UNIVERSITY OF CALIFORNIA SAN DIEGO

Learning Generalizable Dexterous Manipulation

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

in

Electrical Engineering (Intelligent Systems, Robotics, and Control)

by

Yuzhe Qin

Committee in charge:

Professor Hao Su, Co-Chair
Professor Xiaolong Wang, Co-Chair
Professor Nikolay Atanasov
Professor Henrik Christensen
Professor Michael Tolley

2024

University of California San Diego

2024

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my advisors, Prof. Xiaolong Wang and Prof. Hao Su, for their invaluable guidance and support throughout my PhD journey.

My sincere thanks to Prof. Su, who has been a constant source of inspiration and mentorship since my master's studies. Your guidance ignited my initial interest in robot learning, laying the foundation for my research pursuits. I am particularly grateful to Prof. Wang for his unwavering support and mentorship during my PhD. As one of the first PhD students in your lab, I deeply appreciate the time and effort you invested in my development as an independent researcher. Your insights on research topics and advice on career development were instrumental in shaping my dissertation. I am incredibly fortunate to have had the opportunity to learn from both of you. Your combined expertise and unique perspectives have enriched my research experience and helped me grow as a scholar.

I am profoundly grateful to my PhD committee members, Prof. Nikolay Atanasov, Prof. Henrik Christensen, and Prof. Michael Tolley, for their insightful contributions and unwavering support throughout this journey. Their guidance has been invaluable.

I extend my heartfelt thanks to my collaborators, including colleagues and friends at institutions like Google Everyday Robots and NVIDIA's Seattle Robotics Lab. Their camaraderie, inspiration, and shared wisdom significantly enriched my research experience. I especially want to acknowledge Runyu Ding, Jiyue Zhu, Jun Wang, Ying Yuan, Haichuan Che, Binghao Huang, Yueh-Hua Wu, and Rui Chen, with whom I had the pleasure of co-leading research projects and sharing many insightful discussions. I am also immensely grateful to my labmates, Fanbo Xiang, Jiayuan Gu, Tongzhou Mu, Chengzhe Jia, Ruihan Yang, and Yang Fu. The discussions we shared were always insightful and incredibly helpful for my research. I want to express my sincere gratitude to Yu-Wei Chao, Wei Yang, Dieter Fox, Daniel Ho, and Kaichun Mo, whose mentorship extended far beyond UC San Diego. Their guidance was not only helpful but essential to my success.

Finally, my deepest thanks go to my mother and father. Words cannot express how grateful

I am for your unwavering and unconditional love. You have been my guiding light through every challenge, the foundation of my resilience, and the source of my hope and perseverance. Thank you for always believing in me, even when the journey felt like a marathon with no end in sight.

Chapter 2, in full, is a reprint of the material published in the 2023 Robotics: Science and Systems (RSS): “AnyTeleop: A General Vision-Based Dexterous Robot Arm-Hand Teleoperation System” (Yuzhe Qin, Wei Yang, Binghao Huang, Karl Van Wyk. Hao Su, Xiaolong Wang, Yu-Wei Chao, Dieter Fox). The dissertation author was the primary investigator and author of both papers.

Chapter 3, in full, is a reprint of the material published in the 2024 Conference on Computer Vision and Pattern Recognition (CVPR): “CyberDemo: Augmenting Simulated Human Demonstration for Real-World Dexterous Manipulation” (Jun Wang*, Yuzhe Qin*, Kaiming Kuang, Yigit Korkmaz, Akhilan Gurumoorthy, Hao Su, Xiaolong Wang). The dissertation author was the project co-lead and author of this paper.

Chapter 4, in full, is a reprint of the material published in the 2022 Conference on Robot Learning (CoRL): “DexPoint: Generalizable Point Cloud Reinforcement Learning for Sim-to-Real Dexterous Manipulation” (Yuzhe Qin*, Binghao Huang*, Zhao-Heng Yin, Hao Su, Xiaolong Wang). The dissertation author was the primary investigator and author of this paper.

VITA

2014-2018	Bachelor of Science in Mechanical Engineering, Shanghai Jiao Tong University
2018-2020	Master of Science in Electrical Engineering (Intelligent Systems, Robotics, and Control), University of California San Diego
2020-2024	Doctor of Philosophy in Electrical Engineering (Intelligent Systems, Robotics, and Control), University of California San Diego

PUBLICATIONS

[1] Jun Wang, Yuzhe Qin, Kaiming Kuang, Yigit Korkmaz, Akhilan Gurumoorthy, Hao Su, and Xiaolong Wang. Cyberdemo: Augmenting simulated human demonstration for real-world dexterous manipulation. *arXiv preprint arXiv:2402.14795*, 2024

[2] Ying Yuan, Haichuan Che, Yuzhe Qin, Binghao Huang, Zhao-Heng Yin, Kang-Won Lee, Yi Wu, Soo-Chul Lim, and Xiaolong Wang. Robot synesthesia: In-hand manipulation with visuotactile sensing. *arXiv preprint arXiv:2312.01853*, 2023

[3] Entong Su, Chengzhe Jia, Yuzhe Qin, Wenxuan Zhou, Annabella Macaluso, Binghao Huang, and Xiaolong Wang. Sim2real manipulation on unknown objects with tactile-based reinforcement learning. *arXiv preprint arXiv:2403.12170*, 2024

[4] Lirui Wang, Yiyang Ling, Zhecheng Yuan, Mohit Shridhar, Chen Bao, Yuzhe Qin, Bailin Wang, Huazhe Xu, and Xiaolong Wang. Gensim: Generating robotic simulation tasks via large language models. *arXiv preprint arXiv:2310.01361*, 2023

[5] Zehao Zhu, Jiashun Wang, Yuzhe Qin, Deqing Sun, Varun Jampani, and Xiaolong Wang. Contactart: Learning 3d interaction priors for category-level articulated object and hand poses estimation. *arXiv preprint arXiv:2305.01618*, 2023

[6] Yuzhe Qin, Wei Yang, Binghao Huang, Karl Van Wyk, Hao Su, Xiaolong Wang, Yu-Wei Chao, and Dietor Fox. Anyteleop: A general vision-based dexterous robot arm-hand teleoperation system. *arXiv preprint arXiv:2307.04577*, 2023

[7] Binghao Huang, Yuanpei Chen, Tianyu Wang, Yuzhe Qin, Yaodong Yang, Nikolay Atanasov, and Xiaolong Wang. Dynamic handover: Throw and catch with bimanual hands. *arXiv preprint arXiv:2309.05655*, 2023

[8] Pengwei Xie, Rui Chen, Siang Chen, Yuzhe Qin, Fanbo Xiang, Tianyu Sun, Jing Xu, Guijin Wang, and Hao Su. Part-guided 3d rl for sim2real articulated object manipulation. *IEEE Robotics and Automation Letters*, 2023

[9] Zhao-Heng Yin, Binghao Huang, Yuzhe Qin, Qifeng Chen, and Xiaolong Wang. Rotating without seeing: Towards in-hand dexterity through touch. *arXiv preprint arXiv:2303.10880*, 2023

[10] Linghao Chen, Yuzhe Qin, Xiaowei Zhou, and Hao Su. Easyhec: Accurate and automatic hand-eye calibration via differentiable rendering and space exploration. *IEEE Robotics and Automation Letters*, 2023

[11] Chen Bao, Helin Xu, Yuzhe Qin, and Xiaolong Wang. Dexart: Benchmarking generalizable dexterous manipulation with articulated objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21190–21200, 2023

[12] Stone Tao, Xiaochen Li, Tongzhou Mu, Zhiao Huang, Yuzhe Qin, and Hao Su. to-executable trajectory translation for one-shot task generalization. *arXiv preprint arXiv:2210.07658*, 2022

[13] Jianglong Ye, Jiashun Wang, Binghao Huang, Yuzhe Qin, and Xiaolong Wang. Learning continuous grasping function with a dexterous hand from human demonstrations. *IEEE Robotics and Automation Letters*, 8(5):2882–2889, 2023

[14] Luobin Wang, Runlin Guo, Quan Vuong, Yuzhe Qin, Hao Su, and Henrik Christensen. A real2sim2real method for robust object grasping with neural surface reconstruction. In *2023 IEEE 19th International Conference on Automation Science and Engineering (CASE)*, pages 1–8. IEEE, 2023

[15] Xiaoshuai Zhang, Rui Chen, Ang Li, Fanbo Xiang, Yuzhe Qin, Jiayuan Gu, Zhan Ling, Minghua Liu, Peiyu Zeng, Songfang Han, et al. Close the optical sensing domain gap by physics-grounded active stereo sensor simulation. *IEEE Transactions on Robotics*, 2023

[16] Yuzhe Qin, Binghao Huang, Zhao-Heng Yin, Hao Su, and Xiaolong Wang. Dexpoint: Generalizable point cloud reinforcement learning for sim-to-real dexterous manipulation. In *Conference on Robot Learning*, pages 594–605. PMLR, 2023

[17] Chieko Sarah Imai, Minghao Zhang, Yuchen Zhang, Marcin Kierebiński, Ruihan Yang, Yuzhe Qin, and Xiaolong Wang. Vision-guided quadrupedal locomotion in the wild with multi-modal delay randomization. In *2022 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 5556–5563. IEEE, 2022

[18] Yuzhe Qin, Yueh-Hua Wu, Shaowei Liu, Hanwen Jiang, Ruihan Yang, Yang Fu, and Xiaolong Wang. Dexmv: Imitation learning for dexterous manipulation from human videos. In *European Conference on Computer Vision*, pages 570–587. Springer, 2022

[19] Yuzhe Qin, Hao Su, and Xiaolong Wang. From one hand to multiple hands: Imitation learning for dexterous manipulation from single-camera teleoperation. *RA-L*, 7(4):10873–10881, 2022

[20] Kaichun Mo, Yuzhe Qin, Fanbo Xiang, Hao Su, and Leonidas Guibas. O2o-afford: Annotation-free large-scale object-object affordance learning. In *Conference on robot learning*, pages 1666–1677. PMLR, 2022

[21] Ziyuan Liu, Wei Liu, Yuzhe Qin, Fanbo Xiang, Minghao Gou, Songyan Xin, Maximo A Roa, Berk Calli, Hao Su, Yu Sun, et al. Ocrtoc: A cloud-based competition and benchmark for robotic grasping and manipulation. *IEEE Robotics and Automation Letters*, 7(1):486–493, 2021

[22] Yijia Weng, He Wang, Qiang Zhou, Yuzhe Qin, Yueqi Duan, Qingnan Fan, Baoquan Chen, Hao Su, and Leonidas J Guibas. Captra: Category-level pose tracking for rigid and articulated objects from point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13209–13218, 2021

[23] Ahmed H Qureshi, Jacob J Johnson, Yuzhe Qin, Taylor Henderson, Byron Boots, and Michael C Yip. Composing task-agnostic policies with deep reinforcement learning. *arXiv preprint arXiv:1905.10681*, 2019

[24] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, et al. Sapien: A simulated part-based interactive environment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11097–11107, 2020

[25] Yuzhe Qin, Rui Chen, Hao Zhu, Meng Song, Jing Xu, and Hao Su. S4g: Amodal single-view single-shot se (3) grasp detection in cluttered scenes. In *Conference on robot learning*, pages 53–65. PMLR, 2020

ABSTRACT OF THE DISSERTATION

Learning Generalizable Dexterous Manipulation

by

Yuzhe Qin

Doctor of Philosophy in Electrical Engineering (Intelligent Systems, Robotics, and Control)

University of California San Diego, 2024

Professor Hao Su, Co-Chair
Professor Xiaolong Wang, Co-Chair

Dexterous manipulation using multi-fingered robotic hands is a crucial area in robotics, aimed at performing intricate tasks with various objects in everyday environments. However, this field presents significant challenges. Modeling the complex contact patterns between a dexterous hand and manipulated objects is difficult, hindering the effectiveness of model-based control methods. Furthermore, the high number of Degrees of Freedom (DoF) in the hand's joints, dramatically increases the complexity of training data-driven policies for dexterous manipulation.

This dissertation addresses the challenging task of learning highly generalizable dexterous manipulation skills applicable across diverse scenarios. We investigate two principal directions to enhance the learning capabilities of dexterous manipulation.

First, we leverage the inherent structural similarities between human and robotic hands, employing human data to guide robot manipulation skills. This approach is motivated by the bio-inspired design of dexterous hands, which offers a unique opportunity to learn from human demonstrations. To facilitate efficient data collection, we develop AnyTeleop, a general vision-based teleoperation system for dexterous robot arm-hand systems. AnyTeleop utilizes readily available devices like web cameras to provide a versatile interface for teleoperating various arm-hand systems. Furthermore, we introduce CyberDemo, a data augmentation technique that expands the original human demonstrations, generating a dataset hundreds of times larger than the initial set. This approach allows for training policies capable of handling a wider range of scenarios without requiring additional human effort.

Second, we explore the potential of using vast amounts of simulated data to learn dexterous manipulation policies. The primary challenge in this direction lies in bridging the domain gap between simulation and the real world, encompassing both dynamics and visual discrepancies. This sim2real gap is particularly pronounced for high DoF dexterous hands. To address this, we propose a sim-to-real reinforcement learning framework, DexPoint, that leverages point cloud and proprioceptive data. This framework integrates multi-modal sensory information into a unified 3D space, preserving the spatial relationships between robot components, sensors, and manipulated objects. This unified representation enables faster policy learning in simulation and smoother transfer to real-world applications.

Chapter 1

Introduction

1.1 Introduction

1.1.1 Generalizable Dexterous Manipulation

Dexterous manipulation is one of the most challenging and important problems in robotics. It aims to enable robots to perform intricate tasks with various objects in everyday environments [99, 21, 2, 26, 29, 8]. Multi-fingered robotic hands, designed to emulate the remarkable capabilities of the human hand, play a pivotal role in achieving this goal [119, 127, 134, 120, 130]. However, the complexity of dexterous manipulation poses significant challenges in both modeling and control.

One of the primary challenges in dexterous manipulation is modeling the intricate contact patterns between the robotic hand and manipulated objects. The high number of contact points and the dynamic nature of these interactions make it difficult to develop accurate and computationally efficient models [20, 23, 70, 91]. Consequently, model-based control methods often struggle to achieve the desired level of dexterity and adaptability.

Another major challenge stems from the high-dimensional action spaces associated with multi-fingered hands. A large number of Degrees of Freedom (DoF) in the hand's joints dramatically increases the complexity of training data-driven policies for dexterous manipulation [13, 100, 21]. This complexity hinders the learning process and makes it challenging to develop policies that can generalize across diverse tasks and environments.

1.1.2 Overview of Techniques and Contributions

This dissertation aims to address the challenge of learning highly generalizable dexterous manipulation skills that can be applied across a wide range of scenarios. The main objectives of this research are:

1. To leverage the structural similarities between human and robotic hands, employing human data to guide robot manipulation skills.
2. To explore the potential of using vast amounts of simulated data to learn dexterous manipulation policies while bridging the domain gap between simulation and the real world.

By pursuing these objectives, we seek to develop novel approaches that enhance the learning capabilities of dexterous manipulation systems, enabling them to perform complex tasks with increased efficiency and adaptability.

Human-Inspired Learning for Dexterous Manipulation

The bio-inspired design of dexterous robotic hands offers a unique opportunity to learn from human demonstrations. By leveraging the inherent structural similarities between human and robotic hands, we can guide the development of robot manipulation skills using human data.

AnyTeleop: A General Vision-Based Teleoperation System To facilitate efficient data collection for human-inspired learning, we introduce AnyTeleop, a versatile vision-based teleoperation system for dexterous robot arm-hand systems. AnyTeleop utilizes readily available devices, such as web cameras, to provide an intuitive interface for teleoperating various arm-hand systems. This approach eliminates the need for specialized hardware and enables the collection of diverse human demonstration data across different environments and tasks.

CyberDemo: Data Augmentation for Human Demonstrations While human demonstrations provide valuable insights into dexterous manipulation strategies, collecting large amounts of human data can be time-consuming and resource-intensive. To address this challenge, we propose CyberDemo, a data augmentation technique that expands the original human demonstrations. By applying various transformations and perturbations to the collected data, CyberDemo generates a dataset that is hundreds of times larger than the initial set. This augmented dataset enables the training of policies that can handle a wider range of scenarios without requiring additional human effort.

Simulator-Based Learning for Dexterous Manipulation

In addition to human-inspired learning, we explore the potential of using vast amounts of simulated data to learn dexterous manipulation policies. Simulated environments offer the advantage of generating large-scale datasets without the constraints and costs associated with real-world data collection. However, the primary challenge in this approach lies in bridging the domain gap between simulation and the real world.

The Sim2Real Gap in Dexterous Manipulation The domain gap between simulation and the real world, known as the sim2real gap, poses a significant challenge in learning dexterous manipulation policies. This gap encompasses both dynamics and visual discrepancies, which are particularly pronounced for high DoF dexterous hands [161, 163, 26]. The complex contact dynamics and the intricate interactions between the hand and objects are difficult to model accurately in simulation, leading to discrepancies in the learned policies when transferred to the real world.

DexPoint: A Sim-to-Real Reinforcement Learning Framework To address the sim2real gap, we propose DexPoint, a sim-to-real reinforcement learning framework that leverages point cloud and proprioceptive data. DexPoint integrates multi-modal sensory information into a unified 3D space, preserving the spatial relationships between robot components, sensors, and manipulated objects. By representing the environment in this unified manner, DexPoint enables faster policy learning in simulation and smoother transfer to real-world applications.

1.1.3 Contributions and Organization

The main contributions of this dissertation are:

1. The development of AnyTeleop, a general vision-based teleoperation system for efficient human demonstration data collection.

2. The introduction of CyberDemo, a data augmentation technique that significantly expands human demonstration datasets for improved policy learning.

3. The proposal of DexPoint, a sim-to-real reinforcement learning framework that leverages point cloud and proprioceptive data to bridge the domain gap between simulation and the real world.

The remainder of this dissertation is organized as follows: Chapter 2 presents AnyTeleop, detailing the system design, data collection process, and experimental results. Chapter 3 introduces CyberDemo, a novel approach to robotic imitation learning that capitalizes on simulated human demonstrations to master real-world tasks. Chapter 4 presents DexPoint, describing the framework architecture, policy learning approach, and evaluation in both simulated and real-world environments. Finally, Chapter 5 discusses the implications of the research findings, highlights potential future directions, and concludes the dissertation.

Through the development and evaluation of these novel approaches, this dissertation contributes to advancing the field of dexterous manipulation in robotics, paving the way for more adaptable, efficient, and generalizable robotic systems capable of performing complex tasks in everyday environments.

Chapter 2

Collecting Dexterous Manipulation Data with Human Teleoperation

The quest to endow robots with human-like dexterity hinges critically on the quality and diversity of the data used to train them. Dexterous manipulation, involving nuanced interactions between robotic hands and a variety of objects, requires comprehensive datasets that capture a wide range of motions, tactile sensations, and object manipulations under different conditions. However, collecting such detailed and multifaceted data presents significant challenges, particularly when using dexterous robotic hands that mimic the complexity of human hands. Teleoperation systems, which enable human operators to control robots remotely, offer a promising avenue for data collection but often struggle with issues of latency, precision, and the fidelity of sensory feedback, complicating the task of capturing realistic, high-quality manipulation data.

Vision-based teleoperation has emerged as a transformative approach in robotics, enabling systems to mimic human-level dexterity and intelligence in physical interactions with diverse environments. This capability is primarily facilitated by inexpensive camera sensors, making sophisticated robotic manipulation more accessible. Despite these advances, the current landscape of vision-based teleoperation systems faces significant limitations. Most systems are intricately tailored to specific robot models and deployment environments. This customization results in poor scalability, becoming increasingly impractical as the diversity of robot models and operational contexts continues to grow.

In this chapter, we introduce AnyTeleop, a novel teleoperation system characterized by its universality and versatility. AnyTeleop is designed to seamlessly support a wide array of robotic arms, hands, realities (both simulated and real), and camera configurations within a singular, cohesive framework. This system is not only flexible in accommodating various simulators and real hardware but also demonstrates robust performance across these platforms. In real-world applications, AnyTeleop surpasses the capabilities of prior systems engineered for specific robot hardware, achieving higher success rates with the same robotic setups. Additionally, in simulated environments, AnyTeleop outperforms specialized systems, leading to enhanced outcomes in imitation learning tasks. Through these advancements, AnyTeleop sets a new standard for adaptable, high-performance vision-based teleoperation systems, promising significant contributions to the field of robotic manipulation.

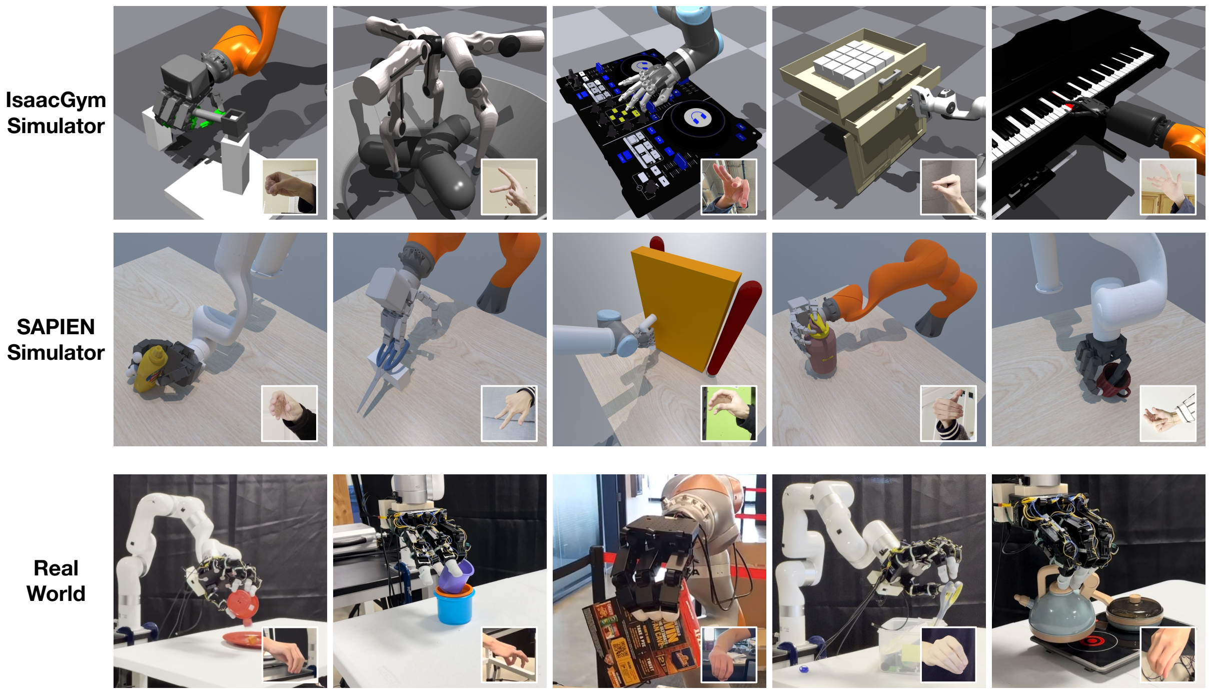


Figure 2.1. AnyTeleop Overview. We present AnyTeleop, a vision-based teleoperation system for a variety of scenarios to solve a wide range of manipulation tasks. AnyTeleop can be used for various robot arms with different robot hands. It also supports teleoperation within different realities, such as IsaacGym simulator (top row), SAPIEN simulator (middle row), and real-world (bottom rows).

2.1 Introduction

A grand goal of robotics is to endow robots with human-level intelligence to physically interact with the environment. Teleoperation [98], as a direct means to acquire human demonstrations for teaching robots, has been a powerful paradigm to approach this goal [65, 40, 171, 53, 85, 25, 60, 7, 94, 133, 159]. Compared to gripper-based manipulators, teleoperating dexterous hand-arm systems poses unprecedented challenges and often requires specialized apparatus that comes with high costs and setup efforts, such as Virtual Reality (VR) devices [5, 53, 45], wearable gloves [76, 77], handheld controller [115, 116, 64], haptic sensors [42, 69, 123, 139], or motion capture trackers [174]. Fortunately, recent developments in vision-based teleoperation [3, 73, 49, 72, 109, 74, 66, 65, 4] have provided a low-cost and more generalizable alternative for teleoperating dexterous robot systems.

Despite the progress, the current paradigm of vision-based teleoperation systems still falls short when it comes to scaling up data collection for robot teaching. First, prior systems are often designed and engineered toward a particular robot model or deployment environment. For example, some systems rely on vision-based hand tracking models trained on datasets collected in the deployed studio [73, 49], and some rely on human-robot retargeting models [170, 43] or collision avoidance models [138] trained for the particular robot at use. These systems will scale poorly as the pool of robot models expands and the variety of operating environments increases. Second, each system is created and coupled with one specific "reality", either only in the real world or with a particular choice of simulators. For example, the HAPTIX [69] motion capture system is only developed for teleoperation in MuJoCo-based environments [147]. To facilitate large-scale data collection with simulation as well as closing sim-to-real gaps, we need teleoperation systems to operate both in virtual (with arbitrary choices of simulators) and in the real world. Finally, existing teleoperation systems are often tailored for single-operator and single-robot settings. To teach robots how to collaborate with other robot agents as well as with human agents, a teleoperation system should be designed to support multiple pilot-robot partners where the robots can physically interact with each other in a shared environment.

In this dissertation, we aim to set the foundation for scaling up data collection with vision-based dexterous teleoperation, by filling in the aforementioned gaps. To this end, we propose *AnyTeleop*, a unified and general teleoperation system (Fig. 2.1), which can be used for:

- *Diverse* robot arm and dexterous hand models;
- *Diverse* realities, i.e. different choices of simulators or the real world;
- Teleoperation from *diverse* geographic locations, via a browser-based web visualizer developed for remote visual feedback;
- *Diverse* camera configurations, e.g. RGB camera with or without depth, single or multiple cameras;
- *Diverse* operator-robot partnerships, e.g. two operators separately piloting two robots to collaboratively solve a manipulation task.

To achieve this goal, we first develop a general and high-performance motion retargeting library to translate human motion to robot motion in real time without learned models. Our collision avoidance module is also learning-free and powered by CUDA-based geometry queries. They can adapt to new robots given only the kinematic model, i.e., URDF files. Second, we develop a web-based viewer compatible with standard browsers, to achieve simulator-agnostic visualization and enable remote teleoperation across the internet. Third, we define a general software interface for visual-based teleoperation, which standardizes and decouples each module inside the teleoperation system. It enables smooth deployment on different simulators or real hardware.

While being very general to support many settings with a single system, our system can still achieve great performance in the experiments. For real-world teleoperation, AnyTeleop can outperform a previous system [138] designed for specific robot hardware **with higher**

Table 2.1. Comparison of Vision-Based Teleoperation System. We compare AnyTeleop's capabilities with related visual teleoperation systems for multi-fingered dexterous robots. "Calib free" means extrinsic calibration is not needed, "No Arm" in the column of "Multiple Arms" means this system can only control hand motion but not arm-hand systems.

	Sensor		**Robot**			**Comminication**	
	Calib Free	Contact Free	Multi Arms	Multi Hands	Collision Free	Remote Teleop	Colab Teleop
DexPilot [49]	✗	✓	✗	✗	✓	✗	✗
Holo-Dex [5]	✓	✓	No Arm	✗	✗	✓	✗
DIME [7]	✗	✓	No Arm	✗	✗	✓	✗
TeachNet [73]	✓	✓	No Arm	✗	✗	✗	✗
Telekinesis [138]	✓	✗	✗	✗	✓	✓	✗
Qin *et al.* [109]	✓	✓	No Arm	✓	✗	✓	✗
MVP-Real [113]	✗	✗	No Arm	✗	✗	✓	✗
Transteleop [72]	✗	✗	✗	✗	✓	✓	✗
Mosbach *et al.* [92]	✗	✗	✗	✗	✗	✓	✗
AnyTeleop	✓	✓	✓	✓	✓	✓	✓

success rates on 8 out of 10 tasks proposed in the previous method, using the same robot as [138]. For simulated environment teleportation, the smoother and collision-free demonstrations collected by AnyTeleop can bring better imitation learning results **with higher success rates on 5 out of 6 tasks** proposed in their paper, compared with a previous system [109] specifically designed for that simulator. Finally, we demonstrate that AnyTeleop can be extended to support collaborative manipulation, which to our best knowledge has neither been achieved in the literature of vision-based teleoperation nor on dexterous hands.

Our system is also packaged to be easily deployable. The containerized design makes installation easy and frees users from handling software dependencies.

2.2 Teleoperation System Overview

Fig. 2.2 illustrates our proposed paradigms of vision-based teleoperation systems. Below we introduce the features and designs of our system which realize the paradigms.

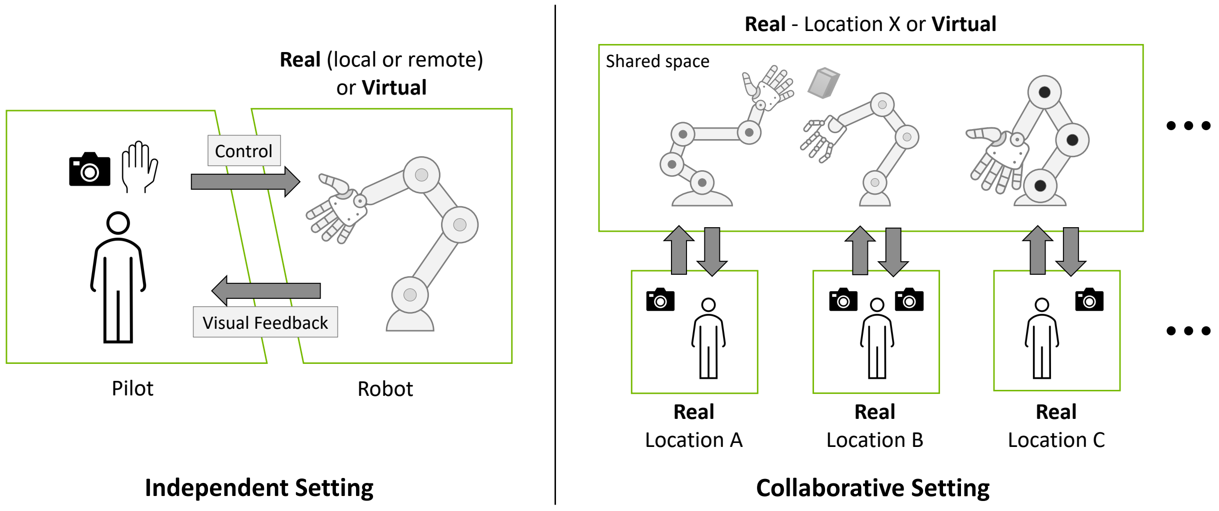


Figure 2.2. Vision-based Teleoperation. Paradigms of vision-based teleoperation systems in *independent* and *collaborative* settings. The system should support any arm-hand models, existed in either virtual or the real world, can operate with flexible camera configurations, provide visual feedback for both local or remote presence, and support multiple robots piloted in a shared space.

2.2.1 System Features

1. **Any arm-hand.** As shown in Fig. 2.1, AnyTeleop is designed for arbitrary dexterous arm-hand systems that are not limited to any specific robot type.
2. **Any reality.** AnyTeleop is decoupled from specific hardware drivers or physics simulators. It can support different realities as visualized in Fig. 2.1.
3. **Anywhere remote teleoperation.** AnyTeleop provides a web-based visualizer to monitor the teleoperation and simulation in standard web browsers, e.g. Chrome.
4. **Any camera configuration.** AnyTeleop can consume data from both RGB and RGB-D cameras, and from either single or multiple cameras. Most importantly, it does not require extrinsic calibration as in most previous systems. This allows more flexible camera configurations and lower deployment overhead.
5. **Any number of operator-robot partnerships**. AnyTeleop supports collaborative settings where operators separately pilot two robots to collaboratively solve a manipulation task.

6. **Simple deployment.** AnyTeleop and all libraries are encapsulated as a docker image that can be downloaded and deployed on any Linux machine, which frees users from handling troublesome dependencies.

We compare AnyTeleop with other vision-based dexterous teleoperation systems. We compare the systems in three dimensions: (i) sensor requirements; (ii) robot-related support; (iii) afforded use cases. Among all these teleoperation systems, AnyTeleop is the only one which can support different robot arms and enable collaborative teleoperation. It is also one of the only two systems that can support different dexterous hands.

2.2.2 System Design

The architecture of the teleoperation system is shown in Fig. 2.3. The teleoperation server (Section 2.3) receives the camera stream from the driver, detects the hand pose, and then converts it to joint control commands. The client receives these commands via network communication and uses them to control a simulated or real robot. The system is designed with three key principles: modularity, communication-focused, and containerization. Modularity is achieved by implementing well-defined input-output interfaces for each sub-component, allowing for wide applicability to different robot arms, dexterous hands, cameras, and realities. Communication-focused design allows for remote and collaborative teleoperation and reduces computation requirements on the operator's side by deploying heavy computations on a powerful server. Finally, the containerized design makes installation and deployment easier compared to other robotics systems with heavy software dependencies.

2.3 Teleoperation Server

The teleoperation server, outlined in Section 2.2, utilizes the RGB or RGB-D data from one or multiple cameras and generates smooth and collision-free control commands for the robot arm and dexterous hand. It consists of four modules: (i) the hand pose detection module, which

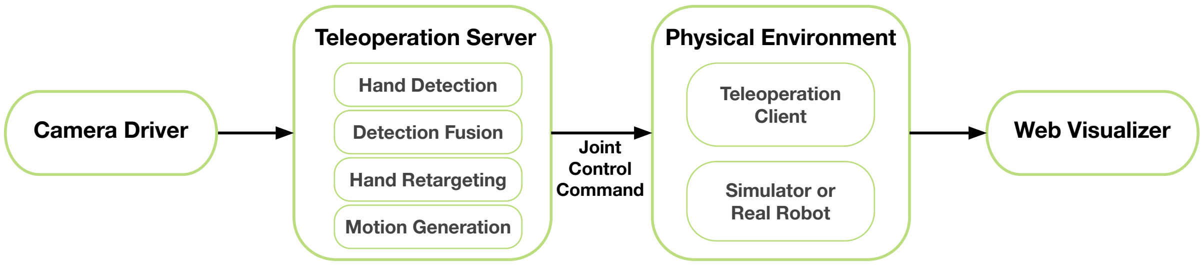


Figure 2.3. System Architecture. AnyTeleop is composed of four components: (i) camera driver, which captures the human hand pose in RGB or RGB-D format; (ii) teleportation server, the core component in our system, which performs hand pose detection and converts detection results to robot control commands; (iii) teleoperated robot, which is either a real robot or a simulated robot in a virtual environment; (iv) web visualizer, which enables remote visualization across the internet.

predicts hand wrist and finger poses from the camera stream, (ii) the detection fusion module, which integrates the results from multiple cameras, (iii) the hand pose retargeting module, which maps human hand poses to the dexterous robot hand, and (iv) the motion generation module, which produces high-frequency control signals for the robot arm. A standardized software interface is defined for all four modules to facilitate flexibility and generalizability in AnyTeleop .

2.3.1 Hand Pose Detection

The hand pose detection module offers a unique feature to utilize input from various camera configurations, including RGB or RGB-D cameras, and single or multiple cameras. The design principle is to leverage more information, such as depth, and additional cameras, to improve performance when available. But it can also perform the task with minimal input, i.e. a single RGB camera. The detection module has two outputs: local finger keypoint positions in the wrist frame and global 6D wrist pose in the camera frame. The finger keypoint detection only requires RGB data while the wrist pose detection can optionally use depth information to achieve better results.

Finger Keypoint Detection. Our finger keypoint detection utilizes MediaPipe [169], a lightweight, RGB-based hand detection tool that can operate in real-time on a CPU. The MediaPipe detector can accurately locate 3D keypoints of 21 hand-knuckle coordinates in the wrist frame and 2D keypoints on the image.

Wrist Pose Detection from RGB-D. We use the pixel positions of the detected keypoints to retrieve the corresponding depth values from the depth image. Then, utilizing known intrinsic camera parameters, we compute the 3D positions of the keypoints in the camera frame. The alignment of the RGB and depth images is handled by the camera driver. With the 3D keypoint positions in both the local wrist frame and global camera frame, we can estimate the wrist pose using the Perspective-n-Point (PnP) algorithm.

Wrist Pose Detection from RGB only. The orientation of the hand can be computed analytically from the local positions of the detected keypoints. However, determining the wrist position in the camera frame can be challenging without explicit 3D information. To enhance MediaPipe for global wrist pose estimation, we adopt the approach used in FrankMocap [121] by incorporating an additional neural network that predicts the weak perspective transformation scale of the hand. The weak perspective transformation approximates the original perspective camera model by assuming that the observed object is farther from the camera than its size. Together with intrinsic parameters, this scale factor can be used to approximate the 3D position of the hand. The wrist position computed this way has a larger error than depth camera, but it is still sufficient for many downstream teleoperation tasks.

Visualization of Hand Pose Detection We visualize the hand pose detection results in Figure 2.4. We showcase five typical cases, which include: (i) a hand spreading out the fingers for teleoperation initialization, (ii) fingers facing downwards in preparation for a top-down grasp, (iii) a precision grasp using the thumb and index finger, (iv) a power grasp using all five fingers, and (v) a failure case where the hand is positioned vertically relative to the camera plane.

2.3.2 Detection Fusion

The detection fusion module integrates multiple camera detection results. Self-occlusion can be a problem when performing hand pose detection, especially when the hand is perpendicular to the camera plane. Using multiple cameras can alleviate this problem by providing additional

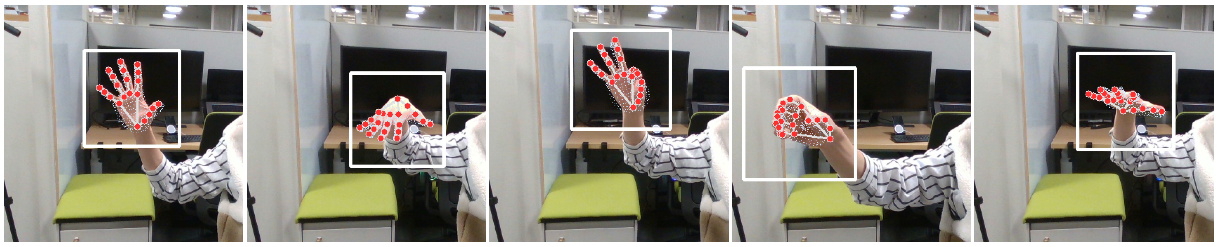


Figure 2.4. Hand Pose Detection Visualization. This figure visualizes the hand detection results, with the white bounding box highlighting the predicted area and red points marking the identified finger key points. The hand skeleton is represented by the grey lines connecting the key points. Additionally, the small grey points depict the 2D projection of 3D vertices from the SMPL-X hand model. The figure showcases five diverse cases, from left to right: (i) a hand spreading out the fingers to initiate teleoperation, (ii) fingers facing downwards in preparation for a top-down grasp, (iii) a precision grasp using the thumb and index finger, (iv) a power grasp using all five fingers, and (v) a failure scenario where the hand is positioned vertically relative to the camera plane.

views. However, there are two main challenges in fusing multiple detection results: (i) each camera can only estimate the hand pose in its own frame and (ii) there is no straightforward metric to quantify the confidence of each detection result.

To overcome the first challenge, we perform an auto-calibration process using the human hand as a natural marker. We use the first N frames of hand detection results from multiple cameras to calculate the relative rotation between each camera, expressed in $SO(3)$. We find that although the absolute position of detected hand pose is not so accurate in RGB-only setting, the relative motion between consecutive frames is more robust. With orientation between each camera, we can transform the detected relative motion from different cameras into a single frame.

To address the second challenge, we use the SMPL-X [103] hand shape parameters predicted from the detection module, as inspired by Qin *et al.* [109]. During teleoperation, the true shape parameters should remain constant for a given operator, but the predicted values can contain errors during self-occlusion. We observe that larger shape parameter prediction errors often correspond to larger pose errors. To approximate the confidence score, we take the mean of the estimated shape parameters in the first N frames as a reference and compute the error between the predicted shape parameters and the reference. Implementation-wise, we require the

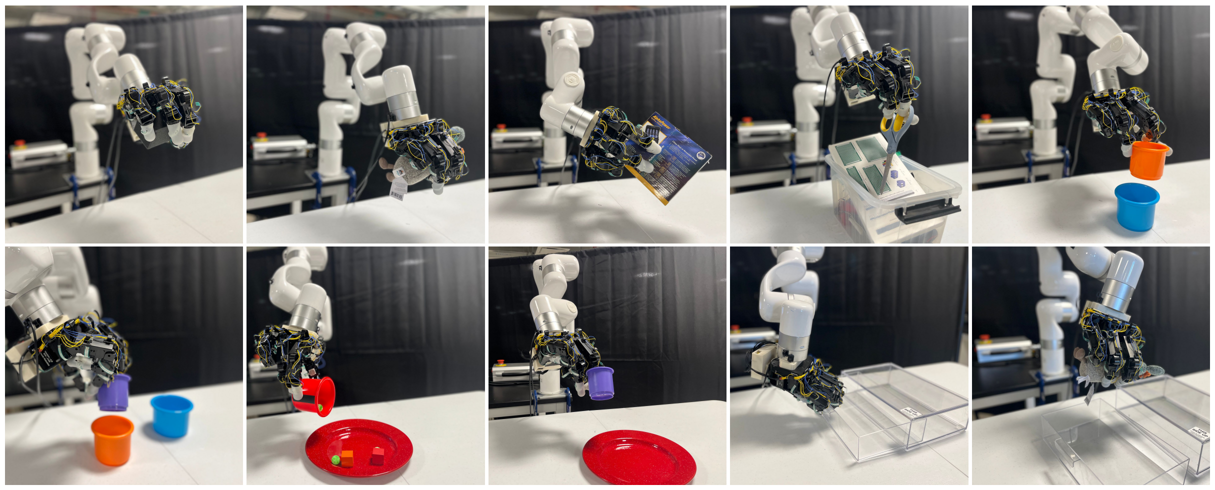


Figure 2.5. Real Robot Teleoperation Tasks. We replicate the ten manipulation tasks proposed in Sivakumar *et al.* [138] using same or similar objects. Top row, left to right: Pickup Box Object, Pickup Fabric Toy, Box Rotation, Scissor Pickup, Cup Stack. Bottom row, left to right: Two Cup Stacking, Pouring Cubes onto Plate, Cup Into Plate, Open Drawer and Open Drawer and Pickup Cup.

operator to spread their fingers during the first N frames to ensure an accurate reference value of shape parameters. The fusion module then selects the relative motion captured by the camera with the highest confidence score and forwards it to the next module. In implementation, we choose $N = 50$.

2.3.3 Hand Pose Retargeting

The hand pose retargeting module maps the human hand pose data obtained from perception algorithms into joint positions of the teleoperated robot hand. This process is often formulated as an optimization problem [110, 49], where the difference between the keypoint vectors of the human and robot hand is minimized. The optimization can be defined as follows:

$$\begin{aligned} \min_{q_t} \sum_{i=0}^{N} ||\alpha v_t^i - f_i(q_t)||^2 + \beta ||q_t - q_{t-1}||^2 \\ \text{s.t.} \quad q_l \leq q_t \leq q_u, \end{aligned} \tag{2.1}$$

where q_t represents the joint positions of the robot hand at time step t, v_t^i is the i-th keypoint vector for human hand computed from the detected finger key points, $f_i(q_t)$ is the i-th forward kinematics function which takes the robot hand joint positions q_t as input and computes the i-th keypoint vector for the robot hand, q_l and q_u are the lower and upper limits of the joint position, α is a scaling factor to account for hand size difference. An additional penalty term with weight β is included to improve temporal smoothness. When retargeting to a different morphology, such as a Dclaw in Figure 2.1, we need to specify the key point vectors mapping between the robot and human fingers manually. It is worth noting that this module only considers the robot hand.

We demonstrate the results of hand pose retargeting in Figure 2.11. The figure displays seven gestures being performed using four different dexterous hands.

2.3.4 Motion Generation

Given the detected wrist and hand pose, our goal is to generate smooth and collision-free motion of robot arm to reach the target Cartesian end-effector pose. Real-time motion generation methods are required to have a smooth teleoperation experience. In the prior work of [49], the robot motion is driven by Riemannian Motion Policies (RMPs) [118, 32] that can calculate acceleration fields in real-time. However, accelerations towards a particular end-effector pose do not guarantee natural trajectories. In this work, we adopt CuRobo [144], a highly parallelized collision-free robot motion generation library accelerated by GPUs, to generate natural and reactive robot motion in real-time. In AnyTeleop, the motion generation module receives the Cartesian pose of the end-effector at a low frequency (25 Hz) from the hand detection and retargeting modules, and generates collision-free joint-space trajectories within joint limits at a higher frequency (120 Hz). The generated trajectories are ready for safe execution by impedance controllers on either a simulated or real robot.

2.4 Web-based Teleoperation Viewer

To better support the teleoperation tasks, we implement a web-based visualization module to facilitate remote and collaborative teleoperation, especially for teleoperation in simulated environments. It has the following features: (i) browser-based viewer, which makes it easily accessible remotely; (ii) synchronized visualization, i.e. two operators working on the same collaborative task should see the same scene synchronously from their own local viewports. The viewer is developed based upon the `meshcat` [37] library and utilizes `Three.js` [36] for rendering. The visualization server ports the simulation results onto the browser after each simulation iteration. Operators can get visual feedback from the browser window and move their hands to control the corresponding robot.

2.4.1 Web-based Teleoperation Viewer

In this section, we demonstrate how the web-based visualizer provides accessibility and multi-view support for teleoperation through its lightweight rendering and capability to run in multiple browser windows. Figure 2.6 shows screenshots of the web-based visualizer when it is used to visualize the five IsaacGym tasks depicted in Figure 2.1.

Lightweight Rendering vs High Visual Quality. The design of our web-based viewer prioritizes accessibility and convenience, as it can be used on any device with a browser and provides minimal but sufficient rendering capabilities for teleoperation. Although the rendering quality may not be as advanced as the original simulator viewer, simulation states can be saved for offline rendering to produce high-quality visual data. For example, in visual reinforcement learning tasks using RGB images as inputs, the rendered data can be generated using a more powerful engine such as a ray tracer after the teleoperation is completed.

Multi-View Support for Teleoperation. In teleoperation, human operators often require a clear understanding of the spatial relationships between objects and robots to make informed decisions. This information can be provided through multi-view rendering, which is a widely

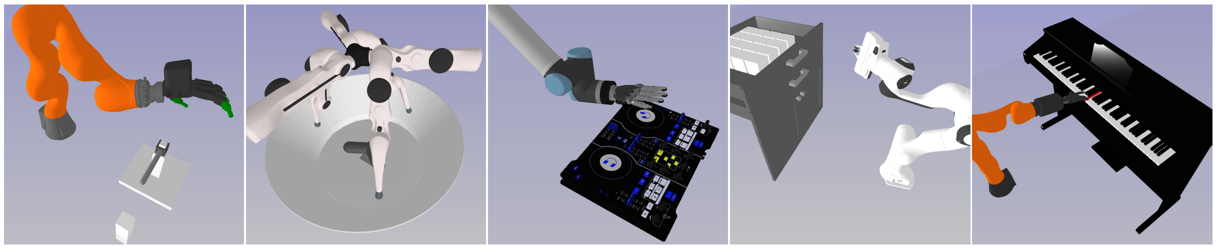


Figure 2.6. Web-based Viewer. The visualization of the teleoperation process can be performed in the web-based viewer, which features the five tasks from the IsaacGym tasks shown in Figure 2.1. The viewer utilizes the *three.js* library for real-time rendering through a web browser.

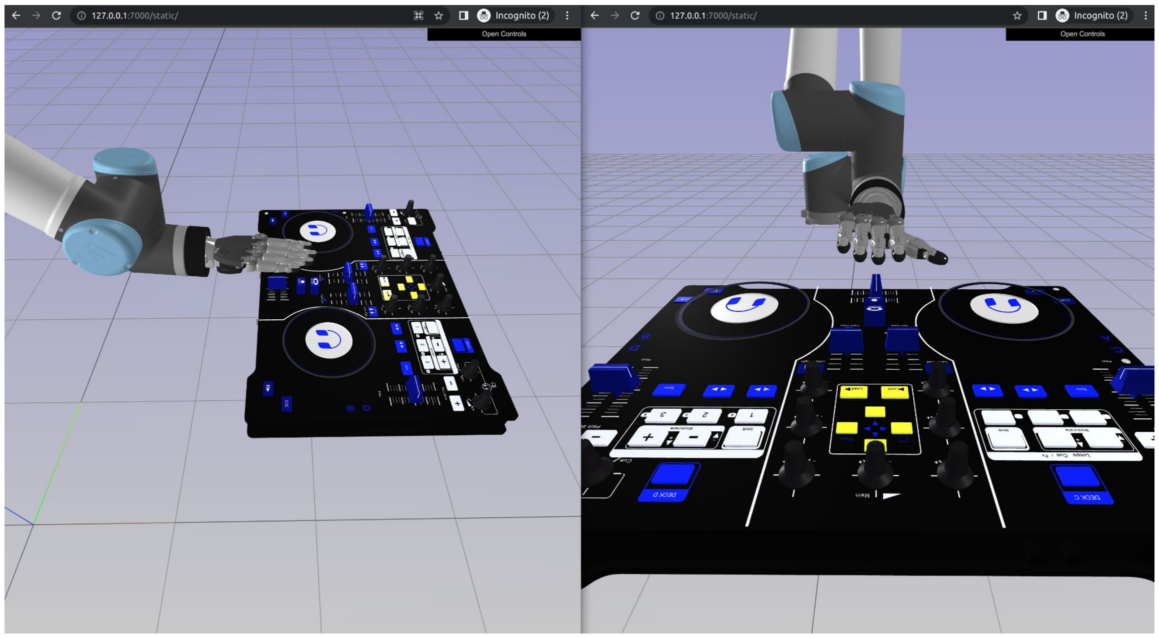


Figure 2.7. Multi-view Support with More Browser Windows. The web-based viewer offers multiple views to enhance the operator's understanding of the 3D object relationships. Additional views can be accessed by simply opening more browser windows.

used technique in previous teleoperation works [72, 109, 153]. Our web-based viewer offers multi-view support to the operator by simply opening multiple browser windows. As shown in Figure 2.7, an example of the operator using two views to perform a manipulation task is displayed. The operator is able to open as many windows as needed to enhance their teleoperation experience.

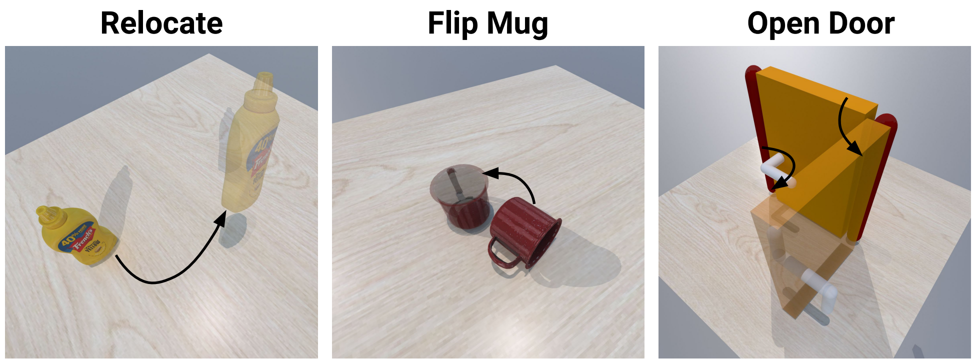


Figure 2.8. Imitation Learning Experiments in SAPIEN Environments. The figure visualizes the three tasks we use for both teleoperation data collection and imitation learning. The transparent object represents the goal of the task while the black arrow represents the steps of the task.

Table 2.2. Profiling Results. We profile different modules inside the teleoperation server on both desktop and laptop. The time is measured when all teleoperation modules are run on the same computer simultaneously.

		Desktop	Laptop
HardWare	GPU	RTX 3090	RTX 2070
	CPU	i9-10980XE	i7-8750
	Memory	32GB	32GB
	Modules	Time (ms)	Time (ms)
Profiling	Hand Pose (RGB)	26 ± 5	34 ± 5
	Hand Pose (RGB-D)	27 ± 5	35 ± 5
	Fusion	1 ± 0	1 ± 0
	Retargeting	9 ± 7	10 ± 9
	Motion	8 ± 3	11 ± 5

2.5 System Evaluation

2.5.1 Profiling Analysis

We perform profiling on modules mentioned in Section 2.3 on a desktop and a laptop. As shown in Table 2.2, the most time-consuming module is hand pose detection, which runs on a GPU for real-time inference. The designed maximum frequency for hand pose detection is 25Hz, so both the desktop and laptop can meet the requirement. Both the retargeting module and the fusion module run at the same frequency as the hand detection module due to the publisher and

Table 2.3. Real Robot Teleoperation Results. We replicate the experiment settings and tasks in [138] and compare with [138]. For the baseline method, we use the success rate reported in their paper [138]

Task	AnyTeleop	Telekinesis [138]
Pickup Box Object	**1.0**	0.9
Pickup Fabric Toy	**1.0**	0.9
Box Rotation	**0.6**	**0.6**
Scissor Pickup	**0.8**	0.7
Cup Stack	**0.9**	0.6
Two Cup Stacking	**0.7**	0.3
Pouring Cubes onto Plate	**0.7**	**0.7**
Cup Into Plate	**1.0**	0.8
Open Drawer	**1.0**	0.9
Open Drawer and Pickup Object	**0.9**	0.6

subscriber logic. For best performance, the motion generation module should run at 120Hz but can still work with a lower frequency. Notably, we found it difficult to achieve this throughput when running all these modules on the same computer. Luckily, with our communication-oriented design, we can run the control modules on a separate machine to achieve the best performance.

2.5.2 Real Robot Teleoperation

In this section, we will test our AnyTeleop system across a wide range of real-world tasks that covers diverse objects and manipulation skills. Besides, we will compare our teleoperation performance of AnyTeleop with a similar teleoperation system. A fair comparison of real-robot tasks is often very challenging due to the difficulty in replicating the baseline methods delicately. To ensure a more fair comparison, we replicate the ten manipulation tasks proposed in Robotic Telekinesis [138] with the same XArm6 robot, Allegro hand, and similar objects. A trained operator attempts to solve these tasks using AnyTeleop system. The ten tasks are visualized in Fig. 2.5. Same as [138], we run each task ten times for AnyTeleop and use a single Intel RealSense camera. For the baseline method, we directly use the results reported in their paper.

As shown in Table 2.3, AnyTeleop can get a higher success rate of 8/10 tasks and the same success rate on 2/10 compared with the baseline. Although AnyTeleop is designed to be

Table 2.4. Comparison of Camera Configurations. We evaluate the teleoperation performance on the Play Piano task with different camera configurations.

Camera Configuration	Completion Time	Error Percentage (%)
Single RGB	109*s*	28.1%
Single RGB-D	87*s*	21.8%
Two RGB-D	**74s**	**12.5**%

more general, it can still outperform the baseline system that was specifically designed for the XArm6-Allegro hardware. We find that the major advantage of our system is the capability to handle objects with thin-walled structures, such as the cup-stack, two-cup-stacking, and cup-into-plate tasks. Our optimization-based retargeting module can close the distance between fingertips, which makes grasping the cup more stable. However, network-based retargeting can hardly translate the fine-grained precision grasp from human to robot, which leads to a lower success rate.

2.5.3 System Evaluation on Camera Configurations

In this section, we examine the impact of different camera configurations on the teleoperation performance of our system, AnyTeleop, which is capable of supporting diverse configurations including RGB, RGB-D, and single or multiple cameras. Even with a minimal configuration, i.e. a single RGB camera, the system can still perform effectively. Additionally, by adding more resources, such as multiple cameras, our system can achieve better performance.

We use the *Play Piano* task implemented in IsaacGym [80] as the evaluation scenario, which requires the robot hand to press piano keys in a specific order. The task is shown in Figure 2.1. To quantify performance, we introduce two task metrics: (i) completion time, i.e. the elapsed time from start to finish, and (ii) the percentage of incorrect key presses, which measures the number of incorrectly pressed keys relative to the total number of keys.

A trained operator performs the task ten times for each camera configuration. As reported in Table 2.4, with additional information, such as depth, and an increasing number of cameras, the task can be completed faster and with fewer errors, which demonstrates that our system allows users to easily trade-off between efficiency and system cost based on their use case.

2.6 Applications

2.6.1 Imitation Learning

The most important application of the proposed system is imitation learning from demonstration. We can first collect demonstrations on several dexterous manipulation tasks and then use the data to train imitation learning algorithms. In this experiment, we will show that the teleoperation data collected using our AnyTeleop can better support downstream imitation learning tasks. In the following subsection, we will first introduce the experiment setting and baseline and then discuss the experimental results.

Baseline and Comparison. To fairly compare with previous teleoperation systems, we need to align both the task setting and robot configuration precisely. It is often challenging for real-robot hardware but much easier for a simulated environment. Thus, we choose a recent vision-based teleoperation work [109] that can be used for simulated robots as our baseline. It is worth noting that we are comparing two teleoperation systems via the demonstration data collected by each system. Thus, we compared with the baseline by training the same learning algorithm on different demonstration data collected via the baseline system and our teleoperation system. We follow [109] to choose Demo Augmented Policy Gradient (DAPG) [114] as the imitation algorithm. We also compare it with a pure reinforcement learning (RL) based algorithm from [108] which does not utilize demonstrations. We provide the same dense reward for RL training as previous work [109].

Manipulation Tasks. We directly use the manipulation tasks proposed by the baseline work [109] for comparison, which include three tasks: (i) *Relocate*, where the robot picks an object on the table and moves it to the target position; (ii) *Flip Mug*, where the robot needs to rotate the mug for 90 degrees to flip it back; (iii) *Open Door*, where the robot needs to first

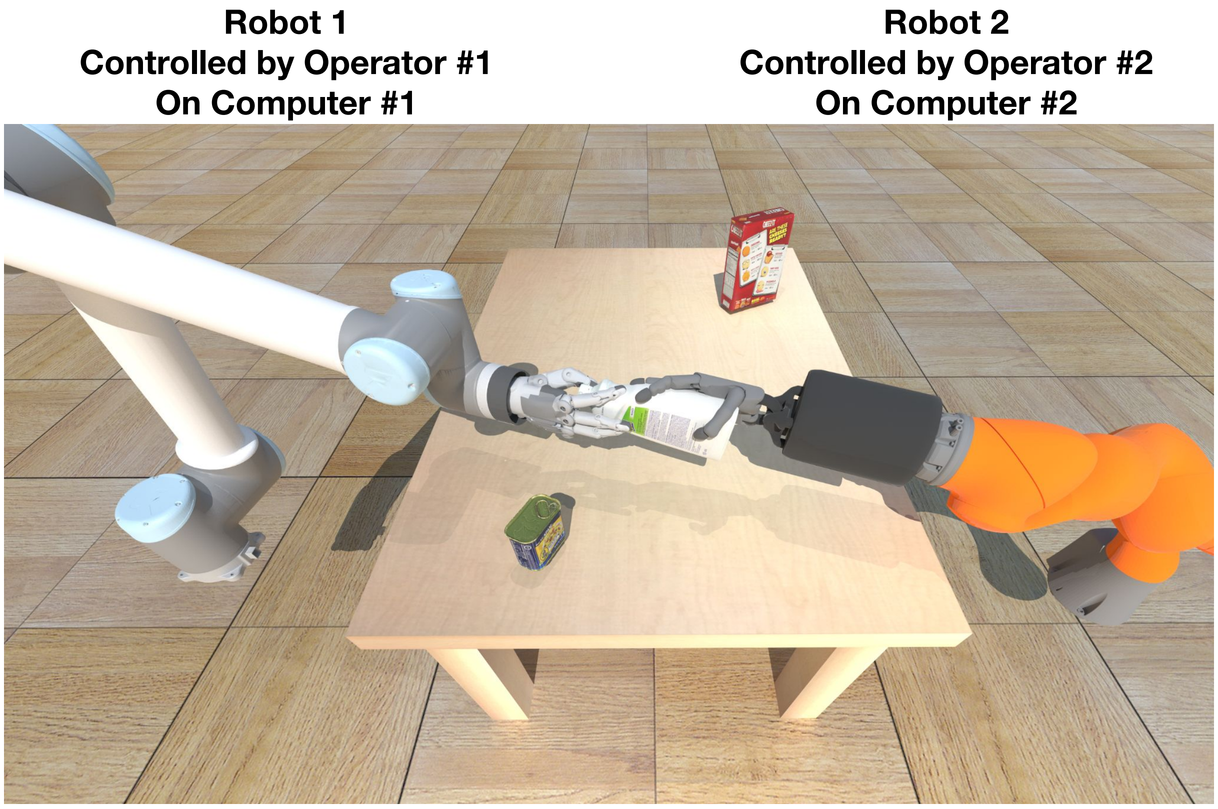


Figure 2.9. Collaborative Teleoperation for Handover Task. Operator #1 act as the UR10-Schunk robot and operator #2 acts as the Kuka-Shadow robot. In this task, the operator #2 needs to pick up an object on the table and handoverit to operator #1.

rotate the lever to unlock the door, and then pull it to open the door. The manipulated objects in all three tasks are randomly initialized and the target position is also randomized in *Relocate*. Each manipulation task has two variants: the floating-hand variant and the arm-hand variant. The floating-hand is a dexterous hand without a robot arm that can move freely in space. The arm-hand means the hand is mounted on a robot arm with a fixed base, which is a more realistic setting.

Demonstration Details. For the baseline teleoperation system [109], we directly use the demonstration collected by the original authors with 50 demonstration trajectories for each task. The baseline system only utilizes a single RGB-D camera. For fairness, we also collect 50 trajectories for each task using the single camera setup. The baseline system can only handle floating hands and they propose a demonstration translation pipeline to convert the

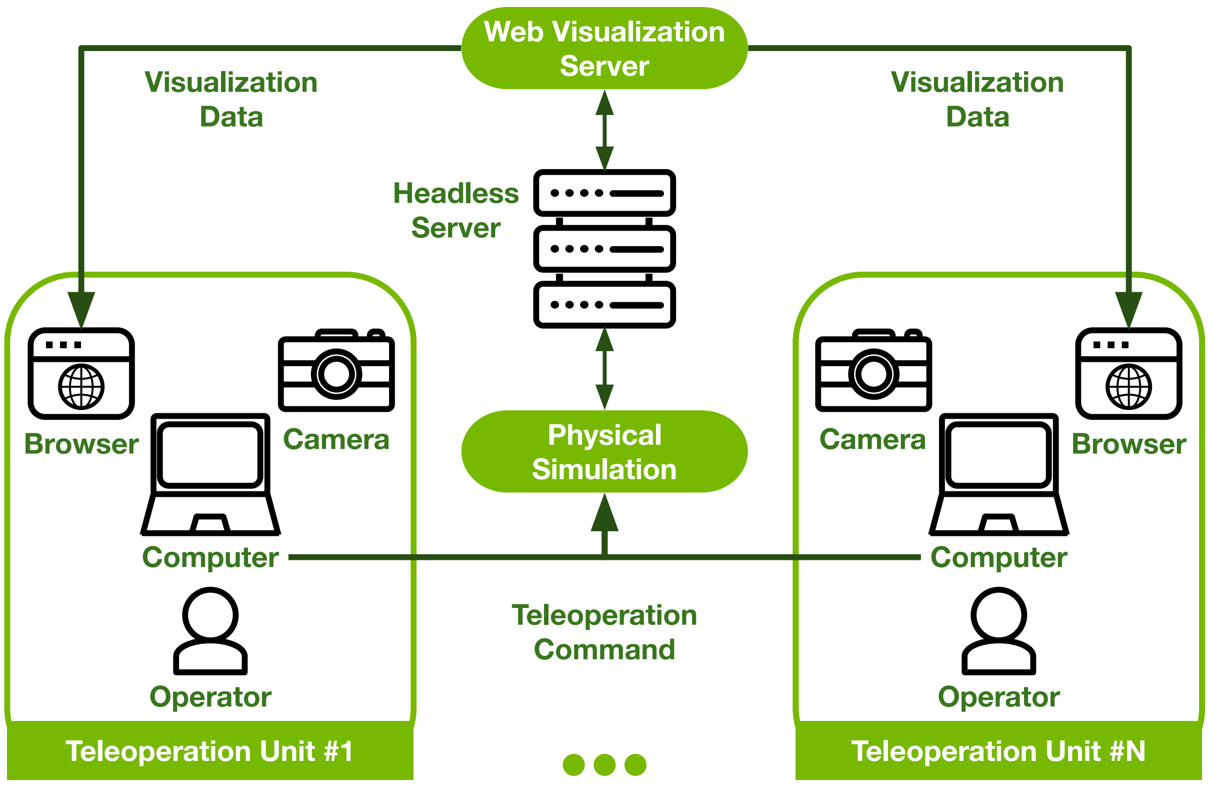


Figure 2.10. Collaborative Teleoperation System. Our system can be extended to collaborative manipulation tasks even when operators are not in the same physical location. Each operator can use a local computer with camera to detect the hand pose and send the detection results to a central server. Meanwhile, they can use the web browser to visualize the current simulation environment, including the robot controlled by other operators.

demonstration with floating hands to demonstrations with arm-hand. For our AnyTeleop , we collect demonstrations using the arm-hand setting and convert the demonstration to floating hand so that the demonstration can be used by both the floating-hand variant and arm-hand variant.

Results and Discussion. For each method on each task, we train policies with three different random seeds. For each policy, we evaluate it on 100 trials. More details about the success metrics can be found in [109]. As shown in Table, the imitation learning algorithm trained on demonstration collected by AnyTeleop can outperform baseline and RL on most tasks with one exception. Compared with the demonstration collected via the baseline system, our system has two benefits that contribute to better performance in imitation learning: (i) The collected trajectory is more smooth, which means that the state-action pairs are more consistent and easier to be consumed by the network. (ii) Different from the baseline, our system explicitly supports teleoperation with arm-hand system and guarantees no self-collision. On the contrary, the baseline system utilizes retargeting to generate joint trajectory for robot arm, which may lead to several self-collision for robot arm. Thus we can observe significant performance gain of our system for manipulation tasks with arm-hand. For the flip mug task, the difficulty of collecting demonstration with arm is much larger than with a floating hand, which influences the demonstration quality.

2.6.2 Collaborative Manipulation

Collaborative manipulation is a key technology for the development of human-robot systems [149]. Collecting demonstration data for collaborative manipulation tasks has been a challenging task since it requires multiple operators to work together seamlessly. With our modularized and extensible system design and web-based visualization, our system enables convenient data collection on collaborative tasks, even if operators are not in the same physical location. In this section, we show that our teleoperation system can be extended to a collaborative setting where multiple operators coordinate together to perform manipulation tasks. We choose human-to-robot handover as an example as shown in Fig. 2.9. In this setting, operator #1 control a robot hand, and operator #2 control a human hand.

Collaborative Teleoperation System Design. Fig. 2.10 illustrates the system architecture for multi-operator collaboration, which includes two components. (i) Teleoperation Units: It is composed of a computer that is connected to at least one camera and a human operator. In each teleoperation unit, the human operator will watch the real-time visualization on a web browser and move the hand accordingly to perform manipulation tasks. (ii) Central Server: it runs the physical simulation and the web visualization server. The detection results from multiple teleoperation units are sent to the server and converted into robot control commands based on the pipeline in Section 2.3. Meanwhile, the web visualizer server will keep synchronized with the simulated environment and maintain the visualization resources as described in Section 2.4.

Acknowledgement

Chapter 2, in full, is a reprint of the material published in the 2023 Robotics: Science and Systems (RSS): “AnyTeleop: A General Vision-Based Dexterous Robot Arm-Hand Teleoperation System” (Yuzhe Qin, Wei Yang, Binghao Huang, Karl Van Wyk. Hao Su, Xiaolong Wang, Yu-Wei Chao, Dieter Fox). The dissertation author was the primary investigator and author of this paper.

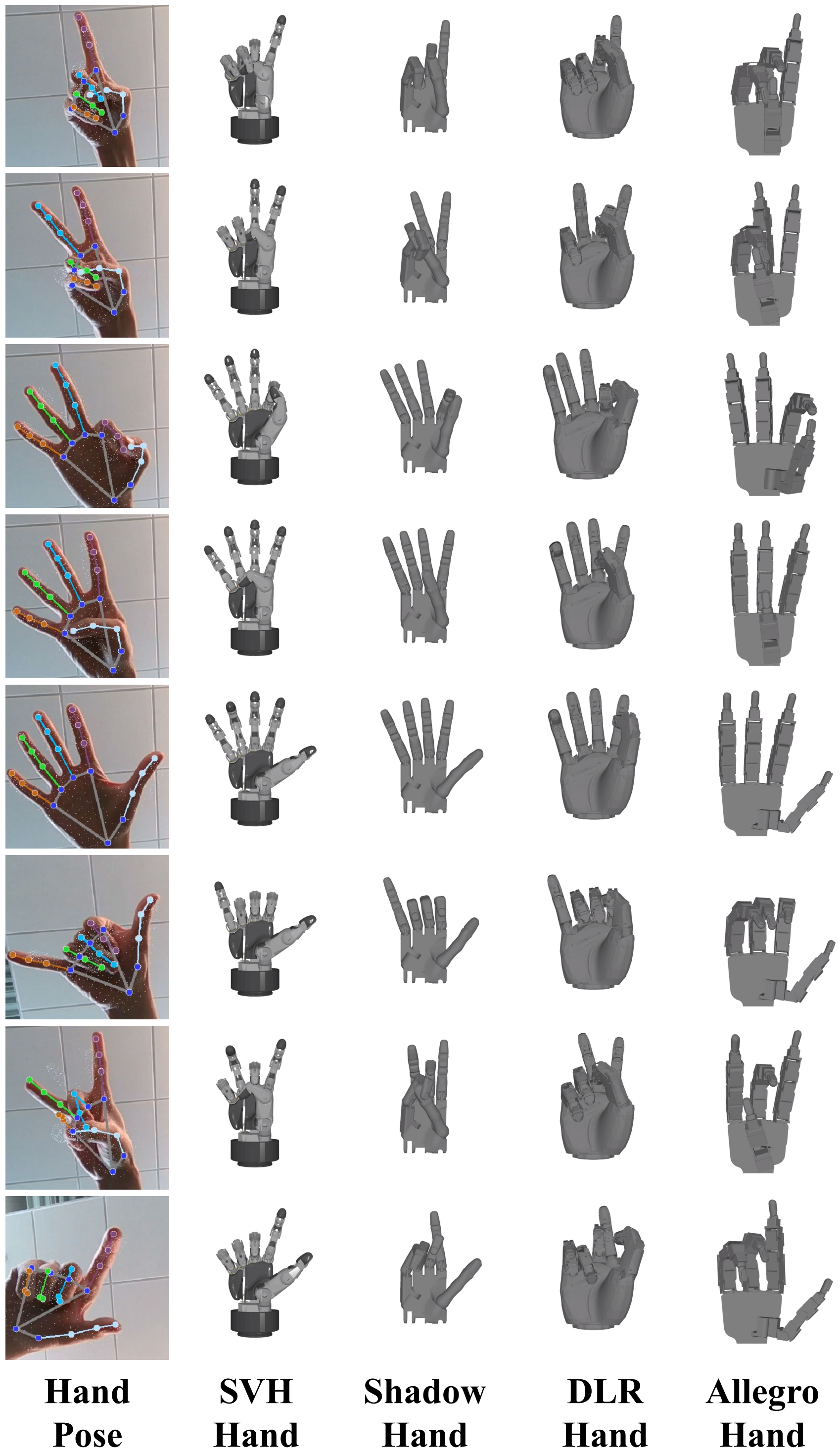


Figure 2.11. Visualization of Hand Pose Retargeting. The figure presents the results of hand pose retargeting for seven gestures and four different dexterous robot hands. The four hands are displayed in order from left to right: (i) Schunk SVH hand; (ii) Shadow Hand; (iii) DLR Hand; (iv) Allegro Hand.

Chapter 3

Augmenting Human Teleoperation Demonstration for Real-World Robot Manipulation

In this chapter, we present CyberDemo, an innovative approach to robotic imitation learning that capitalizes on simulated human demonstrations to master real-world tasks. CyberDemo represents a significant advancement in leveraging virtual environments for the training of robotic systems, employing extensive data augmentation techniques to bridge the gap between simulated training and real-world application. This method demonstrates superior performance compared to traditional in-domain real-world demonstrations, adeptly handling a diverse array of physical and visual conditions during task execution.

The affordability and convenience of data collection in simulated environments do not compromise the effectiveness of CyberDemo. On the contrary, it achieves higher success rates across a variety of tasks compared to baseline methods and exhibits remarkable generalizability, even with objects that were not included in the initial training set. For instance, CyberDemo effectively manipulates novel tetra-valve and penta-valve objects, despite the training demonstrations exclusively involving tri-valves. This capability underscores the robustness of the approach and its potential to generalize well beyond the training data.

Through the development and evaluation of CyberDemo, this chapter underscores the transformative potential of simulated human demonstrations in enhancing the capabilities of robots to perform dexterous manipulation tasks in real-world settings. Our findings indicate that by effectively harnessing the power of simulation for training purposes, we can significantly expand the operational range and effectiveness of robotic systems in handling complex and varied tasks that they might encounter in everyday environments.

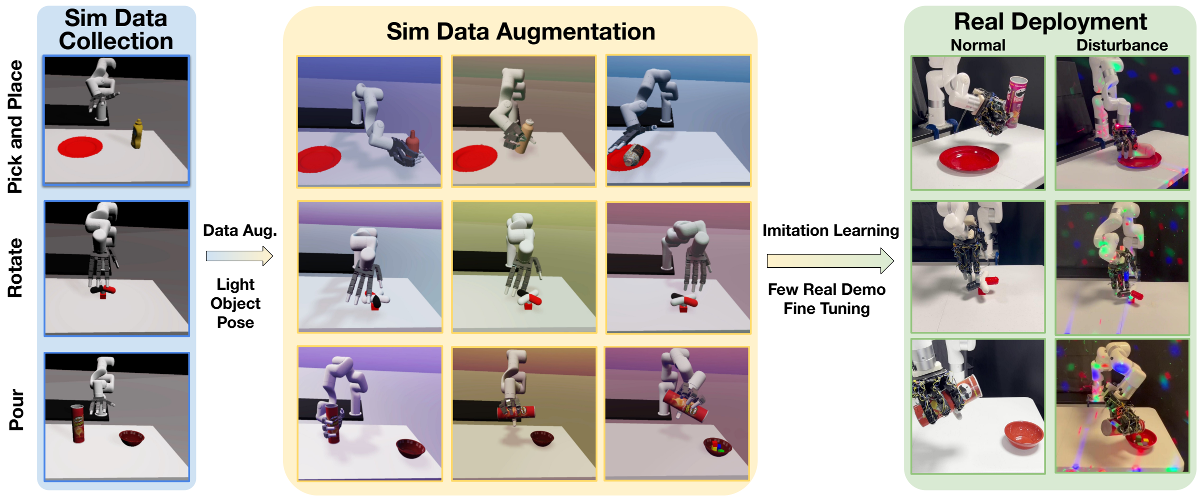


Figure 3.1. CyberDemo Overview. CyberDemo is a novel pipeline for learning real-world dexterous manipulation by using simulation data. First, we collect human demos in a simulated environment (blue region), followed by extensive data augmentation within the simulator (yellow region). Then, the imitation learning model, trained on augmented data and fine-tuned on a few real data, can be deployed on a real robot.

3.1 Introduction

Imitation learning has been a promising approach in robot manipulation, facilitating the acquisition of complex skills from human demonstration. However, the effectiveness of this approach is critically dependent on the availability of high-quality demonstration data, which often necessitates substantial human effort for data collection [16, 101, 12]. This challenge is further amplified in the context of manipulation with a multi-finger dexterous hand, where the complexity and intricacy of the tasks require highly detailed and precise demonstrations.

In imitation learning, in-domain demonstrations, which refer to the data collected directly from the deployment environment, are commonly used for robot manipulation tasks [86]. It is generally believed that the most effective way to solve a specific task is to collect demonstrations directly from the real robot on that task. This belief has been upheld as the gold standard, but we wish to challenge it. We argue that collecting human demonstrations in simulation can yield superior results for real-world tasks, not only because it does not require real hardware and can be executed remotely and in parallel, but also due to its potential to enhance final task performance by employing simulator-only data augmentation [87, 62, 124, 71, 109, 84]. This allows the generation of a dataset that is hundreds of times larger than the initial demonstration set. However, while existing studies employ the generated dataset to train in-domain policies within the simulation, the sim2real challenge of transferring policies to the real world remains an unresolved problem.

In this dissertation, we study the problem of how to utilize simulated human demos for real-world robot manipulation tasks. We introduce **CyberDemo**, a novel framework designed for robotic imitation learning from visual observations, leveraging simulated human demos. We first collect a modest amount of human demonstration data via teleoperation using low-cost devices in a simulated environment. Then, CyberDemo incorporates extensive data augmentation into the original human demonstration. (i) Image-level augmentation, utilizing common techniques such as color jitter and random crop. (ii) Scene-level augmentation, where we randomize the visual materials, light conditions, and camera viewpoint in the simulated scene, then replay the original demonstration to render new images. (iii) Kinematics-level augmentation, where we randomize the pose of the robot and objects, including some out-of-distribution states, and recalculate the robot action for these states using inverse kinematics.

The augmented set covers a broad spectrum of visual and physical conditions not encountered during data collection, thereby enhancing the robustness of the trained policy against these variations. These augmentation techniques are also designed with the downstream sim2real transfer in mind. We employ a unique curriculum learning strategy to train the policy on the augmented dataset, then fine-tune it using a few real-world demos (3-minute trajectories), facilitating effective transfer to real-world conditions. Surprisingly, when transferred to the real world, this policy outperforms those trained on in-domain demonstrations collected under identical real-world settings. While policies trained on only real-world demonstrations may suffer from variations in lighting conditions, object geometry, and object initial pose, our policy is capable of handling these without the need for additional human effort.

Our system, which utilizes a low-cost motion capture device for teleoperation (i.e., RealSense camera) and demands minimal human effort (i.e., a 30-minute demo trajectory), can learn a robust imitation learning policy. Despite its affordability and minimal human effort requirements, CyberDemo can still achieve better performance on the real robot. Compared with pre-trained policies, e.g. R3M [95] fine-tuned on real-world demonstrations, CyberDemo achieves a success rate that is 35% higher for quasi-static *pick and place* tasks, and 20% higher for non-quasi-static *rotate* tasks. In the generalization test, while baseline methods struggle to handle unseen objects during testing, our method can rotate novel tetra-valve and penta-valve with 42.5% success rate, even though human demonstrations only cover tri-valve (second row of Figure 3.1). Our method can also manage significant light disturbances (last column of Figure 3.1). In our ablation study, we observe that the use of data augmentation, coupled with an increased number of demonstrations in the simulator, results in superior performance compared to an equivalent increase in real-world demonstrations. To foster further research, we will make our code and human demonstration dataset publicly available.

To summarize, the main contributions of our work are as follows:

1. We introduce a novel pipeline for imitation learning from human demonstration collected in the simulator.

2. We propose a collection of demo augmentation methods in the simulator that enhances the robustness and generalizability of learned policy.

3. We show that collecting simulated human demonstrations can also be super beneficial to real-world robotics, even for complex dexterous manipulation tasks.

This work demonstrates that human demonstrations collected in a simulator offer substantial advantages over traditional real-world teleoperation for robot manipulation learning. Simulated demonstrations are not only safer and easier to collect, eliminating the need for physical hardware and enabling remote, parallel data acquisition, but also exhibit superior effectiveness in training manipulation policies for real-world deployment. While direct teleoperation in the real world remains a common practice, our findings highlight the significant cost and effectiveness benefits of leveraging simulated environments for demonstration collection in robotic imitation learning.

3.2 Related Work

Data for Learning Robot Manipulation. Imitation learning has been proven to be an effective approach to robotic manipulation, enabling policy training with a collection of demonstrations. Many works have focused on building large datasets using pre-programmed policies [59, 168, 35, 47, 61], alternative data sources such as language [57, 140, 137, 136] and human video [110, 133, 97, 11, 132] or extensive real-world robot teleoperation [16, 101, 6, 8, 12, 60, 41, 79, 86]. However, such works predominantly targeted parallel grippers. Collecting large-scale demonstration datasets for high-DoF dexterous hands continues to be a significant challenge. Meanwhile, data augmentation presents a viable strategy to improve policy generalization by increasing the diversity of data distribution. Previous studies have applied augmentation in low-level visual space [117, 54, 34, 135], such as color jitter, blurring, and cropping, while more recent works propose semantic-aware data augmentation with generative models [162, 82, 30, 14, 31, 177]. However, these augmentations operate at the image level and are not grounded in physical reality. CyberDemo extends data augmentation to the trajectory level using a physical simulator, accounting for both visual and physical variations. Concurrent to our work, MimicGen [84] proposes a system to synthesize demonstrations for long-horizon tasks by integrating multiple human trajectories. However, it confines demonstrations to in-domain learning, i.e., it only trains simulation policies with simulated demos without transferring to real robots. In contrast, our work aims to harness simulation for real-world problem-solving. We exploit the convenience of simulators for collecting robot demonstrations and employ a sim2real approach to transfer these demos to a dexterous robot equipped with a multi-finger humanoid hand. Our research emphasizes a general framework that leverages simulated demonstrations for real-world robot manipulation.

Pre-trained Visual Representation for Robotics Recent progress in large-scale Self-Supervised Learning [51, 18, 50] has enabled the development of visual representations that are advantageous for downstream robotic tasks [131, 167, 164]. Several studies have focused on pretraining on non-robotic datasets, such as ImageNet [38] and Ego4D [46], and utilizing the static representations for downstream robot control [95, 102, 157]. Other research has focused on pre-training visual representations on robot datasets, using action-supervised self-learning objectives that depend on actions [133, 128], or utilizing the temporal consistency of video as a learning objective [160, 125, 141, 129]. These investigations primarily aimed to learn features for effective training of vision-based robotic manipulation. In addition to training visual representations on offline datasets, some researchers have also explored learning the reward function to be used in reinforcement learning [166, 9, 88, 83, 68]. Unlike prior studies, our work diverges by utilizing simulation data for pre-training rather than employing Self-Supervised Learning for representation learning. This not only enhances the learning of image representations but also incorporates task priors into the neural network through the use of action information. By pre-training in simulated environments, the manipulation policy can better generalize to new objects with novel geometries and contact patterns.

Sim2Real Transfer The challenge of transferring skills from simulation to real-world scenarios, known as sim2real transfer, has been a key focus in robot learning. Some approaches have employed system identification to build a mathematical model of real systems and identify physical parameters [63, 55, 26, 106, 75, 152]. Instead of calibrating real-world dynamics, domain randomization [146, 104] generates simulated environments with randomized properties and trains a model function across all of them. Subsequent research demonstrated that the selection of randomization parameters could be automated [48, 1, 165, 22]. However, due to the extensive sample requirements to learn robust policies, domain randomization is typically used with RL involving millions of interaction samples. Domain adaptation (DA) refers to a set of transfer learning strategies developed to align the data distribution between sim and real. Common techniques include domain adversarial training [44, 148] and the use of generative models to make simulated images resemble real ones [54, 15]. Most of these DA approaches focus on bridging the visual gap. However, the challenge of addressing the dynamics gap remains significant. The sim2real gap becomes even more pronounced for dexterous robotic hands that have high-DoF actuation and complex interaction patterns [48, 161, 105, 163]. In this work, we extend the concept of domain randomization to human demonstration collected in the simulator and focus on data augmentation techniques that can effectively utilize the simulation for transfer to a real robot. We demonstrate that there can be a significant benefit in collecting human demonstration in the simulator, despite the sim2real gap, instead of solely relying on real data.

3.3 CyberDemo

In CyberDemo, we initially gather human demonstrations of the same task in a simulator through teleoperation (Section 3.3.1). Taking advantage of the simulator's sampling capabilities and oracle state information, we enhance the simulated demonstration in various ways, increasing its visual, kinematic, and geometric diversity, thereby enriching the simulated dataset (Section 3.3.2). With this augmented dataset, we train a manipulation policy with Automatic Curriculum Learning and Action Aggregation (Section 3.3.3).

3.3.1 Collecting Human Teleoperation Data

For each dexterous manipulation task in this work, we collect human demonstrations using teleoperation in both simulated and real-world environments. For real-world data, we utilize the low-cost teleoperation system referenced in [111]. This vision-based teleoperation system

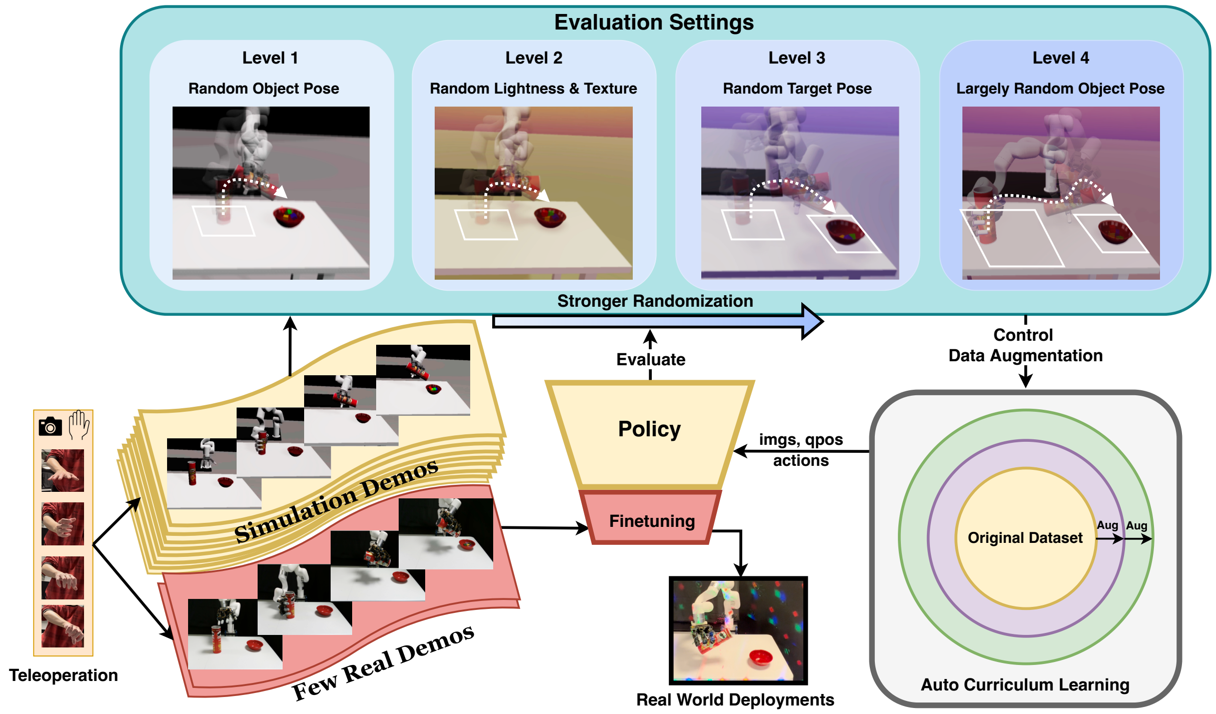


Figure 3.2. CyberDemo Pipeline. First, we collect both simulated and real demonstrations via vision-based teleoperation. Following this, we train the policy on simulated data, incorporating the proposed data augmentation techniques. During training, we apply automatic curriculum learning, which incrementally enhances the randomness scale based on task performance. Finally, the policy is fine-tuned with a few real demos before being deployed to the real world.

solely needs a camera to capture human hand motions as input, which are then translated into real-time motor commands for the robot arm and the dexterous hand. We record the observation (RGB image, robot proprioception) and the action (6D Cartesian velocity of robot end effector, finger joint position control target) for each frame at a rate of 30Hz. For this work, we collect only three minutes of robot trajectories for each task on the real robot.

For data in simulation, we build the real-world task environments within the SAPIEN [156] simulator to replicate the tables and objects used in real scenarios. It is worth noting that, for teleoperation, there is no requirement of reward design and observation spaces as in reinforcement learning settings, making the process of setting up new tasks in the simulator relatively simple. We employ the same teleoperation system [111] to collect human demonstrations in the simulator.

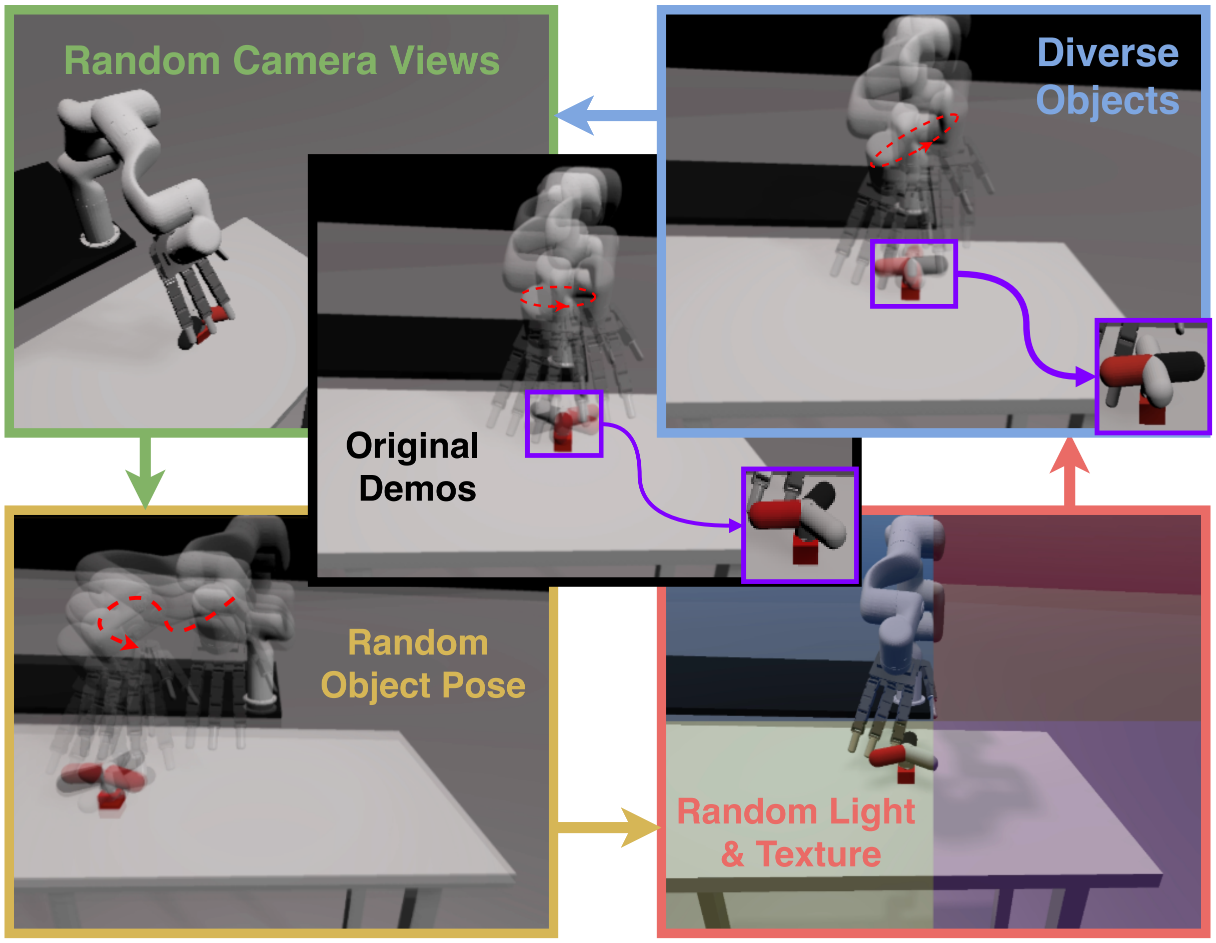


Figure 3.3. Data Augmentation. Our dataset augmentation encompasses four dimensions: (a) random camera views, (b) diverse objects, (c) random object pose, (d) random light and texture.

3.3.2 Augmenting Human Demo in Simulator

Unlike real-world data collection, where we are limited to recording observations of physical sensors, such as camera RGB images and robot proprioception, the simulation system enables us to record the ground-truth state and contact information within the virtual environment. This unique benefit of simulation provides a more comprehensive data format for the simulated demonstrations compared to its real-world counterparts. Thus, we can take advantage of demonstration replay techniques on these simulated demonstrations, which are not feasible with real-world data.

When developing data augmentation techniques in the simulator, it is essential to keep in mind that the ultimate goal is to deploy the trained policy to a real robot. The augmentation should accordingly focus on the visual and dynamical variations that are likely to be encountered in the real world. Moreover, we aim for the manipulation policy to generalize to novel objects not encountered during the data collection process. For example, manipulating the tetra-valve when collecting data only on the tri-valve in Figure 3.3. Specifically, we chose to augment the lighting conditions, camera views, and object textures to enhance the policy's robustness against visual variations. In addition, we modified the geometric shape of the objects and the initial poses of the robot and objects to improve the policy's robustness against dynamical variations as follows:

Randomize Camera Views. Precisely aligning camera views between demo collection and final evaluation, not to mention between simulation and reality, poses a significant challenge. To solve this problem, we randomize the camera pose during training and replay the internal state of the simulator to render image sequences from new camera views. Unlike standard image augmentation techniques such as cropping and shifting, our method respects the perspective projection in a physically realistic manner.

Random Light and Texture. To facilitate sim2real transfer and improve the policy's robustness against visual variations, we randomize the visual properties of both lights and objects (Figure 3.3, lower right). Light properties include directions, colors, shadow characteristics, and ambient illumination. Object properties include specularity, roughness, metallicity, and texture. Similar to camera view randomization, we can simply replay the simulation state to render new image sequences.

Add Diverse Objects. In this approach, we replace the manipulated object in the original demos with novel objects (Figure 3.3 upper right). However, directly replaying the same trajectory would not work as the object shape is different. Instead, we perturb the action sequence from the original demo with Gaussian noises to generate new trajectories. These trajectories provide reasonable manipulation strategies but are slightly different from the original one. With the highly cost-effective sampling in the simulator, we can enumerate the perturbation until it is successful. It is important to note that this technique is feasible with real-world demonstrations.

Randomize Object Pose. A common approach in reinforcement learning to enhance generalizability involves randomizing the object pose during reset. Augmenting imitation learning data to achieve a similar outcome, however, is less intuitive. Denote $T_A^B \in SE(3)$ as the pose of frame B relative to frame A. The original object pose is $T_W^{O_{old}}$, the newly randomized object pose is $T_W^{O_{new}}$, and the original end effector pose is $T_W^{R_{old}}$. The objective is to handle the object pose change $T_W^{O_{new}}(T_W^{O_{old}})^{-1}$. A simple strategy can be first moving the robot end effector to a new initial pose, $T_W^{R_{new}} = T_W^{O_{new}}(T_W^{O_{old}})^{-1}T_W^{R_{old}}$. Then, the relative pose between the robot and the object aligns with the original demonstration, enabling us to replay the same action sequence to accomplish the task. Although this method succeeds in generating new trajectories, it offers minimal assistance for downstream imitation learning. The new trajectory is always composed of two segments: a computed reaching trajectory to the new end effector pose $T_W^{R_{new}}$, and the original trajectory. Given that different augmented trajectories often share a significant portion of redundancy, they fail to provide substantial new information to learning algorithms.

To address this, we propose **Sensitivity-Aware Kinematics Augmentation** to randomize object poses for human demonstrations. Instead of appending a new trajectory ahead of the original one, this method amends the action for each step in the original demo to accommodate the change in object pose $T_W^{O_{new}}(T_W^{O_{old}})^{-1}$. The method includes two steps: (i) Divide the entire trajectory into several segments and compute the sensitivity of each segment; (ii) Modify the end effector pose trajectory based on the sensitivity to compute the new action.

(i) **Sensitivity Analysis for Trajectory Segments.** Sensitivity pertains to the robustness against action noise. For example, a pre-grasp state, when the hand is close to the object, has higher sensitivity compared to a state where the hand is far away. The critical insight is that it is simpler to modify the action of those states with lower sensitivity to handle the object pose variation $\Delta T = T_W^{O_{new}}(T_W^{O_{old}})^{-1}$. The robustness (the multiplicative inverse of sensitivity) of a trajectory segment ψ can be mathematically defined as follows:

$$\begin{aligned} \psi_{seg} &= \exp(\max \delta_a) \quad \text{s.t.} \quad \mathbf{eval}(\tau') = 1 \\ \tau' &= \{a_1, a_2, ..., a'_n, ..., a'_{n+K-1}, ..., a_N\} \\ \forall i \in seg \quad a'_i &= a_i + \delta_a \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, 1) \end{aligned} \tag{3.1}$$

In this equation, we divide the original action trajectory τ with length N into M segments, each segment with size $K = N/M$. Then we perturb the action within a segment seg by adding Gaussian noise of scale δ_a to the original action $\{a_m, a_{n+1}, ..., a_{n+k-1}\}$ while keeping all the actions outside of this segment unchanged to generate perturbed trajectory τ'. We assume the action space is already normalized to $[-1, 1]$ and **eval** is a binary function indicating whether an action trajectory can successfully solve the task. Intuitively, a demonstration segment is more sensitive if a smaller perturbation can cause it to fail. This sensitivity guides us on how to adjust the action to handle a new object pose. In practice, we incrementally escalate the noise scale δ_a applied to the original action trajectory until the task fails to determine max δ_a

(ii) **New End Effector Pose Trajectory.** To accommodate the new object pose, the total pose change of the end effector should be the same as the change in the object pose ΔT. Each action contributes a small part to this change. We distribute this "task" to each step based on sensitivity:

$$\begin{aligned} \overline{\psi}_{seg_j} &= \frac{\psi_{seg_j}}{\sum_{j=1}^{M} \psi_{seg_j}}, \quad \forall seg_j \\ \Delta T_j &= exp(\overline{\psi}_{seg_j} \log(\Delta T)/K) \\ a_i^{new} &= a_i f_i(\Delta T_j) \end{aligned} \tag{3.2}$$

In this equation, $\overline{\psi}_{seg_j}$ is the normalized robustness, ΔT_j represents the pose modification for each step, with all states in the same segment being responsible for the same amount of modification "task" to compute new action a_i^{new}. f_i is a similarity transformation in $SE(3)$ space that converts the motion from the world frame to the current end effector frame. Intuitively, segments with higher robustness are tasked with more significant changes.

Please note that all the actions discussed above pertain solely to the 6D delta pose of the end effector and do not include the finger motion of the dexterous hand. For tasks such as pick-and-place or pouring, which also involve a target pose (e.g., the plate pose in pick-and-place or the bowl pose in pouring), we can apply the same augmentation strategy to the target pose (as illustrated in Level 3 of Fig. 3.2).

3.3.3 Learning Sim2Real Policy

Given an augmented simulation dataset, we train a visual manipulation policy that takes images and robot proprioception as input to predict the robot's actions. In human teleoperation demonstrations, robot movements are neither Moravian nor temporally correlated. To deal with this issue, our policy is trained to predict action chunks rather than per-step actions, using Action Chunking with Transformers(ACT) [173]. This approach produces smoother trajectories and reduces compounding errors.

Despite our data augmentation's capacity to accommodate diverse visual and dynamic conditions, a sim2real gap remains for the robot controller. This gap becomes more challenging in our tasks, where the end effector is a high-DoF multi-finger dexterous hand. This controller gap can significantly impact non-quasi-static tasks like rotating a valve, as shown in the second row of Figure 3.1. To close this gap, we fine-tune our network using a small set of real-world demonstrations (3-minute trajectory). However, due to the discrepancies in data collection patterns of human demos between simulation and reality, direct fine-tuning on real data risks overfitting. To ensure a smoother sim2real transfer, we employ several techniques, which will be discussed subsequently.

Automatic Curriculum Learning. Curriculum learning and data augmentation techniques are often used together to provide a smoother training process. Following the spirit of curriculum design in previous reinforcement learning work [1, 48], we devise a curriculum learning strategy applicable to our imitation learning context. Prior to training, we group the augmentations in Section 3.3.2 into four levels of increasing complexity, as depicted in Figure 3.2. We begin training from the simplest level, $L = 0$, signifying no augmentation, and then evaluate the task success rate after several steps of training. The evaluation difficulty aligns with the current level of L. When the success rate surpasses a pre-defined threshold, we advance to the next level, which brings greater augmentation and harder evaluation. If the success rate fails to reach the threshold, we create additional augmented training data and stays at the current level. We continue this iterative process until all levels are completed. To prevent endless training, we introduce a fail-safe N_{max}: if the policy repeatedly fails during evaluation for N_{max} times, we also progress to the next level. This curriculum learning approach significantly depends on data augmentation techniques to generate training data dynamically with suitable levels of randomization. This concept stands in contrast to typical supervised learning scenarios, where data is pre-established prior to training. This on-demand data generation and customization highlights the advantage of simulation data over real-world demonstrations.

Action Aggregation for Small Motion. Human demonstrations often include noise, especially during operations involving a dexterous hand. For example, minor shaking and unintentional halting can occur within the demonstration trajectory, potentially undermining the training process. To solve this, we aggregate steps characterized by small motions, merging these actions into a single action. In practice, we set thresholds for both end-effector and finger motions to discern whether a given motion qualifies as small. Through the aggregation process, we can eliminate small operational noises from human actions, enabling the imitation learning policy to extract meaningful information from the state-action trajectory.

3.4 Experiment Setups

Our experimental design aims to address the following key queries:

(i) How does simulation-based data augmentation compare to learning from real demon-

Table 3.1. Main Comparison on Real Robot for Pick and Place Task. In our study, we compare the performance across four distinct tasks: (a) Pick and Place Bottle, and (b) Pick and Place Can (exploring different grasping approaches). We perform evaluations of the models in four levels of real-world scenarios. These levels included: (a) Level 1: In Domain, (b) Level 2: Out of Position, (c) Level 3: Random Light, and (d) Level 4: Out of Position and Random Light.

	Pick and Place Mustard Bottle (Single Object)				Pick and Place Tomato Soup Can (Single Object)			
	Level 1	Level 2	Level 3	Level 4	Level 1	Level 2	Level 3	Level 4
R3M	2 / 20	0 / 20	0 / 20	0 / 20	7 / 20	3 / 20	4 / 20	0 / 20
PVR	4 / 20	0 / 20	0 / 20	0 / 20	4 / 20	0 / 20	3 / 20	0 / 20
MVP	2 / 20	0 / 20	3 / 20	1 / 20	7 / 20	2 / 20	4 / 20	2 / 20
Ours	**7 / 20**	**6 / 20**	**8 / 20**	**5 / 20**	**14 / 20**	**11 / 20**	**13 / 20**	**13 / 20**

Table 3.2. Main Comparison on Real Robot for Rotating and Pouring Tasks. In our study, we compare the performance across four distinct tasks: (a) Pouring (grasping a bottle and pouring its contents into a bowl), and (b) Rotating the tri-valve. We perform evaluations of the models in four levels of real-world scenarios. These levels included: (a) Level 1: In Domain, (b) Level 2: Out of Position, (c) Level 3: Random Light, and (d) Level 4: Out of Position and Random Light.

	Pouring				Rotating			
	Level 1	Level 2	Level 3	Level 4	Level 1	Level 2	Level 3	Level 4
R3M	3 / 20	0 / 20	0 / 20	0 / 20	11 / 20	2 / 20	6 / 20	2 / 20
PVR	2 / 20	0 / 20	1 / 20	0 / 20	8 / 20	3 / 20	5 / 20	1 / 20
MVP	1 / 20	1 / 20	3 / 20	2 / 20	8 / 20	4 / 20	10 / 20	6 / 20
Ours	**9 / 20**	**4 / 20**	**10 / 20**	**7 / 20**	**15 / 20**	**10 / 20**	**17 / 20**	**13 / 20**

strations in terms of both robustness and generalizability?

(ii) How does our automatic curriculum learning contribute to improved policy learning?

(iii) What is the ideal ratio between simulated and real data to train an effective policy for a real-world robot?

3.4.1 Dexterous Manipulation Tasks

We have designed three types of manipulation tasks in both real-world and simulated environments, including two quasi-static tasks (pick and place, pour) and one non-quasi-static task (rotate). For the experiments, we utilize an Allegro hand attached to an XArm6. The action space comprises a 6-dim delta end effector pose of the robot arm and a 16-dim finger joint position of the dexterous hand, with PD control employed for both arm and hand.

Pick and Place. This task requires the robot to lift an object from the table and position it on a plate (first row of Figure 3.1). Success is achieved when the object is properly placed onto the red plate. We select two objects during data collection and testing on multiple different objects.

Rotate. This task requires the robot to rotate a valve on the table (second row of Figure 3.1). The valve is constructed with a fixed base and a moving valve geometry, connected via a revolute joint. The task is successful when the robot rotates the valve to 720 degrees. We use a tri-valve in data collection and test on tetra-valves and penta-valves.

Pour. This task requires the robot to pour small boxes from a bottle into a bowl (third row of Figure 3.1). It involves three steps: (i) Lift the bottle; (ii) Move it close to the bowl; (iii) Rotate the bottle to dispense the small boxes into the bowl. Success is achieved when all four boxes have been poured into the bowl.

For each task, we have designed levels for both data augmentation (Section 3.3.2) and curriculum learning (Section 3.3.3).

3.4.2 Baselines

Our approach can be interpreted as an initial pretraining phase using augmented simulation demonstrations followed by fine-tuning with a limited set of real data. It is natural to compare our method with other pre-training models for robotic manipulation. We have chosen three representative vision pre-training models. For all of them, we utilize the pre-trained model provided by the author and then fine-tune it using our real-world demonstration dataset.

PVR is built on MoCo-v2 [28], using a ResNet50 backbone [52] trained on ImageNet [122].

MVP employs self-supervised learning from a Masked Autoencoder [50] to train visual representation on individual frames from an extensive human interaction dataset compiled from multiple existing datasets. MVP integrates a Vision Transformer [39] backbone that segments frames into 16x16 patches.

Table 3.3. Ablation on Data Augmentation. We evaluated the benefits of data augmentation using auto-curriculum learning across various levels of difficulty. ”Original” represents the training distribution, ”Light” introduces random light noise, and ”Position” adds position augmentation during evaluation. We conducted 200 simulations and 20 real-world tests to assess performance in both simulated and practical settings.

	Test in Sim				Test in Real			
Levels / Demos	L1	L2	L3	L4	Original	Light	Position	Position + Light
[1] / 100	78%	0%	0%	0%	20%	5%	0%	0%
[1, 2] / 330	73%	75%	10.5%	7.5%	15%	25%	0%	0%
[1,2,3] / 550	58%	66.5%	43.5%	21%	15%	15%	5%	15%
[1,2,3,4] / 810	92.5%	81%	63%	49%	35%	30%	30%	40%

R3M proposes a pre-training approach where a ResNet50 backbone is trained using a mix of time-contrastive learning, video-language alignment, and L1 regularization. This model is trained on a large-scale human interaction videos dataset from Ego4D [46].

3.5 Results

3.5.1 Main Comparison

Augmented simulation data markedly boosts real-world dexterous manipulation. As depicted in Table 3.1 and Table 3.2, our methodology outperforms the baselines trained exclusively on real data in the in-domain setting (Level 1), exhibiting an average performance boost of 31.67% averaged on all tasks. Additionally, in other settings like random lighting (Level 2), out of position (Level 3), combined random lighting and out of position (Level 3) R3M, and MVP, the baselines exhibit a significant drop in success rates. In contrast, our method shows resilience to these variations, underscoring the efficacy of simulation data augmentation. Not only does this approach bridge the sim2real gap and amplify performance in in-domain real-world tasks, but it also significantly improves manipulations in out-of-domain real-world scenarios. By integrating augmentation for both visual and dynamic variations, our method successfully navigates challenges and delivers impressive results.

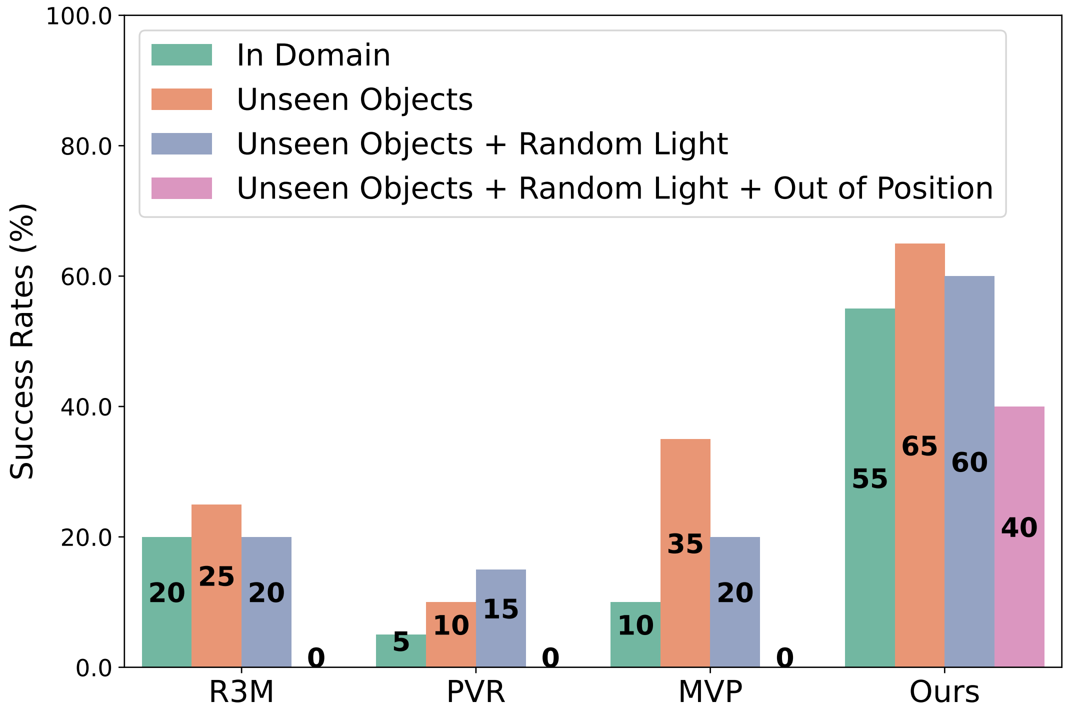


Figure 3.4. Generalization to Novel Objects for Pick and Place. We compare our approach with the baselines in scenarios involving novel objects, random light disturbances, and random object positions.

3.5.2 Generalization to Novel Objects

By incorporating data augmentation techniques, such as including diverse objects in simulation, our model can effectively manipulate unfamiliar objects, even when transitioning to a real-world context. As shown in Figure 3.4 and 3.5, the baseline methods grapple with more complex real-world situations. In the most challenging scenario, rotating novel objects under random light conditions and new object positions, only one baseline method manages to solve it by chance with a 2.5% success rate. In contrast, our method still accomplishes the task with a success rate of 30%.

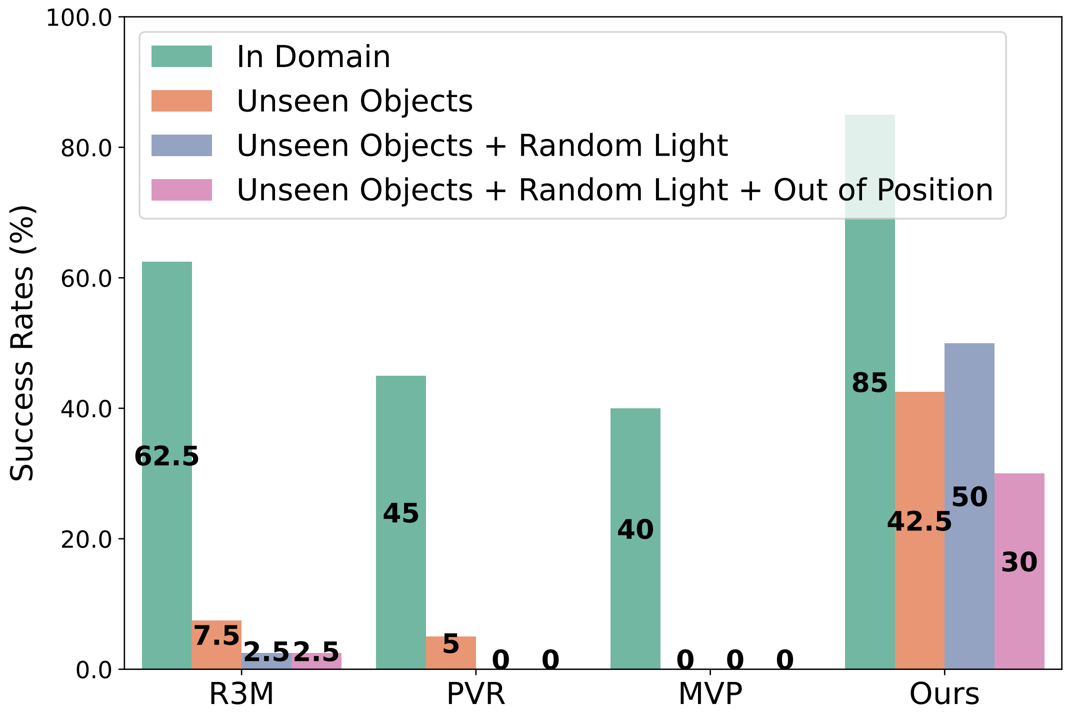


Figure 3.5. Generalization to Novel Objects for Rotating. The experimental setup for this task mirrors that of the ”Generalization to Novel Objects for Pick and Place” experiments.

3.5.3 Ablation on Data Augmentation

To evaluate the effectiveness of the data augmentation techniques, we perform an ablation study where our policy is trained with four levels of augmentation. As depicted in Table 3.3, the policy performs better in both the simulation and real-world settings with increased data augmentation, and the policy trained on all four levels excels in all metrics. Interestingly, the policy manages to solve simpler settings more effectively even in simulation when more randomness is introduced in the training data. These experiments underscore the importance of simulator-based data augmentation.

Table 3.4. Ablation on Auto-Curriculum Learning. We compare three different settings: (1) Auto Curriculum Learning based on the success rate. (2) Auto Curriculum Learning based on Data Generation Rate(the ratio of successfully generated trajectories to the total number of attempts). (3) Automatic Domain Randomization only based on Data Generation Rate.

	Test in Sim				Test in Real
Method	Level 1	Level 2	Level 3	Level 4	Out of Position + Random Light
ACL (Task)	**80%**	**61%**	43.5%	57%	**35%**
ACL (Data)	19.5%	30%	**75%**	**66%**	20%
ACL wo CL(Data)	20%	22%	32.5%	15%	5%

3.5.4 Ablation on Auto-Curriculum Learning

In this experiment, we evaluate the policy's effectiveness by testing it 200 times in simulations and conducting 20 real-world tests. As shown in Table 3.4, employing curriculum learning with auto-domain randomization solely based on the data generation rate yields inferior results compared to the approach based on model performance.

Acknowledgement

Chapter 3, in full, is a reprint of the material published in the 2024 Conference on Computer Vision and Pattern Recognition (CVPR): "CyberDemo: Augmenting Simulated Human Demonstration for Real-World Dexterous Manipulation" (Jun Wang*, Yuzhe Qin*, Kaiming Kuang, Yigit Korkmaz, Akhilan Gurumoorthy, Hao Su, Xiaolong Wang). The dissertation author was the primary investigator and author of this paper.

Chapter 4

Generalizable Dexterous Manipulation from Visual Sim2Real

In this chapter, we introduce a sophisticated sim-to-real framework designed to enhance the capabilities of robotic systems in performing dexterous manipulation with a high degree of generalization to new objects within the same category. Central to our framework is the innovative use of point cloud inputs and dexterous robotic hands to train the manipulation policy. This approach is augmented by two novel techniques aimed at bolstering the learning process and ensuring effective sim-to-real transfer.

Firstly, we employ imagined hand point clouds as augmented inputs, enriching the training data and providing a more comprehensive learning experience. Secondly, we have developed novel contact-based rewards that specifically encourage successful interaction between the robot's dexterous hands and the objects being manipulated. These rewards are designed to enhance the tactile feedback essential for delicate manipulation tasks, thereby improving the robot's performance in real-world scenarios.

We conduct a thorough empirical evaluation of our method using the Allegro Hand, a sophisticated robotic hand designed for complex manipulation tasks. Our tests involve grasping novel objects in both simulated environments and real-world settings, demonstrating the framework's robustness and versatility. To the best of our knowledge, this chapter describes the first policy learning-based framework that achieves such notable generalization results with dexterous hands.

Through detailed analysis and testing, this chapter highlights the potential of our sim-to-real framework to significantly advance the field of robotic dexterous manipulation. The techniques developed here pave the way for more adaptive, efficient, and skilled robotic systems capable of handling a diverse range of manipulation tasks in real-world applications.

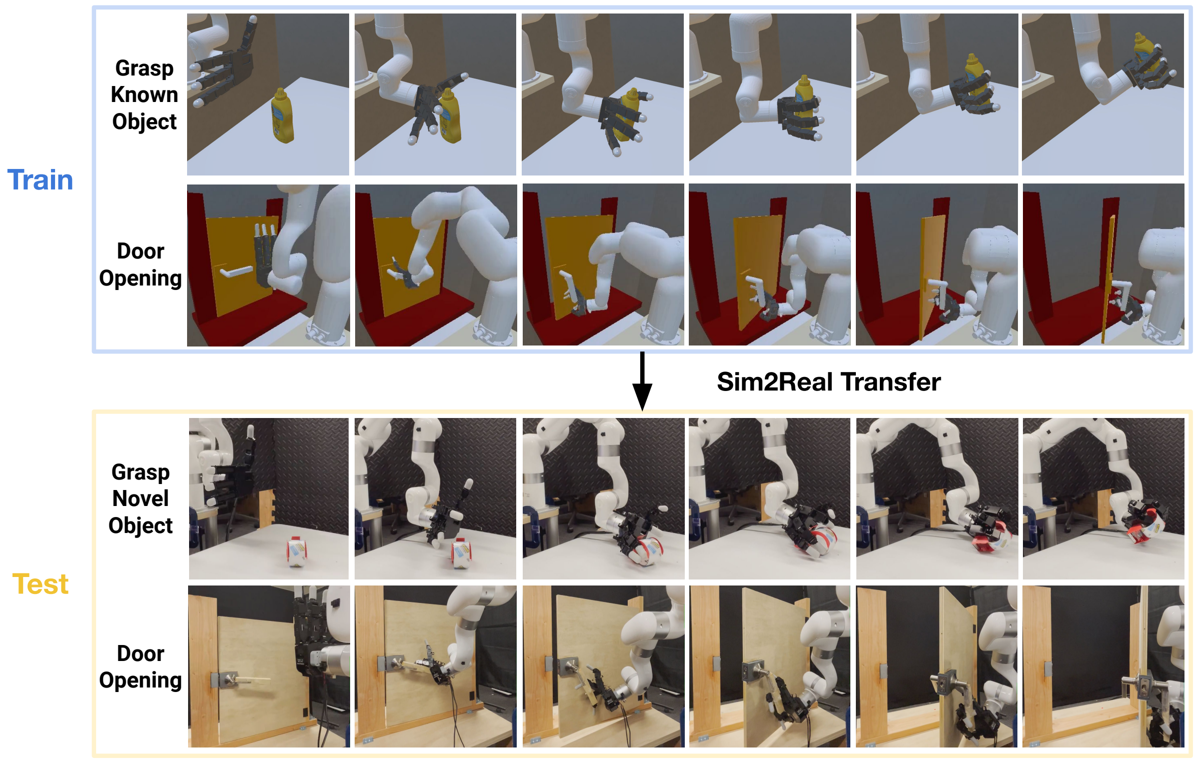


Figure 4.1. DexPoint Overview. We introduce a reinforcement learning method that takes the point cloud as input for two manipulation tasks: grasping and door opening. By introducing several techniques in the policy learning process, our point cloud-based policy trained purely in simulation can successfully generalize to novel objects and transfer to real world without any real-world data.

4.1 Introduction

Dexterous manipulation has remained to be one of the most challenging problems in robotics [2]. While multi-finger hands create ample opportunities for robots to flexibly manipulate objects in our daily life, the nature of the high degree of freedom and high-dimensional action space creates significant optimization challenges for both search-based planning algorithms and policy learning algorithms. Recent efforts using model-free Reinforcement Learning have achieved encouraging results on complex manipulation tasks [2, 175]. However, it still faces many challenges in generalizing to diverse objects and being deployed on multi-finger hands in the real world.

For example, the dexterous manipulation framework proposed by OpenAI et al. [2] can solve in-hand manipulation of Rubik's Cube with RL and transfer to the real robot hand. However, the policy is only trained with one particular object and it is not able to *generalize to diverse objects*. To achieve cross-object generalization, recent efforts proposed to learn robust 3D point cloud representations [27, 155, 56, 93] with diverse objects using RL in simulation. While point cloud input has also been shown easier for Sim2Real transfer [172] given its focus on geometry instead of texture, the assumption on the access of complete object point clouds and ground truth states limit the transferability of above methods to the *real robot deployment*. Among these works, Chen et al. [27] showed that cross-object generalization is achievable in simulation without knowing the shape of the object to grasp, but its requirement of real-time access to the object states is itself a very challenging robot perception problem, especially under large occlusions during hand object interaction.

In this dissertation, we provide a sim-to-real reinforcement learning framework for generalizable dexterous manipulation, using two tasks with the Allegro Hand[119]: (i) object grasping where the test objects have not been seen during training; (ii) door opening where the test doors have levers of novel shape that has not been used in training. The tasks are visualized in Figure 4.1.

We perform our studies by training a point cloud-based reinforcement learning policy in the grasping and door-opening task. With this approach, we list the three key discoveries of our framework for learning generalizable point cloud policy below:

(i) We justified that it is possible to achieve *direct sim-to-real transfer* for a dexterous manipulation policy with category-level generalizability when we use point cloud as the data representation.

(ii) Raw point clouds captured by sensors often come with heavy occlusions and noise: only a very small portion of the points from the observation represent the robot fingers. We propose to *imagine* the complete robot finger point clouds according to the robot kinematic model and use them to augment the occluded real point cloud observations. We find that explicitly augmenting the input by *imagined* points can help achieve better robustness and sample efficiency for reinforcement learning.

(iii) Different from existing works that add contact information to the input of RL, we design a novel reward using contact pair information without adding contact to the observation. This practice remarkably improves sample efficiency as well as learning stability and avoids the dependency on contact sensors that are often unavailable for real robot models.

4.2 Approach

Our objective is to train a generalizable point cloud policy on a dexterous robot hand-arm system that is able to grasp a wide range of objects or open a closed door with RL. We aims at Sim-to-Real transfer without any real-world training or data. During testing, the robot can only access the single-viewed point cloud and the robot's proprioception data. As is discussed before, training such a policy comes with numerous technical challenges, including reward design and imperfect point cloud information. In this work, we propose a novel reward design technique based on contact and imagined point cloud model to deal with these challenges.

Preliminaries: We model the dexterous manipulation problem as a Partially Observable Markov Decision Process (POMDP) $\mathcal{M} = (\mathcal{O}, \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{T}, \mathcal{U})$. Here, $\mathcal{O}$ is the observation space, $\mathcal{S}$ is the underlying state space, $\mathcal{A}$ is the action space, $\mathcal{R}$ is the reward function, $\mathcal{T}$ is the transition dynamics, and $\mathcal{U}$ generates agent's observation. At timestep t, the environment is at the state $s_t \in \mathcal{S}$. The agent observes $o_t \sim \mathcal{U}(\cdot|s_t) \in \mathcal{O}$. The agent takes action a_t and receives reward $r_t = \mathcal{R}(s_t, a_t)$. The environment state at timestep $t+1$ then transit to $s_{t+1} \sim \mathcal{T}(s_t, a_t)$. The

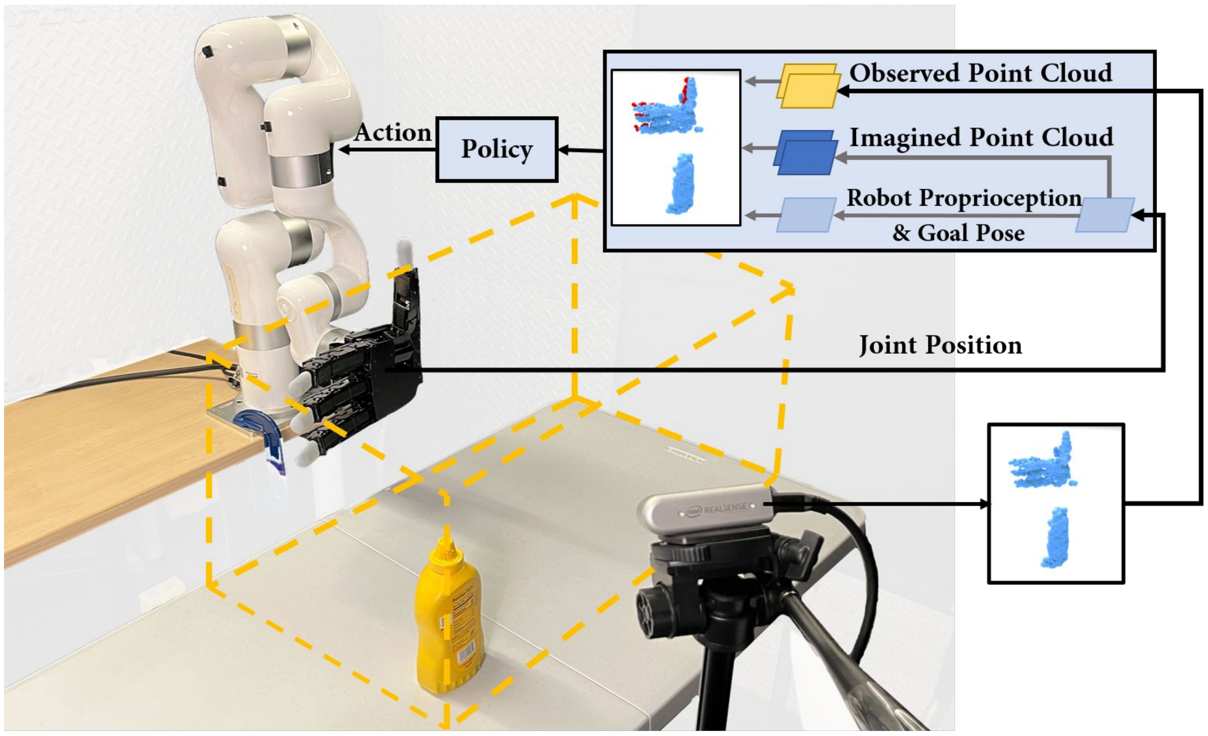


Figure 4.2. Real-experiment Setup. We use an Allegro Hand attached to an XArm6 and a RealSense D435 camera facing forward the robot.

objective of the agent is to maximize the return $\sum_{t=0}^{T} \gamma^t r_t$, where γ is a discount factor.

System Setup: Many previous works on learning-based dexterous manipulation attach the hand to a fixed platform to simplify the experiment environment in the real world. In this work, we create a more flexible and powerful dexterous manipulation system that includes both the robotic hand and the arm (Figure 4.2). Concretely, we attach the Allegro Hand to an XArm6 robot. Allegro Hand is a 16-DoF anthropomorphic hand with four fingers and XArm6 is a 6-DoF robot arm. We place a RealSense D435 camera at the right front of the robot to capture the point cloud. This setup brings additional challenges to RL exploration and Sim2Real deployment. We use SAPIEN [156] platform which uses a full-physics simulator to build the environment of the whole system. The simulation time step is $0.005s$ to ensure stable contact simulation. Each control step lasts for $0.05s$.

Tasks and Objects: We expect our robot to perform the grasping task over a diverse set of objects and to open a locked door by rotating the lever. In the grasping task, we first select a random object from an object dataset and place it on the table. The robot is then required to move it to a target pose. Moreover, the robot should be able to generalize to different initial states, so we randomize both the initial pose and the goal pose for each trial. In simulation experiments, we use bottles and cans from both ShapeNet [19] dataset and YCB [17] dataset. In real-world experiments, we use novel unseen objects to test the policy. In the door opening task, the robot hand is required first to rotate the lever to unlock the door latch and then pull the lever in a circular motion. We use three doors to test the policies both in the simulation and the real world, only one is used for training and the other two are unseen doors. We also randomize the initial pose for each trial.

Observation Space: The observation contains both visual and proprioceptive information with four modalities: (1) Observed point cloud provided by the camera; (2) Proprioception signals of the robot including joint positions and end-effector position; (3) Imagined hand point cloud proposed in Sec. 4.2.2; (4) Object goal position provided in each trial. All the information is accessible on the real robot. The dimension of each observation modality is shown in Figure 4.3.

Action Space: The action is responsible for controlling both the 6-DOF robot arm and the 16-DOF hand. It has 6 + 16 = 22 dimension in total. The robot arm is parametrized by the 6D translation and rotation of the end-effector relative to a reference pose. We use the damped least square inverse kinematics solver with a damping constant $\lambda = 0.05$ to compute the joint motion. Each finger joint of the Allegro hand is controlled by a position controller. Both robot arm and hand are controlled by PD controllers.

Network Architecture: The network architecture is visualized in Figure 4.3,

4.2.1 Reward Design with Oracle Contact

Since we aim to solve the dexterous manipulation problem with pure RL, the reward design is central to the method. We need a good reward function to ensure proper interaction between the robotic hand and the object. The whole interaction process consists of two phases. The first phase is to simply reach the object. The second phase is to grasp the object and move it to the target, which is more challenging. For the first phase, we encourage reaching the following reaching rewards:

$$r_{\text{reach}} = \sum_{\text{finger}} \frac{1}{\epsilon_r + d(\mathbf{x}_{\text{finger}}, \mathbf{x}_{\text{obj}})}. \tag{4.1}$$

Here, $\mathbf{x}_{\text{finger}}$ and $\mathbf{x}_{\text{obj}}$ are the Cartesian position of each fingertip and the target object. Note that $\mathbf{x}_{\text{obj}}$ is available when we perform training in simulation. However, using this reward alone cannot ensure proper contact for grasping. For example, the robot can touch the object with the back of the hand rather than the palm and then get stuck in this local minimum. Therefore, we introduce a novel contact reward to guarantee meaningful contact behavior:

$$r_{\text{contact}} = \textbf{IsContact}(\text{thumb}, \text{object}) \ \textbf{AND} \left(\sum_{\text{finger}} \textbf{IsContact}(\text{finger}, \text{object}) \geq 2 \right). \tag{4.2}$$

This contact reward function outputs a boolean value in $\{0, 1\}$. It outputs 1 only if the thumb is in contact with the object and there is more than one finger in contact with the object. Intuitively, it encourages the robot to cage the object within its fingers. In this case, the robot can quickly find out stable grasping and lift the object to the target location. The lifting behavior is encouraged by

$$r_{\text{lift}} = r_{\text{contact}} \textbf{Lift}(\mathbf{x}_{\text{obj}}, \mathbf{x}_{\text{target}}). \tag{4.3}$$

The **Lift** function is basically in the form of Equation 4.1 and the main difference is that it will return a large reward value upon task completion. The overall reward function is a weighted combination of the terms above plus a control penalty:

$$\mathcal{R} = w_{\text{reach}} r_{\text{reach}} + w_{\text{contact}} r_{\text{contact}} + w_{\text{lift}} r_{\text{lift}} + w_{\text{penalty}} r_{\text{penalty}}. \tag{4.4}$$

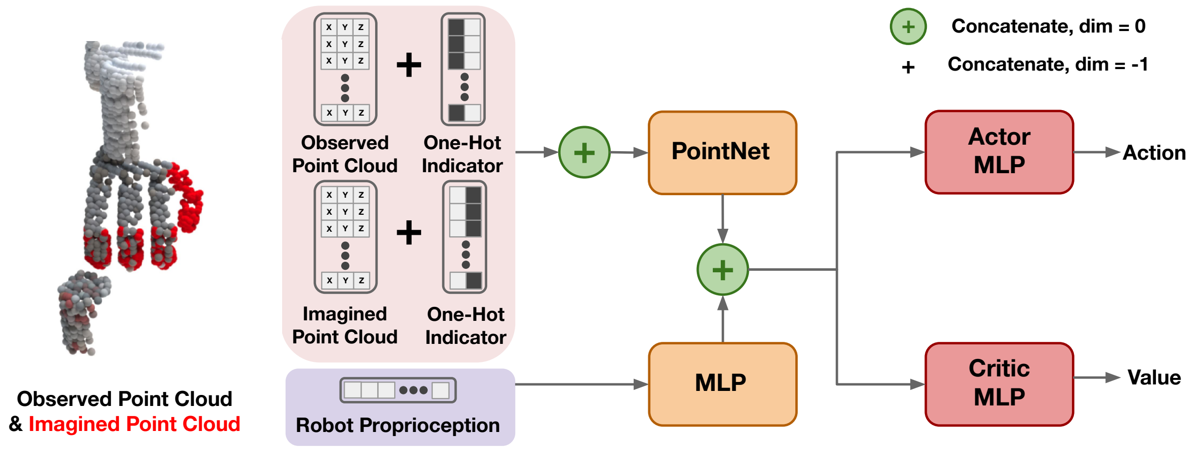
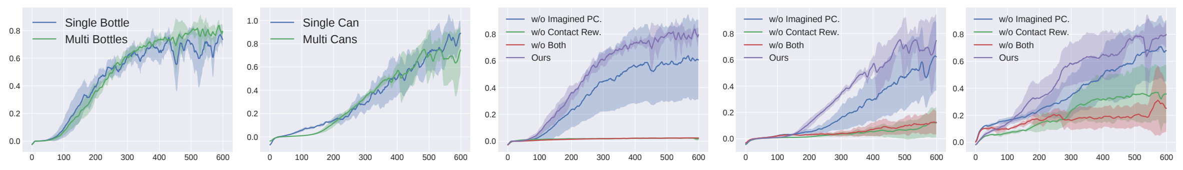


Figure 4.3. Network Architecture. Our feature extractor takes the observed point cloud, imagined point cloud, robot proprioception, and goal pose as input to output a feature embedding. Both actor and critic take the same feature to predict action and value. The red point represented the imaged point cloud of the robot hand. Note that our network does not require RGB information.

4.2.2 Imagined Hand Point Cloud

The usage of point cloud comes with two challenges. The first challenge is occlusion, which may occur to both the object under manipulation and the hand itself. When the robot hand is interacting with an object, the fingers may be occluded by the object. Since we do not assume tactile sensors in this work, this occlusion problem can be serious. The second challenge is the low point cloud resolution during RL training, where we can only use a limited number of points due to the memory limit. In this case, the number of points from the hand finger may not be adequate to precisely capture the spatial relationship between the robot and the object. We propose a simple yet effective method to handle both issues in a unified manner. Our idea is to use an imagined hand point cloud in the observation to help the robot to *see the interaction*.

We provide one example in Figure 4.3. Black points indicate the point cloud captured by the camera, in which some important details of fingers are missing. These missing details provide crucial information about the interaction. Though such interaction information can also be inferred by combining the information from both the proprioception and visual input, we find that the best way is to synthesize these missing details. Concretely, we can compute the pose



(a) (b) (c) (d) (e)

Figure 4.4. Training Curves. The first two plots show the single-object and multi-object training curve of (a) bottle category and (b) can category. The right three plots show the ablation results on the (c) grasping bottle (d) grasping can and (e) door opening. The x-axis is the training iterations and the y-axis is the normalized episodic return. The shaded area indicates standard error and the performance is evaluated on five random seeds.

for each finger link via forward kinematics given the joint position from the robot joint encoder and the robot kinematics model. Then, we synthesize the imagined point cloud (blue points in Figure 4.3) by sampling the points from the mesh of each finger link. This process is possible in both simulation and the real world.

4.2.3 Training

We adopt Proximal Policy Optimization (PPO) [126] to train the agent in simulation, and then deploy to real without real-world fine-tuning. The network architecture is illustrated in Figure 4.3. Both value and policy networks share the same visual feature extraction backbone. We concatenate the observed point cloud with the imagined hand point cloud together as the input to the feature extractor. We also attach a one-hot encoding to each point which indicates whether it is observed or an imagined point.

4.3 Experiments

4.3.1 Experimental Setup

Point Cloud Pre-processing: To enable smooth transfer from simulation to real-world, we apply the same data preprocessing procedure to the point cloud captured by the camera. It involves four steps: (i) Crop the point cloud to the work region with a manually-defined bounding-box; (ii) Down-sample the point cloud uniformly to 512 points; (iii) Add a distance-dependent

Table 4.1. Experiment on Multi-object Training. We evaluate the policy trained with single and multiple objects on the bottle (upper two rows) and can (bottom two rows) categories with point cloud input. We test them on both known or novel objects. The success rate is reported on 5 seeds.

Settings	Bottle		Can	
	Known Obj.	Novel Obj.	Known Obj.	Novel Obj.
Single Obj. Training	0.81 ± 0.09	0.60 ± 0.06	0.96 ± 0.04	0.63 ± 0.18
Multi Obj. Training	**0.83** ± **0.16**	**0.81** ± **0.15**	**0.93** ± **0.07**	**0.68** ± **0.09**

Gaussian noise to the simulated point cloud to improve the sim2real robustness; (iv) Transform point cloud from the camera frame to the robot base frame using camera pose. In simulation, we use the ground-truth camera pose with multiplicative noise for frame transformation. In the real-world, we perform hand-eye calibration to get the camera extrinsic parameters.

Evaluation Criterion: We evaluate the performance of a policy by its success rate. For grasping tasks, a task is considered a success if $d_{ot} < 0.05m$ in simulation, where d is the distance between object position and goal position. In real world, the task is considered as success if the XY position of the object is within 5cm from the target position and the height of the object is at least 15cm from table top. For the door opening, the task is considered as success if the door is opened to at least around 45 degrees.

EigenGrasp Baseline: We choose the EigenGrasp [33] as the grasp representation. Given an object mesh model, we use the GraspIt [89] to search valid grasp for Allegro Hand. Then, we use the RRTConnect [67] motion planner implemented in OMPL [143] to plan a joint trajectory to the pre-grasp pose and then plan a screw motion from pre-grasp pose to the grasp pose. Finally, we close all fingers based on searched grasp pose and lift the object to the target. Note that different from our approach, the baseline method **requires complete object model to search for grasps and ground-truth object pose** to align the grasp pose in the robot frame. To evaluate the performance of baseline on novel objects, we first build a grasp database on ShapeNet bottle and can categories using GraspIt. Given the sensory data of a new object, we search for the most similar objects in the dataset and use the query grasps for the novel object.

Here we compare the performance of our method with baselines in the real-world.

Training: We train RL in two settings for grasping: (i) training on a single object (ii) training on multiple objects jointly. For the single object grasping, we perform experiments on both “tomato soup can” and “mustard bottle” from YCB. For the multi-object training, we choose 10 objects from the can or bottle categories of ShapetNet. We randomly choose one object to train for each episode in the multi-object training. Here, we use can and bottle as experimental subjects since they represent two different basic grasping patterns [96] for anthropomorphic hand: precision grasp and power grasp. For door opening, we only train the policy on the door with fixed lever geometry.

4.3.2 Comparison of Single-object and Multi-object Training

We plot the training curve of RL in Figure 4.4 (a) and (b). In general, our method can learn to grasp and move the object to the goal pose within 600 iterations, where each iteration contains 20K environment steps. Then we evaluate the policy trained on both known and novel objects, and the results are shown in Table 4.1. We run 100 trials to compute the average success rate. On grasping known objects, we find that agent trained on single-object outperforms the agent trained on multi-object by a small margin, for both learning efficiency and final success rate. However, agent trained on multi-objects does much better at grasping novel objects. Our results suggest that using multiple object during training is important, and is of great importance for novel object generalization.

4.3.3 Ablation Results in Simulation

We ablate two key innovations of the work: the reward design with oracle contact and the imaged hand point cloud. We perform experiments on four different variants: (i) without imagined point cloud; (ii) without contact-based reward design; (iii) without both imagined point cloud and contact-based reward design; (iv) our standard approach with both techniques. Note that the variant (iii) is an approximation of [27] in our environments and tasks. We compare both

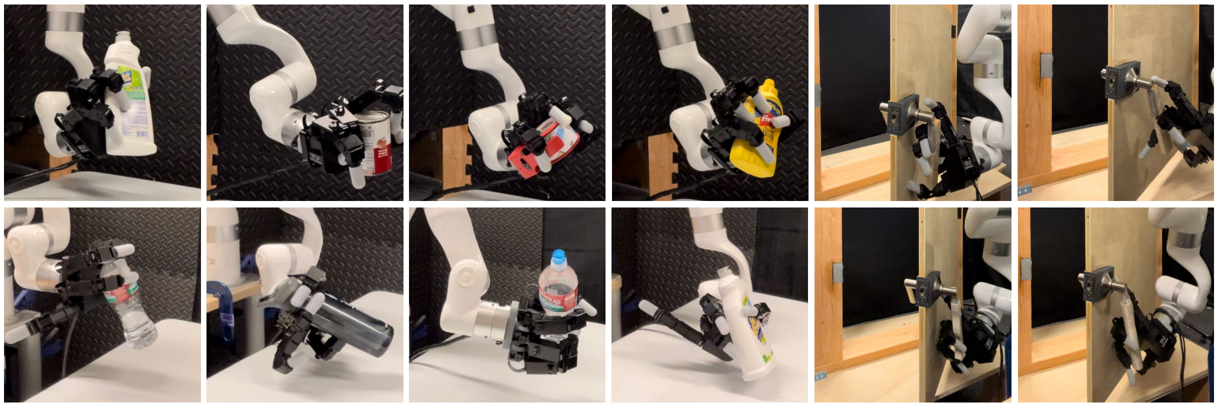


Figure 4.5. Real-experiment: We evaluate our point cloud policy on various unseen objects.

Table 4.2. Ablation Study: We investigate the influence of contact-based reward design and imaged point cloud. We evaluate the success rate on both known and novel objects under four settings: (i) without imaged point cloud; (ii) without contact reward; (iii) without both; and (iv) with both.

Settings	**Bottle**		**Can**		**Door**	
	Known Obj.	Novel Obj.	Known Obj.	Novel Obj.	Known Obj.	Novel Obj
w/o Imagined PC.	0.60 ± 0.46	0.56 ± 0.51	0.91 ± 0.17	0.63 ± 0.07	0.14 ± 0.28	0.11 ± 0.27
w/o Contact Rew.	0.00 ± 0.00	0.00 ± 0.00	0.06 ± 0.08	0.03 ± 0.06	0.21 ± 0.26	0.20 ± 0.25
w/o Both	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
Ours	**0.83 ± 0.16**	**0.81 ± 0.15**	**0.93 ± 0.07**	**0.68 ± 0.09**	**0.92 ± 0.06**	**0.79 ± 0.11**

the learning curve and the evaluation success rate of these four variants. For the grasping task, Figure 4.4 (c) and (d) show the results on bottle and can categories, and Table 4.2 shows the success rate. Our findings can be summarized as follows.

First, we find that contact reward information is of vital importance for training the point cloud RL policy on the multi-finger robot hand. Without using contact reward, the agent can hardly learn anything (red and green curve in the figure) and get nearly zero success rate during evaluation for both bottle and can categories. By encouraging contact between fingers and objects, the RL agent can avoid getting stuck in local minimums and learn meaningful manipulation behavior.

Second, the imagined point cloud can also improve the training and test performance for both categories, though it is not so important as the contact reward. As is shown in the learning curves of Figure 4.4 (c) and (d), the policy utilizing the imagined point cloud as input can learn

Table 4.3. Real-World Grasping Experiment. This experiment consists of 3 categories, which incorporate 26 objects: 10 bottles, 6 cans, and 10 other objects in multiple mixed categories.

Method	Bottle	Can	Mixed Category
EigenGrasp Oracle	0.66	0.45	N/A
EigenGrasp	0.50	0.41	N/A
Single Obj. Train	0.75 ± 0.06	0.75 ± 0.08	N/A
Multi Obj. Train	**0.87 ± 0.03**	**0.83 ± 0.13**	**0.73 ± 0.12**

Table 4.4. Real-World Door Opening Experiment. This experiment consists of 3 different doors, the first one on the left is used for training and the other two are for testing.

Settings	Original Door	Novel Door 1	Novel Door 2
Single Door Train	0.72 ± 0.07	0.60 ± 0.03	0.67 ± 0.01

faster in the early stage of the training and show much smaller variances. An interesting fact is that the imagined point cloud is more beneficial for the bottle category than the can category. One possible reason is that grasping a bottle requires multi-finger coordination to perform a power grasp. Such coordination is highly dependent on the detailed finger information provided by the imagined point cloud. The imagined point cloud can help the agent to better see the fingers even if they are occluded by the object, e.g., fingers behind the object.

The experiments on the Door Opening task also support these findings. As shown in Figure 4.4 (e), the policy trained with contact-based reward and imaged hand point cloud outperforms other ablation methods, which also demonstrates the effectiveness of our two key designs. Compared with the grasping experiments, we can observe larger variations during policy training. One possible reason is that door opening suffers from heavier occlusion than grasping when the door lever is grasped by the robot hand, which influences the temporal consistency of the PointNet feature.

4.3.4 Real-World Evaluation

We perform sim2real experiments to evaluate the performance of our method in the real world. As is shown in Figure 4.2, we attached an Allegro hand onto a XArm-6 robot arm to grasp the object on the front table. We apply the same data-preprocessing steps for both simulated

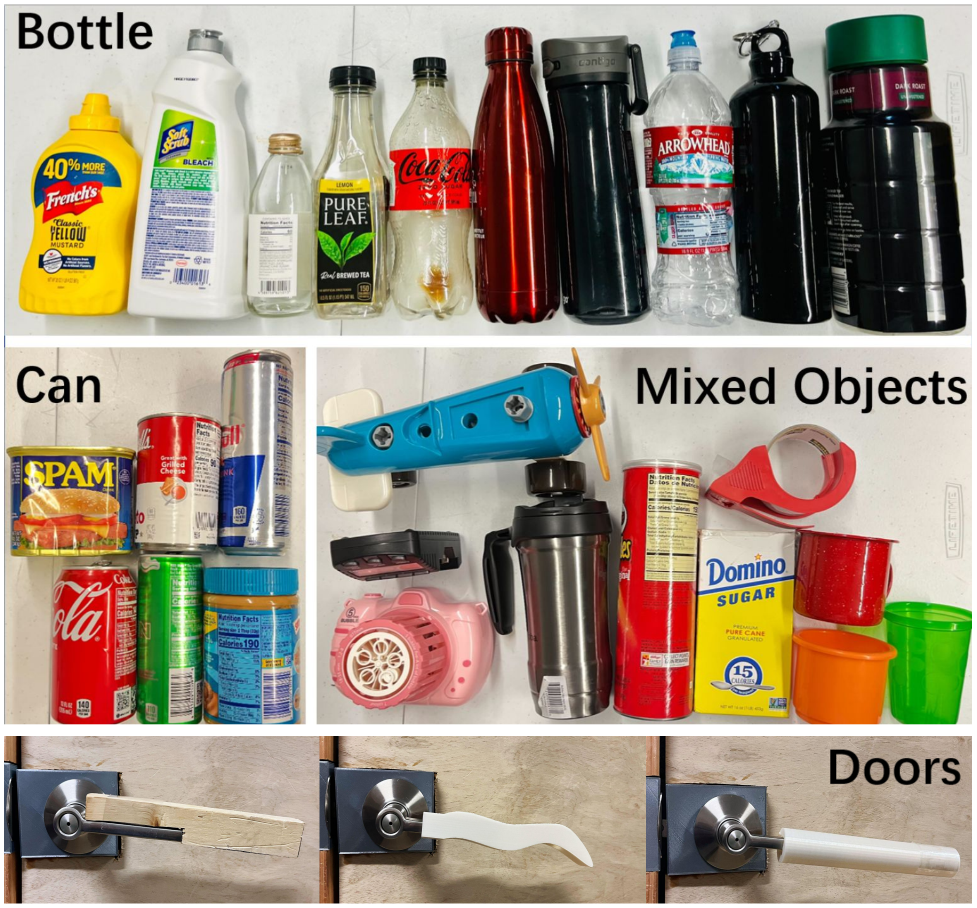


Figure 4.6. Objects Sets for Real World Experiments. The objects within each category for real-world evaluation. The bottle, can and mixed objects are used for grasping while the doors are used for door opening.

environment and real-world as mentioned in Sec.4.3.1.

The task execution sequence is visualized on the bottom row of Figure 4.1. Both Single Obj. Training and Multi Obj. Training will be evaluated in the corresponding category, while policies training with multiple objects even evaluated in the Mixed category with unseen objects. We run 10 independent trials seeds for each object-policy pair.

The real-world evaluation results are shown in Table 4.3 and Table 4.4. The policy trained in the simulator with point cloud input can directly transfer to the real-world without fine-tuning. For both two tasks, our policy can even deploy on objects that have never been seen during the training. Moreover, We find that for grasping, training on multiple objects can ensure better performance than single-object training. Compared with the EigenGrasp baseline shown in Table 4.3, our policies trained on both single-object and multi-object performs better, even if the EigenGrasp baseline use the object model information. Since EigenGrasp + motion planning is a open-loop manipulation policy, small error in object and scene modeling, e.g. initial object pose or object geometry, can lead to a failure in final grasp results. In contrast, our methods works in a closed-loop fashion with point cloud observation, which does not require privilege knowledge about the object.

4.4 Simulation Environment Details

This section details the simulation environment and training procedures used in our experiments.

4.4.1 Object Set.

For single-object experiments, we utilize object models from the YCB dataset [17]. For multi-object experiments, we curate a dataset of 50 objects from the ShapeNet dataset [19], using 10 for training and 40 for testing in simulation. The test objects used in real-world experiments are visualized in Figure 4.6.

Object Pre-processing. To ensure compatibility with the simulation environment and physical plausibility, we preprocess all object models using the following steps:

Scaling: ShapeNet objects lack real-world scale information. We address this by scaling each object based on its bounding box diagonal length, ensuring consistency within object categories.

Convex Decomposition: We decompose both YCB and ShapeNet objects into convex parts using V-HACD [81] with default parameters. This facilitates stable and efficient physical simulation. Objects with more than 40 convex parts are excluded.

4.4.2 Reward.

As described in Section 4.2.1, the reward function for our manipulation task comprises four components:

$$\mathcal{R} = w_{\text{reach}} r_{\text{reach}} + w_{\text{contact}} r_{\text{contact}} + w_{\text{lift}} r_{\text{lift}} + w_{\text{penalty}} r_{\text{penalty}}. \tag{4.5}$$

Reach Reward (r_{reach}): Encourages the robot hand to approach both the object and the target location.

Contact Reward (r_{contact}): Rewards establishing and maintaining stable contact with the object.

Lift Reward (r_{lift}): Rewards lifting the object, calculated as the difference between the object's current and initial height ($h_{\text{current}} - h_{\text{init}}$). This reward is active only when the contact reward is non-zero, ensuring proper grasping before lifting. **Action Penalty (r_{penalty}):** Discourages large control actions and promotes smooth movements ($-||a||_2^2$).

4.5 Reinforcement Learning Details

This section details the reinforcement learning algorithm and network architecture used in our experiments.

4.5.1 RL Training.

We utilize Proximal Policy Optimization (PPO) for training our point cloud-based manipulation policy. Table 4.5 summarizes the PPO hyperparameters. We use an on-policy training setup for all experiments except for the policy distillation experiments.

4.5.2 Network Architecture

Our RL agent employs a PointNet-based architecture for processing point cloud inputs. Figure 4.3 illustrates the network structure. We concatenate visual features extracted by PointNet

Table 4.5. DexPoint Hyper-parameters of PPO reinforcement learning training.

Parameter	Mini-Batch Size	Learning Rate	Clip Range	Horizon	Epoch	Steps
Value	500	3e-4	0.8	200	10	10

Table 4.6. DexPoint Network Architecture of PPO Reinforcement Learning Training.

Module	Architecture	Output Dim
Visual Feature Extractor	PointNet Local Channel: (64, 128, 256)	256
	PointNet Global Channel: (256,)	256
State Feature Extractor	MLP: (64, 64)	64
Actor	MLP: (64, 64)	64
Critic	MLP: (64, 64)	64

with proprioceptive features processed by a Multi-Layer Perceptron (MLP). This combined feature representation is then shared by both the value and policy networks for predicting state values and actions, respectively. Table 4.6 provides details of the network architecture.

Acknowledgement

Chapter 4, in full, is a reprint of the material published in the 2022 Conference on Robot Learning (CoRL): "DexPoint: Generalizable Point Cloud Reinforcement Learning for Sim-to-Real Dexterous Manipulation" (Yuzhe Qin*, Binghao Huang*, Zhao-Heng Yin, Hao Su, Xiaolong Wang). The dissertation author was the primary investigator and author of this paper.

Chapter 5

Finale

In conclusion, my research endeavors in the complex field of dexterous robotic manipulation surpass mere technological advancement, aiming to bridge the gap between robotic functionality and human-like subtlety. The methodologies we've developed, grounded in learning from direct human teleoperation, extracting nuanced maneuvers from human videos, and the innovative implementation of visuotactile integration, tackle the multifaceted challenges inherent in robotic manipulation. This work does not only strive to solve the intricate technical issues but also significantly reduces the gap between simulations and the ever-unpredictable real-world scenarios.

The journey thus far has been enlightening, revealing both expected and unforeseen challenges. Our approach has centered on creating systems that can adapt and learn in conditions that closely mimic human interaction environments. The use of human teleoperation data has provided a direct pathway to understanding and replicating human-like dexterity in robots, offering an immediate and impactful avenue for robotic teaching. Meanwhile, the analysis of human manipulation videos has unlocked a treasure trove of data, enabling our robots to learn from a vast array of human actions without the need for direct interaction or intervention.

The integration of visuotactile feedback within these systems is particularly promising. By combining tactile sensations with visual inputs, we have started to see robots that can perform tasks with a sensitivity and adaptability that were previously unattainable. This breakthrough is pivotal in our quest to minimize the discrepancies between simulated training and real-world application, ensuring that the skills acquired by robots are not only robust but also universally applicable.

Looking ahead, the potential expansions of this research are vast and varied. I am particularly excited about further leveraging the rich data provided by human demonstrations to enhance the generalizability of robotic manipulation policies. There is an untapped potential in extending the use of human motion priors, which could allow robots to adapt their movements and strategies across a wider range of objects and scenarios, effectively bridging the gap between different realities, such as transitioning from simulated environments to the tangible world.

Moreover, the concept of assistive neural teleoperation is set to revolutionize the way we interact with and utilize robots. By embedding goal-conditioned policies within teleoperation systems, we can significantly refine how robots interpret and execute human commands. This advancement could lead to more efficient, effective, and intuitive robotic systems, capable of performing complex tasks with minimal human oversight.

As we continue to push the boundaries of what robotic systems can achieve, the implications of this research extend far beyond the confines of laboratories and test environments. The practical applications in industries such as manufacturing, healthcare, and domestic service are profound. Robots equipped with the ability to perform complex, delicate tasks reliably and safely could transform numerous aspects of everyday life and work.

In essence, the core of my research lies in its steadfast dedication to transforming robotic manipulation into a domain where robots are not merely tools but are partners and collaborators, enhancing human capabilities and taking on roles that complement and augment human efforts. The road ahead is fraught with challenges, yet it is replete with immense potential. I remain committed to exploring these opportunities with the utmost rigor and creativity, driven by the vision of creating robotic systems that not only emulate human dexterity but also surpass human limitations in precision, consistency, and adaptability.

Bibliography

[1] Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

[2] OpenAI: 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. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

[3] Dafni Antotsiou, Guillermo Garcia-Hernando, and Tae-Kyun Kim. Task-oriented hand motion retargeting for dexterous manipulation imitation. In *ECCV Workshops*, 2018.

[4] Reuben M Aronson and Henny Admoni. Gaze complements control input for goal prediction during assisted teleoperation. In *Robotics science and systems*, 2022.

[5] Sridhar Pandian Arunachalam, Irmak Güzey, Soumith Chintala, and Lerrel Pinto. Holo-dex: Teaching dexterity with immersive mixed reality. *arXiv preprint arXiv:2210.06463*, 2022.

[6] Sridhar Pandian Arunachalam, Irmak Güzey, Soumith Chintala, and Lerrel Pinto. Holo-dex: Teaching dexterity with immersive mixed reality. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5962–5969. IEEE, 2023.

[7] Sridhar Pandian Arunachalam, Sneha Silwal, Ben Evans, and Lerrel Pinto. Dexterous imitation made easy: A learning-based framework for efficient dexterous manipulation. *arXiv preprint arXiv:2203.13251*, 2022.

[8] Sridhar Pandian Arunachalam, Sneha Silwal, Ben Evans, and Lerrel Pinto. Dexterous imitation made easy: A learning-based framework for efficient dexterous manipulation. In *2023 ieee international conference on robotics and automation (icra)*, pages 5954–5961. IEEE, 2023.

[9] Yusuf Aytar, Tobias Pfaff, David Budden, Thomas Paine, Ziyu Wang, and Nando De Freitas. Playing hard exploration games by watching youtube. *Advances in neural information*

processing systems, 31, 2018.

[10] Chen Bao, Helin Xu, Yuzhe Qin, and Xiaolong Wang. Dexart: Benchmarking generalizable dexterous manipulation with articulated objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21190–21200, 2023.

[11] Homanga Bharadhwaj, Abhinav Gupta, and Shubham Tulsiani. Visual affordance prediction for guiding robot exploration. *arXiv preprint arXiv:2305.17783*, 2023.

[12] Homanga Bharadhwaj, Jay Vakil, Mohit Sharma, Abhinav Gupta, Shubham Tulsiani, and Vikash Kumar. Roboagent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking. *arXiv preprint arXiv:2309.01918*, 2023.

[13] Antonio Bicchi. Hands for dexterous manipulation and robust grasping: A difficult road toward simplicity. *IEEE Transactions on robotics and automation*, 2000.

[14] Kevin Black, Mitsuhiko Nakamoto, Pranav Atreya, Homer Walke, Chelsea Finn, Aviral Kumar, and Sergey Levine. Zero-shot robotic manipulation with pretrained image-editing diffusion models. *arXiv preprint arXiv:2310.10639*, 2023.

[15] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3722–3731, 2017.

[16] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[17] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. The ycb object and model set: Towards common benchmarks for manipulation research. In *2015 international conference on advanced robotics (ICAR)*, pages 510–517. IEEE, 2015.

[18] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

[19] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[20] Henry J Charlesworth and Giovanni Montana. Solving challenging dexterous manipulation tasks with trajectory optimisation and reinforcement learning. In *International Conference on Machine Learning*, pages 1496–1506. PMLR, 2021.

[21] Nikhil Chavan-Dafle and Alberto Rodriguez. Sampling-based planning of in-hand manipulation with external pushes. In *Robotics Research: The 18th International Symposium ISRR*, pages 523–539. Springer, 2020.

[22] Yevgen Chebotar, Ankur Handa, Viktor Makoviychuk, Miles Macklin, Jan Issac, Nathan Ratliff, and Dieter Fox. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8973–8979. IEEE, 2019.

[23] Claire Chen, Preston Culbertson, Marion Lepert, Mac Schwager, and Jeannette Bohg. Trajectotree: Trajectory optimization meets tree search for planning multi-contact dexterous manipulation. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8262–8268. IEEE, 2021.

[24] Linghao Chen, Yuzhe Qin, Xiaowei Zhou, and Hao Su. Easyhec: Accurate and automatic hand-eye calibration via differentiable rendering and space exploration. *IEEE Robotics and Automation Letters*, 2023.

[25] Sean Chen, Jensen Gao, Siddharth Reddy, Glen Berseth, Anca D Dragan, and Sergey Levine. Asha: Assistive teleoperation via human-in-the-loop reinforcement learning. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 7505–7512. IEEE, 2022.

[26] Tao Chen, Megha Tippur, Siyang Wu, Vikash Kumar, Edward Adelson, and Pulkit Agrawal. Visual dexterity: In-hand reorientation of novel and complex object shapes. *Science Robotics*, 8(84):eadc9244, 2023.

[27] Tao Chen, Jie Xu, and Pulkit Agrawal. A system for general in-hand object re-orientation. In *Conference on Robot Learning*, pages 297–307. PMLR, 2022.

[28] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

[29] Yuanpei Chen, Chen Wang, Li Fei-Fei, and C Karen Liu. Sequential dexterity: Chaining dexterous policies for long-horizon manipulation. *arXiv preprint arXiv:2309.00987*, 2023.

[30] Zoey Chen, Sho Kiami, Abhishek Gupta, and Vikash Kumar. Genaug: Retargeting behaviors to unseen situations via generative augmentation. *arXiv preprint arXiv:2302.06671*, 2023.

[31] Zoey Chen, Karl Van Wyk, Yu-Wei Chao, Wei Yang, Arsalan Mousavian, Abhishek Gupta, and Dieter Fox. Learning robust real-world dexterous grasping policies via implicit shape augmentation. *arXiv preprint arXiv:2210.13638*, 2022.

[32] Ching-An Cheng, Mustafa Mukadam, Jan Issac, Stan Birchfield, Dieter Fox, Byron Boots, and Nathan Ratliff. Rmp flow: A computational graph for automatic motion policy generation. In *Algorithmic Foundations of Robotics XIII: Proceedings of the 13th Workshop on the Algorithmic Foundations of Robotics 13*, pages 441–457. Springer, 2020.

[33] Matei Ciocarlie, Corey Goldfeder, and Peter Allen. Dexterous grasping via eigengrasps: A low-dimensional approach to a high-complexity problem. In *Robotics: Science and systems manipulation workshop-sensing and adapting to the real world*, 2007.

[34] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. arxiv 2018. *arXiv preprint arXiv:1805.09501*, 1805.

[35] Murtaza Dalal, Ajay Mandlekar, Caelan Garrett, Ankur Handa, Ruslan Salakhutdinov, and Dieter Fox. Imitating task and motion planning with visuomotor transformers. *arXiv preprint arXiv:2305.16309*, 2023.

[36] Brian Danchilla and Brian Danchilla. Three. js framework. *Beginning WebGL for HTML5*, pages 173–203, 2012.

[37] Robin Deits. Meshcat. https://github.com/rdeits/meshcat, 2018.

[38] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[39] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[40] Guanglong Du, Ping Zhang, Jianhua Mai, and Zeling Li. Markerless kinect-based hand tracking for robot teleoperation. *International Journal of Advanced Robotic Systems*, 9(2):36, 2012.

[41] Frederik Ebert, Yanlai Yang, Karl Schmeckpeper, Bernadette Bucher, Georgios Georgakis, Kostas Daniilidis, Chelsea Finn, and Sergey Levine. Bridge data: Boosting generalization of robotic skills with cross-domain datasets. *arXiv preprint arXiv:2109.13396*, 2021.

[42] Jean Elsner, Gerhard Reinerth, Luis Figueredo, Abdeldjallil Naceri, Ulrich Walter, and

Sami Haddadin. Parti-a haptic virtual reality control station for model-mediated robotic applications. *Frontiers in Virtual Reality*, 3, 2022.

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

[44] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030, 2016.

[45] Zaid Gharaybeh, Howard Chizeck, and Andrew Stewart. *Telerobotic control in virtual reality*. IEEE, 2019.

[46] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18995–19012, 2022.

[47] Jiayuan Gu, Fanbo Xiang, Xuanlin Li, Zhan Ling, Xiqiang Liu, Tongzhou Mu, Yihe Tang, Stone Tao, Xinyue Wei, Yunchao Yao, et al. Maniskill2: A unified benchmark for generalizable manipulation skills. *arXiv preprint arXiv:2302.04659*, 2023.

[48] Ankur Handa, Arthur Allshire, Viktor Makoviychuk, Aleksei Petrenko, Ritvik Singh, Jingzhou Liu, Denys Makoviichuk, Karl Van Wyk, Alexander Zhurkevich, Balakumar Sundaralingam, et al. Dextreme: Transfer of agile in-hand manipulation from simulation to reality. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5977–5984. IEEE, 2023.

[49] 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 *ICRA*, 2020.

[50] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.

[51] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[52] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[53] Hooman Hedayati, Michael Walker, and Daniel Szafir. Improving collocated robot teleoperation with augmented reality. In *International Conference on Human-Robot Interaction*, 2018.

[54] Daniel Ho, Kanishka Rao, Zhuo Xu, Eric Jang, Mohi Khansari, and Yunfei Bai. Retinagan: An object-aware approach to sim-to-real transfer. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10920–10926. IEEE, 2021.

[55] Binghao Huang, Yuanpei Chen, Tianyu Wang, Yuzhe Qin, Yaodong Yang, Nikolay Atanasov, and Xiaolong Wang. Dynamic handover: Throw and catch with bimanual hands. *arXiv preprint arXiv:2309.05655*, 2023.

[56] Wenlong Huang, Igor Mordatch, Pieter Abbeel, and Deepak Pathak. Generalization in dexterous manipulation via geometry-aware multi-task learning. *arXiv preprint arXiv:2111.03062*, 2021.

[57] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023.

[58] Chieko Sarah Imai, Minghao Zhang, Yuchen Zhang, Marcin Kierebiński, Ruihan Yang, Yuzhe Qin, and Xiaolong Wang. Vision-guided quadrupedal locomotion in the wild with multi-modal delay randomization. In *2022 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 5556–5563. IEEE, 2022.

[59] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2):3019–3026, 2020.

[60] Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pages 991–1002. PMLR, 2022.

[61] Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. Vima: General robot manipulation with multimodal prompts. *arXiv*, 2022.

[62] Amlan Kar, Aayush Prakash, Ming-Yu Liu, Eric Cameracci, Justin Yuan, Matt Rusiniak, David Acuna, Antonio Torralba, and Sanja Fidler. Meta-sim: Learning to generate synthetic datasets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4551–4560, 2019.

[63] Manuel Kaspar, Juan D Muñoz Osorio, and Jürgen Bock. Sim2real transfer for reinforcement learning without dynamics randomization. In *2020 IEEE/RSJ International*

Conference on Intelligent Robots and Systems (IROS), pages 4383–4388. IEEE, 2020.

[64] Bachir El Khadir, Jake Varley, and Vikas Sindhwani. Teleoperator imitation with continuous-time safety. *arXiv preprint arXiv:1905.09499*, 2019.

[65] Jonathan Kofman, Siddharth Verma, and Xianghai Wu. Robot-manipulator teleoperation by markerless vision-based hand-arm tracking. *International Journal of Optomechatronics*, 2007.

[66] Jonathan Kofman, Xianghai Wu, Timothy J Luu, and Siddharth Verma. Teleoperation of a robot manipulator using a vision-based human-robot interface. *IEEE transactions on industrial electronics*, 52(5):1206–1219, 2005.

[67] James J Kuffner and Steven M LaValle. Rrt-connect: An efficient approach to single-query path planning. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 2, pages 995–1001. IEEE, 2000.

[68] Sateesh Kumar, Jonathan Zamora, Nicklas Hansen, Rishabh Jangir, and Xiaolong Wang. Graph inverse reinforcement learning from diverse videos. In *Conference on Robot Learning*, pages 55–66. PMLR, 2023.

[69] Vikash Kumar and Emanuel Todorov. Mujoco haptix: A virtual reality system for hand manipulation. In *International Conference on Humanoid Robots (Humanoids)*, 2015.

[70] Vikash Kumar, Emanuel Todorov, and Sergey Levine. Optimal control with learned local models: Application to dexterous manipulation. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 378–383. IEEE, 2016.

[71] Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Michael Lingelbach, Jiankai Sun, et al. Behavior-1k: A benchmark for embodied ai with 1,000 everyday activities and realistic simulation. In *6th Annual Conference on Robot Learning*, 2022.

[72] 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. *IEEE Transactions on Cybernetics*, 2022.

[73] 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 *ICRA*, 2019.

[74] Jacky Liang, Ankur Handa, Karl Van Wyk, Viktor Makoviychuk, Oliver Kroemer, and Dieter Fox. In-hand object pose tracking via contact feedback and gpu-accelerated robotic

simulation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6203–6209. IEEE, 2020.

[75] Jacky Liang, Saumya Saxena, and Oliver Kroemer. Learning active task-oriented exploration policies for bridging the sim-to-real gap. *arXiv preprint arXiv:2006.01952*, 2020.

[76] Hangxin Liu, Xu Xie, Matt Millar, Mark Edmonds, Feng Gao, Yixin Zhu, Veronica J Santos, Brandon Rothrock, and Song-Chun Zhu. A glove-based system for studying hand-object manipulation via joint pose and force sensing. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6617–6624. IEEE, 2017.

[77] Hangxin Liu, Zhenliang Zhang, Xu Xie, Yixin Zhu, Yue Liu, Yongtian Wang, and Song-Chun Zhu. High-fidelity grasping in virtual reality using a glove-based system. In *2019 international conference on robotics and automation (icra)*, pages 5180–5186. IEEE, 2019.

[78] Ziyuan Liu, Wei Liu, Yuzhe Qin, Fanbo Xiang, Minghao Gou, Songyan Xin, Maximo A Roa, Berk Calli, Hao Su, Yu Sun, et al. Ocrtoc: A cloud-based competition and benchmark for robotic grasping and manipulation. *IEEE Robotics and Automation Letters*, 7(1):486–493, 2021.

[79] Corey Lynch, Ayzaan Wahid, Jonathan Tompson, Tianli Ding, James Betker, Robert Baruch, Travis Armstrong, and Pete Florence. Interactive language: Talking to robots in real time. *IEEE Robotics and Automation Letters*, 2023.

[80] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, et al. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021.

[81] Khaled Mamou and Faouzi Ghorbel. A simple and efficient approach for 3d mesh approximate convex decomposition. In *2009 16th IEEE international conference on image processing (ICIP)*, pages 3501–3504. IEEE, 2009.

[82] Zhao Mandi, Homanga Bharadhwaj, Vincent Moens, Shuran Song, Aravind Rajeswaran, and Vikash Kumar. Cacti: A framework for scalable multi-task multi-scene visual imitation learning. *arXiv preprint arXiv:2212.05711*, 2022.

[83] Priyanka Mandikal and Kristen Grauman. Dexvip: Learning dexterous grasping with human hand pose priors from video. In *Conference on Robot Learning*, pages 651–661. PMLR, 2022.

[84] Ajay Mandlekar, Soroush Nasiriany, Bowen Wen, Iretiayo Akinola, Yashraj Narang, Linxi

Fan, Yuke Zhu, and Dieter Fox. Mimicgen: A data generation system for scalable robot learning using human demonstrations. *arXiv preprint arXiv:2310.17596*, 2023.

[85] Ajay Mandlekar, Danfei Xu, Roberto Martín-Martín, Yuke Zhu, Li Fei-Fei, and Silvio Savarese. Human-in-the-loop imitation learning using remote teleoperation. *arXiv preprint arXiv:2012.06733*, 2020.

[86] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. *arXiv preprint arXiv:2108.03298*, 2021.

[87] Ajay Mandlekar, Yuke Zhu, Animesh Garg, Jonathan Booher, Max Spero, Albert Tung, Julian Gao, John Emmons, Anchit Gupta, Emre Orbay, et al. Roboturk: A crowdsourcing platform for robotic skill learning through imitation. In *Conference on Robot Learning*, pages 879–893. PMLR, 2018.

[88] Oier Mees, Markus Merklinger, Gabriel Kalweit, and Wolfram Burgard. Adversarial skill networks: Unsupervised robot skill learning from video. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4188–4194. IEEE, 2020.

[89] Andrew T Miller and Peter K Allen. Graspit! a versatile simulator for robotic grasping. *IEEE Robotics & Automation Magazine*, 11(4):110–122, 2004.

[90] Kaichun Mo, Yuzhe Qin, Fanbo Xiang, Hao Su, and Leonidas Guibas. O2o-afford: Annotation-free large-scale object-object affordance learning. In *Conference on robot learning*, pages 1666–1677. PMLR, 2022.

[91] Igor Mordatch, Zoran Popović, and Emanuel Todorov. Contact-invariant optimization for hand manipulation. In *Proceedings of the ACM SIGGRAPH/Eurographics symposium on computer animation*, pages 137–144, 2012.

[92] Malte Mosbach, Kara Moraw, and Sven Behnke. Accelerating interactive human-like manipulation learning with GPU-based simulation and high-quality demonstrations. In *Humanoids*, 2022.

[93] Tongzhou Mu, Zhan Ling, Fanbo Xiang, Derek Yang, Xuanlin Li, Stone Tao, Zhiao Huang, Zhiwei Jia, and Hao Su. Maniskill: Generalizable manipulation skill benchmark with large-scale demonstrations. *arXiv preprint arXiv:2107.14483*, 2021.

[94] Katharina Muelling, Arun Venkatraman, Jean-Sebastien Valois, John Downey, Jeffrey Weiss, Shervin Javdani, Martial Hebert, Andrew B Schwartz, Jennifer L Collinger, and J Andrew Bagnell. Autonomy infused teleoperation with application to bci manipulation. *arXiv preprint arXiv:1503.05451*, 2015.

[95] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

[96] John R Napier. The prehensile movements of the human hand. *The Journal of bone and joint surgery. British volume*, 38(4):902–913, 1956.

[97] Anh Nguyen, Dimitrios Kanoulas, Luca Muratore, Darwin G Caldwell, and Nikos G Tsagarakis. Translating videos to commands for robotic manipulation with deep recurrent neural networks. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3782–3788. IEEE, 2018.

[98] Günter Niemeyer, Carsten Preusche, Stefano Stramigioli, and Dongjun Lee. Telerobotics. *Springer handbook of robotics*, pages 1085–1108, 2016.

[99] Allison M Okamura, Niels Smaby, and Mark R Cutkosky. An overview of dexterous manipulation. In *IEEE International Conference on Robotics and Automation. Symposia Proceedings*, 2000.

[100] OpenAI, Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafa l Józefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, Jonas Schneider, Szymon Sidor, Josh Tobin, Peter Welinder, Lilian Weng, and Wojciech Zaremba. Learning dexterous in-hand manipulation. *arXiv*, 2018.

[101] Abhishek Padalkar, Acorn Pooley, Ajinkya Jain, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anikait Singh, Anthony Brohan, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.

[102] Simone Parisi, Aravind Rajeswaran, Senthil Purushwalkam, and Abhinav Gupta. The unsurprising effectiveness of pre-trained vision models for control. In *International Conference on Machine Learning*, pages 17359–17371. PMLR, 2022.

[103] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. A. Osman, Dimitrios Tzionas, and Michael J. Black. Expressive body capture: 3d hands, face, and body from a single image. In *CVPR*, 2019.

[104] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 3803–3810. IEEE, 2018.

[105] Haozhi Qi, Ashish Kumar, Roberto Calandra, Yi Ma, and Jitendra Malik. In-hand object rotation via rapid motor adaptation. In *Conference on Robot Learning*, pages 1722–1732. PMLR, 2023.

[106] Haozhi Qi, Brent Yi, Sudharshan Suresh, Mike Lambeta, Yi Ma, Roberto Calandra, and Jitendra Malik. General in-hand object rotation with vision and touch. In *Conference on Robot Learning*, pages 2549–2564. PMLR, 2023.

[107] Yuzhe Qin, Rui Chen, Hao Zhu, Meng Song, Jing Xu, and Hao Su. S4g: Amodal single-view single-shot se (3) grasp detection in cluttered scenes. In *Conference on robot learning*, pages 53–65. PMLR, 2020.

[108] Yuzhe Qin, Binghao Huang, Zhao-Heng Yin, Hao Su, and Xiaolong Wang. Dexpoint: Generalizable point cloud reinforcement learning for sim-to-real dexterous manipulation. In *Conference on Robot Learning*, pages 594–605. PMLR, 2023.

[109] Yuzhe Qin, Hao Su, and Xiaolong Wang. From one hand to multiple hands: Imitation learning for dexterous manipulation from single-camera teleoperation. *RA-L*, 7(4):10873–10881, 2022.

[110] Yuzhe Qin, Yueh-Hua Wu, Shaowei Liu, Hanwen Jiang, Ruihan Yang, Yang Fu, and Xiaolong Wang. Dexmv: Imitation learning for dexterous manipulation from human videos. In *European Conference on Computer Vision*, pages 570–587. Springer, 2022.

[111] Yuzhe Qin, Wei Yang, Binghao Huang, Karl Van Wyk, Hao Su, Xiaolong Wang, Yu-Wei Chao, and Dietor Fox. Anyteleop: A general vision-based dexterous robot arm-hand teleoperation system. *arXiv preprint arXiv:2307.04577*, 2023.

[112] Ahmed H Qureshi, Jacob J Johnson, Yuzhe Qin, Taylor Henderson, Byron Boots, and Michael C Yip. Composing task-agnostic policies with deep reinforcement learning. *arXiv preprint arXiv:1905.10681*, 2019.

[113] Ilija Radosavovic, Tete Xiao, Stephen James, Pieter Abbeel, Jitendra Malik, and Trevor Darrell. Real-world robot learning with masked visual pre-training. *arXiv preprint arXiv:2210.03109*, 2022.

[114] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. In *RSS*, 2018.

[115] Daniel Rakita, Bilge Mutlu, and Michael Gleicher. A motion retargeting method for effective mimicry-based teleoperation of robot arms. In *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*, pages 361–370, 2017.

[116] Daniel Rakita, Bilge Mutlu, and Michael Gleicher. Remote telemanipulation with adapting viewpoints in visually complex environments. *Robotics: Science and Systems XV*, 2019.

[117] Kanishka Rao, Chris Harris, Alex Irpan, Sergey Levine, Julian Ibarz, and Mohi Khansari.

Rl-cyclegan: Reinforcement learning aware simulation-to-real. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11157–11166, 2020.

[118] Nathan D Ratliff, Jan Issac, Daniel Kappler, Stan Birchfield, and Dieter Fox. Riemannian motion policies. *arXiv preprint arXiv:1801.02854*, 2018.

[119] WONIK Robotics. Allegro robot hand. https://www.wonikrobotics.com/research-robot-hand.

[120] Active Robots. Ar10 humanoid robotic hand. https://www.active-robots.com/ar10-humanoid-robotic-hand.html.

[121] Yu Rong, Takaaki Shiratori, and Hanbyul Joo. Frankmocap: Fast monocular 3d hand and body motion capture by regression and integration. *arXiv preprint arXiv:2008.08324*, 2020.

[122] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

[123] M Salvato, Negin Heravi, Allison M Okamura, and Jeannette Bohg. Predicting hand-object interaction for improved haptic feedback in mixed reality. *IEEE Robotics and Automation Letters*, 7(2):3851–3857, 2022.

[124] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9339–9347, 2019.

[125] Karl Schmeckpeper, Oleh Rybkin, Kostas Daniilidis, Sergey Levine, and Chelsea Finn. Reinforcement learning with videos: Combining offline observations with interaction. *arXiv preprint arXiv:2011.06507*, 2020.

[126] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[127] Schunk. Schunk svh robot hand. https://schunk.com/us_en/gripping-systems/highlights/svh.

[128] Max Schwarzer, Nitarshan Rajkumar, Michael Noukhovitch, Ankesh Anand, Laurent Charlin, R Devon Hjelm, Philip Bachman, and Aaron C Courville. Pretraining representations for data-efficient reinforcement learning. *Advances in Neural Information Processing*

Systems, 34:12686–12699, 2021.

[129] 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 *2018 IEEE international conference on robotics and automation (ICRA)*, pages 1134–1141. IEEE, 2018.

[130] ShadowRobot. Shadowrobot dexterous hand. In *https://www.shadowrobot.com/products/dexterous-hand/*, 2005.

[131] Rutav Shah and Vikash Kumar. Rrl: Resnet as representation for reinforcement learning. *arXiv preprint arXiv:2107.03380*, 2021.

[132] Lin Shao, Toki Migimatsu, Qiang Zhang, Karen Yang, and Jeannette Bohg. Concept2robot: Learning manipulation concepts from instructions and human demonstrations. *The International Journal of Robotics Research*, 40(12-14):1419–1434, 2021.

[133] Kenneth Shaw, Shikhar Bahl, and Deepak Pathak. Videodex: Learning dexterity from internet videos. In *Conference on Robot Learning*, pages 654–665. PMLR, 2023.

[134] Kenneth Shaw and D Pathak. Leap hand: Low-cost, efficient, and anthropomorphic hand for robot learning. *Submission, ICRA*, 2023.

[135] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

[136] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Cliport: What and where pathways for robotic manipulation. In *Conference on Robot Learning*, pages 894–906. PMLR, 2022.

[137] Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. Progprompt: Generating situated robot task plans using large language models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11523–11530. IEEE, 2023.

[138] Aravind Sivakumar, Kenneth Shaw, and Deepak Pathak. Robotic telekinesis: learning a robotic hand imitator by watching humans on youtube. *arXiv preprint arXiv:2202.10448*, 2022.

[139] Hyoung Il Son, Antonio Franchi, Lewis L Chuang, Junsuk Kim, Heinrich H Bulthoff, and Paolo Robuffo Giordano. Human-centered design and evaluation of haptic cueing for teleoperation of multiple mobile robots. *IEEE transactions on cybernetics*, 43(2):597–609, 2013.

[140] Simon Stepputtis, Joseph Campbell, Mariano Phielipp, Stefan Lee, Chitta Baral, and

Heni Ben Amor. Language-conditioned imitation learning for robot manipulation tasks. *Advances in Neural Information Processing Systems*, 33:13139–13150, 2020.

[141] Adam Stooke, Kimin Lee, Pieter Abbeel, and Michael Laskin. Decoupling representation learning from reinforcement learning. In *International Conference on Machine Learning*, pages 9870–9879. PMLR, 2021.

[142] Entong Su, Chengzhe Jia, Yuzhe Qin, Wenxuan Zhou, Annabella Macaluso, Binghao Huang, and Xiaolong Wang. Sim2real manipulation on unknown objects with tactile-based reinforcement learning. *arXiv preprint arXiv:2403.12170*, 2024.

[143] Ioan A Sucan, Mark Moll, and Lydia E Kavraki. The open motion planning library. *IEEE Robotics & Automation Magazine*, 19(4):72–82, 2012.

[144] Balakumar Sundaralingam, Siva Hari, Adam Fishman, Caelan Garrett, Karl Van Wyk, Valts Blukis, Alexander Millane, Helen Oleynikova, Ankur Handa, Fabio Ramos, Nathan Ratliff, and Dieter Fox. CuRobo: Parallelized collision-free robot motion generation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, June 2023.

[145] Stone Tao, Xiaochen Li, Tongzhou Mu, Zhiao Huang, Yuzhe Qin, and Hao Su. to-executable trajectory translation for one-shot task generalization. *arXiv preprint arXiv:2210.07658*, 2022.

[146] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017.

[147] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *IROS*, 2012.

[148] Joanne Truong, Sonia Chernova, and Dhruv Batra. Bi-directional domain adaptation for sim2real transfer of embodied navigation agents. *IEEE Robotics and Automation Letters*, 6(2):2634–2641, 2021.

[149] Albert Tung, Josiah Wong, Ajay Mandlekar, Roberto Martín-Martín, Yuke Zhu, Li Fei-Fei, and Silvio Savarese. Learning multi-arm manipulation through collaborative teleoperation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9212–9219. IEEE, 2021.

[150] Jun Wang, Yuzhe Qin, Kaiming Kuang, Yigit Korkmaz, Akhilan Gurumoorthy, Hao Su, and Xiaolong Wang. Cyberdemo: Augmenting simulated human demonstration for real-world dexterous manipulation. *arXiv preprint arXiv:2402.14795*, 2024.

[151] Lirui Wang, Yiyang Ling, Zhecheng Yuan, Mohit Shridhar, Chen Bao, Yuzhe Qin, Bailin Wang, Huazhe Xu, and Xiaolong Wang. Gensim: Generating robotic simulation tasks via large language models. *arXiv preprint arXiv:2310.01361*, 2023.

[152] Luobin Wang, Runlin Guo, Quan Vuong, Yuzhe Qin, Hao Su, and Henrik Christensen. A real2sim2real method for robust object grasping with neural surface reconstruction. In *2023 IEEE 19th International Conference on Automation Science and Engineering (CASE)*, pages 1–8. IEEE, 2023.

[153] Dong Wei, Bidan Huang, and Qiang Li. Multi-view merging for robot teleoperation with virtual reality. *IEEE Robotics and Automation Letters*, 6(4):8537–8544, 2021.

[154] Yijia Weng, He Wang, Qiang Zhou, Yuzhe Qin, Yueqi Duan, Qingnan Fan, Baoquan Chen, Hao Su, and Leonidas J Guibas. Captra: Category-level pose tracking for rigid and articulated objects from point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13209–13218, 2021.

[155] Yueh-Hua Wu, Jiashun Wang, and Xiaolong Wang. Learning generalizable dexterous manipulation from human grasp affordance. *arXiv preprint arXiv:2204.02320*, 2022.

[156] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, et al. Sapien: A simulated part-based interactive environment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11097–11107, 2020.

[157] Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked visual pre-training for motor control. *arXiv preprint arXiv:2203.06173*, 2022.

[158] Pengwei Xie, Rui Chen, Siang Chen, Yuzhe Qin, Fanbo Xiang, Tianyu Sun, Jing Xu, Guijin Wang, and Hao Su. Part-guided 3d rl for sim2real articulated object manipulation. *IEEE Robotics and Automation Letters*, 2023.

[159] Jianglong Ye, Jiashun Wang, Binghao Huang, Yuzhe Qin, and Xiaolong Wang. Learning continuous grasping function with a dexterous hand from human demonstrations. *IEEE Robotics and Automation Letters*, 8(5):2882–2889, 2023.

[160] Lin Yen-Chen, Andy Zeng, Shuran Song, Phillip Isola, and Tsung-Yi Lin. Learning to see before learning to act: Visual pre-training for manipulation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7286–7293. IEEE, 2020.

[161] Zhao-Heng Yin, Binghao Huang, Yuzhe Qin, Qifeng Chen, and Xiaolong Wang. Rotating without seeing: Towards in-hand dexterity through touch. *arXiv preprint arXiv:2303.10880*, 2023.

[162] Tianhe Yu, Ted Xiao, Austin Stone, Jonathan Tompson, Anthony Brohan, Su Wang, Jaspiar Singh, Clayton Tan, Jodilyn Peralta, Brian Ichter, et al. Scaling robot learning with semantically imagined experience. *arXiv preprint arXiv:2302.11550*, 2023.

[163] Ying Yuan, Haichuan Che, Yuzhe Qin, Binghao Huang, Zhao-Heng Yin, Kang-Won Lee, Yi Wu, Soo-Chul Lim, and Xiaolong Wang. Robot synesthesia: In-hand manipulation with visuotactile sensing. *arXiv preprint arXiv:2312.01853*, 2023.

[164] Zhecheng Yuan, Zhengrong Xue, Bo Yuan, Xueqian Wang, Yi Wu, Yang Gao, and Huazhe Xu. Pre-trained image encoder for generalizable visual reinforcement learning. *Advances in Neural Information Processing Systems*, 35:13022–13037, 2022.

[165] Sergey Zakharov, Wadim Kehl, and Slobodan Ilic. Deceptionnet: Network-driven domain randomization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 532–541, 2019.

[166] Kevin Zakka, Andy Zeng, Pete Florence, Jonathan Tompson, Jeannette Bohg, and Debidatta Dwibedi. Xirl: Cross-embodiment inverse reinforcement learning. In *Conference on Robot Learning*, pages 537–546. PMLR, 2022.

[167] Yanjie Ze, Nicklas Hansen, Yinbo Chen, Mohit Jain, and Xiaolong Wang. Visual reinforcement learning with self-supervised 3d representations. *IEEE Robotics and Automation Letters*, 8(5):2890–2897, 2023.

[168] Andy Zeng, Pete Florence, Jonathan Tompson, Stefan Welker, Jonathan Chien, Maria Attarian, Travis Armstrong, Ivan Krasin, Dan Duong, Vikas Sindhwani, et al. Transporter networks: Rearranging the visual world for robotic manipulation. In *Conference on Robot Learning*, pages 726–747. PMLR, 2021.

[169] Fan Zhang, Valentin Bazarevsky, Andrey Vakunov, Andrei Tkachenka, George Sung, Chuo-Ling Chang, and Matthias Grundmann. Mediapipe hands: On-device real-time hand tracking. *arXiv preprint arXiv:2006.10214*, 2020.

[170] Haodong Zhang, Weijie Li, Yuwei Liang, Zexi Chen, Yuxiang Cui, Yue Wang, and Rong Xiong. Human-robot motion retargeting via neural latent optimization. *CoRR*, 2021.

[171] 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 *ICRA*, 2018.

[172] Xiaoshuai Zhang, Rui Chen, Ang Li, Fanbo Xiang, Yuzhe Qin, Jiayuan Gu, Zhan Ling, Minghua Liu, Peiyu Zeng, Songfang Han, et al. Close the optical sensing domain gap by physics-grounded active stereo sensor simulation. *IEEE Transactions on Robotics*, 2023.

[173] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

[174] Wenping Zhao, Jinxiang Chai, and Ying-Qing Xu. Combining marker-based mocap and rgb-d camera for acquiring high-fidelity hand motion data. In *Proceedings of the ACM SIGGRAPH/eurographics symposium on computer animation*, pages 33–42, 2012.

[175] Henry Zhu, Abhishek Gupta, Aravind Rajeswaran, Sergey Levine, and Vikash Kumar. Dexterous manipulation with deep reinforcement learning: Efficient, general, and low-cost. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3651–3657. IEEE, 2019.

[176] Zehao Zhu, Jiashun Wang, Yuzhe Qin, Deqing Sun, Varun Jampani, and Xiaolong Wang. Contactart: Learning 3d interaction priors for category-level articulated object and hand poses estimation. *arXiv preprint arXiv:2305.01618*, 2023.

[177] Zhengbang Zhu, Hanye Zhao, Haoran He, Yichao Zhong, Shenyu Zhang, Yong Yu, and Weinan Zhang. Diffusion models for reinforcement learning: A survey. *arXiv preprint arXiv:2311.01223*, 2023.