

**Project Thesis**  
PRO-080

**Hand-held Data Collection  
for  
Learning of Manipulation Tasks**

**Tragbare Datensammlung für  
das Lernen von Manipulationsaufgaben**

by  
Jannik Jorge Grothusen

Supervisors: Prof. Dr.-Ing. R. Seifried (TUHH)  
Dr.-Ing. D.-A. Dücker (TUM)

Technical University of Munich (TUM)  
Munich Institute of Robotics and Machine Intelligence  
Prof. Dr.-Ing. S. Haddadin

Hamburg University of Technology (TUHH)  
Institute of Mechanics and Ocean Engineering  
Prof. Dr.-Ing. R. Seifried

Munich, July 2024





# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Motivation . . . . .	1
1.2	Problem Statement . . . . .	2
1.3	Thesis Structure . . . . .	3
<b>2</b>	<b>Concepts in Learning for Manipulation</b>	<b>5</b>
2.1	Formalizing Manipulation Learning Tasks . . . . .	5
2.2	Learning from Human Demonstrations . . . . .	8
<b>3</b>	<b>State of the Art</b>	<b>13</b>
3.1	Learning Algorithms and Datasets . . . . .	13
3.2	Data Collection Methods . . . . .	16
3.3	Discussion . . . . .	20
<b>4</b>	<b>Design of a Demonstration Interface</b>	<b>25</b>
4.1	Conceptual Overview . . . . .	26
4.2	Recording of Transferable Observations . . . . .	29
4.3	Human-to-Robot Action Mapping . . . . .	30
4.4	Workflow and Data Processing . . . . .	34

<b>5</b>	<b>Evaluations</b>	<b>39</b>
5.1	Validation Strategy . . . . .	39
5.2	Action Data Accuracy . . . . .	41
5.3	Data Collection Efficiency . . . . .	48
5.4	Discussion . . . . .	49
<b>6</b>	<b>Conclusion</b>	<b>51</b>
6.1	Summary . . . . .	51
6.2	Outlook . . . . .	53
	<b>Bibliography</b>	<b>54</b>
	<b>Appendix</b>	<b>63</b>
A.1	Contents Archive . . . . .	63

# Chapter 1

## Introduction

### 1.1 Motivation

Robot manipulation is central to achieving the promise of robotics. The very definition of a robot requires that it has actuators, which it can use to interact with the world around it. The potential applications for autonomous manipulation are immense: robots capable of manipulating their environment could be deployed in hospitals, elder- and child-care, factories, outer space, restaurants, service industries, and homes.

Given the wide variety of deployment scenarios and unsystematic environmental variations in human spaces, an effective manipulation robot must operate in unpredictable and unfamiliar environments for both the robot and its designers [KroemerNiekumKonidaris21]. Traditional robot control methods involve modeling domain dynamics and deriving mathematically-based algorithms. While theoretically robust, these methods heavily depend on the accuracy of the world model [ArgallEtAl09].

The real world often contains too much variation to formulate an accurate model of the environment, the objects in it, or the skills required to manipulate them in advance. *Learning* offers an effective solution by developing statistical algorithms that can learn from examples and generalize to unseen data, allowing robots to perform tasks without explicit instructions. Researchers have thus focused on how a robot should learn to manipulate the world around it. This research has ranged from learning individual manipulation skills from human demonstrations to learning abstract descriptions of manipulation tasks suitable for high-level planning and discovering an object's functionality through interaction, among other objectives.

The goal of robot learning is to teach a robot to select an action based on its current world state. This mapping between world states and actions is called a *policy* and is fundamental to many robotics applications. A particularly promising approach, which has seen great success in recent years, is *Learning from Demonstration* (LfD). Within LfD, a policy is learned from examples provided by a teacher, usually an expert in the demonstrated task. The performance of learned policies heavily depends on the scale and diversity of these demonstrations [KhazatskyEtAl24]. This suggests that creating large, diverse, high-quality robot manipulation datasets is crucial for developing more capable and robust robotic manipulation policies.

However, creating such datasets is challenging. Unlike vision or language data, training manipulation policies typically requires robot manipulation data with recorded observations and actions, which cannot be easily obtained from the internet. Collecting robot manipulation data in diverse environments involves logistical and safety challenges, especially when deploying robots outside controlled laboratory settings. Additionally, collecting data at scale requires considerable investments in hardware and human labor for supervision, particularly for gathering demonstration data. Consequently, even the most general robot manipulation policies today are mostly trained on data collected in controlled, lab-like environments with limited scene and task diversity.

## 1.2 Problem Statement

A fundamental trade-off in acquiring robotic manipulation data is balancing diversity against transferability. Recent works like [PadalkarEtAl23] and [KhazatskyEtAl24] have shown that human manipulation capability can be a suitable source of demonstration data. Especially, when aiming to scale robotic manipulation skills using machine learning. Although humans would be able demonstrators, the challenge in robotics is that, to date, it is expensive and tedious to collect demonstration data from humans in a robot-compatible format.

Currently, data collection platforms struggle to balance the diversity of collected actions and their transferability to effective robot policies for two reasons:

- 1. Embodiment Gap:** There is a fundamental difference between human actions and robot capabilities, known as the embodiment gap. Methods like shadowing human motions or learning from human videos, although easy to scale, do not translate well to robot actions due to differences in embodiment, resulting in inefficient policy learning.

**2. High Setup Costs and Expertise Requirements:** Data collection methods, such as the widely used teleoperation approach, are resource-intensive and require specialized knowledge to operate. Even though these systems collect robot-native data, their complexity limits the ability to collect diverse demonstrations at scale.

The objective of this thesis is to develop a suitable concept for an intuitive and effective data collection interface that reduces the embodiment gap, captures transferable data, and supports the learning of diverse manipulation skills for robots.

To achieve this, a sensorized hand-held gripper interface is proposed that simultaneously minimizes the embodiment gap while remaining intuitive and flexible. The system eliminates the need for a physical robot during data collection because demonstrations are recorded in the end-effector space. The main contributions of this work include: (1) a novel hardware adaptation equipped with state-of-the-art sensors to enable intuitive high-quality data acquisition in real-time and (2) a comprehensive software suite for easy workflow control and multimodal data processing. The chapter also outlines the system’s hardware setup, workflow, and data processing methods

### 1.3 Thesis Structure

The remainder of this thesis is organized as follows. First, Chap. 2 explains the fundamentals of learning manipulation tasks from human demonstrations. Then, Chap. 3 presents an extensive review on the state of the art of robot learning for manipulation, including a discussion on recent data collection methods. Based on this, in Chap. 4, a concept for a sensorized hand-held gripper interface is developed. The proposed system design enhances state-of-the-art methods in three key aspects: it allows for real-time data acquisition with both visual and depth information, streamlines workflow with user-friendly control interfaces, and boosts accessibility by accommodating various end-effectors. In Chap. 5, the developed system is evaluated in several experiments for its accuracy and efficiency. The results of these experiments are then analyzed and discussed in detail. Finally, the findings of this thesis are summarized and future potential work is discussed in Chap. 6.

In the spirit of fostering further research and collaboration, all hardware and software systems are provided open-source to facilitate evaluation and replication of this work: <https://github.com/J4nn1K/demonstration-interface>.





# Chapter 2

## Concepts in Learning for Manipulation

As introduced in Chap. 1, learning is a highly effective solution for providing robots with autonomous manipulation skills. This approach is particularly valuable when the robot’s environment or tasks exhibit too much variation to allow for the creation of an accurate world model. The process of learning a manipulation skill consists of two steps: (1) acquiring a dataset and (2) deriving a policy from it. The scope of this thesis lies in the first step: acquiring data.

Since policy performance is closely tied to the quality of the dataset it was trained on, it is important to understand the connection between policy and data. This is why this chapter aims to provide the reader with a comprehensive understanding of how the learning of manipulation tasks works as a whole. This chapter will present the fundamentals of learning manipulation tasks (Sec. 2.1) and specifically how this can be done using human demonstrations (Sec. 2.2), which is the method that this thesis builds on.

### 2.1 Formalizing Manipulation Learning Tasks

Robot learning problems can be formulated as Partially-Observable Markov Decision Processes (POMDP) [KaelblingLittmanCassandra98]. A POMDP is a generalization of the individual Markov Decision Processes [Howard60] and models the relationship between an agent and its environment. Formally, a POMDP is a 7-tuple  $(S, A, T, R, \Omega, O, \gamma)$ , where  $S$  is a set of states;  $A$  is a set of actions;  $T(s' | s, a)$  is a transition function, giving the probability distribution over states  $s'$  reached after executing action  $a$  in state  $s$ ;  $R(s, a, s')$  is a reward function, expressing the immediate reward for executing action  $a$  in state  $s$  and

transitioning to state  $s'$ ;  $\Omega$  is a set of observations;  $O(o \mid s', a)$  is a set of conditional observation probabilities, giving the probability of receiving observation  $o$  which depends on the new state  $s'$  and on the just taken action  $a$ ; and  $\gamma \in [0, 1]$  is a discount factor expressing the agent’s preference for immediate over future rewards [KroemerNiekumKonidaris21].

Here, the goal of learning is to find a deterministic control policy,  $\pi : O \rightarrow A$ , that maps observations to actions so as to maximise the *return*, or discounted sum of future rewards  $\sum_{i=0}^{\infty} \gamma^i r_i$ , for that specific problem.

According to [KroemerNiekumKonidaris21], learning problems for manipulation can typically be placed into one of five broad categories:

1. Learning to define the state space
2. Learning a transition model of the environment
3. Learning motor skills
4. Learning to characterize a motor skill
5. Learning compositional and hierarchical structure

This work focuses on learning motor skills. When learning motor skills, the robot attempts to learn a motor control policy that directly achieves some goal. This goal ranges from learning task-specific solution policies to policies that can produce a solution to any task in a task family given a context vector, to useful motor skills that constitute a component of the solution policy but are not themselves a complete solution. In the following, different policy parameters will be discussed. A more in-depth discussion can be found in [KroemerNiekumKonidaris21].

### State and Observation Spaces

Modelling manipulation tasks requires representations of the robot’s environment and the objects that it is manipulating. These representations serve as the basis for learning skill policies. Within-task variations are captured by the *state space* (those features that a manipulation action can change); across-task variations are captured as part of the *context space* (attributes that are fixed in any specific task but aid generalization across pairs of tasks).

As they are capable of manipulating their surroundings, robots can use actions to enhance their perception of the environment. Robot perception is therefore broadly divided into passive and interactive perception, with the key difference being whether or not the robot physically interacts with the environment. The term *passive perception* refers to the process of perceiving the environment without exploiting physical interactions with it. For example, recognizing and localizing objects in a scene based on a camera image. In *interactive perception*, the

robot physically interacts with its surroundings to obtain a better estimate of the environment. For example, a robot may push an object to better estimate its constraints or lift an object to estimate its weight. Robots can use a wide range of sensor modalities to observe the effects of its interactions, including haptic, tactile, vision, and audio.

### Action Spaces

The robot ultimately needs to send a control signal to its actuators to perform actions in the physical world. These signals may, for example, define the torque for an electric motor [LevineEtAl16]. The outputs of policies often do not work directly at the level of actuator signals. This is why, in practice, an additional controller is often placed between the policy and the actuator. Using an additional controller allows the robot to leverage a large body of prior work on control for robotics [SpongHutchinsonVidyasagar20]. Example controllers include simple linear PID controllers as well as more complex model-based admittance and impedance controllers.

Both desired position and force information can be defined in the joint space or a Cartesian task space for the end-effector. For manipulation tasks, it is often easier to generalize interactions with objects across the robot’s workspace using a Cartesian action space [Mason81]. For example, with a Cartesian action space, applying an upward force on a grasped object would be the same action anywhere in the robot’s workspace. The controller is then responsible for mapping these desired signals into the joint space for actuation. For a joint-space action policy, the robot would need to learn different joint torques depending on the arm’s current configuration.

The inclusion of a controller also allows the robot to use a policy that operates at a lower frequency than the controller. While the low-level controllers may operate at 100s or 1000s of Hertz, the policies can operate at lower frequencies. For policies operating at lower frequencies, an additional interpolation step may be used to guide the controller between the desired values.

### Policy Structure

In robotic manipulation, specific parameterizations are often used that restrict the representational power of the policy; if these restrictions respect the underlying structure of the task, generalization and data efficiency are often improved without significantly impacting asymptotic performance. The spectrum of policy structures ranges from highly general (but often sample-inefficient) to highly constrained (but potentially more sample-efficient) representations. An overview on policy structures is given in [KroemerNiekumKonidaris21].

Recent works are predominantly using neural networks. Neural networks can be categorized as *generic fixed-size parametric policy representations*. These representations make stronger assumptions about the complexity and structure of the policy. They are the most flexible and data-driven but they also typically require large amounts of data to produce high-quality generalization. Deep neural networks in particular have become ubiquitous for learning in robotics for two reasons; (1) they can be used to represent state spaces for manipulation tasks with high dimensional observation spaces and (2) they are highly effective at combining data from multiple sensor modalities or information sources. For manipulation tasks, robots often use this approach to merge information between passive modalities (e.g., vision) and more interactive modalities (e.g., touch and haptics), or to incorporate additional task information (e.g., instructions).

## 2.2 Learning from Human Demonstrations

The problem of learning a mapping between world state and actions lies at the heart of many robotics applications. The development of policies by hand is often very challenging and as a result machine learning techniques have been applied to policy development.

In contrast to reinforcement learning, which learns from a robot’s experiences in the world (or a model of it), imitation learning aims to learn about tasks from demonstration trajectories. This can be thought of as a form of programming, but one in which the user simply shows the robot what to do instead of writing code to describe the desired behavior.

The simplest way to use demonstration data to learn a motor skill is to use it as supervised training data to learn the robot’s policy. This is commonly called behavioral cloning [HayesDemiris94, SchaalEtAl05, RossGordonBagnell11]. The demonstration  $d_i \in \mathcal{D}$  provides a set of observation-action pairs  $(o_i, a_i)$ , that can be used as training data for a supervised learning algorithm to learn policy parameters that should, ideally, be able to reproduce the demonstrated behavior in novel scenarios.

The *Learning from Demonstrations* (LfD) problem is constructed after [ArgallEtAl09] as follows. A dataset is composed of example executions of the task by a demonstration teacher (Fig. 2.1, left). The world consists of states  $S$  and actions  $A$ , with the mapping between states and actions being defined by a transition function  $T$ . It is assumed that the state is not fully observable. The learner instead has access to observed state  $O$ , through the mapping  $M : S \rightarrow O$ . A policy  $\pi : O \rightarrow A$  selects actions based on observations of the world state. A single cycle of policy execution at time  $t$  is shown in Fig. 2.1 (right).

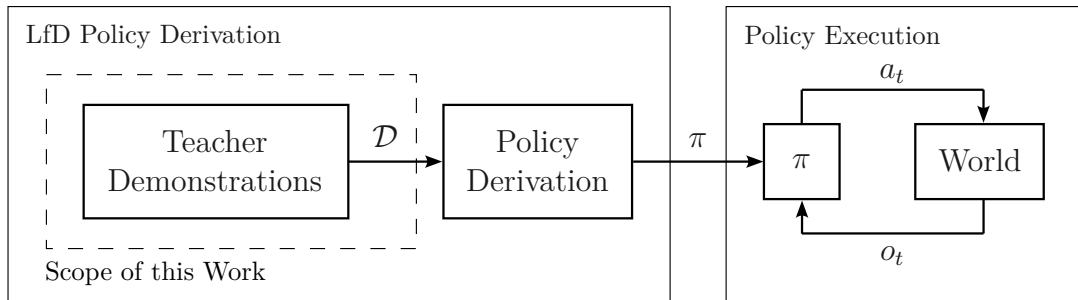


Figure 2.1: Control policy derivation and execution. The teacher’s demonstrations in  $\mathcal{D}$  are used to derive a policy  $\pi$ . During policy execution at timestep  $t$  the policy  $\pi$  is interacting with the world via an action  $a_t$  and receives an observation  $o_t$ .

Learning from demonstration data has been extensively studied in several different settings, because it can enable the robot to leverage the existing task expertise of (potentially non-expert) humans to (1) bypass time-consuming exploration that would be required in a reinforcement learning setting, (2) communicate user preferences for how a task ought to be done, and (3) describe concepts, such as a good tennis swing, that may be difficult to specify formally or programmatically [KroemerNiekumKonidaris21].

The next two sections will formalize the two steps needed for learning a manipulation task from demonstrations; (1) how the dataset is built and (2) how the observation to action mapping is learned.

### 2.2.1 Gathering Demonstrations

There are various techniques for executing and recording demonstrations. The dataset is composed of observation-action pairs recorded during teacher executions of the desired behaviour. Exactly *how* they are recorded, and *what* the teacher uses as a platform for the execution, varies greatly across approaches. Section 3.2 will provide an overview on different approaches.

For LfD to be successful, the states and actions in the learning dataset must be usable by the learner. In the most straightforward setup, the states and actions of the teacher executions map directly to the learner. In reality, however, a direct mapping will often not be possible, as the learner and teacher will likely differ in sensing or mechanics. The issue of *correspondence* deals with the identification of a mapping between the teacher and the learner that allows the transfer of information from one to the other.

In [ArgallEtAl09] correspondence is defined with respect to two mappings, shown in Fig. 2.2: the record mapping  $g_R(o, a)$  and the embodiment mapping  $g_E(o, a)$ .

Furthermore LfD data acquisition approaches can be split into two categories based on the embodiment mapping, and thus the execution platform; (1) *Demonstration*: There is no embodiment mapping  $g_E(o, a) = I(o, a)$ , because demonstration is performed on the actual robot learner (or a physically identical platform) and (2) *Imitation*: there exists an embodiment mapping  $g_E(o, a) \neq I(o, a)$ , because demonstration is performed on a platform which is not the robot learner (or a not physically identical platform).

When teacher executions are *demonstrated* the robot records from its own sensors as its body executes the behavior. There exists no embodiment mapping issue between the teacher and learner but there may exist a non-direct record mapping, however, for state and/or actions, if the states experienced (actions taken) by the demonstrator are not recorded directly, and must instead be inferred from the data. Based on this distinction, two common approaches for providing demonstration data to the robot learner can be identified: (1a) *Teleoperation*: the teacher operates the robot learner platform and the robot’s sensors record the execution; thus  $g_R(o, a) = I(o, a)$  and (1b) *Shadowing*: the robot learner records the execution using its own sensors while attempting to match or mimic the teacher motion as the teacher executes the task; thus  $g_R(o, a) \neq I(o, a)$ .

For *imitation* approaches embodiment issues do exist between the teacher and learner. These approaches can further be divided based on whether the record mapping is the identity or not: (2a) *Sensors on teacher*: sensors located on the executing body are used to record the teacher execution; thus  $g_R(o, a) = I(o, a)$  and (2b) *External observation*: sensors external to the executing body are used to record the execution. These sensors may or may not be located on the robot learner; thus  $g_R(o, a) \neq I(o, a)$ .

## 2.2.2 Deriving a Policy

Given a dataset of state–action examples that have been acquired using one of the methods described in the previous section, a policy can be derived. LfD has seen the development of three core approaches to deriving policies from demonstration data: (1) learning an approximation to the state-action mapping – a *mapping function*, (2) learning a model of the world dynamics and deriving a policy from this information – a *system model*, and (3) a sequence of actions can be produced after learning a model of action pre- and post-conditions – a *plan*. As already mentioned in Sec. 2.1 this work focuses on deriving skill policies. This is why only mapping functions are discussed in further detail. See [ArgallEtAl09] for more details on the other approaches.

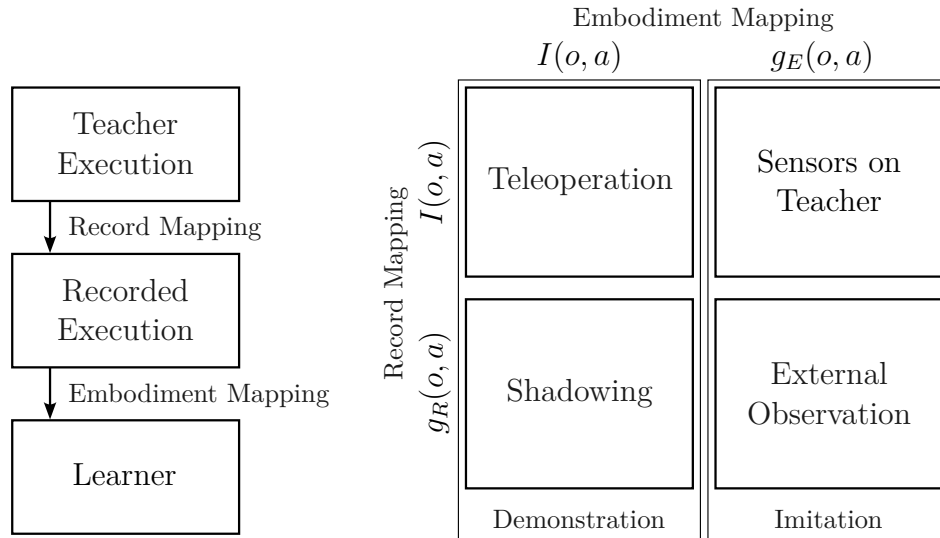


Figure 2.2: Intersection of the record and embodiment mappings (right). The left and right columns represent an identity (*Demonstration*) and non-identity (*Imitation*) embodiment mapping. Each column is then subdivided by an identity (top) or non-identity (bottom) record mapping. Typical approaches to providing data are listed within the quadrants.

The mapping function approach to policy learning calculates a function that approximates the observed state to action mapping,  $f : O \rightarrow A$ , for the demonstrated behavior. The goal of this type of algorithm is to reproduce the underlying teacher policy, which is unknown, and to generalize over the set of available training examples such that valid solutions are also acquired for similar states that may not have been encountered during demonstration.

According to [ArgallEtAl09] mapping approximation techniques fall into two categories depending on whether the prediction output of the algorithm is discrete or continuous. *Classification* techniques produce discrete output, and *regression* techniques produce continuous output. Many techniques for performing classification and regression have been developed outside of LfD and the reader is referred to [HastieEtAl09] for a full discussion.

*Classification* approaches categorize their input into discrete classes, thereby grouping similar input values together. In the context of policy learning, the input to the classifier is robot states and the discrete output classes are robot actions. Classification methods can be applied at three action control levels: basic motion control, action primitives, and complex behaviors.

*Regression* approaches map demonstration states to continuous action spaces. Similar to classification, the input to the regressor are robot states, and the continuous output are robot actions. Since the continuous-valued output often results from combining multiple demonstration set actions, typically regression approaches apply to low-level motions and not high-level behaviors.

In recent years, policy learning from demonstration, in its simplest form, is usually formulated as the *supervised regression task of learning to map observations to actions* [ChiEtAl23]. In practice, the unique nature of predicting robot actions – such as the existence of multi-modal distributions, sequential correlation, and the requirement of high precision – makes this task distinct and challenging compared to other supervised learning problems. The next section will review recent attempts to address this challenge.



# Chapter 3

## State of the Art

To provide the reader with an understanding of how manipulation data is actually collected and used to train models, this chapter will review recent works in the field of policy derivation and data collection. First, to provide an understanding of what is needed to make a manipulation policy work, state-of-the-art representations and dataset are presented in Sec. 3.1. Then, recent data collection methods are presented in Sec. 3.2 and discussed in Sec. 3.3.

### 3.1 Learning Algorithms and Datasets

Recall from Sec. 2.1 that the essence of learning robot behavior is to find a control policy  $\pi : O \rightarrow A$ , that maps observations to actions. It will be now discussed how these observations, actions, and policies look like in practice.

#### Learning Algorithms

As introduced in Sec. 2.1, a policy representation must be chosen for learning manipulation tasks. In recent years, deep neural networks have become the predominant policy representation for learning manipulation policies. This is why the following discussion will only focus on neural networks or *models*. The reader is referred to [KroemerNiekumKonidaris21] for a survey on other policy representations.

Practical applications of policy learning often require hand-engineered components for perception, state estimation, and low-level control. To address this issue works have focused on so called *end-to-end* policy learning. End-to-end policies directly map raw observations like RGB-images to low-level actions like motor torques.

This pixel-to-action formulation is particularly suitable for manipulation, because manipulation often involves objects with complex physical properties, such that learning the manipulation policy is much simpler than modeling the whole environment. The first works that learned manipulation from visual demonstrations were [LevineEtAl16], Behavioral Cloning (BC) [TorabiWarnellStone18], and Visual Imitation through Nearest Neighbors (VINN) [PariEtAl21].

The success of transformer models [VaswaniEtAl17] in natural language processing (NLP) and computer vision motivated a number of recent works that collected demonstration datasets and trained transformer-based policies on them. The major advantages of these models lie in handling multi-modal observations as inputs and scaling with large, diverse datasets. An example of this are Behavior Transformers (BeT) [ShafiullahEtAl22].

Recent advancements have notably elevated the transformer model’s capabilities, namely the prediction of *action chunks* and the *denoising diffusion* training objective. Action Chunking with Transformers (ACT) [ZhaoEtAl23] predicts a sequence of actions with a transformer decoder and is able to perform single fine real-world manipulation tasks with 80-90% success, with only 10 minutes worth of demonstrations. Extending the multi-modal flexibility by integrating the transformer backbone with a denoising diffusion model [HoJainAbbeel20] facilitates long-horizon motion planning. This model type is called Diffusion Policy (DP) [ChiEtAl23] and it is a powerful parametrization for multimodal action output distributions that combine expressivity with scalability. Diffusion Policy is achieving similar success rates to ACT [FuZhaoFinn24, HaFlorenceSong23, GhoshEtAl24]. A very recent method called Vector-Quantized Behavior Transformer (VQ-BeT) [LeeEtAl24] improves on models such as BeT and Diffusion Policies and is therefore setting a new state-of-the-art.

Papers have also focused on broadening the generalization abilities of robot policies. This can mean to perform low-level visuomotor control across tasks, environments, and robotic systems. These models are usually equipped with the ability to understand and follow language instructions [StepputtisEtAl20, NairEtAl22a, MeesEtAl22], often by learning language-conditioned policies. Some of them are incorporating Internet-scale pre-trained Vision-Language-Models (VLMs) to boost generalization and enable emergent semantic reasoning. Recent examples include RT-2-X (VLM-based) [BrohanEtAl23, PadalkarEtAl23], OCTO (DP-based) [GhoshEtAl24], RoboAgent (ACT-based) [BharadhwajEtAl23], and OpenVLA (VLM-based) [KimEtAl24]. These models are often called Vision-Language-Action (VLA) models or generalist policies. They can be as small as 27 M parameters in OCTO and become as large as the 55 B parameter version of RT-2-X.

Since observation and action spaces vary significantly across robots, generalist policies are often using a normalized action space. The most common one being a 7-dimensional action vector controlling the end-effector ( $x, y, z$ , roll, pitch, yaw, and gripper opening or the rates of these quantities) [PadalkarEtAl23]. This way, an output of the model can be interpreted (de-normalized) differently depending on the embodiment used. A common observation space are recent camera images and language instructions.

A notable trend in prior studies is their reliance on large training datasets, often involving hundreds of demonstrations per task, to train robust transformer policies. This is likely due to limited 3D spatial reasoning when just using camera images as observations. A line of work has been focusing on improving 3D spatial reasoning by including depth information like RGB-D images as input to the manipulation policy. These policies are also using more specific perceptual representations. PerAct [ShridharManuelliFox23] for example is using voxelized point clouds and 3D convolutional networks and Act3D [GervetEtAl23] a multi-scale 3D feature cloud. The very recently proposed Robotic View Transformer 2 (RVT-2) [GoyalEtAl24] is using a point cloud reconstruction and multiple virtual views to learn precise manipulation tasks. RVT-2 can learn high-precision tasks with as little as 10 demonstrations and is achieving a new state-of-the-art on RL Bench [JamesEtAl20] improving the success rate from 65% to 82%. A very similar approach is taken by Equivariant Diffusion Policy [YangEtAl24] which uses a scene point cloud and the robot pose as input and can generalize to novel objects and scenes after learning from just 5 minutes of human demonstrations in each task.

### Datasets

A key ingredient for training robot policies is robot training data. In computer vision and NLP, training on large and diverse datasets scraped from the internet yields models that work in a wide range of new tasks. Similarly, in robot manipulation, a number of recent works have demonstrated that larger, more diverse robot training datasets enable us to push the envelope on policy generalization, including positive transfer to new objects, instructions, scenes, and embodiments [BharadhwajEtAl23, PadalkarEtAl23, KhazatskyEtAl24]. But in contrast to vision and language data that can be scraped from the web, obtaining robot data at scale is challenging and often involves significant investments in hardware and human labor.

In recent years, there have been multiple efforts for building robot manipulation datasets of increasing scale and diversity. See Tab. 3.1 for an overview. The datasets are either collected via scripted and autonomous policies (RoboNet [DasariEtAl19], MT-Opt [KalashnikovEtAl21]) or human teleoperation (RT-1 [BrohanEtAl22], RH20T [FangEtAl23]).

Table 3.1: Recent publicly available robotic manipulation dataset show an increase in scene diversity. *Scenes* refers to the number of unique robot work spaces. Dobb·E is the only dataset which was collected with a hand-held tool. It consists of a wide variety of scenes which indicates the ability to collect diverse data using hand-held tools.

Dataset	# Trajectories	# Scenes	Method
RoboNet (2019)	162k	10	scripted
MT-Opt (2021)	800k	1	scripted/learned
RT-1 (2022)	130k	2	human teleop
RH20T (2023)	13k	7	human teleop
RoboSet (2023)	98.5k	11	human/scripted
BridgeData V2 (2023)	60.1k	24	human/scripted
<b>Dobb·E (2023)</b>	<b>5.6k</b>	<b>216</b>	<b>human tool-based</b>
Open X-Embodiment (2023)	1.4M	311	dataset aggregation
DROID (2024)	76k	564	human teleop

RoboSet [BharadhwajEtAl23] and BridgeData V2 [WalkeEtAl23] contain a combination of scripted and teleoperated demonstrations. To this date, there is only one public dataset which was collected using a hand-held device; Dobb·E [ShafiullahEtAl23]. Works like the Open X-Embodiment [PadalkarEtAl23] and DROID [KhazatskyEtAl24] aim to scale data diversity by being reusable across institutions. Nevertheless, it is unclear if this method is able to scale enough to enable similar breakthroughs as seen recently in natural language processing or computer vision.

## 3.2 Data Collection Methods

This section will discuss four data acquisition techniques used in research to learn from human demonstrations; (1) Visual Demonstrations from Human Video, (2) Shadowing and Motion Capture, (3) Teleoperation, and (4) Hand-held Tools. Note that the term *teleoperation* also means *remote control of a robot* and thus is often used in conjunction with all kinds of data acquisition techniques. This work will stick to the categorization introduced in Sec. 2.2, meaning that not all "teleoperation" works will be categorized as teleoperation. Also, data collection methods often cannot be strictly categorized into groups because the definition of record- and embodiment-mappings changes with the choice of observation and action spaces.

Recall from Sec. 2.2 and Fig. 2.1 that teacher executions are separated in *demonstration*, meaning that the demonstration is performed on the actual robot learner and *imitation*, meaning that the demonstration is performed on a platform which is not the robot learner (or a not physically identical platform).

### Visual Demonstrations from Human Video

Utilizing in-the-wild video data (e.g. YouTube videos) can be an attractive way to acquire very large amounts of demonstration data. The most common way is to learn from diverse passive human demonstration videos. To infer action data from passive human video, previous works use hand pose detectors [BahlGuptaPathak22, WangEtAl23, ShawBahlPathak23], or combined human videos with in-domain teleoperated robot data [SchmeckpeperEtAl20, QinEtAl22]. To bridge the embodiment gap and transfer actions between humans and robots recent works used hand pose retargeting [QinEtAl22] and extraction of embodiment-agnostic keypoints [XiongEtAl21]

At this point it should be noted that even if inferring actions from in-the-wild video data is an ongoing effort, Internet-scale video data is successfully used for pre-training of visual representations in robot learning [NairEtAl22b, RadosavovicEtAl23]. Most model architectures including RT-1 [BrohanEtAl22], Diffusion Policy [ChiEtAl23], and ACT [ZhaoEtAl23] are applying pre-trained Convolutional Neural Networks (CNNs) to input observation data [LeCunBengioothers95, LeCunBengioHinton15, PerezEtAl18].

### Shadowing and Motion Capture

Recall from 2.2 that *shadowing* means that the demonstration is performed on the actual robot learner that mimics the teacher’s demonstrated motions while recording from its own sensors. This method is particularly useful when training robots with similar form factors to human beings. The human-like morphology of humanoids presents a unique opportunity to leverage the vast amounts of human motion and skill data available for training, bypassing the scarcity of robot data.

Recent works in this field include OmniH2O [HeEtAl24], HumanPlus [FuEtAl24], and Open-TeleVision [ChengEtAl24]. See Fig. 3.1 for a images. To bridge the physical gaps between humanoids and humans in morphologies and actuation these works are translating motion from human to robot using motion retargeting libraries like *dex-retargeting* [QinEtAl23] or by training low-level policies for following human motions [FuEtAl24].

Besides for humanoid form factors, recent works have also focused on vision-based shadowing for control of single robot hands and hand-arm systems. The main works in this field are DexPilot [HandaEtAl20], Holo-Dex [ArunachalamEtAl23a], DIME [ArunachalamEtAl23b], and AnyTeleop [QinEtAl23]. All of these works are focusing on capturing the dexterous abilities of human hands.



Figure 3.1: Recent data collection methods utilizing shadowing. In all three works humanoid robots are mimicing human actions using vision-based motion retargeting.

### Teleoperation

Recall from 2.2 that *teleoperation* means that the teacher operates the robot learner platform (or a physically identical platform) and the robot’s sensors record the execution. The simplest form of teleoperation is kinesthetic teaching [KormushevCalinonCaldwell11, SteinmetzMontebelliKyrki15]; a user demonstrates a new skill to a robot by manually guiding the robot’s arm through the motion.

A more common approach for teleoperation is the utilization of interfaces such as a 3D spacemouse [3Dconnexion24], VR or AR controllers [Meta24], and haptic devices [Force Dimension24] which abstract away the kinematic constraints on the robot. Their control (record) mapping usually lies in the task space (i.e. the end-effector pose) and is therefore the identity mapping. The RH20T dataset [FangEtAl23] was collected using a Force Dimension sigma.7 haptic device. This device provides force feedback to the operator but greatly constraints the workspace (see Fig. 3.2). Datasets like BridgeData V2 [WalkeEtAl23] and DROID [KhazatskyEtAl24] were collected using Meta’s Quest 2 VR controllers. The robot platforms in these works are mostly equipped with parallel jaw grippers. The HATO platform [LinEtAl24] uses Quest 2 to control multi-fingered hand hardware with touch sensing. See Fig. 3.2 for pictures of these systems.

Another recent and promising approach are leader-follower (i.e. *puppeteering*) systems. Works in this category include ALOHA (unilateral control) [ZhaoEtAl23, AldacoEtAl24] and GELLO (bilateral control) [WuEtAl23]. They are using scaled kinematically equivalent structures, allowing the user to interact with the controller as if they were directly controlling the target arm, as in kinesthetic teaching. ALOHA and GELLO are pictured in Fig. 3.2.

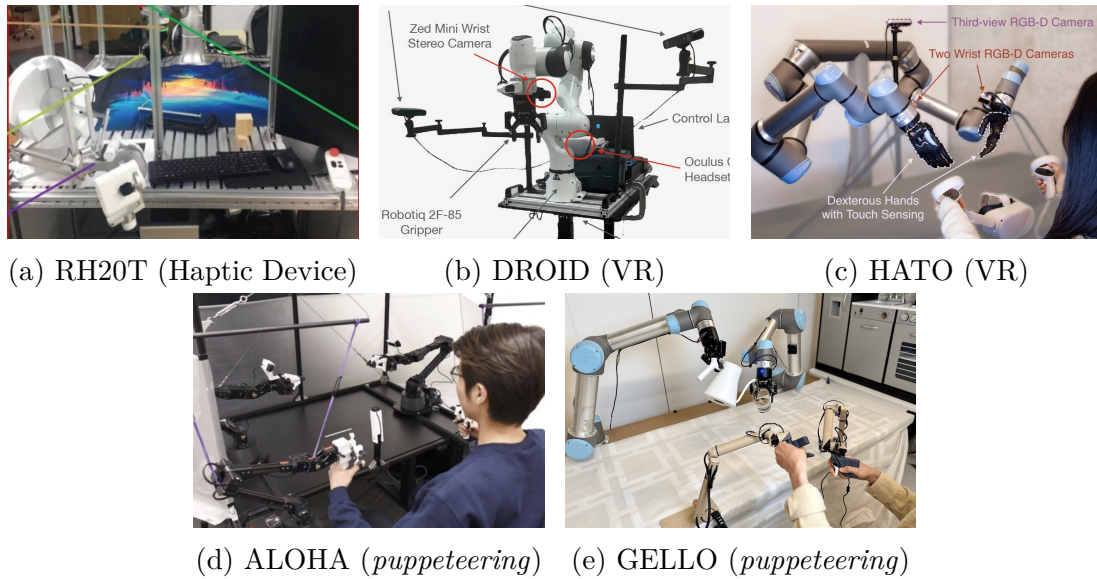


Figure 3.2: Data collection platforms utilizing teleoperation.

### Hand-Held Tools

Recently, using hand-held grippers as a data collection interface has emerged as a promising middle-ground alternative between in-the-wild human videos and teleoperation. These devices are constructed after robotic end-effectors and try to minimize the embodiment gap between data collection and robot inference while remaining intuitive and flexible to use (see Fig. 3.3). Depending on the observation-action-space recorded by the hand-held tool, these devices can loosely be categorized as teleoperation because their data space is identical to the corresponding robot platform. Hand-held data collection is currently limited to the commonly used end-effector pose and gripper state as action space because it is not built on a kinematic robot structure. Nevertheless, when using this action space there is no morphological gap that needs to be bridged by a mapping between human and robot actions.

Hand-held data collection devices are very intuitive and portable by nature. This makes them very suitable for collecting diverse demonstration datasets in different environments. First works in this field were *grabber tool* (see Fig. 3.3 proposed by [SongEtAl20]) and [YoungEtAl21]. These devices struggled with insufficient visual context and action imprecision. Recently, Dobb·E [ShafiullahEtAl23] proposed a reacher-grabber tool mounted with an iPhone to collect single-arm demonstrations for the Stretch robot platform [KempEtAl22]. Dobb·E demonstrates policy deployment for quasi-static tasks and requires environment-specific policy fine-tuning.

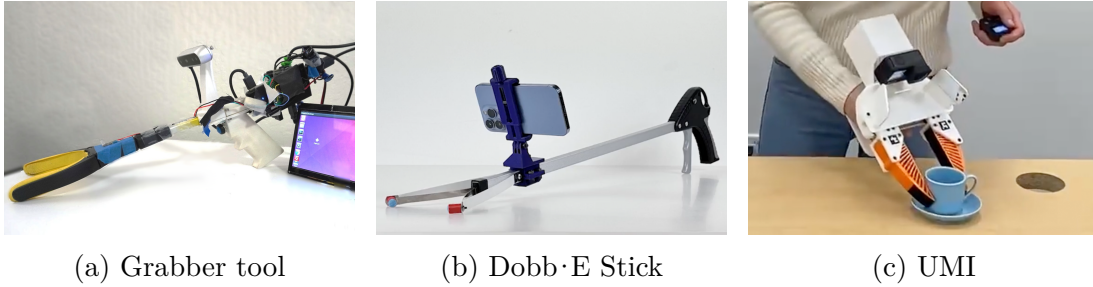


Figure 3.3: Hand-held data collection devices.

Dobb·E’s limitations are addressed by the Universal Manipulation Interface (UMI) [ChiEtAl24]; a hand-held data collection and policy learning framework that allows direct transfer from in-the-wild human demonstrations to deployable robot policies. UMI’s data collection hardware takes the form of a trigger activated, handheld 3D printed parallel jaw gripper with soft fingers, mounted with a GoPro camera as the only sensor and recording device (see Fig. 3.3). UMI is constructed after the WSG-50-110 gripper by Weiss Robotics [Weiss Robotics24]. For bimanual manipulation, UMI can be extended with another gripper.

### 3.3 Discussion

As pointed out in Sec. 3.1 Transformer-based model architectures have become very capable policy representations for learning of manipulation tasks. When trained on high-quality demonstrations they are capable of performing precise manipulation and are able to generalizing to unseen tasks, environments, and embodiments. These advancements are mainly based on recent efforts to collect high-quality and diverse demonstration datasets. Section 3.2 presented different large-scale robotic datasets and elaborated on various techniques for collecting robot data. Based on these findings, the following section will discuss what the most suitable method for scaling up data collection is and what downsides this method currently has.



Before discussing the data collection methods in detail, a few general observations on robotic data collection and policy learning from recent works will be noted:

- **Diversity in Tasks and Environments:** Diversity is the most crucial characteristic for demonstration datasets. Multiple works have shown that data diversity boosts generalization ability and thus policy performance [PadalkarEtAl23, KhazatskyEtAl24].
- **Action Space:** While using joint-space action representations can be beneficial for a very limited set of tasks, employing end-effector pose actions has been found sufficient for learning effective manipulation policies [PadalkarEtAl23, ChiEtAl24].
- **Camera Position:** Research has been conducted on various camera positions and wrist-mounted camera data has proven to be sufficient, as demonstrated by [ChiEtAl24].
- **Depth Data:** As proven by [GoyalEtAl24] and [YangEtAl24] utilizing additional information beyond just RGB images, such as depth data, significantly improves policy effectiveness
- **End-effectors:** Although parallel-jaw grippers sometimes limit the range of achievable motions compared to multifingered hands [BicchiKumar00], they have been shown to be more than suitable even for complex tasks, as evidenced by [ZhaoEtAl23].

### 3.3.1 Discussion of Data Collection Approaches

**Visual Demonstrations from Human Video.** While collecting videos of humans performing manipulation tasks directly is relatively inexpensive and easy to scale, overcoming the morphology gap between robots and humans remains challenging. Learning from video demonstrations has three major downsides; (1) video demonstrations lack explicit action information which is crucial for learning generalizable policies, (2) the evident embodiment gap between humans and robots hinders action transfer from videos to robots and (3) the observation gap induced by the embodiment gap introduces inevitable mismatch between train and inference time observation data, worsen the transferability of the resulting policies.

**Shadowing and Motion Capture.** Mimicking human motions can be useful when working with high degree of freedom action spaces (like in multi-fingered hands or humanoid form factors) because this level of complexity would otherwise be even harder to transfer from demonstrations to robot actions. Even though

shadowing might be a suitable for such action spaces the main downside is similar to an issue of learning from human video; explicit action information is hard to derive in highly complex action spaces. Solving this problem often introduces high computational cost and noticeable latency which reduces intuitiveness for the operator.

**Teleoperation.** Collecting demonstrations by teleoperation is the most common method to acquire robot data right now. VR controllers are very easy to use and haptic devices even enable force feedback. But because of the morphological differences between these control devices and the robots these systems can be unintuitive to new users. Additionally, haptic devices are usually very expensive and greatly constrain the workspace. Leader-follower platforms provide an intuitive answer to these problems. Because of their kinematically equivalent structure they are easy to use and enable direct action-transferability. While being the dominant method for robotic data collection right now, teleoperation systems restrict our ability to efficiently collect large-scale diverse data in the wild. Their reliance on real robots during data collection limits the type and number of environments the system can gain access to. Furthermore, these systems are often build around a specific robot platform which limits the ability for embodiment diversity.

**Hand-held Tools.** Most data collection setups struggle with in-the-wild capabilities and thus are limited in their ability to intuitively collect diverse manipulation data. Hand-held devices emerged as a promising alternative. They eliminate the need for physical robots during data collection and offer a more portable interface for in-the-wild robot teaching. Hand-held grippers collect data in the end-effector task space that is directly transferable to different robot embodiments (e.g. 6DoF or 7DoF robot arms). They are simultaneously minimizing the embodiment gap while remaining intuitive and flexible. Nethertheless hand-held data collection comes with two major downsides; (1) kinematic limits of the downstream deployment robots are unknown at the time of data collection (valid but hardware-infeasible actions can be recorded) and (2) action recovery imprecision (most devices rely on monocular structure-from-motion (SfM) which often struggles to recover precise global action due to motion blur or insufficient texture).

### 3.3.2 Conclusion

Due to their transferable action space and intuitive and portable nature, hand-held tools seem to be the most promising approach for collecting large amounts of diverse task demonstrations. Besides the above mentioned problems of hand-held systems, three issues in previous works (especially UMI [ChiEtAl24]) are identified, which limit the device’s ability to be a suitable demonstration interface:

- **Limited Sensing and Feedback Capabilities:** While state-of-the-art teleoperation systems employ sophisticated sensors to enable real-time feedback during data collection and the ability to record depth data [KhazatskyEtAl24], the hand-held system UMI uses a GoPro as its only sensor. In this case data processing and filtering can only be done post data collection.
- **Complicated Workflows:** To collect data with UMI a four-step process needs to be done prior to data collection; (1) GoPro preparation, (2) time-code synchronization, (3) Recording of a mapping video, and (4) recording of a gripper calibration video. This is already a short workflow but it still consist of more steps than just pressing a button to record a session. Additionally, the usage of a GoPro forces the operator to use the GoPro’s control interfaces. This can become infeasible when collecting bimanual data since the operator has no free finger to press the GoPro’s record button.
- **End-effector Specificity:** UMI is constructed after a specific gripper, namely WSG-50-110 from Weiss Robotics [Weiss Robotics24]. This limits accessibility and transferability to new gripper types.



# Chapter 4

## Design of a Demonstration Interface

In this chapter, a concept for a hand-held data collection framework for demonstration of manipulation tasks is presented. The system aims to address the issues pointed out in Sec. 3.3, namely limited sensing capabilities, complicated operator workflows, and no adaptability to other end-effectors. For this, we start by setting out the features, we believe are necessary for an improved hand-held manipulation demonstration system:

- **Intuitive:** The system is intuitive for a human operator to use without training. The system can be taken into any environment where a human is able to operate. The human demonstrator is able to perform dexterous manipulation tasks with it.
- **Capable:** The ability to capture and transfer natural and complex human manipulation skills to a high-quality robot-native data format. The system should be forward compatible to real-time data processing.
- **Adaptable:** Researchers and enthusiasts should be able to reproduce the system and use data to train their own robots, even with different sensors and end-effectors.

Section 4.1 gives a high-level conceptual overview of the proposed system. Section 4.2 and 4.3 go into more detail on how the observation space and action recovery system were designed. Section 4.4 presents the data processing pipeline and software suite.

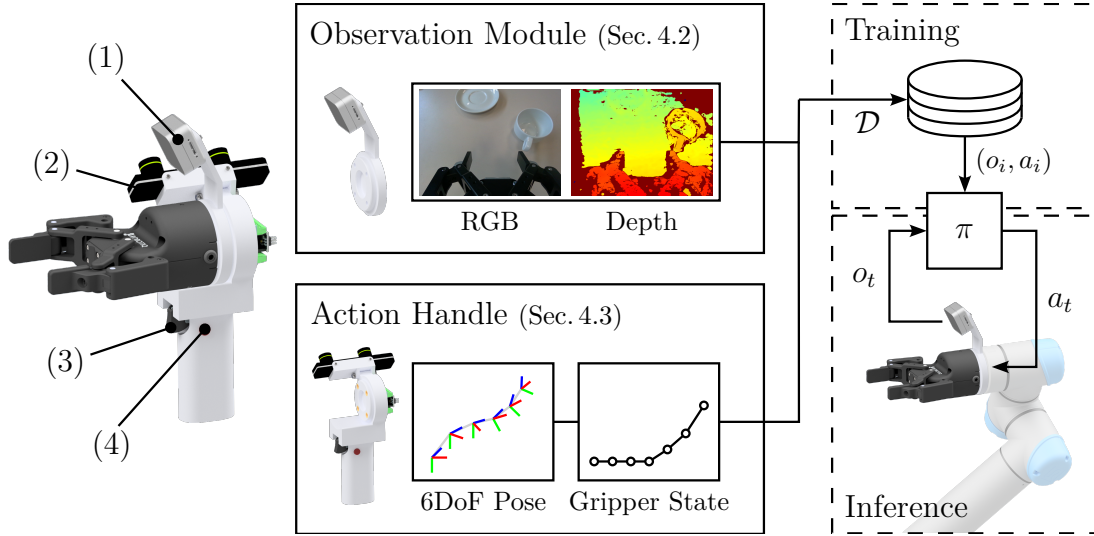


Figure 4.1: Conceptual overview of the data collection framework that consists of an end-effector, the Observation Module and the Action Handle. It is equipped with four sensors: (1) a RGB-D camera, (2) a tracking camera with IMU, (3) a trigger for gripper control, and (4) a record button. The recorded observations and actions can be used as a dataset  $\mathcal{D}$  for training a policy  $\pi$  (not scope of this work).

## 4.1 Conceptual Overview

Based on the features derived above this section will present a high-level conceptual overview of the proposed hand-held data collection framework. Recall from Sec. 2.2 that the goal of a data collection system is to provide the learner with demonstrations  $d_i \in \mathcal{D}$  consisting of observation-action pairs  $(o_i, a_i)$ . During test time the learner tries to predict actions based on given observations.

The proposed data collection device takes the form of a 3D-printed handle that is equipped with four sensors; (1) a camera to record the device’s observation space, (2) a tracking camera to recover motion data, (3) a trigger for gripper control, and (4) a push button to start and stop recordings (see Fig. 4.1, left). Because of its modular mounting method, the device can be trivially adapted for different end-effectors. The following paragraphs will go into more detail on the different features of the framework.

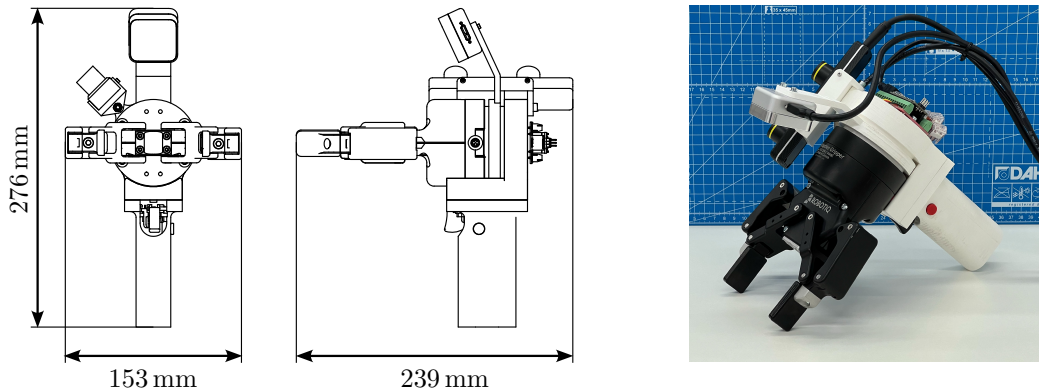


Figure 4.2: Left: schematic front and side view of the device, annotated with external dimensions. Right: a photograph of the real prototype.

**Observation Module.** This component records the observation space (i.e. RGB-D camera images) during data collection and test time. It is equipped with a RGB-D camera and mounted on the data collection device in the same way it will be mounted on the robot during policy inference. This way it is assured that the model receives the same observations during training and test time. Section 4.2 will go into more detail on the design of this component.

**Action Handle.** This component takes care of recovering robot-native actions from human motion. Using visual-inertial simultaneous localization and mapping (SLAM) [Durrant-WhyteBailey06] a 6 degree of freedom (DoF) pose is estimated and recorded. Additionally the Action Handle is equipped with a trigger for parallel-jaw gripper control in continuous space and a push button to easily start and stop recordings. Section 4.3 will go into further detail on how the actions are recovered.

**Mounting.** To account for different end-effector types the mounting method of the system is constructed after ISO 9409-1 [ISO04]. This norm defines a circular plate as mechanical interface for exchangeable, hand-mounted end-effectors. This way it can be ensured that the Observation Module is mountable on most industrial robots and that many standard end-effectors can be attached to the device.

**End-effector.** One of the shortcomings of previous hand-held systems was identified to be the need for a specific end-effector (see Sec. 3.3). The proposed mounting method enables the developed system to be usable with different end-effectors. As a first step, this work utilizes parallel-jaw grippers because of their simplicity and durability.

Component	Weight	Cost
ZED Mini Stereo Camera	63 g	630 €
RealSense D405 Depth Camera	58 g	300 €
Electronics (Arduino, Wires etc.)	25 g	30 €
3D-printed Parts	158 g	20 €
Screws and Inserts	16 g	10 €
Gripper (Robotiq 2F-85)	900 g	5300 €
	<b>1220 g</b>	<b>6290 €</b>

Table 4.1: A list of used components including their weight and price. Off-board components like cables and the external computer are not listed here because they are interchangeable and do not significantly contribute to the device’s weight.

**Data Infrastructure.** To facilitate a light-weight software suite the whole codebase is written in Python (with the exception of C-like code running on the Arduino for sensor-readouts). The code does not depend on robotics middleware like the Robot Operating System (ROS). Data processing and workflow control are implemented using a node-like software structure which is based on Python’s `multiprocessing` package. Multi-processing is chosen because of its simple and direct ability to implement concurrency and messaging. Section 4.4 will go into more detail on the software implementation.

**Assembly.** The device weighs 1220 g, with an external dimension of  $239\text{ mm} \times 153\text{ mm} \times 276\text{ mm}$  (L×W×H). The cost of the device without a gripper and external computer is around 990 €. When factoring in a gripper like the Robotiq 2F-85 and a computer, for example Nvidia’s Jetson Orin Nano Development Kit, the end-price is approximately 7000 €. This is significantly cheaper than setups with robots like DROID [KhazatskyEtAl24] and ALOHA [ZhaoEtAl23], which cost around \$20000 but more expensive than other hand-held setups like UMI [ChiEtAl24], which is reported to cost \$298. Table 4.1 provides a detailed list of used components. Figure 4.2 shows the dimensions and a real picture of the device. CAD-files of the 3D-printed parts and the codebase can be found here:

<https://github.com/J4nn1K/demonstration-interface>.





Figure 4.3: Left: observation camera field of view (FoV). Right: example RGB and depth image observations.

## 4.2 Recording of Transferable Observations

The Observation Module is responsible for providing observations to a policy during training and test time. The proposed device minimizes the observation embodiment gap by using the Observation Module for human demonstrations and robot deployment. This is possible due to the wrist-mount camera position. More on this will be discussed below. This section will highlight the design considerations of the two main features of the Observation Module; (1) the camera positioning, and (2) the inclusion of depth information, in addition to common RGB-image observations.

### Wrist-mounted Camera Position

One of the challenges of robotic data collection as pointed out in Chap. 1 is an embodiment gap. As already pointed out above, the proposed device is minimizing the observation gap by using the same Observation Module for data collection and test time. This way, the observations recorded by the wrist-mount camera are indistinguishable between data collection and deployment. On the right of Fig. 4.1 the mounting during deployment is pictured.

Besides reducing the embodiment gap, using a wrist-mounted camera has additional benefits; (1) mechanical robustness and easy deployment: because the camera is fixed relative to the fingers, mounting the observation module and gripper on robots does not require camera-robot-world calibration and (2) portability: without the need for an external static camera, as they are commonly used in state-of-the-art systems like DROID [KhazatskyEtAl24], the system is simplified and becomes highly portable. Recent works like [ChiEtAl24] have even observed that during training with a moving camera, the policy learns to focus on task-relevant objects or regions instead of background structures. As a result of this, the final policy naturally becomes more robust against distractors at inference time.

### RGB and Depth Channels

As seen in Sec. 3.1 depth perception can significantly boost policy efficiency. While state-of-the-art teleoperation platforms like ALOHA 2 [AldacoEtAl24] and DROID [KhazatskyEtAl24] are utilizing wrist-mounted depth cameras to capture depth information, hand-held devices currently only employ RGB-cameras. The hand-held system UMI [ChiEtAl24] tries to integrate depth perception by using side mirrors for implicit stereo. The device proposed in this thesis will directly employ a depth camera to capture direct and high-quality depth information.

### Camera Selection

A wide range of cameras are used in robot learning platforms for wrist-mounted observations. Commonly used cameras include but are not limited to; simple Web-cameras like Logitech’s C922x Pro, which is used by ALOHA [ZhaoEtAl23], Intel’s RealSense D405, which is used by ALOHA 2 [AldacoEtAl24], and ZED Mini from StereoLabs, which is employed in DROID [KhazatskyEtAl24]. UMI [ChiEtAl24] is build on observations from a GoPro.

Initial experiments using the ZED Mini from StereoLabs as the observation camera showed insufficient depth channel quality on short ranges ( $< 30$  cm) even though StereoLabs is reporting an effective depth range down to 10 cm [StereoLabs24]. In comparison, depth data from Intel’s RealSense D405 was of seemingly sufficient quality. An explanation for this is can be that the D405 has a way shorter baseline (distance between the two lenses), which can lead to a higher accuracy when calculating depth from disparity in short ranges. Since the D405 captures sufficient image and depth data, it was chosen to be the observation camera in this thesis.

## 4.3 Human-to-Robot Action Mapping

The Action Handle maps human actions into a robot-native format. It records a 7-dimensional action space that consists of continuous 6D Cartesian end-effector motion as well as a discrete dimension to control the opening and closing of the gripper. The handle is constructed for use with a right hand but can trivially be adapted for left hands and also bi-manual use. The index finger controls the trigger, which’s state is then mapped to the gripper’s opening and closing motions. The thumb is used to push the record button.

This section will explain the design of the four main features of the Action Handle; (1) the modular mounting interface, (2) the action recovery using SLAM, (3) the trigger for gripper control and (4) the record button.

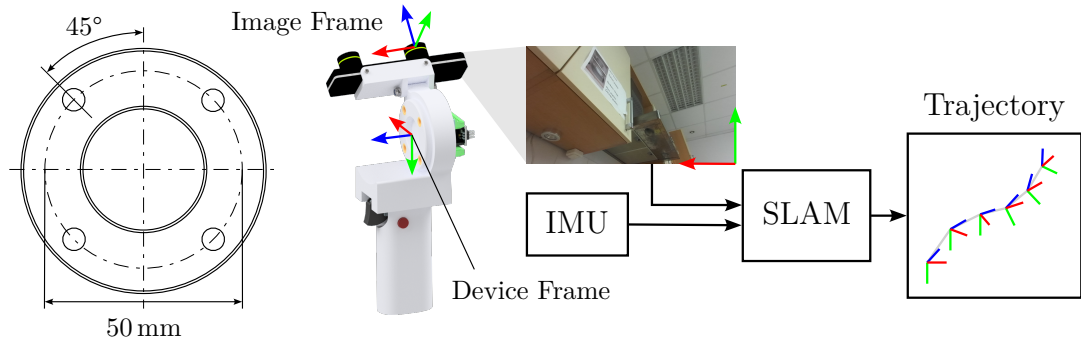


Figure 4.4: Features of the Action Handle. Left: ISO 9401-1-50-4-M6 mounting interface for end-effectors. Right: visual-inertial SLAM recovers pose data from human motion using camera images of the environment and IMU data.

### Modular Mounting

To create a modular and truly robot-native data collection method, the main mechanical mounting interface of the proposed system is constructed after a typical robot flange. Similar to reusing the Observation Module for data collection and robot deployment, this design choice facilitates a minimized embodiment gap between the hand-held handle and the robot during deployment. The mounting is the same in both scenarios.

Most industrial manipulators that are used in research like Franka Robotics' Panda and Universal Robots' UR-series are utilizing the same mechanical interface for end-effectors. This interface is described by ISO 9409-1:2004 [ISO04] and is used as mounting method on the discussed Action Handle. The norm ISO 9409-1 defines the main dimensions, designation and marking for a circular plate as mechanical interface. It is intended to ensure the exchangeability and to keep the orientation of hand-mounted end-effectors.

The dimensions of the mechanical interface can be described by a specific designation code; ISO 9401-1 -  $d_1$  -  $N$  -  $d_4$ . In this designation code  $d_1$  stands for the pitch circle diameter,  $N$  for the number of thread holes, and  $d_4$  for the thread of the hole. Commonly used robots like Panda and the UR-series are build on the designation code ISO 9401-1-50-4-M6. This specific version of the ISO 9401-1 is pictured on the left in Fig. 4.4.

### Pose Estimation using SLAM

There are different options to estimate a pose in 3D space. Localization can be done on-board a system, typically by utilizing cameras and IMU-data (accelerometer and gyroscope) or off-board using external cameras which are usually static. External localization can be very accurate, with sub-millimeter errors at a very

high temporal resolution of hundreds of Hertz. An example for such a system are the Vicon’s Vero motion capture cameras [Vicon24]. These high-speed cameras are placed around a capture area to record the positions of reflective markers attached to a subject. The cameras emit infrared light (invisible to the naked eye), which is reflected back by the markers and can be captured to record the position of the markers. The pose of VR controllers in systems like Meta’s Quest 2 are recorded in a similar fashion: the Quest 2 headset is equipped with multiple outward-facing cameras that capture the environment and infrared LEDs on the controllers [Meta24].

Off-board localization has three major downsides for the discussed application; (1) it depends on external sensors and usually requires markers (often infrared-based), (2) these systems can be very expensive and hard to set up, and (3) they are limited to workspaces in which they are placed - moving them can require significant effort. To facilitate portability and intuitiveness for the proposed device a on-board localization method is chosen; visual-inertial SLAM.

Visual simultaneous localization and mapping is a method that uses visual data from cameras to estimate pose information. The process begins with capturing (depth) images from a camera. Features (distinctive points or areas) in the images are detected using algorithms like ORB [CamposEtAl21]. These features are described in a way that allows for matching between frames and to establish correspondences. Using the correspondences, the system estimates the camera’s movement between frames. A more in-detail explanation on visual SLAM can be found in [TaketomiUchiyamaIkeda17]. Data from inertial measurement units (IMUs) can be used to refine the SLAM-algorithm’s pose estimation.

During first experiments that tried to estimate pose data from the Observation Module’s front-facing camera (in that case, a ZED Mini), resulted in un-usable tracking data. This is likely due to insufficient structure in recorded images. These test demonstrations were recorded in the common tabletop manipulation setting (i.e. looking down at a table). Due to a lack of visual features in these observations, the SLAM-algorithm was likely unable to detect correspondences and therefore unable to calculate accurate pose data.

Previous works like UMI [ChiEtAl24] addressed this issue by employing a fisheye lens, that, due to its wide field-of-view (FoV) of  $155^\circ$ , still records visual features of the environment, even when facing a plain surface. This thesis addresses this by using a dedicated tracking camera (see Fig. 4.4) which faces away from the operator and the workspace to record sufficient visual features of the environments. The capabilities of this method will be analysed in Sec. 5.2.

The tracking pipeline is implemented using StereoLab’s ZED Mini stereo camera [StereoLabs24] and the ZED-SDK’s positional tracking API. The camera is equipped with an inertial sensor capable of providing accelerations and angular

velocities in motion. StereoLabs' positional tracking API is closed-source and therefore no details on the implementation of the visual-inertial SLAM algorithm can be presented here. During data collection, the estimated pose is transformed from the tracking camera's image frame to a more robot-native frame at the center of the mounting plate (see Fig. 4.4).

### Intuitive Gripper Control

To enable interaction with the real world besides movements, the end-effector needs to be controlled in a deterministic way. Since the discussed system should be usable with multiple end-effectors (different parallel-jaw grippers), a general mapping from human input to opening and closing of the gripper needs to be defined. In this thesis, a discrete range  $[0, 100] \in \mathbb{R}$  is used as an intermediate representation to map the operator's input to the gripper state. The range corresponds to: 0 - *fully opened* and 100 - *fully closed*. With this representation we can ensure a standardized gripper interface, independent of the gripper's finger stroke.

The operator's input is acquired using a standard gamepad trigger. It is designed to provide analog input, allowing for variable control rather than a simple binary on/off state. This is achieved through the use of a rotary potentiometer, which translates the physical displacement of the trigger into an electrical signal. This analog electrical signal is digitalized using a Arduino Nano and sent to a connected computer via serial communication. Section 4.4 will go into more detail on how the signal is processed.

### Workflow Control

A shortcoming of UMI [ChiEtAl24] has been identified in its workflow control. The recording of a demonstration must be initiated using the attached GoPro's button or a connected GoPro Remote. As recordings need to be started and stopped for every demonstration, this interaction method can become tedious, especially during long data collection sessions with many demonstrations. Furthermore, bimanual use of UMI-grippers becomes infeasible without a second operator. This is because, when holding the handles of the devices, the operator cannot reach the record buttons. To address this issue, this thesis proposes a button placed inside the handle near the operator's thumb. This design allows recordings to be easily started with the same hand that is holding the device, enabling bimanual use with only a single operator.

Similar to the trigger, the button is connected to the Arduino Nano on the back of the device. A binary on/off signal is recorded and sent to a connected computer via serial communication. Section 4.4 will provide more details on how these signals are processed.



(a) Typical tabletop demonstration setup. (b) Demonstration of a manipulation task.

Figure 4.5: The proposed system can be easily set up using two power supplies, four USB-cables, and a second computer (left) for workflow control.

## 4.4 Workflow and Data Processing

This section will describe the intended device usage and how the signal and data processing backend looks like.

### Setup and Workflow

The hardware setup is straight forward; when the Action Handle and Observation Module are already assembled, an end-effector can be mounted to the device using four M6-screws. Then, four USB-cables are used to connect the gripper, the two cameras, and the micro-controller, to a computer respectively. The computer needs to run Linux as its operating system and have a CUDA-enabled Nvidia GPU. In this work, the Nvidia Jetson Orin Nano Development Kit is used (see Fig. 4.5a, at the top right of the table). It is small and therefore very portable, but still capable of performing all computations needed for the usage of the device, namely signal processing, the SLAM algorithm, and recording of data streams. For energy, two power supplies are needed; one for the computer and one for the gripper.

For workflow control, it is easier to connect a second computer (e.g. a laptop, see Fig. 4.5) to the system using ethernet. This computer can communicate to the system via a SSH-tunnel and will be used to control the workflow and to easily download collected demonstration data. Because the second computer is only connected via ethernet, it can be placed anywhere in the network.

To start a data collection *session*, only a single Python-script, `record_session.py`, needs to be executed on the computer. The script takes care of launching all processes necessary for data collection, namely: sensor interfaces, hardware calibration, control loops, and recording pipelines.

After a few seconds, in which the gripper will automatically open and close for calibration and in which the data streams from all sensors are verified, the device is ready for recording.

An *episode* recording can be started by the operator using a push on the record button. This will start a recording loop at a user-specified frequency. All sensor readings for observations and actions are synchronized to a specific timestamp and saved in memory. When the operator is done with one demonstration, a second press on the record button will stop the episode. The recorded data will now be saved from memory to disk in a HDF5 format. For 20 seconds of demonstration time, saving usually takes less than one second. Details on the data format will be discussed below.

### Data Processing

As already briefly introduced in Sec. 4.1, the software suite, which is the processing backend of the device, is designed in a lightweight fashion to support efficiency and to require minimal dependencies. Four major component classes are programmed in Python to interface with the hardware: `Handle`, `Tracker`, `Camera`, and `Gripper`. `Handle` provides an abstract interface to the micro processor, which sends trigger and button values via serial communication. `Tracker` provides an interface to the tracking camera and takes care of coordinate transformations. `Camera` is directly forwarding image data from the observation camera. Finally, `Gripper` is interfacing with the end-effector using a low-level Modbus RTU protocol.

The component classes are used in `record_session.py` to interface with the hardware at an abstract level. Figure 4.6 shows the dataflow within this Python program. This program, which is the main process, starts five sub-processes using Python's `multiprocessing` package. Each sub-processes is handling one of the following functions: `read_handle`, `read_tracker`, `read_camera`, `control_gripper`, and `record_data`. The first four are interfacing with the hardware via the above defined component classes. Their outputs are written to shared objects between the processes using `multiprocessing.Manager`. These object can then be read by the fifth process, `record_data`, that takes care of workflow control (i.e. the recording state) and the aggregation, synchronization, and export of recorded data. In this process, a simple state machine waits for a press of the record button and then starts to save the recorded data into memory at a defined frequency. This is done by reading the shared objects, which are written by the other processes. When the button is pressed again, `record_data` will create HDF5 datasets, write the data from memory into them, and save them to a `.h5` file. The choice of this data format is discussed below.

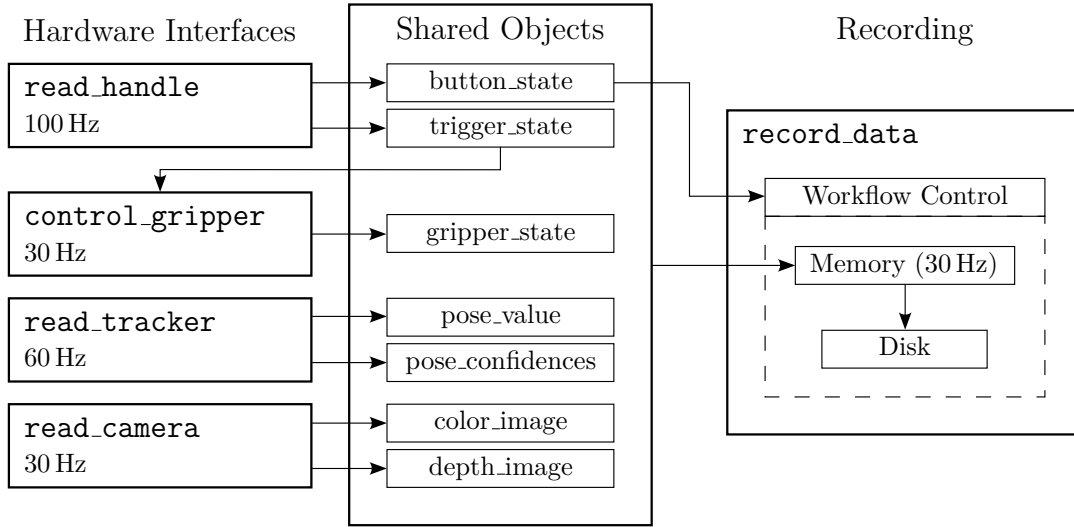


Figure 4.6: Schematic dataflow within the main program. Four processes (left) are interfacing with hardware and writing their data to shared objects (center). These shared objects are read by the recording process (right) at a defined frequency of 30 Hz and saved to disk when the episode is finished.

A few things should be noted here. Most of the recent datasets like DROID [KhazatskyEtAl24] and OXE [PadalkarEtAl23] provide datapoints at a low frequency around 10 to 15 Hz. This is mainly due to the current in-efficiency of large policies. Their inference often can not run faster than 10 Hz for computational reasons. The proposed recording pipeline in this thesis runs at 30 Hz to enable future-proof data collection at a higher temporal resolution. Running the SLAM algorithm in `read_tracker` at targeted frequency of 60 Hz, has proven to result in better pose data than running it a lower frequency like 30 Hz.

For debugging and post-processing the software suite includes a few notable programs: (1) multiple visualization methods, which can easily be used to visualize data after recording, (2) a web viewer, which broadcasts the image streams to the network and helps to validate camera data, and (3) testing scripts for every individual component.



Name	Shape	Data Type	Size
<b>timestamp</b>	<b>1</b>	<b>uint64</b>	<b>8 bytes</b>
image_timestamp	1	uint64	8 bytes
<b>color_image</b>	<b>480×640×3</b>	<b>uint8</b>	<b>912,600 bytes</b>
<b>depth_image</b>	<b>480×640</b>	<b>uint16</b>	<b>614,400 bytes</b>
pose_timestamp	1	uint64	8 bytes
<b>pose_value</b>	<b>4×4</b>	<b>float64</b>	<b>128 bytes</b>
pose_confidence	1	uint8	1 byte
trigger_timestamp	1	uint64	8 bytes
trigger_state	1	uint8	1 byte
gripper_timestamp	1	uint64	8 bytes
<b>gripper_state</b>	<b>1</b>	<b>uint8</b>	<b>1 byte</b>

Table 4.2: Data structure of a single timestep. Each sensor reading is saved with a corresponding timestamp to preserve the information at what time exactly the reading was made. Datatypes most relevant to policy learning are bold.

### Data Format

The data is recorded at 30 Hz. Each trajectory contains the following elements:

- A RGB camera stream at  $640 \times 480$  resolution.
- A depth camera stream at  $640 \times 480$  resolution.
- The end-effector poses (6 DoF) relative to the initial pose at  $t_0$ .
- The robot gripper states (1 DoF).
- The timesteps at which the datapoints are recorded.

The data is saved in the commonly used Hierarchical Data Format version 5 (HDF5). This format is suitable for robot learning data because it allows for the storage of large and complex datasets in a single file, and therefore making data organization and access more straightforward. HDF5 supports a hierarchical structure, enabling the creation of datasets and groups that mirror the logical organization of robot learning data. In our case every data stream is stored in its own dataset within the HDF5 file. The rows of Tab. 4.2 each correspond to a dataset in this file, which stores multiple of these values. A single datapoint of the trajectory consists of 1,527,171 bytes. This equals to 1.456 MB per timestep. At a recording frequency of 30 Hz, one episode with a typical length of 20s, is approximately 870 MB large.



# Chapter 5

## Evaluations

The goal of this thesis is to create an intuitive and effective data collection interface that reduces the embodiment gap, gathers transferable data, and facilitates the learning of various manipulation skills for robots. Chapter 4 proposed a data collection framework, which takes the form of a sensorized hand-held gripper. This chapter aims to evaluate the proposed system under two different aspects; (1) How accurate can we recover robot-native actions from human motions? And (2) how efficient is the data collection process? First, Sec. 5.1 describes the over-arching evaluation strategy. This is followed by analyses on the ability to accurately recover actions in Sec. 5.2 and on the data collection efficiency in Sec. 5.3. Finally, the experimental results are discussed in Sec. 5.4.

### 5.1 Validation Strategy

Different aspects are important to consider when evaluating a framework for its ability to record demonstrations. This section will provide a short discussion on what these aspects are, how this is usually done in other works, and how this chapter aims to validate the proposed system.

As identified by recent works like UMI [ChiEtAl24], two aspects are central to evaluating a data collection framework:

**Accuracy (Sec. 5.2):** How well can robot-native actions be recovered from human demonstrations? How does the framework perform in diverse environments? – Data quality goes hand in hand with transferability to an effective robot policy.

**Efficiency (Sec. 5.3):** How fast can manipulation data be recorded? Is the usage intuitive and operator-friendly? – Efficiency in data collection usually corresponds to the scale at which demonstrations can be collected.

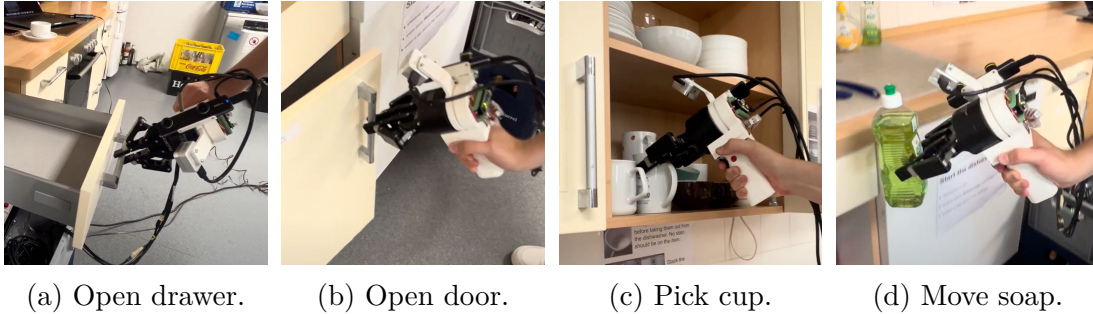


Figure 5.1: Example demonstrations of various tasks. The system’s demonstration capabilities are only limited by the operator’s motion range, the parallel gripper’s physical limits, and the need for a power outlet.

When considering recent works (see Chap. 3) for validation strategies, it can be noted, that these works often evaluate their data collection frameworks on how well recorded demonstrations transfer to effective robot policies. For this, there is a line of work on evaluating policy performance for manipulation. Examples for this are RL Bench [JamesEtAl20], MetaWorld [YuEtAl20], and FMB [LuoEtAl24]. These benchmarks aim to assess the learned policy on manipulation tasks, but not the ability to actually perform and record demonstrations. This thesis focuses on data collection and not policy learning. Therefore, only the ability to record demonstrations is evaluated.

The proposed framework’s ability to capture manipulation skills is studied with a single manipulation task: *cup arrangement*. Evaluating the demonstration capabilities using only one task is sufficient because demonstrating other task is trivial and shown in Fig. 5.1.

The *cup arrangement* task is specifically suitable because it is also used to evaluate the UMI gripper in [ChiEtAl24] and therefore provides a quantitative baseline to compare our device’s data collection efficiency to the state-of-the-art framework UMI. Additionally, the task can be used for the qualitative analysis of the action recovery system since it is a manipulation consisting of complex and diverse motions.

The *cup arrangement* task is defined as follows: Place a cup on the saucer with its handle facing to the upper right of the robot or operator (see Fig. 5.2). Task success is defined as: the cup is placed upright on the saucer with its handle oriented inside the upper right quadrant of the saucer. This task tests the system’s ability to demonstrate both prehensile (pick and place) and non-prehensile actions (i.e., pushing to re-orientate the cup).

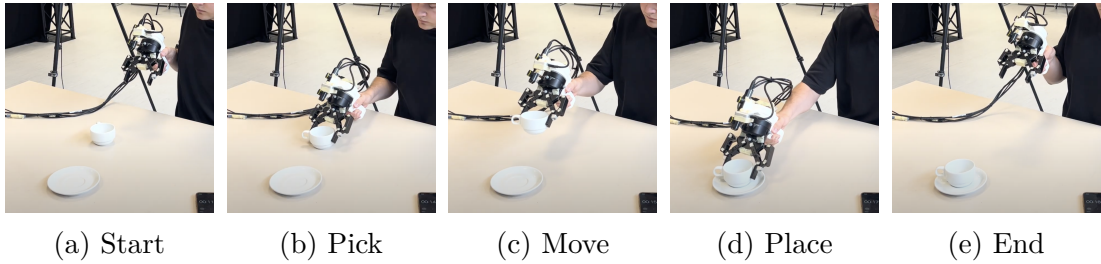


Figure 5.2: *Cup Arrangement*: The task is to place a cup on the saucer with its handle facing to the upper right.

As pointed out in Sec. 3.1, one of the most important characteristics of demonstration datasets is data diversity. To test the system’s ability to capture diverse scenes, we evaluate in five different environments, which are shown in Fig. 5.3. When data collection is possible in any environment, we refer to this as *in-the-wild*.

In addition to performing the *cup arrangement* task, we utilize two different methods for quantifying the system’s accuracy; (1) a motion capture system, which provides an accurate ground truth trajectory for comparison (but is limited to a specific lab setting) and (2) a contraption which guides the system through a predefined motion path, therefore also providing a ground truth path (but without timestamps).

## 5.2 Action Data Accuracy

Demonstration data quality usually corresponds to transferability to an effective robot policy. To evaluate the quality of collected demonstrations, we analyze the developed system in two ways; quantitatively (Sec. 5.2.1) and qualitatively (Sec. 5.2.2). Quantitatively, pre-defined motions are performed in five environments, which can then be compared to the device’s estimated pose data. Additionally, for the Vicon Lab environment, ground truth trajectories are collected while performing the *cup arrangement* task (which is stochastic by nature). Qualitatively, the *cup arrangement* task is performed in all five environments. Then, the recorded pose data can be used to discuss qualitative characteristics.

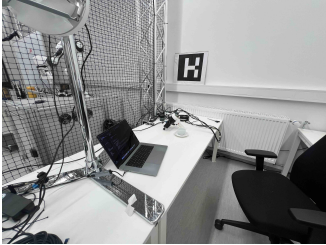






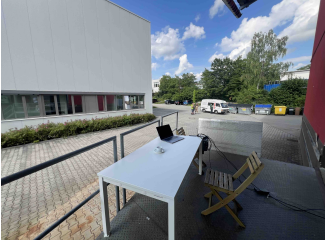

Environment	Tracking Camera View	
		<p>Lab (Small)</p> <ul style="list-style-type: none"> <li>- artificial lighting</li> <li>- artificial features</li> <li>- short distances (1-2 m)</li> </ul>
		<p>Lab (Large)</p> <ul style="list-style-type: none"> <li>- artificial lighting</li> <li>- natural features</li> <li>- long distances (&gt;5 m)</li> </ul>
		<p>Lab (Vicon)</p> <ul style="list-style-type: none"> <li>- artificial &amp; natural lighting</li> <li>- natural features</li> <li>- medium distances (3-5 m)</li> </ul>
		<p>Kitchen</p> <ul style="list-style-type: none"> <li>- artificial lighting</li> <li>- natural features</li> <li>- short distances (&lt;2 m)</li> </ul>
		<p>Outside</p> <ul style="list-style-type: none"> <li>- natural lighting</li> <li>- natural features</li> <li>- long distances (&gt;10 m)</li> </ul>

Figure 5.3: Experimental environments and corresponding exemplary views recorded by the tracking camera.

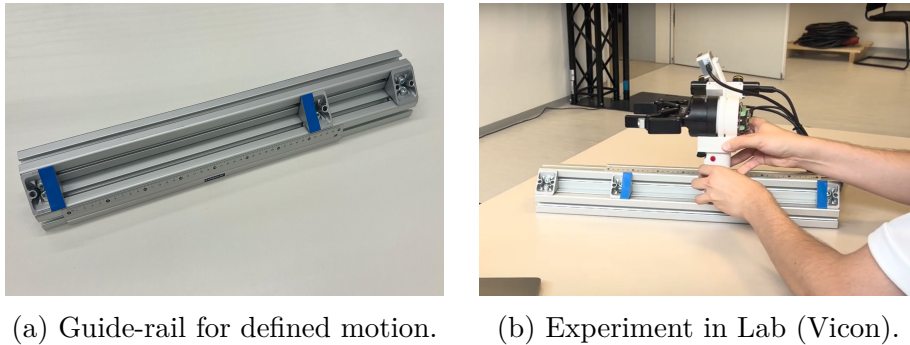


Figure 5.4: A contraption, that guides the device’s motion along a specified distance, is used to quantify pose accuracy in different environments.

### 5.2.1 Quantitative Analysis

To quantify the accuracy of the system’s pose estimation, a guided motion is performed in all five environment. This motion is executed using what is shown in Fig. 5.4 – a contraption that guides the device along a straight line for a specified distance.

After the guard-rail is placed in the workspace, the protocol for this experiment is as follows: 1. fix the device at the start of the guide-rail, 2. start recording a demonstration using the record button, 3. move the device along the guide-rail until the it reaches the specified stop at 30 cm, 4. move it backwards until it reaches the starting point, and 5. stop the recording. This process is repeated three times in all five environments.

In Fig. 5.5, the recorded positions during this experiment are depicted. Three different behaviours can be observed here. The paths recorded in Lab (Small) and Kitchen are very much following the ground truth. In contrast, the paths recorded in Lab (Large) and Lab (Vicon) show a noticeable drift in position. Finally, the position data recorded in the Outside environment shows a path, which is unable to track the ground truth.

Figure 5.6 visualizes the error of the positional tracking during the experiment. Since there are no timestamps available to match the trajectory in the temporal domain, a point-to-curve euclidean distance is chosen as a metric for comparison. Since the pose data from the Outside environment is identified to be impractical in Fig. 5.5, it is omitted here. The observations from above are confirmed using the error metric; paths recorded in Lab (Small) and Kitchen are able to track the ground truth with an error of below 5 mm. For paths from Lab (Large) and Lab (Vicon) a maximum error of 30 mm can be observed at the time, when the device reaches the end of the guard-rail.

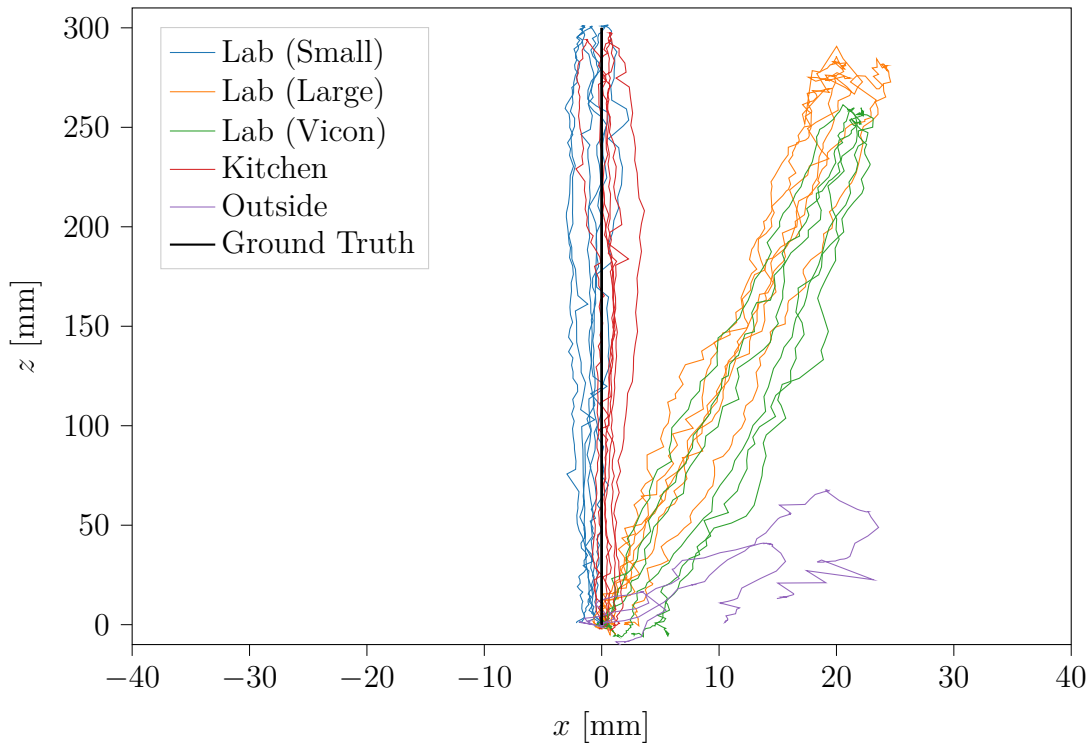


Figure 5.5: Recorded paths and ground truth in the  $x$ - $z$ -plane for five different environments. The device is moved 300 mm along the  $z$ -axis (back and forth). This motion was done three times in every environment.

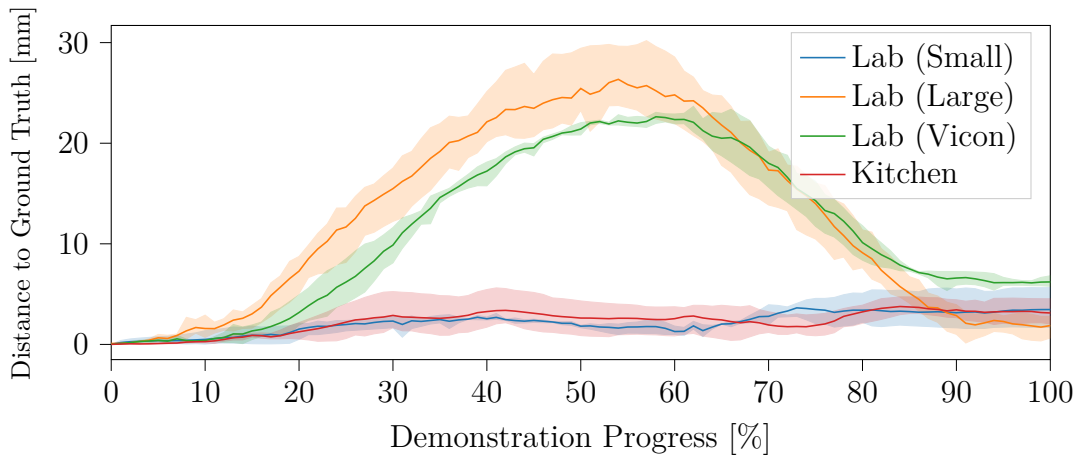


Figure 5.6: Tracking error during the demonstration. The error is calculated using a point-to-curve euclidean distance to the nearest point on the ground truth path.



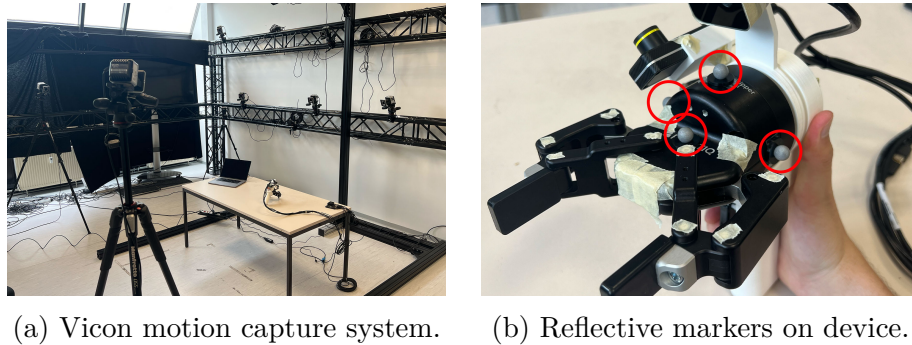


Figure 5.7: Experimental setup using a Vicon motion capture system. It is used to track the movements of reflective markers placed on the proposed device.

In addition to guard-rail method, which is only able to provide a ground truth for one-dimensional motions, a Vicon motion capture system [Vicon24] was used in the Lab (Vicon) environment to acquire accurate ground truth measurements for complex motions. For this, reflective markers are placed on the device. A network of cameras emits infrared light, which reflects off the markers and back to the cameras. The system then triangulates the position of each marker in three-dimensional space by analyzing the multiple viewpoints from different cameras. This data is processed to create an accurate ground truth pose of the device.

For measurements using the Vicon system the following protocol is used: 1. start recording a demonstration using the record button, 2. perform the *cup arrangement* task, 3. stop the recording, and 4. rearrange the cup and saucer to another initial state for the task. This process was repeated 24 times to get a comprehensive dataset that captures a variety of movements and interactions.

Figure 5.8 shows the positional tracking errors between the estimated pose of the demonstration interface and the actual pose, which was recorded using the Vicon motion capture system. A mean tracking error of around 30 mm with peaks up to 100 mm can be observed. There is no significant change in the pose difference during the progression of the demonstration. Using the measurements, a average tracking error of 32.83 mm is calculated for the *cup arrangement* task in the Lab (Vicon) environment.

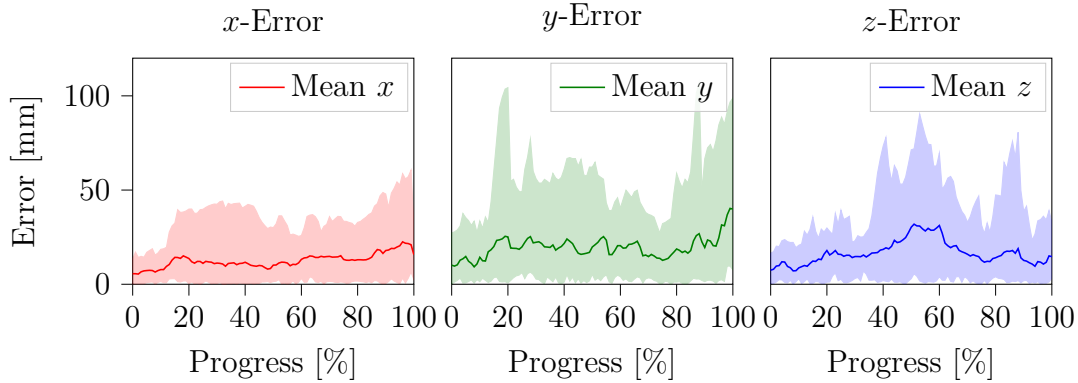


Figure 5.8: Positional tracking error per spatial axis in the Lab (Vicon) environment while performing the *cup arrangement* task.

### 5.2.2 Qualitative Analysis

In Sec. 5.2.1, two methods are used to quantitatively analyze the positional tracking capabilities of the developed system. Both methods are limited in their ability to fully evaluate the tracking ability: the guard-rail method is limiting the ground truth data to one-dimensional motions and the Vicon motion capture system can only be used in one environment, Lab (Vicon). To address this, this section will compare recorded trajectories from all environments qualitatively.

For this the following experiment was conducted: in every environment the *cup arrangement* task was performed three times by the same operator with random initial cup and saucer states. This way, a diverse dataset of trajectories was collected which makes qualitative comparison between the different environments possible.

Figure 5.9 shows a random *cup arrangement* trajectory for each of the five environments. Here, similar results to the ones in Sec. 5.2.1 can be observed. The trajectories recorded in Lab (Small) and Kitchen are look very smooth and do not contain noticeable twitches or noise. In comparison, the trajectories from Lab (Large) and Lab (Vicon) do not share these characteristics. They show unnatural twitching and noise data. Finally, the trajectory recorded in the Outside environment does not represent any reasonable motion.

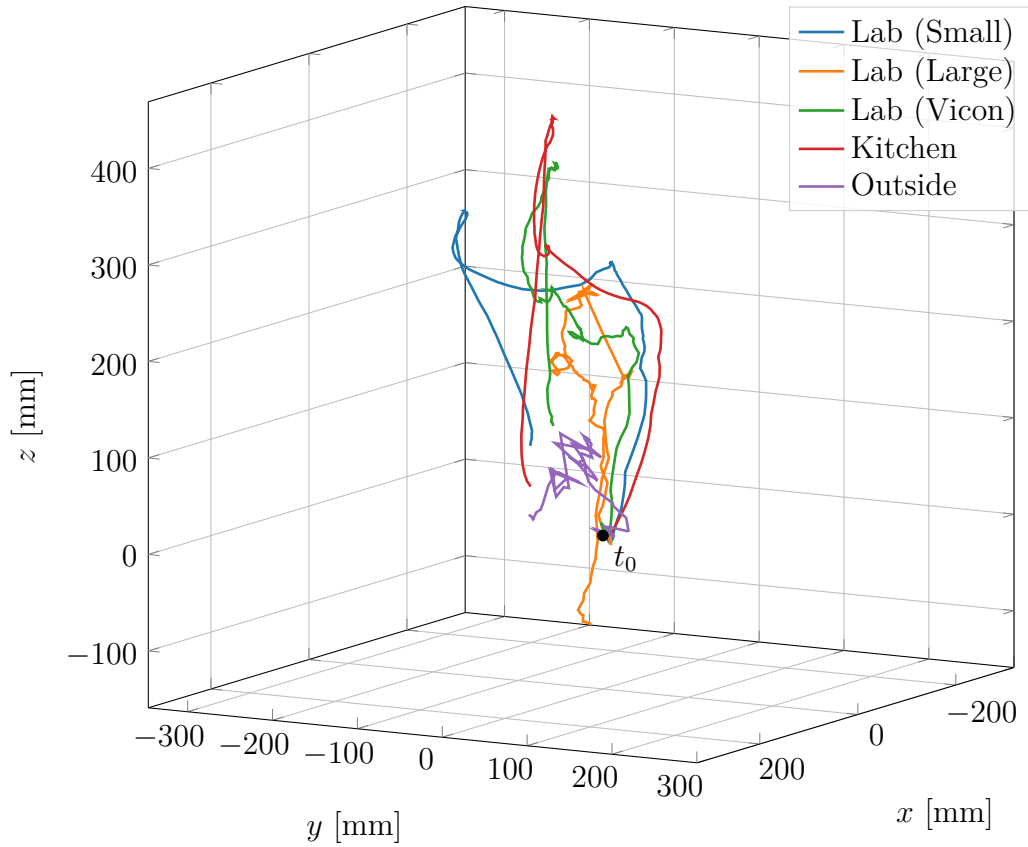


Figure 5.9: Qualitative comparison of *cup arrangement* trajectories in different environments. Trajectories recorded in Lab (Small) and Kitchen are smooth and twitch-free. Trajectories from Lab (Large) and Lab (Vicon) look noisy and include small twitches, but still represent reasonable motion. The trajectory recorded in the Outside environment does not show a feasible motion.

## 5.3 Data Collection Efficiency

For acquiring large and diverse datasets it is important that the method of acquisition is scalable. Metrics for how scalable a data collection approach is, can be the cost of a setup and how accessible its components are. Another metric for scalability is the efficiency at which demonstrations can be recorded. For this, completion times or throughput of demonstrations are usually considered. This section will quantify and compare the throughput of the proposed system to other recent data collection methods.

To measure the throughput at which manipulation tasks can be demonstrated, we conduct a simple user study with three participants. Each user was granted a brief 1-minute orientation session, introducing them to the basics of the device. This is then followed by a 2-minute practice phase, allowing them to gain familiarity with the device and its usage. Following the practice phase, the participants begin the task execution phase, with a time limit of 3 minutes. During this time the users perform the following steps: 1. start the recording of an episode, 2. perform the *cup arrangement* task, 3. stop the recording, and 4. rearrange the cup and saucer to a new initial position. These steps are repeated until the 3 minutes time are over. Note that the time taken to reset the environment, randomize objects, and handle system faults are also counted in this experiment to accurately represent the real-world data collection throughput.

Figure 5.10 shows the results of the user study. The three participants were able to collect 16, 13, and 11 demonstrations in 3 minutes. This results in an average of 267 collections per hour with the proposed device. The state-of-the-art handheld gripper UMI [ChiEtAl24] reports an average of 111 collections per hour for the *cup arrangement* task. For proper comparison with UMI, a new baseline for human hand collection was established: 680 collections per hour. UMI reports a 48% speed of the human hand. Using the established baseline, the proposed device reaches 39% speed of the human hand. Note, that this comparison is relative and the proposed device can capture more than double the amount of absolute demonstrations collected by UMI within an hour.

During the user study, feedback was collected from the participants. All participants reported that the use of the device was intuitive and therefore easy to learn. As a downside, the participants reported that the data collection becomes strenuous after a couple of minutes because of the device's front-facing center of gravity (caused by the gripper).

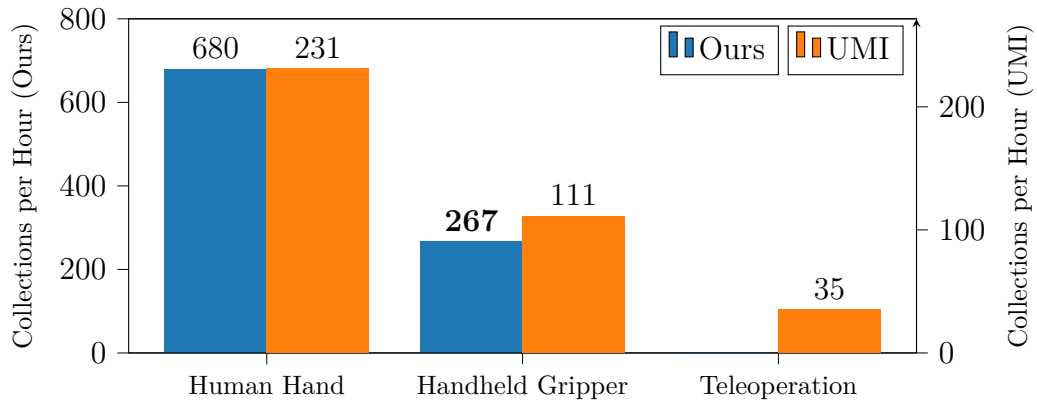


Figure 5.10: Throughput comparison. Collections per hour for the *cup arrangement* task are compared to the human hand speed and the results reported by UMI [ChiEtAl24]. Teleoperation is done with a 3D space-mouse.

## 5.4 Discussion

This chapter aimed to analyze the proposed system’s effectiveness for collecting manipulation data. For this, two central aspects were evaluated: data quality and the efficiency of data collection. This section will briefly discuss the results and try to identify shortcomings of the developed device.

### Action Data Accuracy

The action recovery system was evaluated using quantitative and qualitative methods. Section 5.2 described how three different behaviours are observed. First, accurate and smooth action recovery at tracking errors below 5 mm for the Lab (Small) and Kitchen environments is observed. Second, tracking behaviour with positional drifts and twitches for the Lab (Large) and Lab (Vicon) environments. An average trajectory tracking error of 33 mm was identified in Lab (Vicon) using a motion capture system. Finally, when performing demonstrations in the Outside environment, the recorded trajectory data does not represent any reasonable motion.

When looking at the characteristics of the used environments, a correlation between tracking performance and distance to features in the environment can be observed. In Lab (Small) and Kitchen, the tracking camera is recording features like a wall or a cabinet in close proximity. For Lab (Large) and Lab (Vicon), features like the wall are further away. In the Outside environment, the tracking camera mainly records the sky. Note, the SLAM algorithm used to recover pose data is using RGB-images, IMU data, and depth information. An explanation for the observed behaviour can be that the SLAM algorithm, and therefore tracking

performance, heavily depend on features in the depth domain – when recording demonstrations in close proximity to the environment’s features, tracking performance is noticeably better than if the features are further away from the device.

Since the source code of the tracking algorithm, used by the ZED Mini tracking camera, is not publicly available, no information on the importance of depth data for tracking could be gathered.

The state-of-the-art hand-held data collection system UMI [ChiEtAl24] reports a SLAM tracking accuracy of 6.1 mm, determined using a motion capture benchmark. Note, UMI performs pose recovery in post-processing, after recording the demonstrations, utilizing ORB-SLAM3 [CamposEtAl21]. In comparison, the system proposed in this thesis is capable of real-time action recovery with an accuracy of below 5 mm in environments where features are close to the device.

### Data Collection Efficiency

The speed at which demonstration can be collected was analyzed during a user study with three participants. After a short orientation and practice phase, the users performed the *cup arrangement task* for three minutes. On average, the proposed device enables data collection at a speed of 267 demonstrations per hour (doing *cup arrangement*). In comparison, the throughput when demonstrating this task with UMI [ChiEtAl24] reported to be 111 demonstrations per hour. That means, the developed device more than doubles the data collection speed compared to UMI. A reason for this is likely the improved workflow control using a thumb button. With the proposed device, the operator does not need a second hand to start and stop data recordings.

To compare the device’s throughput to the human hand speed, a baseline was established: 680 collections per hour for the *cup arrangement task*. This is more than double the speed reported by UMI for the human hand baseline. The discrepancy may be due to the stochastic nature of the *cup arrangement task*, as different operators may perform the task in varying ways. Therefore, comparing absolute demonstration speeds may not fully capture the efficiency of the devices. To address this, relative speeds of data collection compared to the human hand speed should be considered. For the developed method, a relative speed of 39% was identified. This is less than the relative speed of 48% reported by UMI.

While the developed method captures demonstrations faster than UMI in absolute terms, it is slower when compared to human speed. Note that a limitation of this analysis is the small sample size of participants and the number of demonstrations compared to UMI’s analysis. Future work could involve a more extensive analysis with a larger sample size.

# Chapter 6

## Conclusion

This thesis covered the design of a suitable concept for portable data collection using a sensorized hand-held gripper interface. This new device extends the capabilities of state-of-the-art interfaces in three ways. First, it enables real-time data acquisition, including visual and depth information. Second, it simplifies workflow control by introducing easy-to-use control interfaces. Finally, it improves accessibility by supporting the use of different end-effectors. The following provides a concise summary of the thesis' findings and concludes with an outlook on promising directions for future work.

### 6.1 Summary

Starting from the initial problem statement, the fundamentals of learning manipulation tasks from demonstration were presented in Chap. 2. It begins by formalizing manipulation learning tasks using POMDPs and describing the components such as state and action spaces, and policy structure. Various approaches to learning from demonstrations (LfD) are then discussed, highlighting the distinction between demonstration and imitation methods. The chapter outlines methodologies for gathering demonstrations, addressing the correspondence problem, and details methods for deriving policies from demonstration data, focusing on mapping functions for policy learning.

A review of the state-of-the-art policy learning and data collection methods was presented in Chap. 3. It discussed learning algorithms and datasets, highlighting the shift towards deep neural networks and transformer-based models due to their ability to handle multimodal observations and scale with large, diverse datasets. Various data collection methods were examined, including visual demonstrations, shadowing, teleoperation, and hand-held tools, each with its ad-

vantages and limitations. Hand-held tools were identified as a promising data collection approach because of their portability and intuitive use. Limitations of hand-held methods were identified in sensing, feedback, workflow complexity, and end-effector specificity.

Based on this, a concept concept for an intuitive and effective data collection interface was developed in Chap. 4. The proposed system features a 3D-printed handle equipped with a RGB-D camera, a tracking camera, a gripper control trigger, and a recording button. These elements facilitate the capture and transfer of human manipulation skills into a robot-native data format and ensure forward compatibility with real-time data processing. The modular design supports various end-effectors and follows ISO standards to maintain compatibility with commonly used robots. Detailed explanations cover the Observation Module’s design, including the strategic placement of the wrist-mounted camera to minimize the observation gap, and the Action Handle’s mapping of human actions using visual-inertial SLAM for pose estimation. The chapter also outlines the system’s hardware setup, workflow, and data processing methods.

In order to obtain first insights on the effectiveness of the proposed system, the device was evaluated in Chap. 5. This chapter assesses the system’s accuracy in recovering robot-native actions from human motions and the efficiency of the data collection process. The validation strategy includes quantitative and qualitative analyses, focusing on a specific manipulation task, *cup arrangement*, performed in various environments. Quantitative tests using guided motions and a Vicon motion capture system show that the system performs well in environments with nearby features but struggles in featureless or large spaces. Qualitative analysis further supports these findings. Efficiency is assessed through a user study, showing the developed system allows fast data collection but is slightly less efficient relative to human hand speed when compared to other hand-held methods. The chapter concludes by discussing the correlation between tracking performance and environmental features.

To conclude, this thesis presented an effective data collection system for acquiring demonstrations for the learning of manipulation tasks.



## 6.2 Outlook

This thesis offers several promising directions for future work. One key area is the further experimentation and evaluation of transferring recorded data to effective robot policies. This involves collecting a larger dataset of demonstrations across various manipulation tasks. Using this expanded dataset, a manipulation policy such as ACT [ZhaoEtAl23] or Diffusion Policy [ChiEtAl23] could be trained and assessed on state-of-the-art manipulation benchmarks for robot learning.

Multiple ablation studies should be conducted to analyze the performance of the manipulation policy concerning the data collected by the proposed hand-held device. Factors to be examined include camera positioning and field of view, the use of depth data, temporal frequency of data points, and the use of different end-effectors.

Regarding action data accuracy, the SLAM-based pose estimations have proven to be the most fragile part of the action transfer process, indicating a need for further improvements. Comparative studies of different cameras and SLAM implementations could be beneficial. Another approach could involve testing different localization methods, such as external static tracking cameras. The use of a wrist-mounted fish-eye lens to capture sufficient visual context, as demonstrated by UMI [ChiEtAl24], appears to be a particularly promising approach.

For data collection efficiency, our user study was limited to inexperienced users who received only brief training. Additional, more comprehensive training could significantly improve users' proficiency in using the device, and this could be explored in future studies.

A notable shortcoming of hand-held data collection devices is the potential to collect valid but hardware-infeasible trajectories (see Sec. 3.3). In previous works like UMI, this issue was addressed through post-processing and filtering of the data. The device proposed in this thesis enables real-time data processing and could simulate the kinematic constraints of downstream robots. These calculations could then be used to provide immediate feedback to the operator during data collection, alerting them to any infeasible paths being demonstrated.



# Bibliography

- [3Dconnexion24] 3Dconnexion: Spacemouse. <https://3dconnexion.com/dk/spacemouse/>, 2024. Accessed: 2024-07-05.
- [AldacoEtAl24] Aldaco, J.; Armstrong, T.; Baruch, R.; Bingham, J.; Chan, S.; Draper, K.; Dwibedi, D.; Finn, C.; Florence, P.; Goodrich, S.; et al.: Aloha 2: An enhanced low-cost hardware for bimanual teleoperation. arXiv preprint arXiv:2405.02292, 2024.
- [ArgallEtAl09] Argall, B.D.; Chernova, S.; Veloso, M.; Browning, B.: A survey of robot learning from demonstration. *Robotics and autonomous systems*, Vol. 57, No. 5, pp. 469–483, 2009.
- [ArunachalamEtAl23a] Arunachalam, S.P.; Güzey, I.; Chintala, S.; Pinto, L.: Holo-dex: Teaching dexterity with immersive mixed reality. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5962–5969, IEEE, 2023.
- [ArunachalamEtAl23b] Arunachalam, S.P.; Silwal, S.; Evans, B.; Pinto, L.: Dexterous imitation made easy: A learning-based framework for efficient dexterous manipulation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5954–5961, IEEE, 2023.
- [BahlGuptaPathak22] Bahl, S.; Gupta, A.; Pathak, D.: Human-to-robot imitation in the wild. arXiv preprint arXiv:2207.09450, 2022.
- [BharadhwajEtAl23] Bharadhwaj, H.; Vakil, J.; Sharma, M.; Gupta, A.; Tulsiani, S.; Kumar, V.: Roboagent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking. arXiv preprint arXiv:2309.01918, 2023.
- [BicchiKumar00] Bicchi, A.; Kumar, V.: Robotic grasping and contact: A review. In *Proceedings 2000 ICRA. Millennium conference. IEEE international conference on robotics and automation. Symposia proceedings (Cat. No. 00CH37065)*, Vol. 1, pp. 348–353, IEEE, 2000.

- [BrohanEtAl22] Brohan, A.; Brown, N.; Carbajal, J.; Chebotar, Y.; Dabis, J.; Finn, C.; Gopalakrishnan, K.; Hausman, K.; Herzog, A.; Hsu, J.; et al.: Rt-1: Robotics transformer for real-world control at scale. arXiv preprint arXiv:2212.06817, 2022.
- [BrohanEtAl23] Brohan, A.; Brown, N.; Carbajal, J.; Chebotar, Y.; Chen, X.; Choromanski, K.; Ding, T.; Driess, D.; Dubey, A.; Finn, C.; et al.: Rt-2: Vision-language-action models transfer web knowledge to robotic control. arXiv preprint arXiv:2307.15818, 2023.
- [CamposEtAl21] Campos, C.; Elvira, R.; Rodríguez, J.J.G.; Montiel, J.M.; Tardós, J.D.: Orb-slam3: An accurate open-source library for visual, visual-inertial, and multimap slam. IEEE Transactions on Robotics, Vol. 37, No. 6, pp. 1874–1890, 2021.
- [ChengEtAl24] Cheng, X.; Li, J.; Yang, S.; Yang, G.; Wang, X.: Open-television: Teleoperation with immersive active visual feedback. arXiv preprint arXiv:2407.01512, 2024.
- [ChiEtAl23] Chi, C.; Feng, S.; Du, Y.; Xu, Z.; Cousineau, E.; Burchfiel, B.; Song, S.: Diffusion policy: Visuomotor policy learning via action diffusion. In Proceedings of Robotics: Science and Systems (RSS), 2023.
- [ChiEtAl24] Chi, C.; Xu, Z.; Pan, C.; Cousineau, E.; Burchfiel, B.; Feng, S.; Tedrake, R.; Song, S.: Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. In Proceedings of Robotics: Science and Systems (RSS), 2024.
- [DasariEtAl19] Dasari, S.; Ebert, F.; Tian, S.; Nair, S.; Bucher, B.; Schmeckpeper, K.; Singh, S.; Levine, S.; Finn, C.: Robonet: Large-scale multi-robot learning. arXiv preprint arXiv:1910.11215, 2019.
- [Durrant-WhyteBailey06] Durrant-Whyte, H.; Bailey, T.: Simultaneous localization and mapping: part i. IEEE robotics & automation magazine, Vol. 13, No. 2, pp. 99–110, 2006.
- [FangEtAl23] Fang, H.S.; Fang, H.; Tang, Z.; Liu, J.; Wang, J.; Zhu, H.; Lu, C.: Rh20t: A robotic dataset for learning diverse skills in one-shot. In RSS 2023 Workshop on Learning for Task and Motion Planning, 2023.
- [Force Dimension24] Force Dimension: Haptic devices. <https://www.forcedimension.com/products>, 2024. Accessed: 2024-07-05.
- [FuEtAl24] Fu, Z.; Zhao, Q.; Wu, Q.; Wetzstein, G.; Finn, C.: Human-plus: Humanoid shadowing and imitation from humans. arXiv preprint arXiv:2406.10454, 2024.

- [FuZhaoFinn24] Fu, Z.; Zhao, T.Z.; Finn, C.: Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation. arXiv preprint arXiv:2401.02117, 2024.
- [GervetEtAl23] Gervet, T.; Xian, Z.; Gkanatsios, N.; Fragkiadaki, K.: Act3d: 3d feature field transformers for multi-task robotic manipulation. In 7th Annual Conference on Robot Learning, 2023.
- [GhoshEtAl24] Ghosh, D.; Walke, H.; Pertsch, K.; Black, K.; Mees, O.; Dasari, S.; Hejna, J.; Xu, C.; Luo, J.; Kreiman, T.; Tan, Y.; Chen, L.Y.; Sanketi, P.; Vuong, Q.; Xiao, T.; Sadigh, D.; Finn, C.; Levine, S.: Octo: An open-source generalist robot policy. In Proceedings of Robotics: Science and Systems, Delft, Netherlands, 2024.
- [GoyalEtAl24] Goyal, A.; Blukis, V.; Xu, J.; Guo, Y.; Chao, Y.W.; Fox, D.: Rvt-2: Learning precise manipulation from few demonstrations. arXiv preprint arXiv:2406.08545, 2024.
- [HaFlorenceSong23] Ha, H.; Florence, P.; Song, S.: Scaling up and distilling down: Language-guided robot skill acquisition. In Conference on Robot Learning, pp. 3766–3777, PMLR, 2023.
- [HandaEtAl20] Handa, A.; Van Wyk, K.; Yang, W.; Liang, J.; Chao, Y.W.; Wan, Q.; Birchfield, S.; Ratliff, N.; Fox, D.: Dexpilot: Vision-based teleoperation of dexterous robotic hand-arm system. In 2020 IEEE International Conference on Robotics and Automation (ICRA), pp. 9164–9170, IEEE, 2020.
- [HastieEtAl09] Hastie, T.; Tibshirani, R.; Friedman, J.H.; Friedman, J.H.: The elements of statistical learning: data mining, inference, and prediction, Vol. 2. Springer, 2009.
- [HayesDemiris94] Hayes, G.M.; Demiris, J.: A robot controller using learning by imitation. University of Edinburgh, Department of Artificial Intelligence Edinburgh, UK, 1994.
- [HeEtAl24] He, T.; Luo, Z.; He, X.; Xiao, W.; Zhang, C.; Zhang, W.; Kitani, K.; Liu, C.; Shi, G.: Omnih2o: Universal and dexterous human-to-humanoid whole-body teleoperation and learning. arXiv preprint arXiv:2406.08858, 2024.
- [HoJainAbbeel20] Ho, J.; Jain, A.; Abbeel, P.: Denoising diffusion probabilistic models. Advances in neural information processing systems, Vol. 33, pp. 6840–6851, 2020.

- [Howard60] Howard, R.A.: Dynamic programming and markov processes. John Wiley, 1960.
- [ISO04] ISO: Iso 9409-1 manipulating industrial robots - mechanical interfaces - part 1: Plates, 2004. Accessed: 2024-07-05.
- [JamesEtAl20] James, S.; Ma, Z.; Arrojo, D.R.; Davison, A.J.: Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, Vol. 5, No. 2, pp. 3019–3026, 2020.
- [KaelblingLittmanCassandra98] Kaelbling, L.P.; Littman, M.L.; Cassandra, A.R.: Planning and acting in partially observable stochastic domains. *Artificial intelligence*, Vol. 101, No. 1-2, pp. 99–134, 1998.
- [KalashnikovEtAl21] Kalashnikov, D.; Varley, J.; Chebotar, Y.; Swanson, B.; Jonschkowski, R.; Finn, C.; Levine, S.; Hausman, K.: Mt-opt: Continuous multi-task robotic reinforcement learning at scale. *arXiv preprint arXiv:2104.08212*, 2021.
- [KempEtAl22] Kemp, C.C.; Edsinger, A.; Clever, H.M.; Matulevich, B.: The design of stretch: A compact, lightweight mobile manipulator for indoor human environments. In *2022 International Conference on Robotics and Automation (ICRA)*, pp. 3150–3157, IEEE, 2022.
- [KhazatskyEtAl24] Khazatsky, A.; Pertsch, K.; Nair, S.; Balakrishna, A.; Dasari, S.; Karamcheti, S.; Nasiriany, S.; Srirama, M.K.; Chen, L.Y.; Ellis, K.; et al.: Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.
- [KimEtAl24] Kim, M.J.; Pertsch, K.; Karamcheti, S.; Xiao, T.; Balakrishna, A.; Nair, S.; Rafailov, R.; Foster, E.; Lam, G.; Sanketi, P.; et al.: Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.
- [KormushevCalinonCaldwell11] Kormushev, P.; Calinon, S.; Caldwell, D.G.: Imitation learning of positional and force skills demonstrated via kinesthetic teaching and haptic input. *Advanced Robotics*, Vol. 25, No. 5, pp. 581–603, 2011.
- [KroemerNiekumKonidaris21] Kroemer, O.; Niekum, S.; Konidaris, G.: A review of robot learning for manipulation: Challenges, representations, and algorithms. *Journal of machine learning research*, Vol. 22, No. 30, pp. 1–82, 2021.
- [LeCunBengioHinton15] LeCun, Y.; Bengio, Y.; Hinton, G.: Deep learning. *nature*, Vol. 521, No. 7553, pp. 436–444, 2015.

- [LeCunBengioothers95] LeCun, Y.; Bengio, Y.; et al.: Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, Vol. 3361, No. 10, p. 1995, 1995.
- [LeeEtAl24] Lee, S.; Wang, Y.; Etukuru, H.; Kim, H.J.; Shafiullah, N.M.M.; Pinto, L.: Behavior generation with latent actions. *arXiv preprint arXiv:2403.03181*, 2024.
- [LevineEtAl16] Levine, S.; Finn, C.; Darrell, T.; Abbeel, P.: End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, Vol. 17, No. 39, pp. 1–40, 2016.
- [LinEtAl24] Lin, T.; Zhang, Y.; Li, Q.; Qi, H.; Yi, B.; Levine, S.; Malik, J.: Learning visuotactile skills with two multifingered hands. *arXiv preprint arXiv:2404.16823*, 2024.
- [LuoEtAl24] Luo, J.; Xu, C.; Liu, F.; Tan, L.; Lin, Z.; Wu, J.; Abbeel, P.; Levine, S.: Fmb: A functional manipulation benchmark for generalizable robotic learning. *arXiv preprint arXiv:2401.08553*, 2024.
- [Mason81] Mason, M.T.: Compliance and force control for computer controlled manipulators. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 11, No. 6, pp. 418–432, 1981.
- [MeesEtAl22] Mees, O.; Hermann, L.; Rosete-Beas, E.; Burgard, W.: Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, Vol. 7, No. 3, pp. 7327–7334, 2022.
- [Meta24] Meta: Meta quest 2. <https://www.meta.com/de/en/quest/products/quest-2/>, 2024. Accessed: 2024-06-18.
- [NairEtAl22a] Nair, S.; Mitchell, E.; Chen, K.; Savarese, S.; Finn, C.; et al.: Learning language-conditioned robot behavior from offline data and crowd-sourced annotation. In *Conference on Robot Learning*, pp. 1303–1315, PMLR, 2022.
- [NairEtAl22b] Nair, S.; Rajeswaran, A.; Kumar, V.; Finn, C.; Gupta, A.: R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.
- [PadalkarEtAl23] Padalkar, A.; Pooley, A.; Jain, A.; Bewley, A.; Herzog, A.; Irpan, A.; Khazatsky, A.; Rai, A.; Singh, A.; Brohan, A.; et al.: Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.

- [PariEtAl21] Pari, J.; Shafiullah, N.M.; Arunachalam, S.P.; Pinto, L.: The surprising effectiveness of representation learning for visual imitation. arXiv preprint arXiv:2112.01511, 2021.
- [PerezEtAl18] Perez, E.; Strub, F.; De Vries, H.; Dumoulin, V.; Courville, A.: Film: Visual reasoning with a general conditioning layer. In Proceedings of the AAAI conference on artificial intelligence, Vol. 32, 2018.
- [QinEtAl22] Qin, Y.; Wu, Y.H.; Liu, S.; Jiang, H.; Yang, R.; Fu, Y.; Wang, X.: Dexmv: Imitation learning for dexterous manipulation from human videos. In European Conference on Computer Vision, pp. 570–587, Springer, 2022.
- [QinEtAl23] Qin, Y.; Yang, W.; Huang, B.; Van Wyk, K.; Su, H.; Wang, X.; Chao, Y.W.; Fox, D.: Anyteleop: A general vision-based dexterous robot arm-hand teleoperation system. arXiv preprint arXiv:2307.04577, 2023.
- [RadosavovicEtAl23] Radosavovic, I.; Xiao, T.; James, S.; Abbeel, P.; Malik, J.; Darrell, T.: Real-world robot learning with masked visual pre-training. In Conference on Robot Learning, pp. 416–426, PMLR, 2023.
- [RossGordonBagnell11] Ross, S.; Gordon, G.; Bagnell, D.: A reduction of imitation learning and structured prediction to no-regret online learning. In Proceedings of the fourteenth international conference on artificial intelligence and statistics, pp. 627–635, JMLR Workshop and Conference Proceedings, 2011.
- [SchaalEtAl05] Schaal, S.; Peters, J.; Nakanishi, J.; Ijspeert, A.: Learning movement primitives. In Robotics Research. The Eleventh International Symposium: With 303 Figures, pp. 561–572, Springer, 2005.
- [SchmeckpeperEtAl20] Schmeckpeper, K.; Xie, A.; Rybkin, O.; Tian, S.; Daniilidis, K.; Levine, S.; Finn, C.: Learning predictive models from observation and interaction. In European Conference on Computer Vision, pp. 708–725, Springer, 2020.
- [ShafiullahEtAl22] Shafiullah, N.M.; Cui, Z.; Altanzaya, A.A.; Pinto, L.: Behavior transformers: Cloning  $k$  modes with one stone. Advances in neural information processing systems, Vol. 35, pp. 22955–22968, 2022.
- [ShafiullahEtAl23] Shafiullah, N.M.M.; Rai, A.; Etukuru, H.; Liu, Y.; Misra, I.; Chintala, S.; Pinto, L.: On bringing robots home. arXiv preprint arXiv:2311.16098, 2023.
- [ShawBahlPathak23] Shaw, K.; Bahl, S.; Pathak, D.: Videodex: Learning dexterity from internet videos. In Conference on Robot Learning, pp. 654–665, PMLR, 2023.



- [ShridharManuelliFox23] Shridhar, M.; Manuelli, L.; Fox, D.: Perceiver-actor: A multi-task transformer for robotic manipulation. In Conference on Robot Learning, pp. 785–799, PMLR, 2023.
- [SongEtAl20] Song, S.; Zeng, A.; Lee, J.; Funkhouser, T.: Grasping in the wild: Learning 6dof closed-loop grasping from low-cost demonstrations. *Robotics and Automation Letters*, 2020.
- [SpongHutchinsonVidyasagar20] Spong, M.W.; Hutchinson, S.; Vidyasagar, M.: *Robot modeling and control*. John Wiley & Sons, 2020.
- [SteinmetzMontebelliKyrki15] Steinmetz, F.; Montebelli, A.; Kyrki, V.: Simultaneous kinesthetic teaching of positional and force requirements for sequential in-contact tasks. In 2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids), pp. 202–209, 2015.
- [StepputtisEtAl20] Stepputtis, S.; Campbell, J.; Phielipp, M.; Lee, S.; Baral, C.; Ben Amor, H.: Language-conditioned imitation learning for robot manipulation tasks. *Advances in Neural Information Processing Systems*, Vol. 33, pp. 13139–13150, 2020.
- [StereoLabs24] StereoLabs: Zed mini camera overview datasheet, 2024. Accessed: 2024-07-05.
- [TaketomiUchiyamaIkeda17] Taketomi, T.; Uchiyama, H.; Ikeda, S.: Visual slam algorithms: A survey from 2010 to 2016. *IPSJ transactions on computer vision and applications*, Vol. 9, pp. 1–11, 2017.
- [TorabiWarnellStone18] Torabi, F.; Warnell, G.; Stone, P.: Behavioral cloning from observation. *arXiv preprint arXiv:1805.01954*, 2018.
- [VaswaniEtAl17] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I.: Attention is all you need. *Advances in neural information processing systems*, Vol. 30, 2017.
- [Vicon24] Vicon: Motion capture camera. <https://www.vicon.com/hardware/cameras/>, 2024. Accessed: 2024-07-07.
- [WalkeEtAl23] Walke, H.R.; Black, K.; Zhao, T.Z.; Vuong, Q.; Zheng, C.; Hansen-Estruch, P.; He, A.W.; Myers, V.; Kim, M.J.; Du, M.; et al.: Bridgedata v2: A dataset for robot learning at scale. In Conference on Robot Learning, pp. 1723–1736, PMLR, 2023.
- [WangEtAl23] Wang, C.; Fan, L.; Sun, J.; Zhang, R.; Fei-Fei, L.; Xu, D.; Zhu, Y.; Anandkumar, A.: Mimicplay: Long-horizon imitation learning by watching human play. *arXiv preprint arXiv:2302.12422*, 2023.

- [Weiss Robotics24] Weiss Robotics: Wsg series. <https://weiss-robotics.com/servo-electric/wsg-series/>, 2024. Accessed: 2024-07-05.
- [WuEtAl23] Wu, P.; Shentu, Y.; Yi, Z.; Lin, X.; Abbeel, P.: Gello: A general, low-cost, and intuitive teleoperation framework for robot manipulators. arXiv preprint arXiv:2309.13037, 2023.
- [XiongEtAl21] Xiong, H.; Li, Q.; Chen, Y.C.; Bharadhwaj, H.; Sinha, S.; Garg, A.: Learning by watching: Physical imitation of manipulation skills from human videos. In 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 7827–7834, IEEE, 2021.
- [YangEtAl24] Yang, J.; Cao, Z.a.; Deng, C.; Antonova, R.; Song, S.; Bohg, J.: Equibot: Sim (3)-equivariant diffusion policy for generalizable and data efficient learning. arXiv preprint arXiv:2407.01479, 2024.
- [YoungEtAl21] Young, S.; Gandhi, D.; Tulsiani, S.; Gupta, A.; Abbeel, P.; Pinto, L.: Visual imitation made easy. In Conference on Robot Learning, pp. 1992–2005, PMLR, 2021.
- [YuEtAl20] Yu, T.; Quillen, D.; He, Z.; Julian, R.; Hausman, K.; Finn, C.; Levine, S.: Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In Conference on robot learning, pp. 1094–1100, PMLR, 2020.
- [ZhaoEtAl23] Zhao, T.Z.; Kumar, V.; Levine, S.; Finn, C.: Learning fine-grained bimanual manipulation with low-cost hardware. arXiv preprint arXiv:2304.13705, 2023.

# Appendix

## A.1 Contents Archive

There is a folder **PRO\_080\_Grothusen/** in the archive. The main folder contains the entries

- **PRO\_080\_Grothusen.pdf**: the pdf-file of the thesis PRO-080.
- **Data/**: a folder with all the relevant data, programs, scripts and simulation environments.
- **Latex/**: a folder with the \*.tex documents of the thesis PRO-080 written in Latex and all figures (also in \*.svg data format if available).
- **Presentation/**: a folder with the relevant data for the presentation including the presentation itself, figures and videos.



## **Erklärung**

Ich, Jannik Jorge Grothusen (Student der Mechatronik an der Technischen Universität Hamburg, Matrikelnummer 52576), versichere, dass ich die vorliegende Projektarbeit selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel verwendet habe. Die Arbeit wurde in dieser oder ähnlicher Form noch keiner Prüfungskommission vorgelegt.

---

Unterschrift

---

Datum