8-1-2010

Modeling of micro-scale touch sensations for use with haptically augmented reality

Adam Woodrow Weissman

Follow this and additional works at: http://scholarworks.rit.edu/theses

Recommended Citation

This Thesis is brought to you for free and open access by the Thesis/Dissertation Collections at RIT Scholar Works. It has been accepted for inclusion in Theses by an authorized administrator of RIT Scholar Works. For more information, please contact rit scholar works@rit.edu.
Modeling of Micro-scale Touch Sensations for use with Haptically Augmented Reality

by

Adam Woodrow Weissman

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Computer Engineering

Supervised by

Associate Professor Dr. Shanchieh Jay Yang
Department of Computer Engineering
Kate Gleason College of Engineering
Rochester Institute of Technology
Rochester, New York
August 2010

Approved by:

Dr. Shanchieh Jay Yang, Associate Professor
Thesis Advisor, Department of Computer Engineering

Dr. Roy Melton, Lecturer
Committee Member, Department of Computer Engineering

Dr. Edward Brown, Assistant Professor
Committee Member, Department of Electrical Engineering
Thesis Release Permission Form

Rochester Institute of Technology
Kate Gleason College of Engineering

Title:

Modeling of Micro-scale Touch Sensations for use with Haptically Augmented Reality

I, Adam Woodrow Weissman, hereby grant permission to the Wallace Memorial Library to reproduce my thesis in whole or part.

___________________________________________________________
Adam Woodrow Weissman

___________________________________________________________
Date
Abstract

Modeling of Micro-scale Touch Sensations for use with Haptically Augmented Reality

Adam Woodrow Weissman

Supervising Professor: Dr. Shanchieh Jay Yang

Possessing dexterity and sensory perceptions, the human hand is a versatile tool that can grasp, hold, and manipulate objects using various postures and forces interacting with the environment. Many industrial tasks are replacing human hands with anthropomorphic robotic hands. In skillful tasks such as micro surgical operations, a master-slave interface system of robotic hands is required to emulate a human hand’s dexterity by using glove controllers with force sensors for telemanipulation. Although these interface techniques are widely applied for large scale robots, little has been accomplished for micro-scale robots due to the constraints and complexity imposed by miniaturization.

To provide sensible haptic control and feedback from robots at the micro-level, this work investigates the intricacies associated with the use of micro-scale robotic actuators with the intention of using them with haptic feedback systems. This work also develops a system model to test the ability of computing elements that emulate a microrobotic hand’s tactile perception of stiffness. An interface glove was used to collect control data from the user, which was used alongside a Matlab model to simulate the operation and control of two different microhand designs. In order to control the microhand device accurately, feedback from simulated sensors was used to affect the airflow of the pneumatic system driving the displacement of the microhand. Four major components were developed for the overall system. The glove interface gives the operator a method to interact with the system. The microhand modeling took place in two components. The first component was the
model of the microhand itself. The other component needed was a pneumatic subsystem to drive the microhand operation. The final major component developed was a graphical user interface to give the operator feedback as to what is happening in the target environment. The integration of all of these components allows for experimentation of the intricacies of operating with these microhand devices.

The investigation of this micro-haptic system shows that some parameters make the system perform faster and more accurately than others. Metrics such as percent error and settling time of the displacement of one micro-finger are shown to measure success of each method. Future improvements for this system could include the integration of pneumatically controlled balloon micro-actuators with the operator’s glove interface or implementing more accurate contact mechanics into the model.
Contents

Abstract ................................................................. iii

Glossary ............................................................... x

1 Introduction .......................................................... 1
  1.1 Motivation ......................................................... 2
  1.2 Project Description ............................................ 3

2 Related Work ......................................................... 5
  2.1 Haptic Interface Systems ....................................... 5
  2.2 Microhand Devices ............................................. 10
  2.3 Micro-haptic Sensor ............................................ 11
  2.4 Control Methodologies ......................................... 12
    2.4.1 MEMS Control ............................................ 12
    2.4.2 Pneumatic Control ....................................... 13
    2.4.3 Micro-hand Control ...................................... 14
  2.5 Stiffness ......................................................... 14

3 System Design ...................................................... 17
  3.1 Overview ....................................................... 17
  3.2 Haptic Interface ............................................... 19
    3.2.1 Haptic Control ......................................... 19
    3.2.2 Haptic Display .......................................... 21
  3.3 Calibration ..................................................... 22
  3.4 Control Methodology ......................................... 23
    3.4.1 Environment Modeling .................................. 24
    3.4.2 Solenoid Modeling ....................................... 25
    3.4.3 Air Muscle Modeling .................................... 26
    3.4.4 Graphical User Interface and Verification .......... 27
    3.4.5 Haptic Display .......................................... 27
# Results and Analysis

4.1 Lu versus Konishi Model .................................................. 30
4.2 Stiffness ................................................................. 33
4.3 Sensor Noise ............................................................ 38
4.4 Regulator Limits .......................................................... 42
4.5 Input Type ................................................................. 45
4.6 Controller Type ........................................................... 53
4.7 Summary ................................................................. 60

# Conclusions and Future Work ............................................. 61

Bibliography ................................................................. 63

A P5 Matlab Integration Code .............................................. 66
B Active Calibration Matlab Code ........................................ 69
C Data Selection Matlab Code .............................................. 72
D Data Selection GUI Matlab Code .................................... 77
E Virtual Reality Model (VRML code) .................................... 81
# List of Tables

2.1 Folgeraiter Duty Cycle Results [14] ................................................. 10

4.1 Base Line Performance Metrics ....................................................... 31
4.2 Stiffness Performance Metrics ........................................................ 33
4.3 Sensor Noise Performance Metrics ............................................... 39
4.4 Regulator Performance Metrics ...................................................... 43
4.5 Input Type Performance Metrics ..................................................... 46
4.6 Performance Metrics of Different Controllers ................................. 53
# List of Figures

1.1 System Diagram of Haptic Interface and Virtual Environment for Micro-hand System .................................................. 4

2.1 Fundamental Sensations that make up the Sense of Touch [28] ............... 5
2.2 Cohn Spatial Discrimination Results [11] ........................................ 8
2.3 Folgheraiter Current Intensity Results [14] ...................................... 9
2.4 Folgheraiter Frequency Results [14] .............................................. 9
2.5 Direction of Motion of Differing Designs of Micro-finger Joints .......... 11
2.6 Stress versus strain of a various types of cells [1] ............................. 16

3.1 Conceptual Feedback Loop Control Model .................................... 18
3.2 Essential Reality P5 Glove [6] ................................................... 20
3.3 Simulink Control Model .......................................................... 23
3.4 Simulink Model for Pneumatic Solenoid ...................................... 25
3.5 Lu measured force versus pressure data [22] ................................. 26
3.6 Konishi measured force versus pressure data [18] .......................... 27
3.7 Data Set and Regression Method Selection Tool .............................. 28
3.8 Graphical User Interface .......................................................... 28

4.1 Percent Displacement using Lu Dataset with Soft (2) Object ......... 31
4.2 Error using Lu Dataset with Soft (2) Object ................................. 32
4.3 Error using Konishi Dataset with Soft (2) Object ......................... 32
4.4 Error using Lu Model and an Object of Stiffness=2 ....................... 33
4.5 Error using Lu Model and an Object of Stiffness=10 ................. 34
4.6 Error using Lu Model and an Object of Stiffness=100 ............... 35
4.7 Error using Konishi Model and an Object of Stiffness=2 ........... 35
4.8 Error using Konishi Model and an Object of Stiffness=10 ........ 36
4.9 Error using Konishi Model and an Object of Stiffness=100 .... 37
4.10 Error using Slightly Noisy Lu Dataset ....................................... 38
4.11 Error using Highly Noisy Lu Dataset ......................................... 39
4.12 Error using Slightly Noisy Konishi Dataset ................................. 40
4.13 Error using Highly Noisy Konishi Dataset ................................. 41
# Glossary

<table>
<thead>
<tr>
<th>H</th>
<th>Having to do with the sensations associated with touch, p. 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCFM</td>
<td>Standard Cubic Foot per Meter, p. 25.</td>
</tr>
<tr>
<td>PSI</td>
<td>Pounds per Square Inch, p. 26.</td>
</tr>
<tr>
<td>TENS</td>
<td>Transcutaneous Electric Nerve Stimulation, p. 8.</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

A challenging problem in the field of robotics and micro-robotics is to incorporate natural user interfaces allowing easy and sensible operation. Three types of interfaces have been researched extensively: verbal, visual, and haptic. Verbal interfaces allow users the ability to give commands orally to a robot and allow the robot to give feedback orally [26]. Visual interfaces allow users to visualize the state of the robot and the results of its actions as well as to provide stimulation through non-contact and non-oral means [26]. The third type, haptic interfaces, allows the sensation of touch to be perceived through electronics and mechanical apparatuses. These natural interfaces are typically applied to robot systems of a macroscopic scale.

With the MEMS and NEMS fields gaining interest, robotic systems are shrinking to microscopic and nano-scopic scales and finding new applications, such as minimally invasive surgery. With new forms and paradigms for robotics, new interfaces and control methods also need to be investigated. Traditional interfaces, data collection methods, and signal processing techniques developed for large scale robots may not be suitable for use in micro-scale robotics due to the constraints imposed by the size of the robot.
1.1 Motivation

One group of robotic systems that could benefit from advancements in haptic interfaces are tele-operated surgical robot systems because current surgical systems provide only visual feedback to the surgeon during operation. This is an issue because surgeons often rely on their sense of touch to perform operations [20]. By adding a form of haptic interface to these systems, surgeons could have more situational awareness of the target environment during a surgery.

In some models of the daVinci system [2], there is currently one level of haptic stimulation to the operator, force feedback. Force feedback in this case can be described as providing the operator resistance on a tool control when the tool cannot comply with the operator’s command. An example of this would be trying to cut different materials with a pair of scissors. When cutting a piece of paper, the operator feels very little resistance and the intended action is being completed. When trying to cut a sheet of steel, the operator cannot make the scissors close due to the resistance of the steel. One desired improvement [3] is to expand this capability by allowing to the operator to feel a range of resistances or stiffnesses driven by an actuator in the target environment.

Another complication to this problem is the effect of miniaturization of the robots’ tools on the haptic experience for the end user. Each generation of tool on these robotic surgical systems, like the daVinci Surgical System [4], shrinks to offer better minimally invasive surgery and shorten patient recovery time [4]. This is an issue because microscopic instruments, such as the microhand developed by Lu et al.[23] [22], may emulate the natural dexterity of a human hand but do not interact the same way a human hand interacts with its environment.

Miniaturization of the end effector also affects the robotic control scheme. Borovic et al.[10] cites three specific issues when designing control systems for MEMS devices:
MEMS systems are small and often very quick.

MEMS sensors can be large and change the dynamics of the system.

In most applications, one goal of a MEMS system is to be self contained. This constrains the controller to be as small as possible.

An added benefit to having a surgeon interact directly with computing elements is that simulations can be easily integrated into the same hardware. This is beneficial because training can be done on the same equipment used for an actual surgery without risk of damage to a patient or the equipment. Another added complexity to this problem could be providing meaningful feedback, both visual and haptic, to the surgeon. To do this, accurate models of how objects feel and respond to force need to be developed and evaluated.

1.2 Project Description

By investigating the the dynamics of micro-actuators with their target environments as they relate to touch and stiffness, insights to constraints, limitations, and other design decisions can be found to guide the design of future micro-scale haptic robotic systems. To investigate the sensation of touch at a microscopic scale, a system component level design was constructed. Surveys of haptic systems and haptic system components were conducted to identify different parameters in such a system. These surveys provided information on different system architectures, control techniques, and technology options which became the basis for the component models for a haptic input device, a display interface, microhand control system, and a model of the microhand environment.

This investigation looked into the attribute of stiffness with regard to touch. To accomplish this, models and modeling techniques were investigated and developed to produce
accurate and meaningful feedback to the user. Also needed was a means to collect the data supplied by the user to control the microhand model, and a display to supply feedback to the user. By collecting performance data, such as rise time, settling time, percent overshoot, and steady state error, the different model parameters can be compared and evaluated to suggest trends when designing physical micro-scale haptic robotic systems. See Figure 1.1 for a block diagram of the system interactions.

Figure 1.1: System Diagram of Haptic Interface and Virtual Environment for Microhand System

Since many of the components of the proposed system are non-linear, each component was modeled and pieced together instead of modeling the behavior of the entire system. This ensures that any piece can be changed in the future without affecting other subsystems. A similar approach has been taken by Borovic et al.[10]. This is particularly important for the microhand environment modeling; in the future, an actual microhand device will be integrated into the system. When this happens, actual force and displacement data will be passed to the elasticity calculating algorithm instead of using the modeled values.
Chapter 2

Related Work

2.1 Haptic Interface Systems

Haptic systems have been gaining interest in recent years. Many projects look to explore one aspect of touch, such as stiffness or texture. Shirado and Maeno [28] describe four sensations that make up the sense of touch: roughness, coldness, moistness, and stiffness. These sensations are visualized in 2.1. This investigation focuses on the sensation of stiffness.

Figure 2.1: Fundamental Sensations that make up the Sense of Touch [28]
To be able to give the user touch based feedback, a few options were considered. For applications based on the stiffness of an object, two methods are widely used. These methods are balloon actuation and pin actuation [8]. Each method has advantages and disadvantages. Benali et al. [8] describe four categories of haptic devices in their study: pin actuated, vibratory, bubble actuated (by pneumatics or by some liquid), and energy based (thermal or electrical injection).

Pin actuated devices often employ motors to raise or lower a series of pins to cause different haptic responses. Typically, force sensations, such as stiffness or deformation, are delivered by pin actuated devices. Braille machines often use this form of device, due to the similar final response generated by Braille and pin actuated devices. One disadvantage to this type of system is that the motors that drive the pins need to be located very near the human-machine interface. This requirement causes pin actuated devices to be bulky.

Vibratory devices cause the operator’s body to oscillate at specific frequencies to imply a feeling of roughness or friction. These devices are not used interchangeably with other types, because of the specific type of sensations they generate.

Bubble actuated devices manipulate the properties of a substance within a bubble to alter the size of the bubble. Most common are pneumatic bubbles, which force air into the bubble and change the internal pressure. Other bubble actuators heat or cool a liquid inside the bubble that causes the liquid to expand or contract. Pneumatic bubbles can be hard to implement because many tubes may be necessary for the pneumatics to be effective.

Energy based haptic devices inject an electrical current or transfer heat into the user. These devices are typically very invasive and can be dangerous if not handled properly. Energy devices, though, can cause very different haptic responses, such as vibrations, warmth, or even pain.

Fan et al. [13] at UCLA have developed a system for helping people who are missing lower limbs. They implement a prosthetic that allows a more natural stride by adding haptic
sensors and a display array to a stock leg prosthetic. Their paper describes augmenting a lower limb prosthetic with force sensors on the foot, which allowed the electronics to sense how the patient’s weight was being distributed through a step. These data were then used to produce a signal to stimulate the user with PDMS pneumatic balloon actuators. Four balloons were mounted onto a cuff attached to the user’s leg and were inflated with solenoids and regulators controlled by a micro-controller. The balloons were placed in similar positions as compared to the locations of the sensors on the foot. This work is important because it shows the feasibility of communicating haptic information through electronics and mechanical actuators.

King et al. [20] [12] have developed a system that augments the daVinci surgical robot [4] with a balloon array and force sensors to provide haptic feedback to the operator. The force sensors were added to one of the gripper tools on the surgical robot and PDMS pneumatic balloon actuators were added to the controls of the daVinci robot. This research begins to look at concerns raised when interacting with an environment that is not macro scale, although the system uses a macro-level force sensor and a 1-to-1 mapping of measured force to applied balloon force.

Other groups, such as Cohn et al. [11], have tried to characterize the use of a pneumatic array as a human feedback device. In their work, a 5x5 pneumatic piston array was created and tested to find suitable process and operation parameters that would stimulate the user optimally. The first experiment of Cohn et al. was to test pattern recognition on the piston array. Out of 149 trials, 77% of patterns were identified correctly. This also shows the feasibility of communicating haptic information through electronic and mechanical means. The second experiment presented was in relative force discrimination. In this set of tests, two array elements were actuated with different pressures, and the test subject was to identify which element had a greater force. This set of trials resulted in a 75% success rate, and they estimated that a human’s force sensitativity is approximately 37mN. Cohn’s third set of
tests evaluated the position discrimination of a human finger. This was done by stimulating a section of the array and translating the actuated section. The subject was then asked to identify the direction of translation. Figure 2.2 shows the percentage correct versus the distance between original actuation and the translated actuation of that series of trials. As shown in Figure 2.2, as the distance increased, the subject was more able to identify the translation. The 75% correct point occurred at approximately 0.14mm, showing the spatial resolution of a human fingertip. This work is important because it is an early attempt to characterize the human finger’s sensing abilities. Knowing these capabilities will result in devices that maximize efficiency in terms of stimulation to the user, design time, and materials used in construction.

![Figure 2.2: Cohn Spatial Discrimination Results](image)

Other forms of tactile stimulation have also been investigated. Folgheraiter *et al.*[14] developed a TENS haptic feedback system, in which electrical signals were used to generate touch sensations. TENS devices inject an electrical signal of varying currents, voltages, and frequencies to the skin to stimulate the different touch receptors. The advantage that
Folgheraiter gives to this type of systems is that the sensations of touch can be recreated without the need to recreate the physical properties that make up that sensation. She goes on to conduct some experiments aimed to characterize different parameters of the electrical signal as they relate to sensations they generate to the subject. In the first set of trials, the effects of frequency and current were tested. As would be expected, as more current is applied, the subject’s sensation was more intense. This effect is shown in Figure 2.3. The frequency of the signal seemed to affect the intensity of the sensation less than the current. The frequency, though, affected the type of sensation, as shown in Figure 2.4.

![Figure 2.3: Folgheraiter Current Intensity Results](image1)

The next experiment Folgheraiter et al. conducted was to characterize the effects of duty cycle on the induced sensation. For this experiment they injected a signal of 10Hz or 50Hz,
middle or high intensity (these current values changed from person to person), and 10% or 90% duty cycle. Then two signals were injected sequentially varying only the duty cycle, and the subject was to give a response comparing one signal to the other. Such responses included lower, stronger, softer, harder, faster, slower, and equal. See Table 2.1 for results of this experiment. Folgheraiter asserted that typically the lower duty cycle gives a slower but clearer or more identifiable sensation, whereas the higher duty cycle generates a more intense and faster but also smoother or less identifiable sensation.

<table>
<thead>
<tr>
<th>Duty Cycle</th>
<th>L</th>
<th>St</th>
<th>So</th>
<th>H</th>
<th>F</th>
<th>SI</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>10%</td>
<td>11</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>90%</td>
<td>0</td>
<td>9</td>
<td>8</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 2.1: Folgeraiter Duty Cycle Results [14]

This work is interesting because it employs a more exotic method of generating sensation with some success. This is an interesting method due to the fact that it is not recreating the physical situation that would evoke such a sensation, which is important because it may not need such bulky equipment as other methods do.

### 2.2 Microhand Devices

One goal of this research was to visualize the use of a microhand device in minimally invasive surgery. For simulation purposes, a microhand device needed to be modeled. Two different microhand designs were investigated. Each of the different apparatuses uses different mechanics to mimic the motion of a human joint.

Lu et al. [22] [23] built a microhand device that caused actuation when a balloon, which joined two stiff segments together, inflated on the outside of the device. This caused the angle between the two rigid segments to decrease.

A similar device was developed by Jeong and Konishi [16] [18]. Their device utilized the bi-directional motion that occurred when different pressures were applied to the PDMS bubble. Figure 2.5 shows the different ways that PDMS bubble actuators can move.
Lu’s device closely follows the operation in Figure 2.5(a). Jeong and Konishi’s device closely follows the operation outlined in Figure 2.5(b). The motion of these designs as analyzed by each respectively [22] [16] is used later to approximate the force produced as pressure injected into the microhand model.

2.3 Micro-haptic Sensor

To control the microhand and feed information back to the operator, a set of sensors is required to determine actual force and displacement values in the target environment. There has been work done in miniaturization of sensors to operate at micro-scale environments, which could be used in this system. Specifically, displacement could be measured via a strain gauge sensor, which is similar to the method used to measure displacement in the operator’s glove interface. Lin et al.[21] and Kang et al.[19] both describe different methods to create strain gauge sensors for use in MEMS and NEMS applications. Force
sensors will measure the amount of force being exerted on the object of interest by the microhand. Simon [29] created a MEMS capacitive cantilever sensor meant to measure force on a microhand structure.

### 2.4 Control Methodologies

A control system is needed to accurately move the microhand to the desired location. There is much research into the control of MEMS devices [10] [27], some research into the control of pneumatic devices [30], and very little on the control of micro-hand devices [17]. A combination of concepts from each of these fields was used to control the micro-hand device model.

#### 2.4.1 MEMS Control

As more research goes into the applications and manufacturing of MEMS devices, more research also occurs on how to best control these devices. Of interest in this application, Borovic et al.[10] discuss issues to consider when designing control systems for MEMS devices and Senturia et al.[27] identify issues with modeling the operation of MEMS devices and explore ways to improve.

In their work, Borovic et al.[10] make arguments that due to complexity and size constraints open loop or pre-shaped control schemes should be used unless a high degree of accuracy or a high tolerance for parameter uncertainties is needed. Borovic et al.provide the example of the control of an optical comb device, they explore the performance of the operation of the optical comb using an open loop, pre-shaped open loop, and closed loop control scheme. They assert that the specific application of this device and the requirement for accuracy in position of the object being actuated by the optical comb will determine the control scheme to be used. They also portray a balance between the complexity and real-estate required by the controller and the performance tolerance of the system. For simple actions with the optical comb, the complexity and size of a closed loop controller would
not be necessary because the performance of the system was within tolerable limits.

For the system being described in this work, the size of the controller is not constrained due to the nature of pneumatic devices. One advantage to pneumatic devices in this situation is that some components can be off loaded from the target environment [8]. Another credit for closed loop control in this situation, is the need for a high degree of positional accuracy. One of the requirements for this work is to achieve a high degree of accuracy because of its potential use in MIS.

Senturia et al. [27] suggests that one issue that faces MEMS designers and MEMS control designers is the lack of modeling strategies and tools to reflect accurately the environment MEMS devices would operate in. Some forces that act on macroscopic object interactions are so small they are negligible to the interaction. When the environment is miniaturized, these forces are no longer negligible and may alter the way the objects in the environment interact. Accounting for this would make for incredibly complex models and long running simulation. Senturia suggests Finite Element Analysis be considered for such situations. This has also been seen in other surgical haptic systems [9].

2.4.2 Pneumatic Control

Impedance control schemes are often used when controlling pneumatic actuators. As defined by Tzafestas et al. [30], impedance control is “essentially a position control scheme where force feedback is used to modify the apparent inertia of the robot seen from the environment.” They go on to describe impedance control as indirectly controlling the force exerted by actuator to reach a desired actuator position. This is particularly useful when used with pneumatic systems, because the force exerted by the actuator cannot be electrically controlled directly. Often there is a regulator between with an electrical input that can control air-flow to the actuator which affects the force exerted.

Tzafestas et al. [30] also suggest that derivative gain controllers, such as PD or PID
controllers, are not suitable for MEMS or force sensing pneumatic applications. This is because MEMS interactions are often of very small magnitudes which change quickly and which are of the same magnitude as sensor noise. Because of this quickly changing signal, derivative gain controllers would dominate the control signal and always be a high value. This would cause the controller to act erroneously. A simplified controller based on this idea is analyzed for use in this work.

2.4.3 Micro-hand Control

Jeong and Konishi [17] talk briefly about the control of their micro-finger design. Their discussion focuses on the equipment needed for control of the devices as opposed to the control theory used in the operation of the micro-finger. When used with a guide robot, movements beyond a specific range were handled by the macro-scale guide robot that the micro-finger would be attached to. Smaller scale motions would be performed by the micro-finger. Jeong and Konishi parallel this in their operator interface; motion commands by the operator’s finger would be sent to the micro-finger to be performed, and wrist, elbow, and shoulder motions would be performed by the guide robot. By using this arrangement, the micro-finger has a larger range of operation and therefore has a larger work area.

2.5 Stiffness

As stated before, the focus of this investigation will be on the modeling of micro-actuators, along with micro-sensors and the control system, as they monitor and report on the stiffness of the target object. Stiffness, as defined by Rand and Burton [25], is the resistance an object has to deformation. One of the major relationships that describes how objects change or deform due to load forces imposed upon them is Hooke’s Law. When motion is constrained to one dimension Hooke’s law can be described as follows in Equation 2.1.
\[ k = \frac{F}{\Delta x} \]  

(2.1)

In this equation \( F \) is the force being exerted on the object expressed in Newtons, \( \Delta x \) is the amount displacement that results expressed in Meters, and \( k \) is the stiffness constant expressed in Newtons per Meter.

Rand and Burton [25] go on to explain a method for measuring stiffness of cells that pull a section of a red blood cell into a pipette and measure the force exerted to suck the cell into the pipette and the distance the cell is sucked into the pipette. These values are then entered into a modified version of Hooke’s Law to obtain an effective stiffness value of the cell.

What makes biological applications more interesting when analyzing stiffness is that the stiffness value of the object is not constant [15]. In these non-linear stiffness cases, the instantaneous stiffness at a given load force is defined by Equation 2.2.

\[ C(\epsilon) = \frac{d\sigma(\epsilon)}{d\epsilon} \]  

(2.2)

Here \( \epsilon \) is the strain on the object, \( C(\epsilon) \) is the instantaneous stiffness for a given \( \epsilon \) in terms of Pascals, \( \frac{d}{d\epsilon} \) is the derivative with respect to the strain which is defined as \( \frac{\partial x}{x} \) and is unitless, and \( \sigma \) is the stress imposed on the object which is defined as \( \frac{d\text{Force}}{dx} \). [15] An example stress versus strain curve is shown in Figure 2.6. By understanding how objects deform, an estimation of how a micro-actuator will interact with a target object can be made.

The next chapter applies the concepts described in the previous sections to create a model for approximating harness at a microscopic scale. The model will employ the data associated with the microhand designs to create a robust estimate of how much force a microhand can exert upon an object.
Figure 2.6: Stress versus strain of various types of cells [1]
Chapter 3

System Design

3.1 Overview

The system was designed using a functional paradigm, where every functional component has a corresponding software component, which was modeled empirically. As described by Borovic et al.[10], By utilizing a feedback loop in the system, the microhand more reliably moves to the desired grasp. With the feedback loop, if an object being grasped by the microhand is harder than the system is estimating it to be, then the system can adjust and increase the pressure being applied to the microhand. This, consequently, ensures the microhand acts as the user intends it to, even if the object is harder than expected.

To operate the microhand, a pneumatic system is needed, which would include a way to compress air to specific pressure (such as a compressed air tank and compressor), a pressure regulating solenoid, and tubing to connect the three. The microhand includes force sensors in order to measure the pressure being applied to the object of interest. Some solenoids also include the capability to estimate the pressure within the pneumatic bubble actuators in the microhand. With this capability the angle of deflection within the microhand can also be estimated. By comparing the current position of the microhand to the position at which the user desires the microhand to be, an error value can be calculated. This error value is then used for the feedback control loop, either increasing or decreasing the air flow being fed to the microhand.

Each component needed by an actual system is modeled to ensure similar operation...
in simulation as in physical components. This was done by analyzing each individual component. Figure 3.1 shows the conceptual feedback loop model for the system. For this experiment the operator will control a glove interface device. The data from the glove device is compared against the actual deformation of the microhand and target object. The difference of these values drives a control scheme, which would generate air flow and air pressure to the micro-hand. The air flow and pressure can be converted into a force and deformation, which is compared to the operator’s new glove value. Simulink was used to model the control loop and each of the components within the loop. The method of modeling each component is described in the following sections.
3.2 Haptic Interface

As discussed previously, there are many different forms of haptic input and output devices. In this situation, it is desired to transmit the sensation of stiffness. This leads to two choices of feedback actuation, many choices for a human control device that will allow the operator to manage the position of target environment actuation, and a distinct set of sensors that will be necessary to monitor the target environment.

3.2.1 Haptic Control

A Human-Machine Interface (HMI) is needed in order to have an operator interact with the system. Since the operator is controlling a hand like machine, it would be natural to control the hand machine based on the movement of the operator’s hand. The most common way to monitor the movements of a human hand is through the use of a glove-computer interface. There are many different glove interfaces commercially available, and they each measure different hand attributes using different methods.

An Essential Realities P5 glove was used for user input to the system. The P5 glove was chosen because of the following characteristics.

- Natural way to sense a user’s hand’s motion
- Easy to use programming API
- Low Cost

Many of the glove devices, including the P5, employ strain sensors to measure the curl of the user’s fingers. The API for the P5 includes direct transformations to the system for the velocity and acceleration of the curl of the user’s fingers, which are useful for providing input to the environmental models described later. IR LEDs are also used on the glove to allow the system to find the three dimensional coordinates of the glove. The glove also processes the position data to produce roll, pitch, yaw, velocities, and accelerations of all
position attributes [6]. Figure 3.2 shows the P5 glove system.

![Figure 3.2: Essential Reality P5 Glove [6]](image)

In Donal A. Norman’s book “The Design of Everyday Things” [24], Norman describes the two elements to creating a good user interaction: a logical conceptual model and visibility of the outcome. Norman describes a logical conceptual model made of a set of affordances, constraints, and mappings, which each need to be properly defined. In this situation, the glove interfaces afford for movement in each finger. This action is constrained by the mobility of the human hand and the bending ability of the strain sensors. The most logical mapping of the finger bending affordance to the desired outcome (deflection of the micro-finger) is to map the the portion of the operator’s finger deflection directly to the portion of the microhand’s finger deflection.

The other aspect Norman describes, visibility, deals with how the operator is able to perceive the intended use of the device and the outcome of his or her actions [24]. Norman gives an example of the temperature of a refrigerator’s freezer as an example of the visibility concept. When the freezer temperature control was set to 5, an operator would expect the temperature to remain constant. Norman observed that the temperature fluctuated based on the refrigerator temperature control as well as the freezer control, making this a poor interface. The intended use of this system, given that the operator is presented with a glove interface and a micro-hand device, would be a linear mapping between the glove and the microhand. From an operator perspective, it would not make sense for the microhand to
bend to 50% when the operator is barely deflecting in the glove. For the outcome visibility, a graphical user interface provides the operator an approximate view of the effect of his or her actions via the microhand device. For the scope of this thesis, the glove interface drives a simulation which in turn drives the graphical user interface, not the actual microhand device.

3.2.2 Haptic Display

Many different forms of haptic displays were discussed in previous sections. Since the target sensation to be transmitted is stiffness, pin and balloon style actuation devices would be the most suitable methods to reproduce stiffness from the target environment. Many factors must be considered when selecting between the two methods. One critical factor between bubble and pin actuation is the amount of equipment needed and the proximity of that equipment. Pin actuation requires motors to be located in very close proximity to the tactile display, whereas balloon actuation can have solenoids at some distance away to control pressure of the balloon. Proximity is important because having motors attached to the glove to provide the haptic display would make the glove very bulky and difficult to wear and operate. Another major factor in the decision is the availability of the equipment used to make the actuators. The bubble actuation method employs a device that can be made with the same equipment as is used to fabricate the microhand devices, unlike the pin actuation device that would require many commercial parts. Because of these reasons, balloon actuation was chosen because of the availability of the equipment used for balloon actuation and lack of pin actuation equipment. Research outside the scope of this thesis is dedicated to the design and integration of a lightweight and wearable tactile display based on balloon pneumatics.

To control the solenoids, some form of electrical signal needs to be generated by the computer. Since standard computers cannot generate analog electrical signals, an interface needs to be used. A few different solutions were investigated, including micro-controllers,
FPGAs, and data acquisition systems. Ultimately, a data acquisition system (DAQ) from National Instruments [5] was selected. The DAQ was chosen because of its ease of integration, reliability, and ease of learning when the project is continued. Additionally, the DAQ allows for development on a single platform rather than on a dual platform consisting of a host computer and an embedded device. Also, the DAQ can be easily addressed from any software written on the host computer and thus eliminates the need to define a protocol.

3.3 Calibration

Calibration in this application is the ability for the system to adjust to different operators’ input. For instance, if an operator does not have the ability to drive the output of the glove interface to its maximum value, the system should be able to map the extent of the operator’s control ability to the maximum output of the system. This configuration greatly enhances the operator’s ability and also allows the system to transform the output signal of the glove interface into a form more usable by the control loop. Since the microhand is on the scale of a millimeter, and the glove interface is of a much larger scale, a logical transformation would be to use the percent of allowable deflection in the glove interface as the percent of allowable deflection in the microhand.

The active calibration was done by setting a moderate maximum and minimum value at the start of program execution. As the user flexes his or her finger in the glove interface, the maximum and minimum are replaced by the respective maximum and minimum values the operator reaches. The current value was compared against the current maximum and minimum and a percent was generated. Equation 3.1 shows the equation used to generate the percent deflection of the glove interface, which was then used for the input of the microhand control loop.
\[
\text{PercentDeflection} = \frac{\text{DeflectionCurrent} - \text{DeflectionRunningMinimum}}{\text{DeflectionRunningMaximum}}
\]  

(3.1)

### 3.4 Control Methodology

Again, by utilizing a feedback loop in the system, the microhand will more reliably move to the desired grasp. With the feedback loop, if an object grasped by the microhand is harder than the system is estimating it to be, then the system can adjust and increase the pressure being applied to the microhand. Figure 3.3 shows the implementation of the feedback loop in Simulink.

![Figure 3.3: Simulink Control Model](image)

In this situation, the goal of the controller is to have the output deflection follow the input deflection. To accomplish this goal, a simple controller is needed, such as a proportional controller. Other controllers are investigated in the Results and Analysis section.
3.4.1 Environment Modeling

In order to produce accurate results, an accurate environment model needs to be developed, including the microhand, the object to be grasped, and the interaction between the two. As described earlier, there are many techniques used to describe the way an object will deform when under load. Ideally, a method like finite element analysis [9] should be used to estimate the three dimensional deformation of the target object and the micro-finger.

Due to the complexity and the induced performance considerations, a simpler approximation by utilizing Hooke’s law in one dimension was used. One of the more common methods to measure stiffness based on the Hooke’s Law paradigm is a procedure called Instrumented Indentation (II). In II an indenter of known specific parameters is pressed into the object of interest. The indenter is pushed to a known force, and the displacement, or the distance the indenter indented the object of interest, is recorded. The force and displacement are then used in an equation to approximate the stiffness of the object of interest. [31]

A modified version of this procedure, still based on Hooke’s Law, is used in this system. Instead of using the prescribed indenter, the microhand finger is used, the force is estimated in this simulation, and the displacement generated in the control loop is used to calculate an effective stiffness with Equation 3.2 [22]. In a physical implementation of this system, force sensors and displacement sensors would be used to determine the effective stiffness of the object and to drive the visual user interface. Equation 3.3 shows the relationship that will be used to approximate the deformation induced by a load force.

\[
\text{Effective Stiffness} = \frac{\text{Force}}{\text{Deformation}} \quad (3.2)
\]

\[
\Delta x = \frac{F}{k} \quad (3.3)
\]
In this equation $\Delta x$ is the deformation, $F$ is the forced induced by the microhand, and $k$ is the hardness.

For this application an effective stiffness is suitable because both effective stiffness and $II$, are arbitrary mappings of calculated values. This method was employed by Lu [22] to estimate how stiff the micro-finger acted as air was injected into the bubbles of the microhand. Since the value does not correspond to a direct measurable value and as long as this calculation is kept consistent throughout the system, any mapping of force and displacement to stiffness can be used.

3.4.2 Solenoid Modeling

To drive the pneumatic micro-hand, a series of components are needed to supply the air flow. Such components include an air tank or compressor to be able to increase the pressure of the air in the micro-hand, a regulator to control the air flow and indirectly the pressure, and tubing to connect the air tank to the regulator and the regulator to the micro-hand. In a physical system, the electrical signal from the input translation software and controller will be sent to an electrically controlled solenoid, that acts to regulate the airflow sent to the microhand. Since the system is using feedback on the deflection of the microhand, the air flow rate does not need to monitored or used in a local feedback loop to regulate the pressure, as long as minimum and maximum pressure and airflow parameters are honored. Based on these criteria, a model can be made for the solenoid, see Figure 3.4.

![Figure 3.4: Simulink Model for Pneumatic Solenoid](image)

The solenoid can regulate only from 0 SCFM to 1.5 SCFM [7] and can also vent the accumulated pressure. Therefore the input rate is regulated from -0.5 to 1.5 SCFM. The
PDMS balloon can handle only a specific amount of pressure before the balloon bursts. Both Lu [22] and Konishi [18] gave air pressure versus force data up to 60 PSI. This is the nominal pressure for operation of the microhand according to both.

### 3.4.3 Air Muscle Modeling

Lu and Konishi both regressed the force data points they collected as they increased pressure in the PDMS balloon of their micro-finger devices. Figure 3.5 shows the Pressure versus Force data, and Figure 3.6 shows the Force versus Pressure data as measured by Konishi.

![Figure 3.5: Lu measured force versus pressure data [22]](image)

These data are included in a variable look-up table that could switch between the data sets and also between regression methods on the data. Figure 3.7 shows the user interface that allows for changing data sets and regression methods. Also editable in this window is the model’s stiffness.

By transforming the pressure value into a force value, another calculation can be made to get the deformation on the object of interest. Equation 3.1 is used to generate a value of estimated deformation. In a system utilizing an actual microhand, the force exerted on the object of interest would be measured, as would the deflection of the microfinger. Between these two values a stiffness could be estimated, as described below.
3.4.4 Graphical User Interface and Verification

A Graphical User Interface allows the operator a greater amount of situational awareness. In future iterations, the deformation information will then be transformed to produce a signal to generate a pressure from the pneumatic systems, which will in turn invoke a haptic response on the user [9].

In this simulation Matlab’s Virtual Reality Toolbox is used to create an approximation of how the target object will deform based on the operator’s input. The visualization uses the deformation calculated by the environment model and maintains the object’s volume to produce a visual interface for the operator, which can be seen in Figure 3.8.

3.4.5 Haptic Display

In this experiment, the haptic display is not an important part of the system, unlike a real system where the haptic display has a vital role in the situational awareness of the operator. In future projects, the haptic display could be integrated into the glove control interface to provide the user accurate information about the target environment. Benali-Koudja et al.[8] made a survey of different forms of haptic devices at the time of their paper. Each invoked
different sensations and has different advantages and disadvantages as described previously in Chapter 2.
Chapter 4

Results and Analysis

The following sections investigate the limitations and robustness of the proposed system. To evaluate the robustness and to find the limitations of the system, the following attributes’ effects were investigated.

- Difference between Lu micro-hand model and Konishi micro-hand model
- Sensor noise
- Stiffness of the object
- Input type
- Controller type
- Regulator limits

The primary metric that will be shown in the upcoming sections is error. The error is defined by Equation 4.1. The error can be used to derive the other metrics used. Rise time is defined in Equation 4.2. Settling time is defined in Equation 4.3. Overshoot is defined in Equation 4.4. Steady state error is defined by Equation 4.5.

\[
\text{Error(\%)} = |\%\text{Displacement}_{\text{Actual}} - \%\text{Displacement}_{\text{Desired}}| \tag{4.1}
\]

\[
\text{RiseTime (s)} = \text{Time}_{\text{SignalAt90\%OfMaximum}} - \text{Time}_{\text{SignalAt10\%OfMaximum}} \tag{4.2}
\]
\[ \text{Settling Time} (s) = \text{Time}_{\text{Signal Settles To Within 5\% Of Final Value}} - \text{Time}_{\text{Input Applied}} \quad (4.3) \]

\[ \text{Percent Overshoot} (\%) = \left| \frac{\text{Displacement}_{\text{Signal Maximum}} - \text{Displacement}_{\text{Steady State}}}{\text{Displacement}_{\text{Steady State}}} \right| \quad (4.4) \]

\[ \text{Steady State Error} (\%) = \frac{\text{Average Error}_{\text{Signal Settles To Within 5\% Of Final Value}}}{\text{Average Error}_{\text{Settles To Within 5\% Of Final Value}}} \quad (4.5) \]

### 4.1 Lu versus Konishi Model

As discussed previously, the fundamental mechanism for motion is different between the device Dr. Lu [22] created and the device Dr. Konishi [18] created. Lu’s device placed bubble actuators on the inside of the device so that upon inflation the bubbles will cause the links of the micro-finger to curl inward. This can be seen in Figure 4.1 and Figure 4.2.

Konishi’s device worked slightly differently in that the bubbles were on the outside of the device so that when inflated, the micro-finger would curl outward and then inward. This action is shown in Figure 4.3.

This is particularly interesting, because it is very obvious that the micro-hand moves in the opposite direction as intended as seen by the still increasing error after the desired position is changed. This is due to the fact that this device, as mentioned previously, moves outward, then inward. The point where the error begins to decrease is also the point that the micro-hand begins to move inward.

A base line set of metrics can be found in Table 4.1.
Figure 4.1: Percent Displacement using Lu Dataset with Soft (2) Object

Table 4.1: Base Line Performance Metrics

<table>
<thead>
<tr>
<th></th>
<th>Lu Dataset</th>
<th>Konishi Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rise Time (s)</td>
<td>0.01</td>
<td>3.3</td>
</tr>
<tr>
<td>Overshoot (%)</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Settling Time (s)</td>
<td>0.01</td>
<td>3.4</td>
</tr>
<tr>
<td>Steady State Error</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4.1: Base Line Performance Metrics
Figure 4.2: Error using Lu Dataset with Soft (2) Object

Figure 4.3: Error using Konishi Dataset with Soft (2) Object
4.2 Stiffness

The stiffness of the object of interest is one of the primary factors affecting the ability of the system to adapt to changes in desired position of the micro-hand. As would be expected, the harder the object is, the more force is needed to deform the object to the desired position, as seen by Figures 4.4, 4.5, 4.6, 4.7, 4.8, and 4.9. Table 4.2 compares the performance metrics of the different models and stiffness.

![Figure 4.4: Error using Lu Model and an Object of Stiffness=2](image)

<table>
<thead>
<tr>
<th></th>
<th>Lu H=2</th>
<th>Lu H=10</th>
<th>Lu H=100</th>
<th>Konishi H=2</th>
<th>Konishi H=10</th>
<th>Konishi H=100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rise Time (s)</td>
<td>0.01</td>
<td>0.06</td>
<td>0.60</td>
<td>3.3</td>
<td>3.6</td>
<td>4.0</td>
</tr>
<tr>
<td>Overshoot (%)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Settling Time (s)</td>
<td>0.01</td>
<td>0.07</td>
<td>0.68</td>
<td>3.3</td>
<td>3.6</td>
<td>4.0</td>
</tr>
<tr>
<td>Steady State Error (%)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4.2: Stiffness Performance Metrics

An item of particular note in Table 4.2 is that as the stiffness is increased, the total settling time of the Konishi simulations does not increase significantly. This is due to the bi-directional motion of Konishi’s device. Since the Konishi device moves in the opposite direction than the intended motion, the microhand loses contact with the object for a period of time. In this region of operation, the microhand motion is determined purely by the
Figure 4.5: Error using Lu Model and an Object of Stiffness=10

pressure of the pneumatic joints. By losing contact with the object, the stiffness of the object does not affect the motion of the microhand until microhand returns contact with with the object.
Figure 4.6: Error using Lu Model and an Object of Stiffness=100

Figure 4.7: Error using Konishi Model and an Object of Stiffness=2
Figure 4.8: Error using Konishi Model and an Object of Stiffness=10
Figure 4.9: Error using Konishi Model and an Object of Stiffness=100
4.3 Sensor Noise

Sensor error was also added to the model to make the simulated model more realistic. Two different values of error were investigated, 1 (nano-Newton for force, nano-meter for displacement) and 0.1 nN or nm, respectively, for both a soft object and hard object. The performance metrics are summarized in Table 4.3. The noise in these cases causes the performance to be slightly degraded in rise time, steady state error, and overshoot. Figures 4.10, 4.11, 4.12, and 4.13 show that as amplitude of the noise is increased, the steady state error increases.

Figure 4.10: Error using Slightly Noisy Lu Dataset
Figure 4.11: Error using Highly Noisy Lu Dataset

Table 4.3: Sensor Noise Performance Metrics

<table>
<thead>
<tr>
<th></th>
<th>Ideal Lu Soft</th>
<th>Ideal Lu Hard</th>
<th>Lightly Noisy Lu Soft</th>
<th>Lightly Noisy Lu Hard</th>
<th>Highly Noisy Lu Soft</th>
<th>Highly Noisy Lu Hard</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rise Time (s)</td>
<td>.01</td>
<td>.595</td>
<td>.01</td>
<td>.01</td>
<td>.53</td>
<td>.49</td>
</tr>
<tr>
<td>Overshoot (%)</td>
<td>0</td>
<td>0</td>
<td>6.00</td>
<td>9.44</td>
<td>2.27</td>
<td>3.20</td>
</tr>
<tr>
<td>Settling Time (s)</td>
<td>.01</td>
<td>.68</td>
<td>.02</td>
<td>.01</td>
<td>.66</td>
<td>.63</td>
</tr>
<tr>
<td>Steady State Error (%)</td>
<td>0</td>
<td>0</td>
<td>.015</td>
<td>.010</td>
<td>.0135</td>
<td>.0467</td>
</tr>
</tbody>
</table>
Figure 4.12: Error using Slightly Noisy Konishi Dataset
Figure 4.13: Error using Highly Noisy Konishi Dataset
4.4 Regulator Limits

One remedy to the loss of performance due to an increase in the stiffness of the object of interest is to find a regulator that is able to push air into the microhand faster, make the pressure in the microhand bubbles to increase, and make the finger bend faster. Three situations are presented using different limits for the pneumatic regulators, see Figures 4.14, 4.15, and 4.16. Table 4.4 shows the difference in performance using different regulators. As to be expected, the regulator that is able to produce the highest amount of air flow to the microhand is the most able to respond to change in the system.

Figure 4.14: Error using Low Flow (Limit = 1) Regulator
Figure 4.15: Error using Medium Flow (Limit = 10) Regulators

<table>
<thead>
<tr>
<th>Regulator Limit (SCFM)</th>
<th>Low Flow</th>
<th>Medium Flow</th>
<th>High Flow</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rise Time (s)</td>
<td>.06</td>
<td>.55</td>
<td>5.475</td>
</tr>
<tr>
<td>Overshoot (%)</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Settling Time (s)</td>
<td>.07</td>
<td>.68</td>
<td>6.84</td>
</tr>
<tr>
<td>Steady State Error (%)</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4.4: Regulator Performance Metrics
Figure 4.16: Error using Medium Flow (Limit = 100) Regulators
4.5 Input Type

To show the robustness of the system and the ability of the system to respond to change, different input values should be used. Previous examples set the desired displacement to 50% at a given time. A more interesting input to this system would be a sine wave, due to the increase and decrease of the desired displacement as well as the ability to evaluate the delay between the change in desired displacement and the actual displacement. The sine wave input is presented here compared to the standard step input at both a soft stiffness (2) and a stiffer stiffness (100) in Figures 4.17, 4.18, 4.19, 4.20, 4.21, 4.22, 4.23, and 4.24. Table 4.5 summarizes the performance with the various input types. With the stiffer objects (stiffness = 100) and a very dynamic input (sine wave) the model has a hard time adapting to the desired input. The high amount of error in Figure 4.25 shows that the micro-hand and controller are not able to keep up with the rate of change of the operator input.

![Figure 4.17: Error using Lu Model of Stiffness=2 with Step input](image)

The most interesting input type that could be used with the system is actual operator input. The following images show the output of the system when driven by a set of operator’s input, see Figure 4.26. The operator performed 5 repetitions of a finger curl and extension.
As seen in the figure, when the operator performed the curl or extension action, the error jumped. The model adapted to the change in approximately 3 seconds which is consistent with the settling time of the Lu microhand interacting with a stiff (100) object.
Figure 4.19: Error using Konishi Model of Stiffness=2 with Step input
Figure 4.20: Error using Konishi Model of Stiffness=2 with Sine input

Figure 4.21: Error using Lu Model of Stiffness=100 with Step input
Figure 4.22: Error using Lu Model of Stiffness=100 with Sine input

Figure 4.23: Error using Konishi Model of Stiffness=100 with Step input
Figure 4.24: Error using Konishi Model of Stiffness=100 with Sine input
Figure 4.25: Error using Konishi Model of Stiffness=100 with Sine Input Close Up
Figure 4.26: Error using Lu Model of Stiffness=100 with Glove Input
4.6 Controller Type

Many different controller types exist. One task of this thesis was to evaluate different basic controller types to see the effect on the system. Three different controller types were evaluated: a proportional controller, a PI controller, and a PD controller. Each type was tried with multiple values of gain also. Figures 4.27, 4.28, 4.29, 4.30, 4.31, and 4.32 show that as the proportional gain is increased, the settling time and rise time decrease. Table 4.6 shows the difference in performance between the forms of controllers.

![Figure 4.27: Error using Proportional Controller, P=10](image)

<table>
<thead>
<tr>
<th></th>
<th>P=10</th>
<th>P=100</th>
<th>P=1000</th>
<th>P=10000</th>
<th>P=100,I=1</th>
<th>P=100,I=10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rise Time (s)</td>
<td>.061</td>
<td>.012</td>
<td>.0115</td>
<td>.0115</td>
<td>.0117</td>
<td>.0117</td>
</tr>
<tr>
<td>Overshoot (%)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Settling Time (s)</td>
<td>.1500</td>
<td>.0337</td>
<td>.0152</td>
<td>.0137</td>
<td>.0328</td>
<td>.0764</td>
</tr>
<tr>
<td>Steady State Error (%)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1.5e-5</td>
<td>3.5e-4</td>
</tr>
</tbody>
</table>

Table 4.6: Performance Metrics of Different Controllers

As seen in Table 4.6, the controller that had the best performance, in regard to settling time, was the P=10000 proportional controller. As the proportional gain increases, the effect of the error is increased, causing the airflow to increase quicker. Even though the P=10000 controller performed best, the return on the increase in proportional gain from
Figure 4.28: Error using Proportional Controller, P=100

1000 to 10000 was diminished. The diminished return on performance is caused by the limits imposed on the system by the air flow regulator. Even though the error is being multiplied by proportional gain, the regulator can provide only so much airflow and therefore limits how fast the system can adapt. This limit would cause a controller of P=100000 to have a similar performance as the P=10000 controller. Figures 4.33 and 4.34 show the air flow usage. The proportional controller in this case outperforms the PI controller because an integral gain emphasizes the past states of the system. This causes the system to be slightly slower when adapting to change and can cause some steady state error, which is also shown in 4.6.

Any iteration utilizing a derivative gain, like PD or PID, failed to simulate, as predicted by Tzafestas et al.[30]. Tzafestas suggested that since the signal-to-noise ratio is low for MEMS sensors, that a derivative gain would be very large compared to proportional gain or integral gain. See Figure 4.35 for the partial simulation leading up to the derivative gain, which caused the simulation to become unstable.
Figure 4.29: Error using Proportional Controller, P=1000

Figure 4.30: Error using Proportional Controller, P=10000
Figure 4.31: Error using PI Controller, P=100 I=1

Figure 4.32: Error using PI Controller, P=100 I=10
Figure 4.33: Air Flow using P Controller, P=1000
Figure 4.34: Air Flow using P Controller, P=10000
Figure 4.35: Error using PID Controller, $P=100$, $I=1$, $D=1$
4.7 Summary

This chapter showed many different attributes that could be varied in this system to demonstrate the robustness of the model and also identify trends that could also apply to a physical system. As seen before, the model described is able to emulate two different forms of micro-actuators, allow for various stiffness values of the object to be grasped, allow for different inputs to be applied to the system, and incorporate the effect of noise on the modeled sensors. By testing the model with different values for the pneumatic regulators, and control scheme trends were identified that could be used in a physical system. To maintain performance of the system as the stiffness of the target object is increased then the regulator must be able to handle more pushing more air quicker than with a soft object. By testing different values of controller, it was seen that more proportional gain caused the response time of the system to decrease up to a certain point at which the controller was taxing the regulator to its limit, causing no further performance gain.
Chapter 5

Conclusions and Future Work

This thesis has shown the need and feasibility of using haptic interfaces, especially in the fields of medicine and human-robot interaction. The goal of this thesis was to formulate, model, and characterize a foundation system to simulate the control a micro-actuator component able to be outfitted with sensors able to deliver information to a haptic feedback component. Key metrics for evaluating the effectiveness of the model are the latencies associated with the simulated Matlab model and consistency of the model (as described by the lack of steady state error in simulation). As seen in the previous chapters, certain parameters in the model are more accurate and timely than others given specific inputs. For example, a very stiff object requires a strong air regulator in order for the system to adapt to input. Another example is that for all cases a high gain proportional controller is suitable to control the micro-hand based on input.

Many improvements can be made to this system. These improvements can be categorized into three areas: incorporating hardware into the system, improving model physics, or enhancing user interaction. The proposed system was designed with the intention of some day being implemented in actual hardware. The system was also designed to be modular, so that a segment could be removed and then actual hardware could take its place. Such hardware components that could be brought into the system include:

- A pneumatic system including an air compressor, regulator, and microhand
- Hardware controller
Haptic Feedback Component

There are at least two potential improvements that could be made to enhance user interaction. This thesis did not include the integration of a haptic interface system, although one was suggested. Future work could include the addition of a physical haptic user feedback interface into the system. This would allow for more situational awareness by the operator by giving the operator sensory feedback from two sources: visually and haptically. In this proposed system, a basic graphical user interface system was included. Another improvement to this system would be to develop a much more powerful graphical user interface system to approximate the target environment more accurately. An interesting topic would be how best to portray the available information across both a graphical user interface with a haptic interface system.

The third set of improvements relates to the core simulation engine of this system. Basic physical models were used to develop this system. There is a much larger field of physics based on the interaction of objects, even at a microscopic scale. This thesis scratches only the surface of what is possible with Contact Mechanics. Further investigation of contact mechanics could greatly improve the accuracy of the proposed system. Similarly, the effects of all five fingers of a hand, rather than just a single finger, could be incorporated in the model.
Bibliography


[5] High-speed m series multifunction daq 16-bit, up to 1.25 ms/s, up to 80 analog inputs, 2007.


Appendix A

P5 Matlab Integration Code

#define S_FUNCTION_NAME  p5sim
#define S_FUNCTION_LEVEL 2

#include "simstruc.h"
#include "include\P5\P5dll.h"

P5State *state = 0;

/** MATLAB GENERATED **/
static void mdlInitializeSizes(SimStruct *S)
{
    /* See sfuntmpl_doc.c for more details on the macros below */

    ssSetNumSFcnParams(S, 0); /* Number of expected parameters */
    if (ssGetNumSFcnParams(S) != ssGetSFcnParamsCount(S)) {
        /* Return if number of expected != number of 
         * actual parameters */
        return;
    }

    ssSetNumContStates(S, 0);
    ssSetNumDiscStates(S, 0);

    if (!ssSetNumOutputPorts(S, 1)) return;
    ssSetOutputPortWidth(S, 0, 1);

    ssSetNumSampleTimes(S, 1);
    ssSetNumRWork(S, 0);
    ssSetNumIWork(S, 0);
    ssSetNumPWork(S, 0);
ssSetNumModes(S, 0);
ssSetNumNonsampledZCs(S, 0);

ssSetOptions(S, 0);
}

static void mdlInitializeSampleTimes(SimStruct *S)
{
    ssSetSampleTime(S, 0, CONTINUOUS_SAMPLE_TIME);
    ssSetOffsetTime(S, 0, 0.0);
}

#undef MDL_INITIALIZE_CONDITIONS
#if defined(MDL_INITIALIZE_CONDITIONS)
    static void mdlInitializeConditions(SimStruct *S)
    {
    }
#endif /* MDL_INITIALIZE_CONDITIONS */
/** END MATLAB GENERATED **/

#define MDL_START
#if defined(MDL_START)
    static void mdlStart(SimStruct *S)
    {
        P5_Init();
        state = P5_GetStatePointer(0);
        P5_SetUnits(P5_CM);
    }
#endif /* MDL_START */

static void mdlOutputs(SimStruct *S, int_T tid)
{
    real_T *y = (real_T*) ssGetOutputPortSignal(S, 0);
    y[0] = state->finger[1];
}

#undef MDL_UPDATE /* Change to #undef to remove function */
#if defined(MDL_UPDATE)
  static void mdlUpdate(SimStruct *S, int_T tid)
  {
  }
#endif /* MDL_UPDATE */

#undef MDL_DERIVATIVES /* Change to #undef to remove function */
#if defined(MDL_DERIVATIVES)
  static void mdlDerivatives(SimStruct *S)
  {
  }
#endif /* MDL_DERIVATIVES */

static void mdlTerminate(SimStruct *S)
{
  P5_Close();
}

#ifdef MATLAB_MEX_FILE /* Is file being compiled as MEX-file? */
#include "simulink.c" /* MEX-file interface mechanism */
#else
#include "cg_sfun.h" /* Code generation registration function */
#endif
Appendix B

Active Calibration Matlab Code

```matlab
%% MATLAB GENERATED &&
function calib(block)
  setup(block);

function setup(block)

% Register number of ports
block.NumInputPorts = 1;
block.NumOutputPorts = 1;

% Setup port properties to be inherited or dynamic
block.SetPreCompInpPortInfoToDynamic;
block.SetPreCompOutPortInfoToDynamic;

% Override input port properties
block.InputPort(1).Dimensions   = 1;
block.InputPort(1).DatatypeID   = 0; % double
block.InputPort(1).Complexity  = 'Real';
block.InputPort(1).DirectFeedthrough = true;

% Override output port properties
block.OutputPort(1).Dimensions = 1;
block.OutputPort(1).DatatypeID  = 0; % double
block.OutputPort(1).Complexity = 'Real';

% Register parameters
block.NumDialogPrms = 0;

% Register sample times
```
% [0 offset] : Continuous sample time
% [positive_num offset] : Discrete sample time
%
% [-1, 0] : Inherited sample time
% [-2, 0] : Variable sample time
block.SampleTimes = [-1 0];

block.RegBlockMethod('PostPropagationSetup', @DoPostPropSetup);
block.RegBlockMethod('InitializeConditions', @InitializeConditions);
block.RegBlockMethod('Start', @Start);
block.RegBlockMethod('Outputs', @Outputs); % Required
block.RegBlockMethod('Update', @Update);
block.RegBlockMethod('Derivatives', @Derivatives);
block.RegBlockMethod('Terminate', @Terminate); % Required

%end setup

function DoPostPropSetup(block)
block.NumDworks = 1;

    block.Dwork(1).Name = 'x1';
    block.Dwork(1).Dimensions = 1;
    block.Dwork(1).DatatypeID = 0; % double
    block.Dwork(1).Complexity = 'Real'; % real
    block.Dwork(1).UsedAsDiscState = true;

% END MATLAB GENERATED &&

function InitializeConditions(block)
global thres_low;
global thres_high;
    thres_low = 20;
    thres_high = 60;
%end InitializeConditions

function Start(block)
%endfunction
function Outputs(block)
global thres_low;
global thres_high;
data = block.InputPort(1).Data(1);
if data < thres_low
    thres_low = data;
elseif data > thres_high
    thres_high = data;
end
block.OutputPort(1).Data(1) = ...
(data - thres_low)/(thres_high - thres_low);

%end Outputs

%% MATLAB GENERATED %%

function Update(block)
block.Dwork(1).Data = block.InputPort(1).Data;

%end Update

function Derivatives(block)
%end Derivatives

function Terminate(block)
%end Terminate
Appendix C

Data Selection Matlab Code

```matlab
%% MATLAB GENERATED %%
function simsfunc(block)
    setup(block);

%endfunction

function setup(block)

    % Register number of ports
    block.NumInputPorts = 1;
    block.NumOutputPorts = 2;

    % Setup port properties to be inherited or dynamic
    block.SetPreCompInpPortInfoToDynamic;
    block.SetPreCompOutPortInfoToDynamic;

    % Override input port properties
    block.InputPort(1).Dimensions = 1;
    block.InputPort(1).DatatypeID = 0; % double
    block.InputPort(1).Complexity = 'Real';
    block.InputPort(1).DirectFeedthrough = true;

    % Override output port properties
    block.OutputPort(1).Dimensions = 1;
    block.OutputPort(1).DatatypeID = 0; % double
    block.OutputPort(1).Complexity = 'Real';
    block.OutputPort(2).Dimensions = 1;
    block.OutputPort(2).DatatypeID = 0; % double
```
block.OutputPort(2).Complexity = 'Real';

% Register parameters
block.NumDialogPrms = 0;

block.SampleTimes = [-1 0];

block.RegBlockMethod('PostPropagationSetup', @DoPostPropSetup);
block.RegBlockMethod('InitializeConditions', @InitializeConditions);
block.RegBlockMethod('Start', @Start);
block.RegBlockMethod('Outputs', @Outputs); % Required
block.RegBlockMethod('Update', @Update);
block.RegBlockMethod('Derivatives', @Derivatives);
block.RegBlockMethod('Terminate', @Terminate); % Required
block.RegBlockMethod('SetInputPortSamplingMode', ... @SetInputPortSamplingMode);

%end setup

function SetInputPortSamplingMode(block, idx, fd)
    block.InputPort(idx).SamplingMode = fd;
end

function DoPostPropSetup(block)
    block.NumDworks = 1;

    block.Dwork(1).Name = 'x1';
    block.Dwork(1).Dimensions = 1;
    block.Dwork(1).DatatypeID = 0; % double
    block.Dwork(1).Complexity = 'Real'; % real
    block.Dwork(1).UsedAsDiscState = true;
end

function InitializeConditions(block)
%end InitializeConditions

% END MATLAB GENERATED %
function Start(block)
    dshandle = dataset;
    dsdata = guidata(dshandle);

    block.OutputPort(2).Data(1) = 0;
    block.OutputPort(1).Data(1) = ...
    str2double(get(dsdata.stiffness, 'String'));
    %block.Dwork(1).Data = 0;

%endfunction

function Outputs(block)
    dshandle = dataset;
    dsdata = guidata(dshandle);

    % Changes Data from mg to uN
    ludataset = [0 10 15 20 25 30 35 40 45 50 55 60; ... 0 8 9 12 14 17 34 60 77 89 107 122]'.*9.80665E-03;
    kondataset = [0 12 24 36 50 62 75;
                      0 -.05 -.07 .05 .18 .3 1.33]';
    dataset = [];
    if get(dsdata.ludata,'Value') == 1
        dataset = 0;
    elseif get(dsdata.konishidata,'Value') == 1
        dataset = 1;
    elseif get(dsdata.mckibbendata,'Value') == 1
        dataset = 2;
    end

    output = 0;
    if dataset == 0
        dataX = ludataset(:,1);
        dataY = ludataset(:,2);
    elseif dataset == 1
        dataX = kondataset(:,1);
        dataY = kondataset(:,2);
    end

    if get(dsdata.polyreg,'Value') == 1
        polytype = strcat('poly', get(dsdata.orderfield,'String'));

f = fittype(polytype);
fit1 = fit(dataX, dataY, f);
output = feval(fit1, block.InputPort(1).Data);
elseif get(dsdata.expreg,'Value') == 1
    f = fittype('exp1');
    fit1 = fit(dataX, dataY, f);
    output = feval(fit1, block.InputPort(1).Data);
elseif get(dsdata.piecewisereg,'Value') == 1
    x = block.InputPort(1).Data;
    if(dataset == 0)
        if(x <= 30)
            output = 17/30*x*9.80665E-03;
        else
            output = ((122-17)/30*x-105)*9.80665E-03;
        end
    elseif(dataset == 1)
        if(x <= 24)
            output = -.07/24*x;
        else
            output = (.33+.07)/(75-24)*x-.2582;
        end
    end
end

% Outputs the stiffness from the selection GUI
block.OutputPort(1).Data(1) = ... str2double(get(dsdata.stiffness, 'String'));
% Outputs the regressed data values
block.OutputPort(2).Data(1) = output;

%end Outputs

% % MATLAB GENERATED %
function Update(block)
    block.Dwork(1).Data = block.InputPort(1).Data;
%end Update

function Derivatives(block)
function Terminate(block)

%end Terminate
Appendix D

Data Selection GUI Matlab Code

```matlab
%% MATLAB GENERATED FROM DESIGN &&
function varargout = datasel(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name', mfilename, ...
    'gui_Singleton', gui_Singleton, ...
    'gui_OpeningFcn', @datasel_OpeningFcn, ...
    'gui_OutputFcn', @datasel_OutputFcn, ...
    'gui_LayoutFcn', [], ..., ...
    'gui_Callback', []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

function datasource_OpeningFcn(hObject, eventdata, handles, varargin)
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% --- Outputs from this function are returned to the command line.
function varargout = datasource_OutputFcn(hObject, eventdata, handles)
```
% Get default command line output from handles structure
varargout{1} = handles.output;

function edit4_Callback(hObject, eventdata, handles)
% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), ...
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit3_Callback(hObject, eventdata, handles)
% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), ...
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function stiffness_Callback(hObject, eventdata, handles)
% --- Executes during object creation, after setting all properties.
function stiffness_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), ...
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function orderfield_Callback(hObject, eventdata, handles)

% --- Executes during object creation, after setting all properties.
function orderfield_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), ...
    get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes during object deletion, before destroying properties.
function stiffness_DeleteFcn(hObject, eventdata, handles)
function stiffness_ButtonDownFcn(hObject, eventdata, handles)

% END MATLAB GENERATED %

% --- Executes when selected object is changed in regpanel.
function regpanel_SelectionChangeFcn(hObject, eventdata, handles)
switch get(eventdata.NewValue,'Tag') % Get Tag of selected object.
    case 'polyreg'
        set(handles.orderfield,'Enable','on');
    case 'expreg'
        set(handles.orderfield,'Enable','off');
    case 'piecewisereg'
        set(handles.orderfield,'Enable','off');
    otherwise
end

% --- Executes when selected object is changed in uipanel9.
function uipanel9_SelectionChangeFcn(hObject, eventdata, handles)
switch get(eventdata.NewValue,'Tag') % Get Tag of selected object.
    case 'ludata'
        set(handles.polyreg,'Enable','on');
        set(hObject,'expreg','Enable','on');
        set(hObject,'piecewisereg','Enable','on');
        if (get(hObject,'polyreg','Value') == 1)
            set(handles.orderfield,'Enable','on');
        else
            set(handles.orderfield,'Enable','off');
        end
    end
end

case 'konishidata'
    set(handles.polyreg,'Enable','on');
    set(handles.expreg,'Enable','on');
    set(handles.piecewisereg,'Enable','on');
    if (get(handles.polyreg,'Value') == 1)
        set(handles.orderfield,'Enable','on');
    else
        set(handles.orderfield,'Enable','off');
    end
otherwise
end

% --- Executes on key press with focus
% on stiffness and none of its controls.
function stiffness_KeyPressFcn(hObject, eventdata, handles)

% --- Executes on button press in writetofile.
function writetofile_Callback(hObject, eventdata, handles)
% Hint: get(hObject,'Value') returns toggle state of writetofile
Appendix E

Virtual Reality Model (VRML code)

#VRML V2.0 utf8
## V-REALM BUILDER GENERATED BASED ON DESIGN ##

DEF Bckg Background {
groundAngle [ ]
groundColor 0.8 0.8 0.8
skyAngle [ ]
skyColor 0.8 0.8 0.8
}
Viewpoint {
fieldOfView 0.785398
jump TRUE
position 0 1.5 6
description "Ball"
}
DEF finger Transform {
translation 0 1.25 0
children [
    Transform {
        translation 0 0 0
        children Shape {
            appearance Appearance {
                material Material {
                    ambientIntensity 0.2
                    diffuseColor 1 0 0
                }
            }
        }
    }
}

geometry Sphere {
radius 0.25
}

Transform {
  translation 2.23517e-008 0.5 0
  children Shape {
    appearance Appearance {
      material Material {
        ambientIntensity 0.2
        diffuseColor 1 0 0
      }
    }
  }

  geometry Cylinder {
    height 0.875
    radius 0.25
    bottom FALSE
  }
}

Transform {
  translation 0 -0.125 0
  children [
    Shape {
      appearance Appearance {
        material Material {
          ambientIntensity 0.2
        }
      }
    }

    geometry Box {
      size 2 0.25 2
    }
  ]
}
Anchor {
  children []
  bboxCenter 0 0 0
  bboxSize -1 -1 -1
  parameter [ ]
}

DEF ball Transform {
  translation 0 0.5 0
  children Shape {
    appearance Appearance {
      material Material {
        ambientIntensity 0.2
        diffuseColor 0.172755 0.263285 0.8
      }
    }
    geometry Sphere {
      radius 0.5
    }
  }
}