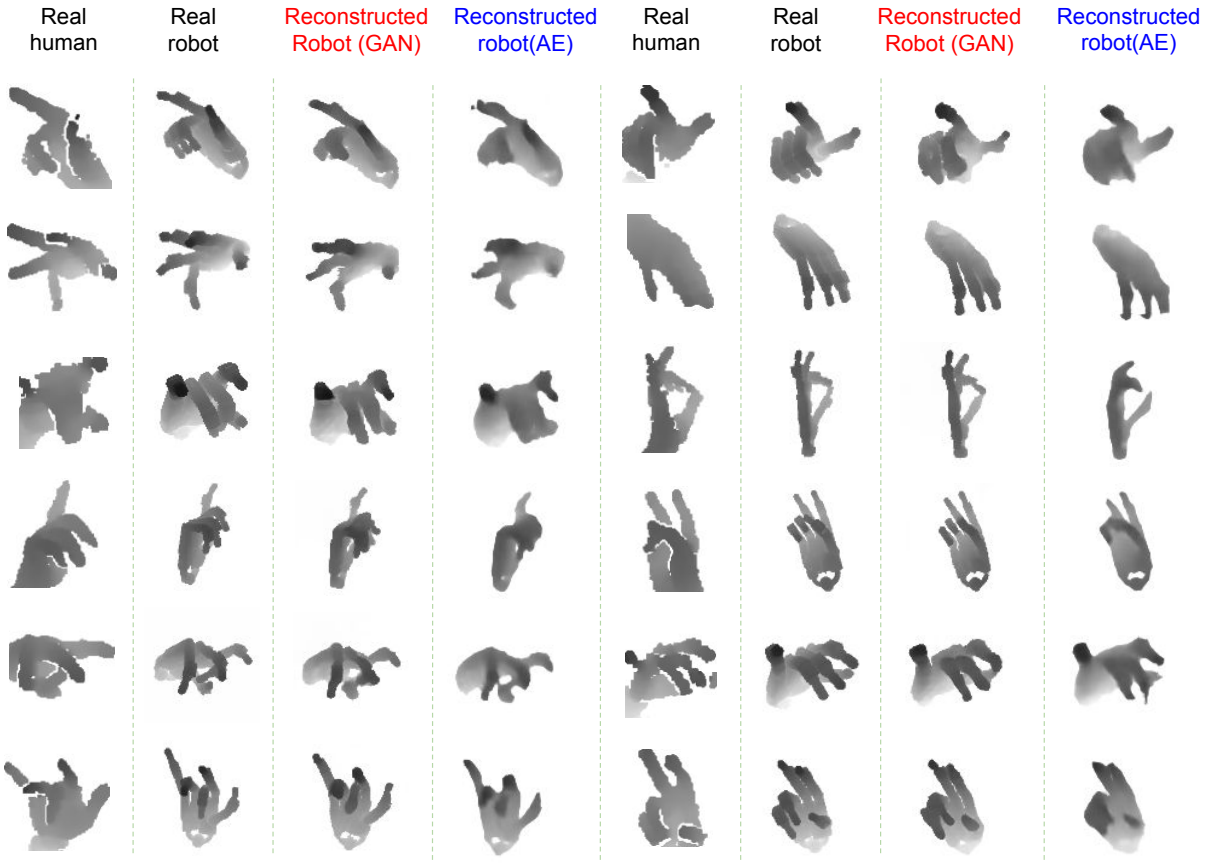
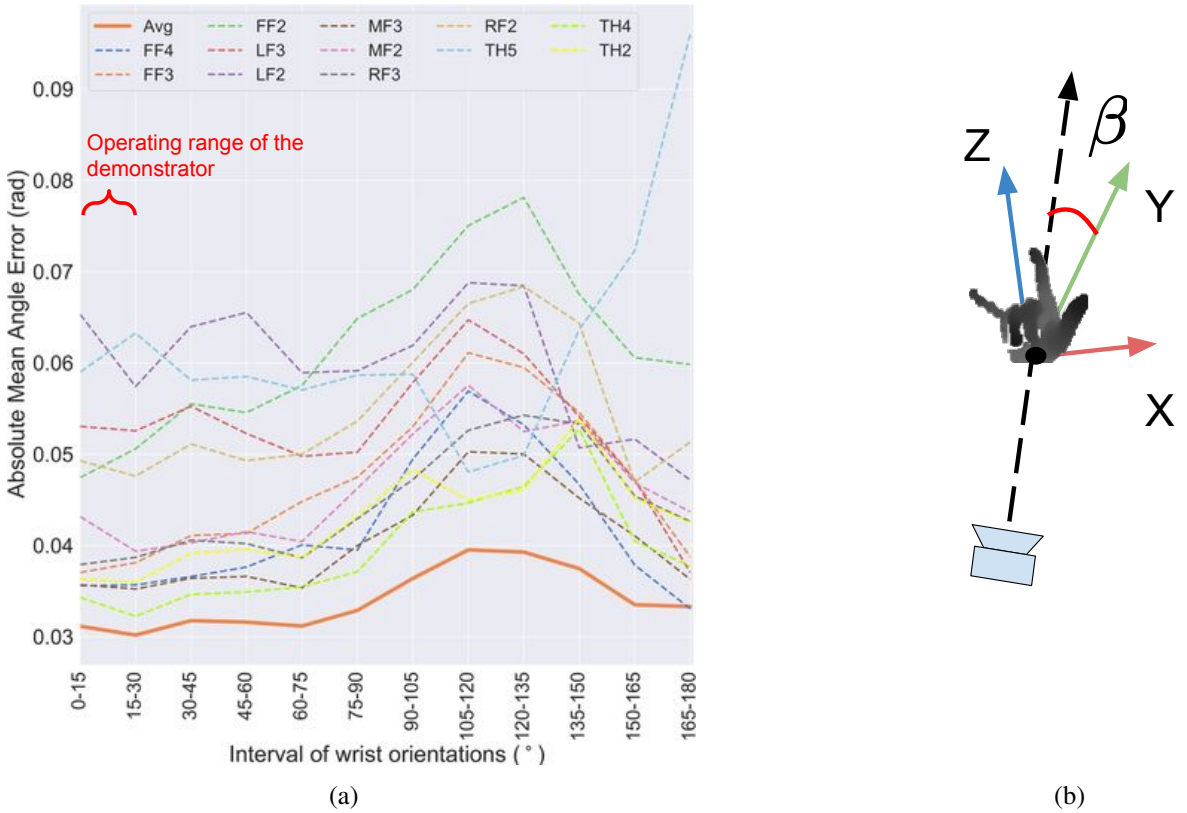


Figure 5.11 – Reconstructed robot images generated from Transteleop and GANteleop.

5.4.4 Analysis Based on Viewpoint

Since the inaccuracy issues due to the self-occlusion of the fingers result from the camera viewpoint to a certain extent, the mean error of images from different camera viewpoints was examined. The test dataset was divided into twelve portions based on the angle between the camera direction and the Y-axis of the hand. Fig. 5.12(b) exhibits the average absolute angle error on the individual joint (except for joints LF5, LF4, MF4, RF4, and TH3, which have a relatively smaller mean error than other joints) tested on the twelve sub-datasets. The mean errors of all joints manifest a noticeable rise when the viewpoints are in the $[75°, 150°]$ range because of the amount of self-occlusion. Especially, the mean errors of thumb joint 5, which is one of the essential joints for manipulation, is 0.096 rad in the $[165°, 180°]$ range. Surprisingly, most joints perform well at the $[150°, 180°]$ range.



(a) (b)

Figure 5.12 – (a) The absolute average angle error on the individual joint tested on twelve intervals of viewpoints. (b) β is used to indicate the angle of the camera viewpoint, which is the angle between the camera direction and the Y-axis of the hand.

To figure out this phenomenon, we analyze the average number of occluded joints for each sub-dataset, indicating the posture complexity. A joint occlusion is defined by thresholding the distance between the joint's depth annotation value and its re-projected depth value. The average number of the occluded joints out of 19 joints is shown in Table. 5.2. Apparently, there are some straightforward human hand images with lower posture complexity at $[150°, 180°]$ viewpoint range in our test dataset, therefore the mean error at this range is lower than expected. If we only focus on the $[0°, 150°]$ viewpoint range, this result reveals that it is easier to determine hand poses in a good camera viewpoint. Therefore, it is beneficial to keep the operating range of the human hand in the $[0°, 30°]$ viewpoint range.

Ori.(°)	0-15	15-30	30-45	45-60	60-75	75-90
Num	4.43	4.61	4.99	5.44	5.94	6.43
Ori.(°)	90-105	105-120	120-135	135-150	150-165	165-180
Num	6.38	5.91	5.64	5.32	3.75	3.65

Table 5.2 – The average number of occluded joints out of 19 joints tested on twelve intervals of viewpoint angle

5.5 Discussion

This chapter further studies deep-learning-based human hand pose estimation (Research Question Q1) and presents the preprocessing methods, network structure, training objectives, and evaluations of a novel robot joint estimation model, Transteleop. Transteleop observes the human hand through a low-cost depth camera then generates joint angles and depth images of the paired robot hand through an image-to-image translation process. A keypoint-based reconstruction loss explores the resemblance in appearance and anatomy between human and robotic hands and enriches the local features of reconstructed images. The accuracy evaluation in three metrics confirms that Transteleop learns more indicative visual representations than other baselines. Therefore, applying Transteleop to the markerless vision-based teleoperation for the anthropomorphic robot hand will most likely accomplish interesting manipulation tasks. In the next chapter, combing with a robot arm teleoperation method, Transteleop shows a good performance in robot hand-arm manipulation applications. Code of the Transteleop model is available at [W15].

The viewpoint analysis verifies that it is easier to determine hand poses in a good camera viewpoint. This result suggests that maintaining the human hands captured from the best viewpoint and at a suitable distance during the experiments will increase the joint accuracy, thereby fostering more effective dexterous manipulation.

Chapter 6

Multimodal Hand-Arm Teleoperation System Based on Vision and IMU

With the end-to-end learning networks (TeachNet and Transteleop) available, the kinematic mapping from the human hand to the robot hand is learned, and markerless vision-based teleoperation for the anthropomorphic hand is achieved. The challenge to extending the hand teleoperation system to a hand-arm system lies in the good design of systematic control. This chapter describes how to teleoperate the PR2 robot [W5] by the end-to-end model integrated with an IMU-based arm control system.

Section 6.1 elaborates on the goal, the design and the hardware platform of the hand-arm teleoperation system. Section 6.2 covers the software usage of the IMU-based device, deals with how to send the motion data from a Windows system to a Linux system through the ROS platform, and explicates the control strategies of the robot arm. With a 3D-printed camera holder described in section 6.3, the teleoperator is not limited to a fixed workspace anymore. Finally, section 6.4 shows the efficiency and stability of the proposed multimodal teleoperation system by a variety of complex manipulation tasks that go beyond simple pick-and-place operations. The experimental video is available at [W12].

6.1 System Description

Our goal is to build a robotic hand-arm teleoperation system in which the teleoperator performs natural hand motions for a series of hand-arm coordinated manipulation tasks in an unlimited visual workspace. To set up such a system, the markerless end-to-end vision-based model, Transteleop, is formulated to teleoperate the anthropomorphic hand. Subsequently, we should determine a real-time and smooth arm teleoperation method. Based on the survey in section 2.1.1, IMU-based devices are suitable to achieve accurate control of the robotic arm, and they are convenient to implement. Therefore, an IMU-based device is chosen to control the arm. Let I_H be the image of a human demonstrating hand poses of manipulation tasks as observed by a depth camera. In this system, the vision part aims to take advantage of a neural model that takes I_H as inputs and predicts joint angles Θ of the robot hand, while the IMU part intends to map the absolute motion of the human arm to the robot arm.

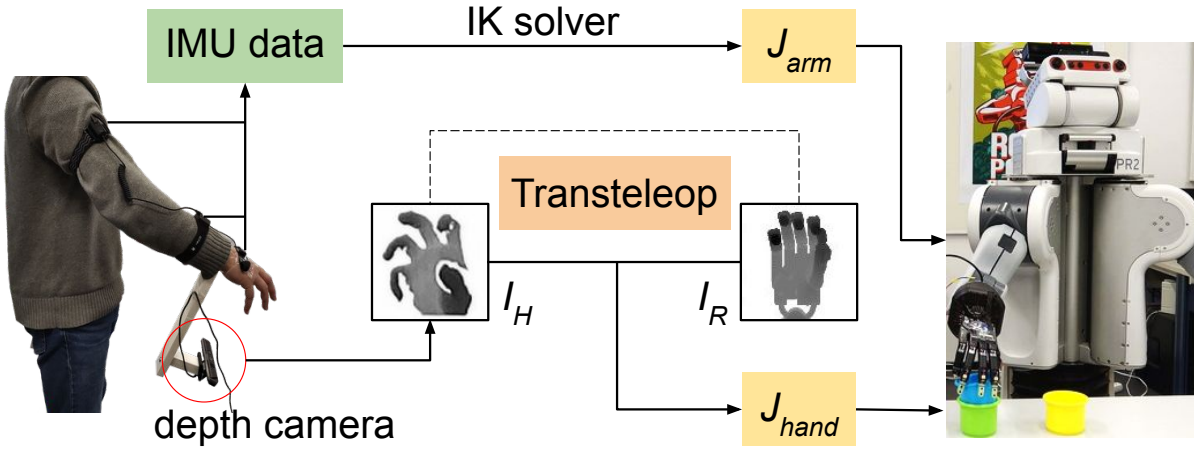


Figure 6.1 – The multimodal teleoperation system is built on the markerless vision-based model, Transteleop, which predicts the joint angles of an anthropomorphic hand, and an IMU-based arm control method. Transteleop gains meaningful pose information from depth images of a human hand I_H based on an image-to-image translation structure and predicts the joint angles J_{hand} of the robot hand. This multimodal system implements different manipulation tasks such as pick and place, cup insertion, object pushing, and dual-arm handover tasks on a PR2 robot with a Shadow hand installed on its right arm.

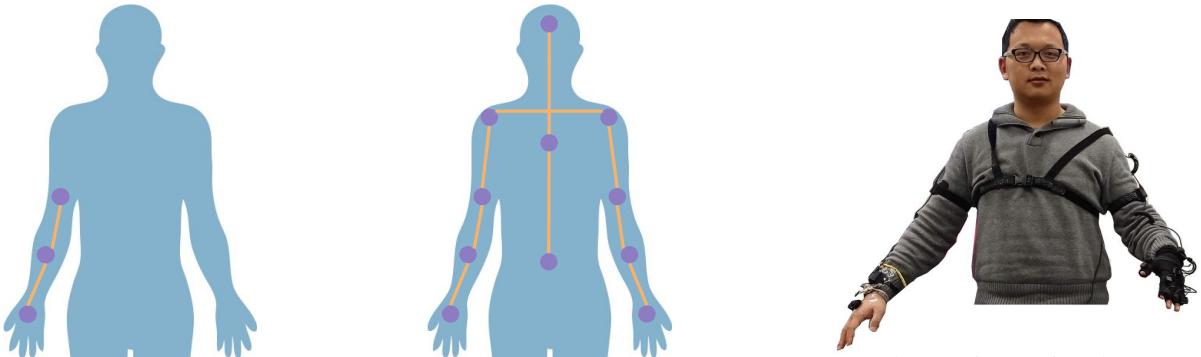
One requirement of vision-based methods is that the hand of the teleoperator must stay inside the limited view range of the camera system. This restriction impedes the operator from completing manipulation tasks that need a wide working area. To achieve a dexterous hand-arm teleoperation system without workspace constraint, we develop a camera holder to mount the camera on the arm of a human. In conjunction with the Transteleop method, a camera holder, and IMU-based arm teleoperation, this multimodal system can not only maintain the natural motion of the human fingers but also allow for flexible arm motion. Fig. 6.1 illustrates the framework of proposed method for hand-arm teleoperation. The robot system used in this work is a PR2 robot with a 19-DoF Shadow hand mounted on its right arm. Unlike the 7-DoF left arm of PR2, the right arm of PR2 only has 5 DoF due to the attached Shadow hand. In addition to that, five Syntouch Biotac tactile sensors [W37] are retrofitted at the fingertips of the Shadow hand.

6.2 IMU-based Arm Teleoperation Method

We chose the IMU-based Perception Neuron (PN) system as the arm-control device. According to the manufactures, PN system is one of the most versatile, adaptable motion capture systems in the world. It is prevalent in body tracking, animation, and VR game interaction [W26]. Each Neuron sensor in the capture system is an IMU composed of a gyroscope, accelerometer, and magnetometer. The output rate is 60 Hz or 120 Hz depending on whether the number of active Neurons is larger than 17. A supporting Hub collects motion data from the Neuron sensors then sends the data to a computer via USB or wireless connection through any standard 2.4 G wireless router.

6.2.1 Axis Neuron Software

The Axis Neuron software used to communicate with Perception Neuron hardware can be downloaded from [W25] and installed in the Windows system. As we use the ROS platform in the Ubuntu system to control the robot, the Axis Neuron is therefore installed in a Windows virtual machine by VMware Workstation 15 Player [W45] in a Linux computer. In our experiments, we use the PN to control one arm or both arms of the PR2 robot. Hence, we use the suit's single-arm mode (3 neurons) or upper-body mode (11 neurons). The three neurons for single-arm mode are sufficient to capture the motion of the right palm, right upper arm, and right forearm. In upper-body mode, six neurons are used to capture motions of two arms, two neurons used for the left and right shoulder, one neuron for the spine, one neuron for the stomach, and one neuron for the head. The illustration of the sensor distribution on the human body is shown in Fig. 6.2. In our experiment, we glue the palm sensor to the back of the human hand instead of wearing a glove in order to keep the hand markerless in the images. The neurons at stomach and head are not required in our setup.



(a) Single-arm mode (b) Upper-body mode (c) A human is wearing the upper-body suit

Figure 6.2 – Illustration of the neuron distribution on the human body in (a) single-arm mode and (b) upper-body mode. The purple dots indicate the position of the neurons. (c) demonstrates the upper-body suit when a human is wearing it.

After the human user puts on the motion capture suite and connects it through the Hub to the Axis Neuron, the user can choose or customize the body dimensions (see "Body Size Manager" window in Fig. 6.3) and calibrate the suit by following the instructions in the Axis Neuron manual [W24]. Fig. 6.3 shows the interface of the Axis Neuron in single arm mode. The Axis Neuron not only receives and processes the motion data but also exports the motion data to some third-party software by BVH (Biovision hierarchical data) and FBX (Filmbox) formats. In our experiments, Axis Neuron broadcasts BVH data, which includes the Euler angle and the displacement of each link, through the Transmission Control Protocol (TCP). The illustration of data broadcast is shown in Fig. 6.4.

6.2.2 Data Broadcast Through ROS

Consequently, the next crucial question is how to read the BVH data in the Windows virtual machine then publish it to the Ubuntu host computer. We tackle this cross-system connectivity

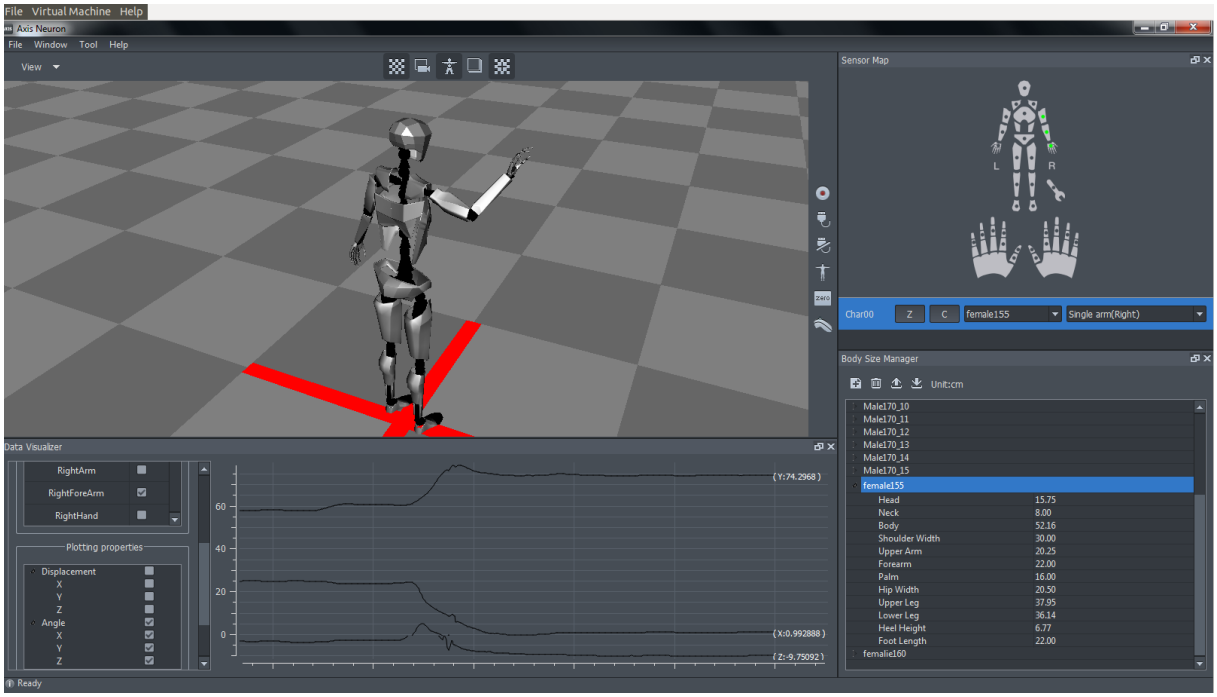


Figure 6.3 – Interface of the Axis Neuron software in single-arm mode. The main "View" window shows a humanoid robot that performs the same movements as the human. The "Sensor Map" window illustrates the position of activated sensors. The "Data Visualizer" window plots the real-time XYZ angles of the right forearm. The "Body Size Manager" window saves the customized body measurements of the demonstrators.

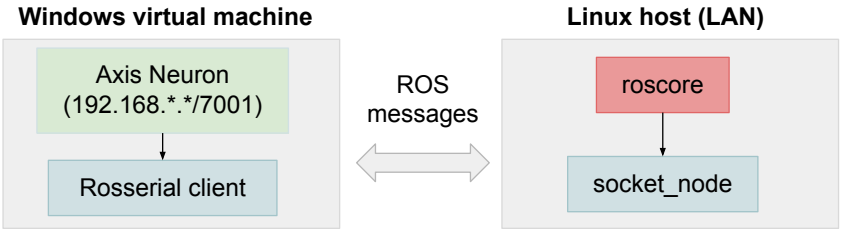


Figure 6.4 – Data broadcast from Axis Neuron to the Linux host machine.

issue using the rosserial ROS package [W4]. Rosserial is a ROS communication protocol that transmits standard ROS messages based on ROS topics or services among different platforms over a serial interface. The roserial_server package provides the host-side rosserial connection in C++ and deals with publishing and subscribing for a connected rosserial-based client [W30]. The client can run on different platforms, such as Windows, various Arduino boards.

On the Linux host, a roscore executable and the socket node from the rosserial package, which provides an interface to multiple TCP clients, need to be run. A rosserial-enabled client, which reads the PN BVH data and publishes the motion data as ROS messages, is executed after installing basic ROS libraries on the Windows virtual machine. To correctly run the client, it is essential to configure the correct broadcast port of the Axis Neuron and the IP address of the ROS master. The publish rate of the motion messages is set to 50 Hz. Accordingly, the motion data is published on the Linux host and can be used for controlling the robot arm. In spite of not

wearing the whole-body suit, the 6D pose of other unmeasured bodies parts is broadcast with constant values determined by the body dimension of the demonstrator. The detailed codes and command lines are presented in [W13].

6.2.3 Arm Tracking

After setting up the hardware and software configuration of the PN system, the Euler angles of the arm joints are obtained. One final calibration process involves registering the pose movements of the human arm with the robot system. This is accomplished by matching the human spine frame to the torso link of the PR2 robot, as shown in Fig. 6.5. The spine link is defined in the PN software, and the robot torso link is set in the robot Unified Robot Description Format (URDF) file. We assume the transformation ${}^{spine}T_h$ of the right human palm with respect to the human spine link is the same as the transformation ${}^{torso}T_r$ of the wrist of the robot arm in relation to the torso link of the robot. As the lengths of the human arm and the robot arm are different, we update ${}^{torso}T_r$ by the link length of the robot upper arm and forearm. Given the transformation ${}^{torso}T_r$, we compute the joint angles of the robot arm J^{ik} (including two wrist joints of Shadow hand) by feeding this pose to the bio-ik solver. After this, we set the angular velocity $\mathcal{V}_t$ of each joint by calculating and scaling the feedforward joint difference between the desired joint angles of the current frame J_t^{ik} and of the previous frame J_{t-1}^{ik} and the feedback joint difference between the desired joint angle of the current frame J_t^{ik} and of the current robot joint state J_t^{robot}.

$$\mathcal{V}_{n,t} = \delta_1 \cdot (J_{n,t}^{ik} - J_{n,t-1}^{ik}) + \delta_2 \cdot (J_{n,t}^{ik} - J_{n,t}^{robot}), \tag{6.1}$$

where n is the n-th joint of the arm, δ_1, δ_2 account for the scaling factors of each velocity term.

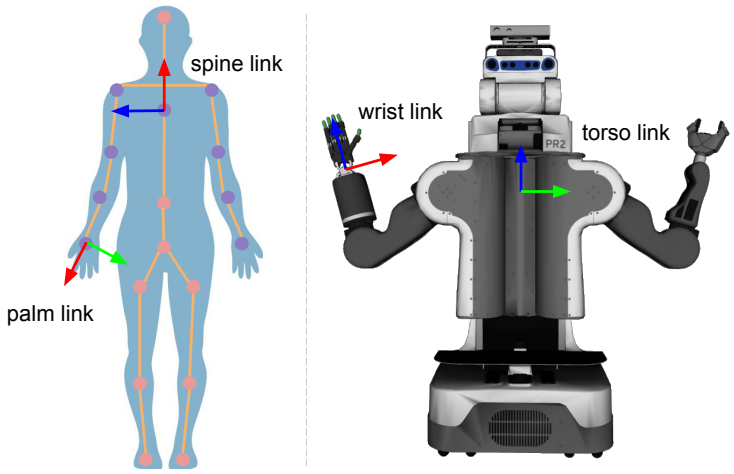


Figure 6.5 – Illustration of the registration between the human right hand and the PR2 right robot arm. The human spine frame is set to match the torso link of the PR2 robot, and the frame of the right human palm is registered to the frame of the wrist of the robot arm. The dots on the human body diagram represent all possible body parts that Perception Neuron can track. The purple ones are where the Neurons have been activated in this setup, and the pink ones are not.

6.3 Camera Holder

The expected dexterous hand-arm system requires that the teleoperator works in an unlimited visual workspace. Paradoxically, the field of view of the depth camera is usually restricted. The depth sensor used in our experiments is the Intel RealSense SR300, whose horizontal and vertical depth fields of view is 73° and 59°, respectively. The operating range is from 0.3 m to 2 m. Compared to the Intel RealSense F200 camera used in chapter 4, the SR300 camera is more advanced and supports shorter exposure time, and allows dynamic motion up to $2\,m/s$ based on a newer 640×480@30 Hz VGA (video graphics array) depth mode. If the operators implement teleoperation tasks requiring a wide workspace, the human hand will disappear from the camera's view range. We solve this problem with a cheap 3D-printed camera holder (see Fig. 6.6), which can be mounted on the forearm of the teleoperator by inelastic polyester straps. The installation of this camera holder is shown in all experimental figures in the following section 6.4. The camera holder's weight and length are 248 g, and 35 cm. During the teleportation experiments, the camera will move along with the forearm. The other side benefit is that the camera holder facilitates an almost optimal viewpoint to capture human hands.

Figure 6.6 – The camera holder is used to mount the camera on the human arm.

6.4 Manipulation Experiments

The multimodal teleoperation system described above was systematically evaluated across four types of physical tasks that analyze precision and power grasps, prehensile and non-prehensile manipulation, and dual-arm handover tasks. For the control of the arm, we set $\delta_1 = 0.7, \delta_2 = 0.1$. The frequency of the arm's velocity control is 30 Hz. The starting poses of the human arm were always consistent with the starting pose of the robot arm. Meanwhile, the arms of the robot always started and ended at almost similar poses over every task. The frequency of the hand's trajectory control is set to 30 Hz. One female and two male testers have participated in the following robotic experiments, and each task was randomly performed by one of them. All objects used in the experiments are not modeled into the planning scene, so there is no collision avoidance between these objects and the robot.

1) Pick and place. For this task, we prepared two testing scenarios: pick a Pringles can and place it in a red bowl on the same table; pick a cube on the table and place it on top of a brick on a box. The height of the box is 225 mm. The first scenario requires the power grasp skills of the robot, and the second scenario needs the precision grasp skills of the robot and a wide enough workspace for the teleoperator.

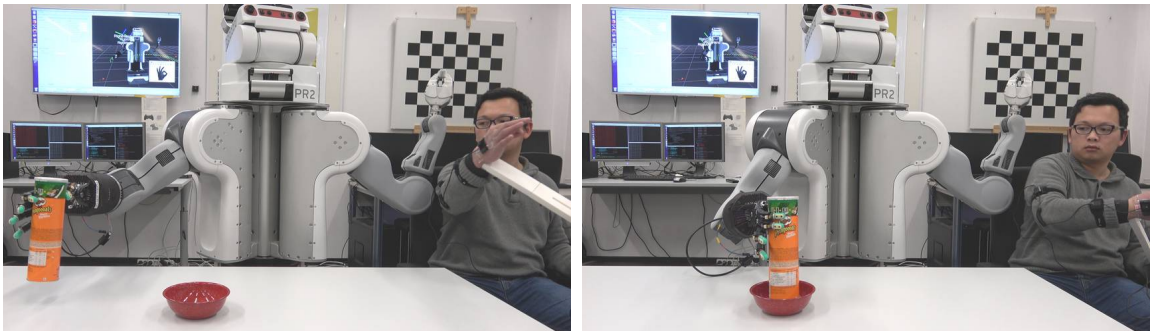


Figure 6.7 – The PR2 robot picks a Pringles can and places it in a bowl. The Pringles and the bowl are set on the same table.

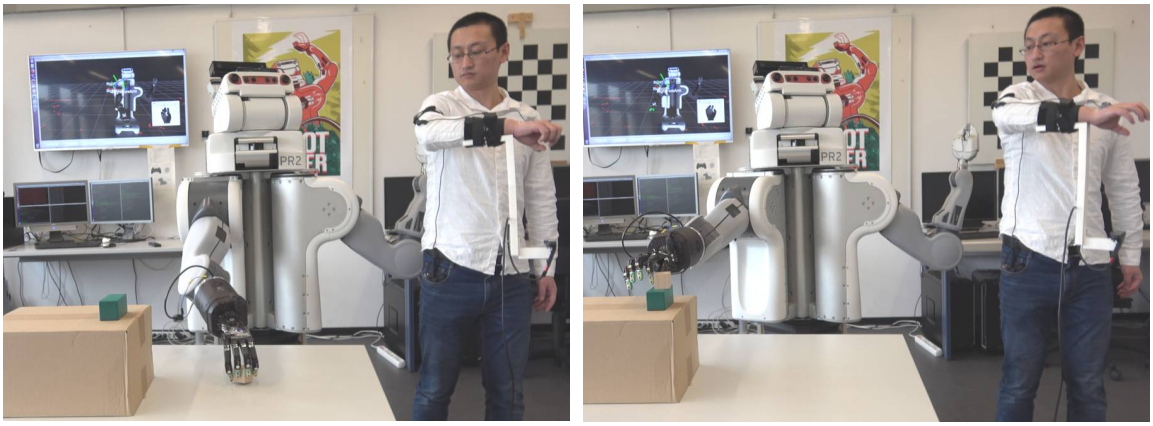


Figure 6.8 – The PR2 robot picks a wooden cube on the table and places it on a rectangular brick on a box. The height of the box is 225 mm.

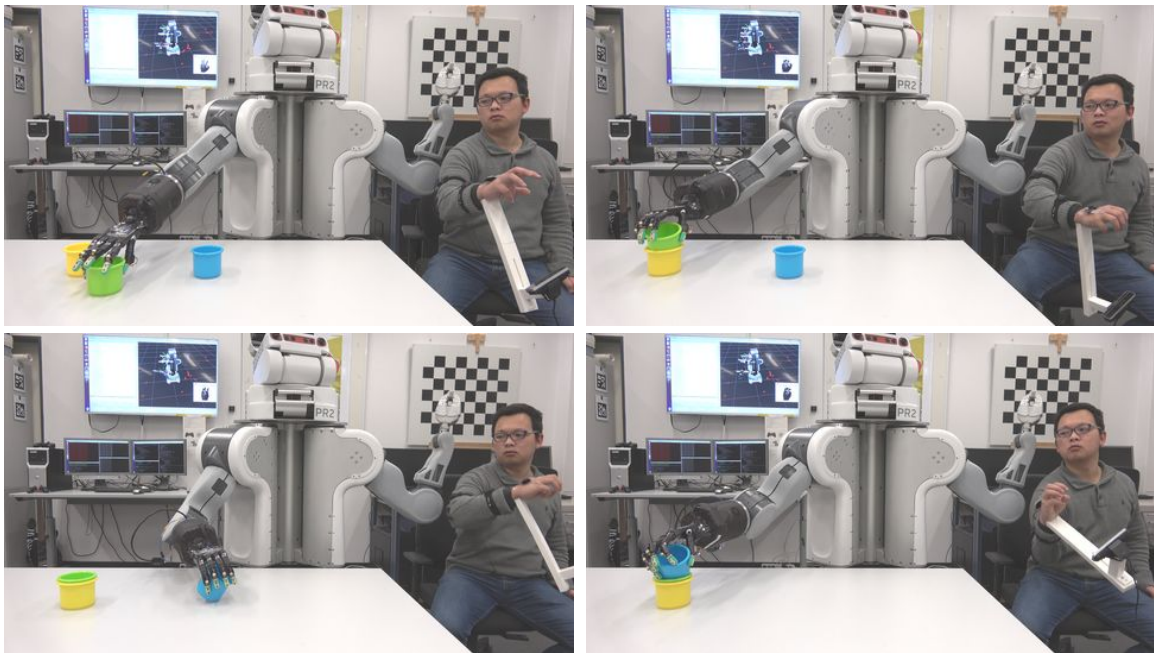


Figure 6.9 – The PR2 robot inserts three cups into each other.

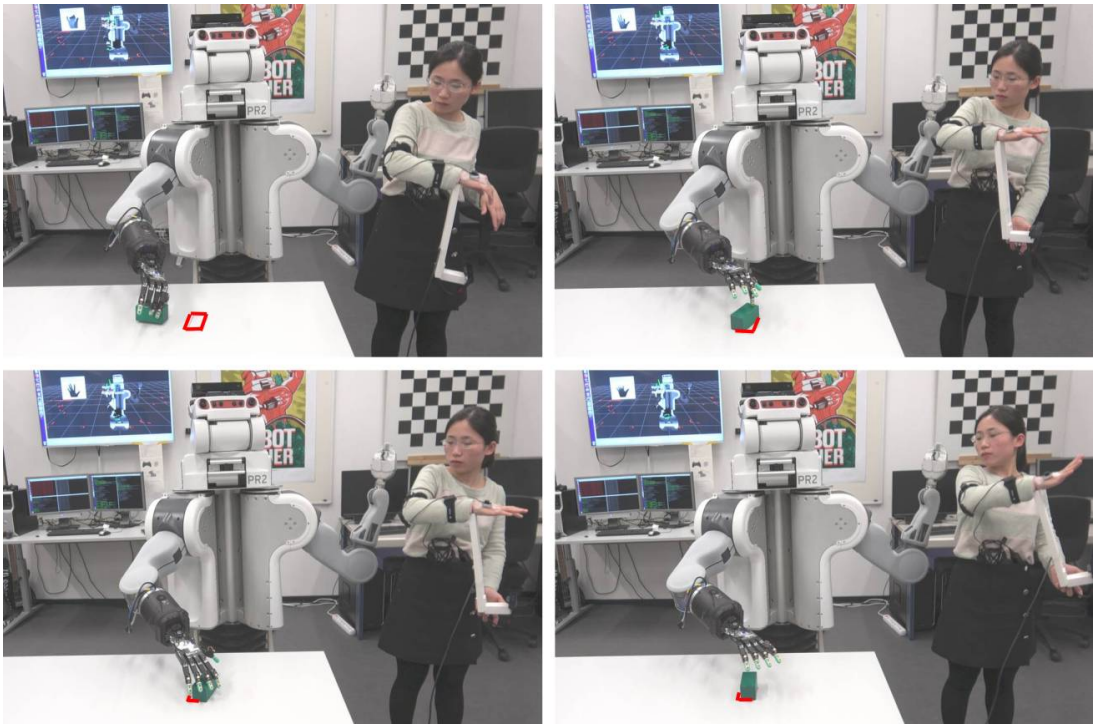


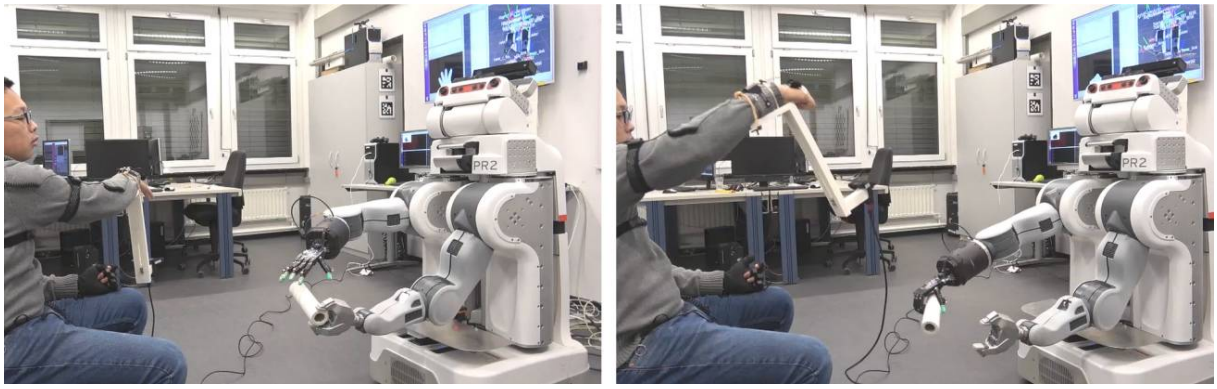
Figure 6.10 – The robot pushes a rectangular brick to a specified pose. The red rectangular on the table represents the target pose of the brick.

2) Cup Insertion. Three concentric cups are to be inserted into each other in order of size from smallest to largest. The cups are randomly put on the table. This task examines the abilities of precision grasp and release.

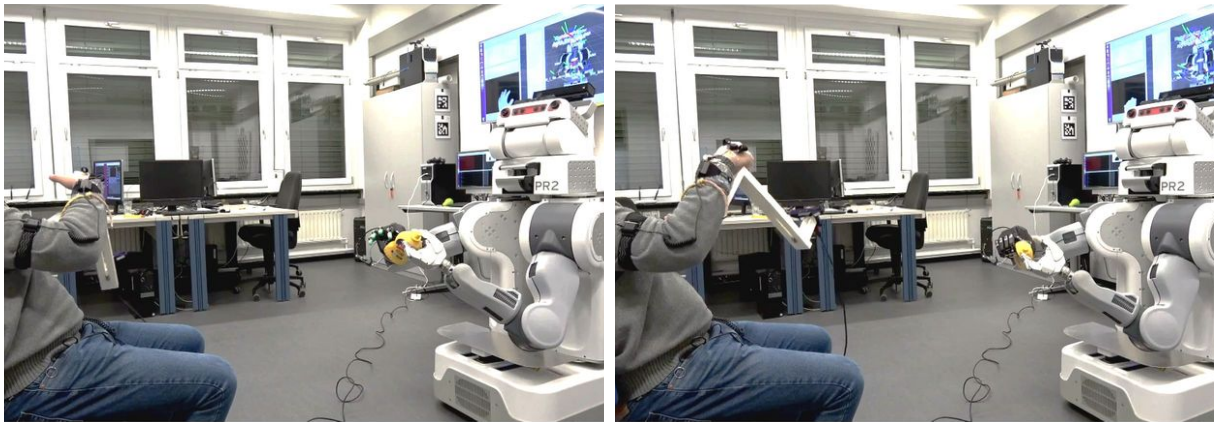
3) Object pushing. We set a random initial pose of a brick then push the brick into a designated position. The goal pose is labeled as a red rectangle. The size of the rectangle is as same as the size of the brick. This task examines the challenges of pushing, sliding, and precision grasping. During the task, a lots of slight adjustment of the brick pose were conducted when the brick was close to the goal area.

4) Dual-arm handover. The left robot arm hands a roll of paper or a mustard bottle over to its right hand. The operator also exploits the PN setup to control the left arm and left gripper of the PR2. This task tests the coordination ability of the whole teleoperation system. Owing to our system's mobility, the human can sit face to face instead of parallel to the robot to better visualize. Fig. 6.11 qualitatively demonstrates this experimental scenario. Notably, in the handover task, the teleoperators needed five complete warm-up trials to adapt to the opposite operating direction of the robot.

Similar to [63], the operators performed a warm-up training phase for each task with five non-consecutive attempts before the real testing trials. For more manageable tasks such as pick and place, the operators could complete the task well after three trials. However, for the handover task, the teleoperator took more trials to adapt to the opposite operating direction of the arm. Each task was conducted five times by one demonstrator. Fig. 6.7 to Fig. 6.11 qualitatively demonstrate the experimental results of our hand-arm system.



(a) The PR2 robot hands over a roll of paper from its left gripper to its right hand.



(b) The PR2 robot hands over a mustard bottle from its left gripper to its right hand.

Figure 6.11 – Screenshots of handover tasks.

	pick1	pick2	cup	pushing	handover
Ave. time (s)	18.5	37.2	25.5	62.0	36.33
Ave. success rate	100%	100%	100%	80%	60%

Table 6.1 – Average completion time and success rate of each task

Table 6.1 numerically shows the average completion time a teleoperator took to finish a task, and the success rate. The completion time was calculated when the robot started to move until it went back to the starting pose. The high success rate and short completion time for the two pick and place tasks, and for the cup insertion task indicate that our system has the ability of precision and power grasps. Compared to the two pick and place tasks shown in Figs. 6.7 and 6.8, the cube is smaller than the Pringles can, and the brick is much smaller than the bowl. Therefore, the robot needed a longer time to precisely grasp the cube and find a correct place to land the cube. During the pushing task, the robot could quickly push the brick close to the target position using multiple fingers. Nevertheless, the operators took a long time to deal with the orientation of the brick in order to make the pushing error lower than 5 mm. The handover task achieved a relatively low success rate, mainly because of the imprecise control of the left gripper, so the robot accidentally lost the object during the handover interaction. These results reflect the fact that the visual-based method is more suitable for multi-finger control than the IMU-based method.

6.5 Discussion

The chapter presents a multimodal hand-arm teleoperation system that consists of the novel vision-based hand pose regression network (Transteleop) and an IMU-based arm tracking method. Regarding the arm tracking, the velocity commands of each arm joint are generated by calculating and scaling the feedforward joint difference and the feedback joint difference (Research Question Q2). Moreover, the human-robot arm calibration was achieved by matching the human spine frame to the torso link of the PR2 robot (Research Question Q2). Therefore, the human movements on the floor do not matter for the arm control because only the relative transformation from the human spine to the right wrist was considered. Thanks to a self-designed camera holder, the integration between the hand and the arm is achieved, making this whole system mobile and unrestricted to the field of view of the depth camera (Research Question Q3). In theory, the human can stand or move anywhere inside of the sensing area of the PN system. Finally, the demonstration of the teleoperation system across three trained human demonstrators on four tasks, i.e., pick and place, cup inserting, pushing a brick, and dual-arm object handover, was successfully performed (Research Question Q4). These robotic experiments verify that Transteleop is an efficient and feasible vision-based teleoperation algorithm, and the IMU-based arm control was reliably implemented on the PR2 arms.

Although our method performs well in real-world tasks, it still has some limitations. The Neuron sensors in the PN system should keep away from magnetic field environments, such as metal tables, computers, or any electrical machines. Since the humans stand near the robot, the IMU data are prone to drift over time, so the demonstrators have to recalibrate the wearing suit from time to time. It would be better to find another arm teleoperation method that is insensitive to the working environment and efficient to use. The 248 g and 35 cm 3D-printed camera holder and the 108 g depth sensor are worn on the human's forearm, bringing additional physical burden for the teleoperator. Since the inelastic polyester straps became a bit loose after long time experiments, sometimes the small rotation of the human forearm could not drive the camera holder to move along. Moreover, the rotation of the human forearm is not consistent with the rotation of the hand palm. Therefore, the camera cannot capture the human hand at the best viewpoint all the time, resulting in hand images with severe self-occlusion. However, the self-occlusion of the fingers is one of the primary reasons causing inaccuracy issues of the vision-based pose estimation algorithms, especially when a single and fixed camera provides the visual data. We hypothesize that a controlled active system would potentially replace the camera holder and solve the extra burden and inaccuracy problem. The controlled active system can be set up so that a camera is mounted on the end-effector of a manipulator, and this manipulator automatically tracks the human hand. One of the primary concerns of vision-based teleoperation is the absence of haptic feedback because the control loop is only supervised by the user. However, the haptic devices are usually expensive and have to be worn on the human hand. As a low-cost alternative to haptic feedback, the visual channels could be implemented to visualize the magnitude of pressure or force on each fingertip of the Shadow hand [128].

Chapter 7

Hand-Arm Teleoperation System Based on Active Vision

Last chapter presents a multimodal hand-arm teleoperation system combined with a vision-based hand estimation model and an IMU-based arm control method. However, the inaccuracy issues due to the self-occlusion of the fingers have not been tackled. On average, the mean errors of the state-of-the-art hand pose estimation algorithms by a single camera are less than 10 mm, but only when the angle between the camera direction and the human hand is less than 30° [171]. The experiment in section 5.4.4 also reveals that it is beneficial to keep the operating range of the human hand in the $[0^\circ, 30^\circ]$ viewpoint range. A well-calibrated multi-camera system is commonly used to perceive rich information, but it cannot avoid extreme viewpoints for the target objects and always encounters issues like time synchronization and long processing time [141]. Another option is to ensure that the camera always captures the human hand from the best viewpoint and at an optimal distance, capitalizing on the more straightforward pose estimation owing to the unobstructed fingers. The aim of active vision systems is to manipulate the viewpoints of the camera and gain the best observation [3]. To thoroughly solve the limited viewpoint issues in vision-based teleoperation, developing a controlled active vision system at the local site would be beneficial.

Therefore, this chapter aims to develop a novel vision-based hand-arm teleoperation system that captures the human hands from the best viewpoint and from a suitable distance. This teleoperation system consists of the end-to-end hand pose regression network Transteleop and a controlled active vision system. Section 7.1 investigates the background of active vision in robotics and how to implement an active system into a hand-arm teleoperation application. Section 7.2 discusses three potential hardware choices that can be used for hand tracking. Later, section 7.3 covers the hardware preparation and software communication in the proposed teleoperation system. Section 7.4 and section 7.5 then explain the control strategies for the robot arms. A precision analysis is given in the following section 7.6.1. Last but not least, a variety of complex manipulation tasks, i.e., pick and place, tower building, pouring, sweeping, and moving or sliding the fader of a MIDI mixer are demonstrated in section 7.6.2 to show the practicality and stability of the proposed teleoperation system.

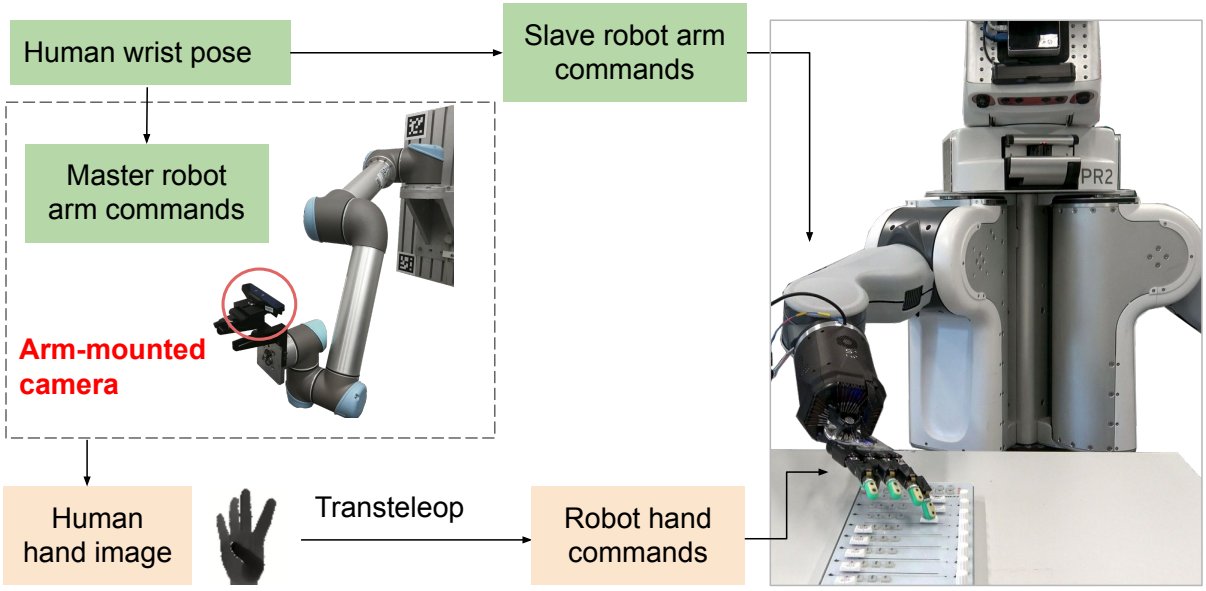


Figure 7.1 – The pipeline of the proposed active-vision-based hand-arm teleoperation system. The human demonstrator teleoperates the slave hand by the end-to-end hand pose estimation network Transteleop and controls the slave robot arm by the relative trajectory control based on the operator's wrist motions. To solve inaccuracy issues of the hand pose estimation caused by self-occlusion of the fingers, we introduce the controlled active vision system to obtain the best hand information. The active vision system consists of a depth camera mounted on a robot arm and a real-time trajectory generation method. This teleoperation system enables the slave robot to finish different types of manipulation tasks such as pouring, sweeping, and sliding a fader.

7.1 Active Vision for Teleoperation

Active vision has been widely used in object tracking [169], robotic grasping [2], human-robot interaction [34], and simultaneous localization and mapping (SLAM) [33]. It aims to select an optimal image viewpoint by moving the vision sensor to an optimal pose for facilitating the associated applications. In common active vision systems, the vision sensor is either mounted on the end effector of a manipulator as a hand-eye system or as a Pan-Tilt robot. Calli et al. [20] utilized the curvature information from the silhouette of unknown objects to update the robot pose using active vision for obtaining exemplary grasping configurations. Recently, some work has optimized the camera viewpoint based on reinforcement learning techniques for grasping pose generation and robotic pushing tasks [19, 29]. In the human-robot interaction scenario, active vision usually strengthens the robot's ability to detect the humans' presence and interpret their motion or emotions [16]. Latif et al. [84] proposed the eye-gaze tracking interface TeleGaze to teleoperate mobile robots based on the visual information from the two Pan-Tilt-Zoom cameras on the robots. To improve the operational performance and increase the immersive feeling, Huang et al. [68] established an active vision system based on a Pan-Tilt-Zoom video camera to track the target in a space robot teleoperation task automatically. Instead of applying active vision to observe a remote site, we investigate how to build a controlled active vision setup at the operator site to capture the human hand from favorable viewpoints.

In this chapter, we devise a markerless vision-based hand-arm teleoperation system in conjunction with the hand pose estimation method Transteleop and a real-time active vision system (see Fig. 7.1). To coordinate active perception and track the human hand from optimal viewpoints, we set up an active vision system where a depth camera is mounted on the end-effector of a robot arm. We will prove the reliability and practicality of the proposed teleoperation system by network evaluation, trajectory analysis, and non-trivial robot experiments, including pick and place, tower building, pouring, sweeping, and MIDI mixer fader sliding, across two trained demonstrators.

7.2 Design Choices for Hand Tracking

The goal for our active vision system is to locate where the human hand is, in other words, the 6D global pose estimation of the human hand. A precise 6D global hand pose estimation algorithm would be ideal, but most hand pose estimation methods only discover the 3D position of the human wrist without orientation. Leap Motion Controller [W42] is a known hand tracking device that can estimate joints and reconstruct the skeletal model of two hands at the same time. Its field of view is $140° \times 120°$, which is larger than typical depth cameras. Nevertheless, when some joints are obstructed, Leap Motion generates inaccurate pose estimation results. Alternatively, using some state-of-the-art motion tracking hardware, e.g., marker-based motion tracking system, could be a better choice. In this section, three motion tracking choices have been tested are discussed and depicted in Fig. 7.2.

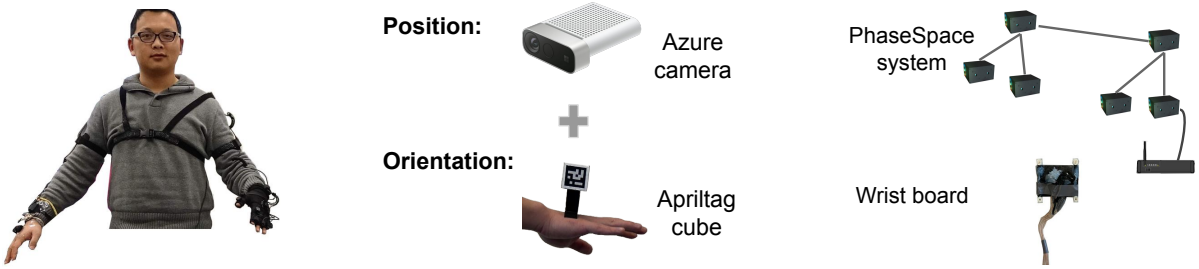


Figure 7.2 – Potential hardware choices of arm tracking. (a) The motion data from the PN system is heavily influenced by the magnetic environment caused by the robot. (b) Apriltag detection fails in real-time tracking for moving objects. (c) The PhaseSpace tracking system with a wrist board could be a better choice.

In chapter 6, the PN motion tracking system and the sophisticated human-robot registration show excellent performance for robot arm teleoperation. Even if existing body estimation algorithms can obtain the pose of the human spine, it is impossible to get accurate registration between the human arm and the robot arm. Additionally, in the active vision scenario, the human stands close to the master robot arm all the time so that the magnetic environment would heavily interfere with the IMU motion data.

The Azure Kinect camera [W22] provides a human body tracking SDK (Software Development Kit), which outputs 6D poses of 32 joints of the human body. Despite this body tracking

algorithm only retaining high accuracy on the joint position estimation, the orientation of the spine joint is empirically reliable. Thereby, a possible solution is to utilize the human body tracking SDK with the cheap visual fiducial system AprilTag [118], whose off-the-shelf detection algorithm computes the accurate 3D position and the orientation of the tags with respect to the camera. The Apriltag can be made in the shape of a cube consisting of a tag bundle composed of six different markers on each face. Then, the Azure Kinect camera estimates the pose of the right human hand based on the AprilTag cube fixed at the back of that human hand. However, we found that the Apriltag detection only works well on static tags during the testing. Once the hand is moving, the detection will be interrupted intermittently probably due to the motion blur.

Alternatively, a marker-based motion tracking system can be a smart choice to pursue reliable and real-time pose detection. The PhaseSpace Impulse X2E motion tracking system [W29], which is unrivaled in speed, precision, and flexibility, provides up to 960 Hz motion data based on several linear-detector-based cameras. A marker on the palm is not as easily obscured as a customized glove with multiple markers when the human hand performs specific gestures. And a marker on the back of the hand cannot affect the depth images of the fingers.

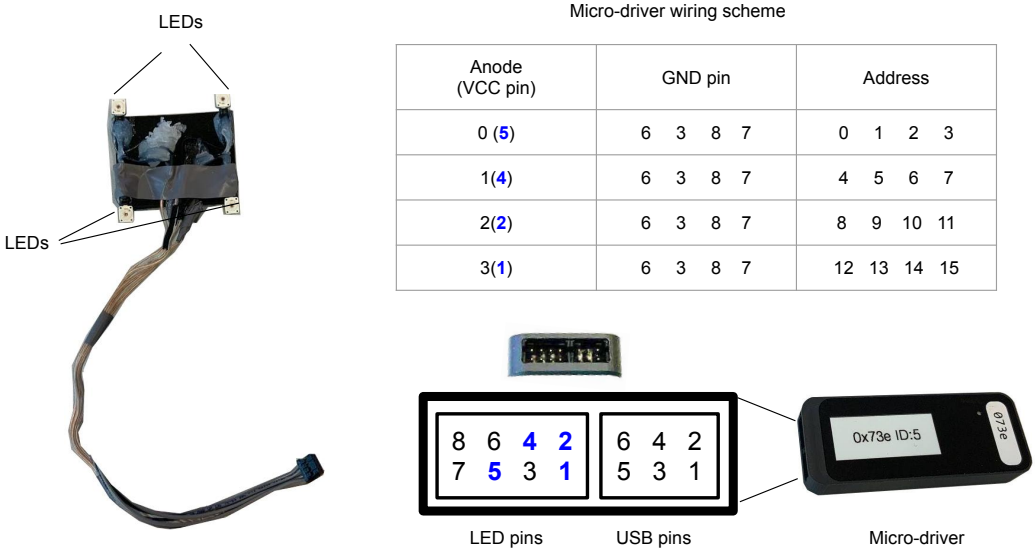


Figure 7.3 – Wring scheme of the customized wrist marker. The left photo shows the wrist marker. The blue and bold numbers are the anodes pins. The combination of one anode and one ground pin determines the address of one LED, which will be recorded in the session profile.

With the discussion above, we take the hardware combination of the PhaseSpace system and a wrist marker as the hardware support of the arm tracking method. Therefore a customized wrist marker with four red LED lights was designed. The shape of the upper layer on this wrist marker is 6 cm×4 cm rectangular, and the layer close to the hand back is slightly arched. Four red LED lights are glued to four corners of the marker. The marker ID, marker name, all LED IDs, positions of four red LED lights were configured and recorded in a YAML file. Meanwhile, the inherent object origin was implicitly defined. To let the PhaseSpace system track the wrist marker, the marker should connect to one micro-driver, which will be activated when the PhaseSpace system is on. Based on the wiring scheme of the micro-driven in shown in

Fig 7.3, we built the wiring connection of the marker. A session profile, which was saved in the PhaseSpace server, has to be created to describe the ID of the associated micro-driver, anodes, addresses and IDs of each LED, and marker name. After all the configuration and hardware preparation, the connected LEDs will flash in different patterns that the cameras can differentiate. Then, this marker's position can be calculated based on all recognized LEDs.

7.3 System Infrastructure

Our goal is to build an agile vision-based teleoperation system in which the teleoperator performs natural finger motions and unrestricted arm actions for a series of manipulation tasks that can be performed in an unlimited workspace. The overall hardware setup is shown in Fig. 7.4.

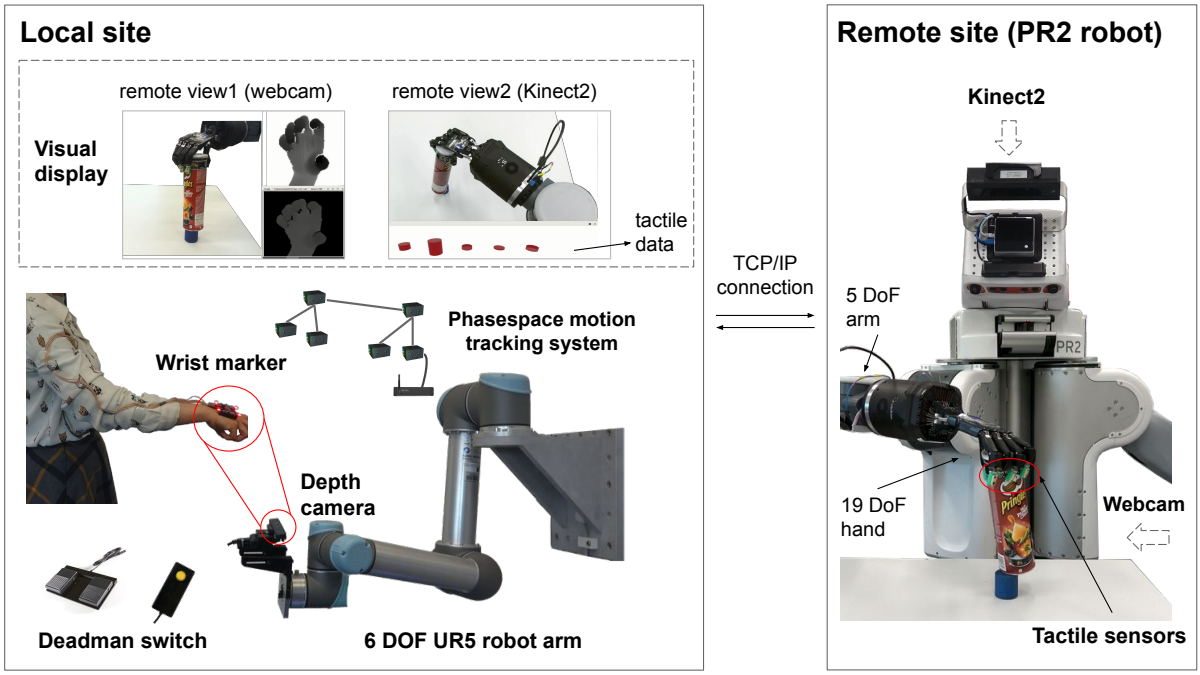


Figure 7.4 – Overall hardware setup of the proposed hand-arm teleoperation system based on hand pose estimation and active vision.

The local site setup (also see Fig. 7.5) consists of a 6-DoF UR5 collaborative robot arm with a RealSense SR300 depth sensor, a PhaseSpace motion tracking system, a 3D-printed lightweight LED wrist marker, two monitors, and two deadman switches. The human teleoperator stands in front of the UR5 robot and keeps a safe distance, while the UR5 robot arm, which possesses a certified safety system, is used to track the human's right hand autonomously. At the remote site, the slave robot is a PR2 robot with a 19-DoF Shadow robot hand mounted on its 5-DoF right arm. A Kinect2 RGBD camera mounted on the PR2 head, and a webcam located to the right of the PR2 observe the robot's motions from the top and side viewpoints. In our setup, the human stands in front of the UR5 robot and can only see the PR2 from the visual displays (see the working environment illustration in Fig. 7.6).

For wrist tracking, the PhaseSpace motion tracking system (320 Hz) estimates the right human hand's 6D pose based on the wrist marker pasted to the back of the human hand. The PhaseSpace

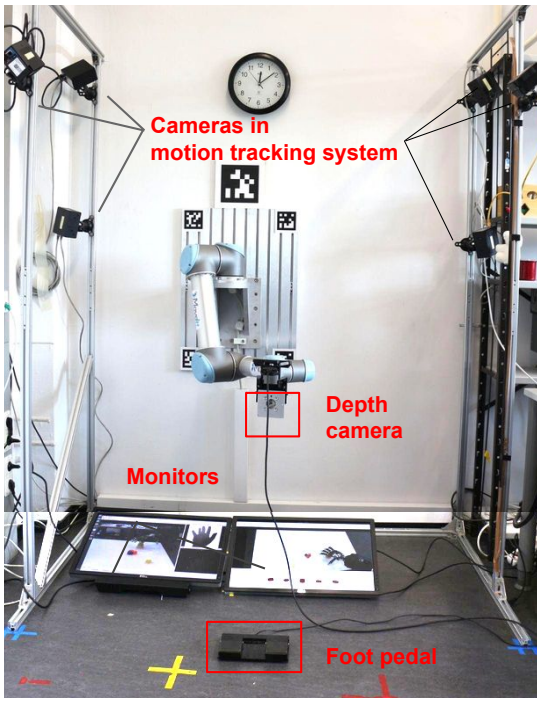


Figure 7.5 – Front view of the hardware at the local site.

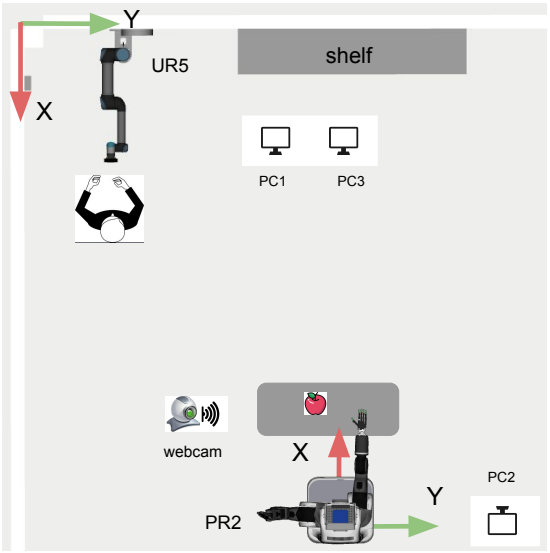


Figure 7.6 – Illustration of teleoperation working environment from the top view.

motion tracking system is registered to the robot coordinate system through a well-designed registration board with predistributed LEDs lights and a printed Apriltag, as shown in the center of Fig. 7.7. The external camera pose T_c is determined by the Apriltag bundles near the UR5 base T_r. Then the pose of Apriltag on the registration board and the pose of the board T_b are estimated by the Apriltag detection and the PhaseSpace system, respectively. The pose ${}^rT_p$ of the PhaseSpace system with respect to the robot base can be calculated by:

$$ {}^rT_p = {}^r\ T_c\ {}^cT_a\ {}^aT_b\ {}^bT_p, \tag{7.1}$$

where ${}^aT_b$ is the fixed transfromation from the Apriltag coordinate O_a to the registration board origin O_b. In terms of the calibration between the linear cameras in the PhaseSpace system and the pose ${}^rT_p$, the in-the-shelf MasterClinet software and a calibration wand were used. More implementation details can be find from [W28].

For finger tracking, the SR300 depth sensor is mounted on the end-effector of the robot arm to capture depth images of the human hand. There is a self-designed camera holder connecting the depth sensor and the UR5 end-effector (see Fig. 7.8). The holder is designed also to be suitable for the case where the end-effector is equipped with a gripper. The pose of the SR300 camera is calculated based on the camera holder's model and the rough relative position between the holder and the camera. We do not require to calibrate the camera precisely as long as the camera is guaranteed to capture the human hand all the time. Regarding the feedback, two monitors are used to visualize the real-time status of the remote site and the depth images of the human hand. The visual remote site is captured by the Kinect2 camera and a webcam in the remote side. In addition to the two visual streams of the robot state from the top view and the right view, real-time force feedback from the robot's fingertips is also graphically represented

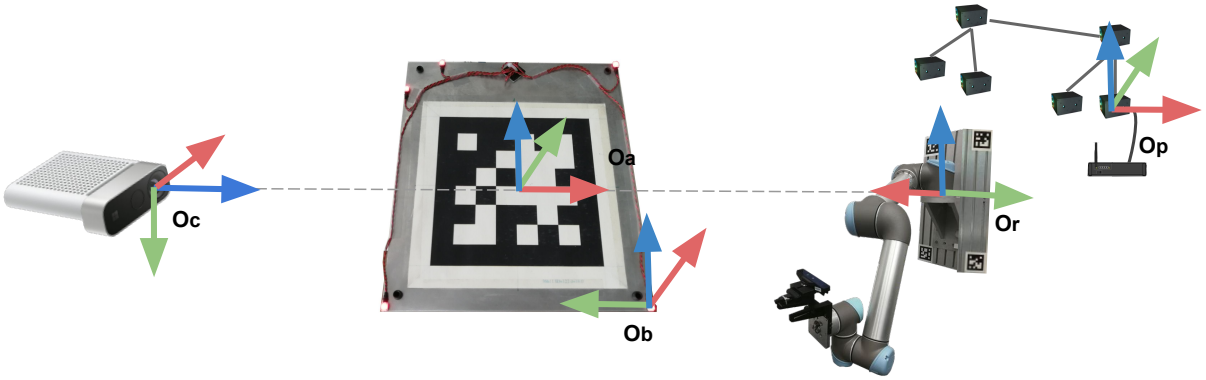


Figure 7.7 – Illustration of how to register the PhaseSpace system to the robot system. The well-designed registration board with predistributed LEDs lights and a printed Apriltag in the middle of this figure connects to the external camera and the PhaseSpace system.

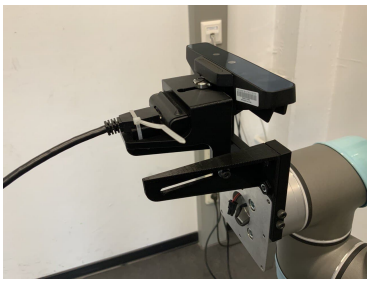


Figure 7.8 – Camera holder in active vision system.

by five cylinders, whose height changes along with the magnitude of the force. The fingertip force reads from the fluid pressure of the BioTac sensor. Regarding the robot emergency control, a footpedal and a deadman switch are used. The left pedal is used to stop the UR5 robot, the right pedal and the switch activate or deactivate the right PR2 arm and the Shadow hand.

This hardware setup works across three computers under the same local area network, and the data is communicated between these computers via ROS. The main ROS topics on each computer and data communication are depicted in Fig. 7.9. PC1 and PC3 belong to the local site, while PC2 is at the remote site. PC1 and PC2 control the UR5 robot and the PR2 robot, respectively. There is one ROS master each running at both sites. PC1 publishes the 6D global hand poses, and PC2 generates real-time trajectory commands for the PR2 right arm based on the global hand poses. Moreover, PC1 generates the Shadow hand's joint commands based on the human hand images, and the Shadow hand imitates human hand gestures at the remote site. PC3 is used for feedback visualization and to control the SR300 depth camera. To discover the running ROS masters in the local network and exchange messages, the master_discovery_fkie [W39] and master_sync_fkie [W40] ROS packages are used in conjunction. The master_discovery_fkie package connects multiple ROS masters by sending messages to a defined multicast group and creating echoes when the different ROS masters respond. The master_sync_fkie package synchronizes the local ROS master to remote ROS masters discovered by the master_discovery node. Those ROS topics, which are used to communicate among different ROS masters, can be specified by a ROS parameter called "sync_topics" when running the master_sync node. It is worth mentioning that these packages require all ROS nodes and services with no duplicate names

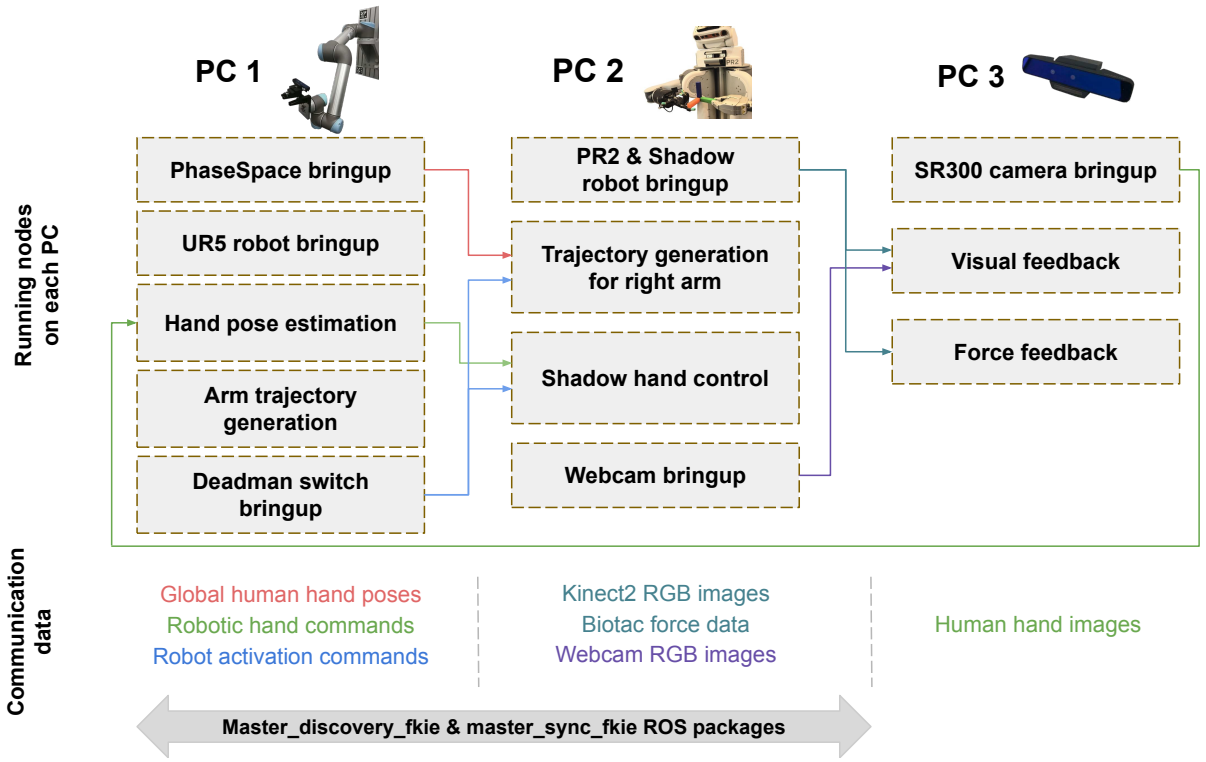


Figure 7.9 – Main ROS nodes on each computer and data communication among three computers. The colored lines represents the data transfer between two nodes. The types of the transferred data are listed in the corresponding colored texts. The two robots are running on two independent ROS masters but communicate via master_discovery_fkie and master_sync_fkie ROS packages.

among all ROS masters. Supported by these two packages, some essential ROS messages, i.e., the 6D human hand pose, the robot hand commands, and the sensing feedback are synchronized on both sides.

7.4 Controlled Active Vision System

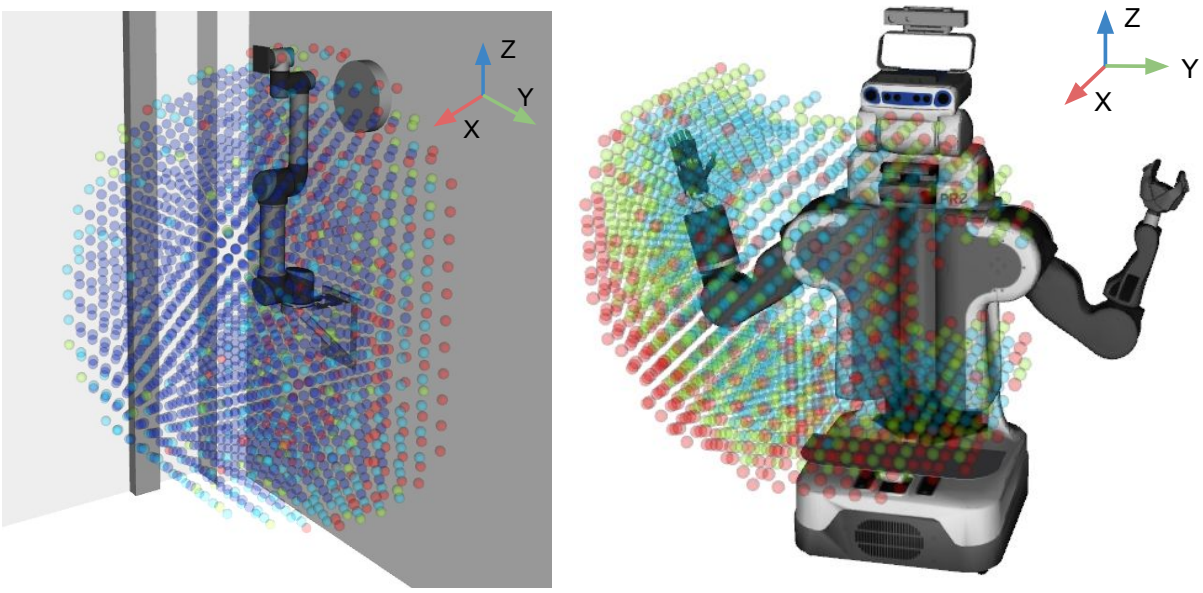
The real-time active vision system allows the camera to capture the right human hand at optimal viewpoints by involving a moving vision sensor, mounted on the end-effector of the robot arm. In such a tracking system, three crucial issues should be considered:

1) whether the robot can smoothly follow the human hand in real-time;

2) whether the robot keeps a safe distance from the human;

3) whether the UR5 robot arm can satisfy the required workspace of the manipulation tasks.

Regarding the first issue, the frequency of the PhaseSpace motion tracking system ensures the fast and reliable identification of the human hand. Then the goal pose of the robot end-effector is 40 cm back along the Y-axis of the hand (see Fig. 3.6), which corresponds to the optimal view distance for the SR300 camera. Secondly, 30 Hz joint-space trajectory generation is achieved by the inverse kinematics solver bio-ik [131]. Except for the pose goal, a regularization goal is



(a) (b)

Figure 7.10 – Visualization of the UR5 workspace and PR2 workspace in our setup from the third-person viewpoint. The blue, green, yellow, and red spheres indicate that the robot end-effector can reach that position with more than 50, 20, 10, and equal to 1 orientation(-s).

additionally considered to keep the joint-space solutions close to the current robot state. To avoid redundant solutions, the Z-axes of the forearm link and the second wrist link are required to point away from the inverse directions of their initial state. Then the real-time 6D poses of the end-effector are translated online into joint-space robot commands, which are required to be as close as possible to the current robot configuration. In Cartesian space, the translation and angular motions are constrained by velocity and acceleration limits. Besides that, a maximum velocity constraint in joint space is also employed.

On top of the certified safety system of the UR5, the 40 cm distance between the hand and end-effector and the trajectory constraints also provide a strong safety guarantee. To avoid rapid motions caused by instability of the PhaseSpace system or accidental human errors, the position movements more than 50 cm along any axes or the orientation change over 60° around any axes between two frames are omitted. Besides, a collision object whose volume covers the area of the human hand is added into the planning scene and updates its pose in real-time. During the experiments, we will check whether the target poses are in the collision area before every execution. Moreover, the human can immediately press the deadman's switch (left foot pedal) to stop the UR5 robot.

To figure out the overall workspace of the system, we constructed a reachability map of the UR5 and the right arm of the PR2 by creating grid-poses in the environment, and calculating valid IK solutions for the poses. In our setup, the UR5 robot is mounted on a wall near a corner and the PR2 is standing in an unconstrained space. The blue, green, yellow, and red spheres in Fig. 7.10 represent that the robot end-effector can reach that position with more than 50, 20, 10, and equal to 1 orientation(-s). Note that the UR5 arm can reach a large part of its overall

workspace in many different orientations (see blue spheres in Fig. 7.10(a)). Compared to the UR5 workspace, apparently, only a few blue spheres scatter in the PR2 workspace due to mechanic limitations. To utilize most of the PR2 workspace, relative control is implemented for the right PR2 arm and absolute control for the UR5 robot. Therefore, the right PR2 arm only performs the incremental motion of the human arm after the demonstrator presses the right foot pedal, and the UR5 robot will track the human arm motion online. In this way, the human demonstrator can always move arms into a comfortable motion range. At the end, we summarize the overall usage of the PhaseSpace system in Fig. 7.11.

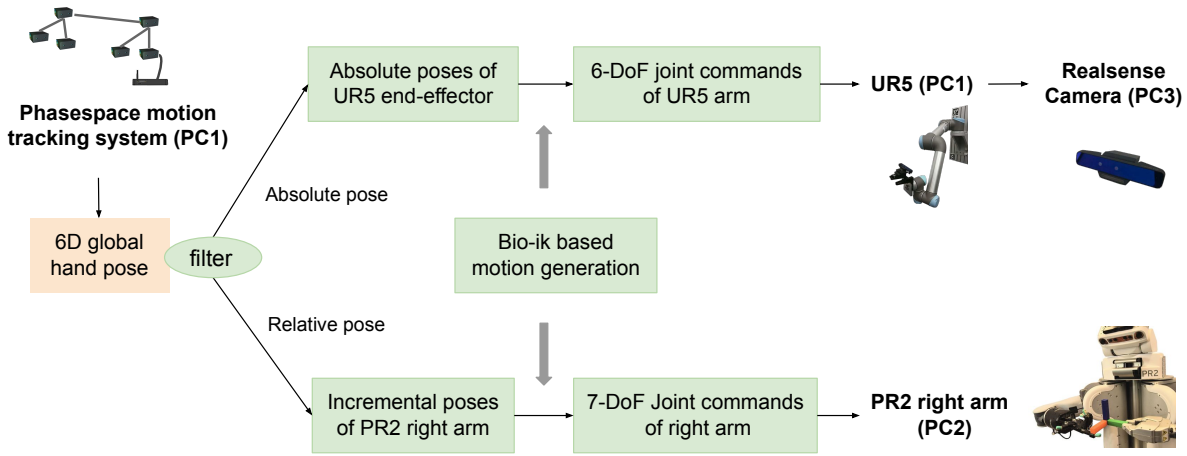


Figure 7.11 – Overall usage of the PhaseSpace system. PC* in the brackets means which PC this hardware connects to.

7.5 Slave Robot Motion Generation

As discussed in section 7.4, in view of a fine coordination between two robots and a limited workspace of the PR2 robot, we employ relative control for the slave robot. Therefore, an initial registration of the human wrist pose with the slave wrist is not necessary, and the local site does not constrain the workspace of the slave robot. The slave robot only moves when the human presses the foot pedal. This foot pedal secures the robot and allows the potential adjustments at the local site, e.g., possible self-collision of the UR5 robot, close to the workspace boundary of the UR5 robot.

Similar to the motion of the UR5, given the human wrist pose acquired through the Phasespace system, plus the regularization goal, the bio-ik solver computes the joint angles of the slave arm under velocity and acceleration constraints in Cartesian space and the joint space. A minor difference is that the feedforward and feedback joint angle differences are considered to calculate the joint velocity in joint space, similar to section 6.2.3.

7.6 Robot Experiments

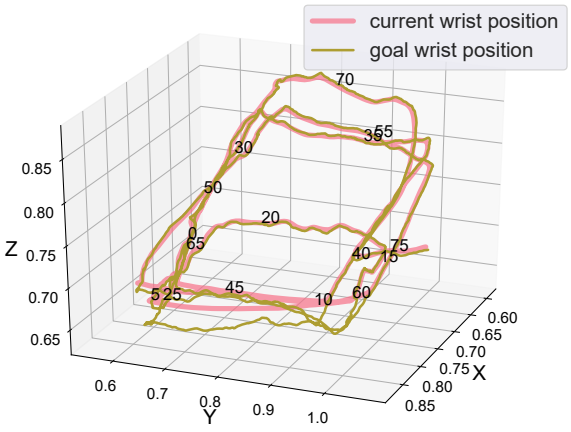
In this section, we rigorously examine the proposed teleoperation system by precision analysis of robot trajectories and five elaborate experiments (pick and place, tower building, pouring,

sweeping, and pushing) that test precision and power grasp, prehensile and non-prehensile manipulation. In Cartesian space, the maximum linear velocity, angular velocity and linear acceleration are 0.2 m/s, 2 rad/s and 2.0 m/s^2 for both robots. The maximum velocity in the joint space of the UR5 is 3 rad/s. The velocity limits of five joints on the right PR2 arm are the default values from the PR2 manual [W5]. The control frequencies of the UR5 robot, the Shadow hand, and the slave arm are all 30 Hz.

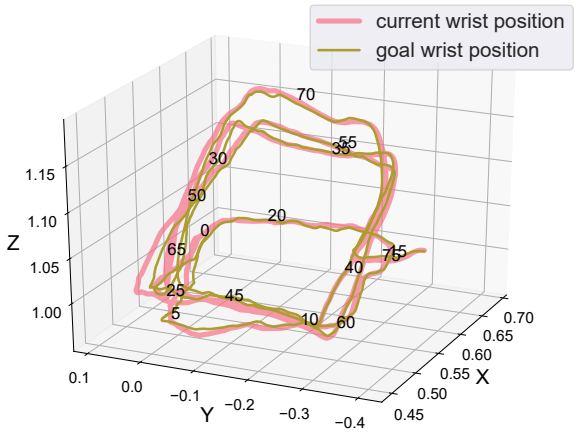
7.6.1 Precision Analysis

To check if the robot could track human hands in real-time, it is essential to evaluate the precision of this teleoperation system quantitatively. We recorded the end-effector trajectories of both the UR5 and the PR2 while the right human hand performed specific motions. Based on the same movements of the human arm, the right PR2 arm was tested starting from the center and the side of its workspace and with different weights (0.5, 1) of the regularization goal, respectively. Speaking of which, the weight of the end-effector pose goal is 1.

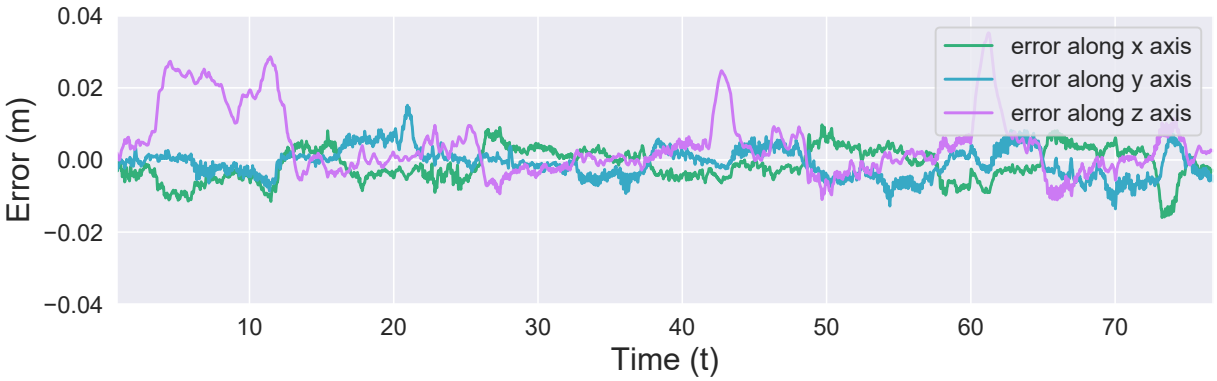
Fig. 7.12(a) presents the example end-effector trajectories of the UR5 while a human moves the hand at the local site. Figs. 7.12(b), 7.13(a), 7.13(b) plot the corresponding end-effector trajectories of the PR2 with different parameters. The starting poses of these three PR2 trajectories are slightly different because the start of recording time is a bit mismatched. The frame coordinate of these trajectories is parallel to the base frame of the UR5. The goal wrist positions in both figures are the smoothed goals after filtering the Cartesian constraints. The plots of current wrist positions are the real robot trajectories. From Fig. 7.12(a) and Fig. 7.12(c), we can see that the UR5 follows the human hand well in most cases. During around 3-12 s and 41-43 s, where the human hand is moving through a sharp corner and the UR5 robot is stretching a bit, the tracking error is up to 3 cm. The probable reasons are that 1) the regularization goal, which tries to keep the joint-space solutions as close as possible to the current robot configuration, in our trajectory generation method and the servo parameter set in the UR5 driver are acting together to smoothen the trajectory. 2) the closer the robot is to its workspace boundary along the X-axis, the greater the trajectory error. Luckily, the existing 3 cm position error does not affect the camera tracking the human hand at all because there is still a 40 cm distance between the camera and the human hand. When the right PR2 arm starts from the center of its workspace with regularization weight 0.5, the PR2 arm conducts the motion commands most precisely as depicted in Fig. 7.12(b) and Fig. 7.12(d). The average L1 tracking error along three axes of the right PR2 arm is 1.8 cm. With these configurations, the right PR2 arm is capable of following human motions and conducting most manipulation tasks, such as pick and place or pouring. When the weight of the regularization goal is equal to the pose goal, the trajectory is smoothed a lot at the sharp corner around 13-19 s, resulting in almost 10 cm error along the X-axis in Figs. 7.13(a), 7.13(c). When the right PR2 arm starts at 30 cm along the negative Y-axis from the workspace center, the robot cannot reach the goals in the top right corner due to mechanical difficulties, as shown in Figs 7.13(b), 7.13(d). Hence, setting the weight of the regularization goal to 0.5 and starting the robot from the center area of its workspace are beneficial.



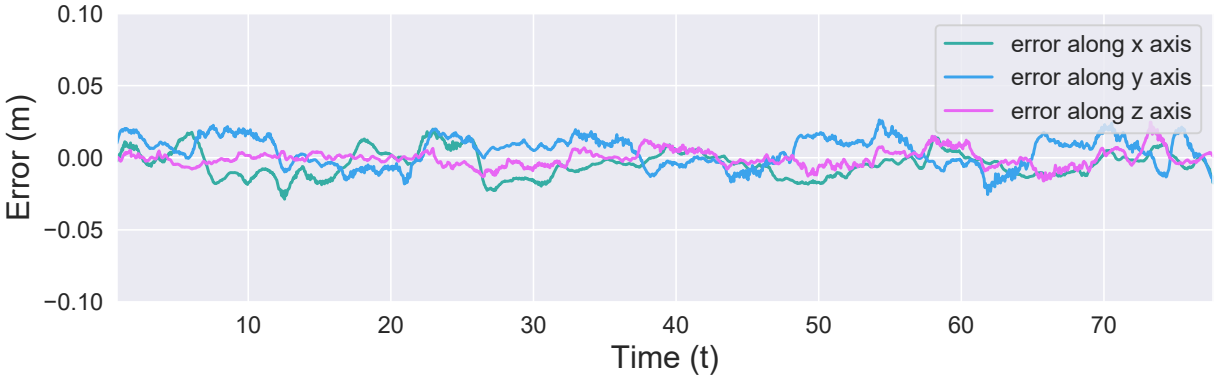
(a) Example end-effector trajectories of the UR5



(b) Example end-effector trajectories of the right PR2 arm (middle start, regularization 0.5)

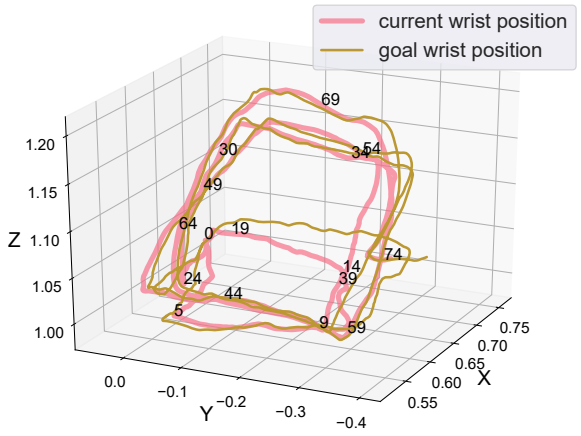


(c) Camera position error along X, Y, Z axes of the UR5 tracking

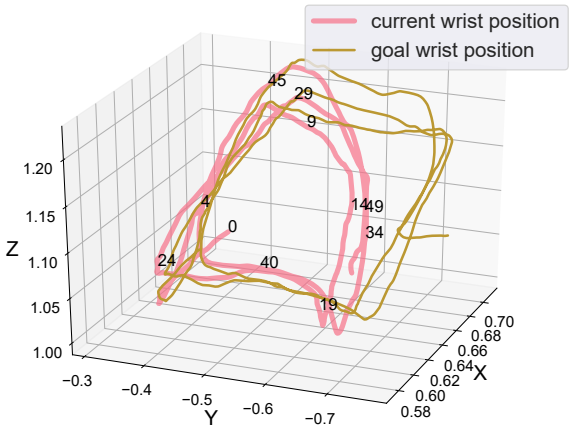


(d) Position error along X, Y, Z axes of the PR2 motion (middle start, regularization 0.5)

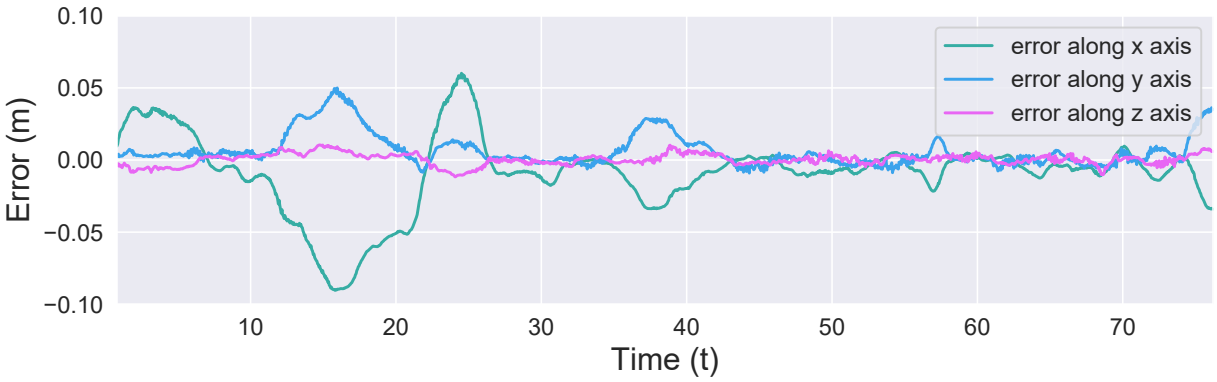
Figure 7.12 – Trajectory analysis of the UR5 and the right PR2 arm. The right PR2 arm starts from the center of its workspace with regularization weight 0.5. The numbers overlapping on the trajectories in (a) and (b) mean the corresponding execution time (s).



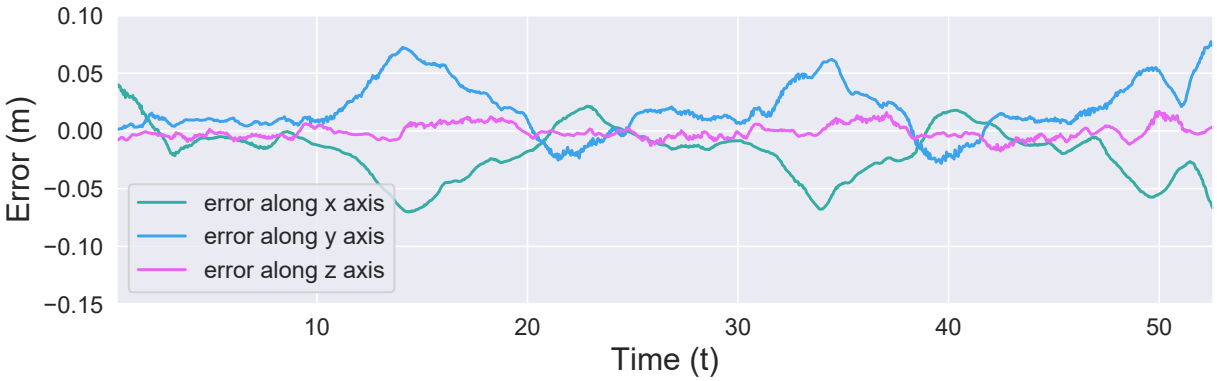
(a) PR2 trajectories (middle start, regularization 1)



(b) PR2 trajectories (side start, regularization 0.5)



(c) Position error along X, Y, Z axes of the PR2 motion (middle start, regularization 1)



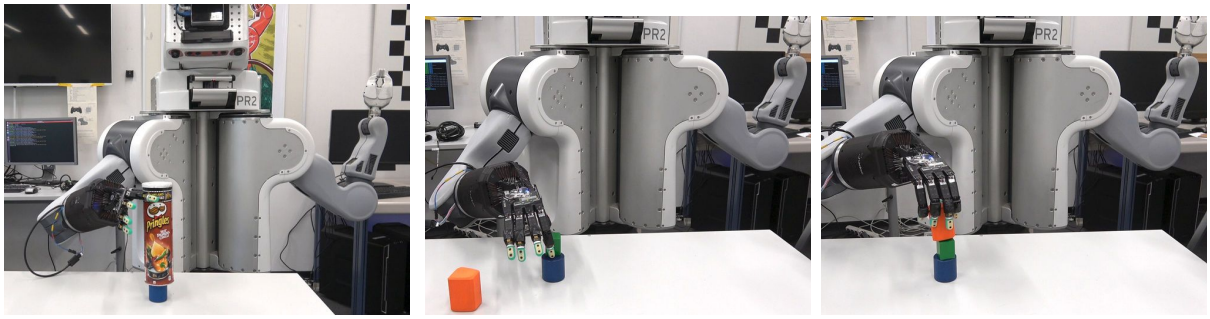
(d) Position error along X, Y, Z axes of the PR2 motion (side start, regularization 0.5)

Figure 7.13 – Trajectory analysis of the right PR2 arm when (a) and (c) the weight of the regularization goal is 1; (c) and (d) starting from the side of its workspace. The numbers overlapping on the trajectories in (a) and (c) mean the corresponding execution time (s).

7.6.2 Manipulation Experiments

To verify the reliability of the teleoperation system, real-world experiments across five types of physical tasks were performed by a female and a male adult. The operators need to get familiar with the active tracking system first and then go through a warm-up phase for each task with ten non-consecutive attempts before the real testing. As a matter of fact, one of the primary concerns of vision-based teleoperation systems is their lack of haptic feedback. Since the visual could be a low-cost alternative to haptic feedback, we visualize the pressure values on each fingertip from the BioTac sensors during the manipulation process as a clue to force feedback.

1) Pick and place. In this experiment, the robot grasped a Pringles can with a radius of 4 cm then placed it on the top of a blue cylinder with a radius of 2.5 cm. We teleoperated the robot to grasp the Pringles can from the top and from the right side by power grasp, as shown in Fig. 7.4 and Fig. 7.14(a). Compared to the pick and place task in Chapter 6, the difficulty level in this chapter is higher because the bottom object has a smaller diameter, thus requiring extremely stable releasing.



(a) (b)

Figure 7.14 – (a) The PR2 robot picks a Pringles can and places it on the top of a blue cylinder. (b) The PR2 robot stacks three different objects on top of each other.

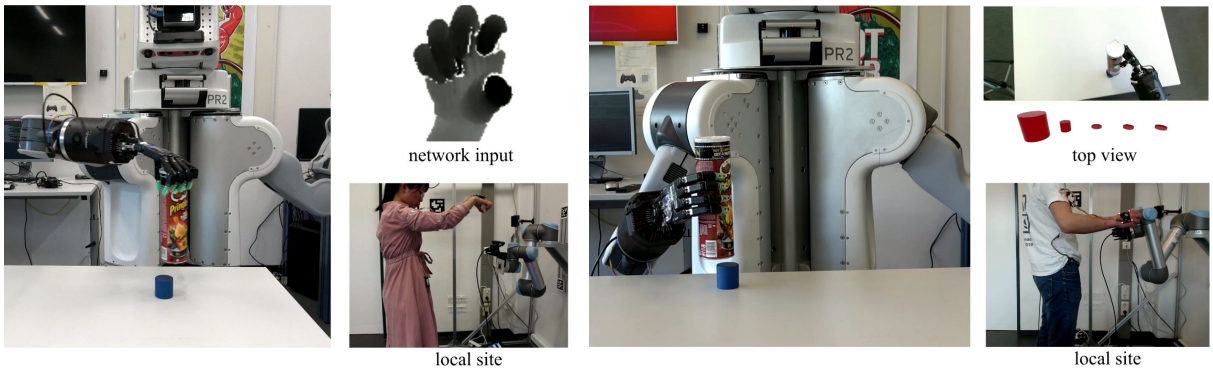


Figure 7.15 – Example scenes with the local site, remote site, network input, and the force feedback in the pick and place task and tower building task.

2) Tower building. This experiment requires the robot to stack three different objects on top of each other, see Fig. 7.14(b). The robot used a precision grasp for the small green block and conducted a power grasp for the irregular ellipsed orange block. Since bigger objects were required to be placed on top of the smaller cylinder in experiments 1 and 2, these two tasks strictly inspected the grasping ability and stable object release. The right image in Fig 7.15 shows the scene with the local site, remote site, and the force feedback in the tower building this task. On the one hand, the force feedback helped the human users to know when the robot was touching the object. On the other hand, the force feedback reduced the risk of robot damage due to excessive force.

3) Pouring. In this task, the robot grasped a cup filled with rice, poured the rice into a bowl, and then placed the empty cup on a box. Fig. 7.16 visualizes the teleoperation process of this task. The human was supposed to turn the right hand 90° clockwise, move along the Y-axis of the UR5 robot, update the hand pose to a grasping pose, then slowly rotate the right wrist to assume a pouring pose. To fulfill this task, the UR5 robot was required to track the human hand simultaneously. Overall, this pouring task mainly examines the stability of the tracking system.

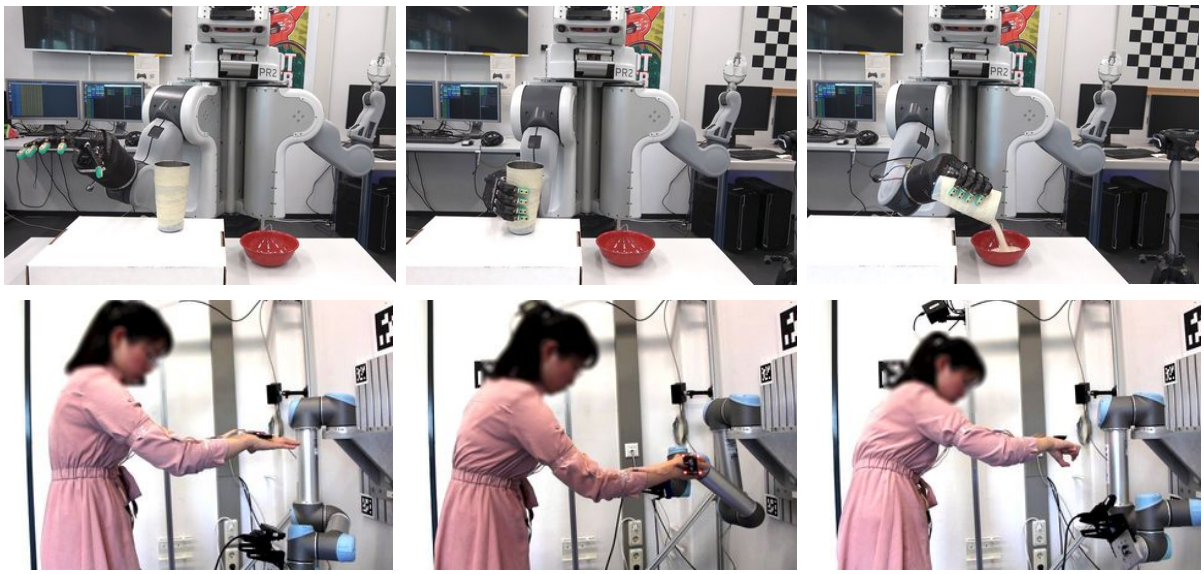


Figure 7.16 – From left to right, three images in the upper row show the PR2 robot grasping a cup filled with rice and pouring the rice into a bowl, and the images in the lower row visualize the real-time human status at the local site.

4) Sweeping. The robot grasped the brush and swept three small blocks to a specific place, then placed the brush on the table. The contact force between the brush and the table surface should be mild. This task contains the challenges of pushing, sliding, and precision grasping (see Fig. 7.17).

5) Sliding a MIDI mixer fader. Figs. 7.1 and 7.18 illustrate the fader sliding task by the first finger of the Shadow hand. The width and length of the fade are 0.4 cm and 1 cm. Fig. 7.19 displays the human status in the local site and the top visual scene of the remote site during the experiments, and the visual haptic feedback of the five robot fingertips. Obviously, in the top scene, the critical experimental area is easily occluded by the robot hand itself. Therefore, the

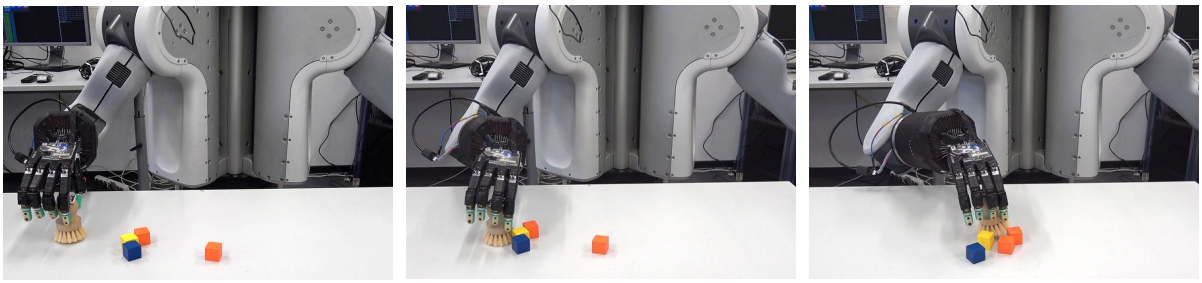


Figure 7.17 – The PR2 robot grasps the brush and sweeps three blocks to the orange block at the right side of the image.

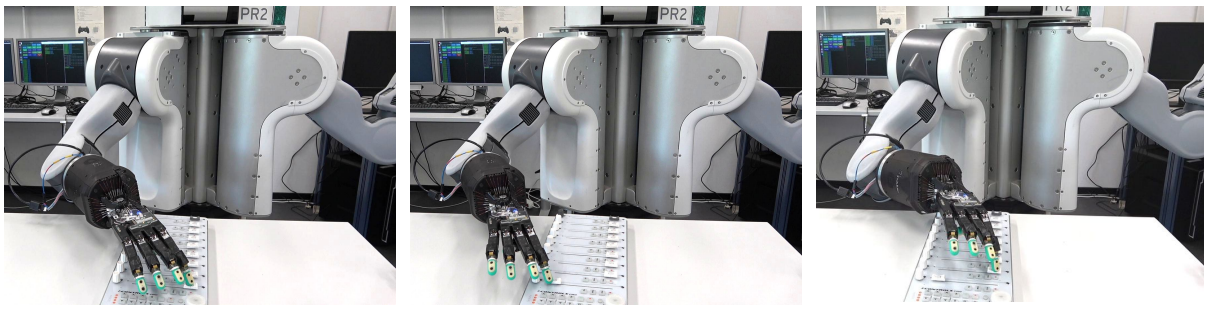
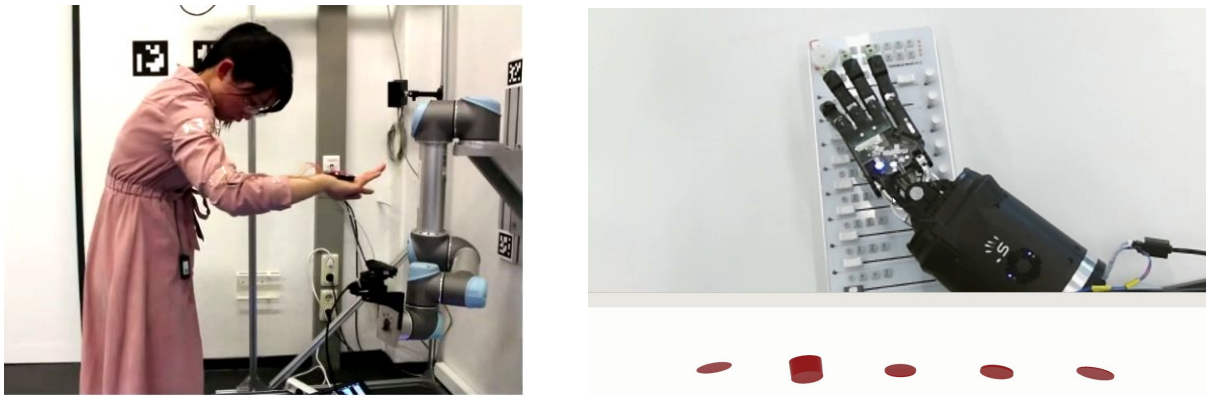


Figure 7.18 – The PR2 robot slides the mixer fader from left to right using its first finger.



(a) (b)

Figure 7.19 – (a) The human status at the local site in the fader sliding task. (b) shows the top scene from the Kinect on the head of the PR2 and the visual haptic feedback. The five red cylinders from left to right qualitatively illustrate the fingertip pressure of the thumb, first finger, middle finger, ring finger, and little finger.

remote manipulation states are heavily dependent on the side webcam and the haptic feedback. Besides that, humans hardly control their hands to move along an exact straight line. Hence, the robot usually cannot slide a fader from left to right in one go. To improve the success rate, we downscaled the human movements by three times.

	pick	tower	pouring	sweeping	fader-1	fader-2	fader-3
Ave. time(s)	52.5	102.8	49.2	75.3	15.5	36.5	85.5
Ave. success rate	90	60	100	100	80	60	10

Table 7.1 – Average completion time and success rate of each task

Table 7.1 numerically shows the average completion time and success rate of all tasks. The high success rates of the pick and place task, pouring task, and sweeping tasks indicate that our system is able to perform precision grasps robustly, power grasps, placing, sliding, and robust tracking. In the tower building task, the human needs to place the objects on a smaller surface with proper force, and the robot could accidentally ruin the tower. Therefore this task took the longest time and achieved a relatively low success rate. For the fader sliding task, fader-* means how many faders the robot continuously slide from left to right. We observe that more fades the robot continuously slid, the lower success rate we got and the longer time the robot required. And the time used to find the next fader occupies half of the completion time in the fader sliding task. The downscaled human finger movements and haptic feedback helped to improve the robot performance, but the limitation of the visual feedback and the accuracy of the current robot system are still insufficient for this fine manipulation task.

7.7 Discussion

This chapter presents a robotic arm teleoperation method developed on vi, integration of hand-arm teleoperation system, and system verification (Research Questions Q2 and Q3). Two trained human demonstrators successfully performed real-world experiments on five tasks, i.e., pick and place, tower building, pouring, sweeping, and fader sliding (Research Question Q4). In conclusion, the robot experiments indicate that the active vision system continuously adjusts the camera pose to guarantee that the human hand can be captured by the optimal point of view. Moreover, the most experimental results verify the excellent efficacy of the proposed dexterous hand-arm teleoperation system. However, during the experiments, even though the users observed the robot status from the simultaneous display of two camera views (top view and side view), one view possibly becomes non-informative when the robot hand partially occluded the object from the top or the side. In this case, visual force feedback is indeed helpful. However, visual force feedback is less efficient than haptic force feedback, especially when interacting with soft objects. After a period of experiments, the users are somehow exhausted to focus on multiple visual displays simultaneously. In summary, the robot-motion feedback is a limitation of the current vision-based teleoperation system. In the future, improving visual feedback via immersive devices, autonomously inspecting remote work situations, and using audio channels as auditory force feedback or workspace boundary alarm will be useful. Furthermore, integrating adaptive force control strategies into the teleoperation system could achieve safer and more compliant operation [177]. Until now, all experiments were finished under the position control, where the position commands from the joint regression models will directly apply to the robot joints. In other words, the task dynamics in all manipulation experiments were neglected. To consider the task dynamics and achieve compliant teleoperation, the next chapter integrates an adaptive control strategy into the current vision-based teleoperation system and demonstrates the compliant teleoperation on several robotic tasks in simulation and real world.

Chapter 8

Evaluation Experiments on Compliant Teleoperation by Adaptive Force Control

We humans can spontaneously adapt our hand pose and force to interact with environments in a compliant manner during daily manipulation tasks. However, the position control strategy used in the previous chapters is not suitable for the complex task of acquiring compliant grasping and manipulation skills of a multi-finger robot. Consequently, if we would like to endow a robot with human-like skills, one promising solution is to combine adaptive control strategies that allow the robot to deal with physical and dynamic interactions with the environment compliantly. Thus, this chapter studies extending the proposed teleoperation models to compliant teleoperation by combining adaptive control strategies and conducts additional evaluation experiments on the Shadow hand.

Firstly, section 8.1 depicts the human-in-the-loop learning-control approach for compliant teleoperation. After a retrospect on grasping and manipulation based on force/torque control in subsection 8.2.1, the chosen adaptive force controller, which is derived from the computation model inspired by the human motor learning principles [167], is presented in subsection 8.2.2. Furthermore, to validate whether compliant teleoperation could yield better performances than teleopertion under the position control, section 8.3 and 8.4 describe the implementation results in both simulation and real-world tasks. Since the chosen adaptive force controller is designed for the robot hand, all experiments are only tested on the Shadow hand. Compared to robot experiments in previous chapters, the in-hand manipulation task of opening-a-cap is firstly tested.

8.1 System Overview

A compliant teleoperation system should allow taking an image as the input and output the desired force commands for the robot hand. The pipeline of the proposed system is shown in Fig. 8.1. The proposed teleoperation system is a learning-control approach combining a vision-based teleoperation system with adaptive force control. For the learning part, the end-to-end model TeachNet developed in chapter 4 is employed to learn the mapping relation between the human hand pose and the joint angles of the Shadow robot hand. It is worth mentioning that both TeachNet and Transteleop are qualified to be used in this pipeline. An adaptive force

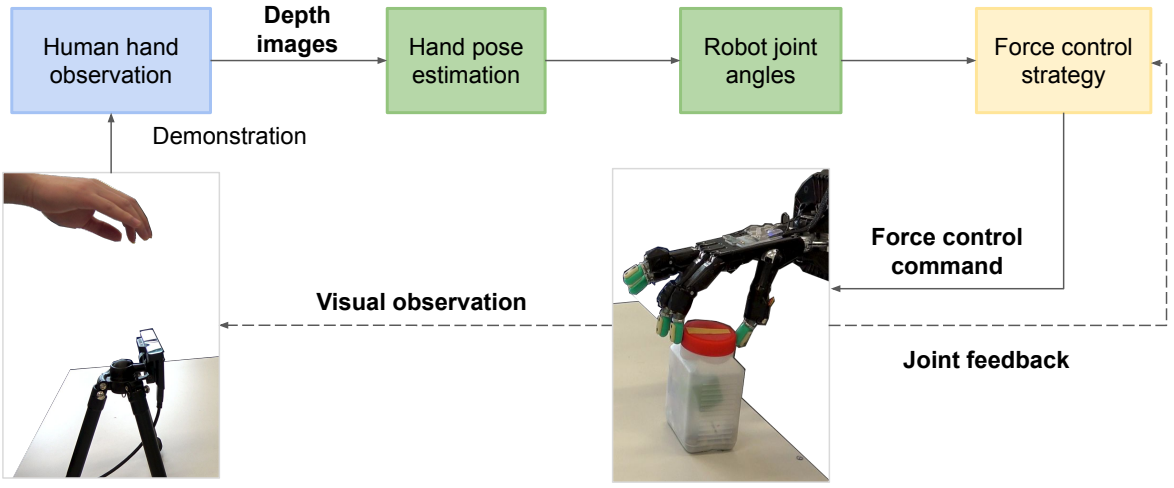


Figure 8.1 – The pipeline of compliant teleoperation by adaptive force control. The human demonstrator guides the robot hand to complete a task through vision-based teleoperation. A camera is used to track the hand pose during the demonstration. The neural network model TeachNet is used to map the human hand pose to the robot hand joint angles. Subsequently, a force control strategy is applied to generate the desired force commands for the robot hand during the demonstration loop.

control strategy that predicts the next-step desired control command based on the desired joint angles and the current robot states is applied to deal with task dynamics during hand grasping and manipulation. The force controller used in this chapter is derived from the computation model inspired by human motor learning principles. The control variables in the controller, i.e., impedance and feedforward terms, are simultaneously adapted online and combined to generate the force/torque commands, which are subsequently sent to the robot hand in the joint space. Even though this force controller does not exploit tactile feedback, the simulation and real-world experiments show that the manipulation stability with the compliant teleoperation is better than TeachNet with the existing position control and the fixed-gain-based force control.

8.2 Adaptive Force Control Strategy

8.2.1 Grasping and Manipulation based on Force/Torque Control

It is a significant goal in robotic manipulation research to augment robots with human-like dexterous and compliant behavior for many tasks in everyday life. In recent years, numerous attempts have been published towards this goal, but some issues have not been fully addressed yet, especially concerning grasping and manipulation with a multi-finger robot hand [49, 183, 132].

An impedance-model-based force controller has been used in robotic manipulator control for some physical interaction tasks [50]. However, its use in controlling multi-fingered robot hands for grasping and manipulation tasks has not been thoroughly investigated yet. Recent studies illustrate that force control strategies increase the grasping stability and robustness [44] and achieve a good grasp stability [144] and in-hand manipulation [87] for the haptic exploration by a multi-finger robotic hand. In [160], an object-level impedance controller has been developed

and shown to be effective and robust in robot grasping. Li et al. [86] improved the controller by dividing the impedance into two parts: one for stable grasping and another for manipulation. Furthermore, the desired impedance is estimated using supervised learning based on the data collected from the human demonstration in advance. Pfanne et al. [123] proposed an object-level impedance controller based on in-hand localization, which improved the ability to avoid contact slippage through adjusting the desired grasp configurations. Garate et al. [53] proposed regulating the control of the grasping impedance (stiffness) by regulating both the robot hand pose and the finger-joint stiffness. By adapting the magnitude and the geometry of the grasp stiffness, the desired stiffness profile could adapt the hand configuration for stable grasping. With the support of the Omega3 haptic device, Michel et al. [106] presented an adaptive impedance control with learned state-varying stiffness for the bilateral teleoperation of contact tasks featuring continuous interaction with the unknown environment by a 7 DoF KUKA light-weighted robot. However, these force controllers may not be suitable for a vision-based teleoperation system, where the controller needs to dynamically and quickly respond to the changes of the human hand pose to predict the desired force commands. Consequently, the contribution of this work is to explore the regulation of the impedance (stiffness) and the feedforward term online during the process of robot grasping or manipulation, which cannot be learned in advance or through exploration.

8.2.2 Methodology

Recently, a biomimetic control strategy inspired by the findings of human motor learning in the muscle space has been developed and proved to be an effective method for compliant robot manipulation. Neuroscience has discovered that humans can simultaneously adapt their arm impedance and feedforward force to minimize motion error and interaction force with external environments under a specific set of constraints [17]. Based on this principle, a biomimetic force controller was first proposed in [167] which allowed the robots to deal with both stable and unstable interactions through the adaptation of the impedance and feedforward term in the force controller. Li et al. [94] further improved this controller and implemented it to deal with several physical interaction tasks such as cutting and drilling by a redundant robot manipulator. However, the biomimetic control strategy has not been utilized for a dexterous robot hand with multiple DoFs. Here, the biomimetic force controller is extended to enable compliant grasping and manipulation from human hand teleoperation.

Controller Formulation

Each finger of the robot hand is to be considered controlled independently using an impedance-based force controller in the joint space [94]. The control input, τ_c, is composed of three components the robot dynamics τ_0, an impedance term u, and a feedforward force v. Therefore, we have

$$\tau_c = \tau_0 - u - v, \tag{8.1}$$

where

$$\tau_0 = M\ddot{q}_e + C\dot{q}_e + G - \Gamma\varepsilon, \tag{8.2}$$

with a symmetric positive-definite matrix Γ with minimal eigenvalue. M, C), and G denote the inertia, the Coriolis and centrifugal forces, and the gravitational force, respectively. $\dot{q}_e$ is an auxiliary variable, $\dot{q}_e = \dot{q}_d - \zeta e$.

The impedance is determined in a PD (Proportional-Derivative) form,

$$u = K_s e + K_d \dot{e}, \tag{8.3}$$

with

$$\begin{cases} K_s = diag\{K_{s,1}, K_{s,2}, \cdots, K_{s,N}\} \\ K_d = diag\{K_{d,1}, K_{d,2}, \cdots, K_{d,N}\} \\ v = \{v_1, v_2, \cdots, v_N\} \end{cases}, \tag{8.4}$$

and

$$e = q - q_d, \;\; \dot{e} = \dot{q} - \dot{q}_d, \tag{8.5}$$

where $K_s \in R^{N\times N}$ and $K_d \in R^{N\times N}$ denote the stiffness and damping matrix, respectively. $e \in R^{N\times 1}$ and $\dot{e} \in R^{N\times 1}$ represent the errors of the joint angles and velocities between the current ($q \in R^{N\times 1}$ and $\dot{q} \in R^{N\times 1}$) and desired ($q_d \in R^{N\times 1}$ and $\dot{q}_d \in R^{N\times 1}$) ones.

Then, all the compliant profiles are parameterized (i.e., K_s, K_d, and v) as [177],

$$K_{s,i} = \theta_{k,i}^T g, \;\; K_{d,i} = \theta_{d,i}^T g, \;\; v_i = \theta_{v,i}^T g, \tag{8.6}$$

where $\theta_k \in R^{N\times N_g}$, $\theta_d \in R^{N\times N_g}$, and $\theta_v \in R^{N\times N_g}$ denote the parameters corresponding to the compliant profiles, i.e., stiffness, damping and feedforward force, respectively. And $g \in R^{N_g\times 1}$ is the Gaussian basis, and is determined by

$$[g]_{n_g} = \frac{\omega_n(s)}{\sum_{n=1}^{N_g} \omega_n(s)}, \tag{8.7}$$

with

$$\omega_n(s) = \exp(-0.5h_n(s - c_n)^2), \tag{8.8}$$

where s is the variable that can be calculated by $\dot{s} = -s$. N_g is the total number of the Gaussian models, and c_n and h_n are the centers and widths of the basis.

Cost Definition

The parameters θ_k, θ_d, and θ_v need to be adapted at each time step based on the desired and current robot states to generate the desired control force. To do so, the tracking error cost and interaction dynamics cost are considered.

First, for the minimization of the tracking error, the following cost which is often used in the robot control domain is defined,

$$L_e = \frac{1}{2}\varepsilon^T M(q)\varepsilon, \tag{8.9}$$

where ε is a sliding error, determined by $\varepsilon = \dot{e} + \zeta e$, and ζ is a positive constant. $M(q) \in R^{N\times N}$ is the inertia matrix of the robot dynamics model.

Then, the following cost to deal with the interaction dynamics is formulated,

$$L_c = \frac{1}{2}\tilde{\Phi}^T \Theta^{-1} \tilde{\Phi}, \tag{8.10}$$

Online generation of the force control command
Input
The learned optimal TeachNet model f_m;
The constant coefficients: Θ_k, Θ_d, Θ_v, and ζ;
The Gaussian basis g.
Begin
Initialize the parameters θ_k, θ_d, and θ_v;
While *online teleoperation* **do**
Sense an image of the human hand pose I_H;
Calculate the desired pose of the robot hand;
Get the robot current states;
Calculate the sliding error ε;
Update parameters θ_k, θ_d and θ_v;
Generate the desired force control commands τ_c
Send the effort commands to the robot joint space.
End
End

Table 8.1 – The force control strategy for robot compliant teleoperation

where

$$\tilde{\Phi} = \Phi - \Phi^* = [\tilde{\theta}_k^T, \tilde{\theta}_d^T, \tilde{\theta}_v^T]^T, \tag{8.11}$$

with

$$\Phi = [\bar{\theta}_k^T, \bar{\theta}_d^T, \bar{\theta}_v^T]^T, \tag{8.12}$$

and

$$\Phi^*(t) = [\bar{\theta}_k^{*T}, \bar{\theta}_d^{*T}, \bar{\theta}_v^{*T}]^T, \tag{8.13}$$

where $\theta_k^*(t)$, $\theta_d^*(t)$, and $\theta_v^*(t)$ denote the desired parameters, and $\bar{(\cdot)}$ denotes the row average vectors of the corresponding parameters. The matrix Θ is determined according to,

$$\Theta = \mathrm{diag}(\Theta_k \otimes \mathrm{I}, \Theta_d \otimes \mathrm{I}, \Theta_v \otimes \mathrm{I}), \tag{8.14}$$

where $\Theta_k \in R^{N\times N}$, $\Theta_d \in R^{N\times N}$, and $\Theta_v \in R^{N\times N}$ are symmetric positive-definite matrices which are manually set in the experiments. I is an identity matrix.

The updating goal is therefore to minimize the overall cost, i.e., $\min \|L_c + L_e\|$. Finally, let the derivative of the cost equal zero to obtain the following updating laws. For the n-th ($n \in [0, \cdots, N]$) DoF, there have

$$\dot{\theta}_{k,n}^T = \Theta_{k,n}\varepsilon_n e_n g, \tag{8.15}$$

$$\dot{\theta}_{d,n}^T = \Theta_{d,n}\varepsilon_n \dot{e}_n g, \tag{8.16}$$

$$\dot{\theta}_{v,n}^T = \Theta_{v,n}\varepsilon_n g. \tag{8.17}$$

Table 8.1 summarizes the procedure of the force control strategy.

8.3 Simulation Experiments

A real-sim setup was established for the simulation experiments. The human demonstrator adjusted the hand pose to guide the simulated robot hand to complete the task under visual feedback during each task. This was done using a virtual Shadow motor hand in the Gazebo simulator with the ODE engine. In our usage, the Shadow robot hand was torque-controlled under the Teach mode. The simulation environment was run on the Ubuntu 18.04 system with a CPU Intel Core i5-8500 and an NVIDIA 1050 Ti GPU. Note that while virtual environments are dominated by physics (e.g., object weights and surface frictions), the absence of force feedback makes the tasks rather challenging [54], as even slight inaccuracies on joint angles from TeachNet may result in failed interactions. The experiment video is available at [W11].

Door opening. This task requires the human demonstrator to guide the robot hand to open a door by pulling the handle in the X-direction. Since the base and the wrist of the Shadow hand were fixed in our simulation environment, this task can only rely on the fingers' interaction with the handle to open the door. The robot can open the door smoothly by teleoperation with the adaptive force control strategy, as depicted in Figs. 8.2 and 8.3. On the other hand, under the position mode, the teleoperator can only occasionally open the door and fails to make the door open as wide as under the force control mode. Fig. 8.3(b) shows the positions of the contacts between the robot hand and the door handle in the X-Y plane. It is observed that the contact points are almost evenly distributed along the X-axis under compliant teleoperation, suggesting a stable interaction between the robot hand and the handle during the execution of the task.

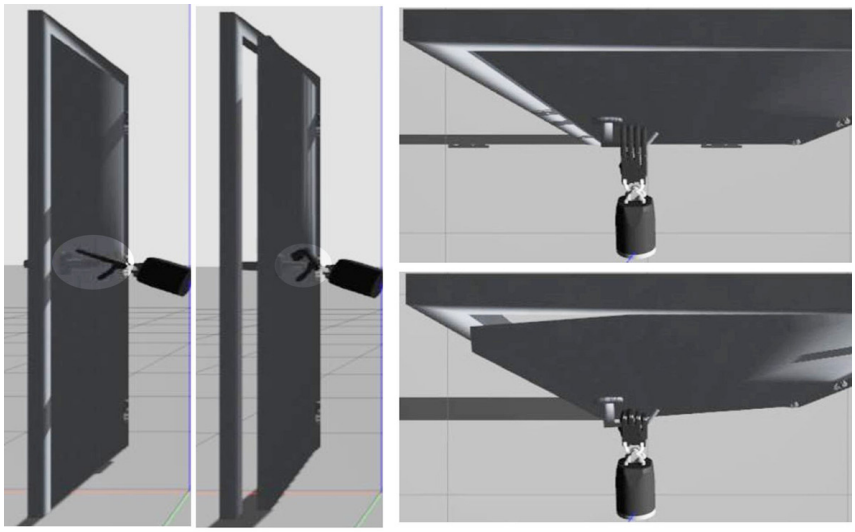


Figure 8.2 – The initial and final configurations during the door opening task from the (a) side and (b) top view, respectively.

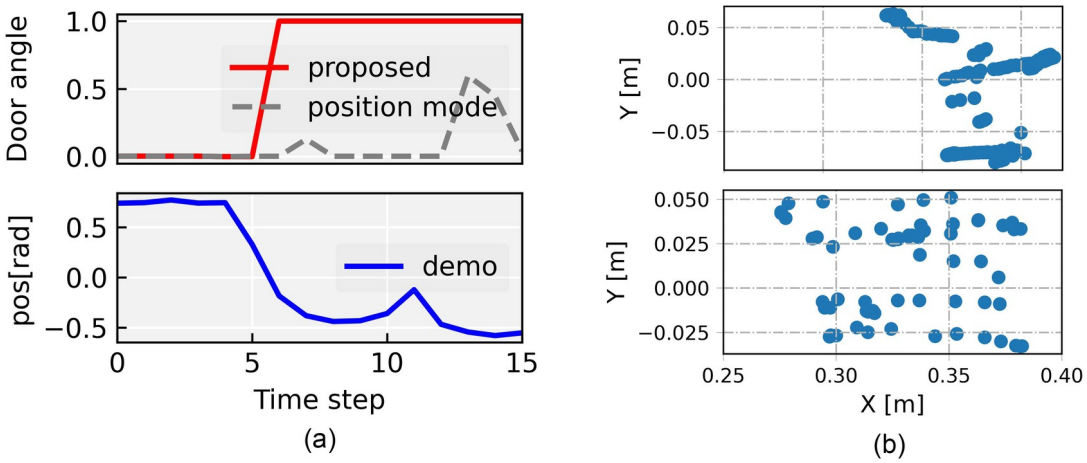


Figure 8.3 – (a) The upper row shows the normalized angle changes of the door with respect to the world frame, the lower row represents the general trend of the changes of the demonstrated joint angles from TeachNet. For the sake of better visualization, the door angles are normalized to [0, 1], and joint angles of the robot are reduced to one dimension using PCA. (b) shows the positions of the contact points under the position (upper row) and force (lower row) control modes.

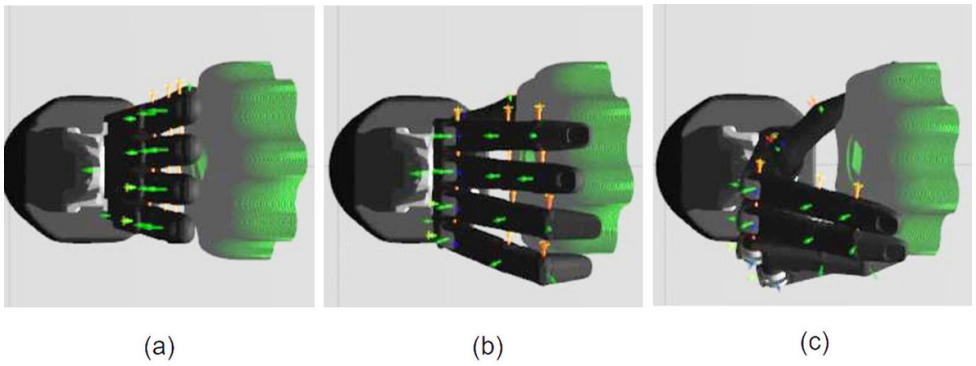


Figure 8.4 – Screenshots of the cap turning task. (a), (b) and (c) denote the initial, middle, and final configurations.

Cap turning. In this task, the robot hand was teleoperated by the human demonstrator to turn a cap using five fingers. The frame of the cap was fixed in Gazebo, and the cap can be rotated in the X-Z plane. When screwing a cap, humans usually adapt the motion of both arm and hand coordinately to complete this task. More importantly, the rotation of the wrist joint plays a vital role during the turning process. In our teleoperation system, however, the fixed base and wrist of the Shadow hand made this task more challenging than usual. The robot hand was guided to make contact with the cap using a proper configuration and then to adapt the movements of all the fingers to turn the cap. The fingers can move coordinately and cooperate well with each other to complete the task using the proposed force control strategy (see Fig. 8.4 as an example). Furthermore, the position distribution of the contact points obtained by the compliant teleoperation is more caplike (see Fig. 8.5).

Mouse touching. To further explore the compliance achieved by the proposed learning-control approach, the performances of a mouse touching experiment were investigated. As shown in Fig. 8.6, the robot hand contacts a curved surface by touching a mouse. The experiment

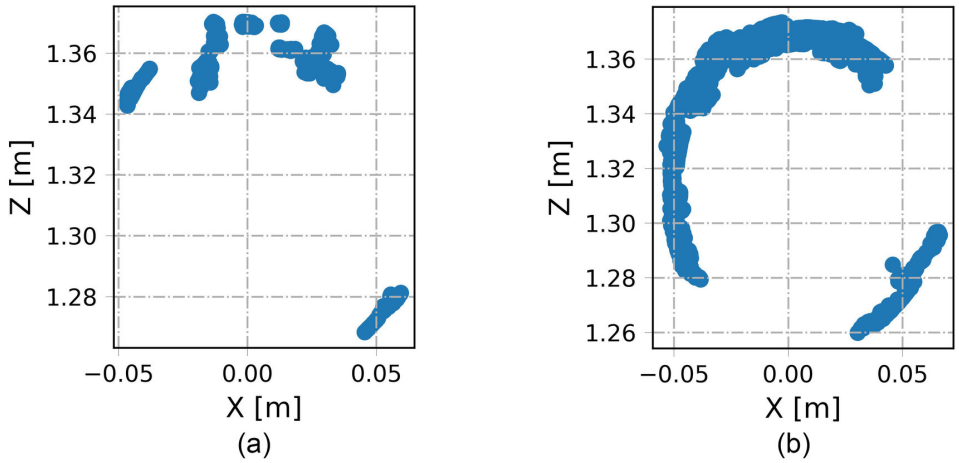


Figure 8.5 – The positions of the contact points in the X-Z plane during the turning process, under the (a) position control and (b) adaptive force control modes, respectively.

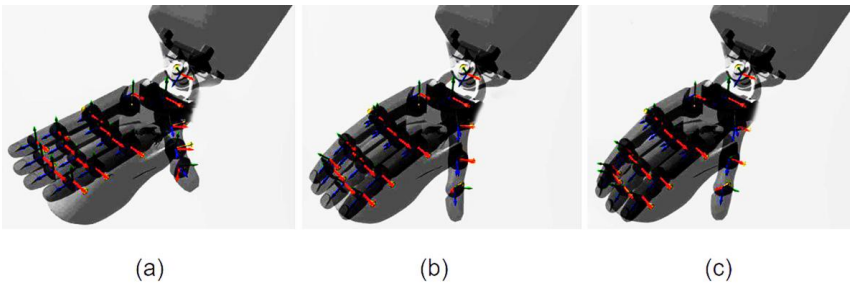


Figure 8.6 – The screenshots of the touching-a-mouse task. (a), (b) and (c) denote the initial, middle and final configurations.

mainly focuses on achieving stable contacts with small contact forces between the hand and the mouse surface. The robot hand is expected to touch the surface of the target object in a more human-like manner. To evaluate the impact of the adaptive control strategy, the task is also conducted under three different control strategies: (a) with the adaptive control; (b) with force control but with a fixed-gain-based impedance controller, which has been often applied in robotics; and (c) the position control mode. Under each condition, the task is repeated ten times. Under each condition, the task is repeated ten times. During each test, the contact points and forces are recorded for evaluation of the performances.

Fig. 8.7 manifests that under the adaptive control mode, the contact points of each local region are distributed in a more clustered way than that under the position mode, with comparatively low contact forces. Under the position control mode, there are obvious slippery points with larger contact forces due to the rigid interaction with the mouse of the robot hand. The fixed-gain-based control mode obtains a moderate performance with several slippery contact points, although the contact forces are smaller than the case under the direct position control mode. The contact forces from these tests under each control condition were collected and used to calculate the maximum and average forces. The results (see Fig. 8.7(d)) demonstrate significantly lower average as well as maximum forces with the proposed compliance teleoperation system.

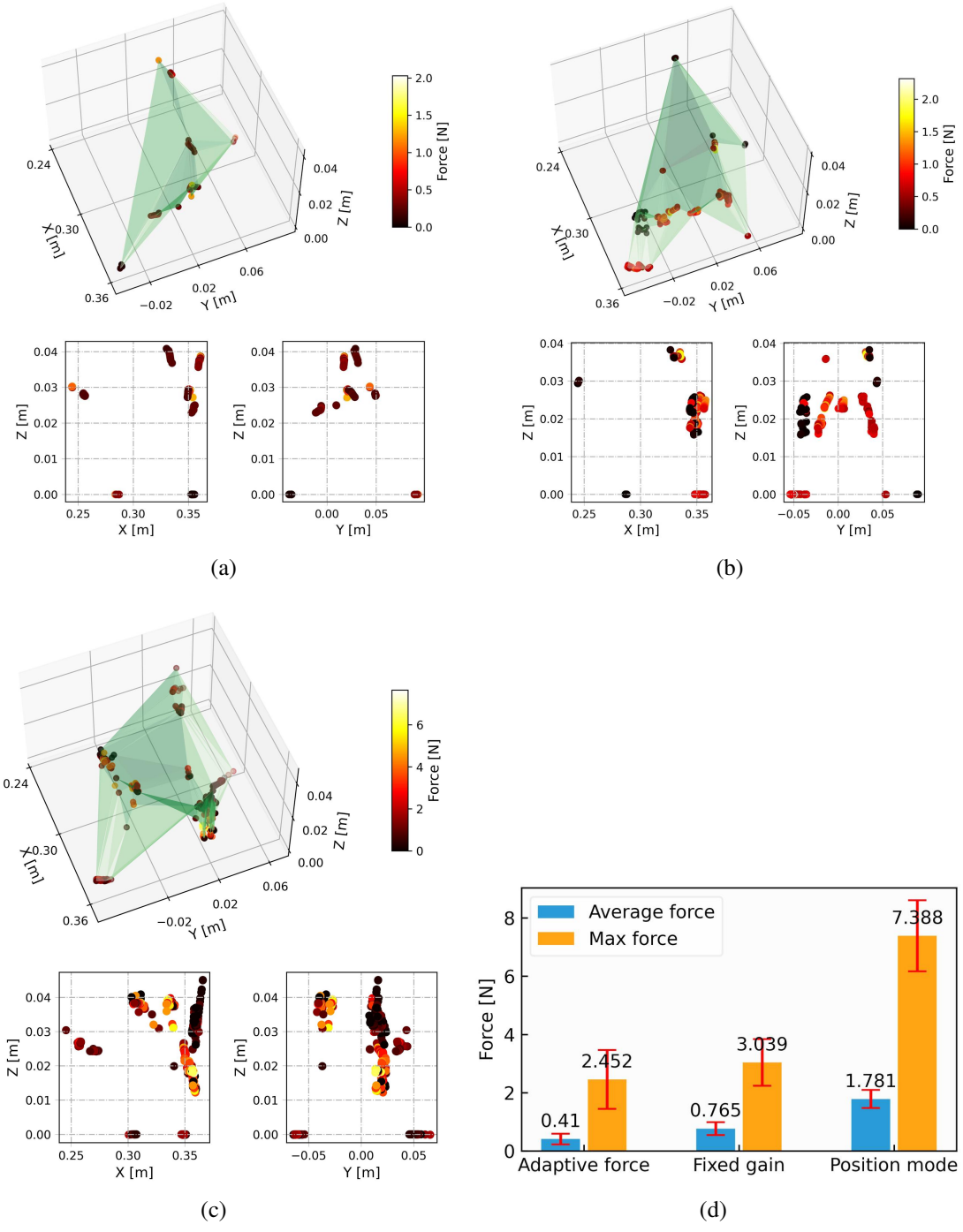


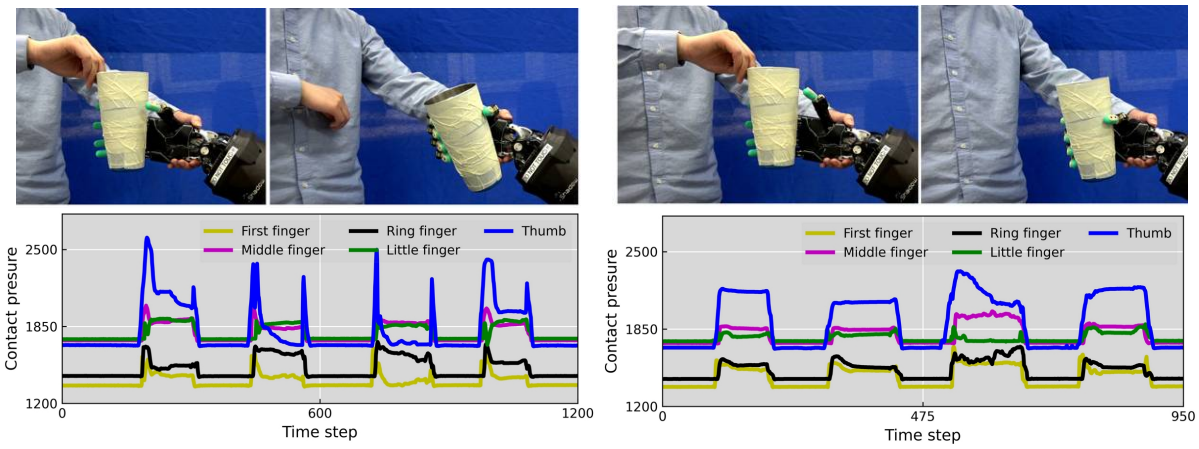
Figure 8.7 – The distribution of the contact points and the contact force during the touching process under the (a) adaptive force control, (b) fixed-gain based force control, and (c) position control conditions, respectively. The upper row show the results in the 3D space, and the lower row shows the corresponding results which are projected to the $x - z$ and $y - z$ planes. (d) shows the max and average contact forces during the touching phase under the three control conditions.

8.4 Real Robot Validation

Instead of a motor-driven robot hand, the Shadow hand is a tendon-driven robot. Applying a force controller to a tendon-driven robot hand is usually more challenging than applying to a motor-driven robot hand in the joint space due to the highly nonlinear characteristics. So far, very few results have been reported in the literature. Typically, Deshpande et al. developed a force-optimized joint controller [36] for the tendon-driven robot ACT Hand and applied it to control one joint (i.e., the MCP joint) of that hand in several tracking experiments. In [123], the authors proposed using force control to increase the grasping stability of the tendon-driven Shadow hand. If perturbations are applied to the grasped object, the joint control force could be increased straightforwardly, as one example of overcoming the perturbations. In the experiment, the Shadow hand was controlled under the effort-control mode, i.e., the TEACH mode. The joint effort measured from the Shadow hand is the difference between the two gauge readings rather than motor torque. To implement compliant teleoperation, the outputs from the force controller were mapped to the effort control commands or PWM value in a linear manner, instead of directly sending the outputs to the joints. The maximum PWM (Pulse Width Modulation) value of each joint was set same for the effort-control mode and the position-control mode.

Grasping. As tested in chapter 4, the robot hand succeeded in grasping objects with different shapes and sizes, using both power grasp and precision grasp. The dynamical grasp process and contact force from the tactile sensors on the fingertips are observed to compare the grasping performance of the adaptive force control with the position control mode. During each grasp, the human partner held the cup in a very similar pose, and another subject teleoperated the robot hand to grasp and took over the cup from the human hand (see Fig. 8.8). The same subject teleoperated the robot hand under two different modes, using a very similar grasp posture. The task was repeated four times, and the subject kept using as similar as possible grasp postures for two control modes. The measured contact pressure force in Fig. 8.8 indicates that under the position control mode, the robot hand tends to rigidly grasp the cup with a larger contact force (especially for the thumb), resulting in an obvious bump at the touching stage of the grasp. Under the adaptive force mode, on the other hand, the contact force keeps more steady along the grasping process. Then, a soft plastic cup was also grasped under two control modes (see Fig. 8.9). It turns out that under the position mode, the robot hand gets the cup deformed much more easily after contact than under the adaptive force control mode.

Screw pouring. To illustrate how the force control strategy deals with external disturbances, a pouring task was then performed. The robot hand was teleoperated to hold a cup, and the human partner poured a set of screws into the cup, as shown in Fig. 8.10. The total weight of the screws was about 0.6 Kg. With the increasing weight, the thumb tends to slip slightly. Once the slip happens, the pose error becomes larger, the force controller correspondingly increases the command effort automatically to overcome the slip, thus maintaining a stable holding posture. Fig. 8.11(a) visualizes the joint angles (both measured ones from the Shadow robot and estimated ones from TeachNet) and the generated command of the fourth joint of the thumb (i.e., THJ4) during the pouring process. The angle of THJ4 changes obviously when slippage occurs along the vertical direction of the surface of the cup. The command effort is adapted correspondingly to overcome the slipping so that the thumb can be stabilized. However, the control effort profiles stay comparatively low if no slippage happens, see the profiles of the

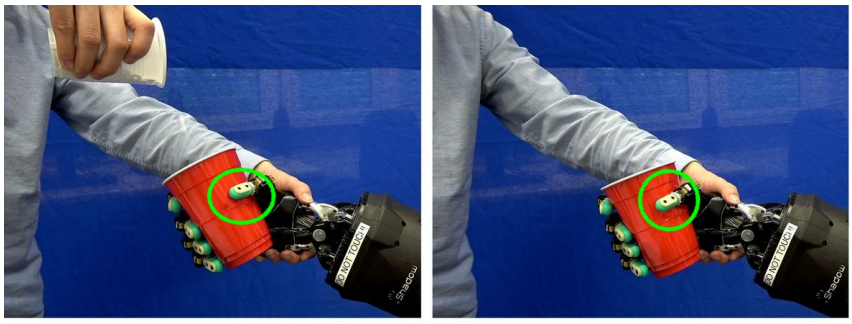


(a) (b)

Figure 8.8 – The upper row illustrates the grasping task under (a) position and (b) adaptive force control modes. The lower row shows the measured contact pressure of the five fingertips from the tactile sensors while continuously grasping the rigid cup four times.

(a) (b)

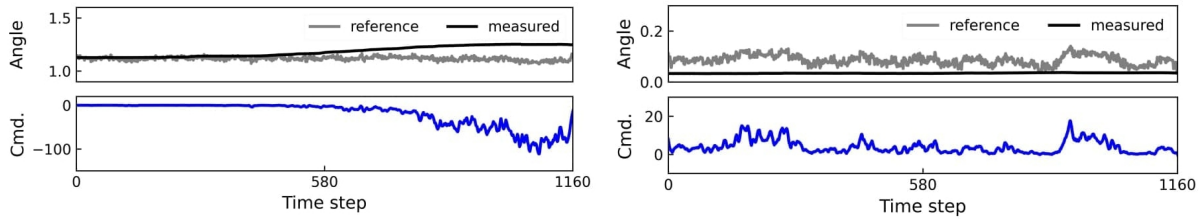
Figure 8.9 – Grasping a soft plastic cup under (a) position and (b) adaptive force control modes. The left depth images are the inputs of the TeachNet model.



(a) (b)

Figure 8.10 – Screenshots of the screw pouring task. (a) and (b) show the situation before and after pouring, respectively. With the increasing weight, the thumb slipped slightly.

fourth joint of the first finger (i.e., FFJ4) as an example in Fig. 8.11(b). These results suggest that the proposed compliant teleoperation system is able to generate appropriate effort commands to overcome the external disturbances adaptively.



(a) (b)

Figure 8.11 – The measured joint angles (real), the estimated joint angles from TeachNet (ref.), and the joint effort commands of joint (a) THJ4 and (b) FFJ4 in the screw pouring task. Cmd. refers to the commanded efforts.

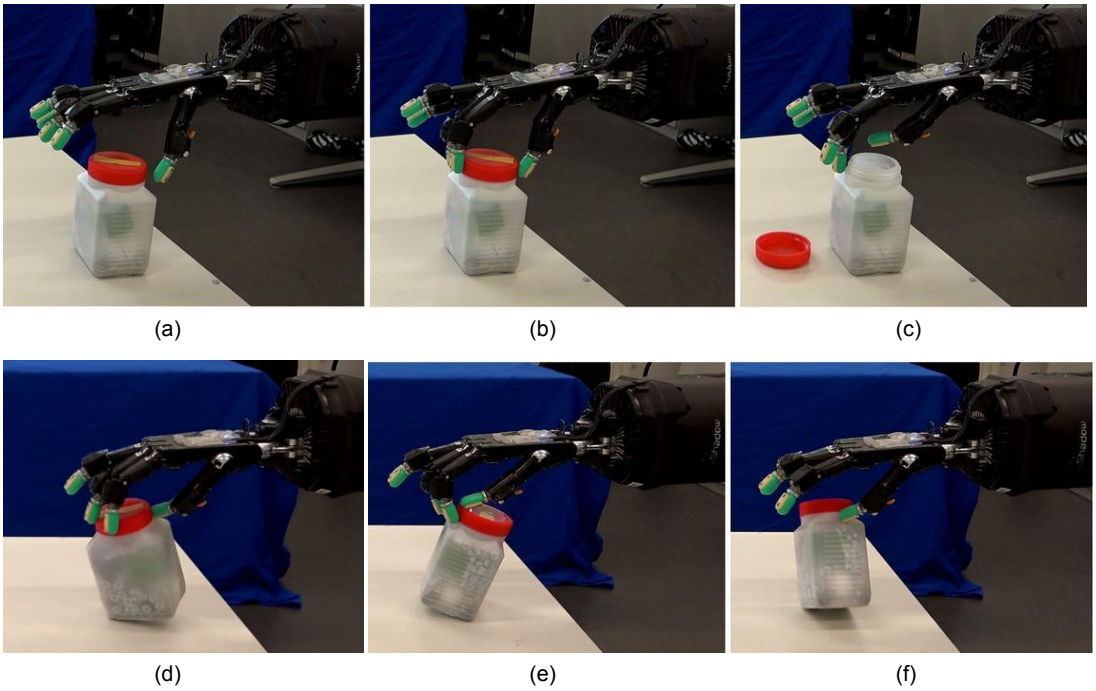


Figure 8.12 – Screenshots of the cap opening task. (a), (b) and (c) display the initial, middle and final configurations under the adaptive force mode, respectively. (d)-(f) show several typical rigid interaction examples under the position mode.

Cap opening. Finally, the task of opening a bottle cap was carried out, requiring great dexterity and compliance by the robot hand. Only three fingers (i.e., TH, FF, and MF) of the robot hand were used. To compare the performance of teleoperation under adaptive force control and position control, only one subject teleoperated the robot in this task. The results demonstrate that the robot can open the bottle cap under the proposed adaptive force control mode (see Fig. 8.12(a)-(c)). By contrast, under the position mode, the robot fingers tend to push the bottle away easily and to interact with the cap quite rigidly (see, Fig. 8.12(d)-(f)), due to their lack of flexibility and dexterity.

8.5 Discussion

This chapter describes compliant robotic teleoperation based on the markerless vision-based teleoperation model, TeachNet, and an adaptive force control strategy (Research Question Q4). The learning-control approach takes a depth image of the human hand as the input and predicts the desired force control commands instead of directly outputting the motion control policies. The force control strategy adapts the compliant profiles (impedance and feedforward) online and step-by-step in the force controller, based on the pose difference between the human hand and the robot hand.

The compliant teleoperation has been evaluated in several simulation tasks and real-world robotic tasks. The results show that it can perform better than the prevalent, state-of-the-art position control mode for robot-compliant grasping and manipulation. Even though we only tested the integration of the TeachNet model and the force control strategy, the Transteleop model can be the other alternative of the hand pose estimation algorithm in this learning-control system. It is worth mentioning that the robot hand in the simulation environment is motor-driven, but the real-world Shadow hand is tendon-driven. Despite the differently driven mechanisms, the compliant teleoperation with the adaptive force controller performs well both in the simulated and real-world hands.

However, the drawbacks of the vision-based teleoperation, e.g., the absence of tactile feedback, still exist in this integrated system. We can estimate the interaction force between the robot hand and its environment from tactile signals collected from the tactile sensors mounted on the tips of the Shadow motor hand. The estimated force information can then be included in the control loop as a feedback variable to increase the interaction dexterity. Regarding the adaptive force control strategy we employed, it is one of the state-of-the-art works in recent years. But the control strategy needs to be manually set the open parameters, which would affect the performances. In the future, we may test one adaptive force control strategy that online optimizes the pre-set open parameters by optimization techniques, e.g., reinforcement learning. The goal is to further improve the robot's capability of dexterous manipulation during human-in-the-loop teleoperation.

Chapter 9

Conclusions and Outlook

In this thesis, two dexterous hand-arm teleoperation systems in which markerless vision-based teleoperation plays a pivotal role were developed and successfully used for dexterous manipulation such as grasping, pushing, sliding, and cap opening. In conclusion, the research questions raised in section 1.3 can be answered as follows:

Q1: Teleoperation Methods for Anthropomorphic Hand: A markerless vision-based teleoperation method and joint-to-joint mapping were applied to the anthropomorphic robotic hand. Two CNN-based models which learn the pose features of the robotic hand from the visual perception of humans were proposed. TeachNet handled the differences in appearance and anatomy between human and robotic hands with a consistency loss function, while Transteleop bridged the kinematic disparities between the robot hand and the human hand by an image-to-image translation process. To train the proposed two end-to-end learning models, two pairwise human-robot hand datasets that include pairs of depth images in the same gesture and corresponding joint angles of the robot hand were built. Meanwhile, an efficient optimized mapping method was designed to match the Cartesian position and the link direction of the Shadow hand from the human hand pose and properly consider possible self-collisions.

Q2: Teleoperation Methods for Robotic Arm: Using an IMU suit and a motion tracking system, two robotic arm teleoperation approaches were implemented on the right arm of the PR2 robot. The arm trajectories were smooth and real-time through proper velocity and acceleration constraints and trajectory regularization built upon the bio-ik optimization solver. In the IMU setup, the human-robot registration was accomplished by matching the human spine link and the robot torso link. In the motion tracking system, only period incremental motions of the human were performed on the robot arm, which is parallel to the human.

Q3: Integration of Hand-Arm Teleoperation System: In this thesis, the remote hand-arm robot system is a PR2 robot with a Shadow hand mounted on the right arm. The main problem of combining the vision-based robotic hand control with the robotic arm teleoperation is that a single camera's limited field of view restricts the workspace of the human arm and the robot arm. Thus, a 3D printed camera holder and an active vision system were successively designed to avoid constraining the robot workspace by the field of view of the depth camera and perceive better human finger motions.

Q4: System Verification: The accuracy of the TeachNet and Transteleop models was compared with other state-of-the-art methods through three commonly used metrics on testing datasets. The stability and efficiency of the whole hand-arm teleoperation systems were tested over a variety of complex manipulation tasks, including grasping, placing, inserting, pushing, sliding, pouring, handover, and bottle opening. The success rate and time cost of the robot experiments were recorded and analyzed. Furthermore, compliant teleoperation was achieved by a learning-control approach combining the end-to-end model with an adaptive force controller. The performance of the learning control approach tested in several simulation and real-world tasks indicates that compliant teleoperation is more reliable than the current widely-used position control mode for obtaining compliant grasping and manipulation and shows more potential for the proposed vision-based teleoperation models.

The major contribution of this work is using an end-to-end learning scheme in markerless vision-based teleoperation for anthropomorphic hands and generalizing this scheme to dexterous hand-arm teleoperation systems. The end-to-end learning approach explicitly specifies two efficient deep neural networks, which exploit the geometrical resemblance between human hands and the robotic hand and learn the kinematic mappings between them. Combining these hand control models and the real-time arm control methods, the dexterous hand-arm teleoperation systems were established to be accurate, efficient, and robust. The schematic conclusion of this thesis is visualized in Fig. 9.1. While there is still plenty of room for promotion and application of the teleoperation system to other laboratories, this thesis makes a sound argument that the proposed systems are taking a concrete step in the right direction.

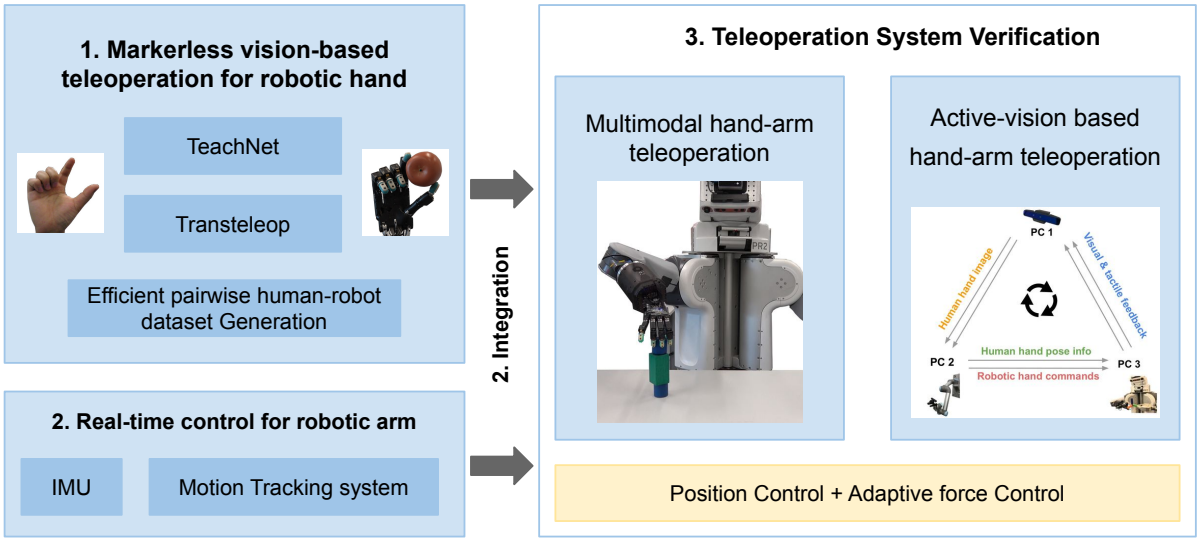


Figure 9.1 – Schematic diagram of thesis conclusion.

9.1 Limitations

Despite the previous chapters presenting meaningful and promising results, some issues currently prevent a more user-friendly teleoperation system using the technologies and hardware setup discussed in this thesis.

The main limitation is robot feedback, including visual feedback and force feedback. In all teleoperation experiments, visual feedback was achieved by directly observing the remote scenario (the users stood close to the remote site) or the real-time stream captured by two cameras. Neither of them provided a comprehensive observation of the operating state. While the robot hand occludes the object from the viewpoint of the human or the camera, visual feedback is entirely or partially lost. This imperfect visual observation could be improved by increasing the number of observing cameras in the remote scene and using virtual reality technologies to achieve immersive teleoperation.

Visual force feedback has been used to promote better sensing of the contacting condition in chapter 7 as a cheap and convenient substitute to haptic sensors. However, force feedback in the visual channel is revealed to be less efficient than haptic force feedback, especially when interacting with soft objects [128]. To perform more complicated robotic tasks, using tactile feedback would be helpful. However, how to maintain the natural and uncustomized advantages of vision-based teleoperation needs to be further studied.

The other limitation from the user's perspective is the heavy control burden. All experiments fully harnessed the cognition of humans and employed a direct control strategy for the robot. Even though direct control is seamless and continuous, it puts all control burden on the users, which is unacceptable in a long-term teleoperation task.

The hardware complexity determines the users' adaptation time to the teleoperation system, the preparation time, and the hardware cost. The 6D global pose of the human hand was obtained by the IMU device in chapter 6 and the PhaseSpace motion tracking system in chapter 7. If the same camera could estimate the 6D global pose and hand joints, the hardware setup would become a lot simpler. However, most hand pose estimation methods are either imprecise or only discover the 3D position of the human wrist without orientation. Therefore, studying a deep-learning-based 6D global hand pose estimation algorithm would be an interesting research topic.

9.2 Future Research

The usability and efficiency of the hand-arm teleoperation system should be further improved based on the limitation summarized in the last section. Regarding robot feedback in teleoperation, several possible developments can be studied in the future.

- Elaborate camera installation. The most naive method to improve visual feedback is by increasing the number of cameras. However, too many visual displays would be a cumbersome visual burden for the users. One camera installed on the robot wrist could be a good choice.

- Immersive perception by virtual reality. Virtual reality promises to view the working environment in 3D and gains a greater understanding of how the remote site works. Creating a realistic and natural observation of the remote site without disorienting lag would be a promising approach [157]. The realistic observation should contain the real-time robot states, object states, and even contacting conditions.
- Slip detection. Slip detection by the tactile sensors on the robot's fingertips could help the robot automatically adjust finger positions or increase joint forces during the manipulation tasks.
- Haptic fingercap. Instead of haptic gloves, haptic fingercaps are a promising device that could be used in our vision-based teleoperation system [7]. Building a mapping relation between the tactile data from Biotac sensors and the vibration sensors of the fingercaps, the haptic feedback around the fingertip will boost the performance of dexterous manipulation.
- Audio alarm. Using audio channels such as auditory force feedback or a workspace boundary alarm could also lessen the human visual burden.

The control strategy of the teleoperation can be updated from direct control to shared control, which combines user commands and remote autonomy. Intention recognition of the users could be developed for repetitive tasks, such as closing and opening the robot hand, placing the object, moving to a specific position. Moreover, task autocorrection associated with slip detection, force estimation, or mixed reality is worth highlighting. For the integration with adaptive force control, a control strategy considering tactile feedback and online optimizing the pre-set open parameters would benefit the robot performance.

The 6D hand wrist estimation is similar to 6D object pose estimation with unknown objects models. Apart from developing algorithms based on existing state-of-the-art object pose estimation methods [67], under the premise of ensuring real-time performance, an essential future extension is using multimodal inputs or 3D inputs.

A further improvement could be more dexterous manipulation implementation, such as in-hand cube rotation and electrical screw-driver operation. Collecting a new human hand dataset with comprehensive tool-use postures would help. For robot groundtruth generation, a retargeting method that optimizes the positioning of fingertips (both distance and direction among fingertips) could be studied [63].

Finally, a crossmodal teleoperation scheme is an important research direction [W44, 66]. For example, teleoperation by visual and neurophysiological signal fusion. Some work has represented how to decode user's intentions from neurophysiological signals using the promising brain–machine interfaces (BMIs) and translate them into a control signal for the robotic device [154]. However, the integration between BMI systems and robotics is still at its infancy as the robotic application only extends to picking and placing by a robot arm. Therefore, combining vision data and neurophysiological signals such as EEG to teleoperate a dexterous hand will be an exciting achievement.

Appendix A

Basics of Convolution Neural Networks

The aim of this chapter is to provide the reader with some basic knowledge and extended applications of Convolutional Neural Network (CNN). The proposed TeachNet and Transteleop models in this thesis are built based on CNNs.

A CNN is a type of artificial neural network and is commonly used for image classification, image segmentation, natural language processing, and data analysis. It also has been applied to many research fields, e.g., computer science, meteorology, medical systems, and robotics. Inspired by biological processes, the connectivity pattern between neurons of CNNs is is similar to the organization of animal cortical neurons. Many CNNs of different structures have been proposed, and ten well-known networks are listed in chronological order in Fig. A.1.

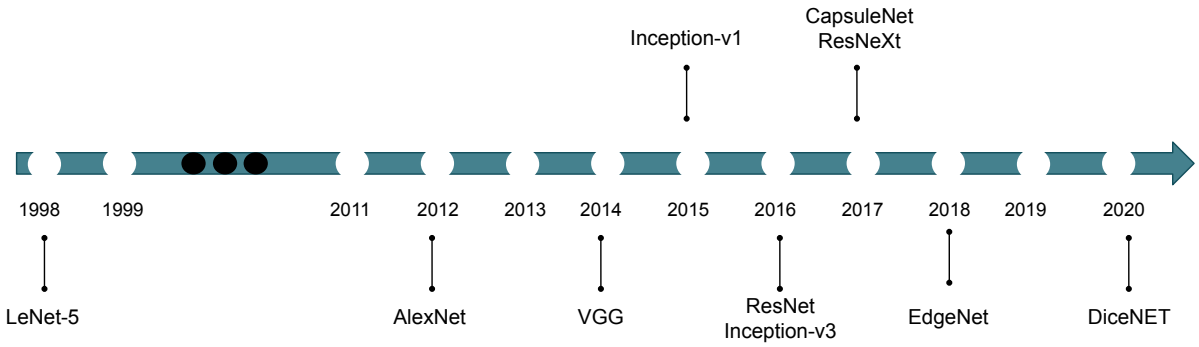


Figure A.1 – Time diagram of ten well-known CNN with different architectures [85, 81, 142, 149, 64, 150, 165, 134, 22, 104].

This chapter starts with the foundation of modern CNNs, LeNet-5 (A.1) and with the Residual Neural Network (ResNet) which solves the saturated accuracy problem when network depth increases (A.2). Afterwards, section A.3 presents several applications of CNNs taking depth images as inputs and describes how depth images are used in specified tasks. Moreover, CNNs with end-to-end learning schemes are introduced in section A.4.

A.1 LeNet-5

LeNet-5 is the foundation of modern CNNs and was proposed by Yann LeCun in 1998 [85]. Initially, it was introduced to recognize hand-written numbers. LeNet-5 has seven layers, as illustrated in Fig. A.2. Its architecture has become a standard CNN design template: stacks convolutional layers with activation functions, pooling layers for feature extraction, and ends the network with one or more fully-connected layers for forming the outputs. Based on the structure of LeNet-5, the common layers in CNNs are introduced as follows.

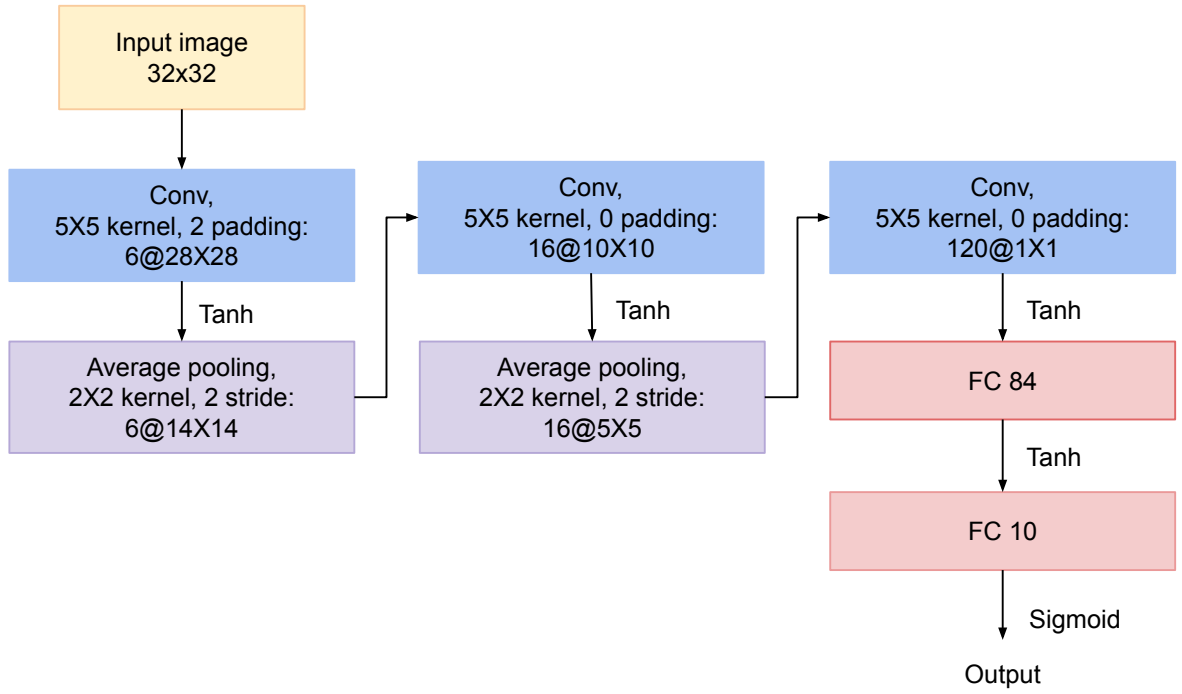


Figure A.2 – The architecture of LeNet-5. It has seven layers, three convolutional layers, two average pooling layers, and two fully-connected layers.

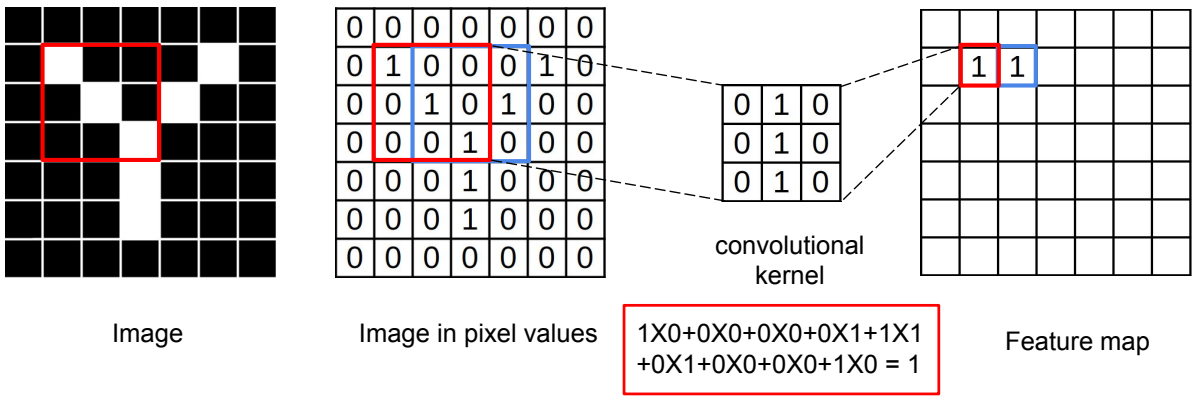


Figure A.3 – Convolution operation.

A.1.1 Convolutional Layer

The convolutional layer is the main building block used in CNN. A convolution is an operation that applies a filter (a kernel) to an input and results in a map of activations called a feature map. The filter is a spatial matrix that slides over the entire input data, calculates the dot product with the sub-region of input data, and gets the output as the matrix of dot products. Similar to animal cortical neurons, the artificial neurons only respond to a part of the surrounding cells in their receptive fields.

The procedure of a convolutional operation is illustrated in Fig. A.3. On the one hand, the weights of one kernel are identical for each pixel in the next layer. This is the "Parameter Sharing" feature of a convolutional layer. On the other hand, every pixel at the next layer is connected to pixels in a local area at the current layers, rather than all pixels from the current layer. This "Sparsity of connections" characteristic of a convolutional layer harnesses the properties of an image that a group of nearby pixels has better information than grouped distant pixels. These two characteristics explain why convolutional layers cause fewer parameters than traditional artificial neural networks. The formulation of a 2D convolution is

$$\mathcal{F}_{2d}(i,j) = \sum_{i=0}^{k}\sum_{j=0}^{k} G(i,j) \otimes H(k-i,k-j), \tag{A.1}$$

where k is the kernel size, usually 3, 5 or 7; G is the image matrix; H is the kernel matrix, and (i, j) is the matrix position of the output feature map.

How many pixels the filter moves in each step is the *stride*. By default, the stride is 1. Apparently, after several convolutional layers, the feature maps get shrunk. The other outcome is that the corner pixels in the input images only get covered by the filter once, while the middle pixels get covered multiple times. Therefore, the corner information of the inputs could be lost. Consequently, *padding* is introduced before the convolutional process. The padding size means how many pixels are added to the border of the input image. Usually, the values of the padding pixels are 0.

The inputs of LeNet-5 are $32 \times 32 \times 3$ RGB images. The first convolutional layer uses six convolution kernels of 5×5 with 2 padding pixels and 1 stride. It means that the 32×32 input image will be converted into 34×34, and the filter moves 1 pixel forward in each step. So the output from the first convolutional layer is $28 \times 28 \times 6$.

A.1.2 Pooling Layer

The role of a pooling layer is sub-sampling the feature map and getting rid of spatial variance, which enables the machine to recognize the object robustly. The larger the kernel size, the larger the receptive field and the smaller the output feature map. The two most common pooling approaches are max pooling and average pooling. The max/average pooling layer gets the maximum/average value in its receptive field layer. In LeNet-5, an average pooling layer with 2×2 kernel follows a convolutional layer. The stride of the pooling is 2, so the 2×2 receptive fields are non-overlapping. Then the output from the first pooling layer in LeNet-5 is $14 \times 14 \times 6$.

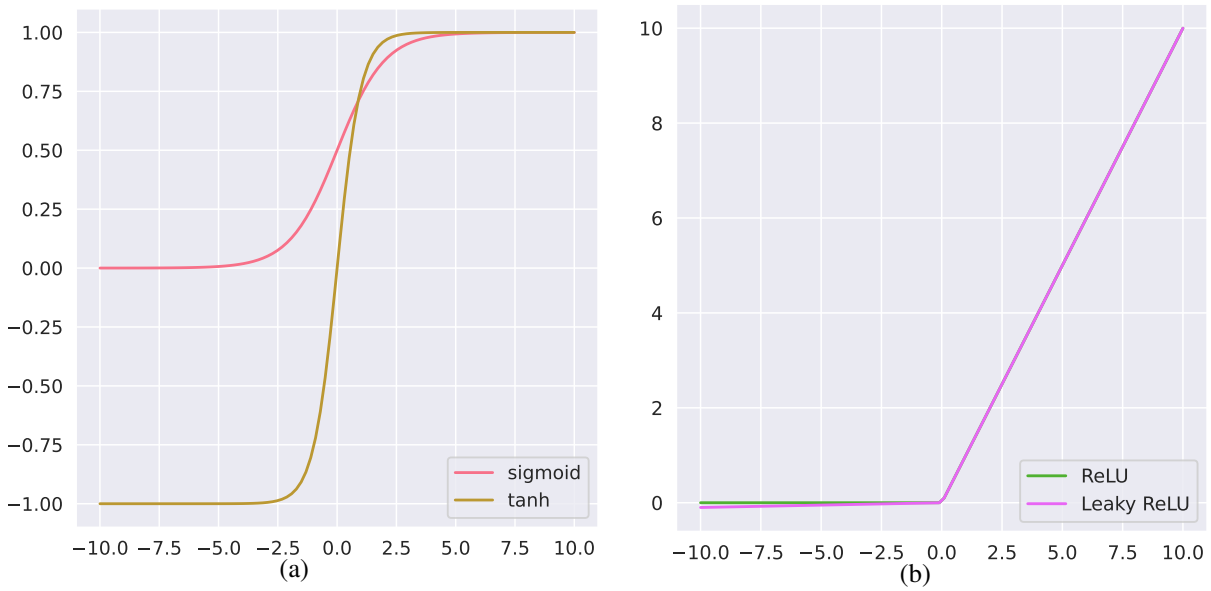


Figure A.4 – Visualization of the four nonlinear activation functions Sigmoid, Tanh, ReLU, and Leaky ReLU.

A.1.3 Fully-connected Layer

In LeNet-5, after two convolutional layers and two pooling layers, there are three FCs, which comply with the data extracted by previous layers to produce the final output. A FC flattens the multichannel feature maps and connects all inputs from one layer to every activation unit of the next layer. Furthermore, each connection has its own weight. If one neuron in a FC layer updates, all values in the next layer will be affected. In contrast, if one neuron in a convolutional layer updates, it only affects the neighboring neurons within the receptive field of the convolutional kernel. In addition, the same weights on the kernel are applied to every neuron. Therefore, a convolutional layer usually has fewer parameters than a fully-connected layer. For example, the first and second convolutional layers have 156, 1516 parameters, respectively, but the first FC layer has 10164 parameters.

A.1.4 Nonlinear Activation Function

Non-linear activation functions introduce non-linearity into the network and allow the network to create complex mappings between high-dimensional inputs and low-dimensional outputs. Every convolutional layer in LeNet-5 follows a non-linear activation function, the hyperbolic tangent Tanh. The output layer follows a sigmoid function, whose output is $[0, 1]$. The common non-liner activation functions are Sigmoid, Tanh, Rectified Linear Unit (ReLU), and Leaky ReLU. Their equations and visualization are shown in equation A.2 - equation A.5 and Fig. A.4.

The Sigmoid function is especially suitable for predicting the probability in a two-class classification task. The Tanh function outputs values in the range of $[-1, 1]$. One disadvantage of the Sigmoid and Tanh functions is value saturation. Saturation means that larger values are clamped to 1, and small ones are clamped to 0 or =1. Further, Sigmoid and Tanh functions are sensitive to input changes around the input's midpoint (0.5 or 0). The ReLU function returns

the same input value directly, or returns 0 if the input is 0.0 or negative. Therefore, ReLU is a piecewise linear function. ReLU has become the most popular activation function because of the simple computation, sparse representation of returning 0 values, and piecewise linear behavior. However, the main limitation of ReLU is that all the negative values become 0 immediately, which reduces the ability of the model to fit or train data appropriately. A node may output an activation value of 0.0 during the whole training process, referred to as the "dying ReLU" phenomenon. The Leaky ReLU aims to solve the dying ReLU issue and relaxes the non-linear output of the function to allow for small negative values.

$$\mathcal{A} = \tanh(x), \tag{A.2}$$

$$\mathcal{A} = \frac{1}{1 + \mathrm{e}^{-\mathrm{x}}}, \tag{A.3}$$

$$\mathcal{A} = \begin{cases} 0 & \text{if } x < 0 \\ x & otherwise \end{cases} \tag{A.4}$$

$$\mathcal{A} = \begin{cases} 0.01x & \text{if } x < 0 \\ x & otherwise \end{cases} \tag{A.5}$$

where x is the input unit.

In summary, we conclude the benefits of CNNs:

1) CNNs reduce the number of trainable network parameters because of weight sharing the convolutional layers and, in turn, help the network to enhance generalization and to avoid overfitting problems.

2) CNNs effectively extract highly organized and highly reliant features from high-dimensional data.

3) Large-scale network implementation is much easier with CNNs than with other traditional artificial neural networks.

4) Moreover, in general, CNNs decrease the need for human effort to develop functionalities of the complex system.

A.2 Residual Neural Network

From 2012, various CNN architectures have been proposed, and CNN-based algorithms have been applied to different research fields. From the 7-layer LeNet-5 to the 8-layer AlexNet [81], the 19-layer VGG [142], and the 27-layer Inception-v1 [149], the main trend of CNNs structure is becoming deeper. In these neural network models, the first several convolutional layers learn low-level features of input images, such as edges, dots, and curves. The later convolutional layers extract high-level features to recognize object types and shapes. Nevertheless, researchers found that simply stacking convolutional layers brings even worse training results on many datasets. As the depth of the network increases, the accuracy saturates and then drops rapidly [64].

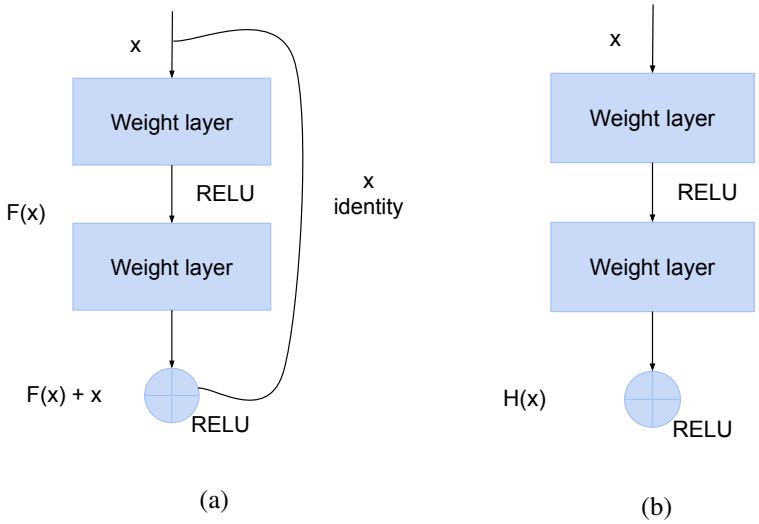


Figure A.5 – A building block in (a) a residual neural network and (b) a plain network.

A deep residual neural network (ResNet) was designed to ease the degradation problem. The critical point of a residual neural network is explicitly reconstructing the layers as learning residual functions regarding the layer inputs instead of learning unreferenced functions. As illustrated in Fig. A.5, utilizing a skip connection to jump over two layers and an element-wise addition formulates operation $F(x) + x$. Function $F(x)$ can contain two layers or three layers, and these layers can be FC layers or convolutional layers. Further experiments and comparisons have proved that residual networks with this identity mapping are easier to optimize and obtain higher accuracy than plain networks with the same network depth. Note that ResNet utilizes Batch Normalization after every convolutional layer before the activation operation to stabilize the learning process and to speed up the learning process. BN is achieved by calculating the mean and standard deviation of each input variable for each mini-batch and performing standardization by these statistics.

A.3 Convolution Neural Networks using Depth Images

Unlike RGB images with three channels: red, green, and blue, depth images have only one channel in which each pixel associates to a distance between the image plane and the corresponding object in the image. With the popularity of depth sensors, depth images have become easy to obtain, so neural networks that take depth maps as input are also developing rapidly in many applications, e.g., human hand pose estimation, image segmentation, and robotic grasping. Human hand pose estimation based on depth images of a bare hand has been elaborated in section 2.4. The depth images in human hand pose estimation are cropped and usually augmented by rotation, translation, and jittering process.

Depth images fed into image segmentation CNNs usually are complete from the observing scene and augmented by multiplicative gamma noise. Xie et al. [164] developed a two-stage CNN, UOIS-Net, for unseen object instance segmentation. UOIS-Net first generated object instance center votes through a depth seeding network and assembled them into rough initial

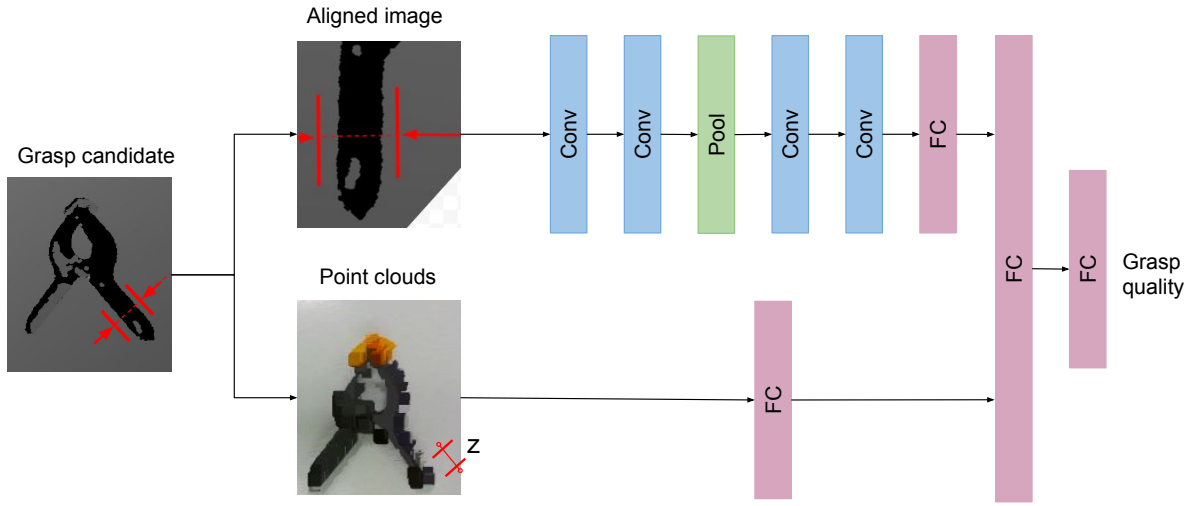


Figure A.6 – Architecture of the grasp quality convolutional neural network GQ-CNN [101]. Adapted image from [101].

masks. Secondly, a region refinement network combining these initial masks and RGB images produced accurate segmentation masks. This framework outperformed the other state-of-the-art methods through quantitative and qualitative comparison experiments on 2D and 3D test datasets.

Regarding grasp pose detection, CNNs using depth images mainly focus on how to represent the grasp candidates by depth images effectively. In 2017, Mahler et al. [101] proposed a grasp quality convolutional neural network (GQ-CNN) to estimate grasp quality based on depth images in a Dex-Net 2.0 grasp planner (see Fig. A.6). The input depth images of GQ-CNN are the depth images centered on the grasp pose center and aligned with the grasp axis orientation. The top-down grasp proposals are generated by antipodal points on the object boundary based on force closure. The other input of GQ-CNN is the gripper depth extracted from the scene point clouds of the observing scene. In over 1,000 physical evaluations, the Dex-Net 2.0 grasp planner achieved a 99% success rate on a test set of 40 novel objects. Similar to GQ-CNN, Ten Pas et al. [120] built a grasp detection CNN to score the 6D grasp candidates for unknown objects. The grasp candidates are represented as three depth images by the geometry of the observed object surfaces and unobserved volumes within the closing area of the gripper.

A.4 End-to-end Learning

With the development of deep and complex neural networks, end-to-end learning has become a fashion because it blurs the traditional boundaries between learning machines and other processing components [58]. End-to-end learning aims to integrate all modules into one neural network, representing the complex target system and bypassing explicit modeling.

The traditional pipeline of a speech recognition system 1) extracts features from the audio inputs, then 2) detects the individual phoneme and 3) composites words, and 4) outputs the text transcripts at the end. An obvious limitation of the traditional method is that each module must be optimized separately under different criteria. Instead, an end-to-end learning algorithm can

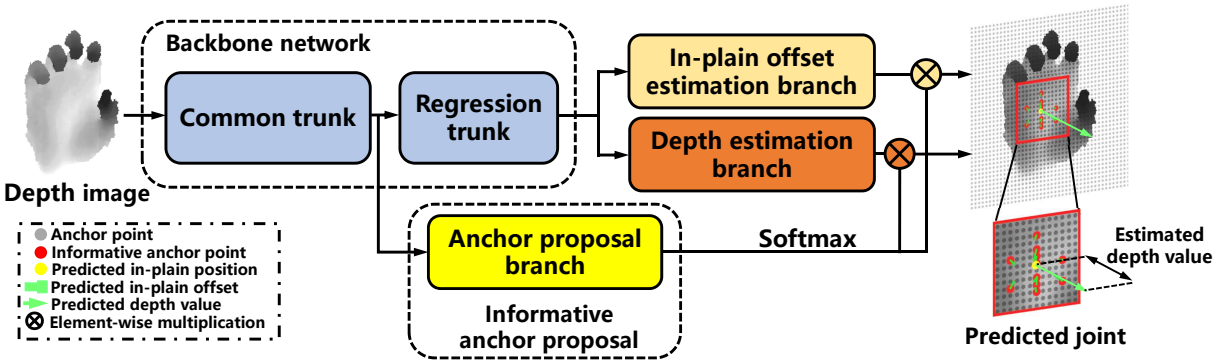


Figure A.7 – Architecture of the A2J model in an end-to-end manner [166]. Reprinted image: ©2019 IEEE.

directly generate the transcripts from the input audios using a sophisticated RCNN. In the hand pose estimation field, earlier methods usually detected the joint positions on 2D heatmaps, then got the 3D joint positions by complex optimization-based post-processing [153]. In recent years, 3D pose estimation networks in end-to-end manners have been exponentially increasing. For example, Xiong et al. [166] proposed a novel anchor-based anchor-to-joint hand pose estimation network (A2J) in the end-to-end learning fashion. The architecture of A2J is shown in Fig. A.7. The backbone network is formed on ResNet-50. The 3D hand keypoints are predicted by the in-plain offset prediction branch and depth estimation branch. The anchor proposal branch assigns weights to all anchor points, which can be assumed as local regressors towards the joints from different viewpoints and distances. Regarding robotic grasping, most existing methods solve the 6-DoF grasping problem for an unseen object by grasp pose generation and grasp pose detection in two steps. Wu et al. [163] designed an end-to-end grasp proposal network built on a novel grasp proposal module that defines anchors of grasp centers at a discrete set of regular 3D grid corners. Moreover, the real-world grasping experiments verified that their dedicated grasp proposal module outperformed the current state-of-the-art grasping methods. In the imitation learning field, Finn et al. [51] presented how to fine-tune vision-based policies end-to-end from one-shot demonstration using model-agnostic meta-learning on several robotic manipulation tasks.

To summarize, the elegant but somewhat brute-force end-to-end learning directly targets the goal system, but every enhancement comes with a price. The architectures of end-to-end models are usually more complicated and have more parameters than single-module networks, resulting in training difficulties.

Appendix B

List of Publications

Here is the list of publications included in this thesis:

- Shuang Li, Xiaojian Ma, Hongzhuo Liang, Michael Görner, Philipp Ruppel, Bin Fang, Fuchun Sun, and Jianwei Zhang. "Vision-based teleoperation of Shadow dexterous hand using end-to-end deep neural network". In: *IEEE International Conference on Robotics and Automation (ICRA)*. Montreal, 2019, pp. 416–422.
- Shuang Li, Jiaxi Jiang, Philipp Ruppel, Hongzhuo Liang, Xiaojian Ma, Norman Hendrich, Fuchun Sun, and Jianwei Zhang. "A mobile robot hand-arm teleoperation system by vision and IMU". in: *IEEE International Conference on Intelligent Robots and Systems (IROS)*. Las Vegas, 2020, pp. 10900–10906.
- Shuang Li, Norman Hendrich, Hongzhuo Liang, Philipp Ruppel, Changshui Zhang, and Jianwei Zhang. "A dexterous hand-arm teleoperation system based on hand pose estimation and active vision". In: *IEEE Transactions on Cybernetics* (2021). Under Review.
- Chao Zeng, Shuang Li, Yiming Jiang, Qiang Li, Zhaopeng Chen, Chenguang Yang, and Jianwei Zhang. "Learning compliant grasping and manipulation by teleoperation with adaptive force control". In: *IEEE International Conference on Intelligent Robots and Systems (IROS)*. Prague, 2021, pp. 717–724.
- Chao Zeng, Shuang Li, Zhaopeng Chen, Chenguang Yang, Sun Fuchun, and Jianwei Zhang. "Multi-fingered robot hand compliant manipulation based on vision-based demonstration and adaptive force control". In: *IEEE Transactions on Systems, Man and Cybernetics: Systems* (2021). Under Review.

Here is the list of publications that was generated during the period of PhD but is not included in this thesis:

- Shuang Li, Hongzhuo Liang, and Jianwei Zhang. "Path planning for wheeled mobile service robots based on improved genetic algorithm". In: *International Convention on Rehabilitation Engineering and Assistive Technology*. Shanghai, 2018, pp. 249–252.
- Hongzhuo Liang, Xiaojian Ma, Shuang Li, Michael Görner, Song Tang, Bin Fang, Fuchun Sun, and Jianwei Zhang. "PointnetGPD: detecting grasp configurations from point sets". In: *International Conference on Robotics and Automation (ICRA)*. 2019, pp. 3629–3635.

- Hongzhuo Liang, Shuang Li, Xiaojian Ma, Norman Hendrich, Timo Gerkmann, Fuchun Sun, and Jianwei Zhang. "Making sense of audio vibration for liquid height estimation in robotic pouring". In: *IEEE International Conference on Intelligent Robots and Systems (IROS)*. 2019, pp. 5333–5339.
- Hongzhuo Liang, Chuangchuang Zhou, Shuang Li, Xiaojian Ma, Norman Hendrich, Timo Gerkmann, Fuchun Sun, and Jianwei Zhang. "Robust robotic pouring using audition and haptics". In: *IEEE International Conference on Intelligent Robots and Systems (IROS)*. 2020, pp. 10880–10887.
- Hongzhuo Liang, Lin Cong, Norman Hendrich, Shuang Li, Fuchun Sun, and Jianwei Zhang. "Multifingered Grasping Based on Multimodal Reinforcement Learning". In: *IEEE Robotics and Automation Letters* (2021). To appear, pp. 1–8.
- Chao Zeng, Shuang Li, Bin Fang, Chen Zhaopeng, and Jianwei Zhang. "Generalization of Robot Force-Relevant Skills through Adapting Compliant Profiles". In: *IEEE Robotics and Automation Letters* (2021). To appear, pp. 1–8.

Appendix C

Glossary

AR Augmented Reality

BN Batch Normalization

CMC Carpometacarpal

CNN Convolutional Neural Network

DIP Distal Interphalangeal

DoF Degree of Freedom

EEG Electroencephalography

EMG Electromyography

FC Fully-connected Layer

FF First Finger

GAN Generative Adversarial Network

IMU Inertial and Magnetic Measurement Unit

ICP Iterative Closest Point

IP Interphalangeal

LF Little Finger

MCP Metacarpophalangeal

MF Middle Finger

MR Mixed Reality

MSE Mean Squared Error

PCA Principal Component Analysis

PIP Proximal Interphalangeal

PN Perception Neuron

PSO Particle Swarm Optimization

RCNN Recurrent Convolutional Neural Network

ReLU Rectified Linear Unit

RF Ring Finger

ROS Robot Operating System

STN Spatial Transformer Network

TCN Time-contrastive Network

TMC Trapeziometacarpal

TH Thumb

URDF Unified Robot Description Format

VAE Variational Autoencoder

VR Virtual Reality

References

[1] Tahir Abbas, Vassilis-Javed Khan, Ujwal Gadiraju, Emilia Barakova, and Panos Markopoulos. “Crowd of oz: a crowd-powered social robotics system for stress management”. In: *Sensors* 20.2 (2020), p. 569.

[2] Peter K Allen, Aleksandar Timcenko, Billibon Yoshimi, and Paul Michelman. “Automated tracking and grasping of a moving object with a robotic hand-eye system”. In: *IEEE Transactions on Robotics and Automation* 9.2 (1993), pp. 152–165.

[3] John Aloimonos, Isaac Weiss, and Amit Bandyopadhyay. “Active vision”. In: *International Journal of Computer Vision* 1.4 (1988), pp. 333–356.

[4] Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. “Learning dexterous in-hand manipulation”. In: *The International Journal of Robotics Research* 39.1 (2020), pp. 3–20.

[5] Dafni Antotsiou, Guillermo Garcia-Hernando, and Tae-Kyun Kim. “Task-oriented hand motion retargeting for dexterous manipulation imitation”. In: *European Conference on Computer Vision Workshop (ECCVW)*. 2018.

[6] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. “A survey of robot learning from demonstration”. In: *Robotics and Autonomous Systems* 57 (2009), pp. 469–483.

[7] Oscar Javier Ariza Nunez. “Wearable haptic technology for 3D selection and guidance”. PhD thesis. Universität Hamburg, 2020. URL: https://ediss.sub.uni-hamburg.de/handle/ediss/8926.

[8] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. “Segnet: A deep convolutional encoder-decoder architecture for image segmentation”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.12 (2017), pp. 2481–2495.

[9] Seungryul Baek, Kwang In Kim, and Tae-Kyun Kim. “Augmented skeleton space transfer for depth-based hand pose estimation”. In: *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 8330–8339.

[10] Hadi Beik-Mohammadi, Matthias Kerzel, Benedikt Pleintinger, Thomas Hulin, Philipp Reisich, Annika Schmidt, Aaron Pereira, Stefan Wermter, and Neal Y Lii. "Model mediated teleoperation with a hand-arm exoskeleton in long time delays using reinforcement learning". In: *IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. 2020, pp. 713–720.

[11] Yoshua Bengio, Aaron Courville, and Pascal Vincent. "Representation learning: A review and new perspectives". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.8 (2013), pp. 1798–1828.

[12] Alexandre Bernardino, Marco Henriques, Norman Hendrich, and Jianwei Zhang. "Precision grasp synergies for dexterous robotic hands". In: *IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE. 2013, pp. 62–67.

[13] Paul J Besl and Neil D McKay. "Method for registration of 3-D shapes". In: *Sensor fusion IV: Control Paradigms and Data Structures*. Vol. 1611. International Society for Optics and Photonics. 1992, pp. 586–606.

[14] Matteo Bianchi, Robert Haschke, Gereon Büscher, Simone Ciotti, Nicola Carbonaro, and Alessandro Tognetti. "A multi-modal sensing glove for human manual-interaction studies". In: *Electronics* 5.3 (2016), p. 42.

[15] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. "End to end learning for self-driving cars". In: *arXiv preprint arXiv:1604.07316* (2016).

[16] Ludovic Brethes, Paulo Menezes, Frédéric Lerasle, and J Hayet. "Face tracking and hand gesture recognition for human-robot interaction". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2004, pp. 1901–1906.

[17] Etienne Burdet, Rieko Osu, David W Franklin, Theodore E Milner, and Mitsuo Kawato. "The central nervous system stabilizes unstable dynamics by learning optimal impedance". In: *Nature* 414.6862 (2001), pp. 446–449.

[18] Yujun Cai, Liuhao Ge, Jun Liu, Jianfei Cai, Tat-Jen Cham, Junsong Yuan, and Nadia Magnenat Thalmann. "Exploiting spatial-temporal relationships for 3D pose estimation via graph convolutional networks". In: *IEEE International Conference on Computer Vision (ICCV)*. 2019, pp. 2272–2281.

[19] Berk Calli, Wouter Caarls, Martijn Wisse, and P Jonker. "Viewpoint optimization for aiding grasp synthesis algorithms using reinforcement learning". In: *Advanced Robotics* 32.20 (2018), pp. 1077–1089.

[20] Berk Calli, Martijn Wisse, and Pieter Jonker. "Grasping of unknown objects via curvature maximization using active vision". In: *IEEE International Conference on Intelligent Robots and Systems (IROS)*. 2011, pp. 995–1001.

[21] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. "OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields". In: *IEEE transactions on pattern analysis and machine intelligence* 43.1 (2019), pp. 172–186.

[22] Justin Cappos, Matthew Hemmings, Rick McGeer, Albert Rafetseder, and Glenn Ricart. "EdgeNet: A global cloud that spreads by local action". In: *Symposium on Edge Computing (SEC)*. IEEE. 2018, pp. 359–360.

[23] Miguel A Carreira-Perpinan and Geoffrey Hinton. "On contrastive divergence learning". In: *International Workshop on Artificial Intelligence and Statistics*. PMLR. 2005, pp. 33–40.

[24] Ilaria Cerulo, Fanny Ficuciello, Vincenzo Lippiello, and Bruno Siciliano. "Teleoperation of the SCHUNK S5FH under-actuated anthropomorphic hand using human hand motion tracking". In: *Robotics and Autonomous Systems* 89 (2017), pp. 75–84.

[25] Ritwik Chattaraj, Bikash Bepari, and Subhasis Bhaumik. "Grasp mapping for Dexterous Robot Hand: A hybrid approach". In: *International Conference on Contemporary Computing (IC3)*. 2014, pp. 242–247.

[26] Weiya Chen, Chenchen Yu, Chenyu Tu, Zehua Lyu, Jing Tang, Shiqi Ou, Yan Fu, and Zhidong Xue. "A survey on hand pose estimation with wearable sensors and computer-vision-based methods". In: *Sensors* 20.4 (2020), pp. 1074–1098.

[27] Xinghao Chen, Guijin Wang, Cairong Zhang, Tae-Kyun Kim, and Xiangyang Ji. "SHPR-Net: deep semantic hand pose regression from point clouds". In: *IEEE Access* 6 (2018), pp. 43425–43439.

[28] Fai Chen Chen, Alain Favetto, Mehdi Mousavi, Elisa Ambrosio, Silvia Appendino, Diego Manfredi, Francesco Pescarmona, Giuseppe Calafiore, and Basilio Bona. "Human Hand: Kinematics, Statics, and Dynamics". In: *International Conference on Environmental Systems*. 2011, pp. 1–10.

[29] Ricson Cheng, Arpit Agarwal, and Katerina Fragkiadaki. "Reinforcement learning of active vision for manipulating objects under occlusions". In: *Conference on Robot Learning (CoRL)*. 2018, pp. 422–431.

[30] Sachin Chitta, Ioan Sucan, and Steve Cousins. "Moveit![ros topics]". In: *Robotics & Automation Magazine* 19.1 (2012), pp. 18–19.

[31] Seung Keun Cho, Hong Zhe Jin, Jang Myung Lee, and Bin Yao. "Teleoperation of a mobile robot using a force-reflection joystick with sensing mechanism of rotating magnetic field". In: *IEEE Transactions On Mechatronics* 15.1 (2009), pp. 17–26.

[32] Wang Dangxiao, Guo Yuan, Liu Shiyi, Yuru Zhang, Xu Weiliang, and Xiao Jing. "Haptic display for virtual reality: progress and challenges". In: *Virtual Reality & Intelligent Hardware* 1.2 (2019), pp. 136–162.

[33] Andrew J Davison and David W. Murray. "Simultaneous localization and map-building using active vision". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24.7 (2002), pp. 865–880.

[34] Yiannis Demiris and Bassam Khadhouri. "Hierarchical attentive multiple models for execution and recognition of actions". In: *Robotics and Autonomous Systems* 54.5 (2006), pp. 361–369.

[35] Xiaoming Deng, Yuying Zhu, Yinda Zhang, Zhaopeng Cui, Ping Tan, Wentian Qu, Cuixia Ma, and Hongan Wang. "Weakly Supervised Learning for Single Depth-Based Hand Shape Recovery". In: *IEEE Transactions on Image Processing* 30 (2020), pp. 532–545.

[36] Ashish D Deshpande, Jonathan Ko, Dieter Fox, and Yoky Matsuoka. "Control strategies for the index finger of a tendon-driven hand". In: *The International Journal of Robotics Research* 32.1 (2013), pp. 115–128.

[37] Endri Dibra, Silvan Melchior, Ali Balkis, Thomas Wolf, Cengiz Oztireli, and Markus Gross. "Monocular RGB hand pose inference from unsupervised refinable nets". In: *IEEE Conference on Computer Vision and Pattern Recognition Workshop (CVPRW)*. 2018, pp. 1075–1085.

[38] Christian RG Dreher, Mirko Wächter, and Tamim Asfour. "Learning object-action relations from bimanual human demonstration using graph networks". In: *IEEE Robotics and Automation Letters* 5.1 (2019), pp. 187–194.

[39] Guanglong Du, Ping Zhang, Jianhua Mai, and Zeling Li. "Markerless kinect-based hand tracking for robot teleoperation". In: *International Journal of Advanced Robotic Systems* 9.2 (2012), pp. 36–45.

[40] Kuo Du, Xiangbo Lin, Yi Sun, and Xiaohong Ma. "CrossInfoNet: Multi-task information sharing based hand pose estimation". In: *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 9896–9905.

[41] Anany Dwivedi, Gal Gorjup, Yongje Kwon, and Minas Liarokapis. "Combining electromyography and fiducial marker based tracking for intuitive telemanipulation with a robot arm hand system". In: *IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. 2019, pp. 1–6.

[42] Anany Dwivedi, Dasha Shieff, Amber Turner, Gal Gorjup, Yongje Kwon, and Minas Liarokapis. "A Shared Control Framework for Robotic Telemanipulation Combin-

ing Electromyography Based Motion Estimation and Compliance Control". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2021, pp. 9467–9473.

[43] Ali Erol, George Bebis, Mircea Nicolescu, Richard D Boyle, and Xander Twombly. "Vision-based hand pose estimation: A review". In: *Computer Vision and Image Understanding* 108.1-2 (2007), pp. 52–73.

[44] Shaowei Fan, Haiwei Gu, Yuanfei Zhang, Minghe Jin, and Hong Liu. "Research on adaptive grasping with object pose uncertainty by multi-fingered robot hand". In: *International Journal of Advanced Robotic Systems* 15.2 (2018), pp. 1–16.

[45] Bin Fang, Xiao Ma, Jiachun Wang, and Fuchun Sun. "Vision-based posture-consistent teleoperation of robotic arm using multi-stage deep neural network". In: *Robotics and Autonomous Systems* 131 (2020), p. 103592.

[46] Bin Fang, Fuchun Sun, and Huaping Liu. "Robotic teleoperation systems using a wearable multimodal fusion device". In: *International Journal of Advanced Robotic Systems* 14.4 (2017).

[47] Linpu Fang, Xingyan Liu, Li Liu, Hang Xu, and Wenxiong Kang. "JGR-P2O: joint graph reasoning based pixel-to-offset prediction network for 3D hand pose estimation from a single depth image". In: *European Conference on Computer Vision (ECCV)*. Springer. 2020, pp. 120–137.

[48] Manuel Ferre, Rafael Aracil, Carlos Balaguer, Martin Buss, and Claudio Melchiorri. *Advances in Telerobotics*. Vol. 31. Springer, 2007.

[49] F Ficuciello, A Migliozzi, G Laudante, P Falco, and B Siciliano. "Vision-based grasp learning of an anthropomorphic hand-arm system in a synergy-based control framework". In: *Science robotics* 4.26 (2019).

[50] Fanny Ficuciello, Luigi Villani, and Bruno Siciliano. "Variable impedance control of redundant manipulators for intuitive human–robot physical interaction". In: *IEEE Transactions on Robotics* 31.4 (2015), pp. 850–863.

[51] Chelsea Finn, Tianhe Yu, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. "One-shot visual imitation learning via meta-learning". In: *Conference on Robot Learning (CoRL)*. PMLR. 2017, pp. 357–368.

[52] Jeremy A Fishel, Toni Oliver, Michael Eichermueller, Giuseppe Barbieri, Ethan Fowler, Toivo Hartikainen, Luke Moss, and Rich Walker. "Tactile Telerobots for Dull, Dirty, Dangerous, and Inaccessible Tasks". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2020, pp. 11305–11310.

[53] Virginia Ruiz Garate, Maria Pozzi, Domenico Prattichizzo, Nikos Tsagarakis, and Arash Ajoudani. "Grasp stiffness control in robotic hands through coordinated optimization

of pose and joint stiffness". In: *IEEE Robotics and Automation Letters* 3.4 (2018), pp. 3952–3959.

[54] Guillermo Garcia-Hernando, Edward Johns, and Tae-Kyun Kim. "Physics-based dexterous manipulations with estimated hand poses and residual reinforcement learning". In: *IEEE International Conference on Intelligent Robots and Systems (IROS)*. 2020, pp. 9561–9568.

[55] Liuhao Ge, Yujun Cai, Junwu Weng, and Junsong Yuan. "Hand PointNet: 3D hand pose estimation using point sets". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 8417–8426.

[56] Liuhao Ge, Zhou Ren, and Junsong Yuan. "Point-to-point regression pointnet for 3D hand pose estimation". In: *European Conference on Computer Vision (ECCV)*. 2018, pp. 475–491.

[57] Guido Gioioso, Gionata Salvietti, Monica Malvezzi, and Domenico Prattichizzo. "Mapping synergies from human to robotic hands with dissimilar kinematics: an approach in the object domain". In: *IEEE Transactions on Robotics* 29.4 (2013), pp. 825–837.

[58] Tobias Glasmachers. "Limits of end-to-end learning". In: *Asian Conference on Machine Learning*. PMLR. 2017, pp. 17–32.

[59] Francisco Gomez-Donoso, Sergio Orts-Escolano, and Miguel Cazorla. "Accurate and efficient 3D hand pose regression for robot hand teleoperation using a monocular RGB camera". In: *Expert Systems with Applications* 136 (2019), pp. 327–337.

[60] Claudia González, J Ernesto Solanes, Adolfo Munoz, Luis Gracia, Vicent Girbés-Juan, and Josep Tornero. "Advanced teleoperation and control system for industrial robots based on augmented virtuality and haptic feedback". In: *Journal of Manufacturing Systems* 59 (2021), pp. 283–298.

[61] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial nets". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2014, pp. 2672–2680.

[62] Hengkai Guo, Guijin Wang, Xinghao Chen, Cairong Zhang, Fei Qiao, and Huazhong Yang. "Region ensemble network: Improving convolutional network for hand pose estimation". In: *IEEE International Conference on Image Processing (ICIP)*. 2017, pp. 4512–4516.

[63] Ankur Handa, Karl Van Wyk, Wei Yang, Jacky Liang, Yu-Wei Chao, Qian Wan, Stan Birchfield, Nathan Ratliff, and Dieter Fox. "Dexpilot: Vision based teleoperation of dexterous robotic hand-arm system". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2020, pp. 9164–9170.

[64] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778.

[65] Kazuki Higashi, Keisuke Koyama, Ryuta Ozawa, Kazuyuki Nagata, Weiwei Wan, and Kensuke Harada. "Functionally divided manipulation synergy for controlling multi-fingered hands". In: *IEEE International Conference on Intelligent Robots and Systems (IROS)*. 2020, pp. 9190–9197.

[66] Focko L Higgen, Philipp Ruppel, Michael Görner, Matthias Kerzel, Norman Hendrich, Jan Feldheim, Stefan Wermter, Jianwei Zhang, and Christian Gerloff. "Crossmodal pattern discrimination in humans and robots: a visuo-tactile case study". In: *Frontiers in Robotics and AI* 7 (2020).

[67] Tomas Hodan, Frank Michel, Eric Brachmann, Wadim Kehl, Anders GlentBuch, Dirk Kraft, Bertram Drost, Joel Vidal, Stephan Ihrke, Xenophon Zabulis, et al. "BOP: Benchmark for 6D object pose estimation". In: *European Conference on Computer Vision (ECCV)*. 2018, pp. 19–34.

[68] Cheng Huang, Huaping Liu, Fuchun Sun, and Yuming Sheng. "Space Robot Teleoperation Based on Active Vision". In: *Practical Applications of Intelligent Systems*. Springer, 2014, pp. 229–240.

[69] Yongqiang Huang, Matteo Bianchi, Minas Liarokapis, and Yu Sun. "Recent data sets on object manipulation: A survey". In: *Big Data* 4.4 (2016), pp. 197–216.

[70] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. "Image-to-image translation with conditional adversarial networks". In: *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 1125–1134.

[71] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. "Spatial transformer networks". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2015, pp. 2017–2025.

[72] Divye Jain, Andrew Li, Shivam Singhal, Aravind Rajeswaran, Vikash Kumar, and Emanuel Todorov. "Learning Deep Visuomotor Policies for Dexterous Hand Manipulation". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2019, pp. 3636–3643.

[73] Roland S Johansson and J Randall Flanagan. "Coding and use of tactile signals from the fingertips in object manipulation tasks". In: *Nature Reviews Neuroscience* 10 (2009), pp. 345–359.

[74] Evangelos Kazakos, Christophoros Nikou, and Ioannis A Kakadiaris. "On the fusion of RGB and depth information for hand pose estimation". In: *IEEE International Conference on Image Processing (ICIP)*. 2018, pp. 868–872.

[75] James Kennedy and Russell Eberhart. “Particle swarm optimization”. In: *IEEE International Conference on Neural Networks*. Vol. 4. 1995, pp. 1942–1948.

[76] Diederik P Kingma and Max Welling. “Auto-encoding variational bayes”. In: *International Conference on Learning Representations (ICLR)*. 2014.

[77] Thomas N Kipf and Max Welling. “Semi-supervised classification with graph convolutional networks”. In: *International Conference on Learning Representations (ICRL)*. 2017, pp. 1–14.

[78] Futoshi Kobayashi, Keiichi Kitabayashi, Hiroyuki Nakamoto, Fumio Kojima, Wataru Fukui, Nobuaki Imamura, and Tadashi Maeda. “Multiple joints reference for robot finger control in robot hand teleoperation”. In: *IEEE International Symposium on System Integration (SII)*. 2012, pp. 577–582.

[79] Futoshi Kobayashi, Hiroyuki Nakamoto, Fumio Kojima, Tadashi Maeda, Nobuaki Imamura, and Hidenori Shirasawa. “Teleoperation of universal robot hand with pinching force stabilization”. In: *Simulation and Modeling Related to Computational Science and Robotics Technology* 37 (2012), pp. 172–184.

[80] Jonathan Kofman, Xianghai Wu, Timothy J Luu, and Siddharth Verma. “Teleoperation of a robot manipulator using a vision-based human-robot interface”. In: *IEEE Transactions on Industrial Electronics* 52.5 (2005), pp. 1206–1219.

[81] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in Neural Information Processing Systems (NeurIPS)* 25 (2012), pp. 1097–1105.

[82] Dennis Krupke. “Mixed reality technologies for novel forms of human-robot interaction”. PhD thesis. Universität of Hamburg, 2019. URL: https://ediss.sub.uni-hamburg.de/handle/ediss/9128.

[83] Dennis Krupke, Frank Steinicke, Paul Lubos, Yannick Jonetzko, Michael Görner, and Jianwei Zhang. “Comparison of multimodal heading and pointing gestures for co-located mixed reality human-robot interaction”. In: *IEEE International Conference on Intelligent Robots and Systems (IROS)*. 2018, pp. 5003–5009.

[84] Hemin Omer Latif, Nasser Sherkat, and Ahmad Lotf. “TeleGaze: Teleoperation through eye gaze”. In: *IEEE International Conference on Cybernetic Intelligent Systems (CIS)*. 2008, pp. 1–6.

[85] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[86] Miao Li, Hang Yin, Kenji Tahara, and Aude Billard. "Learning object-level impedance control for robust grasping and dexterous manipulation". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2014, pp. 6784–6791.

[87] Qiang Li, Robert Haschke, Bram Bolder, and Helge Ritter. "Grasp point optimization by online exploration of unknown object surface". In: *IEEE International Conference on Humanoid Robots (Humanoid)*. 2012, pp. 417–422.

[88] Rui Li, Zhenyu Liu, and Jianrong Tan. "A survey on 3D hand pose estimation: cameras, methods, and datasets". In: *Pattern Recognition* 93 (2019), pp. 251–272.

[89] Shaoqiang Li, Wenliang Guo, Hao Liu, Tao Wang, Zhou YuanYuan, Tao Yu, Chongyang Wang, Yongming Yang, Nanshan Zhong, Nuofu Zhang, and Shiyue Li. "Clinical application of an intelligent oropharyngeal swab robot: implication for the COVID-19 pandemic". In: *European Respiratory Journal* 56.2001912 (2020).

[90] Shuang Li, Norman Hendrich, Hongzhuo Liang, Philipp Ruppel, Changshui Zhang, and Jianwei Zhang. "A dexterous hand-arm teleoperation system based on hand pose estimation and active vision". In: *IEEE Transactions on Cybernetics* (2021). Under Review.

[91] Shuang Li, Jiaxi Jiang, Philipp Ruppel, Hongzhuo Liang, Xiaojian Ma, Norman Hendrich, Fuchun Sun, and Jianwei Zhang. "A mobile robot hand-arm teleoperation system by vision and IMU". In: *IEEE International Conference on Intelligent Robots and Systems (IROS)*. Las Vegas, 2020, pp. 10900–10906.

[92] Shuang Li, Hongzhuo Liang, and Jianwei Zhang. "Path planning for wheeled mobile service robots based on improved genetic algorithm". In: *International Convention on Rehabilitation Engineering and Assistive Technology*. Shanghai, 2018, pp. 249–252.

[93] Shuang Li, Xiaojian Ma, Hongzhuo Liang, Michael Görner, Philipp Ruppel, Bin Fang, Fuchun Sun, and Jianwei Zhang. "Vision-based teleoperation of Shadow dexterous hand using end-to-end deep neural network". In: *IEEE International Conference on Robotics and Automation (ICRA)*. Montreal, 2019, pp. 416–422.

[94] Yanan Li, Gowrishankar Ganesh, Nathanaël Jarrassé, Sami Haddadin, Alin Albu-Schaeffer, and Etienne Burdet. "Force, impedance, and trajectory learning for contact tooling and haptic identification". In: *IEEE Transactions on Robotics* 34.5 (2018), pp. 1170–1182.

[95] Hongzhuo Liang, Lin Cong, Norman Hendrich, Shuang Li, Fuchun Sun, and Jianwei Zhang. "Multifingered Grasping Based on Multimodal Reinforcement Learning". In: *IEEE Robotics and Automation Letters* (2021). To appear, pp. 1–8.

[96] Hongzhuo Liang, Shuang Li, Xiaojian Ma, Norman Hendrich, Timo Gerkmann, Fuchun Sun, and Jianwei Zhang. "Making sense of audio vibration for liquid height estimation

in robotic pouring". In: *IEEE International Conference on Intelligent Robots and Systems (IROS)*. 2019, pp. 5333–5339.

[97] Hongzhuo Liang, Xiaojian Ma, Shuang Li, Michael Görner, Song Tang, Bin Fang, Fuchun Sun, and Jianwei Zhang. "PointnetGPD: detecting grasp configurations from point sets". In: *International Conference on Robotics and Automation (ICRA)*. 2019, pp. 3629–3635.

[98] Hongzhuo Liang, Chuangchuang Zhou, Shuang Li, Xiaojian Ma, Norman Hendrich, Timo Gerkmann, Fuchun Sun, and Jianwei Zhang. "Robust robotic pouring using audition and haptics". In: *IEEE International Conference on Intelligent Robots and Systems (IROS)*. 2020, pp. 10880–10887.

[99] David G Lowe. "Distinctive image features from scale-invariant keypoints". In: *International Journal of Computer Vision* 60.2 (2004), pp. 91–110.

[100] Jing Luo, Zhidong Lin, Yanan Li, and Chenguang Yang. "A teleoperation framework for mobile robots based on shared control". In: *IEEE Robotics and Automation Letters* 5 (2019), pp. 377–384.

[101] Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio Ojea, and Ken Goldberg. "Dex-Net 2.0: Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics". In: *Robotics: Science and Systems (RSS)*. 2017, pp. 12–16.

[102] Jameel Malik, Ibrahim Abdelaziz, Ahmed Elhayek, Soshi Shimada, Sk Aziz Ali, Vladislav Golyanik, Christian Theobalt, and Didier Stricker. "Handvoxnet: deep voxel-based network for 3D hand shape and pose estimation from a single depth map". In: *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 7113–7122.

[103] Cassie Meeker, Thomas Rasmussen, and Matei Ciocarlie. "Intuitive hand teleoperation by novice operators using a continuous teleoperation subspace". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2018, pp. 5821–5827.

[104] Sachin Mehta, Hannaneh Hajishirzi, and Mohammad Rastegari. "DiCENet: Dimension-wise convolutions for efficient networks". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2020).

[105] Damien Michel, Ammar Qammaz, and Antonis A Argyros. "Markerless 3D human pose estimation and tracking based on RGBD cameras: an experimental evaluation". In: *International Conference on PErvasive Technologies Related to Assistive Environments*. ACM. 2017, pp. 115–122.

[106] Youssef Michel, Rahaf Rahal, Claudio Pacchierotti, Paolo Robuffo Giordano, and Dongheui Lee. "Bilateral teleoperation with adaptive impedance control for contact tasks". In: *IEEE Robotics and Automation Letters* 6.3 (2021), pp. 5429–5436.

[107] Mehdi Mirza and Simon Osindero. "Conditional generative adversarial nets". In: *arXiv preprint arXiv:1411.1784* (2014).

[108] Gyeongsik Moon, Ju Yong Chang, and Kyoung Mu Lee. "V2V-PoseNet: voxel-to-voxel prediction network for accurate 3D hand and human pose estimation from a single depth map". In: *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 5079–5088.

[109] Gyeongsik Moon, Shoou-I Yu, He Wen, Takaaki Shiratori, and Kyoung Mu Lee. "InterHand2. 6M: A dataset and baseline for 3D interacting hand pose estimation from a single RGB image". In: *European Conference on Computer Vision (ECCV)*. Springer. 2020, pp. 548–564.

[110] Franziska Mueller, Florian Bernard, Oleksandr Sotnychenko, Dushyant Mehta, Srinath Sridhar, Dan Casas, and Christian Theobalt. "GANerated hands for real-time 3D hand tracking from monocular RGB". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 49–59.

[111] Franziska Mueller, Micah Davis, Florian Bernard, Oleksandr Sotnychenko, Mickeal Verschoor, Miguel A Otaduy, Dan Casas, and Christian Theobalt. "Real-time pose and shape reconstruction of two interacting hands with a single depth camera". In: *ACM Transactions on Graphics (TOG)* 38.4 (2019), pp. 1–13.

[112] Günter Niemeyer, Carsten Preusche, Stefano Stramigioli, and Dongjun Lee. "Telerobotics". In: *Springer Handbook of Robotics*. Springer, 2016, pp. 1085–1108.

[113] Markus Oberweger and Vincent Lepetit. "DeepPrior++: Improving fast and accurate 3D hand pose estimation". In: *IEEE International Conference on Computer Vision Workshop (ICCVW)*. 2017.

[114] Markus Oberweger, Gernot Riegler, Paul Wohlhart, and Vincent Lepetit. "Efficiently creating 3D training data for fine hand pose estimation". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 4957–4965.

[115] Markus Oberweger, Paul Wohlhart, and Vincent Lepetit. "Generalized feedback loop for joint hand-object pose estimation". In: *IEEE transactions on Pattern Analysis and Machine Intelligence* 42.8 (2019), pp. 1898–1912.

[116] Markus Oberweger, Paul Wohlhart, and Vincent Lepetit. "Hands deep in deep learning for hand pose estimation". In: *arXiv preprint arXiv:1502.06807* (2015).

[117] Iason Oikonomidis, Nikolaos Kyriazis, and Antonis A Argyros. "Efficient model-based 3D tracking of hand articulations using kinect". In: *British Machine Vision Conference (BMVC)*. 2011.

[118] Edwin Olson. "AprilTag: A robust and flexible visual fiducial system". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2011, pp. 3400–3407.

[119] Paschalis Panteleris, Iason Oikonomidis, and Antonis Argyros. "Using a single RGB frame for real time 3D hand pose estimation in the wild". In: *IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE. 2018, pp. 436–445.

[120] Andreas ten Pas, Marcus Gualtieri, Kate Saenko, and Robert Platt. "Grasp pose detection in point clouds". In: *The International Journal of Robotics Research* 36.13-14 (2017), pp. 1455–1473.

[121] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. "Context encoders: Feature learning by inpainting". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 2536–2544.

[122] Esteban Peña-Pitarch, Neus Ticó Falguera, and Jingzhou Yang. "Virtual human hand: model and kinematics". In: *Computer Methods in Biomechanics and Biomedical Engineering* 17.5 (2014), pp. 568–579.

[123] Martin Pfanne, Maxime Chalon, Freek Stulp, Helge Ritter, and Alin Albu-Schäffer. "Object-Level Impedance Control for Dexterous In-Hand Manipulation". In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 2987–2994.

[124] Lucia Pigini, David Facal, Lorenzo Blasi, and Renzo Andrich. "Tele-operated robots in elderly care at home: A survey on needs and perceptions of elderly people and caregivers". In: *Everyday Technology for Independence and Care*. IOS Press, 2011, pp. 542–549.

[125] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. "Pointnet: Deep learning on point sets for 3D classification and segmentation". In: *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 652–660.

[126] Chen Qian, Xiao Sun, Yichen Wei, Xiaoou Tang, and Jian Sun. "Realtime and robust hand tracking from depth". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014, pp. 1106–1113.

[127] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. "You only look once: Unified, real-time object detection". In: *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 779–788.

[128] Paul Richard, Grigore Burdea, Daniel Gomez, and Philippe Coiffet. "A comparison of haptic, visual and auditive force feedback for deformable virtual objects". In: *Internation Conference on Automation Technology (ICAT)*. Vol. 49. 1994, p. 62.

[129] Robert N Rohling, John M Hollerbach, and Stephen C Jacobsen. "Optimized fingertip mapping: a general algorithm for robotic hand teleoperation". In: *Presence: Teleoperators & Virtual Environments* 2.3 (1993), pp. 203–220.

[130] Javier Romero. "From Human to Robot Grasping". PhD thesis. KTH Royal Institute of Technology, 2011. URL: https://www.diva-portal.org/smash/get/diva2:459204/FULLTEXT01.pdf.

[131] Philipp Ruppel, Norman Hendrich, Sebastian Starke, and Jianwei Zhang. "Cost functions to specify full-body motion and multi-goal manipulation tasks". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2018, pp. 3152–3159.

[132] Philipp Ruppel and Jianwei Zhang. "Learning object manipulation with dexterous hand-arm systems from human demonstration". In: *IEEE International Conference on Intelligent Robots and Systems (IROS)*. 2020, pp. 5417–5424.

[133] Lars Ruthotto and Eldad Haber. "An introduction to deep generative modeling". In: *GAMM-Mitteilungen* 44.2 (2021), e202100008.

[134] Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. "Dynamic routing between capsules". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2017, pp. 2672–2680.

[135] Mithileysh Sathiyanarayanan and Sharanya Rajan. "MYO Armband for physiotherapy healthcare: A case study using gesture recognition application". In: *International Conference on Communication Systems and Networks (COMSNETS)*. 2016, pp. 1–6.

[136] Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, Sergey Levine, and Google Brain. "Time-contrastive networks: Self-supervised learning from video". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2018, pp. 1134–1141.

[137] Pratyusha Sharma, Deepak Pathak, and Abhinav Gupta. "Third-person visual imitation learning via decoupled hierarchical controller". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2019, pp. 2593–2603.

[138] Thomas B Sheridan. *Telerobotics, automation, and human supervisory control*. MIT press, 1992.

[139] Aaron P Shon, Keith Grochow, and Rajesh PN Rao. "Robotic imitation from human motion capture using gaussian processes". In: *IEEE International Conference on Humanoid Robots (Humanoid)*. 2005, pp. 129–134.

[140] Miguel A Simao, Olivier Gibaru, and Pedro Neto. "Online recognition of incomplete gesture data to interface collaborative robots". In: *IEEE Transactions on Industrial Electronics* 66.12 (2019), pp. 9372–9382.

[141] Tomas Simon, Hanbyul Joo, Iain Matthews, and Yaser Sheikh. "Hand keypoint detection in single images using multiview bootstrapping". In: *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 1145–1153.

[142] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (2014).

[143] Laura Smith, Nikita Dhawan, Marvin Zhang, Pieter Abbeel, and Sergey Levine. "AVID: learning multi-stage tasks via pixel-level translation of human videos". In: *Robotics: Science and Systems (RSS)*. 2020.

[144] Nicolas Sommer and Aude Billard. "Multi-contact haptic exploration and grasping with tactile sensors". In: *Robotics and Autonomous Systems* 85 (2016), pp. 48–61.

[145] Adrian Spurr, Jie Song, Seonwook Park, and Otmar Hilliges. "Cross-modal deep variational hand pose estimation". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 89–98.

[146] Srinath Sridhar, Antti Oulasvirta, and Christian Theobalt. "Interactive markerless articulated hand motion tracking using RGB and depth data". In: *IEEE international conference on computer vision (ICCV)*. 2013, pp. 2456–2463.

[147] Christopher Stanton, Anton Bogdanovych, and Edward Ratanasena. "Teleoperation of a humanoid robot using full-body motion capture, example movements, and machine learning". In: *Australasian Conference on Robotics and Automation*. Vol. 8. 2012, pp. 51–60.

[148] Xiao Sun, Yichen Wei, Shuang Liang, Xiaoou Tang, and Jian Sun. "Cascaded hand pose regression". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 824–832.

[149] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going deeper with convolutions". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 1–9.

[150] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. "Rethinking the inception architecture for computer vision". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 2818–2826.

[151] Danhang Tang, Hyung Jin Chang, Alykhan Tejani, and Tae-Kyun Kim. "Latent regression forest: Structured estimation of 3D articulated hand posture". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014, pp. 3786–3793.

[152] Ajay Kumar Tanwani and Sylvain Calinon. "A generative model for intention recognition and manipulation assistance in teleoperation". In: *IEEE International Conference on Intelligent Robots and Systems (IROS)*. 2017, pp. 43–50.

[153] Jonathan Tompson, Murphy Stein, Yann Lecun, and Ken Perlin. "Real-time continuous pose recovery of human hands using convolutional networks". In: *ACM Transactions on Graphics (ToG)* 33.5 (2014), pp. 1–10.

[154] Luca Tonin, Felix Christian Bauer, and José del R Millán. "The role of the control framework for continuous teleoperation of a brain–machine interface-driven mobile robot". In: *IEEE Transactions on Robotics* 36.1 (2019), pp. 78–91.

[155] Ruben Villegas, Jimei Yang, Duygu Ceylan, and Honglak Lee. "Neural kinematic networks for unsupervised motion retargetting". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 8639–8648.

[156] Jorn Vogel, Daniel Leidner, Annette Hagengruber, Michael Panzirsch, Berthold Bauml, Maximilian Denninger, Ulrich Hillenbrand, Lioba Suchenwirth, Peter Schmaus, Marco Sewtz, et al. "An Ecosystem for Heterogeneous Robotic Assistants in Caregiving: Core Functionalities and Use Cases". In: *IEEE Robotics & Automation Magazine* 28.3 (2020), pp. 12–28.

[157] Dong Wei, Bidan Huang, and Qiang Li. "Multi-View Merging for Robot Teleoperation With Virtual Reality". In: *IEEE Robotics and Automation Letters* 6.4 (2021), pp. 8537–8544.

[158] Aaron Wetzler, Ron Slossberg, and Ron Kimmel. "Rule of thumb: deep derotation for improved fingertip detection". In: *British Machine Vision Conference (BMVC)*. 2015.

[159] David Whitney, Eric Rosen, Elizabeth Phillips, George Konidaris, and Stefanie Tellex. "Comparing robot grasping teleoperation across desktop and virtual reality with ROS reality". In: *Robotics Research*. Springer, 2020, pp. 335–350.

[160] Thomas Wimbock, Christian Ott, and Gerd Hirzinger. "Analysis and experimental evaluation of the intrinsically passive controller (IPC) for multifingered hands". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2008, pp. 278–284.

[161] Tytus Wojtara and Kenzo Nonami. "Hand posture detection by neural network and grasp mapping for a master slave hand system". In: *IEEE International Conference on Intelligent Robots and Systems (IROS)*. Vol. 1. 2004, pp. 866–871.

[162] Michael T Wolf, Christopher Assad, Matthew T Vernacchia, Joshua Fromm, and Henna L Jethani. "Gesture-based robot control with variable autonomy from the JPL BioSleeve". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2013, pp. 1160–1165.

[163] Chaozheng Wu, Jian Chen, Qiaoyu Cao, Jianchi Zhang, Yunxin Tai, Lin Sun, and Kui Jia. "Grasp Proposal Networks: An End-to-End Solution for Visual Learning of Robotic Grasps". In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 33. 2020, pp. 13174–13184.

[164] Christopher Xie, Yu Xiang, Arsalan Mousavian, and Dieter Fox. "Unseen object instance segmentation for robotic environments". In: *IEEE Transactions on Robotics* (2021), pp. 1–17.

[165] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. "Aggregated residual transformations for deep neural networks". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 1492–1500.

[166] Fu Xiong, Boshen Zhang, Yang Xiao, Zhiguo Cao, Taidong Yu, Joey Tianyi Zhou, and Junsong Yuan. "A2j: Anchor-to-joint regression network for 3D articulated pose estimation from a single depth image". In: *IEEE International Conference on Computer Vision (ICCV)*. 2019, pp. 793–802.

[167] Chenguang Yang, Gowrishankar Ganesh, Sami Haddadin, Sven Parusel, Alin Albu-Schaeffer, and Etienne Burdet. "Human-like adaptation of force and impedance in stable and unstable interactions". In: *IEEE transactions on Robotics* 27.5 (2011), pp. 918–930.

[168] Linlin Yang, Shile Li, Dongheui Lee, and Angela Yao. "Aligning latent spaces for 3D hand pose estimation". In: *IEEE International Conference on Computer Vision (ICCV)*. 2019, pp. 2335–2343.

[169] Alper Yilmaz, Omar Javed, and Mubarak Shah. "Object tracking: a survey". In: *ACM Computing Surveys* 38.4 (2006), pp. 13–57.

[170] Cheol-Hwan Yoo, Seowon Ji, Yong-Goo Shin, Seung-Wook Kim, and Sung-Jea Ko. "Fast and accurate 3D hand pose estimation via recurrent neural network for capturing hand articulations". In: *IEEE Access* 8 (2020), pp. 114010–114019.

[171] Shanxin Yuan, Guillermo Garcia-Hernando, Björn Stenger, Gyeongsik Moon, Ju Yong Chang, Kyoung Mu Lee, Pavlo Molchanov, Jan Kautz, Sina Honari, Liuhao Ge, et al. "Depth-based 3D hand pose estimation: From current achievements to future goals". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 2636–2645.

[172] Shanxin Yuan, Qi Ye, Björn Stenger, Siddhant Jain, and Tae-Kyun Kim. "Bighand2.2M benchmark: hand pose dataset and state of the art analysis". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 4866–4874.

[173] Mohammad Kassem Zein, Majd Al Aawar, Daniel Asmar, and Imad H Elhajj. "Deep learning and mixed reality to autocomplete teleoperation". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2021, pp. 4523–4529.

[174] Chao Zeng, Shuang Li, Zhaopeng Chen, Chenguang Yang, Sun Fuchun, and Jianwei Zhang. "Multi-fingered robot hand compliant manipulation based on vision-based demonstration and adaptive force control". In: *IEEE Transactions on Systems, Man and Cybernetics: Systems* (2021). Under Review.

[175] Chao Zeng, Shuang Li, Bin Fang, Chen Zhaopeng, and Jianwei Zhang. "Generalization of Robot Force-Relevant Skills through Adapting Compliant Profiles". In: *IEEE Robotics and Automation Letters* (2021). To appear, pp. 1–8.

[176] Chao Zeng, Shuang Li, Yiming Jiang, Qiang Li, Zhaopeng Chen, Chenguang Yang, and Jianwei Zhang. "Learning compliant grasping and manipulation by teleoperation with adaptive force control". In: *IEEE International Conference on Intelligent Robots and Systems (IROS)*. Prague, 2021, pp. 717–724.

[177] Chao Zeng, Hang Su, Yanan Li, Jing Guo, and Chenguang Yang. "An Approach for Robotic Leaning Inspired by Biomimetic Adaptive Control". In: *IEEE Transactions on Industrial Informatics* (2021).

[178] Heng Zhang, Zeming Zhao, Yang Yu, Kai Gui, Xinjun Sheng, and Xiangyang Zhu. "A feasibility study on an intuitive teleoperation system combining IMU with sEMG sensors". In: *International Conference on Intelligent Robotics and Applications (ICIRA)*. Springer. 2018, pp. 465–474.

[179] Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. "Deep imitation learning for complex manipulation tasks from virtual reality teleoperation". In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 5628–5635.

[180] Xingyi Zhou, Qingfu Wan, Wei Zhang, Xiangyang Xue, and Yichen Wei. "Model-based deep hand pose estimation". In: *International Joint Conference on Artificial Intelligence (IJCAI)*. 2016, pp. 2421–2427.

[181] Yidan Zhou, Jian Lu, Kuo Du, Xiangbo Lin, Yi Sun, and Xiaohong Ma. "Hbe: Hand branch ensemble network for real-time 3D hand pose estimation". In: *European Conference on Computer Vision (ECCV)*. 2018, pp. 501–516.

[182] Jun-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darrell, Alexei A Efros, Oliver Wang, and Eli Shechtman. "Toward multimodal image-to-image translation". In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2017, pp. 465–476.

[183] Katie Z Zhuang, Nicolas Sommer, Vincent Mendez, Saurav Aryan, Emanuele Formento, Edoardo D'Anna, Fiorenzo Artoni, Francesco Petrini, Giuseppe Granata, Giovanni Cannaviello, et al. "Shared human–robot proportional control of a dexterous myoelectric prosthesis". In: *Nature Machine Intelligence* 1 (2019), pp. 400–411.

[184] Christian Zimmermann and Thomas Brox. "Learning to estimate 3D hand pose from single RGB images". In: *IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 4903–4911.

List of Web-Adresses

[W1] Phantom Auto. *Phantom Auto website*. Accessed: October 2021. URL: https://phantom.auto/.

[W2] Manus Company. *OptiTrack website*. Accessed: September 2021. URL: https://www.manus-meta.com/optitrack.

[W3] Shadow Robot Company. *Shadow Dexterous Hand Technical Specification*. Accessed: September 2021. URL: https://www.shadowrobot.com/wp-content/uploads/shadow_dexterous_hand_technical_specification_E1_20130101.pdf.

[W4] Michael Ferguson. *Roserial wiki*. Accessed: September 2021. URL: http://wiki.ros.org/rosserial.

[W5] Willow Garage. *PR2 User Manual*. Accessed: September 2021. URL: https://www.clearpathrobotics.com/assets/downloads/pr2/pr2_manual_r321.pdf.

[W6] Gazebo. *Gazebo website*. Accessed: September 2021. URL: http://gazebosim.org/.

[W7] HaptX. *HaptX website*. Accessed: September 2021. URL: https://haptx.com/.

[W8] CyberGlove Systems Inc. *CyberGlove website*. Accessed: December 2021. URL: http://www.cyberglovesystems.com/cyberglove-ii/.

[W9] Intel. *Intel RealSense Camera SR300 website*. Accessed: September 2021. URL: https://ark.intel.com/content/www/us/en/ark/products/92329/intel-realsense-camera-sr300.html.

[W10] Microsoft Kinect. *Microsoft Kinect Wikipedia*. Accessed: September 2021. URL: https://en.wikipedia.org/wiki/Kinect.

[W11] Shuang Li. *Experimental Video of Compliant Teleoperation in Simulation*. Accessed: October 2021. URL: https://www.youtube.com/watch?v=xL9BvPGIKxE.

[W12] Shuang Li. *Experimental Video of the Multimodal Hand-Arm Teleoperation System*. Accessed: October 2021. URL: https://www.youtube.com/watch?v=rAj2IWl2ezs.

[W13] Shuang Li. *Perception Neuron Software Setup*. Accessed: October 2021. URL: https://git.crossmodal-learning.org/shuang.li/perception_neuron_ros-master.

[W14] Shuang Li. *TeachNet Github repository*. Accessed: October 2021. URL: https://github.com/TAMS-Group/TeachNet_Teleoperation.

[W15] Shuang Li. *Transteleop Github repository*. Accessed: October 2021. URL: https://smilels.github.io/multimodal-translation-teleop/.

[W16] Deutsches Zentrum für Luft- und Raumfahrt (DLR). *EDAN System*. Accessed: September 2021. URL: https://www.dlr.de/rm/en/desktopdefault.aspx/tabid-11670/20388_read-47709/.

[W17] Deutsches Zentrum für Luft- und Raumfahrt (DLR). *Force-Feedback joystick*. Accessed: October 2021. URL: https://www.dlr.de/rm/en/desktopdefault.aspx/tabid-3805/.

[W18] Deutsches Zentrum für Luft- und Raumfahrt (DLR). *HUG System*. Accessed: September 2021. URL: https://www.dlr.de/rm/desktopdefault.aspx/tabid-11704/#gallery/34276.

[W19] Deutsches Zentrum für Luft- und Raumfahrt (DLR). *ROKVISS*. Accessed: October 2021. URL: https://www.dlr.de/rm/en/desktopdefault.aspx/tabid-15723/#gallery/35441.

[W20] Deutsches Zentrum für Luft- und Raumfahrt (DLR). *Rollin' Justin System*. Accessed: September 2021. URL: https://www.dlr.de/rm/en/desktopdefault.aspx/tabid-11427/#gallery/35411.

[W21] Deutsches Zentrum für Luft- und Raumfahrt (DLR). *ROTEX*. Accessed: October 2021. URL: https://www.dlr.de/rm/en/desktopdefault.aspx/tabid-3827/5969_read-8744.

[W22] Microsoft Azure Kinect DK. *Azure Kinect DK website*. Accessed: September 2021. URL: https://azure.microsoft.com/en-us/services/kinect-dk/.

[W23] NDI. *NDI trakSTAR website*. Accessed: September 2021. URL: https://www.ndigital.com/products/3d-guidance/.

[W24] Noitom Perception Neuron. *Axis Neuron manual*. Accessed: September 2021. URL: https://neuronmocap.com/system/files/software/Axis%20Neuron%20User%20Manual_V3.8.1.5.pdf.

[W25] Noitom Perception Neuron. *Axis Neuron Software download*. Accessed: September 2021. URL: https://neuronmocap.com/content/axis-neuron.

[W26] Noitom Perception Neuron. *Perception Neuron website*. Accessed: September 2021. URL: https://neuronmocap.com/.

[W27] HANDS19 Challenge Organizers. *HANDS19 Challenge*. Accessed: September 2021. URL: https://sites.google.com/view/hands2019/challenge#h.p_adfpp7VAhgAL.

[W28] PhaseSpace. *PhaseSpace software*. Accessed: September 2021. URL: https://www.phasespace.com/software/.

[W29] PhaseSpace. *PhaseSpace website*. Accessed: September 2021. URL: https://www.phasespace.com/.

[W30] Mike Purvis. *Roserial_server wiki*. Accessed: September 2021. URL: http://wiki.ros.org/rosserial_server.

[W31] Shark Robotics. *Colossus website*. Accessed: October 2021. URL: https://robots.ieee.org/robots/colossus/.

[W32] SotfBank Robotics. *NAO Robot Website*. Accessed: September 2021. URL: https://www.softbankrobotics.com/emea/en/nao.

[W33] RViz. *RViz ROS Wiki*. Accessed: September 2021. URL: http://wiki.ros.org/rviz.

[W34] Shadow Robot Company. *Shadow Robot Company website*. Accessed: September 2021. URL: https://www.shadowrobot.com/dexterous-hand-series/.

[W35] Shadow Robot Company. *Shadow Robot github repository*. Accessed: October 2021. URL: https://github.com/shadow-robot.

[W36] Intuitive Surgical. *Intuitive Surgical website*. Accessed: December 2021. URL: https://www.davincisurgery.com/.

[W37] SynTouch Inc. *BioTac manual*. Accessed: September 2021. URL: https://syntouchinc.com/wp-content/uploads/2020/09/SynTouch-Product-Manual-BioTac-2020-09-23.pdf.

[W38] Istituto Italiano di Tecnologia. *iCub Robot Website*. Accessed: December 2021. URL: https://icub.iit.it/.

[W39] Alexande Tiderko. *Master_discovery_fkie*. Accessed: October 2021. URL: http://wiki.ros.org/master_discovery_fkie.

[W40] Alexande Tiderko. *Master_sync_fkie*. Accessed: October 2021. URL: http://wiki.ros.org/master_sync_fkie.

[W41] Global Times. *Smart throat-swabbing robot*. Accessed: October 2021. URL: https://www.globaltimes.cn/content/1182175.shtml.

[W42] Ultraleap. *Leap Motion Controller*. Accessed: October 2021. URL: https://www.ultraleap.com/product/leap-motion-controller/.

[W43] Ultraleap. *Ultraleap*. Accessed: September 2021. URL: https://www.ultraleap.com/.

[W44] University of Hamburg. *Crossmodal Learning (CML) project website*. Accessed: August 2019. URL: https://www.crossmodal-learning.org.

[W45] Vmware. *VMware Workstation Player download*. Accessed: September 2021. URL: https://www.vmware.com/de/products/workstation-player/workstation-player-evaluation.html.

List of Figures

1.1 Teleoperation applications 2
1.2 Aims of this thesis 4
1.3 Thesis structure 6

2.1 Overview of state-of-the-art of robotic teleoperation and data-driven 3D hand pose estimation 8
2.2 System architecture of the tactile telerobot 10
2.3 The teleoperation examples of the DexPilot system 12
2.4 Three control architectures in teleoperation 15
2.5 Different teleoperation control architectures in three robotic systems from DLR 16
2.6 Comparison of hand pose estimation and its similar fields 18
2.7 Human pose estimation by Openpoe 19
2.8 Common four input modalities in 3D hand pose estimation 21
2.9 Network architecture of DeepPrior++ 23
2.10 Schematic diagrams of augmented skeleton space transfer for depth-based hand pose estimation 24
2.11 The overall architecture of the HCRNN model 25

3.1 Visulization of human hand skeleton and its kinemtatic model 31
3.2 Visulization of Shadow dexterous hand and its kinematic chain 32
3.3 Pipeline for dataset generation 33
3.4 Example data from the BigHand2.2M dataset in 2D depth images and 3D point clouds 35
3.5 Gazebo environment and depth images of Shadow hand from nine viewpoints. . 36
3.6 Visualization of hand frame. 37
3.7 Examples of paired human-robot hand datasets 37
3.8 Angle distribution of THJ5, THJ4, FFJ3, MFJ3, RFJ3, and LFJ3 39
3.9 Visualization of hand frame 40

4.1 TeachNet-based teleoperation architecture 43
4.2 Architecture of TeachNet neural network 44
4.3 Architecture of the encoder module in TeachNet model 44
4.4 Architecture of the discriminator network D for soft consistency loss 46
4.5 Comparison of joint angle and distance accuracy between TeachNet and other baselines 48
4.6 Comparison of average angle error between TeachNet and other baselines . . . 49
4.7 Hand pose analysis from TeachNet. 50

4.8 The depth image of the human hand captured from different distances regarding the RealSense F200 camera . . . 51
4.9 Teleoperation results using the virtual Shadow hand with position control . . . 52
4.10 Screenshots of grasping experiments using TeachNet . . . 52
4.11 human status and objects in the in-hand grasp and release task . . . 53
4.12 Teleoperation examples on real-world robot. . . . 54

5.1 Translation results of an image-to-image translation model . . . 58
5.2 Architecture of GAN and Autoencoder model . . . 59
5.3 Architecture of the Transteleop model . . . 61
5.4 Architecture of a spatial transformer module . . . 62
5.5 Example images by an PCA-based rotation . . . 62
5.6 The heatmap of scaling matrix α . . . 64
5.7 Architecture of the GANteleop model . . . 66
5.8 Comparison of joint angle and distance accuracy between Transteleop and other baselines . . . 66
5.9 Comparison of average angle error between Transteleop and other baselines . . . 67
5.10 Example images generated by the STN preprocessing module in Transteleop . . . 67
5.11 Reconstructed robot images generated from Transteleop and GANteleop . . . 69
5.12 Transtelep analysis based on viewpoint . . . 70

6.1 Framework of the multimodal hand-arm teleoperation system . . . 74
6.2 Illustration of the neuron distribution on the human body . . . 75
6.3 Interface of Axis Neuron software in single-arm mode . . . 76
6.4 Data broadcast from Axis Neuron . . . 76
6.5 Illustration of the registration between the human right hand and the PR2 right robot arm . . . 77
6.6 Self-designed 3D-printed camera holder . . . 78
6.7 Screenshots of pick-a-Pringles-can-in-a-bowl task . . . 79
6.8 Screenshots of pick-a-wooden-cube-on-a-brick task . . . 79
6.9 Screenshots of cup-inserting task . . . 79
6.10 Screenshots of pushing task . . . 80
6.11 Screenshots of handover task . . . 81

7.1 Pipeline of the dexterous hand-arm teleoperation system with active vision. . . . 84
7.2 Potential hardware choices of arm tracking . . . 85
7.3 Wring scheme of the customized wrist marker . . . 86
7.4 Hardware setup of the dexterous hand-arm teleoperation system with active vision 87
7.5 Front view of the hardware at the local site . . . 88
7.6 Illustration of teleoperation working environment from the top view . . . 88
7.7 Illustration of how to register the PhaseSpace system to the robot system . . . 89
7.8 Camera holder in active vision system . . . 89
7.9 Visualization of software architecture . . . 90
7.10 Visualization of the UR5 workspace and PR2 workspace . . . 91
7.11 Overall usage of the PhaseSpace system . . . 92
7.12 Trajectory analysis of the UR5 and the right PR2 arm . . . 94

7.13 Trajectory analysis of the right PR2 arm with different starting position and weight of the regularization goal . . . 95
7.14 Screenshots of pick and place task and tower building task . . . 96
7.15 Example scenes in the pick and place task and tower building task . . . 96
7.16 Screenshots of pouring task . . . 97
7.17 Screenshots of sweeping task . . . 98
7.18 Screenshots of fader sliding task . . . 98
7.19 Screenshots of picking-and-placing task and tower-building task . . . 98

8.1 Pipeline of compliant teleoperation by adaptive force control . . . 102
8.2 The initial and final configurations during the door opening task. . . . 106
8.3 Angle changes and position distribution of the contact point in the door opening task . . . 107
8.4 Screenshots of the cap turning task . . . 107
8.5 The position distribution of the contact points in cap turning task . . . 108
8.6 The screenshots of the mouse touching experiments . . . 108
8.7 The contacting analysis in the mouse touching experiments . . . 109
8.8 Grasping task under two control modes . . . 111
8.9 Illustration of grasping a soft plastic cup under two control modes . . . 111
8.10 Screenshots of the pouring-screws-into-a-cup task . . . 111
8.11 Real-time joint angles and effort commands of joint THJ4 and FFJ4 in screw pouring task . . . 112
8.12 Screenshots of the cap opening task . . . 112

9.1 Schematic diagram of thesis conclusion . . . 116

A.1 Time diagram of important CNN architectures . . . 119
A.2 LeNet-5 architecture . . . 120
A.3 Convolution operation . . . 120
A.4 Visualization of the four nonlinear activation functions Sigmoid, Tanh, ReLU, and Leaky ReLU . . . 122
A.5 A building block in a residual neural network . . . 124
A.6 Architecture of the grasp quality convolutional neural network GQ-CNN . . . 125
A.7 Architecture of the A2J model in an end-to-end manner . . . 126

List of Tables

2.1 Comparison of existing 3D hand pose estimation datasets 20

3.1 Joint ranges of human hand proposed by [28] and Shadow hand with BioTac sensor 32

4.1 Accuracy under high-precision conditions 49
4.2 Average time (s) a novice took to grasp and release an object 53

5.1 Angle/distance accuracy under high-precision conditions and average angle/distance error 68
5.2 The average number of occluded joints 70

6.1 Average completion time and success rate of each task in the multimodal teleoperation system 81

7.1 Average completion time and success rate of each task in the hand-arm teleoperation system with the active vision system 99

8.1 The force control strategy for robot compliant teleoperation 105

Acknowledgements

Eidesstattliche Erklärung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Dissertationsschrift selbst verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Ich versichere weiterhin, dass ich die Arbeit vorher nicht in einem anderen Prüfungsverfahren eingereicht habe und die eingereichte schriftliche Fassung der Fassung auf dem elektronischen Speichermedium entspricht.

Ort, Datum

Unterschrift