NORTHWESTERN UNIVERSITY

Active Intent Disambiguation, Control Interpretation, and Arbitration for Assistive Robotics

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

DOCTOR OF PHILOSOPHY

Field of Mechanical Engineering

By

Deepak Edakkattil Gopinath

EVANSTON, ILLINOIS

June 2022

ABSTRACT

Active Intent Disambiguation, Control Interpretation, and Arbitration for Assistive Robotics

Deepak Edakkattil Gopinath

The goal of this dissertation is to develop models, algorithms, and interaction protocols to improve the efficacy and quality of Human-Autonomy Interaction (HAI) in the domain of assistive robotics. In this domain, the most common control paradigm is that of *manual teleoperation* using control interfaces such as joysticks, switch-based head arrays, and sip-and-puff. However, manual teleoperation can become physically and cognitively burdensome to the human due to the limitations of the control interface, inherent complexities of the assistive machine, and motor impairments. Although introducing full robotics autonomy could be a viable approach to ameliorate these challenges, a more attractive control paradigm is that of *shared autonomy* in which the human and the autonomous agent *share* control responsibilities, thereby ensuring that the human still has agency.

This dissertation focuses on two important aspects of a shared-autonomy assistive system, namely, *intent inference* and *control arbitration*. In a shared autonomy human-robot team, the autonomous agent's ability to *infer human intent* accurately is critical for

providing correct and timely assistance to the human. However, due to the sparsity, low information content, and imperfections in the control signals, accurate intent inference is rather difficult. To improve the autonomous agent's ability to infer user intent this dissertation introduces the idea of *intent disambiguation*. Algorithms and protocols for intent disambiguation are designed to alter or nudge the human's decision-making context in a principled manner so that the signals generated by the human contain more information regarding underlying intent. Intent disambiguation algorithms endow the autonomous agent with active learning capabilities. More meaningful and informative teleoperation signals from the human enables the autonomous agent to perform accurate intent inference which in turn improves the assistance provided *to* the human. In Chapter 5 a heuristic approach for intent disambiguation that reasons over the space of control modes is introduced. Building upon this idea, in Chapter 6 a more rigorous formalism of intent disambiguation that is grounded in information-theoretic principles is presented.

Another important topic that this dissertation addresses is the question of how to share robot control between the human and the autonomous agent. Upon successful user intent inference the autonomous agent can rely on different types of autonomous controllers to generate appropriate assistance towards the user's intended goal. Yet, in a shared control setting the autonomous control signal needs to be combined with the human's control signal in some fashion. *Control sharing* in a shared control assistive system lies on a continuum with full manual teleoperation on one end of the spectrum to fully autonomous solution on the other. How exactly should control be arbitrated between the human and the autonomous agent is a critical question that impacts the overall performance of the human-autonomy team. Implicitly, the question could be framed as a *constrained optimization* problem in which the optimization objective is to balance both task-related metrics (such as successful task completion, minimizing effort, and minimizing energy expenditure) and subjective metrics (such as satisfaction, sense of agency, and user's assistive preferences). To this end, in Chapter 7 we propose a *human-in-the-loop* solution to this constrained optimization problem in which the human uses an easy-to-understand, interpretable protocol to optimize the arbitration parameters according to their own optimality criteria to achieve their desired outcomes.

In the assistive setting, robot teleoperation is facilitated using physical control interfaces. Autonomous agents benefit a great deal if they can make a distinction between *conceptual* and *physical* aspects of interface operation. To address this problem, this thesis introduces the notion of *interface awareness* into probabilistic models of interface-mediated robot teleoperation. In Chapter 4, the autonomous agent relies on an interface-aware robot teleoperation model to reason about user intent at the level of interface signals and then provide appropriate types of modifications and corrections to faulty or unintended interface operation that arise due to lack of motor skill, motor impairments or inherent noise in the physical interface.

As a last contribution, Chapter 8 presents a software tool for conducting web-based crowdsourced human-robot interaction experiments. Data collection in the domain of assistive robotics can be an arduous task and the goal of this tool is to facilitate rapid prototyping and testing of novel algorithms developed for assistive autonomous agents.

To summarize, the work presented in this dissertation tackles different challenges that arise in human-autonomy interaction in the context of interface-mediated assistive robot teleoperation and proposes algorithms and protocols to improve the overall human experience of interacting with an assistive robot.

Acknowledgements

This dissertation would not have been possible without the unconditional support, trust, love, and encouragement from my parents K. Gopinathan, Rajiny Gopinath and my big brother Brijesh Gopinath. Growing up, they provided me with an environment in which I could flourish without restrictions and dream big. They inculcated in me a sense of curiosity and responsibility that played a big role in my artistic and academic pursuits in the last twenty years or so. In moments of self-doubt, all I need to do to feel energized and confident is to close my eyes and think of them.

I would not know what my life would have been like during my Ph.D. years, if I was not with my beloved partner Ranjani Prabhakar. You are my rock. You have made me a better person, have kept me grounded, and helped me understand myself a lot better with your unconditional support and love. I cannot wait for the next phase of our lives to begin. I also want to thank Ranjani's parents, Krishnaswamy Prabhakar and Kalyani Prabhakar for always making me feel special. I also want to thank Dr. Janani Prabhakar and Dr. Sangeet Khemlani for always being there for me and ready to geek out on anything!

Little did I know that when I walked into the Manufacturing Institute Auditorium at Georgia Tech to attend Dr. Brenna Argall's IRIM seminar on assistive robots in 2015 that my life was going to take a turn. What started off as a chance meeting turned into an exciting intellectual adventure over the last six years. I could not have asked for a better advisor than Dr. Argall during my Ph.D. I will forever be grateful that she let me be and choose my own research path during my time in Argallab. The singlemost thing that I looked forward to when I started my Ph.D was the random walk that I wanted to perform in the space of ideas and knowledge; and unsurprisingly Dr. Argall was there with me every step along the way. She has been there as a patient listener, always offering words of support and encouragement, especially in my lowest moments.

The summer of 2018 was when I first had the chance to work in an industry setting as a research intern at Toyota Research Institute (TRI) in Cambridge, MA. To say it was baptism by fire is an understatement. I am extremely thankful to Dr. Guy Rosman and Dr. Simon Stent for giving me the opportunity to be part of the Risk Aware Driving team and putting faith in my abilities and entrusting me with those projects. It almost felt like I was doing a parallel Ph.D and I could not have asked for a more enriching, thrilling, and exciting experience. I also feel extremely privileged to have had a close working relationship with TRI ever since my first internship in 2018.

In the summer of 2017, I attended a two-week summer school in computational neuroscience (Cosmo 2017) organized by Dr. Konrad Kording, Dr. Gunnar Blohm, and Dr. Paul Schrater at University of Minnesota. The most energising and intellectually stimulating two weeks ever! Cosmo 2017 changed my scientific perspective for good. I developed a better sense for computational modeling and it helped me open up to the world of psychology, cognitive science, and neuroscience and inspired me to pursue new directions in my own research. I am so grateful to be part of that wonderful cohort. You all made a huge impact on me!

My transition into AI and robotics research came by way of my creative and technological explorations in arts, especially in music. I am deeply indebted to the person who kickstarted it all, Dr. Richard Boulanger. Dr. B (as he is fondly known) once told me that when you do a Ph.D, although the size of the set of things you know might increase, its relative size to everything that you *do not* know will shrink. I now fully understand what he meant then! I also want to thank Dr. Gil Weinberg for introducing me into the world of robotics and giving me the opportunity to be part of the amazing robotic musicianship group at Georgia Tech. Being part of the robotic drumming prosthesis project is what sparked my interest in human-robot interaction, shared autonomy, and assistive robotics. I also want to thank two of my dearest music teachers, the late Ralph Peterson Jr. and the late Tibor Pusztai for all the wisdom that they have imparted to me.

Pursuing a Ph.D can be a lonely pursuit and it was important for me to have the friends that I have in my life; knowing that I can always reach out to them for anything. I want to thank my dearest friend Dr. Vishnu Sreekumar for a lifetime of friendship, brotherhood, and inspiration. I won't forget all those hours you spent explaining Bayes Theorem to me. I think I put that to good use in this thesis! You should be proud!

I also want to thank the wonderful community of friends and squash players that I met at Lakeshore Sports and Fitness in downtown Chicago. Particularly, I am so grateful to have met my coach and dear friend Luis Sanchez. I couldn't have asked for a better person to be my best friend outside of my work! Always full of fun, practical advice, and wisdom!

I feel so fortunate to be among my colleagues and friends in Argallab. I want to sincerely thank my current and former labmates, Alexander Broad, Mahdieh Nejati Javaremi, Siddarth Jain, Ahmetcan Erdogan, Michael Young, Chris Miller, Jongmin Lee, Temesgen Gebrekristos, Andrew Thompson, and Larisa Loke for the wonderful times. I will miss our lunch conversations, opinionated debates, and occasional happy hours on Fridays. I have learned a lot from you all and your support has been invaluable for me during these last few years.

Lastly, I want to thank the administrative staff at Shirley Ryan AbilityLab and Northwestern University's Mechanical Engineering department for making our lives easier. I am grateful to everyone who took time off their busy lives to be participants in the studies that I conducted.

Dedication

To my dear parents

Table of Contents

List of Tables

List of Figures

CHAPTER 1

Introduction

Robots are ubiquitous in modern society and have revolutionized the relationship between humans and machines. Robotics automation, particularly in industrial settings, has resulted in enhanced manufacturing productivity, improved quality assurance, and increased worker safety. As a result of robotics automation, manufacturing jobs are undergoing a paradigm shift as workers can switch their focus to more creative aspects of manufacturing such as engineering, programming, and management. However, compared to a few decades ago, in the present day, robots have transitioned out of the rigid, structured, and specialized industrial environments to the more rich, complex and unpredictable day-to-day human environments and have impacted diverse domains of human endeavor such as healthcare [**101**], entertainment [**67**], and home robotics [**59**].

The impact is even more significant in the domain of assistive and rehabilitation robotics in which the potential to drastically enhance the quality of life for people suffering from motor impairments is immense. Numerous assistive and rehabilitation machines ranging from powered and smart wheelchairs [**121**], exoskeletons [**130**] and assistive robotic arms [**96**] can help to promote independence, boost self-esteem [**36**] and extend mobility and manipulation capabilities of motor-impaired individuals. These machines help such individuals to regain a sense of agency and can revolutionize the way they interact with society at large. The work presented in this thesis concerns this particular application domain of robotics.

The standard usage of these assistive machines still relies on manual teleoperation by the human. Manual teleoperation refers to the fact that motion control of the robot is *fully* entrusted with the human. In the assistive domain, manual teleoperation is typically enacted through a *control interface* such as a joystick, switch-based head array or a sip-and-puff. In such settings these assistive robots are complex mechanical devices without any kind of intelligent capabilities and are treated as extensions of human motor abilities. Although it is true that teleoperated assistive robots themselves can be a valuable tool to enhance the quality of Activities of Daily Living (ADL) tasks [**91**], one of the most difficult conundrums is that the greater the motor impairment of the end-user, the more limited the control interfaces that are available for them to use [**7**]. The manual control of these assistive machines can become physically and cognitively difficult due to the low-dimensionality, sparsity and bandwidth of the control interface. The inherent complexity in robot dynamics, uncertainties in the physical world, and the physical limitations of the user add to the difficulty as well. There is clearly a need for improving the human-robot interaction experience in the assistive setting for wider user adoption and efficacy of these machines.

This thesis recognizes the need to address this gap and focuses on the development of computational models, algorithms, and interaction protocols that aim to improve the human-autonomy interaction in the assistive domain. A naive way to address this problem would be to introduce full *robot autonomy*; the intelligence that enables robots to accomplish a task independently without requiring explicit instructions from a human. This holds considerable promise as a tool to offset (and in some cases restore) the above-mentioned limitations. Advances in the fields of machine learning and artificial intelligence have helped to endow these assistive machines with better decision-making and prediction capabilities when interacting with humans in real-world settings [**123**].

However, there is a growing consensus that the users of assistive technologies *do not* prefer to cede full authority to the autonomous partner during task execution. Particularly, for motor-impaired users, the need to retain a sense of agency using residual motor function is important for improving self-belief and uplifting self-esteem [**29**]. Therefore, in such cases, the introduction of *shared control* seeks to find a middle ground between full teleoperation and full autonomy by offloading only some aspects of task execution to the autonomy [**43, 170**].

1.1. Motivation

Shared-control describes a general approach for having an autonomous agent alongside a human in the control loop of a machine, for example an assistive robot. In a shared-control system, the primary function of the autonomous agent is to compensate for any shortcomings in overall task performance, improve user experience, ensure safety, and to offload the cognitive and physical burden of full manual control of the machine. The task responsibility is split between the user and the autonomous agent usually with an aim to reduce human effort in accomplishing the task. The exact nature of workload distribution depends on the robotic platform, the domain, and user needs. For example, it is possible that humans and autonomous agents complement each other's capabilities and coordinate their actions in a hierarchical fashion. In a smart wheelchair navigation task, the human can be responsible for high-level actions such as selecting a navigation goal in the world, and the autonomous agent can be entrusted to perform low-level planning and control that would drive the wheelchair to the chosen navigation goal. Another paradigm that can be adopted is a blending-based shared control framework in which the human and autonomous agent actions are arbitrated at the control signal (or the policy) level directly. In a blending-based shared control system, the control signals from the human and the autonomous agent are issued in the same space. Yet another approach is to adopt a turn-taking paradigm in which the autonomous agent and the human switch roles intermittently. Turn-taking allows the human to take rest (important for long-term continued use) and also provides an opportunity for the human to observe the autonomous agent's actions without any interference with their own actions potentially helping the human to build better mental models of the autonomous agent's behavior.

1.1.1. Intent Disambiguation

Regardless of the control allocation paradigm, the usefulness of shared-control human-machine systems typically relies on the autonomous agent's ability to infer the user's underlying needs and intentions as clearly as possible. Autonomous agents perform intent inference in a passive manner by combining information available through various sensor streams. Particularly in the assistive domain, inferring user intent becomes significantly challenging as the user input is low-dimensional (due to inherent limitations in the available control interfaces and motor abilities), filtered (due to electromechanical filtering to reduce oscillations and noise), or sparse (due to physical limitations, motor impairments and noisy actuation channels that result in signal dropouts). When the information content available in sensor streams becomes sparse then it is helpful to design human-autonomy interaction protocols that can perform *active* intent inference. By adding an active element to the interaction, the autonomous agent is endowed with the capability to alter or nudge the human's decision-making context such that the humans are coaxed to generate actions that will reveal the underlying intent to the autonomous agent more clearly.

To that end, this thesis develops shared-control paradigms in which autonomous agents are endowed with algorithms to perform *intent disambiguation*, which is a mechanism to perform active inference of human internal state. This thesis presents two different formulations of intent disambiguation: (a) a heuristic approach to disambiguate intent over the space of control modes, and (b) an information-theoretic approach to disambiguate user intent over the entire state space. Additionally, this thesis also presents new methods to perform intent inference inspired by dynamic field theory, particularly effective for dealing with sparse signals.

1.1.2. User-Driven Shared Control

Another critical design consideration for shared-control algorithms is the proper characterization and customization of individual instances of human-autonomy teams. The need for customization arises because users differ in their physical abilities, knowledge level, preferences, and desired amount of assistance. Therefore, a one-size-fits-all approach is unlikely to work for different end users. To address the problem of customization of assistance levels, this thesis develops a lightweight, interpretable, and interactive verbal procedure which leverages the human in the control loop to optimize for appropriate levels of assistance according to user preference. The procedure allows for continual adjustment of assistance parameters as and when required by the user without having to retrain or redesign the autonomous agent.

1.1.3. Interface-Level Intent Inference

During control-interface mediated robot teleoperation, the idiosyncrasies of the control interface have a profound effect on how end users can communicate their intent effectively. Each control interface relies on a unique physical activation mechanism and places different physical and cognitive requirements on the end user's capabilities. Mechanical wear and tear can also degrade the quality of signal output from an interface. An autonomous agent tasked with the responsibility of assisting the user during task execution will benefit from utilizing more accurate models that capture the distinction between the user's conceptual understanding of how to use a control interface and the physical understanding (motor skill) of how to operate a control interface. To that end, this thesis also introduces an interface-aware probabilistic teleoperation model to perform interface-level intent inference.

To summarize, the high-level research questions this thesis aims to tackle are as follows:

- What algorithms and human-autonomy interaction protocols can improve the autonomous agent's ability to accurately infer intent when the human input is low-dimensional and sparse?
- How can we identify and characterize the limitations in the control of and the operation of a control interface and their impact on various downstream tasks in the context of shared-control robot teleoperation?

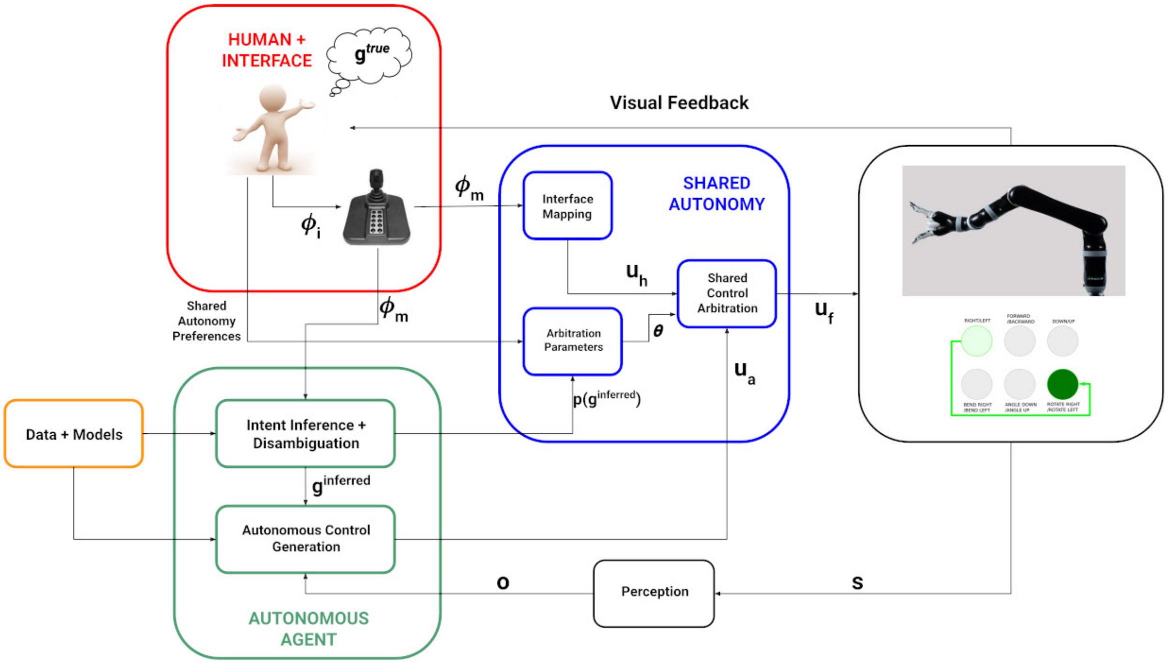


Figure 1.1. Key components of a shared-autonomy human-robot system. Chapter 4 addresses the human's interaction with the control interface. Chapters 5 and 6 address various challenges related to intent inference from low-dimensional control signals by developing new algorithms for intent inference and disambiguation. The question of optimizing the arbitration function parameters is addressed in Chapter 7. Chapter 8 proposes a crowdsourced solution to the problem of large-scale data collection for human-robot interaction experiments.

- How can we leverage the human in the loop to optimize the arbitration parameters in a shared control system with low computational overhead?

1.2. Dissertation Outline

This thesis starts with an in-depth discussion of related work and other helpful background material in Chapter 2. This sets the context in order to situate the specific problems addressed in the later chapters properly. Chapter 3 formalizes the shared-control human robot system in which robot teleoperation is mediated using a control interface.

Chapter 4 explores the idea of *interface-awareness* and investigates its role in improving interface-level intent inference in the context of control-interface mediated robot teleoperation.

Insights: The signal that emanates from the control interface during robot teleoperation is masked by the physical and cognitive limitations of the user and the mechanical particularities of the interface which in turn have profound impact on various downstream tasks that rely on this signal.

Contributions: This chapter introduces a generative model for the human's physical interaction with a control interface and distinguishes between the *conceptual* understanding of and the *physical* aspect of operating a control interface. By reasoning over the unobserved human intentions at the level of interface signals using model-based Bayesian inference techniques, a new assistance system that provides customized modifications to the measured interface signals is presented.

Results: This chapter presents a set of simulation-based results in which different variants of the model are investigated. The results of a 10-person human subject study show that the proposed assistance paradigms help to significantly reduce task completion time, number of mode switches, cognitive workload, and improve overall user satisfaction when operating the robot. The work presented in this chapter is done in equal collaboration with Mahdieh Nejati Javaremi. The details of workload distribution is presented in the chapter.

Chapter 5 addresses various challenges that arise when autonomous agents attempt to perform user intent inference from limited bandwidth, low fidelity interface-level signals generated by the human via the control interface. Particularly, the chapter develops algorithms and interaction protocols for autonomous agents to help themselves before assisting end users.

Insights: This chapter introduces the notion of *inverse legibility* in which the assistance provided by the autonomous agent is intended to *extract* more legible, intent-expressive control commands *from* the human. These intent expressive commands are then used for *intent disambiguation* via control mode selection.

Contributions: This chapter develops a heuristic metric to characterize control modes according to their ability to disambiguate human intent. An interaction protocol is also proposed in which the user is in control of when to activate the disambiguation algorithm. Additionally, the chapter also presents a novel approach to intent inference inspired by *dynamic field theory.*

Results: The results of an 8-person human subject study reveal that the disambiguation system helps to significantly reduce task effort and is of greater utility for more limited control interfaces and complex tasks.

Chapter 6 builds upon the need for intent disambiguation presented in Chapter 5 and presents an algorithm for interface-aware intent disambiguation over environment states that comprise of both robot locations as well as control modes.

Insights: The main insight presented in this chapter is that intent disambiguation can be cast in information-theoretic terms and can be performed not just over the subset of control modes but instead over the entire world state that comprises of robot location as well. Additionally, by situating the algorithm within a turn-taking based shared-autonomy paradigm the autonomous agent is able to help itself and enhance its ability to infer user's goal more accurately.

Contributions: The chapter presents (a) an interface-aware intent disambiguation algorithm grounded in information-theoretic principles and (b) a turn-taking based adaptive assistance protocol that utilizes the intent disambiguation algorithm alongside blending-based shared-control.

Results: This chapter presents results in two different settings. Pure simulation-based results validate the design of the disambiguation metric. Results of a 9-person human subject study in a simulated robot environment reveal that a turn-taking based shared control paradigm endowed with the disambiguation capabilities results in lower task effort as measured by objective metrics such as mode switches and task completion times.

Chapter 7 describes the idea of *human in the loop* optimization of shared control parameters in the domain of assistive robotics.

Insights: The key insight in this work is that customization of shared-control assistance for specific users is necessary due to the diversity in user needs. When framing the problem of human-autonomy interaction as one of reward optimization, explicit representation of the user's reward function is *not* necessary as long as we can leverage the human in the loop to optimize the arbitration parameters.

Contributions: The chapter presents an iterative verbal optimization protocol initiated by the end-user to optimize the control arbitration parameters. This approach attempts to address the human-AI value misalignment issue by directly giving the human agency over how the autonomous agent should behave.

Results: The results in this study reveal surprising aspects of user preferences; particularly in that the customized levels of assistance did not result in the most optimal task performance as measured by task completion times and number of mode switches. That is, the true reward function that the human optimizes is likely more complex than a simple time-optimal or minimum-effort cost function, indicating the need to investigate the exact specification of the true cost function that is being optimized by the human in a shared control system. The work presented in this chapter is done in collaboration with Siddarth Jain. The details of workload distribution is presented in the chapter.

Chapter 8 of this thesis also makes a software contribution in the form of a new web-based software tool for crowd-sourced Human-Robot Interaction (HRI) experiments.
Insights: Development of novel HRI algorithms is only meaningful and useful if they can be tested rigorously in a larger context. There are many factors that can limit access to end-user evaluation, for example, their ability to travel to the research site, scheduling conflicts, and health issues. At the time of the writing of this thesis, the COVID-19 pandemic also raised significant health concerns for researchers to conduct in-person research studies.
Contributions: This chapter presents *RemoteHRI*, a Javascript-based open-source software framework for crowdsourced human-robot interaction experiments in a web-browser. *RemoteHRI* uses state-of-the-art ReactJS framework to build standard simulated environments for HRI research and provides the researcher with a flexible set of software tools for rapid prototyping and quick deployment of online experiments. The code implementation of this project is primarily accomplished by Finley Lau.

Chapter 9 presents a discussion of future directions for the field.

To summarize, the work presented in this dissertation tackles different challenges that arise in human-autonomy interaction in the context of interface-mediated assistive robot teleoperation and proposes algorithms and protocols to improve the overall human-autonomy interaction experience.

CHAPTER 2

Background and Related Work

This chapter presents an overview of existing literature that has informed the work in this thesis. We will touch upon shared control in the human robot interaction, human behavior modeling, intent recognition, approaches to task allocation, and autonomous policy generation.

2.1. Shared Control in Human Robot Interaction

Traditionally, shared-control in human-machine systems is described as Levels of Autonomy (LoA) [**15, 143**]. LoAs provide a principled taxonomy to classify human-machine interaction by taking into consideration the decision-making responsibilities entrusted to each party, how much information regarding the context is utilized, and also the types of actions enacted by each agent [**52, 53, 54**]. LoAs range from full manual control of a robot by the user (pure teleoperation) on one end to full autonomous control capable of accomplishing a diverse set of tasks in a variety of environments (potentially containing other humans as well) on the other. However, this is still a simplified way of understanding shared-control systems, as the rich and multi-dimensional space of human-machine interaction is reduced to a single dimension [**114**]. Depending on the roles played by the participating agents in a shared-control system, there are multiple fine-grained notions of semi-autonomous systems that exist such as collaborative control, mixed-initiative control, and supervisory control. Going beyond this uni-dimensional approach, Schilling et al. [**140**] advocate for a multidimensional perspective in characterizing shared control systems, particularly with an emphasis on quantifying emergent interaction patterns. The authors present a number of other dimensions such as shared resources, adaptivity, and predictability, to name a few, which are of importance.

Overall, in shared-control systems complementary abilities of humans and autonomous agents are leveraged to jointly accomplish various tasks such that the human-autonomy team is typically *more* capable than either the human or the autonomous agent on their own. That is, the autonomous agent is typically in the control loop of the robot and presumes tight coupling between the human and the agent for successful task accomplishment [**40**]. Abbink et al., in a recent survey paper, defined shared control as one in which "*...humans(s) and robots(s) are interacting congruently in a perception-action cycle to perform a dynamic task, that either the human or the robot could execute individually under ideal circumstances*" [**1**]. However, the presumption that the human partner could execute the task ideally is often violated. For example, in the domain of assistive robotics, the end-user population consists of people with severe motor impairments as a result of spinal cord or brain injuries or neuromotor degenerative diseases who therefore have limits on their ability to control assistive robots. Shared control, as opposed to fully autonomous solutions, is a viable approach to provide a sense of agency to the end user by entrusting some level of robot control with them.

Research in shared control systems span a wide range of topics such as perception [**83, 84, 86**], inference [**3, 129**], user-behavior modeling [**24, 169**], and customization [**32**], to name a few. The work in this thesis primarily concerns inference, personalization, and interaction modeling. The next section provides an overview of various applications of shared control in human-machine systems.

2.2. Applications of Shared Control

Shared control for human-machine systems have applications in various domains. Navigation assistance systems, often found in aviation systems [**160, 161**], semi-autonomous vehicles [**106**], and smart wheelchairs [**45, 78**], are forms of shared control, in which the human is typically responsible for providing the navigation goals and the underlying autonomy is responsible for navigating to the specified goal in a safe and efficient manner [**46**]. Such systems can rely on predefined maps of the world or could rely on SLAM-like techniques [**119**] to build them on-the-fly. Shared control can be used for ensuring safety of the human as well as the machine [**25**]. For example, in semi-autonomous driving, autonomy can leverage its faster computing capabilities to detect and reason about potentially unsafe features in the environment during navigation and provide automatic obstacle avoidance when appropriate thereby keeping the user safe [**150**]. Safety protocols, such as obstacle avoidance are particularly important in situations where pure robot teleoperation is hard due to complex dynamics of the system [**134**], lack of skill, or inherent motor impairments. Within assistive robotics, the type and amount of assistance will vary depending on the use case, the robotic platform, and the control interface used, as well as user preferences. Autonomy can play the role of a teacher and can assist the end user to enhance rehabilitation outcomes and facilitate skill development [**111**].

The standard usage of these assistive machines relies on manual teleoperation typically enacted through a control interface such as a joystick. However, the greater the motor impairment of the user, the more limited are the interfaces available for them to use. These interfaces (for example, sip-and-puffs and switch-based head arrays) are low-dimensional, discrete interfaces that can only operate in subsets of the entire control space. These subsets are referred to as *control modes*. The dimensionality mismatch between the interface and the robot's controllable dimensions necessitates the user to switch between control modes during teleoperation. This is known as *mode switching* and has been shown to increase the cognitive and physical burden and to affect task performance negatively [**128**]. In order to offset the drop in performance due to shifting focus (also known as task switching) from the task at hand, due to switching between different control modes, various mode switch assistance paradigms have been proposed. A simple time-optimal mode switching scheme is shown to improve task performance [**71**]. Machine learning techniques are utilized to learn mappings from robot state to control modes preferred by human users [**85**]. Robot operation in certain control modes can also help the autonomous agent to infer the user's intent more accurately and confidently, especially in scenarios where the inference of user intent is exclusively informed by human's control commands issued via the limited control interfaces [**64**]. Additionally, the autonomous agent can provide assistance in the form of filtering the noise induced in the system due to interface operation. The source of noise could be electro-mechanical wear and tear of the equipment or due to performance degradation as a result of cognitive or physical fatigue.

To that end, in this thesis, we introduce the notion of *inverse legibility*, in which the assistive roles are switched (temporarily) and the human-generated actions *help the autonomous agent* to deliver assistance more effectively [**64**]. Particularly to help with mode switching, we introduce an intent disambiguation metric to characterize the intent disambiguation capabilities of a control dimension/control mode. By having the user operate the robot in the disambiguating control mode, the control commands become more *intent-expressive* and as a result the autonomous agent is able to infer the user's intent more accurately and subsequently step in and provide appropriate assistance.

Another important design consideration is that of customization of assistance to individual user needs. The need for customization in the space of assistive and rehabilitation robotics arises because a one-size-fits-all approach is unlikely to work for all end users. Motor impairments vary significantly between people as the nature of injury or disease is never the same. Therefore, in order to generalize better and to ensure higher user satisfaction and adoption, this thesis presents a protocol for human-in-the-loop customization of control allocation parameters. We utilize a blending-based shared control scheme in which the linear blending factor is a function of the probability of the predicted goal (agnostic to the type of intent inference algorithm used). We assume a parameterized piecewise-linear function and develop an iterative procedure with which the user is able to tune the parameters of the arbitration function to their own satisfaction and preference [**65**].

The next section discusses various approaches to human behavior modeling and intention recognition and their impact on improving the joint performance of the shared-control human-machine system.

2.3. Human Behavior Modeling and Intent Inference for Shared Control

Teamwork in a shared-control system is enhanced when the team members understand each other's intentions and goals. However, there are particular challenges that arise in HRI due to the differences in the mental and physical capabilities of humans and robots [**72**]. Robots can deal with such challenges by maintaining models of human cognition and behavior [**89**] spanning different timescales and levels. The main purpose of behavior modeling is so that it can be used for prediction. Endowing the autonomous agent with this predictive power can enable it to anticipate human needs and actions and respond accordingly.

In [**72**] the authors utilize 'Marr's levels of analysis' [**110**] to categorize models for human behavior into three distinct categories, namely: computational, algorithmic, and implementational. Categorization of human models using Marr's level of analysis clarifies what aspects of human behavior are being modeled. Computational-level techniques are ideal for scenarios that benefit from the knowledge of normative behavior that humans ought to exhibit. These models typically rely on simplistic assumptions of perfectly rational behavior and treat human idiosyncrasies and deviations from the norm as observational noise. Algorithmic level analysis, on the other hand, seeks to delve into the processing constraints that agents have and how they lead to systematic errors thereby providing better insight into *why* agents deviate from normative behavior. However, the algorithmic models typically work well over shorter timescales and therefore are not suited for modeling human behavior that lasts over longer timescales.

Within the computational category, one of the most common methodologies is to adopt simple probabilistic models that attempt to model very low-level short-time horizon behaviors (such as reaching motions performed by humans during a manipulation task). For example, the human could be modeled as a noisily optimal agent that noisily optimizes a goal-dependent cost function [**49**]. Such a model is used by the autonomous agent to infer user's intent in which the predicted goal is the one with the lowest cost given the user's control input. Optimality principles are particularly attractive because of their success in the domain of motor control [**167**] and also because they provide a principled approach to how agents ought to behave. This type of framework, however, requires well-defined cost functions that provide succinct descriptions of the task at hand. Cost functions can be hard-coded under assumptions of rationality, hand-designed by domain experts, or learned from human demonstrations using techniques such as inverse reinforcement learning [**175**] and inverse optimal control [**51**]. Data-driven approaches utilizing conventional machine learning algorithms are also successfully used to recognize human behavior in a wide variety of domains such as social robotics [**112**] and assistive robotics [**63**]. More recently, data-driven approaches based on Koopman operators are used to learn models of joint human-machine systems [**24**]. Koopman operator based approaches scale well to high-dimensional spaces as the computational complexity does not grow with the amount of data.

Recently, advances in deep learning have inspired a variety of approaches to trajectory prediction in many application domains. Deep networks have high expressive power and are usually trained using a supervised learning approach via gradient descent based back-propagation. In general, to accomplish trajectory prediction, these networks take the context information, which consists of environment state and user control signal (when available) as input and predict the state and action trajectory for some amount of time into the future. Network architectures such as variational auto-encoders [**37**], and conditional variational auto-encoders [**80**], are designed so that they learn rich latent space representations that encode the essential information needed for prediction. However, many of these approaches require large amounts of data for proper training and suffer from generalizability issues. Additionally, in order to handle the non-stationarities of the data generation process it is necessary to continually update and train the model which comes with its own infrastructure and computational overhead.

In this thesis, we adopt a diverse set of approaches to model human behavior focussing on simplicity and computational feasibility. In the chapter on human-driven customization (Chapter 7) of shared control parameters [**65**], we rely on ideas from optimal control theory [**97**] to model human behavior. We assume that humans act optimally with respect to an *unknown* (to the autonomy) reward function, but then entrust the human with the responsibility to optimize the control allocation parameters directly. On the other hand, in the work on mode switch assistance for intent disambiguation (Chapter 5), the assumed teleoperation model is one in which the human optimizes for shortest path to goal given the constraints of the control interface. In Chapter 6, we model human teleoperation of the robot as a interface-dependent Markov Decision Process in which the reward function simultaneously optimizes the Manhattan distance to the goal and the number of mode switches executed during teleoperation.

In general, computational approaches are well suited for situations in which one is primarily concerned about what the human is doing without necessarily reasoning about the underlying causes for the behavior. In the algorithmic category, Hidden Markov Models (HMMs) and other Markov-based approaches are common choices for modeling human behavior [**93**]. HMMs are powerful due to their ability to express latent variables and can be used for efficient online inference of hidden states given a set of observations. An extension of HMMs that can be useful for modeling human-robot collaboration is the Partially Observable Markov Decision Process (POMDP) framework [**155**]. In a POMDP setting, the human is typically modeled as noisily optimizing a reward function and picks actions at every timestep according to the optimal policy. For example, POMDP based models are used for assessing the human partner's trust in the autonomous partner [**35**]. POMDPs in which human intent is treated as a latent state can be used the autonomous agent for performing model-based intent inference. By performing online inference on these latent states, the autonomous agent can incorporate human intentions into its own decision making process thereby implicitly taking into account the user's preferences and goals. In general, inference over latent variables in a POMDP framework is performed via Bayesian methods in which a prior distribution over the latent variables is updated via Bayes Theorem upon receiving new evidence [**48**]. The choice of likelihood function typically encodes the human's actions/preferences given the state. A generalization of HMMs, known as Dynamic Bayesian Networks (DBNs), have also been used successfully to model human-robot collaboration. DBNs are attractive due to their ability to handle multivariate, mixed-observability variables and to represent temporal interdependencies [**120**]. DBNs have been successfully utilized to model human beliefs, desires, and learning which can then be used for intent prediction and understanding cognitive phenomena such as concept learning [**156**]. In general, modeling approaches that rely on Bayesian networks offer principled ways to incorporate all the variables that are relevant to describe the phenomena and to reason about them.

Most research aimed at advancing teleoperation systems has focused on creating novel devices that improve upon various aspects of the teleoperation system, such as improved signal decoding in a brain machine interface [**148**], haptic feedback for increased transparency [**2**], or improved fixed mappings from user inputs to control commands in a redundant body machine interface [**127**]. Little work in assistive robotics has distinguished between the intended versus the produced command signal measured through the interface. Work in driver behavior modeling has investigated higher-level (action-level) inference, for example, to classify and predict driver actions [**125**]. Another work considers the uncertainty in human grasp intent to provide appropriate autonomous robotic grasp plans [**18**]. Unlike the work presented in Chapter 4, they assume that the human is physically capable of producing their intended commands and the source of uncertainty is due to detection noise. Previous work has modeled a person's internal beliefs about a dynamic system, and uses a dynamics transfer function in order to provide the assistance that leads to a desired human action or learning outcome [**131, 133**]. In these works, it is assumed that any suboptimal human command is due to a mismatch between their internalized and the true dynamics model, plus there is no control sharing—the autonomous agent alone is issuing commands based on the inferred user intent.

2.4. Intent Disambiguation and Information Gathering in Shared Control

Shared-control systems often require a good estimate of the human's intent—for example, their intended reaching target in a manipulation task or a target goal location in a navigation task [**105**]. Intent can be explicitly communicated by the user [**38**] via various modalities such as laser pointers, click interfaces, and in some cases natural language [**33**]. Intent can also be inferred from the user's control signals and other environmental cues using various algorithms [**87**]. However, when the autonomous agent relies on control signals emanating from the control interfaces to perform intent inference it becomes particularly hard due to the low bandwidth, sparsity, and dimensionality of these interface signals.

The idea behind intent disambiguation is to alter the *context* such that the human is nudged in some manner to assist the autonomous agent in its intent inference capability. The intent disambiguation algorithms presented in Chapters 5 and 6 leverage the underlying synergies that are inherent in human-autonomy cooperation. In the context of human-human cooperative teams, the notion of shared intentionality—one in which all parties involved in a collaborative team share the same intention or goal and have a joint commitment towards it—is crucial to make task execution more seamless and efficient [**163, 164**]. This principle is relevant to successful human-robot interaction as well. From the robot's perspective, the core idea behind our intent disambiguation approach is one of *"Help Me, Help You"*—that is, if the user can help the autonomous agent with intent inference via more intent-expressive actions, then the autonomous agent, in turn, can provide appropriate task assistance more swiftly and accurately. In human-robot interaction, the legibility and predictability of robot motion *to* the human is investigated [**47**] with various techniques to generate legible robot motion proposed [**75**]. Our work instead investigates the idea of *inverse legibility* [**64**] in which the assistance scheme is intended to bring out more legible intent-expressive control commands *from* the human.

Eliciting more legible and information-rich control commands from the user to improve intent estimation is closely related to active learning in which a learning agent seeks to identify data samples that will maximally inform it about the true hypothesis that it attempts to learn. Designing optimal control laws that maximize information gain can be accomplished by having the associated reward structure reflect some measure of information gain [**10**]. Autonomous robots designed for exploration and data acquisition tasks can benefit from exploring more information-rich regions in the environment. If the spatial distribution of information density is known *a priori*, information maximization can be accomplished by maximizing the ergodicity of the robot's trajectory with respect to the underlying information density map [**115, 116**]. Intent disambiguation is closely related to the ideas presented by Brooks et al. [**28**] in which the authors seek to balance information gathering actions with goal-oriented actions within a shared autonomy context. The objective of this work is to identify *autonomous actions* to quickly ascertain the user's goal, whereas in the disambiguation algorithm presented in Chapter 6, we optimize over future states from which *subsequent human actions* will result in information gathering about the latent goal.

Probing algorithms have been designed to elicit information-rich signals in a collaborative workspace setting, for example, an autonomous car interacting with a human driver at a traffic intersection [**138**]. In the context of reinforcement learning, information-theoretic approaches based on mutual information maximization have been used to train agents to hide and share intentions in a cooperative/adversarial multi-agent setting [**152**]. Coaxing humans to generate information rich signals in an interaction setting is also related to how teachers engage in *showing* as opposed to simply *doing* behavior when interacting with a learner [**73, 74**]. In such settings, *showing* behavior is distinct from simply *doing* because the former explicitly optimizes for the student's confidence in what is being taught. The disambiguation algorithms presented in Chapter 5 and Chapter 6 elicit the human to *show* their latent goal *implicitly* by identifying states in which their *doing* behavior is maximally informative and thus amounts to *showing*.

2.5. Approaches to Task Allocation in Shared Control

In a shared-control setting, task responsibility is split between the human and the autonomous agent. Different approaches exist for control allocation that depend on the application domain, user preferences, and robot platform.

The most common methods to share control between the human and autonomous system include (a) hierarchical methods in which the human selects the high-level goal and the autonomous agent generates the low-level control, (b) control partitioning schemes, and (c) blending the user controls and the autonomy commands. In the hierarchical paradigm, control allocation typically occurs at the task level in which high-level task goals are entrusted with the human and the autonomy takes care of the low-level control of the robot [**5, 173**]. For example, in the assistive domain, smart wheelchair users use a click-based interface [**146**] to select a desired destination in the world or a laser pointer [**38**] to point to a desired navigation goal. The autonomous agent can then generate the global as well as local plans utilizing any state-of-the-art motion planners. In the domain of table-top manipulation users use natural language [**22, 137**] to specify the desired grasp or manipulation target which then combined with object recognition [**21**] and motion planning modules can produce the desired robot trajectory [**23**]. In the domain of robotic wheelchair research, the high-level goals typically are navigation goals [**145**], while control partitioning, for example, places the control of speed with the user and heading with the autonomy [**144**].

Blending-based approaches seek to arbitrate between human and autonomy actions at the control signal level (or the policy level) directly [**44, 104, 108**]. A commonly used arbitration scheme is one in which the final control command issued to the robot is a linear combination of human and autonomy control commands. The arbitration itself should be contextual and depends on the autonomous agent's confidence in itself and in the user as well as on the particulars of the user [**49**]. The blending parameter can be fixed or a parameterized function of the context and the autonomy's confidence in its prediction of the user's intent [**68**]. By casting shared-control allocation in a broader theoretical framework, a mathematical model for probabilistic shared control in complex dynamic environments is proposed in [**165**]. In this work, the interactive relationship between the human, autonomy and the environment is modeled as a undirected graphical model. The paper also introduces the notion of an *interaction* function between the operator and the autonomy that captures the "agreeability" between the human and autonomy. For specific forms of the interaction function, linear blending is recovered as a special case of the more general framework. Another approach to arbitration is to treat the interaction function as an all-or-nothing-at-all pass through filter. This approach, commonly referred to as a Maxwell's demon approach in the literature [**20, 60**], blocks the human control signal if it disagrees with the autonomous control signal with respect to some pre-defined metric. The metric is designed such that it quantifies the human control signal's impact on the future safety of the system or efficiency in successfully completing the task. Control blending paradigms often are employed for behaviors like obstacle avoidance [**34**].

Control sharing in the case of robotic arms most commonly involves user-specification of a target (such as an object) [**82**] or pose correction [**16**], and autonomous generation of robot control commands. Approaches that partition the control space may, for example, place the control of end-effector position in the vertical direction with the human and in the $x - y$ plane with the autonomy [**50**]. Control blending is less straightforward—because the user rarely is able to issue a control signal with high enough dimensionality to cover all control dimensions of the robot (e.g. 6D)—but recently is gaining interest. Moreover, there are approaches which study specifically the customization of how this control blending happens [**103**]. The amount of control blending often is determined using an arbitration function that is based, for example, on the autonomy's confidence in its prediction of the user's goal [**48**]. Our work similarly employs an arbitration function to dictate the amount of control sharing.

Optimization techniques are adopted to generate different strategies for control sharing; for example, formulating the problem as a POMDP and inferring a distribution over goals [**89**], using pseudo-navigation functions for collaborative control [**58**] or concatenating energy-optimal motion primitives to create optimal trajectories [**102**]. Although these approaches result in improved task performance (completion time, control effort), the assistance schemes are mixed in terms of user acceptance. In particular, there are instances of assistance resulting in higher user dissatisfaction [**89**], and users preferring to be in control and more cautious [**102**]. In other studies users find the assistance at times to be uncooperative and tolerate a loss of control only for a significant improvement in performance [**172**].

In an attempt to construct more realistic reward functions, others, inspired by design research, incorporate a measure of "discomfort" into the optimization reward function [**69**]. However, the specific form of the reward function is platform dependent and is not generalizable to all assistive devices. Despite an improvement in task performance, none of the above reward function formulations were able to guarantee high user satisfaction (with the exception of domain-specific discomfort [**69**]). The need for higher user satisfaction is crucial for the acceptance of robot autonomy by the end users in the assistive domain. This gap motivates our approach to engage the end user in the optimization procedure.

2.6. Models for Autonomous Policy Generation

Mathematical models are also widely used for learning policies responsible for generating the autonomous agent's control actions. In addition to successful task completion, autonomy would benefit from its actions that are legible, natural, and safe. *Learning from Demonstration* (LfD) [**8**] provides a framework to learn autonomous policies directly from user-provided demonstration data. For example, imitation learning paradigms can be utilized to develop end-to-end systems that can directly map high dimensional states to actions [**17**]. A more generalizable approach is to cast the problem within the framework of *inverse reinforcement learning* in which the goal of the algorithm is to recover the user's reward function [**175**]. Policies that optimize long-term accumulated reward (solution to the forward reinforcement learning (RL) problem) improve the robustness and generalization capabilities of the autonomous agent. Closely related to the RL approach is to utilize a control-theoretic framework to derive optimal policies for a given task. Standard optimal control theory techniques are model-based; that is, they presume the existence of a dynamics model and solve for the optimal policy with respect to a specified cost function [**97**]. State-of-the-art RL techniques, on the other hand, can be model-free and can resort to sampling-based techniques to derive optimal policies [**171**]. In Chapter 7, the autonomous agent utilizes an LfD algorithm to learn task-specific control policies.

Planning-based approaches such as probabilistic road maps [**92**] and rapidly exploring random trees [**99**] are also widely used for generating motion plans (for both robotic manipulators as well as mobile robots). In addition to task accomplishment, in order to enhance user experience robot motion plans typically need to possess various other desired characteristics. Interaction dynamics in a shared-control system can become more seamless if the autonomous agent is able to make its intentions legible to the user. To that end, researchers have attempted to mathematically define legibility and predictability of robot motion [**47**]. Similarly, safety is of paramount importance when robots work in close proximity to humans. Therefore, behaviors such as obstacle avoidance are incorporated into navigation plans [**151**]. For this purpose, the on-board perception system relies on object detection and recognition models to identify objects of interest and obstacles thereby characterizing favorable and unfavorable parts of the state space of the robot during navigation [**118**].

Potential fields based approaches are also effective as they are computationally lightweight and produce more intuitive trajectories that correspond to straight line paths in Euclidean space [**95**]. More recent work have incorporated ideas from differential geometry to treat obstacles as local deformation in the geometry of the workspace and aims to derive motion policies directly in a curved Riemannian workspace [**132**]. The effect of the obstacle is to curve the geodesics (straight line paths) around itself as determined by the local curvature induced by the obstacle [**109**]. Chapters 5 and 6 utilize potential field approaches to generate autonomous control commands which then get combined with human control commands within a blending-based shared control arbitration paradigm.

CHAPTER 3

Formalizing Control Interface Mediated Assistive Robotic Manipulation

Shared control assistive robotics consist of four main actors: **human**, **control interface**, **robot**, and the **autonomous agent**. The work in this thesis investigates different aspects of the interplay between these actors; with a focus on (a) algorithms to improve the autonomous agent's ability to infer human intent via intent disambiguation algorithms, (b) characterization of interface-level stochasticity to provide assistance to handle unintended interface operation, and (c) customization of control arbitration parameters based on user preference.

This chapter provides a detailed overview of the commercially available control interfaces and delves deeper into some unique limitations that arise in manual teleoperation due to constraints imposed by these control interface. We will also discuss how the notion of *control modes* arise as a consequence of the *dimensionality mismatch* between the control interface and the control space of the robot. Additionally, this chapter provides a unifying mathematical framework based on coupled dynamic Bayesian Networks to reason about various phenomena that arise in human-autonomy interaction in shared autonomy settings.

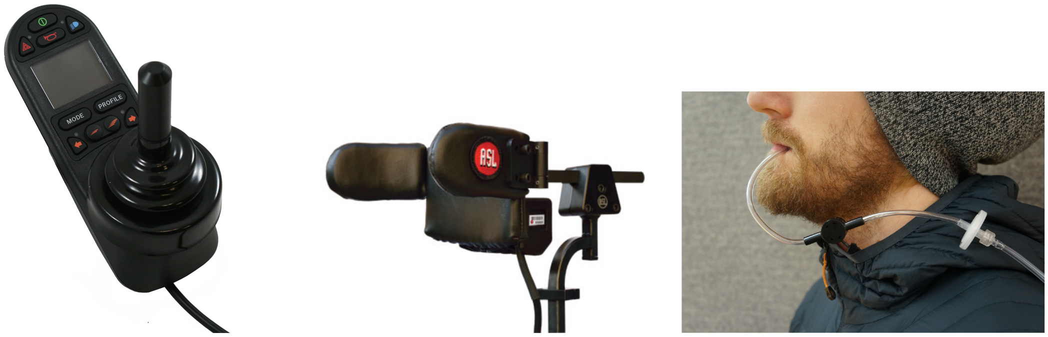


Figure 3.1. Commercially used control interfaces for assistive robot teleoperation. (Left) Joystick - 2D/3D, (Center) Switch-based Head Array - 1D/2D, and (Right) Sip-and-Puff - 1D.

3.1. Formalizing Control Interfaces and Control Modes

We will first describe how to characterize different control interfaces and illustrate how control modes arise. We will then present a framework to formalize robot teleoperation using limited control interfaces.

3.1.1. Overview

In the domain of assistive robotics, the motor impairments that arise due to spinal cord or brain injuries and neuro-degenerative diseases place huge restrictions on the type and kind of control interfaces that are available for the subjects to use. The most popular control interfaces that are commercially available are joysticks, sip-and-puffs, and switch-based head arrays (Figure 3.1). The primary factor that determines the control interface that is available for use by an end user is their level of motor impairment. It is typically the case that *the greater the motor impairment of the user, the more limited are the interfaces available for them to use.*

Let us now look at some of the features that characterize a given control interface. Each control interface can be characterized by its **dimensionality**, which is the *maximum number of controllable dimensions that can be simultaneously activated*, and the set of **interface-level actions** that are available. Depending on the interface, the set of interface-level actions can be either be **continuous** or **discrete**. End users utilize the control interface to realize their *task-level intentions* (for example, *move the robot forward* or *rotate the robot end-effector downwards*) and consequently communicate them to the autonomous agent (and the robot) by executing the corresponding interface-level actions. Typically, the mapping between task-level actions and interface-level actions is predefined and deterministic. Users become proficient at using the interface and memorize the mapping via repeated practice.

From the perspective of the autonomous agent, the incoming interface-level actions that the user generates contain information regarding the user's task-level intentions (or in other words, interface-level actions can be treated as noisy observations of the human's latent intent) and can be used for performing intent inference. Lastly, these interface-level actions received by the autonomous agents are mapped to low-level control commands (joint velocities, end-effector velocities, wheel velocities, *et cetera*) that are applied to the robot controllers which then result in robot motion (or a mode switch).

The commercially available interfaces shown in Figure 3.1 are of lower (or equal) dimensionality when compared to the dimensionality of the assistive robots. An implication of this is that if the dimensionality of the robot control space is **greater** than the dimensionality of the interface, it would necessitate the control space of the robot to be partitioned into smaller subsets for full control of the robot. Each such subset is referred to as a **control mode**. By ensuring that the union of control modes spans the robot's entire control space, users can achieve full control of the robot. For a given interface-robot combination, the total number of control modes will depend on the dimensionality of the robot control space and also of the control interface. This dimensionality mismatch between the interface and the robot's controllable dimensions requires the user to **switch** between control modes during manual teleoperation. This is referred to as **mode switching**. Mode switching increases the physical and cognitive burden as the user needs to switch attention from the task controlling the robot to the task of switching modes [**9, 117**].

Therefore, for high-dimensional robot control using low-dimensional interfaces users are required to execute two types of task-level actions, namely, (a) **control actions** that result in robot motion, and (b) **mode switch actions** that result in transition of the current active control mode to the desired mode. End users execute these task-level actions via pre-specified interface-level actions (such a button presses on a joystick, or a hard puff in a sip-and-puff device). Due to the mechanical and physical constraints, it is usually the case that users are unable to execute both control actions and mode-switch actions simultaneously.

In general, for a given state, the set of available task-level control actions and mode-switch actions depend on the state and also the *mode-switching paradigm*. Three most commonly used mode switching paradigms are: **direct**, **unidirectional cyclic**, **bidirectional cyclic**. Each of the paradigms differ as follows:

- **Direct**: The user can switch to any control mode directly from any other control mode. Typically, joysticks utilize this paradigm, in which each control mode is assigned to a single button that can be pressed independently.
- **Unidirectional Cyclic**: There is a pre-specified ordering of control modes that the user can step through sequentially one at a time without skipping any control mode in the ordered set. Therefore, at any given mode, the user can only transition to the "next" mode in the order. At the end of ordered set a mode switch will typically result in a wraparound to the control mode that occupies the first position in the ordered set.
- **Bidirectional Cyclic**: Similar to unidirectional cyclic, except that the user can switch to the "previous" mode in the order and wraparound happens in both directions as well.

3.1.2. Formalizing Interface-Mediated Robot Teleoperation

Let $\mathcal{Q}$ denote the robot state space and $\mathcal{K}$ denote the set of controllable dimensions of the robot. Let d be the dimensionality of the control interface such that $d < |\mathcal{K}|$. Due to the dimensionality mismatch, the control space is partitioned into a set of control modes denoted as $\mathcal{M}$, such that $\bigcup m = \mathcal{K}$, where each $m \in \mathcal{M}$ is a subset of $\mathcal{K}$. The full state space of the interface-robot system is then denoted as $\mathcal{S} = \mathcal{Q} \times \mathcal{M}$. Note that $|m| \leq d \ \forall \ m \in \mathcal{M}$; that is, all dimensions that constitute a control mode m are accessible through the control interface.

Let $\mathcal{A}$ be the set of all task-level actions available for the interface-robot combination. $\mathcal{A}$ can be decomposed as $\mathcal{A}_q \times \mathcal{A}_m$, where $\mathcal{A}_q$ is the set of **task-level control actions** and $\mathcal{A}_m$ is the set of **mode switch** actions. Task-level control actions bring about state transitions in the robot state space $\mathcal{Q}$ and result in *robot motion.* Mode switch actions result in transitions in $\mathcal{M}$ and determine the *active control mode*; or in other words, the dimension(s) in which robot motion is possible. Let $\mathbf{\Delta}$ denote the mode-switching paradigm, and therefore the subset of actions (primarily the mode switch actions) available in a given state $(q, m) \in \mathcal{Q} \times \mathcal{M}$ is denoted as $\mathcal{A}(q, m, \mathbf{\Delta}) \subset \mathcal{A}$.

Let $\mathbf{\Phi}$ be the set of all **interface-level** actions. We introduce two variables $\phi_i \in \mathbf{\Phi}$ (intended interface-level action) and $\phi_m \in \mathbf{\Phi}$ (measured interface-level action) to distinguish between what interface-level action the user *intended* to execute via the interface and the actual interface-level action that gets measured. Note that this distinction becomes important because ϕ_i captures the cognitive aspect of using a control interface; that is, the mental knowledge of what needs to be done with the interface to achieve a particular task-level action, whereas ϕ_m encodes the physical aspect of operating an interface. ϕ_m, in general, is different from the *intended* interface-level action, ϕ_i, because of various factors such as motor noise, and electro-mechanical wear and tear in the interface, that can corrupt the signal and also the user's ability to execute ϕ_i.

Let us ground this presented formalism in a specific example. Consider the end-effector control of a **3-DoF** robotic arm using a **1D sip-and-puff** device with a **bidirectional cyclic** mode-switching paradigm. The robot state space $\mathcal{Q}$ is $\mathbb{R}^3$. Under the assumption that the robot is operated in the task space, $\mathcal{K}$ corresponds to $\{x, y, z\}$, with $|\mathcal{K}| = 3$. Since the interface is one dimensional $d = 1$; we have $d < |\mathcal{K}|$. The dimensionality mismatch requires $\mathcal{K}$ needs to be partitioned into **three** subsets or control modes, such that each control mode allows for robot control in the single dimension containing in the mode. Therefore, the set of modes $\mathcal{M} = \{\{x\}, \{y\}, \{z\}\}$. Note that $\bigcup m = \mathcal{K}$ and $|m| \leq d \;\; \forall \;\; m \in \mathcal{M}$. The full state space of the robot-interface system is $\mathbb{R}^3 \times \mathcal{M}$. Let us denote x as the **Right-Left**, y as the **Forward-Backward** and z as the **Up-Down** directions. The set of all task-level control actions $\mathcal{A}_c$ consist $\{\textbf{moveRight}, \textbf{moveLeft}, \textbf{moveForward}, \textbf{moveBackward}, \textbf{moveUp}, \textbf{moveDown}\}$. The set of all control-level mode-switch actions $\mathcal{A}_m$ consists of actions denoted as ms_m, which represent a mode-switch (ms) action that would lead to a transition to mode $m \in \mathcal{M}$; that is, $\mathcal{A}_m = \{ms_{\{x\}}, ms_{\{y\}}, ms_{\{z\}}\}$.

In this example, since each mode corresponds to a single dimension of the robot's control space, in a given state (q, m), effectively there are only two task-level control actions available; (a) move robot in the positive direction along the dimension in $m \in \mathcal{M}$, denoted as **movePos**, which corresponds to **moveRight**, **moveForward**, and **moveUp** actions in $\{x\}$, $\{y\}$, and $\{z\}$ respectively, and (b) move robot in the negative direction along the dimension corresponding to m, denoted as **moveNeg**, which corresponds to **moveLeft**, **moveBackward**, and **moveDown** actions in $\{x\}$, $\{y\}$, and $\{z\}$ respectively. Likewise, since the interface is 1D and the mode-switching paradigm is bi-directional cyclic there are only two mode-switch actions available; (a) switch mode to the 'next' mode in sequence (ms_{m+1}), denoted as **modeNext**, which corresponds to $ms_{\{y\}}$, $ms_{\{z\}}$, and $ms_{\{x\}}$ mode switch actions in $\{x\}$, $\{y\}$, and $\{z\}$ respectively and (b) switch mode to the 'previous' mode in the sequence (ms_{m-1}), denoted as **modePrev**, which corresponds to $ms_{\{z\}}$, $ms_{\{x\}}$, and $ms_{\{y\}}$ mode switch actions in $\{x\}$, $\{y\}$, and $\{z\}$ respectively. The set of all interface-level actions available for the 1D sip-and-puff are (a) **Hard Puff** (b) **Hard Sip** (c) **Soft Puff** (d) **Soft Sip**. One possible mapping between interface-level actions and task-level controls actions and mode switch actions can be (a) **Hard Puff** $\longrightarrow$ **modeNext**, (b) **Hard Sip** $\longrightarrow$ **modePrev**, (c) **Soft Puff** $\longrightarrow$ **movePos**, (d) **Soft Sip** $\longrightarrow$ **moveNeg**.

3.2. Human-Autonomy Interaction as Coupled Perception-Action Loops

In the assistive domain, robot teleoperation is typically enacted through a control interface. The actions executed by the human may depend on a variety of factors such as the partial observations of the environment, internal goals and desires, task specifications, constraints of the control interface, and so on and so forth. Upon taking an action the environment state evolves due to the inherent stochastic dynamics and the process repeats in time. In essence, the interaction between the human and the environment can be thought of as a **perception-action loop unfolding in time**. Most typically, humans plan their action in the task space which then gets mapped into interface-level actions and subsequently to low-level control actions. In a similar fashion, the autonomous agent also receives partial information about the environment state (which includes the robot and the human) through various types of sensors such as cameras, joint encoders, and interface signals from control interfaces. The autonomous agent can then perform inference over latent variables such as the user's intended goal and preferences, and take different interventions to control the robot alongside the human to achieve various desired outcomes. This interaction between the autonomous agent and the environment is yet another perception-action loop unfolding in time.

Perception-action cycle is considered to be the fundamental logic of the central nervous system, in which perception and action processes are closely interlinked [**41**]. Perception leads to action, and action leads to perception. In the context of Human-Autonomy Interaction (HAI), as the perception-action loop unfolds, the human and the autonomous agent interact with each other via *explicit and implicit exchange of information.* Both agents continually infer the other's latent states and goals. For example, in situations in which the human goal is not explicitly specified, the autonomous agent has to infer the human's internal goal from available sensor data. Similarly, the human might not be fully aware of the autonomous agent's collaboration strategy and will have to infer its decision-making logic to effectively cooperate and coordinate with it. This joint interaction can be modeled as *coupled perception-action loops.* Within this coupled system, one agent's environment subsumes the other agent. In order to motivate how the notion of perception-action loops can be used to describe HAI in a shared-control setting, let us consider a concrete example of assistive robotic manipulation.

Consider a scenario in which a motor-impaired human and an autonomous agent jointly control an assistive robotic arm to perform table-top manipulation. For simplicity, let us assume that shared control is achieved via a linear control blending paradigm in which the human's low-level control command and the autonomous agent's low-level control command are linearly combined to produce a final control command that is issued to the robot controllers. Let us also assume that the assistive robotic arm is mounted on a wheelchair in which the person is seated. The task is to reach for and grasp certain objects (the true target is only known to the human) on a table. The perception component with respect to the human is as follows. From the seated position, the person is able to see and *perceive* various aspects of the environment such as the positions and orientations of the various table-top objects, the shape and color of the objects, and the pose of the robot's end-effector. The human's line of sight could be obstructed due to occlusion from the robot and other factors and as a result the human only receives partial information regarding the true state of the environment. Based on observations about the world state, and their intentions, the human will execute actions in order to bring about a change in the environment. These actions are relayed to the robot via the control interface and the low-level motor controller that converts interface-level actions into low-level control signals. This constitutes the perception-action loop involving the human and the environment.

Now let us analyze the same scenario from the perspective of the autonomous agent. In the shared control assistive domain, the role of the autonomous agent is defined: it is to assist the human towards their desired goal by sharing robot control responsibilities with the human. Let us assume that the autonomous agent obtains partial observations about the environment state via on-board cameras. Additionally, the control signals that are generated by the human via the control interface also function as sensory inputs to the autonomous agent. Fusing all these sensory inputs, the autonomous agent is able to infer the user's latent intentions and execute appropriate assistive actions alongside the human to bring about change in the environment. This constitutes the perception-action loop involving the autonomous agent and the environment. Since both the human and the autonomous agent interact with the same physical system (the assistive robot and the other objects in the world), they implicitly exchange information regarding each other's control strategies, preferences, as well as intentions.

This thesis contributes a formalized framework that represents our insight that HAI is a coupled perception-action loop, and casts this representation as a Causal Bayesian Network (CBN) (Figure 3.2). The nodes of the network represent the relevant variables

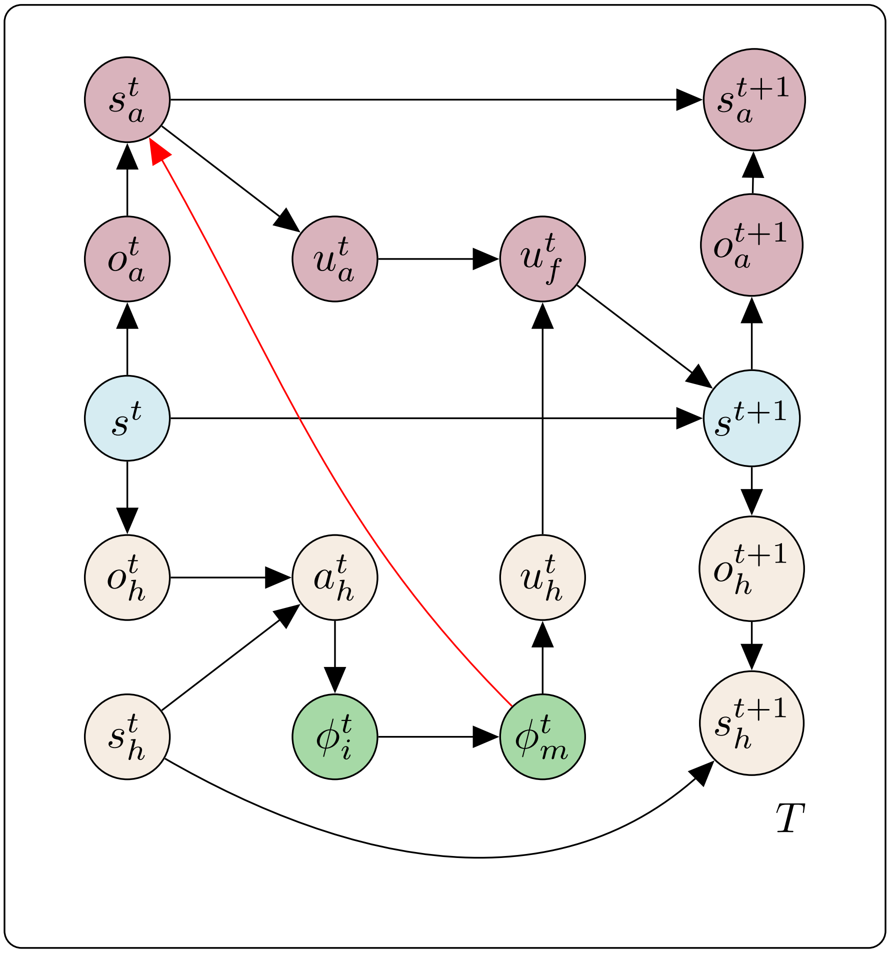


Figure 3.2. Our model of Human-Autonomy Interaction as a coupled perception-action loop that unfolds in time represented as a causal Bayesian network. The nodes represent various relevant variables that interact with each other at discrete time steps. The beige nodes are variables associated with the human whereas the dark pink nodes are associated with the autonomous agent.

(latent and observed) pertaining to both the human and the autonomous agent, and the edges represent the probabilistic influence they have on each other [**124**]. In Figure 3.2, s^t refers to the world state that includes the robot state, active mode state, and other objects and agents in the environment. s_h^t and s_a^t denote the internal state of the human and the autonomous agent respectively. These capture the intentions, preferences and strategies deployed by the human and the autonomous agent during interaction. o_h^t and o_a^t are the noisy observations of the true world state that are accessible to the human and the autonomous agent respectively. a_h^t represents the task-level actions executed by the human. ϕ_i^t and ϕ_m^t are the intended and measured interface-level actions. ϕ_m^t is accessible to the autonomous agent (indicated by the red edge) and allows it to infer the latent internal state of the human. u_h^t denotes the low-level human control command as a result of transforming the ϕ_m^t. u_a^t represents the autonomous agent's control command typically generated via some autonomous policy. u_f^t denotes the arbitrated control command that results in the evolution of the world state from s^t to s^{t+1}. Note that this CBN represents one of the many different ways in which a human and an autonomous agent could interact.

Knowledge about the variables in the model is represented as probabilistic distributions thereby making the model amenable to information-theoretic analysis by which we can quantify the *information dynamics* that unfolds during HAI. Once information flow is quantified, the autonomous agent's behavior can be designed with an aim to *shape* the information flow in the joint system towards desired specifications and outcomes. By characterizing the information dynamics in HAI using the proposed framework, we can:

(1) Develop a framework that will provide a systematic and principled approach to the design of the autonomous agent, in which its actions are interpreted as appropriately timed *interventions* with an aim to modulate the information flow between the human and the autonomous agent, thereby improving overall task performance.

(2) Bring different aspects of HAI, such as intent inference, skill acquisition, control arbitration, transparency, and cooperation under a single umbrella in order to shed light on the more fundamental low-level descriptors and characteristics of human-autonomy teaming.

The subsequent chapters present solutions to address various challenges that arise in HAI, such as intent estimation and disambiguation, interface awareness, and customized optimization of arbitration function in an attempt to facilitate uninterrupted information flow between the agents partaking in this coupled perception-action dance.

CHAPTER 4

Interface-Aware Intent Inference for Customized Handling of Unintended Actions in Assistive Robots

The following chapter presents an assistance system that reasons about a user's *intended* actions during control interface mediated robot teleoperation and provides appropriate interventions for modifying unintended behavior. The key contribution of this chapter is the mathematical formalization of the notion of *interface awareness* in assistive robot teleoperation models. We accomplish this by making a clear distinction between the user's *intended* and *measured* interface-level actions. The results of a human-subject study suggest that the customized assistance paradigms helped to significantly reduce objective task effort, reduce cognitive workload and user frustration and improve overall satisfaction. The work presented in this chapter was done in equal collaboration between Deepak Gopinath and Mahdieh Nejati Javaremi.[1]

4.1. Introduction

In shared-control human-machine systems, the end-users are partially responsible for the control of the robot. In the assistive domain, users exert this agency using different kinds of control interfaces. The choice of interface is typically determined by the extent

[1] The individual contribution breakdown is as follows: 1) Gopinath developed the code for human-subject study and testing environment, performed analysis of simulation data, and was the proctor for the study. 2) Nejati developed the resources and code for the training environments, sip-and-puff teleoperation suite, prepared the surveys on Qualtrics, and performed the statistical analysis of the study data. 3) Gopinath and Nejati equally contributed to model refinement, experiment design, and manuscript writing.

of motor impairment and the residual motor function. These interfaces essentially act as conduits by which information regarding the user's intentions *flow* into the robot and the assistive autonomous agent. In addition to its utility as a control signal, the interface signal often is used in various other capacities by the autonomous agent, such as input to an inference engine. This implies that any deviations—in magnitude, direction or timing—between the signal intended by the human and the signal measured by the autonomous agent can have rippling effects throughout the system pipeline. Deviations between intended and produced human motions have been extensively studied [**39, 56, 88, 168**] and can arise due to cognitive as well as physiological factors [**4**]. For motor-impaired people, inherent physical limitations can increase the likelihood of accidental deviations from intended commands, which can lead to unwanted robot behavior. Therefore, in a shared autonomy system it is important for the autonomous agent to make decisions based on *intended*—as opposed to measured and executed—interface actions to improve the quality and efficacy of the interaction.

Critically in this domain, the mechanism by which the human signal is generated is often ignored and assumed to be a black box. However, operating a control interface and successfully teleoperating a robot requires the user to be able to *physically activate* the interface—whether via button presses, joystick deflections, or screen taps—and have a good grasp of the control mappings between interface-level actions and the task-level actions. Typically, models that reason about human behavior within this context do not represent the activation mechanisms of the interface or the control mappings. The following example will highlight why it is important to reason about these factors if we are to design an assistive functionality for an autonomous agent.

Let us reconsider the interface-robot system presented in Section 3.1.2. In order to successfully teleoperate the robot, the user should know about the predefined control mappings; that is, the user should have a thorough *conceptual* understanding of what interface-level physical action they should execute in order to achieve a particular task-level action. A novice user might not remember the control mappings properly and could choose a wrong interface-level action to accomplish a desired task-level action. It could also be that they have perfect knowledge of what interface-level action needs to be executed, however, due to motor deficits, lack of motor skills, or even fatigue, they are unable to execute it properly. Lastly, there is also the possibility of equipment malfunction that can introduce signal noise, despite the user having perfect knowledge of the control mappings and motor skills to execute the correct interface-level action. All of these issues highlight the need for explicit modeling of the user's physical interaction with the interface and that *interface-awareness* is a key component to the design of successful assistive shared-autonomy algorithms.

The contributions of this chapter are threefold:

- *Modeling the User's Physical Interface Operation*: We mathematically formulate the user's *physical interaction* with the interface during teleoperation and characterize various factors that affect how the intended user inputs get altered through the interface before being measured by the autonomous agent. User-specific models of the user's understanding of the true control mappings and stochastic deviations are built from data collected from individuals to customize the assistive algorithms.

- *Model-Based Inference of Intended Input*: Using the interface-aware teleoperation model and prior knowledge of the user's high-level behavior encoded as a task-level action policy, probabilistic reasoning is used to infer the user's latent *intended* interface-level actions to deduce unintended deviations from expected behavior.
- *Customized Interface-Level Assistance*: We formulate two methods to provide appropriate modifications to the measured human control input in an online fashion. The assistance algorithms are personalized because the user-specific probabilistic models encode the unique characteristics of a particular user's interaction behavior with the robot.

4.2. Interface-Aware Bayesian Intent Inference and Assistance

In this section, we will describe the mathematical model of the user's physical interaction with a control interface during robot teleoperation and the assistive algorithm that uses this model to provide customized assistance.

4.2.1. Modeling the User's Physical Interface Operation

Figure 4.1 depicts the generative probabilistic graphical model of a user's physical interaction with a control interface during robot teleoperation at any time t.

Let s^t represent the world state and a^t represents the action primitives that are defined in the task space that the user intends to execute at time step t. u^t is the low-level control

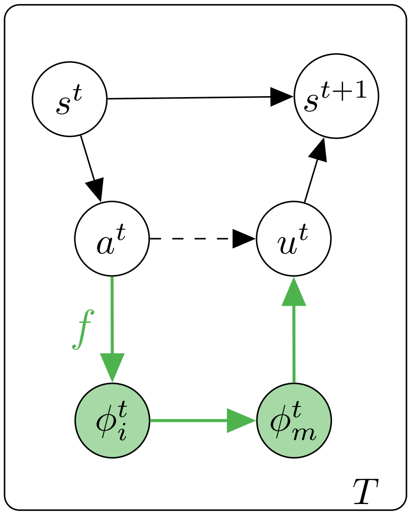


Figure 4.1. Probabilistic graphical model depicting user-robot interaction via a control interface. Teleoperation is typically modeled simply as $a^t \rightarrow u^t$ (dashed edge). We additionally capture physical interaction with the interface (green nodes).

commands (velocities or torques issued to the robot). $\phi_i^t \in \Phi$ is the *intended interface-level physical action* initiated by the user that aims to achieve a^t and is *unobservable* to the autonomous agent.

The set of available interface-level physical actions depends on the physical modality used for activating an interface. The set of interface-level physical actions for commonly used interfaces are presented in Table 4.1. $\phi_m^t \in \Phi$ is the *measured interface-level physical action* produced by the user and is fully *observed* by the autonomous agent. The novel contribution of this model is in (a) the explicit modeling of the interface-dependent physical mechanisms that generate u_h^t and (b) in distinguishing the latent ϕ_i^t from the measured ϕ_m^t. In a noise-free setting, ϕ_i^t and ϕ_m^t are equivalent. However, in practice, ϕ_m^t may deviate from ϕ_i^t due to biases resulting from motor-impairment, stress or equipment malfunction, to name a few. Modeling this distinction is important because the

Sip and Puff	Head Array	2D joystick
Soft Puff, Soft Sip, Hard Puff, Hard Sip	Press Left Button/Right Button/Back Button	Push Joystick at an angle θ^{joy} in the 2D plane, Press Button

Table 4.1. Interface-level actions for commonly used control interfaces.

user wants ϕ_i^t to cause the world state s^t to transition to s^{t+1} whereas in actuality ϕ_m^t is responsible for transition. This discrepancy can lead to transitions into undesirable world states which can increase user frustration.

We model the interactions between these variables using a probabilistic graphical model. Specifically, $p(\phi_i^t|a^t)$ captures the user's *internal model* of the true control mapping (which is static and deterministic and denoted as f). Users acquire this internal model via training. $p(\phi_m^t|\phi_i^t)$ captures the stochastic deviations in the user's interface-level physical actions when using the control interface and can be interpreted as a *user input distortion model.* $p(\phi_i^t|a^t)$ and $p(\phi_m^t|\phi_i^t)$ can be personalized by fitting the distributions to user-specific data. This model can be used by the autonomous agent to improve its understanding of the human partner, which in turn can enhance its decision making capabilities. Note that, we can incorporate partial observability of the world by introducing an observation variable with a corresponding observation function that determines how the observations are generated from the state s^t. The world state could be partially observable due to various reasons; for example line of sight occlusions, lack of transparency of the autonomous agent's actions, limited sensing capabilities, among others.

4.2.2. Bayesian Inference of a^t from ϕ_m^t

In a shared control scenario, the signal emanating from the interface (ϕ_m^t) is accessible to the autonomous agent. In the previous section, we established how the intended task-level action gets altered through the interface due to various noise factors. If the autonomous agent needs to be effective, the assistance provided by it should augment the human's *intended actions* and not the *measured* actions. Therefore, the specific question we seek to address is as follows: Given the measured interface-level physical action issued by the user ϕ_m^t what is the probability distribution over the task-level actions, a^t? More precisely, we are interested in estimating the probability distribution $p\big(a^t|\phi_m^t\big)$. Concretely, using Bayes theorem, we have

$$p\big(a^t|\phi_m^t\big) \propto p\big(\phi_m^t|a^t\big)p\big(a^t\big) \tag{4.1}$$

Marginalizing $p\big(\phi_m^t|a^t\big)$ over ϕ_i^t we have,

$$p\big(\phi_m^t|a^t\big) = \sum_{\phi_i^t \in \Phi} p\big(\phi_m^t, \phi_i^t|a^t\big) \tag{4.2}$$

Due to the conditional independence of a^t and ϕ_m^t Equation 4.2 becomes

$$p\big(\phi_m^t|a^t\big) = \sum_{\phi_i^t \in \Phi} p\big(\phi_m^t|\phi_i^t\big)p\big(\phi_i^t|a^t\big) \tag{4.3}$$

and plugging Equation 4.3 in Equation 4.1 we have,

$$p\big(a^t|\phi_m^t\big) = \eta p\big(a^t\big) \sum_{\phi_i^t \in \Phi} p\big(\phi_m^t|\phi_i^t\big)p\big(\phi_i^t|a^t\big) \tag{4.4}$$

where η is the normalization factor. We also have

$$p\big(a^t\big) = \sum_{s^t \in \mathcal{S}} p\big(a^t|s^t\big)p\big(s^t\big) \tag{4.5}$$

and combining Equation 4.5 with Equation 4.4 we have

$$p\big(a^t|\phi_m^t\big) = \eta \sum_{s^t \in \mathcal{S}} p\big(a^t|s^t\big)p\big(s^t\big) \left[\sum_{\phi_i^t \in \Phi} p\big(\phi_m^t|\phi_i^t\big)p\big(\phi_i^t|a^t\big) \right]. \tag{4.6}$$

4.2.3. Interpreting Conditional Probability Distributions

Each one of the three conditional probability distributions that appear in the right hand side of Equation 4.6 have intuitive interpretations.

$p\big(a^t|s^t\big)$ is the *control policy* the user maintains internally. A novice user's control policy could be a random policy initially, due to lack of familiarity with or understanding of how the system works. With training, the user's policy will gradually stabilize and converge to an optimum—with respect to an internal cost function [**79, 81, 98**]. For example, when performing table-top manipulation with an assistive robotic arm, the user may initially give actions that lead the robotic arm into singularities that need to be corrected with additional actions (less optimal); but as they become more experienced, they will learn to accomplish the task by avoiding such configurations (more optimal).

Additionally, the user might try to execute actions to minimize the task completion time and the number of mode switches performed (additional criteria to optimize).

$p\big(\phi_i^t|a^t\big)$ captures the user's *internal model* of the true control mapping (denoted as f) from task-level action primitives to the intended interface-level physical actions. Users acquire an internal model of this mapping (which is static and deterministic) via training [**126**]. For example, when using a 2-DoF linearly proportional joystick to control a 2-D wheelchair, the control-mapping from the action primitive to interface-level physical actions is intuitive to most people (to move the wheelchair forward, deflect the joystick forward; to move forward faster, deflect the joystick forward *more*). However, using the same interface to control a 2-D lunar lander (from the OpenAI Gym suite) can be less intuitive (pushing the joystick in one direction fires the vertical thruster, while pushing the joystick in the other direction fires the horizontal thruster). However, using a sip-and-puff interface can be less intuitive because the physical actions (regulating air pressure while blowing into and sipping from a tube) do not have a one-to-one correspondence to the task-level action primitives and are less transparent to the user.

Lastly, $p\big(\phi_m^t|\phi_i^t\big)$ captures the stochastic deviations of the *measured* interface-level physical actions from the *intended* interface-level physical actions. This conditional probability distribution can be interpreted as the *user input distortion model.* These deviations can be due to fatigue, delayed or faulty memory retrieval, or features of the interface. These conditional probabilities can be personalized by fitting the distributions to user-specific data.

4.2.4. Customized Handling of Unintended Physical Actions

The motivation for our framework described in Section 4.2.2 is to improve the control of complex robotic machines with limited interfaces used by people with motor-impairments. Equation 4.6 can be used within a shared-control assistive paradigm to infer the human's true task-level intent (concurrently the intended interface-level action) and, if necessary, provide assistance to reduce the cognitive and physical burden of dealing with unintentional deviations during robot teleoperation.

The inference and assistance scheme is outlined in Algorithm 1. Using Equation 4.6, at every time step t we compute the likelihood of $a^t \in \mathcal{A}$ conditioned on the observed ϕ^t_m (line 2). The action primitive corresponding to the maximum of the distribution is inferred to be the intended action $a^t_{inferred}$, and using the true control mapping function f we compute $\phi^t_{inferred}$ (lines 3-4). In Algorithm 2, the autonomy intervenes only if (a) $\phi^t_{inferred}$ is different from ϕ^t_m and (b) the uncertainty of prediction, computed as the entropy H of the distribution, is less than a predefined threshold ϵ. Otherwise, ϕ^t_m will be passed through the control pipeline unaltered. The appealing characteristic of the proposed control-sharing algorithm is that the user is maximally in control, which potentially can improve user satisfaction and acceptance [**26**]. Most notably, our assistance system remains as close to the manual system as possible and does not rely on augmenting the human-robot team with high fidelity sensors to improve prediction accuracy. Furthermore, when the autonomy steps in, it does so only to provide commands closest to the user's true intentions (which they were unable to issue correctly themselves).

We implement and evaluate two assistive shared-control paradigms.

Algorithm 1 Infer Intended Commands

```
function INFER_INTENDED_COMMAND($t, \phi^t_m$)
    compute $p\big(a^t|\phi^t_m\big)$                                  ▷ equation 4.6
    $a^t_{inferred} \leftarrow argmax\big((p\big(a^t|\phi^t_m\big)\big)$
    $\phi^t_{inferred} \leftarrow f\big(a^t_{inferred}\big)$                     ▷ true control mapping
    return $\phi^t_{inferred}$
```

Algorithm 2 Handle Unintended Commands

```
function HANDLE_UNINTENDED_COMMANDS($t, \phi^t_m$)
    $\phi^t_{inferred}$ = INFER_INTENDED_COMMAND($t, \phi^t_m$)
    if $\phi^t_{inferred} \neq \phi^t_m$ then
        if $H\big(p\big(a^t|\phi^t_m\big)\big) < \epsilon$ then                   ▷ uncertainty is low
            if filtered then
                $\phi^t_{modified} = 0$
            else if corrective then
                $\phi^t_{modified} = \phi^t_{inferred}$
        else
            return $\phi^t_m$
    else
        return $\phi^t_m$
    return $\phi^t_{modified}$
```

4.2.4.1. Filtered autonomy. If ϕ^t_m is deemed as unintended with certainty, filter (block) this command, $\phi^t_{modified} = 0$, i.e., no motion or mode switching occurs.

4.2.4.2. Corrective autonomy. If ϕ^t_m is deemed as unintended with certainty, correct this command, $\phi^t_{modified} = \phi^t_{inferred}$, i.e., resulting in application of the inferred intended action.

4.3. Simulation-Based Algorithm Evaluation

In order to gain a deeper insight into how different hyper-parameters—such as noise levels in $p\big(\phi^t_i|a^t\big)$ and $p\big(\phi^t_m|\phi^t_i\big)$—affect the overall performance of our proposed assistance algorithm, we designed a simulation-based experiment. We chose a path-navigation task for this simulation-based evaluation (shown in Figure 4.3) and assumed that an sip-and-puff (SNP) interface was being used for robot teleoperation. The domain of task-level and interface-level actions for an SNP are defined in Section 4.4.1. Task-level action primitives (a^t) and intended interface-level physical actions (ϕ_i^t) were sampled from the generative model shown in Figure 4.1. ϕ_i^t was corrupted according to $p(\phi_m^t|\phi_i^t)$ to generate ϕ_m^t. In our simulations, we assumed the existence and knowledge of a fully deterministic optimal policy (i.e. $p(a^t|s^t)$ is known).

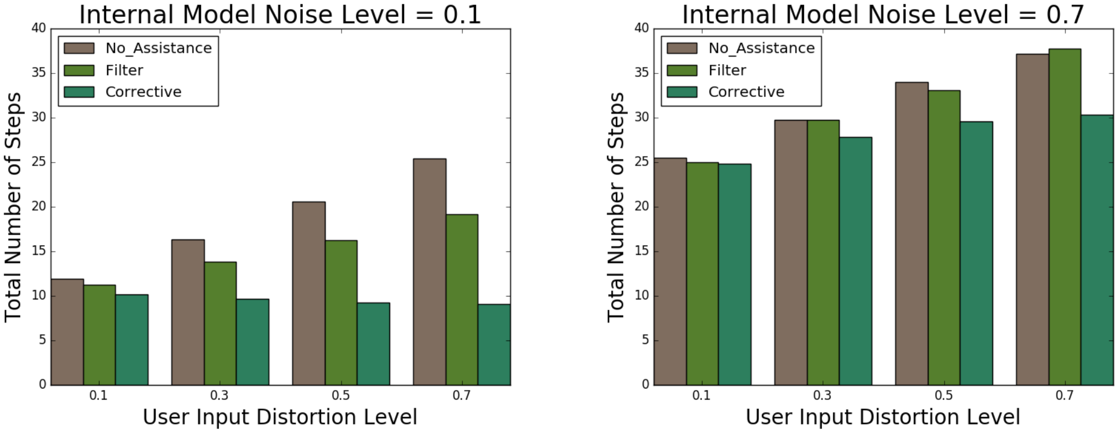
4.3.1. Simulation Setup

The human teleoperation of the robot using the control interface was modeled as a Markov Decision Process where the world state $s^t \in \mathcal{S}$ was defined as a 3-tuple (p_n^t, θ^t, m^t), where $p_n \in [p_0 \ldots p_{N+1}]$ denoted discrete locations that represented the way-points (including the start and the end) of the path, N was the number of turns, $\theta^t \in [0, -\pi/2, \pi/2]$ was the discrete orientations available to the point robot and $m^t \in [m_1, m_2, m_3]$ denoted the currently active mode. The action space $\mathcal{A}$ was identical to the space defined at the end of Section 4.2.1. For a perfect agent, $p(\phi_i^t|a^t)$ and $p(\phi_m^t|\phi_i^t)$ are delta distributions (due to lack of any distortion or noise). In reality due to various factors such as stress, fatigue or hardware issues, the variance of these distributions will be higher. In our simulations, the distributions $p(\phi_i^t|a^t)$ and $p(\phi_m^t|\phi_i^t)$ were modeled as mixture distributions comprised of a delta and a uniform distribution. The weight factor ($w_{uniform}$) used for the mixture distributions was treated as a scalar simulation parameter.[2] In our simulations,

[2] $w_{uniform} = 0.0$ and $w_{uniform} = 1.0$ denote a pure delta and uniform random distribution, respectively.

Parameter	Range of Values
N - Number of Turns	[1,2,3]
Assistance Type	[Filtered, Corrective, No Assistance]
$w_{uniform}$ in $p(\phi_i \mid a)$	[0.1, 0.3, 0.5, 0.7]
$w_{uniform}$ in $p(\phi_m \mid \phi_i)$	[0.1, 0.3, 0.5, 0.7]

Table 4.2. Ranges of different simulation parameters.



(a)

(b)

Figure 4.2. Total number of state transitions for two different noise levels in $p(\phi_i|a)$ - 0.1 (left) 0.7 (right).

the amount of random noise injected into $p(\phi_i^t|a^t)$ and $p(\phi_m^t|\phi_i^t)$ was treated as a scalar simulation parameter. Table 4.2 indicates the ranges of all simulation parameters.

4.3.2. Simulation Results

We evaluate the performance of our assistance algorithm as measured by the total number of state transitions during a trial, under different assistance conditions and investigated the effect of different levels of random noise in $p(\phi_i^t|a^t)$ and $p(\phi_m^t|\phi_i^t)$ on the amount of assistive interventions and accuracy of intent prediction.

$p(\phi_i^t \mid a^t)$ noise	$p(\phi_m^t \mid \phi_i^t)$ noise			
	0.1	0.3	0.5	0.7
0.1	5.7	21.1	41.0	54.9
0.3	0.0	7.4	13.3	24.9
0.5	0.0	0.0	5.6	6.6
0.7	0.0	0.0	0.0	0.0

Table 4.3. Percentage of Assistive Interventions (in %).

$p(\phi_i^t \mid a^t)$ noise	$p(\phi_m^t \mid \phi_i^t)$ noise			
	0.1	0.3	0.5	0.7
0.1	92.9	100.0	100.0	100.0
0.3	72.2	82.4	100.0	100.0
0.5	58.8	51.3	76.5	100.0
0.7	45.4	40.6	36.1	74.4

Table 4.4. Prediction Accuracy of Intended Commands (in %).

Figure 4.2 reveals that a more accurate internal model (where $p(\phi_i^t|a^t)$ has low corruption noise), in general, will help the user to perform better. For a given $p(\phi_i^t|a^t)$, the corrective assistance paradigm has the highest performance, followed by filtered and no-assistance. The difference in performance between the assistance paradigms decreases as the noise in $p(\phi_i^t|a^t)$ increases, illustrating the need for proper training and acquisition of accurate internal models. These insights guide our experimental design explained in detail in the next section. Tables 4.3 and 4.4 indicate the percentage of assistive interventions and prediction accuracy are more sensitive to noise in $p(\phi_i^t|a^t)$ than $p(\phi_m^t|\phi_i^t)$, once again reinforcing the need for proper training for the user so that they acquire a good understanding of the true control mapping from task-level actions to interface-level physical actions.

4.4. Experimental Design

We conducted a human-subject study ($n = 10$) to evaluate our inference algorithm and assistance paradigms in terms of overall task performance and user preference. All participants gave their informed, signed consent to participate in the experiment which was approved by Northwestern University's Institutional Review Board. Each study session consisted of three phases. Phase 1: Training and data collection to model $p\big(\phi_i^t|a^t\big)$, Phase 2: Training and data collection to model $p\big(\phi_m^t|\phi_i^t\big)$, and Phase 3: Assistance evaluation phase in which the subjects controlled a 3D point robot in a simulated navigation environment using the SNP interface under three different assistance conditions.

4.4.1. Experimental Testbed

For the evaluation task, we designed a simulated navigation environment with three control dimensions (Figure 4.3) [**66**]. Participants operated a 1D SNP interface. We chose the 1D SNP interface for reasons of difficulty and accessibility as this often is the only device accessible to those with severe motor impairments. The subjects used the SNP to operate the 3D point robot's motion along two translational dimensions denoted as x, y and a rotational θ dimension, one at a time. The dimensionality mismatch between the control interface and the robot necessitated the control space to be partitioned into smaller subsets called *control modes*. Motion was restricted only along those dimensions that belong to the currently active control mode. The subject used the interface to activate different modes by switching between them. The set $\mathcal{A}$ of task-level action primitives consisted of (a) clockwise mode switch, (b) counter-clockwise mode switch, (c) positive direction motion, and (d) negative direction motion. The set Φ of interface-level physical actions available for a sip-and-puff comprised of (a) hard sip, (b) soft sip, (c) hard puff, and (d) soft puff. The true correspondence between a^t and ϕ_i^t was deterministic (denoted as $f(\cdot)$) and predefined. When using an SNP the likelihood of generating unintended control actions was quite high because of (a) the same input modality was used for both robot motion control as well as mode switching, (b) inherent difficulty in breath regulation, and (c) factors such as fatigue, stress, and saliva gathered in the straw.

4.4.2. Learning Personalized Distributions

We designed two tasks to capture the personalized distributions $p\big(\phi_i^t|a^t\big)$ and $p\big(\phi_m^t|\phi_i^t\big)$ from user data.

4.4.2.1. Personalized Internal Control Mapping Model. Participants were first trained on the true control mapping ($f(\cdot)$) during a standardized training phase. The training consisted of three phases: (a) learning about the action primitives (a^t) for the 3D experimental task-space (Figure 4.3), (b) learning about the interface-level physical actions (ϕ^t) available through the interface, and lastly (c) learning the mapping between ϕ^t and a^t. The training was followed by six blocks of testing trials. During testing, the user was shown a graphical depiction of a^t, and instructed to select the correct ϕ^t. Each block consisted of all the available actions in a randomly balanced order. A training refresher was provided between blocks. To account for the effect of time-induced stress on $p\big(\phi_i^t|a^t\big)$, each trial in each of the three blocks had a time limit of five seconds. Stress can affect memory retrieval, and time constraints have been shown to be the main limitation of working memory since processing and storage compete for limited resources [**14**]. The distribution $p\big(\phi_i^t|a^t\big)$ was modeled using data collected during the testing phase in which

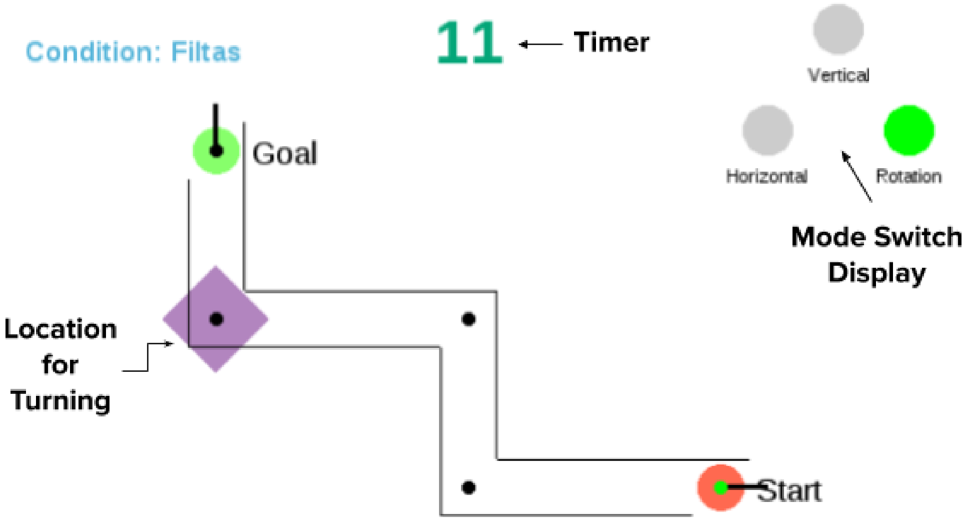


Figure 4.3. An example trial in the human-subject study navigation environment. Feedback regarding the current active mode was displayed on the screen (on the top right corner).

we assumed that the user's internal model is quasi-stationary. The subjects repeated the training and the testing protocol until they met a minimum level of proficiency. This was to ensure that when subject performed the simulation study in Phase 3, they had a good understanding of what interface-level physical actions were needed to generate the required task-level actions (or equivalently low-level control commands) to successfully complete the trial.

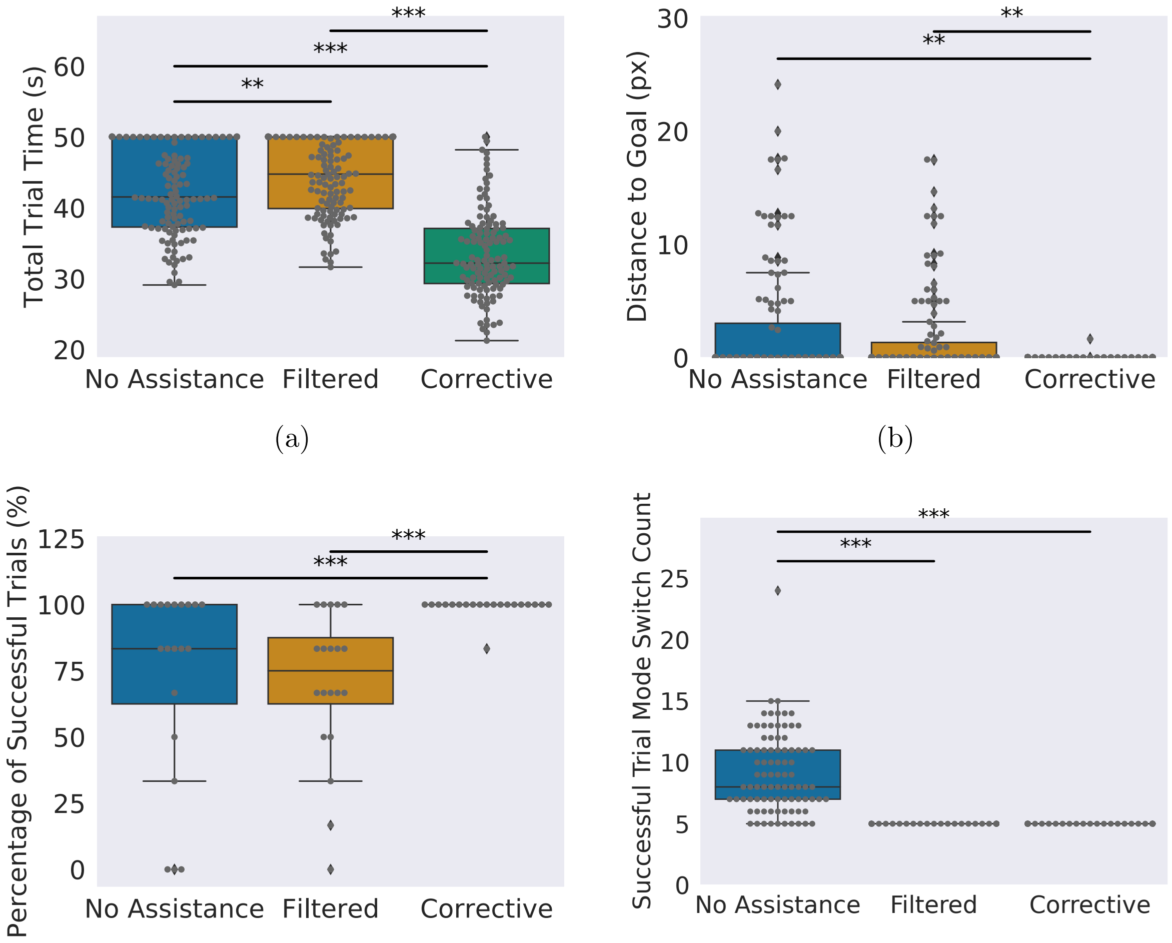
4.4.2.2. Personalized User Input Distortion Model. Participants were trained on the operation of the control interface in order to ensure a good understanding of physical aspects of using the interface. During training, participants were asked to issue different interface-level actions and were provided feedback on how they performed. During testing, the user was shown an interface-level action on the screen (e.g., "Soft Puff") as a prompt and asked to generate the same action through the interface, with no feedback on performance. Similar to the experiment in Section 4.4.2.1, to monitor the effect of time-induced stress on $p\big(\phi_i^t|\phi_m^t\big)$, each trial had a time limit of five seconds. The distribution $p(\phi_m^t|\phi_i^t)$ was modeled using the data collected during this testing phase.

4.4.3. Assistance Evaluation

In the evaluation task, the subject controlled the motion of the 3D point robot along predefined paths from a start pose to a goal pose (Figure 4.3. For each trial the start and end positions were randomized. The initial control mode was selected at random, and restricted to be different than the mode corresponding to the optimal first action in order to normalize the difficulty of starting the trial in different configurations and ensure balance across action types. Users performed the evaluation task under three conditions: (1) *no assistance*, (2) *filtered* assistance, and (3) *corrective* assistance. The subject was required to rotate the point robot to the target orientation at one of the corners (highlighted in violet). Subjects were prompted to complete the task with the least number of mode switches and in a timely manner. A trial was deemed successful if the robot was at the goal pose within the allotted time limit (50 seconds). Subjects performed six blocks (two blocks per assistance condition) of six trials each. In total, we collected 360 trials (120 trials per assistance condition). After each block, the subjects were required to respond to a NASA-TLX questionnaire.

4.5. Results

We analyze group performances using the non-parametric Kruskal-Wallis test and perform the Conover's test post-hoc pairwise comparisons to find the strength of significance. For all figures, $* : p < 0.05$, $** : p < 0.01$, and $*** : p < 0.001$.



(a)

(b)

(c)

(d)

Figure 4.4. Objective task performance metrics grouped by assistance condition. (a) Total trial time with maximum trial time capped at 50s. (b) Distance to the goal at the end of trials. (c) Percentage of successful trials. (d) Total number of mode switches during successful trials. The optimal number of mode switches was 5 for all trials. All metrics improve significantly with the corrective assistance condition.

4.5.1. Objective Task Performance Metrics

To evaluate the effectiveness of our algorithm on overall task performance, we compare (a) the total task completion times, (b) the remaining distance to the desired goal position at the end of each trial, (c) the percentage of successful trials under each assistance condition, and (d) the total number of mode switches for successful trials, across the three conditions (Figure 4.4).

As seen in Figure 4.4a, the total trial time is shortest under *corrective* assistance, increases with the *no assistance*, and is largest under the *filtered* assistance paradigm. One likely reason for the latter observation is that under *filtered* assistance, the repeated issuance of suboptimal mode switch commands is repeatedly blocked, while under *no assistance* consecutive suboptimal mode switches passes through the control pipeline unaffected and eventually results in transition into the desired control mode. For example, in our experimental setup, two counter-clockwise mode switches is equivalent to a single clockwise mode switch, and vice versa. Under the *corrective* scheme, if the autonomous agent is confident in its prediction, it will automatically correct the sub-optimal commands, and therefore no time is lost due to the unintended commands.

Figures 4.4b-4.4c show the remaining distance to goal at the end of the trial and the percentage of trials successfully finished by each subject, respectively. Both of these metrics improve significantly under the *corrective* assistance condition. With *corrective* assistance, almost all users have a 100% success rate, therefore it is expected that the distance to goal will be zero.

The *filtered* and *corrective* assistance paradigms are comparable when looking at the total number of mode switches during successful trials (Figure 4.4d). Both assistance conditions are optimal with respect to the number of mode switches, which is five for all trials. Under *no* assistance, despite successful task completion, the number of executed mode switches is up to three times the optimal number.

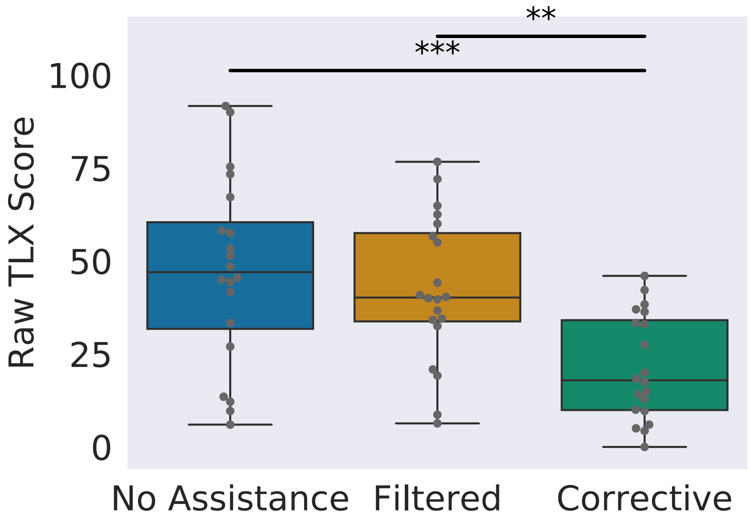


Figure 4.5. An example trial in the human-subject study navigation environment. Feedback regarding the current active mode was displayed on the screen (on the top right corner).

4.5.2. Perceived Workload

We use the raw NASA-TLX as a subjective measure of perceived workload [**70**]. Larger TLX scores indicate higher perceived workloads. During corrective assistance, the autonomous agent offloads some of the subject's cognitive burden by correcting unintended actions—as evident by the significant reduction in the user's perceived workload (Figure 4.5). During filtered assistance, although the autonomous agent gives feedback by way of blocking unintended actions, the user is still responsible for issuing all correct commands.

4.5.3. User Acceptance of Assistive Autonomy

We evaluate user preferences and acceptance of our assistive paradigms using a survey questionnaire (Figure 4.6). The statements are rated on a 7-point Likert scale from

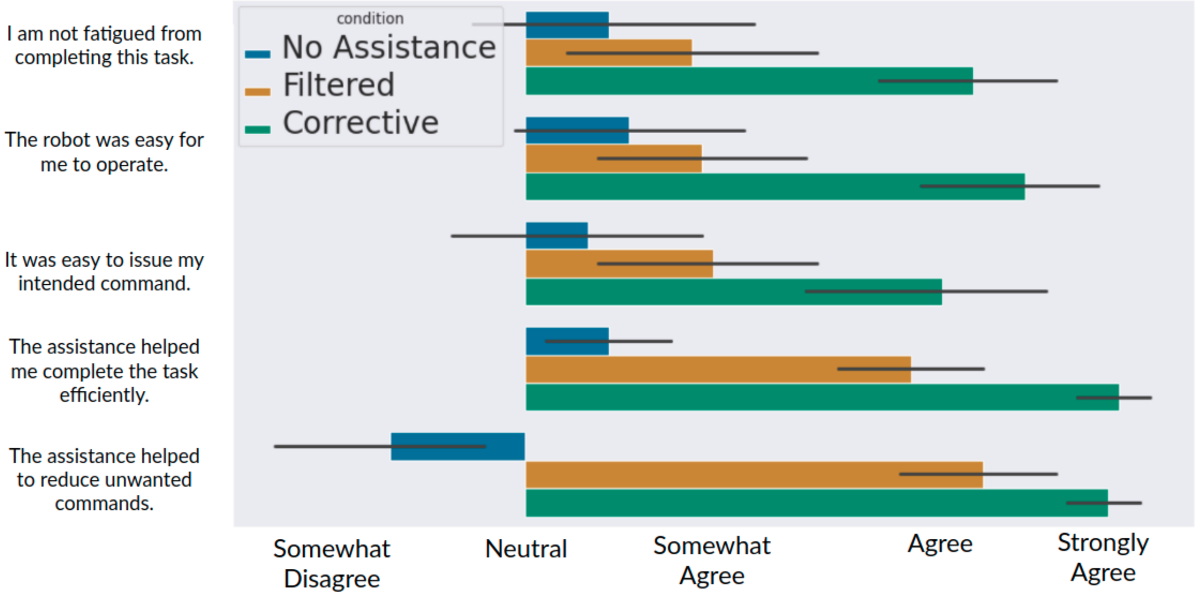


Figure 4.6. Average user response to post-task questionnaire. The bars indicate standard deviation.

strongly disagree (1) to strongly agree (7). Although some of the objective measures of task performance between *filtered* assistance and *no assistance* are comparable, the users rate that the *filtered* assistance helps them complete the task more efficiently and is easier to operate than under *no assistance*. Overall, the participants show a strong preference for the *corrective* assistance. This strong preference for *corrective* assistance is unsurprising because, for the study environment at any given state, there was only a single optimal action to be executed. If the user issued an action different from the optimal action it was immediately corrected and thus progress towards task completion was never impeded.

4.6. Discussion

Our results suggest that each of the assistance paradigms have unique advantages that are crucial for end users of assistive devices. In shared control, the user is partially in control of the robot at all times. This means that robot teleoperation using a control interface is a necessary skill that end-users should possess. Learning teleoperation skills requires appropriate types of feedback and since the *filtered* paradigm blocks all user inputs that *do not* correspond to optimal actions, a user operating the interface under this condition can learn to issue the correct commands progressively. That is, the *filtered* paradigm will play the role a *strict* teacher. The filtered paradigm can also be used within a rehabilitative setting as a teleoperation training paradigm to assist in maximal skill acquisition. By contrast, the *corrective* paradigm might help users who have plateaued in their teleoperation skill, or in cases where efficient and successful task completion is critical. For example, consider a scenario in which a wheelchair user is trying to cross a busy cross-walk with only a few seconds remaining before the cross-walk light turns red. One can imagine that this will be a high stress situation which can potentially result in unintended operation of the control interface. If the autonomous agent in the wheelchair interprets the unintended commands as the true intended commands by the user, these could result in wheelchair motions that might cause accidents. If the wheelchair is endowed with the ability to reason about the intended goal (which is to cross the street safely) then by relying on the *corrective* paradigm, the user will be able to cross the street safely as unintended commands will be modified properly. With corrective assistance a person with a recent motor impairment could begin to operate an assistive device *earlier* in their rehabilitation journey despite the lack of skill in interface operation, thereby accelerating their mobility independence. When used in tandem within an adaptive shared-control framework, the *filtered* and the *corrective* assistance paradigms have the potential to improve the overall quality of assistive device operation while also encouraging skill development and making independent operation of the device more accessible.

One of the limitations of the interface-aware teleoperation model presented in this chapter is that it assumes that the subject's internal model is stationary. For simplicity, we ignore how learning can improve the internal model thereby changing the baseline performance. Likewise, fatigue and hardware related issues (such as accumulation of saliva in the sip-and-puff straw) can add time-varying noise to the input distortion model during the course of interface operation. There are two possible ways to address this problem. We could rely on online learning strategies in which model parameters are tweaked on-the-fly as new data becomes available. Secondly, we could also model hyperparameters related to skill-level and fatigue into the model and jointly infer them as well given that we have sufficiently accurate models for learning and fatigue dynamics.

Another observation that is not reflected in the analysis presented is how subjects alter their control strategies depending on the assistance condition. For example, in the *corrective* assistance condition, some subjects quickly realize that even a 'wrong' command is promptly corrected by the assistance system resulting in the optimal action. This is fine as long as such a strategy is adopted for quick gains, however, in the long run resorting to such strategies to 'exploit' the autonomous agent's assistance can degrade their own ability and understanding of how to operate the robot. To mitigate this, the assistance system could keep track of the statistics of the commands issued by the user and decide to temporarily deactivate the corrective assistance functionality to ensure that the user does not exploit the feature.

4.7. Conclusions

This chapter presented a probabilistic graphical model of control interface mediated robot teleoperation that distinguished between intended versus measured user interface actions. Additionally, this chapter introduced two assistance paradigms that reason about stochastic deviations in user input and provide customized assistance in a shared-control framework. The efficacy of the assistance paradigms were evaluated both in simulation and via a 10-person human subject study. Our results indicated that the assistance conditions were helpful in improving various objective task metrics such as task completion times, number of mode switches performed, final distance to goal, and percentage of successful trials. More importantly, the assistance paradigms also reduced the perceived cognitive workload and user frustration, and improved overall user satisfaction.

CHAPTER 5

An Algorithmic Framework for Intent Disambiguation via Mode Switching: A Heuristic Approach

This chapter introduces the idea of *intent disambiguation* in the context of assistive robotics. In an assistive shared autonomy system, the effectiveness of an autonomous agent's assistance depends on how well it can infer the user's intent. The challenge in these systems, however, is that the *sensory* input that the autonomous agent relies on for intent inference is highly limited due to the inherent physical limitations of the end-user as well as the idiosyncrasies (low-dimensionality, small bandwidth, sparsity) of the control interfaces used for robot teleoperation. With *intent disambiguation* algorithms the autonomous agent seeks to extract *intent expressive* signals from the user which in turn helps the autonomous agent to infer intent more accurately and subsequently offer more effective assistance.

This chapter presents a heuristic approach for intent disambiguation by leveraging the fact that goal-oriented robot control *when restricted to different control modes* can contain different levels of intent expressiveness. To facilitate disambiguation, the *disambiguation metric* developed in this chapter enforces an ordering on the set of control modes according to their ability to disambiguate human intent. The initial simulation-based analysis of the algorithm reveals that the choice of belief update algorithms has a huge impact on the effectiveness of the disambiguation algorithm. Hence as a secondary contribution, this chapter also presents a novel intent inference algorithm inspired by *dynamic field theory* that works in conjunction with the disambiguation scheme. We also present results from an eight person human subject study as well as simulation results that investigate the usefulness of the proposed algorithm.

5.1. Introduction

Assistive and rehabilitation machines—such as robotic arms and smart wheelchairs—have the potential to transform the lives of millions of people with severe motor impairments [**101**]. With rapid technological advancements in the domain of robotics these machines have become more capable and complex, and with this complexity the control of these machines has become a greater challenge.

The standard usage of these assistive machines relies on manual teleoperation typically enacted through a control interface. However, the greater the motor impairment of the user, the more limited are the interfaces available for them to use. These interfaces (for example, sip-and-puffs and switch-based head arrays) are low-dimensional, at times discrete, and can typically only operate in subsets of the entire control space (referred to as *control modes*). The dimensionality mismatch between the interface and the robot's controllable dimensions necessitates the user to switch between control modes during teleoperation and this has been shown to impact the cognitive and physical burden of operation and to affect task performance negatively [**71**].

The introduction of an intelligent autonomous agent to these assistive machines can alleviate some of the above-mentioned issues. More specifically, with *shared* autonomy the task responsibility is shared between the user and the underlying autonomous agent.

However, for the autonomous agent to be effective, it needs to have a good understanding of the user's needs and intentions. That is, *intent inference* is critical to ensure appropriate assistance.

In this work, we consider use-case scenarios in which the autonomous agent's inference of user intent is *exclusively* informed by the human's control commands issued via the control interface. As an example, in the domain of assistive robotic manipulation, these control commands are typically mapped to the end-effector (or joint) velocities and result in robot motion. Motion carries information regarding underlying intent. However, intent inference becomes particularly challenging when the user input is low-dimensional and sparse—as is the case with the more limited interfaces available to those with severe motor impairments—because the robot motion will likely be more discontinuous and jagged, and carries less *direct* information regarding the underlying human intent. While to augment the human-robot system with high-fidelity sensors could enhance the autonomous agent's intent inference capabilities, for reasons of user adoption and cost, within the assistive domain we intentionally design the assistance add-ons to be as invisible and close to the manual system as possible. The need for intent *disambiguation* arises as the autonomous agent needs to reason about all possible goals before issuing appropriate assistance commands.

The key insight in this chapter is that for *goal-oriented robot teleoperation control commands in certain control modes are more intent expressive than others and therefore may help the autonomous agent to improve inference accuracy.* This is the notion of *inverse legibility* in which human-generated actions *help the autonomous agent* to infer the human's intent unambiguously.

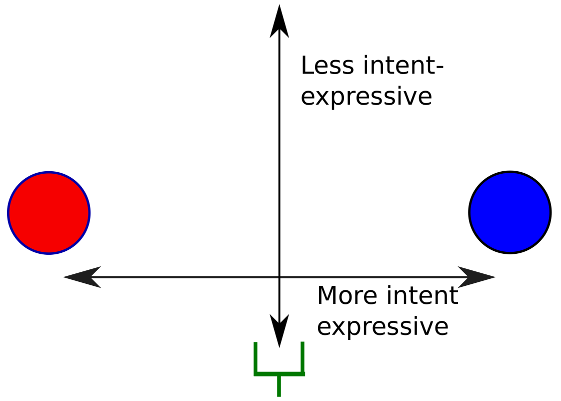


Figure 5.1. Illustration of goal disambiguation along various control dimensions. Any motion of the end-effector (green) along the y-axis will not help the system to disambiguate the two goals (A and B). However, motion along the x-axis provides cues as to which goal.

Consider the hypothetical reaching experiment illustrated in Figure 5.1. Since the spatial locations of the goals are maximally spread along the horizontal axis, any human control command issued along the horizontal dimension conveys a great amount of information about the intended goal to an observer. In other words, motion along the horizontal dimension is more *intent expressive* and will help to draw accurate inference more quickly and confidently. This approach to more seamless human-robot interaction exploits the underlying implicit information exchange between partners that are inherent to task execution with shared intentions. The idea is that, more accurate inference enables the autonomous agent to assist the human more effectively, and thereby improve overall task performance. This is important especially in assistive robotics, wherein the purpose of the autonomous agent is to bridge gaps in control proficiency that result from human impairments and limited control interfaces. The main contributions of this work are as follows:

(1) We develop a control mode selection algorithm which selects the control mode *for* the user, in which the user-initiated motion will help the autonomy to *maximally disambiguate* intent by eliciting more *intent expressive* control commands from the human.

(2) We present results from a pilot study conducted to evaluate the efficacy of the disambiguation algorithm.

(3) We propose a novel field-theoretic approach to intent inference based on ideas from *dynamic field theory* in which the time evolution of the probability distribution over goals is specified as a continuous-time constrained dynamical system that obeys the principle of maximum entropy in the absence of user control commands.

5.2. Mathematical Formalism for Intent Disambiguation

This section develops a heuristic disambiguation metric to induce an ordering on the set of control modes according to their ability to disambiguate human intent and frames intent disambiguation as a problem of determining the control mode that is able to maximally disambiguate between goals.

5.2.1. Notation

Let $\mathcal{G}$ be the set of all candidate goals with $n_g = |\mathcal{G}|$ and g^i refers to the i^{th} goal with $i \in [1, 2, \ldots, n_g]$. A *goal* in this context represents the human's underlying intent. Specifically, in assistive robotic manipulation, as the robotic arm first must reach toward and grasp discrete objects in the environment, intent inference is the estimation of the belief over all possible discrete goals (objects) in the environment. At any time t, the autonomous agent maintains a probability distribution over goals denoted by $\boldsymbol{p}(t)$ such that $\boldsymbol{p}(t) = [p^1(t), p^2(t), \ldots, p^{n_g}(t)]^T$ where $p^i(t)$ denotes the probability associated with goal g^i. The probability $p^i(t)$ represents the robot's belief that goal g^i is the human's intended goal.

Let $\mathcal{K}$ be the set of all controllable dimensions of the robot and k^i represents the i^{th} control dimension where $i \in [1, 2, \ldots, n_k]$ with $n_k = |\mathcal{K}|$. The limitations of the control interface necessitate $\mathcal{K}$ to be partitioned into control modes. Let $\mathcal{M}$ be the set of all control modes with $n_m = |\mathcal{M}|$. Additionally, let m^i be the i^{th} control mode where $i \in [1, 2, \ldots, n_m]$. Each control mode m^i is a subset of $\mathcal{K}$ such that $\bigcup\limits_{i=1}^{n_m} m^i = \mathcal{K}$.[1]

In this work, we assume a kinematic model for the robot and the kinematic state (the robot's end-effector pose) at any time t is denoted as $\boldsymbol{x}_r(t) \in \mathbb{R}^3 \times \mathbb{S}^3$ and consists of a position and orientation component, where $\mathbb{S}^3$ is the space of all unit quaternions. The pose for goal $g \in \mathcal{G}$ is denoted as $\boldsymbol{x}_g \in \mathbb{R}^3 \times \mathbb{S}^3$. The low-level control command issued by the human via the control interface is denoted as $\boldsymbol{u}_h$ and is mapped to the Cartesian velocity of the robot's end-effector.[2] For a 6-DoF robotic arm, $\boldsymbol{u}_h \in \mathbb{R}^6$. The autonomous control policy generates an autonomy control command which is denoted as $\boldsymbol{u}_a \in \mathbb{R}^6$. The control command issued to the robot, which is a synthesis of $\boldsymbol{u}_h$ and $\boldsymbol{u}_a$ is denoted as $\boldsymbol{u}_f \in \mathbb{R}^6$. The control command that corresponds to a unit velocity vector along control dimension $k \in \mathcal{K}$ is denoted as $\boldsymbol{e}^k$.

[1] Note that a dimension $k \in \mathcal{K}$ can be an element of multiple control modes.

[2] Note that, in the formalism presented in this chapter, we ignore interface-level actions and reason about intent directly using the low-level control commands. Interface-aware intent disambiguation is presented in Chapter 6.

5.2.2. Disambiguation Metric

The disambiguation metric developed in this section is a *heuristic* measure that characterizes the intent disambiguation capabilities of robot motion along particular control dimensions within particular control modes. Specifically, we define disambiguation metric $D_k \in \mathbb{R} \; \forall \; k \in \mathcal{K}$. We further explicitly denote disambiguation measures for both positive and negative motions along k as D_k^+ and D_k^- respectively. We also define a disambiguation metric $D_m \in \mathbb{R}$ for each control mode $m \in \mathcal{M}$.

By virtue of design, the disambiguation metric D_m is a measure of how useful user initiated robot motion in control mode m would be to the autonomous agent's ability to perform intent inference. In this work the computation of D_k depends on four features (denoted as Γ_k, Ω_k, Λ_k, and Υ_k), that capture different aspects of the *shape* of a projection of the probability distribution over intent. These projections and computations are described in detail in Section 5.2.3 and Section 5.2.4, and as pseudocode in Algorithm 3.

5.2.3. Forward Projection of $\boldsymbol{p}(t)$

The first step in the computation of D_k is a model-based forward projection of the probability distribution $\boldsymbol{p}(t)$ from the current time t_a to times t_b and t_c (Algorithm 3, line 4) where $t_a < t_b < t_c$.[3] We consider two future times in order to compute short-term (t_b) and long-term (t_c) evolutions of the probability distribution. The application of unit velocity $\boldsymbol{e}^k$ results in probability distributions $\boldsymbol{p}_k^+(t_b)$ and $\boldsymbol{p}_k^+(t_c)$, and $-\boldsymbol{e}^k$ results in $\boldsymbol{p}_k^-(t_b)$ and

[3] *UpdateIntent()* in Line 4 is implemented using Equation 5.7 discussed in detail in Section 5.4.2. *SimulateKinematics()* assumes that the end-effector kinematics is same as that of a point-like robot. All parameters which affect the computation of $\boldsymbol{p}(t)$ are denoted as $\boldsymbol{\Theta}$.

Algorithm 3 Intent Disambiguation

```
Require: $\boldsymbol{p}(t_a), \boldsymbol{x}_r(t_a), \Delta t, t_a < t_b < t_c, \boldsymbol{\Theta}$
 1: for $k = 0 \dots n_k$ do
 2:     Initialize $D_k = 0$, $t = t_a$, $\boldsymbol{u}_h = \boldsymbol{e}^k$
 3:     while $t \leq t_c$ do
 4:         $\boldsymbol{p}_k(t + \Delta t) \leftarrow \text{UpdateIntent}(\boldsymbol{p}_k(t), \boldsymbol{u}_h; \boldsymbol{\Theta})$
 5:         $\boldsymbol{x}_r(t + \Delta t) \leftarrow \text{SimulateKinematics}(\boldsymbol{x}_r(t), \boldsymbol{u}_h)$
 6:         if $t = t_b$ then
 7:             Compute $\Gamma_k, \Omega_k, \Lambda_k$
 8:         if $t = t_c$ then
 9:             Compute $\Upsilon_k$
10:         $t \leftarrow t + \Delta t$
11:     Compute $D_k$
```

$\boldsymbol{p}_k^-(t_c)$, where the subscript k captures the fact that the projection is the result of the application of a control command only along control dimension k.

5.2.4. Features of D_k

To compute the disambiguation metric for a control dimension, we design four features that encode different aspects of the *shape* of the probability distribution as it evolves under motion in a specific control dimension k. For each control dimension k, each of the four features is computed for projections along both positive and negative directions independently. The four features are computed in lines 7 and 9 in Algorithm 3.

1) *Maximum:* The maximum of the projected probability distribution $\boldsymbol{p}_k(t_b)$ is a good measure of the robot's *overall certainty* in accurately predicting human intent. The maximum of the distribution is given by

$$\Gamma_k = \max_{1 \leq i \leq n_g} p_k^i(t_b) \tag{5.1}$$

(i.e., the statistical mode of this discrete distribution). A higher value implies that the robot has a greater confidence in its prediction of the human's intended goal.

2) *Pairwise separation:* More generally, disambiguation accuracy benefits from a larger separation, Λ_k, between goal probabilities. The quantity Λ_k is computed as the *sum of the pairwise distances* between the n_g probabilities.

$$\Lambda_k = \sum_{i=1}^{n_g} \sum_{j=i}^{n_g} |p_k^i(t_b) - p_k^j(t_b)| \tag{5.2}$$

Λ_k is particularly helpful if the difference between the largest probabilities fails to disambiguate.

3) *Difference between maxima:* Disambiguation accuracy benefits from greater differences between the first and second most probable goals. This difference is denoted as

$$\Omega_k = \max(\boldsymbol{p}_k(t_b)) - \max(\boldsymbol{p}_k(t_b) \setminus \max(\boldsymbol{p}_k(t_b))) \tag{5.3}$$

and Ω_k becomes particularly important when the distribution has multiple modes and a single measure of maximal certainty (Γ_k) alone is not sufficient for successful disambiguation.

4) *Gradients:* Γ_k, Ω_k, and Λ_k are local measures that encode shape characteristics of the short-term temporal projections of the probability distribution over goals. However, the quantity $\boldsymbol{p}_k(t)$ can undergo significant changes upon long-term continuation of motion along control dimension k. The spatial gradient of $\boldsymbol{p}_k(t)$ encodes this propensity for change

and is approximated by

$$\frac{\partial \boldsymbol{p}_k(t)}{\partial x_k} \simeq \frac{\boldsymbol{p}_k(t_c) - \boldsymbol{p}_k(t_b)}{x_k(t_c) - x_k(t_b)}$$

where x_k is the component of robot's projected displacement along control dimension k. The greater the difference between individual spatial gradients, the greater will the probabilities deviate from each other, thereby helping in disambiguation. In order to quantify the "spread" of gradients we define Υ_k as

$$\Upsilon_k = \sum_{i=1}^{n_g} \sum_{j=i}^{n_g} \Big| \frac{\partial p_k^i(t)}{\partial x_k} - \frac{\partial p_k^j(t)}{\partial x_k} \Big| \tag{5.4}$$

where $|\cdot|$ denotes the absolute value.

5) *Computation of D_k and D_m:* The individual features Γ_k, Ω_k, Λ_k, and Υ_k are combined to compute D_k in such a way that, by design, higher values of D_k imply greater disambiguation capability for the control dimension k. More specifically,

$$D_k = \underbrace{w \cdot (\Gamma_k \cdot \Lambda_k \cdot \Omega_k)}_{\text{short-term}} + \underbrace{(1 - w) \cdot \Upsilon_k}_{\text{long-term}} \tag{5.5}$$

where w is a task-specific weight that balances the contributions of the short-term and long-term components. In our implementation, w is empirically set to 0.5. Equation 5.5 is computed twice, once in each of the positive ($\boldsymbol{e}^k$) and negative directions ($-\boldsymbol{e}^k$) along k, and the results (D_k^+ and D_k^-) are then summed to compute D_k.

The computation of D_k is performed for each control dimension $k \in \mathcal{K}$. The disambiguation metric D_m for control mode m then is calculated as

$$D_m = \sum_{k \in m} D_k$$

and the control mode with highest disambiguation capability m^* is given by $m^* = \text{argmax}_m D_m$ and $k^* = \text{argmax}_k D_k$ gives the control dimension with highest disambiguation capability k^*. Disambiguation mode m^* is the control mode the autonomous agent chooses *for* the human to better estimate their intent.

5.3. Simulation-Based Analysis of the Impact of Choice of Intent Inference on Disambiguation

This section presents some of the preliminary simulation-based analysis results in which we qualitatively investigate whether the proposed disambiguation algorithm selects control modes that are useful for intent disambiguation for a given goal configuration. Additionally, we also evaluate the impact of certain simplification assumptions in our algorithm and how different choices of intent inference mechanisms affect the accuracy (in terms of picking the most useful mode for disambiguation) of the algorithm.

5.3.1. Heuristic-Based Confidence Functions

Confidence functions based on distance measures and alignment between the human and the autonomous agent's control commands provide interpretable, simple, and computationally lightweight methods to maintain estimates over the user's intended goal. In order to investigate how different choices of intent inference methods affect the accuracy of the disambiguation algorithm, we start with simple choices of confidence functions. Simulation based analysis is performed using these confidence functions which reveals areas of improvement. Informed by the simulation results, a more robust and accurate inference method based on dynamic field theory (used in the human-subject study experiment) is proposed in the next section.

We implement two confidence functions for the simulation-based analysis reported in Section 5.3.2 and 5.3.3. A simple proximity-based confidence function used extensively in the literature [**48, 49**] is

$$\textbf{C1}\text{: } c(\boldsymbol{x}_r, \boldsymbol{x}_g) = \max(0, 1 - \frac{\|\boldsymbol{x}_r - \boldsymbol{x}_g\|}{R})$$

where $\boldsymbol{x}_r$ is the current position of the robot, $\boldsymbol{x}_g$ is the location of goal g, R is the radius of a sphere beyond which the confidence is always 0 and $\|\cdot\|$ is the Euclidean norm. We refer to this confidence function as **C1**.

A weakness of **C1** is that it only considers current position and ignores any cues regarding human intent present in the control command itself. A confidence function that instead incorporates the human's control command will contain more information. One such function aims to capture the "directedness" of the human control command towards a goal position. We refer to this as **C2** and is given by,

$$\textbf{C2}\text{: } c(\boldsymbol{x}_r, \boldsymbol{x}_g, \boldsymbol{u}_h) = \boldsymbol{u}_h \cdot (\boldsymbol{x}_g - \boldsymbol{x}_r)$$

where $\boldsymbol{u}_h$ is the human control command.

5.3.2. Impact of Choice of Confidence Function

In order to qualitatively assess the soundness of our algorithm, we perform simulations in which k^* is computed at 2000 uniformly sampled points in the workspace of a simulated version of the robotic arm described in Section 5.5. The workspace is approximated as a

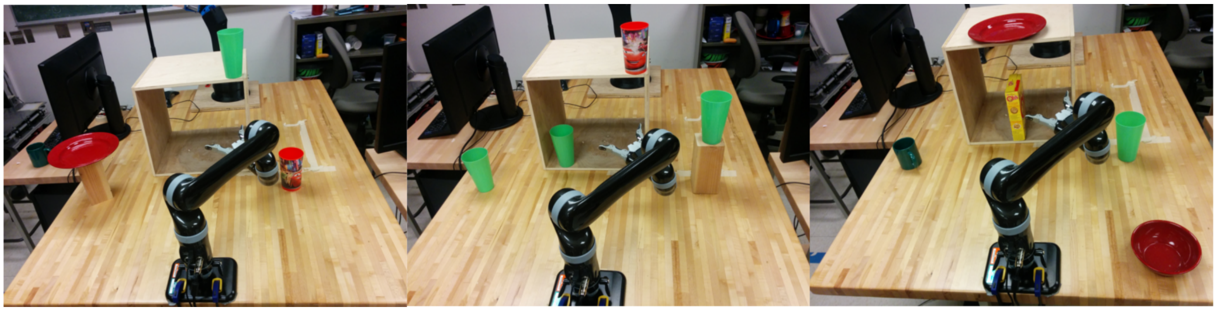


Figure 5.2. Tasks for simulation analysis. From Left to Right: Easy, Hard, Hardest.

$1.2\times0.6\times0.7m^3$ volume in front of the robot. Three different goal configurations were used for simulations. Since the target orientations are the same for all goals, disambiguation only happens in the translational dimensions and therefore is reduced to a 3D problem. Confidence functions **C1** ($R = 0.3m$) and **C2** are evaluated using a goal configuration shown in Figure 5.2 (middle column).

Figure 5.3 shows the results of the simulation. For the goal configuration used in the simulation, the goal positions are spread out maximally along the x and z axes. Intuitively, the system will be able to infer the human's intent quicker if the human control command is either along the x or the z axis. Table 5.1 further reports the number of times the algorithm picked each of the three control dimensions, for each confidence function.

These results shed light on the efficacy of a confidence function in properly capturing human intent. The choice of confidence functions can greatly affect the computation of k^* and m^*, and so the effectiveness of our disambiguation algorithm is intimately linked to the inference methodology of different choices of confidence function.

Under confidence function **C1** the information is equally spread throughout all control dimensions (Table 5.1), because **C1** contains less information with respect to the user's

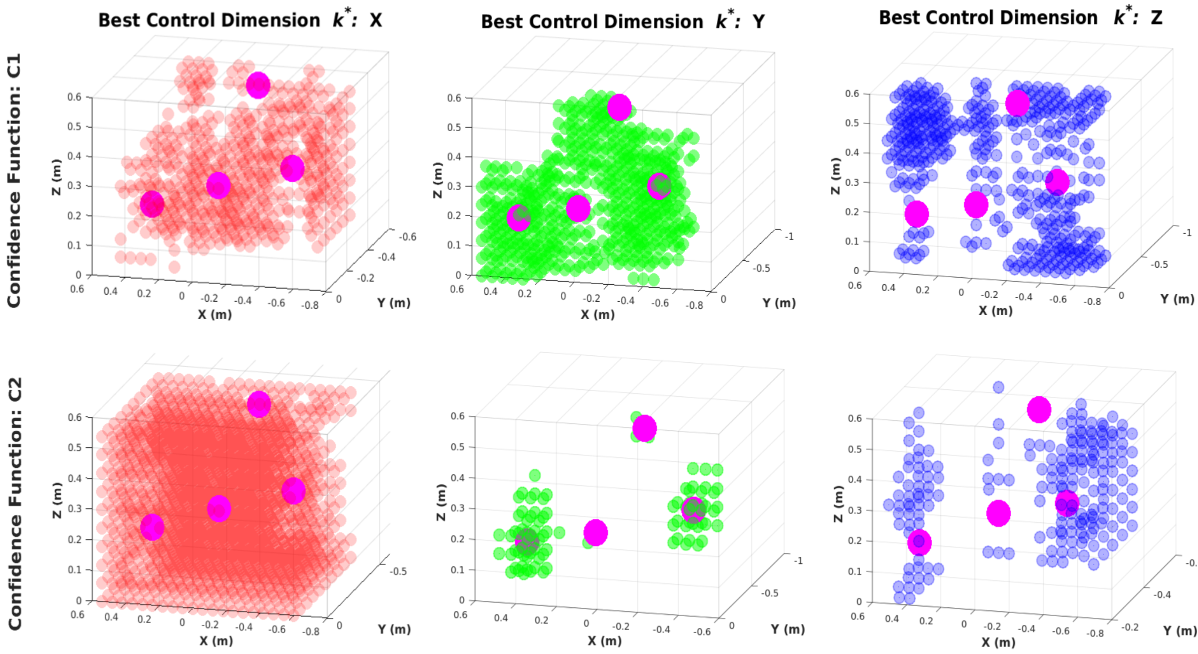


Figure 5.3. Control dimensions best able to disambiguate intent. Using confidence functions **C1** (top row) and **C2** (bottom row). Left column: k^* is X. Middle Column: k^* is Y. Right Column: k^* is Z. Magenta spheres indicate the goal locations (intent).

Best control dimension distribution				
Confidence Function	**X**	**Y**	**Z**	**NULL**
C1	579	615	446	360
C2	1711	93	196	0

Table 5.1. Best control dimension distribution for two different confidence functions.

selected *motion* and therefore also their intended goal. Furthermore, **C1** has “null” spaces where all confidences are identically equal to zero—and therefore neither disambiguation nor intent inference is possible. By contrast, using **C2**, x is identified as the preferred dimension in 1711 out of 2000 samples, and z in 196 of the remaining 289 samples, which indicates that the confidence function along with the disambiguation algorithm is able to

n_g	3	4	5
Accuracy%	89.24	87.09	86.11

Table 5.2. Disambiguation accuracy for off-axis motions

select the optimal disambiguating dimensions over 95% of the time. The algorithm picked y only when the robot is directly in front of a goal.

5.3.3. Robustness of Computing D_m from D_k

In our algorithm, the computation of D_m (Equation 5.2.4) only considers motion projected along orthogonal dimensions: the axes of each dimension k contained in mode m. However, the user can generate a control command in any arbitrary direction within the control mode; that is, the robot can be moved along any direction spanned by the control dimensions in m. In order to assess the robustness of computing D_m only using the orthogonal dimensions, we perform simulations in which m^* is computed for 500 uniformly spaced locations in the robot workspace. At each of those points, 100 random control commands feasible in m^* are generated and applied to perturb the robot. Finally, at each of these perturbed positions the best control mode is once again computed.

If the best mode in the perturbed position is indeed mode m^*, then the simplification does not adversely affect the identification of the disambiguating mode. Table 5.2 summarizes the number of times a match occurs for different configurations of the workspace (with $n_g = 3, 4,$ and 5). While the simplification does hold for 85-90% of off-axis motions, we also observe a trend where performance drops as the number of goals increases. Intuitively this makes sense because disambiguation between goals will become harder with a larger number of goals in the scene and it becomes necessary to consider potential robot motions that are not just along the orthogonal dimensions.

5.3.4. Discussion

Our simulation results indicate a strong correlation between the intent inference power of a given confidence function and the disambiguation power of our algorithm. It is unsurprising that confidence functions which are information-poor approximations of human intent also perform less robustly when disambiguating between those approximations. Moreover, the algorithm could be used to pre-compute the disambiguating modes offline. Informed by the findings of the simulation results presented, in the next section, we introduce a novel intent inference mechanism that draws inspiration from dynamic field theory. The proposed inference mechanism aims to overcome the limitations of heuristic confidence functions by incorporating memory via recurrent interactions terms as well as robustness to noise in the external input.

5.4. Intent Inference

Since the disambiguation power of our algorithm is closely linked to the fidelity of the underlying intent inference mechanism, in this section, we propose a novel intent inference scheme inspired by *dynamic field theory*. By having the autonomous agent maintain a probability distribution over goals, we implicitly model the human as a Partially Observable Markov Decision Process (POMDP) in which all the uncertainty in the user's state is concentrated in the user's intended goal. By maintaining and updating a probability distribution over goals the autonomous agent can reason about the human's latent state (internal goal) during trial execution. Inference over goal states typically is done using a recursive Bayesian belief update which determines how the distribution evolves over time. Here, we introduce a novel approach to compute the time evolution of a probability distribution over goals as a *constrained dynamical system* which serves as an alternative to the recursive Bayesian update scheme.

5.4.1. Dynamic Field Theory

In Dynamic Field Theory (DFT) [**142**], variables of interest are treated as dynamical state variables. To represent the information about these variables requires two dimensions: one which specifies the value the variables can attain and the other which encodes the *activation level* or the amount of information about a particular value. These *activation fields* (also known as dynamic neural fields) are analogous to probability distributions defined over a random variable.

Following Amari's formulation [**6**] the dynamics of an activation field $\phi(x, t)$ are given by

$$\tau \frac{\partial \phi(x,t)}{\partial t} = \int dx' b(x - x') \sigma(\phi(x', t)) - \phi(x,t) + h + S(x,t) \tag{5.6}$$

where x denotes the variable of interest, t is time, τ is the time-scale parameter, $b(x - x')$ is the interaction kernel, $\sigma(\phi)$ is a sigmoidal nonlinear threshold function, h is the constant resting level, and $S(x, t)$ is the external input. The interaction kernel mediates how activations at all other field sites x' drive the activation level at x. Two types of interactions are possible: excitatory (when interaction is positive) which drives up the activation, and inhibitory (when the interaction is negative) which drives the activation down.

Historically, dynamic neural fields were conceived to explain cortical population neuronal dynamics based on the hypothesis that the excitatory and inhibitory neural interactions between local neuronal pools form the basis of cortical information processing.

These activation fields possess unique characteristics that make them ideal candidates for modeling the time evolution of $\boldsymbol{p}(t)$. First, a peak in the activation field can be *sustained* even in the absence of external input due to the recurrent interaction terms. Second, information from the past can be *preserved* over much larger time scales quite easily by tuning the time-scale parameter thereby endowing the fields with memory. Third, the activation fields are *robust* to disturbance and noise in the external input [**141**]. We harness these characteristics to specify a model for smooth temporal evolution of $\boldsymbol{p}(t)$ in the next section.

5.4.2. Field-Theoretic Intent Inference

Our insight is to use the framework of dynamic neural fields to specify the time evolution of the probability distribution $\boldsymbol{p}(t)$, in which we treat the individual goal probabilities $p^i(t)$ as constrained dynamical state variables such that $p^i(t) \in [0, 1]$ and $\Sigma_1^{n_g} p^i(t) = 1$. We refer to this approach as the *field-theoretic intent inference.*

The full specification of the field is given by

$$\frac{\partial \boldsymbol{p}(t)}{\partial t} = \frac{1}{\tau}\Big[\underbrace{-\boldsymbol{P}^T_{n_g \times n_g} \cdot \boldsymbol{p}(t)}_{\text{goal transition dynamics}} + \underbrace{\frac{1}{n_g} \cdot 1_{n_g}}_{\text{rest state}} \Big] + \underbrace{\boldsymbol{\lambda}_{n_g \times n_g} \cdot \sigma(\boldsymbol{\xi}(\boldsymbol{u}_h; \boldsymbol{\Theta}))}_{\text{excitatory + inhibitory}} \tag{5.7}$$

where time-scale parameter τ determines the memory capacity and decay behavior, $\boldsymbol{P}_{n_g \times n_g}$ is the state transition matrix for the embedded Markov chain that models the goal transitions as jump processes, 1_{n_g} is a vector of dimension n_g containing all ones, $\boldsymbol{u}_h$ is the human control input and $\boldsymbol{\Theta}$ represents all other task-relevant features, $\boldsymbol{\lambda}$ is the control matrix that controls the excitatory and inhibitory aspects, $\boldsymbol{\xi}$ is a function that encodes the nonlinearity through which human control commands and task features affect the time evolution, and σ is a biased sigmoidal nonlinearity given by $\sigma(\boldsymbol{\xi}) = \frac{1}{1+e^{-\boldsymbol{\xi}}} - 0.5$.

Our design of $\boldsymbol{\xi}$ is informed by what features of the human control input and environment effectively capture the human's underlying intent. We choose the *directedness* of the robot motion towards a goal, the *agreement* between the human and the autonomous agent's commands, and the *proximity* to a goal. The *directedness* component looks at the shortest straight line path towards a goal g, whereas the *agreement* serves as an indicator of how similar (measured as a dot product) the human and the autonomous agent's signals are to each other. One dimension i of $\boldsymbol{\xi}$ is defined as

$$\xi^i(\boldsymbol{u}_h; \boldsymbol{\Theta}) = \underbrace{\frac{1+\eta}{2}}_{\text{directedness}} + \underbrace{\boldsymbol{u}_h^{rot} \cdot \boldsymbol{u}_{a,g^i}^{rot}}_{\text{agreement}} + \underbrace{\max(0, 1 - \frac{\left\|\boldsymbol{x}_{g^i} - \boldsymbol{x}_r\right\|}{R})}_{\text{proximity}}$$

where $\eta = \frac{\boldsymbol{u}_h^{trans} \cdot (\boldsymbol{x}_{g^i} - \boldsymbol{x}_r)^{trans}}{\left\|\boldsymbol{u}_h^{trans}\right\| \left\|(\boldsymbol{x}_{g^i} - \boldsymbol{x}_r)^{trans}\right\|}$, $\boldsymbol{u}_{a,g^i}$ is the robot autonomy command for reaching goal g^i, *trans* and *rot* refer to the translational and rotational components of $\boldsymbol{u}_h$, $\boldsymbol{u}_h$ or $\boldsymbol{x}$, R is the radius of the sphere beyond which the proximity component is always zero, $\|\cdot\|$ is the Euclidean norm and $\boldsymbol{\Theta} = \{\boldsymbol{x}_r, \boldsymbol{x}_{g^i}, \boldsymbol{u}_{a,g^i}\}$ At every time-step, constraints on $p^i(t)$ are enforced such that $\boldsymbol{p}(t)$ is a valid probability distribution. The most confident goal g^* is computed as $g^* = \text{argmax}_i \, p^i(t) \; \forall \; i \in [1, \ldots, n_g]$.

5.4.3. Field-Theoretic Intent Inference for Assistive Robotics

In this work we assume that the autonomous agent's inference of user intent solely relies on user issued low-level robot control commands. In the domain of assistive robotics, it is quite often the case that the user input is highly discontinuous (due to fatigue, motor impairments, stoppage for mode switches, *et cetera*). Therefore, it is important to reason about belief over goals also in the *absence* of useful information.

According to the *principle of maximum entropy*, in the absence of testable information (no control commands issued by the user and a uniform global prior), the belief should converge to a uniform distribution over time. In the absence of $\boldsymbol{u}_h$, using Equation 5.7 and an appropriately chosen time-scale parameter τ, $\boldsymbol{p}(t)$ converges to a uniform distribution by correctly ignoring outdated information. The rate at which the distribution decays to a uniform distribution is controlled by τ.

By contrast, the standard discrete-time recursive belief update equation as implemented in [**87**] is

$$p(g^t|\boldsymbol{u}_h^t) = \eta p(\boldsymbol{u}_h^t|g^t) \sum_{g^{t-1} \in \mathcal{G}} p(g^t|g^{t-1}) p(g^{t-1}|\boldsymbol{u}_h^{t-1})$$

where η is a normalization factor, $p(\boldsymbol{u}_h^t|g^t)$ is a likelihood function, and $p(g^t|g^{t-1})$ is the goal transition probability. In the recursive belief update, when $\boldsymbol{u}_h = 0$ and the likelihood function is uniform, it can be shown that the posterior distribution over goals converges to the stationary distribution of the goal transition matrix. The stationary distribution is not necessarily uniform and can introduce unwanted biases in the inference.

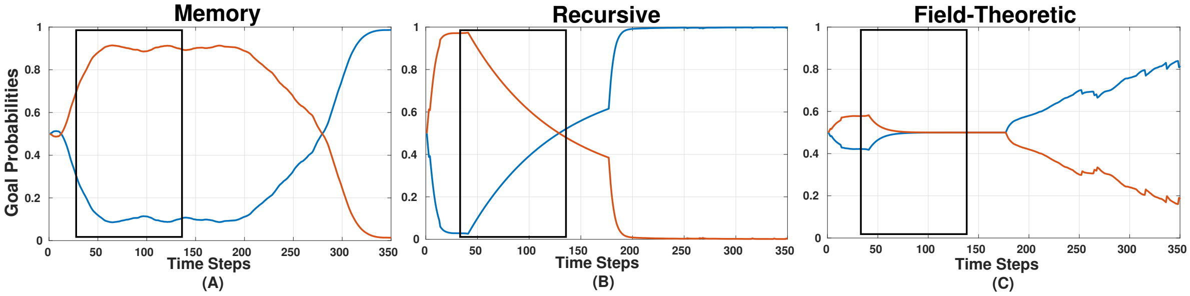


Figure 5.4. Inference comparison: Goal probabilities (blue and orange lines) for (A) Memory-based prediction, (B) Recursive Bayesian belief update and (C) Field-theoretic inference in a scene with two goals. Black rectangular boxes indicate times of zero control velocity, with varying effects on the inference schemes: (A) little change (since the cost function is purely distance-based), and convergence to (B) the stationary distribution of the goal transition matrix $\boldsymbol{P}$ and (C) the uniform distribution as dictated by the principle of maximum entropy.

Knowledge of task-level semantics can provide informative global priors that can further improve the accuracy of the inference mechanism. Our field-theoretic approach additionally can encode a task-level global prior in the 'rest state' term. For example, in a pick-and-place task, the initial goal distribution could be biased towards the object that needs to be picked. In order to evaluate the performance of our field-theoretic inference approach a quantitative comparison to (a) memory-based prediction [**49**] and (b) recursive belief updating [**87**] was implemented using point robot simulation in $\mathbb{R}^3$. The human was modeled as issuing a control command that noisily optimizes a straight-line path towards the intended goal. Signal dropout was simulated by randomly zeroing out control commands and τ was set to be 10. Additionally, $\boldsymbol{u}_h$ was set to be zero for a randomly chosen section of each trial in order to compare the convergence behavior of different approaches. The number of goals varied between three and five. Goal transitions were randomly sampled every five to eight time steps. The average length of the simulated trajectories was 615 time steps and 500 trials were simulated. Inference accuracy was computed as the fraction of total trial time (excluding when $\boldsymbol{u}_h = 0$) for which an algorithm correctly inferred the ground truth goal.

Results for field-theoretic inference outperformed memory-based prediction significantly (87.46% vs. 59.15% respectively) and were comparable to recursive belief updating (87.43%). Figure 5.4 shows an illustrative example of goal inference using the various methods. One can see that when there is no control command issued, the field-theoretic approach alone converges to a uniform distribution in agreement with the principle of maximum entropy.

5.5. Study Methods

In this section, we describe the study methods used to evaluate the efficacy of the disambiguation algorithm.

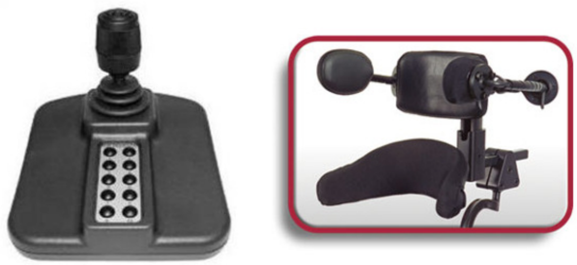
Participants: For this study eight subjects were recruited (mean age: 31 ± 11, 3 males and 5 females). All participants gave their informed, signed consent to participate in the experiment, which was approved by Northwestern University's Institutional Review Board.

Hardware: The experiments were performed using the MICO 6-DoF robotic arm (Kinova Robotics, Canada), specifically designed for assistive purposes. The software system was implemented using the Robot Operating System (ROS) and data analysis was performed in MATLAB. The subjects teleoperated the robot using two different control interfaces: a 2-axis joystick and a switch-based head array, controlling the 6D Cartesian velocity of the end-effector (Figure 5.5). An additional button was provided to request the mode switch assistance.

The joystick generated 2D continuous control signals. Under joystick control the full control space was partitioned into five control modes that were accessed via button presses. The switch-based head array consisted of three switches embedded in a headrest, operated via head movements, and generated 1D discrete signals. Under head array control the full control space was partitioned into seven control modes. The back switch was used to cycle between the different control modes, and the switches to the left and right controlled the motion of the robot's end effector in the positive and negative directions along a selected control dimension.

Tasks: Two different categories of tasks were evaluated.

Single-step: The aim was to reach one of five objects on the table, each with a pre-specified target orientation (Figure 5.6, Left).



Control Mappings

Mode	**Head Array**	**Joystick**
1	v_x	v_x, v_y
2	v_y	v_x, v_z
3	v_z	ω_z, ω_y
4	ω_z	ω_x
5	ω_y	gripper
6	ω_x	—
7	gripper	—

Figure 5.5. A 2-axis joystick (left) and switch-based head array (center) and their control mapping and operational paradigms (right). v and ω indicate the translational and rotational velocities of the end-effector, respectively.

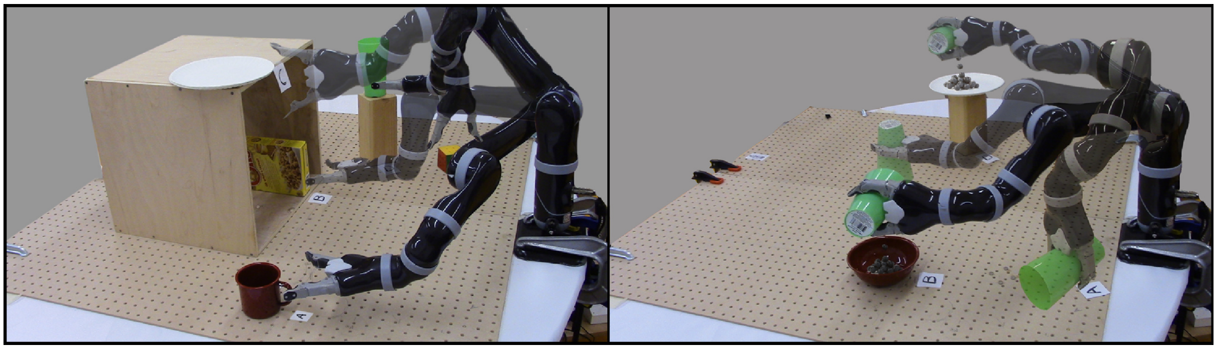


Figure 5.6. Study tasks performed by subjects. *Left:* Single-step reaching task. *Right:* Multi-step pouring task.

Multi-step: Each trial began with a full cup held by the robot gripper. In the first stage, the subjects were required to pour the contents of the cup into one of two containers, and subsequently in the second stage, the cup was to be placed at one of the two specified locations and with a particular orientation (Figure 5.6, right).

Switching Paradigms: Two kinds of mode switching paradigms were evaluated in the study.

Manual: During task execution the user performed all mode switches.

Disambiguation: The user either performed a mode switch manually or requested a switch to the *disambiguation* mode. The user was free to issue disambiguation requests at any time during the task execution, upon which the algorithm identified and switched the current control mode to the best disambiguation mode m^* by invoking Algorithm 3. During the trial, the user also was allowed to switch control modes using a manual mode switch at any time as well. The only requirement was that, the user request disambiguation at least once during the task execution.

Shared Control: Control-level assistance was always active for both mode switch assistance paradigms (manual and disambiguation). We used a blending-based shared-control paradigm in which the final robot control command was a linear composition of the human control command and an autonomous control command. The amount of assistance was directly proportional to the probability of the most confident goal g^*, and thus to the strength of the intent inference. The probability distribution over goals, $\boldsymbol{p}(t)$, was updated using Equation 5.7 as outlined in Section 5.4.2 and the most confident goal was computed as $g^{\operatorname{argmax}_i p^i(t)}$. Therefore, if intent inference improved as a result of goal disambiguation, more assistance would be provided by the autonomous agent.

Specifically, the autonomous control policy generated control command $\boldsymbol{u}_a \leftarrow f_a(\boldsymbol{x}_r)$ where $f_a(\cdot) \in \mathcal{F}_a$, and $\mathcal{F}_a$ was the set of all control behaviors corresponding to different tasks. $\mathcal{F}_a$ could be derived using a variety of techniques such as *Learning from Demonstrations* [**8**], motion planners [**76**] or navigation functions [**135**].

In our implementation, the autonomous agent's control command $\boldsymbol{u}_{a,g}$ was generated using a simple potential field which was defined in all parts of the state space [**95**]. Every goal g was associated with a potential field γ_g which treated g as an attractor and all other goals in the scene as repellers. The autonomy command was computed as a summation of the attractor and repeller velocities and operated in the full 6D Cartesian space. For potential field γ_g, the attractor velocity is given by

$$\dot{\boldsymbol{x}}_r^{attract} = \boldsymbol{x}_g - \boldsymbol{x_r}$$

where $\boldsymbol{x}_g$ is the location of goal g.[4] The repeller velocity was given by

$$\dot{\boldsymbol{x}}_r^{repel} = \sum_{i \in \mathcal{G} \setminus g} \frac{\boldsymbol{x_r} - \boldsymbol{x}_{g^i}}{\mu(\left\|\boldsymbol{x_r} - \boldsymbol{x}_{g^i}\right\|^2)}$$

where $\dot{\boldsymbol{x}}_r$ indicated the velocity of the robot in the world frame, μ controlled the magnitude of the repeller velocity and $\|\cdot\|$ is the Euclidean norm. The autonomy command associated with goal g was then computed as a summation of these attractor and repeller velocities and was given by

$$\boldsymbol{u}_{a,g} = \dot{\boldsymbol{x}}_r^{attract} + \dot{\boldsymbol{x}}_r^{repel}.$$

γ_g operated in the full six dimensional Cartesian space, and treated position and orientation as independent potential fields.

Under blending, the final control command $\boldsymbol{u}_f$ issued to the robot was computed as

$$\boldsymbol{u}_f = \alpha \cdot \boldsymbol{u}_{a,g^*} + (1 - \alpha) \cdot \boldsymbol{u}_h \tag{5.8}$$

where g^* was the most confident goal. Similar to $\boldsymbol{u}_h$, the autonomy command $\boldsymbol{u}_{a,g^*} \in \mathbb{R}^6$ was mapped to the 6D Cartesian velocity of the end-effector. The blending factor α was a piecewise linear function of the probability $p(g^*)$ associated with g^* and was given by

$$\alpha = \begin{cases} 0 & p(g^*) \le \rho_1 \\ \frac{\rho_3(p(g^*) - \rho_1)}{\rho_2 - \rho_1} & \text{if} \quad \rho_1 < p(g^*) \le \rho_2 \\ \rho_3 & \rho_2 < p(g^*) \end{cases}$$

[4] In position space, the '–' operator computes the difference between the goal position and current robot position in $\mathbb{R}^3$. In orientation space, the '–' operator computes the *quaternion difference* between the goal orientation and the current robot orientation.

with $\rho_i \in [0, 1]\ \forall\ i \in [1, 2, 3]$ and $\rho_2 > \rho_1$. In our implementation, we empirically set $\rho_1 = \frac{1.2}{n_g}, \rho_2 = \frac{1.4}{n_g}$ and $\rho_3 = 0.7$.

Study protocol: A within-subjects study was conducted using a fractional factorial design in which the manipulated variables were the tasks, control interfaces, and the switching paradigm conditions. Each subject underwent an initial training period that lasted approximately 30 minutes. The training period consisted of three phases and two different task configurations. The subjects used both interfaces to perform the training tasks.

Phase One: The subjects were asked to perform a simple reaching motion towards a single goal in the scene. This phase was intended for the subjects to get familiarized with the control interface mappings and teleoperation of the robotic arm.

Phase Two: Subjects were asked to perform a simple reaching motion towards a single goal in the scene in the presence of blending-based autonomous assistance.

Phase Three: Subjects were able to explore the disambiguation request feature during a reaching task to observe the effects of the mode switch request and subsequent change in robot assistance. Multiple objects were introduced in the scene. Subjects were explicitly informed that upon a disambiguation request the robot would select a control mode that would help the autonomy determine the subject's intended goal and thereby enable it to assist the user more effectively.

During the testing phase, each subject performed both tasks using both interfaces under the *Manual* and *Disambiguation* paradigms. All trials started in a randomized initial control mode and robot position. The ordering of control interfaces and paradigms was randomized and counterbalanced across all subjects. Three trials were collected for the *Manual* paradigm and five trials for the *Disambiguation* paradigm. On an average,

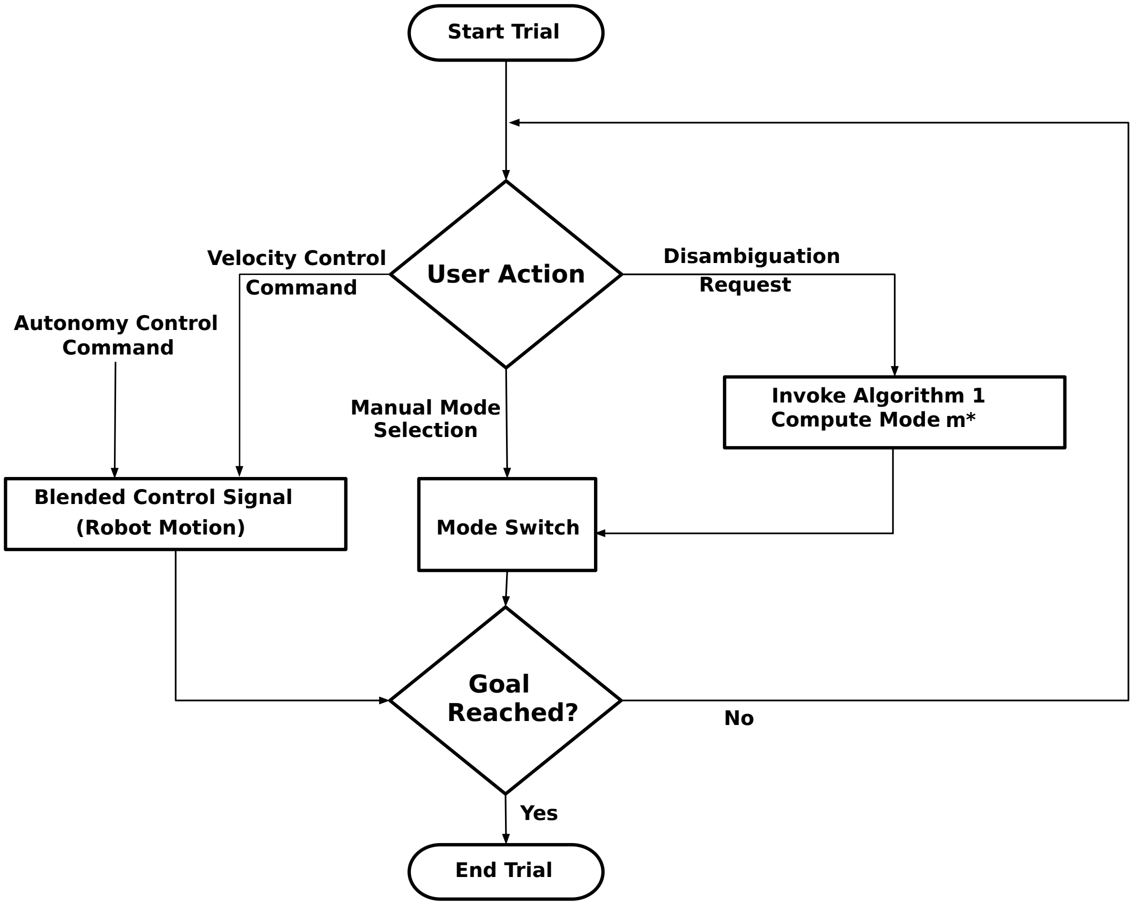


Figure 5.7. Flow chart depicting user action sequence during a single trial. The user could issue a (a) velocity control command, resulting in intent inference followed by generation of an autonomy signal and then blended control signal, and causes robot motion or (b) manual mode switch or (c) disambiguation request, both resulting in a control mode switch.

each trial lasted approximately 10-40s depending on the starting position of the robot and the specified reaching target. At the start of each trial, $p^i(t)$ for $i \in [1, 2, \ldots, n_g]$ was initialized as $\frac{1}{n_g}$. During the trial as the user teleoperated the robot, $\boldsymbol{p}(t)$ was updated according to Equation 5.7 online at each time step. Figure 5.7 captures how a single trial unfolds in time.

Metrics: The objective metrics used for evaluation included the following.

- *Number of mode switches*: The number of times a user switched between various control modes during task execution. This metric captures one of the main factors that contributes to the cognitive and physical effort required for task execution in assistive robotic manipulation [**71**].
- *Number of disambiguation requests*: The number of times a user pressed the disambiguation request button.
- *Number of button presses*: The sum of *Number of mode switches* and *Number of disambiguation requests.*
- *Skewness*: A higher-order moment used to quantify the asymmetry of any distribution. Used to characterize how much the temporal distribution of disambiguation requests deviates from a uniform distribution.
- *Task completion time*: Time taken to complete the task successfully. This metric is an indicator of how well the human and the autonomous agent work together.

Additionally, at the end of each testing phase, subjective data was gathered via a brief questionnaire. Users were given the following statements regarding the usefulness and capability of the assistance system to rate according to their agreement on a 7-point Likert scale.

- **Q1** - Control modes chosen by the system made task execution easier.
- **Q2** - The robot and I worked together to accomplish the task.
- **Q3** - I liked operating the robot in the control modes chosen by the system.

Subjects were also asked to indicate their preference in the following questions.

- **Q4** - Which interface was the hardest to operate?

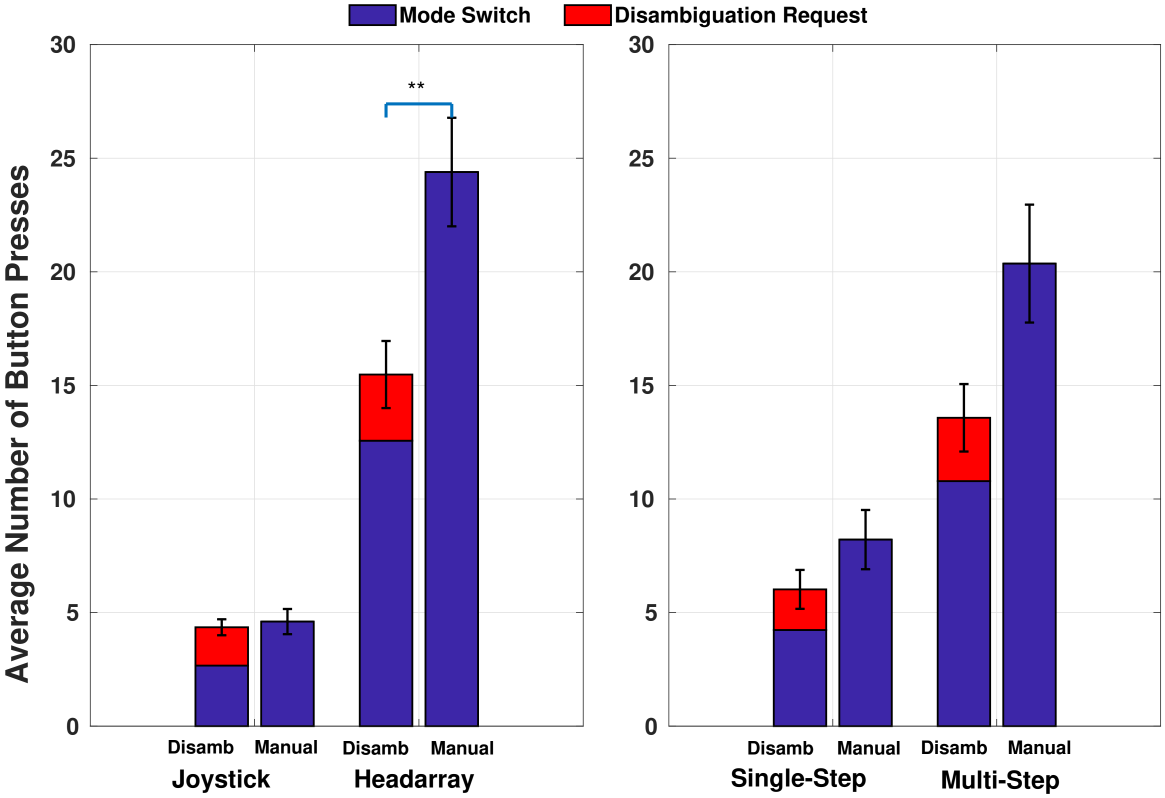


Figure 5.8. Average number of button presses, *Disambiguation* and *Manual* paradigms. *Left:* Grouped by control interfaces. *Right:* Grouped by tasks.

- **Q5** - For which interface was the assistance paradigm the most useful?
- **Q6** - Which one of the schemes do you prefer the most?
- **Q7** - Which one of the schemes is the most user-friendly?

5.6. Results

In this section we present results from our human-subject study. The study results indicate that the disambiguation request system is of greater utility for *more limited control interfaces* and *more complex tasks*. Subjects demonstrate a wide range of disambiguation request behaviors with a common theme of relying on disambiguation assistance

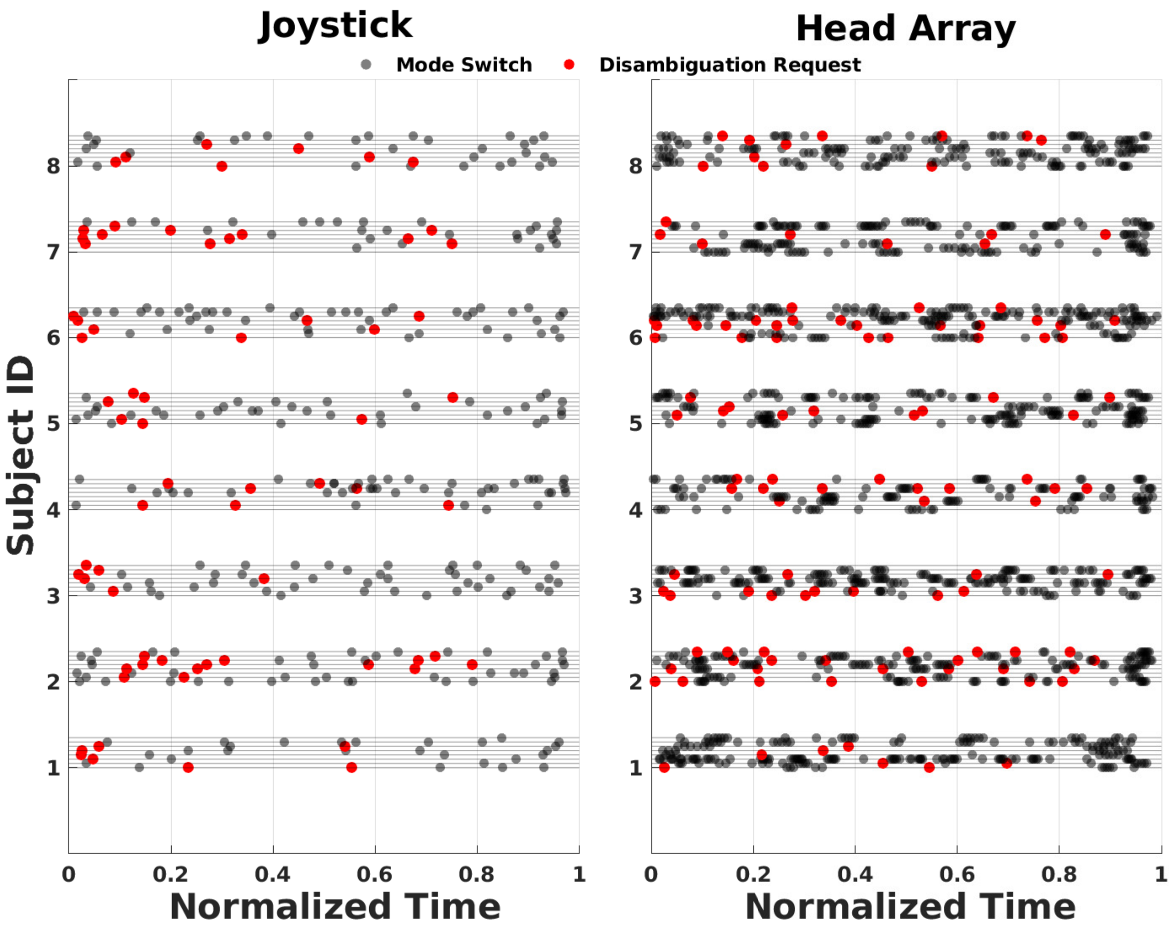


Figure 5.9. Temporal pattern of button presses for joystick (left) and head array (Right) during the multi-step task on a trial-by-trial basis for all subjects. For each subject, each light gray horizontal line represents a single trial. Eight trials per subject, for each interface.

earlier in the trials. Furthermore, the survey results show that operating the robot in the disambiguating mode make task execution easier and that users prefer the *Disambiguation* paradigm to the *Manual* paradigm. Statistical significance is determined using the Wilcoxon Rank-Sum test in where (***) indicates $p < 0.001$, (**) $p < 0.01$, and (*) $p < 0.05$.

Impact of Disambiguation: A statistically significant decrease in the number of button presses is observed between the *Manual* and *Disambiguation* paradigms when using the head array (Figure 5.8, left). Due to the low-dimensionality of the head array and cyclical nature of mode switching, the number of button presses required for task completion is inherently high. The disambiguation paradigm is helpful in reducing the number of button presses likely due to higher robot assistance that is present in the disambiguating control mode. For the joystick, statistically significant differences between the two paradigms are observed for the number of manual mode switches ($p < 0.05$). However, this gain is offset by the button presses that are required to make disambiguation requests. When grouping by task, the general trend of a decrease in the number of button presses is more pronounced for the more complex multi-step task (Figure 5.8, right). Although not statistically significant, we also observe that the autonomy has a slightly higher control authority (as measured by α) during the disambiguation trials ($\alpha = 0.27 \pm 0.16$) when compared to the manual trials ($\alpha = 0.25 \pm 0.16$).

These results suggest that disambiguation is more useful as the control interface becomes more limited and the task becomes more complex. Intuitively, intent prediction becomes harder for the robot when the control interface is lower dimensional as it does not reveal a great deal of information about the user's underlying intent. By having the users operate the robot in the disambiguating control mode, the autonomous agent is able to elicit more intent-expressive control commands from the human which in turn helps in accurate goal inference and subsequently appropriate assistance.

Temporal Distribution of Disambiguation Requests: In Figure 5.9 the frequency and density of button presses (disambiguation requests plus mode switches) are much

Table 5.3. Skewness of the temporal distribution of disambiguation requests.

	Single-step	Multi-step
Joystick	0.63	0.57
Head Array	0.35	0.22

higher for the more limited control interface (head array). We observe that a higher number of disambiguation requests correlates with the more limited interface and complex task. The subjects also demonstrate a diverse range of disambiguation request behaviors, in regards to both (a) when the disambiguation requests are made and (b) with what frequency (e.g., Subject 1 vs. Subject 2, Joystick). The variation between subjects is likely due to different factors such as the user's comfort in operating the robot and understanding of the disambiguating mode's ability to recruit more assistance from the autonomy.

The temporal distribution of disambiguation analyzes *when* the subject requested assistance during the course of a trial. The skewness of the temporal distribution of disambiguation requests reveals a higher concentration of requests during the earlier parts of a trial (Table 5.3) for both interfaces and tasks.[5] However, under head array control the temporal distribution is less skewed, indicating that the need for disambiguation request persists throughout the trial, likely due to the extremely low-bandwidth of the interface.

Performance: No statistical difference is observed in task completion times between the *Manual* and *Disambiguation* paradigms (Figure 5.10). However, the variance in the task completion times in general is lower under the *Disambiguation* conditions (with the

[5] A uniform temporal distribution corresponds to a trial in which the disambiguation requests are uniformly spread out during the course of task execution. The skewness of a uniform distribution is zero.

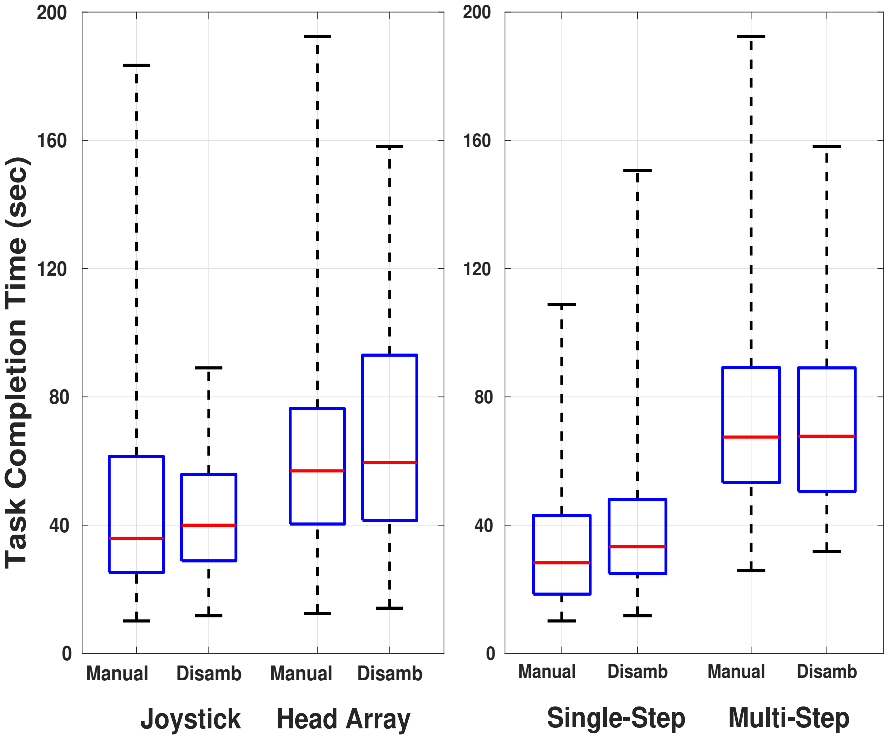


Figure 5.10. Task completion times. *Disambiguation* and *Manual* paradigms. *Left:* Grouped by control interfaces. *Right:* Grouped by tasks.

exception of single-step), indicating more consistent task performance when disambiguation requests are utilized. The task success is 92.50% (148 out of 160 trials) and 97.92% (94 out of 96 trials) for the *Disambiguation* and *Manual* paradigms respectively. 13 out of the 14 unsuccessful trials occur during the more complex multi-step task. Figure 5.11 provides illustrative examples of the time evolution of goal probabilities and demonstrates how operation in the disambiguating mode can very quickly elevate one goal probability above the threshold for providing autonomy assistance. Figure 5.11 (right) demonstrates how, at times, subjects do not leverage the capabilities of the disambiguating mode and

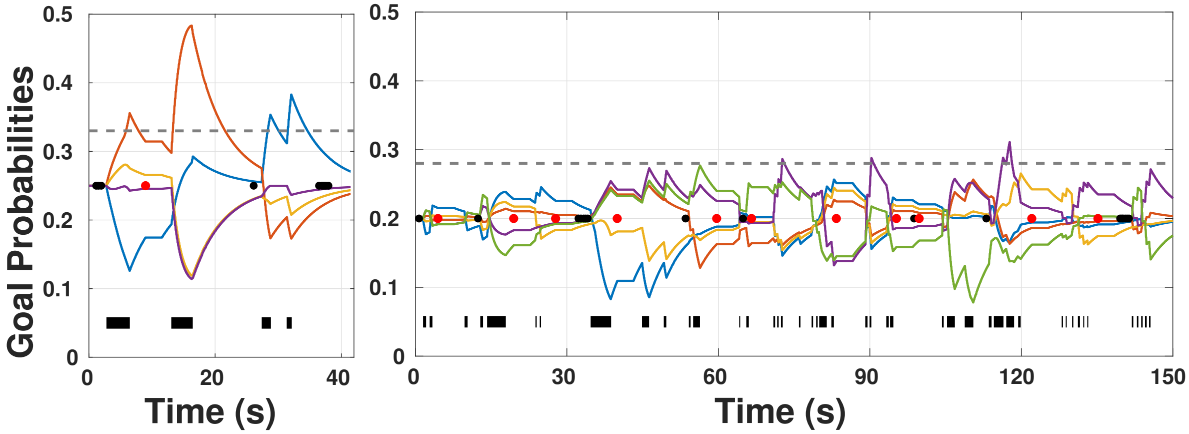


Figure 5.11. Time evolution of goal probabilities. Plot annotations include the minimum threshold for robot assistance (gray dashed line), disambiguation requests (red dots), manual mode switches (red dots), and indications of non-zero human control commands (black segments below the plotted data).

immediately perform a manual mode switch, without ever issuing any control commands in the disambiguating mode.

User Survey: Table 5.4 summarizes the results of the user survey. Users agree that task execution is easier during disambiguation trials (**Q1**, 4.88±0.95) and that operation under disambiguating modes is enjoyable (**Q3**, 5.00±1.15). User responses strongly validate the effectiveness of the blending-based shared control scheme (**Q2**, 6.19±0.75). Unsurprisingly, all users feel that it is harder to control the robot using the head array (**Q4**) and rate the utility value of the disambiguation paradigm to be higher for robot control with the head array (**Q5**). Although the subjects overwhelmingly prefer the *Disambiguation* to the *Manual* paradigm (**Q6**) only four out of the eight subjects find the *Disambiguation* paradigm to be user-friendly (**Q7**). One possible explanation is a lack of transparency regarding why the autonomous agent chose the disambiguating mode.

Table 5.4. Subjective Survey Results

	Across Tasks	Single-step	Multi-step
Q1	4.88 ± 0.95	4.88 ± 0.99	4.88 ± 0.99
Q2	6.19 ± 0.75	6.25 ± 0.89	6.13 ± 0.64
Q3	5.00 ± 1.15	5.25 ± 1.28	4.75 ± 1.03
Q4	Head Array	Head Array	Head Array
Q5	Head Array	Head Array	Head Array
Q6	Disambiguation	Disambiguation	Disambiguation
Q7	Disamb/Manual	Disamb/Manual	Disamb/Manual

5.7. Discussion

The disambiguation algorithm presented in this chapter can be utilized in any human-robot system in which there is a need to disambiguate between the different states of a discrete hidden variable (for example, a set of discrete goals in robotic manipulation or a set of landmarks in navigation tasks). The disambiguation algorithm assumes the existence of a discrete set of parameters (for example, control modes for robotic manipulation or natural language based queries for navigation) that can help the intent inference mechanism to precisely converge to the correct solution. Although the disambiguation algorithm is task-agnostic—because it relies exclusively on the shape features of the probability distribution over the hidden variable—the disambiguation is only as good as the efficacy of the inference algorithm that is used. In our experience, the choice of cost functions and domain-specific heuristics used for inference need to be appropriate for the task at hand. During our implementation development, the efficacy of the disambiguation algorithm degraded when we used only a subset of the four features to inform the disambiguation metric. This only reinforces the need for a combination of different shape features for successful disambiguation.

Another observation from the subject study is how often participants submitted a disambiguation request and then chose not to operate in the selected mode—effectively not letting the robot help them. This under-utilization phenomenon is illustrated in Figure 5.11 (right). One possible explanation is the subject's lack of understanding of how their control commands can help the robot to understand their intent. It is likely that a good grasp of the assistance mechanism is critical for providing intent-expressive control commands to the autonomy—underlining the need for extensive and thorough training for greater transparency in the human-robot interaction, so that the human has a clear understanding of how and why the autonomous agent chooses a specific assistance strategy.

The training can be made more effective in a few different ways. For example, the subjects could be explicitly informed of the task relevant features (directedness, proximity *et cetera*) that the autonomous agent relies on for determining the amount of assistance to offer. Knowledge of these features might motivate the users to leverage the assistance offered when operating in the disambiguating mode more.

The inherent time delays associated with the computation of the disambiguation mode (approximately 2-2.5s) might have been a cause for user frustration. Half of the subjects did report that the disambiguation system was not user-friendly. To improve upon this delay, a large set of disambiguating modes could be precomputed for different parts of the workspace, goal configurations, and goal priors ahead of time, which then could be available as a lookup table during task execution. Automated mode switching schemes that eliminate the need for manual button presses altogether might also be a viable option for significantly reducing task effort.

5.8. Conclusions

In this chapter, we presented the idea of *intent disambiguation assistance* via control mode selection for a shared-control assistive robotic arm. The aim of the control mode selection algorithm was to elicit more *intent-expressive* control commands from the user by placing control in those control modes that *maximally disambiguate* between the various goals in a scene. A pilot user study was conducted with eight subjects to evaluate the efficacy of the disambiguation system. The study results indicated a decrease in task effort in terms of the number of button presses when the disambiguation system was active. As the last contribution, we also presented a novel intent inference mechanism inspired by *dynamic field theory* that works in conjunction with the proposed disambiguation algorithm.

Informed by the findings and shortcomings of the work presented in this chapter, in the next chapter a more refined algorithmic framework to perform intent disambiguation is presented.

CHAPTER 6

Information Theoretic Intent Disambiguation via Contextual Nudges for Assistive Shared Control

In the previous chapter, a heuristic was designed to evaluate the intent disambiguation capabilities of different control modes and the users were responsible for activating the algorithm during their interaction with the autonomous agent. This chapter further develops the ideas presented in Chapter 5 and formalizes the notion of *interface-aware intent disambiguation* by rigorously grounding it in information-theoretic principles. The disambiguation metric proposed in this chapter enables the autonomous agent not only to reason about the disambiguation capabilities of control modes but also of the entire state space that constitutes robot location as well. This chapter also proposes a turn-taking based Human-Autonomy Interaction (HAI) protocol in which the autonomous agent activates the proposed disambiguation algorithm during its turn when the uncertainty about its prediction of user intent is high. We present results from a nine person human subject study which suggest that disambiguation (a) helps to significantly reduce task effort, as measured by number of mode switches, task completion times, and number of turns executed by the human and (b) enables the autonomous agent to provide accurate assistance with greater confidence and contribution to the overall control signal. The work presented in this chapter was done in collaboration between Deepak Gopinath and Andrew Thompson.[1]

6.1. Introduction

A fundamental problem in robotics is that of state estimation from noisy sensor data [**13**]. The primary goal of any state estimation algorithm is to reduce the uncertainties that arise from noisy measurements and this can become particularly hard due to limited information channels and associated hardware constraints. In HAI scenarios, usually, state estimation performed by the autonomous agent not only involves estimation of the environment state, but also of the unobserved latent human state that encodes their goals, beliefs, and intentions. Particularly, in the domain of shared autonomy assistive robots, the effectiveness of an assistive autonomous agent depends on how well it is able to infer the user's intentions *unambiguously* from the control interface signals that are generated by the human.

In Chapter 5 we motivated the need for intent disambiguation in the context of human-machine interaction as a strategy to improve the autonomous agent's ability to infer human intent from low-dimensional, low-bandwidth signals generated via limited control interfaces. In this chapter, we frame *intent disambiguation* as a problem of optimally *nudging* the user's environment (decision making context) such that their subsequent control interface actions are guaranteed to result in *maximal information gain* regarding

[1] The individual contribution breakdown is as follows: 1) Development of the initial idea, disambiguation metric, and code was done by Deepak Gopinath, who was also the primary proctor for the subject study. 2) Andrew Thompson assisted in study protocol design and preparation, and benchmarking the computational efficiency of the algorithm.

the user's latent intentions. In this work, we also explicitly incorporate the impact of the control interface's inherent noisy characteristics on information gain.

The key contributions of this chapter are three-fold:

(1) We propose an interface-aware information-theoretic framing of the problem of *intent disambiguation*.

(2) We propose a turn-taking based HAI protocol in which the autonomous agent utilizes the proposed disambiguation metric to *help itself* when uncertain about its prediction of human intent.

(3) We present results from a nine person human subject study that evaluated the effectiveness of the proposed disambiguation metric and turn-taking based protocol.

The proposed disambiguation metric and turn-taking based interaction protocol is validated in simulation as well as with a human subject-study.

6.2. Intent Disambiguation as an Algorithm for Nudging

When the autonomous agent uses the intent disambiguation algorithm to nudge the robot into maximally disambiguating states, it is an example of how a *decision-making* context is altered to indirectly affect the decisions taken by the human. This phenomenon, known as *nudging*, is extensively studied in the sphere of behavioral economics [**157**], public policy [**55**], smart technologies [**113**], and business marketing [**90**]. Originally proposed by Thaler and Sunstein in [**158**], *nudging* is the mechanism by which any aspect of the choice architecture is modified in an attempt to influence peoples' behaviors in a predictable manner. Choice architecture refers to the organization and presentation of the choices that a decision maker have. Note that, *nudging* is not same as introducing an arbitrary number of constraints, but rather it is an attempt to influence the decision maker's choice without limiting the choice set or making other alternatives more costly. In the domain of robotics, applications of nudge theory have been explored in the context of social robotics, particularly with respect to the ethics and morality of nudging human when they interact with robots [**136, 153**]. More recently, a computational account of optimal nudging was proposed in [**31**], in which nudging was framed as modifications to the costs of different cognitive operations.

In the case of human-autonomy interaction scenario discussed in this chapter, the *context* is the environment state in which the human is required to generate actions. Altering context amounts to changing some aspects of the environment state. Moreover, the context change is initiated by the autonomous agent. That is, the autonomous agent *intervenes* to alter the context in specific ways in order to influence the human to act in a certain manner. In the case of control interface mediated robot teleoperation, a contextual nudge could be a change to the active mode, or in the robot location, or both. In Chapter 5, we focus on *nudges* that are restricted to mode changes, whereas in the formulation presented in this chapter the changes to the context affect both the robot location and mode. Additionally, nudges are introduced not only to help the human make better decisions, but also to help the autonomous agent itself which in turn will benefit the human.

6.3. Assistance via Turn Taking

In a shared autonomy system in which both the human and the autonomous agent control the *same physical device* at the *same time*, for the human to isolate the autonomous agent's contribution to the overall control signal can become hard as feedback is limited. To estimate the autonomous agent's control contribution requires either a good mental model of the autonomous agent's policy and understanding of the control arbitration scheme, or mentally *subtracting* away the effect of their own issued controls. In human-human interaction, it has been established that humans rely on *theory of mind* mechanisms to explain other agents' actions. Humans could rely on such theory of mind explanations to make sense of the autonomous agent's behavior as well [**11, 12**]. However, all of these assume access to observations that are not corrupted by the human's own actions. In turn-taking, the bulk of the autonomous agent's actions are executed during its own turn *without any contribution from the human* and as a result it becomes easier for an observer to infer the latent assistance strategies from *uncorrupted* trajectory rollouts initiated by the autonomous agent.

In general, turn taking is one of more common HAI protocols used in a variety of robotics subfields. Fluency in HAI based on information flow is studied [**147, 159**]. Assistive robots utilize turn-taking as an interaction paradigm for therapy of children with autism spectrum disorder [**149**]. Conversational turn-taking robots that rely on gestures and natural language modalities are extensively used to understand social dynamics of human-robot interaction [**100**]. An additional motivation for the target domain is that turn-taking could provide a natural framework which affords periods of rest for the human. Taking sufficient rest becomes particularly important (especially with limited interfaces such as a sip-and-puff) as continuous manual teleoperation is physically taxing and can quickly result in fatigue.

6.4. Mathematical Formalism

In this section, we present the mathematical formulation of our interface-aware information theoretic intent disambiguation algorithm. A probabilistic graphical model of limited control-interface mediated robot teleoperation is presented in Section 6.4.1. In Section 6.4.2 we describe the recursive Bayesian intent estimation algorithm used by the autonomous agent to determine the user's intended goal. We also present a turn-taking based human-autonomy interaction protocol that will embed the disambiguation algorithm as a part of the autonomous agent's interaction strategy.

6.4.1. Modeling Limited Control-Interface Mediated Robot Teleoperation

Figure 6.1 depicts the probabilistic graphical model of control interface mediated robot teleoperation by a goal-directed human.

We model the human teleoperating the robot towards a goal $g \in \mathcal{G}$ using a limited control interface as an **interface dependent goal-directed Markov Decision Process** (MDP) denoted by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}_g, \gamma, \rho_0, \Delta)$, where $\mathcal{R}_g : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the goal-dependent reward, $\gamma \in [0, 1)$ is the discount factor, Δ is an interface-dependent parameter that determines the mode-switching behavior and ρ_0 is the initial state distribution. We model the robot-interface system as a deterministic dynamical system with a transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$, where $\mathcal{S} = \mathcal{Q} \times \mathcal{M}$ is the state space that comprises of the robot location ($\mathcal{Q}$) and the set of control modes ($\mathcal{M}$). $\mathcal{A}$ is the set of all task-level actions.

$\mathcal{A}$ can be decomposed as $\mathcal{A}_q \times \mathcal{A}_m$ where $\mathcal{A}_q$ is the set of *task-level control actions* and $\mathcal{A}_m$ is the set of *control-level mode switch actions*. As a result, $\mathcal{T}$ is a hybrid dynamical system consisting of $\mathcal{T}_q : \mathcal{Q} \times \mathcal{A}_q \rightarrow \mathcal{Q}$, which determines how the task-level control actions result in *motion* or equivalently changes to the robot location and $\mathcal{T}_m : \mathcal{M} \times \mathcal{A}_m \rightarrow \mathcal{M}$ which determines how control-level mode switch actions determine the current active mode. Additionally, we also define two feature extractors $\Psi_q(s) = q$ and $\Psi_m(s) = m$ $\forall s = (q, m) \in \mathcal{S}$. Additionally, we also have $\mathcal{G} \subset \mathcal{Q}$, so that goals are defined as desired locations for the robot.

We solve for the goal-dependent optimal policy (which is a mapping from state s to a distribution over actions a), denoted as π^g_{optim} using standard value iteration and treat the goal-dependent human's stochastic policy, $p(a|s, g)$ to be an ϵ-greedy policy that can be written as

$$p(a|s, g) = (1 - \epsilon)\pi^g_{optim} + \epsilon\pi_{unif} \tag{6.1}$$

where π_{unif} is a uniform policy and $\epsilon \in (0, 1)$.

6.4.2. Recursive Bayesian Goal Inference

At any time t, the human's true intended goal is latent and unobservable to the autonomous agent and as such it maintains a belief over goals denoted as b^t_g. The process of goal inference amounts to computing the posterior over goals given the history of measured interface-level actions, $\phi^{0:t}_m$ and states $s^{0:t}$. More precisely, we can use Bayesian

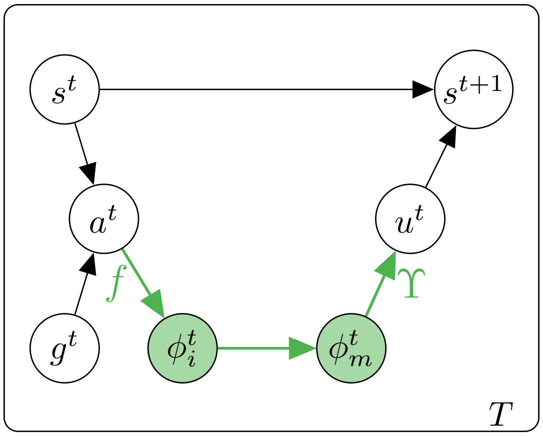


Figure 6.1. Probabilistic graphical model depicting a specific user's goal directed interactions with the robot via the control interface at single time step t. The nodes and edges that model the physical aspect of controlling the interface are highlighted in green. In this graphical depiction s^t subsumes both q^t (the robot state) as well as m^t (the control mode). ϕ_i^t and ϕ_m^t are the *intended* and measured interface-level physical actions respectively. These variables encode the specific physical activation mechanisms needed to generate a signal using an interface. More details on interface-level physical actions to be found in Section 4.2.1 of Chapter 4

inference to compute the posterior $b_g^t = p(g^t|\phi_m^{0:t}, s^{0:t})$ as

$$b_g^t \propto p(g^t|\phi_m^{0:t-1}, s^{0:t})p(\phi_m^t|g^t, \phi_m^{0:t-1}, s^{0:t}) \tag{6.2}$$

and using the conditional independence assumptions encoded in the model shown in Figure 6.1 we can remove the dependence of $p(\phi_m^t|g^t, \phi_m^{0:t-1}, s^{0:t})$ in Eq. 6.2 on $\phi_m^{0:t-1}$ and $s^{0:t-1}$. Then Eq. 6.2 becomes,

$$b_g^t \propto p(g^t|\phi_m^{0:t-1}, s^{0:t})p(\phi_m^t|g^t, s^t). \tag{6.3}$$

Marginalizing over a^t and ϕ_i^t we can express Eq. 6.3 as

$$b_g^t = \eta \cdot p(g^t|\phi_m^{0:t-1}, s^{0:t}) \sum_{a^t \in \mathcal{A}} \sum_{\phi_i^t \in \Phi} p(\phi_m^t|\phi_i^t) p(\phi_i^t|a^t) p(a^t|s^t, g^t) \tag{6.4}$$

with η is the normalization factor. $p(\phi_i^t|a^t)$ capturing the *user's internal model of the true mapping* (denoted as f) between task level actions and interface-level physical actions. $p(\phi_m^t|\phi_i^t)$ is the *user input distortion model*, which captures the stochastic deviations of the *measured* interface-level physical actions from the *intended* interface-level physical actions. $p(g^t|\phi_m^{0:t-1}, s^{0:t})$ can be recast as,

$$\begin{aligned} p(g^t|\phi_m^{0:t-1}, s^{0:t}) &= \sum_{g^{t-1} \in \mathcal{G}} p(g^t, g^{t-1}|\phi_m^{0:t-1}, s^{0:t}) \\ &= \sum_{g^{t-1} \in \mathcal{G}} p(g^t|g^{t-1}, \phi_m^{0:t-1}, s^{0:t}) p(g^{t-1}|\phi_m^{0:t-1}, s^{0:t}) \\ &= \sum_{g^{t-1} \in \mathcal{G}} p(g^t|g^{t-1}) \underbrace{p(g^{t-1}|\phi_m^{0:t-1}, s^{0:t-1})}_{b_g^{t-1}} \end{aligned} \tag{6.5}$$

under the assumption that g^t only depends on g^{t-1} and that the state at time t does not influence the goal at time $t-1$. Combining Eq. 6.5 with Eq. 6.4 the recursive Bayesian update for goal inference is given by

$$b_g^t = \eta \left[\sum_{g^{t-1} \in \mathcal{G}} b_g^{t-1} \cdot p(g^t|g^{t-1}) \right] \sum_{a^t \in \mathcal{A}} \sum_{\phi_i^t \in \Phi} p(\phi_m^t|\phi_i^t) p(\phi_i^t|a^t) p(a^t|s^t, g^t). \tag{6.6}$$

Under the assumption that the goal transition probability is a delta distribution, the above equation can be further simplified as

$$b_g^t = \eta \cdot b_g^{t-1} \sum_{a^t \in \mathcal{A}} \sum_{\phi_i^t \in \Phi} p(\phi_m^t | \phi_i^t) p(\phi_i^t | a^t) p(a^t | s^t, g^t). \tag{6.7}$$

6.4.3. Disambiguation Metric

We formalize intent disambiguation as a characterization of the states in the state space $\mathcal{S}$ according to their potential to *extract* interface signals ϕ_m from the user that provide the most information about the latent goal $g \in \mathcal{G}$. Intent disambiguation is particularly useful when the measured control signals, ϕ_m are very noisy and sparse. An autonomous agent that uses the intent disambiguation algorithm can nudge the robot into *maximally disambiguating* state(s), such that *subsequent* actions executed by the user will help the autonomous agent accurately infer human intent.

In order to perform intent disambiguation, the autonomous agent needs to have a notion of the amount of information contained in ϕ_m^t about g^t conditioned on the current state and the past history.

To be precise, at any given time t, if $\phi_m^{0:t-1}$ and $s^{0:t-1}$ represent the history of interface-level actions and states that the autonomous agent have observed, then the **conditional mutual information**, $I(\phi_m^t; g^t | \phi_m^{0:t-1}, s^{0:t-1}, s^t)$ between ϕ_m^t and g^t conditioned on $\phi_m^{0:t-1}$ and $s^{0:t-1}$ and the current state s^t measures the amount of information obtained about g^t by observing ϕ_m^t. Using the standard definition of conditional mutual information we then have

$$I(\phi_m^t; g^t|\phi_m^{0:t-1}, s^{0:t-1}, s^t) = \sum_{\phi_m^t\in\Phi}\sum_{g^t\in\mathcal{G}} p(\phi_m^t, g^t|\phi_m^{0:t-1}, s^{0:t-1}, s^t)\log\frac{p(\phi_m^t|g^t, \phi_m^{0:t-1}, s^{0:t-1}, s^t)}{p(\phi_m^t|\phi_m^{0:t-1}, s^{0:t-1}, s^t)}. \tag{6.8}$$

The term $p(\phi_m^t, g^t|\phi_m^{0:t-1}, s^{0:t-1}, s^t)$ can be rewritten as

$$p(\phi_m^t, g^t|\phi_m^{0:t-1}, s^{0:t-1}, s^t) = p(g^t|\phi_m^{0:t-1}, s^{0:t-1}, s^t)p(\phi_m^t|g^t, \phi_m^{0:t-1}, s^{0:t-1}, s^t). \tag{6.9}$$

By using the conditional independence assumptions encoded in the model, we can further simplify $p(\phi_m^t|g^t, \phi_m^{0:t-1}, s^{0:t-1}, s^t)$ to $p(\phi_m^t|g^t, s^t)$ and $p(\phi_m^t|\phi_m^{0:t-1}, s^{0:t-1}, s^t)$ as $p(\phi_m^t|s^t)$ by removing the dependence on $\phi_m^{0:t-1}, s^{0:t-1}$. Therefore, Eq. 6.8 becomes

$$I(\phi_m^t; g^t|\phi_m^{0:t-1}, s^{0:t-1}, s^t) = \sum_{g^t\in\mathcal{G}} p(g^t|\phi_m^{0:t-1}, s^{0:t-1}, s^t) \sum_{\phi_m^t\in\Phi} p(\phi_m^t|g^t, s^t)\log\frac{p(\phi_m^t|g^t, s^t)}{p(\phi_m^t|s^t)} \tag{6.10}$$

Marginalizing over g^{t-1} the first term on the right hand side of Eq. 6.10 can be expressed as

$$\begin{aligned}\sum_{g^t\in\mathcal{G}} p(g^t|\phi_m^{0:t-1}, s^{0:t-1}, s^t) &= \sum_{g^t\in\mathcal{G}}\sum_{g^{t-1}\in\mathcal{G}} p(g^t, g^{t-1}|\phi_m^{0:t-1}, s^{0:t-1}, s^t)\\ &= \sum_{g^t\in\mathcal{G}}\sum_{g^{t-1}\in\mathcal{G}} p(g^t|g^{t-1}, \phi_m^{0:t-1}, s^{0:t-1}, s^t)p(g^{t-1}|\phi_m^{0:t-1}, s^{0:t-1}, s^t) \qquad (6.11)\end{aligned}$$

and under the assumptions that the goal transition probability only depends on the previous goal, that the state at time t does not have an influence on the belief over g^{t-1} (violates causality), Eq. 6.11 can be simplified as

$$\sum_{g^t \in \mathcal{G}} p(g^t|\phi_m^{0:t-1}, s^{0:t-1}, s^t) = \sum_{g^{t-1} \in \mathcal{G}} p(g^{t-1}|\phi_m^{0:t-1}, s^{0:t-1}) \sum_{g^t \in \mathcal{G}} p(g^t|g^{t-1}). \tag{6.12}$$

By combining Eq. 6.12 with Eq. 6.10 we finally have

$$\begin{aligned} (6.13)\quad & I(\phi_m^t; g^t|\phi_m^{0:t-1}, s^{0:t-1}, s^t) = \\ & \sum_{g^{t-1} \in \mathcal{G}} p(g^{t-1}|\phi_m^{0:t-1}, s^{0:t-1}) \sum_{g^t \in \mathcal{G}} p(g^t|g^{t-1}) \sum_{\phi_m^t \in \Phi} p(\phi_m^t|g^t, s^t) \log \frac{p(\phi_m^t|g^t, s^t)}{p(\phi_m^t|s^t)} \\ & = \sum_{g^{t-1} \in \mathcal{G}} b_g^{t-1} \sum_{g^t \in \mathcal{G}} p(g^t|g^{t-1}) \sum_{\phi_m^t \in \Phi} p(\phi_m^t|g^t, s^t) \log \frac{p(\phi_m^t|g^t, s^t)}{p(\phi_m^t|s^t)}. \end{aligned}$$

Note that the quantity on the right hand side of Eq. 6.13 is the expectation of the Kullback-Leibler divergence between $p(\phi_m^t|g^t, s^t)$ and $p(\phi_m^t|s^t)$ and therefore Eq. 6.13 is equivalent to

$$I(\phi_m^t; g^t|\phi_m^{0:t-1}, s^{0:t-1}, s^t) = \mathbb{E}_{g^{t-1} \sim b_g^{t-1}, g^t \sim p(g^t|g^{t-1})} D_{\mathrm{KL}} \left[p(\phi_m^t|g^t, s^t) | p(\phi_m^t|s^t) \right] \tag{6.14}$$

and can be estimated using Monte Carlo techniques by generating samples according to the generative model in Figure 6.1.

For an arbitrary state $s \in \mathcal{S}$ and a given history of interface actions ($\phi_m^{0:t-1}$) and states ($s^{0:t-1}$) we define the full disambiguation metric $D : \mathcal{S} \rightarrow \mathbb{R}$ as

$$D(s) = I(\phi_m^t; g^t | \phi_m^{0:t-1}, s^{0:t-1}, s) - \lambda \sum_{g^{t-1} \in \mathcal{G}} b_g^{t-1} \cdot \left\| g^{t-1} - \Psi_q(s) \right\| . \tag{6.15}$$

The first term on the right hand side is the conditional mutual information described earlier and the second term can be interpreted as a regularization term with λ being the regularization coefficient. The regularization term helps the optimizer to navigate an ill-defined optimization landscape which can occur if the mutual information term is identical for all states in the optimization domain. In our implementation, $\mathcal{Q}$ is the space of robot position in which case the regularization term $\|g^{t-1} - \Psi_q(s)\|$ is the distance from the robot position q to the goal position g^{t-1}. And hence, the optimizer favors disambiguating states that are closer to the goal region. Note that $D(s)$ is a conditional metric that depends on the history of interface actions and states.

The maximally disambiguating state s^* is the optimizer of Eq.6.15 and is given by

$$s^* = \underset{s \in \mathcal{S}}{\operatorname{argmax}}\, D(s) \tag{6.16}$$

Note that, the transition from the current state s^{t-1} to s^* could be (a) only a mode switch, (b) only a change in robot location, or (c) a combination of a mode switch and a change in robot location.

Simulation-based Validation of $D(s)$: We compared how well $D(s)$ was able to match an intuitive ground truth for what a maximally good disambiguating state should be. For a local neighborhood of each $s \in \mathcal{S}$, we computed $s*$ using Eq. 6.16 and compared it to a ground truth that was computed by picking the state in the same neighborhood that provided maximum expected difference between first and second maxima of b_g^t over a single time step. We ran a total of 100 simulations for $|\mathcal{G}|$ ranging from 3 to 30 (in steps of 3), and $|\mathcal{S}| = 200$ and the match percentage was 100%.

6.5. Shared Control via Contextual Nudges

We propose an adaptive assistance strategy for the autonomous agent. During its turn (a) if the autonomous agent is not confident, it computes s^* in a local neighborhood of the current state and nudges the robot to s^* and (b) if confident in its prediction of the human intent it transitions the robot towards the predicted goal. Within a shared-control system, at any time t after having observed $\phi_m^{0:t-1}$ and $s^{0:t-1}$, the autonomous agent could choose to nudge the robot into s^*. The autonomous agent can use any kind of autonomous controller to generate an autonomous control command, denoted as u_a^t to accomplish robot motion from s^{t-1} to s^* during its turn. Subsequent actions (ϕ_m^t) executed by the human from s^* will extract maximum information regarding g^t and by doing so, the agent implicitly *helps itself* to provide accurate assistance in the future. A threshold on intent prediction confidence (as measured by entropy of the belief distribution over goals) determines whether to follow the first or the second strategy at each time step. With this strategy, during its turn, the autonomous agent can continue to contribute to task progress, in addition to providing control-blending based assistance during the human's turn. Robot motion and mode switches executed by the autonomous agent during its turn can also provide valuable information *to* the human about the assistance strategies used by the autonomous agent, therefore potentially improving transparency and cooperation.

Algorithm 4 Turn-Taking Interaction with Active Intent Disambiguation

```
Require: $t = 0$
 1: if human-turn then
 2:     $a_h^t \sim p(a|s^t, g^t)$                            ▷ task-level action [human]
 3:     $\phi_i^t \sim p(\phi_i^t|a^t)$                        ▷ intended interface action [human]
 4:     $\phi_m^t \sim p(\phi_m^t|\phi_i^t)$                   ▷ measured interface action [human]
 5:     $u_h^t = \Upsilon(\phi_m^t)$                           ▷ control command corresponding to $\phi_m^t$
 6:     Update $b_g^t$ using Eq. 6.4                            ▷ Bayesian belief update
 7:     $u_a^t = \Xi(\text{argmax}_g\, b_g^t)$                  ▷ control signal to achieve inferred goal, $g'$
 8:     $u_f^t = \alpha \cdot u_a^t + (1 - \alpha) \cdot u_h^t$  ▷ shared autonomy via control blending
 9:     $s^{t+1} \sim \mathcal{T}^u(s^t, u_f^t)$               ▷ state transition using blended control signal
10:     $t = t + 1$
11: if autonomy-turn then
12:     if $H(b_g^t) > \kappa$ then              ▷ if not confident, nudge into disambiguating state
13:         Compute $s^*$ using Eq. 6.16
14:         $s^{t+1} = s^*$
15:     else                                       ▷ if confident, nudge towards inferred goal
16:         $s^{t+1} = (\beta \cdot g' + (1 - \beta) \cdot \Psi_q(s^t), \Psi_m(s^t))$
```

The turn-taking based interaction protocol is outlined in Algorithm 4. Task execution begins with the human. Lines 2-4 show how a human generates task-level actions (a_h^t) and utilizes a control interface to generate interface-level actions (ϕ_t^t and ϕ_m^t) which then get converted to low-level robot control commands (u_h^t) via a transformation function denoted as Υ (Line 5). The autonomous agent utilizes a goal-dependent policy (Ξ) to generate the autonomous command denoted as u_a^t (Line 7). Blending based shared control is available during the human's turn with the blending factor α determined by the autonomous agent's confidence in its prediction of human intent (Line 8). The blending factor α is a strictly non-decreasing piecewise linear function of the probability $p(g')$ associated with the inferred goal g' and is given by

$$\alpha = \begin{cases} 0 & p(g') \leq \rho_1 \\ \frac{\rho_3(p(g')-\rho_1)}{\rho_2-\rho_1} & \text{if} \quad \rho_1 < p(g') \leq \rho_2 \\ \rho_3 & \rho_2 < p(g') \end{cases}$$

with $\rho_i \in [0,1] \;\forall\; i \in [1,2,3]$ and $\rho_2 > \rho_1$. In our implementation, we empirically set $\rho_1 = \frac{1.1}{n_g}, \rho_2 = \frac{1.2}{n_g}$ and $\rho_3 = 0.8$. Note that, higher confidence in prediction results in higher values of α. The arbitrated command, u_f^t then results in robot state transition according to a transition function $\mathcal{T}^u$. During the autonomous agent's turn (Line 11) depending on the confidence of its prediction of intended goal (as measured by the entropy of the belief distribution) the autonomous agent does one of two things, (a) if prediction uncertainty is high (Line 12), then the agent nudges the robot to a maximally disambiguating state (s^*) in the local neighborhood of the current state (Lines 13-14), (b) if prediction uncertainty is low, then the agent moves the robot along the direction from the current location ($\Psi_q(s^t)$) to the inferred goal location by a distance determined by β, leaving the active mode unchanged ($\Psi_m(s^t)$) in Line 16. After the autonomous agent successfully nudges the robot to the target state, the turn is handed back to the human and the task execution continues until goal is achieved.

6.6. Illustration of the Intuitiveness of $D(s)$

In this section, we provide an illustration of how the proposed disambiguation metric matches our intuitions of what the maximally disambiguating states are for different priors over goals in a simple environment. The simulated environment under consideration is a 2D gridworld with three goal locations in which the robot state space $\mathcal{Q}$ is the (x, y)

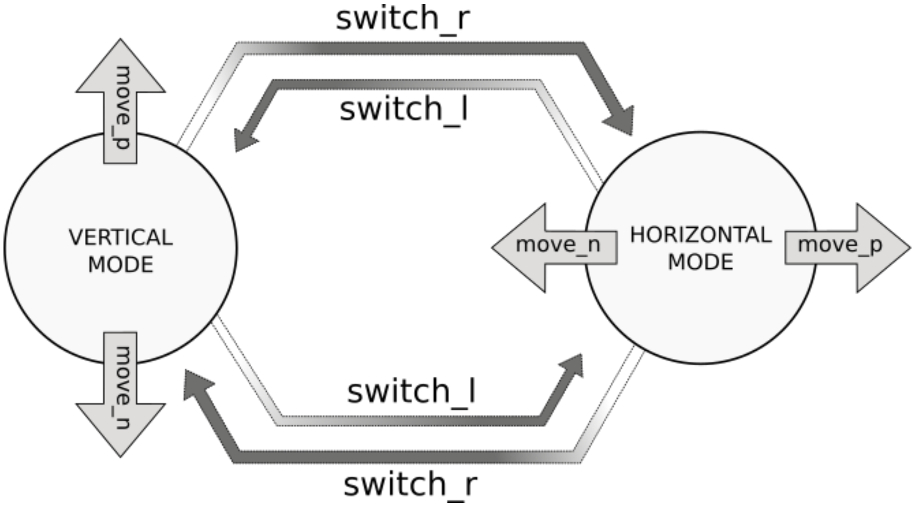


Figure 6.2. Mode switching diagram for 2D grid world with a 1D sip-and-puff interface using a bi-directional mode switching paradigm.

coordinate of each grid cell (Figure 6.5). The simulated teleoperation interface is a 1D interface with bi-directional mode-switching capabilities. [2]

Since the dimensionality of the interface is lower than the dimensionality of the robot the control space is partitioned into two modes: specifically, $\mathcal{M} = \{\textit{Horizontal}, \textit{Vertical}\}$, allowing for motion along x and y dimensions respectively. The action space $\mathcal{A}$ consist of $\mathcal{A}_c = \{\textit{move-positive}, \textit{move-negative}\}$ that consists of actions that allow positive and negative motion along the active dimension at any given time and $\mathcal{A}_q = \{\textit{switch-right}, \textit{switch-left}\}$ which consists of mode-switch actions that result in mode switches. Note that for an ordered set $\mathcal{M}$, *switch-right* results in a transition from mode m_i to m_{i+1} with wrap around to the first element of the set and *switch-left* results in a mode transition from m_i to m_{i-1} with wraparound to the last element of the ordered set. In this scenario

[2] Note that we are just exploring one possible configuration of goals in this illustrative example. The proposed metric is applicable for *any* configuration as well as number of goals.

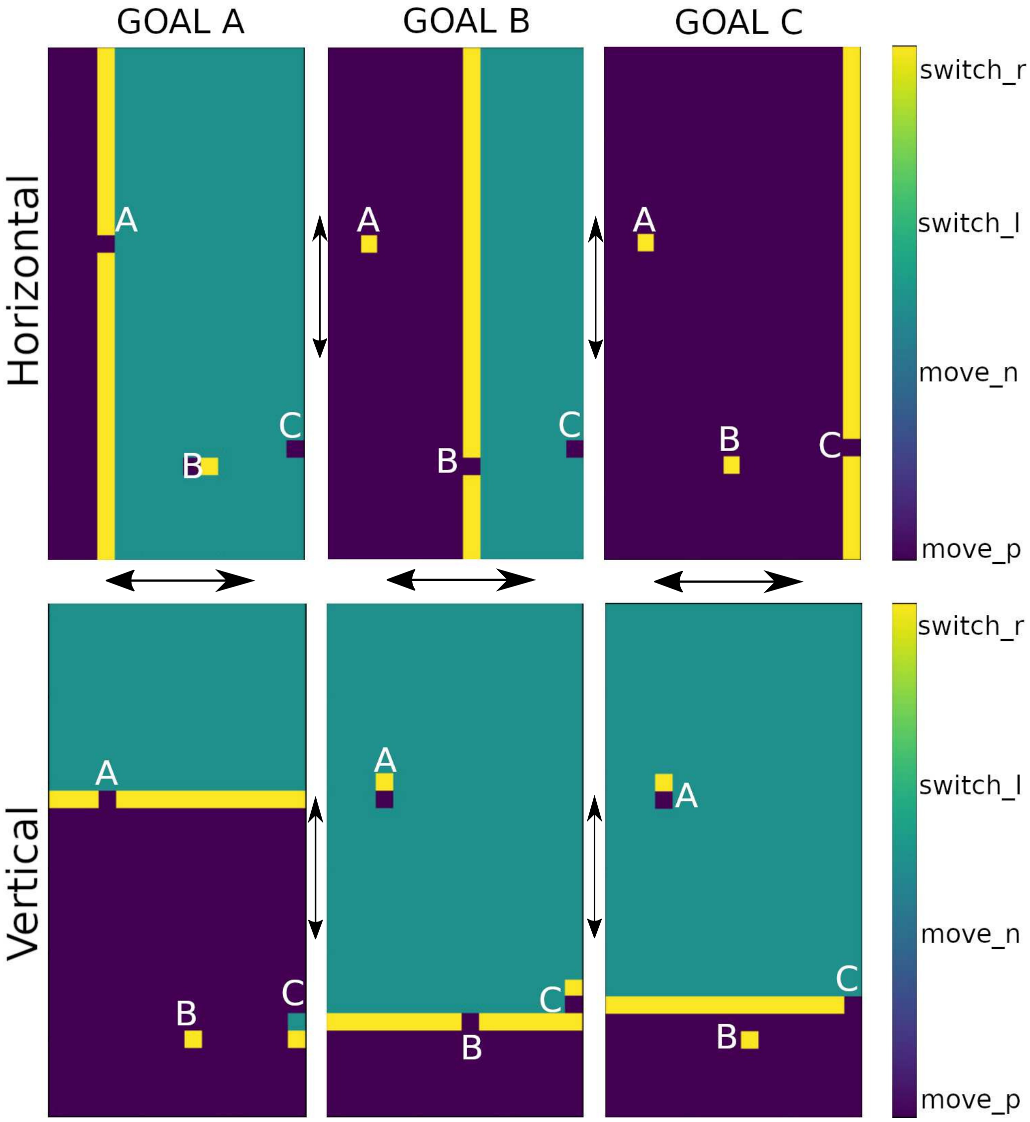


Figure 6.3. Learned policy using value iteration for the each of the goals in the goal configuration shown in Figure 6.5.

with just two control modes a mode switch in one mode simply results in a transition to the other mode (Figure 6.2).

We model the simulated human as an MDP described in Section 6.4.1 and use value iteration [**154**] to obtain a goal-dependent policy, $p(a|s,g)$. The reward function $\mathcal{R}$ is

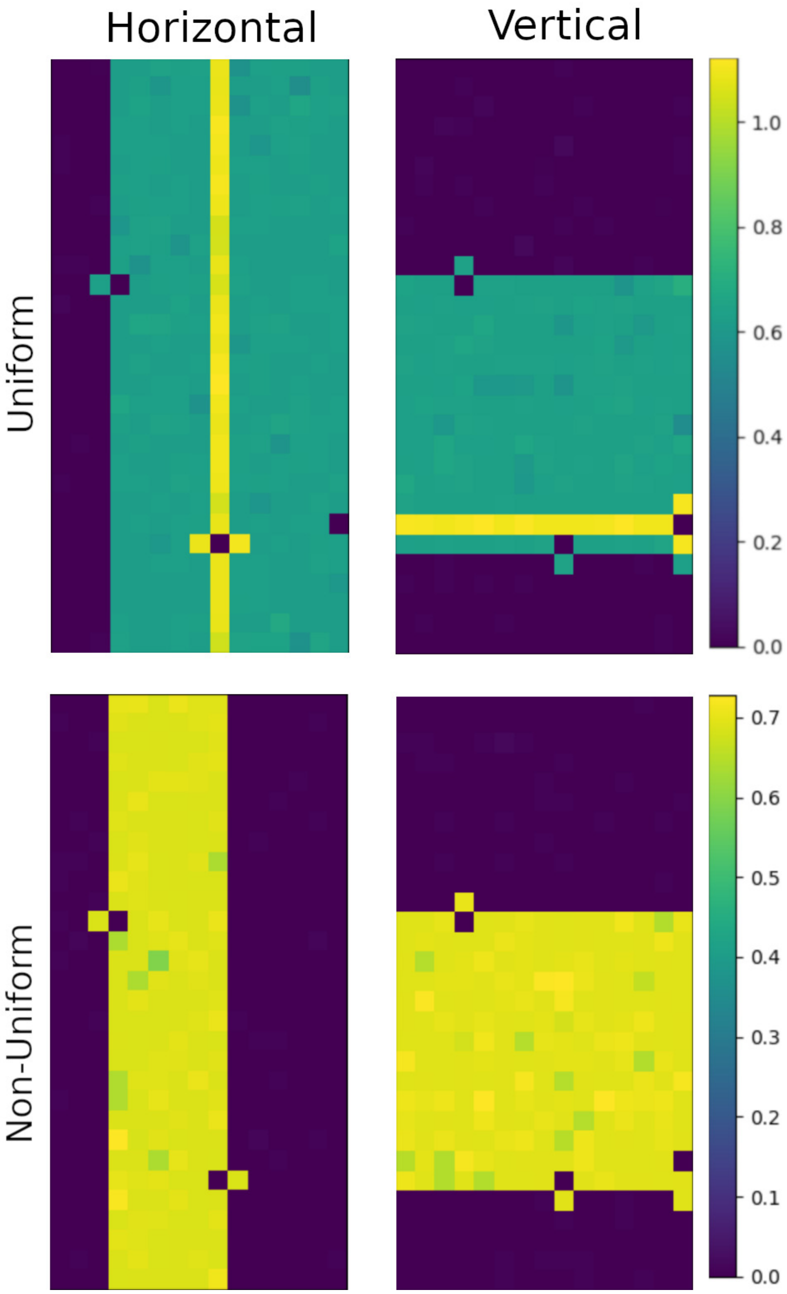


Figure 6.4. Top Row: $D(s)$ computed for all states in the grid world for uniform prior for each mode. Bottom Row: $D(s)$ computed for all states in the grid world for nonuniform prior for each mode. Note that λ was set to be 0.0 to highlight the contribution from the mutual information term.

designed to minimize the distance travelled (according to Manhattan distance) as well as the number of mode switches executed. The optimal policy shown in Figure 6.3 is obtained using value iteration. Upon inspection, we can see that following the policy from any states results in straight line paths and minimal number of mode switches. The top row of Figure 6.4 shows the disambiguation metric computation for all states (visualized for each mode separately) for a uniform prior over goals and $\lambda = 0$.

Within the bounds of the goal region, we see that the maximally disambiguating states for the *Horizontal* mode correspond to the states that are aligned with goal B along the y dimension; which meets our sanity check as any movement left in these states would suggest goal A, any movement right would suggest goal C and a mode switch to vertical motion would suggest goal C. Similarly, the maximally disambiguating states for the *Vertical* mode are states that are aligned with goal C along the x dimension and in this case we once again see that any movement up suggests goal A, any movement down suggests goal B and a mode switch to horizontal motion would suggest goal C. Since no two goals have the same action mapped to them, they allow for maximal goal disambiguation.

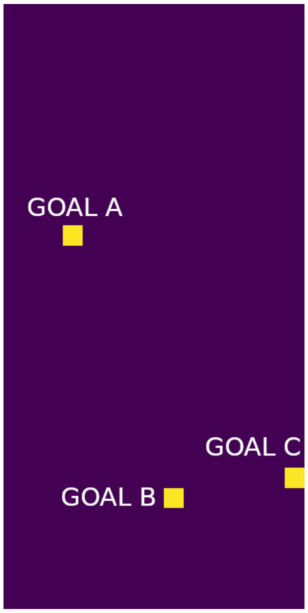


Figure 6.5. Goal configuration used for simulation using a 15 × 30 2D grid.

We also see that in Figure 6.4 (top and bottom) there are states in which the disambiguation metric is identically equal to zero (dark blue color). Under the assumption that the human approximately behaves like the MDP policy shown in Figure 6.3, we can see that for any states outside of the horizontal and the vertical limits of the goal region, there is a high likelihood that the human would choose the *same* action in those states regardless of the intended goal. As the actions are *indistinguishable*, an observer will not be able to determine (from executed actions alone) the true intended goal with certainty.

In the bottom row of Figure 6.4, the priors are non-uniform with the probability associated with goal C set to be zero. This simulates a scenario in which the history of states and actions have already caused the Bayesian update of belief to assign zero probability to goal C. Effectively, it becomes a question of disambiguating between goals A and B. The disambiguation metric computed for this scenario correctly ignores goal C completely.

6.7. Experimental Design

Each study session consisted of four phases: Phase 1: Training and data collection to model $p\big(\phi_i^t|a^t\big)$. Phase 2: Training and data collection to model $p\big(\phi_m^t|\phi_i^t\big)$. Phase 3: Familiarization with teleoperation, control blending and turn-taking based interaction protocol. Phase 4: Algorithm evaluation. In Phase 4, the subjects performed a navigation task using a 3D point robot using the sip-and-puff interface towards pre-defined goals under two experimental conditions. In total, we collected 432 trials (216 per evaluation condition). We conducted a human subject study ($n = 9$) to evaluate our turn-taking based interaction protocol that deploys the disambiguation algorithm. All participants gave their informed, signed consent to participate in the experiment which was approved by Northwestern University's Institutional Review Board.

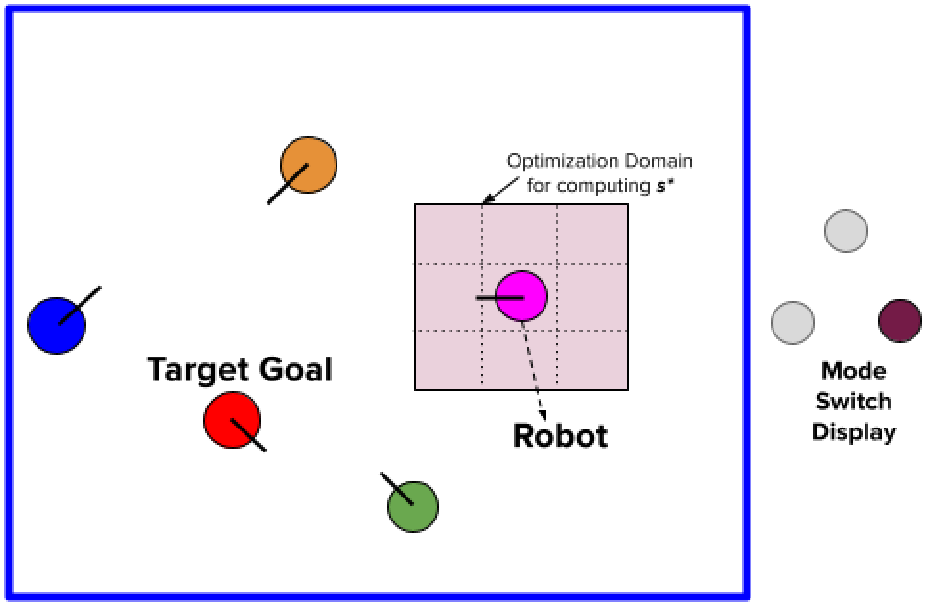


Figure 6.6. Simulated navigation environment. Both clockwise as well as counter-clockwise mode switches are possible. The mode switch display highlights the current active mode. The shaded 3x3 grid around the robot denotes the optimization domain for computing the maximally disambiguating state s^*.

6.7.1. Experimental Setup

We designed a simulated navigation environment (Figure 6.6) in which subjects operate a 1D sip-and-puff (SNP) device to (a) control a 3 Degrees-of-Freedom (DoF) point robot's motion along two translational (x, y), and one rotational (θ) dimensions, one at a time towards a pre-defined goal, g_{true} (shown in red) and (b) to perform mode switches. We opted for SNP, as it is one of the most information limited interfaces used by people with severe motor impairments. For this environment, $\mathcal{M} = \{Horizontal, Vertical, Rotational\}$. In order to compute s^* the continuous 3D robot state space was discretized into a 10×10×8 grid that represented 10×10 x-y grid locations and eight discrete orientations for each cell.

Note that this discretization was for the computation of $D(s)$ only; the robot positions, velocities, and goal positions were all continuous valued.

In order to facilitate seamless turn-taking between the subject and the autonomous agent, a text display presented the subject with information regarding the state of interaction and the environment's boundary transitions from blue to red over a fixed time period ($\sim$3-4s) to remind the subjects that they should hand over the turn. At the beginning of the subject's turn they were allowed to wait hence providing time for planning, or simply rest. Handover to the autonomous agent was triggered by not issuing any commands for $\sim$1.5-2s *whenever the subject deems it is appropriate*. During the subject's turn, a linear control blending based assistance was present. The autonomous control policy was generated using the algorithm developed by Huber et al., in [**77**] and operated in the full 3D space.

6.7.2. Training protocol

Learning Personalized Distributions: Data collection for learning the model for the interface operation (estimating the distributions $p(\phi_m|\phi_i)$ and $p(\phi_i|a)$ was done according to the procedures described in Section 4.4.2.

Familiarization with environment and robot control: Participants first were trained on the physical mechanism of operating the interface. Subsequently, they became familiarized with the environment and gained practice in both robot teleoperation as well as in interacting with the autonomous agent via control blending during the turn-taking process.

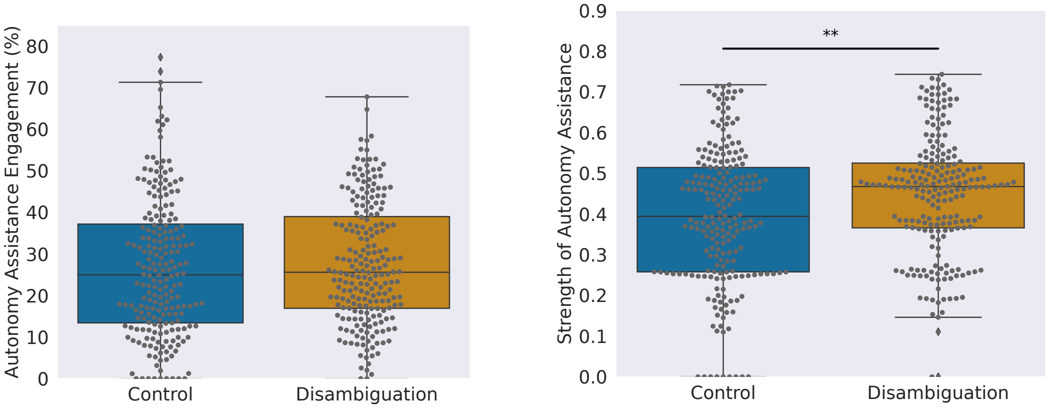
6.7.3. Algorithm Evaluation

In the evaluation task, the subject controlled the motion of a 3-DoF point robot to reach a 3-D goal (Figure 6.6). The number of the goals varied from three to four. For each trial, the starting position of the robot was randomized and diametrically opposite from the goal region. A trial always started and ended with the subject's turn. Subjects performed the evaluation task under two conditions: (a) *Disambiguation* and (b) *Control*.

Disambiguation Condition: During the autonomous agent's turn, the procedure outlined in Lines 12-16 of Algorithm 2 was activated with the constraint that the optimization domain for computing s^* was a local neighborhood grid of size 3×3 centered around the state s^t at the beginning of the autonomous agent's turn. λ was set to be 1.0.

Control Condition: During the autonomous agent's turn, the robot was nudged towards the goal with the highest probability by issuing autonomous control commands. If multiple goals were tied for highest probability, then the mean of all those tied goals was the target of the nudge. The distance nudged was sampled randomly from within the fixed size local neighborhood used in the *Disambiguation* condition.

A trial was deemed successful if the robot's pose coincided with the red goal (both position and orientation) within a predefined threshold. Subjects performed six blocks of eight trials each. After each block, the subjects were asked to respond to a NASA-TLX questionnaire and a post-task survey in which they were queried about their subjective evaluation of how well the autonomous agent was able to assist them during the task. We further evaluated the effectiveness of the disambiguation algorithm according to the following metrics.



(a)

(b)

Figure 6.7. (a) Percentage of time that autonomy assistance is engaged during a trial. (b) Strength of autonomy assistance (as measured by the blending factor α) during a trial. Box plots show median and quartiles. The black dots represent the individual data points.

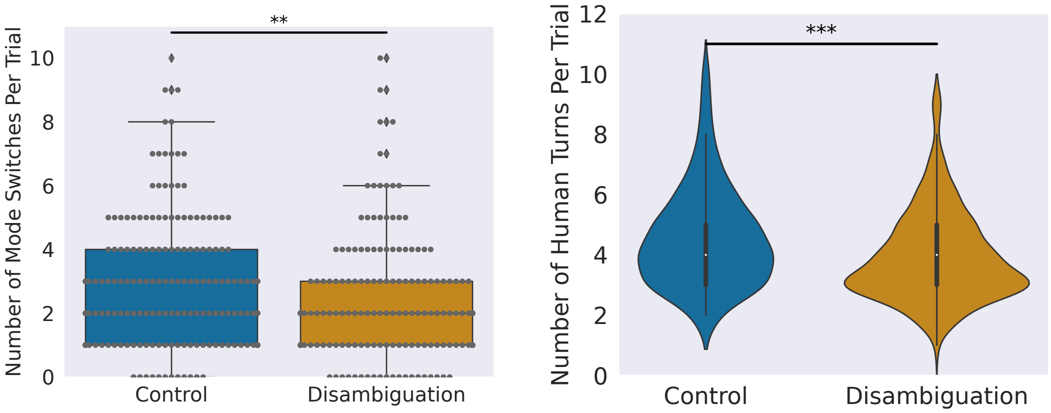
Assistance Engagement: Fraction of time the autonomous agent activates assistance towards the true goal ($\alpha > 0$ with $g' = g_{red}$) during the human's turn in a trial.

Strength of Assistance: Average value of the blending factor α over all time steps when both goal inference is correct and blending assistance is active.

Number of Mode Switches: The number of human-initiated mode switches; a significant contributor to the cognitive and physical effort required for task execution.

Human Effort: Fraction of total trial time that subjects spent operating the robot in order to accomplish the task.

Task Completion Time: Total time taken to complete the task successfully.



(a)

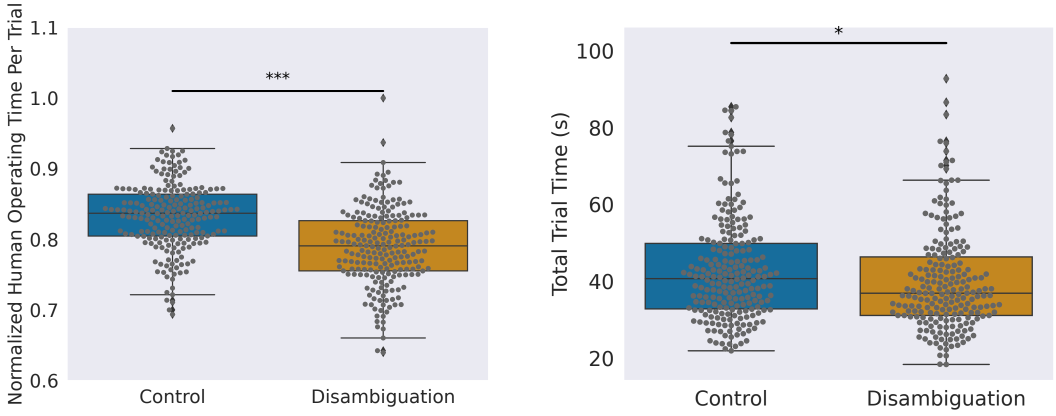
(b)

Figure 6.8. (a) Number of mode switches executed by the subject during a trial. (b) Number of human turns per trial.

6.8. Results

We analyze group performances using the non-parametric Kruskal-Wallis test and perform the Conover's test post-hoc pairwise comparisons to find the strength of significance. For all figures, $* : p < 0.05$, $** : p < 0.01$, and $*** : p < 0.001$.

Figure 6.7a shows the fraction of time the autonomous agent activates blending assistance towards the correct goal during the human's turn. For any time t if $\alpha > 0$ and $g' = g_{true}$ then assistance is activated. We do not observe any statistically significant difference between the two experimental conditions. However, in Figure 6.7b the strength of assistance offered by the autonomous agent towards the correct goal as measured by the average value of the blending factor α over a trial, is higher for *Disambiguation* compared to the *Control* condition ($p < 0.01$). Since α is an non-decreasing function of the probability associated, higher values of $alpha$ implies that $p(g')$ is higher as well. This indicates that in the *Disambiguation* condition, autonomous agent is able to be more confident in its



(a)

(b)

Figure 6.9. (a) Fraction of total trial time that subjects spent operating the robot. (b) Total task completion time per trial.

prediction of true goal and therefore provide stronger assistance (greater control contribution by the autonomous agent to the overall control signal) without overtly engaging with the human for more time. Figure 6.10 illustrates how the goal probability associated with the true goal (in red) increased rapidly during the human's turn immediately following the autonomous agent's turn during which it nudged the robot into s^*. Additionally, the overall task success for *Disambiguation* condition (93.05%) was higher than the *Control* condition (90.2%).

A statistically significant decrease in the number of mode switches is observed between the *Disambiguation* and *Control* conditions as seen in Figure 6.8a. In Figure 6.8b We also observe that subjects perform a fewer number of turns under the *Disambiguation* condition as compared to the *Control* condition. In Figure 6.9a we also observe a statistically significant decrease in the amount of time the subjects spent operating the robot in the *Disambiguation* condition, measured as a fraction of total trial time.Overall, task completion time is also lower under the *Disambiguation* condition (Figure 6.9b).

Fewer mode switches, fewer number of turns, and faster trial times likely correlate with less human effort. In the *Disambiguation* condition, the subject is able to execute actions that are maximally informative about the true goals because the autonomous agent nudges the robot into maximally disambiguating states. Task completion time is shorter because the autonomous agent is able to infer human intent with more confidence and therefore provide stronger and more accurate assistance towards the true goal. Manual mode switches become unnecessary once control blending is activated by the autonomous agent, the robot is also able to move in all three dimensions simultaneously.

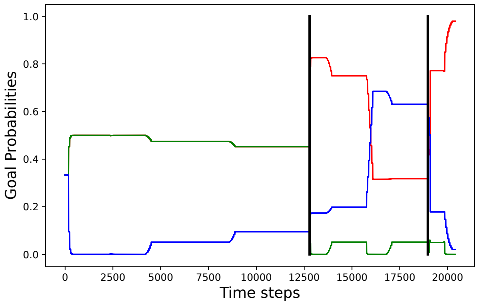


Figure 6.10. Evolution of goal probabilities for a *Disambiguation* trial. The black vertical line indicates the start of the human turn after the autonomous agent nudged the robot into s^* during in the previous turn. Note that at the start of the trial the probabilities associated with the green and red goals coincide.

6.8.1. Subjective Task Metrics

We use the raw NASA-TLX as a subjective measure of perceived workload [**70**]. Although the mean score for the *Disambiguation* condition (32.75) is slightly lower than the *Control* (34.45) condition, we do not observe a statistically significant difference. We evaluate user preferences and acceptance of our shared-control assistive paradigms using a questionnaire (Figure 6.11). The statements are rated on a 7-point Likert scale from strongly disagree

(1) to strongly agree (7). Overall, the subjects rate the *Disambiguation* condition higher than the *Control* condition when it comes to the agent's ability to *figure out* the human's intended goal faster. However, subjects think that it is in the *Control* condition that the autonomous agent is more effective in helping them *move* towards the desired goal.

6.9. Discussion

The computation of the disambiguation metric critically depends on whether it can be empirically estimated from a generative model. Our results indicate that despite having an approximate model for human behavior (one in which the human is assumed to minimize path distance to goal and the number of mode switches), the autonomous agent was able to leverage the model and successfully compute reasonably good disambiguating states. Performance would likely improve further if more accurate models of human behavior—learned from large amounts of data using state-of-the art machine learning techniques—are used in conjunction with the proposed disambiguation algorithm.

Although turn-taking allows the user to observe the autonomous agent's actions and acquire a mental model of the autonomous agent's policy, sufficient training and priming is important so that the subject's expectation of the autonomous agent's policy is close to the true policy. With more training and practice, the human-autonomy team can achieve common ground faster and the human will be incentivized to work in a cooperative manner and *leverage* the assistance offered by the autonomous agent.

In Eq. 6.16, the domain over which the optimization occurs can vary depending on whether the autonomous agent wants to move the robot in a small neighborhood from the

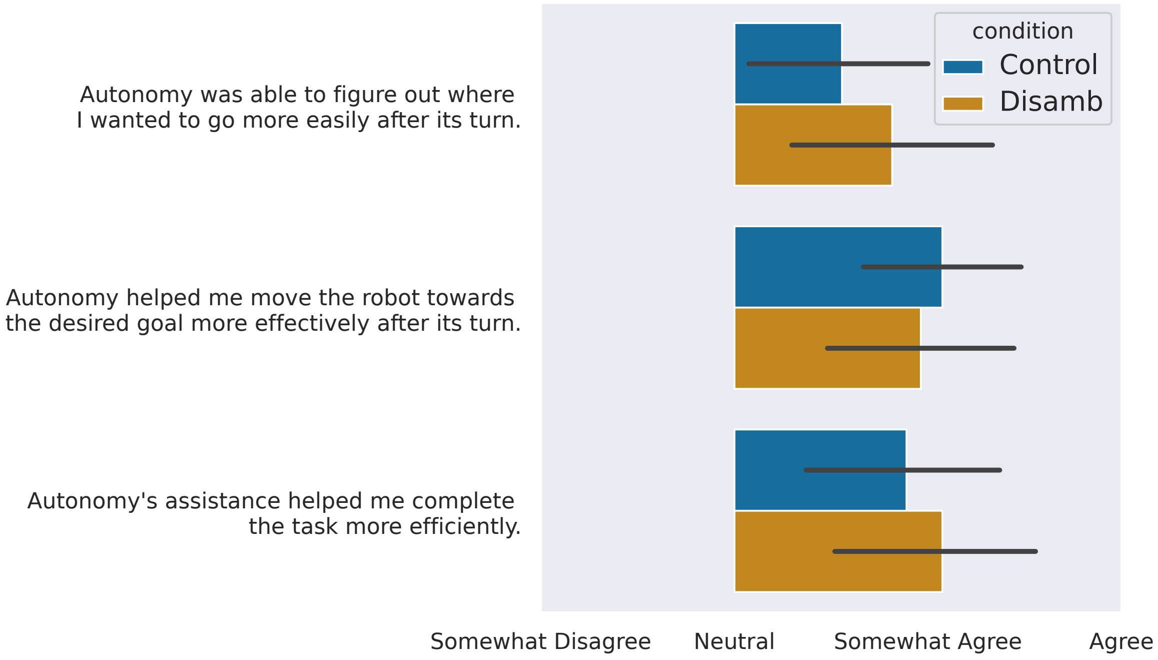


Figure 6.11. Average user response to post-task survey questions. The bars indicate standard deviation.

current position or whether it plans to execute large scale motions. In the small neighborhood condition, the idea is that the autonomous agent's actions would be interpreted by the user as small nudges as opposed to a complete takeover of user control. Disambiguation over the entire state space might, however, reduce human effort significantly by reducing the overall number of turns and also the fraction of total time the human operates the device. It might be beneficial to allow the user to pick the optimization domain depending on their preference. Yet another algorithmic modification would be to reason over larger time horizons; but this would come at a higher computational cost.

Prior task structure can be leveraged to further simplify the computation of the disambiguation metric. For example, in an assistive robotic arm, although the full task space is six dimensional, prior constraints on task execution (the subject reaches for an object before grasping it) essentially will help to accomplish disambiguation in the three dimensional translation space. Disambiguation is especially useful during the earlier parts of task execution when the uncertainty in inference is the highest; and in a typical manipulation task, moving in translation space to perform a reaching motion happens earlier.

6.10. Conclusions

This chapter has presented a novel interface-aware intent disambiguation algorithm grounded in information-theoretic principles. The primary goal of this algorithm was to elicit maximally informative control signals from the user by placing them in states that have the highest disambiguation capabilities as determined by the metric. The chapter also introduced a turn-taking based human-autonomy interaction protocol in which the autonomous agent utilized the proposed disambiguation metric to extract information rich actions from the human when uncertain about its prediction of human intent. The efficacy of both the proposed algorithm and protocol was evaluated via a nine person human subject study. The results indicated that the disambiguation system resulted in a statistically significant decrease in task effort in terms of the number of manual mode switches executed and the fraction of time the users spent operating the robot.

CHAPTER 7

Human-in-the-loop Optimization for Shared Autonomy

This chapter addresses the question of leveraging the human in the loop of a human-robot team to optimize the parameters of the control arbitration scheme in a shared autonomy setting. The key contribution of this chapter is the mathematical formalization of user-driven optimization of shared autonomy in assistive robotics as a nonlinear optimization problem, in which the optimizer is the *human in the loop.* By framing the problem in such terms, we leave the exact nature of the user's reward function unspecified. The results of a 17-person subject study reveal that all subjects were able to converge to a valid setting of the assistance parameters, thereby suggesting the existence of individualized optimal solutions. Most interestingly, the amount of assistance that users optimized for did not always correspond to optimal *objective* task performance as measured by task completion time and number of mode switches performed. The work presented in this chapter was done in collaboration between Deepak Gopinath and Siddarth Jain. [1]

[1] The individual contribution breakdown is as follows: 1) Gopinath developed the mathematical framework and formalism, ported and implemented the SEDs algorithm for LfD and did the code development for the shared autonomy pipeline, was the primary proctor for the human-subject study, analyzed the data/generated the plots. 2) Jain assisted with subject recruitment, proctoring the experiment, LfD data collection, assisted with figure generation and writing the related works section of the published manuscript. Additionally, many thanks to Jessica Presperin Pedersen, OTR/L, ATP/SMS, for recruiting the SCI subjects, and to Samuel Schlesinger for assistance during the subject study.

7.1. Introduction

For people with severe motor impairments as a result of spinal cord or brain injuries, assistive machines such as robotic arms or powered wheelchairs are crucial for reducing their dependence on caretakers and increasing the ability to perform activities of daily living. Since users differ in their physical abilities and desired amount of assistance, *customization* of the amount of assistance is critical for successful adoption of assistive shared control systems. A blanket solution that targets the average end user will likely remain suboptimal in the individual case and therefore *user-specific optimization* of the parameters that influence the overall human autonomy interaction is an important area of investigation.

To start off, pre-defined assistance levels (as defined by different parameter presets) can be good starting points, but may not remain optimal for the user in the long term. For example, the subjects' motor abilities will likely change—either degrade (e.g., due to degenerative disease) or improve (e.g., due to successful rehabilitation). As a result, the need for assistance may increase or decrease. One way to accomplish customization is to tune the system parameters which will bring about a change in the final human-robot team behavior. A straightforward choice of optimality criterion is to consider task-related performance metrics such as task completion time and effort. Such metrics, however, may not capture user-related metrics like comfort, preference, or satisfaction.

Our insight is that if we entrust the responsibility of customization to the users, they are given the opportunity to tune the system in such a way that the optimal interaction—according to their personal optimality criterion—will emerge. Moreover, the user-driven customization of assistance can be user-dependent in addition to being task-dependent.

By doing so, we also *empower* the user with the ability to directly influence the overall experience of interacting with the autonomous agent, which can potentially help to improve personal satisfaction and self-esteem.

To ground our formalism, in this chapter we present a first implementation of a customizable blending-based shared control system, in which the reasoning between the user control and the autonomous agent's control is a parameterized function of autonomous agent's confidence in its inference of human intent. Our interactive user-driven customization system maps verbal cues from the human to changes in these parameter values.

The contributions of this chapter are three-fold:

- We ground the problem of user-driven customization of shared control arbitration parameters as a non-linear policy optimization problem in which reasoning between the user control and the autonomous agent's actions is a tunable function of confidence in the autonomous agent's ability to infer human intent.
- We present a lightweight interactive user-driven customization system that maps verbal cues from the end user to progressively fine-grained discrete adjustments of the parameter values.
- Lastly, we also present results from a 17-person (uninjured and spinal cord injured) subjects study that evaluated the customization system.

7.2. Proposed Framework

Principles from optimal control theory have been successfully used to account for different aspects of human motor control such as arm trajectory formation, posture control and locomotion [**61, 162, 167**]. The underlying motivation in using optimal control theory is that biological systems have evolved to produce motor commands that optimize motor behavior with respect to the task at hand [**162**]. The extension of this reasoning is that the optimizing principles are operating over control commands to the robot effector rather than motor commands to the human muscles, when a human operates an assistive robot to replace their lost motor function,

We frame the customization formalism within the language of policy optimization (or equivalently, optimal control theory) not only because of this biological parallel, but also because it allows for the analysis of the effects of the various design decisions and the components of a shared control system in a rigorous manner.

7.2.1. Mathematical Formalism

Let $\boldsymbol{q}^t$ be the robot state at time t. The low-level control command issued by the human to the robot is denoted as $\boldsymbol{u}_h^t$ and by the autonomous agent according to a predefined set of autonomous policies is denoted as $\boldsymbol{u}_a^t$.[2] Let $\boldsymbol{\theta}^t$ be the set of tunable arbitration parameters that will affect the manner in which control is shared between the human and the autonomous agent and consequently the final control command $\boldsymbol{u}_f^t$ that results in robot motion.

At any time t, the control signal from the autonomous agent is generated according to goal-dependent policy $f_g(\cdot) \in \mathcal{F}_g$,

$$\boldsymbol{u}_a^t \leftarrow f_{g^*}(\boldsymbol{q}^t) \tag{7.1}$$

[2] In this chapter, we ignore interface-level effects altogether and directly reason in the space of low-level control commands.

where $\mathcal{F}_g$ is the set of all control behaviors corresponding to different goals (denoted as g), and g^* is the user's inferred goal at time t. Numerous options exist for class of policy generation functions; in this work, we employ a Learning from Demonstration approach described in detail in Section 7.2.7.

In the previous chapters, we saw that we can model the human as an Markov Decision Process (MDP) optimizing a reward function defined in the task-level state and action space. The policy is defined as mapping from states to distributions over task-level actions. These task-level actions are transformed into interface-level actions and then into low-level control commands via deterministic mapping functions. In this chapter, we ignore all the intermediate transformations and directly reason about human intent in the space of low-level control commands. Most importantly, in this chapter, although we model the human as optimizing an interface dependent goal-directed MDP, we leave the reward function unspecified and hence the autonomous agent does *not* rely on an *a priori* rational model of human behavior.

The shared control system makes use of an arbitration function $\boldsymbol{\beta}(\cdot)$, parameterized by $\boldsymbol{\theta}^t$

$$\boldsymbol{u}_f^t \leftarrow \boldsymbol{\beta}_{\boldsymbol{\theta}^t}(\boldsymbol{u}_h^t, \boldsymbol{u}_a^t) \tag{7.2}$$

to arbitrate between the control commands from the human and the autonomous agent's policy to produce the final control command $\boldsymbol{u}_f^t$ executed by the robot.

A key insight in our formulation is that, for a time–varying arbitration function $\boldsymbol{\beta}(\cdot)$, the parameters $\boldsymbol{\theta}$ themselves can be functions of time and therefore may be *interpreted* *as control signals.* The transition function for the world state can be written as

$$\boldsymbol{q}^{t+1} = \mathcal{T}^u(\boldsymbol{q}^t, \boldsymbol{\theta}^t; \boldsymbol{u}_h^t, \boldsymbol{u}_a^t) \tag{7.3}$$

where $\mathcal{T}^u$ is, in general, a nonlinear function. In Equation 7.3, $\boldsymbol{u}_h^t, \boldsymbol{u}_a^t$ are treated as given and $\boldsymbol{\theta}^t$ is treated as a *control signal.* That is, by altering $\boldsymbol{\theta}^t$ the resulting state at the next time step would be different. The problem of finding the optimal $\boldsymbol{\theta}^t$ that will generate the optimal human-robot interaction experience for the user and task performance (according to the reward function) thus may be formulated as a policy optimization problem over $\boldsymbol{\theta}^t$.

Policy optimization approaches assume that the existence of a reward function. In general, let $\mathcal{R}_h$ denote the reward function that the human optimizes. The discounted N-step total return G_h^t can be simply written as

$$G_h^t = \sum_{k=t,k=t+N} \mathcal{R}_h(\boldsymbol{q}^k, \boldsymbol{\theta}^k; \boldsymbol{u}_h^k, \boldsymbol{u}_a^k) \tag{7.4}$$

The true reward function could depend on additional factors such as user satisfaction, motivation *et cetera.* The optimization domain for the tunable parameters $\boldsymbol{\theta}^t$ can be denoted as $\boldsymbol{\Theta}^t$.

7.2.2. User-Driven Policy Optimization

Typically in a policy optimization setting, the optimization is performed over control signals that drive state transitions. That is, the optimization objective is find a policy (a mapping from states to actions) that maximizes the total return. In our scheme, the control commands from the human and the autonomous agent ($\boldsymbol{u}_h^t$ and $\boldsymbol{u}_a^t$) are treated as given quantities and the objective is to optimize the arbitration parameters $\boldsymbol{\theta}^t$.

In this chapter, we, furthermore, make no attempt to determine the exact nature of the reward function $\mathcal{R}_h$. There might be a number of unmeasurable latent factors that could influence the structure of the human's reward function, and determining the exact mathematical form for the reward, likely, is an intractable problem. Making any kind of approximation to simplify the reward function in turn will affect the robustness and efficacy of the assistive system. Since we do not want to reduce the assistive capabilities of our system, and we have a human in the loop, our insight is that *the optimization can be performed by the user themselves, instead of adopting off-the-shelf policy optimization algorithms.* Thus, there is no need to concretely define $\mathcal{R}_h$ for an optimizer; instead the representation of the reward function is maintained by the end user as they optimize the parameters $\boldsymbol{\theta}^t$ until the desired behavior is achieved. In this user-driven customization system, the overall effect of parameter optimization is that of changing the assistance offered by the robot.

The details of the optimization domain and procedure are presented in Sections 7.2.3 and 7.2.4. Robot policy learning is accomplished offline using a Learning from Demonstration algorithm described in Section 7.2.7.

7.2.3. Command Arbitration

In our implementation, the arbitration function $\beta(\cdot)$ that reasons between the human and the autonomous agent control signals is a linear blending function given by,

$$\beta_{\boldsymbol{\theta}}(\boldsymbol{u}_h^t, \boldsymbol{u}_a^t) \triangleq (1 - \alpha_{\boldsymbol{\theta}}) \cdot \boldsymbol{u}_h^t + \alpha_{\boldsymbol{\theta}} \cdot \boldsymbol{u}_a^t \tag{7.5}$$

where $\alpha_{\boldsymbol{\theta}} \in [0, 1]$ is itself a function parameterized by $\boldsymbol{\theta}$.[3] Note that $\alpha_{\boldsymbol{\theta}} = 0$ corresponds

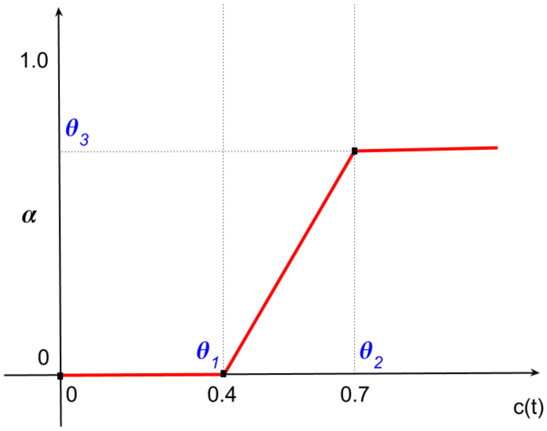


Figure 7.1. A prototypical arbitration function, parameterized by $\boldsymbol{\theta} = \{\theta_1, \theta_2, \theta_3\}$.

to full teleoperation, and $\alpha_{\boldsymbol{\theta}} = 1$ to full autonomy.

The majority of *arbitration functions* $\alpha_{\boldsymbol{\theta}}$ can be reduced to the functional form pictured in Figure 7.1, characterized by a set of three parameters $\{\theta_1, \theta_2, \theta_3\}$ and independent variable c^t[**48**]. Computation of c^t is described in Section 7.2.5. The parameter set determines:

- θ_1: The minimum value of c^t above which control blending is performed.
- θ_2: The value of c^t above which α is maximum and constant.
- θ_3: The maximum value of α for any value of c^t.

Note that $\theta_3 = 0$ corresponds to constant teleoperation (irrespective of the value of c^t). The relationship between c^t and $\alpha_{\boldsymbol{\theta}}$ is linear between θ_1 and θ_2, and the slope of this linear relation determines how *aggressively* the robot assumes control. The parameter bounds are such that $\forall i,\ \theta_i^t \in [0, 1]$ and $\theta_1 \leq \theta_2$. In the study presented in this chapter, the parameters are tuned only between tasks and are unchanged during task execution; that is, $\theta_i^t = \theta_i^{t_0},\ \forall t \in [t_0, t_f]$, where t_0 and t_f denote the start and end of a trial. Essentially, the parameters are *constant* functions of time. These constraints specify the

[3] The time index t is dropped from $\boldsymbol{\theta}^t$ for brevity in notation.

optimization landscape for the parameters and in Section 7.2.4 we will see how the user performs constrained optimization of the arbitration parameters through an interactive procedure. The arbitrated signal $\boldsymbol{u}_f^t$ is the velocity of the end-effector in Cartesian space, which eventually is converted to joint-space velocities via inverse kinematics.

7.2.4. User-Driven Optimization of the Arbitration Parameters

In our interactive user-driven optimization procedure, verbal commands from the user are mapped to systematic changes in $\boldsymbol{\theta}$ by the system operator. Note that, in this implementation we do not focus on how this optimization procedure is facilitated. One could imagine an AI system with natural language processing capabilities replacing the system operator altogether.

A change in assistance level can be achieved by modulating one or more of the $\theta_i \in \boldsymbol{\theta}$, according to $\theta_i = \theta_i \pm \delta\theta_i$. In our implementation, at initialization $\delta\theta_i = 0.1$. The value of $\delta\theta_i$ is adaptive, and is halved if a request to increase assistance is immediately followed by a request to decrease and vice versa (in order to avoid oscillatory behavior). After each optimization step, the user observes the change in the behavior of the shared-control system which then informs their next optimization step. Note that the reward function that is being optimized is internally represented by the user and influences the magnitude and direction of the optimization step.

Table 7.1 provides a few example mappings between common verbal cues, the parameters changed and the values of $\delta\theta$. We chose to modulate more than one parameter at a time as it helps to make the change in assistance level more perceivable to the user.

Verbal Cue	Parameters Changed	Amount of change
“More”	$\theta_3 \uparrow$, $\theta_2 \downarrow$, $\theta_1 \downarrow$	$\delta\theta \leftarrow \delta\theta$
“Less”	$\theta_3 \downarrow$, $\theta_2 \uparrow$, $\theta_1 \downarrow$	$\delta\theta \leftarrow \delta\theta$
“Little More”	$\theta_3 \uparrow$, $\theta_2 \downarrow$, $\theta_1 \uparrow$	$\delta\theta \leftarrow \frac{1}{2}\delta\theta$
“Little Less”	$\theta_3 \downarrow$, $\theta_2 \uparrow$, θ_1 (*no change*)	$\delta\theta \leftarrow \frac{1}{2}\delta\theta$

Table 7.1. Mappings from verbal cues to parameters changed ($\uparrow$ indicates a positive $\delta\theta$ and $\downarrow$ denotes a negative $\delta\theta$)

7.2.5. Intent Inference

In our implementation, the variable c^t that influences the value of $\boldsymbol{\alpha}$ is *the autonomous agent's confidence in its inference of the user's intended goal.* The confidence c^t is computed at each time step that the human provides a control signal, i.e. whenever $\boldsymbol{u}_h^t \neq \emptyset$. In our implementation, c^t is computed as

$$c^t \triangleq w_1(\boldsymbol{u}_h^t \cdot \boldsymbol{u}_a^t) + w_2(\boldsymbol{e}^{-d}) \tag{7.6}$$

where d is the Euclidean distance between the end effector and an inferred target location at time t, and $c^t \in [0, 1]$. The first term in (7.6) provides a measure of agreement or alignment between the user-generated commands and the commands generated from the autnomous policy.[4] The second term encodes the nearness to the target. Parameters w_1 and w_2 are task-specific weights.

At each execution step this confidence measure is computed for all candidate goals in the scene, $g \in \mathcal{G}$, resulting in a distribution of confidences $c_g^t \in \mathcal{C}$ over the candidate goals.[5] To compute these confidences, each control behavior f_g generates a command

[4] Commands $\boldsymbol{u}_h^t$ and $\boldsymbol{u}_a^t$ are first smoothed using a moving average filter ($0.6s$), so that small command changes do not affect the confidence measure drastically.

[5] In the pilot study the candidate goals are objects placed at predefined positions in front of the robot. Our system also is able to autonomously perceive object positions using an onboard perception system and can use these as candidate goals.

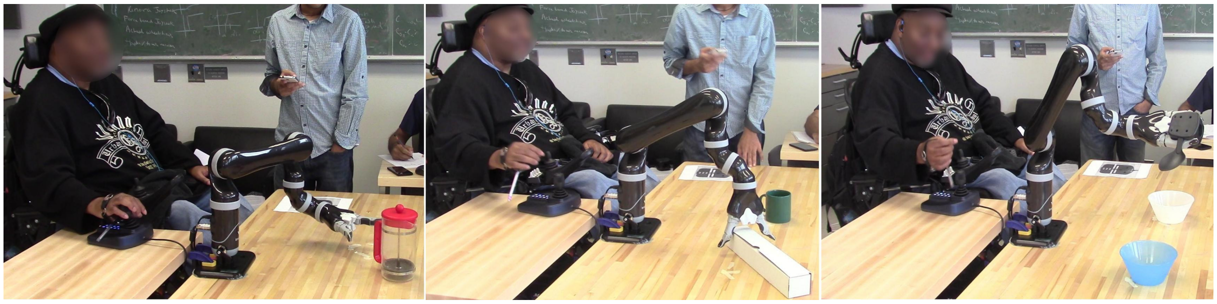


Figure 7.2. Study tasks performed by SCI participant. *Left to right:* Simple Reaching (R), Reaching for Grasping (RfG), Reaching for Scooping (RfS).

(where f_g aims to reach candidate goal g) which is used in the calculation of c_g^t according to (7.6). The candidate goal that has the highest computed confidence is selected as the inferred goal and is denoted as a g^*. That is,

$$g^* = argmax_{g \in \mathcal{G}} c_g^t \tag{7.7}$$

7.2.6. Control Interface and Mapping

The human control command $\boldsymbol{u}_h^t$ is enacted via a 3-axis joystick operated under two different mapping paradigms (Table 7.2). In this work, we treat the interface as noiseless and that the user's intended task-level actions are deterministically mapped to interface-level actions and subsequently to low-level control signals without any deviations. The joystick signals are mapped to the translational and rotational velocities of the end-effector in Cartesian space. The first paradigm uses only two of the three axes (no twist)—because many end users lack the hand function to perform twisting—and accordingly defines four 2D modes to cover the six control dimensions of the robot arm. We refer to this as the *2D mapping paradigm*. The second uses all three of the joystick axes under two 3D modes and is referred to as the *3D mapping paradigm*.

Control Mappings		
Mode	**3D**	**2D**
1	v_x, v_y, v_z	v_x, v_y
2	$\omega_x, \omega_y, \omega_z$	v_x, v_z
3	—	ω_x, ω_y
4	—	ω_z

Table 7.2. Operational paradigms for the teleoperation interface.

7.2.7. Derivation of the Autonomy Policy

The autonomous agent's control command $\boldsymbol{u}_a^t$ is generated from an autonomous control policy. While any number of techniques may be employed to derive the behavior functions in $\mathcal{F}_g$, there are some limitations on the form that $f_g(\cdot)$ should take. Attempting to return the robot to a pre-planned path (as many planners do) is less effective in shared-control systems where the deviations from the planned paths occur as a result of user initiated actions—this likely would be unwelcome to the user. Instead replanning would need to happen fast enough not to stall the task execution. We, therefore, advocate the use of real-time control policies which are defined in all parts of the state space.

Our current implementation favors dynamical systems formulations. The autonomous robot policies are learned from human demonstrations using an approach known as *Stable Estimator of Dynamical Systems (SEDS)* [**94**]. In SEDS, the target poses are modeled as attractors of a dynamical system. For each task or goal g, a set of demonstrations are collected by kinesthetically moving the robot. For the purposes of model-learning using SEDS, the world state is represented as a concatenated tuple ($\boldsymbol{q}^t$, $\dot{\boldsymbol{q}}^t$) consisting of the joint angles and velocities, where $\boldsymbol{q}^t \in \mathbb{T}^6$ (six-dimensional torus) and $\dot{\boldsymbol{q}}^t \in \mathbb{R}^6$. For each task, the SEDS algorithm learns the parameters of a time-independent dynamical system which models the joint velocities as a function of joint angles. So

$$\dot{\boldsymbol{q}}(t) \leftarrow f_g(\boldsymbol{q}(t)) \tag{7.8}$$

and $\boldsymbol{u}_a^t \triangleq K(\dot{\boldsymbol{q}}(t))$ where $K(\cdot)$ is the forward kinematics of the robot arm (since human teleoperation and control blending both happen in the end-effector Cartesian space). The dynamical system ensures the existence of the policy over the entire workspace, and that the robot trajectories follow the general contour of the task demonstrations.

7.3. Study Methods

The experiments were performed using the MICO robotic arm (Kinova Robotics, Canada) specifically designed for assistive purposes. The system was implemented using the Robot Operating System (ROS) and model learning was performed using MATLAB. The maximum end effector translational velocity along any axis was capped at 20 cm/s.

7.3.1. Task Descriptions

Three tasks were developed for the study (Fig. 7.2).

Simple Reaching (R): The user teleoperated the robotic arm to reach a single object (coffee carafe) placed in front of the robotic arm. The purpose of this task was to get the user accustomed to the control interface and to the different assistance levels provided by the system. At the end of the task, the assistance level that the user preferred was noted.

Reaching for Grasping (RfG): The user teleoperated the robotic arm to reach one of two objects on the table with a pose suitable for grasping, as the robot arm provided assistance. There was a near object (mug) and a far object (box), each of which required a different

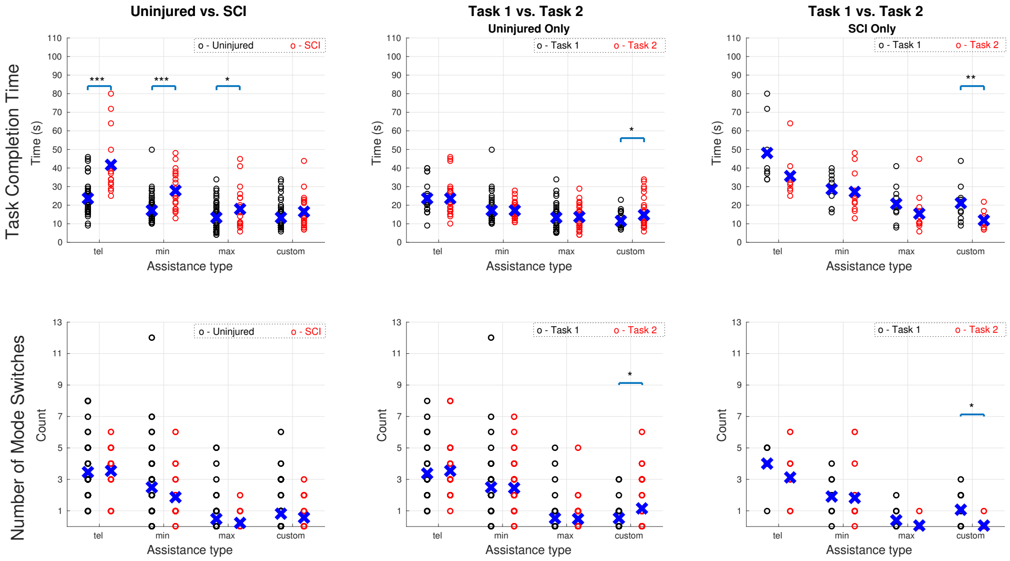


Figure 7.3. Task completion time (top row) and number of mode switches (bottom row) for uninjured vs. SCI subjects (first column), Task 1 vs. Task 2 for uninjured subjects only (second column), Task 1 vs. Task 2 for SCI subjects only (third column).

orientation of the gripper for grasping (side and top, respectively) and accordingly also different approach trajectories during reaching.

Reaching for Scooping (RfS): The user teleoperated the robotic arm to reach for one of two objects on the table with a pose suitable for a scooping motion, as the robot arm provided assistance. There was a near object and a far object (both bowls), each of which required a different approach trajectory. For this task, the end effector of the robotic arm was fitted with a spoon which had to be inserted into the bowl.

7.3.2. Study Protocol and Metrics

Subjects: For this exploratory study 17 subjects were recruited—13 uninjured control subjects (mean age: 26 ± 4, eight males and five females) and four spinal cord injury

(SCI) subjects (mean age: 35 ± 14, all males, C3-C5 injury levels). Seven of the uninjured subjects (five males, two females) and three of the SCI subjects used the *3D* interface paradigm, and the remaining subjects used the *2D* paradigm. All participants gave their informed, signed consent to participate in this experiment, which was approved by Northwestern University's Institutional Review Board.

Protocol: Each user performed all three tasks. The purpose of the practice task (R) was to get the user accustomed to the control interface and assistance system. Data was then collected on the remaining two tasks (RfG, RfS). The order of presentation for the RfG and RfS tasks was randomized and balanced across subjects, to avoid ordering effects.

Before the RfG and RfS trials, the user was first asked to operate the system in full teleoperation mode (*tel*) and also under three predefined assistance levels (*min, mid* and *max*). After this phase, the subject was given the option to customize the assistance level. Changes in assistance levels were communicated verbally to the system operator resulting in the parameter changes as outlined in Table 7.1. The user then tested the customized assistance level by executing the task. This customization procedure was repeated until the user was satisfied and lasted on an average 10 and a maximum of 15 minutes, resulting in assistance level *custom*. Data collection began only after this customization process was completed. Three trials were collected for *min, max* and *custom* assistance levels.[6] A typical session lasted approximately 1-1.5 hours. For the first (non-practice) task, the baseline from which customization began was the *mid* level assistance, with level *custom* being the result after customization. For the second task, customization began at this

[6] For one SCI participant one less trial was recorded for *min* assistance level during the first task due to a clerical error.

level *custom* from the first task as the baseline, with the option to further customize resulting in level *custom* for the second task.

Metrics: A number of objective metrics were evaluated this study. *Task Completion time* is the amount of time spent accomplishing a task. *Mode Switches* refers to the number of times the subject switched between the various modes of the control interface (Table 7.2). Mode switches additionally, is an indirect measure of the effort put forth by the user. At the end of the study, subjective data was gathered via a brief questionnaire. Users were given statements about the assistance system to rate on a 7-point Likert scale (1 is low, 7 is high), according to their agreement. The questions primarily concerned the utility value of the assistance system (*U1*), the system's accuracy in goal perception (*CA1*) and its understanding of what the user was trying to accomplish (*CA2*), and the contribution from the user (*CO1*) and the system (*CO2*) in task accomplishment.

7.4. Results

Here we report the results of our pilot study.[7] An improvement in task performance with customization is demonstrated, and a number of other interesting observations are noted. Task performance metrics for different assistance levels (denoted by *min*, *max* and *custom* in the plots) and teleoperation (*tel)* are analyzed across different subject groups, tasks and control interfaces. Note that the *custom* assistance level always lies in between (or is equal to) *min* and *max*. Statistical significance is determined by Welch t-tests for Figures 7.3-7.4 and two sided Wilcoxon Rank-Sum Test for Figure 7.5, where (***) indicates $p < 0.001$, (**) $p < 0.01$, and (*) $p < 0.05$.

[7] The video of the study can be found at http://argallab.smpp.northwestern.edu/index.php/publications/

7.4.1. Observations across Uninjured and SCI subjects

Insight into Reward Function: In this study, 17 subjects perform 34 rounds of customization in total. For seven customization rounds the mean *custom* task completion time is greater (by at least one standard error) than that of *max*. Similarly, the number of mode switches for *custom* is greater than that of *max* for 14 customization rounds. This indicates that subjects are not always optimizing for standard performance metrics—because there does exist a parametrization (*max*) which is known to the subjects and performs better with respect to these metrics. This provides insight that the true reward function that the user is optimizing likely is more complex than a simple time-optimal or minimum-effort reward function.

Task Performance: In Figure 7.3 (first column), the difference between uninjured and SCI subjects' task completion times drops steadily from *tel* to *custom* assistance levels. The t-tests reveal that while the difference between uninjured and SCI is statistically significant for *tel* (p = 5.1e-4), *min* (p = 6.5e-5) and *max* (p = 0.027), this difference disappears with the *custom* (p = 0.096) assistance level. That is, with customized assistance, the performance of SCI subjects is *statistically equivalent* to that of uninjured subjects. The variance in the data also *decreases* with customized assistance, showing the performance to become more consistent.

Interestingly, for mode switches there is no statistical difference between uninjured and SCI subject data for any of the assistance levels. This suggests that the number of mode switches is primarily determined by the nature of the task and control interface, and not the state of injury. However, SCI subjects do take more time than uninjured subjects to perform the same number of mode switches.

7.4.2. Observations across Tasks

Figure 7.3 (second and third columns) shows how task completion times and number of mode switches change between the first and second task for uninjured and SCI subjects. A statistically significant difference in performance only is observed for *custom* assistance, for both groups. Interestingly, SCI subjects show an *improvement* in task completion times ($p = 7.8e\text{-}3$) and mode switches ($p = 8.9e\text{-}3$) between the first and second tasks, whereas uninjured subjects exhibit a performance *decrease*. These changes in performance can be explained by the changes in assistance amount that result from the between-task customization (discussed further in Section 7.4.4).

7.4.3. Observations across Control Interfaces

Figure 7.4 (first column) shows the task completion times and mode switches for subjects using the *2D* and *3D* interfaces. Different operational modes do not seem to have an effect on task completion times, as both groups are statistically equivalent—*despite* the fact that for mode switches the difference between the *2D* and *3D* interfaces is significant. The second column of Figure 7.4 shows a *within-interface* performance comparison between *tel* and the different assistance levels. For all levels assistance significantly helps to reduce the number of mode switches during task execution.

The comparable task completion times may be explained by the fact that easier control compensates for time lost during mode switches. That is, due to the greater number of mode switches required for the *2D* interface compared to the *3D* interface, more time is taken performing mode switches. However, the number of dimensions simultaneously

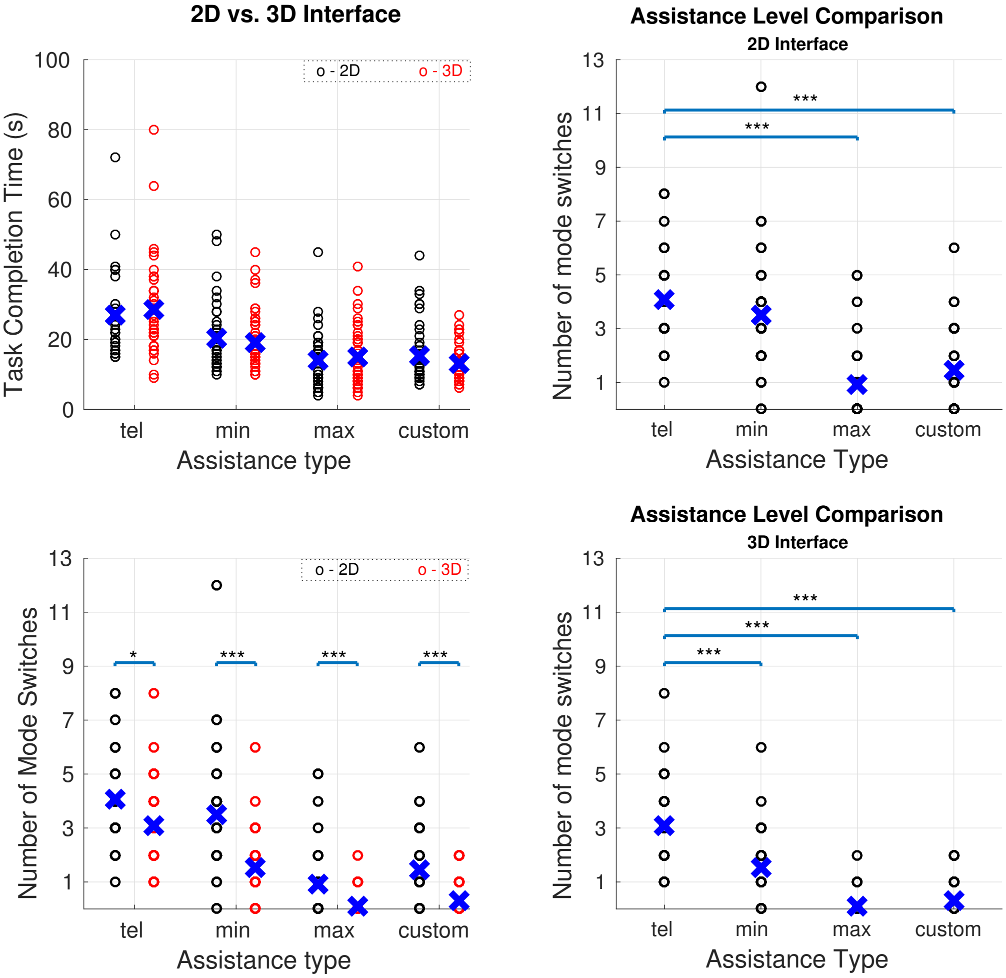


Figure 7.4. Left Column: Task completion time (top) and number of mode switches (bottom) for the 2D vs. 3D interfaces. Right Column: Within-interface assistance comparison for the 2D (top) and 3D (bottom) interfaces.

controlled is less for the *2D* interface compared to the *3D* interface, which makes the control easier.

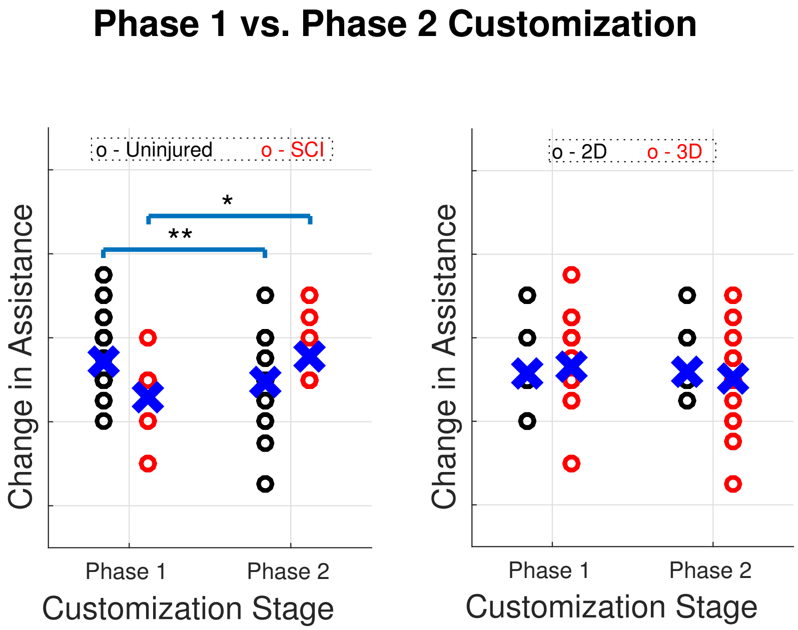


Figure 7.5. Relative change in assistance parameters during customization for Uninjured vs. SCI subjects (left) and 2D vs. 3D interfaces (right).

7.4.4. Relative Change in Parameters during Customization

Figure 7.5 shows the change in amount of assistance (parameter values) during customization for uninjured and SCI subjects. While SCI subjects on average increase the amount of assistance ($p = 0.020$) during the second phase of customization, uninjured subjects choose to reduce the amount of assistance ($p = 0.006$). By contrast, there are no noticeable changes in the amount of assistance when using the *2D* versus *3D* interface. Injury thus seems to be the primary factor in how subjects choose to change the customized assistance level, and the mapping paradigm seems to have little effect. Furthermore, it is interesting that uninjured subjects choose to reduce assistance in spite of an associated decrease in task performance.

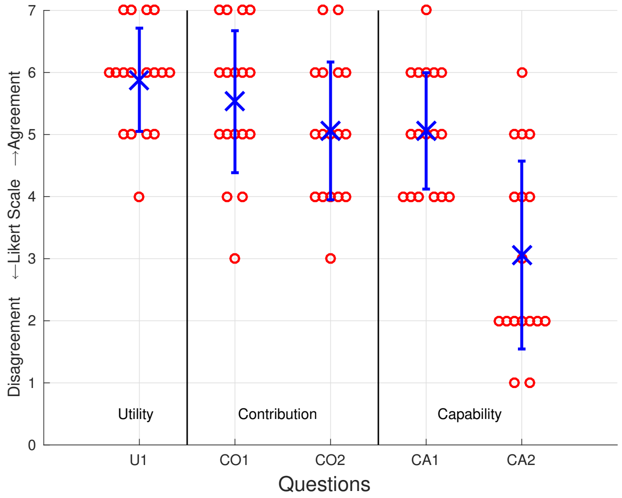


Figure 7.6. User responses on perceived utility, contribution and capability.

7.4.5. User Survey

Users rate (Fig. 7.6) the utility value of the assistance system fairly high (mean = 5.9±0.8) indicating that in general having assistance is favored. The users also think that the system is able to perceive goals accurately (mean = 5.1±1.8) and the inability to estimate human intent is fairly low (mean = 3.1 ± 1.1). The users also feel that they play an important part in accomplishing the task (mean = 5.5 ± 1.0), almost comparable to the contribution from assistance (mean = 5.1 ± 1.6), maybe indicating that they are not prepared to relinquish control altogether.

7.5. Discussion

One of the key findings of the study presented in this chapter is that individuals seek to optimize for rewards that are more complex than simple time-optimal or minimum-effort reward functions. Reward learning is a promising area of research in which the objective is to learn the human's reward function from data. Numerous algorithms have been developed to perform learning from suboptimal data [**30**], include active queries from the human to accelerate learning [**107**] *et cetera.* Despite all the progress, the perils of reward misspecification is still high [**122**]. That is, if the autonomous agent optimizes its behavior for a misspecified model of human behavior, it could have a detrimental effect on the human-robot interaction. Rather than opting for expensive data collection procedures to model human behavior and the relevant underlying latent factors to learn the reward structure, in this work, we simplify the problem by removing the need to determine the exact nature of the reward function. By circumventing this step and leveraging the human in the loop to be the optimizer, by way of the interactive verbal optimization procedure the user is still able to achieve high levels of task performance and satisfaction. The light-weight aspect of the procedure makes it flexible and robust to changes to user preferences that can happen over different time scales. A machine learning based approach may require infrastructure that can collect data, train models and update model parameters in an online fashion, but potentially incurring prohibitive costs. A simple analogy would be apt here; consider the air conditioning system in a car and imagine a scenario in which you could talk to the system by saying ‘a little cooler’, ‘a little warmer’ etc., to adjust the temperature setting. Depending on the day, your mood, what you are wearing etc., you might prefer a different temperature setting. A fully autonomous solution would require massive sensing capabilities that can infer the user’s state and then make adjustments to the air conditioning system according to the inferred preference. However, a simpler solution would be to just have the user be in control and change the temperature themselves. The slight cognitive and physical overhead in the latter approach is offset by more precision in the desired outcome, higher user satisfaction and even more important in the context of a shared control system, *the feeling of being in control.*

7.6. Conclusions

In this work, we formalized human robot interaction in shared autonomy within the framework of policy optimization. Furthermore, we introduced a system for user-driven customization as a constrained nonlinear policy optimization problem within this framework. Unlike standard policy optimization procedures in which the form of the reward function is assumed to be known, in this work no such assumptions were made. Instead, the end user was allowed to directly perform the optimization procedure. The aim was that this will lead to higher user satisfaction, and an increased sense of agency, which was crucial for the acceptance of novel technologies in the assistive domain. An interactive user-driven customization procedure was developed to ground the formalism and the results of the user study were presented. Results showed that all subjects were able to converge to an optimal assistance paradigm and an improvement in task performance with customization also was demonstrated.

CHAPTER 8

A JavaScript Framework for Crowdsourced Human-Robot Interaction Experiments: ***RemoteHRI***

This chapter presents a software tool that was developed in light of the COVID-19 pandemic to facilitate Human-Robot Interaction (HRI) research remotely. Software infrastructure is a critical piece for conducting proper research in this space of human-autonomy interaction and is often neglected in favor of pure algorithmic research. In this chapter, we present *RemoteHRI*, a JavaScript-based software framework for conducting HRI experiments in a web browser. Built with HRI researchers in mind, *RemoteHRI* includes a flexible set of software tools that allows for rapid prototyping and quick deployment of a wide range of laboratory-like experiments that can be run online. *RemoteHRI* uses the state-of-the-art ReactJS [1] framework to build standard HRI stimulus environments such as grid worlds, differential drive cars, and robotic arms. As a result, the researcher can solely focus on the experimental design thereby saving valuable time and effort. Code for *RemoteHRI* is available at `https://github.com/argallab/RemoteHRI`. The work presented in this chapter was done in equal collaboration between Deepak Gopinath and Finley Lau.[2]

[1] https://reactjs.org/

[2] 1. Gopinath was responsible for envisioning the entire software pipeline, identifying use-cases and functionality, and worked on adding new environments. 2. Lau was the primary software developer and was responsible for implementing both the client and the server side applications.

8.1. Introduction

Evaluating progress in science and engineering heavily depends on proper experimentation protocols. In the HRI domain, experiments are needed both to understand the human decision-making process while interacting with autonomous agents and to evaluate the success of robotics autonomy algorithms in interacting with humans and other agents.

In an ideal situation, researchers conduct HRI experiments with robotic systems in the real world. These provide researchers with rich data that encodes the sensing and actuating complexities of robot operation and human interaction in the real world. However, designing and conducting HRI experiments on real robotic systems come with a great deal of challenges, especially in the academic setting.

First, real robotic systems are expensive, and academic labs rarely own multiple robots of the same type. This drastically limits the number of studies that could be run in parallel, thereby making data collection extremely slow. Second, subject recruitment for academic studies can suffer from biases due to lack of diversity in the recruitment pool. Third, in-person subject studies typically require researchers to be in close physical contact with the subjects. However, in light of the COVID-19 pandemic, it became advisable to maintain stricter social distancing protocols due to health concerns, rendering an in-person study practically impossible. Particularly, in the sub-domain of assistive robotics, the end-user population (people with motor-impairments as a result of trauma or neurodegenerative disease) can belong to high-risk immunocompromised groups and as such participation in an in-person study can pose significant health risks.

By contrast, simulation-based experiments are less expensive and can help to design more targeted in-person studies. By conducting experiments online, researchers can parallelize data collection and achieve greater diversity in the recruitment pool. Furthermore, online settings remove the need for in-person supervision, alleviating the health risks of being in close proximity to others. Exploratory HRI studies conducted in a simulated environment can also provide strong priors, in terms of baseline models for human behavior and initial evaluation of robotics algorithms. These studies additionally inform resourceful design of a real-world experiment. However, the realism of a simulator is closely related to the quality and accuracy of the underlying physics engine used by the simulation. As a result, the human-robot interaction that unfolds in a simulated HRI experiment could have systematic biases.

Performing human subject studies in a simulated environment is not novel. For example, *jsPsych* [**42**] is a popular JavaScript library used by researchers in the field of psychology to perform simple online behavioral experiments. However, the stimulus plugins available in *jsPsych* do not cover the full space of rich and complex stimuli needed for HRI. The field of HRI research currently lacks a comprehensive software framework for conducting simulation-based experiments and relies on one-off solutions developed for specific projects, leading to wasted time and repeated effort. Although primarily used for reinforcement learning research, OpenAI Gym [**27**] provides a suite of simulated environments that are widely used for HRI experiments [**20, 139**]; however, the JS interface needed for crowd-sourced browser-based online studies is still under development.

RemoteHRI provides HRI researchers a set of tools to design and conduct common HRI studies in a seamless manner. The framework is designed with the researcher in mind, specifically focusing on ease of use and rapid prototyping. *RemoteHRI* derives its inspiration from frameworks such as *jsPsych* and supports flexible experiment design, an easy-to-use researcher interface, and prepackaged standard HRI stimulus environments. *RemoteHRI* uses ReactJS for client-side rendering, which allows for increased flexibility, speed, and ease of development through its management of state and structural organization of components (groups of elements displayed on the screen) as compared to vanilla JavaScript.

The key features of *RemoteHRI* are as follows:

- **Researcher-Centered Design**: *RemoteHRI* was built with researchers from different backgrounds and programming experience in mind and provides a low entry point for easily prototyping and building HRI experiments, without having to develop the client-side application (which typically requires researchers to learn to use graphics APIs, and physics libraries) or the server-side application.
- **Modular/Extensible**: *RemoteHRI* is highly modular in that the different applications of the framework operate independently of each other and may be replaced with an equivalent module. For example, the *RemoteHRI* server could be replaced with a custom-built server without affecting the client-side application. Furthermore, implementation of different stimulus types follows predefined template types and therefore the framework can be easily extended to work with new types of stimuli.
- **Unified Researcher Interface**: Regardless of the specific experimental domain or design choice, the researcher specifies experimental flow through a single JSON file, which we refer to as the *Experiment.json* file (discussed more in detail in

Section 8.4). *RemoteHRI* provides a GUI interface and a set of utility functions allowing researchers to quickly generate the *Experiment.json* files.

- **Plug and Play**: *RemoteHRI* experiments are completely controlled by the *Experiment.json* file. By specifying the stimuli and their respective properties used in the experiment in the *Experiment.json* file, researchers can create entire experiments without changing the client-side application.

8.2. HRI Experiment Design

HRI researchers may have varying goals for their experiments. Two possible purposes are, (a) data collection experiments (for example, human teleoperation data of robotic manipulators to build data-driven computational models of human decision making) [**174**] or (b) algorithm evaluation experiments (for example, evaluation of a shared control robot policy for assistive robotic manipulators) [**65**].

A quick analysis of different types of experiments conducted in the field of HRI reveals various commonalities in experimental design [**19, 62, 88, 166**]. We identify four different phases in the majority of these experiments:

- **Consenting Phase**: This is common for any experiment done in an academic setting. During this phase, the researcher describes the experiment in detail to the subject either in writing or verbally. After evaluating the risks and benefits of the experiment, the subject chooses whether to participate in the study. During this phase, researchers typically also collect non-identifiable demographic and subject-specific information, such as age, gender, and race.

- **Training Phase**: In the training phase, the participant gets familiarized with the experimental setup. This phase also helps researchers establish a baseline performance, which could be a useful measure in data analysis.
- **Testing Phase**: The testing phase typically consists of multiple blocks of experimental trials under different experimental conditions. This phase may serve to collect data (for example, human teleoperation data of robotic manipulators to build data-driven computational models of human decision making) or to evaluate an algorithm's performance (for example, evaluation of a shared control robot policy for assistive robotic manipulators).
- **Survey Phase**: The participant is presented with questions related to their experience of interacting with the robotic system during the study.

Note that in a given experiment, possibly with the exception of the consenting phase, phases may occur multiple times in no particular order. *RemoteHRI* recognizes the need for such combinatorial flexibility in experimental design and offers researchers an easy approach to specify any experimental flow.

8.3. Framework Modules

RemoteHRI consists of two main software modules: (a) the client-side application and (b) the server-side application. Participants interact with the client-side application in their browsers during an experiment, while the server-side application ensures the proper delivery of experiment content. In the following subsections, we will describe the details of the client-side and the server-side applications.

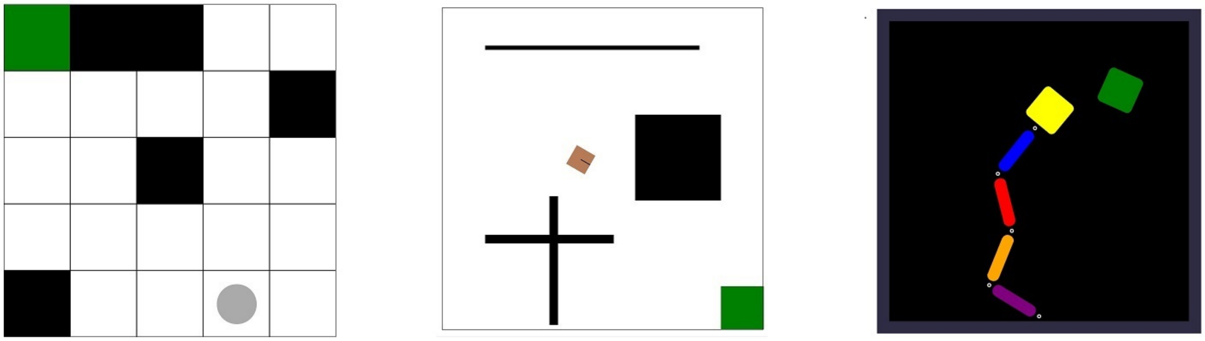


Figure 8.1. Examples of active stimuli in *RemoteHRI*. *Left*: *DiscreteGridWorld* stimulus with static obstacles. *Middle*: *DifferentialDriveRobot* stimulus with static obstacles for navigation tasks. *Right*: *RoboticArm* stimulus for reaching tasks. The goal states are indicated in green.

8.3.1. Client

The client-side application is responsible for presenting the stimulus for each trial during the course of the experiment. In *RemoteHRI* we consider two main classes of stimuli:

- **Passive:** A passive stimulus is an environment in which a participant ‘passively’ observes the presented stimuli (audio/video/text). For example, participants could be shown images of two different humanoid robots and asked which one has more human-like features. Participant responses to a passive stimulus depend on the stimulus type and can take many forms such as single answer/multiple choice, multiple answer/multiple choice, 5- or 7-point Likert scale, or free text response. *RemoteHRI* provides a suite of common passive stimuli such as videos, audio recordings, image and text displays, surveys/questionnaires, *et cetera*.
- **Active:** An active stimulus is an environment in which there is at least one entity that is actuated by the participant via some form of control input, by

> an autonomous agent, or both. We refer to a stimulus containing only a human-controlled entity as *H-Active* and a stimulus containing only an autonomy-controlled entity as *A-Active*. A stimulus environment containing entities controlled by both a human and an autonomous agent is referred to as *HA-Active*. An active stimulus environment may have one or more active entities. *RemoteHRI* includes implementations of some of the standard active environments used in simulated HRI experiments, such as discrete grid-worlds with simulated point robots, differential-drive robots, and planar robotic arms. The implementation of these standard environments is shown in Figure 8.1.

Environments can also be classified based on whether an active agent is operating in a static (*S-Env*) or dynamic (*D-Env*) environment. For example, a human-controlled point robot operating in a discrete grid-world with moving obstacles is an *H-Active/D-Env* stimulus whereas an autonomy-controlled planar robotic arm performing reaching motion towards a fixed goal location is an *A-Active/S-Env* stimulus.

RemoteHRI comes with standard implementations of planning and control algorithms to autonomously control agents in an active environment. Similarly the framework also implements simple algorithms such as random walks and periodic motion for the control of dynamic aspects of the environment. The researcher can readily activate any of these implemented algorithms in the *Experiment.json* file.

Implementation details: The ReactJS client-side application consists of a modular component structure allowing variable display of stimuli on the screen (Figure 8.2).

An experiment is rendered through a React component called *Content*, which represents the space on the browser screen belonging to the experiment's content. Depending on the specification in the *Experiment.json* file, this content can take the form of various stimulus components, such as *DiscreteGridWorld*, *DifferentialDriveRobot*, or *RoboticArm*. Each of these stimulus components contains the same child component structure (*Header/Instructions*, *StimulusView*, *Continuation*), providing for consistency and ease of implementation for new stimulus types. The header/instructions component specifies each trial's title and instructions for the participant. The stimulus view renders the specified stimulus, such as the grid world, differential-drive robot world, or robotic arm world. The continuation component specifies how the participant may proceed to the next trial after completion of a trial, such as by clicking a button or pressing any key.

Since the client-side application is only involved with taking a JSON specification for a trial and rendering the appropriate stimulus on the screen, all experimental flow logic can be abstracted to the server application. This allows for the modular plug-and-play feature of *RemoteHRI*, as researchers can easily use the provided client-side application without modification to render their experiments.

8.3.2. Server

The experiment flow for each participant in an experiment is managed by the server application. It is built using the *Nodus Ponens* framework, a light, full-stack framework for running high-level reasoning and cognitive science experiments in Node.js.[3]

The key features of the *RemoteHRI* server are as follows:

- **Experimental Flow Management**: By using the specification in the *Experiment.json* file, the server is able to construct a list of stimuli trials to present to

[3] Code available at https://www.npmjs.com/package/nodus-ponens. Designed by Sangeet Khemlani. Distributed under the Creative Commons License.

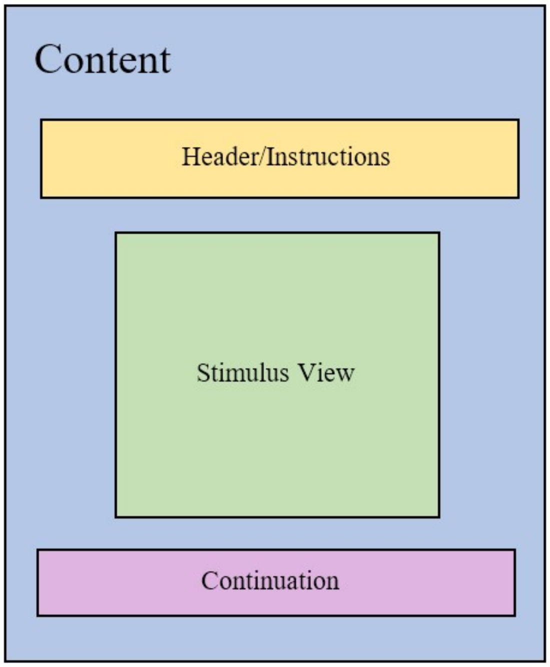


Figure 8.2. Diagram showing structural rendering layout of the client-side application.

each participant. It handles randomization and construction of the unique order in which trials are presented to a particular participant. In addition to assigning a unique ID to each participant, it also keeps track of where participants are in the experiment in order to correctly serve subsequent trials.

- **Data Collection**: The server collects both incomplete and complete data on a trial-per-trial basis. After a participant finishes a trial, the server stores the collected data from the client-side application into a JSON file containing all information collected from the current trial as well as any previous ones.
- **Session Management**: The server keeps track of a participant's experiment data through browser sessions. Notably, this allows for participants to reconnect

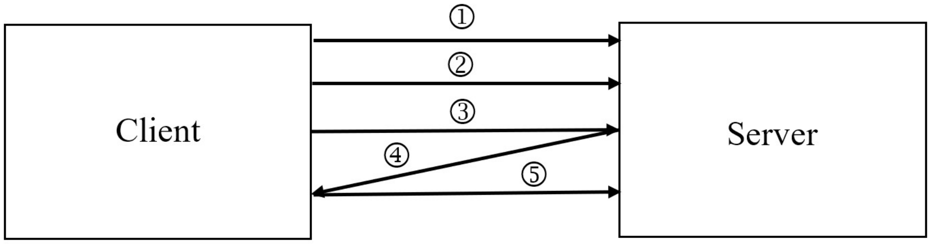


Figure 8.3. Diagram showing communication between the client-side application and the server. 1) Client sends initial request to server for experiment name. 2) Client sends request to start experiment. 3) Client sends request for first trial. 4) Server responds with JSON data for first trial. 5) Client sends collected trial data back to server, along with request for next trial.

to an experiment and continue from where they left off, if a researcher specifies that this is allowed.

8.4. Experiment Specification

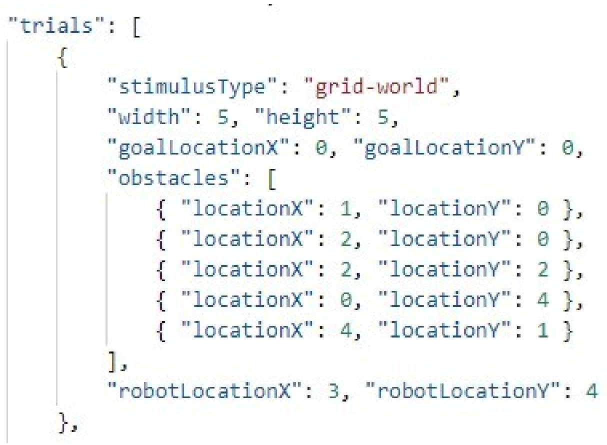
Experiment.json: This JSON file completely specifies an experiment. It contains information about experiment structure as well as trial specifications.

- **Experiment Structure**: *RemoteHRI* provides functionality to group trials through block-level organization. It also exposes boolean flags to specify whether blocks should be shuffled as well as whether trials within a block should be shuffled. It also allows for specification of trials (such as pre/post block survey questionnaires) preceding and following a trial block, regardless of whether the trials inside of the block are shuffled.
- **Trial Specifications**: The *Experiment.json* file contains information relevant to rendering the starting state of the stimulus presented in each trial. For example, in a grid world stimulus, the *Experiment.json* file would specify the width and

height of the world and the starting positions of all objects in the world (Figure 8.4). Each stimulus type has its own JSON schema to completely specify its state.

The *Experiment.json* file can be generated by any means to create a JSON file. This can include manually writing a file, a GUI-based tool provided through *RemoteHRI*, or scripts in any programming language that can write to a JSON file. This allows for flexibility and scalability for the researcher to create an experiment, regardless of their programming experience or the scale of their experiment.

The *Experiment.json* file also allows for minimal configuration when extending *RemoteHRI* with a new stimulus type. Due to the freedom in schema provided by JSON files, a researcher only needs to implement the client-side rendering of their new stimulus using ReactJS and decide what properties may be specified in the *Experiment.json* file.



```
"trials": [
    {
        "stimulusType": "grid-world",
        "width": 5, "height": 5,
        "goalLocationX": 0, "goalLocationY": 0,
        "obstacles": [
            { "locationX": 1, "locationY": 0 },
            { "locationX": 2, "locationY": 0 },
            { "locationX": 2, "locationY": 2 },
            { "locationX": 0, "locationY": 4 },
            { "locationX": 4, "locationY": 1 }
        ],
        "robotLocationX": 3, "robotLocationY": 4
    },
```

Figure 8.4. Snapshot of an example *Experiment.json* file specification for a grid-world experiment. The configuration specified corresponds to the *DiscreteGridWorld* in Figure 8.1.

8.5. Conclusions

In this chapter, we presented *RemoteHRI*, a JavaScript framework for designing and deploying crowdsourced HRI experiments online. We anticipate that *RemoteHRI* can also help in lowering the barrier to entry for HRI research and increase equity in terms of broadened participation from underrepresented HRI researchers. Future extensions of *RemoteHRI* will include functionality to use custom input devices such as joysticks and sip-and-puffs [**57**] in addition to standard keyboard and mouse input. This specifically targets the subdomain of assistive robotics in which studies with motor-impaired subjects are of paramount importance for the successful adoption of assistive technologies. Future iterations of the framework will also include 3D simulated environments.

CHAPTER 9

Final Words

The work presented in this dissertation touches upon three distinct aspects of a shared-autonomy assistive system, namely (a) the intent inference capabilities of the autonomous agent, (b) the interaction between the human and the control interface, and (c) the design of the arbitration scheme for control sharing.

9.1. Nudging for Human-AI Alignment

Intent inference is of paramount important in any shared autonomy assistive system that does not have transparent human goals as the primary goal of an assistive autonomous agent is to help the user achieve *their desired goals and intentions.* Chapters 5 and 6 attempt to tackle a particularly thorny problem faced by autonomous agents, which is that of successfully inferring intent from *limited* control signals that are generated by the user. The limitations arise due to various reasons such as (a) inherent physical constraints in the expressivity of the control interface as a result of unique physical form factors, (b) lack of motor skills due to insufficient practice and experience, and (c) motor limitations that arise due to spinal cord or brain injuries.

The key idea behind *intent disambiguation* developed in these chapters is that it is a mechanism by which the autonomous agents could alter the *context* in which users are issuing their task-related actions. For example, simply by placing the user's control in a different control mode or by moving a robot to a different location, the signals generated by the human carry new meaning and information and as a result the autonomous agent can perform intent inference more accurately. Such a nudge in the context does not restrict the set of actions available to the user. This is important in order to retain as much agency as possible with the user. As a downstream consequence of such a contextual nudge, the user receives more assistance in return. The effectiveness of this strategy could further be enhanced by making the user aware of whether they are being nudged by the autonomous agent.

Quite often, an autonomous agent's ability to assist the user is limited by the lack of fidelity of the models of human behavior that are available for it to use. Although data-driven methods based on state-of-the-art machine learning algorithms offer the promise of the development of generalized robotics capabilities, the applicability of these methods is severely limited by the availability of data or how laborious data collection and curation can be. On other hand, autonomous agents could easily be endowed with hand-engineered models of human behavior designed by domain experts using limited amounts of data and strong priors. Typically, the gap between these models and the actual human behavior could be large, thereby resulting in low utility value of these models for the autonomous agent. The notion of nudging can again be leveraged in such scenarios to bridge the gap between the model predictions and human behavior by nudging the humans to behave in accordance to the model output. Effective training protocols that incentivize humans to be cooperative can help them meet the autonomous agent's capabilities halfway and as a result bring a great deal of benefit to the overall human-robot team.

Nudging is applicable at the *conceptual* level as well and it could be utilized to bridge the gap between what the user already knows about the technological possibilities of assistive robotics and what the future might be like. User-centric iterative design processes are highly encouraged in a domain such as assistive robotics because technological success is closely tied to how useful they are to the end user. Quite often researchers spend valuable time and effort in building algorithms and robot functionalities that end up having very low user acceptance and adoption rates. Hence, it is very important to generate *minimum viable solutions* that can be quickly prototyped and deployed. These minimum viable solutions typically utilize heuristics-based approaches that are easy to implement and therefore can act as *conceptual nudges* on the end user. They provide a sneak peek into the innovative possibilities of assistive technology and can encourage the user to adopt them even in the short term. Additionally, minimum viable solutions informed by heuristics from other fields such as cognitive science, psychology, and behavioral economics can become instrumental to making incremental progress when data-driven methods are not applicable.

9.2. Interface Awareness in Human-Robot Interaction

The notion of *interface awareness* in assistive robotics was initially explored by Nejati et al., in [**88**] in which the usage characteristics of different interfaces are investigated in various command-following and trajectory-following tasks. In Chapter 4 we take the idea of interface awareness a step further by explicitly modeling the *physical activation* aspect of interface within a model of robot teleoperation in order to improve the intent inference capabilities of the autonomous agent. The ideas introduced in Chapter 4 go beyond the realm of assistive robotics and are applicable to any situation in which there is any kind of intermediate physical mechanism that facilitates human-machine interaction. For example, in the domain of autonomous driving, if we want to fully characterize how humans drive, in addition to the modeling the driver's cognitive decision making processes, it is also necessary to evaluate how well they are *physically* able to turn the steering wheel or physically press a gas pedal using their foot. Noise in the motor system that arises due to lack of motor control or motor skill can alter how human intentions get communicated to the external world that includes the autonomous agent. In order to extract the true signal from the noise the autonomous agent will have to explicitly model the noise characteristics. Quite often, physical aspects of interface usage are overlooked in the HRI domain because we assume that humans are always physically capable of executing the necessary physical actions when operating an interface. The model presented in Chapter 4 makes a clear distinction between *conceptual* knowledge of doing a task and the *physical* aspect of actually executing the task. Without the conceptual understanding, the human would not know what motor skill to activate in the first place. However, even if the conceptual understanding is thorough, unless the human can physically perform the action, the resulting change in environment might be undesired. Even in non-assistive domains, autonomous agents could be designed to smooth out errors and imprecisions that arise due to the noise in physical activation when a human controls a machine.

The work presented in this dissertation scratches the surface of what is possible in the domain of human-autonomy interaction. It is my hope that researchers will strive to find a balance between pushing the envelope in terms of computational modeling, algorithm and engineering design on one hand and building useful technologies that have direct impact on improving the lives of the end user on the other hand. To that end, not only should researchers be aware of the limitations of the technologies that are being developed but also realize that fully algorithmic decision-making solutions may not have the highest utility value for the human, in which case it is simply better to have the human be in control of the decisions.

References

[1] Abbink, D. A., Carlson, T., Mulder, M., de Winter, J. C., Aminravan, F., Gibo, T. L., and Boer, E. R. A topology of shared control systems finding common ground in diversity. *IEEE Transactions on Human-Machine Systems*, 99 (2018), 1–17.

[2] Abi-Farrajl, F., Henze, B., Werner, A., Panzirsch, M., Ott, C., and Roa, M. A. Humanoid teleoperation using task-relevant haptic feedback. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2018).

[3] Acharya, U., Kunde, S., Hall, L., Duncan, B. A., and Bradley, J. M. Inference of user qualities in shared control. In *2018 IEEE International Conference on Robotics and Automation (ICRA)* (2018), IEEE, pp. 588–595.

[4] Ajzen, I., Brown, T. C., and Carvajal, F. Explaining the discrepancy between intentions and actions: The case of hypothetical bias in contingent valuation. *Personality and Social Psychology Bulletin 30*, 9 (2004), 1108–1121.

[5] Akinola, I., Chen, B., Koss, J., Patankar, A., Varley, J., and Allen, P. Task level hierarchical system for BCI-enabled shared autonomy. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)* (2017), IEEE, pp. 219–225.

[6] Amari, S.-i. Dynamics of pattern formation in lateral-inhibition type neural fields. *Biological Cybernetics 27*, 2 (1977), 77–87.

[7] Argall, B. D. Turning assistive machines into assistive robots. In *Quantum Sensing and Nanophotonic Devices XII* (2015), vol. 9370, International Society for Optics and Photonics, p. 93701Y.

[8] Argall, B. D., Chernova, S., Veloso, M., and Browning, B. A survey of robot learning from demonstration. *Robotics and Autonomous Systems 57*, 5 (2009), 469–483.

[9] Arrington, C. M., and Logan, G. D. The cost of a voluntary task switch. *Psychological Science 15*, 9 (2004), 610–615.

[10] Atanasov, N., Le Ny, J., Daniilidis, K., and Pappas, G. J. Information acquisition with sensing robots: Algorithms and error bounds. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (2014).

[11] Baker, C. L., Saxe, R. R., and Tenenbaum, J. B. Action understanding as inverse planning. *Cognition 113*, 3 (2009), 329–349.

[12] Baker, C. L., Tenenbaum, J. B., and Saxe, R. R. Goal inference as inverse planning. In *Proceedings of the Cognitive Science Society* (2007).

[13] Barfoot, T. D. *State estimation for robotics.* Cambridge University Press, 2017.

[14] Barrouillet, P., Bernardin, S., and Camos, V. Time constraints and resource sharing in adults working memory spans. *Journal of Experimental Psychology: General* (2004), 83–100.

[15] Beer, J. M., Fisk, A. D., and Rogers, W. A. Toward a framework for levels of robot autonomy in human-robot interaction. *Journal of Human-Robot Interaction 3*, 2 (2014), 74.

[16] Bien, Z., Chung, M.-J., Chang, P.-H., Kwon, D.-S., Kim, D.-J., Han, J.-S., Kim, J.-H., Kim, D.-H., Park, H.-S., Kang, S.-H., Lee, K., and Lim, S.-C. Integration of a rehabilitation robotic system (KARES II) with human-friendly man-machine interaction units. *Autonomous Robots 16*, 2 (2004), 165–191.

[17] Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316* (2016).

[18] Bowman, M., Li, S., and Zhang, X. Intent-uncertainty-aware grasp planning for robust robot assistance in telemanipulation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (2019).

[19] Breazeal, C., Kidd, C. D., Thomaz, A. L., Hoffman, G., and Berlin, M. Effects of nonverbal communication on efficiency and robustness in human-robot teamwork. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2005), IEEE, pp. 708–713.

[20] Broad, A., Abraham, I., Murphey, T., and Argall, B. Data-driven koopman operators for model-based shared control of human–machine systems. *The International Journal of Robotics Research 39*, 9 (2020), 1178–1195.

[21] Broad, A., Argall, B., and Creation, D. Geometry-based region proposals for accelerated image-based detection of 3d objects. In *RSS Workshop on Deep Learning* (2016).

[22] Broad, A., Arkin, J., Ratliff, N., Howard, T., and Argall, B. Real-time natural language corrections for assistive robotic manipulators. *The International Journal of Robotics Research 36*, 5-7 (2017), 684–698.

[23] Broad, A., Arkin, J., Ratliff, N., Howard, T., Argall, B., and Graph, D. C. Towards real-time natural language corrections for assistive robots. In *RSS Workshop on Model Learning for Human-Robot Communication* (2016).

[24] Broad, A., Murphey, T., and Argall, B. Learning models for shared control of human-machine systems with unknown dynamics. *arXiv preprint arXiv:1808.08268* (2018).

[25] Broad, A., Murphey, T., and Argall, B. Operation and imitation under safety-aware shared control. In *Workshop on the Algorithmic Foundations of Robotics* (2018).

[26] Broad, A., Murphey, T., and Argall, B. Highly parallelized data-driven MPC for minimal intervention shared control. *Proceedings of Robotics: Science and Systems (RSS)* (2019).

[27] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. OpenAI gym. *arXiv preprint arXiv:1606.01540* (2016).

[28] Brooks, C., and Szafir, D. Balanced information gathering and goal-oriented actions in shared autonomy. In *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)* (2019), IEEE, pp. 85–94.

[29] Brose, S. W., Weber, D. J., Salatin, B. A., Grindle, G. G., Wang, H., Vazquez, J. J., and Cooper, R. A. The role of assistive robotics in the lives of persons with disability. *American Journal of Physical Medicine & Rehabilitation 89*, 6 (2010), 509–521.

[30] Brown, D., Goo, W., Nagarajan, P., and Niekum, S. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In *International Conference on Machine Learning* (2019), PMLR, pp. 783–792.

[31] Callaway, F., Hardy, M., and Griffiths, T. Optimal nudging. In *CogSci* (2020).

[32] Canal, G., Alenyà, G., and Torras, C. Personalization framework for adaptive robotic feeding assistance. In *International Conference on Social Robotics* (2016), Springer, pp. 22–31.

[33] Cantrell, R., Scheutz, M., Schermerhorn, P., and Wu, X. Robust spoken instruction understanding for HRI. In *2010 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI)* (2010), IEEE, pp. 275–282.

[34] Carlson, T., and Demiris, Y. Collaborative control for a robotic wheelchair: Evaluation of performance, attention and workload. *IEEE Transactions on Systems, Man, and Cybernetics: Part B 42*, 3 (2012), 876–888.

[35] Chen, M., Nikolaidis, S., Soh, H., Hsu, D., and Srinivasa, S. Planning with trust for human-robot collaboration. In *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction* (2018), ACM, pp. 307–315.

[36] Chen, T. L., Ciocarlie, M., Cousins, S., Grice, P. M., Hawkins, K., Hsiao, K., Kemp, C. C., King, C.-H., Lazewatsky, D. A., Leeper, A. E., et al. Robots for humanity: using assistive robotics to empower people with disabilities. *IEEE Robotics & Automation Magazine 20*, 1 (2013), 30–39.

[37] Chen, X., Xu, J., Zhou, R., Chen, W., Fang, J., and Liu, C. Trajvae: A variational autoencoder model for trajectory generation. *Neurocomputing 428* (2021), 332–339.

[38] Choi, Y. S., Anderson, C. D., Glass, J. D., and Kemp, C. C. Laser pointers and a touch screen: intuitive interfaces for autonomous mobile manipulation for the motor impaired. In *Proceedings of the International SIGACCESS Conference on Computers and Accessibility* (2008).

[39] Churchland, M. M., Afshar, A., and Shenoy, K. V. A central source of movement variability. *Neuron 52*, 6 (2006), 1085–1096.

[40] Cipriani, C., Zaccone, F., Micera, S., and Carrozza, M. C. On the shared control of an emg-controlled prosthetic hand: analysis of user–prosthesis interaction. *IEEE Transactions on Robotics 24*, 1 (2008), 170–184.

[41] Cutsuridis, V. Cognitive models of the perception-action cycle: A view from the brain. In *The 2013 International Joint Conference on Neural Networks (IJCNN)* (2013), IEEE, pp. 1–8.

[42] De Leeuw, J. R. jsPsych: A JavaScript ibrary for creating behavioral experiments in a web browser. *Behavior Research Methods 47*, 1 (2015), 1–12.

[43] Demeester, E., Hüntemann, A., Vanhooydonck, D., Vanacker, G., Van Brussel, H., and Nuttin, M. User-adapted plan recognition and user-adapted shared control: A Bayesian approach to semi-autonomous wheelchair driving. *Autonomous Robots 24*, 2 (2008), 193–211.

[44] Desai, M., and Yanco, H. A. Blending human and robot inputs for sliding scale autonomy. In *ROMAN 2005. IEEE International Workshop on Robot and Human Interactive Communication, 2005.* (2005), IEEE, pp. 537–542.

[45] Devigne, L., Aggravi, M., Bivaud, M., Balix, N., Teodorescu, C. S., Carlson, T., Spreters, T., Pacchierotti, C., and Babel, M. Power wheelchair navigation assistance using wearable vibrotactile haptics. *IEEE Transactions on Haptics 13*, 1 (2020), 52–58.

[46] Devigne, L., Pasteau, F., Carlson, T., and Babel, M. A shared control solution for safe assisted power wheelchair navigation in an environment consisting of negative obstacles: a proof of concept. In *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)* (2019), IEEE, pp. 1043–1048.

[47] Dragan, A. D., Lee, K. C., and Srinivasa, S. S. Legibility and predictability of robot motion. In *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction (HRI)* (2013).

[48] Dragan, A. D., and Srinivasa, S. S. *Formalizing assistive teleoperation.* MIT Press, 2012.

[49] Dragan, A. D., and Srinivasa, S. S. A policy-blending formalism for shared control. *The International Journal of Robotics Research 32*, 7 (2013), 790–805.

[50] Driessen, B. J. F., Kate, T. K. T., Liefhebber, F., Versluis, A. H. G., and van. Woerden, J. A. Collaborative control of the manus manipulator. *Universal Access in the Information Society 4*, 2 (2005), 165–173.

[51] Dvijotham, K., and Todorov, E. Inverse optimal control with linearly-solvable mdps. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)* (2010), pp. 335–342.

[52] Endsley, M. R. From here to autonomy: lessons learned from human–automation research. *Human Factors 59*, 1 (2017), 5–27.

[53] Endsley, M. R. Level of automation forms a key aspect of autonomy design. *Journal of Cognitive Engineering and Decision Making 12*, 1 (2018), 29–34.

[54] Endsley, M. R., and Kaber, D. B. Level of automation effects on performance, situation awareness and workload in a dynamic control task. *Ergonomics 42*, 3 (1999), 462–492.

[55] Ewert, B., Loer, K., and Thomann, E. Beyond nudge: advancing the state-of-the-art of behavioural public policy and administration. *Policy & Politics 49*, 1 (2021), 3–23.

[56] Faisal, A. A., Selen, L. P., and Wolpert, D. M. Noise in the nervous system. *Nature Reviews Neuroscience 9*, 4 (2008), 292–303.

[57] Farnet, M. G., McWilliams, H. R., and Stutz, J. J. Sip and puff mouse, Aug. 19 2008. US Patent 7,412,891.

[58] Fernández-Carmona, M., Peula, J. M., Urdiales, C., and Sandoval, F. Towards a shared control navigation function: efficiency based command modulation. In *International Work-Conference on Artificial Neural Networks* (2015), Springer, pp. 185–198.

[59] Fischinger, D., Einramhof, P., Papoutsakis, K., Wohlkinger, W., Mayer, P., Panek, P., Hofmann, S., Koertner, T., Weiss, A., Argyros, A., et al. Hobbit, a care robot supporting independent living at home: First prototype and lessons learned. *Robotics and Autonomous Systems 75* (2016), 60–78.

[60] Fitzsimons, K., Tzorakoleftherakis, E., and Murphey, T. D. Optimal human-in-the-loop interfaces based on maxwell's demon. In *2016 American Control Conference (ACC)* (2016), IEEE, pp. 4397–4402.

[61] Flash, T., and Hogan, N. The coordination of arm movements: an experimentally confirmed mathematical model. *The Journal of Neuroscience 5*, 7 (1985), 1688–1703.

[62] Ghassemi, M., Triandafilou, K., Barry, A., Stoykov, M. E., Roth, E., Mussa-Ivaldi, F. A., Kamper, D. G., and Ranganathan, R. Development of an EMG-controlled serious game for rehabilitation. *IEEE Transactions on Neural Systems and Rehabilitation Engineering 27*, 2 (2019), 283–292.

[63] Goil, A., Derry, M., and Argall, B. D. Using machine learning to blend human and robot controls for assisted wheelchair navigation. In *2013 IEEE International Conference on Rehabilitation Robotics (ICORR)* (2013), IEEE, pp. 1–6.

[64] Gopinath, D., and Argall, B. Mode switch assistance to maximize human intent disambiguation. In *Robotics: Science and Systems* (2017).

[65] Gopinath, D., Jain, S., and Argall, B. D. Human-in-the-loop optimization of shared autonomy in assistive robotics. *IEEE Robotics and Automation Letters 2*, 1 (2017), 247–254.

[66] Gopinath, D., Nejati Javaremi, M., and Argall, B. Chicago, IL, USA. customized interface-aware assistance (v1.0). [Online]. Available: `https://github.com/argallab/customized_interface_aware_assistance.git`, 2020.

[67] Gopinath, D., and Weinberg, G. A generative physical model approach for enhancing the stroke palette for robotic drummers. *Robotics and Autonomous Systems 86* (2016), 207–215.

[68] Gopinath, D. E., and Argall, B. D. Active intent disambiguation for shared control robots. *IEEE Transactions on Neural Systems and Rehabilitation Engineering 28*, 6 (2020), 1497–1506.

[69] Gulati, S., Jhurani, C., Kuipers, B., and Longoria, R. A framework for planning comfortable and customizable motion of an assistive mobile robot. In *Proceedings in the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2009).

[70] Hart, S. G. NASA-task load index (NASA-TLX); 20 years later. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* (2006).

[71] Herlant, L. V., Holladay, R. M., and Srinivasa, S. S. Assistive teleoperation of robot arms via automatic time-optimal mode switching. In *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction (HRI)* (2016).

[72] Hiatt, L. M., Narber, C., Bekele, E., Khemlani, S. S., and Trafton, J. G. Human modeling for human–robot collaboration. *The International Journal of Robotics Research 36*, 5-7 (2017), 580–596.

[73] Ho, M. K., Cushman, F., Littman, M. L., and Austerweil, J. L. Communication in action: Planning and interpreting communicative demonstrations. *Journal of Experimental Psychology: General* (2021).

[74] Ho, M. K., Littman, M., MacGlashan, J., Cushman, F., and Austerweil, J. L. Showing versus doing: Teaching by demonstration. *Advances in Neural Information Processing Systems 29* (2016), 3027–3035.

[75] Holladay, R. M., Dragan, A. D., and Srinivasa, S. S. Legible robot pointing. In *The IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)* (2014).

[76] Hsu, D., Kindel, R., Latombe, J.-C., and Rock, S. Randomized kinodynamic motion planning with moving obstacles. *The International Journal of Robotics Research 21*, 3 (2002), 233–255.

[77] Huber, L., Billard, A., and Slotine, J.-J. Avoidance of convex and concave obstacles with convergence ensured through contraction. *IEEE Robotics and Automation Letters 4*, 2 (2019), 1462–1469.

[78] Hüntemann, A., Demeester, E., Vander Poorten, E., Van Brussel, H., and De Schutter, J. Probabilistic approach to recognize local navigation plans by fusing past driving information with a personalized user model. In *2013 IEEE International Conference on Robotics and Automation* (2013), IEEE, pp. 4376–4383.

[79] Ito, M. Internal model visualized. *Nature 403*, 6766 (2000), 153–154.

[80] Ivanovic, B., Leung, K., Schmerling, E., and Pavone, M. Multimodal deep generative models for trajectory prediction: A conditional variational autoencoder approach. *IEEE Robotics and Automation Letters 6*, 2 (2020), 295–302.

[81] Jagacinski, R. J., and Miller, R. A. Describing the human operator's internal model of a dynamic system. *Human Factors 20*, 4 (1978), 425–433.

[82] Jain, A., and Kemp, C. C. EL-E: an assistive mobile manipulator that autonomously fetches objects from flat surfaces. *Autonomous Robots 28*, 1 (2010), 45–64.

[83] Jain, S., and Argall, B. Automated perception of safe docking locations with alignment information for assistive wheelchairs. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems* (2014), IEEE, pp. 4997–5002.

[84] Jain, S., and Argall, B. Grasp detection for assistive robotic manipulation. In *2016 IEEE International Conference on Robotics and Automation (ICRA)* (2016), IEEE, pp. 2015–2021.

[85] Jain, S., and Argall, B. Robot learning to switch control modes for assistive teleoperation. In *RSS 2016 Workshop on Planning for Human-Robot Interaction: Shared Autonomy and Collaborative Robotics* (2016).

[86] Jain, S., and Argall, B. Estimation of surface geometries in point clouds for the manipulation of novel household objects. In *Proceedings of the RSS 2017 Workshop on Spatial-Semantic Representations in Robotics, Cambridge, MA, USA* (2017), vol. 16.

[87] Jain, S., and Argall, B. Recursive Bayesian human intent recognition in shared-control robotics. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2018), IEEE, pp. 3905–3912.

[88] Javaremi, M. N., Young, M., and Argall, B. D. Interface operation and implications for shared-control assistive robots. In *2019 IEEE 16th International Conference on Rehabilitation Robotics (ICORR)* (2019), IEEE, pp. 232–239.

[89] Javdani, S., Srinivasa, S. S., and Bagnell, J. A. Shared autonomy via hindsight optimization. In *Proceedings of the Robotics Science and Systems (RSS)* (2015).

[90] Johnson, E. J., Shu, S. B., Dellaert, B. G., Fox, C., Goldstein, D. G., Häubl, G., Larrick, R. P., Payne, J. W., Peters, E., Schkade, D., et al. Beyond nudges: Tools of a choice architecture. *Marketing Letters 23*, 2 (2012), 487–504.

[91] Katz, S. Assessing self-maintenance: activities of daily living, mobility, and instrumental activities of daily living. *Journal of the American Geriatrics Society 31*, 12 (1983), 721–727.

[92] Kavraki, L. E., Kolountzakis, M. N., and Latombe, J.-C. Analysis of probabilistic roadmaps for path planning. In *Proceedings IEEE International Conference on Robotics and Automation (ICRA)* (1996), vol. 4, IEEE, pp. 3020–3025.

[93] Kelley, R., Tavakkoli, A., King, C., Nicolescu, M., Nicolescu, M., and Bebis, G. Understanding human intentions via hidden Markov models in autonomous mobile robots. In *Proceedings of the 3rd ACM/IEEE International Conference on Human Robot Interaction* (2008), ACM, pp. 367–374.

[94] Khansari-Zadeh, S. M., and Billard, A. Learning stable nonlinear dynamical systems with Gaussian mixture models. *IEEE Transactions on Robotics 27*, 5 (2011), 943–957.

[95] Khatib, O. Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research 5*, 1 (1986), 90–98.

[96] Kim, D.-J., Hazlett, R., Godfrey, H., Rucks, G., Portee, D., Bricout, J., Cunningham, T., and Behal, A. On the relationship between autonomy, performance, and satisfaction: Lessons from a three-week user study with post-sci patients using a smart 6dof assistive robotic manipulator. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (2010), IEEE, pp. 217–222.

[97] Kirk, D. E. *Optimal control theory: an introduction*. Springer, 1970.

[98] Kluzik, J., Diedrichsen, J., Shadmehr, R., and Bastian, A. J. Reach adaptation: what determines whether we learn an internal model of the tool or adapt the model of our arm? *Journal of Neurophysiology 100*, 3 (2008), 1455–1464.

[99] Kuffner, J. J., and LaValle, S. M. Rrt-connect: An efficient approach to single-query path planning. In *Proceedings of IEEE International Conference on Robotics and Automation (2000)* (2000), vol. 2, IEEE, pp. 995–1001.

[100] Lala, D., Inoue, K., and Kawahara, T. Smooth turn-taking by a robot using an online continuous model to generate turn-taking cues. In *2019 International Conference on Multimodal Interaction* (2019), pp. 226–234.

[101] LaPlante, M. P., et al. Assistive technology devices and home accessibility features: prevalence, payment, need, and trends. *Advance Data from Vital and Health Statistics* (1992).

[102] Lawitzky, M., Kimmel, M., Ritzer, P., and Hirche, S. Trajectory generation under the least action principle for physical human-robot cooperation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (2013).

[103] Li, Q., Chen, W., and Wang, J. Dynamic shared control for human-wheelchair cooperation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (2011).

[104] Li, S., Bowman, M., and Zhang, X. A general arbitration model for robust human-robot shared control with multi-source uncertainty modeling. *arXiv preprint arXiv:2003.05097* (2020).

[105] Liu, C., Hamrick, J. B., Fisac, J. F., Dragan, A. D., Hedrick, J. K., Sastry, S. S., and Griffiths, T. L. Goal inference improves objective and

perceived performance in human-robot collaboration. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems* (2016), International Foundation for Autonomous Agents and Multiagent Systems, pp. 940–948.

[106] Loewenau, J., Gresser, K., Wisselmann, D., Richter, W., Rabel, D., and Durekovic, S. Dynamic pass predictiona new driver assistance system for superior and safe overtaking. In *Advanced Microsystems for Automotive Applications*. Springer, 2006, pp. 67–77.

[107] Lopes, M., Melo, F., and Montesano, L. Active learning for reward estimation in inverse reinforcement learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (2009), Springer, pp. 31–46.

[108] Losey, D. P., Srinivasan, K., Mandlekar, A., Garg, A., and Sadigh, D. Controlling assistive robots with learned latent actions. In *2020 IEEE International Conference on Robotics and Automation (ICRA)* (2020), IEEE, pp. 378–384.

[109] Mainprice, J., Ratliff, N., and Schaal, S. Warping the workspace geometry with electric potentials for motion optimization of manipulation tasks. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2016), IEEE, pp. 3156–3163.

[110] Marr, D. *Vision: A computational investigation into the human representation and processing of visual information.* MIT Press, 1982.

[111] Masiero, S., Celia, A., Rosati, G., and Armani, M. Robotic-assisted rehabilitation of the upper limb after acute stroke. *Archives of Physical Medicine and Rehabilitation 88*, 2 (2007), 142–149.

[112] Matarić, M. J., Eriksson, J., Feil-Seifer, D. J., and Winstein, C. J. Socially assistive robotics for post-stroke rehabilitation. *Journal of NeuroEngineering and Rehabilitation 4*, 1 (2007), 5.

[113] Mele, C., Spena, T. R., Kaartemo, V., and Marzullo, M. L. Smart nudging: How cognitive technologies enable choice architectures for value co-creation. *Journal of Business Research 129* (2021), 949–960.

[114] Miller, C. A. The risks of discretization: what is lost in (even good) levels-of-automation schemes. *Journal of Cognitive Engineering and Decision Making 12*, 1 (2018), 74–76.

[115] Miller, L. M., and Murphey, T. D. Trajectory optimization for continuous ergodic exploration. In *American Control Conference (ACC)* (2013).

[116] Miller, L. M., Silverman, Y., MacIver, M. A., and Murphey, T. D. Ergodic exploration of distributed information. *IEEE Transactions on Robotics 32*, 1 (2016), 36–52.

[117] Monsell, S. Task switching. *Trends in Cognitive Sciences 7*, 3 (2003), 134–140.

[118] Muller, U., Ben, J., Cosatto, E., Flepp, B., and Cun, Y. L. Off-road obstacle avoidance through end-to-end learning. In *Advances in Neural Information Processing Systems* (2006), pp. 739–746.

[119] Mur-Artal, R., Montiel, J. M. M., and Tardos, J. D. Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics 31*, 5 (2015), 1147–1163.

[120] Murphy, K. P., and Russell, S. Dynamic Bayesian networks: representation, inference and learning.

[121] Nuttin, M., Vanhooydonck, D., Demeester, E., and Van Brussel, H. Selection of suitable human-robot interaction techniques for intelligent wheelchairs. In *Proceedings of 11th IEEE International Workshop on Robot and Human Interactive Communication* (2002), IEEE, pp. 146–151.

[122] Pan, A., Bhatia, K., and Steinhardt, J. The effects of reward misspecification: Mapping and mitigating misaligned models. *arXiv preprint arXiv:2201.03544* (2022).

[123] Park, D., Kim, H., and Kemp, C. C. Multimodal anomaly detection for assistive robots. *Autonomous Robots 43*, 3 (2019), 611–629.

[124] Pearl, J. *Causality*. Cambridge university press, 2009.

[125] Pentland, A., and Liu, A. Modeling and prediction of human behavior. *Neural Computation 11*, 1 (1999), 229–242.

[126] Pierella, C., Casadio, M., Mussa-Ivaldi, F. A., and Solla, S. A. The dynamics of motor learning through the formation of internal models. *PLoS Computational Biology 15*, 12 (2019), e1007118.

[127] Pierella, C., Sciacchitano, A., Farshchiansadegh, A., Casadio, M., and Mussa-Ivaldi, S. Linear vs non-linear mapping in a body machine interface

based on electromyographic signals. In *Proceedings of the IEEE International Conference on Biomedical Robotics and Biomechatronics (BioRob)* (2018), pp. 162–166.

[128] Pilarski, P. M., Dawson, M. R., Degris, T., Carey, J. P., and Sutton, R. S. Dynamic switching and real-time machine learning for improved human control of assistive biomedical robots. In *Proceedings of the IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)* (2012), IEEE, pp. 296–302.

[129] Pitzer, B., Styer, M., Bersch, C., DuHadway, C., and Becker, J. Towards perceptual shared autonomy for robotic mobile manipulation. In *2011 IEEE International Conference on Robotics and Automation* (2011), IEEE, pp. 6245–6251.

[130] Pons, J. L. Rehabilitation exoskeletal robotics. *IEEE Engineering in Medicine and Biology Magazine 29*, 3 (2010), 57–63.

[131] Rafferty, A. N., LaMar, M. M., and Griffiths, T. L. Inferring learners' knowledge from their actions. *Cognitive Science 39*, 3 (2015), 584–618.

[132] Ratliff, N. D., Issac, J., and Kappler, D. Riemannian motion policies. *arXiv preprint arXiv:1801.02854* (2018).

[133] Reddy, S., Dragan, A., and Levine, S. Where do you think you're going?: Inferring beliefs about dynamics from behavior. In *Proceedings of the Neural Information Processing Systems (NeurIPS)* (2018).

[134] Reddy, S., Dragan, A. D., and Levine, S. Where do you think you're going? inferring beliefs about dynamics from behavior. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems* (2018), pp. 1461–1472.

[135] Rimon, E., and Koditschek, D. E. Exact robot navigation using artificial potential functions. *IEEE Transactions on Robotics and Automation 8*, 5 (1992), 501–518.

[136] Robotics, C. S. S., Nørskov, M., et al. Nudging by social robots. *Culturally Sustainable Social Robotics: Proceedings of Robophilosophy 2020 335* (2021), 337.

[137] Ross, R. J., Shi, H., Vierhuff, T., Krieg-Brückner, B., and Bateman, J. Towards dialogue based shared control of navigating robots. In *International Conference on Spatial Cognition* (2004), Springer, pp. 478–499.

[138] Sadigh, D., Sastry, S. S., Seshia, S. A., and Dragan, A. Information gathering actions over human internal state. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2016), IEEE, pp. 66–73.

[139] Schaff, C., and Walter, M. R. Residual policy learning for shared autonomy. *arXiv preprint arXiv:2004.05097* (2020).

[140] Schilling, M., Kopp, S., Wachsmuth, S., Wrede, B., Ritter, H., Brox, T., Nebel, B., and Burgard, W. Towards a multidimensional perspective on shared autonomy. In *2016 AAAI Fall Symposium Series* (2016).

[141] Schöner, G. Dynamical systems approaches to cognition. *Cambridge Handbook of Computational Cognitive Modeling* (2008), 101–126.

[142] Schöner, G., and Spencer, J. *Dynamic thinking: A primer on dynamic field theory*. Oxford University Press, 2015.

[143] Sheridan, T. B., and Verplank, W. L. Human and computer control of undersea teleoperators. Tech. rep., Massachusetts Inst of Tech Cambridge Man-Machine Systems Lab, 1978.

[144] Simpson, R., Levine, S., Bell, D., Jaros, L., Koren, Y., and Borenstein, J. NavChair: An assistive wheelchair navigation system with automatic adaptation. In *Assistive Technology and Artificial Intelligence* (1998), vol. 1458 of *Lecture Notes in Computer Science*, pp. 235–255.

[145] Simpson, R. C. Smart wheelchairs: A literature review. *Journal of Rehabilitation Research and Development 42*, 4 (2005), 423.

[146] Simpson, T., Broughton, C., Gauthier, M. J., and Prochazka, A. Tooth-click control of a hands-free computer interface. *IEEE Transactions on Biomedical Engineering 55*, 8 (2008), 2050–2056.

[147] Skantze, G. Turn-taking in conversational systems and human-robot interaction: a review. *Computer Speech & Language 67* (2021), 101178.

[148] Slutzky, M. W., and Flint, R. D. Physiological properties of brain-machine interface input signals. *Journal of Neurophysiology 118*, 2 (2017), 1329–1343.

[149] Soleiman, P., Moradi, H., Mahmoudi, M., Teymouri, M., and Pouretemad, H. R. Teaching turn-taking skills to children with autism using a parrot-like robot. *arXiv preprint arXiv:2101.12273* (2021).

[150] Storms, J., Chen, K., and Tilbury, D. A shared control method for obstacle avoidance with mobile robots and its interaction with communication delay. *The International Journal of Robotics Research 36*, 5-7 (2017), 820–839.

[151] Storms, J. G., and Tilbury, D. M. Blending of human and obstacle avoidance control for a high speed mobile robot. In *American Control Conference (ACC)* (2014), IEEE, pp. 3488–3493.

[152] Strouse, D., Kleiman-Weiner, M., Tenenbaum, J., Botvinick, M., and Schwab, D. J. Learning to share and hide intentions using information regularization. *Advances in Neural Information Processing Systems 31* (2018), 10249–10259.

[153] SULLINS, J. P., and DOUGHERTY, S. Ethical nudging of users while they interact with robots. *Culturally Sustainable Social Robotics: Proceedings of Robophilosophy 2020 335* (2021), 346.

[154] Sutton, R. S., and Barto, A. G. *Reinforcement learning: An introduction.* MIT press, 2018.

[155] Taha, T., Miró, J. V., and Dissanayake, G. A pomdp framework for modelling human interaction with assistive robots. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (2011), IEEE, pp. 544–549.

[156] Tahboub, K. A. Intelligent human-machine interaction based on dynamic bayesian networks probabilistic intention recognition. *Journal of Intelligent and Robotic Systems 45*, 1 (2006), 31–52.

[157] Thaler, R. H. From cashews to nudges: The evolution of behavioral economics. *American Economic Review 108*, 6 (2018), 1265–87.

[158] Thaler, R. H., and Sunstein, C. R. Nudge: Improving decisions about health, wealth, and happiness, 2008.

[159] Thomaz, A. L., and Chao, C. Turn-taking based on information flow for fluent human-robot interaction. *AI Magazine 32*, 4 (2011), 53–63.

[160] Thomsen, B., Annaswamy, A. M., and Lavretsky, E. Shared control between human and adaptive autopilots. In *2018 AIAA Guidance, Navigation, and Control Conference* (2018), p. 1574.

[161] Thomsen, B. T., Annaswamy, A. M., and Lavretsky, E. Shared control between adaptive autopilots and human operators for anomaly mitigation. *IFAC-PapersOnLine 51*, 34 (2019), 353–358.

[162] Todorov, E. Optimality principles in sensorimotor control. *Nature Neuroscience 7*, 9 (2004), 907–915.

[163] Tomasello, M., and Carpenter, M. Shared intentionality. *Developmental Science 10*, 1 (2007), 121–125.

[164] Tomasello, M., and Moll, H. The gap is social: Human shared intentionality and culture. In *Mind the Gap*. Springer, 2010, pp. 331–349.

[165] Trautman, P. Assistive planning in complex, dynamic environments: a probabilistic approach. In *2015 IEEE International Conference on Systems, Man, and Cybernetics (SMC),* (2015), IEEE, pp. 3072–3078.

[166] Tsui, K. M., Flynn, K., McHugh, A., Yanco, H. A., and Kontak, D. Designing speech-based interfaces for telepresence robots for people with disabilities. In *2013 IEEE 13th International Conference on Rehabilitation Robotics (ICORR)* (2013), IEEE, pp. 1–8.

[167] Uno, Y., Kawato, M., and Suzuki, R. Formation and control of optimal trajectory in human multijoint arm movement. *Biological Cybernetics 61*, 2 (1989), 89–101.

[168] Van Beers, R. J., Haggard, P., and Wolpert, D. M. The role of execution noise in movement variability. *Journal of Neurophysiology 91*, 2 (2004), 1050–1063.

[169] Wang, W., Na, X., Cao, D., Gong, J., Xi, J., Xing, Y., and Wang, F.-Y. Decision-making in driver-automation shared control: A review and perspectives. *IEEE/CAA Journal of Automatica Sinica 7*, 5 (2020), 1289–1307.

[170] Wasson, G., Sheth, P., Alwan, M., Granata, K., Ledoux, A., and Huang, C. User intent in a shared control framework for pedestrian mobility aids. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2003).

[171] Watkins, C. J., and Dayan, P. Q-learning. *Machine Learning 8*, 3-4 (1992), 279–292.

[172] You, E., and Hauser, K. Assisted teleoperation strategies for aggressively controlling a robot arm with 2D input. In *Proceedings of Robotics: Science and Systems* (2012).

[173] Zhang, B., Holloway, C., and Carlson, T. A hierarchical design for shared-control wheelchair navigation in dynamic environments. In *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)* (2020), IEEE, pp. 4439–4446.

[174] Zhang, T., McCarthy, Z., Jow, O., Lee, D., Chen, X., Goldberg, K., and Abbeel, P. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)* (2018), IEEE, pp. 1–8.

[175] Ziebart, B. D., Maas, A. L., Bagnell, J. A., and Dey, A. K. Maximum entropy inverse reinforcement learning. In *AAAI* (2008), vol. 8, Chicago, IL, USA, pp. 1433–1438.