Abstract

Robot Tool Use

Meiying Qin

2022

Using human tools can significantly benefit robots in many application domains. It will endow a home robot with the ability to carry out everyday household activities such as cleaning. It will enable an industrial robot to work in manufacturing or maintenance using human tools such as screwdrivers. It will allow a robot to perform experiments using existing human lab tools in a chemistry lab. Such ability would allow robots to solve problems that they were unable to without tools.

However, robot tool use is a challenging task. Tool use was initially considered to be the ability that distinguishes human beings from other animals (Oakley, 1944). Robot tool use has three challenges: perception, manipulation, and cognition. While both general manipulation tasks and tool use tasks require the same level of perception accuracy, there are unique manipulation and cognition challenges in robot tool use.

In this dissertation, we first define robot tool use and compile a taxonomy of robot tool use. We identify required skills for each sub-type in the taxonomy and review previous studies on robot tool use based on the taxonomy. Next, we demonstrate our work on solving some of the sub-types of robot tool use. We present an integrated system that trains a robot with tool use, transfers the learned skills to novel objects, and can even improvise the usage of novel tools. The robot that utilizes this system can also generalize the learned skills to other robot platforms without additional training. We then investigate more complicated forms of robot tool use. Specifically, we explore how a robot can plan sequential tool use so that the robot can use one tool to retrieve another tool in order to complete a task. We also examine how a robot should choose the most appropriate tool from many possible tool options. Finally,

we investigate the application of learned tool use skills in related applications such as human-robot collaborations.

We conclude this dissertation with the contributions and limitations of our work. We also identify open challenges that remain to be solved as future directions.

Robot Tool Use

A Dissertation
Presented to the Faculty of the Graduate School
of
Yale University
in Candidacy for the Degree of
Doctor of Philosophy

by
Meiying Qin

Dissertation Director: Brian Scassellati

December 2022

Contents

1 Introduction **1**

2 A Definition, a Taxonomy and a Review of Robot Tool Use **7**

2.1 Definition of Robot Tool Use 7

2.2 A Taxonomy of Robot Tool Use 11

2.3 Required Skills of Robot Tool Use 17

2.4 Robot Tool Use Literature 21

2.4.1 Non-Causal Tool Use 22

2.4.2 Causal Tool Use — Single-Manipulation Tool Use 24

2.4.3 Causal Tool Use — Multiple-Manipulation Tool Use 31

2.5 Discussions 35

3 Learning and Reasoning About Single-Manipulation Tool Use **39**

3.1 Introduction 40

3.1.1 Task-Oriented Approach to Tool Use 41

3.1.2 Star 1: Learning and Applying Task-General Tool Use 42

3.1.3 Star 2: Task-General Object Substitution 44

3.1.4 Star 3: Transferring Tool Use Skills to Other Robot Platforms 46

3.2 Methods 47

3.2.1 Preliminaries 48

3.2.2 Representations 49

3.2.3 Star 1: Learning and Applying Task-General Tool Use Skills . 55
3.2.4 Star 2: Task-General Object Substitution 62
3.2.5 Star 3: Tool Use Transfer to Other Robot Platforms 66
3.3 Results 67
3.3.1 Star 1: Learning and Applying Task-General Tool Use Skills . 67
3.3.2 Star 2: Task-General Object Substitution 74
3.3.3 Star 3: Transfer Tool Use To Other Robot Platforms 79
3.4 Discussion 81
3.4.1 Contribution 1: Task-Generality 81
3.4.2 Contribution 2: Data Efficient 82
3.4.3 Contribution 3: Integrative framework 83
3.4.4 Limitations 84
3.5 Summary 85

4 Multiple-Manipulation Tool Use: Sequential Tool Use 86
4.1 Introduction 88
4.2 Methods 92
4.2.1 TAAMP 92
4.2.2 Learning the Action Affordances 95
4.3 Results 97
4.4 Discussion 100
4.5 Summary 102

5 Multiple-Manipulation Tool Use: Tool Selection 103
5.1 Introduction 103
5.1.1 Causality in ML and Robotics 105
5.1.2 Affordance learning 106
5.2 Methods 107

5.2.1 Problem Statement . . . 107
5.2.2 Our approach . . . 109
5.2.3 Experiments . . . 112
5.2.4 Evaluation . . . 116
5.3 Results . . . 116
5.4 Discussion . . . 119
5.5 Summary . . . 120

6 Applying Tool Use Knowledge in the Context of Human-Robot Collaboration 121

6.1 Introduction . . . 122
6.1.1 Taxonomy: Handover Requirements . . . 123
6.1.2 Task-oriented Handovers . . . 126
6.2 Methods . . . 129
6.2.1 Object Model Generation . . . 129
6.2.2 Vision Module . . . 129
6.2.3 Learning Tool Affordances . . . 130
6.2.4 Grasping Configurations . . . 132
6.2.5 Presentation Configurations . . . 133
6.3 Results . . . 134
6.3.1 Robot Validations . . . 134
6.3.2 Survey . . . 138
6.4 Discussion . . . 139
6.5 Summary . . . 140

7 Conclusion 141

7.1 Contributions . . . 141
7.2 Future Work . . . 142

A Summary Tables of Studies in Chapter 2 **144**

List of Figures

1.1 Components of an action. The diagram is adapted from Qin et al. (2021), CC BY. 3

2.1 Tool Use Taxonomy. 15

2.2 The modified affordance model in Montesano et al. (2008) ©2008, IEEE (The IEEE does not require individuals working on a thesis to obtain a formal reuse license). Affordances as relations between (A)ctions, (O)bjects, and (E)ffects that can be used to address different purposes: predict the outcome of an action, plan actions to achieve a goal, or recognize objects or actions. We update the colors of the model and represent manipulation skills with brown, cognition skills with pink, and perception skills with grey. 18

2.3 The different aspects of tool affordances need to be addressed in the subtype of tool use. 20

3.1 Affordance Taxonomy. The structure emerges when observing effect-based motion primitives from different frames of reference. The effects of Non-Pose-Based Tasks does not involve changes in the manipulanda's poses, which is different from Pose-Based Tasks. For Pose-Based Tasks, there are infinitely possible poses changes of the manipulanda in the world frame given different start poses. However, there are only finite possibilities for Finite-Effects Tasks when the effects are considered in the manipulanda frame. For example, a screw may have very different end poses in the world frame when given different start poses. However, when referenced by itself, the effects of a screwdriving task that tightens it only have one effect, which is to move the screw towards the direction of the tip of the screw. In contrast, Infinite-Effects Tasks have infinitely many effects in both the world and the manipulanda frame. Learning different subtypes in the taxonomy requires different causal relations. For example, the tool-manipulanda contact poses of the Finite-Effects Tasks are not determined by the desired effects but by the features of the objects; one can determine how a screwdriver should contact a screw without providing the desired effects. For Infinite-Effects Tasks, the tool-manipulanda contact poses is determined by both object features and the desired effects; one should be provided with the desired location of an object before determining how a stick should contact the object to push it. 51

3.2 Star 1 illustrations. (a) depicts the four component trajectories that comprise a hypothetical demonstration of a pushing task. (b) depicts the parametrization of a contact pose using a nail-hammering task as an example. 52

3.3 Alignment procedure for a hypothetical 2D tool substitution problem. 63

3.4 Demonstration of the variety of tasks learned by the robots using source objects. Star 1 tested a robot learning a wide range of tasks, including (a) knocking, (b) stirring, (c) pushing, (d) scooping, (e) cutting, (f) writing, and (g) screw-driving. 68

3.5 The workspace of (a) UR5e, (b) Baxter, and (c) the Kuka youBot robot are similar. Two Azure Kinect RGB-D sensors are placed on the sides of the workspace. 69

3.6 Different grasping poses of the source tools (Star 1). For each task, that is, knocking, stirring, pushing, scooping, and cutting, at least three different grasping poses were tested. 71

3.7 Different grasping poses of the source tools (Star 1). For each task, that is, knocking, stirring, pushing, scooping, and cutting, at least three different grasping poses were tested. 72

3.8 Substitute objects (Star 2). For each task, that is, knocking, stirring, pushing, scooping, and cutting, three substitute tools and three substitute manipulanda were included in testing. The objects in the yellow frames were used as source objects in Star 3. 75

3.9 Results of aligning substitute objects to source objects (Star 2). The green point clouds are the source objects while the blue point clouds are the substitute objects. Manipulandum substitution for the pushing and scooping task is not geometry-dependent, but goal-dependent, and therefore, the alignment results are excluded in the figure. 76

3.10 Results of tool substitution and manipulandum substitution (Star 2). The bar graphs show the results of using the substitute objects to perform knocking, stirring, pushing, scooping, and cutting. The bars compare Star 2's (blue) performance against the baseline (gray). . . . 77

3.11 Results of tool use generalization across robot platforms (Star 3). The bar graphs include results of the UR5e (green), Baxter (yellow), and youBot (yellow) using the source tool/manipulandum combinations for knocking, stirring, pushing, scooping, and cutting. The pictures at the bottom right demonstrate different robots writing "R" with trained scale and orientation. 80

4.1 A comparison of traditional Task And Motion Planning (TAMP) (a) and the proposed task, affordance, and motion planning (TAAMP) (b). On the right, cartoon vignettes showing the questions addressed at each level of these models with an example problem. In this problem, a robot should push an object from under an immovable container. Task planning chooses the action `push`. Affordance planning determines whether the environment can afford the action at all, independent of who the actor is. Motion planning addresses how to complete the action with this specific robot. 87

4.2 Diagram of the search for action parameters in TAMP and in TAAMP for the action `push`. (Top) In an infeasible task in which a block cannot be pushed from under an immovable container, TAMP must search a high-dimensional space for a solution which includes target poses of the block `p`, grasping pose `g`, and robot configurations `q`. Affordance planning detects that none of the `p`'s are feasible in a generic sense and does not execute motion planning. (Bottom) In a constrained task where the block is located inside a tunnel, TAMP needs to consider a large space, including `p`'s that are either feasible or infeasible. Affordance planning detects the `p`'s that are feasible and performs motion planning only within feasible `p` states. 89

4.3 Eight tasks in our evaluations in simulation. In the unconstrained manipulation task, the robot should place the green block (the "celery") on the immovable blue surface (the "sink") to clean it and then place it on the immovable red surface (the "stove") to cook it. The cyan block (the "radish") is movable but not directly related to the goal. The rest of the tasks shared the same goal. In the constrained manipulation task II, the robot needs to relocate the extra cyan block. In the infeasible manipulation task, the robot cannot access the green block since it is located under a yellow immovable. In the tool-use tasks, the robot should pull the green block that is out of reach with the L-shaped tool (i.e., the unconstrained tool-use task), or push the green block under the immovable orange tunnel (i.e., the constrained tool-use task I), or push the L-shaped tool to expose the grasping part of the L-shaped tool in order to use the L-shaped tool to pull the green block (i.e., the constrained tool-use task II), or cannot complete the task when the green block is located under the yellow immovable container. 98

4.4 Eight tasks in our evaluations with a Kuka youBot arm. 99

5.1 Tool selection via causal inference. A Baxter collaborative robot queries a learned causal model of tool-assisted manipulation using perceptual information from its workspace. Information from the graph is used to select the most appropriate tool for completing its goal. 105

5.2 The causal discovery process. During the observation phase (a) the robot learns a skeleton of the causal graph observing demonstrations performed by a human. During the validation phase (b) the robot attempts to orient the edges of the graph via self-supervised experimentation. Finally, during the augmentation phase (c), the robot introduces a new node (blue) and attempts to incorporate it into its graph via further experimentation. 108
5.3 The tool set. 115
5.4 The learned structure of the SCM. 117
5.5 Tool reasoning results. a) depicts the mean distance of the center of the block to the center of the goal region for each tool. b) depicts the learning curve for a select number of tools given initial training on the hoe. c) depicts how tools were selected and used as a function of the block's position given a static goal. 118

6.1 Our taxonomy of robot-to-human handover requirements. Bottom to top: the basic, intermediate and advanced requirements. 124
6.2 Difficulty levels of task-oriented handovers. 127
6.3 Handover evaluations on five tool-use tasks. (Top) The system was first trained with how to perform the stirring, pushing, cutting, knocking, and screw-driving tasks, rather than demonstrations of handovers. (Bottom) The robot was required to generate handovers for the human receiver to perform subsequent tasks. The handovers were with different levels of difficulty. The 'N/A' either refers to that the tool cannot perform the handover at the difficulty level, or the tool is inappropriate. Each cell shows a demonstration, which shows the handover generated and how the human used the tool to perform the subsequent task. The pictures were taken from the view of the human receiver. . 136

6.4 Comparing handover configurations generated by our system and the typical handovers in previous studies. The figure includes handovers of level I (top), level II (middle), and level III (bottom), with the spoon (left) and with the screwdriver (right) in different tasks. The typical handovers always grasp the same location on a tool and orient the handle of a tool horizontally to the human receiver. In contrast, our configurations are customized to the subsequent tasks and thus require minimum in-hand tool adjustments for the human receiver. 137

List of Tables

2.1 Comparison of stereotyped tool use and flexible tool use based on Hunt et al. (2013)'s descriptions. 13

2.2 Additional skills required in multiple-manipulation tool use beyond single-manipulation tool use. 21

4.1 Results In Simulation. Each cell shows the average or the percentage of thirty trials in the given condition. The time out was set to be 600 seconds. The numbers in the run time cells followed the format of M ± SD. 101

A.1 Summary of Non-causal Tool Use Studies. In this table, we summarize the following aspects: (general learning) whether the study involves any type of learning, including aspects in general manipulation; (learning specifics) whether the study learns any aspect that is specifically for tool use, rather than general manipulation; (tasks) the tool use tasks demonstrated in this study; (dynamics) whether the study considers the dynamics while using tools; (robots) the robot that is used to demonstrated the tool use tasks or relevant aspects in this study. . . . 148

A.2 Study Summary of Causal Tool Use — Single-Manipulation Tool Use — Basic Tool Use. In this table, we summarize the following aspects: (actions) the action representations; (effects) the effect representations; (tools) the tool representations; (Actions $\leftrightarrow$ Effects) how this study learns the relation between actions and effects; (sensory input) the type of sensory input; (dynamics) whether the study considers the dynamics while using tools; (tasks) the tool use tasks demonstrated in this study; (robots) the robot that is used to demonstrated the tool use tasks or relevant aspects in this study. 150

A.3 Study Summary of Causal Tool Use — Single-Manipulation Tool Use — Transferable Tool Use. In this table, we summarize the following aspects: (actions) the action representations; (effects) the effect representations; (tools) the tool representations; (Actions $\leftrightarrow$ Effects) how this study learns the relation between actions and effects; (Tools $\leftrightarrow$ Actions) how this study learns the relation between tools and actions; (sensory input) the type of sensory input; (dynamics) whether the study considers the dynamics while using tools; (tasks) the tool use tasks demonstrated in this study; (robots) the robot that is used to demonstrated the tool use tasks or relevant aspects in this study. . 155

A.4 Study Summary of Causal Tool Use — Single-Manipulation Tool Use — Improvisatory Tool Use. In this table, we summarize the following aspects: (actions) the action representations; (effects) the effect representations; (tools) the tool representations; (Actions $\leftrightarrow$ Effects) how this study learns the relation between actions and effects; (Tools $\leftrightarrow$ Actions) how this study learns the relation between tools and actions; (Tools $\leftrightarrow$ Effects) how this study learns the relation between tools and effects; (sensory input) the type of sensory input; (dynamics) whether the study considers the dynamics while using tools; (tasks) the tool use tasks demonstrated in this study; (robots) the robot that is used to demonstrated the tool use tasks or relevant aspects in this study. . 159

A.5 Study Summary of Causal Tool Use — Multiple-manipulation Tool Use. In this table, we summarize the following aspects: (category) the sub-category of this task in multiple-manipulation tool use; (affordance) how the affordance is learned in this study; (manipulation) extra manipulation skills needed comapred with single-maniplation tool use; (cognition: reasoning) extra reasoning skills needed comapred with single-maniplation tool use; (cognition: planning) extra planning skills needed comapred with single-maniplation tool use; (tasks) the tool use tasks demonstrated in this study; (robots) the robot that is used to demonstrated the tool use tasks or relevant aspects in this study. . . . 162

Acknowledgment

I am thankful to receive tremendous support in completing my graduate studies, seeking the next position, and finishing this dissertation.

First, to my Ph.D. supervisor, Brian Scassellati (Scaz), thank you for all your support in the past six years. Thank you for helping me to become an independent researcher and a critical thinker. These are the two most important skills I learned, and I believe they will help me succeed in my future career. I would also like to thank you for giving me the freedom to explore the two most exciting areas in the world, in my opinion: animal-robot interactions and robot tool use. You also worked hard to bring abundant funding to the lab so that we can work in the utopian and never need to worry about the financial burden when doing research.

To other Yale faculty members, thank you for supporting me as a researcher. Marynel Vázquez, thank you for all your support, from technical details to career advice. My time as a graduate student will be much more challenging without your help. Laurie Santos, thank you for being my second advisor on the project on animal-robot interaction. Thank you for letting my dream come true to be able to work on this project. Thank you for your patience and your great feedback. Your caring for students inspired me to become a teacher. Dragomir Radev (Drago), thank you for supervising me as a teaching assistant and being my referee when I am on the job market. It is my pleasure to be able to work with you, who is highly efficient and reachable for students. Maria Piñango (Ping lao shi), thank you for enlightening

me on the first step in research. Though we did not get a chance to collaborate on research projects, I am very grateful that you taught me foundational skills, such as reading research papers and writing my first research essay, as an undergraduate student when you exchanged to teach at Peking University.

To my wonderful colleagues, thank you all for your daily support and guidance. Having you in my journal as a Ph.D. student was my pleasure: Elena Corina Grigore, Aditi Ramachandran, Sarah Strohkorb Sebo, Jake Brawer, Nicole Salomons, Timothy Adamson, Emmanuel Adeniran, Rebecca Ramnauth, Nicholas Georgiou, Debasmita Ghose, Kate Candon, Kayla Matheus, Ellie Mamantov, Nathan Tsoi, Sydney Thompson, Austin Narcomey, Qiping Zhang, Kate Tsui, Alessandro Roncone, Olivier Mangin, Chien-Ming Huang, Laura Boccanfuso, Marilena Mademtzi, Larissa Hall, Judi Paige, Angie Johnston, Moshe Shay Ben-Haim, Alyssa Arre, Michael Bogese, Yiyun Huang, and Ellen Stumph. Jake and Nicole, thank you for being my closest lab mates. I enjoyed the time that we encouraged and supported each other. Jake, I really appreciate that I can collaborate with you on almost all the projects together in robot tool use, especially during the difficult time of the pandemic. Alessandro Roncone, Chien-Ming Huang, and Angie Johnston, thank you for your academic support that helped me conquer the difficulties.

To other people who supported me, thank you for your advice. Paulo A. Ferreira, Paweł Gajewski, and Frank Guerin, thank you for sharing your experiences in obtaining object models. Toki Migimatsu, thank you for sharing your insight on task and motion planning. James Tierney, thank you for conducting the excellent course on academic writing. This course is one of the best writing courses I have ever attended. You summarized common mistakes and taught us how to identify and correct them. I improved my writing with your tips and communicated my ideas more efficiently with researchers in the field.

To all the students that I have supervised, thank you for all your contributions to

my work and your valuable feedback for me as a mentor: Maxim Baranov, Abigail Waugh, Malak Khan, Catherine de Lacoste-Azizi, Carolyne Newman, Owen Marks, Valerie Chen, Caleb Kim, Skylar Regan, Chavely Calleja, Emani Brown, Maisa Crispino, Nicholas Peters, Zo ë Stublarec, Ryan McGough, William Zhu, and David Eduardo Villarreal.

To whom influenced me most before I started graduate school, thank you for your support. Arnold Rosenbloom, you are one of the best instructors I have ever had in computer science. Both I and Albert were your students, and we learned a lot from your inspiring lectures. I would also like to thank you for sharing your experiences as a teaching stream faculty member, and for your advice on selecting job offers. Jacqueline Brunning, I got so much help from you that I cannot elaborate on all of them. It is so unfortunate that I will never be able to say goodbye to you. However, I understand that you prefer not to hold commemorations. As a memorial to you in a different form, I utilized what you taught me about logic and conducted work in this dissertation which involves reasoning about robot tool use. I will never forget your kindness to students. The best way to memorialize you is to pass on your kindness to my future students, and I believe you will agree with me.

To my parents, my parents-in-law,

and my soulmate, my closest friend, my partner, Albert Wong,

THANK YOU!

I LOVE YOU!!

Chapter 1

Introduction

Many robots are designed to interact with objects in the environment. Recent advances grant robots the ability to perform various tasks ranging from everyday tasks, such as swiping a card (Sukhoy et al., 2012), to professional tasks that require high precision, such as robot surgery (Sarikaya et al., 2017).

Among these tasks, robot tool use is gaining increasing attention. Being able to use human tools such as screwdrivers and scissors can greatly expand the applicability of a robot. Household robots will be able to assist humans better by performing a wider range of tasks with everyday tools; robots in chemistry labs will be able to run more experiments by leveraging the lab tools; manufactory robots will be able to complete more tasks by utilizing construction tools without the need for specialized grippers.

Based on the definitions of tool use in animals, we define tool use as "a robot expands its body schema by attaching or securing an external, unanimated, attached or freely available object in order to achieve a goal of altering the state of other environment objects, updating its own state, or other goals, through purposeful manipulations." (For why we define tool use this way, see chapter 2) In this dissertation, a *tool* refers to the object attached to a robot. A *manipulandum* refers to the object

being manipulated by the tool. An *object* is an umbrella term to include both tools and manipulanda.

Robot tool use requires three skills. The first skill is perception. A robot should identify and localize tools and manipulanda from the environment. For example, to drive a slotted screw, the robot needs to align the slotted screwdriver with the screw. Inaccurate pose perception of the screw will lead to misalignment, resulting in the failure of the tool use action. To successfully drive a screw, position knowledge alone is insufficient. In the above example, the tip of the screwdriver should both be at the position of the top of the screw, and oriented in a way that the flat tip of the screwdriver is aligned with the slot of the screw. Though challenging, the perception requirement is not unique to tool use. General robot manipulation also requires similar perceptual capabilities (Li et al., 2022).

The second skill of robot tool use is manipulation. Manipulation skills focus on how to realize the required kinematics and dynamics of tool use actions. The actions include two components as defined in Qin et al. (2021) and demonstrated in Figure 1.1: the contact poses and the course of the action. The contact poses include tool-manipulandum contact poses and gripper-tool contact poses (i.e., grasping). These poses consider both the translational (e.g., the tip of the pen should contact a point on the paper) and the rotational (e.g., the pen should contact the paper close to perpendicular to the plane of the paper, rather than parallel to it) relations of the tool and the manipulanda, or the tools and the gripper. Manipulation also encompasses both the tool trajectory (i.e., a time series of poses of a tool) and dynamics (i.e., the forces required for successful tool use). Though the manipulation skills required in tool use tasks may share similarity with general manipulation tasks, tool use additionally requires that a robot should update its body schema when a tool is held (Stoytchev, 2003). For example, general manipulation tasks consider exerting certain forces at the end-effector, while tool use tasks concern how the forces be generated at the tool

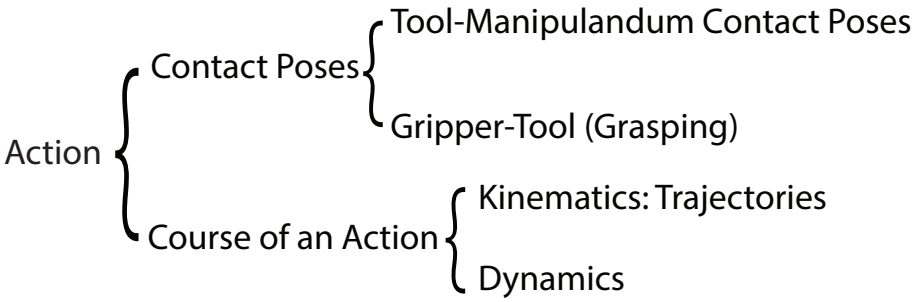


Figure 1.1: Components of an action. The diagram is adapted from Qin et al. (2021), CC BY.

rather than at the end-effector.

The third skill of robot tool use is cognition. This includes reasoning and planning tool use actions given the tasks and available tools. For example, a robot may need to reason about how to grasp the tool to facilitate tool use, and determine the tool-manipulanda contact pose, the trajectory, and the force needed to use tools successfully. The robot also reasons about using a novel tool when learned tools are unavailable. Moreover, the robot should plan to use multiple tools to achieve a goal. Neurological evidence also supports that different cognitive processes were involved when a human uses a tool compared with separate hand and tool actions (Cabrera-Álvarez and Clayton, 2020).

The unique skills required by tool use distinguish tool use from general manipulation tasks (for a review on robot manipulation, see Mason, 2018 and Kroemer et al., 2021). Given the many challenges in robot tool use, this dissertation focuses on cognitive challenges. Specifically, we investigate how to learn different types of tool use tasks in a relatively uniform manner, how to perform planning when a chain of tools needs to be used, and how tool use skills can benefit other sub-areas in robotics such as human–robot collaboration (HRC). In order to gain tool use skills, we focus on learning the interactions between the tools and manipulanda, including tool-manipulanda contact poses as well as the actions of the tools. With learned tool use skills, we explore how to expedite the planning of multiple tools by leveraging

and improving the-state-of-art task and motion planning algorithm (TAMP) (Garrett et al., 2020). After investigating a robot using tools by itself, we examine how tool use skills can facilitate collaborative tool-use tasks when a robot hands over a tool to a human to perform a tool use task.

This dissertation begins with a review of studies in tool use in Chapter 2. We first define robot tool use with insights from animal tool use research. We also devise a taxonomy of tool use tasks, and identify the required skills of each sub-category. We then organize existing tool use literature based on the taxonomy. We present key findings that focus on one type of tool use as well as relevant lines of research in general robot manipulation that share similar techniques. We conclude this chapter by highlighting the open challenges in robot tool use.

The description of our work starts in Chapter 3. In this chapter, we present the TRansferrIng Skilled Tool use Acquired Rapidly (TRI-STAR) framework. It is an integrated, task-general system that can learn a wide range of tool use tasks with minimal training samples. Moreover, it can generalize learned skills to novel tools and manipulanda to perform. Furthermore, our representation allows skills to transfer to other robots without additional training. These three advantages form the three "*stars*" of our system.

Chapter 4 examines the problem of planning a sequence of tool uses in order to complete a final goal. In this chapter, we present the Task, Affordance, And Motion Planning (TAAMP) model, which adds an extra intermediate layer of affordance planning to the classic algorithm of task and motion planning (TAMP). Affordance planning detects what actions can be performed upon the manipulanda given the current context. Therefore, it functions as a filter that restricts the search space of motion planning. TAAMP can benefit both general manipulation tasks as well as tool use tasks in identifying infeasible tasks and seeking solutions under constrained environments.

In Chapter 5, we discuss how a robot should perform tool selection. In this chapter, a robot learns tool use by observing human demonstrations, and generates a tool use model that is represented with directed acyclic graphs. The robot then validates the model by performing experiments with the tools. It may even augment the model by attempting actions that have not been observed. The robot then is presented with multiple tools, and it chooses the most appropriate tool to complete the task by considering both whether a tool can enact required actions in this particular context and how well the tool can afford the action in general.

In Chapter 6, we illustrate the importance of tool use knowledge in applications beyond tool use. We applied the TRI-STAR framework in a human-robot collaboration task to allow a robot to handover a tool to a human receiver to perform subsequent tool use tasks. With tool use knowledge, a robot can generate task-oriented handovers so that the human receiver requires minimum in-hand manipulations after receiving the tools. Previous approaches to handovers mostly learned handovers from demonstrations of hand overs. As a result, the learned handovers may be challenging to generalize to scenarios where the tools need to be used in a novel way. In contrast, our system can generate handovers based on the context.

Chapter 7 discuss our work presented in the previous chapters, including the contributions of our work, as well as the limitations and future work.

This dissertation makes the following contributions to robot tool use:

1. We recognize the unique challenges in robot tool use tasks compared with traditional manipulation tasks.

2. We define robot tool use, develop a taxonomy of robot tool use tasks, and identify required skills of subtypes.

3. We developmen the TRI-STAR framework that learns different types of tool-use tasks uniformly and efficiently.

4. We expansion of classic TAMP to perform tool use tasks that require the use of multiple tools in sequence.

5. We recognize the importance of learning tool use skills and present an application of tool use knowledge in collaborative tool use tasks.

Chapter 2

A Definition, a Taxonomy and a Review of Robot Tool Use[1]

In this chapter, we define robot tool use and introduce a taxonomy of tool use tasks with insights from animal tool use literature. Next, we enumerate the skills required for each type of tool use task. Finally, we provide a comprehensive review of robot tool use literature organized based on the taxonomy. We focus on (1) the unique challenges of tool use tasks compared with general manipulation tasks and (2) current advancements in robot tool use. We conclude this chapter by identifying the open challenges remaining to be solved in robot tool use.

2.1 Definition of Robot Tool Use

Tool use was initially considered a unique behavior only shown in humans (Oakley, 1944). Though still rare, researchers observed tool use behaviors in animals from invertebrates to non-human primates. We focus on the implications of these animal studies for robot tool use, rather than on the implications for animal cognition.

[1]Portions of this chapter are under review: M. Qin, J. Brawer, and B. Scassellati. A Survey of Robot Tool Use.

Therefore, the review of animal tool use studies is not meant to be comprehensive.

While tool use is an intuitive and commonplace part of everyday human affairs, researchers have had difficulty reaching a consensus on a precise definition of tool use. Van Lawick-Goodall (1970, p. 195) defined tool use as "the use of an external object as a functional extension of mouth or beak, hand or claw, in the attainment of an immediate goal," emphasizing the goal-oriented and functional character of tool-use. Alcock (1972, p. 464) revised the definition by specifying the kind of objects that can be used as tools and identifying the scope of the goals: "Tool-using involves the manipulation of an inanimate object, not internally manufactured, with the effect of improving the animal's efficiency in altering the form or position of some separate object."

Beck (1980) identified a number of shortcomings with these definitions. First, only objects that are portable and manipulable should be considered as tools. Under this definition, dropping a stone on an egg would be considered an example of tool use, but pounding a fruit on a tree would not be. The latter case is considered proto-tool-use (Parker and Gibson, 1977). Second, an agent should understand the connection between the goal and the tool. Otherwise, the conditioned behavior of a rat pressing a lever in a Skinner box would be considered tool use, and Beck considered this inappropriate. Third, the tool need not be externally manufactured to the agent using it nor inanimate. Researchers observed that captive apes threw feces toward human intruders, and a chimpanzee utilized the dead body of a colobus monkey to hit a conspecific, suggesting that a live ape could be utilized in a similar way. Beck argued that these behaviors should be considered tool use. Fourth, the goal of tool use can be extended beyond feeding or drinking to other goals such as self-maintenance. As a result, Beck (1980, p. 10) re-defined tool use as "the external employment of an unattached environmental object to alter more efficiently the form, position, or condition of another object, another organism, or the user itself, when the user holds

or carries the tool during or just prior to use and is responsible for the proper and effective orientation of the tool."

For decades, Beck's definition has been accepted widely in the field of animal cognition and was even adopted in early robot tool use studies (e.g., Stoytchev, 2007). Two observations motivated St. Amant and Horton (2008) to propose a new definition of tool use. Krützen et al. (2005) reported that dolphins hold marine sponges in their rostrum in order to prevent potential injuries when probing for food. Breuer et al. (2005) observed that a wild gorilla tested the depth of water with a stick while it walked across a pond. These two behaviors fall outside of Beck's definition of tool use since they do not involve altering the state of another object. St. Amant and Horton were also concerned about Beck's definition that it over-emphasized peripheral aspects of tool use, such as the unattached property; an animal can use a stick that is still attached to a tree as a tool. Moreover, they argued that Beck's definition is vague to determine whether a goal was achieved accidentally. They observed that purposeful behaviors require a continuum of control. Therefore, St. Amant and Horton (2008, p. 1203) re-defined tool use as "the exertion of control over a freely manipulable external object (the tool) with the goal of (1) altering the physical properties of another object, substance, surface or medium (the target, which may be the tool user or another organism) via a dynamic mechanical interaction, or (2) mediating the flow of information between the tool user and the environment or other organisms in the environment." They elaborated that the interactions between tools and manipulanda should be dynamic. Under this definition, stacking boxes to reach bananas is not tool use since the interactions between boxes remains fixed once they have been stacked, while cracking a nut with rock is tool use because the interactions between the nut and the rock is constantly changing.

Influenced by St. Amant and Horton's argument, Shumaker and Walkup joined Beck to revise the Beck's widely accepted definition: "the external employment of an

unattached or manipulable attached environmental object to alter more efficiently the form, position, or condition of another object, another organism, or the user itself, when the user holds and directly manipulates the tool during or prior to use and is responsible for the proper and effective orientation of the tool." (Shumaker et al., 2011, p. 36) We based our definition of robot tool use on this revised definition.

Other definitions of tool use in animal studies exist. Some may simply be shorter versions of these definitions (e.g., Chevalier-Skolnikoff, 1989; Matsuzawa, 1999). Others may disagree with the scope of tool use. For example, Asano (1994) did not restrict the tools to be something being held. This might result in the scope of tool use being overly broad since any behavior may eventually count as tool use, such as walking, which utilizes the ground as the "tool". Lestel and Grundmann (1999) expands the scope of tool use even more by including abstract concepts such as culture as potential tools. These discussions may be too philosophical and lack operational details for robotics research.

We identify three essential points in these definitions. First, tool use must have a goal, despite a lack of consensus regarding a goal's scope. Second, instead of achieving a goal through random exploration, an agent utilizing a tool should understand the connection between the goal and the behavior. Third, the tool should satisfy specific physical criteria, such as being freely manipulable. Based on these points, we define *robot tool use* as:

> **A robot attaches or secures to its end-effector an external, unanimated, freely available object or an object attached to another object, in order to achieve a goal of altering the state of another object, updating its own state, or other goals, through purposeful manipulations.**

Our definition adopts Shumaker et al.'s definition with minor modifications. First, we restrict the tools to be externally manufactured and unanimated. Unlike living

creatures, a robot typically does not produce materials (e.g., feces, spider webs) from its body. We require the tools to be unanimated because an animated object that a robot would most likely manipulate is another robot. We consider this a better fit to the area of multi-agent systems rather than tool use since it involves synchronization and communication between robots. Second, we relax the interactions between the tool and object to be manipulated to be dynamic or static. Therefore, using a container to relocate other objects would count as tool use. Third, we relax the goal of tool use. The scope of the goals in animal tool use was summarized based on animal behavioral observations. Given that tool use in animals is structurally simple even in non-human primates (Fragaszy and Eshchar, 2017), the goals in the above definitions are restricted to altering the state of another object and updating one's knowledge about the environment. In contrast, as robots often utilize human tools, robot tool use is motivated by the same goals for which these tools were designed, goals that can far exceed in scope and complexity those observed in animal studies. On the contrary, robots are required to utilize human tools. The design of human tools is more complex than those used by animals so that human tools may serve purposes beyond the goals identified in animal studies. Therefore, we prefer not to restrict the scope of the goal of robot tool use.

2.2 A Taxonomy of Robot Tool Use

In this section, we overview taxonomies proposed in animal and robotics studies, and present a taxonomy on robot tool use.

Alcock (1972) proposed a dichotomy of tool use in animals: *stereotyped tool use* is seen mostly in invertebrates and fish and *flexible tool use* is typically seen in birds and mammals. Hunt et al. (2013) considered this dichotomy an accurate description of two fundamental types of tool use with different underlying processes, despite being

oversimplified. Stereotyped tool use is inherited and animals only utilize tools in default ways in particular contexts. Examples of stereotyped tool use include antlions throwing sand to capture prey (Alcock, 1972). The species of antlions developed this behavior from the pre-existing non-tool use behavior of random sand flicking in order to maintain their pits. The tool use behavior of antlions throwing sand evolves as a phenotypic change in this species. As a result, these behaviors are widespread across the species of antlions and rarely vary within and across individuals.

In contrast, flexible tool use, which is also referred to as creative tool use, is learning-based that animals explicitly reasoning about the usages based on the context. It is this type of tool use that some believe signals intelligence (Call, 2013) and interests researchers in animal cognition. Chimpanzees' cracking nuts with rocks and fishing termites with sticks are examples of flexible tool use (Biro et al., 2003; Lonsdorf, 2006), as a juvenile chimpanzee acquires such skills by observing its parent(s). Therefore, the learning happens at the level of the individual, rather than at the level of genus. Indeed, each instance of nut cracking or fishing termites can be differentiated even within the same context by the same chimpanzee. Unlike stereotyped tool use, flexible tool use does not share context-dependency and thus can occur across different contexts. We summarized the differences between stereotyped tool use and flexible tool use suggested by Hunt et al. in Table 2.1.

Call (2013) further identified different types of flexible tool use from the perspective of problem-solving in terms of creativity and adaptivity:

- Solving novel problems with old solutions;
- Solving old problems with novel solutions;
- Solving novel problems with novel solutions.

Solutions may include utilizing one tool, selecting a tool from available options, manufacturing a novel tool, or using multiple tools sequentially.

	Stereotyped Tool Use	**Flexible Tool Use**
Distribution	genus level	individual level
Development	phenotypic changes stemed from pre-existing non-tool use behaviors	observational learning
Variability	almost no variations within and between individuals	very different within and between individuals

Table 2.1: Comparison of stereotyped tool use and flexible tool use based on Hunt et al. (2013)'s descriptions.

In contrast to the above taxonomies, Wimpenny et al. (2009) categorized tool use based on the number of tools involved in a problem but overlooked the complexity of the decision process. Boesch (2013) categorized tool use based on four levels of increasing complexity though in some sense reminiscent of Wimpenny et al.'s approach:

- *Simple tool use*: Using one tool, e.g., a chimpanzee uses a twig for fishing termites (Goodall, 1964). The animal only needs to understand the connection between itself and the reward via the tool, which is a first-order problem (Visalberghi and Fragaszy, 2006);

- *Combined tool use*: Using two tools simultaneously, e.g., a capuchin monkey uses a rock to pound a nut on a hard surface (Spagnoletti et al., 2011). The animal needs to consider both spatial relationships concurrently to connect itself with the reward, which is a second-order problem (Visalberghi and Fragaszy, 2006);

- *Sequential tool use*: Using multiple tools one after another, including using a tool for manufacturing another tool, e.g., a chimpanzee using multiple tools in sequence to break a bee hive, open honey chambers, and extract the honey. This behavior not only requires the animal to keep in mind multiple causal

relationships sequentially and choose the correct sequence, but also imposes temporal delay for the reward;

- *Composite tool use*: Combining multiple tools to use as one tool, a tool use behavior yet to be discovered in animals and currently unique to humans.

While the above taxonomies are based on animal studies, Tee et al. (2018, 2022) proposed a categorization based on default usages of tools, and identified three types of tools: category-I tools that "help to amplify/augment certain kinematic or dynamic aspects of functions that are already in an agents repertoire." (p. 6439), category-II tools that are similar to category-I tools but "require actions different from what the agent would have performed, without the tool, to achieve these functions." (p. 6439), and category-III "provide new functions that a human cannot perform without a tool." (p. 6440) As an example, they categorized a vacuum cleaner as a Category-III tool because a robot cannot perform a cleaning task without this tool. However, a vacuum cleaner can be used as a rake to reach objects or as a hammer to hit objects in other contexts. In these contexts, the vacuum cleaner should be classified as category-I tools. Given that this categorization does not consider contexts of tool use, it will be challenging for a system following this categorization to perform flexible tool use, which is context-based.

Based on the taxonomies of animal tool use and the characteristics of robot tool use, we devise a taxonomy as shown in Figure 2.1. We categorize robot tool use into *non-causal tool use* and *causal tool use*, which are similar to stereotyped tool use and flexible tool use in animals, respectively. We changed the terminology for two reasons. Frist, we would like to emphasize the fundamental differences between the two types of tool use behaviors in robots regarding whether robots should understand required causal relations, which are elaborated in Section 2.3. Second, though we consider it necessary for a robot to understand required causal relations in order to achieve behaviors similar to flexible tool use, there is a lack of evidence showing the mechanism

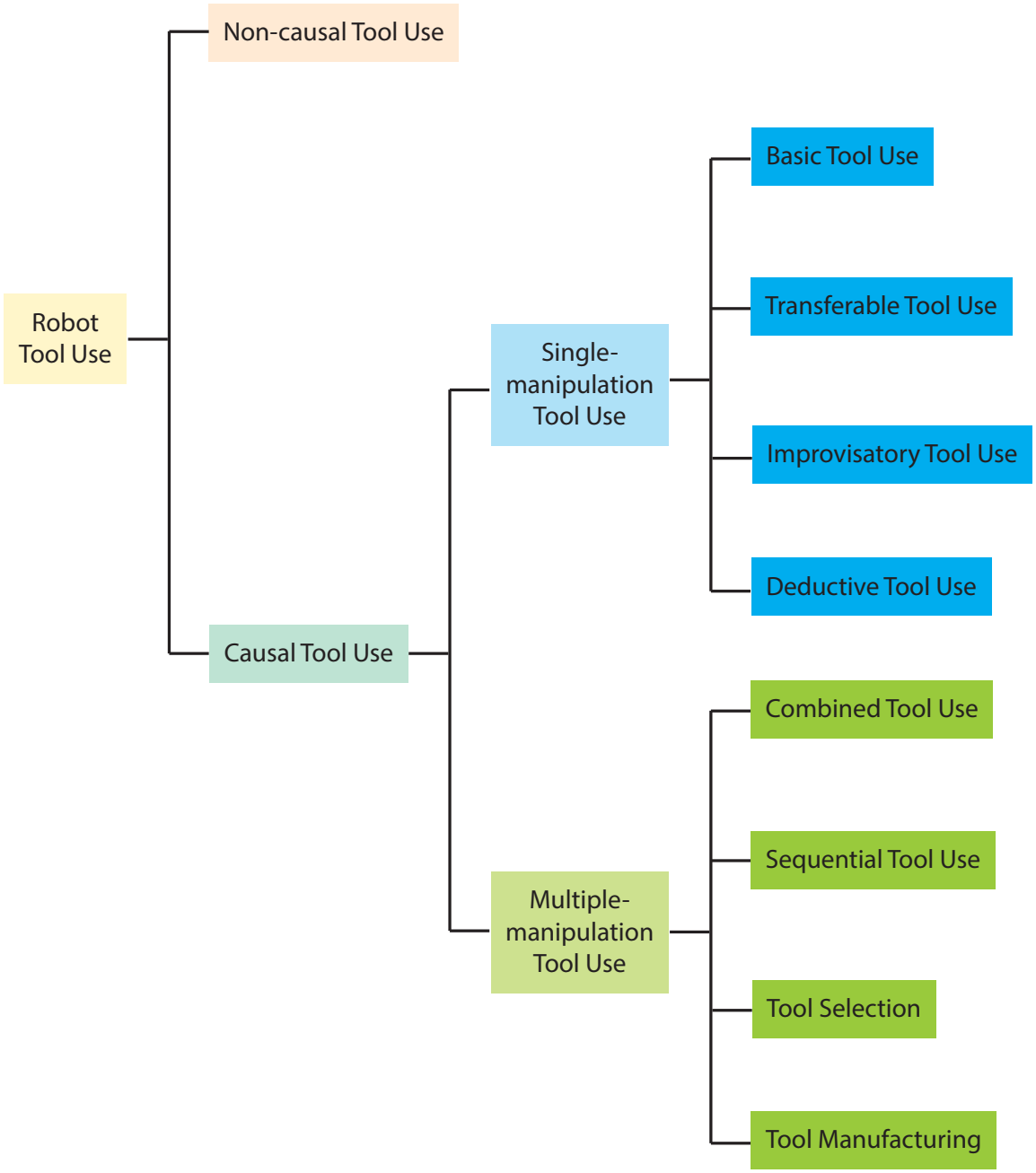


Figure 2.1: Tool Use Taxonomy.

of flexible tool use in animals. Therefore, we would like to avoid claiming that flexible tool use in animals is causal-based, and such discussion is beyond the scope of this dissertation.

We further categorize causal tool use into *single-manipulation tool use* and *multiple-manipulation tool use* based on Boesch's taxonomy. A single manipulation refers to

being presented with a single tool and using the tool to perform one action (e.g., pushing, scooping) in order to achieve one goal, though a robot may observe the usage of multiple tools to learn a task. Multiple manipulations may involve one or any combinations of multiple tools, multiple actions, and goals consisting of multiple sub-goals.

Inspired by Call's taxonomy, we categorize single-manipulation tool use into *basic tool use*, *transferable tool use*, *improvisatory tool use*, and *deductive tool use*. Basic tool use is the most basic form of tool use. In basic tool use, a robot uses a learned tool to solve a learned task, such as pushing a block, striking a xylophone, and cutting a cake. Unlike non-causal tool use that exclusively focuses on the actions, basic tool use focuses on the causal relations between actions and effects. Transferable tool use is a more complicated form of tool use, which aims to transfer learned tool use skills to other intra-category objects that share common form factors (e.g., using mugs of different shapes to pour liquid into different containers). Improvisatory tool use adds further complexity by generalizing learned tool use skills to novel inter-category objects. These objects are generally not designed to perform these tasks, such as using the handle of a screwdriver in place of a hammer to drive a nail. Deductive tool use concerns the problem of using a novel tool to solve a novel task. A robot will not be provided any prior knowledge about the tool or the task, nor given opportunities to learn about them from demonstrations. Instead, the robot should deduct the usage of a tool from its physical knowledge about the world.

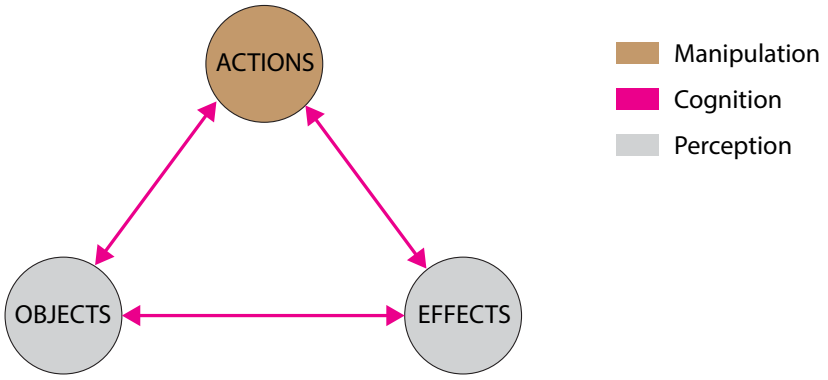
We categorize multiple-manipulation tool use into *combined tool use*, *sequential tool use*, *tool selection*, and *tool manufacturing*. Combined tool use and sequential tool use are similar to the definitions as in Boesch's taxonomy, though sequential tool use does not include constructing a new tool in our definition as it requires more sophisticated manipulation skills. Tool selection refers to the process of choosing the most appropriate tool among many options in order to complete a tool use task. Shu-

maker et al. (2011) defined tool manufacturing as "simply any structural modification of an object or an existing tool so that the object serves, or serves more effectively, as a tool." This definition only includes modifying an existing object. We combine this definition with composite tool use in Boesch's taxonomy, and re-define tool manufacturing as the process of modifying or combining objects or existing tools, with or without the usage of other tools, to complete a tool use task, or to complete the task more efficiently. Different from Shumaker et al. and Boesch's definition, our definition also explicitly includes the possibility of utilizing other tools in the process of manufacturing.

We do not enforce a subdivision of multiple-manipulation tool use by difficulty, unlike a comparable category in Boesch's taxonomy. Boesch was able to rank the categories in animal tool use because the types of tools leveraged by non-human animals are comparatively limited, and the manipulation skills in these animals are usually relatively simple. In robot tool use, the difficulty is dependent on the actual problem to solve, rather than the category that the problem belongs. For example, utilizing two tools in sequence may be simpler than creating a new tool that requires sophisticated manipulation skills. However, a problem that requires planning to use ten tools may be more challenging than a problem that requires a robot to combine two parts as a new tool.

2.3 Required Skills of Robot Tool Use

Our definition of tool use has three important components: objects, goals or desired effects, and manipulations or actions to achieve the goals. These three components agree with the three ingredients of the affordance model defined by Montesano et al. (2008). The affordance model attempts to provide an operational definition of the concept of affordances, whose precise definition is still debatable (for a review, see



inputs	outputs	function
(*O*, *A*)	*E*	Predict effect
(*O*, *E*)	*A*	Action recognition & planning
(*A*, *E*)	*O*	Object recognition & selection

Figure 2.2: The modified affordance model in Montesano et al. (2008) ©2008, IEEE (The IEEE does not require individuals working on a thesis to obtain a formal reuse license). Affordances as relations between (A)ctions, (O)bjects, and (E)ffects that can be used to address different purposes: predict the outcome of an action, plan actions to achieve a goal, or recognize objects or actions. We update the colors of the model and represent manipulation skills with brown, cognition skills with pink, and perception skills with grey.

Jamone et al., 2016 and Zech et al., 2017). The concept was first introduced by Gibson (1979, p. 127) as what the environment "offers the animal, what it provides or furnishes, either for good or ill". Despite the lack of consensus around its definition, the concept of affordances has facilitated much robotic research (Stoytchev, 2005b; Cakmak et al., 2007; Ruiz and Mayol-Cuevas, 2018; Lueddecke et al., 2019; Katz et al., 2014; Moldovan et al., 2012, 2013)

The affordance model by Montesano et al. formulated affordance as three-way relations between objects, actions, and effects. Figure 2.2 shows our modified version of this affordance model with coloring. The coloring captures the three skills required for robot tool use as described in Chapter 1. While generating actions requires manipulation skills, perceiving the effects and the objects demands perception skills. Understanding the connections between the nodes in the model needs cognition

skills. The key to robot tool use is to understand tool affordances.

Each subtype of tool use in our taxonomy addresses different aspects of affordances and requires different skills, as shown in Figure 2.3. Non-causal tool use focuses on generating desired motions, which correspond to the action node in the affordance model. While the manipulation skills are similar to the ones in general manipulation tasks, tool use tasks require additional manipulation skills, such as updating a robot's body schema when a tool is attached to its gripper.

Causal tool use involves learning and applying tool affordances, which focuses on cognition skills. Single-manipulation tool use learns and reasons about affordances. Among the different types of single-manipulation tool use, basic tool use learns how to achieve desired effects with actions. With learned relations between actions and effects, transferable tool use learns the relations between tools and actions in order to adjust actions based on novel tools that share similar form factors with the learned tools. In addition to these two relations, improvisatory tool use requires a robot to understand what specific tool features cause the effects so that it can generalize learned skills to inter-category objects. As a result, improvisatory tool use requires a robot to learn the entire affordance model. These tool use tasks generalize tool use to novel objects by learning and inducing affordance from observations. In contrast, deductive tool use requires a robot to complete a tool use task without prior knowledge using unlearned tools. As a result, the robot has no information to induce affordance and should perform deductive reasoning from general physical rules.

Multiple-manipulation tool use applies rather than learns the affordances. In addition, tool manufacturing requires more sophisticated manipulation skills; sequential tool use and tool selection require higher cognition skills; tool manufacturing requires a high level of manipulation and cognition skills. We summarized the additional skills required in each sub-type in Figure 2.2, and we will elaborate more in Section 2.4.3.

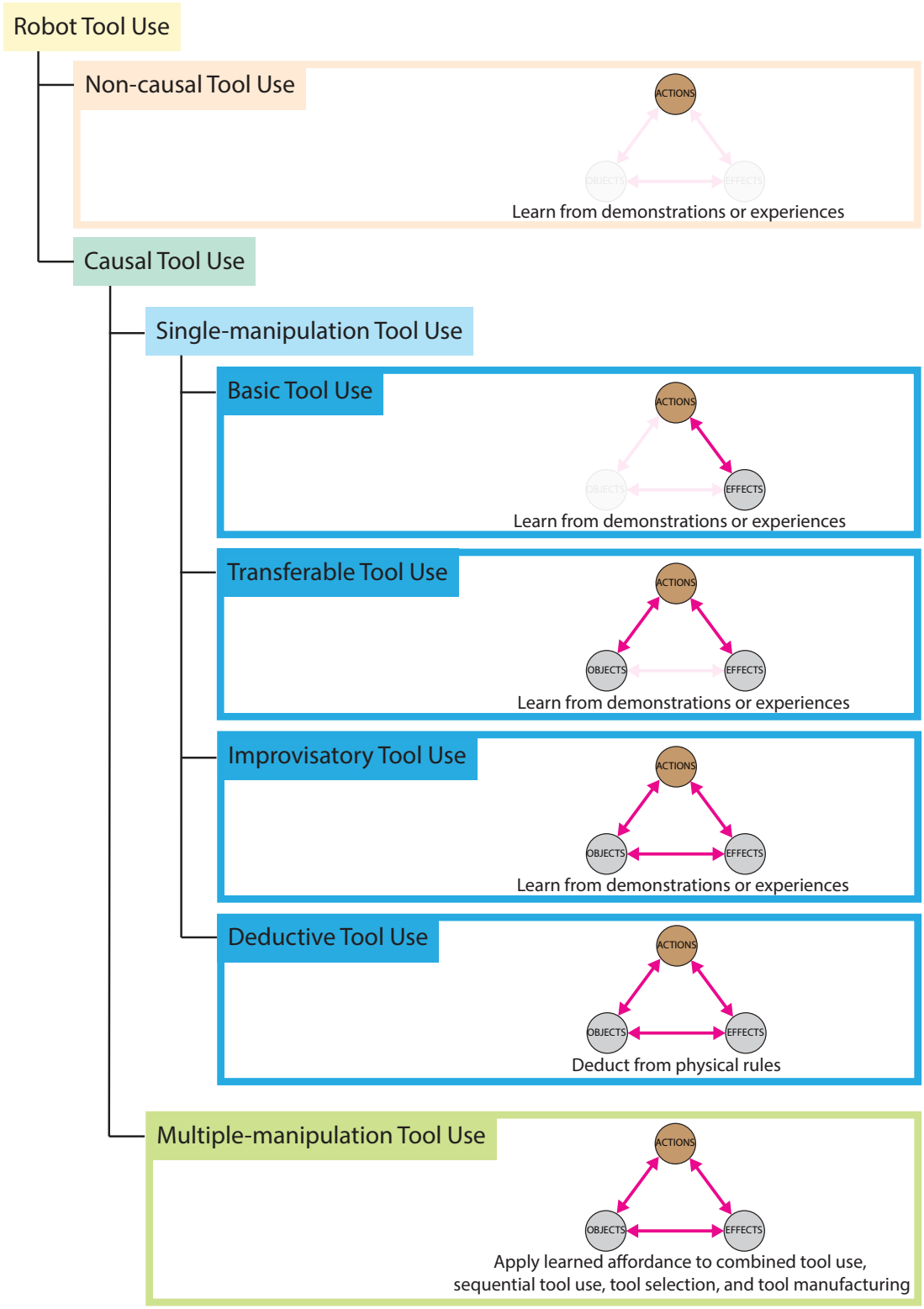


Figure 2.3: The different aspects of tool affordances need to be addressed in the subtype of tool use.

	Additional Manipulation Skills	Additional Cognition Skills	
		Reasoning	Planning
Combined Tool Use	Coordinate the use of multiple tools	Choose appropriate parameters for each tool use	
Sequential Tool Use			Plan the order of using multiple tools
Tool Selection		Choose appropriate tools among many available tools	
Tool Manufacturing	Perform the actions of assembling different parts together as a tool, or modify the current tool	Choose appropriate parts to be assembled and decide where to attach each part, or decide the desired state of the unmodified tool	Planning the order of the manufacturing, which may include sequential tool use

Table 2.2: Additional skills required in multiple-manipulation tool use beyond single-manipulation tool use.

2.4 Robot Tool Use Literature

In this section, we organize robot tool use literature based on our taxonomy. As different levels of assumptions can be made, we would like to point out that techniques that focus on the unique challenge in a higher-level tool use may not necessarily allow a robot to handle the challenges that belong to lower-level tool use. In each section, we focus on the techniques that can solve the unique challenge at each level of tool use. We would also like to emphasize that the purpose of robot tool use is not to mimic or model how animals use tools, but rather to allow robots to use tools.

2.4.1 Non-Causal Tool Use

Non-causal tool use is the ability to use learned tools to solve learned tasks, without understanding the cause-and-effect relationship between the actions and the goals. The purpose of non-causal tool use is to duplicate or reproduce actions with limited variations. It could be achieved by programming a wide variety of tool use actions such as nut fastening in aircraft production that requires high precision (Pfeiffer et al., 2017), stub grinding and deburring with force control (Robertsson et al., 2006), handwriting that involves multi-contact manipulation (Kim et al., 2014), furniture polishing that uses an impedance model (Nagata et al., 2001), generating a collision-free polishing path (Takeuchi et al., 1993), accurately drawing a circle with a compass that involves complex contacts (Kutsuzawa et al., 2017), unfastening screws in collaborative tasks (Li et al., 2020), and pouring based on the volume of liquid (Rozo et al., 2013), or actions relevant to tool use such as grasping a knife resting on a cutting board that requires a high level of dexterity (Xue and Jia, 2020), or segmenting a surgical tool from the background while using it (Su et al., 2018; Garcia-Peraza-Herrera et al., 2017). The purpose of these approaches is to automate one process to facilitate human work. Therefore, the implementations are designed to be highly specific to the task.

However, given the wide range of tasks, it is impractical to program all tool use tasks. Being able to learn these tasks is desired. One approach is to treat tool use tasks the same way as general manipulation tasks and learn the actions accordingly. One of the classic algorithms of learning actions is dynamic movement primitives (DMP) (Schaal, 2006; Ijspeert et al., 2013). DMP leverages the concept of attractors from dynamical systems, and actions are represented as a set of linear differential equations. A more intuitive approach to understanding DMP is to visualize the equations as vector fields, where a trajectory is formed by following the vectors from a starting point to an end point. Each dimension may need to be learned separately and then

coupled together. One advantage of DMP is that the shape of the trajectory can be distorted based on the starting point and the end point. DMP and its variations have been demonstrated with tool use tasks such as swinging a tennis racket (Schaal, 2006; Ijspeert et al., 2002), playing table tennis (Muelling et al., 2010), playing ball-in-a-cup (Kober et al., 2008), pouring liquid (Pastor et al., 2009), and whiteboard cleaning (Kormushev et al., 2011). Algorithms other than DMP have also been employed to represent action primitives, such as probabilistic movement primitives (Paraschos et al., 2013) and Fourier movement primitives (Kulak et al., 2020). Actions were also parameterized as minimal plans to facilitate action interpretation (Guha et al., 2013). To handle tasks that require high manipulation precision such as using chopsticks, model-free imitation learning was chosen (Ke et al., 2021). For tasks that do not require high precision of force or position control, indirect force controllers (Lutscher and Cheng, 2013) or a unified algorithm for dynamic object manipulation (Tsuji et al., 2015) can be considered. While these are methods designed to learn actions, general learning methods such as deep learning (Droniou et al., 2014; Byravan and Fox, 2017) and reinforcement learning (Peters and Schaal, 2006) were also used.

While these studies focus on learning actions, others focus on segmenting continuous actions into action primitives for tool use (Lioutikov et al., 2017; Ramirez-Amaro et al., 2014a). A similar line of research on general manipulation tasks is to recognize the tasks based on the classification of actions (Hu et al., 2014; Shao et al., 2021; Ramirez-Amaro et al., 2014b, 2015; Wölfel and Henrich, 2018; Koch et al., 2022). This approach attempts to ground action profiles to labels, either primitive labels or task labels, and do not relate actions to the effects on the objects being manipulated. For example, the whiteboard swiping action will be characterized as the translational movement of the eraser in this approach, rather than the words being erased, which is the effect. While grounding action profiles to labels is useful in some applications, it does not permit causal tool use.

The above studies treated tool use tasks in the same manner as general manipulation tasks. As a result, they cannot adjust actions based on how the tools are grasped since they do not have tool-related knowledge. In order to accommodate the tools attached to the end-effector, a robot needs to update its body schema to include the tools, or in other words, to calibrate the tool in the gripper. Prior studies focus on updating robot kinematics by considering the tip of a tool (e.g., Kemp and Edsinger (2006)), which is considered the primary contact point between the tool and the environment. Among these studies, some manipulated the tool with kinematic control (Stoytchev, 2003; Nabeshima et al., 2005), and others found it necessary to perform dynamic control (Nabeshima et al., 2007; Karayiannidis et al., 2014; Hoffmann et al., 2014a; Jamone et al., 2013; Kemp and Edsinger, 2006). Despite these studies' success, considering the tool's tip only is insufficient for all tool use tasks. For example, it is insufficient to know where the tip of a mug is when it is used to pour liquid into another container. The mug needs to be tracked with multiple markers attached to it (Lee et al., 2008). Another example is joint tools such as a pair of scissors. In this scenario, a grounded relational representation of the entire tool is needed (Katz et al., 2008). Beyond tool calibration, other studies explored collision detection (Colgate et al., 1995) and obstacle avoidance (Lee and Song, 2021) with tools attached to the gripper, as well as robot motion planning to complete tool use tasks (Holladay et al., 2019; Kobayashi and Hosoe, 2009) or planning for the grasping of the tool (Raessa et al., 2019; Chen et al., 2019; Lin and Sun, 2015) in robot motion generation.

2.4.2 Causal Tool Use — Single-Manipulation Tool Use

Basic Tool Use

Though both basic tool use and non-causal tool use leverage learned tools to solve learned tasks, basic tool use can adjust actions based on the desired effects while non-

causal tool use cannot. In other words, basic tool use requires robots to understand the causal relations between actions and effects. For example, a robot performing basic tool use is able to push an object further away given a target region that is further away, while a robot performing non-causal tool use will simply attempt to duplicate learned actions and does not adjust the actions based on the target region that is further away.

Sinapov and Stoytchev (2008) conducted an early study to explore the relation between actions and effects with motion babbling. They utilized six different tools (T-stick, L-stick, straight stick, L-hook, Y-hook, and an arrow-shaped tool) to relocate a puck with six pre-defined exploratory behaviors (i.e., push, pull, slide-left, slide-right, rotate-left, and rotate-right). For each tool, the robot learned the distribution of movement trajectories of the puck relative to its starting location. Forestier and Oudeyer (2016) employed an active version of Model Babbling to explore the distribution of manipulanda after tool use with two different sticks. These studies focus on the potential distribution of the location of manipulanda, rather than the one-to-one relationship between an action and its effect. Therefore, it is challenging to utilize tools to achieve desired effects with this method.

Other studies learned the one-to-one relation of an action and its effect, though in a quantitative manner. Okada et al. (2006) focused on verifying the effects as success or failure of tool use tasks such as pouring. Pastor et al. (2011) focused on predicting whether an object has been successfully struck by a pool cue or flipped using chopsticks. Studying the relation of an action and its effect in this manner is suitable if the state of the effects is discrete, but may not fit tool use tasks whose effects are continuous such as pushing an object ten centimeters to its right.

Studies that focus on learning the one-to-one relation of an action and its effect in a qualitative way generally employed tasks that result in the relocation of manipulanda. Stoytchev (2005a, 2008) pre-defined eight pulling actions and recorded

the effects of these actions with five different tools into an affordance table. As the actions were discretized, the effects can also be categorized in discretized space. In the evaluation, a robot needed to choose appropriate actions based on the affordance table in order to pull the manipulanda into a goal region, given one of the learned tools. Though Tikhanoff et al. (2013) also leveraged pre-defined actions, they allowed the actions to be parameterized with continuous variables, e.g., a randomly sampled pushing direction. Rather than keeping an affordance table, they used Least Square Support Vector Machines to regress the actions to the effects. Elliott et al. (2016) considered more types of push and pull. They also leveraged two regression techniques: linear regression and Gaussian process regression. Other than pulling and pushing tasks, Elliott and Elliott and Cakmak (2018) explored cleaning tasks to relocate dirt. As the manipulanda are clusters of rigid bodies rather than a single rigid-body manipulandum, they represented the surface as a grid, and trained a pixel-level classifier to predict whether each pixel contains dirt after an action. The robots in the above studies explored tool use by themselves, pre-defined actions are necessary. In contrast, Liu et al. (2018) did not pre-define actions and took the method of imitation learning and learned with deep reinforcement learning.

These studies focus on pushing and pulling tasks. A common feature of these tasks is that the desired effect determines how a tool should contact a manipulandum. Other tool use tasks may permit multiple equally viable ways for a tool to make contact with a manipulandum to achieve the same effect. For example, pouring liquid from different orientations all result in the same effect of a container being filled. Claassens and Demiris (2011) conducted preliminary studies and termed such properties with affordance symmetries. Affordance symmetries are important because a robot will be able to generate different trajectories to complete a tool use task when the learned contact results in collision. However, few studies have explored this direction to our knowledge.

Transferable Tool Use

Transferable tool use describes the ability to take tool use skills trained on an object to other intra-category objects defined by a common form factor. Therefore, the key to transferable tool use is to match the unlearned objects with learned objects.

We first present studies that concern specific types of tool use tasks. Most of these focused on relocation tasks such as pulling and pushing. Mar et al. (2017) and Nishide et al. (2011) leveraged self-organized maps to extract tool features to avoid the need to pre-defining the features. Takahashi et al. (2017) learned a model with a deep neural network that incorporated both grasping information and tool functions. Vogel et al. (2017) searched for the "sweet spot" of a novel baseball bat-like object when used to hit a baseball by sensing the force at the end-effector. Other studies considered pouring tasks. Kroemer et al. (2012) used a kernel-based approach to generalize learned action skills to novel objects. Brandi et al. (2014) performed warping to the point cloud of a learned container to match a novel container. Dong et al. (2019) adjusted pouring behavior by estimating the volume of liquid in the unknown containers. While these studies attempted to transfer tool use skills to novel tools, other studies explored how to act upon novel manipulanda. Gemici and Saxena (2014) sought to transfer cutting skills to food of varying physical properties such as hardness. Elliott et al. (2017) transferred learned surface cleaning actions to different surfaces, including surfaces of different sizes. Li et al. (2018) developed the Push-Net so that the system can push novel objects for re-positioning and re-orientation.

Though these studies demonstrated promising results on specific tool use tasks, it is unknown whether these algorithms could generalize to other types of tool use tasks. Therefore, other researchers investigated algorithms that transfer learned skills more broadly and demonstrated with multiple tool use tasks. Tee et al. (2018, 2022) matched the point cloud of unseen tools to the point cloud of the end-effector and arms of the robot to obtain the usage of the tools. The kPAM/kPAM 2.0 (Gao and

Tedrake, 2021; Manuelli et al., 2019) used keypoints on the tools to represent shared global shape of the category of tools, and tool use skills were inferred from these keypoints. Stückler and Behnke (2014b); Stückler et al. (2016, 2013); Stückler and Behnke (2014a, 2015) considered the point cloud presentation of tools and performed deformable registration with different levels of resolution in order to match the overall shape of the tools.

The approach of these studies requires two steps: one to learn tool use skills as basic tool use, and one to learn the transfer process. Other studies merged the two steps and learned them in one step. Sinapov and Stoytchev (2007) incorporated the shape of the tool when learning tool use models for the pulling task. As a preliminary model, transferr was only demonstrated with tools of the same shape but different sizes. Gonçalves et al. (2014a,b) utilized a Bayesian network to learn how the actions and tool shapes influence the effects. The shape parameters include area, convexity, eccentricity, compactness, circleness, and squareness. Due to the large size of the network, it needed to be reduced to be able to train effectively. They validated their technique with pulling and pushing tasks. Dehban et al. (2016) took a similar approach but overcame the drawback of the need for a discretization of data. To be able to handle grasping, Mar et al. (2015) leveraged support vector machines to map geometric features between learned and novel tools for pulling.

There are pros and cons of these two approaches. Training everything in one step may be more convenient, but the feature space can be quite large and requires more data. Training in a modular way will make it easier to diagnose when the algorithm does not function as intended. It will also make it easier to modify or incorporate new features as the former requires the entire model to be retrained.

Improvisatory Tool Use

Improvisatory tool use describes the ability to use tools in a creative way, which involves generalizing learned tool use skills from objects designed for the tasks to inter-category objects. These objects may not share common form factors with the canonical tools. Therefore, local features of the tools that lead to the desired effects should be identified.

While transferring tool use requires a robot to infer how actions are affected by novel tools given the relation between actions and effects, improvisatory tool use requires a robot to also understand what features of the tools caused the effects, which is the relation between tools and effects. In other words, improvisatory tool use calls for the learning of the full affordance model (Montesano et al., 2007, 2008).

In order to identify local features in unlearned tools, the function of a tool needs to be detected on a per-part basis. Insights can be gained from a related line of research that explores task-oriented grasping of novel objects. These studies made efforts to detect the functional part of a tool in different tasks so that the system can generate different grasping of the same object based on the task (Myers et al., 2015; Song et al., 2010, 2011b,a; Ek et al., 2010; Madry et al., 2012; Song et al., 2015; Murali et al., 2020; Kokic et al., 2017; Detry et al., 2017). Similar to these studies, studies that focus on part detection for tool use also leveraged geometric features. Schoeler and Wörgötter (2015) segmented the tools and used graphs to represent the relations between different tools parts. Nakamura and Nagai (2010) learned the full affordance model. They provided human static demonstrations without showing the course of actions, and detected local features with the Scale Invariant Feature Transform.

Given the functions of each tool part alone, a robot cannot realize improvisatory tool use since a robot have no knowledge about how to orient a tool. The robot needs to combine the tool parts information with tool use knowledge. Due to challenges in modeling grasping, Fitzgerald et al. (2019) achieved the goal with human-guided

adaptation that gained information on how to improvise each tool from human demonstrators. To improvise tool use without the need of human demonstrations for each tool, Agostini et al. (2015) learned actions with a modified DMP and used a Repository of Objects and Attributes with Roles to detect potential usages of a tool. This method is based on matching the global shapes of tools. Though this method can perform some improvisatory tool use (e.g., utilizing a knife vertically for stirring in a way similar to using a spatula), the transfer is limited. Other studies considered both global and local features. Fang et al. (2020) and Xie et al. (2019) took 2D images as input and trained neural networks for improvisatory tool use. While these studies learned tool use skills and tool feature detection together, other attempts learned them in a modular manner. Jain and Inamura (2013) manually pre-defined local features, discretized actions for the pulling and pushing tasks, and trained a robot with a T-shaped tool. They claimed that the skills could be generalized to novel tools, though no demonstration was provided. The Keto framework (Qin et al., 2020) and the GIFT framework (Turpin et al., 2021) generated keypoints on the tools, such as grasping points and function points, based on local features. The robot then planned motion based on the keypoints. However, the keypoint approach may have difficulty on tasks where the tool contact point cannot be readily represented using only one point on the surface, such as a pencil sharpener whose contact is inside the object and the contact is more than a single point. Without using keypoints, Abelha and Guerin (2017), Gajewski et al. (2019), Abelha et al. (2016), and Guerin and Ferreira (2019) characterized the point cloud of a tool by approximating each of its segments with superquadrics and superparaboloids. They parametrized tool use with so-called p-tools, and demonstrated their technique with a wide range of tasks such as hammering and scooping in simulation or on a physical robot. While these studies utilized visual features to transfer tool use, Zhu et al. (2015) included both geometric and physical features such as mass.

Deductive Tool Use

In deductive tool use, a robot should be able to utilize a novel tool to solve a task for which it has no prior knowledge. This is a very challenging task, and no current studies can perform deductive tool use to our knowledge. This type of tool use requires a robot to infer the entire affordance model, which is the relations between actions, effects, and tools, without tool use training samples as in improvisatory tool use.

2.4.3 Causal Tool Use — Multiple-Manipulation Tool Use

Multiple-manipulation tool use involves many different types of tool use. Unlike single-manipulation tool use, the subtypes of multiple-manipulation tool use may not be interrelated. Compared with single-manipulation tool use, they may require more sophisticated manipulation skills and cognition skills such as planning, which are generally not needed in single tool use where only one tool use task is considered. In terms of tool knowledge, they usually require the full model of tool affordance knowledge.

Combined tool use

Combined tool use refers to using multiple tools simultaneously, such as using a fork and a knife to cut a steak. No prior studies have demonstrated combined tool use to our knowledge. We identify two main challenges of combined tool use. The first challenge is at the cognition level. A robot should choose the appropriate parameters for each tool use, such as where to cut with the knife and where to stab the steak with the fork. The second challenge is at the manipulation level, which is how to coordinate the actions of each tool. It involves collision-free motion planning and adjusting the actions of one tool based on the other tool. For example, the force exerted on the fork to stabilize the steak is dependent on the course of the cutting action with the knife. Though generating collision-free motion planning may share

similar techniques in multi-agent systems (for a review, see Rossi et al. (2018); Ismail et al. (2018); Rasheed et al. (2022)), how to choose appropriate parameters and how to coordinate tools are issues specific to tool use and may need to be handled differently from general manipulation tasks.

Sequential tool use

Sequential tool use involves completing multiple tool use tasks in order. Yamazaki et al. (2010) designed an integrated system of daily assistive robots and applied it to the task of tidying and cleaning rooms. This system focused on failure detection and recovery, and manually defined the sequence of tasks to be completed. For a robot to be fully autonomous, the robot should be able to arrange appropriate orders and decide appropriate task parameters for each tool use task since the end state of a task is the start state of the next task.

This requirement falls under the topic of task and motion planning (TAMP) (for a review, see Garrett et al. (2021)). As its name suggests, it integrates low-level motion planning which includes classic robotic manipulation techniques and high-level task planning which belongs to classic AI planning. Task planning aims to find an action skeleton to achieve a goal (e.g., pick up a pencil, use it to write, and put the pen down). Motion planning aims to find motion plans to execute in a robot (e.g., the joint states for each action). TAMP aims to find action parameters to connect task planning and motion planning (e.g., where to grasp the pencil to pick it up so the pencil can be used to write). TAMP currently has two main approaches to find action parameters: the sampling-based approach and the optimization-based approach. The sampling-based approach, which is used in the majority of TAMP studies, samples action parameters and tests the feasibility of the sampled combinations. Therefore, this approach may have difficulty when the solution space is relatively small since the probability of being able to sample the correct solution is small. In contrast, the optimization-

based approach used optimization techniques such as logic-geometric programming (Toussaint et al., 2018) or sequential quadratic programming (Hadfield-Menell et al., 2016). It is able to handle problems with a small solution space more efficiently if the local optima can be handled properly. However, this approach generally requires a longer running time for tasks with many objects due to the increased dimension.

Sequential tool use has been demonstrated with optimization-based TAMP. Toussaint et al. (2018) enabled a robot in simulation to reach a tool that was initially out of reach with another tool in order to grab the target object. While they can handle tasks in a static environment, Migimatsu and Bohg (2020) improved the method with an object-centric approach to adapt to situations where objects were moved by other agents. Though this study was not demonstrated with sequential tool use, it has the potential to be applied to sequential tool use. Due to the current preliminary stage of tool use research, sequential tool use has not been demonstrated with a sampling-based approach to our knowledge.

In the above optimization-based TAMP approach, sequential tool use is only included as a demonstration to validate TAMP methods. Tool use, especially sequential tool use, usually includes multiple objects, which makes it challenging for the optimization-based approach. It is also challenging for the sampling-based approach since tool use tasks generally have a smaller solution space due to the additional constraints of tools. Therefore, alternative TAMP algorithms designed for sequential tool use may be needed due to the special requirements of tool use tasks compared with general manipulation tasks.

Tool Selection

Tool selection is the ability to choose the most appropriate tool among many options. In order to select the most appropriate object to be used as a ram to keep a door open, Levihn and Stilman (2014) identified four properties of a ram. In order to learn the

properties, Wicaksono and Sammut (2016) demonstrated a robot with an instance of the pulling task, and the robot then performed experiments to generate hypotheses about what features are important. As an example of the hypotheses, the hook on the pulling tool should locate on the same side as the manipulanda. The hypotheses are expressed in Horn clauses so that the features are qualitative. To learn the features in a quantitative manner, Saito et al. (2018) learned the full model of tool affordances.

Tool Manufacturing

Tool manufacturing is the ability to complete a tool use task by constructing a tool by combining available materials, modifying a tool, or both. As this process may involve combining different pieces, the manipulation skills required may be similar to the skills in robotic assembly. The peg-in-hole task, which is to insert a peg in a hole, is a standard task in robotic assembly. Researchers have explored methods to improve a robot's performance, such as working with more complex parts with force-guided assembly (Dietrich et al., 2010) and increasing the speed of compliant manipulators (Bös et al., 2017) (For a review on robotic assembly with learning from demonstration, see Zhu and Hu, 2018). Beyond the peg-in-hole task, previous studies also considered the slide-in-the-groove assembly task (Peternel et al., 2015), and robot assembly that leveraged tool use such as hammering and wrenching (Gu et al., 2014).

Nair et al. (2019a,b) studied tool manufacturing by combining available parts. Their system was provided with examples of tool use, and selected appropriate parts as the grasping parts and function parts. The selection was made by comparing the similarity between the available parts and segmented parts of the demonstrated examples. The next step is to combine the parts selected with appropriate orientations. The system then performed tool use tasks to validate the assembled tool. Unlike robotic assembly, the manipulation skills required in these studies are relatively simple. It pre-defined three ways of attaching the different parts: pierce attachment,

grasp attachment, and magnetic attachment. Sammut et al. (2015) designed a robot engineer to perform tool manufacturing. The robot engineer first identified important features of a tool use task, and then constructed the tool using 3D printing.

Tool manufacturing is a complicated task. The task settings of current studies reduce the difficulty of both manipulation and cognition skills. At the manipulation level, a robot may need to combine different parts with simple manipulation skills or leverage an external machine. While in animal or human tool use, the manipulation skills required in tool manufacturing are sophisticated and may even require using other tools. At the cognition level, the choice of available parts discretizes the solution space compared with the task whose solution space is continuous, such as a chimpanzee needed to make a hook to retrieve food. Tool manufacturing also requires tool affordance knowledge to identify important features of a tool to be assembled and requires planning skills to arrange the manipulation actions, especially when sequential tool use is needed.

2.5 Discussions

This chapter defines robot tool use. We also provide a taxonomy of robot use and identify the required skills in each category of tool use. As a summary, non-causal tool use focus on the manipulation skills of using tools. Causal tool use focus on learning or applying affordances. The sub-categories of single-manipulation tool use learn different parts of affordances. Basic tool use learns the actions-effects relation. Transferable tool use focuses on the tools-actions relation in addition to the actions-effects relation. Improvisatory tool use requires the knowledge of the full model. Deductive tool use generates affordances with general knowledge, rather than inducting the model from experiences or demonstrations. While single tool use relies on learning affordances, multiple-manipulation tool use leverages learned affordances

and requires more sophisticated manipulation and/or higher-level cognition skills. In addition, we review literature on robot tool use. To facilitate future tool use studies, we summarized the studies in this chapter in Appendix A.

The study of tool use is still in the preliminary stages, and most studies aim to solve non-causal tool use and basic tool use. We identify the following open challenges in tool use:

1. *How can a robot learn the relations between tool-manipulanda contact poses and effects in transferable tool use?* There is a lack of studies on the relationship between tool-manipulanda contact poses and tool use effects. Most studies focus on the relation between trajectories and effects.

2. *How can an integrative system for improvisatory tool use can handle a wide range of tasks?* While it is challenging to improvise tool use based on either local or global features, it is even more challenging to develop a system that can solve a wide range of tool use tasks. Such a system should decide whether local or global features should be considered, or choose features beyond geometric ones.

3. *How can a robot perform deductive tool use?* The challenge for deductive tool use is the lack of prior experiences. Current techniques for other sub-types of single-manipulation tool use performs inductive reasoning that learns affordances from experiences, and cannot be applied to deductive tool use.

4. *How can a robot perform multiple-manipulation tool use?* Multiple-manipulation tool use requires a robot to perform single-manipulation tool use. In addition, each sub-type in multiple-manipulation tool use requires more sophisticated manipulation skills or higher level cognition skills. Moreover, the additional skills differ among the sub-types of multiple-manipulation tool use.

5. *How can a robot learn the dynamics in causal tool use?* It is already challenging for current studies to consider tasks that can be achieved with only kinematic control. It will be even more challenging to incorporate dynamics as it adds additional dimensions to consider.

6. *Can we design a benchmark database for standard tool use tasks?* It is not trivial to design standard tool use tasks with a benchmark database of object models to facilitate comparisons between different algorithms. The requirements of the tasks should be detailed enough for precise replication. However, detailed requirements may lead to algorithms tailored for these tasks, and loss of generality. Moreover, it is challenging to select representative tools for improvisatory tool use as tools are expected to be used in creative manners. It is also impossible to include all possible tools for a given task due to the almost endless choices of physical objects that can be used as tools. It is also time-consuming to obtain the 3D model of an object.

7. *When and how can tool use knowledge be applied other areas in robotics?* Most studies that are relevant to tool use ignore the affordance model. For example, when learning robot grasping, a system typically observes how a human grasps a tool, rather than inferring how a tool should be grasped based on the subsequent tasks. However, not every study involving tool use requires a robot to learn the full affordance model, and it is important to identify which part of the model should be learned. Moreover, it also requires effort to connect tool use learning module with other modules, such as robot grasping and human-robot-collaboration tasks.

In this dissertation, chapter 3 presents a framework that can perform basic tool use, transferable tool use, and improvisatory tool use to tackle open challenge 1 and part of open challenge 2. Chapter 4 and chapter 5 aim to handle part of open challenge

4. Specifically, Chapter 4 presents a method to generate solutions for sequential tool use more efficiently. Chapter 5 describes a method for tool selection based on causal reasoning. Chapter 6 applies tool use knowledge in a robot-to-human handover task and provides an example to handle open challenge 6.

Chapter 3

Learning and Reasoning About Single-Manipulation Tool Use[1]

In this chapter, we present the TRansferrIng Skilled Tool use Acquired Rapidly (TRI-STAR) framework that aims to handle open challenge 1 and part of open challenge 2 as described in Chapter 2. It is an integrated system that can perform basic tool use, transferable tool use, and improvisatory tool use based on geometric features. TRI-STAR has three primary components: 1) the ability to learn, reason, and apply tool use to a wide variety of tasks from a minimal number of training demonstrations, 2) the ability to generalize learned skills to other tools and manipulated objects, and 3) the ability to transfer learned skills to other robots. These capabilities are enabled by TRI-STAR's task-oriented approach, which identifies and leverages structural task knowledge. We demonstrate this framework with seven tasks that impose distinct requirements on the usages of the tools, six of which were each performed on three physical robots with varying kinematic configurations. Our results demonstrate that TRI-STAR can learn effective tool use from only 20 training demonstrations. In

[1]Portions of this chapter were originally published as: M. Qin*, J. Brawer*, and B. Scassellati. Rapidly Learning Generalizable and Robot-Agnostic Tool-Use Skills for a Wide Range of Tasks. In *Frontiers in Robotics and AI*, 2021. (* equal contributions) (Qin et al., 2021)

addition, our framework generalizes tool use to morphologically distinct objects, as well as transfers them to new platforms, with only minor performance degradation.

3.1 Introduction

Imagine a robot designed to perform household chores. Such a robot will encounter many tasks requiring the use of a wide variety of tools, for example, cutting and stirring ingredients to help with cooking, scooping pet food to care for family pets, and driving screws and hammering nails to assist with house maintenance. In order to be a help and not a hindrance, such a robot would need to be capable of rapidly learning a wide assortment of tasks. In addition, given the complexity of household chores and the diverse range of objects that could be encountered, a robot should be able to generalize learned skills to novel tools and manipulated objects without needing to be retrained. Finally, one might wish to leverage learned skills from other users or transfer a library of accrued skills to a new robot without retraining.

A framework that enables such capabilities would have applications that extend far beyond the household. The search-and-rescue and disaster cleanup domains, for example, could benefit from such capabilities. Since these scenarios can be highly unpredictable and resource-limited, the robot should be able to both learn tool use rapidly and substitute learned tools for alternatives. In addition, the ability to transfer learned skills to other robot platforms will enable rapid deployment of new models to assist or to replace a damaged teammate, regardless of different robot kinematic configurations.

In order to obtain these capabilities, a framework should be able to perform basic tool use, transferable tool use, and improvisatory tool use as described in Chapter 2 and learn the full affordance model. Similar to previous tool use studies, we only consider tool use tasks involving: 1) tools and manipulanda that are unjointed rigid

bodies, 2) the use of contact forces to deterministically change the state of the manipulandum, and 3) a goal that can be accomplished with a single tool action, rather than a series of actions.

We report on a task-general integrative tool use framework, called TRansferrIng Skilled Tool use Acquired Rapidly (TRI-STAR). The framework includes components such as perception, tool use learning, and tool use generalization. These components collectively endow a robot with three capabilities, or *Stars*, aimed at solving challenging and commonplace problems in robot tool use. Star 1 enables basic tool use in a *task-general* manner, that it can learn and apply a wide range of tasks with minimal training. Star 2 allows transferable tool use and improvisatory tool use. It generalizes tool use learned with *source* tools by Star 1 to both *substitute* tools and manipulanda with no additional training, which is *object substitution*. Star 3 is the ability to transfer learned skills directly to other robot platforms, which is *platform generalization*.

3.1.1 Task-Oriented Approach to Tool Use

While some tool use studies are tool-oriented in that they seek to only model the actions node of the affordance model for a specific tool or class of tools (e.g., Stoytchev, 2005a; Zech et al., 2017; Sinapov and Stoytchev, 2008), we opted for a task-oriented approach (Kokic et al., 2017; Detry et al., 2017) that learns the entire affordance model of a task. This is a more natural framing of the problem as the actions to be performed with a tool are driven by the desired effects, not by the specific tool itself. To illustrate, the actions taken with a hammer on a nail differ when one drive the nail into a board or pull the nail out with the claw. In both tasks, the tool (the hammer) and even the manipulandum (the nail) are the same, so differences in how the tool is used can only be explained by the differences in the tasks. In a tool-oriented approach, the tool would have uniquely determined a single action for

both steps.

In a task-oriented approach, effects, objects, and actions are connected through causal relationships. A desired effect causes an agent to select features of objects (e.g., the desired effect of cutting requires a tool to be sharp), and the objects and the desired effects determine a precise action to be taken (e.g., the desired position of a block determines how it should be pushed, and the size of a bowl influences the radius of a stirring motion). While these objects-effects relations, actions-effects relations, and objects-actions relations, respectively, may differ across tasks, they remain constant across instances of a particular task and are useful when learning and generalizing tool use.

Specifying these three relations for each task is impractical and learning these relations for each task can be data intensive. However, the causal structure of this approach implies that tasks with similar desired effects also share common features. Therefore, we compiled an affordance taxonomy that categorizes tasks based on their effects with respect to manipulanda and we will describe the affordance taxonomy (For details, see Section 3.2.2). The advantage of utilizing taxonomic knowledge is that information does not need to be manually specified for new tasks either when learning a task or applying the learned tool use skills. In this way, taxonomic knowledge can help to reduce the training data needed.

3.1.2 Star 1: Learning and Applying Task-General Tool Use

Star 1 is the ability to perform basic tool use, that is, to learn and apply the actions-effects relations in the affordance model using a source tool and manipulandum. This include determining the contact poses and the course of actions depending on the effects. In this section, we first describe relevant studies. They often ignore the causal relations and only models the action node, or ignore action generation. We then describe the challenges in learning basic tool use and briefly describe the tests

we conducted.

Studies focusing on modeling the actions node ignored the contact poses, though they were applied to tool use tasks such as swinging tennis rackets (Ijspeert et al., 2002), batting (Peters and Schaal, 2006), playing ball-in-a-cup (Kober et al., 2008) or table tennis (Muelling et al., 2010), pouring (Pastor et al., 2009; Rozo et al., 2013), writing letters (Lioutikov et al., 2017) or digits (Droniou et al., 2014), and peg-hole insertion (Gao and Tedrake, 2021) with methods such as dynamical movement primitives (Schaal, 2006; Ijspeert et al., 2013) or probabilistic movement primitives (Paraschos et al., 2013). In one study, experimenters hard-coded the contact poses that the end of a peg should align with the top of a hole vertically when learning the peg-hole insertion task (Gao and Tedrake, 2021).

Studies that did not ignore contact poses (Kemp and Edsinger, 2006; Hoffmann et al., 2014b) utilized the tool tip as a simplified representation of the contact area. Yet, in practice, the contact area can comprise any arbitrary area at any location on a tool, such as the tip of a screwdriver, the blade of a knife, the face of a hammer, or the concave surface of a ladle. Moreover, with such a simplification, the relation between the tool and manipulandum is reduced to be the angle of contact, which is insufficient for tasks like screw-driving; a screwdriver should contact a screw not only perpendicular to the head of the screw but also with the correct rotation about the tip axis. Additionally, such simplified representations cannot account for tasks that may have multiple viable contact poses; a hammer may approach a nail from infinitely many orientations about the head axis of the nail and thus have an infinite number of viable contact poses.

While the aforementioned studies did not incorporate the causal relations, studies that focused on these relations did not consider action generation. Sinapov and Stoytchev (2007) and Stoytchev (2008) learned how predefined linear end-effector trajectories of different tools lead to positional changes of a manipulandum. Zech

et al. (2017) attempted to learn relationships between effects and contact poses to aid in tool selection but predefined a contact pose template. Other studies (Gonçalves et al., 2014b,a; Dehban et al., 2016; Moldovan et al., 2013) learned these relations from a probabilistic approach but also with predefined end-effector trajectories.

Star 1 learns and applied the affordance model. We demonstrated seven tasks (knocking, stirring, pushing, scooping, cutting, writing, and screw-driving) that learned with a small number of training samples and tested different types of tool use. This range of tasks tested the learning and application of tool use given different task types, such as stirring, screw-driving and pushing, each corresponding to a type defined in the taxonomy we describe in detail in the methodology. When learning manipulation skills that are described in Chapter 1, we consider the minimum set of tool use that enables a robot to use a tool, which includes the tool trajectory and tool-manipulanda contact poses (henceforth referred to as contact poses).

3.1.3 Star 2: Task-General Object Substitution

Star 2 is the ability to perform transferable tool use that can complete a task, including objects that share a common geometric template (geometrically-similar objects, e.g., mugs differing in shape and size as in Brandi et al. (2014), and improvisatory tool use that share no common form-factor (geometrically-distinct objects, e.g., pushing an object with a cake-cutter rather than a toy rake). To generate actions, an object-substitution algorithm must adjust learned trajectories for tasks such as stirring in a smaller container and produce contact poses. The contact poses for many tasks can be obtained by finding the alignment between the source and substitute objects based on features for tasks such as cutting, but for some tasks like pushing the contact poses are determined by desired effects of the tasks. Similar to previous tool use studies, we focused on geometric features only.

Many previous studies employed task-specific approaches that limited the robot's

ability to transfer to tools that share common form-factors. Some of these approaches required hand-engineered information to find an alignment for each task (e.g., Brandi et al., 2014; Stückler and Behnke, 2014b; Hillenbrand and Roa, 2012). Providing hand-engineered information for each task exhibits two disadvantages. First, algorithms requiring hand-engineered information constrain their user-friendliness for naïve end-users who lack the knowledge to train these algorithms adequately. Second, engineering information for each task is time-consuming and impractical in real-world settings requiring the use of many tools.

Other approaches that can accommodate tools of various shapes usually require prohibitively large amounts of data per task. For example, over 20,000 training examples were needed to learn and generalize in the pushing task (Xie et al., 2019); 18,000 simulated tools were used to generalize tool use in a sweeping and nail-hammering task (Fang et al., 2020); 5,000 vectorized representation tools were used to train a neural network to generalize tool use in the scraping, cutting and scooping tasks (Gajewski et al., 2019; Abelha and Guerin, 2017). Acquiring a large training sample set is infeasible when tasks need to be learned rapidly or when many tasks need to be learned. Moreover, these studies only considered tool substitutions but not manipulandum substitutions limiting their applicability to many real-life tool use applications.

Star 2 performs object substitution by adjusting tool use skills learned by Star 1 using all three relations comprising taxonomic knowledge without additional training. While the actions-effects relations assisted the generation of actions to different task configurations in the same way as in Star 1, the two object-related relations help to generate contact poses and adjust learned trajectories. This ability to adapt trajectories to accommodate substitute objects, as well as the ability to perform tool and manipulandum substitution are two advantages of our approach that are not typically considered in other studies. We evaluated Star 2 with five tasks (knocking, stirring, pushing, scooping, and cutting). The substitute objects differed from the

source objects in size, shape, or a combination of both. We also tested trajectories requiring adjustments based on geometric features of the manipulanda (e.g., stirring and cutting), desired effects (e.g., pushing), and trajectories requiring no adjustments (e.g., hammering).

3.1.4 Star 3: Transferring Tool Use Skills to Other Robot Platforms

Star 3 is the ability to transfer tool use to other robot platforms. This requires a robot-independent representation of tool use. Though learning trajectories and dynamics in the joint state space is common in learning motor skills, such representations make it challenging to transfer learned skills to robots with different hardware configurations. Learning in the Cartesian space is more conducive to cross-platform transfer, though it suffers from practical limitations.

When learning in Cartesian space, prior tool use studies (e.g., Xie et al., 2019; Fitzgerald et al., 2019) used the gripper pose as a proxy for the tool pose to simplify the perception problem. In these studies, rather than learning tool-manipulandum contact poses and tool trajectories, the gripper-manipulandum relative pose and gripper trajectories were used to learn tool use. Using gripper poses assumes that the grasps of a tool remain consistent across training and testing regimes, which is difficult to ensure outside of a controlled lab setting even on the same model of robot. When such assumptions cannot be met and a robot needs to grasp a tool differently, workarounds sometimes are employed such as treating learned tools as novel (Sinapov and Stoytchev, 2008; Mar et al., 2017), which complicates the skill transfer process.

In contrast, a trajectory of a tool, rather than an end-effector, are a flexible and direct representation for the actions. Such a representation is not tied to any particular robot configuration and does not require grasping consistency within or across platforms. This flexibility enables a robot to perform tool use with different

grasps of the same tool. Crucially, this flexibility also extends to transferring skills to other robot platforms.

Star 3 performs tool use transfer from a source robot to a substitute robot by leveraging our platform-agnostic representation of tool use skill. The strength of using such a representation is that it updates a common representational schema (i.e., Cartesian end-effector trajectories) in a simple way, but nevertheless greatly impacts the flexibility and generalizability of tool skills. The process of applying the skills is otherwise the same as in Star 1. We tested the transfer of tool use learned with a Universal Robotics UR5e arm to both a Baxter robot and a Kuka youBot robot with six tasks (knocking, stirring, pushing, scooping, cutting, and writing). These three robots have different degrees of freedom (DoF) and are kinematically distinct. UR5e has 6 DoF, and one arm of Baxter has 7 DoF, which allows the robot to pose its end-effector freely in the 3D space. YouBot without the mobile base has only 5 DoF and thus limits the robot's ability to reach arbitrary poses. Depending on initial conditions, a robot might abort execution or slightly adjust a trajectory if it cannot be fully executed.

3.2 Methods

The TRI-STAR framework focuses on learning geometrically-based tool use via Learning from Demonstration with position control[2]. We first introduce and summarize the representational schemas we use throughout the system which include the affordance taxonomy, trajectory and contact pose-based tool skills, and our 3D model and 6D pose-based object representation. Subsequently we detail the three Stars enabling the primary capabilities of our system.

[2]Source code is available at https://github.com/ScazLab/Frontiers_Robot_Tool_Use.git

3.2.1 Preliminaries

The following terms are used in this method section. Pose describes the translational and rotational state of a 3D rigid body, which is generally represented as a rigid homogeneous transformation matrix in the Lie group $SE(3)$ or exponential representation parameterized as a *screw axis* and an *angle*, which is a concept from screw motion. By the Mozzi-Chasles' theorem (Chasles, 1830), for any arbitrary pose change or transformation, an equivalent screw motion always exists. A screw axis S is a normalized twist (i.e., a spatial velocity) which is a six-dimensional vector consisting of the axis ω of the helix and a linear velocity v at the origin:

$$S = \begin{bmatrix} \omega \\ v \end{bmatrix} \in \mathbb{R}^6$$

More information about the use of exponential representations in robotics can be found in (Lynch and Park, 2017; Siciliano and Khatib, 2016). Screw axes offer a flexible and compact representation of actions. They provide a qualitative description of 3D motion by characterizing the shape of a trajectory such as straight or curved lines, circles, and helices, while the angles quantify the length that a trajectory should follow along the shape defined by the screw axis. As a result, the motion primitives are represented with screw axes in this study.

The *rame of reference*, or *reference frame*, is a coordinate system in which a pose is referenced, and the world frame is the default frame. While this is a standard definition, it is different from previous studies (e.g., Sinapov and Stoytchev, 2007), where the term is used to describe an object of interest. To illustrate the differences, when we refer to things like the "manipulandum frame," for example, we mean the coordinate system with the pose of the manipulandum at the origin, while the object of interests could be anything rather than restricted to the manipulandum as in previous studies.

3.2.2 Representations

Task Representation: Affordance Taxonomy

We developed a taxonomy of affordances to assist learning and application of tool use. Our taxonomy (Figure 3.1) recognizes two fundamental task types using desired effects referenced in the world frame. In our study, we focus on tasks with effects described by pose changes of the manipulandum as they can be easily perceived via the depth cameras. Since the goals for everyday tool use tasks generally require simple motions of the manipulanda, one screw axis can be used to characterize the shape of a effect-directed motion primitive. Non-Pose-Based Tasks are tasks with zero screw axes which represents the case where the pose of a manipulandum (e.g., a bowl) is not changed as a result of the tool usage (e.g., stirring liquid in the bowl) in the world frame. Pose-Based Tasks are tasks with non-zero screw axes such that the pose of a manipulandum changes as a result of tool use, though two further subdivisions emerge when observing in the manipulandum frame. Finite-Effects Tasks such as screw-driving are tasks where a unique screw axis in the manipulandum frame exists to describe actions-effects relations while there are still infinitely many effects in the world frame. Infinite-Effects Tasks, in contrast, like pushing a toy with a rake to the desired location, have infinitely many screw axes in the manipulandum frame to represent effects.

The taxonomy characterizes what causal relations should be considered in the subtypes. Learning tool use in Star 1 is consistent in all three types of tasks, except that Infinite-Effects Tasks require modifications based on the actions-effects relations. The contact pose required when completing a pushing task, for example, depends heavily on the goal pose of the manipulandum. Applying tool use, in contrast, requires different task-specific information depending on task type and the Star. The actions-effects relations specify how actions should be updated based on the desired effects, which

are crucial for Pose-Based Tasks because they determine, for example, the length of the trajectory when driving a screw into a thick piece of wood versus a thinner piece. The objects-effects relations specify which object features are relevant for achieving the desired effect such as a sharp edge of a tool in the case of a cutting task. These relations are important for generating contact poses for object substitution for all tasks except the Infinite-Effects Tasks. These tasks do not require manipulanda-effects relations but require actions-effects relations since the contact pose depends on, for example, where the manipulanda should be pushed to in a pushing task. The object-action relations dictate how the actions should be updated based on different object features, which are relevant for the Non-Pose-Based Tasks such as stirring where the radius of a stirring trajectory is dependent on the size of the container containing the mixture being stirred. These relations capture common features across tasks within each category of the taxonomy and can be used to guide the learning, application, and transferring of tool use to substitute objects.

Actions Representation: Trajectory and Contact Poses

A trajectory consists of four components as shown in Figure 3.2a: 1) the preparation component, which brings the tool in close proximity to the manipulandum, 2) the contact component which initiates contact with the manipulandum, 3) the functional component which acts on the manipulandum, and 4) the finishing component, which moves the tool away from the manipulandum, terminating the trajectory. The main part of the trajectory is the functional component. We represent this component using screw axis representations which are compact and easily adapted for tool use. Though we also included other components, we consider such components peripheral to the tool skill proper and thus are not the focus of this study. Keeping with other tool use studies that either completely ignore such components or hard-code them (e.g., Sukhoy et al., 2012), we represented these components simply using trajectory

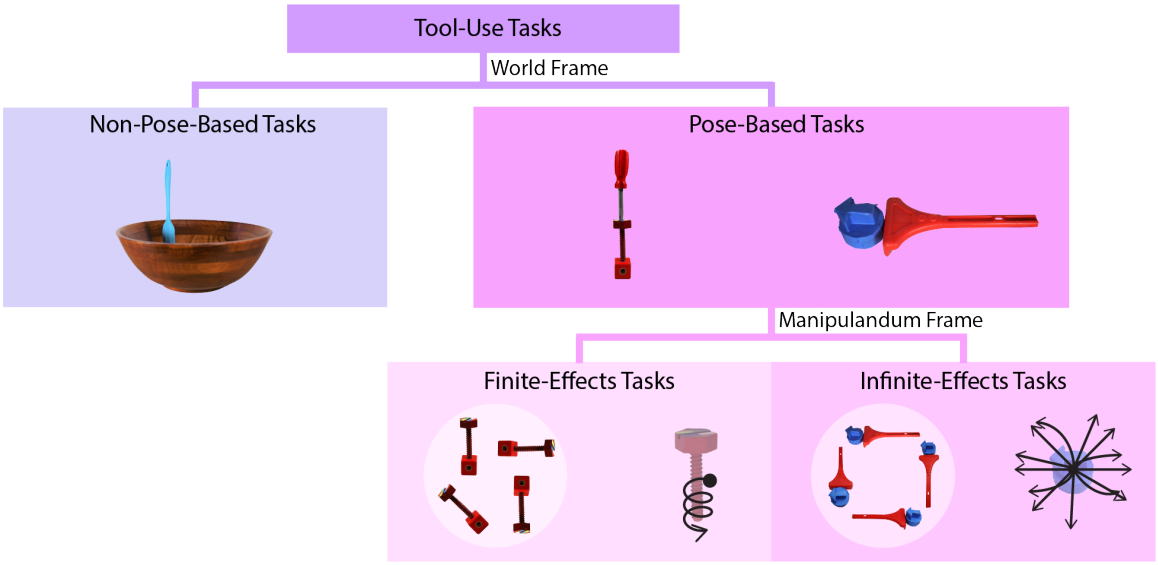
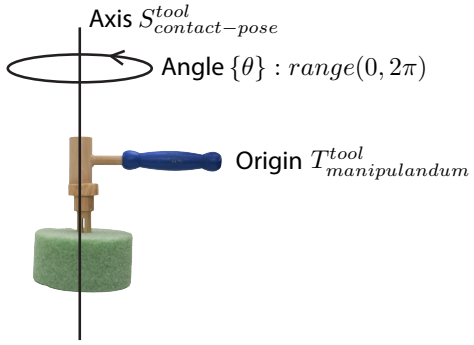


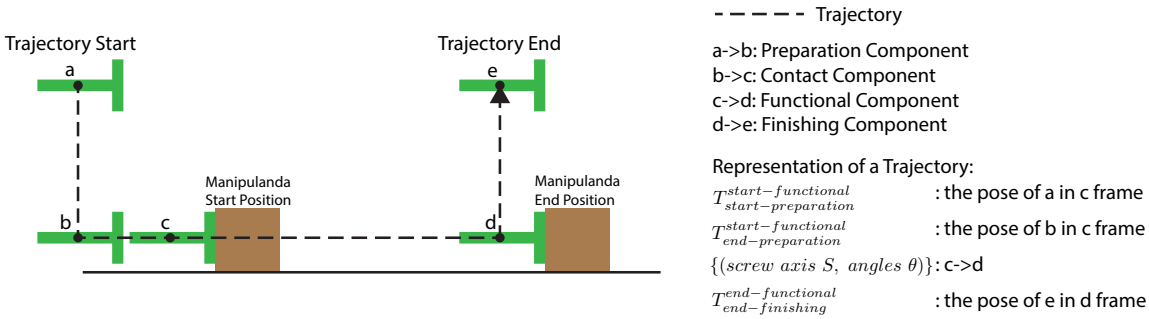
Figure 3.1: Affordance Taxonomy. The structure emerges when observing effect-based motion primitives from different frames of reference. The effects of Non-Pose-Based Tasks does not involve changes in the manipulanda's poses, which is different from Pose-Based Tasks. For Pose-Based Tasks, there are infinitely possible poses changes of the manipulanda in the world frame given different start poses. However, there are only finite possibilities for Finite-Effects Tasks when the effects are considered in the manipulanda frame. For example, a screw may have very different end poses in the world frame when given different start poses. However, when referenced by itself, the effects of a screw-driving task that tightens it only have one effect, which is to move the screw towards the direction of the tip of the screw. In contrast, Infinite-Effects Tasks have infinitely many effects in both the world and the manipulanda frame. Learning different subtypes in the taxonomy requires different causal relations. For example, the tool-manipulanda contact poses of the Finite-Effects Tasks are not determined by the desired effects but by the features of the objects; one can determine how a screwdriver should contact a screw without providing the desired effects. For Infinite-Effects Tasks, the tool-manipulanda contact poses is determined by both object features and the desired effects; one should be provided with the desired location of an object before determining how a stick should contact the object to push it.

end points.

We represent the functional components with a series of segments $\{(screw\ axis\ S,$ $angles\ \theta)\}$ with each segment parameterized with exponential representations of a pose change. The advantage of such representation is twofold. First, since the screw axis includes all six DoF, no coupling between dimensions is needed as in previous methods (Schaal, 2006; Ijspeert et al., 2013; Paraschos et al., 2013). Second, in



(a) Parametrization of a Class of Contact Poses.



(b) Components of a Trajectory and its Parametrization.

Figure 3.2: Star 1 illustrations. (a) depicts the four component trajectories that comprise a hypothetical demonstration of a pushing task. (b) depicts the parametrization of a contact pose using a nail-hammering task as an example.

accordance with other representation schemes, trajectories can also be easily rescaled and rotated. Such representation may not be ideal for other robot manipulation tasks such as pick and place where learned trajectories are flexibly warped based on different start and goal poses. However, this representation is suitable for the tool use domain where trajectories may need to be warped in a structured way based on taxonomic knowledge (e.g., to adapt a learned straight trajectory to push along a curved one required by the desired effects) or extended along the shape outlined by the screw axis such as when driving the same screw into boards of different thicknesses.

The contact poses are represented with equivalence classes of poses, $\{T^{tool}_{man}\}$, that treats all poses formed from rotating around some axis as being equivalent. This is a uniform representation for finite contact poses such as driving screws and infinite

contact poses such as nail-hammering. Each element T_{man}^{tool} is a manipulandum pose in the tool frame (i.e., the tool frame is the pose of the tool when initiating contact with the manipulanda). Such representation is able to accommodate contact areas of any shape located anywhere on a tool and manipulandum as well as represent any orientation between the two objects. The transformations in the same class can be obtained by rotating about an axis S_{cp}^{tool}. As a result, a class of contact poses (shown in Figure 3.2b) is parameterized as an axis S_{cp}^{tool}, a transformation T_{man}^{tool} as the origin, and a group of angles θ such that a viable contact pose can be obtained by rotating an angle θ about the axis S_{cp}^{tool} starting from T_{man}^{tool}. In this way, this class can represent a unique contact pose (i.e., a unique angle which is zero), limited contact poses (i.e., a limited number of angles), or an infinite number of contact poses (i.e., the angles within a range).

Object Representation: 3D Models and 6D Poses

TRI-STAR is designed for a robot to be able to utilize novel tools without prior training. In order to accomplish this the algorithm requires the robot to obtain 3D models of the novel objects under consideration. We used Microsoft Azure RGB-D cameras, which are commonly used and relatively inexpensive sensors, to obtain raw partial 3D point clouds. With the relatively low fidelity of perceived partial point clouds, available methods could not obtain full 3D models of sufficiently good quality. Therefore, it was necessary to design a pipeline to fit our needs.

This pipeline begins with first mounting an object in the robot's end-effector. The robot can then rotate its end-effector around an arbitrary axis to ensure both the back and front of the object is visible to the 3D camera. A series of raw point clouds are obtained while the robot steps through the trajectory. The background in the point clouds is then pruned to obtain the partial point clouds of the objects. Given the pose of the end-effector at each step, the partial point clouds are merged by transforming

these point clouds to the initial pose. To account for noise, we optimize the rotation axis represented as a screw axis S with the Quasi-Newton method (Dennis and Moré, 1977) by minimizing the sum of the Euclidean distances between the bounding boxes of the partial point clouds and the bounding box of the merged point cloud. As parts of the objects are occluded by the robot's own gripper, the robot obtains two such merged scans and registers them to create the final complete scan. Supplemental scans using Autodesk Recap[3] photogrammetry software was also used to obtain point clouds for objects that are challenging for the robot to grasp. Although we attempted to design the entire process to be autonomous, the grasping during scanning and tool use requires an experimenter to assist with mounting an object to the gripper.

To obtain smoothed triangle meshes, the models are post-processed automatically with a script using meshlabxml[4], a python interface to MeshLab[5], similar to a previous study (Gajewski et al., 2019). The point clouds are upsampled with Poisson-disk sampling with input 5,000, meshed with Ball-Pivoting, smoothed with Taubin smoothing, and holes are filled with the default settings. The meshes are then centralized and realigned based on their minimum bounding boxes.

We used a non-marker-based perception system and estimated the pose of the objects from raw sensor input. Two Azure devices are placed on the two sides of the workspace to capture a complete point cloud representation of the workspace. Background and foreground point clouds are retrieved from both sensors. The workspace is isolated, and the desktop is removed with random sample consensus (RANSAC; Fischler and Bolles, 1981) from these point clouds. To obtain a partial point cloud of the manipulanda, the background is subtracted from the foreground point clouds. The pose of the object in the world frame T_{man}^{world} is obtained by rigid registration between the partial point cloud and the full 3D model. The pose with a higher fitting

[3]https://www.autodesk.com/
[4]https://github.com/3DLIRIOUS/MeshLabXML
[5]https://www.meshlab.net/

score, measured by calculating the ratio of inlier point correspondences over the total number of target points, is chosen. If the scores from both sensors are similar, the averaged pose is used.

The method of obtaining tool poses in the end-effector frame is similar to the method above, except for the extra step of removing points belonging to the gripper to isolate the tool. The pose of the tool in the end-effector frame is then obtained with $T_{tool}^{ee} = (T_{ee}^{world})^{-1} \times T_{tool}^{world}$ where $\times$ is matrix multiplication, and the superscript -1 represents matrix inversion, given the perceived pose of the end-effector in the world frame T_{ee}^{world} and the perceived tool pose T_{tool}^{world}. Similar to previous tool use studies, we assume a fixed grasp for a tool once it is in the robot's end-effector.

3.2.3 Star 1: Learning and Applying Task-General Tool Use Skills

In Star 1, our framework categorizes task demonstrations using our taxonomy and leverages taxonomic knowledge of the identified category to learn tool use (i.e., the contact poses and trajectories) and generate actions with desired effects not seen in the training samples. In the following sections, we describe how tool use is learned and applied to novel task configurations. Specifically, we first detail the simulated demonstrations used to train the skills evaluated in this study. Subsequently, we discuss how demonstrations are categorized using our affordance taxonomy and how the corresponding taxonomic knowledge is leveraged to learn trajectory and contact pose representations. Next, we detail how the system utilizes new task configurations to apply learned skills by generating new trajectories and contact poses.

Learning Tool Use Skills

The input data required by our algorithm includes the start and goal pose of the manipulanda in the world frame and the tool trajectories as the keyframes in the

world frame. Twenty simulated training samples per task were provided. Training samples were obtained with kinematic teaching of keyframe demonstrations in simulation. Each sample was a single demonstration of a task using a source tool and manipulandum. The samples were assumed to be successful demonstrations of a task, as no sophisticated outlier removal methods were utilized.

With the start and goal poses of the manipulanda, the system can infer the category of task being demonstrated to be used to guide the learning of trajectories and contact poses. If the effects of all demonstrations are zero vectors, then this task is a Non-Pose-Based Task. Otherwise, it is a Pose-Based Task. If it is the latter, the effects are converted to the manipulandum frame (i.e., the manipulanda frame is the start pose of the manipulanda) and are clustered based on the Euclidean distance between ω parts and the Euclidean distance between v parts of sample screw axes. If a unique cluster is found, then this task is considered a Finite-Effects Task. Otherwise, it is an Infinite-Effects Task.

The trajectory between two adjacent keyframes in a given demonstration is assumed to be interpolated, which may or may not be linear depending on rotational differences between the two frames. The keyframes can include only the start and goal pose of segments or any arbitrary number of midpoints. The keyframes are first merged into segments automatically. The different components of the trajectory are then identified by the framework. However, each component is assumed to have the same shape across demonstrations except for the functional component. Given a demonstrated trajectory comprised of keyframes, the framework first groups the keyframes into segments with similar transformations between keyframes (i.e., the grouping stage). A component might be missing for different types of tasks, which is identified during this grouping stage. Subsequently, each segment, or partial segment, is then parameterized with the appropriate component, and represented with $T_{start-prep}^{start-func}$, $T_{end-prep}^{start-func}$, $\{(screw\ axis\ S,\ angles\ \theta)\}$, and $T_{end-fin}^{end-func}$, as illustrated in

Figure 3.2a (i.e., the parametrization stage).

The first step in the grouping stage is to identify the preparation component and the finishing component, which is to find the start pose of the preparation component $T^{world}_{start-prep}$, goal pose $T^{world}_{end-prep}$ of the preparation component (it is also the start of the contact component $T^{world}_{start-con}$), the start pose of the finishing component $T^{world}_{start-fin}$ (it is also the end of the functional component $T^{world}_{end-func}$), and end pose of the finishing component $T^{world}_{end-fin}$. To do this, the transformations between keyframes in the world frame are converted to the screw motion representation. Adjacent transformations with similar screw axes are merged. The similarity is evaluated with the Euclidean distance between ω parts and the Euclidean distance between v parts of sample screw axes. The merging is done by averaging the screw axis and summing the angles. After merging, the first segment is assumed to be the preparation component, while the last is assumed to be the finishing component. The start and end poses of these components can thus be found.

The second step in the grouping stage is to identify the other components. For Non-Pose-Based Tasks, the rest of the segments are assumed to be the functional component, and the contact component of this type of task is assumed to be a segment with no transformations. For Pose-Based Tasks, the contact poses are assumed to be unchanged once the tool contacts the manipulanda. Therefore, the start of the functional component $T^{world}_{start-func}$ (which is also the end of the contact component $T^{world}_{end-con}$) can be obtained with $T^{world}_{start-man} \times (T^{world}_{end-man})^{-1} \times T^{world}_{end-fin}$. Since the start (i.e., the end of the preparation component) and end (i.e., the start of the functional component) poses of the contact component are known, the contact component is found by interpolating these poses, which is obtained by calculating the screw axis of the transformation between the start and end pose and sampling angles with 1-degree intervals. Although the start and end pose of the functional component is known, the functional component is not a simple interpolation as it may need to follow a

certain trajectory. Therefore, the algorithm allocates the remaining segments to the functional component, after excluding the partial segment belonging to the contact component. The partial segment is found by identifying the overlap between the first proceeding segment of the preparation component and the contact component.

In the parametrization stage, the keyframes are converted to different reference frames for easy application. The start and end pose of the preparation components are converted to the frame of the start pose of the functional component, resulting in $T_{start-prep}^{start-func}$ and $T_{end-prep}^{start-func}$, respectively. The end pose of the finishing component is converted to the frame of the end pose of the functional component, which is $T_{end-fin}^{end-func}$. If multiple segments comprise the functional component, each segment is represented with screw motion and the start pose of this segment is used as the reference frame. As a result, the trajectory of a demonstration is represented using $T_{start-prep}^{start-func}$, $T_{end-prep}^{start-func}$, $\{(screw\ axis\ S,\ angles\ \theta)\}$, and $T_{end-fin}^{end-func}$.

The next step of the parametrization stage is to find a template from all the training samples. The functional components of Infinite-Effects Tasks are ignored, as they are determined by the desired effects, rather than a shared trajectory template. For the rest of the tasks, the number of the segments comprising the functional component should be the same for each task. For the minority of demonstrations that are inconsistent with the number of segments that the majority of the demonstrations are associated with, those samples are excluded. For the remaining valid training samples, each segment of the component derived from different demonstrations is averaged. The transformations $T_{start-prep}^{start-func}$, $T_{end-prep}^{start-func}$, and $T_{end-fin}^{end-func}$ are also averaged from each demonstration.

The above learns the trajectories and now we consider how to learn the contact poses. Our current algorithm assumes a single contact area on the source tool when performing the same task, which could be relaxed in future studies. The contact area of the tool and the manipulandum were determined by proximity. For Infinite-

Effects Tasks like object pushing where task success is contingent on the goals of the manipulandum, a change-of-basis of the start pose of the manipulandum is performed in order to incorporate the goal into its representation so that the contact poses are effect-based. The demonstrated contact poses are then converted to our representation of a class of contact poses using S_{cp}^{tool}, T_{man}^{tool}, and a group of $\{\theta\}$.

For Infinite-Effects Tasks, we perform a change-of-basis on the start poses of the manipulandum before calculating the contact poses in order to account for the goal-directed nature of these tasks. The x axis is chosen to be the moving direction of the manipulanda, which is the normalized v part of a screw axis representing the transformation of the manipulandum from the start to the goal in the world frame. The z axis is chosen to be the direction of standard gravity. If the x axis and z axis are parallel, an arbitrary direction is chosen ahead of time which is not parallel to the standard gravity. The y axis is obtained with the right-hand rule, which is the cross product of x and z. To ensure the perpendicularity between x and z, the z axis is recalculated with the cross product of x and y. The position of the manipulanda remained to be the perceived position.

The contact pose of each demonstration T_{man}^{tool} are obtained by $(T_{start-func}^{world})^{-1} \times T_{start-man}^{world}$ where $T_{start-func}^{world}$ is the tool pose at the start of the functional component and $T_{start-man}^{world}$ is the start pose of the manipulanda. Then the contact poses from each demonstration are converted to our representation, a class of contact poses. The axis between any two contact poses is calculated, and the poses whose axis deviate too much from the majority of axes are excluded. An arbitrary pose, generally the pose of the first demonstration, is chosen as the origin T_{man}^{tool}. The transformations between valid contact poses and this origin are calculated in the origin frame and represented using screw motion. The averaged axis S_{cp}^{tool} is used as the axis of this class. For the angles obtained, if the Kolmogorov-Smirnov test (Daniel, 1990) on the group of angles showed no significant difference from a uniform distribution, then

the range of this angle is used to represent $\{\theta\}$. Otherwise, the groups of angles are clustered using density-based spatial clustering of applications with noise (DBSCAN; Ester et al., 1996), and the mean of each cluster is included in $\{\theta\}$.

Applying Tool Use Skills

To apply the learned tool use with the source tool and manipulanda, configurations of a task should be provided, which includes the start $T^{world}_{start-man}$ and goal pose $T^{world}_{goal-man}$ of the manipulandum $T^{world}_{goal-man}$. The goal pose $T^{world}_{goal-man}$ can be provided by perception (e.g., placed at the desired location) or by the experimenter in the form of a transformation matrix. The start pose $T^{world}_{start-man}$ is always perceived. The goal is always assumed valid for the given task and could be achieved by the given tool.

To use a tool, the contact poses and tool trajectories should be found. The contact poses are generated based on learned contact poses and taxonomic knowledge. Since multiple possible contact poses T^{tool}_{man} exists for each task, multiple corresponding tool trajectories are generated. These tool trajectories are then converted into end-effector trajectories to be executed by the robot given the current perceived tool grasping pose. Trajectories are considered candidates if their functional components can be executed since the complete execution of the functional component is crucial to perform a task. The final trajectory is chosen from the candidates that minimizes the required joint changes. If none of the functional components can be executed in full, the robot simply aborts execution. Otherwise, the robot attempts to execute as many components or partial components as possible since the full execution of other component is not central to successfully complete the task.

Now we consider how to generate a trajectory. Given a contact pose T^{tool}_{man} obtained from above, which is equivalent to $T^{start-func}_{start-man}$, and the start $T^{world}_{start-man}$ and goal pose $T^{world}_{goal-man}$ of a manipulandum, the start $T^{start-man}_{start-prep}$ and end $T^{start-man}_{end-prep}$ of the preparation component in the manipulandum frame is calculated using the

learned trajectories by $(T^{start-func}_{start-man})^{-1} \times T^{start-func}_{start-prep}$ and $(T^{start-func}_{start-man})^{-1} \times T^{start-func}_{end-prep}$ (the information from the learned trajectories are labeled with an enclosed rectangle), respectively. The preparation component in the manipulandum frame is then found by finding the interpolation between its start and end pose. The contact component in the manipulandum frame is obtained using the same method, with its start pose being the end pose of the preparation component and the end pose being the start of the functional component. In terms of the functional component, each segment of $\{(screw\ axis\ S,\ angles\ \theta)\}$ is found by interpolating the learned trajectory and converting those transformations to the manipulandum frame for the Non-Pose-Based Task. For Finite-Effects Tasks, the length of the trajectory, which is the angle in the screw motion representation, is adjusted according to the goal while the learned shape described by the screw axis remains the same. For Infinite-Effects Tasks (e.g., pushing), both the shape and length are determined by the goal with the end pose of the functional component being $(T^{world}_{start-man})^{-1} \times T^{world}_{end-man} \times (T^{start-func}_{start-man})^{-1}$, and the trajectory is found by interpolating the start and end pose. The end pose of the finishing component is calculated using the learned trajectory as $T^{start-man}_{end-func} \times T^{end-func}_{end-fin}$. In the end, each pose in the trajectory $T^{start-man}_{tool}$ is converted to the world frame with $T^{world}_{start-man} \times T^{start-man}_{tool}$. In the writing task introduced in Section 3.3, when a different scale of the trajectory (e.g., write a larger or smaller "R") is requested, the angle θ in $\{(screw\ axis\ S,\ angles\ \theta)\}$ is scaled if the screw axis represents translational changes only, otherwise the v part of the S in $\{(screw\ axis\ S,\ angles\ \theta)\}$ is scaled. This works because the screw axis is in the previous pose's frame, and the v represents the velocity at the origin. To rotate the trajectory (e.g., to produce a tilted "R"), one can simply rotate $T^{world}_{start-func}$. The corresponding start and end pose of other components need to be updated accordingly.

We now shift our attention to generating contact poses. For the learned class of contact pose whose $\{\theta\}$ is composed of discrete values, the contact pose in the matrix

form corresponding to each value is calculated. If $\{\theta\}$ is a range, the contact poses are treated as discrete values by sampling angles from the range by 1-degree intervals. For Pose-Based Tasks, the contact poses are adjusted along the tool direction of motion so that a tool is guaranteed to touch the manipulandum (e.g., when pushing, an irregular object may require a slightly different relative position between the tool and manipulandum). With this method, we are able to generate collision-free contact poses.

3.2.4 Star 2: Task-General Object Substitution

Star 2 utilizes the tool use learned by Star 1 and calculates the appropriate contact poses by finding the alignment between the source and substitute object, and adjusts the tool trajectory by leveraging the relevant taxonomic knowledge identified for each category of tasks. Star 2 requires the same manual inputs as the application in Star 1, which include the start and goal pose of the manipulanda, the desired number of circles for the stirring task, and the desired scale and rotation of the written letter for the writing task, as well as the grasping pose.

Three-step alignment algorithm

For all tasks, except Infinite-Effects Tasks, contact poses are obtained by calculating the alignment between the source and substitute objects. The contact poses of Infinite-Effects Tasks depend on both the desired effects and the alignement. When the two tools are of the same type or share a generic form factor such as two different types of hammers, often considering the entire shape of both tools (i.e., their global features) produces the best results. In the case of tasks like pushing where no generic tool form-factor exists, utilizing features like the contact area (i.e., local features) of the source tool is necessary. Therefore, we designed a three-step alignment algorithm that produces alignments between a source and substitute objects using both global

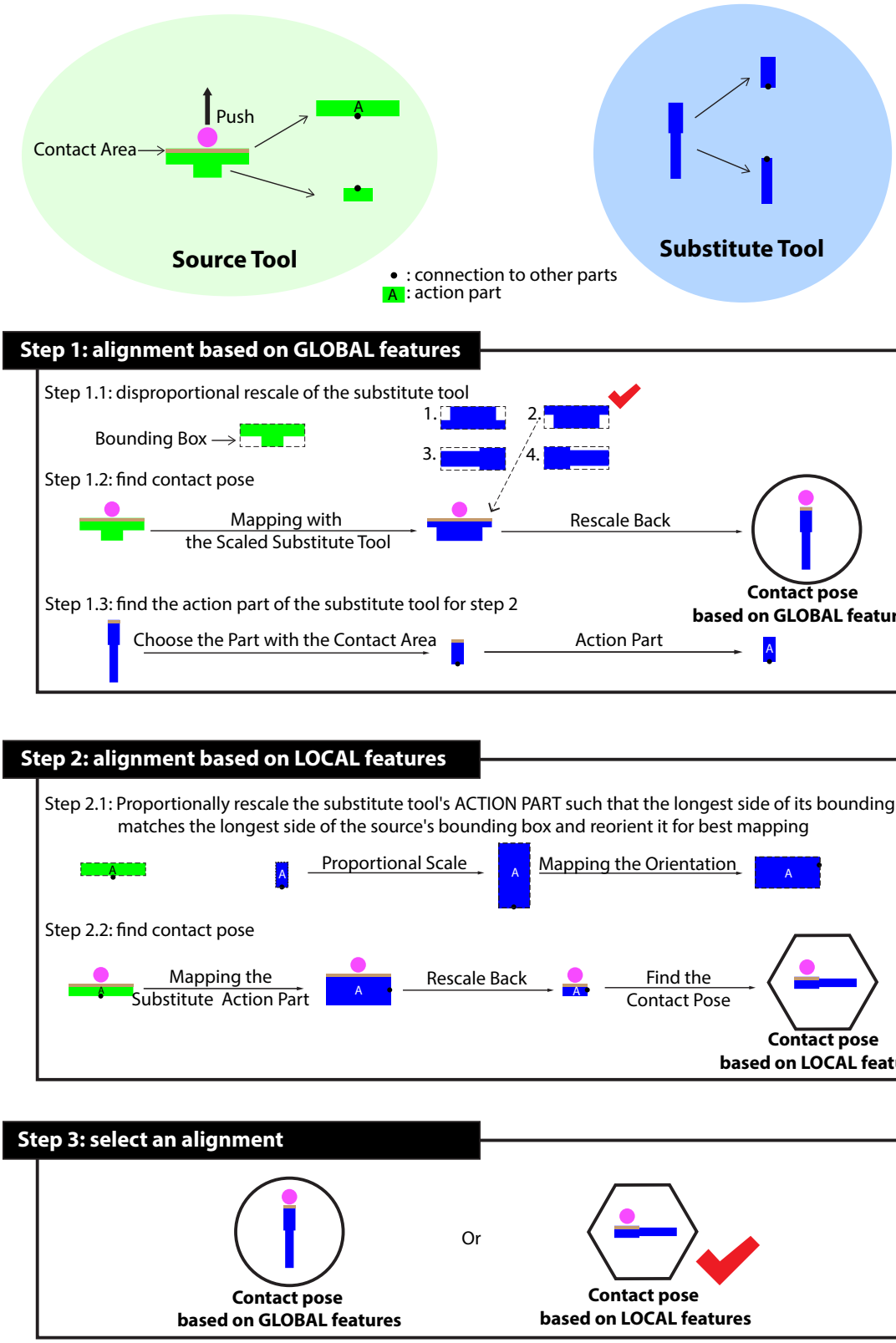


Figure 3.3: Alignment procedure for a hypothetical 2D tool substitution problem.

(step one) and local features (step two), and selects the most appropriate one (step three). Since we consider local features, object meshes need to be segmented prior to applying this algorithm. The application of the three-step alignment algorithm differs slightly for tools and manipulanda.

In order to segment a mesh, we utilized a method similar to a previous study (Abelha and Guerin, 2017) using the shape diameter function (CDF) with the CGAL library[6]. The number of clusters k were ranged from 2 to 8 with step 1, and the smoothness parameter λ ranged from 0.1 to 0.7 with step 0.1. Since no direct relation exists between the number of clusters k and the results of the segmentation, the number of clusters with the greatest number of results was chosen as k_{chosen}. Since, in most instances, the object with only one cluster is undesirable, k_{chosen} was allowed to be one only if the number of results with one cluster was significantly more than the number of clusters with the second greatest number of results. The segmentation was randomly chosen from all the segmentations with k_{chosen} clusters due to similarity.

Figure 3.3 depicts our process for finding contact poses given segmented tool models, by finding the alignment $T_{sub-tool}^{src-tool}$ between the source and substitute meshes. Th goal of the first step is to find the alignment based on global geometric features. In this step, the substitute objects are rescaled disproportionally so that their bounding boxes share the same size as the bounding box of the source objects, and reoriented along the axes of the bounding box. As an object can be rescaled and reoriented in multiple ways, the one that is most similar to the source object is chosen as the alignment based on global features. The similarity is measured by the averaged minimum Euclidean distance between the points of the two point clouds when the centers of the two objects are aligned. The contact area on the substitute object is chosen by proximity to the contact area on the source object. The segment containing the contact area is chosen to be the action part which is used in step two. If the

[6]https://doc.cgal.org/latest/Surface_mesh_segmentation/index.html

contact area is distributed across multiple segments, then the action part is chosen to be the contact area itself rather than any individual segment. As a result, we do not rely on the correctness of the segmentation. The goal of the second step is to find the alignment based on local geometric features. In order to find this alginment and the corresponding contact area, the two action parts are mapped in a similar manner except that the substitute action part is rescaled proportionally, and the alignment of the two action parts uses modified iterative closest point (ICP) registration (Rusinkiewicz and Levoy, 2001). In step three, of the two contact areas found in the two steps, the candidate with the highest similarity score is chosen along with its corresponding contact pose and the alignment of the tools $T_{sub-tool}^{src-tool}$ is thus found.

The manipulanda do not need to be decomposed into action and grasping parts like tools do. Therefore, the contact area is used as the action part, and the algorithm to find the alignment poses of the substitute manipulanda $T_{sub-man}^{src-man}$ is otherwise the same as finding the alignment of the substitute tool. For Infinite-Effects Tasks, the alignment of the manipulanda is not needed since the geometric features of the manipulanda do not decide the alignment. Therefore, it is handled in the same manner as the source manipulanda in that the start pose is updated to incorporate the desired effect. The alignment, in this case, is set to be the identity matrix.

Generating Tool Trajectories

Given the alignment resulting from the three-step alignment algorithm, the trajectory of the substitute tool can be found given the learned source tool trajectory with adjustments based on the taxonomic knowledge if necessary (see Section 3.2.3). With the obtained tool trajectory, the end-effector trajectory is calculated from the tool trajectory in the same way as Star 1, except that the functional component is rescaled based on the size of the substitute manipulandum relative to the source for Non-Pose-

Based tasks.

To find a candidate tool trajectory, an equivalent trajectory of the source tool acting upon an equivalent source manipulanda (i.e., the equivalent start pose and goal pose of the manipulandum is calculated with $T^{world}_{start-sub-man} \times (T^{src-man}_{sub-man})^{-1}$ and $T^{world}_{end-sub-man} \times (T^{src-man}_{sub-man})^{-1}$, respectively) is first found. Then each pose of the trajectory $T^{src-man}_{src-tool}$ is updated with $(T^{src-man}_{sub-man})^{-1} \times T^{src-man}_{src-tool} \times T^{src-tool}_{sub-tool}$ which calculates the trajectory of the substitute tool in the substitute manipulandum frame. The trajectory is then converted to the world frame. For Non-Pose-Based Tasks, the functional component of the trajectory is rescaled based on the relative size of the longest dimension of the source and substitute manipulandum. Multiple candidate tool trajectories are found and each corresponding to a contact pose chosen in the same way as in Star 1. The final tool trajectory is chosen from the candidate tool trajectories in the same way as in Star 1.

3.2.5 Star 3: Tool Use Transfer to Other Robot Platforms

As tool use learned by Star 1 are represented independent of robot configurations, no additional algorithms were needed in order to enable skill transfer to different platforms that could perform the given task. This was assisted via the development of a perception system that obtains the 3D poses of the tools and manipulanda from RGB-D cameras, though in principle, any method that can accurately perceive these poses can be used. With the learned tool use and the perceived grasping, we calculate the end-effector trajectories and control the robot by leveraging existing inverse kinematics and motion planning libraries. In order to simplify motion control across different robot platforms, we implemented a Robot Operating System node that uses the same interface to control all three robots. This interface can be easily extended to accommodate more platforms.

The same mechanisms of partially executing a trajectory or completely aborting it

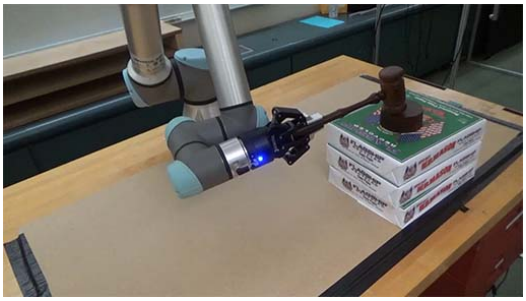
mentioned in Section 3.2.3 also apply when the platforms being transferred to cannot execute the generated actions. Moreover, learning a class of contact poses also helps with finding viable solutions on different platforms. For example, when required to drive an nail with a hammer, a robot can choose to approach a manipulandum from any orientation, even those even not appearing in the training set, which increases the viable kinematic solutions when a robot searches for motion planning.

3.3 Results

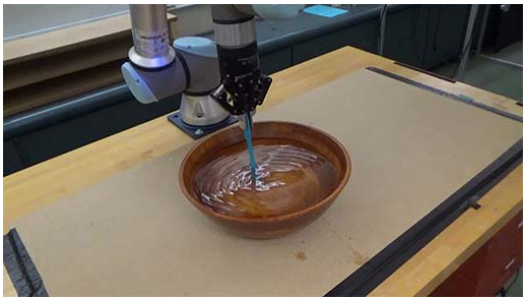
TRI-STAR uses raw sensor data for perception and demonstrated Star 1 with seven tasks trained with minimal training samples via Learning from Demonstration (Argall et al., 2009). We tested Star 2 by providing three substitute tools and three manipulanda for each task. Finally, we conducted experiments for Star 3 that transferred the learned skills to two other robot platforms with different kinematic configurations.

3.3.1 Star 1: Learning and Applying Task-General Tool Use Skills

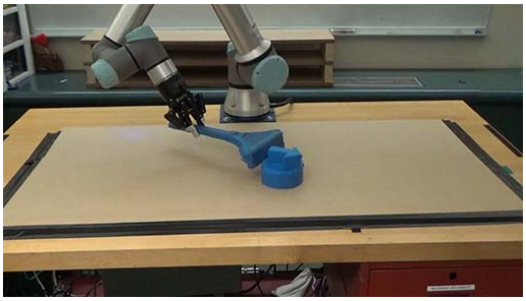
Figure 3.4 shows an example from each of the seven tasks with the source tools and manipulanda, and Figure 3.5a shows the testing environment. Six of the seven tasks were tested on a UR5e robot, and the screw-driving task was demonstrated on a simulated UR5e due to the higher perception accuracy required to align the tip of a screwdriver to the slot on the head of a screw. All tasks tested on the physical robot were evaluated quantitatively except for the writing task, which was included for demonstration purposes only. Creating quantitative metrics was sometimes challenging; while the pushing task could be evaluated with translation errors to the goal as had been done previously (Xie et al., 2019; Fitzgerald et al., 2019), other tasks were previously reported with only binary success or failure results (Pastor et al., 2009;



(a) Knocking.

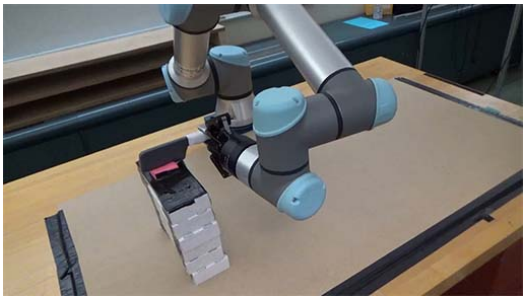


(b) Stirring.



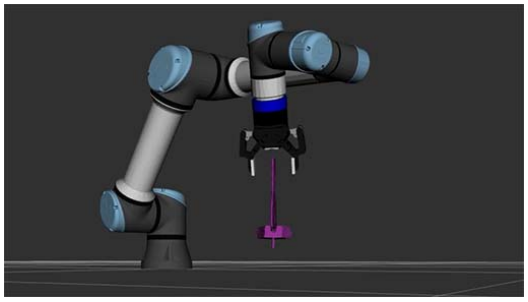
(c) Pushing.

(d) Scooping.



(e) Cutting.

(f) Writing.



(g) Screw-driving.

Figure 3.4: Demonstration of the variety of tasks learned by the robots using source objects. Star 1 tested a robot learning a wide range of tasks, including (a) knocking, (b) stirring, (c) pushing, (d) scooping, (e) cutting, (f) writing, and (g) screw-driving.

(a) The workspace of the UR5e robot.

(b) The workspace of the Baxter robot.

(c) The workspace of the Kuka youBot robot.

Figure 3.5: The workspace of (a) UR5e, (b) Baxter, and (c) the Kuka youBot robot are similar. Two Azure Kinect RGB-D sensors are placed on the sides of the workspace.

Brandi et al., 2014) or success rates over multiple trials (Fang et al., 2020; Gajewski et al., 2019). When evaluating performance quantitatively, we used stricter methods (e.g., using loudness in decibels for the knocking task) when possible.

The five tasks analyzed quantitatively were also compared with a baseline condition. We designed the baseline condition to accord with the common practice across task-general tool use learning frameworks of using the gripper pose as a proxy to the tool pose. Therefore, in the baseline condition, the robot repeated an end-effector trajectory in the task space of a training sample chosen randomly. For the five tasks, we tested ten trials per task per condition. Trials in which the robot was not able to follow the commanded trajectories were excluded. The start and goal poses of the manipulanda were altered in each trial. In both the experimental and baseline condition, the robot held tools with various poses as shown in Figure 3.6, a complexity that was not present in other studies. These poses were provided to the robot by the experimenters in order to impose pose variety (see Section 3.1.4 for motivation), though in principle TRI-STAR can accommodate autonomous grasping. Figure 3.7 summarizes the results. Details of testing each task are described below.

Knocking. A robot is required to strike an object with a hammer to produce sound. The robot successfully completed the task in 10 out of 10 trials in the testing condition, while its performance in the baseline condition was 4 out of 10 trials. We also measured the sound of each knock on the manipulandum using the Sound Meter app with a Samsung tablet placed close to the manipulandum. The average decibels, including the reading from unsuccessful trials, of the testing condition (mean (M) = 82.79 decibel (dB), Standard Deviation (SD) = 2.58 dB) was higher than the baseline condition (M = 32.00 dB, SD = 41.44 dB).

Stirring. A robot should stir the liquid with a spatula to desolve salt. 0.25 tsp salt per liter was added to the room-temperature water and given several seconds to settle. The robot was allowed to stir for one minute or five circles, whichever

Figure 3.6: Different grasping poses of the source tools (Star 1). For each task, that is, knocking, stirring, pushing, scooping, and cutting, at least three different grasping poses were tested.

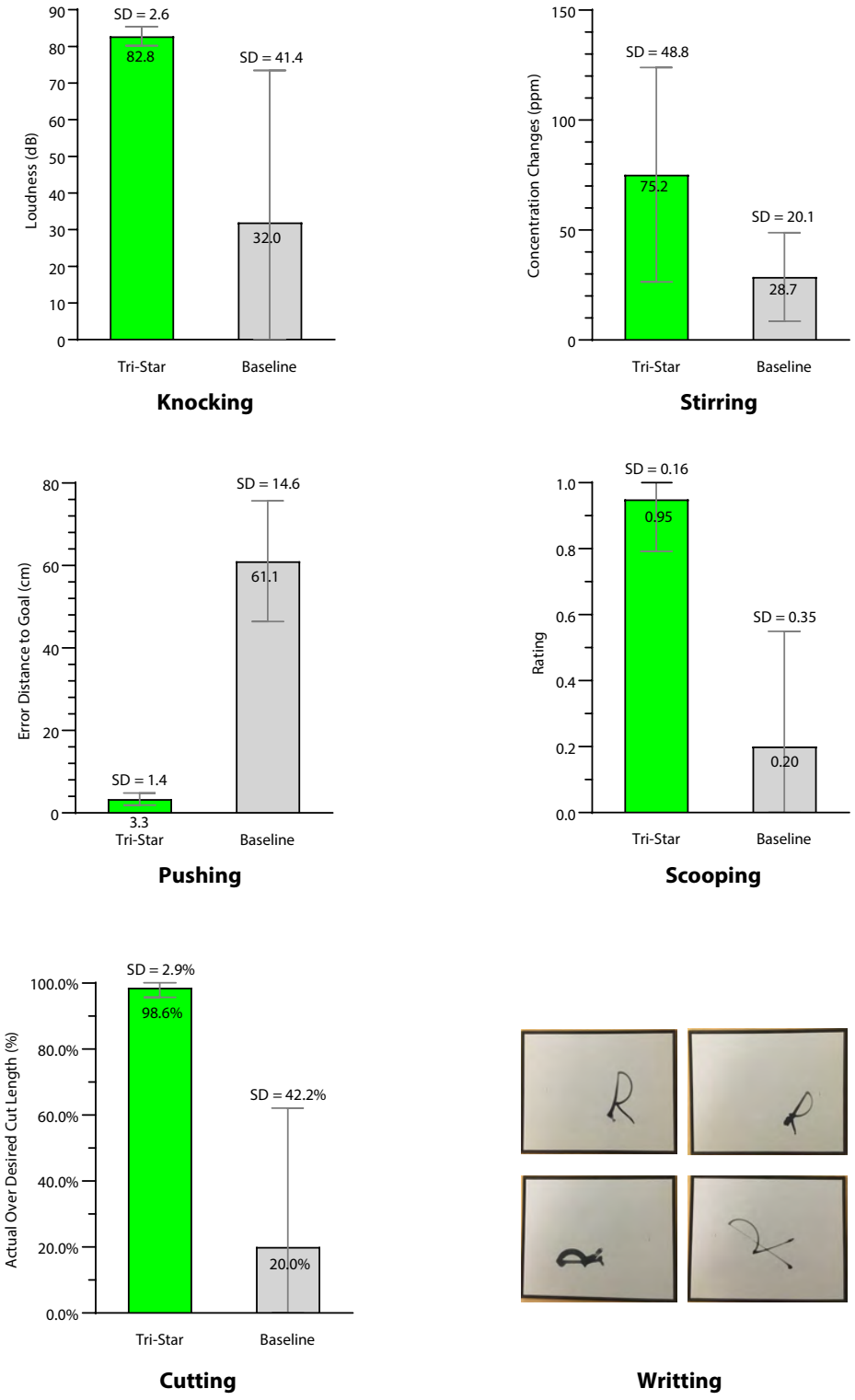


Figure 3.7: Different grasping poses of the source tools (Star 1). For each task, that is, knocking, stirring, pushing, scooping, and cutting, at least three different grasping poses were tested.

lasted longer. Due to kinematic constraints, the grasps in the testing conditions were similar to the training pose. This constraint, along with the enforced grasping pose consistency across training and baseline conditions, resulted in both training and testing conditions completing 10 of 10 trials. We also measured the concentration changes in part per million (ppm) before and after the stirring using a total dissolved solids meter. More salt dissolved in the testing condition (M = 75.20 ppm, SD = 48.79 ppm) than in the baseline condition (M = 28.70 ppm, SD = 20.10 ppm).

Pushing. A robot should push a blue item to the goal position chosen randomly by the experimenters. The manipulandum was pushed closer to the goal position in the testing condition (translation error: M = 3.36 centimeters (cm), SD = 1.45 cm) than in the baseline condition (M = 61.06 cm, SD = 14.62 cm). Our translation error in the testing condition is consistent with a recent study (Xie et al., 2019; M = 6.37 cm, SD = 5.33 cm) which also utilized perceptual data from raw sensor readings. The translation errors were mainly due to perception errors. This is supported by the significantly reduced translation error (M = 0.013 cm, SD = 0.0074 cm) observed when performing the same experiments using a simulated UR5e robot with perfect perception.

Scooping. A robot is required to scoop a rubber duck placed on top of packing peanuts. The performance was rated as 1 if the robot successfully scooped the manipulandum, 0.5 if the rubber duck slipped away but the robot scooped surrounding packing material, and 0 if the robot failed to scoop anything. The robot scooped the manipulandum more successfully in the testing condition (M = 0.95, SD = 0.16) than in the baseline condition (M = 0.20, SD = 0.35).

Cutting. A robot should cut a putty in half. We measured the percentage length of the actual cut over the length of the intended cut. Even with a relaxed criterion accepting cuts as shallow as 1mm depth in the baseline condition, the robot cut the putty more thoroughly in the testing condition (M = 98.62%, SD = 2.91%) than in

the baseline condition ($M = 20.00\%$, $SD = 42.16\%$).

Writing. A robot should write the letter "R" at the chosen location with the required scale and orientation. The required scale and orientation may or may not be included in the training samples. Figure 3.7 shows various letters "R" that the robot wrote.

Screw-driving. The robot is required to drive a screw placed at random locations and orientations in simulation, and the robot is able to complete the task successfully.

3.3.2 Star 2: Task-General Object Substitution

Five tasks (knocking, stirring, pushing, scooping, and cutting) were tested on a UR5e robot. Other than using substitute objects, the experiments and evaluation in Star 2 were the same as those performed in Star 1. For each task, three pairs of substitute objects were tested, and all objects were appropriate for the tasks. In the baseline condition, a random contact area and a contact pose were chosen on each of the substitute objects. The trajectories were generated using the same method as the testing condition. Figure 3.8 shows the source and substitute objects. Figure 3.9 shows the alignment result of each substitute object with the source object in each task. Figure 3.10 summarizes the results of the five tasks. Details of each task are described below.

Knocking. All three substitute tools successfully struck the substitute manipulanda in all trials in the testing condition, while the performance dropped significantly in the baseline condition (i.e., at most 1 out of 10 trials for each tool-manipulandum combination). In a previous study with a similar task (Fang et al., 2020), the highest success rate on nail-hammering was 86.7% of all the substitute tools with tens of thousands of training samples. In the testing condition, the average loudness in the testing condition ($M = 65.62$ dB, $SD = 3.35$ dB) was higher than that of the baseline condition ($M = 4.34$ dB, $SD = 16.50$ dB), while the loudness was not measured in

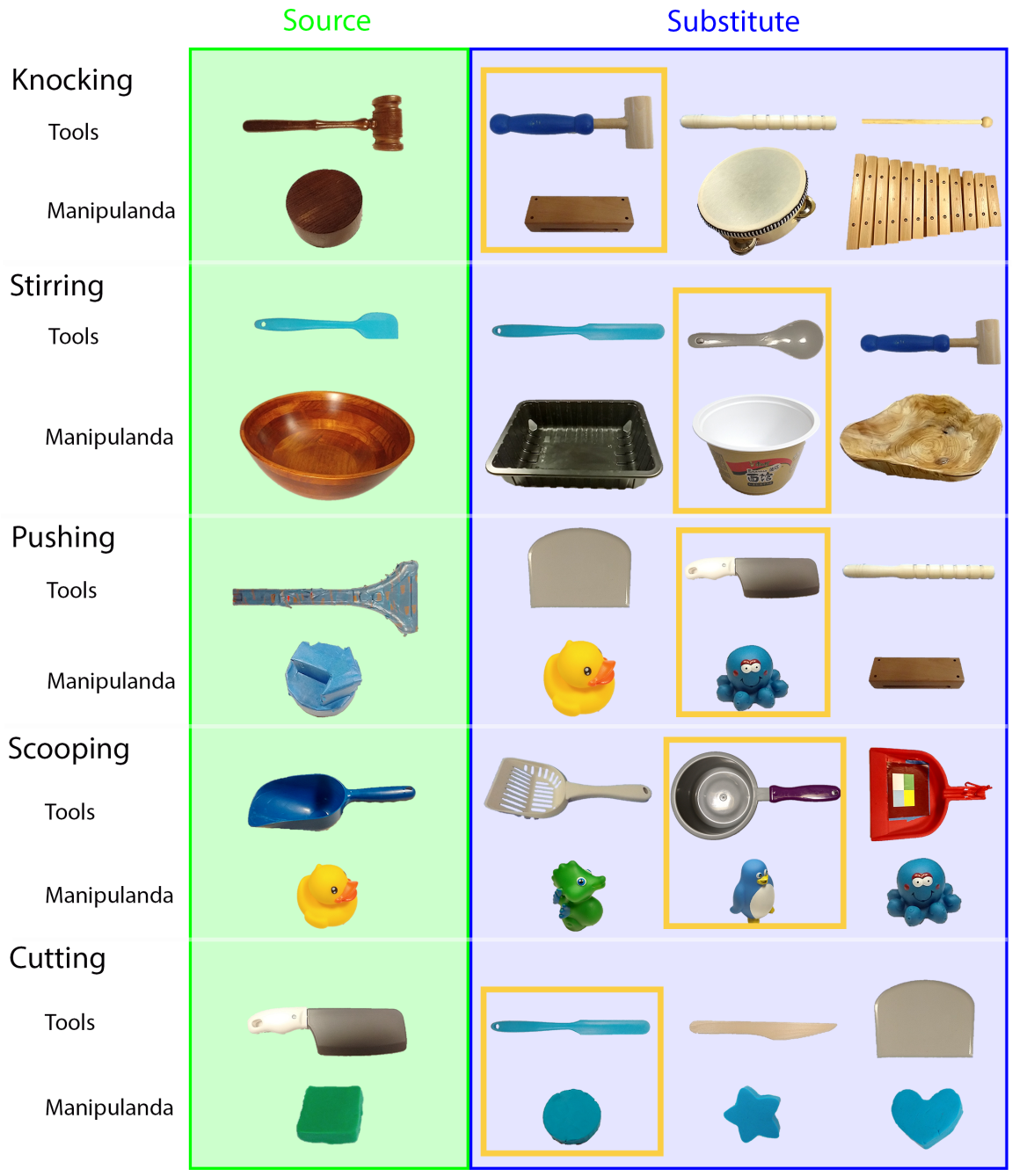


Figure 3.8: Substitute objects (Star 2). For each task, that is, knocking, stirring, pushing, scooping, and cutting, three substitute tools and three substitute manipulanda were included in testing. The objects in the yellow frames were used as source objects in Star 3.

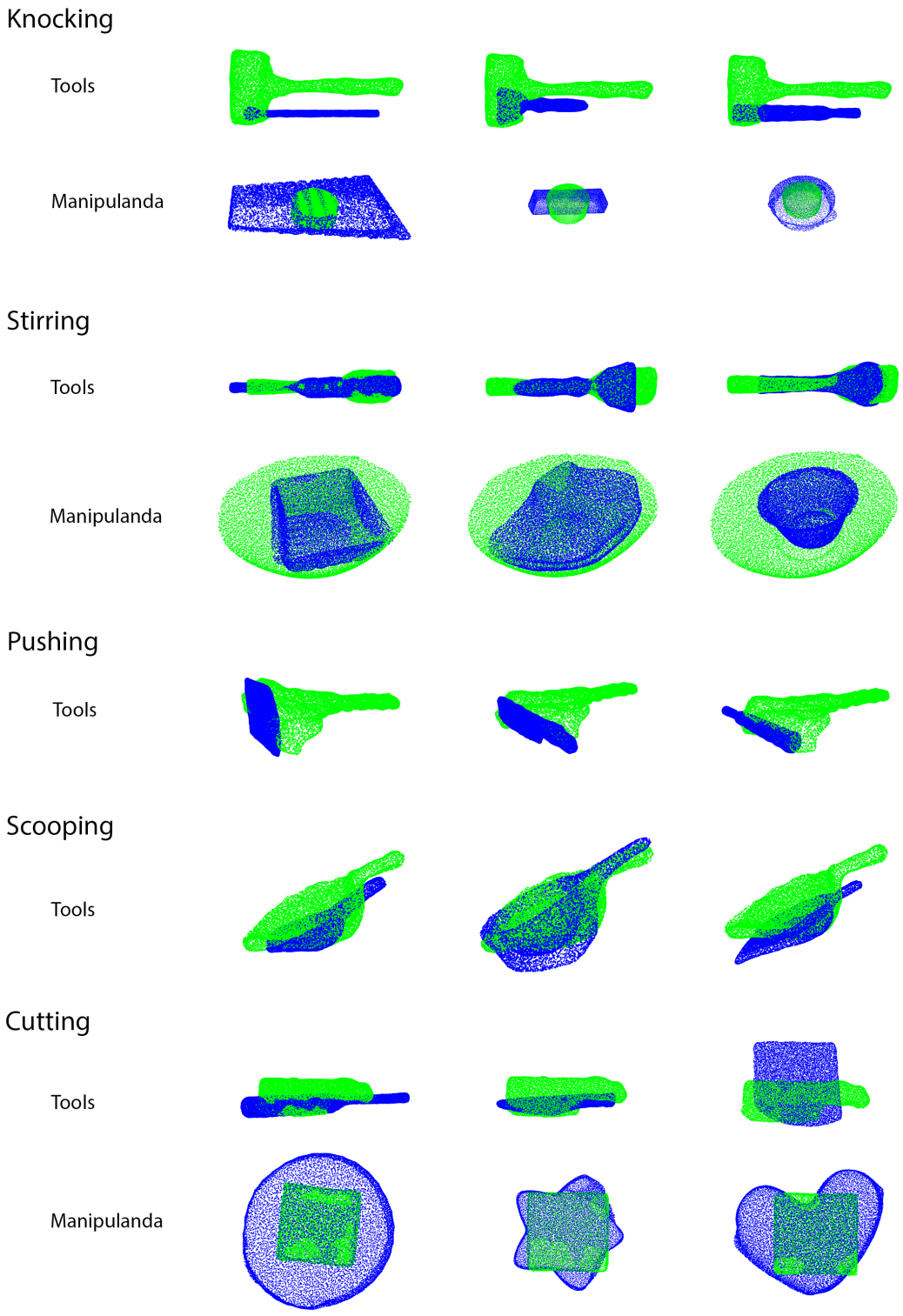


Figure 3.9: Results of aligning substitute objects to source objects (Star 2). The green point clouds are the source objects while the blue point clouds are the substitute objects. Manipulandum substitution for the pushing and scooping task is not geometry-dependent, but goal-dependent, and therefore, the alignment results are excluded in the figure.

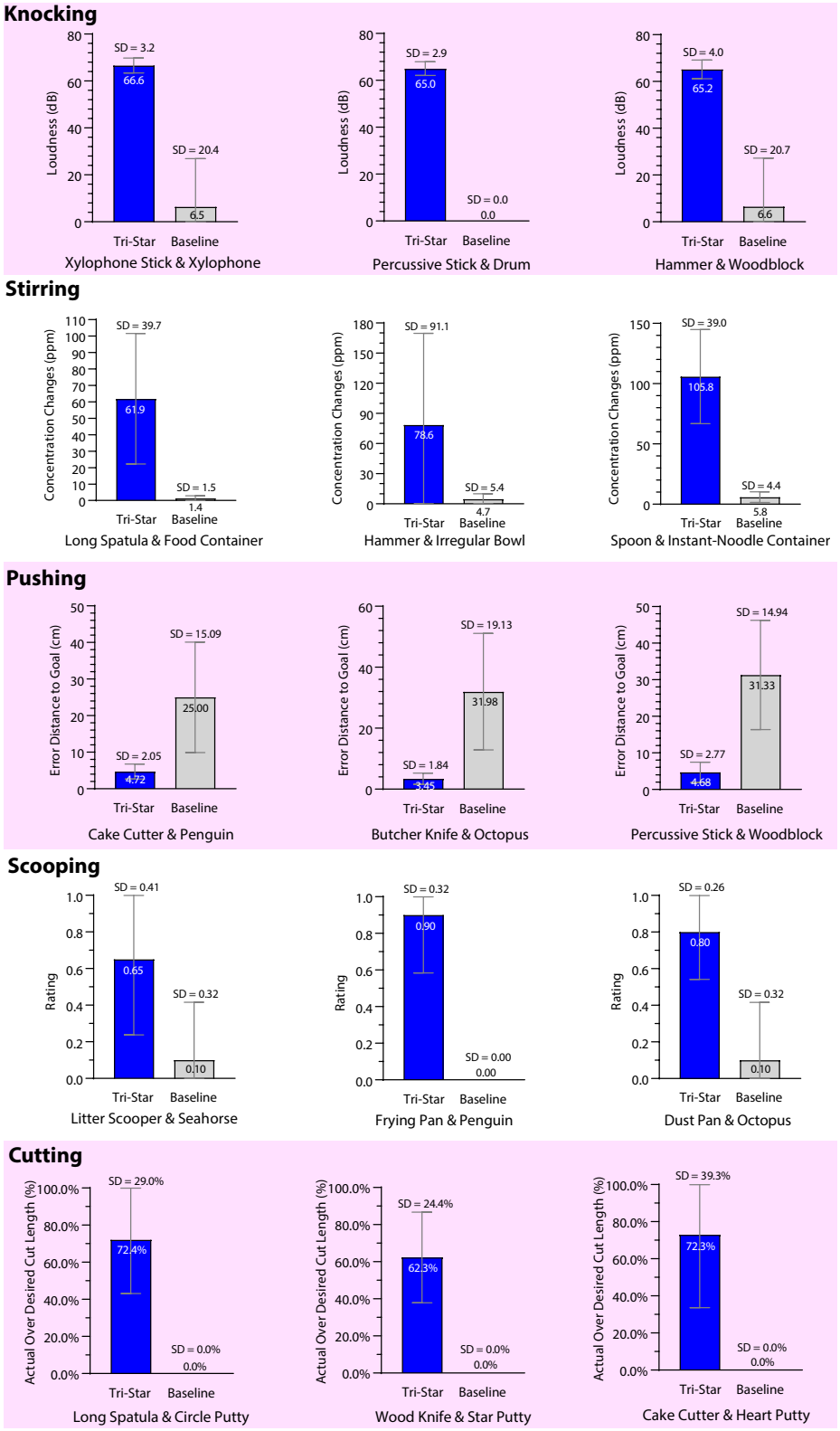


Figure 3.10: Results of tool substitution and manipulandum substitution (Star 2). The bar graphs show the results of using the substitute objects to perform knocking, stirring, pushing, scooping, and cutting. The bars compare Star 2's (blue) performance against the baseline (gray).

the previous study.

Stirring. All three substitute tools successfully stirred the room-temperature salted water in the substitute containers in all trials in the testing condition, while all substitute tools failed to stir in the baseline condition. More salt dissolved in the testing condition (concentration change: $M = 82.10$ ppm, $SD = 62.29$ ppm) than in the baseline condition ($M = 3.97$ ppm, $SD = 4.43$ ppm). We did not encounter another study that performed a similar task.

Pushing. The manipulanda were pushed closer to the goal in the testing condition (translation error: $M = 4.28$ cm, $SD = 2.26$ cm) than in the baseline condition ($M = 29.44$ cm, $SD = 16.24$ cm). In a previous study that also used raw sensor data to perceive the environment (Xie et al., 2019), the translation error using substitute tools and source manipulanda was similar ($M = 5.56$ cm, $SD = 4.13$ cm) to the current study but required more than 10^4 training samples.

Scooping. The substitute tools scooped the substitute manipulanda more successfully in the testing condition (rating: $M = 0.78$, $SD = 0.34$) than in the baseline condition ($M = 0.07$, $SD = 0.25$). In a previous study (Gajewski et al., 2019), the scooping task was tested only in simulation with substitute tools and source manipulanda, and no quantitative results (e.g, success rate) were provided.

Cutting. The robot cut the manipulanda more thoroughly in the testing condition (cut length percentage: $M = 78.33\%$, $SD = 33.95\%$) than in the baseline condition ($M = 6.67\%$, $SD = 25.37\%$) even with relaxed criteria in the baseline condition as mentioned in the Star 1 evaluation. In a previous study (Gajewski et al., 2019), the cutting task was tested only in simulation with substitute tools and source manipulanda, and no quantitative results (e.g., success rate) were provided.

3.3.3 Star 3: Transfer Tool Use To Other Robot Platforms

Six tasks (pushing, stirring, knocking, cutting, scooping, and writing) were used to test skill transfer from a UR5e robot to both a Baxter robot and a Kuka youBot without additional training. Due to the size and payload limitations of Baxter and youBot, source tools different from Star 1 were chosen. The experiments were similar to the ones in Star 1. However, no baseline conditions were included in Star 3, and no comparisons were made with other studies since we did not encounter similar studies. Figure 3.5b and 3.5c show the testing environment of Baxter and youBot. The objects in the yellow frames of Figure 3.8 are the objects tested in Star 3. Star 3 only considered scenarios that the new platforms could complete if they were trained in the same way as the source platform. Therefore, the task configurations of all experiments were within the feasible workspace of the new robots. Figure 3.11 summarizes the results.

Knocking. All three robots successfully completed all trials. The loudness created by the UR5e (M = 75.43 dB, SD = 2.57 dB), Baxter (M = 74.04 dB, SD = 3.95 dB) and youBot (M = 73.89 dB, SD = 7.78 dB) were similar.

Stirring. All three robots successfully completed all trials. The concentration changes of the stirs by Baxter (M = 185.10 ppm, SD = 86.01 ppm) and youBot (M = 176.00 ppm, SD = 35.74 ppm) was slightly higher than the stirs by the UR5e (M = 160.60 ppm, SD = 43.71 ppm).

Pushing. YouBot (translation error: M = 2.40 cm, SD = 1.02cm) pushed the manipulanda slightly closer to the goal than UR5e (M = 3.78 cm, SD = 1.74 cm) or Baxter (M = 4.04 cm, SD = 2.25 cm), which was because of the shorter pushing length by youBot due to limited maximum reach compared with UR5e and Baxter. Since it is open-loop control, fewer errors are accumulated during this shortened course of pushing.

Scooping. UR5e (ratings: M = 0.90, SD = 0.21), Baxter (M = 0.90, SD =

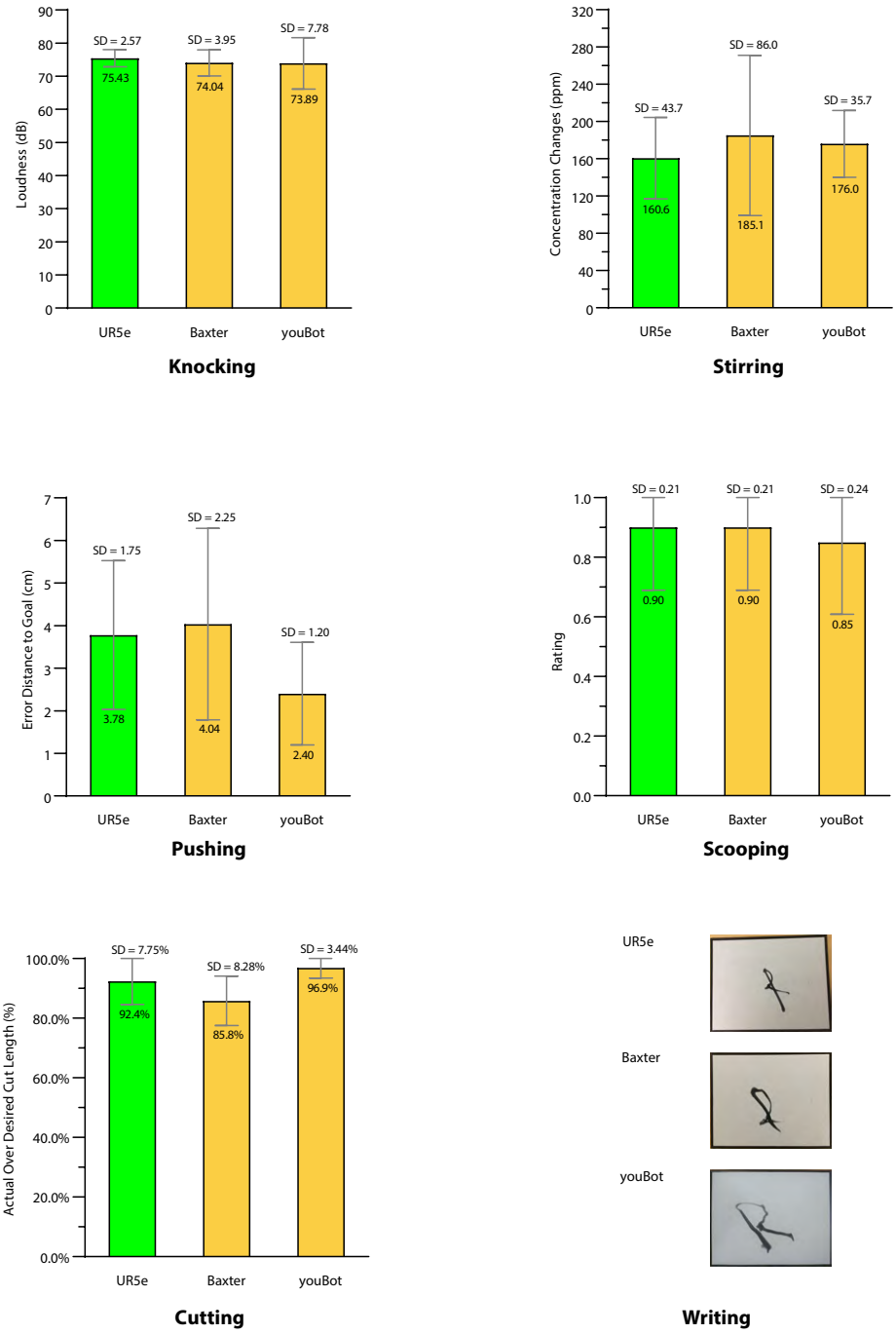


Figure 3.11: Results of tool use generalization across robot platforms (Star 3). The bar graphs include results of the UR5e (green), Baxter (yellow), and youBot (yellow) using the source tool/manipulandum combinations for knocking, stirring, pushing, scooping, and cutting. The pictures at the bottom right demonstrate different robots writing "R" with trained scale and orientation.

0.21) and youBot ($M = 0.85$, $SD = 0.24$) performed equally well.

Cutting. The average cut length percentage cut of UR5e ($M = 92.39\%$, $SD = 7.75\%$) and youBot ($M = 96.92\%$, $SD = 3.44\%$) was slightly longer than Baxter ($M = 85.83\%$, $SD = 8.28\%$), which was due to the difficulty in securing the spatula tightly in Baxter's gripper.

Writing. All three robots were able to repeat the letter "R. " Figure 3.8 shows the letter "R" with the trained scale and orientation written by the three robots.

Screw-driving. All three robots in simulation completed the task successfully.

3.4 Discussion

The results show that the TRI-STAR framework learned a wide range of tasks, generalized the learned skills to substitute tools and manipulanda, and transferred the learned skills across robot platforms. This was achieved by using our task-oriented approach, which includes an affordance taxonomy and identified taxonomic knowledge which specifies knowledge shared across tasks that belong to particular task categories and minimizes the need for knowledge to be defined on a per task basis. We center our discussion around the ways our framework improves upon the state-of-the-art in task-general tool use but also identify limitations of our approach.

3.4.1 Contribution 1: Task-Generality

TRI-STAR is a task-general tool use framework shown to learn, generalize, and transfer tool use for a variety of everyday tasks. Not all tool use algorithms are intended to be task-general as they assume pre-defined knowledge specific to individual tasks at either the learning or generalization stage. Three advances made it possible for TRI-STAR to be task-general. First, we summarized taxonomic knowledge of tasks which enables a multitude of tasks to be learned efficiently including potentially any

undemonstrated tasks covered by one of the known taxonomic categories. Second, TRI-STAR can handle tasks with different contact pose requirements (e.g., pushing, knocking, and screw-driving) and different types of trajectories (e.g., circular periodic trajectories including stirring, linear trajectories including cutting, non-linear trajectories including scooping, trajectories that could be either linear or non-linear including pushing, and complex trajectories with both linear and non-linear segments including writing) which allows a robot to work with a wide range of tasks. Third, TRI-STAR can generalize the tool use to tasks whose substitute tools and manipulanda may be geometrically-similar or geometrically-distinct objects since we made no assumptions about the shape of the objects, unlike previous approaches (Brandi et al., 2014; Fang et al., 2020). An added benefit of generalizing tool use to geometrically-distinct objects is that it can allow a robot to improvise the use of objects such that an object not designed for a task could be used when desired objects are unavailable.

3.4.2 Contribution 2: Data Efficient

Task-general frameworks typically required a large training set size. However, training with a large sample size is time-consuming and thus impractical in time-sensitive domains like search-and-rescue. By leveraging taxonomic knowledge identified for each task category, TRI-STAR required only 20 examples to learn each task, and no additional training samples were needed by Star 2 to generalize the usage to substitute objects or by Star 3 to transfer the skills to other platforms. In contrast, previous studies required over 5,000 (Gajewski et al., 2019), 18,000 (Fang et al., 2020), and 20,000 (Xie et al., 2019) training samples. The small set of training samples needed for each task makes it time-efficient for TRI-STAR to learn new tasks and thus easy to be deployed as an application in the real world. Moreover, TRI-STAR experienced only a minor loss in performance while significantly reducing the necessary training samples.

Star 3 does not require additional training data nor extra algorithmic infrastructure to implement, but rather updates a common representational schema (Cartesian trajectory) that is utilized in many tool use studies. To automate research work, robots have been deployed in chemistry labs (Burger et al., 2020), where tasks and tools are standardized. The ability to transfer skills between robots could save researchers in each lab hundreds of hours of training time as skills could be shared across research labs. For robots in the factory or warehouse, it will be cost-efficient for skills to be transferred to new models without having to shut down the factory in order to debug compatibility-related problems. For other applications, platform-agnostic skill transfer would not merely be a convenience but could open entirely new applications. For example, for in-home robots, the prospect of training every single task by each individual is a nonstarter for most consumers, whereas having access to a shared library of skills may be more acceptable.

3.4.3 Contribution 3: Integrative framework

We demonstrate TRI-STAR's ability to handle all three stars, including tool use learning, tool substitution, and tool use transference to other platforms. Previous studies on task-general tool use either focused on learning basic tool use or tool substitution and typically limit the types of objects considered (e.g., they only consider objects that share similar form-factors or only consider tool but not manipulanda substitution). Other tool use studies tend to be customized to particular tasks, which makes adapting them for the wide variety of tasks a robot might realistically encounter challenging without significant modifications. In contrast, TRI-STAR not only enables all these functionalities within one integrative framework but also removed these limitations. Moreover, our framework encompasses a pipeline which includes important aspects often ignored in other studies such as tool-manipulandum contact pose learning and a perception system customized to the needs of tool use. Our framework

covered important aspects that were not mentioned in previous studies, such as tool-manipulandum contact pose learning. We integrated all of these into TRI-STAR and showed its effectiveness with a wide range of tasks. Being an integrative framework makes it plausible for TRI-STAR to be deployed into real-world contexts.

3.4.4 Limitations

While our results demonstrate the potential of our framework, it has limitations. First, the robot used position control only, rather than force control or feedback control, to learn and complete tasks, which limits its effectiveness on tasks that require consideration of the forces being applied to the manipulanda such as nail-hammering, or tactile feedback such as inserting a key into a lock. Second, our framework only considers the geometric features of the tools and manipulanda and does not consider other properties (e.g., material, weight, texture), which may hinder the robot's ability to choose the most appropriate contact areas for tools like sandpaper that have a single abrasive surface but are otherwise geometrically uniform. Third, although our system calculated the grasping location on the tool, automatic grasping was not demonstrated in the evaluation.

Other limitations also exist for TRI-STAR. First, our framework assumes that all objects, including relevant objects in the environment, are rigid bodies with no joints (i.e., have 0 DoF). This assumption does not allow a robot to handle common tools such as scissors or washcloths or to perform tool use tasks on top of soft surfaces. Second, our framework relies on accurate visual perception and structured environments, which is a common problem for non-marker-based perception systems and is an impediment to handling tasks that require highly accurate perception, such as surgery. Third, object alignment relies on full 3D models though ideally, this system should perform alignments using only partial point cloud data of both geometrically-similar and geometrically-distinct objects. Fourth, TRI-STAR cannot learn the cause-and-

effect relations (e.g., Brawer et al., 2020) that comprise taxonomic knowledge, which does not allow it to, for example, automatically choose between the actions required to stir a liquid versus a heavier mixture like a batter.

3.5 Summary

In this chapter, we presented the TRI-STAR framework that can perform basic tool use, transferable tool use, and improvisatory tool use. Despite its effectiveness with single-manipulation tool use, it does not consider multiple-manipulation tool use, which we will address in the next chapter.

Chapter 4

Multiple-Manipulation Tool Use: Sequential Tool Use[1]

In this chapter, we present a planning algorithm that aims to handle part of open challenge 4 as described in Chapter 2. Specifically, this algorithm helps to expedite the search for valid trajectories in sequential tool use. Task and motion planning (TAMP) is a classic algorithm for solving this type of problem (Garrett et al., 2021) which combines high-level task planning and low-level motion planning to generate robot action plans in different scenarios. The low-level motion planner searches for a solution in a high-dimensional continuous space that includes both robot configurations and environmental variables. However, the high dimensionality of the search space poses challenges in efficiently identifying infeasible tasks or producing solutions for tasks in constrained environments. In order to improve run-time performance for these types of tasks, we propose an intermediate-level *affordance planning* stsep situated between the high-level task planning and low-level motion planning steps. We envision affordance planning as an extension, rather than an alternative, to TAMP,

[1]Portions of this chapter are currently under review: M. Qin, J. Brawer, and B. Scassellati. Using Task, Affordance, and Motion Planning (TAAMP) to Detect Infeasible or Limited Solutions in Affordance-Constrained Environments.

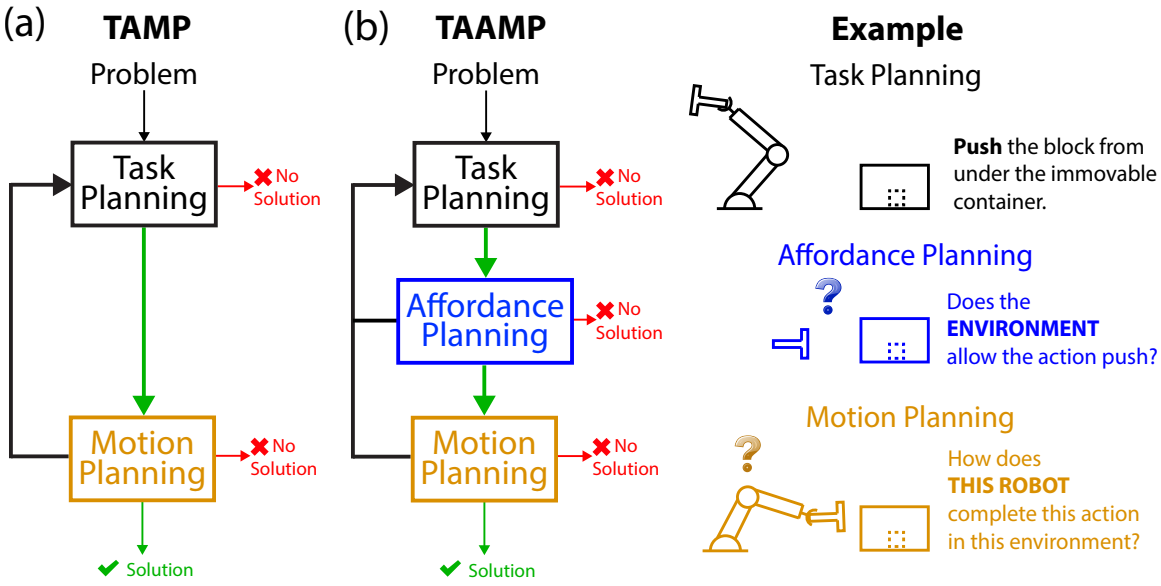


Figure 4.1: A comparison of traditional Task And Motion Planning (TAMP) (a) and the proposed task, affordance, and motion planning (TAAMP) (b). On the right, cartoon vignettes showing the questions addressed at each level of these models with an example problem. In this problem, a robot should push an object from under an immovable container. Task planning chooses the action `push`. Affordance planning determines whether the environment can afford the action at all, independent of who the actor is. Motion planning addresses how to complete the action with this specific robot.

and we call the combined model Task, Affordance, And Motion Planning (TAAMP). Affordance planning addresses whether the environment can afford an action in a general sense, irrespective of specific robot configurations. This actor-agnostic property excludes the need to consider robot configurations, allowing affordance planning to be performed in a lower-dimensional space. Consequently, affordance planning acts as a filter that attenuates the search space of motion planning. We evaluated TAAMP with tasks with different amounts of affordance constraints. Results show that the benefit of this additional affordance level in TAAMP outweighs its extra costs in tasks with constrained affordance or even in infeasible tasks, while minorly impacting performance for tasks with relatively unconstrained affordances.

4.1 Introduction

Task and Motion Planning (TAMP) (Garrett et al., 2021) is a popular method to solve the problem of planning robot motion to achieve particular goals in various environments. It operates by interleaving high-level discrete task planning, which generates abstract action skeletons, and low-level continuous motion planning, which instantiates action parameters and generates robot motion plans. Consider though the infeasible problem in Figure 4.1 in which a robot must push a block located under an immovable container with a tool. In TAMP, the infeasibility of a task is signaled by failing to generate motion plans within the allotted time, which impedes a robot's ability to act in real-time. When searching for viable plans, motion planners in TAMP focus on the robot, and condition the search in the robot configuration space with the constraints provided by the environment and the robot. Searching in this space is generally expensive due to the high dimensionality of robot configurations, especially when the task is infeasible. However, the infeasibility in this problem is not due to robot-related parameters such as robot kinematic constraints or robot grasping poses. The infeasibility of this task is actor-agnostic; that is, it is independent of who is attempting the task (e.g., a robot, an animal, a human) and is solely dependent on the environment. This environment simply does not afford the block being pushed. Similar scenarios also apply to general manipulation tasks such as picking up the block located under the immovable container.

Therefore, we extended the current TAMP model. In our new model of Task, Affordance and Motion Planning (TAAMP, shown in Figure 4.1b), we include an intermediate level, i.e., affordance planning, in between task planning and motion planning. In TAAMP, Task planning and motion planning assume their original functions as in TAMP. More specifically, task planning addresses what actions to choose, if any. Motion planning addresses if or how the actions can be achieved *with the specific robot*. Unlike motion planning, affordance planning shifts the focus

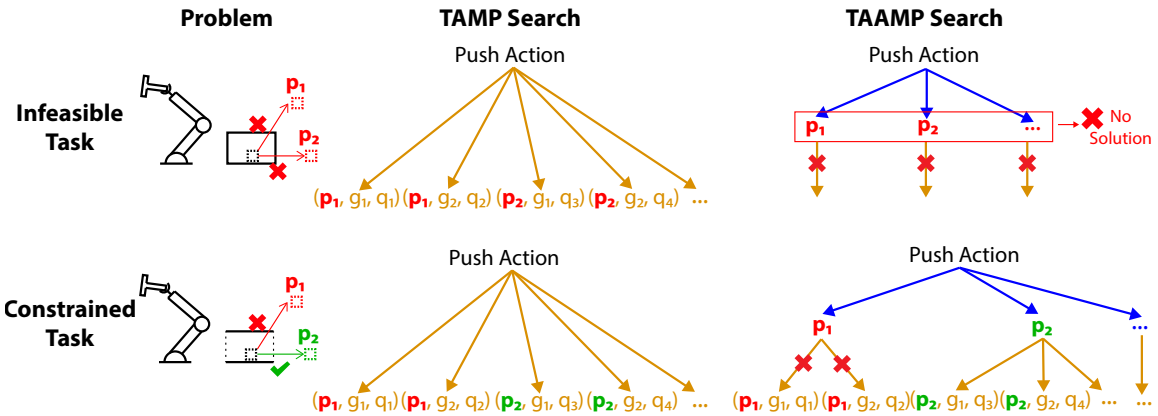


Figure 4.2: Diagram of the search for action parameters in TAMP and in TAAMP for the action `push`. (Top) In an infeasible task in which a block cannot be pushed from under an immovable container, TAMP must search a high-dimensional space for a solution which includes target poses of the block `p`, grasping pose `g`, and robot configurations `q`. Affordance planning detects that none of the `p`'s are feasible in a generic sense and does not execute motion planning. (Bottom) In a constrained task where the block is located inside a tunnel, TAMP needs to consider a large space, including `p`'s that are either feasible or infeasible. Affordance planning detects the `p`'s that are feasible and performs motion planning only within feasible `p` states.

from the robot to the objects to be manipulated, and detects whether specific actions can be achieved at all in a particular environment. Equivalently, affordance planning addresses whether the environment affords the action and/or what are the constraints enforced by the environment *without considering the robot.*

Affordance planning can efficiently detect actions with infeasible affordances, which we illustrate with an example depicted in Figure 4.2 (top). Motion planning in traditional TAMP approaches considers a configuration space of higher dimensionality, which includes object target poses, robot parameters (e.g., grasp poses), and robot kinematic constraints. Such space is challenging for sampling-based approaches since the space is large and the detection of infeasibility may rely on timeout. The high dimensionality makes the computation expensive in optimization-based approaches. In contrast, the intermediate-level affordance planning recognizes that the infeasibility of the task is due to the environment and stops searching. TAAMP does not need to execute the expensive motion planning in this case.

Affordance planning can also benefit actions with constrained affordances, or in other words, actions with a narrow set of possible solutions, as demonstrated in Figure 4.2 (bottom). In this example, the immovable container is replaced with a tunnel so that the object can only be pushed along the tunnel. Traditional TAMP approaches need to search feasible object poses in the high-dimensional motion planning space. Conversely, affordance planning allows a more efficient search for feasible poses in a lower dimension space since the search space does not involve robot configurations. The search result of affordance planning acts then as a filter for motion planning, and reduces the search space of motion planning. Therefore, motion planning is more efficient in generating motion plans since it only considers feasible target poses. In summary, TAAMP benefits planning for constrained tasks by performing a portion of the search in a lower-dimensional space and thus reducing the overall search space in motion planning.

Affordance planning attempts to decouple environment constraints from robot constraints as much as possible to reduce the search space for the motion planner. However, we recognize that some constraints are caused by a mixture of environmental and robotic constraints. In these cases, we use motion planning to handle these environmental constraints. We evaluated the run-time performance and success rates using kitchen tasks in simulation and on a physical robot with different levels of affordance constraints, such as tasks with relatively unconstrained affordances, constrained affordances, and infeasible affordances. Our results suggested that the benefit of affordance planning outweighed the extra cost in infeasible and constrained tasks, while has a minimal impact on the performance of unconstrained tasks.

The detection of infeasible actions in previous studies relied on robot configurations. Hierarchical Planning in the Now (HPN) (Pack and Lozano-Pérez, 2011) and similar planners (Stilman et al., 2007; Stilman and Kuffner, 2008) backtrack when collisions are detected using the swept volume of a specific robot trajectory. In the

planner embedded with causality-based geometric reasoning (Erdem et al., 2011), constraints were added to the planner when specific robot trajectories resulted in failures. When none of the motion plans generated were feasible (Srivastava et al., 2014), they selected one robot trajectory and identified which objects were involved in collisions.

Other studies leveraged constraint-based approaches to TAMP, and the constraints were generally related to samplings of robot configurations. For example, motion planning incorporated both geometric and robot constraints that resulted from the instantiations of previous actions in order to solve the task of stacking multiple cups vertically (Lagriffoul et al., 2012, 2014). A previous study (Bidot et al., 2017) on geometric backtracking connected the constraints on robot configurations with geometric constraints. Iteratively Deepened Task and Motion Planning (IDTMP) (Dantam et al., 2016) updated constraints incrementally to improve run-time performance. The addition and removal of constraints were based on robot motion feasibility.

Recent sampling-based TAMP studies focused on improving the search algorithm to expedite the search. FFRob (Garrett et al., 2018) employed FastForward search. PDDLStream (Garrett et al., 2020) developed an adaptive algorithm that outperformed FFRob, and showed impressive performance on unconstrained tasks. PDDLStream is one of the state-of-the-art algorithms in TAMP, and we compare the performance of our approach with PDDLStream.

The sampling-based TAMP planners mentioned above focused on robots and combined both robotic and environmental constraints in motion planning. Conversely, the intermediate-level affordance planning is actor-agnostic; it focuses on the object to be manipulated and considers constrained affordances furnished by the environment in isolation of the robot. As a result, motion planning of TAAMP, while still proceeding in the high-dimensional configuration space with both robotic and environmental constraints, searches more selectively.

Another set of research employed optimization-based approaches (Toussaint et al., 2018; Migimatsu and Bohg, 2020; Hadfield-Menell et al., 2016). Such algorithms focused on the application of optimization techniques in order to find the optimized solutions of all action parameters at once given an action skeleton. When no solution could be found for the current action skeleton, it continued with the next action skeleton until all action skeletons were exhausted. Therefore, motion planning in optimization-based approaches takes both environment and robot constraints as inputs, which is different from our study in that these constraints are relatively decoupled and affordance planning calculates in a lower-dimensional space.

While the above studies focused on the detection of infeasible solutions or the generation of constrained solutions, other studies explored how to learn what is infeasible. For example, the score space was employed to evaluate the similarity between different problem instances and transferred knowledge from similar problems to generate motion plans (Kim et al., 2019). In another study (Wells et al., 2019), a classifier was trained to classify motion feasibility with the input of robot constraints, task action and the problem instance. Instead of problem instances, the Deep Generative Constraint Sampling (DGCS) (Ortiz-Haro et al., 2022) took the scene image as input. These studies focused on the actors (the robots), while affordance planning focuses on the object to be manipulated.

4.2 Methods

4.2.1 TAAMP

We implemented[2] TAAMP based on PDDLStream, which is one of the state-of-the-art TAMP planners. However, TAAMP can be integrated into other planners, including optimization-based planners.

[2]source code: `https://github.com/ScazLab/TAAMP.git`

Overview of PDDLStream

PDDLStream provides an interface to program TAMP tasks, integrates the contributions of previous studies, and employs efficient search algorithms. It combines standard Planning Domain Definition Language (PDDL) (McDermott et al., 1998) with streams to sample action parameters or to certify predicates. In PDDLStream, a TAMP task is defined with a domain PDDL and a stream PDDL. The domain PDDL is the same as the domain in classic PDDL that mainly consists of predicate definitions and actions. The streams governed the sampling procedures, and three types of streams are relevant: the *function streams* that produce one set of outputs (e.g., given the grasp pose, object pose, and the tool, what is the robot configuration to push an object to the designated position), the *test streams* that certify the boolean value of a predicate (e.g., whether a specific robot trajectory results in collisions), and the *sampling streams* that produce finite or infinite samples (e.g., possible grasping poses given an object or a tool). The sampling procedure and the search are interleaved to find the solution (for more information about PDDLStream, see (Garrett et al., 2020)).

Our Contribution - Affordance Planning

Affordance planning was added as a test stream in each action that required affordance tests. This test stream did not contain any robot-related parameters, such as grasping poses and robot configurations, because affordance planning was actor-agnostic. The affordance tests were generic to all actions such that they returned feasible affordances, which are subsets of possible affordances. *Possible affordances* referred to all possible object pose changes and the corresponding object trajectories allowed by an action without considering collision. The range of possible affordances were determined by how this affordance was trained. While pushing can result in orientation changes of an object in theory, the allowable range did not consider such

changes if the training only involves straight-line pushing. Given that finding additional possible range is not the focus of this study, our affordance planning followed what was seen in the training samples. In order to be *feasible affordances*, the object end pose and the trajectories should be collision-free in the current context. For example, when detecting feasible affordance for a pushing action, it would not consider the vertical pose changes of the object on a flat surface which fell outside of possible affordances, nor end poses that may collide with other objects.

The pose changes were rigid-body transformations, parameterized by screw representations as screw axes and angles in a similar way as in Chapter 3. The screw axes uniformly represented the shape of the trajectory that could be linear, curved, helical, or circular. The angles represented the length of distance traveled along the trajectory. The choice of the frame of reference was based on the type of task, as described below in Section 4.2.2.

The affordance tests employed a sampling-based approach. The affordance tests first checked whether the affordance was constrained in the context of the task by randomly sampling from possible affordances. If the majority of resulting end poses and corresponding trajectories did not result in collisions, the affordance planner determined that the affordance was unconstrained and passed all possible affordances to the motion planner to find robot motion plans. Otherwise, the affordance planner considered the environment to be constrained.

When the affordance was considered to be constrained, the affordance planner conducted a brute-force search to find feasible affordances. We discretized the space of all possible affordances, more specifically, the range of the angle parameters that represent the length of the trajectory in the screw representations, and searched for the instances that did not result in collisions. These feasible affordances were then passed to the motion planner. When generating candidate robot motion plans, the motion planner only considered the goal pose of the objects defined within the feasi-

ble affordances. If the feasible affordances space was empty, the affordance planner announced that the action was infeasible in the current context, without starting the motion planner. We acknowledge that this method, especially the brute-force search, is inefficient, and more efficient methods can be explored in future studies.

Whether the environment allowed a target pose and the corresponding trajectory of an object was decided by whether there were collisions, and collision-checking between meshes can be computationally expensive. Since collisions may only happen when objects are physically close, we only checked for collisions when the distance between two objects was below a threshold. The threshold was not fixed, but was calculated as the sum of half of the diagonal length of the bounding box of each mesh. If the distance between two meshes were below the threshold, we first checked whether the bounding boxes of two meshes were in collision as this is faster than checking collisions between two meshes. If the bounding boxes were in collision, we used meshes to detect collision. We chose oriented bounding boxes over axis-aligned bounding boxes because they represent the shape of the original meshes more faithfully than the axis-aligned bounding boxes. Additionally, calculating oriented bounding boxes can be computationally expensive. Therefore, we stored the oriented bounding box of each object during the planning process since the bounding box is not context-dependent, and applied necessary transformations to the bounding boxes when the pose of the corresponding object changed.

4.2.2 Learning the Action Affordances

Our system learned affordances of actions, though affordances could be pre-defined in TAAMP for simplicity. We chose the TRI-STAR framework presented in Chapter 3 to learn tool affordances, and expanded this framework to learn object affordances so that it can handle both tool-use and general manipulation tasks. However, in principle, any affordance-learning framework can be used with TAAMP.

Overview of the TRI-STAR Framework

TRI-STAR learns how to use tools to achieve a preset goal, such as pushing an object to the desired location or knocking on a target object to make a sound. The tasks considered by TRI-STAR are tool-use tasks that can be completed by one action, rather than by a series of actions. It can handle a wide range of tasks such as knocking, pushing, and driving screws. These three tasks typify three distinct task types in the framework's affordance taxonomy based on whether the object being manipulated results in pose changes. The drum has no pose changes when being striked or knocked with a mallet. In contrast, the shape of the trajectory of a screw in a screw-driving task is uniform relative to the its initial pose (i.e., the object frame). Different still are pushing tasks, which can exhibit a diversity of trajectories in the object frames for the objects to be pushed. These observations suggested the tool affordance can be learned more efficiently in different frames based on the type of task in order to minimize the representation of the tool affordances. The detection of the task type was not pre-defined, but was included in the learning process.

Our Contribution - Learning Object Affordances

In order to evaluate the benefits of affordance planning and its extra costs, we compared our results with a TAMP planner without affordance planning, PDDLStream. Our system learned the affordances of the objects to be manipulated. More specifically, we learned the distribution or range of the possible pose changes of the object. Other than the shape, we also learned the range of the length of the trajectory. Similar to TRI-STAR when learning tool affordances, we utilized different frames of reference based on the taxonomy in TRI-STAR. Moreover, we further categorized tasks with infinitely many possible end poses in the object frame. In our extended taxonomy, this type of task was categorized based on the uniqueness of the pose changes in the world frame. For example, the pushing task may result in infinitely many possible

pose changes given a starting pose. Scooping tasks, in contrast, typically concern vertical poses changes.

4.3 Results

The intermediate-level affordance planner aimed to detect environment constraints rather than constraints related to the robot. Therefore, we evaluated the performance of TAAMP using four types of tasks with different levels of affordance constraints:

- *Unconstrained tasks*: feasible tasks with relatively unconstrained affordances. Traditional TAMP planners should be able to find solutions quickly on these tasks;
- *Constrained tasks I*: constrained tasks that do not require backtracking. That is, no objects other than the target object need to be relocated to complete tasks;
- *Constrained tasks II*: similar to Constrained tasks I but requires backtracking;
- *Infeasible tasks*: infeasible tasks with completely constrained environments that do not provide necessary affordances.

For each type of task, we included a manipulation task and a tool-use task to show that TAAMP can benefit a wide range of tasks. Manipulation tasks were commonly seen in previous TAMP studies, while tool-use tasks were not typical in TAMP studies (for demonstrations of tool-use tasks in TAMP, see Toussaint et al., 2018). We included tool-use tasks because our affordance planning may be especially helpful for this type of task. When a tool is needed to push or pull an object, the object is usually out of reach, or hard to reach due to the constrained environments, e.g., a broom is used to retrieve objects under a sofa. In the latter case, the chance of finding the solution with random sampling is small because only limited target poses are feasible.

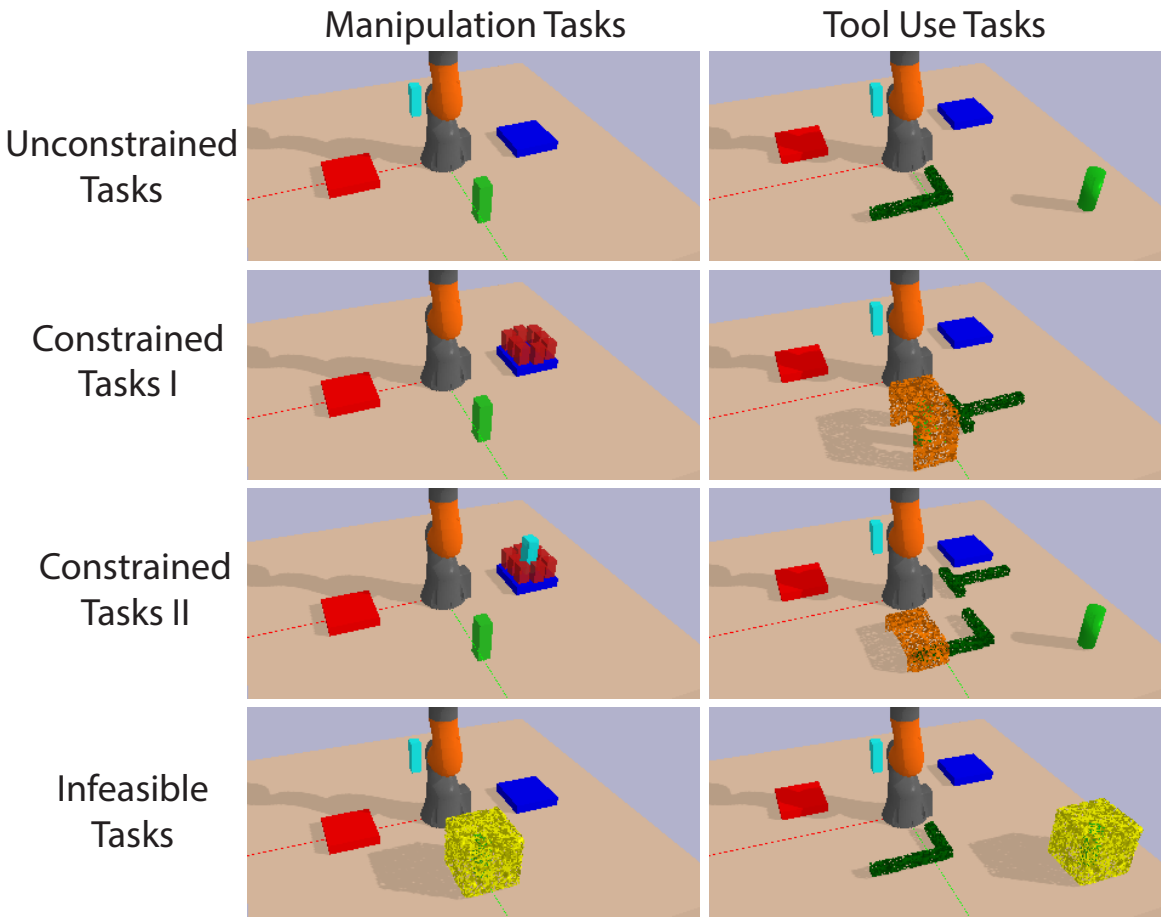


Figure 4.3: Eight tasks in our evaluations in simulation. In the unconstrained manipulation task, the robot should place the green block (the "celery") on the immovable blue surface (the "sink") to clean it and then place it on the immovable red surface (the "stove") to cook it. The cyan block (the "radish") is movable but not directly related to the goal. The rest of the tasks shared the same goal. In the constrained manipulation task II, the robot needs to relocate the extra cyan block. In the infeasible manipulation task, the robot cannot access the green block since it is located under a yellow immovable. In the tool-use tasks, the robot should pull the green block that is out of reach with the L-shaped tool (i.e., the unconstrained tool-use task), or push the green block under the immovable orange tunnel (i.e., the constrained tool-use task I), or push the L-shaped tool to expose the grasping part of the L-shaped tool in order to use the L-shaped tool to pull the green block (i.e., the constrained tool-use task II), or cannot complete the task when the green block is located under the yellow immovable container.

Figure 4.3 shows all eight tasks tested in simulation. The unconstrained manipulation task (top left) is a task originally included in the PDDLStream (Garrett et al., 2020) source code package. In this task, a simulated KUKA iiwa robot arm is ex-

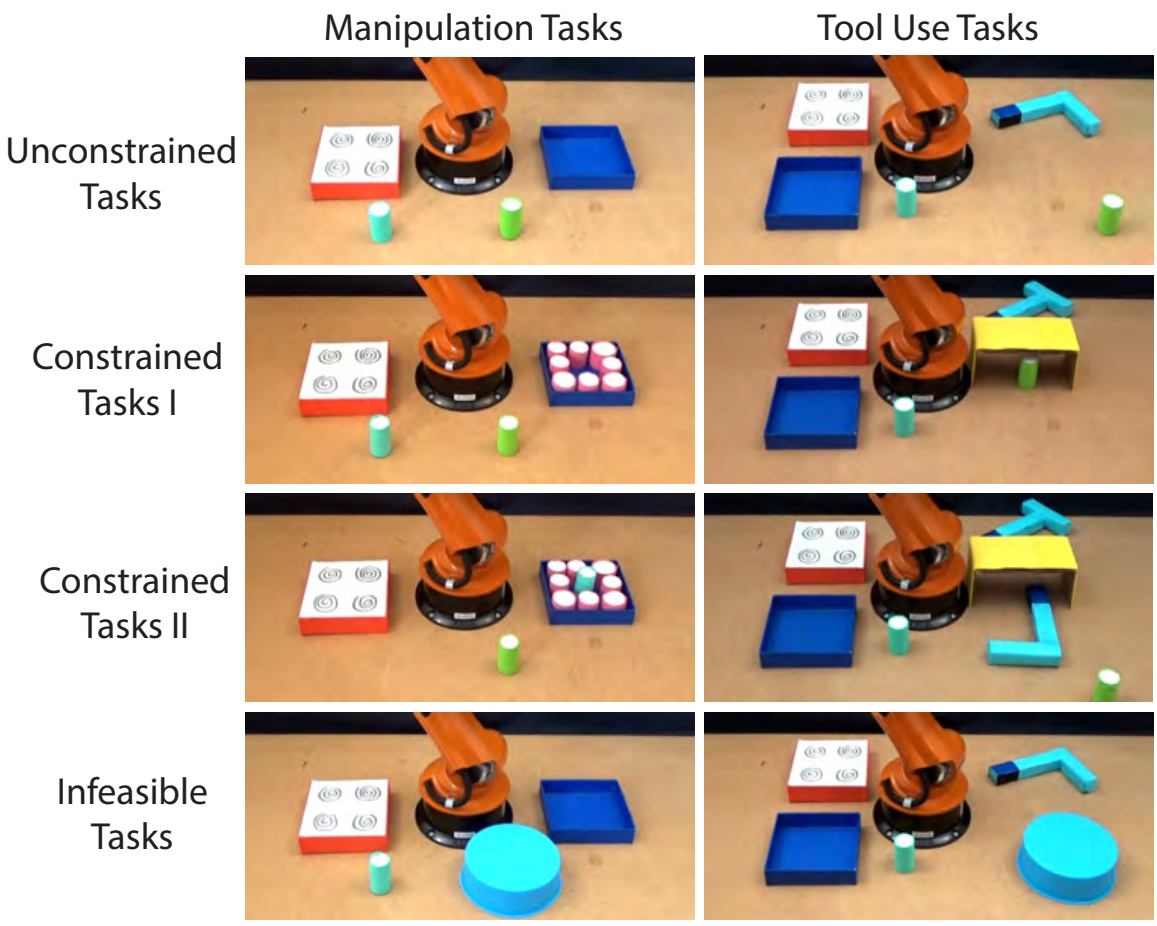


Figure 4.4: Eight tasks in our evaluations with a Kuka youBot arm.

pected to cook a "celery" (the green block) by first placing it on the "sink" (the blue surface) to clean it, and then placing it on the "stove" (the red surface) to cook it. The "radish" (the cyan block) is a movable object but irrelevant to the goal. We utilized this task to guarantee that the task is friendly to PDDLStream. Other tasks, including tool-use tasks, were all variations of this task. We also demonstrated similar tasks on a physical Kuka Youbot robot arm as shown in Figure 4.4.

We trained action affordances on four actions: `pick`, `place`, `push`, and `pull`. The search method we chose for our approach and for PDDLStream was the adaptive algorithm, which was shown to be most efficient in a previous study (Garrett et al., 2020). We executed the scripts on a laptop with an 8th generation Intel Core i9-8950HK CPU with 32 GB memory operating on a Ubuntu 18.04 system. We tested

thirty trials for each task and evaluated run-time performance, as well as the success rate of producing correct plans or correctly recognizing the infeasibility of a task within the allotted time frame. The timeout for each trial was set to ten minutes.

Table 4.1 shows the results in simulation. For infeasible tasks, a success is to identify the infeasibility before timeout. Due to the additional affordance planning, our approach was slightly slower than PDDLStream in unconstrained tasks. However, our approach was more likely to solve constrained and infeasible tasks than the TAMP planner without affordance planning.

4.4 Discussion

We proposed the intermediate-level affordance planning to solve TAMP tasks, which resulted in the three-level TAAMP model. Results showed that the benefit of TAAMP outperformed its additional cost, and thus expedited the generation of motion plans in constrained tasks and the identification of infeasible tasks, while having a minor impact on the performance for unconstrained tasks. Affordance planning of TAAMP should be considered an extension to current TAMP planners, rather than a replacement to motion planning. Given that the task and motion planning in the TAAMP model is the same as other TAMP models, the task planner and the motion planner of TAAMP can be replaced when more efficient planners are developed in the future. Moreover, though we implemented this extra level of affordance planning based on a sampling-based approach, it can be utilized by optimization-based approaches.

Affordance planning is not without limitations. First, affordance planning is designed to handle affordance constraints imposed by the environments, but not constraints imposed by the robot. This extra planning will not improve performance if the scenario is challenging due to robot constraints or a mixture of environment and robot constraints. Second, our current affordance planner is designed based on pose

		Run Time (s)		Success Rate(%)	
		TAAMP (ours)	**TAMP**	**TAAMP (ours)**	**TAMP**
Unconstrained Tasks	**Manipulation**	4.15 ± 0.43	1.66 ± 0.20	100.00	100.00
	Tool-Use	7.02 ± 1.21	14.91 ± 5.67	100.00	100.00
Constrained Tasks I	**Manipulation**	30.82 ± 1.00	493.60 ± 201.91	100.00	30.00
	Tool-Use	16.83 ± 6.52	444.65 ± 227.44	100.00	43.33
Constrained Tasks II	**Manipulation**	68.42 ± 3.86	600.00 ± 0.00	100.00	0.00
	Tool-Use	67.26 ± 44.46	600.00 ± 0.00	100.00	0.00
Infeasible Tasks	**Manipulation**	0.28 ± 0.03	600.00 ± 0.00	100.00	0.00
	Tool-Use	6.97 ± 0.82	600.00 ± 0.00	100.00	0.00

Table 4.1: Results In Simulation. Each cell shows the average or the percentage of thirty trials in the given condition. The time out was set to be 600 seconds. The numbers in the run time cells followed the format of M ± SD.

changes, and improvements can be made to include more state changes of the objects (e.g., color). Third, the algorithm in affordance planning can be designed to be more efficient, or may even include a learning mechanism. These changes will help to increase the benefit of TAAMP in future studies.

Though TAAMP is a good solution as a way of eliminating possible options, we acknowledge that TAMP may be more efficient in unconstrained tasks where it is guaranteed that there is a solution since the solution space is large. The insufficiency of TAAMP in these unconstrained tasks is due to the extra affordance tests. To solve this, one may simultaneously execute both TAMP and TAAMP using multi-threading. The script terminates when either planner returns a result. In this way, both unconstrained tasks and constrained tasks will benefit from the best of the two worlds.

4.5 Summary

In this chapter, we present the TAAMP algorithm, which includes an extra intermediate layer of affordance planning compared with the classic TAMP algorithm. It benefits both general manipulation tasks and tool use tasks, specifically sequential tool use. It may also have the potential to facilitate other types of multiple-manipulation tool use, such as planning for tool manufacturing. While this chapter focuses on sequential tool use, the next chapter focus on another subcategory of multiple-manipulation tool use, tool selection.

Chapter 5

Multiple-Manipulation Tool Use: Tool Selection[1]

In this chapter, we present work that aims to handle part of open challenge 4, which is tool selection. We introduce a method for a robot to reason about tool affordance with an explicit model of cause-and-effect by constructing a structural causal model through a mix of observation and self-supervised experimentation. We demonstrate our method on tool selection tasks, and results suggest that after minimal training examples, our system can preferentially choose new tools based on the context and use these tools for goal-directed object manipulation.

5.1 Introduction

In *The Crow and the Pitcher*, the fifth century B.C.E. Greek poet Aesop wrote of a thirsty crow who ingeniously found relief after dropping stones into a jug of water until the water level was high enough for the crow to drink from it. More recently, the scientific community corroborated this remarkable feat of physics-based instrumen-

[1]Portions of this chapter were originally published as: J. Brawer, M. Qin, and B. Scassellati. A causal approach to tool affordance learning. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8394-8399. IEEE, 2020. (Brawer et al., 2020)

tal problem-solving in New Caledonian Crows (NCC; *Corvus moneduloides*) (Jelbert et al., 2014). Contrary to the flexible reasoning capabilities of crows and other higher species, robot learning and reasoning remain starkly limited despite substantive advancements in statistical machine learning techniques. Additionally, what exactly is learned by these systems remains largely opaque (Marcus, 2018), which not only makes them difficult to debug but poses serious risks in robotics settings where unforeseen behaviors can cause physical harm.

Ideally, robots should be able to acquire knowledge and skills in a way that is both transparent and portable across contexts, but without compromising on the incredible advancements made by the machine learning community. To that end, we have developed a system whereby a robot can learn and exploit a graphical causal model to solve a physical reasoning task. Our approach has a robot learning a structural causal model (SCM) (Pearl, 2000) through a mix of observation and physical exploration. The SCM is used to perform causal inference, which is completed by a group of neural networks that are dynamically constructed and trained as a function of the learned structure of the SCM and the goals of the current task. As a result, our system represents the robot's knowledge in an explicit and explainable way by the directed acyclic graph (DAG) entailed by the SCM, but that also leverages the powerful pattern recognition capabilities of machine learning techniques via the use of neural networks.

We demonstrate our method on a humanoid robot that must build a model of the cause-and-effect relationships underlying tool-assisted manipulation and then use this model to both solve goal-directed manipulation tasks, and quickly learn the affordances of novel tools given a kinematic model of each tool.

We demonstrate that our system is capable of selecting the appropriate tool and action to take to move an arbitrarily placed block into goal regions, as well as leveraging prior learning to bootstrap learning of new tools.

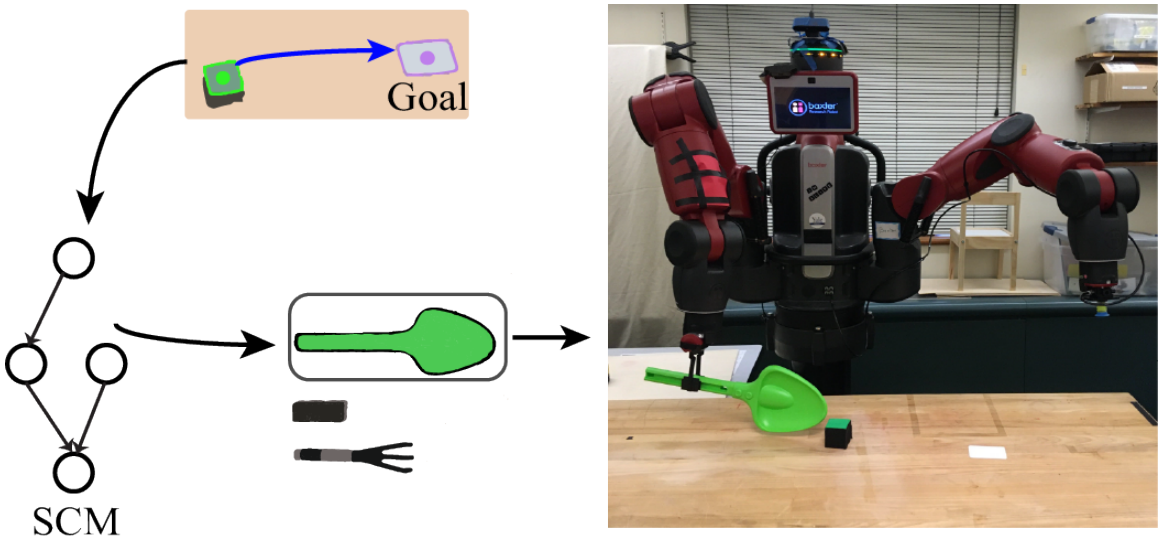


Figure 5.1: Tool selection via causal inference. A Baxter collaborative robot queries a learned causal model of tool-assisted manipulation using perceptual information from its workspace. Information from the graph is used to select the most appropriate tool for completing its goal.

In sum, our contributions are the following:

- A method for a robot to construct a transparent model of causation via a mix of observation and experimental learning.
- A method for performing forward and backward causal inference using a series of dynamically constructed neural networks.
- A method for a humanoid robot using learned causal models to quickly learn and utilize tool affordances of novel tools.

5.1.1 Causality in ML and Robotics

Many researchers have attempted to formalize causal relations over the past century. Here we focus on Pearl's SCM framework to formalize causal relations using directed acyclic graphs (DAGs) that define structural equations between causal variables (Pearl, 2000). Pearl argues that SCMs accommodate sophisticated forms of reasoning, including interventional (e.g., "What if I had done X?") and counterfactual

(e.g., "What if I had acted differently?") (Pearl, 2018). Counterfactual reasoning has been integrated into RL systems; for example, it has been shown to enable the acquisition of effective decentralized multi-agent policies in a credit assignment task (Foerster et al., 2018), and aid in the evaluation of high-risk healthcare policies (Oberst and Sontag, 2019).

SCM-based reasoning has been employed in a robot by Angelov et al. (2019) which used learned SCMs to counterfactually reason about user preferences in their demonstrations of a motion task. Non-SCM based approached to causal reasoning include Xiong et al. (2016), which integrated spatial, temporal, and causal and-or graphs learned from user demonstrations to enable a robot to fold clothing. Nevertheless, causal learning and reasoning for robots remains relatively unexplored despite the active developments from the machine learning and artificial intelligence communities.

5.1.2 Affordance learning

While Bayesian networks have been a popular method for representing affordances (see Andries et al. (2018) for a recent example), an explicitly causal framework is rarely employed despite centrality of cause and effect within many affordance paradigms. As frameworks like SCMs can tease apart causal, and not merely correlational, properties in the environment, a robot equipped with such a framework is better poised to model the effects of its actions on the world, and thus to acquire and utilize affordances. We demonstrate this capability by showing that our system can effectively complete goal-directed tool and action-selection tasks using acquired affordance knowledge.

5.2 Methods

5.2.1 Problem Statement

Our goal is to have a robot model simple tool use behaviors using SCMs, and then subsequently use this model to quickly ascertain the affordances of novel tools. We begin by giving a brief overview of the SCM formalism (refer to Pearl (2000) for more technical details).

The problem of learning an SCM is twofold: the structure of the graph which links variables to other variables – or causes to effects– must be learned, as well as the functional mechanism that formalizes these relationships. Formally, an SCM $\mathcal{M}$ is a tuple $\{\mathcal{U}, \mathcal{V}, \mathcal{F}\}$ where $\mathcal{U}$ is a set of unobserved background features or "exogenous" variables, $\mathcal{V}$ is a set of observed features or "endogenous" variables, and $\mathcal{F}$ is a set of functions that assigns values to variables in $\mathcal{V}$ based on other variables in $\mathcal{U}$ and $\mathcal{V}$. Under the assumption that the causal structure is acyclic, there is a corresponding DAG $\mathcal{G}$ where the nodes in the graph correspond to variables in $\mathcal{U}$ and $\mathcal{V}$ and the edges to the functions in $\mathcal{F}$.

Kocaoglu et al. (2017) demonstrated that feedforward neural networks admit an interpretation as SCMs. We take advantage of this fact by using the causal structure uncovered during learning to guide the construction of series of neural networks that define the structural equations $\mathcal{F}$ underlying the SCM. The advantage of using such a scheme is that it minimizes the assumptions of the generating distributions of the variables of interest, which in many potential real-world scenarios are likely unknown a priori. Exogenous variables $\mathcal{U}$ are represented by the hidden nodes in the neural network.

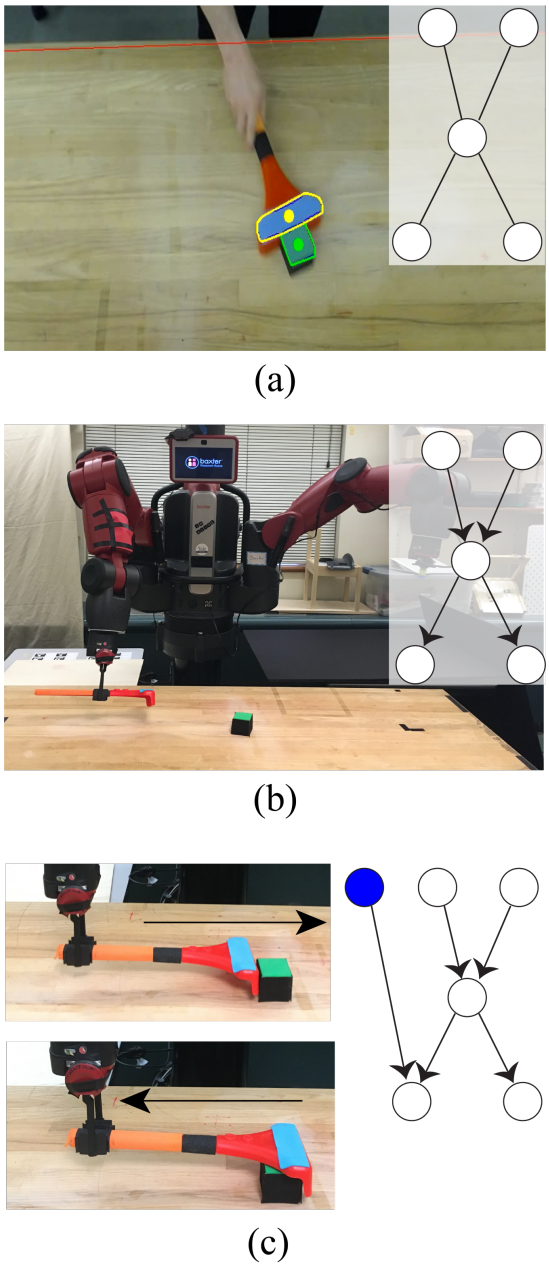


Figure 5.2: The causal discovery process. During the observation phase (a) the robot learns a skeleton of the causal graph observing demonstrations performed by a human. During the validation phase (b) the robot attempts to orient the edges of the graph via self-supervised experimentation. Finally, during the augmentation phase (c), the robot introduces a new node (blue) and attempts to incorporate it into its graph via further experimentation.

5.2.2 Our approach

Causal Learning

Causal learning has three phases: the **observational phase** to the learn an unoriented skeleton of the causal graph, the **validation phase** to oriented the edges of the graph, and the **augmentation phase** to augment the graph with new nodes via experimentation. The goal of the observational phase is to bootstrap learning of causal structure by taking advantage of the fact that it can be inferred (usually within an equivalence class of DAGs under reasonable assumptions of causation, see Eberhardt (2017) for an overview of causal discovery) from passively collected, observational data using standard structural learning algorithms. This helps minimize the amount of interventional data required to fully specify the graph, which may be beneficial, as often collecting this sort of data is difficult to collect for reasons of practicality. The result of this phase is an undirected graph with edges drawn between causally dependent nodes, though the direction of causation may not be known.

Subsequently, the robot enters the self-supervised validation phase, wherein it attempts to orient the graph by collecting interventional data. An edge between nodes V_1 and V_2 is oriented $V_1 \rightarrow V_2$ if $P(V_2|do(V_1 = x)) \neq P(V_2|do(V_1 = y))$ for some interventions x and y, and vice-versa. As interventional samples can be difficult or costly to obtain, it is desirable to be able to preferentially select nodes to intervene on such that the total number of experiments is minimized. To that end we take an active learning approach adapted from Hauser and Bühlmann (2014), which simply intervenes on nodes in the graph produced during the observational phase starting with the nodes with the highest degree, ensuring that the most informative interventions are carried out first.

Finally, in the augmentation phase, the robot attempts to incorporate a new node V_3 into its causal model. Unlike in the validation phase, the goal of the augmentation

phase is not orienting edges that already exist, but rather to add new oriented edges where none had existed previously. An edge between a new node V_3 and an extant node V_1 is added to the model if it is determined that intervening on V_3 effects the value of V_1 using the method described above.

In order to minimize the number of edges to be tested, and thus the chances of falsely identifying causal relationships, we developed a few heuristics for selectively testing nodes. First, nodes are tested in topological sorted order. This ensures that the causal antecedents that have thus far been identified are tested first. In addition, we make the following assumption: if an edge $V_1 \rightarrow V_2$ is found for some nodes V_1, V_2, then we do not test any descendants of V_2. While this has the potential to produce an incomplete causal graph, it avoids the possibility of mistaking indirect causation from $V_1 \rightarrow V_2 \rightarrow V_3$ for some descendant V_3 of V_2, for the direct causal connection $V_1 \rightarrow V_3$.

Causal Inference

Once the causal structure is in place, the functional mechanisms underlying the SCM can be learned, allowing for the robot to perform causal inference. As we wish to minimize our assumptions about these underlying mechanisms, we use neural networks to estimate these functions, using the structure of the graph to guide the structure of the neural network architectures. However, while this allows for reasoning from causes to effects, it is not immediately clear how other forms of causal inference, e.g., diagnostic or 'abductive' reasoning from effects to causes, can be performed. Similarly, in many real-world scenarios, it is often the case that observations have been made for only a subset of variables in the SCM, and it may be desirable to estimate the unknown values using known information.

In order to support these capabilities, the final set of neural network architectures are dependent on not only the causal graph structure G, but the observed and unob-

served variables as well. We treat these unobserved values as a set of "queries" Q, the values of which we would like to estimate. Each node $q \in Q$ is ranked with a "causal score" that is used to ultimately guide the order of inference, as well the construction of the neural networks via a recursive algorithm (refer to algorithm 1). A causal score is a value between 0 and 1, computed as the ratio of the number of q adjacent nodes with observed values to the total number of q adjacent nodes. If the values of all of q's parents are observed, it is automatically assigned a score of 1, as its value can be estimated by standard means. Once the scores are computed, the value of the node with the highest score is inferred, and it is treated as an observed variable, and so can be used to infer the values of other nodes. This process of scoring and inference is repeated until all nodes have been estimated.

Inferring the q values was accomplished using feedforward neural networks, though the structure of these networks may not match the forward structure of the corresponding causal graph in cases where abductive inference is required. We use two heuristics to construct each network f; 1) if all of q's parents are observed, then $q = f(Pa(q))$, where $Pa(X)$ denotes the parent nodes of node X; 2) otherwise, $q = f(V_{observed} \cup V_{collider})$, where $V_{observed}$ is the subset of nodes adjacent to q that have been observed, and $V_{collider}$ is the possibly empty subset of nodes that belong to a collider subgraph with q, as these dependencies may carry useful information about q's state. Thus the size of the input and output layers of any neural network was equal to the number of nodes in $V_{observed} \cup V_{collider}$ and the number of nodes in q (i.e. 1), respectively. In practice, we found that a single hidden layer consisting of 10 nodes worked well across all networks.

Affordance learning

Andries et al. (2018) identify equivalences between affordances based on how similar their effects are on an object when acted upon. We employ a similar approach to

Algorithm 1 Recursive procedure for causal inference.

procedure ANSWERQUERIES(Q, G)
  if Q is empty **then return**
  else
    $(q, score) = max(causalScores(G, Q))$
    if $score == 1.0$ **then**
      $Input = Pa(q)$
    else
      $Input = V_{observed} \cup V_{colliders}$
    $\hat{q} = f(Input)$
    Append $\hat{q}$ to $G.obs$
    $AnswerQueries(Q \setminus q, G)$

learn and represent tool affordances. For some tool t and set of possible actions A, we represent its affordances as a vector $\mathbf{a^t} \in \mathbb{R}^{|A|}$, where each element a_i^t of the vector corresponds to a measure of how well the effects of a tool action align with predictions (here we use the coefficient of determination, r^2). Using a learned causal model of an exemplary tool, a robot can quickly estimate these vectors from a small number of exploratory actions on an object and use these estimates for tool selection and usage given some goal position for the target object.

5.2.3 Experiments

In this section we discuss the design and implementation details of the task used to evaluate our model.

The Task

We would like the robot to 1) model the causal process underlying manipulating objects with tools, and 2) leverage the learned model to more effectively learn and wield new tools. Here we take as examples pushing and pulling as classes of actions the robot can perform. We assume the robot can observe the random variables corresponding to the initial position and final position of a manipulated block, p_{init}

and p_{final}, respectively, as well as the movement vector d, which is defined as the vector that the tool tip travels starting from a small displacement before or after the block to the terminus of the push action. While $p_{init}, p_{final}, d \in \mathbb{R}^2$, we represent each of their dimensions by their own individual nodes, which we express with a superscript, for example, p^x_{init} and p^y_{init} For convenience, when we omit the superscript, we are referring to both dimensions simultaneously. Additionally, while $d \in \mathbb{R}^2$, for simplicity, we limit the push vectors the robot can enact to 12 evenly spaced vectors around the unit circle. This requires that for any estimated value of d, the robot must choose one of these 12 vectors with the smallest angular distance to d to enact. The robot can take actions $push, pull \in A$ in directions d, though does not at the outset know the relationship between these two, or any other variables. Consequently, during inference, d and A are estimated using classifier networks,while p_{init} and p_{final} are estimated using regressor networks.

While it is assumed the robot knows how to perform simple pushes and pulls with each tool, it does not know the effects that tool use entails, nor in what scenarios a given tool is appropriate. While the robot only needs to learn the causal relationships among these 7 variables, the number of possible DAGs is super-exponential in the number of variables (Robinson, 1973), for a total of approximately 1.1×10^9 possible DAG structures, making this structural identification problem non-trivial.

In addition, there are a number of details related to the implementation of three learning phases that are specific to this particular task, outlined below.

Observational phase: The robot observed demonstrations of a block being pushed with a hoe tool (see Figure 5.3) by a human, and tracks p_{init}, p_{final}, and d. This observational data is standardized and passed to a structural learning algorithm in order to get an initial hypothesis of the causal structure. For the purposes of this experiment we use the PC algorithm, a widely used score-based method for causal discovery (Spirtes et al., 2000).

Validation phase: We allow the robot to directly intervene on p_{intial}, p_{final} and d, and limit it only to pushes with the hoe from the previous phase. Each intervention is treated as a randomized controlled trial, where the intervened variable is forced to take on one of two values. Interventions on p_{init} and p_{final} compare interventional distributions for two prespecified positions, where as interventions on d compares the distribution induced by taking a random action vs taking no action at all. As we wish to limit the assumptions we make about the underlying distributions of the variables, we use the Kolmogrov-Smirnoff test to nonparametrically estimate difference between the two interventional distributions. Given that these interventions are used to quickly infer causal relations, and are not themselves rigorous scientific experiments, the $p < .05$ significance convention need not be followed. Here we relax the threshold of significance to $p < .2$, though this may be treated as a hyperparameter which trades-off risk of type I errors for data-efficient estimation.

Augmentation phase: During the augmentation phase, the robot attempts to add the action type A to its causal model. This proceeds in much the same way as the validation phase, except we allow the robot to perform pulls as well as pushes.

Affordance-based tool selection

We would like the robot to choose the best tool, $\hat{t}$, and tool usage $\hat{d}$ given the circumstances of the environment. Using desired movement vector d^*, obtained by querying the causal graph, together with the estimated affordance information $a_x^{t_i}$ for a given tool t_i, the robot can make this selection. This is captured by the heuristic

$$\hat{t}, \hat{d} = \underset{t_i, d_{t_i}}{\arg\min} (2 - a_x^{t_i})(2 - \cos\theta_{d_{t_i} d^*}) \tag{1}$$

Here d_{t_i} is a movement direction the robot is capable of actuating with tool t_i according to its kinematic model. In essence, this heuristic chooses a tool based on

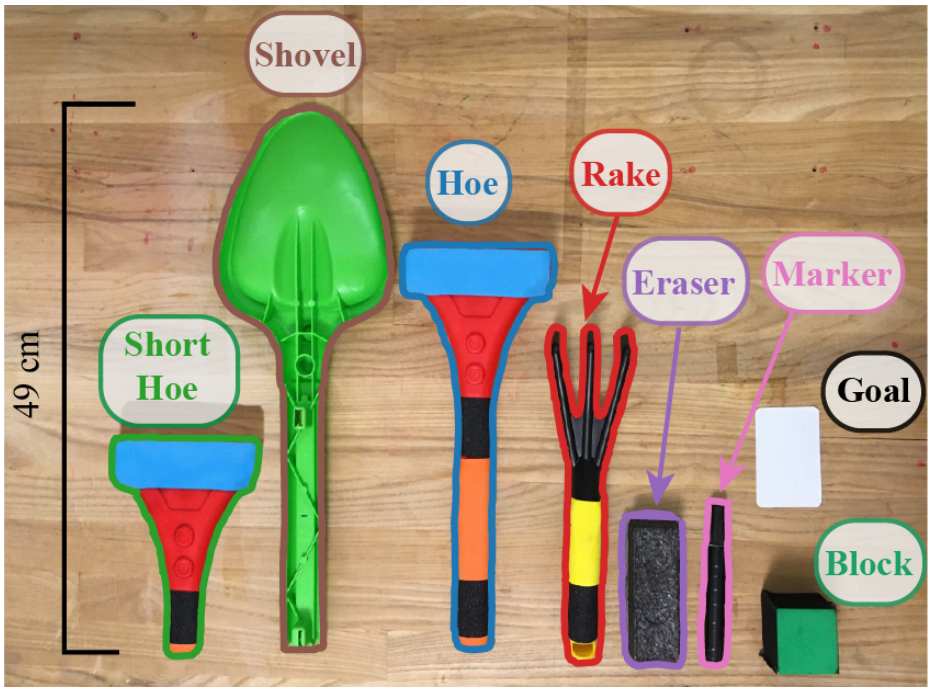


Figure 5.3: The tool set.

the trade-off between how well it affords a desired action in general, and how well it can enact the action in this specific instance.

The workspace

Experiments were performed on a Baxter collaborative robot (Fig 5.1)[2]. Our Baxter model was equipped with a claw gripper for manipulating the tools, as well as a suction gripper for picking and placing the block during the self-supervised learning phases. Informal tests suggested our model's end-effector precision was within $\pm 1cm$. Figure 5.3 depicts the contents of the robot's work space. Actions were performed on a $5cm^3$ wooden block. Initially the robot had access only to a hoe. Additionally, there were 5 morphologically distinct tools the robot's learning was evaluated on. During the evaluation phase, the robot was tasked with pushing the block into a small rectangular goal region measuring approximately, $10cm \times 8.5cm$. The positions of the tool tips, the block, and goal region were tracked with a head-mounted RGB

[2]Source code can be found at https://github.com/ScazLab/crow

webcam using a blob detector calibrated to the colors unique to each object. We used the Causal Discovery Toolbox's (Kalainathan and Goudet, 2019) implementation of the PC algorithm, as well as classifier and regressor neural network implementations from Scikit-learn (Pedregosa et al., 2011).

5.2.4 Evaluation

With our experiments we wished to see how well a robot could use a learned causal model to perform affordance-based reasoning for familiar and unfamiliar tools. In order to do this, we ran three evaluations: 1) Given 8 samples with each of the 5 novel tools, we had the robot choose both the best tool and action to perform given a single fixed goal location, but arbitrary initial positions of the block; 2) given 20 training samples per tool, multiple goal regions and multiple initial block positions, we observed how close could the robot move the block towards the goal region for each tool; 3) we looked at how accurately the robot could predict action effects as a function of training data samples for a subset of the 6 tools.

5.3 Results

Figure 5.4 depicts the learned causal graph. The structure of this graph models two important aspects of the task: 1) the final position of the block is a function of the initial position of the block and the direction in which it is pushed, and 2) the type of action, push or pull, effects the push direction, suggesting the robot has successfully grounded these abstract actions to a concrete sensorimotor effect. The observation phase consisted of 60 sample demonstrations of a human pushing a block. During the validation and augmentation phases, we limited interventions to 20 samples per intervention. During the validation phase the robot performed two interventions, for a total of 40 samples. The augmentation phase consisting of one intervention on

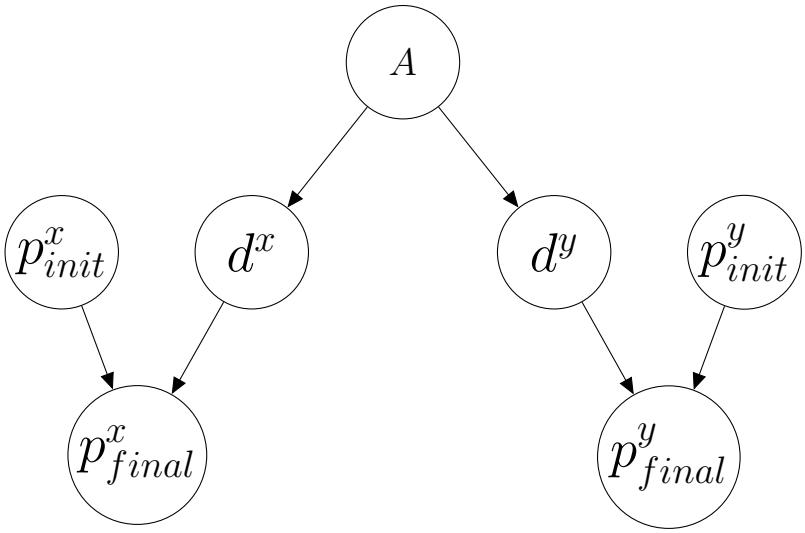
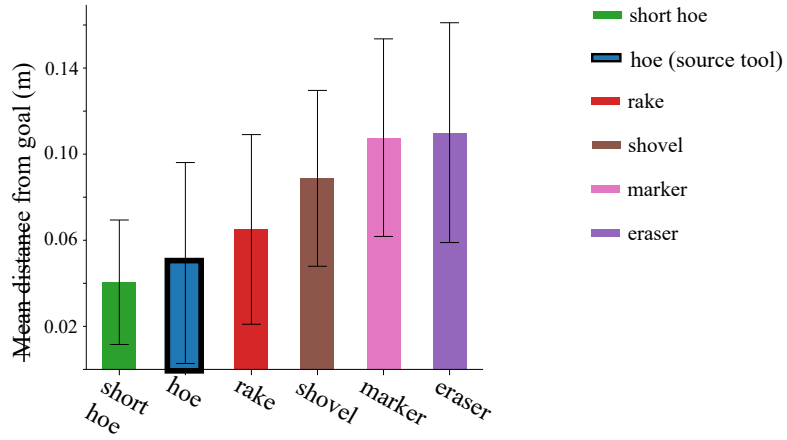


Figure 5.4: The learned structure of the SCM.

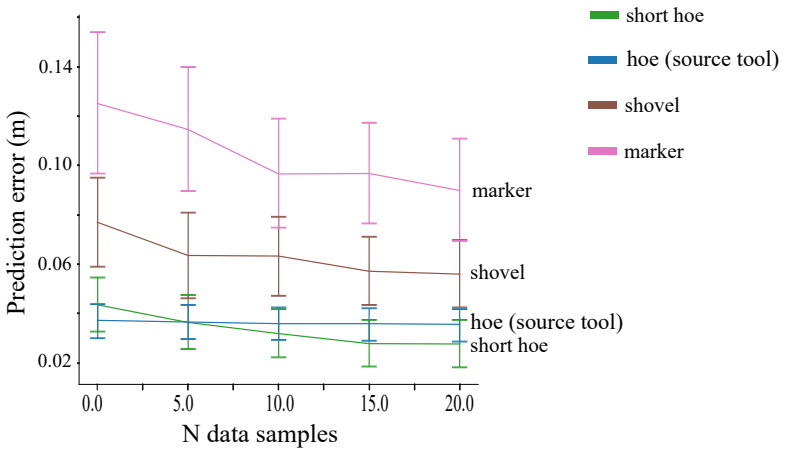
action type, consisted of 20 samples.

Figure 5.5(a) depicts the robot's performance with each tool across 5 goal locations with 5 randomly distributed initial positions per goal location. Remarkably, the top 3 best performing tools, short hoe ($M = 0.04, SD = 0.02$), the hoe ($M = 0.5, SD = 0.04$), and the rake ($M = 0.06, SD = 0.04$), came within a few centimeters of the goal on average, despite relative inaccuracy of the robot and the limited set of movement directions at the robot's disposal. The tool performance roughly track with morphological similarity with the source tool (i.e., the hoe), with the most similar tools producing the best results (refer to Figure 5.3). The learning curve results depicted in Figure 5.3(b) tell a similar story, as the tools that are morphologically most similar to the source tool, and to a lesser extent, the shovel – benefit much more from the prior learning than the marker. Interestingly, the short hoe ultimately performs better than the source tool, though this is likely due to the fact that the short hoe is physically the same tool as the source tool, but just gripped closer to the tool tip, producing more consistent pushes and pulls.

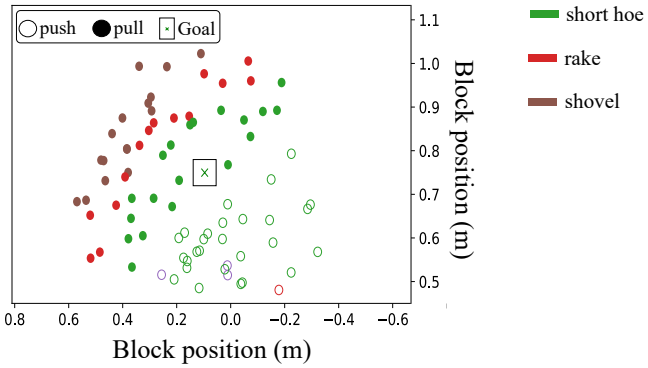
Figure 5.5(c) depicts how our system chose tools and actions as a function of the object's initial position given a fixed goal location. Here we see action selection in line



(a)



(b)



(c)

Figure 5.5: Tool reasoning results. a) depicts the mean distance of the center of the block to the center of the goal region for each tool. b) depicts the learning curve for a select number of tools given initial training on the hoe. c) depicts how tools were selected and used as a function of the block's position given a static goal.

with expectations; the robot chose to push the block when the block was positioned before the goal relative to the robot, and pulled it when it was positioned beyond the goal, further supporting the notion that the system has meaningfully grounded the push and pull actions to sensorimotor effects. Figure 5.5(a) also helps shed light on the tools choices in (c). The robot overwhelmingly preferred the short hoe for pushing and pulling, which makes sense as it is the tool the robot uses most effectively. On the periphery of the workspace, the robot begins choosing the rake for pulling, as the rake is both slightly longer than the hoe, and also the robot's next best tool. The same reasoning for the choice of shovel on the edges of the workspace; it is the longest tool available to the robot and thus the only one capable of completing the desired pull.

5.4 Discussion

In this work, we demonstrated a method for a robot to learn and utilize a causal structural model to rapidly acquire and reason over tool affordances. The results of our experiments suggest that the grounding of actions to effects via the learned causal model enabled the robot to effectively select and use tools preferentially based on the conditions of the workspace.

We believe there are two primary advantages of this method over more common approaches to robot learning: 1) The knowledge acquired through the learning process is explicitly represented and hierarchically organized by virtue of the DAG structure of the SCM; 2) By leveraging this same DAG structure to construct neural networks, the functional mechanisms of the SCM can be learned in a relatively data-efficient way. That is, even in a dense causal network, for a given node of interest, only a subset of the nodes are required to infer the node's value, mitigating the effects of the curse of dimensionality that often plagues machine learning systems. The transparency

of knowledge entailed in 1) is vital in a robotics context, for example in human–robot collaborative contexts where shared expectations amongst collaborators has been demonstrated to be important for team success (Hayes and Scassellati, 2013).

Nevertheless, there were some limitations to this work. There were relatively few causal variables under consideration, and all of them were assumed to be observable. Currently, it is not clear how well our method would scale to learning larger, more complex causal graphs or graphs with latent causal variables. In addition, aspects of the interventional experiments conducted by the robot, including the two initial positions of the block, were hard-coded. Ideally, the robot should be able to autonomously generate its own experiments and choose values to force the interventional variables to take on. This is an important problem, as depending on the generating distribution underlying the model, some interventions may be more informative than others for uncovering causal relationships.

5.5 Summary

In this chapter, we present our work that allows a robot to perform tool selection. While this and the previous chapters focus on tool use, we will next illustrate how tool use knowledge can benefit tasks beyond tool use.

Chapter 6

Applying Tool Use Knowledge in the Context of Human-Robot Collaboration[1]

In this chapter, we turn our attention toward applying learned tool use skills in collaborative human-robot tasks to handle open challenge 6 as described in Chapter 2. Specifically, we address the problem of how a a robot should hand over a tool to a human. We identified different types of handovers and focus on one type of handovers, task-oriented handovers, which incorporate information about subsequent tool use tasks. We identify multiple difficulty levels of task-oriented handovers, and demonstrate that our method can adapt to all difficulty levels, including tasks that match the typical usage of the tool (level I), tasks that require an improvised and unusual usage of the tool (level II), and tasks where the handover is adapted to the pose of a manipulandum (level III). We evaluate the generated handovers with online surveys. Participants rated our handovers to appear more comfortable for

[1]Portions of this chapter were originally published as: M. Qin, J. Brawer, and B. Scassellati. Task-Oriented Robot-to-Human Handovers in Collaborative Tool-Use Tasks. In *Proceedings of the 31st IEEE International Symposium on Robot and Human Interactive Communication*, RO-MAN '22, Naples, Italy, 2022. ACM. (Qin et al., 2022a)

the human receiver and more appropriate for subsequent tasks when compared with typical handovers from prior work.

6.1 Introduction

A robot-to-human handover is a joint action wherein a robot grasps, presents, and transfers an object held in its end-effector to a human receiver. It is a common exercise in numerous applications, including service robots handing flyers to pedestrians (Shi et al., 2013), personal assistive robots handing phones to people with disabilities (Choi et al., 2009), and factory robots handing hammers to collaborators (Koene et al., 2014a). To summarize the different requirements for handovers, we compiled a robot-to-human handover taxonomy (Section 6.1.1). The taxonomy serves the following purposes: 1) it helps to situate our study in the larger picture of robot-to-human handovers; 2) it helps to organize related work on handovers; 3) it may serve as a guide for future systems designed for handovers in terms of what requirements may need to be considered.

This study focused on one specific handover, the *task-oriented handover* that is commonly seen in the context of human-robot collaboration (HRC). A task-oriented handover requires a robot to grasp and present tools in a way that incorporate information about the tasks to be performed by the human receiver. In HRC, the purpose of a task-oriented handover typically is not merely to pass an object to a human, but also to enable the human to use the object to complete tasks. In order to maximize efficiency, the task-oriented handover should allow the human receiver to initiate a subsequent task with minimum in-hand object adjustment. Consequently, handovers of this type are dependent on how the tools should be used. Previous studies on task-oriented handovers generally demonstrated handovers of certain tools, without providing information regarding how the tools are used in the subsequent tasks. How-

ever, as mentioned in recent publications (Ortenzi et al., 2019, 2021), task-oriented handovers have not yet gained enough attention in robot manipulations. As a result, robots' lack of understanding of tool use impedes their ability to generate handovers with novel tools. Therefore, our study aimed to design a system that can generate appropriate task-oriented handovers with from demonstrations of tool use rather than handovers by integrating existing techniques. Furthermore, we also identified multiple levels of difficulties in task-oriented handovers and organized related work accordingly (Section 6.1.2).

We built a system that generates task-oriented handovers. The system learned tool affordances to allow the robot to understand the nature of the subsequent task. In our system, we chose and integrated a tool-affordance learning technique appropriate for handover tasks on a physical robot. We also conducted an online survey to evaluate the handovers executed by the robot. In summary, our contributions are:

1. We defined a taxonomy of handover requirements.

2. Our system generated handovers based on learned tool affordances, rather than handover demonstrations, since task-oriented handovers are dependent on the subsequent tool-use task.

3. With the understanding of how tools should be used, our system was able to handle task-oriented handovers for all three difficulty levels that we identified.

4. Survey participants preferred our handovers and rated them as appearing to be comfortable for a human receiver and appropriate for the subsequent tasks.

6.1.1 Taxonomy: Handover Requirements

A handover is a complex manipulation with various requirements to satisfy. Therefore, we compiled a taxonomy of handover requirements and summarized it in Figure 6.1.

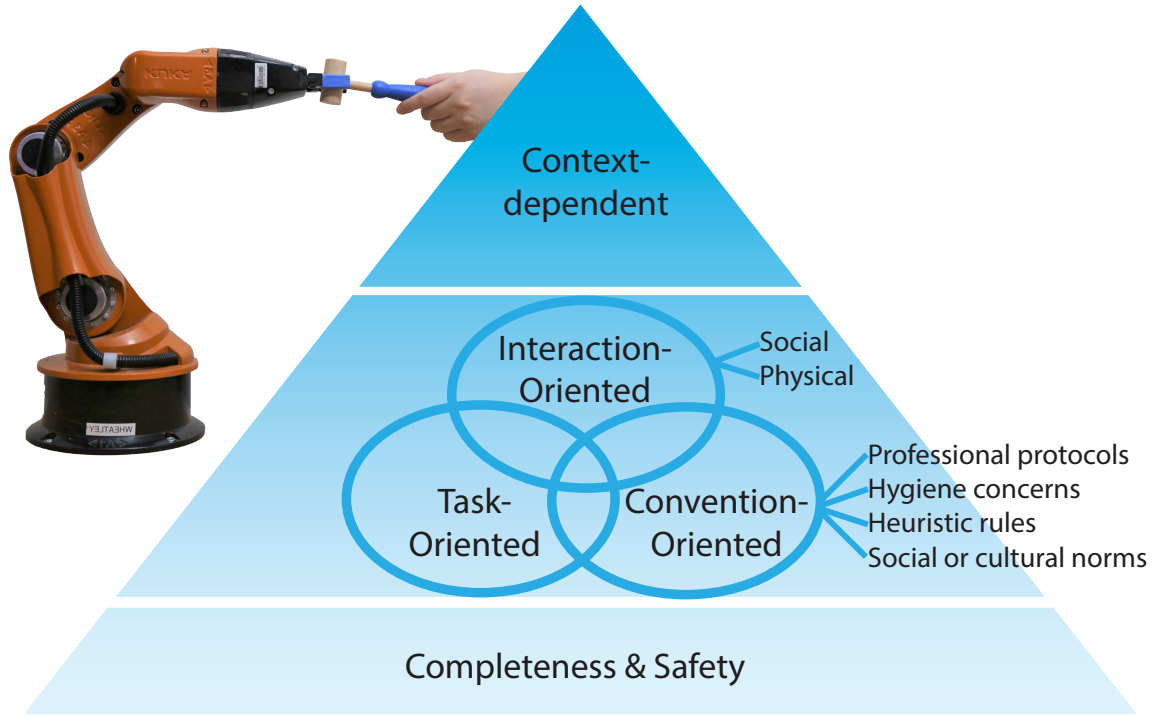


Figure 6.1: Our taxonomy of robot-to-human handover requirements. Bottom to top: the basic, intermediate and advanced requirements.

The requirements at the lower levels should be satisfied first before a higher-level requirement is satisfied. In the taxonomy, the basic requirement is to be complete and safe. A complete handover refers to the successful delivery of an object to the receiver (Huber et al., 2008; Koene et al., 2014b; Chan et al., 2012; Eguiluz et al., 2017; Chan et al., 2013; Konstantinova et al., 2017; Neranon, 2018; Parastegari et al., 2016), and a safe handover requires that no collision with the robot occurs at any time during the course of delivery (Prada et al., 2014; Maeda et al., 2017; Sisbot et al., 2010; Mainprice et al., 2010). This is the focus of most handover studies.

Beyond the basic requirement of completeness and safety, satisfying one or more intermediate requirements will produce appropriate handovers. Compared with the studies focused on basic requirements, fewer handover studies focus on intermediate requirements.

The first intermediate requirement is that handovers should adapt to social or

physical interactions between a human receiver and a robot (i.e., interaction-oriented). The social interactions include sending or perceiving various types of social signals such as eye contact (Strabala et al., 2013; Admoni et al., 2014; Grigore et al., 2013; Cakmak et al., 2011), while the physical interactions involve adjusting where (Aleotti et al., 2012; Koay et al., 2007) or when (Huang et al., 2015) to conduct handovers based on the location or the physical state (e.g., availability) of a human receiver, or generating handovers that comply with human ergonomic needs (Peternel et al., 2017; Bestick et al., 2015). Satisfying these interaction–oriented requirements can help with generating customized handovers that are more comfortable for the receiver.

The second intermediate requirement is that a handover should abide by various conventions (i.e., convention-oriented), including professional protocols (e.g., handing over a surgical tool to a surgeon during a procedure in the operating room), hygiene concerns (e.g., one should not grasp the tines of a fork which will touch food), heuristic rules (e.g., one tend to orient an object horizontally for the receiver), and social or cultural norms (e.g., handing over a gift with a single hand is considered disrespectful). Satisfying convention-oriented requirements can help with generating handovers that match expectations.

The third intermediate requirement is that handovers should incorporate information about subsequent tasks (i.e., task-oriented) (Chan et al., 2014, 2020; Bestick et al., 2016; Ortenzi et al., 2019), which allows the human receiver to perform the subsequent tasks more efficiently. *Our study focuses on this third intermediate requirement, task-oriented handovers, and other requirements are beyond the scope of this study.*

The advanced requirement in our taxonomy is that a handover should be context-dependent. In other words, one should choose one or a combination of intermediate requirements to meet based on the specific context. The intermediate requirements may contradict each other, such that not all requirements can be satisfied simultane-

ously. For example, during a convocation, an assistant hands the diploma to a dean in a way that prioritizes the interaction-oriented requirements so that the dean can receive the diploma more comfortably. However, when the dean hands the diploma to a graduate, the dean will not prioritize the interaction-oriented requirements as the assistant does, but will prioritize the convention-oriented requirements and use both hands to show respect. Therefore, a robot needs to recognize which intermediate requirements are important in the given context and choose one or a combination of intermediate requirements to meet the given context.

6.1.2 Task-oriented Handovers

We identified three levels of difficulties in task-oriented handovers and organized related work on task-oriented handovers accordingly. Figure 6.2 summarizes the difficulty levels and shows examples of each level. Level I is to properly hand over a tool to a human to perform a task typically matched with the tool (e.g., using a screwdriver to drive screws). Since a tool usually has a default usage, level I handovers could be achieved by building or learning a dataset to store handovers (Chan et al., 2020; Bestick et al., 2016; Ortenzi et al., 2019), assuming the dataset was learned with handover demonstrations rather than tool-use demonstrations.

In level II task-oriented handovers, a human receiver may use tools with their default usages, but may also improvise tool-use for tasks not generally associated with the tools (e.g., using a screwdriver to play a xylophone rather than to drive a screw). It is more challenging than level I because a pre-built dataset that can handle level I handovers may not be able to handle level II handovers due to the nearly limitless ways any particular tool can be used in different tasks. More importantly, the dataset may not be able to generalize to level II handovers due to a lack of understanding of how the tools should be used. To realize handovers at this level, a robot should recognize the functional segment of the tool and understand the usage

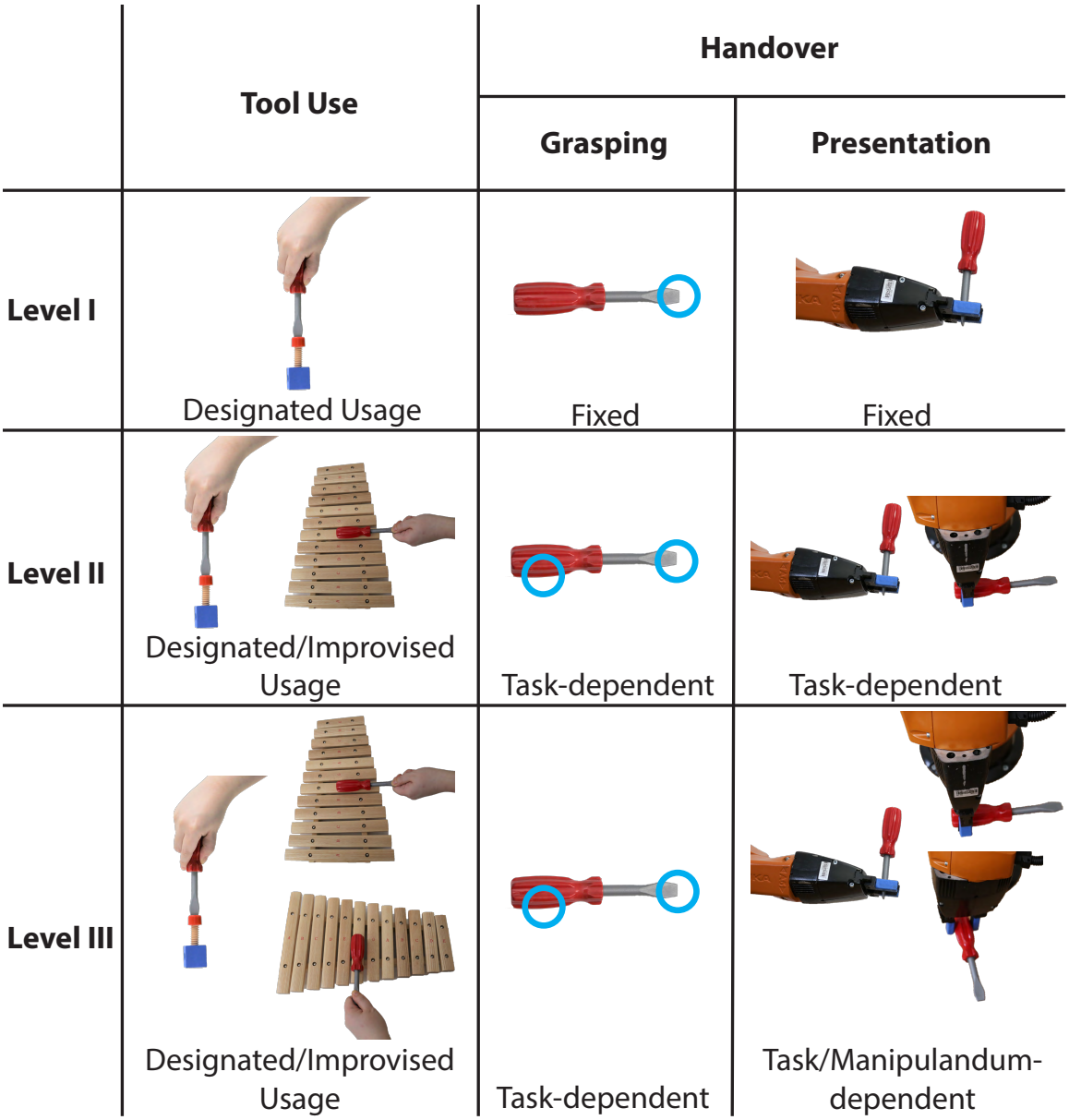


Figure 6.2: Difficulty levels of task-oriented handovers.

to determine the handovers. In other words, learning *tool affordance* is the key to achieving level II task-oriented handover. To our knowledge, only one previous study considered learning tool affordances before performing handovers (Chan et al., 2014). Although only level I handovers were demonstrated, their system may be capable of level II handovers. However, the design of this previous study makes their system impractical to be applied in many HRC scenarios. In that study, a human needed to

demonstrate the usage of the novel tool to the robot in order to determine relevant handover configurations. However, a novel tool to be handed over is generally out of reach of the human receiver, so that a demonstration may be impossible without handing over the tool in the first place.

In addition to level II handover constraints, a robot should adjust the handover configurations based on the pose of the manipulandum (i.e., level III handovers). While some tasks impose consistent orientations irrespective of the tool used (e.g., stirring a pot of broth always requires a vertical tool orientation), the usages of tools in other tasks depend on the pose of the manipulandum (e.g., using a screwdriver to drive a screw placed either vertically into a tabletop or horizontally into a wall). This imposes challenges for previous systems (Chan et al., 2014) because each task was bound with specific handover configurations. Therefore, tool affordance may need to be learned in a different way to allow level III task-oriented handovers.

Previous studies on tool affordance have learned tool-use in various ways. However, they may not be appropriate for task-oriented handovers. Tool affordances were learned as a distribution of the outcomes (Sinapov and Stoytchev, 2008; Stoytchev, 2005a) instead of the relationship between a movement of a tool and the corresponding status change of a manipulandum. With tool affordances learned in this manner, a robot cannot achieve level III handovers because the relation between specific usages and specific contexts is unknown. When this relationship was learned in a previous study (Tikhanoff et al., 2013), it learned in a way that was specific to the learned tools, and it was unknown whether a robot could generalize the learned tools to novel tools. It would be tedious to learn to use every tool prior to handing it over. While parallel Self-Organizing Maps (SOMs) can help with handling novel tools (Mar et al., 2017), novel tools needed to share similar shapes with the training tools, imposing restrictions on what kinds of novel tools a robot could hand over. This problem was overcome by using a large training set (Gajewski et al., 2019; Fang et al., 2020; Xie

et al., 2019), which may be impractical in time-sensitive scenarios to hand over tools.

6.2 Methods

In our system, a robot first learned how to use a tool. Then in a robot-to-human handover task, the handovers were calculated based on how a tool should be used in subsequent tasks, and were then passed on to standard inverse kinematic and motion planning libraries to execute the motion. The tools may even be novel such that the robot never observed their usages in the required task. In this case, the robot first inferred its usage based on how the tools were used in the same task, and then generated corresponding handovers.

6.2.1 Object Model Generation

Preliminary 3D models of the objects were scanned by the robot if possible. A script that utilized MeshLab[2] was used to automatically process the 3D models to smooth, upsample, recenter, and resurface the point clouds into triangular meshes. The 3D models of the tools were then segmented using the shape diameter function (CDF). The objects that could not be scanned by the robot were obtained manually with Autodesk Recap Pro[3]. Detailed procedures for obtaining 3D models can be found in Chapter 3.

6.2.2 Vision Module

To obtain the pose of an object in the scene, a partial point cloud of the object needs to be extracted from the environment. To isolate the partial point cloud, a background point cloud without the object and a foreground point cloud with the object was captured from a depth sensor. Both point clouds were processed to leave

[2]MeshLab: https://www.meshlab.net/
[3]Autodesk software: https://www.autodesk.com/

only the workspace, and the desktop was removed with random sample consensus (RANSAC; Fischler and Bolles, 1981). The partial point cloud of the object was obtained by subtracting the processed background point cloud from the processed foreground point cloud.

After obtaining the partial point cloud of the object, the pose of the object was retrieved by registering the partial point cloud with the triangular mesh of the object using a modified Iterative Closest Point registration (ICP) algorithm (Rusinkiewicz and Levoy, 2001). In this study, the pose of an object was represented with a 4×4 homogeneous transformation matrix $T \in SE(3)$ (superscript: reference frame, subscript: object), and $SE(3)$ represents the special Euclidean group:

$$T = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix}$$

where R is a 3×3 rotation matrix representing the orientation, and p is a vector representing the position. The pose of the tool $T^{world}_{tool\_on\_desk}$ and the manipulandum $T^{world}_{manipulandum}$ in the world frame were perceived when they were placed on the desktop.

6.2.3 Learning Tool Affordances

The system learned tool affordances with the tool use framework called TRansferrIng Skilled Tool use Acquired Rapidly (TRI-STAR) framework described in Chapter 3. The tool use framework includes an affordance taxonomy based on the goal state of the manipulandum in different frames of reference. It learns how a tool acts upon a manipulandum in a task using Learning from Demonstration (LfD). Based on the demonstrations, the framework classifies the tasks according to the affordance taxonomy and learns the tool-use skills or tool affordances accordingly.

In TRI-STAR, the tool affordances include motor skills and contact poses. Motor skills include kinematics skills, such as a trajectory that a tool should follow, and

dynamics skills which considers the forces. Though TRI-STAR currently only considers kinematics skills, dynamics skills are less relevant in handover tasks as the robot may not need to know the force that needs to be exerted while the human collaborator using the tool in order to find the appropriate handover configurations. The other component of tool affordances, which is the contact pose, include the grasping pose of the tool and the tool-manipulandum contact poses while using the tool. The grasping pose is dependent on the tool-manipulandum contact poses. Each segment between the demonstrated key points of the trajectories is represented with exponential representations that parametrize the segment with a screw axis and an angle. The segments are then grouped based on similar screw axes. As a result, the entire trajectory is represented with a series of pairs of a screw axis and an angle. The contact pose is represented by a class. Poses in the same class can be obtained by rotating about an axis. Based on the demonstrations, the framework needs to calculate the axis, choose one pose as the starting pose, and decide the range of rotation allowed about the axis. The range of the rotation depends on the type of task. For example, a slotted screwdriver may contact a slotted screw in two ways, while a hammer may approach a nail from infinitely many directions. Though the representations of the kinematics skills and contact skills are relatively uniform across all tasks, the choice of the frames of reference is dependent on the type of tasks in the affordance taxonomy.

Given novel tools and manipulanda, the key is to find how the object should substitute the learned object. In other words, the system should find the pose of the novel object in the reference frame of the learned object when using the objects in the tool-use task. The substitution is calculated by aligning the source objects and novel objects based on the global or local geometric features. When aligning the objects for global features, the point cloud is stretched or compressed disproportionally along different axes so that the bounding boxes of the objects match. The point cloud of the source object and the substitute object is mapped via modified ICP in order to gain

the best matching result. When aligning the objects for local features, the functional part of the object is stretched or compressed proportionally so that the longest edges of the bounding boxes match. The functional part of the source and substitute object is then mapped via modified ICP. In this way, two substitutions are obtained. One optimizes the global shape, and one optimizes the local features. The system chooses the substitution with a better matching result from these two options.

6.2.4 Grasping Configurations

The grasping configuration, which is the end-effector pose $T^{world}_{ee\_grasp}$ when grasping the tool, includes the orientation $R^{world}_{ee\_grasp}$ and the position $p^{world}_{ee\_grasp}$ of the end-effector.

Grasping Orientations

The tool to be handed over was assumed to be resting on the desk for simplicity. The grippers grasped the tool from above with the fingers perpendicular to the desktop. The opening of the gripper should be perpendicular to the primary axis $\vec{pa}$ of the tool (i.e., the direction of the longest edge of the minimum bounding box of the object), which resulted in the orientation $R^{tool\_adjusted}_{ee\_grasp}$ of the gripper being unchanged in the adjusted tool frame. Given the perceived pose of the tool $T^{world}_{tool\_on\_desk}$, the x axis of the adjusted tool frame $R^{world}_{tool\_adjusted}$ was defined as the unit primary axis of the tool, the z axis was defined as the unit vector opposite to the direction of standard gravity, and the y axis was calculated using the right-hand rule. With the adjusted tool frame, the orientation of the end-effector $R^{world}_{ee\_grasp}$ was calculated as (where $\times$ is matrix multiplication):

$$R^{world}_{ee\_grasp} = R^{world}_{tool\_adjusted} \times R^{tool\_adjusted}_{ee\_grasp}$$

Grasping Positions

The grasping position of the end-effector $p^{world}_{ee\_grasp}$ was initially chosen as the center of the contact area $p^{world}_{tool\_contact}$ of the tool when used on a manipulandum, because the contact area was the part of the tool least likely to be the handle. With learned tool affordances, the TRI-STAR framework calculated the contact area of the tool based on the manipulandum and the subsequent task. The center of the contact area $p^{model}_{tool\_contact}$ was calculated as the center of the minimum bounding box of the contact area. The use of a bounding box reduced bias due to the density of a point cloud. The $p^{world}_{tool\_contact}$ was obtained using $T^{world}_{tool\_on\_desk} \times p^{model}_{tool\_contact}$. To ensure stable grasping, the fingers of the grippers should distribute evenly around the primary axis of the tool. The grasping position needed to be adjusted by projecting $p^{world}_{tool\_contact}$ onto the primary axis $\vec{pa}$ to obtain an adjusted grasping position $p^{world}_{tool\_adjusted\_contact}$:

$$p^{world}_{tool\_adjusted\_contact} = \frac{(p^{world}_{tool\_contact} - p^{world}_{tool\_center}) \cdot \vec{pa}}{\|\vec{pa}\|^2} \vec{pa} + p^{world}_{tool\_center}$$

where $p^{world}_{tool\_center}$ was the center of the minimum bounding box of the tool. The grasping position $p^{world}_{ee\_grasp}$ was set to be $p^{world}_{tool\_adjusted\_contact}$ and the z was set to be the value where the gripper just touched the desktop.

6.2.5 Presentation Configurations

Presentation configurations are the end-effector poses when the robot presents the tool to the human collaborator. In order to minimize in-hand tool adjustment, the orientation of the tool $R^{world}_{tool\_present}$ should be close to the orientation when the human receiver started to use the tool $R^{world}_{tool\_usage}$, while the location of the handover $p^{world}_{tool\_present}$ was pre-set since the human receiver was assumed to be at a fixed location. Each $T^{world}_{tool\_present}$ corresponding to a $T^{world}_{ee\_candidate}$ was calculated using $T^{world}_{ee\_candidate} \times T^{ee}_{tool}$ where $T^{ee}_{tool} = {T^{world}_{ee\_grasp}}^{-1} \times T^{world}_{tool\_on\_desk}$ since the tool was grasped securely so that

T^{ee}_{tool} was unchanged. $R^{world}_{tool\_usage}$ was selected to be the start orientation of the tool trajectory in the world frame. The start orientation $R^{manipulandum}_{tool\_usage}$ of the tool trajectory in the manipulandum frame was generated from the TRI-STAR framework with learned tool affordances. $R^{world}_{tool\_usage}$ was obtained using $R^{world}_{manipulandum} \times R^{manipulandum}_{tool\_usage}$.

6.3 Results

We implemented and tested our system on a Kuka youBot robot without the mobile base. A Microsoft Azure RGB-D camera placed on the side of the workspace was used to perceive the pose of the tools and manipulanda. The human receiver was assumed to stand at a fixed location since adapting to the human location is beyond the scope of this study. In the training stage, the robot was trained with twenty demonstraions per tool in simulation to learn the tool affordances or how to use these tools in five tasks (i.e., stirring, pushing, cutting, knocking, and driving screws). The training tools and manipulanda are shown in Figure 6.3. No additional training was needed to perform handovers after learning the tool affordances. In the testing stage, the robot was required to hand over novel tools to a human to complete tasks and it was informed which task that the human receiver would perform.

6.3.1 Robot Validations

We conducted two experiments. Experiment I tested how the robot handed over a novel tool to complete tasks required either typical (i.e., level I) or improvised (i.e., level II) usage of the tool. Experiment II tested how the robot handed over a novel tool to complete a task with different manipulandum poses (i.e., level III). In order to show that the system can generate different handover configurations of a tool for different subsequent tasks, we chose the same tool to perform as many tasks as possible in the testing phase rather than one novel tool in each task. In experiment

I, a spoon and a screwdriver were chosen as the novel tools. As shown in Figure 6.3, the human receiver was required to perform the stirring, pushing, and cutting tasks with the spoon, and to perform the pushing, knocking, and driving screws tasks with

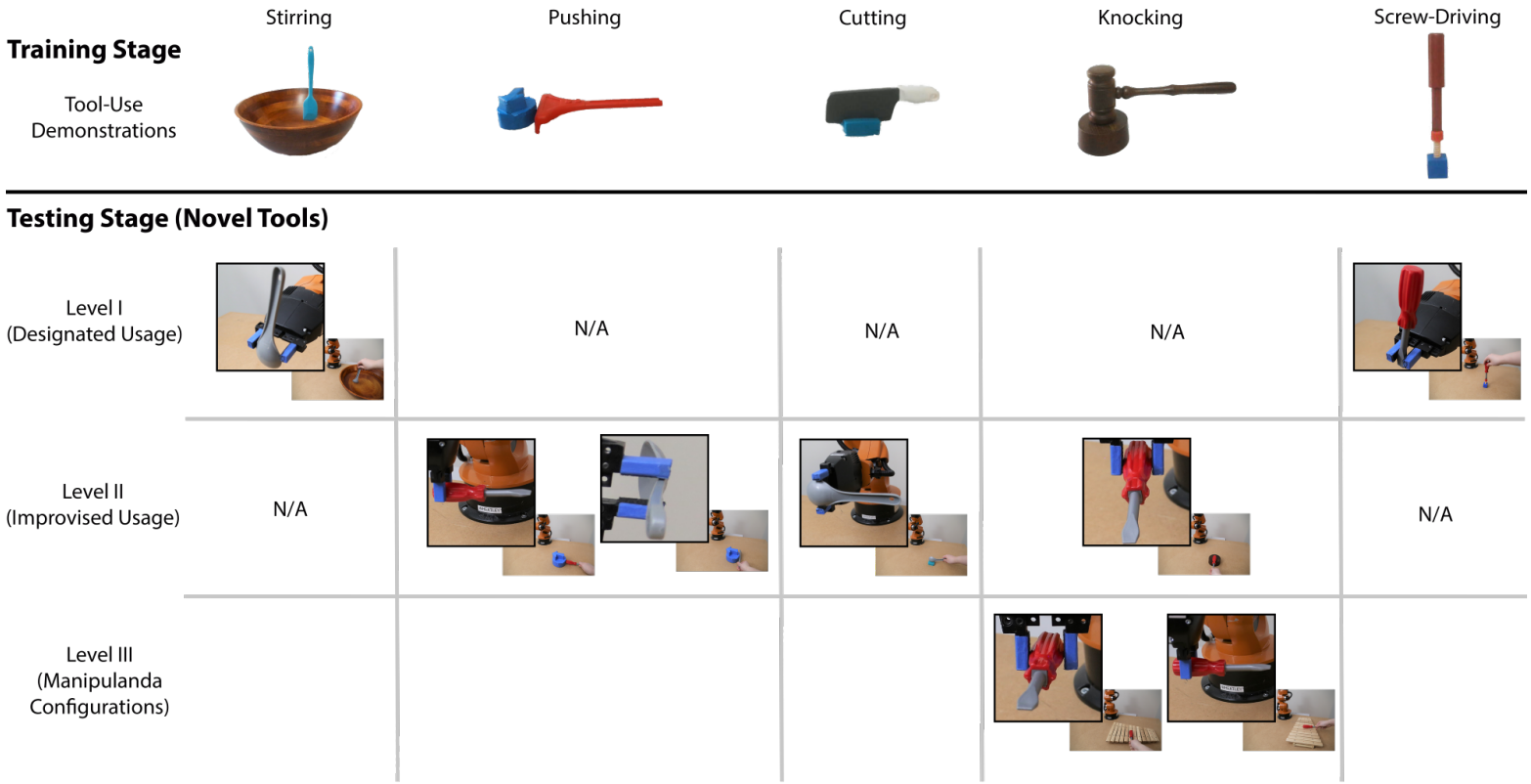


Figure 6.3: Handover evaluations on five tool-use tasks. (Top) The system was first trained with how to perform the stirring, pushing, cutting, knocking, and screw-driving tasks, rather than demonstrations of handovers. (Bottom) The robot was required to generate handovers for the human receiver to perform subsequent tasks. The handovers were with different levels of difficulty. The 'N/A' either refers to that the tool cannot perform the handover at the difficulty level, or the tool is inappropriate. Each cell shows a demonstration, which shows the handover generated and how the human used the tool to perform the subsequent task. The pictures were taken from the view of the human receiver.

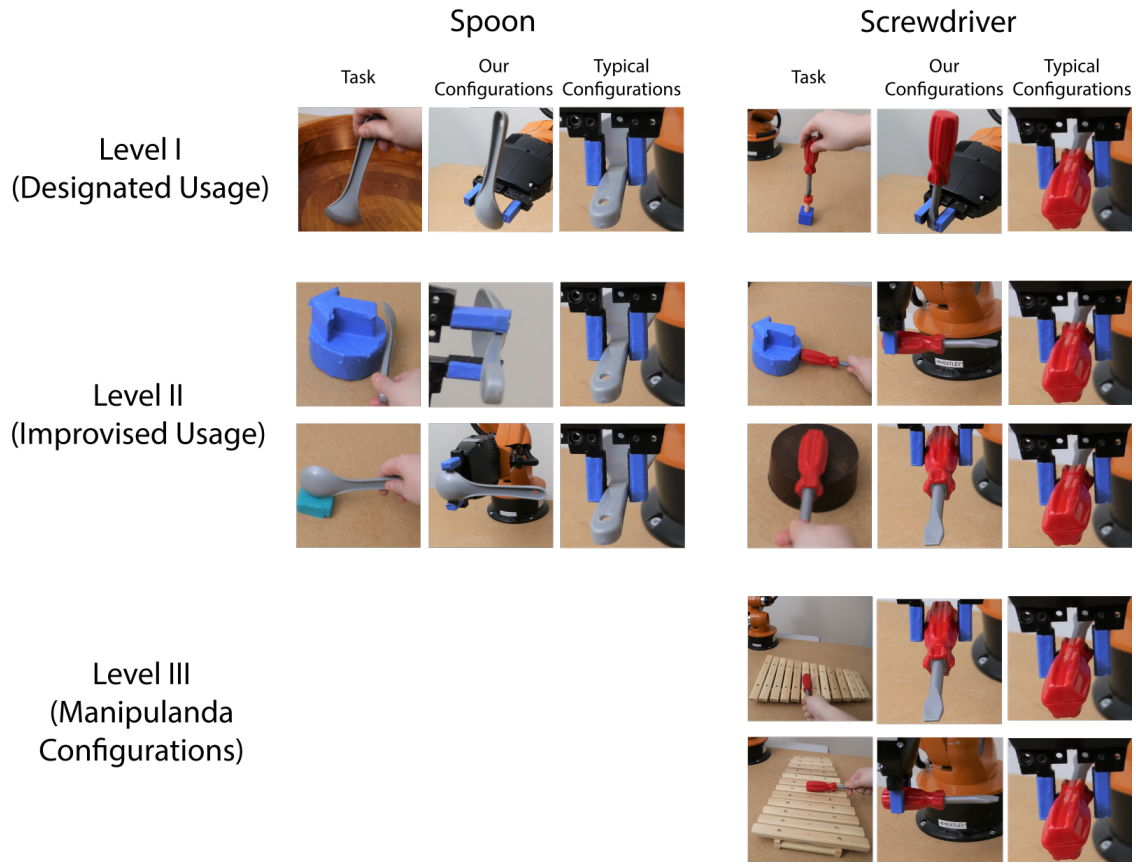


Figure 6.4: Comparing handover configurations generated by our system and the typical handovers in previous studies. The figure includes handovers of level I (top), level II (middle), and level III (bottom), with the spoon (left) and with the screwdriver (right) in different tasks. The typical handovers always grasp the same location on a tool and orient the handle of a tool horizontally to the human receiver. In contrast, our configurations are customized to the subsequent tasks and thus require minimum in-hand tool adjustments for the human receiver.

the screwdriver. A single tool was not required to perform all tasks because some tasks were inappropriate for the tool. In experiment II, even the manipulanda is a novel object, a xylophone, while the tool is the screwdriver. The xylophone was placed with two different orientations.

The results showed that the robot was able to handle level I, level II and level III handovers by adjusting both the grasping and presentation configurations according to the tasks. We compared our configurations with the typical configurations in previous studies as shown in Figure 6.4. While our handover configurations were

customized to the subsequent tasks, typical configurations in previous studies followed heuristic rules that a robot always selected a fixed location on the tool to grasp and oriented the handle horizontally towards the human receiver to present it. Therefore, handovers using our configurations required minimum in-hand tool adjustments when compared with the handovers using typical configurations.

6.3.2 Survey

To evaluate how naïve end-users perceive handovers generated by the robot, we conducted a survey on Amazon Mechanical Turk. Informed consent was obtained electronically. We recruited 70 participants, and each was compensated with $5. Out of the 70 participants, 15 were excluded from data analysis due to failing sanity-check questions. The data from the 55 eligible participants (35 males, 20 females) with an average age of 35.6 years were analyzed. In the survey, the questions were randomized, as were the options in each question. We designed multiple-choice questions (MCQ) and rating questions, which showed pictures or videos of the handovers and how the human receiver uses the tool in the subsequent tasks. The pictures and videos were taken from the view of the human receiver. A sample of the questionnaire can be found here[4]. The MCQ responses were converted to continuous variables and were analyzed with one-sample t-tests to compare with the chance level. The ratings were analyzed with paired samples t-tests.

For experiment I, the participants chose our handovers over the handovers from previous studies 88% of the time ($t(54) = 13.843$, $p < .001$). They were able to predict the subsequent tasks correctly 79% of the time ($t(54) = 11.461$, $p < .001$) given our handovers. On a five-point Likert scale, participants rated our configurations (M = 4.38, SD = 0.87) being more appropriate ($t(54) = 5.650$, $p < .001$) for the subsequent task than the typical configurations (M = 3.04, SD = 1.23). They also rated our

[4]Survey: https://tinyurl.com/surveyforhandover

handovers (M = 4.34, SD = 0.91) to be more comfortable ($t(54) = 5.751$, $p < .001$) for the human receiver than the handovers from previous studies (M = 3.22, SD = 1.13), and the collaboration was perceived to be more fluent ($t(54) = 4.810$, $p < .001$) when the robot used our handovers (M = 4.31, SD = 0.86) than when using the typical configurations (M = 3.24, SD = 1.22). For experiment II, the participants chose preferred handover configurations from two options. Results showed that the participants preferred our handovers 82% of the time ($t(54) = 7.884$, $p < .001$).

6.4 Discussion

We compiled a taxonomy of different requirements for handovers in general, and identified three levels of difficulty for task-oriented handovers in particular. We also integrated a system for task-oriented handovers, and showed that the system was able to handle level I, level II, and level III task-oriented handovers, and thus made it possible for the human receiver to complete subsequent tasks more efficiently with diverse task specifications. Furthermore, the system was trained with tool affordances, rather than demonstrations of handovers, allowing the system to understand the tool-use tasks and generalize the handovers to novel tools. The online survey results showed that participants preferred our handovers over the typical handovers in previous studies.

Our system presents a contribution towards task-oriented handovers. However, we would like to acknowledge the limitations of the current study. We focused on task-oriented handovers, while other handover requirements are beyond the scope of this study. For example, this study focused on task-oriented handovers and did not consider other aspects such as adapting to social signals from the human. Moreover, we acknowledge that the conclusions based on online studies are limited compared with an in-person study.

6.5 Summary

In this chapter, we leverage the TRI-STAR framework from Chapter 3 and utilize the learned tool affordance knowledge to assist a human-robot collaboration task. This work is among the first to demonstrate the importance of understanding tool use in relevant applications, and we believe tool use can benefit other topics in human-robot collaborations as well as other sub-areas in robotics. In the next chapter, we discuss all of our work included in the dissertation. We will focus on our contributions and limitations, as well as highlight future directions.

Chapter 7

Conclusion

Using tools greatly expands a robot's ability and allows it to be deployed in more application domains. Though tool use tasks share commonalities with general manipulation tasks, they have unique challenges to solve. The study of robot tool use is still in a preliminary stage, with many open challenges (Chapter 2).

7.1 Contributions

This dissertation makes the following contributions to address some open challenges:

- We recognize that tool use tasks have unique challenges compared with general manipulation tasks.
- We define robot tool use with insights from animal tool use.
- We compile a tool use taxonomy and identify the different requirements of each sub-type of tool use.
- To address open challenges 1 and 2, we develop the TRI-STAR framework to learn and reason about single-manipulation tool use.

  - In order to perform basic tool use, the framework used a uniform way to represent the actions in different sub-types of single-manipulation tool use tasks. Such representation suits tool use better than the traditional way. Moreover, TRI-STAR learns tool-manipulanda contact poses, which are sometimes ignored in previous studies.
  - In order to perform transferable tool use and improvisatory tool use, TRI-STAR considers both global and local features and how to transfer manipulations skills to substitute objects.
  - While most tool use focuses on one aspect of single-manipulation tool use, TRI-STAR is an integrative system with different modules to address different challenges of tool use.

- To partially address open challenge 4, we develop the TAAMP framework to expedite the search for sequential tool use.
- To partially address open challenge 5, we present a method for tool selection based on causal reasoning.
- To provides a solution to open challenge 6, we illustrate the importance of tool use knowledge with human-robot collaboration tasks, specifically robot-to-human handovers.

7.2 Future Work

Though we made contributions to robot tool use, there are limitations to the solutions. For example, TRI-STAR does not yet consider the dynamics of tool use. In TAAMP, the algorithm of affordance planning should be more efficient. In tool selection, our solution currently only considers relatively few causal relations. When applying tool use knowledge to handover tasks, we did not incorporate requirements

beyond task-oriented handovers. Future work addressing these limitations will significantly enhance the system's applicability or efficiency.

Future work can address other open challenges that are beyond the scope of the work presented in this dissertation. First, deductive tool use has yet to be solved. To achieve deductive tool use, a robot can be provided with physical rules such as the law of conservation of momentum and the work-energy theorem, as in Toussaint et al. (2018) and Levihn and Stilman (2014). The robot can then generate tool affordances from physical rules rather than summarize affordances from demonstrations as in other sub-types of single-manipulation tool use. To make it more interesting, the robot can even be provided with instances to learn these physical rules rather than directly provided with the physical rules. Second, multiple-manipulation tool use is another open challenge with limited studies focusing on this subject. It not only requires a robot to have a complete understanding of affordances as in single-manipulation tool use, but also requires more sophisticated manipulation and cognition skills. Third, a benchmark database is lacking in order to compare and evaluate different tool use systems. Though Abelha and Guerin (2017) provides three databases of 3D tool models with different qualities, the tools are limited to four categories that share common form factors (i.e., tools to roll dough, tools to cut lasagne, tools to hammer nails, tools to lift pancakes). Moreover, each study designs its own tool use tasks. A benchmark of task banks helps compare the capabilities of the systems. Fourth, studies in robot grasping and human-robot collaborations that are relevant to tool use rarely consider incorporating tool use knowledge, but generally only consider the default usage of tools.

To conclude this dissertation, we are excited to advance the study of robot tool use and allow robots to serve our community better.

Appendix A

Summary Tables of Studies in Chapter 2

In this chapter, we presented three summaries tables of the tool use studies in Chapter 2 based on the tool use taxonomy.

ID	Non-causal Tool Use	General Learning?	Learning Specifics?	Tasks	Dynamics?	Robots	Note
1	Pfeiffer et al. (2017)	no	no	fastening bolts in aircraft production	yes	physical: HRP-2Kai humanoid robot	
2	Robertsson et al. (2006)	no	no	stub grinding, deburring	yes	physical: ABB Irb6400 industrial robot with an extended ABB S4CPlus control system	
3	Kim et al. (2014)	no	no	writing	yes	simulation: dynamic simulation	
4	Nagata et al. (2001)	no	no	polishing	yes	physical: an industrial robot JS-10	
5	Takeuchi et al. (1993)	no	no	polishing	yes	physical: an articulate-type polishing robot with 6 degrees of freedom	
6	Kutsuzawa et al. (2017)	no	no	using a compass to draw a circle	yes	physical: a manipulator with six degrees of freedom MOTOMAN-MH3F	
7	Li et al. (2020)	no	no	unfastening screws	yes	physical: KUKA LBR iiwa 14 R800	
8	Rozo et al. (2013)	no	no	pouring	yes	physical: RX60	
9	Xue and Jia (2020)	no	no	n/a	n/a	simulation: UR10 with Shadow Hand	grasping planning
10	Su et al. (2018)	no	no	n/a	yes	physical: the Raven II platform	nision-based surgical tool segmentation
11	Garcia-Peraza-Herrera et al. (2017)	no	no	n/a	n/a	no	surgery tool segmention from 2D image
12	Schaal (2006); Ijspeert et al. (2002)	yes	no	tennis swinging	yes	physical: a 30 DOF Sarcos Humanoid robot	action learning: Dynamic Movement Primitives (DMP)
13	Muelling et al. (2010)	yes	no	playing table tennis	yes	physical: a Barrett WAM arm	action learning: a Mixture of Motor Primitives (MoMP)
14	Kober et al. (2008)	yes	no	playing ball-in-a-cup	yes	simulation: an anthropomorphic SARCOS robot arm	action learning: an augmented version of the dynamic systems motor primitives
15	Pastor et al. (2009)	yes	no	pouring	yes	physical and simulation: demonstrated with a 7-DoF robot arm and reproduced the action with the the Sarcos Slave arm in simulation	action learning: Dynamic Movement Primitives (DMP)
16	Kormushev et al. (2011)	yes	no	surface cleaning	yes	physical: a 25-DOF Fujitsu HOAP-2 humanoid robot	action learning: an extension of Dynamic Movement Primitives (DMP)
17	Paraschos et al. (2013)	yes	no	n/a	yes	physical: a KUKA lightweight robot arm; simulation: 7-link simulated planar robot	action learning: probabilistic movement primitives (ProMP)
18	Kulak et al. (2020)	yes	no	polishing, drawing	yes	physical: a 7-DoF torque-controlled Panda robot	action learning: Fourier movement primitive (FMP)

19	Droniou et al. (2014)	yes	no	writing digits	yes	physical: iCub	action learning: deep neural network
20	Byravan and Fox (2017)	yes	no	n/a	no	no	action learning: deep neural network
21	Guha et al. (2013)	yes	no	slicing, joining, mashing, pouring, stirring	no	no	action learning: minimalist plan
22	Tsuji et al. (2015)	yes	no	turn over pancake with a spatula, for Back and-Forth Gliding Movements of a Spatula to Slide Objects on Top	yes	physical: Motoman-MH3F	action learning: a unified algorithm for generating a variety of movements from planning trajectories that satisfy such conditions
23	Lutscher and Cheng (2013)	yes	no	table wiping, drawing	yes	physical: KUKA LBR-IV lightweight arm	action learning: a generalized programming layer for indirect force controllers (IFCs)
24	Ke et al. (2021)	yes	no	using chopsticks	yes	physical: a custom-built 6-DOF robot	action learning: combating covariate shift in model-free imitation learning
25	Lioutikov et al. (2017)	yes	no	writing letters	yes	physical: a 7-DoF KUKA lightweight armequipped with a five finger DLR HIT Hand II as end effector	action segmentation: Probabilistic Segmentation (ProbS)
26	Ramirez-Amaro et al. (2014a,b, 2015)	yes	no	pouring, cutting, cutting, sprinkling	yes	physical: iCub	action segmentation: semantic reasoning
27	Hu et al. (2014)	yes	no	n/a	no	no	action recognition: soft labeling
28	Shao et al. (2021)	yes	no	pushing, pulling, lifting, pushing, pouring, hitting	yes	simulation: 7-DoF Franka Panda robot arm with a two-fingered Robotiq 2F-85 gripper with pybullet	action recognition: language grounding
31	Wölfel and Henrich (2018)	yes	no	cutting, scratching, drawing, ironing, inserting, pouring, scooping, screwing, drilling	yes	simulation: unknown	action recognition: combined vervalized effects
32	Koch et al. (2022)	yes	no	n/a	no	no	action recognition: a methods-time-measurement based approach
35	Stoytchev (2003)	yes	yes	n/a	no	simulation: a two-joint robot	updating body schema: the tip of a tool
36	Nabeshima et al. (2005)	yes	yes	n/a	no	physical: a customized robot	updating body schema: the tip of a tool

37	Nabeshima et al. (2007)	yes	yes	poking	yes	simulation: a 3-DOF robot	updating body schema: the tip of a tool
33	Kemp and Edsinger (2006)	yes	yes	n/a	yes	physical: Domo	updating body schema: the tip of a tool
34	Jamone et al. (2013)	yes	yes	n/a	yes	simulation: iCub dynamic simulator	updating body schema: the tip of a tool
38	Karayiannidis et al. (2014)	yes	yes	n/a	yes	physical: a 7-DOF velocity controlled manipulator controlled at 130 Hz with a wrist mounted ATI Mini45 6-axis force-torque sensor	updating body schema: the tip of a tool
39	Hoffmann et al. (2014a)	yes	yes	drilling, drawing	yes	physical: the DARPA ARM robot for the drilling task, drawing task: an ST Robotics R17 5-DOF arm equipped with a linear gripper	updating body schema: the tip of a tool
40	Lee et al. (2008)	yes	yes	differet swings with a wooden sword, drinking from a coffee cup, different strokes with a tennis racket forehand stroke, different swings with a golf club	yes	physical: a humanoid robot; simulation: human skeleton	updating body schema: multiple points on tools
41	Katz et al. (2008)	yes	yes	n/a	yes	simulation: a simulated chain robot	updating body schema: the entire tool
42	Colgate et al. (1995)	yes	yes	n/a	n/a	no	collisions detection
43	Lee and Song (2021)	yes	yes	n/a	n/a	physical: Techman TM5-700 6-DOF manipulator	obstacle avoidance
44	Holladay et al. (2019)	yes	yes	hammer pulling, screw driving, wrench turning, knife cutting	yes	physical: ABB YuMi with custom printed finger	motion plannng
45	Kobayashi and Hosoe (2009)	yes	yes	using one object to move another another	no	simulation: 2D circle as a robot manipulator	motion planning
46	Raessa et al. (2019)	yes	yes	n/a	n/a	physical: a manufacturing cell with dual UR3 robots with a Robotiq F85 two finger grip on each arm	grasp planning
47	Chen et al. (2019)	yes	yes	vacuum sucking in order to move objects	n/a	physical and simulation: a dual-arm UR3 Robotiq F-85 parallel finger grippers	grasp planning

48	Lin and Sun (2015)	yes	yes	n/a	n/a	physical and simulation: a real Barrett hand equipped on a 6-DOF FANUC LR MATE 200iC robotic arm	grasp planning

Table A.1: Summary of Non-causal Tool Use Studies. In this table, we summarize the following aspects: (general learning) whether the study involves any type of learning, including aspects in general manipulation; (learning specifics) whether the study learns any aspect that is specifically for tool use, rather than general manipulation; (tasks) the tool use tasks demonstrated in this study; (dynamics) whether the study considers the dynamics while using tools; (robots) the robot that is used to demonstrated the tool use tasks or relevant aspects in this study.

ID	Basic Tool Use	Actions	Effects	Tools	Actions ↔ Effects	Sensory Input	Dynamics?	Tasks	Robots
1	Sinapov and Stoytchev (2008)	six predefined exploratory behaviors: *push*, *pull*, *slide-left*, *slide-right*, *rotate-left*, *rotate-right*	displacement of a puck in 2D	labels	using motion babbling to incrementally learn an adaptive hierarchical representation of the range of the effects	2D image	no	pushing, pulling	simulation: CRS+ A251 arm
2	Forestier and Oudeyer (2016)	dynamic movement primitives (DMP)	displacement in 2D	unknown	an active version of Model Babbling, which is the modular active curiosity-drive model babbling (the MACOB architecture)	n/a	no	pulling with magnetic and hook	simulation: a 2D robot with three joints and a gripper
3	Okada et al. (2006)	predefined action primitives	boolean (success, failure)	point cloud	sensor-based behavior verification system	sequence of 6 Dof coordinates for actions, 2D image for effects	yes	water-pouring, dishwashing	physical: a life-sized humanoid robot HRP2-JSK
4	Pastor et al. (2011)	dynamic movement primitives (DMP)	boolean (success, failure)	unknown	reinforcement learning	pool stroke: information from the Hokuyo laser scanner; box flipping: information from a MicroStrain Intertia-Link attached in side the box	yes	pool stroke, box flipping using two chopsticks	physical: PR2
5	Stoytchev (2005a, 2008)	Eight predefined actions: *extend arm (2 inches)*, *extend arm (5 inches)*, *slide left (2 inches)*, *slide left (5 inches)*, *slide right (2 inches)*, *slide right (5 inches)*, *contract arm (2 inches)*, *contract arm (5 inches)*	3D displacement of an object from two cameras	colors	using motion babbling to learn the affordance table	2D image	no	pushing, pulling	physical: a CRS+ A251 manipulator arm
6	Tikhanoff et al. (2013)	predefined push action with random chosen directions	displacement in 2D	tool tip position	Least Square Support Vector Machine (LSSVM)	3D image	yes	pushing, pulling	physical: iCub

7	Elliott et al. (2016)	ten predefined actions: *front center push*, *front side push right*, *front side push left*, *slide corner push right*, *slide corner push left*, *slide surface push right*, *slide surface push left*, *top pull*, *top side pull right*, *top side pull left*	3D displacement of an object	point cloud	multi-model regression	3D image	no	pushing, pulling	physical: PR2
8	Elliott and Cakmak (2018)	four predefined actions	boolean (success, failure) of each pixel		learning a predictive model of the task outcome from demonstrations	point cloud	no	cleaning (wiping or removing dirt)	physical: PR2; Fetch
9	Liu et al. (2018)	human actions from videos	object states	unknown	imitation learning based on video prediction with context translation and deep reinforcement learning	2D video	no	In simulation: pushing, sweeping, striking; Physical: sweeping, ladling almonds, pushing objects,	physical: a 7-DoF Sawyer robotmixure; simulation: the MuJoCo simulator
10	Claassens and Demiris (2011)	trajectories	3D point	3D point	affordance symmetry	information from an Opti-Track NaturalPoint motion capture system with 8 cameras	no	pouring, cutting	no

Table A.2: Study Summary of Causal Tool Use — Single-Manipulation Tool Use — Basic Tool Use. In this table, we summarize the following aspects: (actions) the action representations; (effects) the effect representations; (tools) the tool representations; (Actions $\leftrightarrow$ Effects) how this study learns the relation between actions and effects; (sensory input) the type of sensory input; (dynamics) whether the study considers the dynamics while using tools; (tasks) the tool use tasks demonstrated in this study; (robots) the robot that is used to demonstrated the tool use tasks or relevant aspects in this study.

ID	Transferable Tool Use	Actions	Effects	Tools	Actions ↔ Effects	Tools ↔ Actions	Sensory Input	Dynamics?	Tasks	Robots
1	Mar et al. (2017)	eight predefined actions: displacing a tool 17 cm along eight different radial direction	object displacement	tool-pose descriptors	assumed	parallel Self-Organizing Maps (SOM) mapping	3D image	no	dragging	simulation: iCub
2	Nishide et al. (2011)	a series of robot's joint angles	a series of features extracted with Self-Organizing Maps (SOM)	dynamics learning module, which is multiple time-scales recurrent neural network (MTRNN)	clustering the calculated parametric bias (PB) value of tools	2D image	yes	pulling	physical: the humanoid robot HRP-2	
3	Takahashi et al. (2017)	a series of robot's joint angles	start and target images	2D image	motor babbling using deep neural network (DNN) and multiple time-scales recurrent neural network (MTRNN)	deep neural network (DNN)	2D image	yes	swinging, pulling	simulation: the humanoid robot ACTOROID
4	Vogel et al. (2017)	unknown (pre-programmed)	unknown	unknown	assumed	optimization of energy transfer	torque and jerk at end effector	yes	hitting tasks like stiking a ball with a bat	physical: DLR LWR III manipulator

5	Kroemer et al. (2012)	dynamic movement primitives (DMP)	unknown	non-parametric representation of surface structures	assumed	kernal logistic regression (KLR)	information from a time-of-flight camera	no	pouring	physical: a seven degrees-of-freedom Motoman robot arm, a seven degrees-of-freedom Motoman robot arm, and a five-fingered Gifu robot hand
6	Brandi et al. (2014)	probablistic motor primitives (ProMP)	unknown	point cloud	assumed	3D warping with pre-defined labels	3D image	no	pouring	physical: a dual-arm robot ; simulation: the Bullet physics engine [20] together with Fluids 2
7	Dong et al. (2019)	tilting angle of the container	estimated volume of the liquid	point cloud	assumed	estimating the volume of the container	3D image	yes	pouring	physical: dual-arm robot system
8	Gemici and Saxena (2014)	unknown (pre-programmed)	unknown	(manipulanda representation) six predefined physical properties: *hardness plasticity*, *elasticity*, *tensile strength*, *brittleness*, *adhesiveness*	assumed	Learning (Manipulanda Action): haptic learning with Dirichlet process and reinforcement learning	haptic inputs	yes	cutting	physical: PR2

9	Elliott et al. (2017)	extracted 2D clearning patterns	markers being removed	(manipulanda representation) size of a surface	assumed	Learning (Manipulanda Action): determine the repetition needed of the cleaning pattern which depends on the size of the surface	3D image	no	surface cleaning	physical: PR2
10	Li et al. (2018)	a tuple of start and end pixel on the image plane	displacement of an object, including rotation	(manipulanda representation) 2D pose and binary mask of an object	assumed	Push-Net	2D image	yes	pushing	physical and simulation: Fetch, Kinova MICO
11	Tee et al. (2018, 2022)	predefined actions: *pull back*, *push forward*, *push sideways*, *lift up*	object displacement	point cloud	assumed	matched the segmented unseen tools to segmented end-effector and arms of a robot with features extracted with 3D Orthogonal Profile Descriptors (OPD) technique	3D image	no	pulling, pushing, lifting	physical: a Olivia III robot; simulation: a planar three-DoF robot
12	Manuelli et al. (2019); Gao and Tedrake (2021)	the desired linear or angular velocity, or the desired force or torque of an oriented keypoint	unknown	keypoints of common form-factors	assumed	keypoint detection and rigid transformation	3D image	yes	whiteboard wiping, (prototool-use) peg-hole-insertion	

13	Stückler and Behnke (2014b); Stückler et al. (2016, 2013); Stückler and Behnke (2014a, 2015)	trajectories	unknown	point cloud	assumed	coherent point drift	3D image	yes	drawing, bottle opening, using a pair of tongs to grasp sausages, sweeping dust,watering plants	physical: cosero
14	Sinapov and Stoytchev (2007)	2D vectors	displacement of an object in 2D	tools in different frames	k-nearest and neighbor and decision tree		2D image	no	pulling	simulation: the open-source dynamic robot simulator BREVE
15	Gonçalves et al. (2014a,b); Dehban et al. (2016)	four pre-defined directional pushes: *left*, *right*, *pull closer*, *push away*	2D displacement of an object	pre-defined shape discriptors: *area*, *convexity*, *eccentricity*, *compactness*, *circleness*, *squareness*	probablistic causal model represented with Bayesian networks		2D image	no	pushing, pulling	physical and simulation: iCub
16	Mar et al. (2015)	length and direction of a push	displacement of an object	pre-defined features (each include sub-features): *based on convex hull*, *based on thinning*, *moments*, *shape descriptors*, *from the angle sinature*, *domain transformations from the distance to the centroid signature*	support vector machine classifiers		2D image	no	pulling	physical and simulation: iCub

Table A.3: Study Summary of Causal Tool Use — Single-Manipulation Tool Use — Transferable Tool Use. In this table, we summarize the following aspects: (actions) the action representations; (effects) the effect representations; (tools) the tool representations; (Actions $\leftrightarrow$ Effects) how this study learns the relation between actions and effects; (Tools $\leftrightarrow$ Actions) how this study learns the relation between tools and actions; (sensory input) the type of sensory input; (dynamics) whether the study considers the dynamics while using tools; (tasks) the tool use tasks demonstrated in this study; (robots) the robot that is used to demonstrated the tool use tasks or relevant aspects in this study.

ID	Improvisatory Tool Use	Actions	Effects	Tools	Actions ↔ Effects	Tools ↔ Actions	Tools ↔ Effects	Sensory Input	Dynamics?	Tasks	Robots	Note
1	Myers et al. (2015)	n/a	n/a	two approaches to learn local shape and geometry primitives: superixel based hierarchical matching pursuit (S-HMP); structured random forests (SRF)	n/a	n/a	n/a	3D image	no	cutting, scooping, containing, pounding	no	task-oriented grasping
2	Song et al. (2010, 2011b,a); Kroemer et al. (2012); Madry et al. (2012); Song et al. (2015)	n/a	n/a	two pre-defined features: *object class*; *object dimension*, *convexity*, *eccentricity*	n/a	n/a	n/a	3D image	no	pouring, tool-use	simulation: a 20-DoF human hand model, a 7-DoF Schunk Dexterous hand model, and an Armar 11 DoF hand	task-oriented grasping
3	Murali et al. (2020)	n/a	n/a	PointNet++ architecture [citation153 42] to represent point cloud and grasp poses; Sematic hierarchy of objects (WordNet)	n/a	n/a	n/a	3D image	no	mixing, sauting with a pan, can opening, spraying	physical and simulation: Sawyer Robot	task-oriented grasping

4	Kokic et al. (2017)	n/a	labels	point cloud	n/a	n/a	n/a	3D image	no	cutting, poking, pounding, pouring	simulation: a Schunk-SDH hand mounted on a KUKA KR5 sixx 850 manipulator	task-oriented grasping;
5	Detry et al. (2017)	n/a	labels	depth image	n/a	n/a	n/a	3D image	no	pouring	no	task-oriented grasping
6	Schoeler and Wörgötter (2015)	n/a	labels	graph representation pf segmented point cloud	n/a	n/a	support vector machines	3D image	no	sieving, cutting, containing, poking, hitting	no	tool part detection
7	Nakamura and Nagai (2010)	contact poses (grasping and tool-manipulandum contact poses)	four predefined features: *color change*, *contour change*, *Barycentric position change*, *change in the number of the work object*	scale invariant feature transform (SIFT) to represent local features	Bayesian networks			unknown	no	cutting, coloring, deformation, transfer, bonding, bonding with coloring	no	tool part detection
8	Fitzgerald et al. (2019)	linear and/or rotational transform model of the end-effector trajectory	n/a	n/a	assumed	inferred from human demonstrations for each tool	provided	unknown	no	sweeping, hooking, hammering	physical: a 7-DOF Jaco2 arm equipped with a Robotiq 85 gripper	

9	Agostini et al. (2015)	semantic event chains (SECs) enriched with object and trajectory information	predicates	segmented image	provided planning operator (PO)	database: a repository of objects and attributes with roles (ROAR)	unknown	yes	cutting, stirring	physical: the KUKA arm platform	
10	Fang et al. (2020)	predefined gripper trajectories	labels	point cloud	assumed	deep neural networks	3D image	no	sweeping, hammering	physical and simultation: a 7-DoF Rethink Robotics Sawyer Arm with a parallel jaw gripper	
11	Xie et al. (2019)	trajectories of end-effectors	position changes of objects in pixels	2D image	deep neural networks		2D image	no	sweeping, wiping, hooking	physical: a Sawyer robot	
12	Jain and Inamura (2013)	five predefined actions: *contract arm*, *slide arm left*, *pull diagonally-1*, *slide arm right*, *pull diagonally-2*	object displacement	three pre-defined features: *horizontal part*, *vertical part*, *corner*	Bayesian networks		2D image	yes	pushing, pulling	no	
13	Qin et al. (2020)	trajectories	labels	keypoints based on local features	assumed	a framework of keypoint representations for tool manipulation (KETO)	3D image	yes	hammering, pushing, reaching	simulation: pybullet	
14	Turpin et al. (2021)	contact poses (grasping and tool-manipulandum contact poses)	labels	keypoints based on local features	assumed	Generilizable Interaction-aware Functional Tool affordances (GIFT)	3D image	no	hooking, reaching, hammering	simulation: a Sawyer robot arm	

15	Abelha and Guerin (2017); Gajewski et al. (2019); Abelha et al. (2016); Guerin and Ferreira (2019)	predefined action profiles	labels	segmented point cloud approximated with superquadrics and superparaboloids	assumed	p-tools	3D image	no	rolling dough, cutting lasagne, hammering nail, lifting pancake, tenderising meat, piercing a potato skin, scooping	simulation: no actual robot	
16	Zhu et al. (2015)	trajectories	object status change	point cloud, mass, volume	ranking function		3D image	yes	chopping wood, shovelling dirt, painting wall	no	
17	Qin et al. (2021)	trajectories segmented with exponential representations of transformations	object displacement for relocation tasks; other properties for other tasks	point cloud	by classifying task type given demonstrations	2-steps substitution algorithm	3D image	no	knocking, stirring, pushing, scooping, cutting, writing, screw-driving	physical and simulation: Baxter, University Robotics UR5e, Kuka youBot	

Table A.4: Study Summary of Causal Tool Use — Single-Manipulation Tool Use — Improvisatory Tool Use. In this table, we summarize the following aspects: (actions) the action representations; (effects) the effect representations; (tools) the tool representations; (Actions $\leftrightarrow$ Effects) how this study learns the relation between actions and effects; (Tools $\leftrightarrow$ Actions) how this study learns the relation between tools and actions; (Tools $\leftrightarrow$ Effects) how this study learns the relation between tools and effects; (sensory input) the type of sensory input; (dynamics) whether the study considers the dynamics while using tools; (tasks) the tool use tasks demonstrated in this study; (robots) the robot that is used to demonstrated the tool use tasks or relevant aspects in this study.

ID	Multiple-manipulation Tool Use	Category	Affordance	Manipulation	Cognition Reasoning	Cognition: Planning	Tasks	Robots	Note
1	Yamazaki et al. (2010)	Sequential Tool Use	assumed	no	no	pre-defined sequence; focus on failure detection and recovery	sweeping a floor	physical: ART daily assistive robot	
2	Toussaint et al. (2018)	Sequential Tool Use	provided in the form of equations	no	no	optimization-based task and motion planning (logic-geometric programming, LGP)	throwing, hitting, hitslide, hitslidesit, pushing	simulation: 14DOF humanoid with two arms	
3	Migimatsu and Bohg (2020)	Sequential Tool Use	provided in the form of equations	no	no	optimization-based task and motion planning (object centric logic-geometric programming, LGP)	n/a	physical and simulation: a 7-dof Franka Panda fitted with a Robotiq 2F-85 gripper	
4	Qin et al. (2022b)	Sequential Tool Use	learned with TRI-STAR (Qin et al., 2021)	no	no	sampling-based task and motion planning (TAAMP)	pushing, pulling	physical: a Kuka youBot robot arm; simulation: a Kuka iiwa robot arm	
5	Levihn and Stilman (2014)	Tool Selection	prespecified tool features; physical laws	no	no	no	using a lever, or other objects to keep a door open	simulation: the dynamic simulator DART	
6	Wicaksono and Sammut (2016)	Tool Selection	formed hypothesis based on observations of tool use, and generated structural similarity score to select tools	no	no	no	pulling	physical and simulation: Baxter	

7	Saito et al. (2018)	Tool Selection	learned tool use with deep learning	no	no	no	pulling, sliding	physical: the NEXTAGE humanoid robot	
8	Brawer et al. (2020)	Tool Selection	learned tool use as a structural causal model (SCM) 8 3	no	no	no	pushing, pulling	physical: Baxter	
9	Dietrich et al. (2010)	Tool Manufacturing	n/a	the analysis of contact models for assembly	no	no	(pseudo-tool use) peg-in-hole with complex parts	physical: the parallel robot Hexa	robot assembly
10	Bös et al. (2017)	Tool Manufacturing	n/a	to increase the achievable speed of compliant manipulators with interatively learned and temporally scaled force control	no	no	(pseudo-tool use) peg-in-hole	physical: an ABB YuMi	robot assembly
12	Peternel et al. (2015)	Tool Manufacturing	n/a	impedance control interface for human-in-the-loop approach	no	no	sliding a bolt fitting inside a groove in order to mount two parts together	physical: Kuka Light Weight Robot, HapticMaster robot	robot assembly
13	Gu et al. (2014)	Tool Manufacturing	learning the relationship of actions and effects with a Portable Assembly Demonatration (PAD) system	no	no	no	screwing, hammering, wrenching	no	robot assembly
14	Nair et al. (2019a,b)	Tool Manufacturing	tool substitution algorithm from Abelha et al. (2016) with an extra step of tool validation phase	pre-determined method to attach parts together, including pierce attachment, grasp attachment and magnetic attachment	using computed scores to determine which parts to choose, and where to connect the parts	not needed as it only involves connecting two parts as a tool	hitting, containing or scooping, screwing, flipping, squeegeeing	physical: a 7-DOF robot arm	

15	Sammut et al. (2015)	Tool Manufacturing	learn affordances by forming hypothesis about important features of tools (Wicaksono, 2020)	constructed the tool with 3D printing	to determine the functional specification and to convert the specification into a design	not needed as the construction is by 3D printing	no evaluation yet	no	

Table A.5: Study Summary of Causal Tool Use — Multiple-manipulation Tool Use. In this table, we summarize the following aspects: (category) the sub-category of this task in multiple-manipulation tool use; (affordance) how the affordance is learned in this study; (manipulation) extra manipulation skills needed comapred with single-maniplation tool use; (cognition: reasoning) extra reasoning skills needed comapred with single-maniplation tool use; (cognition: planning) extra planning skills needed comapred with single-maniplation tool use; (tasks) the tool use tasks demonstrated in this study; (robots) the robot that is used to demonstrated the tool use tasks or relevant aspects in this study.

Bibliography

Abelha, P. and Guerin, F. (2017). Learning how a tool affords by simulating 3d models from the web. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4923–4929. IEEE.

Abelha, P., Guerin, F., and Schoeler, M. (2016). A model-based approach to finding substitute tools in 3d vision data. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2471–2478. IEEE.

Admoni, H., Dragan, A., Srinivasa, S. S., and Scassellati, B. (2014). Deliberate delays during robot-to-human handovers improve compliance with gaze communication. In *Proceedings of the 2014 ACM/IEEE international conference on Human-robot interaction*, pages 49–56.

Agostini, A., Aein, M. J., Szedmak, S., Aksoy, E. E., Piater, J., and Würgütter, F. (2015). Using structural bootstrapping for object substitution in robotic executions of human-like manipulation tasks. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6479–6486. IEEE.

Alcock, J. (1972). The evolution of the use of tools by feeding animals. *Evolution*, pages 464–473.

Aleotti, J., Micelli, V., and Caselli, S. (2012). Comfortable robot to human object hand-over. In *2012 IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication*, pages 771–776. IEEE.

Andries, M., Chavez-Garcia, R. O., Chatila, R., Giusti, A., and Gambardella, L. M. (2018). Affordance equivalences in robotics: a formalism. *Frontiers in neurorobotics*, 12:26.

Angelov, D., Hristov, Y., and Ramamoorthy, S. (2019). Using causal analysis to learn specifications from task demonstrations. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pages 1341–1349. International Foundation for Autonomous Agents and Multiagent Systems.

Argall, B. D., Chernova, S., Veloso, M., and Browning, B. (2009). A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483.

Asano, T. (1994). Tool using behavior and language in primates. *Behavior analysis of language and cognition (eds SC Hayes, LJ Hayes, M. Sato & K. Ono)*, pages 145–148.

Beck, B. B. (1980). *Animal tool behavior: the use and manufacture of tools by animals*. Garland STPM Press, New York.

Bestick, A., Bajcsy, R., and Dragan, A. D. (2016). Implicitly assisting humans to choose good grasps in robot to human handovers. In *International Symposium on Experimental Robotics*, pages 341–354. Springer.

Bestick, A. M., Burden, S. A., Willits, G., Naikal, N., Sastry, S. S., and Bajcsy, R. (2015). Personalized kinematics for human-robot collaborative manipulation. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1037–1044. IEEE.

Bidot, J., Karlsson, L., Lagriffoul, F., and Saffiotti, A. (2017). Geometric backtracking for combined task and motion planning in robotic systems. *Artificial Intelligence*, 247:229–265.

Biro, D., Inoue-Nakamura, N., Tonooka, R., Yamakoshi, G., Sousa, C., and Matsuzawa, T. (2003). Cultural innovation and transmission of tool use in wild chimpanzees: evidence from field experiments. *Animal cognition*, 6(4):213–223.

Boesch, C. (2013). Ecology and cognition of tool use in chimpanzees. *Tool use in animals: Cognition and ecology*, pages 21–47.

Bös, J., Wahrburg, A., and Listmann, K. D. (2017). Iteratively learned and temporally scaled force control with application to robotic assembly in unstructured environments. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3000–3007. IEEE.

Brandi, S., Kroemer, O., and Peters, J. (2014). Generalizing pouring actions between objects using warped parameters. In *2014 IEEE-RAS International Conference on Humanoid Robots*, pages 616–621. IEEE.

Brawer, J., Qin, M., and Scassellati, B. (2020). A causal approach to tool affordance learning. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8394–8399. IEEE.

Breuer, T., Ndoundou-Hockemba, M., and Fishlock, V. (2005). First observation of tool use in wild gorillas. *PLoS biology*, 3(11):e380.

Burger, B., Maffettone, P. M., Gusev, V. V., Aitchison, C. M., Bai, Y., Wang, X., Li, X., Alston, B. M., Li, B., Clowes, R., et al. (2020). A mobile robotic chemist. *Nature*, 583(7815):237–241.

Byravan, A. and Fox, D. (2017). Se3-nets: Learning rigid body motion using deep neural networks. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 173–180. IEEE.

Cabrera-Álvarez, M. J. and Clayton, N. S. (2020). Neural processes underlying tool use in humans, macaques, and corvids. *Frontiers in Psychology*, 11:560669.

Cakmak, M., Dogar, M., Ugur, E., and Sahin, E. (2007). Affordances as a framework for robot control. In *Proceedings of the 7th international conference on epigenetic robotics, epirob'07*.

Cakmak, M., Srinivasa, S. S., Lee, M. K., Kiesler, S., and Forlizzi, J. (2011). Using spatial and temporal contrast for fluent robot-human hand-overs. In *Proceedings of the 6th international conference on Human-robot interaction*, pages 489–496.

Call, J. (2013). Three ingredients for becoming a creative tool user. *Tool use in animals: Cognition and ecology*, pages 3–20.

Chan, W. P., Kakiuchi, Y., Okada, K., and Inaba, M. (2014). Determining proper grasp configurations for handovers through observation of object movement patterns and inter-object interactions during usage. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1355–1360. IEEE.

Chan, W. P., Pan, M. K., Croft, E. A., and Inaba, M. (2020). An affordance and distance minimization based method for computing object orientations for robot human handovers. *International Journal of Social Robotics*, 12(1):143–162.

Chan, W. P., Parker, C. A., Van der Loos, H. M., and Croft, E. A. (2012). Grip forces and load forces in handovers: implications for designing human-robot handover controllers. In *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*, pages 9–16.

Chan, W. P., Parker, C. A., Van der Loos, H. M., and Croft, E. A. (2013). A human-inspired object handover controller. *The International Journal of Robotics Research*, 32(8):971–983.

Chasles, M. (1830). *Mémoire de géométrie pure sur les systèmes de forces, et les systèmes d'aires planes; et sur les polygones, les polyèdres, et les centres des moyennes distances.(Extrait du VIe volume de la Correspondence mathématique et physique des Pays-Bas.).* Mayez.

Chen, H., Wan, W., and Harada, K. (2019). Combined task and motion planning for a dual-arm robot to use a suction cup tool. In *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, pages 446–452. IEEE.

Chevalier-Skolnikoff, S. (1989). Spontaneous tool use and sensorimotor intelligence in cebus compared with other monkeys and apes. *Behavioral and brain sciences*, 12(3):561–588.

Choi, Y. S., Chen, T., Jain, A., Anderson, C., Glass, J. D., and Kemp, C. C. (2009). Hand it over or set it down: A user study of object delivery with an assistive mobile manipulator. In *RO-MAN 2009-The 18th IEEE International Symposium on Robot and Human Interactive Communication*, pages 736–743. IEEE.

Claassens, J. and Demiris, Y. (2011). Generalising human demonstration data by identifying affordance symmetries in object interaction trajectories. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1980–1985. IEEE.

Colgate, J. E., Stanley, M. C., and Brown, J. M. (1995). Issues in the haptic display of tool use. In *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*, volume 3, pages 140–145. IEEE.

Daniel, W. W. (1990). Kolmogorov–smirnov one-sample test. *Applied nonparametric statistics*, 2.

Dantam, N. T., Kingston, Z. K., Chaudhuri, S., and Kavraki, L. E. (2016). Incremental task and motion planning: A constraint-based approach. In *Robotics: Science and systems*, volume 12, page 00052. Ann Arbor, MI, USA.

Dehban, A., Jamone, L., Kampff, A. R., and Santos-Victor, J. (2016). Denoising auto-encoders for learning of objects and tools affordances in continuous space. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4866–4871. IEEE.

Dennis, Jr, J. E. and Moré, J. J. (1977). Quasi-newton methods, motivation and theory. *SIAM review*, 19(1):46–89.

Detry, R., Papon, J., and Matthies, L. (2017). Task-oriented grasping with semantic and geometric scene understanding. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3266–3273. IEEE.

Dietrich, F., Buchholz, D., Wobbe, F., Sowinski, F., Raatz, A., Schumacher, W., and Wahl, F. M. (2010). On contact models for assembly tasks: Experimental investigation beyond the peg-in-hole problem on the example of force-torque maps. In *2010 IEEE/RSJ international conference on intelligent robots and systems*, pages 2313–2318. IEEE.

Dong, C., Takizawa, M., Kudoh, S., and Suehiro, T. (2019). Precision pouring into unknown containers by service robots. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5875–5882. IEEE.

Droniou, A., Ivaldi, S., and Sigaud, O. (2014). Learning a repertoire of actions with deep neural networks. In *4th International Conference on Development and Learning and on Epigenetic Robotics*, pages 229–234. IEEE.

Eberhardt, F. (2017). Introduction to the foundations of causal discovery. *International Journal of Data Science and Analytics*, 3(2):81–91.

Eguiluz, A. G., Rañó, I., Coleman, S. A., and McGinnity, T. M. (2017). Reliable object handover through tactile force sensing and effort control in the shadow robot hand. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 372–377. IEEE.

Ek, C. H., Song, D., Huebner, K., and Kragic, D. (2010). Exploring affordances in robot grasping through latent structure representation. *Vision for Cognitive Tasks, ECCV*.

Elliott, S. and Cakmak, M. (2018). Robotic cleaning through dirt rearrangement planning with learned transition models. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1623–1630. IEEE.

Elliott, S., Valente, M., and Cakmak, M. (2016). Making objects graspable in confined environments through push and pull manipulation with a tool. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 4851–4858. IEEE.

Elliott, S., Xu, Z., and Cakmak, M. (2017). Learning generalizable surface cleaning actions from demonstration. In *2017 26th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 993–999. IEEE.

Erdem, E., Haspalamutgil, K., Palaz, C., Patoglu, V., and Uras, T. (2011). Combining high-level causal reasoning with low-level geometric reasoning and motion planning for robotic manipulation. In *2011 IEEE International Conference on Robotics and Automation*, pages 4575–4581. IEEE.

Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *kdd*, 96(34):226–231.

Fang, K., Zhu, Y., Garg, A., Kurenkov, A., Mehta, V., Fei-Fei, L., and Savarese,

S. (2020). Learning task-oriented grasping for tool manipulation from simulated self-supervision. *The International Journal of Robotics Research*, 39(2-3):202–216.

Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395.

Fitzgerald, T., Short, E., Goel, A., and Thomaz, A. (2019). Human-guided trajectory adaptation for tool transfer. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pages 1350–1358.

Foerster, J. N., Farquhar, G., Afouras, T., Nardelli, N., and Whiteson, S. (2018). Counterfactual multi-agent policy gradients. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Forestier, S. and Oudeyer, P.-Y. (2016). Modular active curiosity-driven discovery of tool use. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3965–3972. IEEE.

Fragaszy, D. and Eshchar, Y. (2017). Tool use in nonhuman primates: natural history, ontogenetic development and social supports for learning. *Evolution of Nervous Systems*.

Gajewski, P., Ferreira, P., Bartels, G., Wang, C., Guerin, F., Indurkhya, B., Beetz, M., and Śniezyński, B. (2019). Adapting everyday manipulation skills to varied scenarios. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 1345–1351. IEEE.

Gao, W. and Tedrake, R. (2021). kpam 2.0: Feedback control for category-level robotic manipulation. *IEEE Robotics and Automation Letters*, 6(2):2962–2969.

Garcia-Peraza-Herrera, L. C., Li, W., Fidon, L., Gruijthuijsen, C., Devreker, A., Attilakos, G., Deprest, J., Vander Poorten, E., Stoyanov, D., Vercauteren, T., et al. (2017). Toolnet: holistically-nested real-time segmentation of robotic surgical tools. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5717–5722. IEEE.

Garrett, C. R., Chitnis, R., Holladay, R., Kim, B., Silver, T., Kaelbling, L. P., and Lozano-Pérez, T. (2021). Integrated task and motion planning. *Annual review of control, robotics, and autonomous systems*, 4:265–293.

Garrett, C. R., Lozano-Perez, T., and Kaelbling, L. P. (2018). Ffrob: Leveraging symbolic planning for efficient task and motion planning. *The International Journal of Robotics Research*, 37(1):104–136.

Garrett, C. R., Lozano-Pérez, T., and Kaelbling, L. P. (2020). Pddlstream: Integrating symbolic planners and blackbox samplers via optimistic adaptive planning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 30, pages 440–448.

Gemici, M. C. and Saxena, A. (2014). Learning haptic representation for manipulating deformable food objects. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 638–645. IEEE.

Gibson, J. J. (1979). The ecological approach to visual perception.

Gonçalves, A., Abrantes, J., Saponaro, G., Jamone, L., and Bernardino, A. (2014a). Learning intermediate object affordances: Towards the development of a tool concept. In *4th International Conference on Development and Learning and on Epigenetic Robotics*, pages 482–488. IEEE.

Gonçalves, A., Saponaro, G., Jamone, L., and Bernardino, A. (2014b). Learning visual affordances of objects and tools through autonomous robot exploration. In

2014 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC), pages 128–133. IEEE.

Goodall, J. (1964). Tool-using and aimed throwing in a community of free-living chimpanzees. *Nature*, 201(4926):1264–1266.

Grigore, E. C., Eder, K., Pipe, A. G., Melhuish, C., and Leonards, U. (2013). Joint action understanding improves robot-to-human object handover. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4622–4629. IEEE.

Gu, Y., Sheng, W., and Ou, Y. (2014). Automated assembly skill acquisition through human demonstration. In *2014 IEEE international conference on robotics and automation (ICRA)*, pages 6313–6318. IEEE.

Guerin, F. and Ferreira, P. (2019). Robot manipulation in open environments: New perspectives. *IEEE transactions on cognitive and developmental systems*, 12(3):669–675.

Guha, A., Yang, Y., Fermu, C., Aloimonos, Y., et al. (2013). Minimalist plans for interpreting manipulation actions. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5908–5914. IEEE.

Hadfield-Menell, D., Lin, C., Chitnis, R., Russell, S., and Abbeel, P. (2016). Sequential quadratic programming for task plan optimization. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5040–5047. IEEE.

Hauser, A. and Bühlmann, P. (2014). Two optimal strategies for active learning of causal models from interventional data. *International Journal of Approximate Reasoning*, 55(4):926–939.

Hayes, B. and Scassellati, B. (2013). Challenges in shared-environment human-robot collaboration. *learning*, 8(9).

Hillenbrand, U. and Roa, M. A. (2012). Transferring functional grasps through contact warping and local replanning. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2963–2970. IEEE.

Hoffmann, H., Chen, Z., Earl, D., Mitchell, D., Salemi, B., and Sinapov, J. (2014a). Adaptive robotic tool use under variable grasps. *Robotics and Autonomous Systems*, 62(6):833–846.

Hoffmann, H., Chen, Z., Earl, D., Mitchell, D., Salemi, B., and Sinapov, J. (2014b). Adaptive robotic tool use under variable grasps. *Robotics and Autonomous Systems*, 62(6):833–846.

Holladay, R., Lozano-Pérez, T., and Rodriguez, A. (2019). Force-and-motion constrained planning for tool use. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7409–7416. IEEE.

Hu, N., Lou, Z., Englebienne, G., Kröse, B. J., et al. (2014). Learning to recognize human activities from soft labeled data. In *Robotics: Science and Systems*.

Huang, C.-M., Cakmak, M., and Mutlu, B. (2015). Adaptive coordination strategies for human-robot handovers. In *Robotics: science and systems*, volume 11. Rome, Italy.

Huber, M., Rickert, M., Knoll, A., Brandt, T., and Glasauer, S. (2008). Human-robot interaction in handing-over tasks. In *RO-MAN 2008-The 17th IEEE International Symposium on Robot and Human Interactive Communication*, pages 107–112. IEEE.

Hunt, G. R., Gray, R. D., Taylor, A. H., et al. (2013). Why is tool use rare in animals. *Tool use in animals: cognition and ecology*, pages 89–118.

Ijspeert, A. J., Nakanishi, J., Hoffmann, H., Pastor, P., and Schaal, S. (2013). Dynamical movement primitives: learning attractor models for motor behaviors. *Neural computation*, 25(2):328–373.

Ijspeert, A. J., Nakanishi, J., and Schaal, S. (2002). Movement imitation with nonlinear dynamical systems in humanoid robots. In *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, volume 2, pages 1398–1403. IEEE.

Ismail, Z. H., Sariff, N., and Hurtado, E. (2018). A survey and analysis of cooperative multi-agent robot systems: challenges and directions. In *Applications of Mobile Robots*, pages 8–14. IntechOpen.

Jain, R. and Inamura, T. (2013). Bayesian learning of tool affordances based on generalization of functional feature to estimate effects of unseen tools. *Artificial Life and Robotics*, 18(1):95–103.

Jamone, L., Damas, B., Santos-Victor, J., and Takanishi, A. (2013). Online learning of humanoid robot kinematics under switching tools contexts. In *2013 IEEE International Conference on Robotics and Automation*, pages 4811–4817. IEEE.

Jamone, L., Ugur, E., Cangelosi, A., Fadiga, L., Bernardino, A., Piater, J., and Santos-Victor, J. (2016). Affordances in psychology, neuroscience, and robotics: A survey. *IEEE Transactions on Cognitive and Developmental Systems*, 10(1):4–25.

Jelbert, S. A., Taylor, A. H., Cheke, L. G., Clayton, N. S., and Gray, R. D. (2014). Using the aesop's fable paradigm to investigate causal understanding of water displacement by new caledonian crows. *PloS one*, 9(3):e92895.

Kalainathan, D. and Goudet, O. (2019). Causal discovery toolbox: Uncover causal relationships in python. *arXiv preprint arXiv:1903.02278*.

Karayiannidis, Y., Smith, C., Vina, F. E., and Kragic, D. (2014). Online contact point estimation for uncalibrated tool use. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2488–2494. IEEE.

Katz, D., Pyuro, Y., and Brock, O. (2008). Learning to manipulate articulated objects in unstructured environments using a grounded relational representation. In *In Robotics: Science and Systems*. Citeseer.

Katz, D., Venkatraman, A., Kazemi, M., Bagnell, J. A., and Stentz, A. (2014). Perceiving, learning, and exploiting object affordances for autonomous pile manipulation. *Autonomous Robots*, 37(4):369–382.

Ke, L., Wang, J., Bhattacharjee, T., Boots, B., and Srinivasa, S. (2021). Grasping with chopsticks: Combating covariate shift in model-free imitation learning for fine manipulation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6185–6191. IEEE.

Kemp, C. C. and Edsinger, A. (2006). Robot manipulation of human tools: Autonomous detection and control of task relevant features. In *Proc. of the Fifth Intl. Conference on Development and Learning*, volume 42.

Kim, B., Wang, Z., Kaelbling, L. P., and Lozano-Pérez, T. (2019). Learning to guide task and motion planning using score-space representation. *The International Journal of Robotics Research*, 38(7):793–812.

Kim, S.-K., Jo, J., Oh, Y., Oh, S.-R., Srinivasa, S., and Likhachev, M. (2014). Robotic handwriting: multi-contact manipulation based on reactional internal contact hypothesis. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 877–884. IEEE.

Koay, K. L., Sisbot, E. A., Syrdal, D. S., Walters, M. L., Dautenhahn, K., and Alami, R. (2007). Exploratory study of a robot approaching a person in the context of handing over an object. In *AAAI spring symposium: multidisciplinary collaboration for socially assistive robotics*, pages 18–24. Stanford, CA.

Kobayashi, Y. and Hosoe, S. (2009). Planning-space shift learning: Variable-space motion planning toward flexible extension of body schema. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3107–3114. IEEE.

Kober, J., Mohler, B., and Peters, J. (2008). Learning perceptual coupling for motor primitives. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 834–839. IEEE.

Kocaoglu, M., Snyder, C., Dimakis, A. G., and Vishwanath, S. (2017). Causalgan: Learning causal implicit generative models with adversarial training. *arXiv preprint arXiv:1709.02023*.

Koch, J., Büsch, L., Gomse, M., and Schüppstuhl, T. (2022). A methods-time-measurement based approach to enable action recognition for multi-variant assembly in human-robot collaboration. *Procedia CIRP*, 106:233–238.

Koene, A., Endo, S., Remazeilles, A., Prada, M., and Wing, A. M. (2014a). Experimental testing of the coglaboration prototype system for fluent human-robot object handover interactions. In *The 23rd IEEE International Symposium on Robot and Human Interactive Communication*, pages 249–254. IEEE.

Koene, A., Remazeilles, A., Prada, M., Garzo, A., Puerto, M., Endo, S., and Wing, A. M. (2014b). Relative importance of spatial and temporal precision for user satisfaction in human-robot object handover interactions. In *Third International Symposium on New Frontiers in Human-Robot Interaction*.

Kokic, M., Stork, J. A., Haustein, J. A., and Kragic, D. (2017). Affordance detection for task-specific grasping using deep learning. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 91–98. IEEE.

Konstantinova, J., Krivic, S., Stilli, A., Piater, J., and Althoefer, K. (2017). Autonomous object handover using wrist tactile information. In *Annual Conference Towards Autonomous Robotic Systems*, pages 450–463. Springer.

Kormushev, P., Nenchev, D. N., Calinon, S., and Caldwell, D. G. (2011). Upper-body kinesthetic teaching of a free-standing humanoid robot. In *2011 IEEE International Conference on Robotics and Automation*, pages 3970–3975. IEEE.

Kroemer, O., Niekum, S., and Konidaris, G. (2021). A review of robot learning for manipulation: Challenges, representations, and algorithms. *The Journal of Machine Learning Research*, 22(1):1395–1476.

Kroemer, O., Ugur, E., Oztop, E., and Peters, J. (2012). A kernel-based approach to direct action perception. In *2012 IEEE international Conference on Robotics and Automation*, pages 2605–2610. IEEE.

Krützen, M., Mann, J., Heithaus, M. R., Connor, R. C., Bejder, L., and Sherwin, W. B. (2005). Cultural transmission of tool use in bottlenose dolphins. *Proceedings of the National Academy of Sciences*, 102(25):8939–8943.

Kulak, T., Silvério, J., and Calinon, S. (2020). Fourier movement primitives: an approach for learning rhythmic robot skills from demonstrations. In *Robotics: Science and Systems*.

Kutsuzawa, K., Sakaino, S., and Tsuji, T. (2017). A control system for a tool use robot: Drawing a circle by educing functions of a compass. *Journal of Robotics and Mechatronics*, 29(2):395–405.

Lagriffoul, F., Dimitrov, D., Bidot, J., Saffiotti, A., and Karlsson, L. (2014). Efficiently combining task and motion planning using geometric constraints. *The International Journal of Robotics Research*, 33(14):1726–1747.

Lagriffoul, F., Dimitrov, D., Saffiotti, A., and Karlsson, L. (2012). Constraint propagation on interval bounds for dealing with geometric backtracking. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 957–964. IEEE.

Lee, D., Kunori, H., and Nakamura, Y. (2008). Association of whole body motion from tool knowledge for humanoid robots. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2867–2874. IEEE.

Lee, Y.-H. and Song, K.-T. (2021). Real-time obstacle avoidance with a virtual torque approach for a robotic tool in the end effector. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8436–8442. IEEE.

Lestel, D. and Grundmann, E. (1999). Tools, techniques and animals: the role of mediations of actions in the dynamics of social behaviours. *Social science information*, 38(3):367–407.

Levihn, M. and Stilman, M. (2014). Using environment objects as tools: Unconventional door opening. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2502–2508. IEEE.

Li, J. K., Lee, W. S., and Hsu, D. (2018). Push-net: Deep planar pushing for objects with unknown physical properties. In *Robotics: Science and Systems*, volume 14, pages 1–9.

Li, R., Pham, D. T., Huang, J., Tan, Y., Qu, M., Wang, Y., Kerin, M., Jiang, K., Su, S., Ji, C., et al. (2020). Unfastening of hexagonal headed screws by a collaborative

robot. *IEEE Transactions on Automation Science and Engineering*, 17(3):1455–1468.

Li, X., Cao, R., Feng, Y., Chen, K., Yang, B., Fu, C.-W., Li, Y., Dou, Q., Liu, Y.-H., and Heng, P.-A. (2022). A sim-to-real object recognition and localization framework for industrial robotic bin picking. *IEEE Robotics and Automation Letters*, 7(2):3961–3968.

Lin, Y. and Sun, Y. (2015). Robot grasp planning based on demonstrated grasp strategies. *The International Journal of Robotics Research*, 34(1):26–42.

Lioutikov, R., Neumann, G., Maeda, G., and Peters, J. (2017). Learning movement primitive libraries through probabilistic segmentation. *The International Journal of Robotics Research*, 36(8):879–894.

Liu, Y., Gupta, A., Abbeel, P., and Levine, S. (2018). Imitation from observation: Learning to imitate behaviors from raw video via context translation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1118–1125. IEEE.

Lonsdorf, E. V. (2006). What is the role of mothers in the acquisition of termite-fishing behaviors in wild chimpanzees (pan troglodytes schweinfurthii)? *Animal cognition*, 9(1):36–46.

Lueddecke, T., Kulvicius, T., and Woergoetter, F. (2019). Context-based affordance segmentation from 2d images for robot actions. *Robotics and Autonomous Systems*, 119:92–107.

Lutscher, E. and Cheng, G. (2013). A practical approach to generalized hierarchical task specification for indirect force controlled robots. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1854–1859. IEEE.

Lynch, K. M. and Park, F. C. (2017). *Modern robotics*. Cambridge University Press.

Madry, M., Song, D., Ek, C. H., and Kragic, D. (2012). "robot bring me something to drink from": object representation for transferring task specific grasps. In *ICRA Workshop on semantic perception, mapping and exploration*, pages 1–6.

Maeda, G. J., Neumann, G., Ewerton, M., Lioutikov, R., Kroemer, O., and Peters, J. (2017). Probabilistic movement primitives for coordination of multiple human–robot collaborative tasks. *Autonomous Robots*, 41(3):593–612.

Mainprice, J., Sisbot, E. A., Siméon, T., and Alami, R. (2010). Planning safe and legible hand-over motions for human-robot interaction. In *IARP/IEEE-RAS/EURON workshop on technical challenges for dependable robots in human environments*.

Manuelli, L., Gao, W., Florence, P., and Tedrake, R. (2019). kpam: Keypoint affordances for category-level robotic manipulation. In *The International Symposium of Robotics Research*, pages 132–157. Springer.

Mar, T., Tikhanoff, V., Metta, G., and Natale, L. (2015). Self-supervised learning of grasp dependent tool affordances on the icub humanoid robot. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3200–3206. IEEE.

Mar, T., Tikhanoff, V., Metta, G., and Natale, L. (2017). Self-supervised learning of tool affordances from 3d tool representation through parallel som mapping. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 894–901. IEEE.

Marcus, G. (2018). Deep learning: A critical appraisal. *arXiv preprint arXiv:1801.00631*.

Mason, M. T. (2018). Toward robotic manipulation. *Annual Review of Control, Robotics, and Autonomous Systems*, 1(1).

Matsuzawa, T. (1999). Communication and tool use in chimpanzees: cultural and social contexts. *The Designs of Animal Communication*, pages 645–671.

McDermott, D., Ghallab, M., Howe, A., Knoblock, C., Ram, A., Veloso, M., Weld, D., and Wilkins, D. (1998). Pddl: The planning domain definition language. Technical report, Technical Report.

Migimatsu, T. and Bohg, J. (2020). Object-centric task and motion planning in dynamic environments. *IEEE Robotics and Automation Letters*, 5(2):844–851.

Moldovan, B., Moreno, P., and van Otterlo, M. (2013). On the use of probabilistic relational affordance models for sequential manipulation tasks in robotics. In *2013 IEEE International Conference on Robotics and Automation*, pages 1290–1295. IEEE.

Moldovan, B., Moreno, P., Van Otterlo, M., Santos-Victor, J., and De Raedt, L. (2012). Learning relational affordance models for robots in multi-object manipulation tasks. In *2012 ieee international conference on robotics and automation*, pages 4373–4378. IEEE.

Montesano, L., Lopes, M., Bernardino, A., and Santos-Victor, J. (2007). Modeling affordances using bayesian networks. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4102–4107. IEEE.

Montesano, L., Lopes, M., Bernardino, A., and Santos-Victor, J. (2008). Learning object affordances: from sensory–motor coordination to imitation. *IEEE Transactions on Robotics*, 24(1):15–26.

Muelling, K., Kober, J., and Peters, J. (2010). Learning table tennis with a mixture of motor primitives. In *2010 10th IEEE-RAS International Conference on Humanoid Robots*, pages 411–416. IEEE.

Murali, A., Liu, W., Marino, K., Chernova, S., and Gupta, A. (2020). Same object, different grasps: Data and semantic knowledge for task-oriented grasping. In *Conference on Robot Learning*.

Myers, A., Teo, C. L., Fermüller, C., and Aloimonos, Y. (2015). Affordance detection of tool parts from geometric features. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1374–1381. IEEE.

Nabeshima, C., Kuniyoshi, Y., and Lungarella, M. (2007). Towards a model for tool-body assimilation and adaptive tool-use. In *2007 IEEE 6th International Conference on Development and Learning*, pages 288–293. IEEE.

Nabeshima, C., Lungarella, M., and Kuniyoshi, Y. (2005). Timing-based model of body schema adaptation and its role in perception and tool use: A robot case study. In *Proceedings. The 4th International Conference on Development and Learning, 2005*, pages 7–12. IEEE.

Nagata, F., Watanabe, K., and Izumi, K. (2001). Furniture polishing robot using a trajectory generator based on cutter location data. In *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No. 01CH37164)*, volume 1, pages 319–324. IEEE.

Nair, L., Balloch, J., and Chernova, S. (2019a). Tool macgyvering: Tool construction using geometric reasoning. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 5837–5843. IEEE.

Nair, L., Srikanth, N. S., Erickson, Z. M., and Chernova, S. (2019b). Autonomous

tool construction using part shape and attachment prediction. In *Robotics: Science and Systems*.

Nakamura, T. and Nagai, T. (2010). Object concept modeling based on the relationship among appearance, usage and functions. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5410–5415. IEEE.

Neranon, P. (2018). Robot-to-human object handover using a behavioural control strategy. In *2018 IEEE 5th International Conference on Smart Instrumentation, Measurement and Application (ICSIMA)*, pages 1–6. IEEE.

Nishide, S., Tani, J., Takahashi, T., Okuno, H. G., and Ogata, T. (2011). Tool–body assimilation of humanoid robot using a neurodynamical system. *IEEE transactions on autonomous mental development*, 4(2):139–149.

Oakley, K. P. (1944). Man the tool-maker. *Proceedings of the Geologists' Association*, 55(2):115–118.

Oberst, M. and Sontag, D. (2019). Counterfactual off-policy evaluation with gumbel-max structural causal models. *arXiv preprint arXiv:1905.05824*.

Okada, K., Kojima, M., Sagawa, Y., Ichino, T., Sato, K., and Inaba, M. (2006). Vision based behavior verification system of humanoid robot for daily environment tasks. In *2006 6th IEEE-RAS International Conference on Humanoid Robots*, pages 7–12. IEEE.

Ortenzi, V., Controzzi, M., Cini, F., Leitner, J., Bianchi, M., Roa, M. A., and Corke, P. (2019). Robotic manipulation and the role of the task in the metric of success. *Nature Machine Intelligence*, 1(8):340–346.

Ortenzi, V., Cosgun, A., Pardi, T., Chan, W. P., Croft, E., and Kulić, D. (2021). Object handovers: a review for robotics. *IEEE Transactions on Robotics*.

Ortiz-Haro, J., Ha, J.-S., Driess, D., and Toussaint, M. (2022). Structured deep generative models for sampling on constraint manifolds in sequential manipulation. In *Conference on Robot Learning*, pages 213–223. PMLR.

Pack, L. and Lozano-Pérez, T. (2011). Hierarchical task and motion planning in the now. In *2011 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1470–1477. IEEE.

Paraschos, A., Daniel, C., Peters, J. R., and Neumann, G. (2013). Probabilistic movement primitives. *Advances in neural information processing systems*, 26.

Parastegari, S., Noohi, E., Abbasi, B., and Žefran, M. (2016). A fail-safe object handover controller. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2003–2008. IEEE.

Parker, S. T. and Gibson, K. R. (1977). Object manipulation, tool use and sensorimotor intelligence as feeding adaptations in cebus monkeys and great apes. *Journal of Human Evolution*, 6(7):623–641.

Pastor, P., Hoffmann, H., Asfour, T., and Schaal, S. (2009). Learning and generalization of motor skills by learning from demonstration. In *2009 IEEE International Conference on Robotics and Automation*, pages 763–768. IEEE.

Pastor, P., Kalakrishnan, M., Chitta, S., Theodorou, E., and Schaal, S. (2011). Skill learning and task outcome prediction for manipulation. In *2011 IEEE international conference on robotics and automation*, pages 3828–3834. IEEE.

Pearl, J. (2000). *Causality: models, reasoning and inference*, volume 29. Springer.

Pearl, J. (2018). Theoretical impediments to machine learning with seven sparks from the causal revolution. *arXiv preprint arXiv:1801.04016*.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830.

Peternel, L., Kim, W., Babič, J., and Ajoudani, A. (2017). Towards ergonomic control of human-robot co-manipulation and handover. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 55–60. IEEE.

Peternel, L., Petrič, T., and Babič, J. (2015). Human-in-the-loop approach for teaching robot assembly tasks using impedance control interface. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 1497–1502. IEEE.

Peters, J. and Schaal, S. (2006). Policy gradient methods for robotics. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2219–2225. IEEE.

Pfeiffer, K., Escande, A., and Kheddar, A. (2017). Nut fastening with a humanoid robot. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6142–6148. IEEE.

Prada, M., Remazeilles, A., Koene, A., and Endo, S. (2014). Implementation and experimental validation of dynamic movement primitives for object handover. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2146–2153. IEEE.

Qin, M., Brawer, J., and Scassellati, B. (2021). Rapidly learning generalizable and robot-agnostic tool-use skills for a wide range of tasks. *Frontiers in Robotics and AI*, 8.

Qin, M., Brawer, J., and Scassellati, B. (2022a). Task-oriented robot-to-human han-

dovers in collaborative tool-use tasks. In *2022 31th IEEE International Conference on Robot & Human Interactive Communication (RO-MAN)*. IEEE.

Qin, M., Brawer, J., and Scassellati, B. (2022b). (under review) using task, affordance, and motion planning (taamp) to detect infeasible or limited solutions in affordance-constrained environments. *Under Review*.

Qin, Z., Fang, K., Zhu, Y., Fei-Fei, L., and Savarese, S. (2020). Keto: Learning keypoint representations for tool manipulation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7278–7285. IEEE.

Raessa, M., Sánchez, D., Wan, W., Petit, D., and Harada, K. (2019). Teaching a robot to use electric tools with regrasp planning. *CAAI Transactions on Intelligence Technology*, 4(1):54–63.

Ramirez-Amaro, K., Beetz, M., and Cheng, G. (2014a). Automatic segmentation and recognition of human activities from observation based on semantic reasoning. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5043–5048. IEEE.

Ramirez-Amaro, K., Beetz, M., and Cheng, G. (2015). Understanding the intention of human activities through semantic perception: observation, understanding and execution on a humanoid robot. *Advanced Robotics*, 29(5):345–362.

Ramirez-Amaro, K., Inamura, T., Dean-León, E., Beetz, M., and Cheng, G. (2014b). Bootstrapping humanoid robot skills by extracting semantic representations of human-like activities from virtual reality. In *2014 IEEE-RAS International Conference on Humanoid Robots*, pages 438–443. IEEE.

Rasheed, A. A. A., Abdullah, M. N., and Al-Araji, A. S. (2022). A review of multi-agent mobile robot systems applications. *International Journal of Electrical & Computer Engineering (2088-8708)*, 12(4).

Robertsson, A., Olsson, T., Johansson, R., Blomdell, A., Nilsson, K., Haage, M., Lauwers, B., De Baerdemaeker, H., Brogardh, T., and Brantmark, H. (2006). Implementation of industrial robot force control case study: high power stub grinding and deburring. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2743–2748. IEEE.

Robinson, R. (1973). Counting labeled acyclic digraphs. In *New Directions in the Theory of Graphs: Proc of the Third Ann Arbor Conf. on Graph Theory*, pages 239–273. Academic Press.

Rossi, F., Bandyopadhyay, S., Wolf, M., and Pavone, M. (2018). Review of multi-agent algorithms for collective behavior: a structural taxonomy. *IFAC-PapersOnLine*, 51(12):112–117.

Rozo, L., Jiménez, P., and Torras, C. (2013). Force-based robot learning of pouring skills using parametric hidden markov models. In *9th International Workshop on Robot Motion and Control*, pages 227–232. IEEE.

Ruiz, E. and Mayol-Cuevas, W. (2018). Where can i do this? geometric affordances from a single example with the interaction tensor. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2192–2199. IEEE.

Rusinkiewicz, S. and Levoy, M. (2001). Efficient variants of the icp algorithm. In *Proceedings third international conference on 3-D digital imaging and modeling*, pages 145–152. IEEE.

Saito, N., Kim, K., Murata, S., Ogata, T., and Sugano, S. (2018). Tool-use model considering tool selection by a robot using deep learning. In *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*, pages 270–276. IEEE.

Sammut, C., Sheh, R., Haber, A., and Wicaksono, H. (2015). The robot engineer. In *ILP (Late Breaking Papers)*, pages 101–106.

Sarikaya, D., Corso, J. J., and Guru, K. A. (2017). Detection and localization of robotic tools in robot-assisted surgery videos using deep neural networks for region proposal and detection. *IEEE transactions on medical imaging*, 36(7):1542–1549.

Schaal, S. (2006). Dynamic movement primitives-a framework for motor control in humans and humanoid robotics. In *Adaptive motion of animals and machines*, pages 261–280. Springer.

Schoeler, M. and Wörgötter, F. (2015). Bootstrapping the semantics of tools: Affordance analysis of real world objects on a per-part basis. *IEEE Transactions on Cognitive and Developmental Systems*, 8(2):84–98.

Shao, L., Migimatsu, T., Zhang, Q., Yang, K., and Bohg, J. (2021). Concept2robot: Learning manipulation concepts from instructions and human demonstrations. *The International Journal of Robotics Research*, 40(12-14):1419–1434.

Shi, C., Shiomi, M., Smith, C., Kanda, T., and Ishiguro, H. (2013). A model of distributional handing interaction for a mobile robot. In *Proceedings of Robotics: Science and Systems*, Berlin, Germany.

Shumaker, R. W., Walkup, K. R., and Beck, B. B. (2011). *Animal tool behavior: the use and manufacture of tools by animals*. JHU Press.

Siciliano, B. and Khatib, O. (2016). Robotics and the handbook. In *Springer Handbook of Robotics*, pages 1–6. Springer.

Sinapov, J. and Stoytchev, A. (2007). Learning and generalization of behavior-grounded tool affordances. In *2007 IEEE 6th International Conference on Development and Learning*, pages 19–24. IEEE.

Sinapov, J. and Stoytchev, A. (2008). Detecting the functional similarities between

tools using a hierarchical representation of outcomes. In *2008 7th IEEE International Conference on Development and Learning*, pages 91–96. IEEE.

Sisbot, E. A., Marin-Urias, L. F., Broquere, X., Sidobre, D., and Alami, R. (2010). Synthesizing robot motions adapted to human presence. *International Journal of Social Robotics*, 2(3):329–343.

Song, D., Ek, C. H., Huebner, K., and Kragic, D. (2011a). Embodiment-specific representation of robot grasping using graphical models and latent-space discretization. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 980–986. IEEE.

Song, D., Ek, C. H., Huebner, K., and Kragic, D. (2011b). Multivariate discretization for bayesian network structure learning in robot grasping. In *2011 IEEE International Conference on Robotics and Automation*, pages 1944–1950. IEEE.

Song, D., Ek, C. H., Huebner, K., and Kragic, D. (2015). Task-based robot grasp planning using probabilistic inference. *IEEE transactions on robotics*, 31(3):546–561.

Song, D., Huebner, K., Kyrki, V., and Kragic, D. (2010). Learning task constraints for robot grasping using graphical models. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1579–1585. IEEE.

Spagnoletti, N., Visalberghi, E., Ottoni, E., Izar, P., and Fragaszy, D. (2011). Stone tool use by adult wild bearded capuchin monkeys (cebus libidinosus). frequency, efficiency and tool selectivity. *Journal of Human Evolution*, 61(1):97–107.

Spirtes, P., Glymour, C. N., Scheines, R., and Heckerman, D. (2000). *Causation, prediction, and search*. MIT press.

Srivastava, S., Fang, E., Riano, L., Chitnis, R., Russell, S., and Abbeel, P. (2014). Combined task and motion planning through an extensible planner-independent interface layer. In *2014 IEEE international conference on robotics and automation (ICRA)*, pages 639–646. IEEE.

St. Amant, R. and Horton, T. E. (2008). Revisiting the definition of animal tool use. *Animal Behaviour*, 75(4):1199–1208.

Stilman, M. and Kuffner, J. (2008). Planning among movable obstacles with artificial constraints. *The International Journal of Robotics Research*, 27(11-12):1295–1307.

Stilman, M., Schamburek, J.-U., Kuffner, J., and Asfour, T. (2007). Manipulation planning among movable obstacles. In *Proceedings 2007 IEEE international conference on robotics and automation*, pages 3327–3332. IEEE.

Stoytchev, A. (2003). Computational model for an extendable robot body schema. Technical report, Georgia Institute of Technology.

Stoytchev, A. (2005a). Behavior-grounded representation of tool affordances. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pages 3071–3076. IEEE.

Stoytchev, A. (2005b). Toward learning the binding affordances of objects: A behavior-grounded approach. In *Proceedings of AAAI symposium on developmental robotics*, pages 17–22. Stanford University Menlo Park.

Stoytchev, A. (2007). *Robot tool behavior: A developmental approach to autonomous tool use*. Georgia Institute of Technology.

Stoytchev, A. (2008). Learning the affordances of tools using a behavior-grounded approach. In *Towards Affordance-Based Robot Control*, pages 140–158. Springer.

Strabala, K., Lee, M. K., Dragan, A., Forlizzi, J., Srinivasa, S. S., Cakmak, M., and Micelli, V. (2013). Toward seamless human-robot handovers. *Journal of Human-Robot Interaction*, 2(1):112–132.

Stückler, J. and Behnke, S. (2014a). Adaptive tool-use strategies for anthropomorphic service robots. In *2014 IEEE-RAS International Conference on Humanoid Robots*, pages 755–760. IEEE.

Stückler, J. and Behnke, S. (2014b). Efficient deformable registration of multi-resolution surfel maps for object manipulation skill transfer. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 994–1001. IEEE.

Stückler, J. and Behnke, S. (2015). Perception of deformable objects and compliant manipulation for service robots. In *Soft Robotics*, pages 69–80. Springer.

Stückler, J., Droeschel, D., Gräve, K., Holz, D., Schreiber, M., Topalidou-Kyniazopoulou, A., Schwarz, M., and Behnke, S. (2013). Increasing flexibility of mobile manipulation and intuitive human-robot interaction in robocup@ home. In *Robot Soccer World Cup*, pages 135–146. Springer.

Stückler, J., Schwarz, M., and Behnke, S. (2016). Mobile manipulation, tool use, and intuitive interaction for cognitive service robot cosero. *Frontiers in Robotics and AI*, 3:58.

Su, Y.-H., Huang, K., and Hannaford, B. (2018). Real-time vision-based surgical tool segmentation with robot kinematics prior. In *2018 International Symposium on Medical Robotics (ISMR)*, pages 1–6. IEEE.

Sukhoy, V., Georgiev, V., Wegter, T., Sweidan, R., and Stoytchev, A. (2012). Learning to slide a magnetic card through a card reader. In *2012 IEEE International Conference on Robotics and Automation*, pages 2398–2404. IEEE.

Takahashi, K., Kim, K., Ogata, T., and Sugano, S. (2017). Tool-body assimilation model considering grasping motion through deep learning. *Robotics and Autonomous Systems*, 91:115–127.

Takeuchi, Y., Ge, D., and Asakawa, N. (1993). Automated polishing process with a human-like dexterous robot. In *[1993] Proceedings IEEE International Conference on Robotics and Automation*, pages 950–956. IEEE.

Tee, K. P., Cheong, S., Li, J., and Ganesh, G. (2022). A framework for tool cognition in robots without prior tool learning or observation. *Nature Machine Intelligence*, pages 1–11.

Tee, K. P., Li, J., Chen, L. T. P., Wan, K. W., and Ganesh, G. (2018). Towards emergence of tool use in robots: Automatic tool recognition and use without prior tool learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6439–6446. IEEE.

Tikhanoff, V., Pattacini, U., Natale, L., and Metta, G. (2013). Exploring affordances and tool use on the icub. In *2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pages 130–137. IEEE.

Toussaint, M. A., Allen, K. R., Smith, K. A., and Tenenbaum, J. B. (2018). Differentiable physics and stable modes for tool-use and manipulation planning. In *Robotics: Science and Systems Foundation*.

Tsuji, T., Ohkuma, J., and Sakaino, S. (2015). Dynamic object manipulation considering contact condition of robot with tool. *IEEE Transactions on Industrial Electronics*, 63(3):1972–1980.

Turpin, D., Wang, L., Tsogkas, S., Dickinson, S., and Garg, A. (2021). Gift: Generalizable interaction-aware functional tool affordances without labels. *arXiv preprint arXiv:2106.14973*.

Van Lawick-Goodall, J. (1970). Tool-using in primates and other vertebrates. In *Advances in the Study of Behavior*, volume 3, pages 195–249. Elsevier.

Visalberghi, E. and Fragaszy, D. (2006). What is challenging about tool use? the capuchin's perspective. *Comparative cognition: Experimental explorations of animal intelligence*, pages 529–552.

Vogel, J., Takemura, N., Höppner, H., van der Smagt, P., and Ganesh, G. (2017). Hitting the sweet spot: Automatic optimization of energy transfer during tool-held hits. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1549–1556. IEEE.

Wells, A. M., Dantam, N. T., Shrivastava, A., and Kavraki, L. E. (2019). Learning feasibility for task and motion planning in tabletop environments. *IEEE robotics and automation letters*, 4(2):1255–1262.

Wicaksono, H. (2020). *A Relational Approach to Tool Creation by a Robot.* PhD thesis, University of New South Wales, Sydney, Australia.

Wicaksono, H. and Sammut, C. (2016). Relational tool use learning by a robot in a real and simulated world. In *Proceedings of ACRA*.

Wimpenny, J. H., Weir, A. A., Clayton, L., Rutz, C., and Kacelnik, A. (2009). Cognitive processes associated with sequential tool use in new caledonian crows. *PLoS One*, 4(8):e6471.

Wölfel, K. and Henrich, D. (2018). Grounding verbs for tool-dependent, sensor-based robot tasks. In *2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 378–383. IEEE.

Xie, A., Ebert, F., Levine, S., and Finn, C. (2019). Improvisation through physical

understanding: Using novel objects as tools with visual foresight. In *Proceedings of Robotics: Science and Systems*, FreiburgimBreisgau, Germany.

Xiong, C., Shukla, N., Xiong, W., and Zhu, S.-C. (2016). Robot learning with a spatial, temporal, and causal and-or graph. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2144–2151. IEEE.

Xue, Y. and Jia, Y.-B. (2020). Gripping a kitchen knife on the cutting board. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9226–9231. IEEE.

Yamazaki, K., Ueda, R., Nozawa, S., Mori, Y., Maki, T., Hatao, N., Okada, K., and Inaba, M. (2010). System integration of a daily assistive robot and its application to tidying and cleaning rooms. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1365–1371. IEEE.

Zech, P., Haller, S., Lakani, S. R., Ridge, B., Ugur, E., and Piater, J. (2017). Computational models of affordance in robotics: a taxonomy and systematic classification. *Adaptive Behavior*, 25(5):235–271.

Zhu, Y., Zhao, Y., and Chun Zhu, S. (2015). Understanding tools: Task-oriented object modeling, learning and recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2855–2864.

Zhu, Z. and Hu, H. (2018). Robot learning from demonstration in robotic assembly: A survey. *Robotics*, 7(2):17.