HAPTIC BEHAVIOR EVALUATION TOOLKIT FOR AUTONOMOUS GROUND ROBOT NAVIGATION

BY

DANIEL J. BROOKS
B.S. WEST VIRGINIA UNIVERSITY (2009)

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF MASTER OF SCIENCE
DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF MASSACHUSETTS LOWELL

Signature of Author: Dan Brooks Date: 5-28-13

Signature of Thesis Supervisor: ______________________
Holly Yanco, Ph.D.

Signatures of Other Thesis Committee Members: ______________________
Georges Grinstein, Ph.D.

____
Jill Drury, Sc.D.

Haptic Behavior Evaluation Toolkit for Autonomous Ground Robot Navigation

Daniel J. Brooks

August 22, 2013

Abstract

A significant amount of research has been conducted regarding the technical aspects of haptic feedback; however, the designing of effective haptic feedback behaviors is not yet well understood from a human-robot interaction perspective. Haptic shared control feedback behaviors for semi-autonomous ground mobile robots sometimes make use of control paradigms that do not appropriately map to the navigation or teleoperation tasks. Also, evaluation of haptic behaviors has not been systematic and often only demonstrates feasibility. As a result, it is difficult to compare between various techniques. We have designed a three-part open-source toolkit to facilitate the investigation of haptic feedback behaviors for navigating semi-autonomous ground robots. Our work consists of 1) a simple hardware modification to turn a popular haptic research device (the Phantom Omni) into a 2D joystick, 2) a ROS software stack for writing arbitrary haptic behaviors for our haptic joystick modification, and 3) a generic experimental design that can be used to evaluate different types of haptic feedback behaviors. Finally, we share our toolkit so that we, as a research community, can better understand users' perceptions and comprehension of haptic behaviors.

Acknowledgments

I would first like to thank Dr. Holly Yanco, not only for her guidance and inspiration throughout this process, but also for providing a creative environment to work in and amazing access to the resources necessary to carry out our research. I would also like to thank Dr. Jill Drury and Dr. Georges Grinstein for being on my thesis committee, and for the insights I have gained through conversation with them.

Next, I would like to thank everyone in the Robotics Lab - both for your friendships and for the help you have given me during this process. Thank you especially to Mike Lunderville for the many hours of work you dedicated to helping me build this system, and to Jordan Allspaw and Mikhail Medvedev for your help debugging problems and running experiments. I would also like to extend a special thanks to Munjal Desai for your friendship and advice. And last but not least, Kate Tsui, for always being there to help every step along the way, and without whom I would not be finished.

Finally, I would like to thank my family and in particular my mother, Lori, for her wisdom, love, patience, and encouragement.

This research was supported in part by the National Science Foundation (IIS-0905228) and the Army Research Office (W911NF-07-1-021).

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Autonomous and semi-autonomous mobile robots are often equipped with sophisticated sensors designed to provide the system with a model of its surrounding environment. This information can then be used for making task-related decisions and conveying information back to the operator. To date, autonomous systems tend to exceed at well defined tasks such as navigation, planning, and obstacle avoidance, usually in fairly structured environments. However, for many current mobile robotic systems, teleoperated control is still largely favored, in part due to a human operator's sophisticated ability to reason about unstructured environments [Sheridan, 1992]. Introducing varying levels of autonomy into a teleoperated system allows for a human operator to make high level decisions while leaving other tasks to the autonomy [Sheridan, 1978]. This shared control introduces its own set of issues which arise from the "out of the loop" problem [Kaber and Endsley, 1997]. That is, when the human operator does not understand why a part of an autonomous system is behaving in a particular manner, usually due to poor situation awareness [Endsley, 1996], to the extent that they are incapable of effectively intervening should it become necessary. Attempts have been made to correct these issues by displaying additional sensor and system state information on the operator control unit (e.g. [Yanco, Keyes, Drury, Nielsen, Few, and Bruemmer, 2007]). However, this strategy is limited due to the finite amount of visual information that an operator can interpret before he or she becomes cognatively saturated.

Shared control systems allow the operator to retain some level of control over the system, though the operator's input is processed, and, in some cases, modified by the autonomy. One typical use of shared control is collision prevention. In this type of shared control, the operator has full control over steering the robot until the robot is in danger of hitting a nearby object. Whenever this occurs, the autonomy overrides any actions by the user that would bring the robot closer to collision while still allowing all other actions by the operator (e.g., "safe mode" [Keyes, 2007; Bruemmer, Dudenhoeffer, and Marble, 2002]). While these systems strive to provide "best of both worlds" solutions, users often find them frustrating, particularly when the system modifies what feels like a direct command.

A robot with any level of autonomy must perform as the user expects from the beginning; it is easy for users to lose trust in a system, which is hard to regain [Desai, 2012]. Research shows there are a number of factors which influence a user's allocation of control to an autonomous robot system including:

- Familiarity [Steinfeld, 2011],
- Mental model [Carlson and Demiris, 2009; Steinfeld, 2011; Norman, 1994],
- Control [Steinfeld, 2011; Norman, 1994],
- Reliability [Steinfeld, 2011; Norman, 1994], and
- Trust [Desai, 2012].

Given the complexity of an autonomous system, it is easy to see why in some cases users would prefer the drawbacks of teleoperation over the benefits of autonomy, especially if the system's behavior does not match a user's expectations (e.g., [Carlson and Demiris, 2010; Lankenau, 2001; Parikh, Grassi Jr, Kumar, and Okamoto Jr, 2005; Viswanathan, Boger, Hoey, Elinas, and Mihailidis, 2007]). Haptic feedback could be used to help mitigate these issues by giving the user insight into the robot's state or intentions. Additionally, haptic technology can be used to create new shared control paradigms that more closely match a user's mental models of how the robot is controlled.

1.1 Haptic Feedback

The area of haptic feedback research is huge, even restricted to work related to robots; a query for the phrase "robot haptic feedback" in Google Scholar will return more than 27,000 results. The space is not only deep but also quite broad, spanning wildly different domains as varied as mechanical hardware designs and implementations, algorithms for remotely feeling surface textures with a robot arm, creating virtual fixtures for assisted manipulator control, and software based stability control systems inspired by physics. Further restriction of the space to the intersection with a query for "mobile robot navigation" yields over 1,500 academic articles. Despite the plethora of literature available, it is still unknown how effective mobile robot navigation with haptic feedback is, and it is difficult to compare the various strategies and implementations. In many cases, tests performed on these systems are little more than feasibility studies that demonstrate the device performing in the manner the authors describe. Even the most in-depth studies usually only measure the time to complete a task, and sometimes the number of collisions, by at most a handful of participants.

Research on mobile robot navigation with haptic feedback has not yet matured to the point where experiments are replicated and repeated by others. Unlike other fields, there are no frameworks or standard methods for evaluating the effects of haptic behaviors or implementations. While fields like computer vision and data mining make use of publicly available data sets for evaluation, the lack of consistency in evaluating haptic behaviors makes it extremely difficult to compare results between studies.

1.2 Problem Statement

For the purposes of this thesis, we focus on the task of operating an unmanned, remotely located ground robot capable of semi-autonomous navigation. Teleoperation [Sheridan, 1978] is typically accomplished using a proportional velocity control device and a live, first-person view of the robot's video feed. Although the concept seems simple, teleoperating a robot at this low level can be cognitively taxing, given the "soda straw" view [Voshell, Woods, and Phillips, 2005]. An operator's visual channel may be oversaturated, particularly in time- and safety-critical domains such as urban search and rescue; robot operators fixate on the video and ignore all other components of the graphical user interface [Keyes, Micire, Drury, and Yanco, 2010]. Like many other researchers, we believe that haptics can be used to provide an additional channel for feedback.

Although a significant amount of research has been conducted regarding the technical aspects of haptic feedback, designing effective haptic feedback behaviors to represent the robot's current state is not yet well understood from a human-robot interaction (HRI) perspective. Further, the way in which users understand and perceive these haptic behaviors is even less well understood. In this thesis, we address the design of a three-part open-source toolkit for investigating haptic behaviors for teleoperating or navigating semi-autonomous ground robots. Our toolkit includes:

- A simple, non-destructive "do-it-yourself" hardware modification to turn a popular haptic research device (the SensAble Phantom Omni) into a 2 degree of freedom (DOF) haptic joystick,
- A ROS software stack for implementing arbitrary behaviors on our haptic joystick modification, and
- A generic experimental design flexible enough to evaluate different haptic feedback behaviors as well as investigate the effects of haptic feedback on user interactions.

1.3 Organization

We begin by briefly surveying the field of haptics and examining the current problems relating to evaluating haptic feedback behaviors for teleoperating or navigating semi-autonomous ground robots in chapter 2. Next, we address the issue of creating an easily reproducible haptic interface appropriate for the task of driving ground robots (chapter 3) and a corresponding software stack for writing haptic behaviors (chapter 4), the combination of which constitutes the first two parts of our toolkit. We then introduce our experiment methodology in chapter 5, which is the third and final component of the toolkit. We discuss one haptic feedback behavior implemented for shared control and the corresponding experiment we conducted in chapter 6, which leveraged all three components of our toolkit. Finally, we conclude by looking at the open research questions and interesting future applications of our work (chapter 7).

Chapter 2

Related Work

Haptic interfaces have been used to control many robotic devices including rotary cranes [Takemoto, Miyoshi, Terashima, and Yano, 2004], wheelchairs [Luo, Hu, Chen, and Lin, 1999], and surgical equipment [Okamura, 2004]. Many different haptic control and feedback strategies have been offered up, and feasibility studies for these methods have been conducted. Unfortunately, these studies often do not provide sufficient insight for determining the most appropriate strategy for any given task.

Environmental feedback strategies try to let users "feel" their surroundings by having nearby objects emit force which is rendered haptically. This force could be a representation of the actual force the remote system is exerting on an object it is in contact with, or it could be a "force field" generated around sensed objects (also known as artificial force reflection [Hong, Lee, and Kim, 1999]) to prevent contact. Alternatively, behavioral feedback strategies strive to use haptic force to represent the state or intentions of the remote system [Barnes and Counsell, 2003]. Position-position control is a control strategy in which the position of a master device is mapped directly (sometimes with scaling) to a remote slave device position. Position-velocity control is control strategy in which the position of a master device is mapped to velocity information for controlling the movement of a remote slave device [Farkhatdinov and Ryu, 2010b]. All of these methods have been used for controlling mobile ground robots (see Table 2.1, Table 2.2, and Table 2.3). These control strategies are discussed further in section 2.3.

There has been a great deal of work in the field of haptic devices and interfaces, especially with respect to research dedicated to keeping haptic interface systems passive, or stable (e.g. [Farkhatdinov and Ryu, 2010b; Elhajj, Xi, and Liu, 2000; Hannaford and Ryu, 2002; Niemeyer and Slotine, 1991]). Stability is a common problem found in bilateral control systems caused by time-delay occurring in network transmissions, which destabilizes the feedback loop between master and slave devices. Other haptic research has focused on building or implementing custom haptic devices (e.g. [Takemoto et al., 2004; Cho, Jin, Lee, and Yao, 2010; Han and Lee, 2007]). Custom design can be quite challenging and time consuming as it requires building hardware, electronics, and software systems. However, relatively little of this work has concentrated on how haptic interfaces impact end users' interactions with the system, also called human-robot interaction (HRI), especially in the case of mobile ground robots. In this chapter, we present an overview of relevant haptics research as it relates to teleoperation and navigation of mobile ground robots.

2.1 Definition and Types of Haptic Devices

Merriam-Webster defines the word "haptic" as "relating to or based on the sense of touch." It is used in the term "haptic interfaces" in the field of human-computer interaction to describe mechanical devices that allow users to kinesthetically interact with an environment. Hayward, Astley, Cruz-Hernandez, Grant, and Robles-De-La-Torre

Table 2.1: Survey of Mobile Ground Robots with Haptic Behaviors (Part 1/3)

Paper	Type*	Haptic Feedback Behavior	Device	Autonomy	Study
[Nguyen and Ryu, 2012]	E-A	Force based on steering angular velocity and speed of vehicle	Phantom Omni vs. custom	teleop	n=4; Compared position-velocity controls
			Logitech Joystick vs. custom		n=4; Compared position-velocity controls
	B	Cornering feel to assist driver	"conventional steering wheel" vs. custom		n=7; Compared interfaces with cornering feel
[Farkhatdinov and Ryu, 2010b]	E-A	Reflective force linearly increases with proximity to obstacle	Phantom Premium	teleop	n=5; Feasibility study comparing non-variable vs. variable haptic feedback
[Cho et al., 2010]	E-A	Reflective force inversely proportional to distance from obstacle	custom	teleop	n=3; Feasibility study comparing joysticks controlled by two-axis hall sensor vs. potentiometers
[Farkhatdinov et al., 2009]	E-A and B	Reflective force inversely proportional to distance from obstacle	Phantom Premium	teleop	n=5; Feasibility study comparing position-position control vs. position-speed control

*E = environmental haptic effect; E-A = artificial force feedback; E-I = impact; B = Behavioral haptic effect

Table 2.2: Survey of Mobile Ground Robots with Haptic Behaviors (Part 2/3)

Paper	Type*	Haptic Feedback Behavior	Device	Autonomy	Study
[Schill et al., 2008b]	B	Joystick fixed in forward position with reflective force inversely proportional to distance from obstacle	Falcon 3D	safe	Feasibility study using spherical optical flow to provide haptic feedback to a joystick
	E-A	Joystick directed by user with reflective force specified by spherical optical flow			
[Farkhatdinov et al., 2008]	E-A	Reflective force inversely proportional to distance from obstacle	Phantom Premium	teleop	n=5; Feasibility study comparing position-position vs. position-speed control
[Mullins et al., 2007]	B	Provides Feedback as a function of motor current draw and IMU data	Phantom Omni	teleop	n=10 (5 skilled, 5 unskilled); Compared the use of "gravwell" feedback to navigate a course
[Mitsou et al., 2006]	E-A	Reflective force inversely proportional to distance from obstacle	Phantom Omni	safe	n=36; Compared haptic feedback vs. no feedback in mazes both with and without visual feedback
[Lee et al., 2005]	E-A	Reflective force inversely proportional to distance from obstacle	Phantom 1.5	teleop vs. safe	n=20; Compared solely haptic feedback vs. haptic feedback and anti-collision operation

*E = environmental haptic effect; E-A = artificial force feedback; E-I = impact; B = Behavioral haptic effect

Table 2.3: Survey of Mobile Ground Robots with Haptic Behaviors (Part 3/3)

Paper	Type*	Haptic Feedback Behavior	Device	Autonomy	Study
[Kim et al., 2004]	E-I	Reflective force proportional to the force of contact on either right or left driving track	2 "haptic joysticks" (models unknown)	teleop	none
[Barnes and Counsell, 2003]	B	Reflective force inversely proportional to distance from obstacle	Immersion IE2000	shared	$n = 10$; Compared teleop, safe, and shared levels of autonomy in a driving course
	E-A			safe	
	E-A			teleop	
[Diolaiti and Melchiorri, 2004]	E-A	Reflective force inversely proportional to distance from obstacle	Phantom (unknown model)	teleop	Comparison of operation with and without haptic feedback
[Rösch et al., 2002]	E-I	Reflective force proportional to force on front bumper of vehicle upon impact	Immersion IE2000	teleop	none
[Luo et al., 1999]	E-A	Reflective force proportional to distance from obstacle; rumble when "encountered expected difficulty"	Microsoft Force Feedback Pro Joystick	teleop	Showed that blind users could effectively navigate hallways using haptic feedback
[Hong et al., 1999]	E-A	Reflective force proportional to distance from obstacle	Immersion Joystick (unknown model)	teleop	none

*E = environmental haptic effect; E-A = artificial force feedback; E-I = impact; B = Behavioral haptic effect

[2004] goes on to further define haptic interfaces as "being concerned with the association of gesture to touch and kinesthesia to provide for communication between the humans and machines," and describes them as being broken down into two categories (Passive and Active) as follows:

> Passive devices are often designed to have programmable dissipation, as a function of position or time. To this category belong the devices having controllable brakes. Another category of passive devices consists of those that rely on non-holonomic constraints (constraints involving velocity). Yet another possibility is to modify the elastic behavior of an element to become harder or softer. The programmability of passive devices comes from the possibility of modifying these constraints under computer control.
>
> As for active devices, the energy exchange between a user and the machine is entirely a function of the feedback control which is applied. Then two categories arise: either the actuators act as a force source (a variable of effort), and position is measured, or the actuators act as a position source and then force is measured. The former case is termed isotonic (force does not change with position) while the latter is called isometric (position does not change with force). Closing the loop around an isotonic device corresponds to specifying an impedance to produce a simulation, and the other case corresponds to an admittance [Hayward et al., 2004, p. 18].

In this work, the term "haptic device" is generally used to refer to programmable devices which are articulated using electric motors, and therefore fall into the category of "active" devices. However, the distinction between passive and active devices is important, since many physical interfaces such as spring centering joysticks and push buttons can be considered haptic interfaces, though dramatically different from the programmable machines we refer to though out this work.

2.2 Bilateral Teleoperation and Passivity

Teleoperation can be defined as human operator controlling a remote slave device through the use of a local master device. Such control is characterized as being bilateral when the master device renders local "feedback" forces to the human operator based on information being sent back from the slave device. Extensive work has been done in the area of bilateral control with respect to teleoperation of mobile robots, arms, and manipulators. Much of this work has focused on passivity, or the property that a system does not generate energy, since introducing energy to the system causes such systems to become unstable. One common source of such instability in bilateral control systems is transmission delay. In 1989, Anderson and Spong [1989] first presented a solution to this problem based on passivity and scattering theory by presenting a control law which could be used to overcome instability. Niemeyer and Slotine Niemeyer and Slotine [1991] introduced the concept of using wave variables to characterize time delay systems and prevent instabilities when constant length time delays are present. Lee, Martinez-Palafox, and Spong [2006] proposed a framework for passivity in a system where a mobile robot used local planning and used a SensAble Phantom Omni to control both simulated and real robots. Kanno and Yokokohji [2012] used wave variables for multilateral control in which multiple masters controlled a single slave arm. They performed a feasibility experiment in which two operators (a leader and a follower) used separate devices (SensAble Phantom Omnis) to feel remote objects (corrugated panel) with a remote arm (a Novint Falcon). Because of its complexity and practicality, the issue of passivity continues to generate a significant amount of literature as researchers explore the problem from different angles (e.g., [Lee et al., 2006; Adams and Hannaford, 1999; Kanno and Yokokohji, 2012; Çavusoglu, Sherman, and Tendick, 2002; Polushin, Liu, and Lung, 2007]).

Passivity remains an import topic of research in the area of bilateral control, since future practical deployment of such systems will depend on reliable, stable operation. However, we consider the issue of passivity to be out of the scope of this work. We circumvent one of the most common sources of instability, network latency, by assuming that the research being conducted is capable of being performed over a high speed local network with a single wireless hop to the robot, in an environment where network traffic and wireless interference can be controlled.

2.3 Control Paradigms

Teleoperation can be broken into two paradigms, position-position and position-velocity control, neither of which are inherently specific to haptic interfaces. In position-position control, the position of the master device maps directly to the position of the slave device, and is primarily used for controlling slave devices with a fixed range of motion, such as robot arms. In this paradigm, movement of the slave device is triggered when the position of the master device changes. When this occurs, a goal position for the slave is generated based on the last position of the master. If the master's position changes again before the slave has reached the goal position, a new goal position is generated based on the updated master position. It is not necessary for both the master and slave devices to be physically similar, although typically the two devices will have similar degrees of freedom. Linear scaling can be applied to control a large manipulator with a miniature version acting as the master device, or to gain precision while controlling a slave device with a smaller workspace then the master.

Position-velocity control refers to the position of the master mapping to a velocity parameter of the slave and is useful for controlling slave devices that have an unbounded range of motion, such as mobile robots. In this paradigm, the master device usually has a reference point against which its position is measured. This measurement is then used to produce a velocity which the slave device attempts to obtain. Position-velocity control is commonly associated with joystick devices which are used to drive skid steer vehicles such as those commonly found on mobile robot platforms.

Haptic feedback can be loosely broken into two overlapping categories, environmental and behavioral feedback. Environmental feedback can be characterized by haptic forces representing remote objects sensed in the environment. The objects could either be real objects sensed by physical contact or virtual objects emitting virtual forces.

Feedback forces which originate from physical contact, or *impact*, may be used to portray pressure being applied by the remote manipulator to an object in the remote environment, or to simulate an object's surface texture. Forces which originate from virtual objects, also called *artificial forces*, are often used to create effects such as force fields or magnetic attractions. Artificial forces are often used to convey a sense impending contact with an actual object.

Behavioral feedback is a less common form of feedback in which forces originate from an autonomous behavior. Behavioral feedback can be used to represent either the state or intentions of the slave device, such as pushing a joystick sideways to indicate the robot wanting to turn towards a waypoint. However, the distinction can be lost in some instances, especially when behaviors are closely tied to the sensing of the environment. For example, creating attractive forces to snap a user's hand to an imaginary magnetic line could either be considered environmental feedback if the line is considered a static imaginary object within the environment, or behavioral feedback if thought of as a dynamic path generated by an autonomous motion planner.

In 1995, Rosenberg [1995] published a paper on the use of virtual fixtures to enhance the performance of a teleoperated task. Human subjects were asked to use a robotic exoskeleton to remotely control a robot arm to place a peg into a board. Several different forms of virtual fixtures were tested against the baseline of using no virtual fixtures to see how they effected the users performance. Virtual fixtures tested included intersecting virtual surfaces, uniform damping fields, variable damping fields, snap-to virtual planes, and snap-to virtual lines. Rosenberg found that all virtual fixtures tested resulted in significant performance improvements over the non-fixture trials, crediting reduced modality as the most likely explanation for the improvement. Snap-to-lines, virtual lines along which the operator's hand would "snap" to by an attractive force when he or she neared the line, resulted in the greatest performance improvement of all virtual fixtures tested, even though they "employ an abstract haptic percept which has no real-world analog."

Takemoto et al. [2004] and Yano, Takemoto, and Terashima [2005] built a custom haptic joystick for controlling a rotary crane. The joystick was built from two AC Servo Motors with harmonic drive and torque sensors attached to the rotational axis. Position-velocity control was used to control the crane, rather than position-position control. A PD (Proportional-Derivative) controller is used to restrict velocity and enforce obstacle avoidance for static obstacles known a priori for both the boom and the rotational components of the crane.

Okamura [2004] discusses haptic technology used in robot-assisted surgery. One use of haptic feedback comes in the form of providing tactile feedback while tying knots.

The level of tension for such knots is extremely important, as is the ability for the remote operator to feel the tension being applied.

Barnes and Counsell [2003] conducted experiments to test the following three hypotheses: (1) that haptic feedback would improve an operator's performance of a teleoperation task, (2) the use of semi-autonomous behaviors would improve the performance of a teleoperation task, and (3) that providing haptic feedback and semi-autonomous behaviors simultaneously would yield the best performance of a teleoperation task. They conducted a series of tests which implemented various configurations including haptic and/or semi-autonomous behaviors in which a virtual robot was controlled using an Immersion Corporation Impulse Engine 2000 haptic joystick. Two different forms of haptic feedback were tested, environmental haptic feedback and behavioral haptic feedback. Environmental haptic feedback was performed by detecting obstacles.

Mitsou et al. [2006] constructed an interface consisting of a 2D haptic joystick model (implemented using a SensAble Phantom Omni device) and overhead view of the the robot and explored environment. The joystick directions correspond to the direction in which the operator wishes the robot to move on the screen - "up," "down," "left," and "right" (much the same way Pac-Man video games are controlled). The robot would attempt to follow the simple commands (those which did not combine the up, down, left, and right actions) until it encountered an obstacle. Obstacles were dealt with by stopping the robot and ignoring any commands that would cause the collision while simultaneously allowing the user to "feel" the obstacle through the haptic feedback. Combined commands, or commands which combined two different directions, were dealt with in a similar fashion. When an obstacle was detected in one of the two directions in which the joystick was being pushed, the direction corresponding to the obstacle would push the joystick back toward the center to inform the operator that the robot could not move in that particular direction. Two experiments were performed with thirty-six participants being asked to navigate a series of equivalently difficult maze environments. In the first experiment, the time to exit the maze was used to measure the effectiveness of the combined command haptic joystick and the same joystick that did not incorporate haptic feedback. The results of this experiment showed that the efficiency was not inhibited by the addition of haptic feedback. In the second experiment participants were asked to "drive-by-feel," without additional visual aid. The participants were able to successfully navigate the mazes (although it took longer to do so), indicating that the haptic feedback in this case was sufficient to understand the spacial layout of the world.

Schill, Mahony, Corke, and Cole [2008a] looked at using optical flow to generate force feedback for controlling their robot, "InsectBot." A Novint Falcon 3D joystick was used as the master device for position-velocity control. Spherical divergence motion cues were used to create forces proportional to an estimated time-to-contact to alert operators to potential collisions. Using comparative flow, forces were generated to act orthogonally to the direction of the robot, causing the robot to automatically center itself between obstacles. The authors conducted several experiments in which the robot was driven down a 1.5m wide corridor with added obstacles to force the robot to drive in "S" curves. Their experiments were centered around testing the functionality of the interface, rather then the user experience, and only showed that their control system was dissipative and prevented their robot from colliding with the environment.

In this chapter, we examined previous work from various areas within the field of haptics research. The majority of this work has focused on the technical aspects of haptic feedback. However, before we can fully leverage the capabilities of this technology, we need to better understand the impact haptic feedback behaviors have on operators from a HRI perspective.

Chapter 3

Hardware Design

A large portion of the research concerning haptic feedback control of mobile ground robots has been performed by researchers using the SensAble Phantom Omni haptic device. The Phantom Omni consists of a 6 degree of freedom arm connected to a stylus which can be held in the user's hand like a pen. All of the research using this device has been performed using the stylus as the interface for steering or navigating the robot. Using such an unorthodox device for controlling the robot makes it both difficult to understand the consequences of the haptic effects and to compare the haptic interface with the ubiquitous joystick interface traditionally used for the task.

3.1 Joystick Types

Joysticks provide a particularly good affordance for control of mobile ground robots since the degrees of freedom provided by a joystick are easily mapped to the translation and rotation movements of the robot. Various types of joystick devices have been used for teleoperating robots, including hand grips, finger controls, thumb controls, and pucks. Hand grip joysticks (also known as flight sticks) are typically larger units which can have 1 (throttles), 2 (joystick), or 3 (such as Novint Falcon) DOF and also typically contain multiple buttons (see Figure 3.1a); manipulating this style of joystick is usually done by articulating the wrist along with slight movements of the arm. Grip joysticks have been used extensively for controlling aircraft and large machinery, in addition to being used for teleoperating robots.

Finger operated joysticks are much smaller than hand grips. These arcade style joysticks come with a wide array of body sizes and styles and occasionally include a button on the top (see Figure 3.1b). Finger joysticks are used by gripping the joystick between the thumb and forefingers, and are commonly built into tabletop operator control units (OCU) or handheld units.

Thumb controls are similar, but are designed to be toggled using just the thumb, and thus may be even smaller than finger joysticks (see Figure 3.1c). These are most commonly found built into video game controls and have seen a recent increase in use for controlling robots since younger generations are very familiar with using these devices.

Finally, pucks are 6 DOF devices (e.g., 3D Connection Space Navigator) commonly used for "flying" a camera view about in exploring a 3D environment such as Google Earth (see Figure 3.1d). Pucks have been used for controlling robots such as the iRobot Packbot in the past, but typically require training to learn to use them effectively. In the Packbot's case, iRobot replaced the puck system with an xBox video game controller which uses thumb controls to operate the robot.

All of these devices inherently have mechanical haptic feedback (meaning that the user is aware of the position in which he or she is holding the device) but only a few have active haptic feedback (usually in the form of a rumble pack). Fewer still are capable of providing discrete haptic forces in multiple degrees of freedom, such as the

(a) Flightstick [1] (b) Finger Joysticks[2] (c) Thumb Controls[3] (d) 6DOF Puck[4]

Figure 3.1: Types of Joystick Devices

discontinued Microsoft Sidewinder Force Feedback joystick.

3.2 Design Considerations

Constructing a robust haptic device from scratch is a non-trivial task; as evident from it being the focus of much research (e.g. [Hayward et al., 2004; Bae, Koo, and Park, 2001; Cho et al., 2010]). The emergence of the 6 DOF SensAble Phantom [SensAble Technologies Inc., 2011] and 3 DOF Novint Falcon [Novint Technologies Inc., 2013] have allowed many more researchers to conduct haptic related research. Researchers have investigated interface design and the effects of haptic feedback with respect to mobile robots using these unmodified commercial off the shelf (COTS) products (e.g., [Schill et al., 2008a; Farkhatdinov and Ryu, 2010a; Diolaiti and Melchiorri, 2003; Mitsou et al., 2006]). These products provide a very convenient alternative for researchers not wishing to develop sophisticated custom hardware devices, and also offer the advantage of support communities and software APIs [SensAble Technologies Inc, 2011; Kyle Machulis, 2010]. However, using a higher DOF device such as a 6 DOF stylus for

[1] [cheapgamehardware.blogspot.com, 2013]
[2] [www.directindustry.com, 2013]
[3] [www.digikey.com, 2013]
[4] [www.trinity3d.com, 2013]

(a) SensAble Phantom Omni[5]

(b) Novint Falcon[6]

Figure 3.2: Commercial Haptic Joysticks

controlling a robot only capable of 2D movement may not be as suitable as using a more traditional and ubiquitous 2 DOF input device such as a joystick [Bowman, Coquillart, Froehlich, Hirose, Kitamura, Kiyokawa, and Stuerzlinger, 2008; Lapointe, Savard, and Vinson, 2011].

With this in mind, we set out to create a 2DOF haptic joystick system specifically designed for the tasks of mobile ground robot teleoperation and navigation that could be shared with other researchers wishing to investigate haptic control of these robots. To facilitate the design process of our system, we considered both physical characteristics and design complexity.

3.2.1 Physical Characteristics

The joystick itself should have the same characteristics as commercial 2 degree of freedom (DOF) joystick products typically used for controlling robots. Traditional 2 DOF joysticks inherently provide users with two forms of haptic feedback - spring centering and hard stops.

The spring force is felt as a small amount of constant pressure as the springs inside

[5] [perso.limsi.fr, 2013]
[6] [www.slipperybrick.com, 2013]

the base of the unit push the handle back towards the center. While small, this pressure provides several important functions. First, feeling any amount of pressure informs the user that the joystick is not centered. Next, the direction of the pressure is inversely related to the direction in which the joystick is being pressed. For example, as the user presses the joystick towards the left, the spring will press back towards the right. Finally, the joystick will "automatically" center itself when the user relaxes their pressure on the unit.

The physical nature of mechanical joysticks also provides a limited range of motion in which the device can be moved. Thus, the user is informed when the device is at its limit because the device comes to a hard stop. During past studies, we have observed that people tend to use this physical boundary to simplify their driving task. The technique of pressing the joystick to its physical limit essentially lowers the dimensionality of the device by keeping the joystick a constant distance away from the center.

Both the spring centering force and hard stops present in mechanical joysticks are important haptic paradigms that needed to be present in the design of our haptic joystick. The base use case scenario for our haptic joystick is for it to be usable as a regular joystick device. Ideally, by designing our haptic joystick to have the same physical parameters as a popular commercial product, a user would not be able to tell the difference between our haptic version (without special haptic effects) and the original product on which it was based. In other words, interacting with the haptic joystick should feel just like a traditional joystick, but with the ability to provide additional information in a haptic manner.

An important consequence of this approach is that the same device can be used to test multiple interface conditions (e.g. haptic and non-haptic). This is especially useful when trying to differentiate between the effects haptic forces play on an interaction from the effects of the software which generates them.

3.2.2 Design Complexity

Our goal was to create an easily reproducible and programmable haptic joystick, such that it could be shared with other researchers who are primarily interested in studying the effects of haptic feedback rather than the devices themselves. It is likely that these researchers already use a COTS product in their research, thus it made sense that a COTS product could be leveraged as a base to build our interface. A COTS product has the advantage of already possessing the necessary computing hardware and software for communicating with the hardware in place.

We decided that all additional parts needed to build the joystick should be readily obtainable. The number of custom components that needed to be manufactured must be kept to a minimum, and those parts should be both easy to make and also easily acquired online. We also specifically wanted to keep our design non-destructive in nature. That is, any modifications made to the haptic device should be completely reversible such that the original device can be put back together without any damage.

Finally, we wanted any custom software written for the haptic joystick to be easy to integrate into existing projects and codebases. As a large portion of the robotics community has embraced OSRF's open source Robot Operating System (ROS) [Quigley, Gerkey, Conley, Faust, Foote, Leibs, Berger, Wheeler, and Ng, 2009], it seemed an appropriate choice for our system. We further discuss our software in chapter 4.

3.3 Haptic Engine

As previously mentioned, building a custom haptic device is a difficult process that requires knowledge and skills in mechanical engineering and electrical design. We decided to approach the problem by building an adapter for an existing COTS product, which reduces complexity and is far easier to replicate.

There were two commercially available haptic platforms that were taken into consideration. First, the Novint Falcon 3D is marketed as a video game joystick. The end effector has a workspace of 4" $\times$ 4" $\times$ 4" and can generate more than 2lbs of force (lbf). The device can be used with Linux, and an open source API for interfacing with the device is available online. The second device considered was the SensAble Phantom Omni. The Omni's workspace measures 6.4" $\times$ 4.8" $\times$ 2.8", with a range of motion well within the user's hand movement when pivoting at the wrist. It has position resolution of greater then 450 dpi and can exert a maximum of 0.75lbf. The Omni can be accessed using the well documented OpenHaptics API.

We elected to use the SensAble Phantom Omni (Figure 3.2a). The Omni was primarily designed for use with simulations and virtual reality and has seen widespread use in haptics research, including control of mobile robots, thus making it an ideal platform on which to build our system. The main disadvantage of this device with respect to our research is that its many degrees of freedom and stylus grip are unsuitable for use as a traditional 2 DOF joystick, as discussed previously. As a result, one of the primary goals of our modification was to reduce the degrees of freedom. Another disadvantage of this device is its size, which is quite bulky compared to most traditional joysticks.

The device comes with the OpenHaptics Toolkit API [SensAble Technologies Inc, 2011], which we discuss in section 4.1. As mentioned earlier, the Phantom was designed for use in virtual environments. Because of this, there are several pitfalls to using the OpenHaptics API to control the arm in the real world, which if not handled appropriately can cause unexpected problems in calculations done on points in the real world (further discussed in chapter 4).

3.4 Joystick Arm Prototype

We selected a C.H. Products Flightstick (see Figure 3.3) on which to model our haptic joystick [Brooks and Yanco, 2012]. This joystick model is relatively common, mechanically simple, and inexpensive. This no-frills product features two push buttons, which could be incorporated into the haptic device. The simple design meant that the joystick grip could easily be removed from the rest of the unit and directly incorporated as part of the haptic joystick.

Figure 3.3: CHProducts Flightstick[7]

In order to convert the 6 DOF Phantom Omni into a 2 DOF device, we designed a modification in which an arm rotating about a pivot point connected the Omni's end effector on one side and the joystick grip on the other. This design constrained the Omni's end effector to move along the surface of a sphere defined by the pivot point as

[7][www.amazon.com, 2013]

its center and arm length as its radius (see Figure 3.4). The body of the grip was large enough that we could mold one end of the arm to fit inside the grip, creating a strong joint.

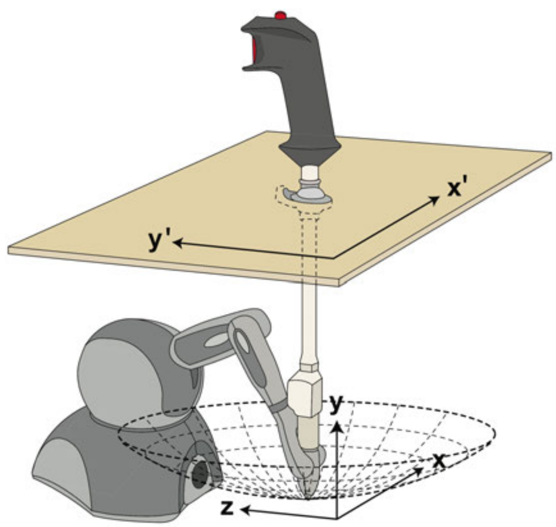


Figure 3.4: Joystick Modification Setup. The Omni end effector is constrained to moving along the surface of a sphere.

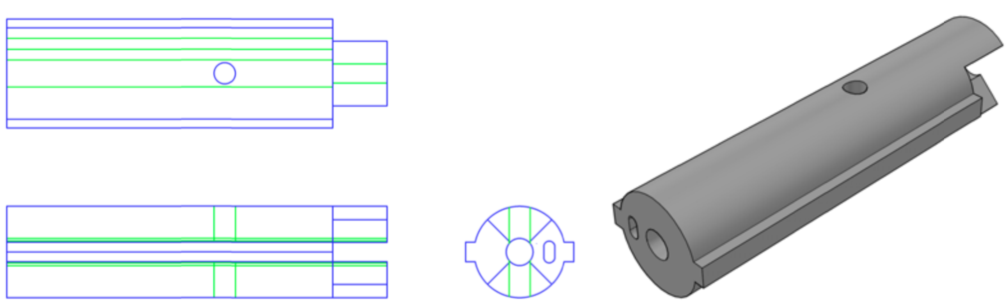


Figure 3.5: Prototype Joystick Grip Insert

The Phantom Omni has a removable stylus attachment connected to the end effector. The stylus can be removed, revealing a 1/4" stereo male barrel jack which can be used to attach other interfaces. We designed an adapter for the arm which would allow us to connect the push buttons on the joystick grip directly to the haptic device.

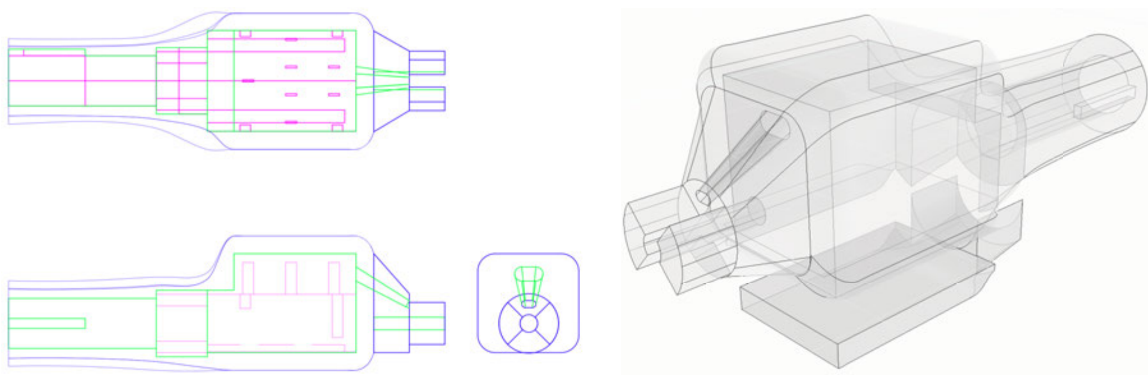


Figure 3.6: Prototype Haptic Adapter

We designed an acrylonitrile butadiene styrene (ABS) plastic shaft that could connect the joystick adapter to the haptic adapter. The arm itself needed to be very rigid so as not to flex when force was generated by the Omni or applied by the user. The shaft was hollow, so that an 1/8" steel rod could be be inserted down the center of the assembly, adding to the rigidity.

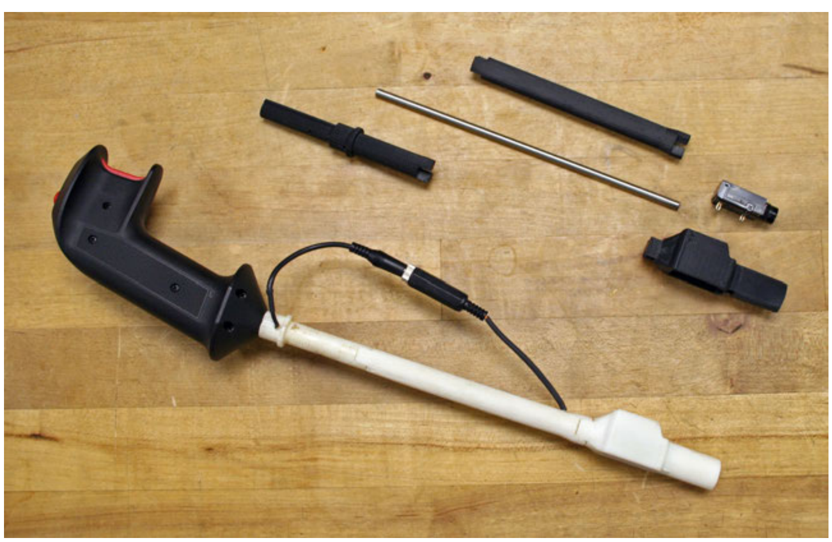


Figure 3.7: Prototype Haptic Joystick Arm

One of our goals was to match the range of motion of the joystick grip to its original range of motion, and simultaneously maximize the range of the haptic end effector (see Figure 3.8). This design requirement had the effect of maximizing the mechanical advantage of the haptic device while simultaneously maximizing the resolution at which it could read the position of the joystick grip. We purchased a ball joint rod end lined with Polytetrafluoroethylene (PTFE, also known as Teflon) that could swivel a maximum of 65 degrees to act as the pivot point, closely approximating the original range of motion of the CHProducts Flightstick.

After completing construction of the initial prototype we were able to make some key observations. The style of grip we had selected encouraged people to hold the joystick in such a way that manipulating it required moving the wrist. Additionally, people tended to grasp very high up on the grip, minimizing the mechanical advantage of the joystick arm. This combination made it very difficult to feel the haptic forces from the device.

3.5 Final Joystick Design

Based on lessons learned from building our prototype, we designed our final joystick. An informal survey of commercial robots designed for military and police use found that finger and thumb style joysticks were the most commonly used style of joysticks for teleoperating mobile robots, and would be a better model for our haptic device.

3.5.1 Model Joystick

Thumb controls have recently seen an increase in popularity for operating robots, which take advantage of younger generation's familiarity with these controls from playing video games. These devices are usually built into handheld controllers (e.g., xBox,

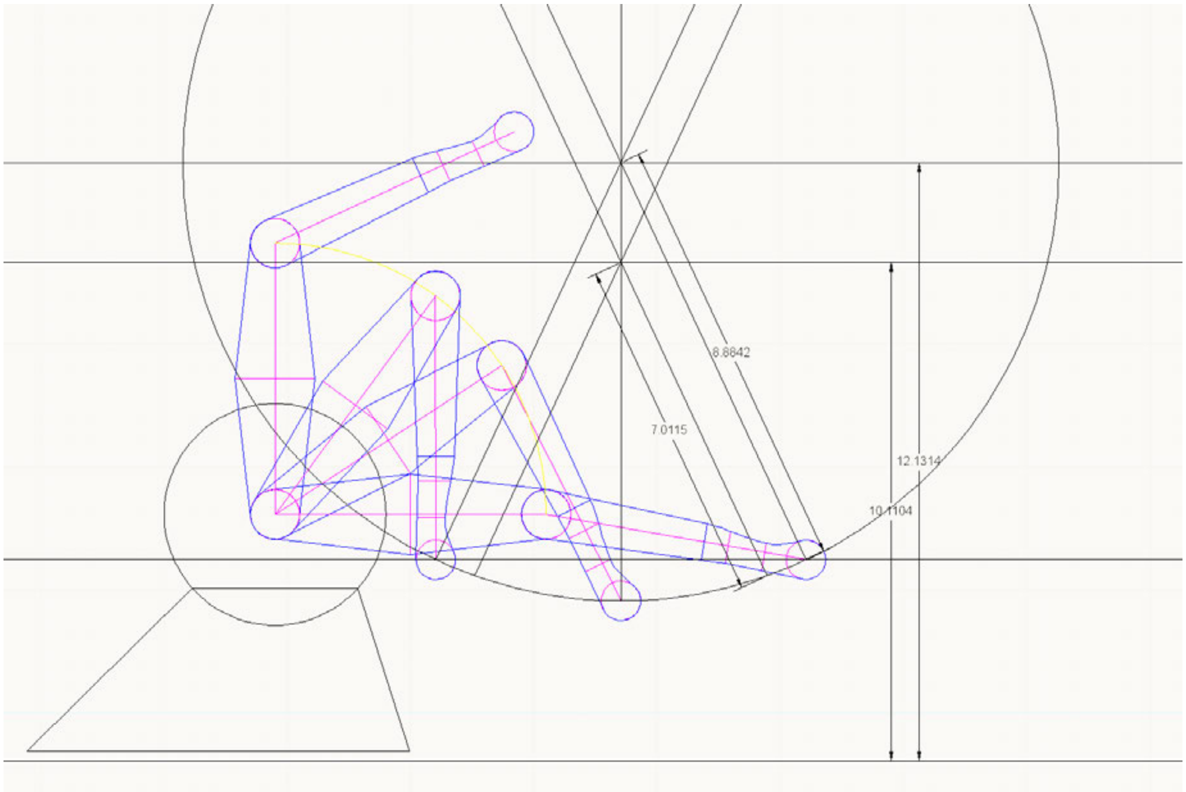


Figure 3.8: Original Haptic Movement Dimensions

Playstation, Nintendo), and incorporating this style joystick into a stationary fixture seemed unnatural. Thus, our final design was based on the CH Products M11L061P finger joystick (Figure 3.9) [CH Products, 2011; Digikey, 2011a, b]. This joystick is commonly found on closed-circuit television (CCTV) products, but has also been used in many other applications including robot control, such as the Inuktun Variable Geometry Tracked Vehicle (VGTV) operator control unit. We refer to this product as our model joystick. It should be noted that, unlike our prototype, our final joystick makes no use of any parts from this actual product; rather, we have created a haptically enabled imitation of it (Figure 3.9).

[8] [Digikey, 2011b]
[9] [Digikey, 2011a]

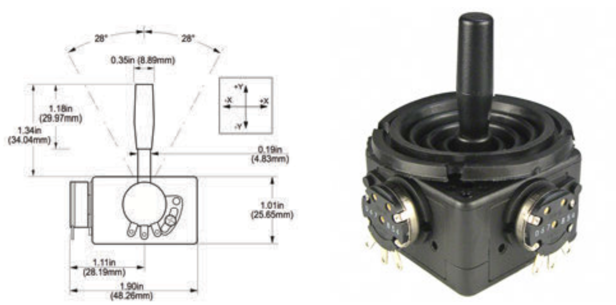


Figure 3.9: CH Products M11L061P[89]

3.5.2 Joystick Arm

An important aspect of a joystick is its physical properties, which in our case are governed by the design of the joystick arm. As mentioned in subsection 3.2.1, two specific characteristics of particular (but not obvious) importance are 1) the physical boundaries of the device which restrict movement beyond certain points, and 2) the spring mechanism which serves as a centering force. In prior experiments, we have observed that users tend to use the physical boundaries of the joystick while driving, which seems to be easier then holding the joystick at an intermediate position. Just below the surface of the user interface, located midway along the arm, is a pivot joint. The pivot joint is located in the same position along the joystick that potentiometers are mounted on our model. Our pivot joint has a 65° circular range of motion which closely approximates the model joystick's 55° square range of motion, which is damped by a circular rubber boot.

The topmost section of the joystick arm is the part the user will regard as "the grip". Extending up through the surface of the user interface, it is capped with an ABS plastic handle and is designed to be the same dimensions as our model joystick. Users tend to push the joystick from one side, relying on the centering force to hold the stick in position. We have reproduced this effect using haptics, as discussed in section 4.1.

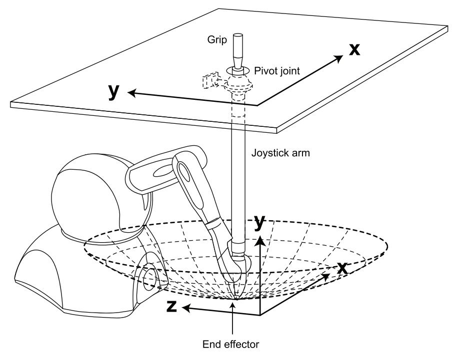


Figure 3.10: Final Joystick Design

The bottom of the arm is an adapter connecting to the end effector of the Phantom Omni. This design allows us to simultaneously measure the position of the joystick and generate haptic forces. The end effector's movement is restricted to rotating spherically below the pivot joint (see Figure 3.10), constraining the Phantom's 6 DOF. We have calculated the length of the joystick arm to maximize the end effector's range of motion, thus also maximizing mechanical advantage and providing the highest resolution for haptic effects.

Attributes	**57°**	**60°**	**63°**	**65°**
Max length from pivot to bottom of the gimble	152mm	166mm	182mm	196mm
Height of pivot above center	127mm	143mm	162mm	177mm
Approximate ratio of forces (Haptic:Joystick)	127:125 (1:1)	141:125 (9:8)	157:125 (5:4)	171:125 (4:3)
Max length of shaft connected to gimble to the bottom of the metal bearing	112mm	126mm	142mm	156mm

Table 3.1: Joystick arm calculations for different ranges of motion

One of the lessons learned during the process of building the prototype joystick was that the joystick arm needed to become more rigid. When pressure from haptic forces was applied, the ABS shaft and steel core design would flex an undesirable amount before the user would become aware of the pressure being applied. Another important

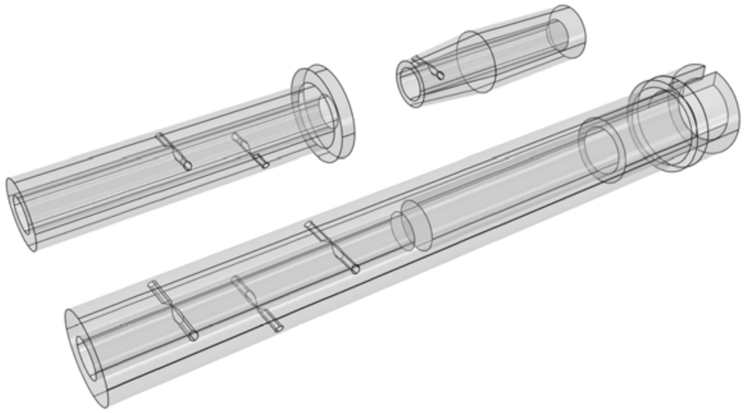


Figure 3.11: ABS Plastic Parts. Counter-clockwise from the top right: Joystick Grip, Pivot Adapter, Omni End Effector Adapter

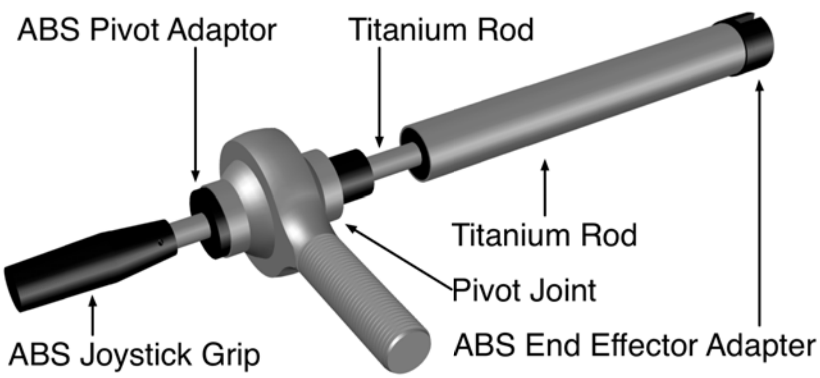


Figure 3.12: Arm Assembly. Plastic parts from Figure 3.11 are shown in black, metal parts are shown in grey.

property of the arm was weight, which needed to be kept to a minimum to maximize the efficiency of system. The larger the mass of the arm, the more its momentum affects haptic behavior. A new arm was designed using stock grade 5 titanium components to both increase rigidity and minimize the weight. These parts were held together with custom fittings made from ABS plastic, encased inside an exterior shell created from titanium tubing.

The titanium parts consist of one 4.75 inch long rod 3/16" in diameter for the shaft, and one 3.25" long tube, 1/2" in diameter, used to reinforce the ABS end effector adapter as an exterior shell. The pivot joint remained a 1/2" PTFE lined ball joint rod end. The titanium and ball joint can be purchased from McMaster-Carr (part numbers #89055K321, #6960T61, and #6960T11), and the titanium can be cut to length. The plastic fittings connecting the pieces of the arm (see Figure 3.12) were printed using a 3D rapid prototyping machine.

3.6 Suspension Mount

The Omni has a limited range of motion (see Figure 3.8), the use of which was optimized in the design of the joystick arm. However, the joystick arm design works under the assumption that the haptic device itself will be located at a specific position relative to the pivot point. While a small amount (e.g. a few centimeters) of device placement error can be accommodated by calibration routines, we needed to implement a system to get the device into the correct general location. We decided to create a suspension mount system that could be used to obtain correct device placement.

Figure 3.13: Final Suspension Mount Design. *Left:* Side View, *Right:* 3D View

The range of motion of the pivot point exactly defines the limits of the joysticks arm and end effector movements; however, incorrect placement of the haptic device could lead to the end effector's movement being prematurely limited by the physical constraints of the haptic device. We empirically determined that the haptic device has the smoothest range of motion when mounted at a slight incline, approximately 13 degrees.

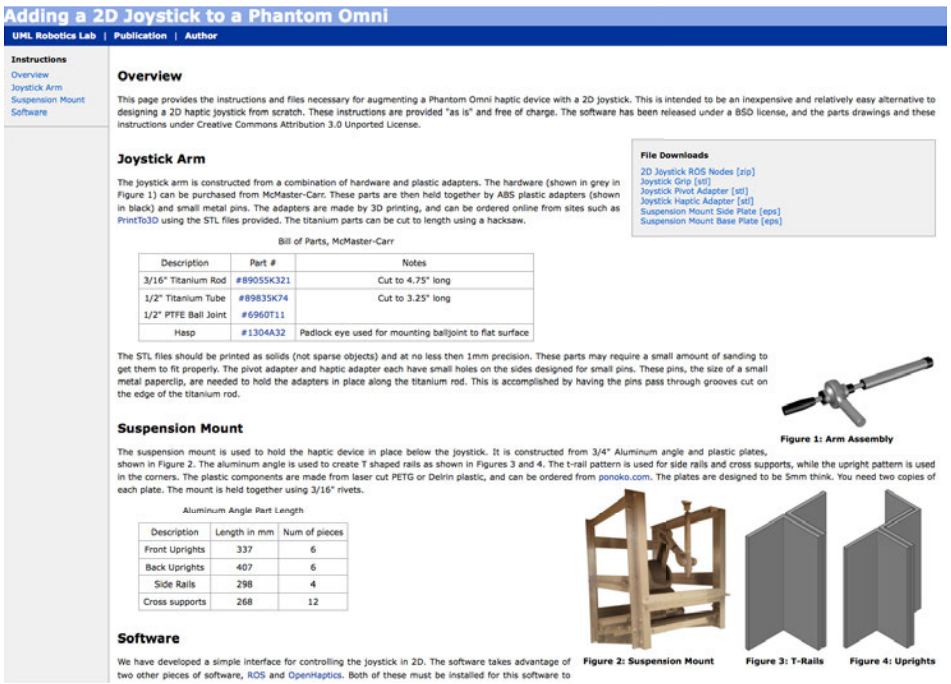
Our early prototypes used for determining the correct position and angle of the haptic device were built with wood and used threaded rod to allow us to easily reconfigure the setup. While the wooden prototype was useful in early stages of development, it quickly became apparent that a more permanent and compact structure was required.

We designed our second prototype using 1/4" PETG plastic and blind rivets. PETG was chosen as it could be precision cut using a laser cutter and was robust against cracking. The mount was designed to be sufficiently wide enough that vertical uprights would not impede the movement of the end effector. Unfortunately, this design lacked sufficient rigidity, causing energy from the haptic device to be absorbed by movement of the mount.

In our final suspension mount design, the PETG frame was replaced with aluminum members to increase rigidity. The final design consisted of an inclined plastic plane, suspended by four aluminum "T" shaped columns spaced wide enough apart to not impede the movement of the end effector. The structure was stiffened with plastic and aluminum bracing on all four sides to reduce movement. The entire mount was constructed from 1/4" laser cut plastic, 3/4" aluminum angle, and 3/16" diameter blind rivets.

3.7 Publishing the design

One of our goals was to create a haptic joystick that was easy to build and duplicate so that other researchers could take advantage of it. We have put together a set of do it yourself (DIY) instructions that we posted on our website located at `www.cs.uml.edu/~dbrooks/haptics`, which includes a bill of parts, the files needed for manufacturing custom components, and links to recommended internet-based services. The haptic joystick adapter design, suspension mount design, instructions, and the associated template files available for download have all been licensed under the Creative Commons Attribution 3.0 Unported License. Parts lists can be found in Appendix A and on our website.



Adding a 2D Joystick to a Phantom Omni

UML Robotics Lab | Publication | Author

Instructions
Overview
Joystick Arm
Suspension Mount
Software

Overview

This page provides the instructions and files necessary for augmenting a Phantom Omni haptic device with a 2D joystick. This is intended to be an inexpensive and relatively easy alternative to designing a 2D haptic joystick from scratch. These instructions are provided "as is" and free of charge. The software has been released under a BSD license, and the parts drawings and these instructions under Creative Commons Attribution 3.0 Unported License.

Joystick Arm

The joystick arm is constructed from a combination of hardware and plastic adapters. The hardware (shown in grey in Figure 1) can be purchased from McMaster-Carr. These parts are then held together by ABS plastic adapters (shown in black) and small metal pins. The adapters are made by 3D printing, and can be ordered online from sites such as PrintTo3D using the STL files provided. The titanium parts can be cut to length using a hacksaw.

File Downloads
2D Joystick ROS Nodes [zip]
Joystick Grip [stl]
Joystick Pivot Adapter [stl]
Joystick Haptic Adapter [stl]
Suspension Mount Side Plate [eps]
Suspension Mount Base Plate [eps]

Bill of Parts, McMaster-Carr

Description	Part #	Notes
3/16" Titanium Rod	#89055K321	Cut to 4.75" long
1/2" Titanium Tube	#89835K74	Cut to 3.25" long
1/2" PTFE Ball Joint	#6960T11	
Hasp	#1304A32	Padlock eye used for mounting balljoint to flat surface

The STL files should be printed as solids (not sparse objects) and at no less then 1mm precision. These parts may require a small amount of sanding to get them to fit properly. The pivot adapter and haptic adapter each have small holes on the sides designed for small pins. These pins, the size of a small metal paperclip, are needed to hold the adapters in place along the titanium rod. This is accomplished by having the pins pass through grooves cut on the edge of the titanium rod.

Figure 1: Arm Assembly

Suspension Mount

The suspension mount is used to hold the haptic device in place below the joystick. It is constructed from 3/4" Aluminum angle and plastic plates, shown in Figure 2. The aluminum angle is used to create T shaped rails as shown in Figures 3 and 4. The t-rail pattern is used for side rails and cross supports, while the upright pattern is used in the corners. The plastic components are made from laser cut PETG or Delrin plastic, and can be ordered from ponoko.com. The plates are designed to be 5mm think. You need two copies of each plate. The mount is held together using 3/16" rivets.

Aluminum Angle Part Length

Description	Length in mm	Num of pieces
Front Uprights	337	6
Back Uprights	407	6
Side Rails	298	4
Cross supports	268	12

Figure 2: Suspension Mount
Figure 3: T-Rails
Figure 4: Uprights

Software

We have developed a simple interface for controlling the joystick in 2D. The software takes advantage of two other pieces of software, ROS and OpenHaptics. Both of these must be installed for this software to

Figure 3.14: Haptic Joystick Website at www.cs.uml.edu/~dbrooks/haptics

Chapter 4

Software Design

The core haptic joystick software consists of device-specific functionality that provides a generic interface for controlling the 2D joystick (see Figure 4.1). This software has been published as Open Source software under the New BSD license [1], and allows for programming arbitrary haptic effects. The robot navigation system written for our case study in chapter 6 makes use of the functionality provided by the software in this chapter.

4.1 SensAble OpenHaptics

The Phantom Omni comes with the powerful OpenHaptics Toolkit [SensAble Technologies Inc, 2011], consisting of Phantom Device Drivers, a Haptic Device API, and Haptic Library API. The Haptic Library API (HLAPI) was designed to compliment OpenGL, allowing scenes rendered in OpenGL to be given physical properties which can be explored though the haptic device. This API is not well adapted for writing software to control a joystick as this was not its intended use case; attempting such a task would

[1] Available at `www.cs.uml.edu/~dbrooks/haptics`

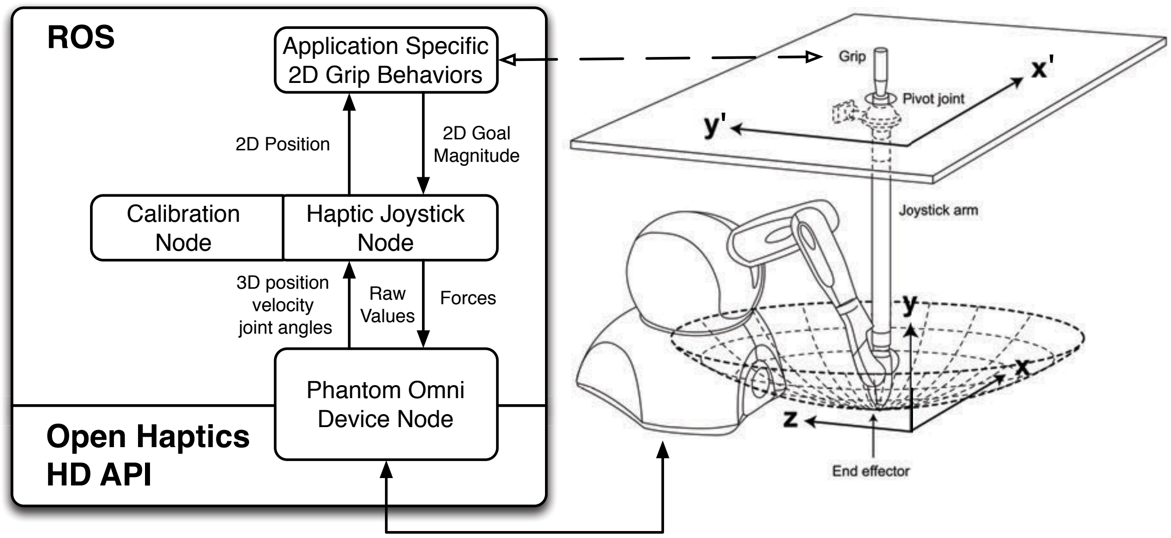


Figure 4.1: Architecture Block Diagram of the Haptic Joystick Software

require an unnecessarily roundabout and cumbersome process. The Haptic Device API (HDAPI) is a low-level API which can be used to read raw information from the device and control motors. This API was written as a support library for the HLAPI and designed to give developers lower level access to the device. Although not immediately obvious, the HDAPI by itself is poorly suited for attempting precision control of the device relative to the physical world, particularly with respect to the reported coordinates of the end effector. This is not actually a shortcoming of the HDAPI or even considered to be an undesirable behavior, since the library was designed for performing calculations relative to a virtual environment rather then the physical world, and the reported values help achieve that objective.

Traditionally, the mobile ground robot driving task has been performed using 2 DOF joysticks, controlling the linear (forward) and angular (turning) velocities along two axes lying on a plane. Using SensAble's HDAPI, we have created a new API which not only exposes HDAPI's low-level control as a ROS node (`haptic_driver`), but allows us to read and control the joystick grip position in 2D (`haptic_joystick`).

4.2 ROS Haptic Driver Node

We wrote an interface that exposes the device-specific functionality of the HDAPI as ROS topics, called the `haptic_driver` node. At the time, there was another ROS node that provided similar functionality called the `phantom_omni` node written by Georgia Tech's Healthcare Robotics Lab [Georiga Tech, 2013]. What started as a handful of edits and fixes to their code quickly turned into a full reimplementation to better suit our application.

One of the most significant problems we encountered was the way the end effector's 3D position was reported. After extensive testing, we found that the HDAPI uses curved axes to report the end effector's position. To get the end effector's position in a Cartesian coordinate system suitable for measuring position in the real world, we used the angles of each joint to calculate our own 3D coordinates. We were able to accomplish this using trigonometry and knowing the lengths of each part of the arm. The angle measurements output by the haptic device – turret, thigh, and shin (see Figure 4.2) – are based on their starting position. When the device is first plugged in, the end effector must be in its dock. We shift these measurements so that all angles

Figure 4.2: Phantom Omni Linkages

are based on a horizontal plane. By measuring the lengths of the shin, L_s and thigh, L_t (where $L_t = L_s = 133.35$ mm), the values of x, y, and z coordinates are computed using Equation 4.2.1.

$$
\begin{aligned}
x &= L_t \cdot \cos(thigh\_ang) \cdot \sin(turret\_ang) \\
&\quad + L_s \cdot \cos(shin\_ang) \cdot \sin(turret\_ang) \\
y &= L_t \cdot \sin(thigh\_ang) + L_s \cdot \sin(shin\_ang) \\
z &= L_t \cdot \cos(thigh\_ang) \cdot \cos(turret\_ang) \\
&\quad + L_s \cdot \cos(shin\_ang) \cdot \cos(turret\_ang)
\end{aligned} \tag{4.2.1}
$$

The `haptic_driver` ROS node publishes information about the device's status over the `/haptic/status` topic in an `OmniStatus` message. The `OmniStatus` message includes the device end effector's 3D `pose` in Cartesian coordinates, the current `velocity` of the end effector, and information provided by the HDAPI such as the `turet_angle`, `thigh_angle`, `shin_angle`, etc. Forces can be sent to the device on the topic `/haptic/feedback` as type `geometry_msgs::Vector3`, specifing the `x`, `y`, and `z` force components. Finally, the `haptic_driver` has been designed to robustly idle in the absence of a ROS master, allowing it to be started independently and remain running while the rest of a ROS system is brought up and down.

4.3 ROS Haptic Joystick Node

Controlling the movement of the joystick grip rotating about the pivot point through the use of 3D direct force vectors proved to be a very complicated and unsustainable method of programming the joystick. For this reason, we developed the `haptic_joystick` node, which provides a much simpler 2D interface for programming arbitrary haptic effects. Our 2D haptic joystick API allows for more abstract control of the joystick without loss of functionality, and will also allow other researchers who choose to use our toolkit to to focus on the nuances of creating haptic joystick behaviors, rather than the complexity of programming the 3D forces.

4.3.1 Calibration

Calibration is a major component to the `haptic_joystick` node, as it provides the 3D to 2D coordinate transformation. The end effector's movement is restricted to the surface of an imaginary sphere centered about the pivot joint (See Figure 4.1). Because the haptic device's location with respect to the pivot point is not known exactly, calibration is required to establish joystick positioning and orientation in our coordinate system. The calibration routine is very simple and consists of moving the joystick grip to the centered, forward, backward, left, and right positions, then moving the joystick grip about randomly.

This movement allows us to define two line segments between opposing maximum points and collect a set of points lying along the spherical range of motion of the end effector. The points collected are passed through a voxel filter, reducing the amount of information to processes and yielding a uniform distribution. To estimate the center and radius of the sphere, which corresponds to the location of the pivot joint and length of our joystick arm respectively, we minimize the difference between the radius and each point's distance from the sphere's center using least squares. Letting $R = x_c^2+y_c^2+z_c^2-r^2$ where x_c, y_c, and z_c are the coordinates of the sphere's center and R is its radius, the equation of the sphere becomes

$$-2x \cdot x_c - 2y \cdot y_c - 2z \cdot z_c + R = -x^2 - y^2 - z^2 \tag{4.3.1}$$

Solving Equation 4.3.1 for n points gives the matrix equation

$$\begin{pmatrix} -2x_1 & -2y_1 & -2z_1 & 1 \\ -2x_2 & -2y_2 & -2z_2 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ -2x_n & -2y_n & -2z_n & 1 \end{pmatrix} \begin{pmatrix} x_c \\ y_c \\ z_c \\ R \end{pmatrix} = \begin{pmatrix} -x_1^2 - y_1^2 - z_1^2 \\ -x_2^2 - y_2^2 - z_2^2 \\ \vdots \\ -x_n^2 - y_n^2 - z_n^2 \end{pmatrix} \tag{4.3.2}$$

A linear algebra approach to least squares gives us estimated values of

$$\begin{pmatrix} x_c \\ y_c \\ z_c \\ R \end{pmatrix} = \left[\begin{pmatrix} -2x_1 & -2y_1 & -2z_1 & 1 \\ -2x_2 & -2y_2 & -2z_2 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ -2x_n & -2y_n & -2z_n & 1 \end{pmatrix}^T \begin{pmatrix} -2x_1 & -2y_1 & -2z_1 & 1 \\ -2x_2 & -2y_2 & -2z_2 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ -2x_n & -2y_n & -2z_n & 1 \end{pmatrix} \right]^{-1} \begin{pmatrix} -x_1^2 - y_1^2 - z_1^2 \\ -x_2^2 - y_2^2 - z_2^2 \\ \vdots \\ -x_n^2 - y_n^2 - z_n^2 \end{pmatrix} \tag{4.3.3}$$

Note that Equation 4.3.3 is always solvable unless all n points are coplanar.

The orientation is calculated by transforming the forward-backward and left-right line segments into the base of an upside-down regular square pyramid inscribed in the calculated sphere. We do this by shifting the maximum points defining these line segments so the segments intersect with each other and the radius touching the sphere's bottom (defined by the point collected when the joystick is centered). The segments are shifted again so that each maximum point is equidistant from the intersection point. This is followed by another shift forcing the maximum points to create a plane that is normal to the radius and touching the sphere's bottom. Thus, the segments are at a right angle to each other. Finally, the points are projected onto the sphere's surface using a ray through each point from the sphere's center.

4.3.2 Coordinate Transforms

With the joystick calibrated, we can then calculate the 2D joystick grip position corresponding to any 3D end effector position and vice versa. Converting from 3D to 2D, the point is first projected onto the plane containing the sphere's center and the forward-backward axis. The planar angle is then calculated as the angle between the ray made by the sphere's center and bottom, and the ray made by the sphere's center and the projected point. The forward-backward output coordinate is measured as the percentage of this angle to the maximum angle possible, giving a range of -1 to 1. A similar calculation is done to find the left-right output coordinate. An algorithm performing the inverse of this operation is used to convert in the opposite direction, from 2D to 3D.

4.3.3 2D Force Control

The joystick grip can be controlled haptically by specifying the (X,Y) target coordinate to which the joystick grip should travel, and the amount of force to use in getting there. We accomplish this by using a "gravity well" to pull the joystick toward the target point. This effect is based on Hooke's Law and made by finding the vector from the current actual position to the target position. A force is applied in that direction with a magnitude proportional to the length of that vector and scaled by the input magnitude of that target. The actual force vector calculation is

$$\mathbf{F} = \frac{\min(|\mathbf{v}|, d_{thresh})}{d_{thresh}} \cdot ((F_{max} - F_{spring}) \cdot m + F_{spring}) \cdot \frac{\mathbf{v}}{|\mathbf{v}|} \tag{4.3.4}$$

where $\mathbf{v}$ is the vector from the current position to the current target, F_{max} is the maximum force allowed, F_{spring} is the minimum force allowed that creates a spring like effect, m is the input magnitude, and d_{thresh} is the distance after which $|\mathbf{v}|$ is no longer used to scale the force. m ranges from 0 to 1 where 0 yields just a spring force, and 1 yields the strongest force allowed. By setting a weak amount of force centered at the origin, we can create the spring centering effect, previously discussed in subsection 3.2.1.

4.3.4 2D Joystick ROS Interface

The `haptic_joystick` node provides the 2D force control interface used to program haptic behaviors for the joystick. The 2D joystick grip position can be read by subscribing to the topic `/joy_pos` which is encoded using the standard ROS convention of `Twist` messages. The forward-backward values are stored in `linear.x`, and the left-right value in `angular.z`; values range from -1.0 (back or left, respectively) to 1.0 (forward or right).

Commanding the 2D joystick grip can be done by publishing a message to the topic `/joyfeedback`. The `JoystickInput` message type specifies the joystick grip's target `linear` and `angular` positions (-1.0 to 1.0), and the amount of force to use as a `magnitude` (0 to 1). `take_control` (boolean) is used to toggle the joystick between haptic and non-haptic modes.

4.4 Publishing the Design

The software described in this chapter is publicly available as open source software on our website at `www.cs.uml.edu/~dbrooks/haptics`, released under a BSD license. The first two toolkit components (Hardware and Software) were presented at the IEEE Conference on Technologies for Practical Robot Applications (TePRA) [Brooks, Lunderville, and Yanco, 2013]. The website also contains instructions for acquiring the necessary pre-installation requirements, and building/installing the software.

Chapter 5

Experiment Template

As we have already asserted, it is very difficult to compare the effectiveness of various haptic behaviors from the literature. One of the major contributing factors to this difficulty is the lack of a consistent method for testing.

We describe an experiment template suitable for conducting a wide range of tests for haptic teleoperation and navigation feedback behaviors. The basic experimental design consists of a within-subjects study in which an operator drives or navigates a semi-autonomous remote mobile robot through a slalom course using two different control methods. This generic task has been used successfully in previous experiments to investigate various factors relating to remote teleoperation at two universities [1] [Desai, 2012; Desai, Kaniarasu, Medvedev, Steinfeld, and Yanco, 2013; Desai, Medvedev, Vázquez, McSheehy, Gadea-Omelchenko, Bruggeman, Steinfeld, and Yanco, 2012].

The experiment has been carefully designed to mitigate the effects of learning during each session, as well as to balance out any biasing effects that might result from running both experimental conditions with each participant. Each participant performs two

[1] Carnegie Mellon University and the University of Massachusetts Lowell

trial runs, and then six runs during which data can be recorded. Eighteen unique sets of participant conditions are used to eliminate any potential biasing by balancing the order of the maps, event sequence variations, and haptic conditions (see Appendix section B.2). Each set of participant conditions consists of a unique ordering of the maps when paired with the map variations. Each map is used exactly once, and each event sequence variation is used exactly twice for each participant. The same joystick device and graphical user interface are used in both control modes, thus by design making the only difference the haptic effects.

5.1 Data Collection

There are a number of important components that should be measured at a minimum when comparing control methods.

5.1.1 Spatial Reasoning

A person's ability to control a remotely located robot is linked with their spacial reasoning (SR) capabilities [Lathan and Tracey, 2002]. An operator with well developed spatial reasoning abilities will find the task of operating a remote robot easier than someone with low spatial reasoning.

The ETS Cube Comparison test [ETS, 1976] is used for measuring participants' SR abilities. This test only needs to be performed once for each participant, and can be administered before or after the experiment.

5.1.2 Workload

The motivations for introducing haptic feedback into the task of teleoperation or navigation of a remote robot may vary, but they share the goal of reducing the operator's workload. The NASA Task Load Index (TLX) [Hart and Staveland, 1988] is a subjective measurement of a participant's perceived workload. The TLX questionnaire asks the participant to rate (on a scale of 0 to 20, with 11=neutral) their Mental Demand, Physical Demand, Temporal Demand, Performance, Effort, and Frustration for an activity. The TLX survey should be administered following each use of the robot. A free online TLX survey is available at `www.playgraph.com/nasatlx` [Sharek, 2011, 2013] and methods for analyzing the survey can be found in Hart [2006].

5.1.3 Situation Awareness

Situation Awareness (SA) is defined as

> "The perception of the elements in the environment within a volume of time and space, the comprehension of their meaning, and the projection of their status in the near future." [Endsley, 1988]

There are three levels of SA - (1) Perception, (2) Comprehension, and (3) Projection. SA can be measured using the Situation Awareness Global Assessment Technique (SAGAT), a technique originally developed for measuring SA in aircraft pilots [Endsley, 1988]. The technique consists of pausing an experiment or simulation, blacking out the interface, and asking the operator a series of questions related to the current situation.

Questions should span all these levels of SA, and there should be enough questions such that a subset can be selected at random from a pool for each SAGAT screen.

SAGAT questions are asked four times during each run, an example of which can be found in subsection 6.8.1.

5.1.4 Task Performance

A number of metrics directly related to the task should be recorded for each run. These metrics are useful as indirect measures of SA and the usability of an interface. Task performance metrics include:

Wrong turns: Number of occurrences in which the robot passed an obstacle on the wrong side

Time to Completion: Time elapsed if the task was successfully completed

Collisions: Number of occurrences in which the robot bumped, scraped, or hit any part of the environment

5.1.5 Subjective Measures

Finally, a number of subjective measures can be recorded. These should be asked after each run, and can be administered along side the TLX questionnaire. Other experiment specific questions can also be added.

- Which autonomous mode did you prefer? (Only if multiple modes are available in each run)
- Please rate how predictable the system was during the last run (1, Unpredictable to 10, Very Predictable)
- Please rate the robot's performance during the last run (1, Very Poorly to 10, Very Well)

- Please rate how well you feel you performed during the last run (1, Very Poorly to 10, Very Well)

5.2 Course Layout

The experiment was designed to be run in a straight, wide, hallway area. The course consists of a series of 8 obstacles placed at regular intervals lengthwise down the center of the hall. The driving task consists of guiding the robot down the hallway, making a u-turn, and returning to the starting point, weaving between the obstacles as in a slalom course.

The course can be configured to match the size of the robot being operated (see Equation 5.2.1). The spacing between obstacles should be approximately 2.5 times the length of the robot L_r, and each obstacle's length L_o should be approximately half the total length of the robot. Obstacles' widths W_o are determined by the width of the hallway and the width of the robot W_r, with the summed amount of space on either side of the robot as it passes by an obstacle approximately equal to 75% of the robot's width.

$$
\begin{aligned}
obs\_spacing &= 2.5 \cdot L_r \\
L_o &= .5 \cdot L_r \\
obstacle\_clearance &= .75 \cdot W_r \\
W_o &= hall\_width - 3.5 * W_r
\end{aligned}
\tag{5.2.1}
$$

While this is spacing allows for plenty of space to maneuver the robot, prior experimentation has indicated that manually navigating the obstacles is a non-trivial task [Desai, 2012].

5.3 Maps and Event Sequence Variations

A total of seven maps (six plus the Trial Map) and four event sequence variations (three plus Trial Variation) were designed to have the same level of task difficulty for each run. For each map, the robot needed to pass a total of 15 obstacles (7 boxes out, u-turn, and 7 boxes back), such as shown in Figure 5.1. Each map needed to have the robot "crossing" the line of boxes a total of 6 times, not including moving to or from the starting point to the first box or going around the u-turn. The robot needed to cross the line of boxes no less then twice and no more then 4 times in any given direction (out or back). In addition, map generation needed to take into account the locations of planned mistakes in each event sequence variation.

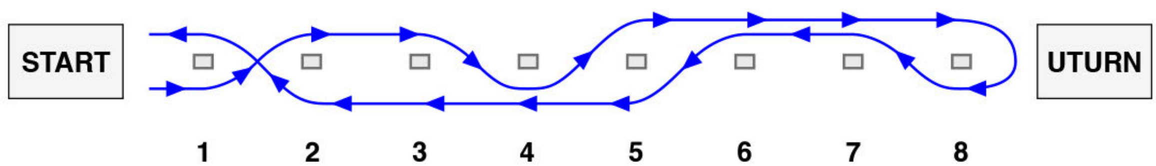


Figure 5.1: Example Course Map. Full set of maps can be found in section B.1

5.3.1 Planned Mistakes

One barrier to the wider adoption of autonomous mobile robot systems is the poor ability of operators to recover control or correct behaviors that may occasionally be necessary when errors occur. This issue is known as the "out-of-the-loop" problem and results from low situation awareness [Kaber and Endsley, 1997]. Based on this, we have designed the experiment to incorporate two artificial autonomy failures in each run to allow us to investigate this type of scenario.

There are two types of mistakes the robot can make (see Figure 5.2). It can go straight when it should have turned (Type S), or it can turn when it should go straight

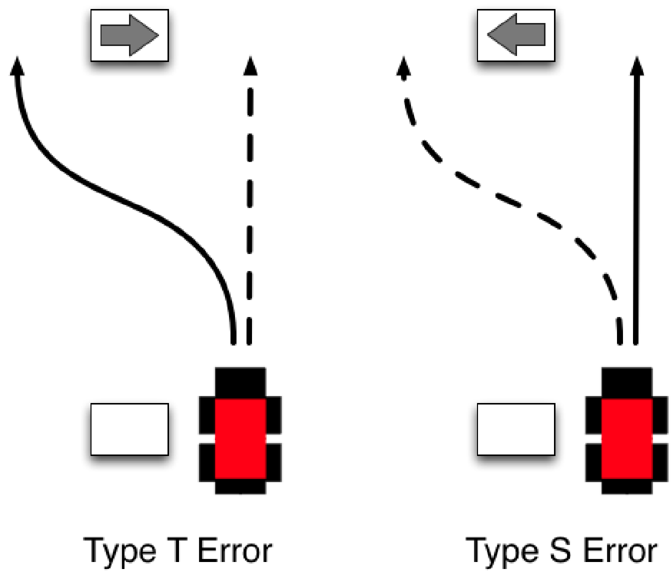


Figure 5.2: Type T and Type S Mistakes. The dashed line shows the correct route the robot *should* have taken based on the direction of the arrow. The solid line shows the route the robot would try to take by "mistake" due to an error in misinterpreting the direction it was supposed to go.

(Type T). Whether the robot makes a Type S or Type T mistake at a specified error point depends on which map is being used. These mistakes are balanced such that each participant experiences an equal number of each type of mistake for both control methods.

With the exception of the trial variation, each variation consists of two points along the course where the robot is programmed to make a mistake. The number of mistakes was selected to be infrequent enough to maintain the operator's trust in the robot's competence [Desai, 2012]. The first mistake happens while passing an obstacle on the way out to the u-turn, and the second on the way back to the start. Each mistake happens in a unique location – in other words, there are six unique places where the robot can make the intentional error.

Variation	**Screen1**	**Screen2**	**Screen3**	**Screen4**
A	YB	NB	NS	YS
B	NS	YS	YB	NB
C	NB	YS	NS	YB

Table 5.1: Event Sequence Variation Error and Placement Pairings. (Y) Error (N) No Error (B) Between Boxes (S) Beside a Box

5.3.2 SAGAT Screen Requirements

The location of SAGAT screen events takes into account each screen's proximity to mistake locations. There are an equal number of SAGAT screens in each direction, and, in each direction, there is one SAGAT screen for which the robot makes an error, and one for which it does not. Half of the SAGAT screens occur between boxes, and the other half occur when the robot is directly beside a box in the act of passing. For each box (except the first and the u-turn), there is exactly one instance where a SAGAT screen occurs beside it. When pairing the error(Y)/no-error(N) and between boxes(B)/beside a box(S) scenarios, there is one of each in each of the event sequence variation (see Table 5.1).

During the two trial runs, only one SAGAT screen should be displayed. This maximizes the amount of continuous driving time the participant receives to allow them to become familiar with the task, while also familiarizing them the SAGAT screens. Because there is only one SAGAT screen in each trial run, additional questions from the pool of possible questions may be asked to help expose the participant to the types of questions to which they will respond.

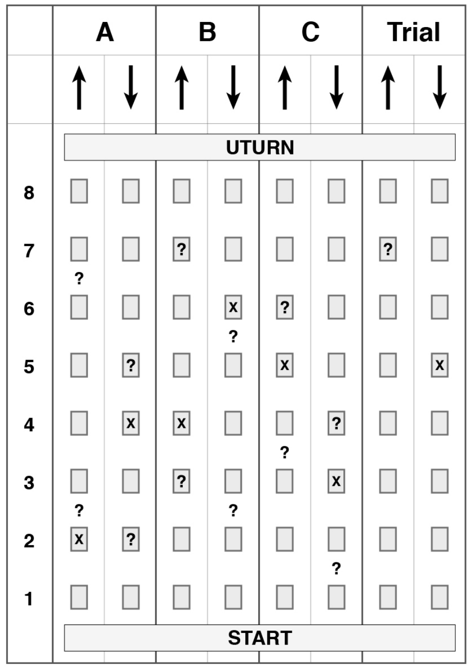


Figure 5.3: Event Sequence Variations A, B, C, and Trial. '?' denotes SAGAT screen locations, and '×' denotes planned mistakes. Variations were applied to maps.

5.3.3 Generating Maps and Variations

Event sequence variations A, B, and C consist of two planned mistakes and four SAGAT screen locations (see Figure 5.3). The trial variation has a single SAGAT screen location and only one planned mistake. A Python script was used to generate all possible map patterns and variation combinations that meet these requirements. One solution was selected, and the requirements were manually verified by hand before beginning this experiment.

5.4 Graphical User Interface

To facilitate comparing results between different experiments, the visual components of the interface should be kept basic and consistent. Based on prior work informing graphical user interface (GUI) design for teleoperation of remote robots [Keyes et al., 2010], we make the following recommendations.

The GUI should consist of a forward facing main video feed, a rear video feed, and distance display (see Figure 5.4). The main video feed should be featured as the primary feature in the center of the interface. An optional control may be added to control movement of the main video feed using a pan/tilt unit. The rear video feed should be in a smaller window placed outside the top right hand corner of the main video feed, analogous to the position of a rear view mirror in a vehicle. The rear video stream may also be mirrored to enhance this effect. Below the main video feed should be a distance display featuring a top down view of the robot (shown as an icon), and a line drawing representing information from the robot's distance sensors (e.g., laser range finders, sonar ring). The distance display should be drawn to scale, and as complete a 360 degree view as possible. Finally, the GUI should have a small runtime clock displaying elapsed time since the beginning of the run. Additional information may be added to the GUI relating specifically to the haptic effects or any secondary tasks added to the experiment, but should be kept to an absolute minimum. It should be noted that the GUI design is not the focus of the experiment, thus is intentionally minimalist.

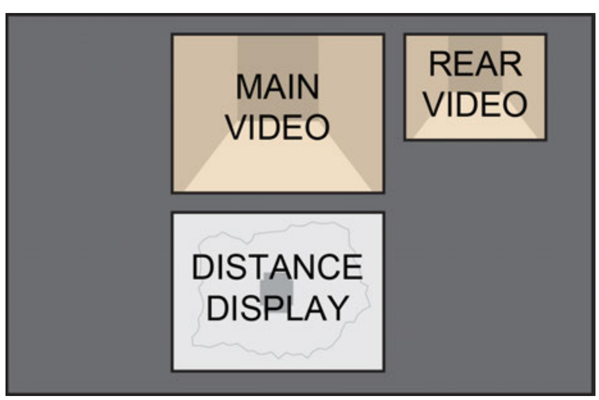


Figure 5.4: GUI Layout

Chapter 6

Toolkit Case Study

Influence of Haptic Shared Control Feedback Behavior on Operator's Situation Awareness

The creation of our toolkit was inspired by the need to facilitate better research practices within the community of researchers investigating haptic control of remote ground robots. We have used our toolkit to design an experiment, both for the sake of research and as a case study.

Our experiment investigates the effects of implementing shared control autonomy with haptic feedback on an operator's situation awareness. The haptic device used in the experiment made use of the hardware adaptation found in chapter 3, and haptic behavior was programmed using the software stack described in chapter 4. The course layout, graphical user interface, and experimental procedure (including collected information) conform to the experiment template in chapter 5.

6.1 Hypotheses

The haptic condition was the independent variable and consisted of alternating between haptic and no haptic feedback (called “Force Feedback” and “Non-Force Feedback” during the experiment), with the starting condition counterbalanced. For both haptic and non-haptic conditions, the total number of and type of mistakes made by the robot were balanced (see subsection 5.3.1) .

Our hypothesis was that haptic interfaces can be leveraged to help improve operator situation awareness. Specifically, we hypothesized that

- Users will have better task performance when operating the robot with the haptic joystick.
- Users will have better SA when operating the robot with the haptic joystick.
- Users will have lower cognitive workload when using the haptic joystick.

Situation Awareness was measured primarily using Endsley’s SAGAT technique [Endsley, 1988], which is one of the toolkit’s recommended data collection methods. Performance was considered a secondary indication of SA.

6.2 Robot

The robot used for this experiment was an iRobot ATRV Jr platform measuring 25 inches wide $\times$ 40 inches long. The ATRV Jr uses a differential (tank style) drivetrain controlled by an rFlex (stock) closed loop motor controller built into the chassis. The chassis was augmented with additional sensors, specifically two cameras, two laser range finders, and a pan tilt unit. A SICK LMS-200 laser range finder, capable of scanning

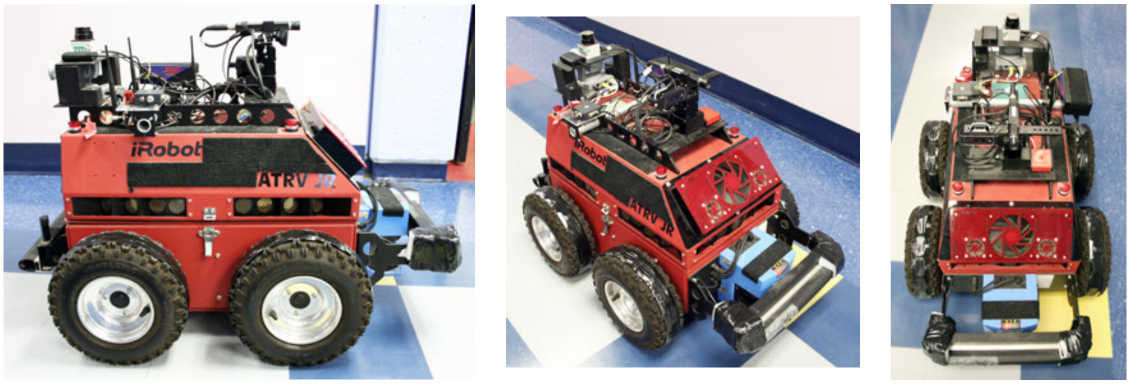
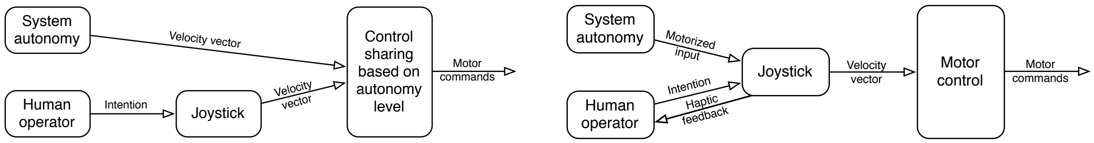


Figure 6.1: iRobot ATRV Jr

180 degrees, was mounted on the front of the robot about 6 inches above the ground. A Hokuyo URG-04LX laser range finder, with a scanning view of 240 degrees, was mounted on rear of the robot facing backwards at a height of 27 inches. The two lasers provided close to 360 degrees of coverage, but created a blind spot on either side of the chassis such that objects near the sides could not be seen; the blind spots did not limit the performance of autonomous navigation. A Directed Perception PTU-D46-17 pan-tilt unit was mounted directly above the front wheels' axle in the center of the robot. A Sony XC-999 camera was mounted on the center of the pan-tilt unit, facing the front of the robot. On the rear of the robot, a backward facing Canon VC-C4 camera was mounted directly below the rear laser.

The robot's onboard computer consisted of a 3.0 GHz Intel Core2Duo Processor with 4GB of memory running an Ubuntu 12.04 Server Installation, with ROS Fuerte. Remote communication with the robot was provided by an 802.11n wireless link between an Apple Airport Extreme 5GHz access point and the robot.



(a) Non-Haptic Shared Control (b) Haptic Shared Control

Figure 6.2: Shared Control Modes

6.3 Shared Control Driving Modes

Two shared control driving modes were tested: a haptic mode and a non-haptic mode. In both cases, the robot was able to navigate itself through the course autonomously, but could be overridden by user input. The same joystick devices, graphical user interface, navigation software, and obstacle avoidance software were used to control the robot in both modes. The joystick emulated the spring centering force of mechanical joysticks, and was capable of applying additional forces in the haptic mode.

The goal was to make the two driving conditions as similar as possible, with the only major distinguishing factor being the presence or absence, respectively, of haptic forces. In the non-haptic mode, the motors were directly controlled by the autonomous software, with speed controlled by pushing the joystick directly forward along the forward-backward axis. The autonomy could be overridden by moving the joystick off-center along the left-right axis or backward along the forward-backward axis (see Figure 6.2a). In haptic mode, the motors were directly controlled by the position of the joystick (see Figure 6.2b); the autonomous software drove by invoking joystick movement (the haptic effect) and could be overridden by pressing forcefully on the joystick. This haptic behavior can be classified as behavioral (rather than environmental) since the effect was generated from the robot's autonomous navigation and obstacle avoidance behaviors rather then reacting directly to the environment.

6.3.1 Autonomous Navigation System

To facilitate our goal of making the two driving conditions as similar as possible, we created an autonomous navigation behavior which could be implemented both with and without haptic effects. The navigation system was comprised of a waypoint style global planner and a local planning system, which created smooth trajectories between goal points and provided obstacle avoidance behaviors.

The courses consisted of a set of obstacles centered down the middle of a straight hallway. Since the robot must steer around each obstacle on its way down the hallway, we defined the robot's path as a list of *wayposes* located on the sides of each obstacle. A waypose consists of a waypoint and an orientation. A waypoint consists of an (x, y) position in world coordinates and a threshold radius which constituted being "close enough" to that point. When the robot arrived at a position satisfying the waypoint condition, it must have also simultaneously met the orientation requirement in order to satisfy the waypose.

Two sets of four wayposes were defined for each obstacle, at each of its four corners. Both waypose sets were oriented to point straight down (or parallel to) the hallway, with one set pointed away from the start and the other towards it. The path along which the robot should travel was selected as the left or right pairs of wayposes for all obstacles from the start to the u-turn and back.

6.3.2 Waypose Path Planning

Because our path is composed of wayposes rather than waypoints, conventional path planning algorithms are not sufficient because they do not consider the robot's goal orientation during the planning stage. It was possible for the robot to rotate in place

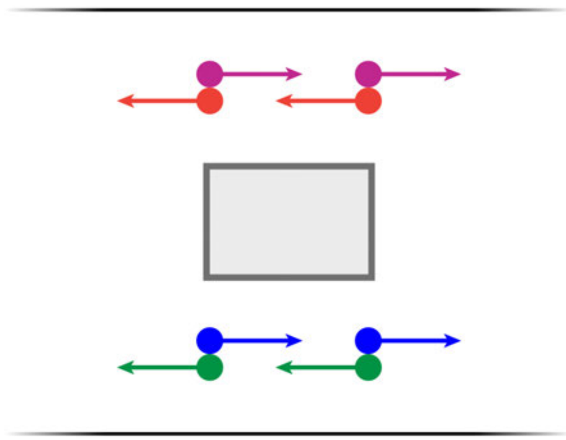


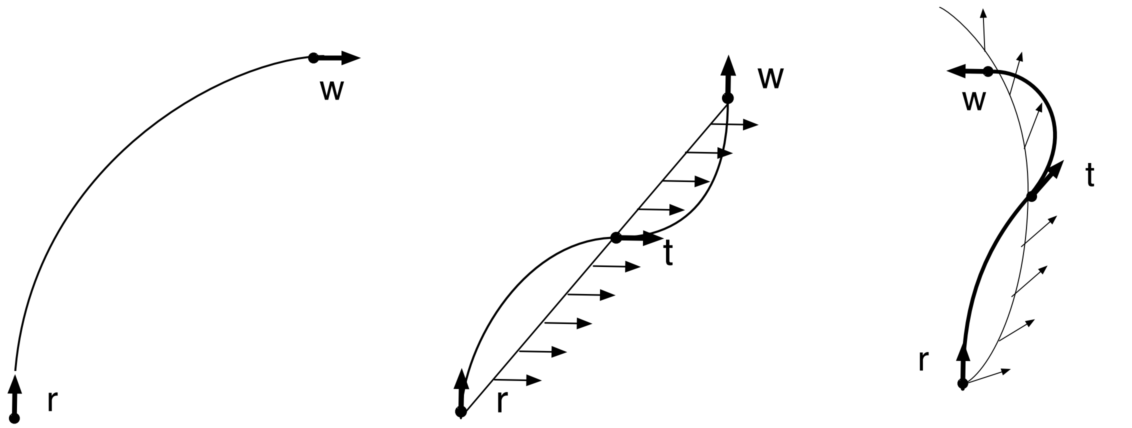
Figure 6.3: Waypose locations around each obstacle. There are four wayposes oriented in each of the two primary directions of travel, set in pairs at the four corners of the obstacle.

upon arrival at the waypoint to match the goal orientation. However, there are two fundamental issues with this approach. First, given the size of the robot, obstacles, and course, the robot would knock over the obstacles. Second, the operator and the autonomy share control of the vehicle, and this behavior is unnatural to a human. To achieve smooth driving, we implemented a "two step" arc planning algorithm based on Kanayama and Hartman [1989].

Symmetric Poses

The algorithm attempts to find a path defined by a circular arc that is tangent to both of the robot's current pose and the target (or next) waypose within some threshold. This is true when

$$sym\_thresh > \left| \left| \frac{r_\theta + w_\theta}{2} \right| - \arctan\left(\frac{w_y - r_y}{w_x - r_x} \right) \right| \tag{6.3.1}$$



(a) Symmetric (b) Non-Symmetric, Parallel (c) Non-Symmetric, Non-Parallel

Figure 6.4: Waypose Possibilities

These two poses are considered to be *symmetric* if such a path, first, can be found, and, second, does not contain obstacles currently within view of the robot's sensors. Appropriate linear and angular velocities are calculated for moving the robot along this arc towards the target waypose (see Figure 6.4a). Otherwise, the two poses are considered *non-symmetric*.

Non-Symmetric Poses and Temporary Wayposes

If the current robot pose and the target waypose are *non-symmetric*, an intermediate waypose must be added. This new waypose is generated such that it is *symmetric* to both the current robot pose and the target waypose, and inserted as a temporary waypose between them (see section 6.3.3). The temporary waypose acts as the target waypose until either it and the current robot pose become non-symmetric, or the robot satisfactorily arrives at the temporary waypose. In either case, the original target waypose is rechecked to see if the robot is *symmetric*, and this process is repeated until the target waypose is achieved. Once the target waypose is achieved, a new waypose is taken from the list defining the robot's path and set as the new target waypose; this process is repeated until the list is empty.

6.3.3 Temporary Waypose Generation

When a temporary waypose must be placed, the next step is to determine of the robot's current pose $\vec{r} =< r_x, r_y, r_\theta >$ and the target waypose $\vec{w} =< w_x, w_y, w_\theta >$ are *parallel*.

Non-Symmetric Parallel Poses

The two poses, $\vec{r}$ and $\vec{w}$, are considered to be parallel if

$$\Delta_{threshold} > |r_\theta - w_\theta| \tag{6.3.2}$$

where r_θ and w_θ are the orientation components of $\vec{r}$ and $\vec{w}$, respectively. For cases in which the two poses are parallel, the new temporary waypose $\vec{t}$ will have an orientation defined as

$$t_\theta = 2 * \arctan(w_y - r_y, w_x - r_x) - \frac{r_\theta + w_\theta}{2} \tag{6.3.3}$$

Its position (t_x,t_y) will lay somewhere on the line segment defined by the points (r_x,r_y) and (w_x, w_y). A set of six evenly spaced discrete points along this line segment are selected as candidate temporary wayposes, $ct = \{\overrightarrow{c_i t} | i \in \mathbb{Z}, 0 \leqslant i < 6\}$ (see Figure 6.4b).

Non-Symmetric Non-Parallel Poses

If the two poses, $\vec{r}$ and $\vec{w}$, are not parallel, then the set of candidate temporary wayposes, ct, will lie along the edge of a circle on which both the current pose and the next waypose also lie (see Figure 6.4c). This circle is defined by Equation 6.3.4, and illustrated in Figure 6.5.

$$\begin{aligned}
c &= \cot(w_\theta - r_\theta) \\
center_x &= \frac{r_x + w_x + c * (r_y - w_y)}{2} \\
center_y &= \frac{r_y + w_y + c * (w_x - r_x)}{2} \\
radius &= |center_{(x,y)} - r_{(x,y)}|
\end{aligned} \tag{6.3.4}$$

Each candidate temporary waypose's orientation is then defined as a point along the edge of the circle as

$$c_i t_\theta = 2 * \arctan(c_i t_y - r_y, c_i t_x - r_x) - r_\theta \tag{6.3.5}$$

Each candidate element in the set of $\{\overrightarrow{c_i t} | i \in \mathbb{Z}, 0 \leqslant i < 6\}$ wayposes is tested to see if any obstacles lie along the path defined by the two arcs tangent to the candidate waypose (i.e., from the robot's current pose $\vec{r}$, and to the next target waypose $\vec{w}$). The first candidate that produces a collision-free path for the robot to follow is selected as the temporary waypose $\vec{t}$.

6.3.4 Obstacle Avoidance and Local Planning

Because the robot's autonomy does not always have direct control over the motors, we refer to the linear and angular velocities that would normally constitute driving commands as the autonomy's *intention*. The intention is calculated using the robot's current pose $\vec{r}$, laser range data, and the current target waypose $\vec{w'}$, defined as either the original target waypose $\vec{w}$ (if $\vec{r}$ is symmetric with $\vec{w}$) or the temporary waypose $\vec{t}$

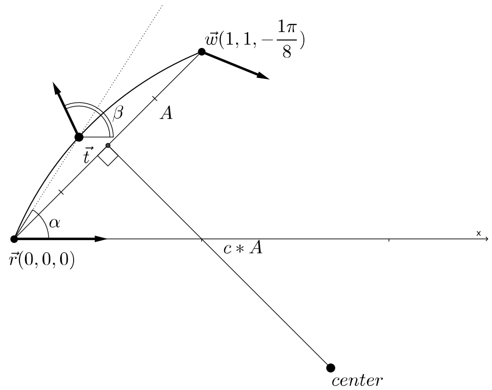


Figure 6.5: Two examples of non-symmetric non-parallel temporary waypoint calculations, showing the effect of changing the orientation of the target waypose has on the center calculation. $t_\theta = \beta = 2 * \alpha$

(if $\vec{r}$ is not symmetric with $\vec{w}$).

From subsection 6.3.2 and subsection 6.3.3, we can assume the robot's current pose $\vec{r}$ is tangent with the next waypose $\vec{w'}$. We can then calculate the intention as the linear and angular velocities needed to directly achieve waypose $\vec{w'}$, and test this trajectory (which we call the *direct path*) for collisions using laser range data.

If we detect that the robot will collide with an obstacle while traveling along the direct path before it reaches the waypose $\vec{w'}$, we calculate the distance from the point of impact (p_x, p_y) to waypose (w'_x, w'_y) and store this value as $d_0 = (p_x, p_y) - (w'_x, w'_y)$. We next test several pre-defined paths defined by arcs with radii $r = \pm[0.5, 1.0, 1.5, 2.5, 4.5, 34.5]$. For each arc a defined by a radius in r, we calculate the closest distance d_a as the shortest distance the arc a comes within reach of the waypose $\vec{w'}$ before a collision occurs. In the case where an arc a passes by the waypose $\vec{w'}$ before a collision, d_a is equal to the shortest distance from the waypose to arc. We then compare the distances calculated for each arc (including the original direct arc) and select the arc which brings us closest to the waypose as $\min(d_0, d_{a0}, d_{a1}, \ldots)$ (see Figure 6.6).

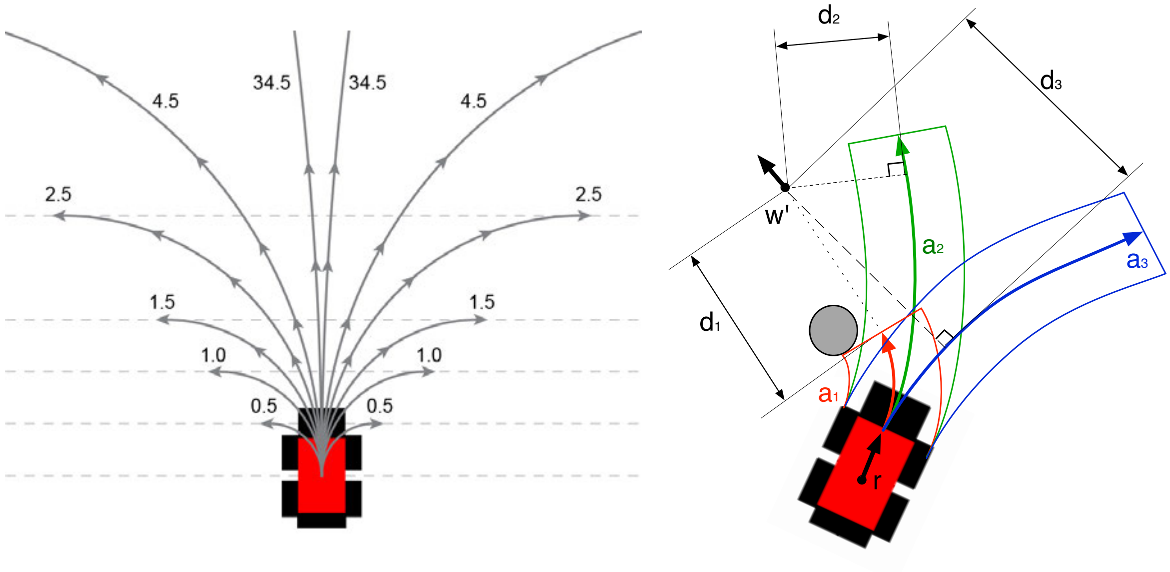


Figure 6.6: Obstacle Avoidance. *Left:* Several arcs are calculated *Right:* the arc that comes closest to the waypose in selected. In this example a_2 would be selected because $d_2 < d_1 < d_3$.

6.3.5 Complying with the User

As the robot approaches each of the eight course obstacles, the autonomy is presented with two fixed goal wayposes, one on either side. One of these fixed goal wayposes will also be in the robot's predefined path. Since control of the robot is shared with a human operator, the autonomy may be overruled at any point; that is, the user can steer in a different direction than the autonomy's calculated intention. There are numerous reasons why the operator may choose to override the autonomy, which can be broken down into two scenarios; we assume that the operator wants to drive down the hallway in the same general direction as the robot (i.e., the user does not want to drive in the opposite direction the robot is currently headed).

In the first scenario, the operator and autonomy both want to drive towards the same side of the next obstacle, but the operator wants to temporarily change the robot's course. Reasons for this situation ocuring might include wanting to move the robot closer or further away from the wall, or making more aggressive turns than the autonomy normally makes. The result is that once the operator returns control to the autonomy, the autonomy should continue steering towards the same fixed waypose.

In the alternative scenario, the operator disagrees with the autonomy about which side the robot should pass on the next obstacle. The most likely reason for this disagreement is the planned mistakes built into the experiment. In this case, because of the disagreement between the predefined waypose and the direction of the arrow in the remote environment, the operator will override the autonomy to steer the robot towards the opposite side of the obstacle. At this point, if the operator returns control to the autonomy, the expected action is for the autonomy to accept this action as a correction and modify its intention to steer towards the fixed waypose chosen by the user.

Whether or not the autonomy should comply with the user is determined by comparing the angle between the robot's current orientation, r_θ, and the current fixed waypose, w_θ, against a threshold. If this threshold is exceeded as the robot turns towards the opposite side of the obstacle, the robot will update the target fixed waypose.

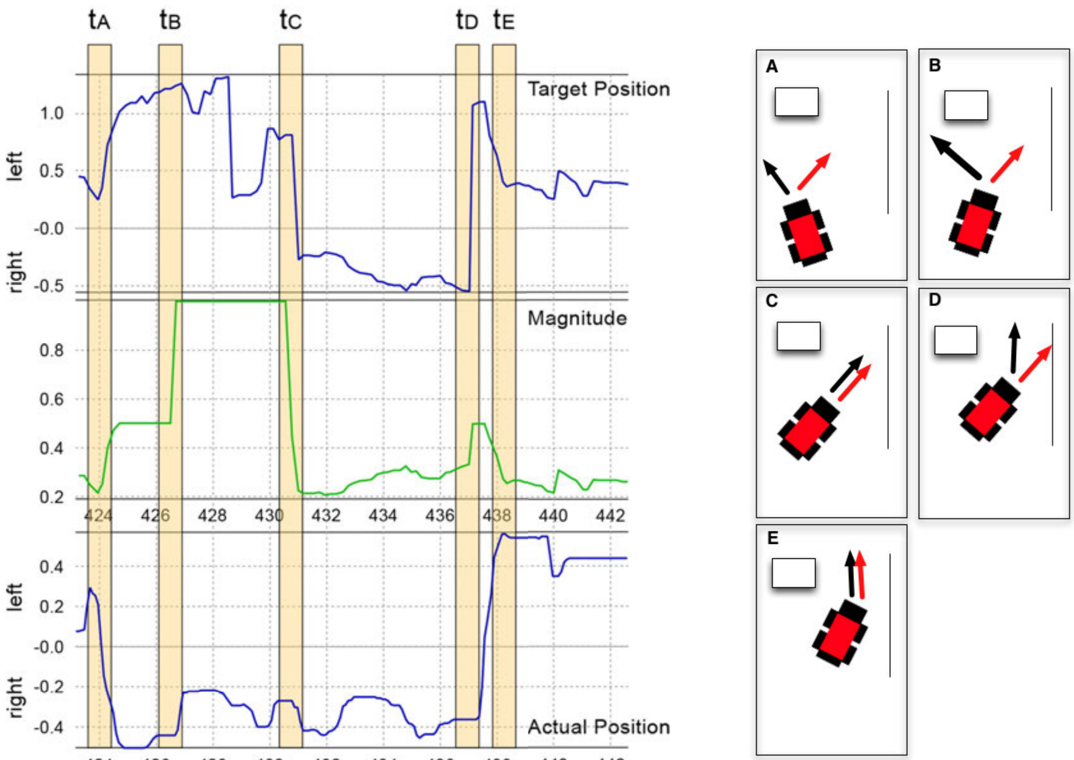
6.3.6 Autonomous Steering in the Non-Haptic Condition

In the non-haptic condition, the joystick's haptic feedback behavior emulates a mechanical joystick's spring centering force. The autonomy is given direct control of the motors by default and will self navigate through the course according to the predefined path at a minimum forward velocity. Whenever the joystick is centered or aligned in the positive vertical direction (i.e., pushed forward), the autonomy will retain control and accept the operator's linear velocity if greater than the minimum. Moving the joystick off-center or backwards revokes control from the autonomy. Once control has been revoked, the operator is granted full position-velocity control (teleoperation), where the vertical (forward-backward) axis controls linear velocity and horizontal (left-right) axis controls angular velocity. Unlike traditional teleoperation however, returning the joystick to a centered position or moving into a forward aligned position returns control to the autonomy, which will continue to drive the robot forward rather than the traditional behavior of coming to a stop. Due to the joystick emulating a spring centering force, if the operator releases the joystick it will self-center in the same way a mechanical joystick behaves.

6.3.7 Autonomous Steering in the Haptic Condition

In the haptic condition, the joystick has absolute control over the robot's motors, and both the autonomy and the human operator use the joystick to control the robot's actions. The autonomy is able to invoke joystick movement using the force feedback mechanism by specifying a target joystick grip location and maximum force magnitude to use in achieving the position (previously discussed in subsection 4.3.3). Autonomous steering forces are combined with a joystick centering force to create a single force effect that the operator can feel. However, unlike the non-haptic condition, if the operator lets go of the joystick grip in this condition, the joystick will not completely recenter. Instead, it will settle into a slightly forward position (necessary to allow the autonomy to continue driving the robot forward), and occasionally shift from side to side under self power to allow the autonomy to steer around obstacles while giving a haptic indication of the robot's intent.

Since the joystick directly controls the robot's motors, the operator can override



(a) Haptic forces. (b) Robot Location

Figure 6.7: Shared-Control Haptic Behavior Example. (Left) *Top:* Target joystick angular position, *Middle:* Magnitude of force (m), *Bottom:* Actual joystick angular position. (Right) Corresponding scenarios at t_n

the autonomy by pressing forcefully on the joystick grip. By pressing the joystick in a forward direction while still allowing the autonomy to move the joystick from side to side, the operator can influence the robot's speed while effectively allowing the autonomy to perform the task of steering. As the actual physical joystick grip location further deviates from the target joystick position determined by the autonomy's intention (i.e., due to the operator pushing on the grip), the haptic force applied to the joystick increases up to a maximum threshold.

The following example demonstrates the shared control behavior of the haptic joystick [Brooks et al., 2013]. The graphs in Figure 6.7a were taken from data collected during our study and show the actual angular position of the joystick grip as well as the desired haptic target and force magnitude exerted over time. In this scenario, the robot was initially steered towards the left (incorrect) side of an approaching obstacle; the autonomy intentionally encouraged the behavior as a result of a planned mistake. The participant, correctly identifying the robot's mistake, began to intervene and correct the mistake by pushing the joystick to the right (t_A). The autonomy, still believing the robot should be headed left, responded by increasing the magnitude of the force pushing the joystick left. As the participant deviated further from the autonomy's desired direction, it increased the force pushing left again (t_B) in an attempt to change the participant's actions. However, as the participant persisted in contradicting the autonomy, the autonomy modified its fixed waypose to comply with the participant's correction (t_C). As the robot crossed the front of the obstacle, it became necessary to turn left to continue down the hallway. The robot again exerted a leftward force (t_D), to which the participant complied. As the participant and autonomy both became satisfied with the rate of turn, the autonomy applied a weak centering force (t_E) to simulate the springs of a physical joystick.

6.4 Course Layout

The course consisted of a hallway 8 feet wide × 92 feet long, with eight obstacles placed at 9 foot intervals down the center of the hall (Figure 6.8). The robot drove down the hallway, made a u-turn, and returned to the starting point, weaving between the obstacles as in a solemn course. Each obstacle was a cardboard box measuring 9 inches wide × 30 inches tall × 20 inches long. Arrows on the boxes indicated on which side the robot should pass each one. Reflective barcodes were placed near the bottom of each

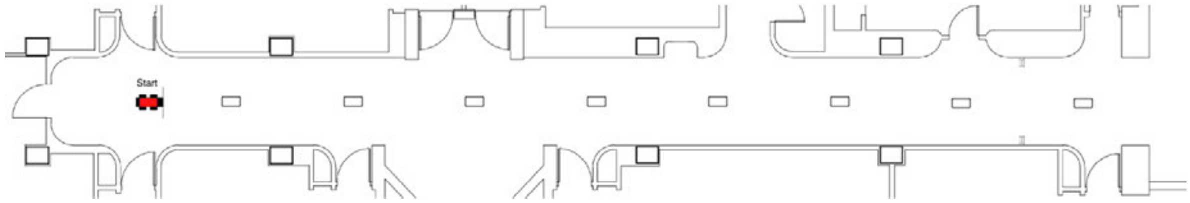


Figure 6.8: Course Layout

box. The robot could pass beside each obstacle with a combined clearance of 18.5 inches between the obstacle and wall (9.25 inches on each side). These parameters conform to the dimension calculations specified by the toolkit's template (see section 5.2). For a more descriptive diagram of the course setup, see appendix section B.3.

6.5 User Interface

The graphical user interface (GUI) used for our experiment was based on an existing interface used in our prior research [Desai, 2012] that met our toolkit guidelines. The GUI was used in all experimental conditions, and consisted of a main video feed, rear video feed, and distance display (see Figure 6.10). The main video feed featured a yellow crosshair with white lines extending out horizontally and vertically from it to indicate the direction in which the camera was pointed. The rear video feed was mirrored and placed at the top right of the main video window, making it like a rear view mirror in a car [Keyes, 2007]. A small runtime clock in the lower right hand corner of the display showed the elapsed time since the beginning of the run.

The distance display consisted of a top down view of the area around the robot. As the main video feed was turned from side to side, the distance display also rotated to indicate the direction the camera was pointed. The distance display consisted of a scaled icon of the robot in the center and black measurement lines extending out

(a) Haptic Location

(b) Physical Control Panel. Driving joystick is on the left, Camera joystick is on the right.

Figure 6.9: Operator Control Station

from each of the robot's four sides. White lines indicated distance measurements in meters to obstacles near the robot; these lines were calculated from the raw range data from the two lasers. Two vectors, colored blue and white, originated from the center of the robot icon. The blue vector indicated the target linear and angular velocity the autonomous system was trying to achieve. The white vector indicated the linear and angular velocity at which the robot was actually driving.

The physical controls consisted of two joysticks and two arcade-style push buttons (see Figure 6.9b). The left joystick was our 2 DOF Haptic Joystick Adapter for the Phantom Omni, and the right joystick is the COTS CH Products M11L061P device our joystick adapter was based on. These joysticks were used for driving (see section 6.3) and controlling the pan and tilt of the main video feed, respectively. The two buttons were located in front of the right joystick. The green button (on the right) re-centered the video camera. The black button was used by participants to note incorrect robot actions (i.e., whenever the robot tried to drive towards the wrong side of a box). The

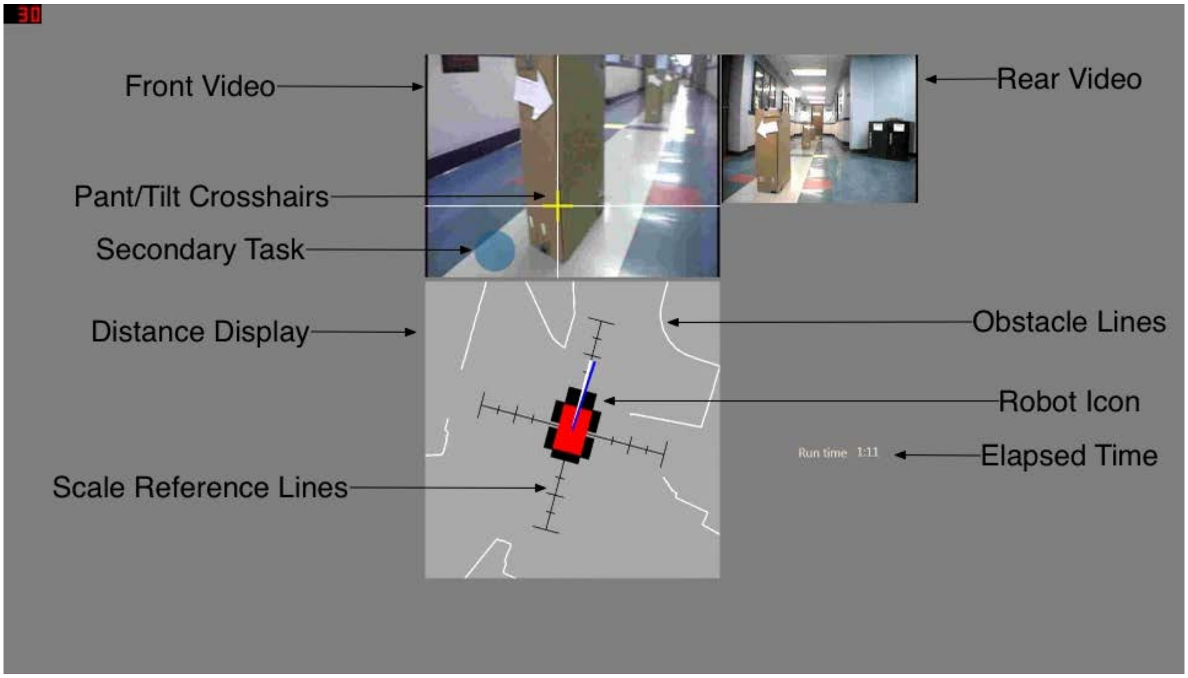


Figure 6.10: Graphical User Interface

COTS joystick and two buttons were interfaced with using an Arduino Uno programmed to stream data over USB to the control machine.

6.6 Tasks

The participant's primary task was to ensure the robot navigated the course following the correct path (as indicated by the arrows) while not hitting anything. Since both modes were autonomous, this task consisted of monitoring the robot's behavior and making corrections as necessary. If the robot did make a mistake, participants were instructed to press the black button and steer the robot towards the correct side of the box.

In addition, participants were asked to perform a secondary tracking task. Translucent blue circles, 70 pixels in diameter, were generated over the main video feed [Desai,

Figure 6.11: Example SAGAT Screen question

2012]. Every 35 seconds, a circle would appear in a pseudorandom location, a distance of 150 pixels away from the crosshair's current location. Whenever one of these circles appeared, participants were instructed to move the yellow crosshair over the circle so that it would disappear. This secondary task forced participants to move the front camera off-center.

Participants also needed to remember information relating to what was happening around the robot such as which side the robot passed the last box on, where the robot was on the course, if the robot had made any recent mistakes, and if the robot had hit anything. Four times during each run (twice in each direction), the robot was stopped and the interface was obscured. The participant was then presented with a series of questions related to information regarding the robot's current situation in the remote environment, as shown in subsection 6.8.1.

6.7 Compensation

Participants received $20 in base compensation for completing the experiment, and had the opportunity to earn an additional $20 based on their performance. For the performance bonus, participants were told that they would start with all $20 of the additional payment, and would lose $5 for each wrong turn the robot made, $1 for each time the robot bumped or scraped a box or wall, $0.50 for each missed tracking circle (i.e. the circle disappeared before they moved the yellow crosshair over it), and $0.50 for each SAGAT question they answered incorrectly. They were told these costs were averaged over all six runs, and then rounded up to the nearest $5.

The actual performance bonus did not penalize participants for incorrect SAGAT questions. Prior to being paid the compensation, participants were not given any indication of their performance level.

6.8 Data Collection

Based on our experiment template, we collected data using SAGAT Questions, pre and post experiment questionnaires, and post run questionnaires. In addition we the logged system state for both the user interface and robot, and recorded observations made by the robot handler.

6.8.1 SAGAT Questions

SAGAT is a technique for measuring an operator's SA, originally developed by Mica Endsley [Endsley, 1988] for air traffic controllers. The technique consists of pausing an experiment or simulation, blacking out the interface, and asking the operator a series

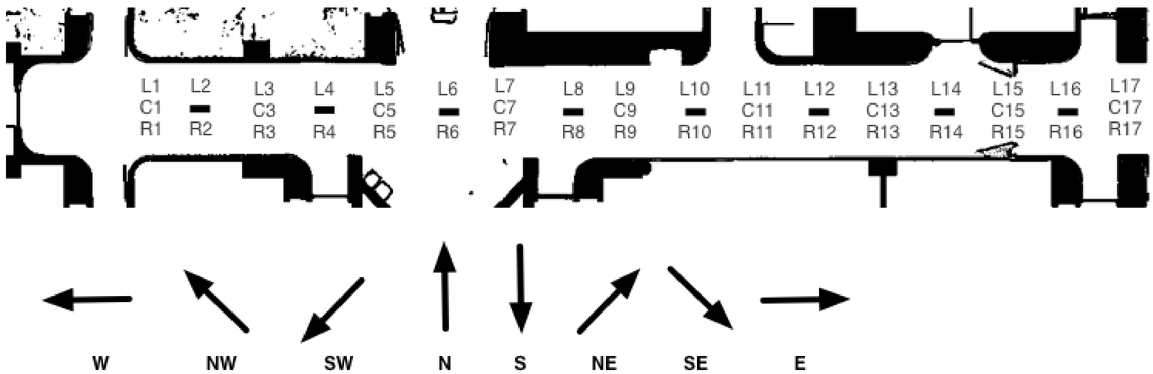


Figure 6.12: SAGAT Question 15: First click on the map the area that best represents the robot's position. Then click on the arrow that best represents the robot's current direction.

of questions related to the current situation.

Due to limitations in the way the software for triggering the SAGAT screens was written, it was necessary to make the location of the second screen (in the forward direction) closer to the u-turn than the location of the first screen (in the return direction) for each condition.

We developed a total of 16 SAGAT questions based on the tasks in this experiment. There are 14 multiple choice questions (four SA level 1 questions, six SA Level 2 questions, and four SA Level 3 questions) and two additional map-based, multi-dimensional questions (both SA Level 2). One multi-dimensional question required the participant to draw the position and orientation of the robot on the course (see Figure 6.12), and the other asked the participant to select the location of the box nearest the robot from 14 discrete positions (see Figure 6.13).

A question was classified as SA Level 1 (perception) versus SA Level 2 (comprehension) if it could be correctly answered by looking at the operator interface, if it was not hidden from view. For example, question 14 (Figure 6.13) is not SA Level 1 because there exist configurations in which being able to see the display would not help to answer the question. Specifically, if the closest box was beside the robot, the blind spot between the front and back lasers would make the box seem to disappear. If this situation was paired with the camera being pointed toward a wall, the participant would not have a way of telling where the closest box was.

Multiple choice questions were presented to the participant on an opaque colored screen, obscuring the user interface (see Figure 6.11). A total of eight questions appeared at each SAGAT screen, shown one at a time in a randomized order (see Figure 6.11). For the eight questions asked on each screen, two were SA Level 1 questions, four were SA Level 2 questions, and two were SA Level 3 questions. Additionally, one of the four SA Level 2 questions was a map question. Some questions had limitations on when they could be asked due to being context specific, as noted in Table 6.1.

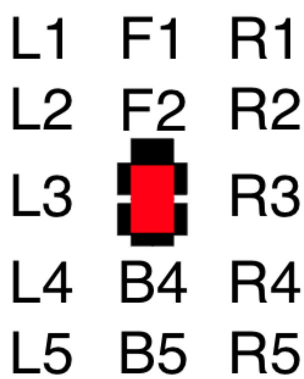


Figure 6.13: SAGAT Question 14: Please click on the area that best represents the location of the box nearest to the robot, keeping in mind that the closest box may be behind you. The square in the center represents the robot.

Num	SA Level	Question	Restrictions and Notes
0	2	What side did the robot pass the last box on?	—
1	2	On which side did the robot pass the second to last box?	—
2	2	Since (the start of the run/the last black screen), has the robot crossed the line of boxes down the middle of the hallway?	Must not be asked on the first screen after the u-turn (could cause ambiguity).
3	1	Is the front camera position currently reset?	—
4	2	Since (the start of the run/the last black screen), has a blue dot in the video screen disappeared before you had the chance to hit it? (Excluding any on the screen when the screen went black)	—
5	2	Since (the start of the run/the last black screen), has the robot bumped or scraped into a box or wall ?	—
6	1	When the screen went black, were you driving the robot at full speed?	Full speed was defined as the joystick stick being pushed all the way in any direction.
7	1	Have you passed the u-turn yet?	Must not be asked just after the beginning of the run, just before the end of the run, or just before the u-turn.
8	1	Are there less than 4 boxes remaining in the direction you are currently headed?	Must not be asked just after the beginning of the run, just before the end of the run, or just before the u-turn.
9	2	Since (the start of the run/the last black screen), have you passed any boxes on the wrong side?	—
10	3	Since (the start of the run/the last black screen), have you had to correct the robot (press the black button or change the robot's course)?	—
11	3	Has the robot's autonomy tried to steer towards the wrong side of the next box?	Must be asked between boxes.
12	3	Did the robot read the barcode on the next box correctly?	Must be asked between boxes.
13	3	Would it be safe to drive the robot forward at full speed for the next 2 seconds?	—

Table 6.1: Single Dimension SAGAT Questions. Response options were "yes," "no," and "I don't know."

Num	SA Level	Question	Restrictions and Notes
14		Please click the area that best represents the location of the box nearest to the robot, keeping in mind that the closest box may be behind you. The square in the center represents the robot. *Additional Verbal instructions:* Rows 1 and 5 represent the closest box being at least one full robot length away. L3 and L5 represent the closest box being directly beside the robot.	See Figure 6.13
14a	2	Is the letter (L, F, R, or B) the same?	Graded as "yes" or "no"
14b	2	Is the row number the same?	Graded as "yes" or "no"
14c	2	Is the answer with one Manhattan distance of the correct answer?	Graded as "yes" or "no"
15		First click on the map the area that best represents the robot's position. Then click on the arrow that best represents the robot's current direction.	See Figure 6.12
15a	2	Is the selected location located within the same row (Left, Center, or Right) as the correct answer?	Graded as "yes" or "no"
15b	2	Is the selected location within the same column (1-17) as the correct location?	Graded as "yes" or "no"
15c	2	Is the selected direction the same as the correct direction?	Graded as "yes" or "no"

Table 6.2: Multi-Dimensional SAGAT Questions

6.8.2 Questionnaires

Three questionnaires were used during the experiment (see section B.4). The questionnaires were administered electronically using custom software. The software required that all questions on a page be answered before moving to the next page. Participants were encouraged to take as much time as they needed to complete the questionnaires.

The pre-experiment questionnaire was primarily concerned with collecting demographic information. Of particular interest were questions about past experience with technology, remote control cars, and video games, as well as their attitude towards using technology [Parasuraman, 2000] and taking risks [Mick and Fournier, 1998] (see

Table B.1 and Table B.2). In addition to the pre-experiment questionnaire, the ETS Cube Comparison test [ETS, 1976] was administered to evaluate participants' spacial reasoning (SR) skills.

The post-run questionnaire consisted of a performance self-assessment, the Muir trust scale [Muir, 1989], and NASA TLX workload questions [Hart and Staveland, 1988] (see Table B.3). This questionnaire was administered after each usage of the robot, including the trial runs.

The post-experiment questionnaire was issued after the participant completed the final post-run survey. This questionnaire examined how participants felt about the two control modes (i.e. haptic and non-haptic) and their ability to use them (see Table B.4).

6.8.3 Logged Information

Over 100 log files and 5GB of data were collected for each participant. Data was collected from the following sources:

- Video/Audio - Zoomed in video camera recording of the participant's hands on the physical controls (i.e., two joysticks and two buttons)
- Video - Screen capture of the participant's driving interface
- System State Recording (ROS bag files) consisting of time indexed data such as robot position and pose, joystick position and feedback forces, and the current waypose target
- Electronic Surveys (pre-experiment, post-run, and post-experiment surveys)
- Cube Comparison Spatial Reasoning Test
- Driving Interface activity log files

- SAGAT Screen responses and correct answers
- Robot Handler Android App Logs
- Scanned Notes taken by the experimenter

An 1800 line Python script was written to aggregate and cross verify information from text file data from over 60 files for each participant. This script generated an aggregate xml data file over 800 lines long for each participant. This information was then imported into a database for querying.

6.9 Procedure

Because the experiment involved operating a remotely located robot, running the experiment required two people - an experimenter and a robot handler. The robot handler was located with the robot. This job included manning the physical emergency stop button and recording the number of times the robot hit an obstacle or took a wrong turn. Between runs, the robot handler reset the robot and course according to the participant condition table (see section B.2).

The experimenter was located with the participant and administered the questionaires. Additionally, he or she was responsible for starting and stopping the software running on the robot, monitoring experiment progress, and trouble shooting the robot. The robot handler and experimenter communicated using 2-way radios. Each session took between 2 and 3 hours to complete.

6.9.1 Pre Experiment Setup

The first trial course was set up prior to the participant's arrival. The course diagram in appendix section B.3 was used to place the boxes in their correct locations in the hallway. Arrows were placed on either side of each box according to the chart in section B.3. Next, the robot was moved into the starting position, along with its charging system. Once the robot was in position, the wireless access point was placed in the hallway and connected with the robot. Light fixture covers were placed over the lights in the stairwells to mitigate interference in the 5.7 to 5.8GHz range.

The experimenter initialized the electronic survey, specified a participant number (e.g. 21), creating an information directory (e.g. ~/Desktop/data/P21), and finally started the "PreExperiment" survey. Next, the experiment followed the instructions found in section B.5 to bring the rest of the system online.

6.9.2 Experiment Protocol

When the participant arrived, he or she was asked to sign an informed consent form, and fill out a pre-experiment survey consisting of demographic information and a spatial reasoning test [ETS, 1976]. Next, the experimenter read aloud from a script which explained the operator interface and task to the participant (see section B.6). The participant was then asked to summarize the instructions. The participant completed two trial runs, once in each driving mode, using the trial specific map and variation. During the trial runs, participants were encouraged to familiarize themselves with the controls and ask questions. The same map and event sequence variation (see Figure 5.3) was used for all participants in both conditions. Once the trial runs were completed, the participant drove the robot a total of six more times, alternating between control modes for each run.

During each run, the robot handler marked each time the robot hit an obstacle or wall or took a wrong turn using custom logging software on an Android phone. This information was sent back after each SAGAT screen was displayed, as well as at the end of the run so that SAGAT questions concerning this information could be evaluated.

The experimenter monitored the progress of each run to ensure that the system was operating properly. Because the system relied on relatively fast network speeds to keep the haptic system stable, the experimenter watched ping times between the server and robot to make sure this effect did not play into interaction between the interface and participant. Additionally, the experimenter monitored the robot's localization and way-point generation, and made notes about anything unusual or interesting that happened during the experiment. During SAGAT screens, the experimenter manually marked the correct answer to multi-dimensional map question asked.

After each run, including the trial runs, participants were asked to fill out a post-run survey. The experimenter would then stop the rosbag logging, shut down the user interface, shut down the robot backend, and place the robot into teleop mode to be moved back into the starting position by the robot handler. Once the robot was in the starting position and plugged in to charge, teleop was shut down and the system was brought back online again for the next run.

Due to the length of the experiment, participants were encouraged to take as many breaks as they would like, and sat while answering the pre-experiment, post-run, and post-experiment questionnaires. After the post-run survey for the last run, participants were given a post-experiment survey to complete. Finally, participants were given an opportunity to ask any remaining questions and were paid compensation in the form of Amazon gift cards.

6.10 Results and Discussion

Although our system has been used by more then 40 people who have logged well over one hundred hours of combined operation time, only 18 of these people participated under the exact conditions described in this chapter and are therefore reported on in this study.

6.10.1 Demographic Analysis

A total of 18 people participated in our experiment – 12 males and 6 females. The average age was 21 with a standard deviation of 2 years. Of the 18 participants, 15 reported being students, 2 worked in retail, and one did not respond. There were 16 right hand dominant participants, 1 left hand dominant, and 1 ambidextrous participant. We had intentionally placed the haptic joystick on the left hand side for two reasons: First, so that people would be forced to interact with the device with their less dexterous hand, and second, to match the movement control on the left and video control on the right paradigm used by video games. All but one participant reported English as their primary language.

Our participants were experienced with technology (Figure 6.15) and had positive attitudes towards technology in general (Figure 6.16). Twelve participants reported having seen robots in person prior to participating in this study. All but 3 participants reported spending at least 20 hours per week using a computer. Only 4 reported spending more then 10 hours per week playing video games (see Figure 6.14).

Participants could be categorized based on whether they had high or low: 1) spatial reasoning abilities (Figure 6.17) or 2) risk seeking tendencies (Figure 6.18) spatial reasoning abilities, however these factors do not appear to be dependent upon each other.

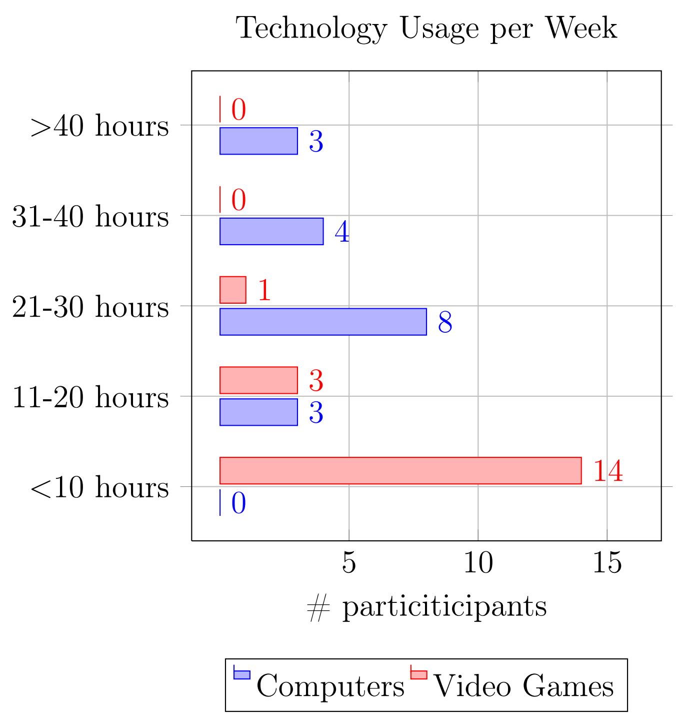


Figure 6.14: Technology Usage per Week

6.10.2 Dependent Variable Analysis

As previously mentioned, our hypothesis was that users would have better SA and task performance while operating the robot with the haptic joystick than with the non-haptic joystick. We also hypothesized that the users' workload would be lower with the haptic joystick, and that users would recognize and correct robot mistakes more quickly when using a haptic joystick.

We found that participants completed the slalom task significantly faster when using the non-force feedback control mode (p<0.001, $t(107)$=5.77 using a two-tailed paired t-test with α=0.05). The participants and also had fewer hits using the non-force feedback joystick (p<0.01, $t(107)$=2.48). Thus, our hypothesis was not supported.

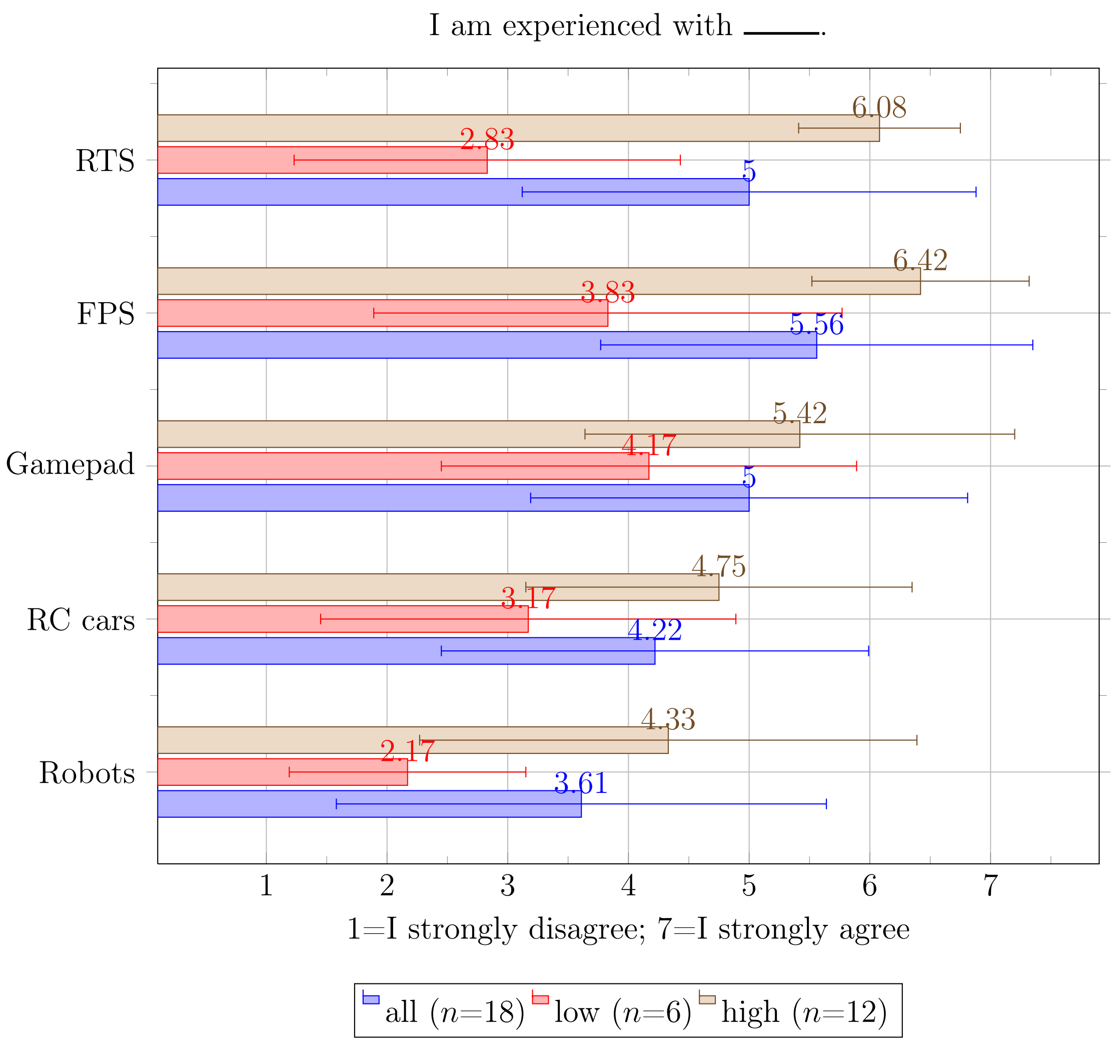


Figure 6.15: Experience with Technology ($\bar{x} \pm 1SD$). *Low*: Below average participants, *High*: Above average participants. See Table B.2 for questionnaire.

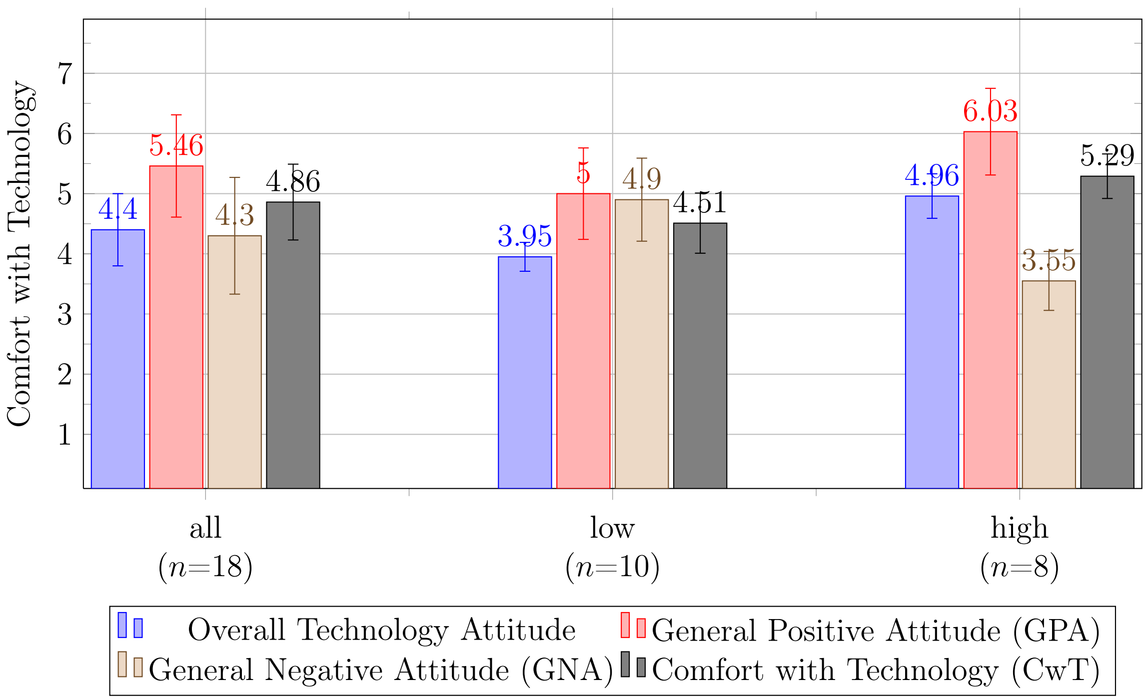


Figure 6.16: Comfort with Technology ($\bar{x}\pm 1SD$). Scale from 1=low to 7=high. Overall Technology Attitude (*all*) $\bar{x} = 4.4$, calculated as the average of the GPA ($\bar{x}$=5.46),GNA ($\bar{x}$=4.3), and CwT ($\bar{x}$=4.86) subscales. CwT and GNA subscales were appropriately negated for the calculation. GNA showed here before negation.

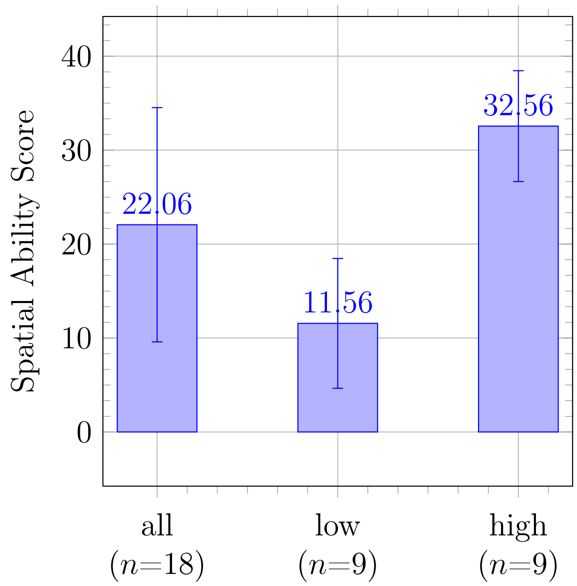


Figure 6.17: Cube comparison ($\bar{x} \pm 1SD$). Score=$num_{correct} - num_{incorrect}$. Overall $\bar{x}$ = 22.06, $\bar{x}$ of participants below average = 11.56, $\bar{x}$ of participants above average = 32.56

.

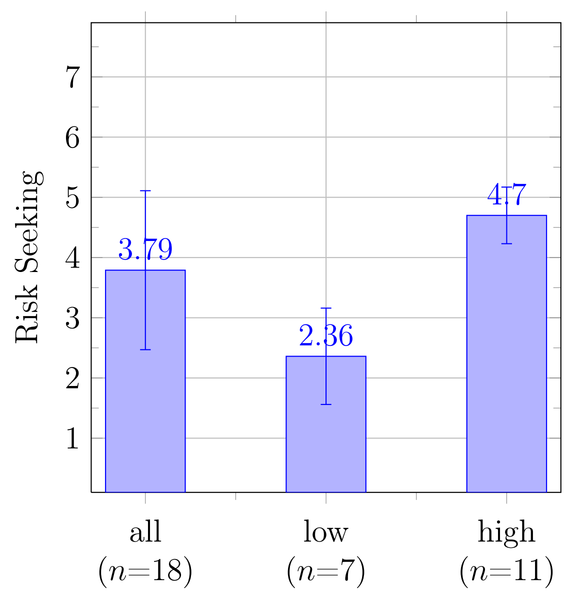


Figure 6.18: Attitudes towards risk ($\bar{x} \pm 1SD$). Scale from 1=low to 7=high. $\bar{x}$ all participants = 3.79. $\bar{x}$ for all participants less then overall average = 2.36. $\bar{x}$ for all participants greater then overall average = 4.7

.

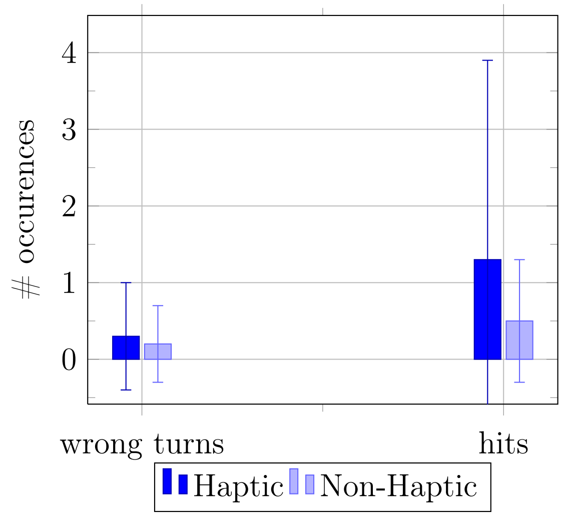


Figure 6.19: Task Performance: Wrong turns and hits ($\bar{x} \pm 1SD$). $\bar{x}$ wrong turn/haptic $= 9.3$; $\bar{x}$ wrong turn/non-haptic $= 0.2$; $\bar{x}$ hits/haptic $= 1.3$; $\bar{x}$ hits/non-haptic $= 0.5$

The difference in task completion time can be explained by the nature of the two driving modes. In the haptic mode, the robot's autonomy would push the joystick about half way forward and from side to side in order to steer the robot. For the participant, we observed that the easiest action would be to allow the robot to autonomously navigate the course, and to loosely place their fingers on the grip and allow the joystick to move their fingers around. For the robot to travel at full speed in this mode, the participant would need to push the joystick all the way forward to its maximal stop point. It is more difficult to feel the side to side forces that the autonomy applies for steering. Therefore, we noted that many participants were willing to sacrifice the robot's speed to make it simpler for them to allow the autonomy to perform the driving task.

In the non-haptic mode, the joystick could remain centered or in any forward vertically aligned position for the robot to autonomously navigate the course, with the joystick position dictating the robot's speed. In this mode, the participant could push the joystick all the way forward and rest it against the hard stop, simultaneously re-

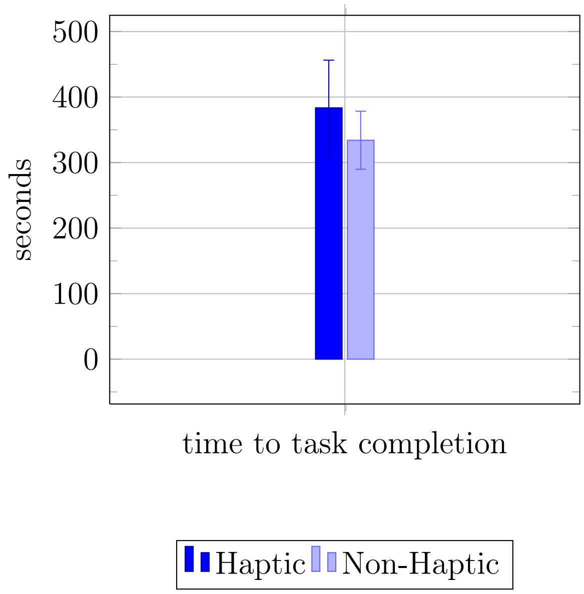


Figure 6.20: Task Performance: Time to completion ($\bar{x} \pm 1SD$). $\bar{x}$ time to completion/haptic = 383.6; $\bar{x}$ time to completion/non-haptic = 334.1

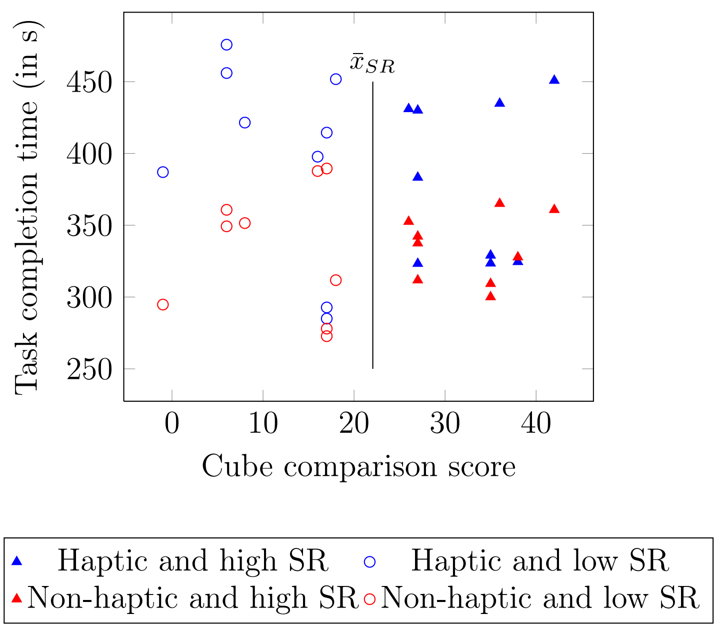


Figure 6.21: Spatial Reasoning (SR) × Task Completion Time. Blue represents the haptic condition, and red the non-haptic one. The triangle represents participants with high spatial reasoning skills (defined as having scored greater than or equal to the sample mean $\bar{x}_{SR}$), and the open circle those scoring below. No correlation exists between spatial reasoning and time to task competition in either haptic condition.

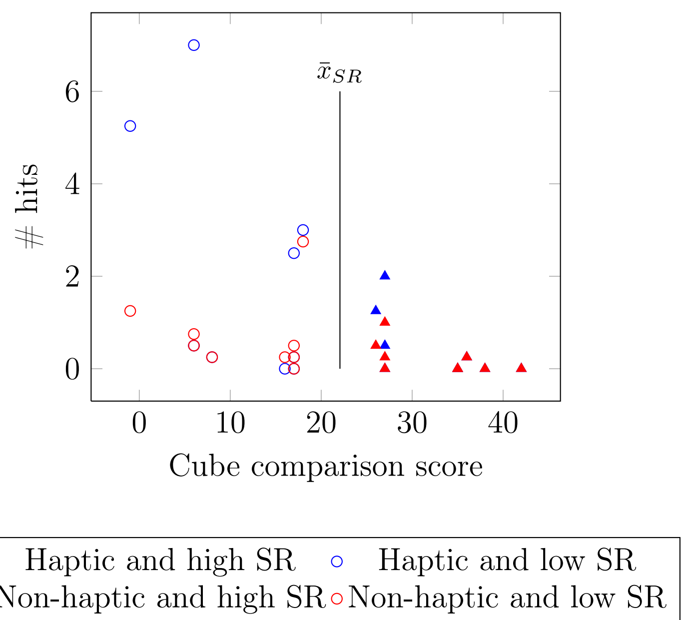


Figure 6.22: Spatial Reasoning (SR) × Hits. There exists a negative strong correlation between spatial reasoning and hits for the haptic joystick condition (Pearson's r=-0.58, p<0.01, $t(16)$=-2.88 using a two-tailed t-test), where strong defined as is $|r| > 0.5$. There is also a medium correlation in the non-haptic condition (r=-0.40), where medium is defined as $|r| > 0.3$; however, this correlation has weak significance (p<0.10, $t(16)$=-1.76 using a two-tailed t-test).

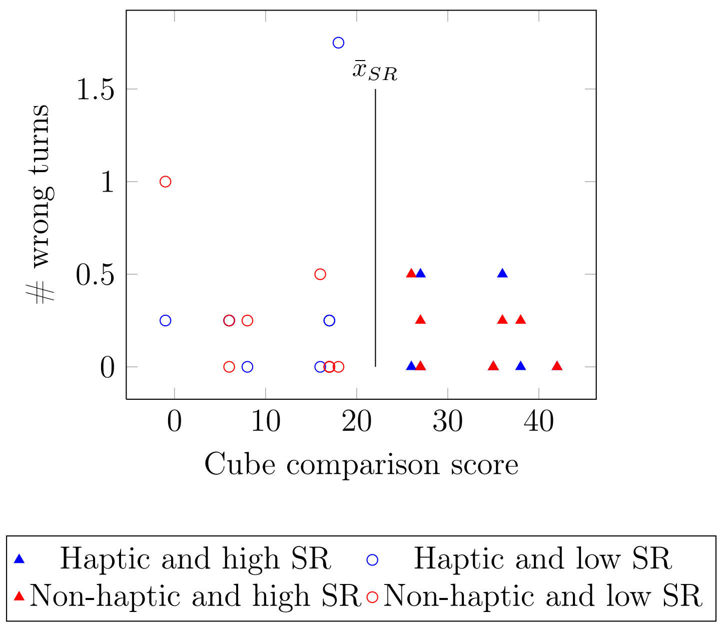


Figure 6.23: Spatial Reasoning (SR) × Wrong Turns. No correlation exists between spatial reasoning and the number of wrong turns in the haptic condition. However, there is a medium correlation between them in the non-haptic condition (r=-0.40, although only weakly significant p<0.1, $t(16)$= -1.76 using a two-tailed t-test).

ducing the dimensionality of the driving task and speeding up the robot's autonomous pace at the same time.

Figure 6.24 indicates no significant difference in participants' reported cognitive workload overall between the haptic and non-haptic control methods (Wilcoxon matched-pairs signed-ranks test; $p \leq 0.57$, $n=52$). This result means our third hypothesis is also not supported; however, we are encouraged that our first haptic shared-control behavior did not cause our participants' workload to increase, and we discuss options for further developing and refining this behavior in section 7.3.

Participants reported having significantly higher levels of confidence in communicating their intentions to the robot using the non-force feedback joystick ($p<0.003$, $t(17)=3.46$ using a two-tailed paired t-test with $\alpha=0.05$) (see Figure 6.25). They also felt more in control of the robot using the non-force feedback joystick ($p<0.024$, $t(17)=2.47$).

We saw no overall or level specific significant differences in the situation awareness as measured by the SAGAT technique (see Figure 6.26). This result is interesting because it provides evidence that the presence of haptic feedback does not necessarily effect an operator's understanding of the state of the remote robot and/or environment compared to a non-haptic implementation of the same system.

Finally, we found that our participants preferred the non-haptic implementation over our haptic implementation (see Table 6.3). This preference was usually determined by the second trial ($n=14$) and remained the preference for the remaining runs. At this point, it is worth reiterating that the underlying autonomy was exactly the same for both implementations, and that the particular haptic implementation was designed to be as similar as possible to the non-haptic implementation. Based on comments both recorded by the experimenter and submitted in our surveys, participants seemed to be

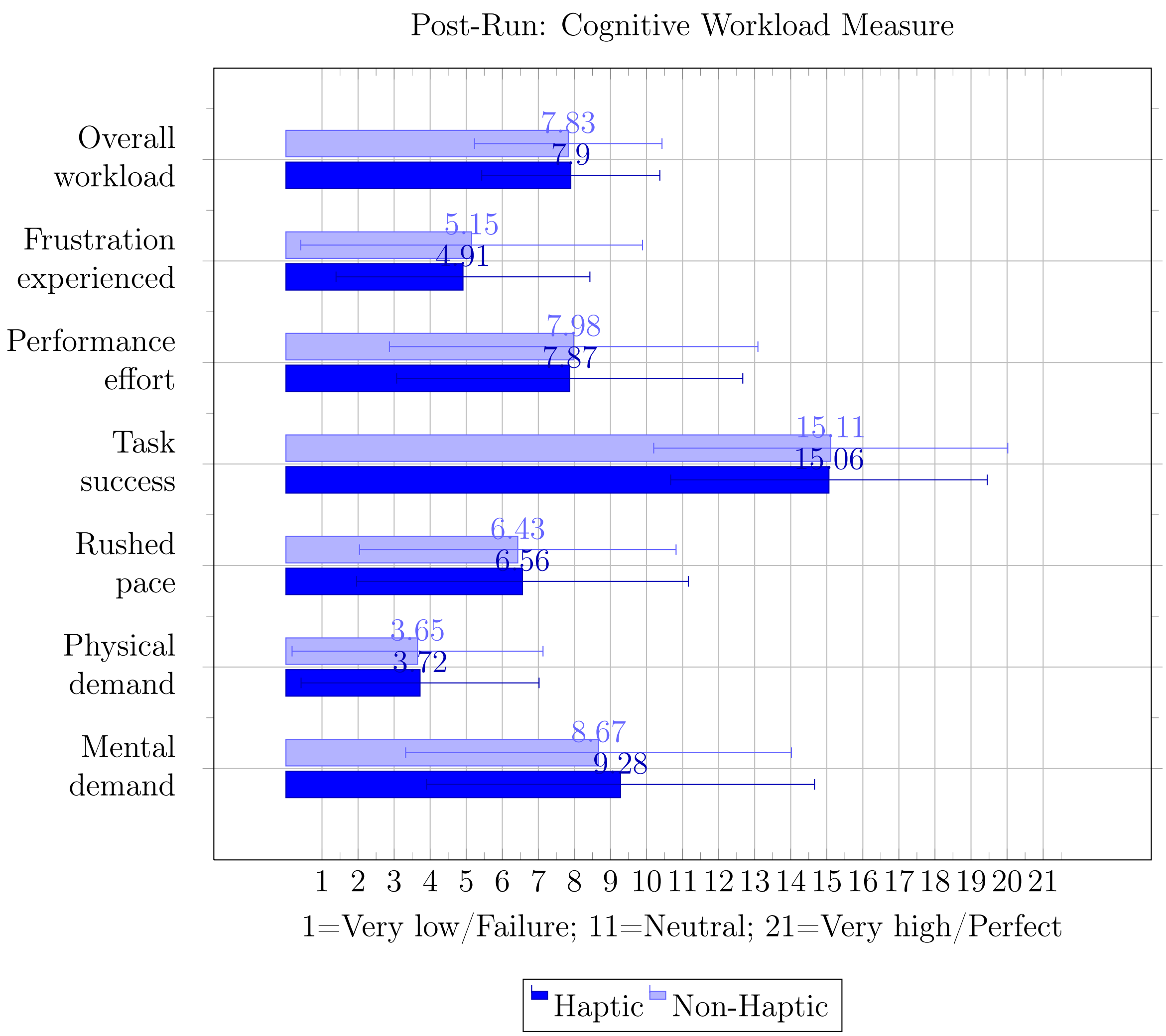


Figure 6.24: Task Load Index ($\bar{x} \pm 1SD$). No significant differences were found between the haptic and non-haptic control modes. Overall workload is defined as the average of the six scales items [Hart, 2006].

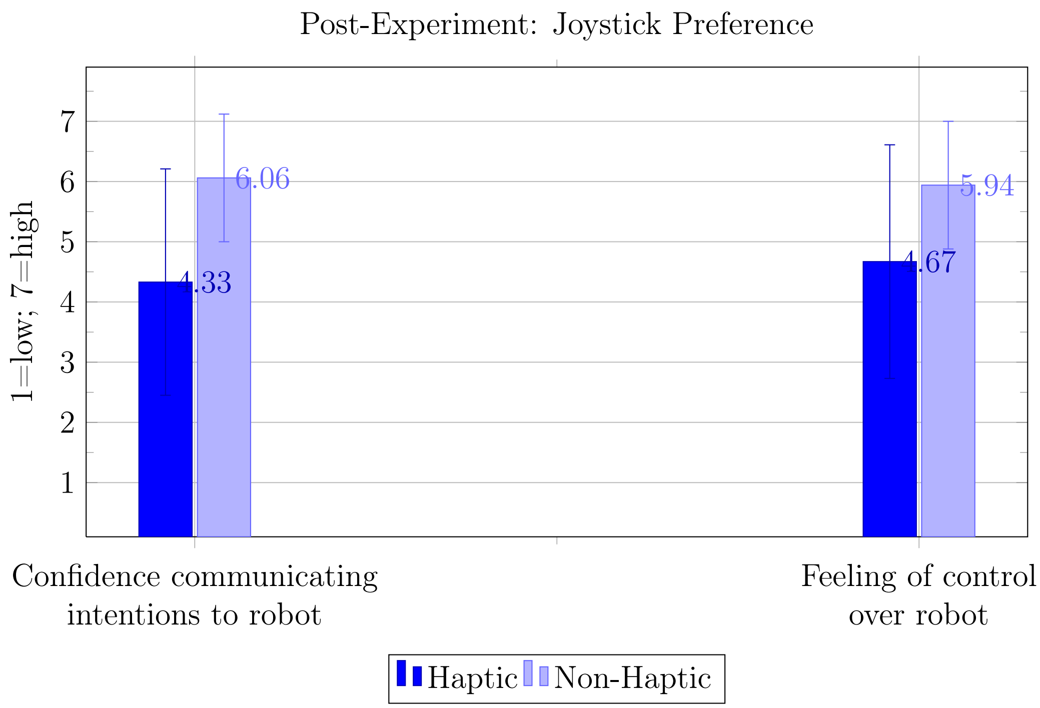


Figure 6.25: Joystick Preference ($\bar{x} \pm 1SD$). $\bar{x}$ for confidence/haptic = 4.33; $\bar{x}$ for confidence/non-haptic = 6.06; $\bar{x}$ for control/haptic = 4.67; $\bar{x}$ for control/non-haptic = 5.94

.

Table 6.3: Post-Run Joystick Preferences

Haptic mode during run:	T1	T2	R1	R2	R3	R4	R5	R6
Force feedback joystick preferred	3	2	0	4	1	0	1	2
Non-force feedback joystick preferred	2	7	8	5	6	6	5	7
No preference	4	0	1	0	2	3	3	0
Non-haptic mode during run:	**T1**	**T2**	**R1**	**R2**	**R3**	**R4**	**R5**	**R6**
Force feedback joystick preferred	0	2	2	0	2	2	2	1
Non-force feedback joystick preferred	2	7	5	9	7	7	7	6
No preference	7	0	2	0	0	0	0	2

most frustrated by the difficulty of controlling the robot when they felt it was necessary to override the system autonomy.

6.10.3 Implications

Our haptic device modification has proven to be robust and seems sufficiently strong. The joystick has withstood the use of more then 30 different users and well over 60 hours of operation in addition to time spent developing and testing. Likewise, the joystick software has also shown itself to be very robust. We used our own toolkit to develop a novel shared control haptic feedback effect which also proved to be sufficiently robust. Due to our assumption that we could overcome problems related to passivity using a high speed network, our software was prone to exhibit slightly erratic behavior when exposed to latencies in excess of 200ms delays. Only during a handful of occasions was the latency significant enough to prevent the operator from completing the run, and in these cases the latency was well over 2000ms; the result of interference radiating on the same wireless frequency we were operating the robot. Although our results did not match our expected hypothesis, the experiment illuminated aspects of HRI haptic research we had not previously considered, which we discuss further as open research questions in chapter 7.

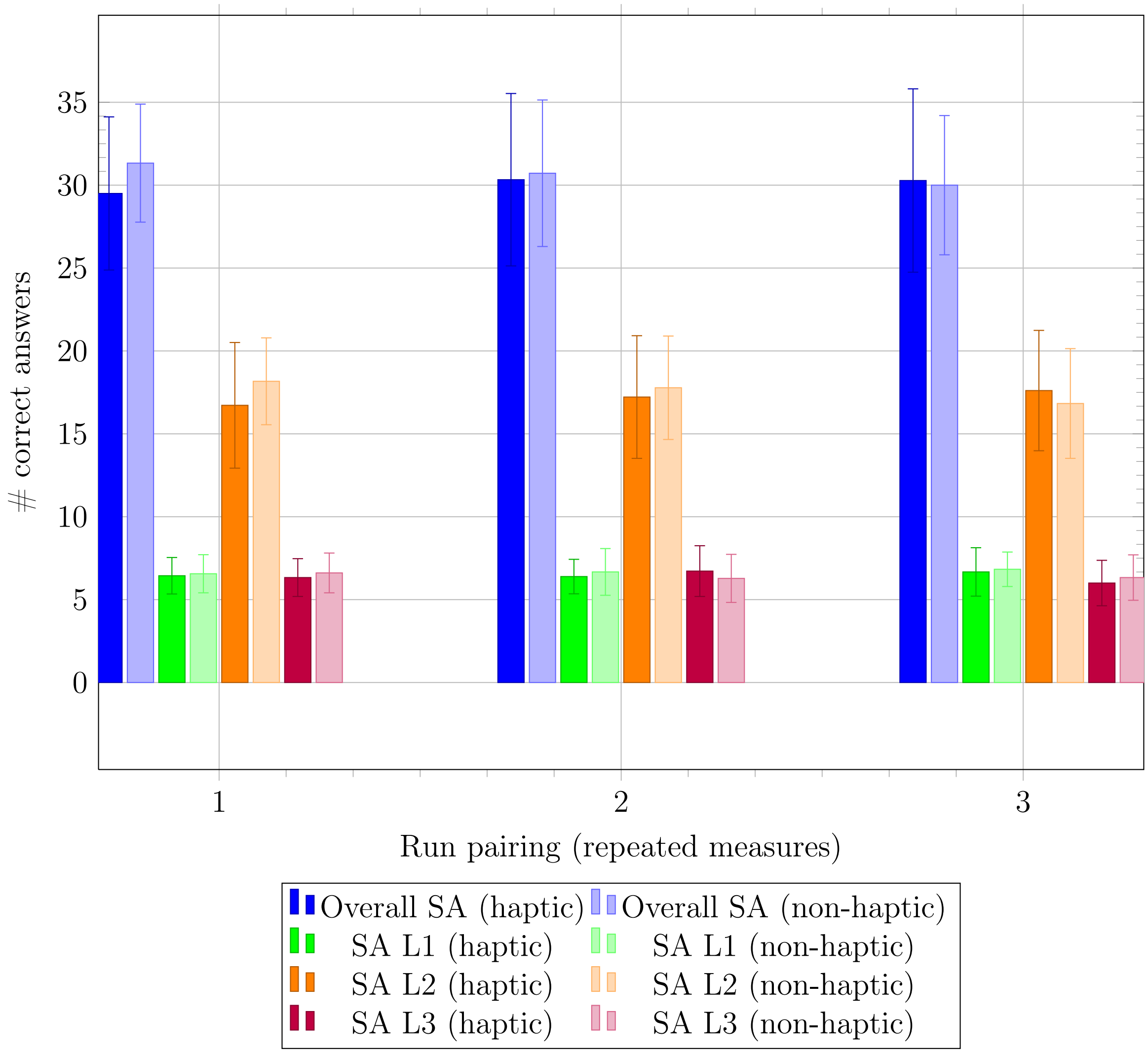


Figure 6.26: SAGAT ($\bar{x} \pm 1SD$). For each run there were 4 screens. At each screen, there were 2 SA Level 1, 4 SA Level 2, and 2 SA Level 3 questions. Values for each run show all correct questions of each SA level summed across all four SAGAT screens. We found no significant difference in the number of correct SAGAT question responses between the haptic and non-haptic control modes. This was also true when using time as a covariant (i.e., run pairing repeated measures). No significant differences existed for overall SA (defined as the sum of correctly answered items for the 4 SAGAT screens in each run), or the L1,L2, and L3 groupings. Our H2 hypothesis was not supported. However, like with workload, our participants' SA did not decrease, and we view these results as a baseline.

Chapter 7

Conclusions

7.1 Contributions

The primary contribution of this thesis is our three part open source haptic behavior evaluation toolkit, consisting of:

1. A hardware adaptation to convert the Phantom Omni into a 2D joystick with the same form factor as the ubiquitous device used to control mobile robots,

2. A corresponding software stack for writing 2D haptic joystick behaviors using our device, written in the nearly ubiquitous Robot Operating System (ROS) to allow easy integration with a wide variety of existing software projects, and

3. A ready-to-use experiment template for running within-subjects user studies to test newly developed haptic effects for ground mobile robots. This experimental design is complete with fully counter-balanced participant conditions and measurement techniques, and has been successfully employed by the research in this thesis and [Desai, 2012].

Our toolkit has been designed to help bring consistency to the experimentation procedures currently being used by haptic researchers investigating mobile robot teleoperation and navigation. This consistency will allow the community to better understand how various haptic behaviors compare with one another and effect a user's experience.

Using our toolkit, we created our second contribution - a novel shared-control haptic behavior for controlling mobile ground robots. The behavior creates fluid driving actions for navigating a known set of wayposes that is capable of guiding an operator along a predefined path. At the same time, the behavior is compliant with operator interventions. Our haptic feedback behavior reflects the robot's internal intentions for shared-control of the autonomous navigation.

Our third contribution is a thorough user study, designed around our the experiment template specified in our toolkit. The user study examined the differences between a haptic and non-haptic implementation of a single shared-control algorithm. The study found no significant differences in the operator's workload or situation awareness measurements between our haptic and non-haptic conditions. However, the study did discover a significant difference in operator performance, with the haptic implementation having an increased number of collisions and longer time to task completion. These findings have brought to light differences in the manner which participants used the controls and have provided insights to ways the haptic feedback behavior design could be improved to match users' expectations.

7.2 Limitations

It is important to define the limitations of the work being presented to prevent inappropriate extensions. The hardware modification was designed specifically for the SensAble

Phantom Omni. While it may be possible for the design to be ported to other devices, this was not our intention during the design process. Additionally, the form factor of the joystick modification is much bulkier than the model finger joystick it was based on. In its current form, it is unsuitable for placing in handheld control units or field use, which is unfortunate, since many commercial mobile robot platforms for military and police use are now making use of handheld units.

The software stack provided in our toolkit has been specifically designed for use with our specific hardware modification, and consequently was not intended to be generic. However, the software stack has been designed such that (1) hardware specific functionality is segregated from the 2D to 3D software, (2) parameter values are easily changed, and (3) our calibration routine minimizes the number of assumptions that were made, which are detailed in this thesis. Modular software design may allow our software stack to be ported to other hardware modifications similar to ours, although this was not the intended functionality.

Our experiment template was intentionally constrained to the task of mobile ground robot teleoperation and navigation supervisory control. Additionally, the use of the visual channel is minimized. While beneficial for advancing the ability to conduct haptic research and replicating experiments, it is also easy to fall victim to "science in a bottle" ecological validity; haptic behaviors should be designed for use in the context of the real world.

7.3 Open Research Questions

The experiment described in the previous chapter focused specifically on a single haptic implementation. During the course of preparing for, running, and evaluating the experiment, many more fascinating ideas were put forward that were unfortunately outside the scope of this work. These ideas range from variations of our current haptic implementation that were suggested by participants of our study, to additional research questions and experiments that might be used for investigating these new questions.

7.3.1 Haptic Variations of Our Shared Control System

We would like to investigate the use of several additional haptic variations of our current shared control behavior. Using our autonomous navigation algorithm, we would like to implement different haptic effects that can be tested using our existing study and then compared against each other. Additionally, we would like to collect information about what people liked and dislike about each effect. Example effects include:

Snap-to steering: One of the concerns stated by participants of our study was the difficulty in persistently working against the steering forces in haptic mode during periods of time when the operator needs to override the autonomy. We propose creating a new version of the haptic driving software which temporarily disengages steering forces during periods of time when the operator is making corrective maneuvers. This could be characterized by creating a "popping" effect to alert the operator that they have disengaged steering forces, and a "snapping" effect to signify the re-engagement of steering forces. The question of how to trigger engaging and disengaging steering forces remains an open question.

Snap-to path: Inspired by Rosenberg's virtual fixtures (in particular the "snap-to lines"), we propose a snap-to path effect. Rather then constantly calculating new paths for the robot based on its current position, a single path is defined which the robot will rigidly follow, similar to the way a train follows rails. This effect is similar to our proposed snap-to steering effect, but engaging and disengaging steering forces is triggered by proximity to the path; deviating from the path is very difficult until you pop off the "rails". You can later pop back onto the path or onto a different path.

Multi-Sensory Cues: The effects listed above are inherently modal, that is, the system has two states - one with steering forces enabled and one with steering forces disabled. Because the control of these modes rely on abstract notions, such as the path described in our proposed "snap-to path" example, it may be difficult for an operator to mentally visualize why the system is behaving in a particular manner. By adding artificial visual indicators (e.g. a line to indicate a path) in our GUI to compliment the haptic feedback, operators may be able to better understand the behavior of the system.

7.3.2 Experiment Variations

While running our study, we conceived of several different research questions and possible experiments which could be used to investigate them. The following are some examples of questions and possible experiments we would like to perform in the future.

Can people tell the difference between known haptic styles? We would like to investigate whether or not people can differentiate between different styles of haptic effects, especially between behavioral and environmental feedback. We propose an experiment in which a participant is asked to identify which mode the system is in, either environmental feedback or behavioral feedback. Before the experiment, a detailed description of each mode should be given to the participant, each followed by a training run in to allow participants to familiarize themselves which each mode. Each participant would then be asked to complete six runs using "randomly selected" modes. At the completion of each run, they will be asked to identify which of the two modes was being used during that run.

Can people detect different unknown haptic styles? As an extension to our proposed experiment to see if people can tell the difference between two modes they have been introduced to, we would like to see if people can differentiate between haptic modes they have not been introduced to. We could accomplish this by describing to participants how different haptic behaviors could be created, without giving specific examples, before the beginning of the experiment. The participants would then be asked to perform our driving task using several different haptic effects, keeping track of how many different effects they believed were tested. At the conclusion of the experiment the participant would be asked how many different effects were used during the experiment.

Does preconception plays into how well a haptic mode is received? We would like to perform an experiment in which we look at how an operators preconceptions about how haptic feedback should work effects their satisfaction with the interaction. We propose investigating this by explaining two different haptic modes to a participant, but not letting them test them. One of the descriptions should exactly match the corresponding haptic mode, while the other description should approximately fit the resulting outcome, but not by accomplishing it in the way described. For example, a path following effect could be described correctly, but the second effect could be described as a behavioral "centering behavior" when in fact it is a deflecting force field environmental behavior. Then, before each run, tell the participant which mode is being used, and after each run ask him or her to rate their current opinion of the mode.

How do people interpret haptic information? What do people believe haptic information is trying to tell them? We propose an experiment in which participants are asked to describe what is going on. Before the experiment, participants would be told there are two haptic modes, but they are given generic identities such as A and B, and not described. During each run, participants are told which mode they are using. At the end of the experiment, participants will be asked to describe what the two modes were and how they worked.

7.3.3 Additional Haptic Modes

Finally we would like to perform head to head comparisons of behaviors described in other research. We would then like to use these behaviors as a baseline against which to compare some behaviors listed below.

Non-absolute control forcefield: This effect would consist of a forcefield style behavior. As the robot nears obstacles, its velocity in the direction towards that obstacle is scaled down until eventually the robot is no longer moving, while movement in directions other then the obstacle are not impacted. The operator feels haptic force as a measure of the difference between what he or she is commanding the robot to do and what the robot is actually doing.

Momentum: This effect models the conservation of momentum, in which the joystick is used to "pull" the robot into motion. A stopped robot would make the joystick very difficult to push away from center. Once moving the robot would want to stay at that velocity, and consequently the joystick would also want to stay in that position. Slowing the robot down or redirecting its movement would exhibit similar behavior.

Discrete directions: This mode is a regular joystick for teleoperation, but "grooves" are dynamically added to the joystick in positions that allow the robot to travel in smooth arcs. These grooves would normally reside at specific positions such as forward, 30 degrees of center, and 60 degrees off center when no immediate obstacles are present, but would slide out of their normal positions into new positions to allow the operator to miss obstacles. This adaptation is an interesting behavior because it has the potential to deliver a very useful assisted steering behavior in non-structured environments that cannot be planned out with wayposes.

As always, research answers far fewer questions then it creates, and perhaps the single largest question that has yet to be adequately answered is what role should force feedback/haptic technology play in shared control systems? Admittedly, this question is rather far-reaching and really encompasses a set of more specific questions aimed at the different aspects of remotely controlling mobile ground robots.

- What information should haptic forces represent?
- How should the haptic force representation relate to the actual state of the remote environment and/or robot?
- How should the haptic forces be implemented, and what are the characteristic differences between implementations?
- When are haptic effects appropriate, and when are they not?
- Are there haptic effects that make sense for all levels of autonomy?

It may be that the answers to these questions are context dependent, with the answers depending on the exact nature of the task being performed. Regardless, further investigation is necessary to understand the answers to these questions.

Bibliography

R. J. Adams and B. Hannaford. Stable haptic interaction with virtual environments. *Robotics and Automation, IEEE Transactions on*, 15(3):465–474, 1999. 2.2

R. J. Anderson and M. W. Spong. Bilateral control of teleoperators with time delay. *Automatic Control, IEEE Transactions on*, 34(5):494–501, 1989. 2.2

B. Bae, T. Koo, and K. Park. Design and control of a two degree of freedom haptic device for the application of PC video games. *Intelligent Robots and Systems*, 2001. 3.2

D. Barnes and M. Counsell. Haptic communication for mobile robot operations. *Industrial Robot: An International Journal*, 30(6):552–563, 2003. 2, 2.3, 2.3

D. A. Bowman, S. Coquillart, B. Froehlich, M. Hirose, Y. Kitamura, K. Kiyokawa, and Stuerzlinger. 3D User Interfaces: New Directions and Perspectives. *Computer Graphics and Applications, IEEE*, 28(6), 2008. 3.2

D. J. Brooks and H. A. Yanco. Design of a haptic joystick for shared robot control. In *Proceedings of the 7th Annual ACM/IEEE International Conference on Human-Robot Interaction*, Mar. 2012. 3.4

D. J. Brooks, M. Lunderville, and H. A. Yanco. Design of a 2D joystick for robot control based on a 6 DOF haptic device. In *Proceedings of the IEEE International Conference on Technologies for Practical Robot Applications (TePRA)*, Apr. 2013. 4.4, 6.3.7

D. J. Bruemmer, D. D. Dudenhoeffer, and J. Marble. Dynamic autonomy for urban search and rescue. In *Proceedings of the AAAI Mobile Robot Workshop*, Edmonton, Canada, 2002. 1

T. Carlson and Y. Demiris. Using visual attention to evaluate collaborative control architectures for human robot interaction. In *New Frontiers in Human Robot Interaction, a symposium at AISB 2009*, pages 38–43, 2009. 1

T. Carlson and Y. Demiris. Increasing robotic wheelchair safety with collaborative control: Evidence from secondary task experiments. In *2010 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5582–5587. IEEE, 2010. 1

M. C. Çavusoglu, A. Sherman, and F. Tendick. Design of bilateral teleoperation controllers for haptic exploration and telemanipulation of soft environments. *Robotics and Automation, IEEE Transactions on*, 18(4):641–647, 2002. 2.2

CH Products. M Series Miniature Resistive Joysticks, 2011. `www.chproducts.com/files/chproducts/brochures/M_CH_low-res.pdf`, accessed Jan. 14, 2013. 3.5.1

cheapgamehardware.blogspot.com, 2013. `cheapgamehardware.blogspot.com`, accessed May 23, 2013. 1

S. K. Cho, H. Z. Jin, J. M. Lee, and B. Yao. Teleoperation of a mobile robot using a force-reflection joystick with sensing mechanism of rotating magnetic field. *Mechatronics, IEEE/ASME Transactions on*, 15(1):17–26, 2010. 2, 2.1, 3.2

M. Desai. *Modeling Trust to Improve Human-Robot Interaction*. PhD thesis, University of Massachusetts Lowell, December 2012. 1, 5, 5.2, 5.3.1, 6.5, 6.6, 3

M. Desai, M. Medvedev, M. Vázquez, S. McSheehy, S. Gadea-Omelchenko, C. Bruggeman, A. Steinfeld, and H. Yanco. Effects of changing reliability on trust of robot systems. In *International Conference on Human-Robot Interaction*. ACM/IEEE, 2012. 5

M. Desai, P. Kaniarasu, M. Medvedev, A. Steinfeld, and H. Yanco. Impact of robot failures and feedback on real-time trust. In *International Conference on Human-Robot Interaction*. ACM/IEEE, 2013. 5

Digikey, 2011a. `http://media.digikey.com/photos/APEM%20Comp%20Photos/M11L061P.jpg`, accessed Jan. 14, 2011. 3.5.1, 9

Digikey, 2011b. `http://media.digikey.com/PDF/Data%20Sheets/APEM%20Components%20PDFs/M11L061P.pdf`, accessed Jan. 14, 2011. 3.5.1, 8

N. Diolaiti and C. Melchiorri. Haptic tele-operation of a mobile robot. In *Robot control 2003: 7th IFAC Symposium (SYROCO'03)*, Wroclaw, Poland, Sept. 2003. 3.2

N. Diolaiti and C. Melchiorri. Haptic tele-operation of a mobile robot. In *Robot Control 2003 (SYROCO'03): A Proceedings Volume from the 7th IFAC Symposium, Wrocław, Poland, 1-3 September 2003*, volume 1, page 7. Pergamon, 2004. 2.3

I. Elhajj, N. Xi, and Y.-H. Liu. Real-time control of Internet based teleoperation with force reflection. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pages 3284–3289, 2000. 2

M. R. Endsley. Situation awareness global assessment technique (SAGAT). *IEEE Aerospace and Electronics Conference (NAECON)*, pages 789–795, 1988. 5.1.3, 6.1, 6.8.1

M. R. Endsley. Automation and situation awareness. *Automation and Human Performance: Theory and Applications*, pages 163–181, 1996. 1

ETS. Kit of factor-referenced cognititve tests, 1976. 5.1.1, 6.8.2, 6.9.2

I. Farkhatdinov and J. Ryu. Improving Mobile Robot Bilateral Teleoperation by Introducing Variable Force Feedback Gain. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5812–5817, 2010a. 3.2

I. Farkhatdinov and J. Ryu. Improving mobile robot bilateral teleoperation by introducing variable force feedback gain. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 5812–5817. IEEE, 2010b. 2, 2.1

I. Farkhatdinov, J.-H. Ryu, and J. Poduraev. Control strategies and feedback information in mobile robot teleoperation. In *17th IFAC World Congress, Seoul, Korea*, pages 14681–14686, 2008. 2.2

I. Farkhatdinov, J.-H. Ryu, and J. Poduraev. A user study of command strategies for mobile robot teleoperation. *Intelligent Service Robotics*, 2(2):95–104, 2009. 2.1

Georiga Tech, 2013. `www.ros.org/wiki/phantom_omni`, accessed May 23, 2013. 4.2

S. Han and J. Lee. Tele-operation of a mobile robot using a force reflection joystick with a single hall sensor. In *The 16th IEEE International Symposium on Robot and Human interactive Communication (RO-MAN)*, 2007. 2

B. Hannaford and J.-H. Ryu. Time-domain passivity control of haptic interfaces. *IEEE Transactions on Robotics and Automation*, 18(1):1–10, Feb. 2002. 2

S. G. Hart. NASA-task load index (NASA-TLX): 20 years later. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 50, pages 904–908. Sage Publications, 2006. 5.1.2, 6.24

S. G. Hart and L. E. Staveland. Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. *Human Mental Workload*, 1(3):139–183, 1988. 5.1.2, 6.8.2

V. Hayward, O. R. Astley, M. Cruz-Hernandez, D. Grant, and G. Robles-De-La-Torre. Haptic interfaces and devices. *Sensor Review*, 24(1):16–29, 2004. 2.1, 3.2

S.-G. Hong, J.-J. Lee, and S. Kim. Generating artificial force for feedback control of teleoperated mobile robots. In *Intelligent Robots and Systems, 1999. IROS'99. Proceedings. 1999 IEEE/RSJ International Conference on*, volume 3, pages 1721–1726. IEEE, 1999. 2, 2.3

D. Kaber and Endsley. Out of the loop performance problems and the use of intermediate levels of automation for improved control system functioning and safety. *Process Safety Progress*, 16, 1997. 1, 5.3.1

Y. Kanayama and B. I. Hartman. Smooth local path planning for autonomous vehicles. In *Robotics and Automation, 1989. Proceedings., 1989 IEEE International Conference on*, pages 1265–1270, 1989. 6.3.2

T. Kanno and Y. Yokokohji. Multilateral teleoperation control over time-delayed computer networks using wave variables. In *Haptics Symposium (HAPTICS), 2012 IEEE*, pages 125–131. IEEE, 2012. 2.2

B. Keyes. Evolution of a Telepresence Robot Interface. Master's thesis, University of Massachusetts Lowell, May 2007. 1, 6.5

B. Keyes, M. Micire, J. L. Drury, and H. A. Yanco. Improving human-robot interaction through interface evolution. In D. Chugo, editor, *Human-Robot Interaction*, chapter 14. InTech, 2010. DOI: 10.5772/8140. 1.2, 5.4

C. Kim, S. Yun, K. Park, C. Choi, and S. Kim. Sensing system design and torque analysis of a haptic operated climbing robot. In *Intelligent Robots and Systems, 2004.(IROS 2004).*, volume 2, pages 1845–1848. IEEE, 2004. 2.3

Kyle Machulis. libnifalcon, 2010. `qdot.github.com/libnifalcon`, accessed Jan. 14, 2013. 3.2

A. Lankenau. Avoiding mode confusion in service robots-the bremen autonomous wheelchair as an example. In *Proceedings of the 7th International Conference on Rehabilitation Robotics (ICORR 2001)*, pages 162–167, 2001. 1

J. F. Lapointe, P. Savard, and N. G. Vinson. A comparative study of four input devices for desktop virtual walkthroughs. *Computers in Human Behavior*, 27(6):2186–2191, Nov. 2011. 3.2

C. E. Lathan and M. Tracey. The effects of operator spatial perception and sensory feedback on human-robot teleoperation performance. *Presence: Teleoperators & virtual environments*, 11(4):368–377, 2002. 5.1.1

D. Lee, O. Martinez-Palafox, and M. W. Spong. Bilateral teleoperation of a wheeled mobile robot over delayed communication network. In *Robotics and Automation, 2006. ICRA*, pages 3298–3303. IEEE, 2006. 2.2

S. Lee, G. Sukhatme, G. J. Kim, and C.-M. Park. Haptic teleoperation of a mobile robot: a user study. *Presence: Teleoperators & Virtual Environments*, 14(3):345–365, 2005. 2.2

R. C. Luo, C.-Y. Hu, T. M. Chen, and M.-H. Lin. Force reflective feedback control for intelligent wheelchairs. In *Intelligent Robots and Systems (IROS 1999).*, volume 2, pages 918–923. IEEE, 1999. 2, 2.3

D. G. Mick and S. Fournier. Paradoxes of technology: consumer cognizance, emotions, and coping strategies. *Journal of Consumer Research*, 25(2):123–143, Sept. 1998. 6.8.2

N. C. Mitsou, S. V. Velanas, and C. S. Tzafestas. Visuo-haptic interface for teleoperation of mobile robot exploration tasks. In *The 15th IEEE International Symposium on Robot and Human Interactive Communication*, pages 157–163, Sept. 2006. 2.2, 2.3, 3.2

B. M. Muir. *Operators' Trust in and Use of Automatic Controllers in a Supervisory Process Control Task*. University of Toronto, 1989. 6.8.2

J. Mullins, B. Horan, M. Fielding, and S. Nahavandi. A haptically enabled low-cost reconnaissance platform for law enforcement. In *Safety, Security and Rescue Robotics, 2007. SSRR 2007. IEEE International Workshop on*, pages 1–6. IEEE, 2007. 2.2

B.-H. Nguyen and J.-H. Ryu. Haptic interface for intuitive teleoperation of wheeled and tracked vehicles. *International Journal of Automotive Technology*, 13(6):949–954, 2012. 2.1

G. Niemeyer and J.-J. Slotine. Stable adaptive teleoperation. *Oceanic Engineering, IEEE Journal of*, 16(1):152–162, 1991. 2, 2.2

D. Norman. How might people interact with agents. *Communications of the ACM*, 37 (7):68–71, 1994. 1

Novint Technologies Inc. Novint Falcon, 2013. `www.novint.com/index.php/novintfalcon`, accessed Jan. 14, 2013. 3.2

A. M. Okamura. Methods for haptic feedback in teleoperated robot-assisted surgery. *Industrial Robot: An International Journal*, 31(6):499–508, 2004. 2, 2.3

A. Parasuraman. Technology Readiness Index (TRI): A multiple-item scale to measure readiness to embrace new technologies. *Journal of Service Research*, 2(4):307–320, May 2000. 6.8.2

S. Parikh, V. Grassi Jr, V. Kumar, and J. Okamoto Jr. Usability study of a control framework for an intelligent wheelchair. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 4745–4750. IEEE, 2005. 1

perso.limsi.fr, 2013. `http://perso.limsi.fr/simard/Images/PhantomOmni.png`, accessed May 23, 2013. 5

I. G. Polushin, P. X. Liu, and C.-H. Lung. A force-reflection algorithm for improved transparency in bilateral teleoperation with communication delay. *Mechatronics, IEEE/ASME Transactions on*, 12(3):361–374, 2007. 2.2

M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng. ROS: an open-source Robot Operating System. *ICRA Workshop on Open Source Software*, 2009. `http://www.ros.org`. 3.2.2

O. J. Rösch, K. Schilling, and H. Roth. Haptic interfaces for the remote control of mobile robots. *Control Engineering Practice*, 10(11):1309–1313, 2002. 2.3

L. B. Rosenberg. Virtual haptic overlays enhance performance in telepresence tasks. In *Photonics for Industrial Applications*, pages 99–108. International Society for Optics and Photonics, 1995. 2.3

F. Schill, R. Mahony, P. Corke, and L. Cole. Virtual force feedback teleoperation of the InsectBot using optical flow. In J. Kim and R. Mahony, editors, *Australasian Conference on Robotics and Automation*, 2008a. 2.3, 3.2

F. Schill, R. Mahony, P. Corke, and L. Cole. Virtual force feedback teleoperation of the insectbot using optical flow. In *Proceedings of the 2008 Australasian Conference on Robotics & Automation*. Australian Robotics and Automation Association Inc., 2008b. 2.2

SensAble Technologies Inc. OpenHaptics Toolkit, 2011. `http://www.sensable.com/products-openhaptics-toolkit.htm`, accessed Nov. 30, 2011. 3.2, 3.3, 4.1

SensAble Technologies Inc. PHANToM Omni, 2011. `http://www.sensable.com/haptic-phantom-omni.htm`, accessed Nov. 30, 2011. 3.2

D. Sharek. A useable, online NASA-TLX tool. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 55, pages 1375–1379. SAGE Publications, 2011. 5.1.2

D. Sharek. NASA-TLX | playgraph, 2013. http://www.playgraph.com/nasatlx, accessed May 22, 2013. 5.1.2

T. B. Sheridan. Human and Computer Control of Undersea Teleoperators. July 1978. 1, 1.2

T. B. Sheridan. *Telerobotics, automation, and human supervisory control*. MIT Press, 1992. 1

A. Steinfeld. Slightly subversive methods for promoting use of autonomy in robots. In *RSS 2011 Workshop On Human-Robot Interaction: Perspectives And Contributions To Robotics From The Human Sciences*, 2011. 1

A. Takemoto, T. Miyoshi, K. Terashima, and K. Yano. Operation assist control system of rotary crane using proposed haptic joystick as man-machine interface. In *Robot and Human Interactive Communication (ROMAN), 2004.*, pages 533–538. IEEE, 2004. 2, 2.3

P. Viswanathan, J. Boger, J. Hoey, P. Elinas, and A. Mihailidis. The future of wheelchairs: Intelligent collision avoidance and navigation assistance. *Geriatrics and Aging*, 10(4):253–256, 2007. 1

M. Voshell, D. D. Woods, and F. Phillips. Overcoming the keyhole in human-robot coordination: Simulation and evaluation. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 49, pages 442–446. SAGE Publications, 2005. 1.2

www.amazon.com, 2013. http://ecx.images-amazon.com/images/I/111X82HDJSL._SL500_SL160_.jpg, accessed May 23, 2013. 7

www.digikey.com, 2013. http://www.digikey.com/Web%20Export/Supplier%20Content/Grayhill_136/ph/grayhill_04J.jpg, accessed May 23, 2013. 3

www.directindustry.com, 2013. http://img.directindustry.com/images_di/photo-g/contactless-inductive-joystick-56536-2595073.jpg, accessed May 23, 2013. 2

www.slipperybrick.com, 2013. http://www.slipperybrick.com/wp-content/uploads/2008/02/novint-falcon-in-black.jpg, accessed May 23, 2013. 6

www.trinity3d.com, 2013. http://www.trinity3d.com/images/3Dconnexion/spacenavigator.jpg, accessed May 23, 2013. 4

H. A. Yanco, B. Keyes, J. L. Drury, C. W. Nielsen, D. A. Few, and D. J. Bruemmer. Evolving interface design for robot search tasks. *Journal of Field Robotics*, 24:779–799, August/September 2007. 1

K. Yano, A. Takemoto, and K. Terashima. Semi-automatic obstacle avoidance control for operation support system with haptic joystick. In *Intelligent Robots and Systems, (IROS 2005).*, pages 1229–1234. IEEE, 2005. 2.3

Appendix A

Hardware

A.1 Haptic Website Instructions

A.1.1 Joystick Arm

The joystick arm is constructed from a combination of hardware and plastic adapters. The hardware can be purchased from McMaster-Carr. These parts are then held together by ABS plastic adapters (shown in black) and small metal pins. The adapters are made by 3D printing, and can be ordered online from sites such as PrintTo3D using the STL files provided on our website. The titanium parts can be cut to length using a hacksaw.

Description	Part #	Notes
3/16" Titanium Rod	#89055K321	Cut to 4.75" long
1/2" Titanium Tube	#89835K74	Cut to 3.25" long
1/2" PTFE Ball Joint	#6960T11	

Table A.1: Haptic Joystick Parts List, McMaster-Carr

The STL files should be printed as solids (not sparse objects) and at no less then 1mm precision. These parts may require a small amount of sanding to get them to fit properly. The pivot adapter and haptic adapter each have small holes on the sides designed for small pins. These pins, the size of a small metal paperclip, are needed to hold the adapters in place along the titanium rod. This is accomplished by having the pins pass through grooves cut on the edge of the titanium rod. For additional information concerning the joystick arm assembly, please see subsection 3.5.2.

A.1.2 Suspension Mount Hardware

The suspension mount is used to hold the haptic device in place below the joystick. It is constructed from 3/4" aluminum angle and plastic plates, shown in Figure 3.13. The

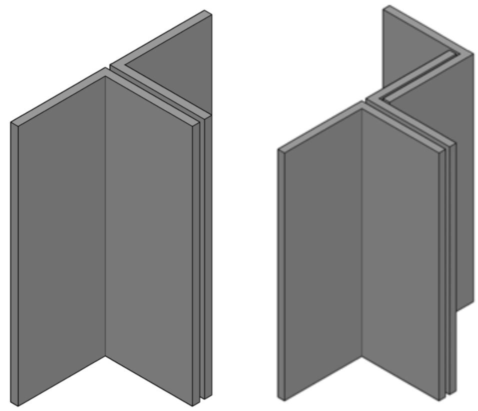


Figure A.1: Aluminum Supports. *Left:* T-Rails, *Right:* Uprights

aluminum angle is used to create T shaped rails as shown in Figure A.1. The t-rail pattern is used for side rails and cross supports, while the upright pattern is used in the corners. The plastic components are made from laser cut PETG or Delrin plastic, and can be ordered from `ponoko.com` using the files available on our website. The plates are designed to be 5mm think. You need two copies of each plate. The mount is held together using 3/16" rivets and washers. For additional information and pictures concerning the suspension mount assembly, please see section 3.6.

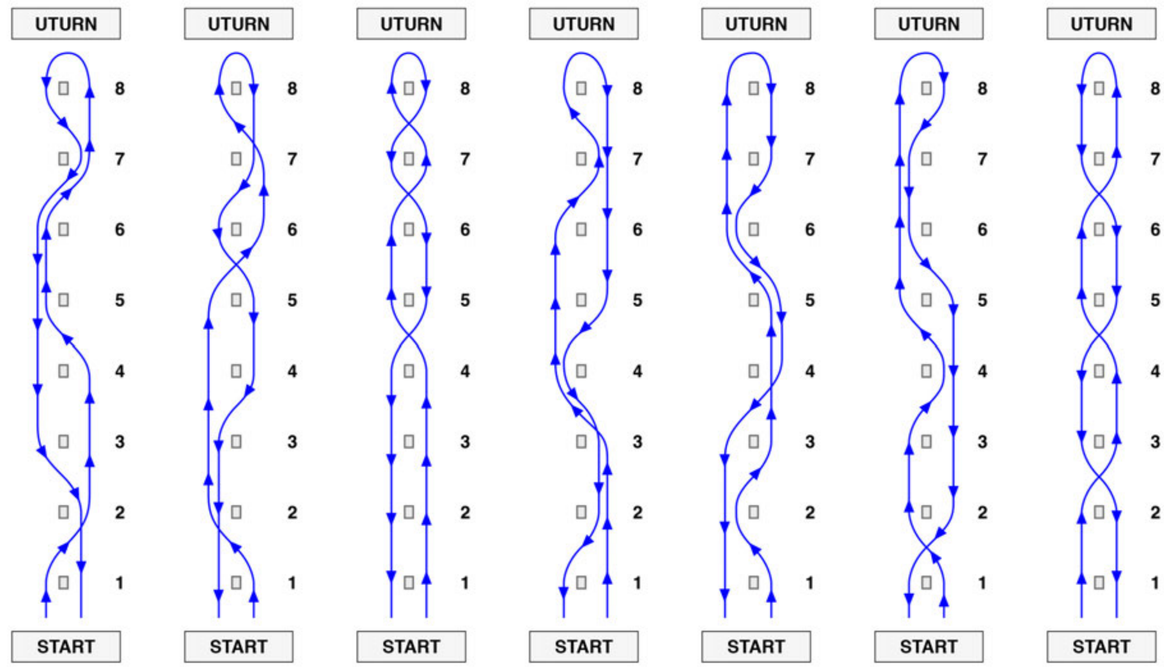
Description	Length in mm	Quantity
Front Uprights	337	6
Back Uprights	407	6
Side Rails	298	4
Cross supports	268	12

Table A.2: Aluminum Angle Parts Length

Appendix B

Experiment Materials

B.1 Maps



(a) Map 0 (b) Map 1 (c) Map 2 (d) Map 3 (e) Map 4 (f) Map 5 (g) Trial

B.2 Participant Conditions

A total of 18 participants were needed for this experiment. Participants 1-9 were used to pilot the study. Due to a number of runs being invalidated due to issues with the network, additional participants were needed to obtain a complete set of 18 runs. Participant conditions repeat starting with P28. These condition spreadsheets may be useful as-is for planning future experiments, but are included here mainly for future reference.

P10	Date			
Run #	**Mode**	**Map**	**Rel**	**Code**
T1	Haptic	T	T	P10-HT1-TT
T2	Non	T	T	P10-NT2-TT
0	Haptic	0	A	P10-H0-0A
1	Non	4	B	P10-N1-4B
2	Haptic	5	C	P10-H2-5C
3	Non	1	A	P10-N3-1A
4	Haptic	2	B	P10-H4-2B
5	Non	3	C	P10-N5-3C

P11	Date			
Run #	**Mode**	**Map**	**Rel**	**Code**
T1	Non	T	T	P11-NT1-TT
T2	Haptic	T	T	P11-HT2-TT
0	Non	3	B	P11-N0-3B
1	Haptic	0	A	P11-H1-0A
2	Non	4	C	P11-N2-4C
3	Haptic	5	B	P11-H3-5B
4	Non	1	A	P11-N4-1A
5	Haptic	2	C	P11-H5-2C

P12				
Run #				
T1	Haptic	T	T	P12-HT1-TT
T2	Non	T	T	P12-NT2-TT
0	Haptic	2	C	P12-H0-2C
1	Non	3	A	P12-N1-3A
2	Haptic	0	B	P12-H2-0B
3	Non	4	C	P12-N3-4C
4	Haptic	5	A	P12-H4-5A
5	Non	1	B	P12-N5-1B

P13	Date			
Run #	**Mode**	**Map**	**Rel**	**Code**
T1	Non	T	T	P13-NT1-TT
T2	Haptic	T	T	P13-HT2-TT
0	Non	1	A	P13-N0-1A
1	Haptic	2	B	P13-H1-2B
2	Non	3	C	P13-N2-3C
3	Haptic	0	A	P13-H3-0A
4	Non	4	B	P13-N4-4B
5	Haptic	5	C	P13-H5-5C

P14	Date			
Run #	**Mode**	**Map**	**Rel**	**Code**
T1	Haptic	T	T	P14-HT1-TT
T2	Non	T	T	P14-NT2-TT
0	Haptic	5	B	P14-H0-5B
1	Non	1	A	P14-N1-1A
2	Haptic	2	C	P14-H2-2C
3	Non	3	B	P14-N3-3B
4	Haptic	0	A	P14-H4-0A
5	Non	4	C	P14-N5-4C

P15	Date			
Run #	**Mode**	**Map**	**Rel**	**Code**
T1	Non	T	T	P15-NT1-TT
T2	Haptic	T	T	P15-HT2-TT
0	Non	4	C	P15-N0-4C
1	Haptic	5	A	P15-H1-5A
2	Non	1	B	P15-N2-1B
3	Haptic	2	C	P15-H3-2C
4	Non	3	A	P15-N4-3A
5	Haptic	0	B	P15-H5-0B

P16	Date			
Run #	**Mode**	**Map**	**Rel**	**Code**
T1	Haptic	T	T	P16-HT1-TT
T2	Non	T	T	P16-NT2-TT
0	Haptic	0	A	P16-H0-0A
1	Non	4	B	P16-N1-4B
2	Haptic	5	C	P16-H2-5C
3	Non	1	A	P16-N3-1A
4	Haptic	2	B	P16-H4-2B
5	Non	3	C	P16-N5-3C

P17	Date			
Run #	**Mode**	**Map**	**Rel**	**Code**
T1	Non	T	T	P17-NT1-TT
T2	Haptic	T	T	P17-HT2-TT
0	Non	3	B	P17-N0-3B
1	Haptic	0	A	P17-H1-0A
2	Non	4	C	P17-N2-4C
3	Haptic	5	B	P17-H3-5B
4	Non	1	A	P17-N4-1A
5	Haptic	2	C	P17-H5-2C

P18	Date			
Run #	**Mode**	**Map**	**Rel**	**Code**
T1	Haptic	T	T	P18-HT1-TT
T2	Non	T	T	P18-NT2-TT
0	Haptic	2	C	P18-H0-2C
1	Non	3	A	P18-N1-3A
2	Haptic	0	B	P18-H2-0B
3	Non	4	C	P18-N3-4C
4	Haptic	5	A	P18-H4-5A
5	Non	1	B	P18-N5-1B

P19	Date			
Run #	**Mode**	**Map**	**Rel**	**Code**
T1	Non	T	T	P19-NT1-TT
T2	Haptic	T	T	P19-HT2-TT
0	Non	1	A	P19-N0-1A
1	Haptic	2	B	P19-H1-2B
2	Non	3	C	P19-N2-3C
3	Haptic	0	A	P19-H3-0A
4	Non	4	B	P19-N4-4B
5	Haptic	5	C	P19-H5-5C

P20	Date			
Run #	**Mode**	**Map**	**Rel**	**Code**
T1	Haptic	T	T	P20-HT1-TT
T2	Non	T	T	P20-NT2-TT
0	Haptic	5	B	P20-H0-5B
1	Non	1	A	P20-N1-1A
2	Haptic	2	C	P20-H2-2C
3	Non	3	B	P20-N3-3B
4	Haptic	0	A	P20-H4-0A
5	Non	4	C	P20-N5-4C

P21	Date			
Run #	**Mode**	**Map**	**Rel**	**Code**
T1	Non	T	T	P21-NT1-TT
T2	Haptic	T	T	P21-HT2-TT
0	Non	4	C	P21-N0-4C
1	Haptic	5	A	P21-H1-5A
2	Non	1	B	P21-N2-1B
3	Haptic	2	C	P21-H3-2C
4	Non	3	A	P21-N4-3A
5	Haptic	0	B	P21-H5-0B

P22	Date			
Run #	**Mode**	**Map**	**Rel**	**Code**
T1	Haptic	T	T	P22-HT1-TT
T2	Non	T	T	P22-NT2-TT
0	Haptic	0	A	P22-H0-0A
1	Non	4	B	P22-N1-4B
2	Haptic	5	C	P22-H2-5C
3	Non	1	A	P22-N3-1A
4	Haptic	2	B	P22-H4-2B
5	Non	3	C	P22-N5-3C

P23	Date			
Run #	**Mode**	**Map**	**Rel**	**Code**
T1	Non	T	T	P23-NT1-TT
T2	Haptic	T	T	P23-HT2-TT
0	Non	3	B	P23-N0-3B
1	Haptic	0	A	P23-H1-0A
2	Non	4	C	P23-N2-4C
3	Haptic	5	B	P23-H3-5B
4	Non	1	A	P23-N4-1A
5	Haptic	2	C	P23-H5-2C

P24	Date			
Run #	**Mode**	**Map**	**Rel**	**Code**
T1	Haptic	T	T	P24-HT1-TT
T2	Non	T	T	P24-NT2-TT
0	Haptic	2	C	P24-H0-2C
1	Non	3	A	P24-N1-3A
2	Haptic	0	B	P24-H2-0B
3	Non	4	C	P24-N3-4C
4	Haptic	5	A	P24-H4-5A
5	Non	1	B	P24-N5-1B

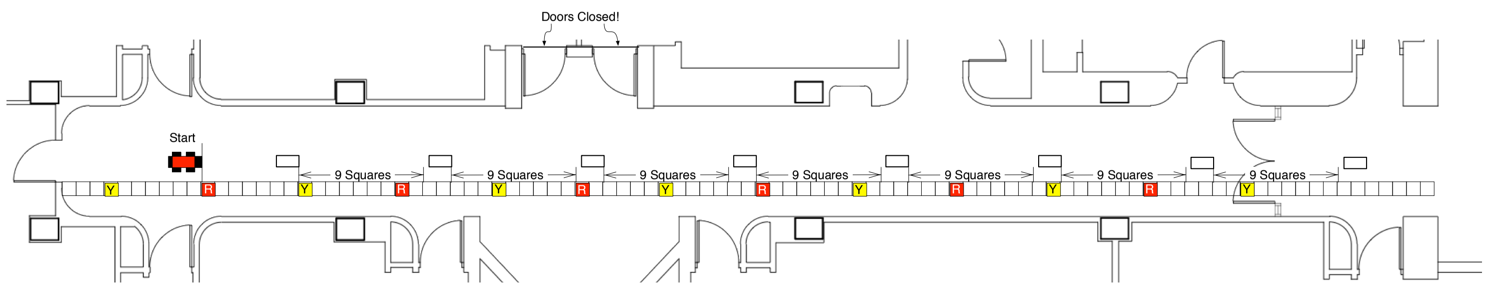
P25	Date			
Run #	**Mode**	**Map**	**Rel**	**Code**
T1	Non	T	T	P25-NT1-TT
T2	Haptic	T	T	P25-HT2-TT
0	Non	1	A	P25-N0-1A
1	Haptic	2	B	P25-H1-2B
2	Non	3	C	P25-N2-3C
3	Haptic	0	A	P25-H3-0A
4	Non	4	B	P25-N4-4B
5	Haptic	5	C	P25-H5-5C

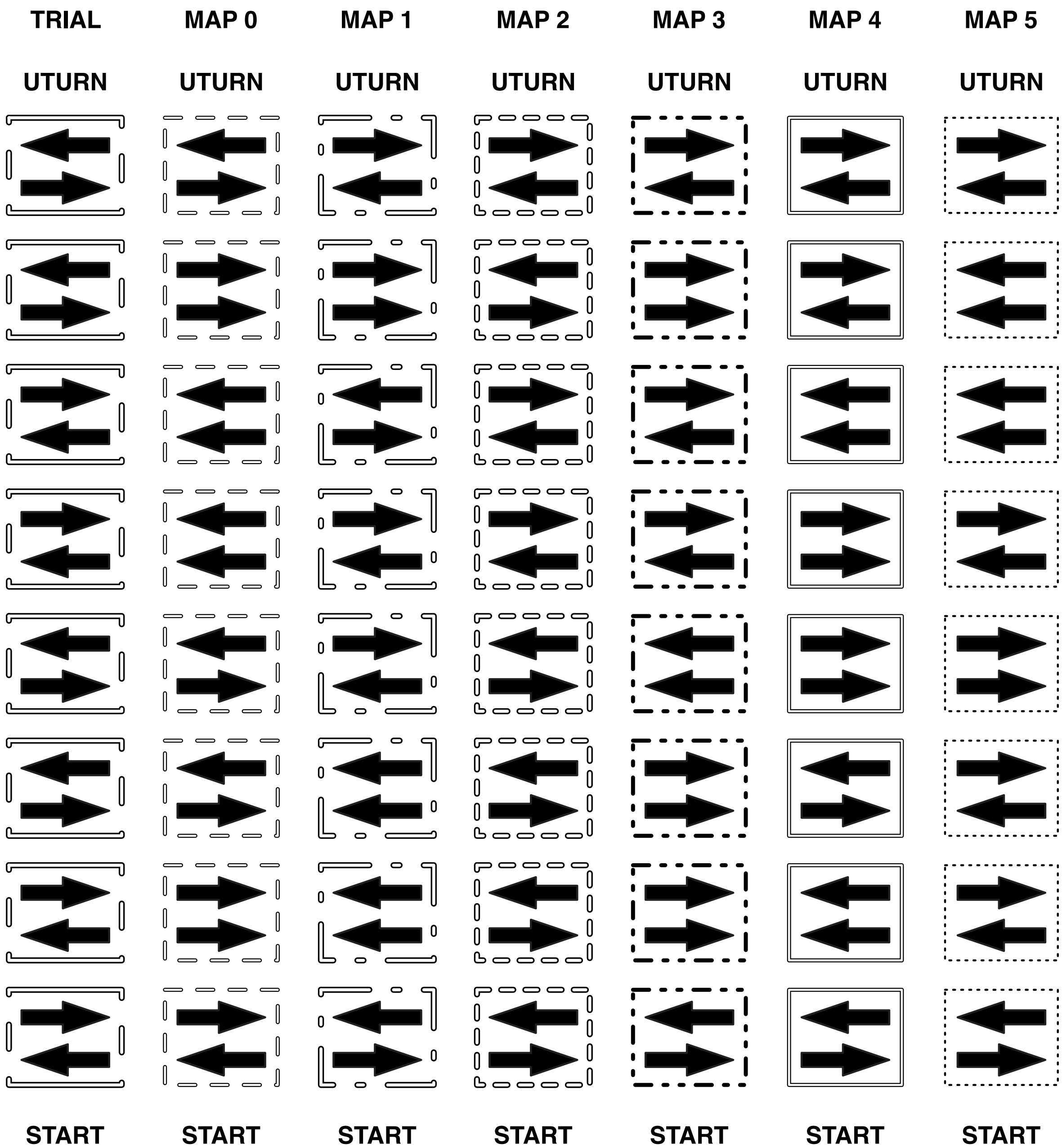
P26	Date			
Run #	**Mode**	**Map**	**Rel**	**Code**
T1	Haptic	T	T	P26-HT1-TT
T2	Non	T	T	P26-NT2-TT
0	Haptic	5	B	P26-H0-5B
1	Non	1	A	P26-N1-1A
2	Haptic	2	C	P26-H2-2C
3	Non	3	B	P26-N3-3B
4	Haptic	0	A	P26-H4-0A
5	Non	4	C	P26-N5-4C

P27	Date			
Run #	**Mode**	**Map**	**Rel**	**Code**
T1	Non	T	T	P27-NT1-TT
T2	Haptic	T	T	P27-HT2-TT
0	Non	4	C	P27-N0-4C
1	Haptic	5	A	P27-H1-5A
2	Non	1	B	P27-N2-1B
3	Haptic	2	C	P27-H3-2C
4	Non	3	A	P27-N4-3A
5	Haptic	0	B	P27-H5-0B

B.3 Course Layout

The following page shows the instructions for setting up the physical layout of the course, which was provided to the robot handler. The course was setup on the 3rd floor of Olsen Hall at the University of Massachusetts Lowell. The robot's starting location is in front of room 302. Box layout is shown using colored floor tiles. The next page has a table showing directions that the arrows should be placed on either side of each box, depending on which map is being used.


Doors Closed!
Start
9 Squares
9 Squares
9 Squares
9 Squares
9 Squares
9 Squares
9 Squares


TRIAL
MAP 0
MAP 1
MAP 2
MAP 3
MAP 4
MAP 5
UTURN
UTURN
UTURN
UTURN
UTURN
UTURN
UTURN
START
START
START
START
START
START
START

B.4 Survey Questions

Question	Type	Response
1) Age	Multiple Choice	18 to 80
2) Gender	Multiple Choice	(1) Male, (2) Female, or (3) Prefer Not to Answer
3) Occupation	Free Response	
4) Computer usage per week	Multiple Choice	(1) $<$ 10 Hours, (2) 11 - 20 Hours, (3) 21 - 30 Hours, (4) 31 - 40 Hours, (5) $>$ 40 Hours
5) Which is your dominant hand?	Multiple Choice	(1) Right, (2) Left, (3) Ambidextrous
6) Is English your primary language?	Multiple Choice	(1) Yes, (2) No
7) Please provide us with your level of experience in the following areas.		
7a) I am experienced with robots	Likert	7 point scale from "Strongly Disagree" to "Strongly Agree".
7b) I am experienced with radio controlled vehicles	Likert	7 point scale from "Strongly Disagree" to "Strongly Agree".
7c) I am experienced with first-person perspective video games	Likert	7 point scale from "Strongly Disagree" to "Strongly Agree".
7d) I am experienced with real time strategy games	Likert	7 point scale from "Strongly Disagree" to "Strongly Agree".
7e) I am experienced with PlayStation / Xbox controllers	Likert	7 point scale from "Strongly Disagree" to "Strongly Agree".
8) Have you seen robots in person before?	Multiple Choice	(1) Yes (2) No
8b) If you answered 'yes' to (8), please explain:	Free Response	
9) On average how many hours in a week do you spend playing video games?	Multiple Choice	(1) "$<$ 10 Hours", (2) "11 - 20 Hours", (3) "21 - 30 Hours", (4) "31 - 40 Hours", (5)"$>$ 40 Hours"
10) Please indicate your level of agreement with each statement regarding risk-taking activity.		
10a) I like to test myself every now and then by doing something a little risky	Likert	6 point scale from "Strongly Disagree" to "Strongly Agree"
10b) Sometimes I will take a risk just for the fun of it.	Likert	6 point scale from "Strongly Disagree" to "Strongly Agree"
10c) I sometimes find it exciting to do things for which I might get into trouble	Likert	6 point scale from "Strongly Disagree" to "Strongly Agree"
10d) Excitement and adventure are more important to me than security	Likert	6 point scale from "Strongly Disagree" to "Strongly Agree"

Table B.1: Pre Experiment Survey Questions (Table 1 of 2)

Question	**Type**	**Response**
11) The following questions are about your attitudes and views towards technology in general. In general, to what extent do you believe that technology…		
11a) Makes life easy and convenient	Likert	7 point scale from "Strongly Disagree" to "Strongly Agree"
11b) Makes life complicated	Likert	7 point scale from "Strongly Disagree" to "Strongly Agree"
11c) Gives people control over their daily lives	Likert	7 point scale from "Strongly Disagree" to "Strongly Agree"
11d) Makes people dependent	Likert	7 point scale from "Strongly Disagree" to "Strongly Agree"
11e) Makes life comfortable	Likert	7 point scale from "Strongly Disagree" to "Strongly Agree"
11f) Makes life stressful	Likert	7 point scale from "Strongly Disagree" to "Strongly Agree"
11g) Brings people together	Likert	7 point scale from "Strongly Disagree" to "Strongly Agree"
11h) Makes people isolated	Likert	7 point scale from "Strongly Disagree" to "Strongly Agree"
11i) Increases personal safety and security	Likert	7 point scale from "Strongly Disagree" to "Strongly Agree"
11j) Reduces Privacy	Likert	7 point scale from "Strongly Disagree" to "Strongly Agree"
12) How well does each of the following phrases regarding technology describe you? Please rate how accurate each is in describing you at the present time		
12a) I like to keep up with the latest technology	Likert	7 point scale from "Strongly Inaccurate" to "Strongly Accurate"
12b) I generally wait to adopt a new technology until all the bugs have been worked out	Semantic Differential	7 point scale from "Strongly Inaccurate" to "Strongly Accurate"
12c) I enjoy the challenge of figuring out high tech gadgets	Likert	7 point scale from "Strongly Inaccurate" to "Strongly Accurate"
12d) I feel confident that I have the ability to learn to use technology	Likert	7 point scale from "Strongly Inaccurate" to "Strongly Accurate"
12e) Technology makes me nervous	Likert	7 point scale from "Strongly Inaccurate" to "Strongly Accurate"
12f) If a human can accomplish a task as well as technology, I prefer to interact with a person	Likert	7 point scale from "Strongly Inaccurate" to "Strongly Accurate"
12g) I like the idea of using technology to reduce my dependence on other people	Likert	7 point scale from "Strongly Inaccurate" to "Strongly Accurate"

Table B.2: Pre Experiment Survey Questions (Table 2 of 2)

Question	Type	Response
1) Please rate how well you feel you performed for the last run	Semantic Differential	7 point scale from Poor to Excellent
2) Please rate the robot's overall performance for the last run	Semantic Differential	7 point scale from Poor to Excellent
3) Which mode would you prefer to use?	Multiple Choice	(1) Force Feedback, (2) Non-Force Feedback, (3) No Preference
4) Please describe all the factors that you think might affect your trust of an autonomous robot.	Free Response	
5) The device I used to control the robot ______ my performance	Semantic Differential	5 point scale from Hindered to Helped
6) Using the device to control the robot was generally ______ .	Semantic Differential	5 point scale from Frustrating to Enjoyable
7) Using this device, I felt that I was in control of the robot.	Semantic Differential	5 point scale from Strongly Disagree to Strongly Agree
8) Generally, I think my performance on the last run ______ .	Semantic Differential	5 point scale from Left Room for Improvement to Was second-to-none
9) To What extent can the system's behavior be predicted from moment to moment?	Semantic Differential	10 point scale from "Not at all" to "Completely"
10) To What extent can you count on the system to do its job?	Semantic Differential	10 point scale from "Not at all" to "Completely"
11) What degree of faith do you have that the system will be able to cope with all system 'states in the future'?	Semantic Differential	10 point scale from "Not at all" to "Completely"
12) Overall how much do you trust the system?	Semantic Differential	10 point scale from "Not at all" to "Completely"
13) How many times did the robot hit objects?	Multiple Choice	0 to 21
14) What percent of the time was the camera aimed straight forward?	Multiple Choice	"0-10", "10-20", "20-30", "30-40", "40-50", "50-60", "60-70", "70-80", "80-90", "90-100"
15) Draw the path that the robot took on the map provided to you	Free Response	drawn on paper
16) How mentally demanding was this task?	Semantic Differential	21 point scale from "Very low" to "Very High"
17) How physically demanding was the task?	Semantic Differential	21 point scale from "Very low" to "Very High"
18) How hurried or rushed was the pace of the task?	Semantic Differential	21 point scale from "Very low" to "Very High"
19) How successful were you in accomplishing what you were asked to do?	Semantic Differential	21 point scale from "Failure" to "Perfect"
20) How hard did you have to work to accomplish your level of performance?	Semantic Differential	21 point scale from "Very low" to "Very High"
21) How insecure, discouraged, irritated, stressed, and annoyed were you?	Semantic Differential	21 point scale from "Very low" to "Very High"

Table B.3: Post Run Survey Questions

Question	**Type**	**Response**
1a) My overall confidence that I could accurately communicate my intentions to the robot with the NON force feedback joystick was ______ .	Semantic Differential	7 point scale from “Low” to “High”
1b) My overall confidence that I could accurately communicate my intentions to the robot with the force feedback joystick was ______ .	Semantic Differential	7 point scale from “Low” to “High”
1c) I felt in control of the robot while using NON Force Feedback	Semantic Differential	7 point scale from “Low” to “High”
1d) I felt in control of the robot while using Force Feedback	Semantic Differential	7 point scale from “Low” to “High”
2a) I would like to operate this robot again.	Likert	7 point scale from “Strongly Disagree” to “Strongly Agree”
2b) The robot was malfunctioning.	Likert	7 point scale from “Strongly Disagree” to “Strongly Agree”
2c) I trust this robot	Likert	7 point scale from “Strongly Disagree” to “Strongly Agree”
2d) I trust robots (in general).	Likert	7 point scale from “Strongly Disagree” to “Strongly Agree”
2e) I will not trust robots as much as I did before.	Likert	7 point scale from “Strongly Disagree” to “Strongly Agree”
3) What are some applications where you think the NON force feedback joystick would be useful? Why?	Free Response	
4) What are some applications where you think the force feedback joystick would be useful? Why?	Free Response	
5) If you had to choose between using either the force feedback mode or the NON force feedback mode to navigate a robot through a slalom course such as this, which would you pick?	Multiple Choice	(1) Force Feedback, (2) Non Force Feedback
6) Please explain your choice for the previous question	Free Response	
7) Do you have any recommendations?	Free Response	

Table B.4: Post Experiment Survey Questions

B.5 Experimenter Instructions

B.5.1 Setup

ssh into robot-lab9 and start a tmux session. Launch the joystick

```
roslaunch haptic_joystick joystick.launch
```

Follow the instructions to calibrate the joystick.

B.5.2 Starting the Run

Paperwork

- Mark down run-id from participant condition table.
- Set the post-run questionnaire for the run

Confirm Setup

- Ask robot handler if the course is setup (if they have not reported already)
- Ask robot handler to unplug the robot

Kill Teleop

Kill Teleop - press control-c.

Start the Backend

inside the tmux session, run

```
$ roslaunch haptic_launch haptics.launch experiment:=true
```

wait to see “JoystickReader: Entering Main Loop” and “Pipeline is live and does not need to PREROLL..” messages.

Start Rviz

Start Rviz on the laptop

```
$ export ROS_MASTER_URI=htt://robot-lab9.lan:11311
$ export ROS_HOSTNAME=(ip address of machine)
$ rosrun rviz rviz
```

Start Logging

- Start logging by pressing the “start logging” button in Rviz.
- Wait to see a long string of topics being recorded show up.

Start UI

- Click “start ui” icon on windows laptop’s desktop
- select the drop down menus for the run
- start fraps

B.5.3 Ending the run

Post Run Survey

Make sure to have the participant fill out the post run survey

Exit the UI

From the black screen, hold down shift and select "End Run". Then press quit.

Stop Fraps

Press F9

Stop Backend Logging

Press the "stop logging" button in Rviz, and close Rviz.

Stop the backend

Start Teleop

```
$ teleop_joy
```

B.6 Instructions Script

Read aloud by experimenter to participant

I'll be reading this script to ensure I don't miss any information. If you have a question, feel free to ask me and I'll do my best to answer them. However, if I can't answer them because they might bias you in some way, then I will answer them once the experiment is over. I'm going to give you a quick overview of the entire process. You have already filled out the pre-experiment questionnaires. Next, I'm going to explain the user interface and task. After that we'll have two practice runs so you can get used to driving the robot. Then there will be six runs each followed by a short questionnaire. Also, feel free to take as many breaks as you'd like. You can stop the experiment whenever you want and there are no penalties for doing so.

This is the user interface you will use to control the robot. The robot has two cameras. The video from the front camera is displayed here [point to the front video feed] and the video from the back camera is displayed here [show the rear video feed]. The rear video stream is mirrored, making it like a rear view mirror in a car. The front video camera can be moved around using the right hand joystick. To move the camera up, push the joystick forward, and to move the camera sideways push the joystick from side to side. As you move the camera, you'll notice on the video feed that the yellow crosshairs and white lines also move. These lines tell you where the camera is pointing. This is important because often people move the cameras to look around and then forget that the camera is off center and keep driving as if the camera is pointed straight. This often causes collisions. Another thing that you probably noticed is the distance display below the main video feed turning as you pan the camera sideways. This is another indicator that tells you which way your camera is pointed. If you press the green button [press the button], it will reset the camera. The distance display shows you what is around the robot and that is helpful while navigating tight spaces. The information displayed is based on accurate sensor data, so if you see a small gap between the robot and the wall, then there is actually a small gap in the real world. Unfortunately, there is a small blind spot between the front and back laser [show that on the screen] so when there are objects close to the side of the robot you will not know their exact location. A blue vector will show up here indicating the direction and speed of where the robot wants to go. A white vector will indicate where you want to go based on your left joystick input - I'll come back to this in just a moment.

Here is a map of the course [gesture to paper map on desk]. These rectangles in the center are boxes around which the robot must drive. The robot starts at this end and will drive all the way to the far end and come back again. Each box has a white arrow on it. The arrow indicates which side of the box the robot should pass on. Each box also has a barcode which the robot can read.

The robot is autonomous: in other words it will drive itself. The robot will avoid hitting things, and can read barcodes on the boxes to determine which side of each box to pass on. Your primary task is to make sure the robot follows the correct path through the course, even when it misreads one of the barcodes. As soon as you notice that the robot is trying to go on the wrong side of a box, press the black button. It will not seem like anything happens when you press this button, but after each run the robot checks to see when this button was pressed so it knows that it had trouble with that barcode during the next run. You only need to press it once for each box the robot makes a mistake with. If the robot does pass on the wrong side of a box, just keep going.

The robot has two modes, both of which are autonomous. In the first mode, the robot will drive autonomously when you press the driving joystick forward [demonstrate]. If you let go of the joystick, the robot will continue to drive forward, but very slowly. If the robot makes a mistake and tries to go on the wrong side of a box, you can correct it by steering with the joystick. In this mode, as long as the joystick is pushed straight forward, the robot will not hit anything. This is not true when you make steering corrections, so you should be careful. The second mode is called force feedback mode. The robot will still drive autonomously, but it must steer itself by moving your joystick using motors below the table. In this mode, the robot will push the joystick forward for you. To stop the robot, you must pull the joystick back to center. As the robot tries to drive itself through the course, it will make many small adjustments to its direction. Since it must drive itself using the joystick, you will be able to feel these small adjustments. For driving in this mode, it is best to loosely grip the joystick about half way up and push forward, allowing the robot to shift the joystick from side to side to steer. You can correct any wrong turns the robot tries to make by pushing more forcefully in the direction you want the robot to turn. Remember, when you notice that the robot is trying to go on the wrong side of a box, press the black button. We will be switching between these two modes each run.

Apart from making sure the robot drives through the boxes correctly, you will need to do a few other tasks. Whenever you see a blue circle appear in the video screen, move the yellow crosshair over it and the circle will disappear. This simulates a professional robot operator's job of visually surveying the environment around the robot. Additionally, you should keep track of which side the robot passed the last box on, which side the robot's autonomy wants to pass the next box on (regardless of it the robot is correct or not), where the robot is on the course, how many times the robot made a mistake reading the boxes, and how many times you think the robot has hit or scraped one of the boxes or walls. Occasionally, we will stop the robot and ask you some of these questions.

For completing the experiment, you will receive 20 dollars in base compensation, with the opportunity to make up to an additional 20 dollars based on your performance. You start out with all 20 extra dollars, and lose money for mistakes. Each time the robot takes a wrong turn, it is minus 5 dollars. Each time the robot hits a box or wall, it is minus 1 dollar. Each time a blue circle disappears from the screen before you have a change to move the yellow cross hair over it, you lose 50 cents. Each time you answer one of our questions incorrectly, you lose 50 cents. However, we average these costs over all 6 runs and then round up to the nearest five dollars.

Do you have any questions?

Please briefly explain to me what you need to do.

We are going to start with the two trial runs. In one, we will use the regular autonomous mode and in the other, we will use the force feed back mode so that you can get used to driving the robot in each mode.

B.6.1 First Time Lights Out (SAGAT Screen) Instructions

Read aloud by experimenter to participant at first instance of questionnaire interruption during first trial run.

Whenever this screen appears the robot automatically stops moving. Please use the mouse to answer the questions on the screen as best you can. If you have trouble understanding what a question is asking, let me know and I will try to explain it better. If you understand the question, but do not know the answer, you may select "I'm not sure", if the option is there.