University of Massachusetts Amherst

ScholarWorks@UMass Amherst

Masters Theses

Dissertations and Theses

July 2016

A Haptic Surface Robot Interface for Large-Format Touchscreen Displays

Mark Price
University of Massachusetts Amherst

Follow this and additional works at: https://scholarworks.umass.edu/masters_theses_2

Part of the Biomedical Devices and Instrumentation Commons, Controls and Control Theory Commons, Electro-Mechanical Systems Commons, Graphics and Human Computer Interfaces Commons, and the Robotics Commons

Recommended Citation

Price, Mark, "A Haptic Surface Robot Interface for Large-Format Touchscreen Displays" (2016). *Masters Theses*. 371.
https://scholarworks.umass.edu/masters_theses_2/371

This Open Access Thesis is brought to you for free and open access by the Dissertations and Theses at ScholarWorks@UMass Amherst. It has been accepted for inclusion in Masters Theses by an authorized administrator of ScholarWorks@UMass Amherst. For more information, please contact scholarworks@library.umass.edu.

A HAPTIC SURFACE ROBOT INTERFACE FOR LARGE-FORMAT TOUCHSCREEN DISPLAYS

A Thesis Presented

by

MARK ANDREW PRICE

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

May 2016

Mechanical and Industrial Engineering

© Copyright by Mark A Price 2016

All Rights Reserved

A HAPTIC SURFACE ROBOT INTERFACE FOR LARGE-FORMAT TOUCHSCREEN DISPLAYS

A Thesis Presented

by

MARK ANDREW PRICE

Approved as to style and content by:

____
Frank C. Sup IV, Chair

____
Ian Grosse, Member

____
Sundar Krishnamurty, Member

____
Sundar Krishnamurty, Department Head
Mechanical and Industrial Engineering

DEDICATION

I am dedicating this thesis to my parents, George and Sheree Price, who have challenged me to learn and excel all my life, and helped me keep my head straight when things went wrong, even the time that I led the pledge of allegiance in a funny voice as a 5th grade student news anchor and was fired by demand of half the school faculty. I am also dedicating it to my twin sister Devon, who is proving her dominance and expertise to the world, but has nothing to prove to me. Finally, I am dedicating this thesis to Jess, who is literally always there for me despite living 3000 miles away.

ACKNOWLEDGMENTS

I would like to thank my advisor, Professor Frank Sup, for his direction and support in building this project up from a basic idea. His advice and instruction, as well as the support of his lab, has helped me to substantially develop my skills creating and working with mechatronic systems. I would also like to thank the other members of my committee, Professor Ian Grosse and Professor Sundar Krishnamurty, for their helpful input since the inception of the project, the inclusion of this work as part of the Center for eDesign, and for challenging me to always consider the scholarly contribution and broader impact of my work.

I would also like to thank Michael White for his partnership, ideas, and resources supplied through FTL Labs, all of which made this project possible. I would like to thank David Thomas at FTL, whose help was instrumental in integrating my device with the Playsurface.

Finally, I would like to thank all of the students in the Center for eDesign and at MRRL for all of their support, assistance, and lessons learned.

ABSTRACT

A HAPTIC SURFACE ROBOT INTERFACE FOR LARGE FORMAT TOUCHSCREEN DISPLAYS

MAY 2016

MARK ANDREW PRICE

B.S.M.E., GEORGIA INSTITUTE OF TECHNOLOGY

M.S.M.E, UNIVERSITY OF MASSACHUSETTS, AMHERST

Directed by: Professor Frank C. Sup IV

This thesis presents the design for a novel haptic interface for large-format touchscreens. Techniques such as electrovibration, ultrasonic vibration, and external braked devices have been developed by other researchers to deliver haptic feedback to touchscreen users. However, these methods do not address the need for spatial constraints that only restrict user motion in the direction of the constraint. This technology gap contributes to the lack of haptic technology available for touchscreen-based upper-limb rehabilitation, despite the prevalent use of haptics in other forms of robotic rehabilitation. The goal of this thesis is to display kinesthetic haptic constraints to the touchscreen user in the form of boundaries and paths, which assist or challenge the user in interacting with the touchscreen. The presented prototype accomplishes this by steering a single wheel in contact with the display while remaining driven by the user. It employs a novel embedded force sensor, which it uses to measure the interaction force between the user and the touchscreen. The haptic response of the device is controlled using this force data to characterize user intent. The prototype can operate in a simulated free mode as well as simulate rigid and compliant obstacles and path constraints. A data architecture has been created to allow the prototype to be used as a peripheral add-on device which reacts to haptic environments created and modified on the touchscreen. The long-term goal of this work is to create a haptic system that enables a touchscreen-based rehabilitation platform for people with upper limb impairments.

TABLE OF CONTENTS

CHAPTER Page

APPENDICES

LIST OF FIGURES

Figure Page

CHAPTER I

INTRODUCTION

A. Research Motivation

The touchscreen has become a ubiquitous human-machine interface. This technology allows intuitive interaction with a virtual space by allowing users to physically manipulate virtual objects with their fingers and hands as they do in the real world. However, physical input is not matched with physical output, forcing the user to rely entirely on visual feedback to control their gestures against an unresponsive, flat screen. Recent work has investigated techniques in introducing haptic feedback – or physical cues that target the user's sense of touch to convey information about the virtual space – to touchscreens. Researchers in this subject focus on two categories of haptic feedback: 1) Tactile feedback: conveying information about surface features (e.g. texture, elevation, temperature), and 2) Kinesthetic feedback: conveying force information that affects the overall motion of the user (e.g. pulling a user toward an objective, braking, redirecting).

Touchscreen haptics approaches tend toward tactile feedback, especially methods that produce fluctuations in friction between the user and the screen surface [1]–[3]. These approaches target the most common touchscreen devices – smartphones, e-readers, and tablets. A target they do not address, however, is the expanding field of touchscreen physical therapy. This field is evolving from methods that employ haptic robot arms and exoskeletons to deliver therapy exercises to patients with neural conditions that affect the upper limbs, including stroke [4]–[6].

Robotic upper limb stroke therapy is one of few alternative therapy approaches supported by substantial evidence to more effective than traditional physical therapy on stroke patients [7]–[9]. Robotic stroke therapy devices accomplish this primarily by providing high intensity, high frequency therapy sessions beyond the level of therapy a patient would receive from a therapist alone [8]. They provide haptic constraints for the patient to constrain their movements to useful paths, and in some cases, actively assist the patient in reaching targets. Another major advantage is the ability to record physical data throughout the therapy sessions. This allows rehabilitation robots to adapt the difficulty of the training to the performance of the patient as well as provide the patient with feedback [10]. The full potential of this ability is still being explored as a means of predicting responders or defining mechanisms of recovery [11]. However, rehabilitation robots are typically large, expensive, permanent installments that require training and supervision to operate. Touchscreen upper limb therapy approaches began as a response to the robotic method by mitigating many of the accessibility problems associated with robotic therapy. However, a touchscreen cannot currently provide haptic guidance as robot-assisted therapy can.

Touchscreen therapy has a number of requirements that prevents it from being compatible with existing touchscreen haptic techniques. It requires large-format touchscreens, which most existing touchscreen haptic approaches are not compatible with. Haptics for rehabilitative therapy also require kinesthetic spatial constraints to restrict user motion along a desired path or toward a goal. The patient must be able to “glide” along the constraint as it resists their motion off the desired path. Kinesthetic-focused touchscreen haptic approaches do exist, but focus on applying braking to resist user motion. This prevents these techniques from constraining a touchscreen user inside a set of smooth spatial constraints as required by therapy applications. A new approach to touchscreen haptics must be developed in order for it to be usable with touchscreen rehabilitation exercises.

B. Research Objective

This thesis focuses on creating a haptic device for a large-format touchscreen capable of presenting a user with smooth kinesthetic constraints, establishing a haptic tool that can be useful for upper limb touchscreen therapy. The device introduces the capability of redirecting and constraining a user as they navigate the touchscreen. It also leverages admittance control principles to provide constraints with compliant behavior in response to user force. This provides a means of delivering physical feedback to a touchscreen user based on the “stiffness” of the haptic objects encountered. The objective of this functionality is to allow for a variable amount of haptic assistance in the desired space, which changes the “difficulty level” of the therapy exercise as well as increases the depth of information gained from interacting with a haptic object. To increase its usability as a therapy tool, the device is designed to respond to haptic obstacles as defined by the touchscreen PC with no modification to the device code. This establishes the haptic device as a pure peripheral, so that therapists and other experts in stroke rehabilitation can design their own therapy exercises on an intuitive touchscreen interface and expect the device to respond correctly without needing to interact with its programming.

The long term goal of the proposed research is to use the presented work as a tool in haptic stroke therapy research to investigate the mechanisms by which users can retrain their motor functions. Little comparison exists between touchscreen and robotic exoskeleton therapy approaches. Without effective touchscreen haptics, this comparison lacks a key element of robotic stroke therapy. By creating a physically interactive game environment, the robot will emulate the effects of full scale rehabilitation robots on a touchscreen. The long-term objective of this research is to make haptic stroke therapy realistic on an intuitive and relatively inexpensive platform and open the possibility of integration with home-based therapy platforms.

C. Research Scope and Approach

This research has four main goals.

1. Identify and target useful haptic effects for stroke rehabilitation based on traditional and robotic therapy approaches
2. Design and fabricate a hand-sized platform for delivering these effects on a flat surface.
3. Fully integrate the robot with the Playsurface and develop control patterns for objects on the screen

4. Create a system enabling rehabilitation exercises using the complete haptic system, using expert clinical feedback.

The first three goals have been met. This thesis focuses on the results of their completion. The fourth goal extends beyond the scope of Master's level work, and offers a path into future work.

This project is being conducted in collaboration with the Center for eDesign at UMass Amherst, the Mechatronics and Robotics Research Laboratory, and in partnership with Michael White from FTL Labs of Amherst, Massachusetts. FTL Labs developed the Playsurface, a touchscreen PC and 35-inch projection display embedded in the surface of a table, providing a large and robust "touch-table". A haptic robot is being designed to interface with this platform in order to fulfill the research objectives. The project is funded by a two-year NSF SBIR Phase 2 grant.

CHAPTER II

BACKGROUND

D. Available Haptic Technology

Existing work in surface haptics on touchscreens is reviewed and compared with upper limb stroke therapy haptic technology in the following sections. Haptic technology for touchscreens focuses on two categories: 1) Tactile fingertip-based feedback, and 2) Kinesthetic braking approaches. The haptic effects that are achievable with these approaches differ than the haptic feedback that is used in stroke therapy robots, which directly actuate user motion. This chapter focuses on the functional gaps between these technologies.

1) Variable Friction Approaches

Varying the amount of friction force acting against a touchscreen user can deliver both tactile and kinesthetic feedback, often to a user's bare finger. Bau *et al*. [1], [12] modulate the electrostatic attraction force between a bare finger and a touchscreen surface by oscillating the voltage level of a transparent electrode beneath a transparent insulator. This method is known as electrovibration, and is capable of simulating textures and surface features. Ultrasonic vibration normal to the surface plane has been used to a similar effect by researchers at Northwestern University [2], [13], [14]. The coefficient of friction between a fingertip and the touchscreen surface can be reduced by increasing the amplitude of the surface vibration, creating a "squeeze film" of air. These processes are illustrated in Figure 1.The magnitude of the friction force generated by these methods is typically below 1 N, making them primarily tactile feedback techniques [15]. However, electroadhesion has been used to increase the normal force between a bare fingertip and an insulated surface by up to 7N. Electroadhesion is a similar technique to electrovibration, in which the electrode voltage is kept steady, and then inverted as the charge begins to leak across the interface [16]. Texture simulation from rapidly oscillating fingertip

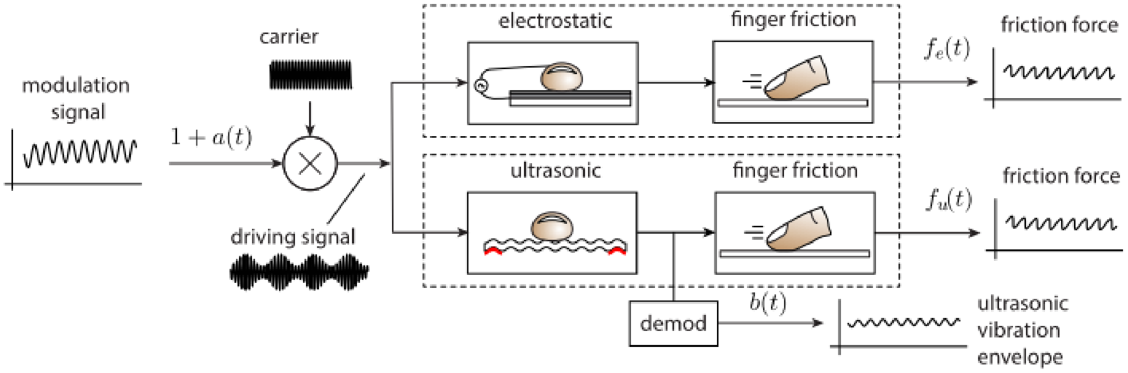


Figure 1. Two methods of touchscreen texture generation. The top branch describes electrostatic friction and the bottom describes ultrasonic friction displayed by a bare finger. [15]

friction is sacrificed for friction force magnitude, making electroadhesion a form of bare-finger kinesthetic feedback.

These methods are limited in that they display the effect to the entire screen at the same time, making the haptic effect useful only when a single touch is used. This eliminates multiple-user or whole-hand applications. However, multi touch and multi-user applications are possible when external hardware that can be controlled independently is introduced. Nakamura [3], [17] achieves multi-finger tactile feedback with electrovibration by employing contact pads on each finger that are energized separately. Marquardt *et al*. [18], [19] created high friction forces by actuating a mechanical brake against the touchscreen surface with their haptic tabletop puck (HTP). The HTP also displays tactile feedback by simulating surface elevation and stiffness with a servo-controlled rod on top of the device that the user touches with a fingertip. This type of device is known as a “height-adjustable tangible” and is primarily a tactile feedback technique [20]. These technologies are shown in Figure 2.

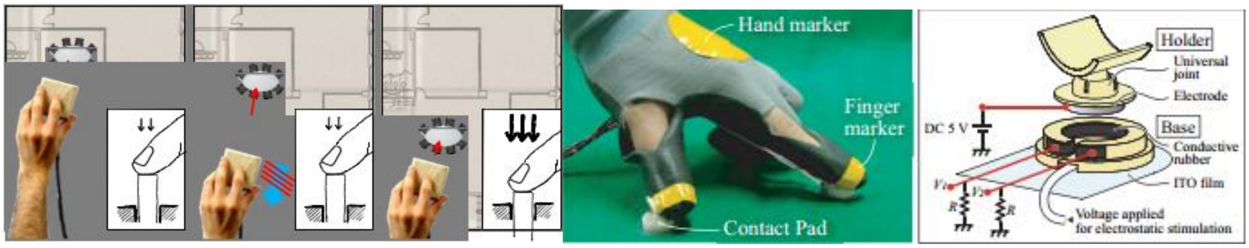


Figure 2. External devices for touchscreen haptics. Left - Haptic tabletop puck operation [18]**. The rod acts as a feedback device and an input device, while a brake provides a kinesthetic response. Right - Nakamura multi-finger electrovibration glove** [3]**. Multiple pads per finger allow multi-touch tactile feedback.**

2) Shear Force Generating Approaches

Another category of touchscreen haptics attempts to generate a lateral force beyond resistance due to friction. In Chubb *et al*. [21], a shear force is exerted on a bare finger in addition to the tactile feedback provided by controlled ultrasonic vibration with the ShiverPaD. This was accomplished by vibrating the touch surface laterally, and timing bursts of ultrasonic vibrations with phases of the lateral vibration to produce higher friction as the surface moved in one direction. In Dai *et al*. [22] a shear force is produced on a bare finger by vibrating the surface laterally and normally simultaneously at identical ultrasonic frequencies on a device called the LateralPaD. The magnitude of the generated force is controlled by varying the phase between the two vibrations. This method and the ShiverPaD method are illustrated in Figure 3. In Mullenbach *et al.* [23], a shear force is produced by physically moving the entire touch display in all planar degrees of freedom while modulating the coefficient of friction with ultrasonic vibration on a device called the ActivePaD. This method requires the touch display to be mounted on frame, and the prototype device has a 0.5 in range of motion, limiting the possible duration of the effect. It does not use timed vibrations to exert a net shear force like the ShiverPaD. In Saga and Deguchi [24] a lateral force is created by using a cable-driven system, directly pulling the user's finger along the surface plane. These approaches all generate lateral forces below 1N, which allows them to simulate edges and surface features, but is not on a scale to constrain user motion.

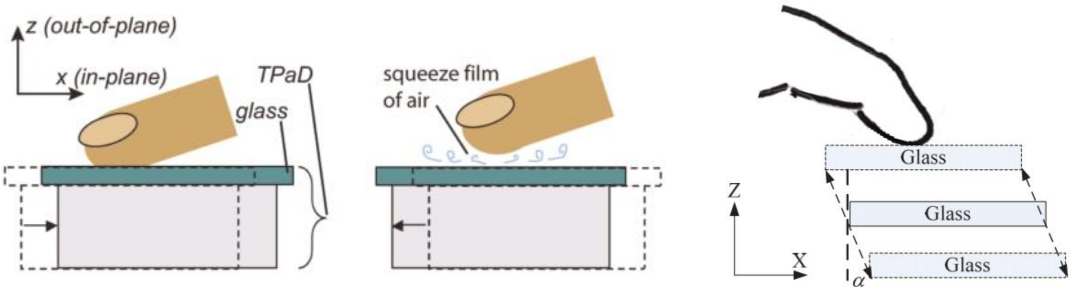


Figure 3. ShiverPaD (left) and LateralPaD (right) methods of producing shear force on a bare finger while modulating friction. [21], [22]

3) Three-dimensional approaches

Work has been done on creating three-dimensional haptic feedback for touchscreens as well. Technology from Microsoft actuates a touchscreen normal to the surface plane, allowing the user to displace the touchscreen itself against a simulated stiffness [25]. MudPad [26] changes the topography of the screen by activating a layer of magnetorheologic fluid with an array of electromagnets. FingerFlux [27] creates tactile feedback near to the surface of a touchscreen with a similar array of electromagnets that react to magnetic caps on a user's fingers. Other methods of midair haptics include using focused ultrasound to create air pressure nodes [28] and emitting targeted vortices into the air [29]. Each of these 3D approaches offers almost exclusively tactile feedback.

Of these works, the variable friction approach has been the most successful at generating forces large enough for kinesthetic feedback. However, sliding friction as a means of displaying haptic surfaces has significant drawbacks. It resists motion in any direction, which can lead to confusing haptic responses to multiple input cues, as Nakamura discusses [3]. This limitation also prevents it from being used to display smooth haptic boundaries, because a user cannot exert a force against a haptic wall and slide unimpeded along it at the same time. Sliding friction is also difficult to control continuously, as sliding friction at low speeds on smooth surfaces tends to have slip-stick behavior. Unless the normal force applied by the user is also measured, the amount of friction force applied is difficult to predict. However, the use of sliding friction is not required in order to generate significant lateral forces with the use of external hardware. While typical kinesthetic haptic interfaces use grounded, mechanically actuated systems that are too cumbersome to use effectively with a touchscreen, Cobots [30]–[32] offer an alternative.

4) Cobots

Cobots are haptic guidance systems designed to work with humans by constraining them to a useful workspace. They are purely dissipative haptic interfaces – mechanical actuation is decoupled from adding energy to the taskspace. Cobots were originally designed to assist assembly workers by constraining a part to a path or within a set of haptic walls while allowing the assembly worker to manually control the movement within those constraints. The project has since expanded into three dimensions and been commercially implemented, but the early work focused on a 2D haptic device called the Cobot unicycle [30]. These projects are shown in Figure 4.

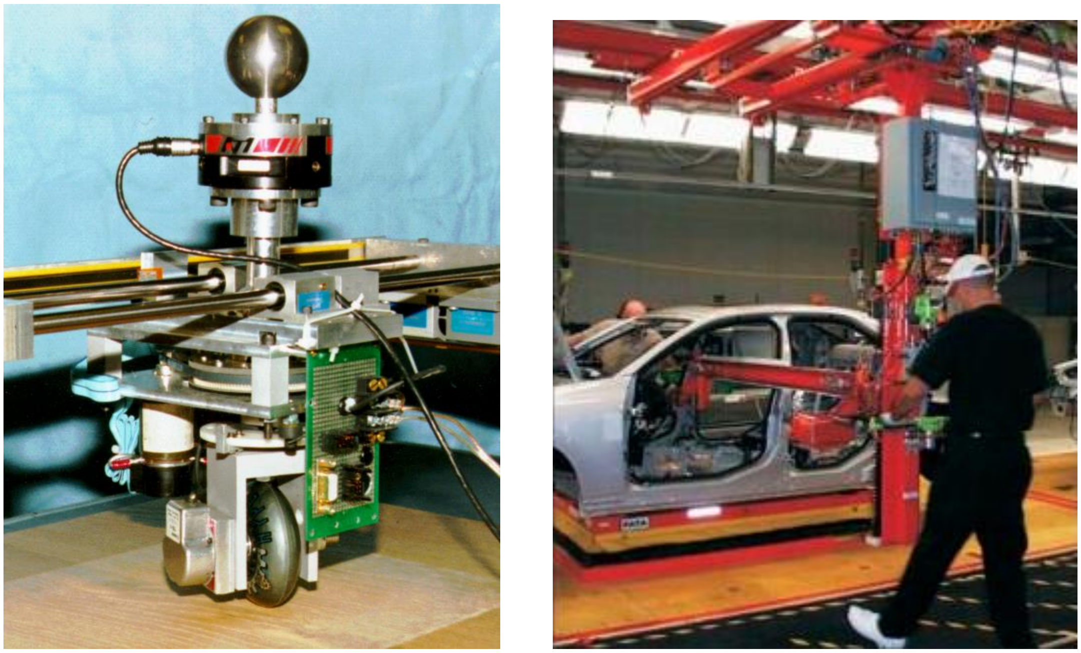


Figure 4. Cobot platforms. Left: The Cobot unicycle. Right: Car cockpit installation assisted by assembly scale Cobot. Produced by commercial offshoot Cobotics. [33]

This Cobot design is an actively steered, but passively driven single wheel, suspended on a two-dimensional gantry, with a single knob- like end effector attached to a 6-axis load cell. The steering angle of the wheel constrains the user moving the Cobot to a single degree of freedom in the direction it is facing. By nature of it having a single steered wheel, controlling the steering speed of the wheel equates directly to controlling the angular velocity of the device when it is in motion. By actively controlling the steering angle, the Cobot is used to constrain the user to an arbitrary 2D path, simulate two degree-of-freedom movement, and confine the device to one side of a wall of arbitrary shape.

Cobot control for the unicycle focuses on two primary control modes – "virtual caster" and "virtual wall". The virtual caster mode attempts to simulate the dynamics of a free particle in 2D space. In this mode, the angular velocity of the Cobot wheel is proportional to the force input by the user, and inversely proportional to the desired simulated mass and the translational velocity [30]. In the virtual wall mode, the Cobot is in caster mode until it travels to the coordinates of an arbitrary wall. It then constrains itself to the boundary of the wall until a force away from the wall is detected [30]. A path trace of a Cobot virtual wall collision is shown in Figure 5.

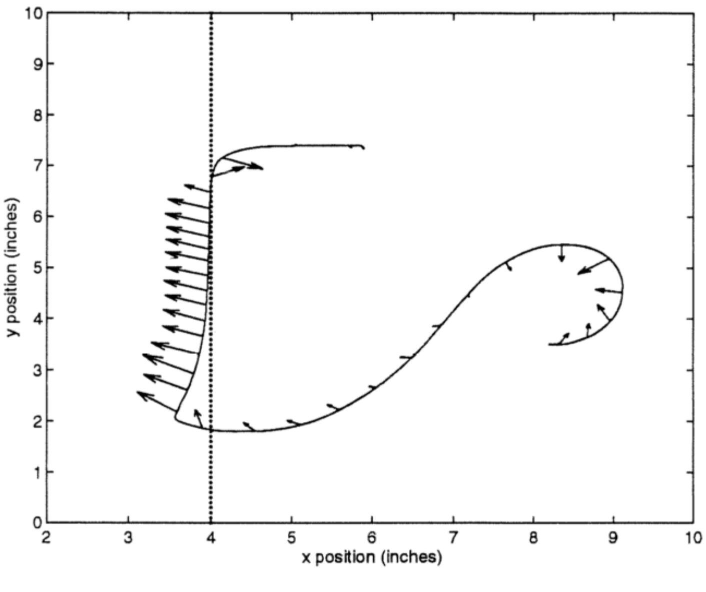


Figure 5. Force vector representation of the Cobot unicycle striking a virtual wall. [30]

The control law for Cobot unicycle path tracking control is based on its penetration into the boundary and its out-of-tangency with the path, and is described by Equation 1.

$$\tilde{\kappa}_{\mathrm{c}} = \frac{2}{L^2}p + \frac{2}{L}\theta \tag{1}$$

$\tilde{\kappa}_{\mathrm{c}}$ in this equation is the commanded curvature error, or the commanded steering speed divided by the translational velocity, from which the curvature of the path is subtracted [30]. L is the “lookahead distance”, or the distance at which the instantaneous radial path commanded by the steering velocity intersects the constraint path. This value functions as a gain parameter. p and θ are “penetration error” and “tangency error”, respectively. These error parameters result in an asymptotic approach to the constraint path.

5) Haptics in Rehabilitation Robots

Haptic feedback in most rehabilitation robots is achieved through motor actuation applying torque to the joints of a robot arm or exoskeleton. Examples of this are the MIT-Manus, ArmIN III, HapticMaster, and the MIME [34]–[37]. This actuation is dictated by using impedance control, which imposes force or torque in response to an external interaction with the device, or its inverse, admittance control (force input, displacement output). Ideally, the robot arm should exert next to no force on the user while they are operating within the desired behavior, and then behave like a spring when the user maneuvers out of the desired point, path, or area. Controlling the robot in this way ensures that it remains safe for a human to use and remain in close contact with, reacting with compliance when the user opposes the robot, and guiding them toward the desired space in proportion to the magnitude of their error. Robots implementing impedance control perform best when they are highly backdrivable – someone with a stroke-impaired limb should be able to move the robot easily. However, they must also be able to exert enough force to fully support and move the patient’s limb if need be. This form of control also demands precise force sensing, usually in multiple axes. These are demanding requirements, and contribute heavily to the expense of these devices. The methods of their use and their effectiveness are discussed in a later section.

E. Current Approaches to Upper Limb Stroke Rehabilitation

Not all upper limb stroke rehabilitation uses haptic devices to increase patient motor gains. This section reviews non-robotic therapy, the use of haptic robots in therapy, and the introduction of touchscreen and home-based therapy approaches.

1) Traditional Stroke Therapy

Stroke rehabilitation begins with an initial assessment of the patient’s motor function immediately after initial recovery from the stroke. Patients with severe motor impairment are not likely to be able to restore arm function, and rehabilitative therapy focuses primarily on teaching compensatory techniques. However, patients with moderate to light motor impairment can achieve significant functional gains in the affected limb if rehabilitative therapy begins early, and the frequency and intensity of the training remains high [38]. At the most fundamental level, successful stroke rehabilitation techniques require the patient to “engage in repetitive and intense use of novel tasks that challenge the stroke survivor to acquire necessary motor skills to use the involved upper limb during functional tasks and activities” [38]. This is traditionally performed during personal sessions with physical or occupational therapists, during which many different established therapy techniques may be chosen that satisfy these criteria. The patient is presented with exercises to increase the strength and flexibility of the affected limb, such as lifting small weights and going through a stretching routine. These exercises can extend to more involved practice of daily functional activities, requiring the patient to grab, manipulate, move, and place objects, sometimes with both arms at once. Examples of these techniques are shown in Figure 6.

While this type of therapy is effective, it is hindered by several limiting factors. Primarily, the amount and intensity of training that a patient needs to gain the full effect of the rehabilitation is difficult to achieve with this method. Therapists work with multiple patients in a day and are subject to fatigue. They are also limited by time; therapy sessions can only be as long as the therapist’s work schedule and number of patients allows. Patients must also travel to the clinic in order to receive therapy, further limiting its accessibility. This problem is exacerbated in rural areas, where rehabilitation clinics are sparse. As a result, most patients do not achieve the amount of motor recovery possible in ideal conditions [39].

Figure 6. Traditional rehabilitation techniques. Left to right - Constraint induced movement therapy, fine motor skill therapy, occupational therapy.

2) Robotic Stroke Therapy

Alternative methods designed to improve on the effectiveness of traditional therapy vary widely. These include, but are not limited to, electromyography (EMG) biofeedback, ballistic or resisted extension therapy, and Bobath exercises [38]. The first of these provides visual feedback to the patient when muscle activation is detected, with the goal of aiding the patient in controlling muscle activation. Ballistic and resisted extension both involve traditional therapy exercises with added elements of difficulty, designed to increase the number of motor units recruited during the exercise. Bobath exercises are therapist guided motions focused on control of the joints, designed to promote smooth voluntary motion and reduce spasticity. However, these methods show no strong evidence of improved patient results when compared to traditional therapy [38]. However, robotic rehabilitation is an exception, consistently and conclusively showing improved motor function in patients over traditional therapy [38]. Robotic stroke rehabilitation was introduced by the MIT-Manus in 1994, a 2 degree-of-freedom robot arm that allows them to navigate a 2D virtual environment displayed by a monitor by moving the end effector with the hand of their affected arm [34]. The Manus confines the patient's hand to a horizontal plane, and engages the affected arm in navigation exercises by passively restricting motion through haptic feedback, as well as providing active assist for more substantially impaired patients. For example, in one therapy exercise, the patient will be required to navigate their cursor – controlled by the end effector of the robot arm – through a maze displayed on the monitor. The walls of the maze are displayed by the robot arm by dynamically varying its mechanical impedance, becoming suddenly stiff and hard to push when colliding with a wall. Clinical studies performed with the Manus refer to this as sensorimotor therapy. For patients with higher levels of motor function, the robot can also perform progressive-resistance therapy, which generates an opposing force during reaching tasks [40].

The Manus established the paradigm for rehabilitation robots: robot arms or grounded exoskeletons that provide haptic feedback to a stroke patient as they navigate a virtual environment. When speaking about stroke rehabilitation robots, devices designed around the basic principles that the Manus established dominate the research field. However, over time, the simulations and games have become more immersive and involved, the task space has been expanded into a full three dimensions, and some rehabilitation robots engage and track hand, wrist, and elbow motion in addition to gross arm movement [35]–[37]. These extensions primarily developed as a means to accelerate the improvement of patients' capabilities to perform activities of daily living (ADLs). A sample of upper limb rehabilitation robots in use for therapy and for research is shown in Figure 7.

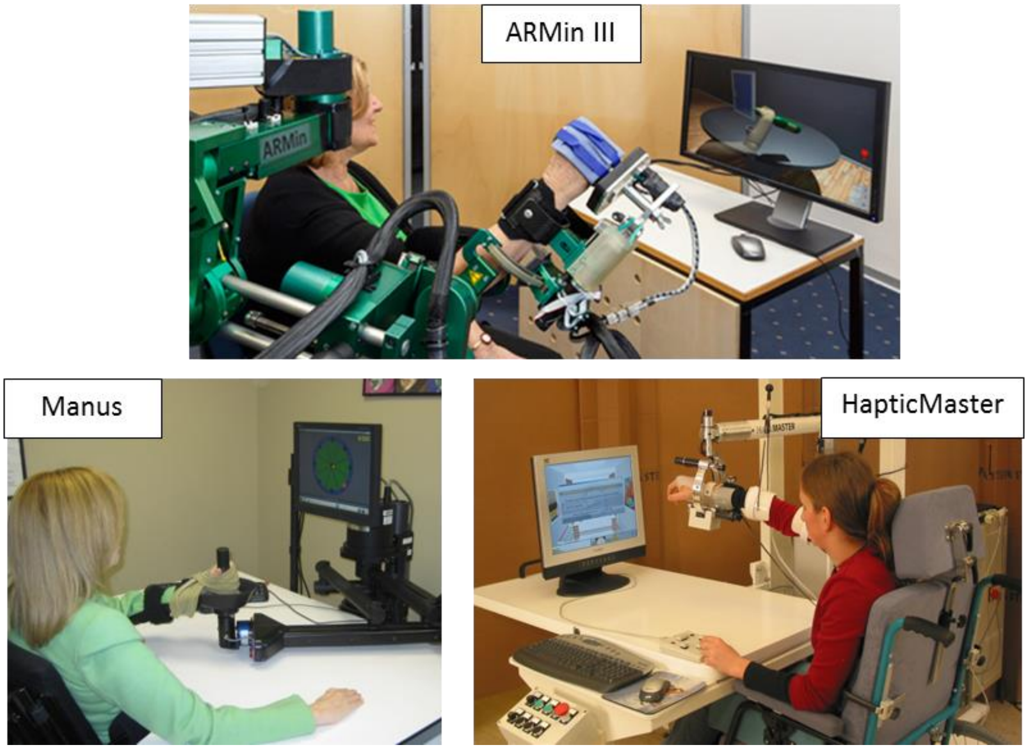


Figure 7. Rehabilitation robots. [34]–[36]

The clinical outcomes of the varying therapy modalities that stroke rehabilitation robots can provide are debatable. Patient improvement from robotic therapy over traditional therapy is statistically significant, but potentially not clinically significant [11]. Stroke therapy using rehabilitation robots also does not show equal improvement in arm function across all measures. Sources conflict as to the areas in which these devices show an improvement over traditional therapy. For example, a systematic review in 2012 by Mehrholz *et al.* for the Cochrane database concludes that robot assistance is more effective when attempting to improve activities of daily living and arm function, but not arm strength [9]. However, another review published in 2008 by Kwakkel *et al.* concludes that while robot assistance is more effective in improving arm function, it does not significantly improve activities of daily living, and makes no comment on arm strength [8]. As noted in these and other such reviews, the lack of large scale multi-center studies to evaluate this technology makes it difficult to derive conclusive results as to its specific benefits. However, the general consensus indicates that stroke rehabilitation robots with physical feedback elicit a significant improvement in motor function in stroke patients when compared with traditional physical therapy [41].

The reasons for this improvement are largely speculative, as the neural mechanisms for motor relearning are not fully understood [11]. The primary advantage that robot-delivered therapy appears to offer is the much higher number of movements the patient must complete in comparison to human-delivered therapy. Therapy robots greatly reduce the therapist fatigue problem, capable of providing very intensive therapy to patients for long periods. However, they also provide an immersive environment for the patient to interact with, helping to maintain patient interest and motivation while keeping the therapy intensity high. Haptic, visual, and auditory cues are all used to provide the patient with real-time feedback and assistance. They have the added advantage of providing quantifiable metrics of patient performance, useful to the patients as feedback and encouragement, and to the therapist as a measure of improvement. Many such rehabilitation robots use these metrics to build an adaptive learning curve into the therapy progression, with rehabilitation games automatically adjusting their difficulty to be optimal for the patient's level of motor function [10]. The MANUS additionally uses metrics like these in a patient assessment program to accurately predict the patient's Fugl-Meyer Assessment score – the most common clinical assessment used on stroke patients to assess motor function [42].

Despite these advantages, devices of this general design suffer from significant hurdles. All of them require patient training to operate, therapist guidance and supervision during therapy, and installation within a medical facility. They are high cost and limited in number, making it infeasible to deliver robot-aided therapy to all stroke patients. Furthermore, use of these devices still requires a patient trip to the clinic, just as traditional therapy does. For these reasons, access to devices like these is limited, and while the therapy performed on them is effective, the scarcity problem remains.

3) Touchscreen and Home-Based Therapy

More recently, as portable, personal electronics have become commonplace, work on improving stoke therapy has focused on utilizing personal computers and touchscreens to create rehabilitation games. This approach allows patients to perform their therapy exercises in a remote session with their therapist, or even unsupervised, with game data and performance logs being uploaded to the therapist to review when they have the time. Remote supervised therapy sessions also tend to be faster than in-person sessions, discounting travel time [39]. Patient compliance in performing therapy exercises at home has also shown to be higher when the patients are receiving telerehabilitation [39]. The devices most commonly used in home-based methods also eliminate many of the usability difficulties associated with rehabilitation robots, as these games are designed to be played on devices that are familiar to the patient. Large-format touchscreen based projects in particular have seen a large rise in interest, due to the naturally physically interactive nature of using a touchscreen, combined with the arm motion required to reach all points on a large area. For example, Spaulding Rehabilitation Clinic and Harvard Medical have begun using a Microsoft Surface – a commercially available large format touchscreen – to display games they create to emulate many of the reach-touch-drag-place games used by rehabilitation robots, in combination with a wearable accelerometer to detect and penalize compensatory behavior [4]. This system is use is shown in Figure 8.

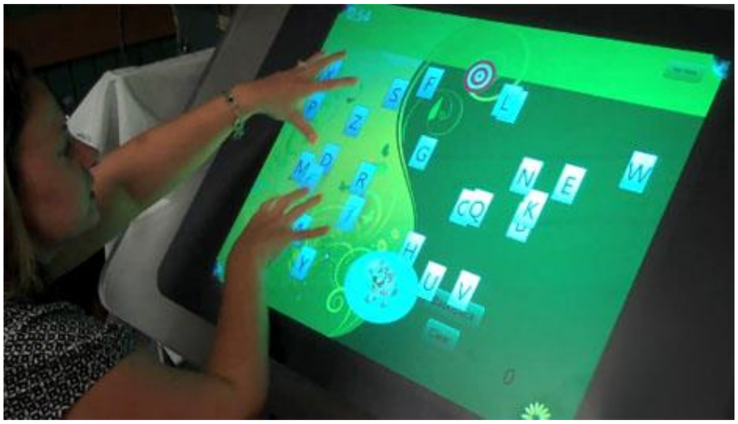


Figure 8. Spaulding touchscreen rehabilitation system. Displayed is a spelling rehab game requiring physical manipulation of letters across the screen. [4]

The group has based this work on several successful touchscreen-based upper limb rehabilitation projects that approach the problem in an extremely similar manner. Several design concepts are borrowed from a study performed by Mumford et al. in 2008, which used a touchscreen rehabilitation platform focused on traumatic brain injury. This work also highlights the use of augmented feedback, shown in Figure 9, in which the patient is presented real-time visual and auditory effects based on the actions they are performing and indicative of their performance. Patients using the system showed promising clinical results versus traditional therapy [6]. Annett et al. have developed a generalized upper limb rehabilitation suite for a system called the AIRtouch – a multitouch tabletop very similar in design to the Playsurface, and present a number of game concepts useful for therapy tasks [43]. These include rhythm games, reach and touch games, tracing games, and real time arcade-style games that require the patient to move and react quickly. The AIRtouch system is also shown in Figure 9.

A visual comparison between any of these projects and the Manus display, even updated for its current commercial form, also demonstrates the drastically lowered barrier to entry for making professional quality games since the years when rehabilitation robots were first being developed. Open source game engines and graphics packages are now easily available and fully documented on the

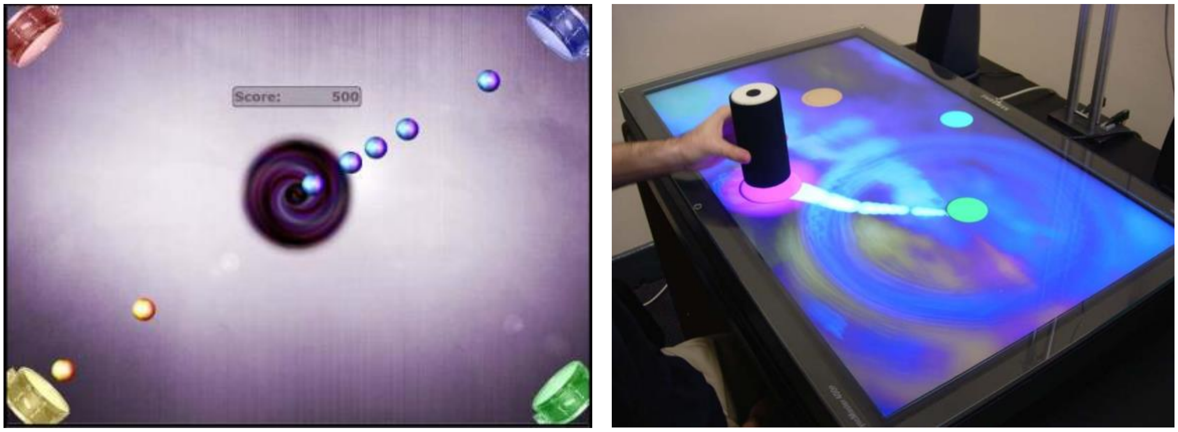


Figure 9. Touchscreen therapy and visual feedback. Left: Mumford augmented feedback [6]**. The size of the aura under the solid object and the ripple speed both correspond to patient action and performance. Right: AIRtouch rhythm game** [43]**. The patient must hit the drums in the corners when the balls from the center reach them.**

internet. This allows engineers and clinical experts to build rehabilitation games on par with commercial apps for download on smartphones and tablets, engaging the patient more effectively and keeping them motivated to continue therapy at the high intensity required for significant functional gains. It has similarly been proposed that the development of rehabilitation-specific games may not be necessary, considering the amount of professionally developed touchscreen and tablet based apps already widely available that engage many of the same tasks [5]. This approach also retains one of the largest advantages of rehabilitation robots in that the patient is engaging with a machine that does not fatigue, and can quantify the patient's performance.

However, to gain these advantages, portable and home-based therapy devices sacrifice the ability to provide haptic feedback. Haptic feedback is one of the fundamental features of rehabilitation robots and is attributable for much of their success – it is the reason that they continued to be developed with actuators rather than as purely passive input devices. Home-based therapy devices are almost exclusively passive input devices.

The Rehabtronics ReJoyce (2011), shown in Figure 10, attempts to emulate a typical rehabilitation robot arm, small enough to be portable and designed to operate with any computer through USB. It is marketed as a telerehabilitation tool, made for webcam sessions with a therapist. It has multiple degrees of freedom and an end-effector designed to engage both gross motor movement and fine motor movement based on the grip configuration the patient uses. However, the device is purely passive, only capable of recording patient input [44].

hCAAR (2014) – Home-based Computer Assisted Arm Rehabilitation – is the only example discovered of a true haptic rehabilitation interface developed specifically for home therapy. It closely mimics the original Manus in design, constraining the user to the horizontal plane. However, rather than simulating passive walls, it actively powers pulls the patient toward the objective, varying this level of assistance based on the patient's performance in an assessment exercise in which no assistance is given

[45]. A pilot trial was conducted, and patient improvement as measured by several metrics was in general slightly below the typical outcome obtained by clinic-based rehabilitation robots [45]. Patient compliance was scattered, as no minimum amount of therapy time was specified and no reminders and limited feedback were given to the patients [45]. Despite the attempt to build a haptic rehabilitation system compact enough to be used at home, the device weighs 54.5 kg in total [45]. This device is also shown in Figure 10.

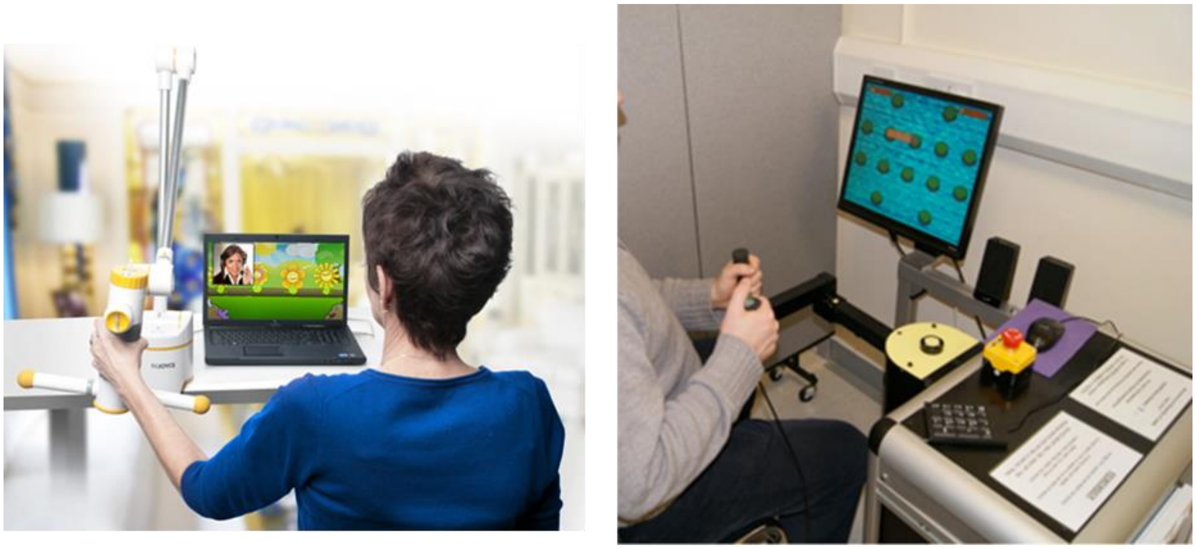


Figure 10. The ReJoyce (left) and hCAAR (right) - attempts to replicate rehabilitation robot arms for home use. [45], [46]

These two devices attempt to deliver the form of a rehabilitation robot arm in a home-based setting, but both ultimately fall short. Multiple degree-of-freedom robot arms are well suited to provide haptic feedback to a patient navigating a virtual environment, but in order to do so, they must be powerfully and precisely actuated, which makes the robot arm out of the weight and price range for a home-based therapy solution. The touchscreen becomes an attractive choice because it offers a high amount of physical engagement in a simple, portable, ubiquitous package. In fact, Rehabtronics released the ReTouch, a large format rehabilitation touchscreen, in 2013, two years after the ReJoyce.

Attempts have been made to provide patients with more physical engagement during touchscreen rehab games, but center primarily around providing passive “tangibles” for the patient to rest on the screen, shown in Figure 11. The Harvard/Spaulding/Ulster project, for example, includes a ball and a wand for the patient to roll or drag across the screen [4]. The Mumford project uses a simple plastic cylinder tracked by the touchscreen. A study conducted by Leitner *et al.* investigated the design of tangible objects for touchscreen interfaces in a rehabilitation setting, and concludes with concept applications for passive 3D blocks [47]. Passive tangibles are the standard in physical interaction with rehabilitative touchscreen applications, and the physical feedback they can provide is primitive at best. No home-based or touchscreen rehabilitative device approaches the level of physical feedback and assistance that rehabilitation robots can provide.

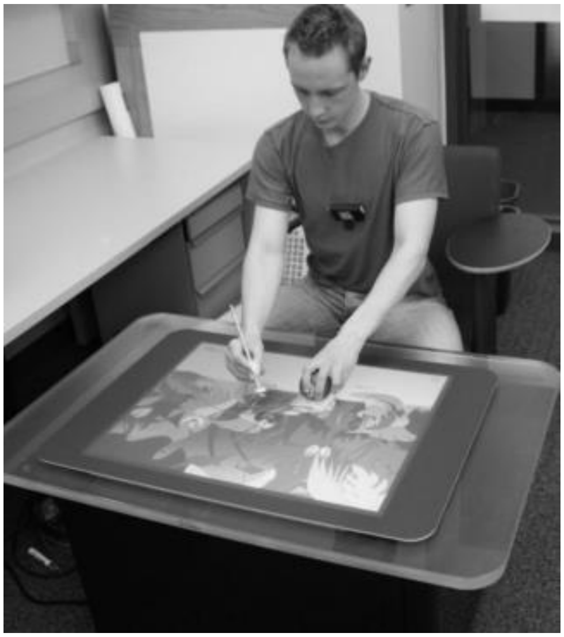


Figure 11. Passive tangibles (a wand and a ball) in use with a rehabilitation touchscreen. [4]

A solution that bridges this gap must deliver a kinesthetic haptic response on a touchscreen, with the ability to constrain user motion within a set of smooth, compliant boundaries. It must be operable with one hand without professional supervision to be a useful home-based therapy tool. It must also be modular and portable, in that it can be attached to any touchscreen capable of interfacing with it and function without reprogramming the device. Finally, it must be affordable, or below the cost of its companion touchscreen, otherwise it suffers the same accessibility problem as other upper limb rehabilitation robots. The design process to arrive at a solution for these requirements and the detailed design of the implemented solution are discussed in the following section.

CHAPTER III

MECHANICAL DESIGN

F. Initial Concepts and Design Process

These are centered on addressing the gap between touchscreen haptics, upper limb rehabilitation robots, and touchscreen rehabilitation approaches. The critical requirements focus on highly responsive force feedback output to the user. The design solution must output and control a reaction force against the user sufficient to stop or redirect whole-arm motion. The sensing requirements are directly related to the control requirements – the device must know how the user is interacting with the touchscreen and their location in the virtual environment. Lastly, manufacturability and costs for the device must be reasonable, it must be compatible with standard touchscreen PCs, and it must be small enough to be justifiable as a touchscreen accessory – its longest dimension should be less than half the touchscreen's shortest dimension, preferably smaller than a human hand. Specific functional requirements that address the performance needs of the complete system are presented in Table 1. Additional detailed design requirements are outlined in a House of Quality, found in Appendix A.

The requirements can be grouped into two overlapping categories: information requirements and physical requirements. Physical design requirements affect the mechanical design, and information requirements affect the electrical, software, and information architecture design. This section will address functional requirements that drive the physical device design.

The most critical requirements deal with the rendering of force to the user. The mechanical output bandwidth requirement refers to the maximum frequency the device can resolve an output force to the user. This metric translates to the resolution of the perceived haptic effect – lower bandwidth increases granularity of the sensation and causes less accurate transitions between haptic effects. This metric drives the actuator used in the device. It is worth noting that by constraining the problem to a 2D surface, the amount of actuator power required to meet this specification is lower than is required of a

Table 1. Functional requirements and specification targets.

	Marginal target	Ideal target
Control/render hand-scale forces(N)	10	20
Detect user input force (N)	1.50	0.10
Exceed voluntary hand motion bandwidth (Hz)	4.0	8.0
Device length (mm)	200	100
Device width (mm)	80	40
Device height (mm)	100	40
Sense current allowable space	Units implementation dependent	
Detect virtual objects (Hz)	50	1000
Structural safety factor	1.2	2.0
Exceed minimum stable controller bandwidth (Hz)	150	1000
Have compatibility with standard PC (Y/N)	Y	Y
Estimated cost (USD)	1000	100

three-dimensional robot arm or exoskeleton, which must support its own weight as well as the user's arm. Converting the kinesthetic control of upper limb rehabilitation robots to a touchscreen platform immediately relaxes the actuation requirements in this way, and significantly contributes to the potential lower cost of the device.

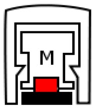
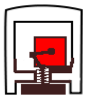
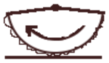
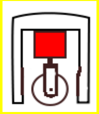
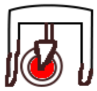
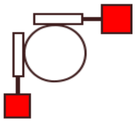
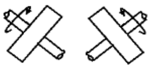
The ability to render and control hand-scale magnitude force is the next highest priority requirement. This is driven by the need for kinesthetic rather than tactile feedback on touchscreens to create a useful haptic touchscreen therapy device.

Two concept categories emerged from these requirements – wheeled concepts and variable friction concepts. Variable friction against the touchscreen surface is by far the most common approach to creating force feedback on touchscreens and served as a natural starting point for design concepts. Concept methods of achieving this included vibration of a high friction rubber surface against the touchscreen, direct servo-applied force of a similar surface down into the touchscreen, and controlling the contact region of a piece with a roughness gradient around its outer surface. A problem with these methods is that the amount of normal force against the touchscreen is user-dependent, so that the amount of friction force would not be able to be accurately controlled. Another vibration concept was generated to deal with this, in which the high friction piece was attached to a known mass, allowed to sit freely on the touchscreen, and vibrated independently of the device housing. These methods were all ultimately discarded due to being unable to address the shortcomings of existing friction-based touchscreen haptic systems, as they operate under the same basic principle. The concept that was evaluated to be most promising was vibration of a high friction piece at lower frequency and higher amplitude than ultrasonic bare finger techniques for a larger range of achievable friction forces. This technique was evaluated with preliminary tests, and the effect did not perform as predicted. The soft high-friction rubber used effectively damped and transferred the vibrations into the acrylic touchscreen surface. It was determined that vibration amplitudes large enough to break contact between the rubber and acrylic and lower perceived friction would be too loud and disruptive in a user's hand to be practical.

Wheeled concepts were inspired by the design of the Cobot unicycle. Variations on the Cobot design were considered in which the steered wheel is braked or actively driven. Other concepts resembled inverse Cobots, with the steering degree of freedom free on a caster and the rolling speed controlled. Breaking further away from Cobot design, one concept involved a free sphere similar to a mechanical mouse. Orthogonal rollers would contact the sphere and be individually driven or braked to control the rolling axis of the sphere. Another concept was considered in which two wheels were steered independently, allowing for control over the device direction and rotation, as well as a form of braking by intentionally aligning the wheels out of parallel.

A concept evaluation matrix including top concepts from both categories is shown in Table 2.

Table 2. Concept evaluation matrix. Each design was scored between 1 and 5 for each metric. Metrics were weighted with a value from 1 to 3 according to the technical importance calculated in the house of quality.

		Output hand-scale forces	Control hand-scale forces	Output frequency bandwidth	Size (higher is smaller)	Cost (Higher is cheaper)	
	Weight	2	3	3	1	1	**Total**
	Vibrating mass, high friction surface	4	2	3	4	5	**32**
	Direct actuation of high friction surface, spring preload	5	2	4	3	3	**34**
	Roughness gradient against surface, position controlled for variable friction	4	2	3	4	5	**32**
	Active steering, passive in wheel direction	4	5	5	3	5	**46**
	Passive (caster) steeting, active drive or braking	5	3	4	4	3	**38**
	Rolling ball, braked or driven with orthoganal rollers	3	4	3	1	2	**30**
	Double steered wheel, passive in wheel direction, brake with wheel misalignment	4	4	5	1	2	**38**

The single-wheel Cobot approach was ultimately chosen as the most suitable design. This approach has the advantage of being able to simulate "frictionless" constraints by taking advantage of the wheel dynamics, which is one of the primary features lacking in current touchscreen haptics. It also requires a single actuator, which limits the complexity and overall cost of the device, as well as reducing its size. This design also scores high on the output bandwidth to cost ratio, because the load on its actuator is entirely composed of friction force against the touchscreen. This allows a less powerful actuator to be used than if it were directly driving or opposing user force.

The size requirement is defined to allow single-hand operation. It is also defined to prevent the device from obscuring a significant portion of the touchscreen. The height restriction is to prevent significant separation of the user from the touchscreen – its aspect ratio should remain low.

The remaining physical requirements double as information requirements – the ability to sense force and its current allowable space. Force sensing is essential because it allows the device to estimate user intent, which is necessary to simulate haptic constraints. For example, the device must be able to tell whether the user is trying to push into a wall or pull off of it so that it can either constrain or free user motion. Force data in two axes is required to calculate a 2D user force vector, which tells the device what direction the user is trying to move in and the magnitude of that intent. Sensing the allowable space is a somewhat simpler requirement to meet, because the touchscreen is capable of keeping track of the location of elements in its virtual space. With a steered wheel design, the device itself must then be able to sense its own steering angle, which it can then place in context of the touchscreen environment to determine its allowable path.

Force measurements are typically obtained with a strain-gauge based load cell. However, this design option was quickly eliminated due to the cost constraint. Two-axis strain gauge load cells have a price point around $2000 and require routine professional calibration. Incorporating one in this haptic device would cause it to be far too costly to justify as a touchscreen haptic accessory. Alternatively, budget force sensing alternatives such as force-sensitive resistive strips, do not give consistent nor accurate force measurements and are only suitable for qualitative force gauging. Alternative force sensing concepts were developed that operate on the same principle as load cells – well characterized strain response to an applied load – but that can operate without high precision strain measurement. Two variations on a two-dimensional floating spring system were developed – an array of preloaded stock helical tension or compression springs, and an array of plastic flexures. The flexure design was chosen due to its high customizability in the 2D force-displacement response, and its elimination of backlash and other assembly inconsistencies. The displacement of the floating body must be measured in two axes. Solutions such as linear potentiometers were considered, but a Hall effect sensor with a paired magnet was chosen due to its ability to function as a contactless position sensor, eliminating any mechanical interference from the force measurement, as well as its low cost.

Steering angle sensing can be accomplished with an optical encoder, or another Hall effect sensor and magnet pair. Use of a potentiometer was eliminated immediately, because the number of rotations it can be turned through is limited. The Hall effect sensor was chosen because of its low cost, its ability to detect absolute angular position, its small size, and its lack of mechanical interference with the steering assembly.

The final design is a hand-sized robot with a single, motor-steered wheel that sits freely atop a touchscreen, shown in Figure 12. The device is completely safe for use by design – it cannot exert a force on the user that is not dissipative. The design prioritizes touchscreen-usability and scaling-down of sensing and actuation costs. This is achieved by reducing the amount of actuator power and sensing precision required from typical rehabilitation robot design through efficient use of additive manufacturing capabilities and constraining the problem to a two-dimensional surface. The robot provides the ability to simulate virtual walls that correspond with objects displayed on the touchscreen by controlling the allowed direction of motion through its wheel angle. It allows the user to supply all of the energy to the system, navigating the virtual environment on the screen under their own power while being steered by the haptic device. Two prototype iterations have been developed from this design. This section reports on the final iteration. A presentation of the previous design iteration can be found in [48]. This prototype with highlighted features is displayed in Figure 12. In the following sections, the details of this prototype are discussed. All part numbers and vendors can be found in Appendix C.

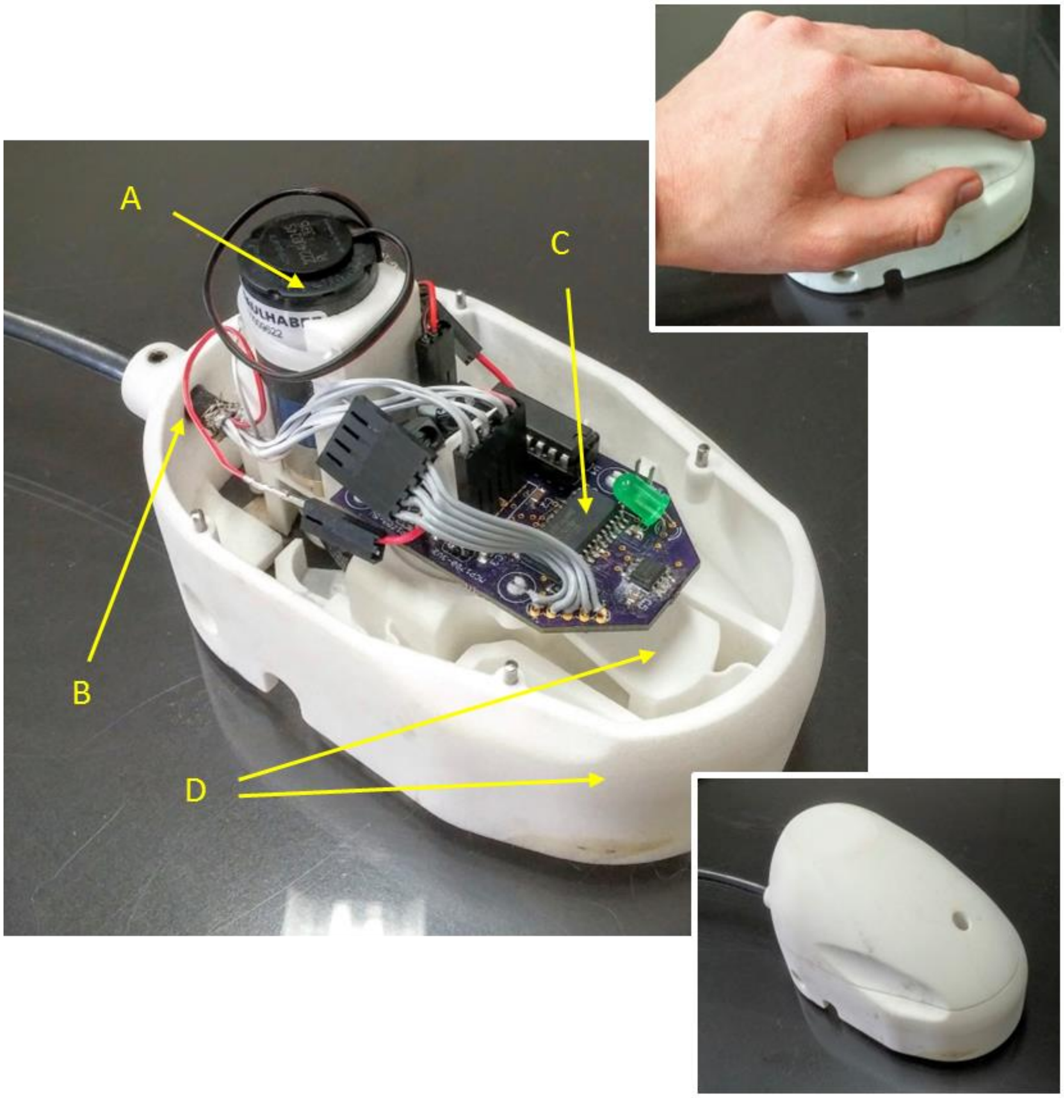


Figure 12. The prototype haptic robot. Major visible components – a) DC steering motor, b. Anchored power and PC communication tether, c) Embedded electronics board, d) Inner and outer stages.

G. Steering Sub-Assembly

Wheel steering is powered by a 24 volt Faulhaber 2224U024SRL brushed DC motor with a 9.7:1 planetary gearhead. The motor is connected to a centrally located hub with a miniature timing belt. The hub is 3D printed out of P2200 Performance nylon powder [49] as a single piece with its timing belt pulley. The hub houses a free-spinning urethane wheel. The steering hub sits inside a Teflon sleeve bearing to minimize internal friction. The bearing is mounted inside a stiff housing, fixed to the frame of the device. The hub assembly is tall relative to the diameter of the hub, and the central axis of the wheel is located well above the bottom edge of the hub. This design provides ample support for any bending moment exerted on the steering hub by a user pushing the device perpendicular to the wheel direction. The belt drive assembly includes two miniature idler pulleys that act as belt tensioners, located on either side of the motor shaft so as to maintain belt tension and prevent slippage during bidirectional motion. This subassembly is shown in Figure 13.

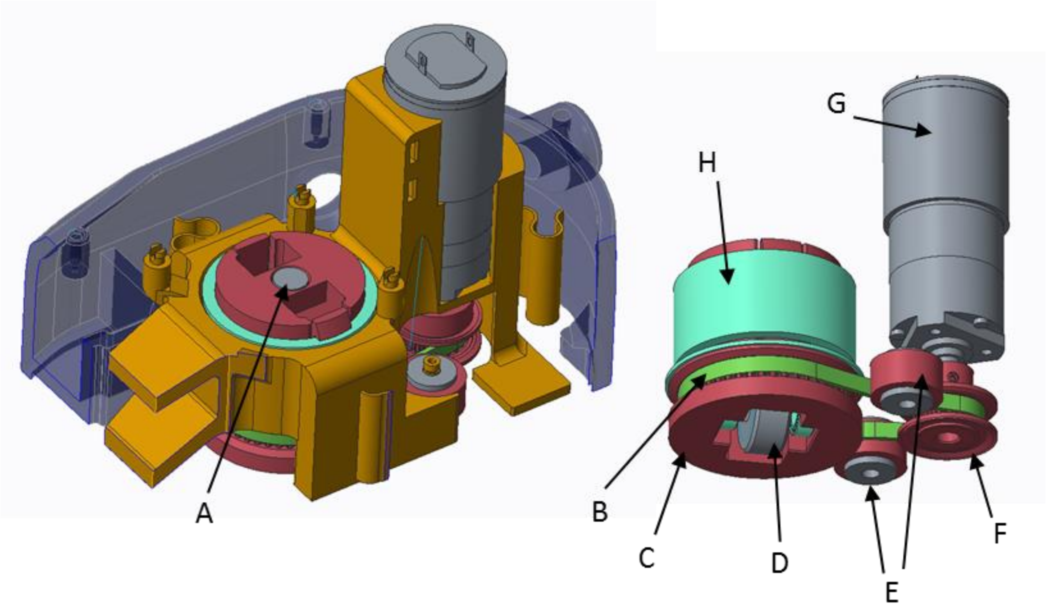


Figure 13. Steering sub-assembly. The right image has structural frames removed. Red indicates a belt pulley, and light blue indicates a bearing surface. A) Steering angle magnet, B) Timing belt, C) Steering hub combined with pulley and snap-fit tabs, D) Urethane wheel mounted in bearings, E) Idler pulleys for bidirectional belt tension, F) Drive pulley, G) DC motor, H) Teflon sleeve bearing.

H. Force Sensor Sub-Assembly

User input force is sensed by employing a two-stage nested design. The outer stage encompasses the device like a shell, giving the user a surface to grasp and push upon. The inner stage houses the wheel and steering assembly, as well as the internal sensor array. The two stages are connected to each other by an array of thin flexures, designed to allow the inner stage to deflect relative to the outer stage when the user exerts a force on the outer stage. This assumes no slipping between the wheel and the rolling surface. The flexure array was designed to provide a uniform spring constant between stages, independent of the deflection direction within a plane parallel to the rolling surface. Any deflection between the stages can be measured and scaled by the known spring constant in order to calculate force from the user.

The individual flexure design analyzed and built takes the form of a thin, tall S-shape. This design was arrived at after a series of cursory finite element analyses on several design concepts, targeted to have the lowest stiffness and most uniform motion in the desired plane. This design and its characteristic dimensions are shown in Figure 14.

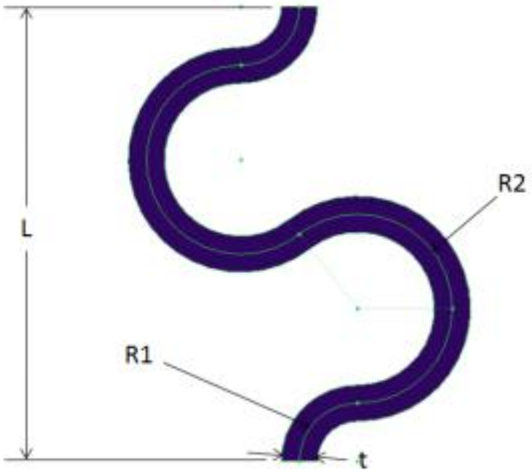


Figure 14. S-flexure geometric parameters.

The flexure stiffness in compression and in shear was first modeled by applying Castigliano's method to curved beam theory, using material properties provided by the manufacturer of the 3D printer used to fabricate the prototype [49]. The full formula for the flexure stiffness is included in Appendix A. This provided an analytical model to optimize for minimum stiffness. Assuming an array of four flexures at 90 degrees to each other in between nested, rigid stages, the deflection of each flexure is constrained to be equivalent to the others. With a loading condition that puts one set of flexures entirely into compression/tension and the other into shear, the objective function for this analysis is described by Equation 2.

$$f(R_1, L, t, h) = k = 2(k_c + k_s) \tag{2}$$

where R_1, L, t, and h are the four independent variables that characterize the shape of the flexure, k is the overall stiffness, and k_c and k_s are the stiffnesses of a single flexure in compression and in shear, respectively. R_2 is not an independent variable, as it has a unique solution dependent on R_1 and L given the existing geometric constraints. Constraint functions limited the designs to maintain enough structural strength to have a yield factor of safety of 1.5, as well as prevent impossible or impractical geometries. These constraint functions are also included in Appendix A.

This optimization was achieved in ModelCenter through a genetic algorithm to converge roughly on an absolute maximum, then through a sequential quadratic programming algorithm to converge on a precise optimal design. The Castigliano approximation began to break down at very high length to thickness ratios due to neglecting the effect of Poisson's ratio, however, so the final flexure design was found iteratively through finite element modeling and physical testing of fabricated designs. A test rig was created to measure the spring constant of selected designs on an Instron tensile test machine, using a 20 N load cell. This model assumes an array of four flexures, two of which are in pure compression or tension, and the other two of which are in pure shear. In order for this model to be valid for the system as a whole, the stiffness of the entire system must not vary as the direction of the force changes, invalidating these assumptions.

The Instron tests confirmed this assumption for a 90 degree sweep at 15 degree intervals. To further validate the constant stiffness assumption, the flexure array was modeled in Creo 3.0 and imported into Creo Simulate for a finite element analysis. This analysis accounts for the out-of-plane deflection of the inner stage relative to the outer stage when the device is compressed into the touchscreen surface. The model was loaded at 14 N on the outer stage with 15 degree intervals through a full 360 degree sweep. The inner stage was constrained to have no deflection. The results of this analysis are plotted in Figure 15. Uncertainty reflects two standard deviations.

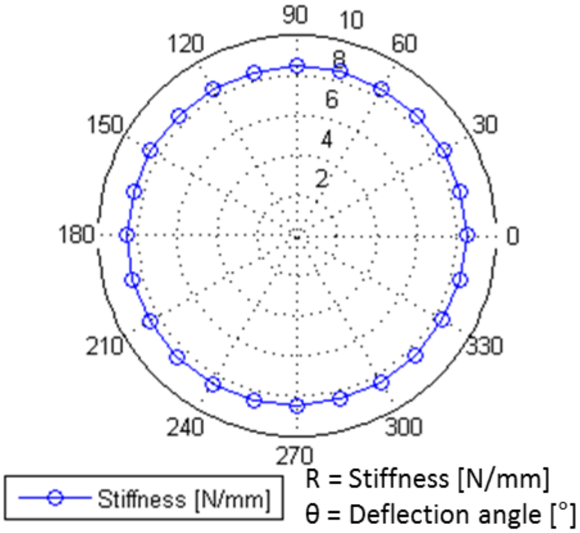


Figure 15. Flexure stiffness with respect to loading angle. Mean stiffness is 8.48 ± 0.03 N/mm.

The implementation of this flexure array in the prototype frame is shown in Figure 16.

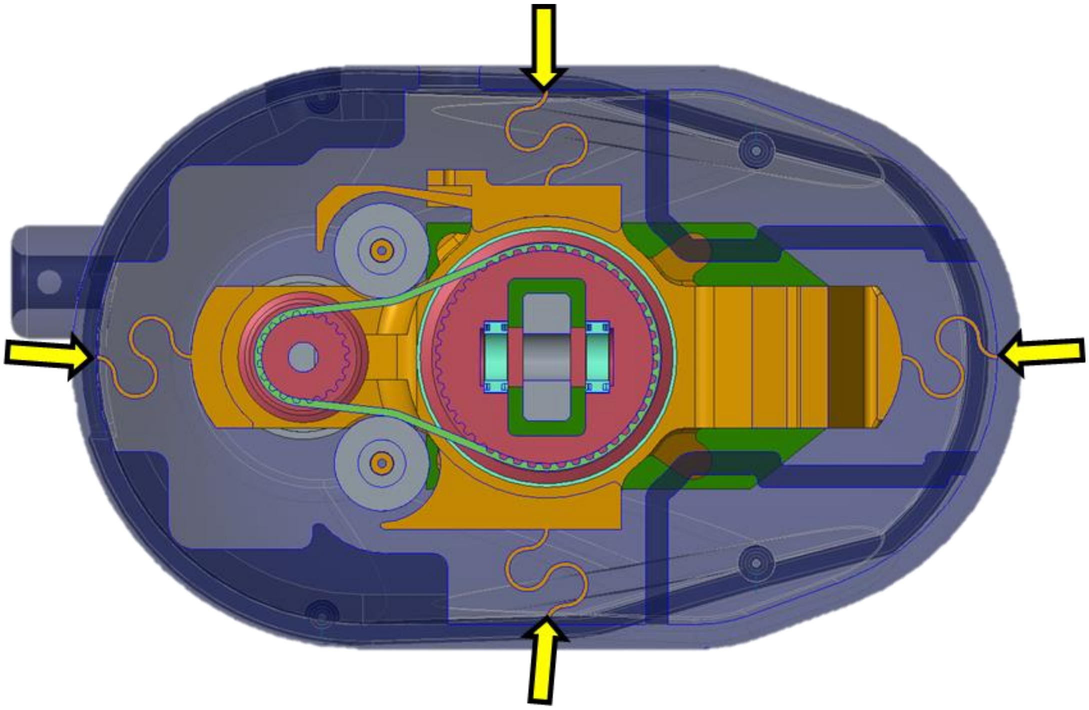


Figure 16. Bottom view of prototype frame with bottom layer of frame cut away. Flexures are highlighted with arrows.

The deflection of the inner stage relative to the outer stage is measured through a Melexis Triaxis Hall effect sensor, mounted on the inner stage, and a neodymium magnet mounted on the outer stage. Each Triaxis Hall sensor is low cost (~$5 USD), and can detect unique magnetic flux values when the magnet is in each of the four quadrants of a plane parallel to the sensor chip. An appropriate distance for the magnet from the sensor chip was obtained by simulating the magnetic flux density field around the selected magnet in COMSOL Multiphysics software. The results of this simulation are shown in Figure 17.

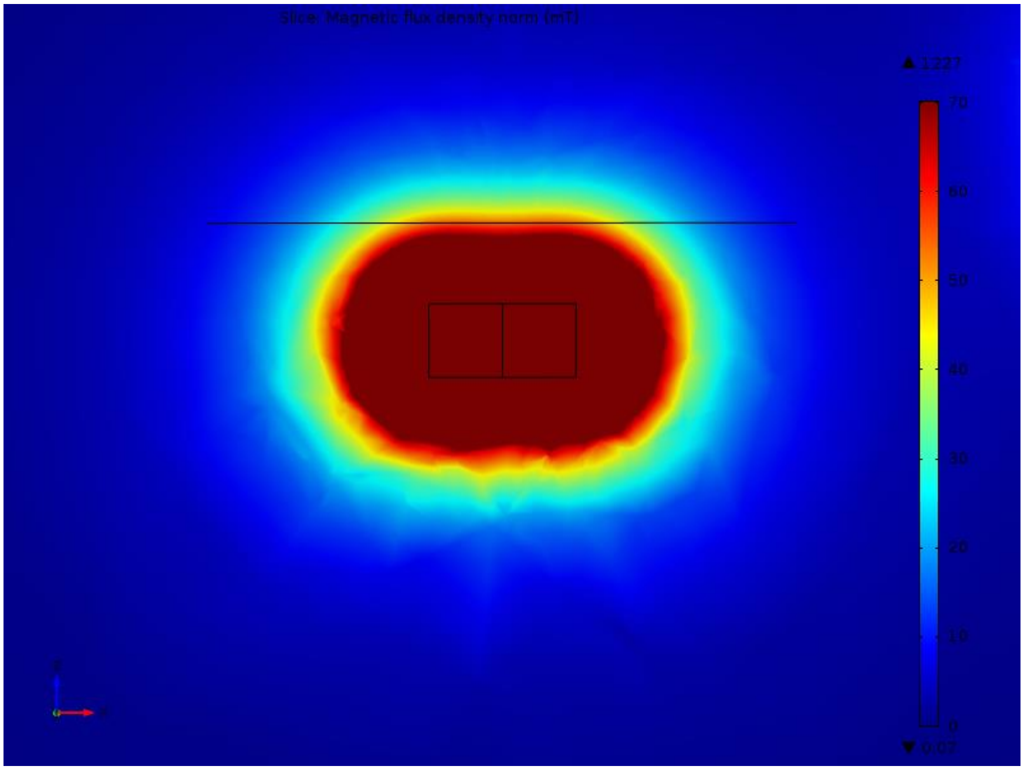


Figure 17. COMSOL simulation of the magnetic field surrounding the Hall effect paired permanent magnet. Dark red reflects flux density above sensor saturation, and dark blue is below the sensor minimum. The line above the magnet is the minimum allowable sensor distance.

To convert these readings to displacement, a model is required to estimate displacement in x and y with 3-axis flux density values as input. An experimental approach was taken rather than extracting simulated magnetic field values from the COMSOL analysis in order to account for sensor imperfections and tolerances in the permanent magnet properties. This was done by mounting the sensor on a 0.001 inch (0.0254 mm) resolution 2-axis micrometer positioning stage. The magnet was mounted above the sensor at the selected distance. A total of 2920 flux density values were recorded for an array of 76 locations across all four quadrants with displacement up to 5.18 mm from the origin. A model for this data was obtained through the software Eureqa [50], which iteratively detects equations to describe a dataset through symbolic regression and plots potential models on an accuracy vs complexity Pareto curve. The model used is described by Equations 3 and 4.

$$x = .000551B_x + \frac{3.12B_x - 51.6 - 0.0838B_y}{120 + B_z} \tag{3}$$

$$y = .000607B_y + \frac{2.97B_y - 5.28 - 0.0495B_x}{70.8 + B_z} \tag{4}$$

In this model, x and y are the x and y position of the magnet relative to the sensor in mm. B values refer to each orthogonal component of the magnetic flux density as measured by the sensor (values range as integers between ±4095). This model has 0.027 mm mean absolute error in x and 0.013 mm in y when used to predict test values in the dataset. These values translate to 0.23 N and 0.11 N when multiplied by the flexure stiffness, respectively. Observed vs predicted value accuracy plots as well as the model placement on the Pareto curve are shown in Figure 18.

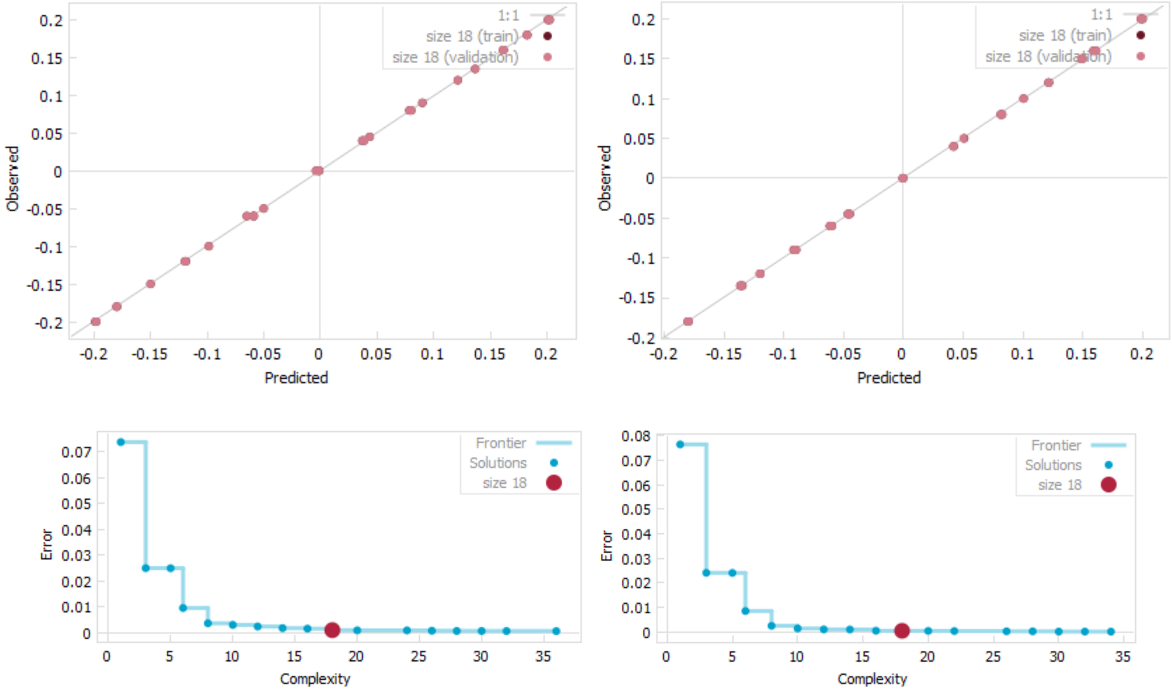


Figure 18. Model accuracy plots as generated by Eureqa. Visible points represent validation points not used to build the model.

Using this model, the Hall effect sensor has a resolution of 0.3 µm, accounting for noise in the sensor reading. The result of this analysis is that the 2-axis force sensor has combined 0.23 N uncertainty in x and 0.12 N uncertainty in y, when sensor resolution, stiffness irregularity, and model accuracy are considered and actual deflection is at 0. This is valid when assuming purely planar deflection and all user force being absorbed by the flexures.

I. Angular Position Sensor Sub-Assembly

The mechanical design allowing the device to sense its steering angle uses an identical Melexis Hall effect sensor to the force sensor, mounted directly above the steering hub. The top of the steering hub houses a diametrically polarized neodymium magnet centered on the steering axis. Magnetic field lines pass through the sensor along the sensor plane, changing orientation as the magnet rotates relative to the sensor. The angle measurement can then be made by comparing the orthogonal magnetic field components in the sensor plane and taking the arctangent of their ratio. The magnet distance to the sensor remains unchanged as the device operates, so magnetic flux density magnitude remains the same as the steering angle changes, greatly simplifying the model. This design provides a contactless, absolute angular position sensor with 0.022 degrees of resolution. However, placing the magnets for both sensors inside the same device causes susceptibility to magnetic interference. The two sensors must be separated by magnetic isolation foil to prevent this effect.

The placement of these sensors in the prototype is shown in Figure 19.

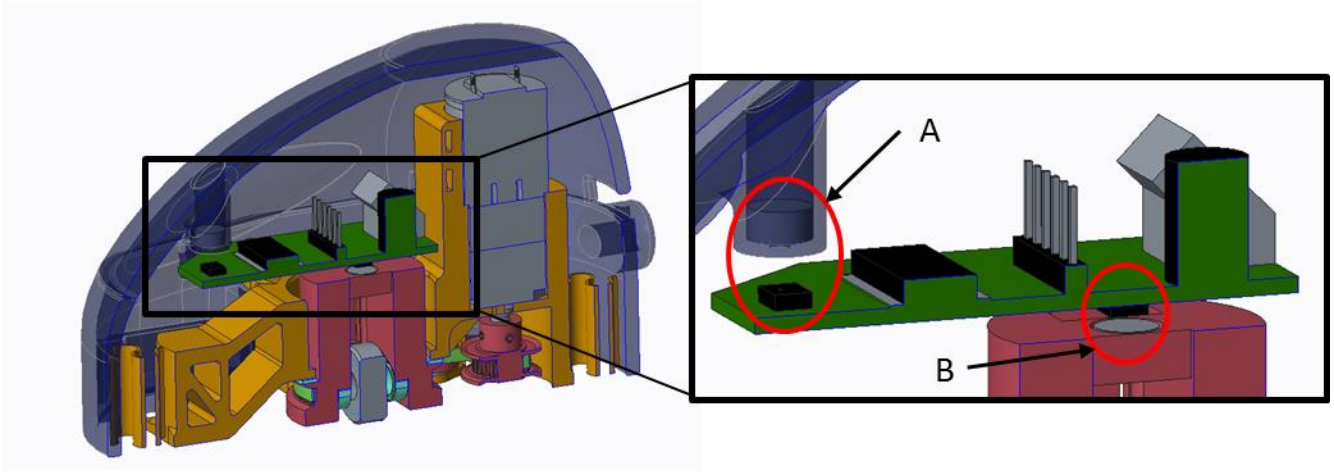


Figure 19. Hall effect sensor placement. A) Force sensor/magnet pair, B) Angular position sensor/magnet pair. Magnetic isolation foil is not depicted.

J. Design for Additive Manufacturing (AM)

This device was designed to be fabricated with an EOS Formiga P110 selective laser sintering machine out of P2200 Performance nylon powder [49]. This method supports complex internal geometries, including overhangs and nested parts, without the use of any separate support material. This machine sinters 0.1 mm thick layers, allowing for finely detailed features. The force-sensor flexures, for example, are less than a millimeter thick and over 10 mm long, which makes their design difficult to machine or to injection mold. Making use of these capabilities, the nested shell design is fabricated as a single piece. This eliminates the need for assembly, as well as the potential for backlash, hysteresis, and contact wear. This has also allowed for the rapid iteration and testing of the flexure array design.

Utilizing AM allows for the combination and rapid customization of typically separate components, and is a significant factor in making the presented design a touchscreen-practical size. For example, the steering hub and the timing pulley that drives it are a single compact piece. Employing this method also allows for a highly customizable motor reduction ratio, as the size of the load pulley can be easily modified and fabricated at will. The same principle has been employed in securing the sensor array to the device frame. Snap fit PCB standoffs have been incorporated into the frame design for ease of modification, to reduce foreign object debris near the sensitive electronics, and to prevent the eventual thread stripping of the soft frame material after frequent removal.

A hybrid approach was taken with portions of the design, where the printed material would not be expected to survive repeated use, and were reinforced with metal parts. The idler pulley shafts, for example, are printed directly onto the inner frame of the prototype, but have steel pins running through the center. These shafts have a high length-to-diameter ratio and are in a perpetual cantilever beam loading condition due to the belt tension, which pushes the limits of the printed material. Assembling metal shafts to the device, however, would likely need to be threaded into the material, which could easily be stripped over time. Adding steel reinforcement to the printed piece allows for a simple and secure press fit into the printed shaft.

The belt-tensioning idler pulleys are another example of hybrid design. One pulley is connected to the device frame with a leaf spring flexure, while the other is mounted rigidly. This configuration allows for adjustable tension with the inclusion of a set screw which exerts a preload onto the leaf spring flexure. A previous iteration of the prototype did not include the set screw, and depended entirely on the flexural properties of the plastic to maintain the appropriate level of belt tension. Minor variations in the pulley center distance and belt length could not be adjusted for. Including a set screw solves this problem. Thread stripping is less of a concern than in the pulley shaft example, because the set screw does not experience a bending load condition and is almost completely embedded in the printed material.

Despite the explicit design for additive manufacturing that the prototype employs, no published DFAM method or algorithm was used to design it such as those proposed by Ponche *et al.* [51] or Adam & Zimmer [52]. The efficiency of the design could likely be improved by employing such a method. This design lends itself as an interesting case study for the effectiveness of DFAM methods and the decision to use AM in general. A discussion on the use of AM to design this device and alternative methods of achieving its critical-to-function constraints can be found in [53].

CHAPTER IV

SYSTEM ARCHITECTURE

K. Surface Robot Interaction with the Touchscreen

The surface robot is not a standalone system; it must interact with a touchscreen in order to function. It depends on the touchscreen for its position, velocity, and orientation in the haptic environment, as well as any properties of haptic objects to render. The presented prototype has been designed to function with a large format touch table called the Playsurface (FTL Labs, Amherst, MA). This platform has a 770 mm by 460 mm rear projection screen, mounted horizontally on a wooden frame. It uses an array of infrared LEDs to flood the screen with IR light, and detects reflections of that light back into the table cabinet with an upward facing IR camera. All touches are registered through image processing of the reflections, allowing the Playsurface to recognize shape, size, and orientation of reflective objects that rest against the surface. This also allows it to recognize approximately 100 touches simultaneously. In this application, the Playsurface detects the position, orientation, and velocity vector of the robot by identifying its unique fiducial marker. This consists of one long and one short reflective strip adhered in parallel to the bottom of the device. The Playsurface draws a line from the large marker to the small marker and uses this vector to determine the device orientation. The midpoint of this line is interpreted as the centroid of the device and is used as a point representation of the device position.

In the closed-loop system, the surface robot detects and interprets its own sensor readings, and communicates these to the touchscreen as wheel angle and force input. The touchscreen also stores the position, size, compliance, and shape of haptic objects displayed on the screen. It communicates the device position, orientation, and velocity to the surface robot relative to the nearest haptic object, along with that object's shape and compliance values. This step tells the device its spatial parameters within the nearest object's relative coordinate system. The device then selects a control mode based on this information and performs it. This allows the device to employ a general response to any object regardless of its parameters without needing the device to "know" where the haptic objects are located beforehand, nor any of their other properties. The complete system architecture and flow of information is illustrated in Figure 20.

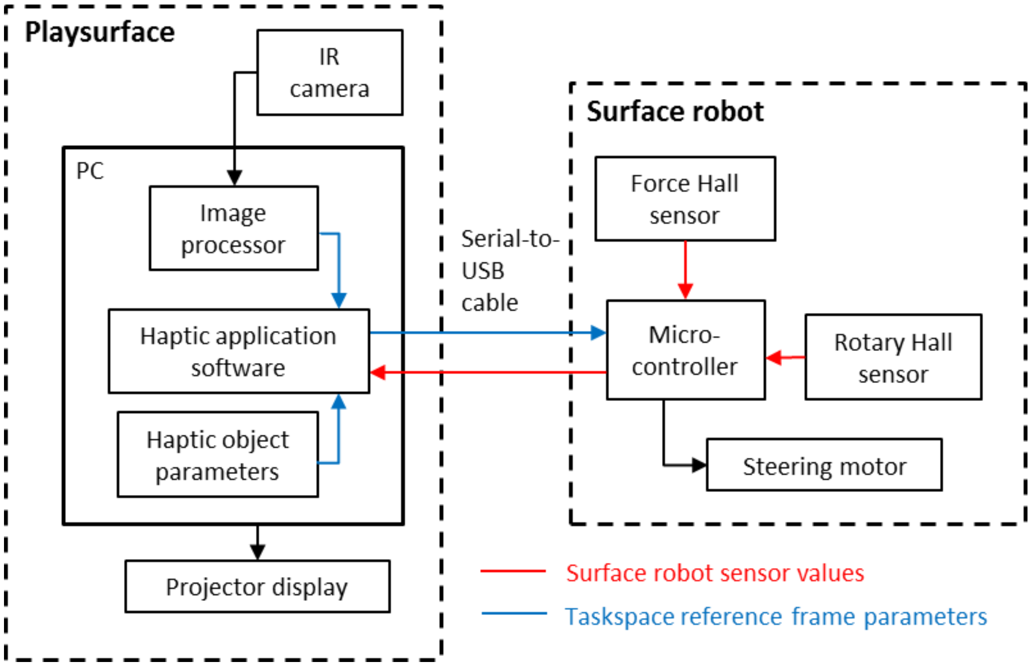


Figure 20. Full system information flowchart. The PC sends the surface robot information about the haptic environment and its place in it. The surface robot sends the PC its internal sensor readings for data logging and visual display to the user.

L. Information flow specifications

The remaining functional requirements not addressed in the mechanical design section are shown in Table 3. The mechanical requirement to "sense current allowable space" has been refined to reflect that this requirement translates to measuring the steering angle of a steered device.

Table 3. Information handling design requirements.

	Marginal target	Ideal target
Sense current allowable space (°)	0.5	0.088
Exceed minimum stable controller bandwidth (Hz)	150	1000
Detect virtual objects (Hz)	50	1000
Have compatibility with standard PC (Y/N)	Y	Y

Steering angle sensing has been discussed, but can now be compared to a design specification. The resolution obtained by the Hall effect sensor implementation is 0.022 degrees, which exceeds the ideal target specification. Noise in the digital signal has been observed at a total uncertainty range of 3 least-significant-bits (LSBs), resulting in a sensor uncertainty of 0.066 degrees, which still outperforms the ideal specification. This value was set by the resolution of a 12-bit optical encoder, which is a typical high-precision angular position sensing solution.

Controller bandwidth is driven by the CPU used to run the device. The surface robot uses a Microchip dsPIC33FJ64MC202 microcontroller to communicate with both Hall effect sensors through SPI and update its stored variables for wheel angle and input force. It updates these readings every 1.5 ms (667 Hz). Direct Memory Access (DMA) functionality is used in the microcontroller to process serial communications while running its control loop simultaneously. Without this feature, the controller bandwidth has a maximum frequency of 181.8 Hz (sample time of 5.5 ms).

The “detect virtual objects” metric refers to the rate at which this information is updated. The touchscreen and the robot communicate via USB-to-UART serial cable. This line of communication was originally intended to be wireless, using 1 mW XBee chips to transfer the serial data through radio frequency. This idea was discarded due to the slow transmission rate and round-trip time, which often exceeded 20 ms. This results in large position lag and unstable control for any reasonable movement speed (greater than 0.1 m/s). The USB-to-UART cable, in comparison, has a round trip time of approximately 1.5 ms with the PC serial port set to its minimum package-received wait time (1ms). The infrared camera in the Playsurface runs at 60 Hz, which sets the maximum frequency for new position information. This is close to the marginal acceptable value of 50 Hz. This prototype requests information from the Playsurface at 133 Hz to exceed the Nyquist frequency of 120 Hz to avoid aliasing. It is worth noting that the Playsurface performs approximately 3 ms of post-processing on each camera frame, meaning that in combination with the communication delay, position information is at least 4 ms old by the time the robot receives it.

The device is PC capable in that once it is initially programmed, it can interface with any touchscreen PC by using a commercial USB-to-serial cable for all further communication. The information structure that allows this is defined by a haptic object-based reference frame that the touchscreen provides to the robot. This reference frame is described in Section IV.D.

M. Position Prediction

In order to mitigate the control instability caused by a 60 Hz position update, a position prediction algorithm is used during controller samples for which new position information is not received from the touchscreen. This is done with a linear extrapolation based on the last known translation velocity and the last detected wheel angle. This position estimator is described by Equations 5 and 6.

$$x_{n+1} = x_n + vt\cos\theta \tag{5}$$

$$y_{n+1} = y_n + vt\sin\theta \tag{6}$$

where θ is the wheel angle with respect to the touchscreen coordinate system, t is the time between controller samples, v is the last known translation speed, and x and y are the absolute x and y-coordinates on of the robot centroid on the touchscreen. Wheel angle is updated at 667 Hz, so the absolute position is updated with this predictor 11 times for every actual touchscreen value. This assumes that changes in robot orientation on the touchscreen are negligible between position updates.

A more complex, second-order predictor has been used, which assumes that the robot path is a circular arc between each sample, tangent to the wheel angle at the current and previous sample. This is described by Equations 7 and 8.

$$x_{n+1} = x_n + \frac{v}{\omega}(\sin\theta_n - \sin\theta_{n+1}) \tag{7}$$

$$y_{n+1} = y_n + \frac{v}{\omega}(\cos\theta_n - \cos\theta_{n+1}) \tag{8}$$

where $\omega = \frac{\theta_{n+1} - \theta_n}{t}$, or the average steering speed between the previous and current sample. This predictor causes unstable behavior in practice, however, due to its singularity when $\omega \approx 0$. Using this predictor only when the change in angle is significant, then using the linear predictor when change in angle is small does not produce any noticeable improvement over using the linear predictor alone. This is likely because with 1.5 ms between samples, the difference between a linear and a radial path is small.

N. Object-Based Reference Frame

In the applied information model, the touchscreen serves as a taskspace reference frame. Without information from the touchscreen, the device can only control wheel angle and measure force relative to its own local coordinate system. With haptic-object-relative position and orientation, the surface robot can transform its measurements to a coordinate system that reflects the taskspace the touchscreen provides. This allows it, for example, to hold its wheel along the vertical dimension of the touchscreen as it is turned manually by the user. This replaces the need for the external frame used by the Cobot, which was used to obtain absolute position data and fix the rotational degree of freedom.

The specific variables that the surface robot receives from the touchscreen are illustrated in Figure 21.

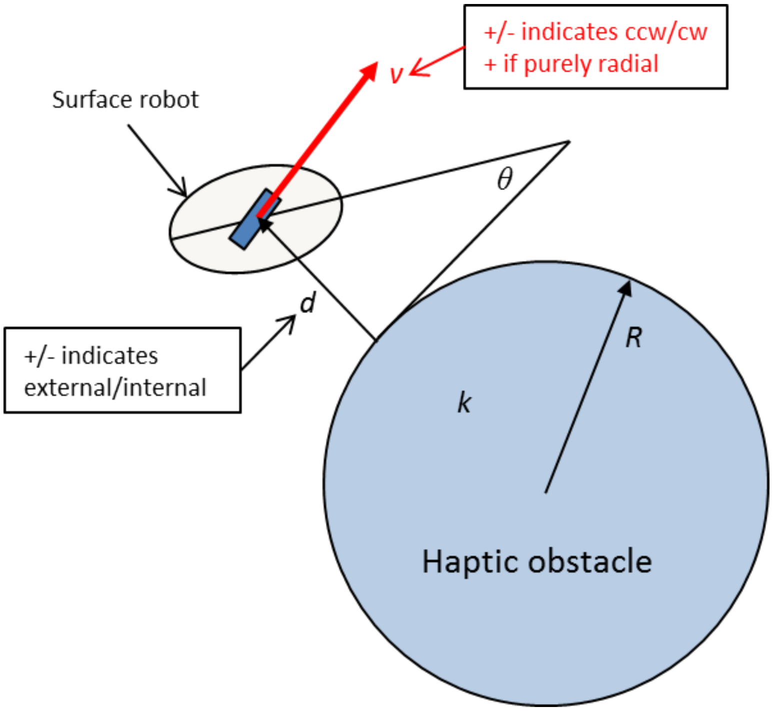


Figure 21. Haptic object reference frame variables with sign conventions. *k* refers to a set of impedance properties discussed in later sections.

The sign conventions are used to minimize the amount of information bits that need to be transferred, which results in a higher control bandwidth. For example, the velocity variable is a scalar magnitude of the robot velocity vector, but for control purposes it is important to know the direction parallel to an object surface that the robot is traveling. This is a single bit of information ("forward" vs. "backward", or "clockwise" vs "counter-clockwise"), so it is reflected in the sign of the scalar quantity. This sign is obtained by performing a cross product on the angular position vector of the robot and its velocity vector, but the vector result of this cross product is not needed, so it is discarded and its sign is attached to the velocity magnitude. Other sign conventions are set to create a counter-clockwise contribution to motor control when positive. In this example, a circular object is used as a reference frame, but these same conventions apply for a curved path with a continually changing radius. Objects described by straight lines have a radius of infinity, and the velocity sign convention reverts to positive with a positive y-component, and negative with a negative y-component. If velocity in y is 0, velocity is positive if traveling in the positive x-direction, and negative in the negative x-direction.

The result is that the haptic environment can stored within the touchscreen due to its superior memory space and its relative usability, and the surface robot will respond to its nearest haptic object. In practice, the Playsurface test platform contains a circular obstacle with parameters that are adjustable with visual sliders. The obstacle can be moved in 2D space, resized, and have its stiffness adjusted. Additional obstacles and shapes are trivial to the device itself, as shown by its ability to control under multiple constraints and straight line geometries in addition to circular paths, demonstrated in later sections. Implementing a fully customizable “haptic sandbox” application is entirely a touchscreen software development project at this point – the surface robot information handling and control would remain unchanged.

Using this coordinate system means that the position predictor must be expressed in terms of the new position variables. This is accomplished with a transformation described by Equations 9-12.

$$T = vt\cos\theta_{wheel} \tag{9}$$

$$N = d + R + vt\sin\theta_{wheel} \tag{10}$$

$$d_{new} = \sqrt{N^2 + T^2} - R \tag{11}$$

$$\theta_{new} = \theta + \tan^{-1}\frac{T}{N} \tag{12}$$

θ_{wheel} is the tangency of the robot wheel to the path, found by subtracting the device-detected steering angle from the body orientation angle. Subtraction is performed rather than addition to maintain the convention that a positive θ_{wheel} contributes to a counter-clockwise motor turn during path control. This effectively creates a temporary coordinate system with tangent and normal components relative to the point on the path the robot is currently radially aligned with. T and N refer to the linearly predicted position of the robot in the tangent and normal axes, respectively. Displacement from the boundary is then calculated by finding the magnitude of this vector and subtracting from the boundary radius. Updated body tangency is then found by calculating the angle swept out from the path center from the initial to the updated position, and adding it to the initial body tangency. This assumes again that the change in body orientation relative to the touchscreen can be neglected between samples. The geometry of these variables is illustrated in Figure 22.

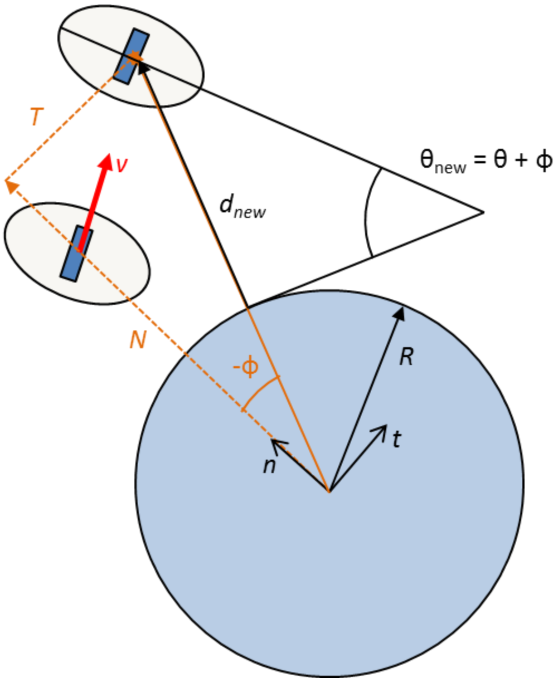


Figure 22. Linear predictor variables in an object reference frame.

This problem is simpler when dealing with objects defined by straight lines. Body orientation does not change in this case. Updated displacement is found using Equation 13.

$$d_{new} = d + vt\sin\theta_{wheel} \tag{13}$$

CHAPTER V

CONTROL

Control of the surface robot expands upon two fundamental control modes established by Colgate *et al*. for the Cobot unicycle: Free mode and path-follow mode [30]. These controllers require modifications to function on the surface robot due to differences in the device design. The primary difference is that this prototype does not have its rotational degree of freedom fixed. While it does calculate the wheel angle with respect to the touchscreen reference frame, a wheel angle disturbance can be introduced by the user, which is not true of the original Cobot, and requires that gains corresponding to wheel angle error be kept relatively low to avoid instability. The surface robot can also travel off of the touchscreen, which causes it to lose its taskspace reference frame and velocity information. The Cobot, in contrast, is physically assembled into its reference frame. An additional control mode has been added to account for this occurrence and allow the robot to function in a limited capacity until contact can be re-established.

Control has also been developed to give constraints defined with path-follow control compliant behavior. This behavior simulates the dynamics of an adjustable mass-spring-damper system. Compliant constraints are not adequately explored for steering-based haptic devices, and come with a number of challenges inherent to the device design. The presented approach overcomes these by applying path-control mode as a position controller inside an admittance control hierarchy. Compliant constraint control is described in detail after the fundamental Cobot controllers and their variations are discussed, beginning with free mode.

A final, important difference is that the surface robot is not fixed to the touchscreen surface and friction between the wheel and the touchscreen is provided only by gravity and the user pushing downward. Every constraint that the surface robot can render can be circumvented by the user by simply lifting the device or sliding the wheel against its rolling direction. These control modes require user participation in maintaining the no-slip condition to function.

O. Free Mode

Free mode is used to control the surface robot so that steers to allow the user to move it along a 2D surface unimpeded. The simplest control approach for this task is a standard position controller, set to align the wheel with the direction of user force. However, due to the free-spinning nature of the wheel, force parallel to the wheel does not statically deflect the stage in that direction, but rather translates into motion of the entire robot. Therefore, the force detected by the sensor is dominated by the component of force perpendicular to the wheel. Using position control in this case results in a constant error near 90 degrees, which the controller can never correct for regardless of how the wheel turns until the force magnitude itself falls to zero. This principle is the foundation of the free mode control approach presented.

The most basic version of this control mode responds only to force input by the user. The device detects a force F from the user, and steers until that force falls below a threshold. The rule used for calculating the steering speed ω is described by Equation 14.

$$\omega = GF \tag{14}$$

G in this case is an arbitrary gain constant used to tune the proportional relationship between force and steering velocity. The commanded steering velocity approaches zero as the wheel approaches alignment with the direction of the user's input force, producing smooth motion and allowing for small adjustments in path. Steering velocity is controlled by a standard PD motor velocity controller.

A shortcoming of Equation 14 is that it does not adequately address selection of the steering direction. Since the wheel is free-spinning, it does not matter whether it faces forward or backward. As a result, the largest angle that the wheel could need to rotate in a single adjustment is 90 degrees. For example, there is no need to turn clockwise 150 degrees to accommodate a near reversal in direction when it can turn 30 degrees counter-clockwise and reach the same result more quickly. This issue is addressed with algebraic transforms, but as previously discussed, the sensor primarily sees a force 90 degrees from the wheel angle. This second issue is more problematic, as neither direction offers a shorter path for a 90 degree rotation. The direction choice is critical if the robot is moving. The only context that allows this decision to be made correctly is the direction of motion, which the surface robot cannot determine on its own. Therefore, a "pseudo-free mode" in which the wheel is always assumed to be spinning in one direction is used when contact with the touchscreen is lost. This results in a usable free mode as long as the user moves the robot in smooth curves without reversing direction. True free mode using velocity information is resumed when communication with the touchscreen is re-established.

P. Particle Mode

An expansion on free mode is "particle mode", which has much of the same functionality as free mode, but simulates the robot as a frictionless particle of some arbitrary mass in the perpendicular-to-path direction. This is a concept known as inertia masking, explored by the Cobot project and their unicycle [31]. The control scheme is expressed in Equation 15:

$$\omega = \frac{F_{\perp}}{uM} \tag{15}$$

The apparent mass of the robot in the direction parallel to the wheel direction cannot be controlled, but the apparent mass of the robot perpendicular to the wheel can be controlled by adjusting the steering speed. This dictates the centripetal acceleration and simulates normal-to-path inertia of a prescribed mass. This effect is limited by the maximum speed of the motor as velocity approaches zero. To handle this singularity, a threshold for u is set, below which the controller operates as if $u$were at the threshold value. The mass of the robot is low enough that simulating a lighter object than itself is difficult to control, as it only becomes noticeable when forcing it into tight curves at high speeds. However, the effect of moving an object that has low inertia in its direction of motion but high inertia to having that direction changed is achievable.

Q. Path-Follow Mode

Path-follow control is implemented directly from Cobot literature, in which the device is programmed to asymptotically approach a defined curve, regardless of the user force input. It does this by correcting for two factors: wheel tangency error and displacement error. When displacement from the desired path is large, the displacement term dominates and the wheel steers toward the path. When displacement is small, the tangency term dominates and the wheel steers to be tangent to the path. This control approach assumes a constant translational velocity u in order to set the instantaneous curvature of the planned approach, and controls the wheel's steering speed rather than its steering angle in order to normalize it against any arbitrary translation velocity. The commanded steering velocity is defined by Equation 16.

$$\omega = u(K_d d_{\mathrm{err}} + K_\theta \theta_{\mathrm{err}} + \kappa_{\mathrm{path}}) \tag{16}$$

The control variables are illustrated in Figure 23.

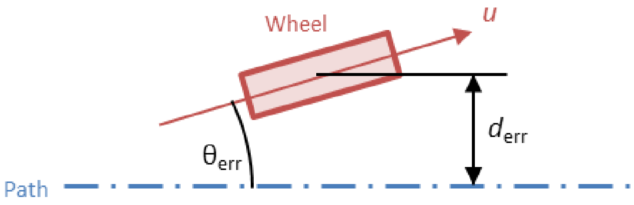


Figure 23. Path-follow control variables. θ_{err}, is the tangency error, and d_{err} is the displacement error. *u* is the scalar translation speed.

This can be expressed in terms of commanded curvature by dividing the translation speed, u, from both sides.

In practice, the displacement term acts like a proportional error term in a position controller, and the tangency term acts like a damping term. A large tangency error constant can introduce steady state error to the path follower, dependent on the dead zone of the motor used. It can also cause instability around zero-displacement, as the controller is fully tangency dominated and can produce large reactions to small angle changes. This condition is almost guaranteed by a user moving the robot along a path, as no user will hold the robot orientation perfectly fixed with respect to the touchscreen reference frame. The path following algorithm also becomes unstable in the presence of displacement errors greater than can be offset by an opposing tangency error. The tangency gain must therefore be carefully selected between these bounds, with its lower limit dictated by the displacement gain. Translation velocity also contributes to instability, with higher path tracking performance at low speeds. This effect is driven by the position update rate that the touchscreen can provide, and can be offset with lower control gains. The controller overall performs with a faster response and steady state error rejection at high gains, but with higher robustness to instability at low gains.

R. Haptic Wall

Free mode and path-follow control can be implemented together to display a wall constraint. If the user penetrates an assigned boundary, the robot switches to path-follow mode to correct and constrain the robot to the boundary path. If the user exerts a force away from the wall while in path-follow mode, free mode resumes and allows the user to pull away. In practice, this setup allows initial penetration of the wall, dependent on how fast the user is moving and the path control gains, and can fail if the user overshoots the wall too quickly for the controller to recover. The “wall approach” and “emergency” control modes have been implemented to mitigate this effect.

1) Wall Approach Mode

The controller anticipates a wall collision for a straight wall by estimating the time to hit the wall based on its current velocity. This allows for a spatial threshold based on the estimated time to collision. Crossing this threshold puts the robot into wall approach mode, in which it steers along a circular path tangent to both the wall and the wheel on the robot. The parameters and geometry of this approach are illustrated in Figure 24.

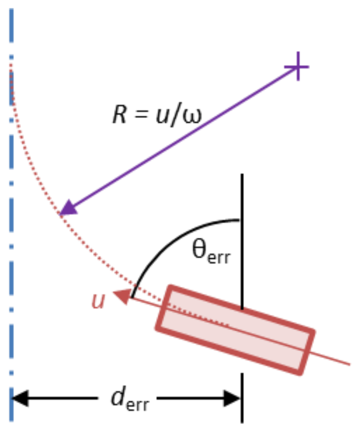


Figure 24. Wall approach control variables. The steering velocity is controlled to move the wheel through the dotted path, described by radius R.

A similar technique is used by the Cobot project, where this circular path is enforced using the path-follow controller [30]. This is implemented as a feedforward steering velocity controller, but could also be implemented as a separate circular path constraint. This controller is described by Equation 17:

$$\omega = \frac{u}{d_{\text{err}}}(1 - \cos\theta_{\text{err}}) \tag{17}$$

where θ_{err} and d_{err} are the same parameters used in the path control algorithm.

2) Fault Mode

If the robot penetrates a haptic wall or path constraint beyond the range with which the path-follow controller can recover, it locks the wheel to be parallel with the wall. This commanded steering position updates with the orientation information supplied by the touchscreen, so the wheel maintains its angle relative to the touchscreen regardless of the rotation of the device. It directs the user directly back toward the boundary when it detects a user force toward the constraint path. This mode is essentially a virtual hard-stop. This is not a true hard-stop, however, due to the non-rigid nature of all constraints the device can present.

S. Compliant Path Control

Creating compliant boundaries with a device of this design is difficult due to the dynamics of the wheel. It allows completely unimpeded motion in one direction while blocking it in all others. Steering angle can be adjusted to control the orthogonal components of the overall motion, but the magnitude of the displacement or velocity is determined by the user. The device cannot, for example, slow a user as they penetrate a wall without deflecting them to the side. It cannot remove energy from the system except through small losses while redirecting it. Two methods have been developed to work around this limitation: 1) Elastic kinetics with inertia masking and 2) Path-follow admittance control.

1) Elastic Kinetics with Inertia Masking

One approach to this problem is to control only the motion perpendicular to the wheel direction. This is similar to the inertia masking principle used in particle mode – variable inertia to changing the direction of the velocity vector can be simulated, but inertia to changing its magnitude is not controllable. This approach results in an attempt to simulate the centripetal acceleration caused by the deflected wall reaction force on the surface robot. The control parameters used for this type of control are illustrated in Figure 25.

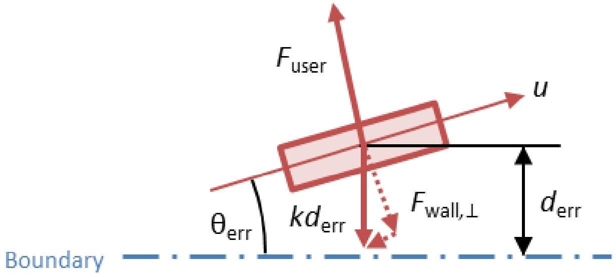


Figure 25. Elastic kinetics with inertia masking control variables. θ_{err}, d_{err}, and u retain the same definitions as in path-follow control. k is the simulated boundary spring constant.

For the sake of this example, the boundary is assumed to be a straight line. The user force perpendicular to the wheel and the wheel-perpendicular component of the simulated wall force are combined to obtain a net force, $F_{user} - F_{wall,\perp}$. This force is equal to the centripetal force that dictates the radius of the robot path, which dictates the centripetal acceleration together with the mass. As in particle mode, this mass is artificial and can be selected to produce a desired behavior. From the kinematics of a particle in motion on a circular path, this acceleration is equal to the angular velocity multiplied by the translation speed. Solving for angular velocity results in a controller based on a commanded steering speed, described by Equation 18.

$$\omega_c = \frac{F_{\text{user}} - k d_{\text{err}} \cos \theta_{\text{err}}}{m u} \tag{18}$$

This approach contains fundamental problems that make it unsuitable for simulating a boundary on its own. First, it cannot correct for a direct approach into the boundary, where the elastic wall force has a zero projection normal to the wheel. In other words, the cosine term falls to 0, eliminating the contribution of the wall to the steering response. Theoretically, the user could push the robot head-on into the wall and meet little to no resistance as long as the trajectory remained nearly perpendicular to the wall boundary. This is because in this scenario, the majority of the elastic force acting against a penetrating particle contributes to a change in velocity magnitude, not direction, which is the component of acceleration that this method is based on neglecting. Second, should the user be heading out of a compliant obstacle, this method depends on the user voluntarily pulling the robot toward the obstacle boundary once the wheel aligns directly outward. This is because, once again, the cosine term becomes 0 when the wheel is normal to the obstacle boundary. The user can just as easily reverse direction to move further into the obstacle, and the controller will not react due to the trajectory remaining normal to the boundary.

This can be addressed by implementing a state controller, where the inertia masked kinetics simulation runs while the robot velocity vector is pointed out toward the boundary, and a separate controller controls the wheel angle to obstruct penetration of the obstacle when the velocity vector is pointed away from the boundary. This controller could take several forms, but for example, the wheel angle could be treated as a valve, where user force below the elastic reaction force acts causes a “closed” condition (the wheel is locked parallel to the boundary), and increasing user force increases the amount by which the valve “opens” (the wheel orients to be more normal to the boundary). This control scheme can be saturated by the wheel orienting normal to the boundary, or assuming a “full open” condition, and so the net force must be normalized against a saturation force. The obstacle penetration component of allowed motion is equal to $\sin\theta$, so for a linear correlation of allowed penetration to net force, the controller is described by Equation 19.

$$\theta = \sin^{-1}\left(\frac{F_{user}\cos\theta) - kp}{F_{max}}\right) \tag{19}$$

This method has implementation problems as well, namely that this controller has no way to correct for penetration errors. The penetration term controls the amount of user force required to "open the valve", but it cannot move back toward the equilibrium position without engaging the inertia masked controller, which has been shown to be unable to enforce a position against user force. Both controllers only consider instantaneous net force – neither one adjusts for accumulated error. This error can be introduced by the finite rotation speed of the motor, delay between samples, or wheel slippage. Also, a state controller between a position-controlled state and a velocity-controlled state is liable to have undesirable effects while transitioning between the two states. For example, this control scheme breaks down entirely if a user rocks the device into the obstacle periodically, with zero force applied in between impulses. This would engage the position controller, allow some displacement into the obstacle by enforcing a nonzero θ, then transition to the velocity controller, which would command a zero steering velocity due to the absence of translational velocity.

An exhaustive search for methods of implementing compliant boundaries has not been completed, and it is possible that a scheme exists which adequately corrects for error accumulation and transition effects while using an inertia masking kinetics approach. However, this problem based around compensating for a fundamental flaw of using inertia masking kinetics for simulating resistive reaction forces – it cannot simulate forces that primarily remove energy from a system. A separate control scheme is required to simulate this effect, implemented with logical conditional statements that quickly grow complex and difficult to manage, with edge effects that can render the controller completely unable to simulate the desired behavior. For these reasons, an alternative approach was developed, based on admittance control in robotic joints.

2) Path-follow admittance control

Admittance control is used in robotics to simulate systems with mechanical impedance, often characterized as inertia, damping, and spring stiffness. The system detects force input by the user, and outputs displacement of the end-effector according to the behavior of a system with those properties. Admittance control is the inverse of impedance control, which detects displacement as defined by the user manipulating an end-effector, and outputs force or torque to simulate the impedance a system with the selected mass, damping, and stiffness would offer. Both versions of this form of simulating spatial constraints are used extensively by upper limb stroke rehabilitation robots. The surface robot design is naturally conducive to admittance control, as it can sense user force, and outputs displacement in the form of allowed movement direction. Particle mode, for example, could be considered a form of admittance control that is limited to simulating inertia.

The following approach is based around the idea that absolute position of the surface robot can be explicitly dictated with the existing path-follow controller, as long that position can be expressed in terms of a two-dimensional path. It uses this position as the output to an admittance controller, which uses net force as the input.

This definition of position holds true when simulating compliant paths if the deformed path as defined by the user force and path impedance variables is used as the commanded position. This also holds true for compliant 2D shapes if the deformed perimeter is defined as a path constraint. For example, on a circular path of radius R and stiffness k, a user pushes the robot toward the center of the path with force F. Assuming that the system has settled to its steady state, the robot should have penetrated F/k units inward. As long as this force is maintained toward the center of the circle, any position along a circle of radius R – F/k is correct. The controller to simulate a compliant circular path is then a controller which takes user force as input, and outputs a radius to the path-follow controller. This scenario is illustrated in Figure 26.

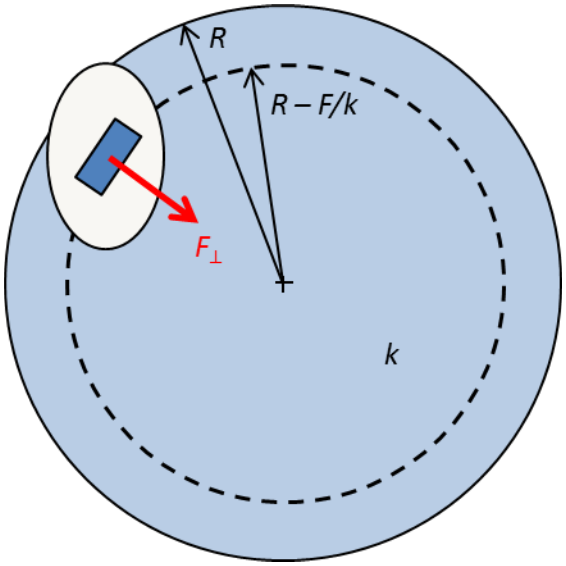


Figure 26. Constraint deflection as a 2D path constraint. The dotted path represents the deformed object perimeter as defined by its impedance properties and the user input force.

To generalize, the output of this controller is commanded path deflection. This allows the controller to apply to linear paths, where the whole path can be displaced in one direction, and to curved paths of continually varying radius, where the radius of each point on the path is increased or decreased by the commanded deflection amount. This controller emulates a massless spring system, as illustrated by Figure 27.

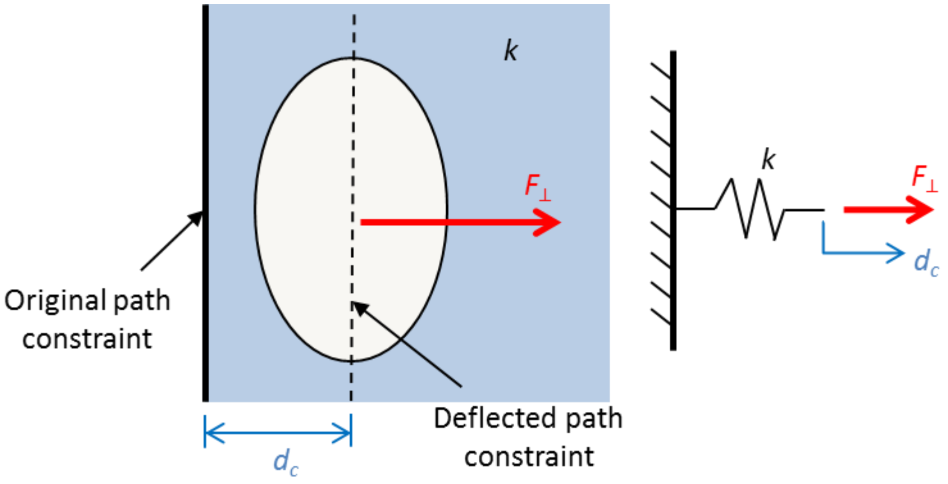


Figure 27. A commanded path deflection admittance controller as a massless spring system. Displacement of the system as created by a force input is rendered as a deflected path constraint for the surface robot.

Commanded deflection is represented by d_c, and $F_\perp$ represents the component of user force perpendicular to the boundary. This controller can be expanded to include inertial and damping properties to simulate a mass-spring-damper system. The commanded deflection then behaves as the system shown in Figure 28.

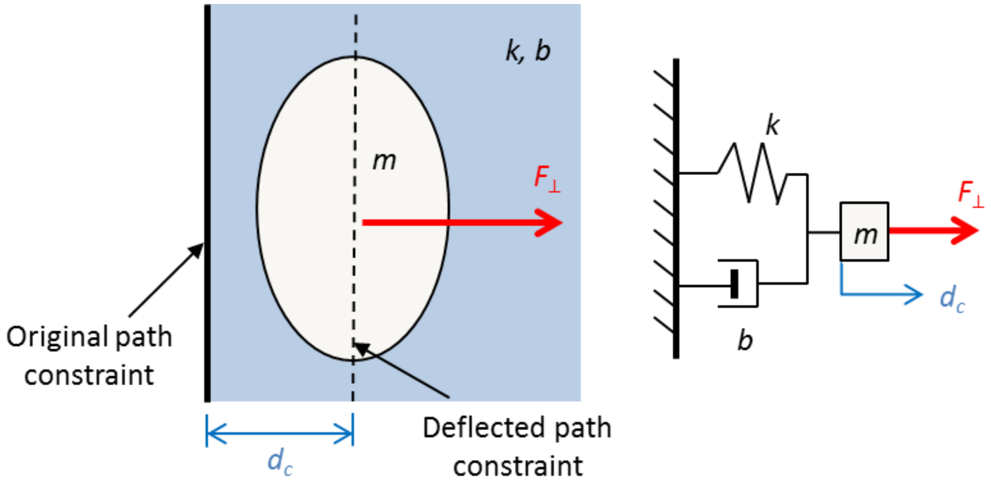


Figure 28. A commanded path deflection admittance controller as a mass-spring-damper system.

This is done by solving for the acceleration of an ideal mass-spring-damper system, and assuming this value remains constant between controller samples. The velocity of the deflecting boundary is calculated with a first order Euler approximation using this value, and the deflection of the boundary is calculated in the same way using the velocity value. This model is described by equations 20 and 21.

$$d_{c,\text{new}} = d_c + \dot{d_c}t \tag{20}$$

$$\dot{d}_{c,\text{new}} = \dot{d}_c + \frac{F_\perp - (kd_c + b\dot{d}_c)}{m}t \tag{21}$$

The commanded deflection and its rate of change are calculated with each controller sample based on the input force measurement. This model increases in fidelity as the sample rate increases, and the prototype controller rate of 667 Hz produces a smooth, accurate response. Desired deflection responses to a step force input with varied mass-spring-damper parameters are plotted in Figure 29.

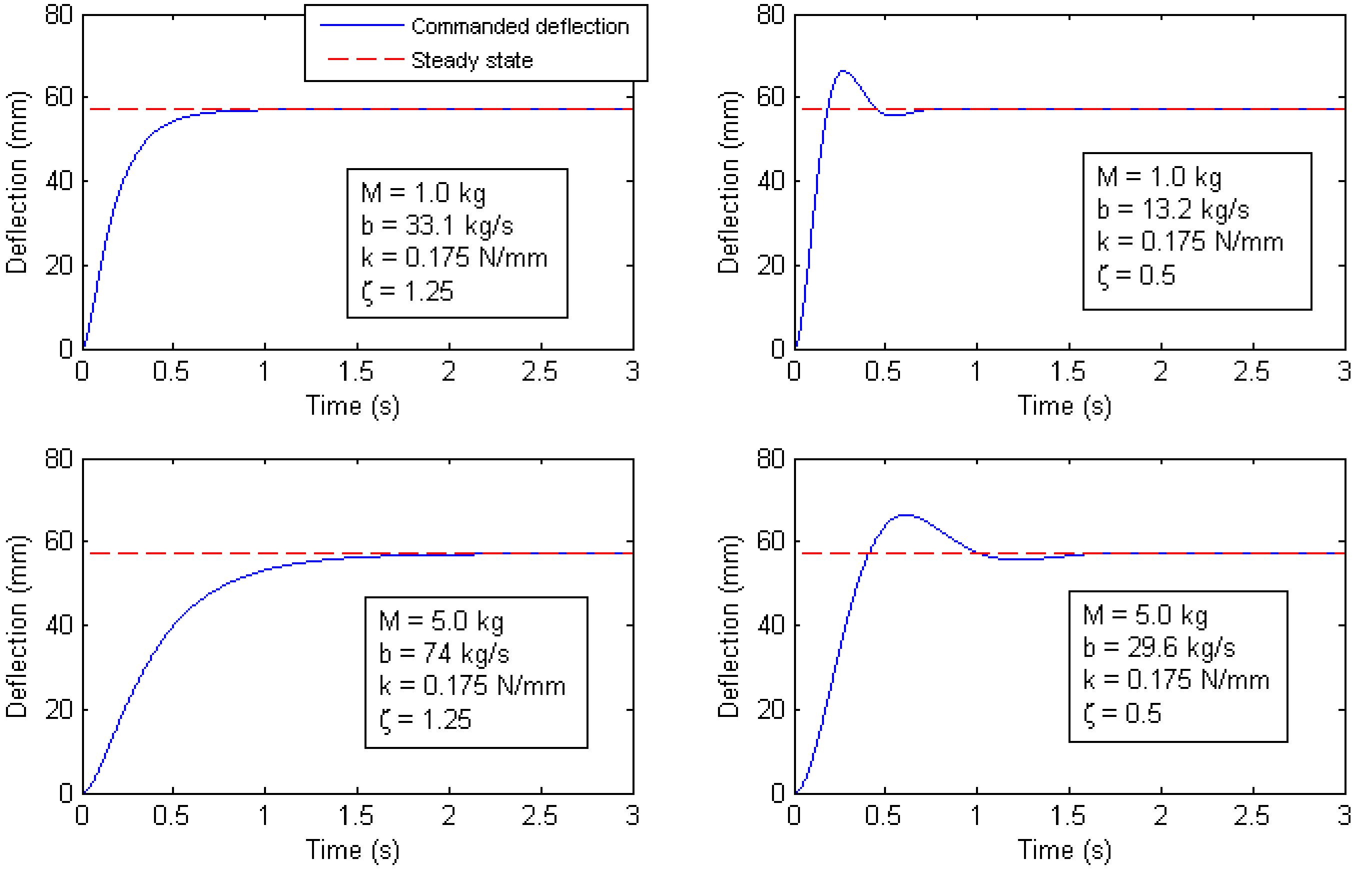


Figure 29. Step response to a 10N perpendicular force input. The spring constant is held static because its primary effect is to change the steady state deflection value. The effect on the damping ratio is controlled by varying b with changing mass.

The effect of this control scheme is that compliant constraints can be simulated with three customizable parameters. The spring element controls the steady state deflection of the constraint, the damping element controls the speed with which the constraint deflects and suppresses oscillation, and the inertial element controls the response to changes in user force as well as the rise time. The inertial element also functions as an effective low-pass filter to the force input signal, rejecting high frequency noise as well as attenuating low frequency periodic inputs. Both the damping and inertial elements contribute to phase lag for periodic inputs. These effects are seen when plotting the commanded deflection for a sine force input, as shown in Figure 30.

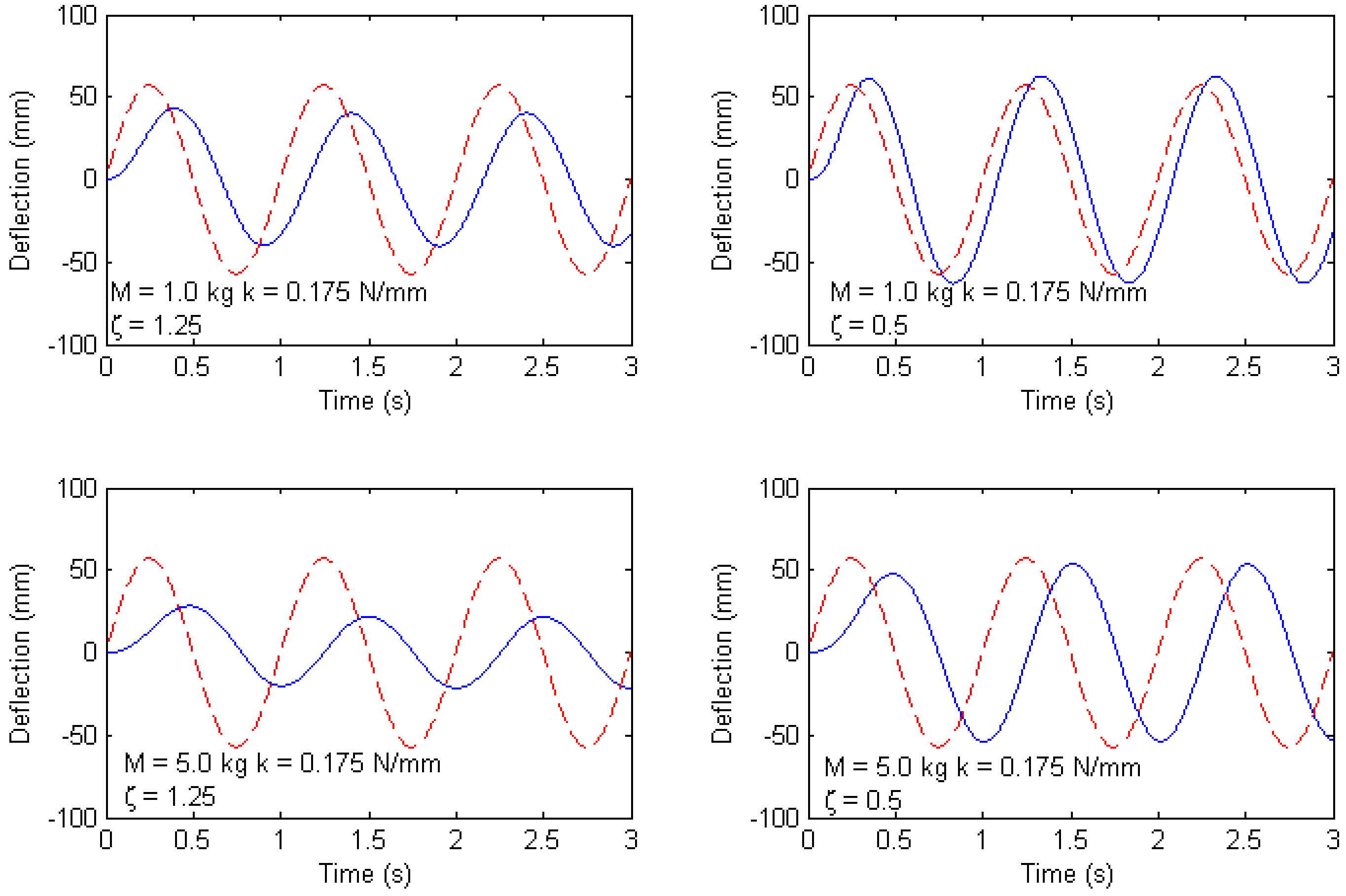


Figure 30. Tracking a sine force input. The dashed reference line is the response of an undamped, massless spring.

This approach has several advantages to the inertia masked kinetics approach. A single controller can simulate the dynamics of a compliant system while robot is pushing into a boundary and being pushed out. The problem of being unable to simulate force in the wheel direction is removed by implementing the path-follow controller, which is designed to asymptotically approach the prescribed path. Also, an explicit desired deflection can be defined, allowing the path-follow controller to correct for errors. A drawback of this approach is that it is dependent on the time constant of the path-follow controller to enforce the desired deflection. Simulating a collision with a highly responsive, underdamped surface can cause the path to deflect more rapidly than path-follow control can enforce the deflection. This is one situation in which directly controlling the steering velocity to simulate particle kinetics could see higher performance, if the other problems with the approach were solved. Another important limitation is that, like every other controller for this device, it must be in motion to have any effect. Therefore it is also possible for the user to be moving the surface robot too slowly for it to "catch up" to its desired deflection, or to be exerting a force normal to the boundary and experience no deflection because the device is stationary.

CHAPTER VI

EVALUATION

The surface robot prototype has been evaluated against its design specifications. These results are presented in two categories – mechanical performance and control performance.

T. Mechanical Performance – Methods

The physical performance characteristics of the prototype are somewhat variable, as they are subject to the amount of vertical force that the user exerts on it. This effect was normalized by establishing a downward force of 20.0 N as the standard by which all metrics were evaluated. This load is sufficient to push the bottom of the outer shell to maximum compression of its felt pads, at which point the wheel is under its maximum vertical preload as determined by the flexure stiffness, regardless of external load.

1) Friction Force Magnitude

Friction force was measured with a handheld 500 N Mark-10 Model M5-100 single-axis force gauge. The wheel surface and the touch table were cleaned and allowed to dry before the test. The prototype was loaded with the standard weight on the touch table surface, and the force gauge sensor tip was fitted with a hook attachment. A fixture with eyelets was fitted around the base of the prototype and secured to the prototype frame with set screws. The force gauge was hooked into an eyelet and the prototype wheel position was locked perpendicular to the pull direction. The force gauge was gradually laterally loaded until the prototype slipped, and the peak value was recorded. The force gauge was made to rest against the touch table surface during testing to ensure that the measured friction force was entirely horizontal. The recorded values represent the mean of 10 measurements, with uncertainty representing two standard deviations.

2) Force Sensor Accuracy

Force sensor accuracy was measured by using the same test conditions. The force gauge was attached to eyelets extending at angles of 0, 60, 90, 150, 195, 255, and 315 degrees from the width axis and about the steering axis. The wheel was locked perpendicular to the pull direction for each angle. The force gauge was manually loaded away from the prototype and a force vector value was recorded simultaneously by the force gauge and the prototype. This procedure was repeated three times at each angle. In addition, 7 zero-force values were recorded. The weight was removed between each pull test to allow any residual deflection maintained by friction between the outer shell and the touch table to resolve back to zero. The experimental setup used for both of these tests is shown in Figure 31.

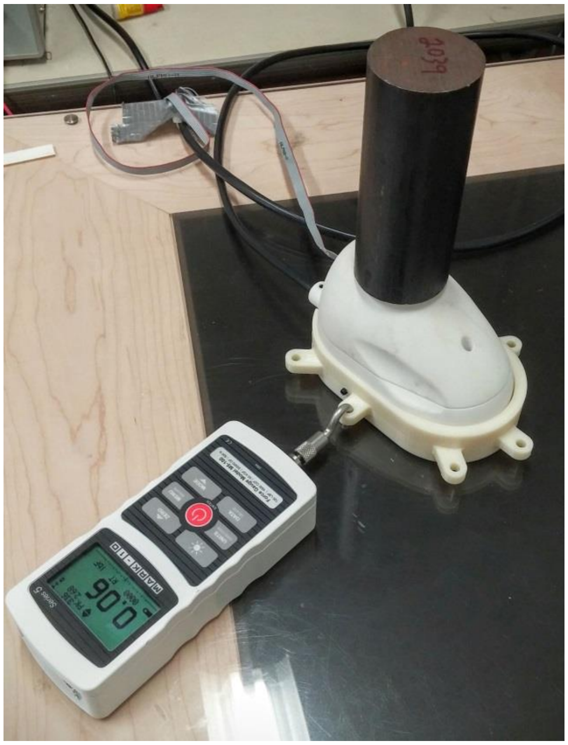


Figure 31. Friction and force sensor validation test setup. The prototype is shown in its test fixture and under its 20 N load weight.

3) Mechanical Output Bandwidth

Mechanical output bandwidth was evaluated by using a position controller to track a sine wave with a peak-to-peak magnitude of 90 degrees – the largest instantaneous position change the motor will be commanded to make. Position data from the onboard angular position sensor was sent via serial connection to a PC during the test. This procedure was performed for sine frequencies from 1 Hz increasing as powers of 2. When the RMS amplitude of the wheel position compared to the RMS amplitude of the ideal sine wave fell below the cutoff amplitude of -3dB, the test was repeated in a binary search until amplitude equal to the cutoff amplitude was found to 0.1 Hz precision. Each trial ran for at least 8 periods with a data sample frequency of 667 Hz.

4) Prototype Dimensions

Prototype dimensions are extracted from the CAD models. Dimensional tolerances introduced by the SLS additive manufacturing process are assumed to be functionally negligible for these metrics.

5) Structural Safety Factor

The structural safety factor was found by simulating the maximum deflection allowed by the prototype frame hard-stops on the simplified flexure array finite element model. The flexures are identified as the critical structural failure point of the design due to their high expected deflection and function in absorbing user force. This model was constrained to have the maximum vertical deflection of the inner stage relative to the outer stage, with the inner stage rigidly constrained against motion in the horizontal plane. The outer stage was constrained to allow only horizontal motion. The flexures were idealized as shells. This analysis was conducted along the long and short axes, as well as at 45 degrees between them. Safety factor represents the maximum von Mises stress calculated for the model compared to the failure strength of the material (50 MPa).

6) Estimated Cost

Estimated cost is the summed price of all prototype components. This is a conservative estimate, as it does not take quantity scaling into account. The cost of the 3D printed components was estimated by exporting their 3D models to Shapeways (www.shapeways.com) – a commercial additive manufacturing company that 3D prints user-uploaded models for them to buy or sell – and reporting the quoted price.

Shapeways offers a service to print parts on a Formiga P100, which is a nearly identical model to the machine that this prototype was fabricated with. This is a realistic approximation of the manufacturing strategy that would be used to produce a quantity of this prototype to sell to users. The total cost to buy all 3D printed parts from Shapeways was added into the cost of the purchased prototype components for the full cost estimate. The cost of labor was not included.

U. Mechanical Performance - Results

Performance metrics are compared with their design specifications in Table 4.

Table 4. Robot mechanical evaluation

	Marginal target	Prototype value	Ideal target
Friction force magnitude (N)	10	10.6 ± 1.0	20
Force sensor mean absolute error (N)	1.50	1.07	0.10
Mechanical output bandwidth (Hz)	4.0	7.0	8.0
Device length (mm)	200	135	100
Device width (mm)	80	63	40
Device height (mm)	100	77.3	40
Structural safety factor	1.2	1.2	2.0
Estimated cost (USD)	1000	439.30	100

This section discusses each performance metric for the final prototype, and compares with the previous iteration when applicable. This comparison is made in order to illustrate the tradeoffs between some metrics when design elements are changed, such as installing a more powerful motor.

1) Friction Force Magnitude

Friction force magnitude is near its marginal level, but is still acceptable for whole arm kinesthetic applications. Interestingly, this value is lower than the friction force supported by the previous prototype iteration (13.7 ± 1.2 N) despite the later iteration being designed with a larger vertical flexure deflection before the outer shell bottoms out against the touchscreen surface. This change was made to increase the amount of friction force rendered by increasing the flexure preload against the wheel. This increase in friction did not happen, because the vertical displacement was decreased in practice due to changes in the felt covering the bottom of the device. First, there is a larger amount of felt covering the bottom surface of the prototype in the later iteration. Felt is used to decrease the friction and prevent abrasion between the outer shell and the touchscreen surface. However, it also absorbs vertical load similar to an array of parallel springs. Increasing the felt surface area effectively increases the number of springs in parallel, stiffening the response and reducing vertical deflection. Also, this system has been further stiffened by using a laser cutter to create felt shapes to fit the corresponding tabs exactly, which has fused the edges of the felt cutouts into a more rigid perimeter.

2) Force Sensor Accuracy

Force sensor accuracy is reported as mean absolute error. This is the mean of the absolute value of the error of each data point. The value reported in the table is the norm of the error in x and in y. Force in the x-direction was found to have a mean absolute error of 0.77 N, and 0.74 N in the y-direction. This is an increase of 0.54 N and 0.62 N in x and y, respectively, from the uncertainty calculated for the ideal model. Actual force data versus embedded force sensor data is plotted in Figure 32.

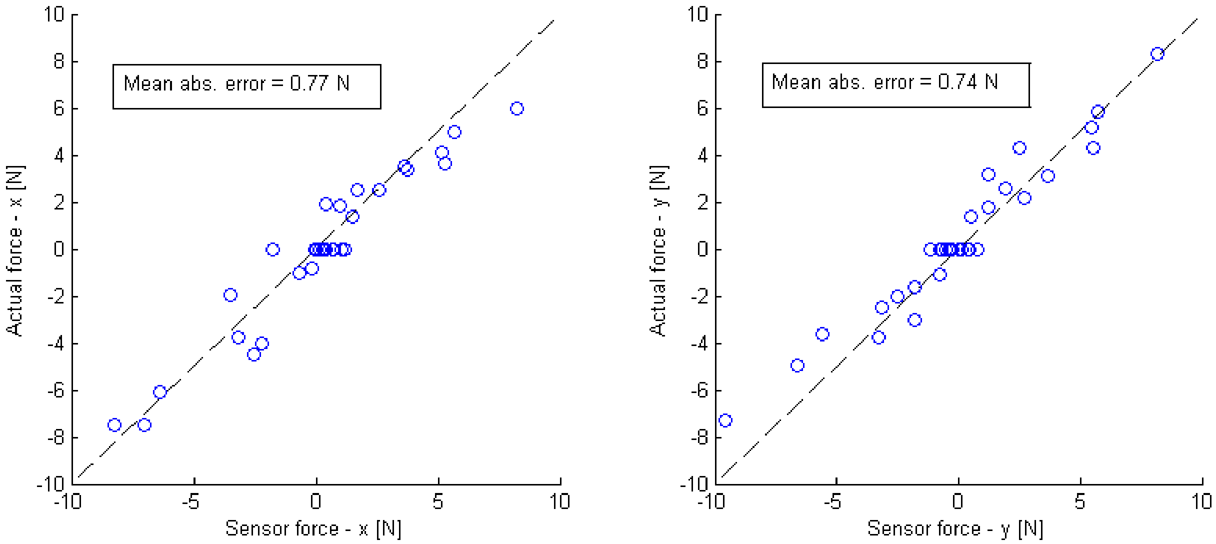


Figure 32. Actual vs. prototype sensed x and y force components.

Another way of looking at this is to test the accuracy of force magnitude and direction. Zero-force data points were discarded for the angle error analysis, because angle measurements have no meaning for a zero force magnitude. These analyses are plotted in Figure 33.

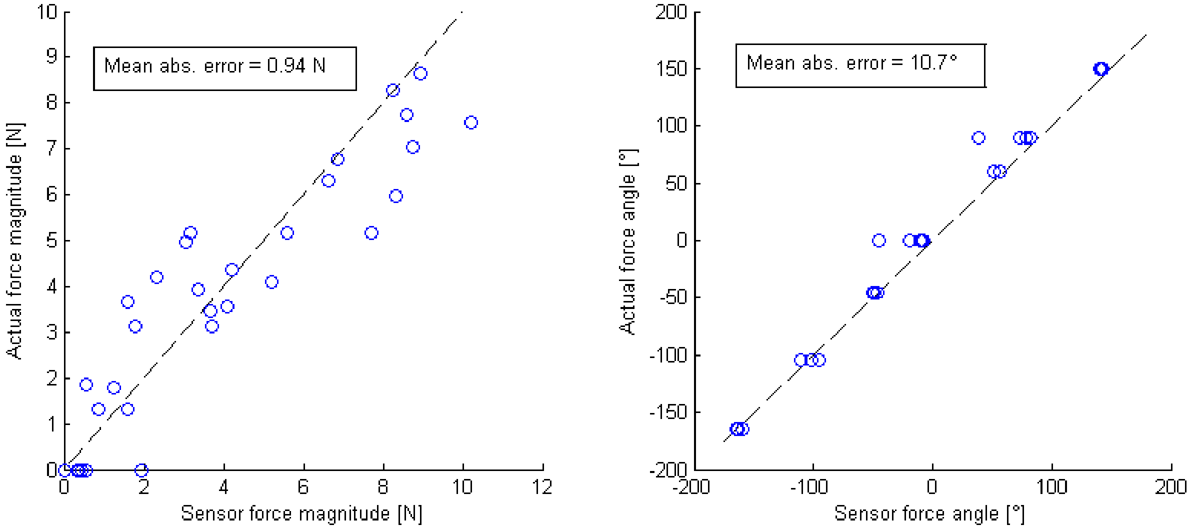


Figure 33. Actual vs. prototype sensed force magnitude and angle.

Imperfections in the force sensor design become apparent when analyzing this data. Force data quality, especially force angle measurements, drops off significantly below force magnitudes of approximately 3 N. When only considering data points with force magnitude above 3 N, force angle error drops from 10.7 to 6.5 degrees. Friction between the outer shell and the touchscreen surface significantly interferes with the force sensor measurement for low-magnitude forces.

Another factor that causes error in this sensor is the assumption of strictly planar movement of the inner stage relative to the outer stage. This assumption holds less true for the current prototype than for the first prototype iteration. In the latest iteration, the flexure stiffness value used to calculate force had to be nearly doubled, and scaling factors had to be added to the x and y-displacement measurements separately in order to achieve a uniform and accurate force measurement response. A detailed force sensor accuracy test was not performed on the first prototype iteration, but realistic force data was being recorded without either of these steps.

The primary cause for this is a change in the wheel width, which was decreased in the second prototype iteration by nearly half (12.3 to 6.4 mm). This was done to decrease the width of the contact area of the wheel against the touchscreen surface, reducing the amount of friction torque the surface exerts on the wheel during steering. However, a side-effect of this is that the soft urethane wheel can more easily bend under user force, allowing the inner stage to tilt out-of-plane as it is deflected. This effect is larger in the x-direction, where the device frame is narrower and the opposing flexures are closer together. This effect accounts for the non-uniform apparent x and y-stiffness, as well as the overall detected stiffness magnitude change. The reduction in friction torque was achieved, but there is a significant tradeoff between wheel bending rigidity and motor output efficiency. An ideal design would contain a wheel with a narrow, high friction surface outside a rigid body.

Another potential error source for this force sensor is offset vertical loading. The prototype design does not contain a flat surface directly above the wheel, so all friction and force tests have been performed with the downward load shifted toward the rear of the device. Figure 34 illustrates the size of the gap caused by the felt pads as well as the slight backward tilt of the outer stage.

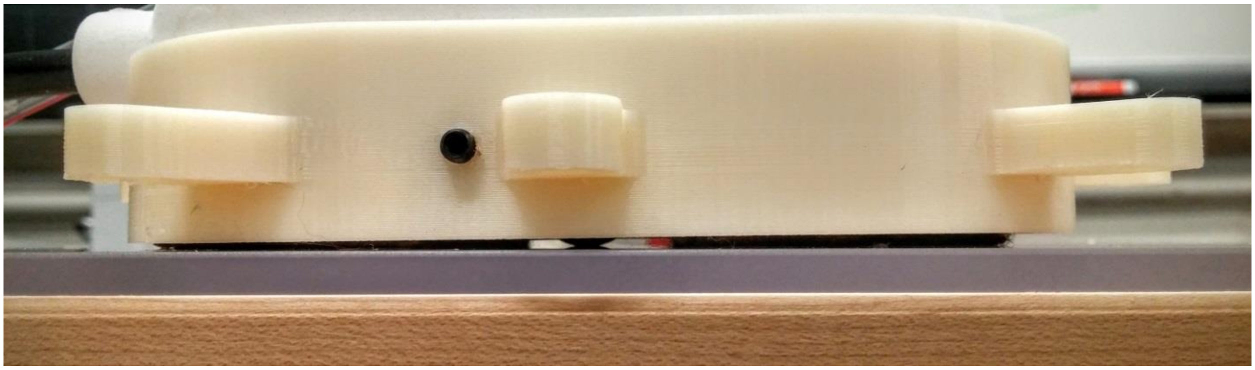


Figure 34. Gap between the bottom of the device and the touchscreen surface under the standard load. Note the increased compression to the left side of the figure.

3) Output Mechanical Bandwidth

The mechanical bandwidth of this prototype is well within range of the specifications. This is significantly improved from the initial prototype, which had a mechanical output bandwidth of 2.5 Hz. The sine-tracking frequency response for both prototypes is plotted in Figure 35.

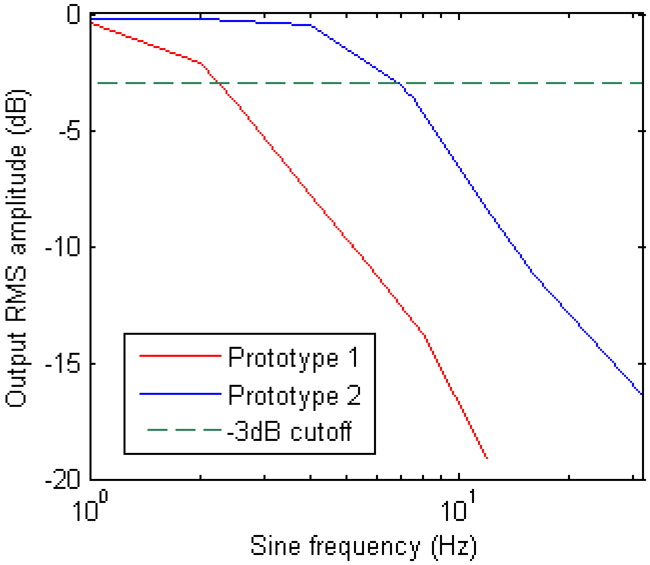


Figure 35. Frequency response for position tracking (90 degree peak-to-peak sine wave).

The updated prototype sine wave position tracking has not yet begun to deteriorate at frequencies where the original prototype was already falling below the cutoff amplitude. This specification was the primary driver behind the updated prototype iteration, because it is essential for delivering high fidelity force-feedback to a user, and the initial prototype was heavily motor-speed limited. This metric also drove the updated wheel width, which was designed to extract more speed from the motor by minimizing friction losses. This effect was achieved – the motor was sized to output a maximum speed of 215 rpm under the same load as the original prototype, and outputs a maximum speed of 250 rpm in the updated prototype.

4) Prototype dimensions

The length and width of the prototype are nearly identical to the length and width of the first iteration. These fall within specifications. Height is significantly increased with the latest prototype, however, due to the addition of a more powerful motor. This is still within specifications, but is approaching a value that significantly separates the user from the touchscreen. However, these dimensions do not account for the fact that the original prototype had all electronics with the exception of the two Hall effect sensors located on a separate breadboard. The current prototype has all of its electronics on a printed circuit board mounted inside the device. A comparison illustrating these differences is shown in Figure 36.

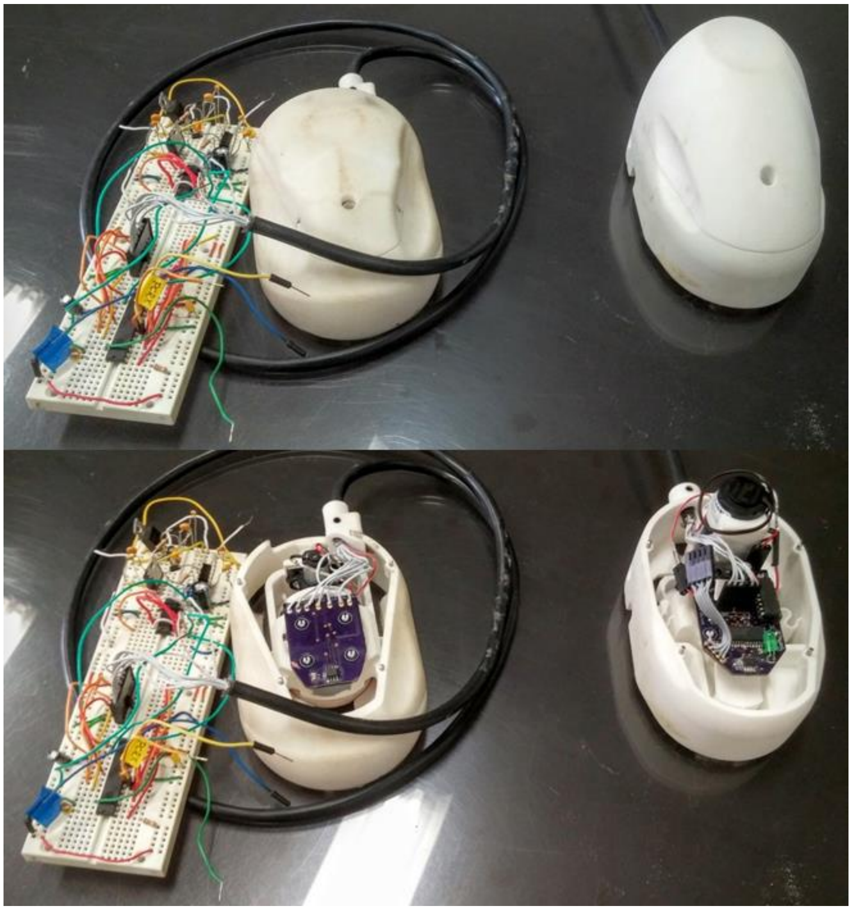


Figure 36. Comparison between the two prototype iterations. The first iteration is on the left with its outboard electronics shown, and the current iteration is on the right.

5) Structural Safety Factor

The maximum allowed flexure deflection results in a safety factor of 1.2, which is marginally within specifications. Under a slip load, which is the maximum load under normal operation, this safety factor increases to 3.2.

6) Estimated Cost

The total, small quantity cost of prototype components is $310.22. Shapeways quotes the printed components at a total of $129.08 to fabricate with a Formiga P100. This combined cost is $439.30, which is within specifications and well below the cost of a typical large-format touchscreen. Detailed costs are shown in Appendix C

V. Control Performance – Methods

Performance of the individual control modes as well as the overall haptic effects has been evaluated. These tests were all performed manually by a single user without a standard weight applied. This is done to fully replicate the operating conditions of the device.

1) Path-Follow Control

Performance of path-follow control has been evaluated for a straight and curved path separately. The prototype was displaced 40 mm off of the path constraint with its wheel parallel to the path. The prototype was then manually moved in a direction at a near-constant speed as the controller guided it onto the path. Because operating speed cannot be explicitly controlled, and the controllers function in the spatial rather than the temporal domain, controller performance metrics were evaluated against path length traveled rather than time passed. Rise distance, settling distance, overshoot percentage, and steady state error were recorded (as opposed to rise time, etc.). Each trial was repeated five times for each path type. Lower gains are used for curved path tracking than for straight path tracking to account for increased instability responses during curved path tracking. Curved path tracking also employs an additional offset term to account for consistent steady state error equivalent to the path curvature value observed during development. The curved path in this test has a radius of 127 mm (5.0 inches). Both controllers were tuned to minimize rise distance in order to increase the response speed to changing path constraints. The same control gains were used for an identical test with the internal position estimation algorithm turned off. This was done to demonstrate the effect of the position update rate on control stability as well as the linear position estimator's ability to mitigate it. An example path trace for this test on each path-type is shown in Figure 37.

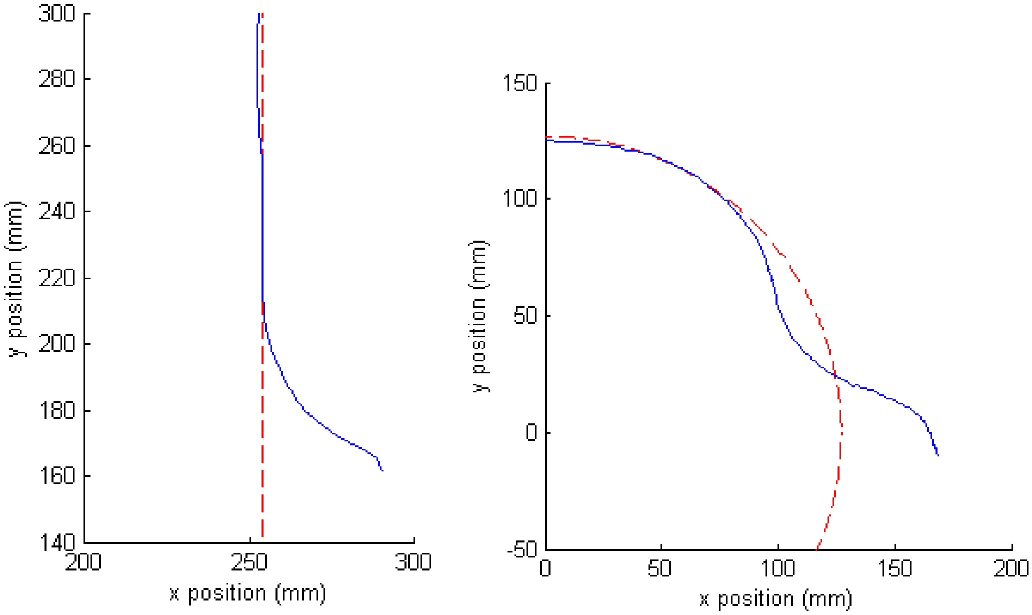


Figure 37. Path-follow controller test trace-through. The solid line represents the prototype position over time. The dotted line represents the path constraint.

2) Rigid Haptic Applications

Three demonstration applications have been designed on the Playsurface to evaluate the haptic rendering of the device: (1) A vertical wall restricting a user from navigating left of a selected x-value, (2) A "slot maze" game, which restricts a user to a series of intersecting straight-line paths at right-angles, and (3) A circular obstacle existing in free space. The wall simulation demonstrates the ability of the device to transition from free mode to path-follow mode in order to render a one-sided haptic constraint. The slot-maze game uses force sensing to determine which path to constrain the user to at an intersection. This demonstrates the use of force data to interpret user intent and display multiple simultaneous spatial constraints. The circular obstacle simulation demonstrates the ability of the device to simulate obstacles in free space, and divert the user around them with the dynamics of a near-frictionless surface.

The performance of the prototype during the test applications was evaluated by recording its position, velocity as detected by the touchscreen, and force data as detected by its onboard sensor. The prototype sensor values were sent to the touchscreen PC via serial communication during the tests. A head-on haptic wall collision and the slot maze game were executed 5 times each. For the wall collision test, the peak force and the wall penetration were recorded for each trial. Performance is compared with the wall collision anticipation control state enabled and disabled. The slot maze and the free-space obstacle are presented as primarily visual demonstrations of haptic applications of the device, as the performance metrics that would define them have been covered by the individual controller evaluations.

3) Path-Follow Admittance Control

Performance of path-follow admittance control has been evaluated by tracking a path with a set of admittance values, and manually applying forces toward both sides of the path. The commanded position of the prototype is recorded along with the actual position of the prototype to quantify the device error in simulating the desired output displacement. This test was conducted for a spring stiffness of 0.212 N/mm and 1.75 N/mm, while simulated mass was held constant at 1 kg and the damping ratio was held at 1.25.

Compliant versions of the rigid haptic applications have also been developed. Single trials of each application are presented at low and high stiffness values to illustrate the effect of admittance control on these applications.

W. Control Performance – Results

Rather than evaluate the controllers against specifications, these evaluations are used to set performance limits of the device. They are also used to confirm the performance improvements introduced by using wall collision anticipation and internal position estimation. Controller performance can be increased by improving design metrics already discussed, specifically the controller sample rate, the static friction force before slip, and the motor bandwidth. The 60 Hz position update rate is the driving limitation of all of the presented results, as demonstrated by the path follow evaluation with and without internal position estimation.

1) Path-Follow Control

The displacement error versus path distance traversed for both straight and curved paths is plotted in Figure 38.

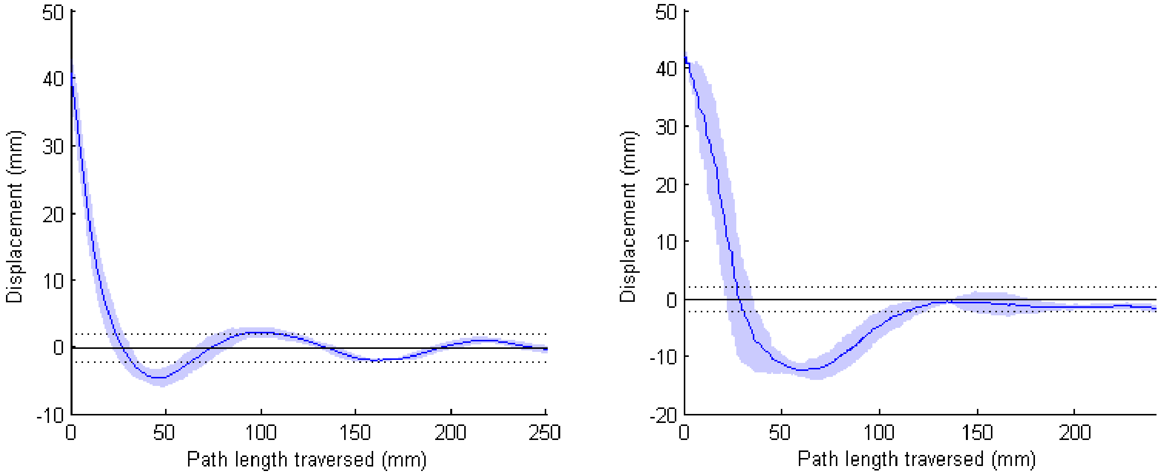


Figure 38: Path approach characteristics. Left - Straight path approach. Right – Radius = 127mm path approach. The transparent patch represents one standard deviation. Dotted lines mark the settling threshold (5% step value).

Performance metrics for these trials are shown in Table 5.

Table 5. Path follow performance metrics for two path curvatures. Uncertainty values represent two standard deviations.

Path curvature (m^{-1})	Rise (mm)	5% Settling (mm)	Overshoot (%)	SS error (mm)
0	26.2 ± 5.1	90.2 ± 98.3	12.4 ± 4.8%	<0.6
7.87	28.6 ± 13.5	135.8 ± 68.4	33.6 ± 6.3%	-1.3

Straight path tracking has significantly less overshoot than curved path tracking, but more oscillation around the steady-state value. This oscillation accounts for the similarity in the settling distance, despite the straight path-follow controller's faster response. It also accounts for the high settling distance variance for both path types, as some trials fall into the settling region after the initial overshoot, and some only after another oscillation outside of it. The oscillation could be damped with a higher tangency gain, but this term is approaching a magnitude that causes significant controller instability. Reduction of both gains is necessary before this effect can be damped further. The steady state error is recorded as less than 0.6 mm because this is the pixel resolution of the camera used to capture position data. Curved path tracking, on the other hand, experiences no oscillation after the initial overshoot, but does have steady state error. This is due to the relatively high tangency term in the controller, as well as possible error in the offset term. Curved path tracking also has higher variance in the approach compared to straight line tracking. Despite the magnitude of the gains in the curved path-follow controller being approximately half the magnitude used in the straight path-follow controller, the rise distance is almost identical. This suggests that lower gains and a relatively high tangency term are desirable if rejecting steady state oscillation is a priority over rejecting overshoot.

The results of this test with position estimation turned off are plotted in Figure 39.

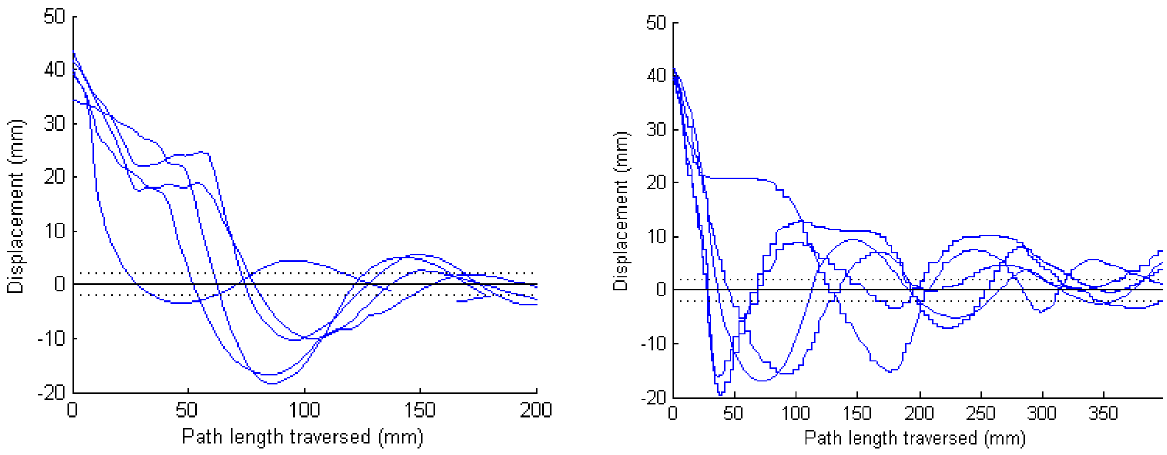


Figure 39. Path approach characteristics, position predicting disabled. Left - Straight path approach. Right – Radius = 127mm path approach. A variance patch was not used to prevent obscuring the inconsistent response.

Performance metrics for these trials relative to the previous set of trials are shown in Table 5.

Table 6. Path follow performance metrics for two curvatures, no position predicting. Rows directly below the reported values represent the percent change from the value recorded with position predicting.

	Path curvature (m^{-1})	**Rise (mm)**	**5% Settling (mm)**	**Overshoot (%)**	**SS error (mm)**
	0	59.2 ± 40.8	No settle	29.5 ± 30.0%	-0.9
Compared to position estimating:	-	+ 126%	-	+ 139%	+ >50%
	7.87	53.0 ± 83.2	No settle	41.8 ± 8.7%	2.5
Compared to position estimating:	-	+ 85.3%	-	+24.4%	+ 92.3%

Using these gains with the pure 60 Hz position update rate results in an inconsistent and marginally stable response that does not converge to within 5% of the initial deflection value. Steady state error is introduced to a path with zero curvature. Rise distance and overshoot increase significantly for both paths, and more than double for the straight line path approach. To improve the performance of these controllers at this sample rate, the gains would need to be reduced, sacrificing response speed for stability. Control performance in this prototype is primarily limited by the position update rate, as evidenced by its dramatic performance decrease when a higher update rate is no longer simulated.

2) Rigid Haptic Applications

A path trace of a typical haptic wall collision with and without collision anticipation is plotted in Figure 40.

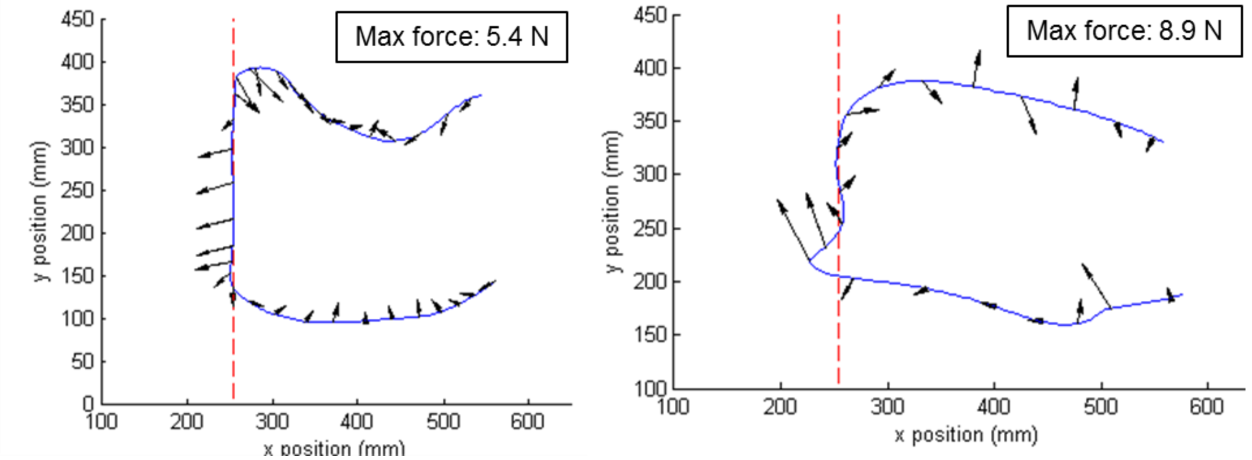


Figure 40: Rigid wall collisions. Left – Collision anticipation on, Right – Collision anticipation off. Arrows represent user force vectors and are scaled to the peak force noted on the plot. The dashed line is the haptic wall.

This application is designed to completely reject user motion into the wall while allowing free motion along or away from it. User force remains small in the free region as the wheel adjusts to correct for it. Force becomes larger upon wall collision, and can reach values up to the slip load of 10.6 N. Initial wall penetration of five trials was 9.1 mm ± 10.1 mm with collision anticipation on, and 31.5 ± 16.6 mm with collision anticipation off. Using the collision anticipation control state reduces the penetration to 29% of the amount allowed without collision anticipation for these trials. The wide variance can be accounted for by variations in approach angle and approach speed. Approach speed in particular significantly affects the transition between free mode and path-follow mode due to the significant delay in position data from the touchscreen.

Path traces of all five slot maze trials are plotted in Figure 41.

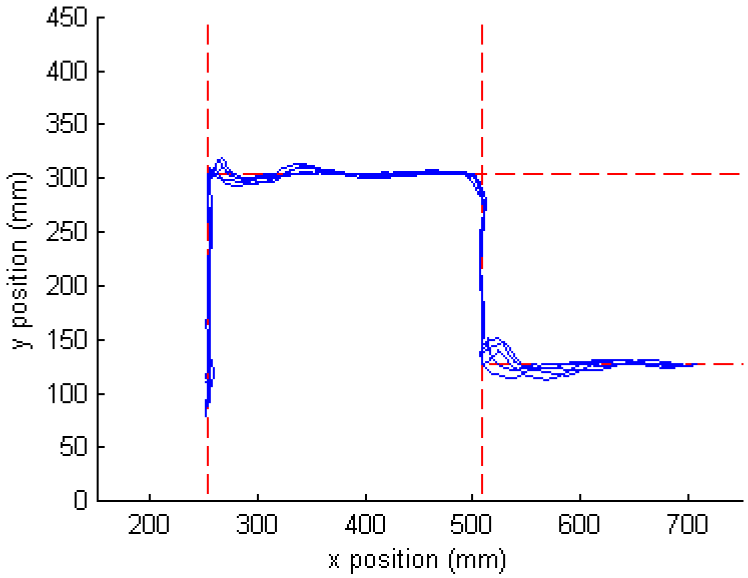


Figure 41. Five slot maze runs. Each trial begins at the lower left and ends at the lower right. Force data is omitted for clarity.

This test reveals some variations in device performance based on its position on the touchscreen. Path tracking falls to nearly zero error on each path, but tracks more tightly in the y-direction. This could be because of distortion of the projected image and mismatched scaling between pixels and length units used by the prototype microcontroller and the Playsurface touch recognition software. Path tracking performance also worsens near the corners of the screen, where the fiducial markers on the prototype are not as consistently tracked by the image processing software, which creates noise in the assigned centroid. However, the device is capable of rendering sharp constraint transitions due to its high motor bandwidth. It also showcases fairly consistent performance of a multi-stage task despite completely manual operation.

A collision with a circular obstacle in free space is plotted in Figure 42. As with the haptic wall collision, force magnitude remains low in the space defined by free mode. The surface robot is smoothly deflected around the edge of the obstacle before continuing in free space on the other side. Steady state error of 5.5 mm is noticeable from this figure. This is due to the steady state error observed in the path-follow controller, as well as the non-rigidity of the physical system under forces close to the wheel slip threshold.

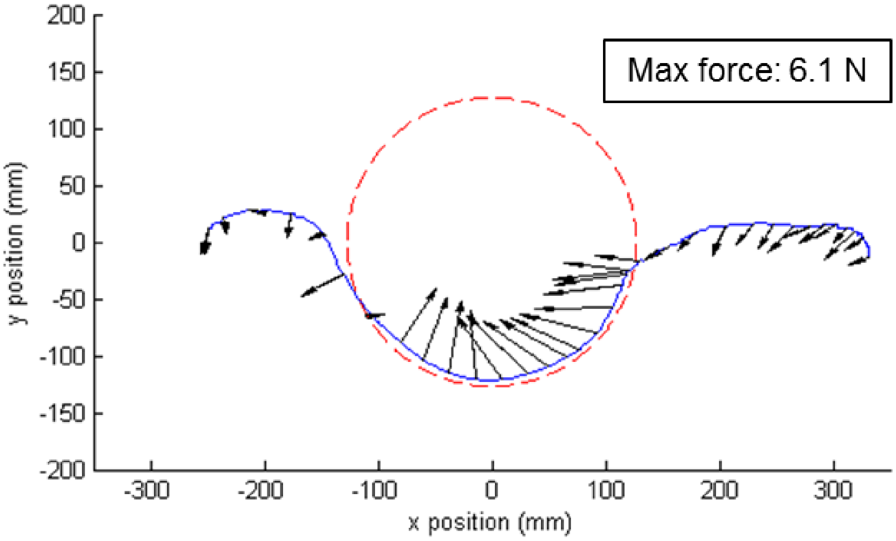


Figure 42. Collision with a rigid circle obstacle, R=127mm.

3) Path-Follow Admittance Control

Results from the low-stiffness path-follow admittance control tests are shown in Figure 43. The mean absolute displacement error during this test was 5.5 mm. The maximum force recorded by the device was 10.3 N. This is near to the maximum force the prototype can render before slipping, which is most likely a major contributor to the reported error. However, this displacement error is small compared to the large scale deflections of the path that the controller allows (about 11% of the maximum commanded deflection). This rapid variation in the desired path is where the output mechanical bandwidth requirement becomes apparent, as the motor must be able to control for positions varying as fast as a user can physical alternate force to the device. It can also be observed from the figure that the device returns to the undeformed path when perpendicular force disappears.

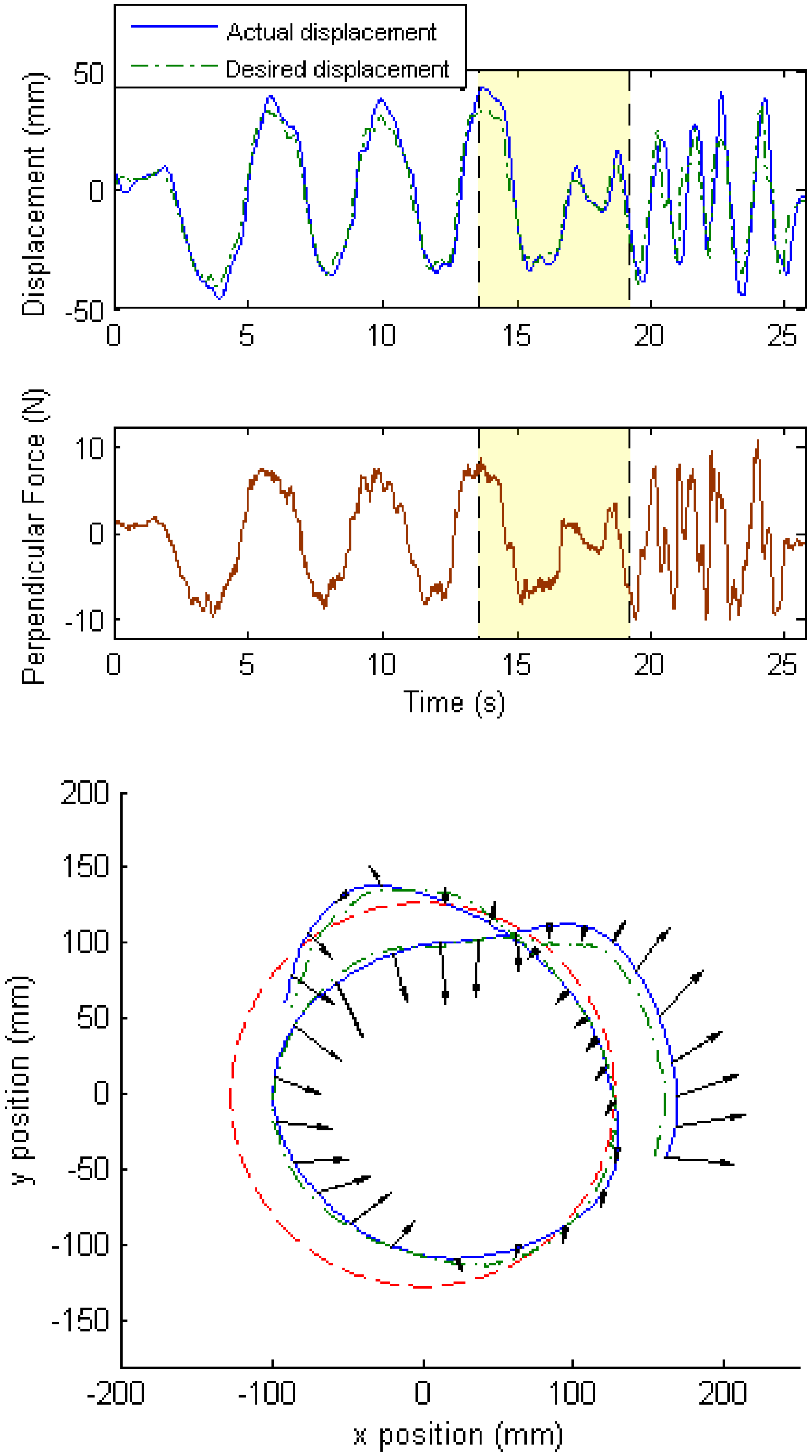


Figure 43: Surface robot response to a low-stiffness circular path (0.212 N/mm). The highlighted section is plotted to show the output of the controller in the taskspace as the surface robot traverses around the path counter-clockwise.

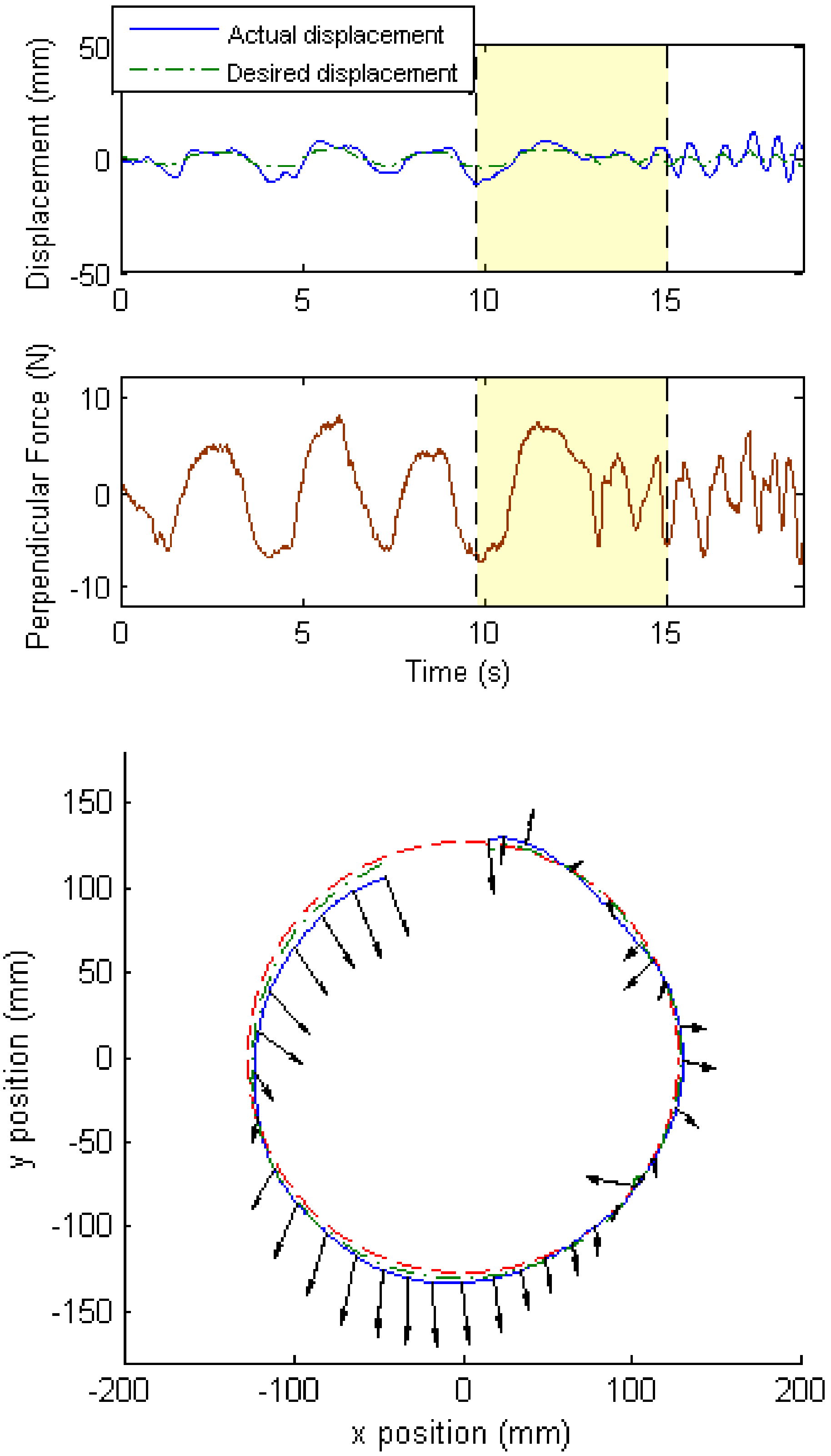


Figure 44: Surface robot to a high-stiffness circular path (k = 1.75 N/mm).

Results from the high-stiffness path-follow admittance control tests are shown in Figure 44. The user is constrained much more tightly to the original constraint despite forces of a similar magnitude to the low-stiffness test. However, deflection error is much more apparent relative to the small commanded deflections, despite having a smaller mean absolute displacement error of 2.65 mm (61% of the maximum commanded deflection). This stiffness is near the upper limit that path-follow admittance control is distinguishable from pure path-follow control. Commanded deflections below 3 mm are liable to be lost in the dynamic path tracking error of the device, which is a result of factors including rise distance, overshoot, and minor wheel slippage. The observed displacement error in the rigid circle collision, for example, is nearly the same as the deflection that this test attempts to dictate.

This control can be applied to any of the demonstrated haptic applications. Absolute mean displacement error using the original path constraint as a reference provides a penalty scoring metric for users attempting to stay within the allowed space. This metric also normalizes for path length and test duration by using the mean error. Low stiffness constraints allow for more penalty accumulation, making them appropriate for more advanced users. In the rehabilitation realm, this translates to a patient motor function metric, and allows for therapists to select an appropriate "difficulty level" for which a patient is capable of scoring a low error sum. For example, low and high-stiffness slot maze runs are plotted in Figure 45.

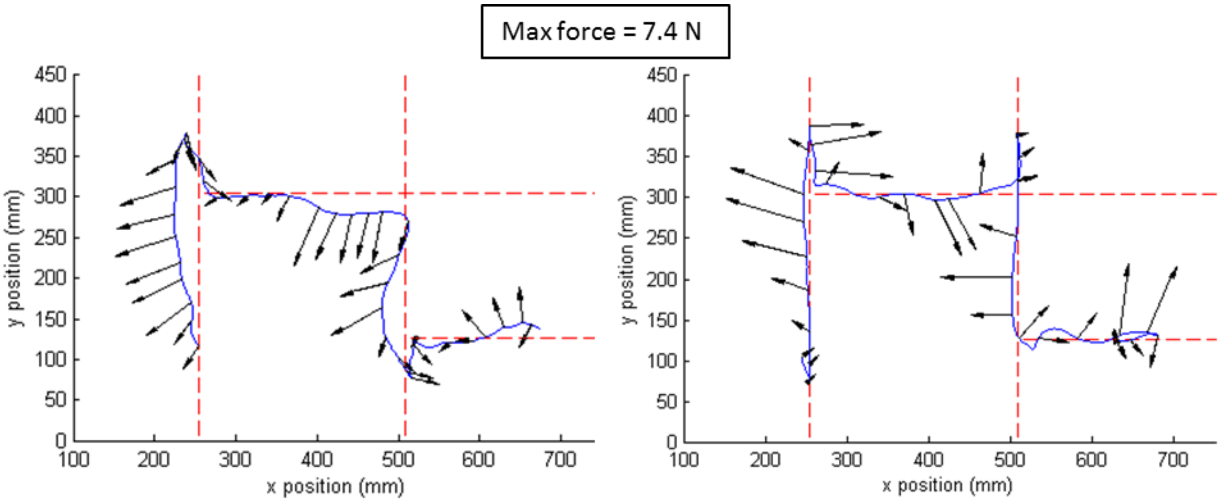


Figure 45. A run through low and high-stiffness compliant slot maze games. Low stiffness: M = 1kg, ζ = 1.25, k = 0.175 N/mm. High stiffness: M = 1kg, ζ = 1.25, k = 1.75 N/mm.

The mean error for the low-stiffness example is 14.1 mm, and the mean error for the high-stiffness example is 3.8 mm. The user exerts a similar amount of perpendicular force in both runs, but if the user goal is to score below 5 mm error before continuing to a higher difficulty, the high stiffness exercise is more appropriate.

The same technique can be applied to spaces defined by haptic walls and obstacles surrounded by free space, as shown in Figure 46.

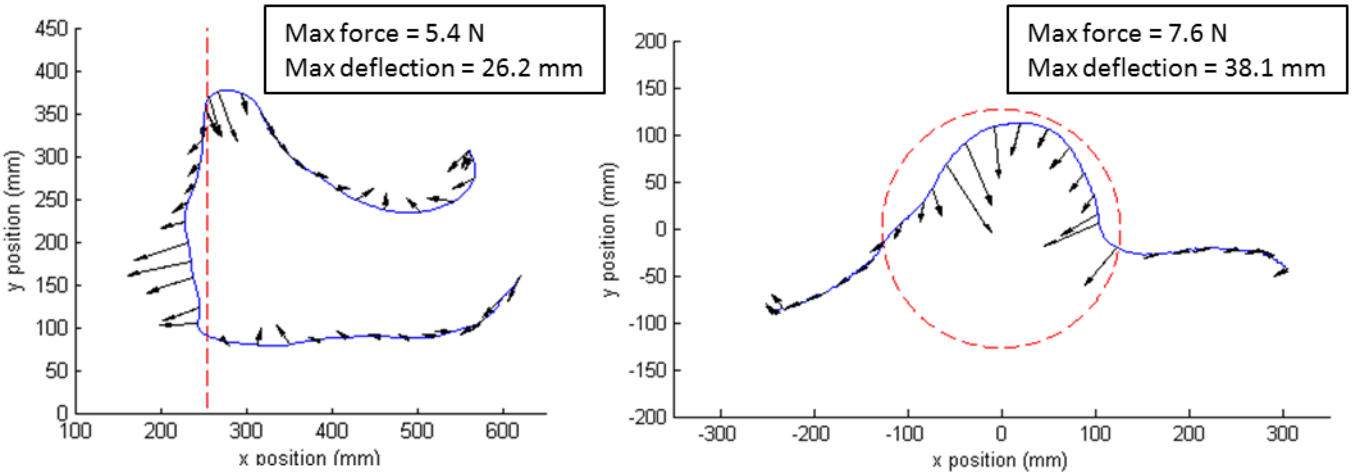


Figure 46. Low stiffness haptic obstacle collisions. Parameters are the same as in the low-stiffness slot maze example.

These tests are meant to serve as an example for potential applications of rigid and compliant spatial constraints as well as a visual demonstration of their performance. The stiffness of the constraints has an upper limit set by the path controller performance and stiffness of the mechanical system. However, the simulation of smooth low-stiffness obstacles capable of emulating the response of a system with damping and inertia is possible with a purely dissipative haptic device. The uses for this technology are in no way limited to the provided applications or example metrics.

CHAPTER VII

CONCLUSIONS

Touchscreen haptics is an emerging field with potential to significantly alter the way humans interface with electronic devices. It is important to address the uses for touchscreens outside of consumer electronics when considering haptic applications. Upper limb stroke therapy techniques have not been able to include haptic features in the migration to touchscreens. The presented device offers a method to provide a useful haptic toolset to touchscreen therapy applications. This allows for a meaningful touchscreen analogue to upper limb stroke therapy robots, and opens up the potential for creating valid comparison studies between touchscreen and robot arm/exoskeleton haptic therapy approaches. Beyond stroke therapy, smooth and compliant kinesthetic constraints on a large-format touchscreen can be used for creating immersive collaborative environments for the purposes of design or education. They can be used for touchscreen accessibility for the vision impaired for applications where whole arm motion is required or spatial constraints are preferred over tactile cues. Another avenue for future work is to move this technology off the touchscreen, providing haptic feedback along any flat surface where global position information can be obtained. This includes operation on surfaces with a projected image, or with webcam applications. However, an important limit of the design is that its haptic constraints can be circumvented or ignored by a user that lifts the device from the surface or forces the wheel to slip. This makes it inappropriate for applications where preventing user motion into a space is essential for safety or other reasons, such as teleoperation of machinery.

Several recommendations are apparent for future designs. For fast and precise controller response, an environmental reference frame must be provided with a higher frequency than 60 Hz, and preferably comparable to the internal controller rate of 667 Hz. Ideally, this system should run at 1 kHz to compete with precision force-feedback devices. The control developed for the device is achievable on a 60Hz reference frame, but this design constraint prevents it from being a precision force-feedback device, making it more appropriate for qualitative force feedback applications (e.g. gaming). Also, there is an optimal point to be found for the size of the wheel, trading off force sensor accuracy with motor speed output. For this design, the motor is sufficiently powerful that a small point of frictional contact is not needed to achieve a high bandwidth. Care must also be taken to ensure that the outer shell of the device can fully bottom out against the riding surface or that the amount of allowed vertical deflection is well-modeled, or the designed friction force will not be achieved. The force sensor can also be improved with a more rigorous and physics-informed modeling approach, potentially mapping out-of-plane motion of the magnet to ensure robustness to mechanical inconsistencies. Other design variations worth exploring include the addition of a drive motor or brake to the steered wheel, and the use of an attractive force between the device and the surface. This second suggestion involves crossover with other touchscreen haptics techniques, such as electrostatic attraction, and allows the possibility of operating the device on a non-horizontal plane.

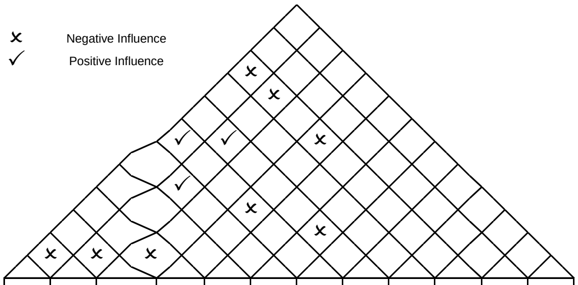
The contribution of this work is a platform for bridging the identified knowledge gap: the inability to output smooth, compliant kinesthetic constraints on a touchscreen platform. The result is a device capable of achieving this through novel contributions to device design, information structure, and steered control methods. Specifically, the design of an embedded force sensor and the path-follow admittance control mode allow for a touchscreen-capable haptic system that addresses this gap. This contribution opens up touchscreen haptics to new applications that require these kinesthetic feedback characteristics, notably upper limb stroke therapy. In addition, a knowledge base has been provided for future kinesthetic touchscreen approaches to build on this technique.

APPENDIX A

HOUSE OF QUALITY

The house of quality used to define the prototype design metrics is shown in Table A.

Table A. Surface robot house of quality.



Customer Attributes, Needs, Requirements, or Demanded Quality	Relative Importance or Weight (1 -- 5)	Control and render single-hand-scale forces	Exceed voluntary hand motion bandwidth	Fit into user hand	Detect user input force	Sense current allowable space	Detect virtual objects	Safety factor of critical structural components	Controller sample rate	Standard PC compatible	Total component cost below touchscreen			Haptic Surface Robot	InMotion ARM (MIT Manus)	ReTouch	TPad Fire
Smooth haptic response	5	M	H		M	M	H		H					3	5	1	5
Display solid objects	4	H	H		L	H	H	M	H		M			3	5	1	1
Respond to user intent of motion	5		H		H	M			H		M			5	5	1	1
Respond to visual environment	5	L			M	M	H		H					3	5	5	5
Intuitive user interface	4		H	H						M				5	1	5	5
Easy to operate with one hand	5	M	H	H				L	H					5	3	5	5
Replaceable - patient can take home	3		H		H	L	M				H			5	1	-	5
Does not require professional operator/supervision	4	M		M				M		H	H			5	1	3	5
Survive regular use by untrained users	5	M						H						3	5	3	5
Display compliant objects	4	M	H		H	H	H		H					3	5	1	1
Customizable haptic environments	4									H				5	1	1	3
Modular/portable	2	L		H				M		H	L			5	1	5	5
Direction of Improvement		+	+	-	-	-	+	+	+		-						
Measurement Units (SI)		N	Hz	mm	N	deg	Hz	-	Hz		$						
Delighted or Ideal Target Specification Values		20	8	100x40x40	0.1	0.088	1000	2.0	1000		100						
Disgusted or Marginal Target Specification Values		10	4	200x80x100	1.5	0.5	50	1.2	150	Y	1000						
Technical Importance		101	150	67	94	88	99	60	140	62	64						

Relative importance values were assigned based on their corresponding attributes' influence on haptic touchscreen therapy. Competing devices were selected to represent an in-use upper limb rehabilitation robot (InMotion ARM), a touchscreen-only therapy system (ReTouch), and a tactile haptic touchscreen tablet (TPad Fire).

APPENDIX B

FLEXURE DEFLECTION MODEL

The deflection for the flexure design in compression and in shear was modeled by applying Castigliano’s method to curved beam theory. This model is described by Equations B.1 and B.2.

$$d_c = 2\frac{P}{AE}\left\{\frac{1}{e_2}\int_{-\frac{\pi}{2}}^{\cos^{-1}-\frac{R_1}{R_2}}[R_2\cos(\theta)+R_1]^2 d\theta+\frac{R_1^2}{e_1}\int_{0}^{\frac{\pi}{2}}\sin^2(\theta)\,d\theta\right\} \tag{B.1}$$

$$\begin{aligned} d_s = \frac{P}{Ae_2E}\Bigg\{ & \int_{-\frac{\pi}{2}}^{\cos^{-1}-\frac{R_1}{R_2}}[L-R_1-R_2-R_2\sin(\theta)]^2 d\theta \\ & + \int_{-\cos^{-1}-\frac{R_1}{R_2}}^{\frac{\pi}{2}}[R_1+R_2-R_2\sin(\theta)]^2 d\theta\Bigg\} \\ & + \frac{P}{Ae_1E}\int_{0}^{\frac{\pi}{2}}\{[L-R_1\sin(\theta)]^2+R_1^2\cos^2(\theta)\}d\theta \end{aligned} \tag{B.2}$$

where $R_2 = \frac{L^2-4LR_1+8R_1^2}{4(L-2R_1)}$, $A = ht$, $e_1 = R_1 - t/\ln\frac{R_1+0.5t}{R_1-0.5t}$, and $e_2 = R_2 - t/\ln\frac{R_2+0.5t}{R_2-0.5t}$. P is the load on the flexure, and the subscripts refer to whether the flexure is acted upon in compression or in shear Note that the thickness of the beam is not neglected, though this effect becomes negligible for flexures at the implemented dimensions. These are: L = 6.35 mm, R1 = 2.54 mm, t = 0.58 mm, h = 20.32 mm.

There are two constraint functions for yielding, one for maximum stress in compression, and one for maximum stress in shear. These are described by Equations B.3 and B.4

$$\begin{aligned} g_1(R_1,L,t,h) = \max\Bigg[& -\frac{P_c}{A}-\frac{P_c(R_1\ +\ R_2\ )(r_n\ -\ r_{2,i})}{Ae_2r_{2,i}}, -\frac{P_c}{A} \\ & +\frac{P_c(R_1\ +\ R_2)(r_{2,o}\ - r_n)}{Ae_2r_{2,o}}\Bigg] - \frac{2}{3}S_y \leq\ 0 \end{aligned} \tag{B.3}$$

$$g_2(R_1, L, t, h) = \max\left[\frac{-P_s L\left(r_n\ -\ r_{1,i}\right)}{Ae_2 r_{1,i}}, \frac{P_s L\left(r_{1,o}\ - r_n\right)}{Ae_2 r_{1,o}}\right] - \frac{2}{3} S_y \leq\ 0 \qquad \text{(B.4)}$$

These values are found by finding the overall deflection from the maximum load, assumed in this optimization to be 20 N, using the total stiffness found in Equation 2, and multiplying by the stiffness in compression or shear as reached by the Castigliano method described earlier. Each r_n is the neutral axis for each curve, defined by the radius and thickness, each r_i is the inner radius of each section, and each r_o is the outer radius. S_y is the yield strength of the material as provided by EOS, the manufacturer of the 3D printer used to fabricate the flexures [49].

APPENDIX C

PROTOTYPE COST BREAKDOWN

Table C presents the full cost of the components of the prototype with small quantity prices. Components that come in packages with a large quantity have been scaled to represent the amount of that component used in one prototype.

Table C. Prototype cost breakdown.

Item	Vendor	P/N	Qty	Price
Miniature timing belt	Stock Drive	A 6Z16M067030	1	$ 2.84
Idler with plain bearing (wheel)	Stock Drive	A 7Z 5-UR625	1	$ 15.65
Miniature timing belt pulley	Stock Drive	A 6N16M020DF3004	1	$ 9.32
Stock Drive shipping			1	$ 8.97
Ball bearing .25" bore, .125" wide	Stock Drive	A 7Y55-PSS3725M	2	$ 12.94
Teflon rod (1.25 in OD, 12 in lg)	ePlastics	PTFENATROD1.250	0.0833	$ 1.30
Teflon sheet (12x12x0.030 in)	ePlastics	PTFENAT0.030X12X12	0.0156	$ 0.07
ePlastics shipping	ePlastics		1	$ 9.99
M2 pins, 8mm long (25 pk)	McMaster-Carr	93600A072	0.16	$ 1.12
M2.5, 4mm lg set screw (50 pk)	McMaster-Carr	92605A064	0.02	$ 0.13
M2.5, 4mm lg shc screw (50 pk)	McMaster-Carr	91290A099	0.06	$ 0.49
1/8" shaft side-mount retaining rings (100 pk)	McMaster-Carr	97431A240	0.02	$ 0.08
M1 pins, 8 mm long (10 pk)	McMaster-Carr	93600A033	0.2	$ 1.87
Printed circuit board	OSH Park		0.3333	$ 5.13
Motor/9.7:1 planetary gearhead	Faulhaber/Micromo	2224U024SRL+20/1 9.7:1+MG10	1	$ 217.20
Triaxis Hall effect sensor 8-SOIC	Melexis (Digikey)	MLX90363KDC-ABB-000-TU-ND	2	$ 9.74
L293D H-bridge	Microchip (Digikey)	497-2936-5-ND	1	$ 3.90
dsPIC33F microcontroller 28-SOIC	Microchip (Digikey)	DSPIC33FJ64MC202-I/SO-ND	1	$ 3.76
Capacitors (1 uF smd)	Digikey	311-1365-1-ND	10	$ 0.66
Resistors (10k smd)	Digikey	541-10KACT-ND	10	$ 0.77
MCP1700 Voltage regulator (3.3V)	Microchip (Digikey)	MCP1700-5002E/TO-ND	1	$ 0.74
Digikey shipping	Digikey		1	$ 3.54
Steering hub	Shapeways		1	$ 10.60
Prototype frame top	Shapeways		1	$ 30.53
Prototype frame bottom	Shapeways		1	$ 87.95
			Total:	$ 439.30

The cost of the prototype is primarily driven by the motor, which accounts for 49% of its total cost. The 3D printed frame and steering hub contribute 29%. The other mechanical and electrical components and their shipping costs contribute the remaining 22%.

REFERENCES

[1] O. Bau, I. Poupyrev, A. Israr, and C. Harrison, “TeslaTouch,” *Proc. 23nd Annu. ACM Symp. User interface Softw. Technol. - UIST ’10*, p. 283, 2010.

[2] L. Winfield, J. Glassmire, J. E. Colgate, and M. A. Peshkin, “T-PaD: Tactile Pattern Display through Variable Friction Reduction,” *Proc. World Haptics 2007*, pp. 421–426, 2007.

[3] T. Nakamura and A. Yamamoto, “Multi-finger surface visuo-haptic rendering using electrostatic stimulation with force-direction sensing gloves,” *IEEE Haptics Symp. HAPTICS*, pp. 489–491, 2014.

[4] A. Dunne, S. Do-Lenh, G. Ó’Laighin, C. Shen, and P. Bonato, “Upper extremity rehabilitation of children with cerebral palsy using accelerometer feedback on a multitouch display,” *2010 Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBC’10*, pp. 1751–1754, 2010.

[5] D. Rand, G. Zeilig, and R. Kizony, “Rehab-let: touchscreen tablet for self-training impaired dexterity post stroke: study protocol for a pilot randomized controlled trial,” *Trials*, vol. 16, no. 1, p. 277, 2015.

[6] N. Mumford, J. Duckworth, R. Eldridge, M. Guglielmetti, P. Thomas, D. Shum, H. Rudolph, G. Williams, and P. H. Wilson, “A virtual tabletop workspace for upper-limb rehabilitation in Traumatic Brain Injury (TBI): A multiple case study evaluation,” *2008 Virtual Rehabil. IWVR*, pp. 175–180, 2008.

[7] G. B. Prange, M. J. a Jannink, C. G. M. Groothuis-Oudshoorn, H. J. Hermens, and M. J. Ijzerman, “Systematic review of the effect of robot-aided therapy on recovery of the hemiparetic arm after stroke.,” *J. Rehabil. Res. Dev.*, vol. 43, no. 2, pp. 171–184, 2006.

[8] G. Kwakkel, B. J. Kollen, and H. I. Krebs, “Effects of robot-assisted therapy on upper limb recovery after stroke: a systematic review.,” pp. 1–11, 2007.

[9] J. Mehrholz, T. Platz, J. Kugler, and M. Pohl, “Electromechanical and Robot-Assisted Arm Training for Improving Arm Function and Activities of Daily Living After Stroke,” *Stroke*, vol. 40, no. 5, pp. e392–e393, 2009.

[10] L. Marchal-Crespo and D. J. Reinkensmeyer, “Review of control strategies for robotic movement training after neurologic injury.,” *J. Neuroeng. Rehabil.*, vol. 6, p. 20, 2009.

[11] D. J. Reinkensmeyer and M. L. Boninger, “Technologies and combination therapies for enhancing movement training for people with a disability,” *J. Neuroeng. Rehabil.*, vol. 9, no. 1, p. 17, 2012.

[12] C. Xu, A. Israr, I. Poupyrev, O. Bau, and C. Harrison, “Tactile display for the visually impaired using TeslaTouch,” *Proc. CHI EA ’11*, pp. 317–322, 2011.

[13] J. Mullenbach, C. Shultz, A. M. Piper, M. Peshkin, and J. E. Colgate, “Surface haptic interactions with a TPad tablet,” *Proc. Adjun. Publ. 26th Annu. ACM Symp. User interface Softw. Technol. - UIST ’13 Adjun.*, pp. 7–8, 2013.

[14] J. Mullenbach, C. Shultz, A. M. Piper, M. Peshkin, and J. E. Colgate, “TPad Fire: Surface Haptic Tablet,” *HAID Haptic Audio Interact. Des.*, no. Figure 2, pp. 103–105, 2013.

[15] D. J. Meyer, M. Wiertlewski, M. a. Peshkin, and J. E. Colgate, “Dynamics of ultrasonic and electrostatic friction modulation for rendering texture on haptic surfaces,” *IEEE Haptics Symp. HAPTICS*, pp. 63–67, 2014.

[16] C. D. Shultz, M. A. Peshkin, and J. E. Colgate, “Surface Haptics via Electroadhesion : Expanding

Electrovibration with Johnsen and Rahbek," *IEEE World Haptics Conf.*, pp. 57–62, 2015.

[17] T. Nakamura and A. Yamamoto, "Multi-finger electrostatic passive haptic feedback on a visual display," *2013 World Haptics Conf. WHC 2013*, no. 1, pp. 37–42, 2013.

[18] N. Marquardt, M. a Nacenta, J. E. Young, S. Carpendale, S. Greenberg, and E. Sharlin, "The Haptic Tabletop Puck : Tactile Feedback for Interactive Tabletops," *Proc. ACM Int. Conf. Interact. Tabletops Surfaces*, pp. 85–92, 2009.

[19] D. Ledo, M. a. Nacenta, N. Marquardt, S. Boring, and S. Greenberg, "The HapticTouch toolkit," *Proc. Sixth Int. Conf. Tangible, Embed. Embodied Interact. - TEI '12*, vol. 1, no. 212, p. 115, 2012.

[20] H. Mi and M. Sugimoto, "HATs : Interact using Height-Adjustable Tangibles in Tabletop Interfaces," *Proc. ACM Int. Conf. Interact. Tabletops Surfaces*, pp. 1–4, 2011.

[21] E. C. Chubb, J. E. Colgate, and M. A. Peshkin, "ShiverPaD: A Glass Haptic Surface That Produces Shear Force on a Bare Finger," *IEEE Trans. Haptics*, vol. 3, no. 3, pp. 189–198, 2010.

[22] X. Dai, J. E. Colgate, and M. A. Peshkin, "LateralPaD: A surface-haptic device that produces lateral forces on a bare finger," *Haptics Symp. 2012, HAPTICS 2012 - Proc.*, pp. 7–14, 2012.

[23] J. Mullenbach, D. Johnson, J. E. Colgate, and M. A. Peshkin, "ActivePaD surface haptic device," *Haptics Symp. 2012, HAPTICS 2012 - Proc.*, pp. 407–414, 2012.

[24] S. Saga and K. Deguchi, "Lateral-force-based 2.5-dimensional tactile display for touch screen," *Haptics Symp. 2012, HAPTICS 2012 - Proc.*, pp. 15–22, 2012.

[25] "Touch to Feel the Virtual World - Microsoft Research." .

[26] Y. Jansen, T. Karrer, and J. Borchers, "MudPad : Tactile Feedback and Haptic Texture Overlay for Touch Surfaces," *Proc. ACM Int.Conf. ITS*, pp. 11–14, 2010.

[27] M. Weiss, S. Voelker, J. Borchers, and C. Wacharamanotham, "FingerFlux: Near-surface Haptic Feedback on Tabletops," *Proc. 24th Symp. User Interface Softw. Technol. - UIST '11*, pp. 615–620, 2011.

[28] B. Long, S. A. Seah, T. Carter, and S. Subramanian, "Rendering volumetric haptic shapes in mid-air using ultrasound," *ACM Trans. Graph.*, vol. 33, no. 6, pp. 1–10, 2014.

[29] R. Sodhi, Poupyrev, M. Glisson, and a Israr, "AIREAL: interactive tactile experiences in free air," *ACM Trans. Graph.*, vol. 32, no. 4, p. 134, 2013.

[30] J. E. Colgate, W. Wannasuphoprasit, and M. a. Peshkin, "Cobots: Robots For Collaboration With Human Operators." 1996.

[31] M. A. Peshkin, J. Edward Colgate, W. Wannasuphoprasit, C. A. Moore, R. Brent Gillespie, and P. Akella, "Cobot architecture," *IEEE Trans. Robot. Autom.*, vol. 17, no. 4, pp. 377–390, 2001.

[32] R. Brent Gillespie, J. Edward Colgate, and M. A. Peshkin, "A general framework for Cobot control," *IEEE Trans. Robot. Autom.*, vol. 17, no. 4, pp. 391–401, 2001.

[33] M. a. Peshkin, J. Edward Colgate, W. Wannasuphoprasit, C. a. Moore, R. Brent Gillespie, and P. Akella, "Cobot architecture," *IEEE Trans. Robot. Autom.*, vol. 17, no. 4, pp. 377–390, 2001.

[34] H. I. Krebs, M. Ferraro, S. P. Buerger, M. J. Newbery, A. Makiyama, M. Sandmann, D. Lynch, B. T. Volpe, and N. Hogan, "Rehabilitation robotics: pilot trial of a spatial extension for MIT-Manus," *J. Neuroeng. Rehabil.*, vol. 1, p. 5, 2004.

[35] T. Nef, M. Guidali, and R. Riener, "ARMin III – arm therapy exoskeleton with an ergonomic shoulder actuation," *Appl. Bionics Biomech.*, vol. 6, no. 2, pp. 127–142, 2009.

[36] R. Van Der Linde, P. Lammertse, E. Frederiksen, and B, "The HapticMaster, a new high-performance haptic interface," *Proc. EuroHaptic, Edinburgh, UK*, pp. 1–5, 2002.

[37] P. S. Lum, C. G. Burgar, M. Van der Loos, P. C. Shor, M. Majmundar, and R. Yap, "MIME robotic device for upper-limb neurorehabilitation in subacute stroke subjects: A follow-up study.," *J. Rehabil. Res. Dev.*, vol. 43, no. 5, pp. 631–642, 2006.

[38] N. Foley, S. Mehta, J. Jutai, E. Staines, and R. Teasell, "Chapitre 10 : Upper Extremity Interventions," in *Evidence-Based Review of Stroke Rehabiliation*, 2013, pp. 1–163.

[39] G. C. Burdea, "Virtual rehabilitation--benefits and challenges.," *Methods Inf. Med.*, vol. 42, no. 5, pp. 519–23, 2003.

[40] S. E. Fasoli, H. I. Krebs, J. Stein, W. R. Frontera, and N. Hogan, "Effects of robotic therapy on motor impairment and recovery in chronic stroke," *Arch. Phys. Med. Rehabil.*, vol. 84, no. 4, pp. 477–482, 2003.

[41] P. Maciejasz, J. Eschweiler, K. Gerlach-Hahn, A. Jansen-Troy, and S. Leonhardt, "A survey on robotic devices for upper limb rehabilitation.," *J. Neuroeng. Rehabil.*, vol. 11, no. 1, p. 3, 2014.

[42] H. Krebs, S. Conroy, C. Bever, and N. Hogan, "Forging Mens et Manus: The MIT Experience in Upper Extremity Robotic Therapy.," in *Neurorehabilitation Technology*, 2011, pp. 125–140.

[43] M. Annett, F. Anderson, D. Goertzen, J. Halton, Q. Ranson, W. F. Bischof, and P. Boulanger, "Using a multi-touch tabletop for upper extremity motor rehabilitation," *Proc. 21st Annu. Conf. Aust. Comput. Interact. Spec. Interes. Gr. Des. Open 24/7 - OZCHI '09*, p. 261, 2009.

[44] J. Kowalczewski, S. L. Chong, M. Galea, and A. Prochazka, "In-Home Tele-Rehabilitation Improves Tetraplegic Hand Function," *Neurorehabil. Neural Repair*, vol. 25, no. 5, pp. 412–422, 2011.

[45] M. Sivan, J. Gallagher, S. Makower, D. Keeling, B. Bhakta, R. J. O¿Connor, and M. Levesley, "Home-based Computer Assisted Arm Rehabilitation (hCAAR) robotic device for upper limb exercises after stroke: results of a feasibility study in home setting," *J. Neuroeng. Rehabil.*, vol. 11, no. 1, p. 163, 2014.

[46] J. Kowalczewski, E. Ravid, and a. Prochazka, "Fully-automated test of upper-extremity function," *Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBS*, pp. 7332–7335, 2011.

[47] M. Leitner, M. Tomitsch, T. Költringer, K. Kappel, and T. Grechenig, "Designing tangible tabletop interfaces for patients in rehabilitation," *CEUR Workshop Proc.*, vol. 415, pp. 1–7, 2008.

[48] M. Price and F. C. Sup IV, "A Robotic Touchscreen Totem for Two-Dimensional Haptic Force Display," in *Haptics Symposium 2016, HAPTICS 2016 - Proceedings*, 2016.

[49] EOS, "PA 2200 Performance 1.0," 2010. [Online]. Available: http://eos.materialdatacenter.com/eo/.

[50] M. Schmidt and H. Lispon, "Distilling Free-Form Natural Laws from Experimental Data," *Science (80-.).*, vol. 324, no. 5923, pp. 81–85, 2009.

[51] R. Ponche, J. Y. Hascoet, O. Kerbrat, and P. Mognol, "A new global approach to design for additive manufacturing," *Virtual Phys. Prototyp.*, vol. 7, no. 2, pp. 93–105, 2012.

[52] G. A. O. Adam and D. Zimmer, “Design for Additive Manufacturing-Element transitions and aggregated structures,” *CIRP J. Manuf. Sci. Technol.*, vol. 7, no. 1, pp. 20–28, 2014.

[53] D. Eddy, J. Calderara, M. Price, S. Krishnamurty, and I. Grosse, “Decision Support System for Additive Manufacturing,” in *ASME International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, 2016, pp. 1–10.