ACTIVE TESTING APPARATUS FOR FORCE AND DISPLACEMENT OF THE FINGERS

by

Brian Douglas Shelby

A thesis submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Master of Science

Department of Mechanical Engineering

The University of Utah

May 2013

The University of Utah Graduate School

STATEMENT OF THESIS APPROVAL

The thesis of **Brian Douglas Shelby**

has been approved by the following supervisory committee members:

Sanford Meek	, Chair	**Jan 18, 2013** Date Approved
William Provancher	, Member	**Jan 18, 2013** Date Approved
Bradley Greger	, Member	Date Approved

and by **Timothy Ameel** , Chair of

the Department of **Mechanical Engineering**

and by Donna M. White, Interim Dean of The Graduate School.

ABSTRACT

Control of a prosthetic device for amputees should be as natural as possible for optimal integration into daily use. A commonly used source of signal for the control of a prosthetic is the amputee’s own electrical activity in muscles, known as electromyogram (EMG) readings. In order for these signals to be correctly interpreted to control the prosthetic, the intended effect of the signals must be understood. A device capable of applying forces and measuring the responses of a finger along a single axis was created with the purpose of gathering data about the mechanical behavior of the hand and relating it to the corresponding EMG signals. Using force and displacement sensors, each device can quantify the behavior of a fingertip. The device is designed such that multiple can be combined into an array for testing several fingers at once, which allows the gathering of complex force and motion data for an entire hand. Data gathered with this device are presented, in which the EMG data are used to predict force, and compared with the actual force. This initial comparison shows the device’s ability to gather data which can improve understanding of the relation of EMG signals to complex motion of the fingers, which in turn will lead to a more natural control of prosthetic hands.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

The author would like to thank the following for their help and support:

Meghan Shelby

Doug and Regan Shelby

Friend and lab partner Thomas Taylor

Professors Sanford Meek, Will Provancher, and Bradley Greger

Mike Knutson

Jim Dotterweich and Dara Scher

1. INTRODUCTION

A developing field of medical devices makes use of decoded electromyography (a recording of electrical activity in muscles known as EMGs) data in order to control prosthetic devices (Musallam, et al. 2004). This technology facilitates amputees' (or possibly those with congenitally absent limbs) control over devices intended to perform lost functions. For example, a transradial amputee can still produce EMG signals in muscles that previously controlled the lost hand. Decoding the EMG signal data allows interpretation of the intended motion of the hand, and signals can be interpreted and sent to a robotic hand to perform the same function.

Using the body's own signals has the benefit of being very natural to the user, meaning that an amputee would need less time becoming accustomed to moving the prosthetic due to a more intuitive interface.

However, the correlation between EMG signals and resulting forces applied by the body is not fully understood. More complex motion of the hand and fingers leads to more complex EMG data. In order to better understand this relationship in the hand, hardware capable of measuring the individual forces and displacements of the fingers for correlation with the EMG signals is necessary. The hardware also needs to be capable of applying forces to the fingers to provide a variety of force and motion stimuli. The design, individual components, and function of such hardware are described in this paper.

1.1. Motivation

Prosthetic devices using the EMG control method are already developed and in production for simple grasping (Ottobock n.d.). However, it is more difficult to differentiate the EMG signals for more complex motion, as in a hand with fingers moving separately. With advances in motion capabilities and degrees of freedom in prosthetic hands (BeBionic n.d.), it is necessary to better understand the EMG signal data of complex motion in the fingers in order to make better use of these electrical and mechanical improvements.

A device capable of applying individual forces simultaneously to each finger of a test subject would create an environment that represents more complex grip or motion activity. Gathering more comprehensive data this way would shed light on the relationship between complex multifinger movements and their corresponding EMG signals. Note that data from this device would also be compatible with emerging technologies making use of motor signals in the brain (Ortmann and Baziyan 2007) (Zhou, et al. 2007) (Andersen, Hwang and Mulliken 2009).

1.2. Background

Devices intended to relate motion to the user's EMG signals have been created before. One device, the Dynamic Skeletal Loading System, was used to apply forces to the arm so the muscular activity of the shoulder could be measured (Meek, Jacobsen and Fullmer 1984). These data were then used in controlling a prosthetic arm.

An advance in myoelectric control of prosthetics came through proportional control of grasping speed and force (Sears and Shaperman 1991). This modification allowed for the speed or force of the grasp to be based on the interpretation of gathered EMG data. This was an improvement over the previous generation of threshold-type hands, whose grip was either fully closed or open, and moved between these states at maximum speed.

Understanding of EMG signals has advanced. There have been improvements in technology for gathering the signals, in understanding of their nature, and in filtering processes to extract more useful data (Reaz, Hussain and Mohd-Yasin 2006). This has allowed for research away from direct serial control, in which the prosthesis' user toggles between motions to be actively controlled. For example, a high signal of the forearm EMG signals could be interpreted as a signal to engage in motion, and the user determines whether the motion is in a grasping device, a wrist or elbow angle, etc. A signal from the opposite muscle group could be used to reverse the motion. However, this method of control was considered awkward and unintuitive (Farrell and Weir 2008).

A previous device, now at the University of Utah, related to the motion of the fingers used simple switches to determine if a finger had moved (Schieber 1991). These data were correlated with neural and EMG signal data (Baker, Yatsenko, et al. 2008) (Baker, Scheme, et al. 2010) (Baker, et al. 2009). This device had the benefit of tracking motion of multiple fingers, but a limitation of the device was its inability to measure the full behavior of the finger beyond whether it had reached a certain point (the switch) or not. The successor to this device, to be described on this paper, will improve on its predecessor by gathering much more detailed data, primarily real-time displacement and velocity values over the majority of a finger's range of motion and the force being exerted by the finger.

The majority of current research into EMG-based prosthetic control (and to some extent, readings taken of signals in the motor cortex) is based on pattern classification, and dates from the 1970s (Almström and Herberts 1978). It focuses on the patterns in the EMG signal associated with a certain motion. For example, a test subject is given a cue to make a fist, or to open his hand. The EMG signals that accompany this motion are gathered and analyzed, and a pattern is sought that can be used to classify similar types of signals as representing a similar motion. The primary benefit of pattern classification is “that it can extract more neural information with fewer monitored EMG signals than simple methods based only on the EMG magnitude” (Huang, et al. 2010). Another benefit is that multiple motions can be interpreted and sent to the prosthetic device. However, in surveyed literature, it was difficult to understand how the different motions (grasps, etc.) were performed. Most papers on pattern classification provided no discussion of how the motion itself was evaluated. Smith, Tenore et al. mentioned the use of a glove with joint angle sensors (Smith, et al. 2008), but most appeared to only cue the motion to be performed using a video of the motion (Zhou, et al. 2007) (Tenore, et al. 2007), if a method was mentioned at all. The lack of standard metrics for the motions makes comparison of results and decoding methods difficult (Micera, Carpaneto and Raspopovic 2010). There is no baseline for how quickly, fully, or forcefully the motion is completed. This shortcoming of typical pattern classification research would be addressed by the device discussed in this paper, which would gather force and position data.

Another difficulty found in pattern classification is difficulty in recognizing the motion of the hand when performed at different force levels (Scheme and Englehart 2011). At high force levels, the EMG signals are difficult to interpret due to tremor in the muscles, and the signals are inconsistent at low forces. The device discussed in this paper would help provide data to better understand the EMG signals by recording the force levels for comparison and analysis alongside the muscle data.

2. HARDWARE GOALS

Gathering data of the motions and forces of the fingers along a single axis is a primary goal of the device. The data to be gathered are displacement and force, for correlation with EMG signal data. The accompanying goal is to be able to apply forces to the fingers to simulate a wider range of conditions for more varied data.

The primary intended user for the device is a trained macaque monkey named Mars. While the device is also capable of interacting with a human, the design of the device must account for the size, safety requirements, and behavior of a monkey.

A goal of the system was to allow each finger to move as naturally as possible. This required the control to react quickly and smoothly with no degradation in behavior over long periods of time, as would be necessary for longer tests. Details of the work towards this goal are given in 7: SOFTWARE DESIGN.

In its application, the device may need to be used by someone with a less technical background. For this reason, a goal of the project was to increase the ease of use with a friendly interface, including switches and knobs for intuitive interaction. See the 5: CONTROL BOX AND USER INTERFACE for more information.

Due to the wide variation in available equipment in different lab set-ups, the creation of a device that is easily integrated with standard types of data gathering systems is a goal of the project.

To summarize the project goals, the device must:

- Gather real-time force, velocity, and displacement data of the fingers along a single axis
- Apply individual forces, velocities, or displacements to the fingers to create varied stimuli for wider range of data
- Be reconfigurable in order to accommodate a variety of ergonomic or functional positions
- Be designed for use with a monkey
- Be safe for test subject
- Be protected from dirt, dust, or unwanted handling
- Exhibit natural motion feel
- Feature intuitive user interface
- Be compatible with wide variety of data acquisition systems

2.1. Customer requirements

In order to meet the general project goals, the device must meet the following specific requirements:

- Measure up to approximately 45 Newton (10 lbf) load without damage to any system components
- Be capable of approximately 150 mm/s (6 in/sec) velocity
- Have approximately 100 mm (4 inches) of travel
- Be able to apply approximately 22 N (5 lbf) force

- Complete independence in each individual device in both data recording and control
- Have BNC data outputs compatible with National Instruments recording equipment
- Finger separation distance: the devices must be designed for fingers that are less than 25mm apart in order to accommodate smaller monkey hands with less distance between fingers than humans
- Number of fingers to measure: the devices need only interact with the thumb and first three fingers. In monkey hands (and to a lesser degree also in humans) the motions of the little finger are interdependent with the motion of the ring finger, thus not providing additional data.
- Finger ring diameter: the devices are to have rings to place the fingers in so that forces can be applied in either direction. The inner diameter of these rings should be 10mm in order to fit the monkey's fingers well, and be easily interchangeable if a different size is desired (i.e. for a human user with larger fingers).

3. PHYSICAL DESIGN

The physical design of the prototype had two iterations: a first to investigate the interactions of the primary components and the feasibility of meeting the motion (speed, force, etc.) goals, and a second which improved on the first with geometry that better fit the ergonomic size constraints.

3.1. Original version

A single prototype was created to gain experience with the components chosen to test their interaction and to understand the behavior of a single unit. This prototype is shown in Figure 1 and Figure 2. For scale, the four cylindrical shafts in Figure 1 for the frame are about 150mm (6 inches) long. The individual components of the prototypes are described in more detail later in 4: HARDWARE.

The single unit prototype's basic shape and function were similar to another device used for haptic feedback at the University of Utah (Sylvester 2006), although the project goals required its size to be much larger than the haptic device. Its behavior had both good and bad traits. The loose tolerances on some of the connectors made the motion very smooth but limited accuracy of sensors, especially the position sensors as the finger ring could move

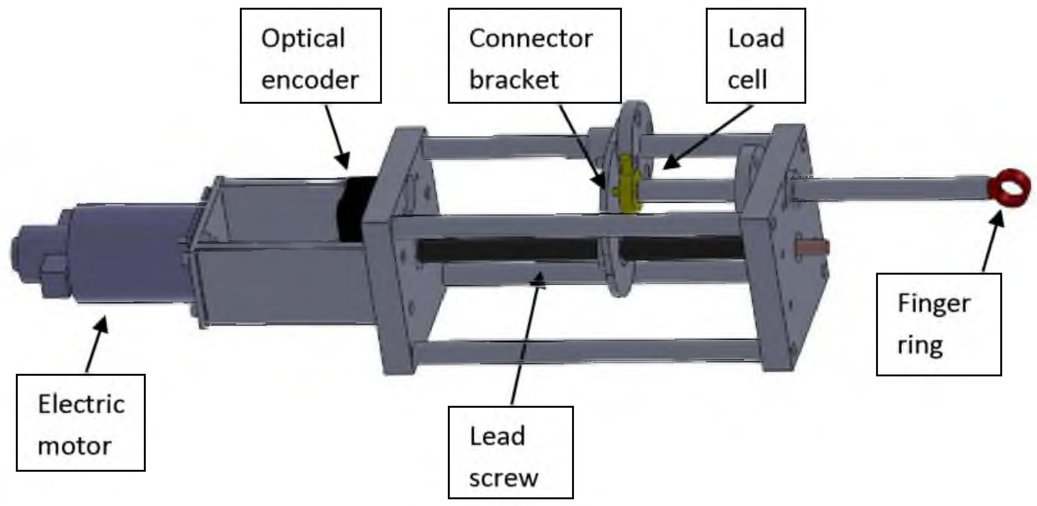


Figure 1: Solidworks model of first prototype

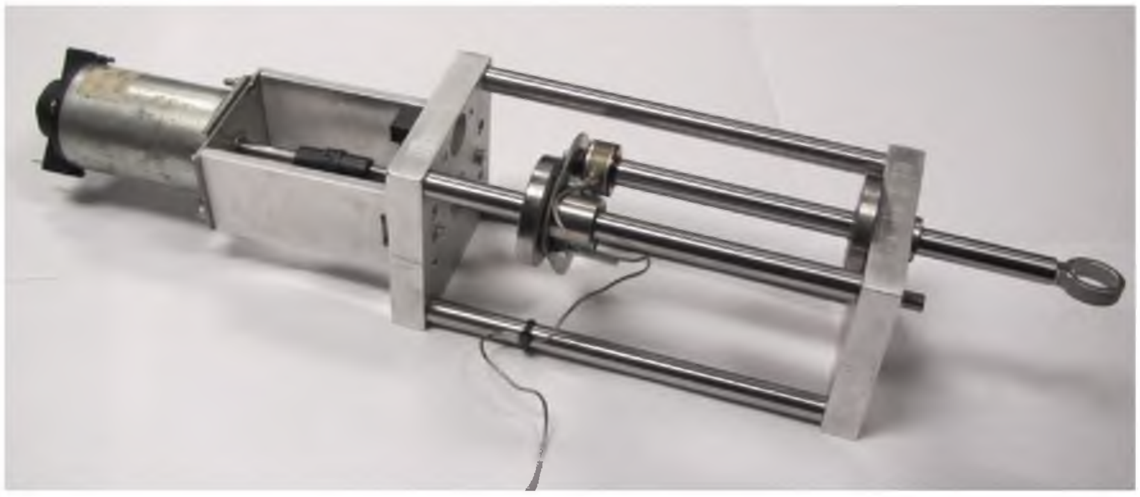


Figure 2: Photo of first device prototype

significantly before the position sensors were able to read a change.

It used the same primary components as the second iteration, namely drive system and sensors (these components are described in detail in subsequent sections). A greater understanding of the interactions of the various components was gained through this iteration.

It was controlled exclusively using a dSpace 1104 board connected to a PC. While this system gives good functionality, it is not universal. Use of a microcontroller to manage the function of the device will enable the system to be compatible with a larger variety of systems.

3.2. Current version

The second iteration incorporated lessons learned about the components from the first iteration and incorporated them in a different geometric design to meet the ergonomic and size constraints. The second system is made of up a modular design. A single unit can be used by itself, or multiple can be bolted together which brings the finger rings close together, to measure the force and location of multiple fingers. Each unit individually measures the force and location of a fingertip placed in the ring seen in Figure 3.

A computer model of the device was created first to identify benefits and disadvantages of various placements of components, as well as to easily evaluate geometrical and tolerance issues.

The basic function is as follows (see Figure 4 for a labeled diagram). A lead screw (dark spiraled shaft in Figure 3) turns and moves a nut (small cylinder concentric to the lead screw, between two hex-head cap screws in Figure 3) back and forth along its length. An electric motor is connected to the lead screw, giving it power to turn. The shaft below the lead screw is to prevent the nut from rotating along with the lead screw, allowing the rotation to turn into linear motion. Connected above the nut is the force sensing load cell, to which is attached a shaft with

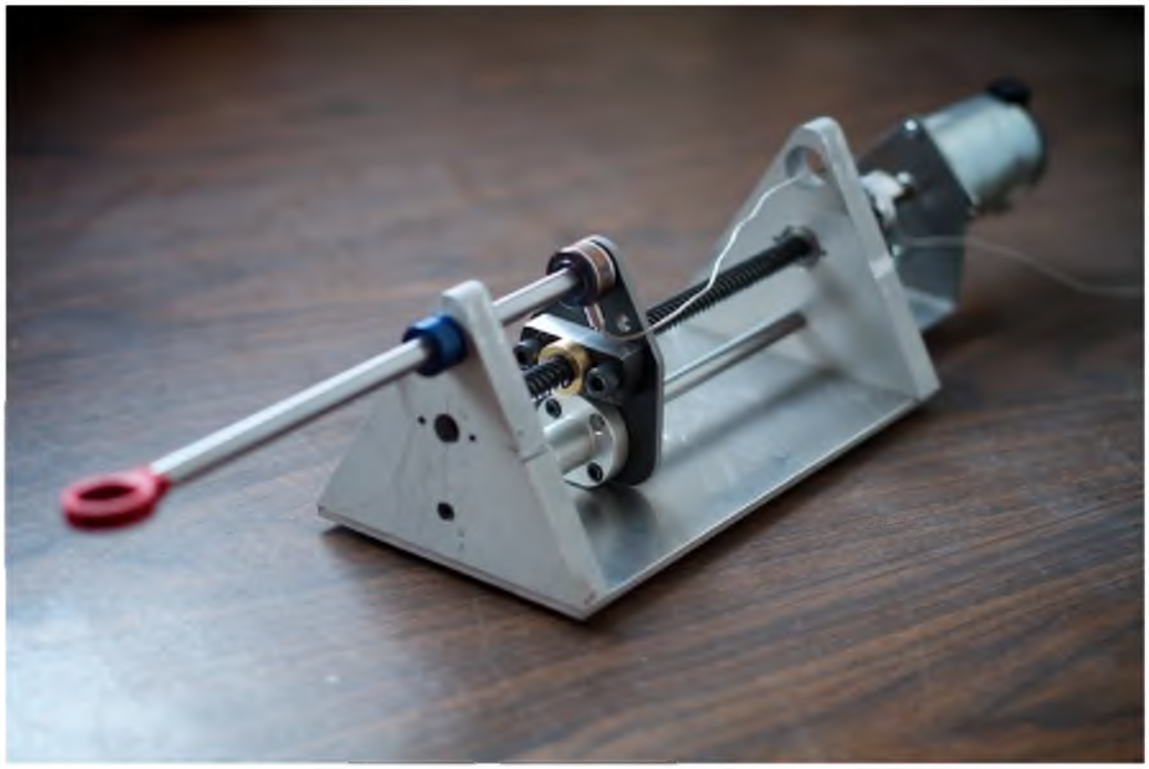


Figure 3: Single unit, mostly extended. Photo courtesy Thomas Taylor

a finger ring on the opposite end. An optical encoder senses the rotation of the lead screw and from this the linear position of the finger ring can be calculated.

The linear bearing located between the finger ring and the force sensor is not ideal for precise force measurements. This placement lessens the sensitivity of the load cell because the moving shaft has friction on it before the force is sensed. This friction can also hold the shaft at a position where it is placing a force on the load cell even if no force is being applied at the ring. This negative aspect of the design was chosen because the load cells are delicate in off-axis forces. Due to the high cost of a load cell and the very likely damage if it were placed on the user end of the shaft, the load cell was removed to a safer location with the tradeoff of some loss of sensitivity. This also removes wires and delicate parts from the reach of the test subject.

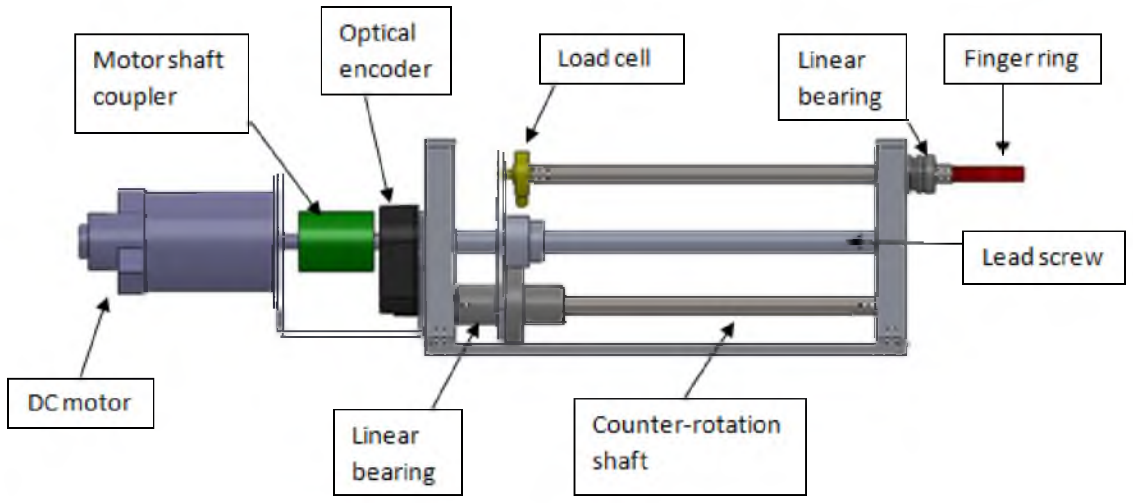


Figure 4: Side view with labeled components; solid model

The design requirement for length of travel was more than any finger would move in its full range of motion, with the hand remaining still. With the components inside the frame taking up some of the room for motion, the length of travel is still nearly 130mm (5 inches). This is enough for the expected finger range, and also allows for the user to start the ring anywhere in a range and still have enough travel for the test. The range of travel is shown visually in Figure 5 and Figure 6.

One of the most difficult aspects in the design of the device was the inability to manufacture the triangular end plates so that the shafts are aligned perfectly. If the three shafts (lead screw, top, and bottom guide shafts) are not extremely close to parallel, the device will have great difficulty traveling down the full length. If the shafts are significantly out of alignment, the device will bind and all motion is impossible. This difficulty was dealt with by leaving room for adjustment at the ends of the two smooth shafts so that they can match the angle of the rigid lead screw. While this does allow for a smooth motion, it also requires the "tuning" of the device during assembly for it to work properly. During use, if a screw is vibrated

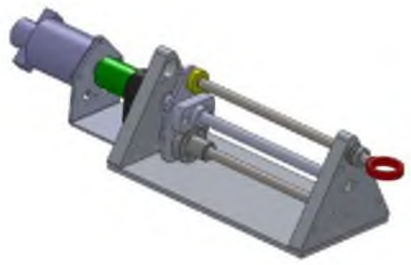


Figure 5: Single unit fully retracted; solid model

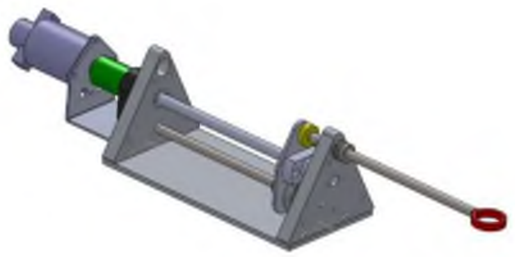


Figure 6: Single unit, fully extended; solid model

loose, it can affect the performance of the system significantly. Locking nuts and washers were used wherever possible to avoid this.

The triangular frame allows for the devices to nest into an array (see Figure 7), putting the testing rings as close as 0.8” (20mm) to each other without interfering. One of the design requirements was to have this distance be less than 25mm in order to accommodate smaller hands with less distance between fingers. Changing the shape or size of the brackets holding the devices together allows for different configurations for either different ergonomics or other functional considerations.

Bolting the devices together has the additional safety benefit of creating a barrier

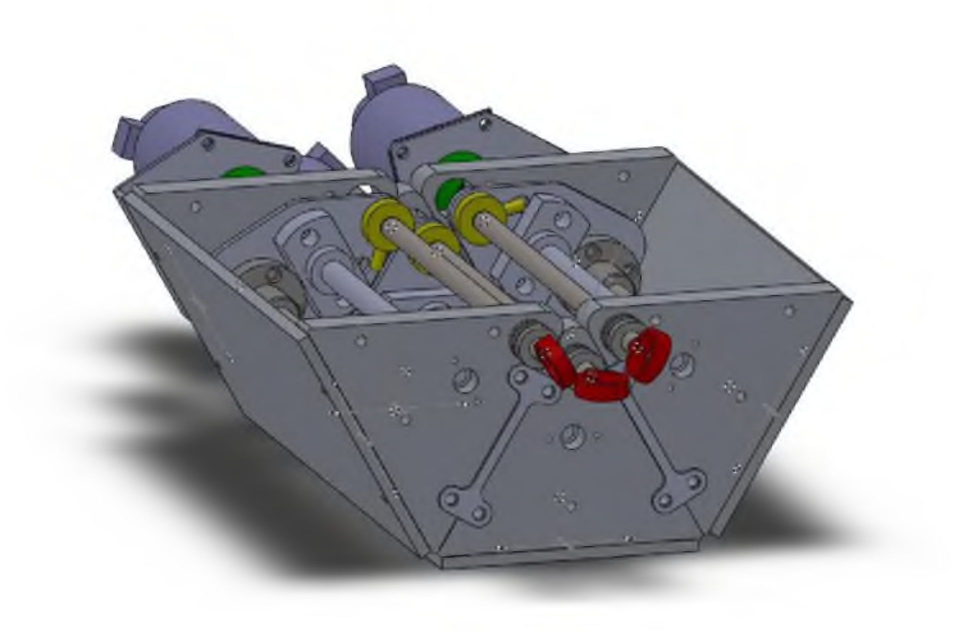


Figure 7: Three attached devices, for testing several fingers; solid model

between the testing area near the finger rings and the moving parts which could damage a hand. This natural barrier can be seen in Figure 8. With a top cover in place, the moving parts could not be reached. This would also keep lubrication on the lead screws from being thrown and dirtying the lab environment.

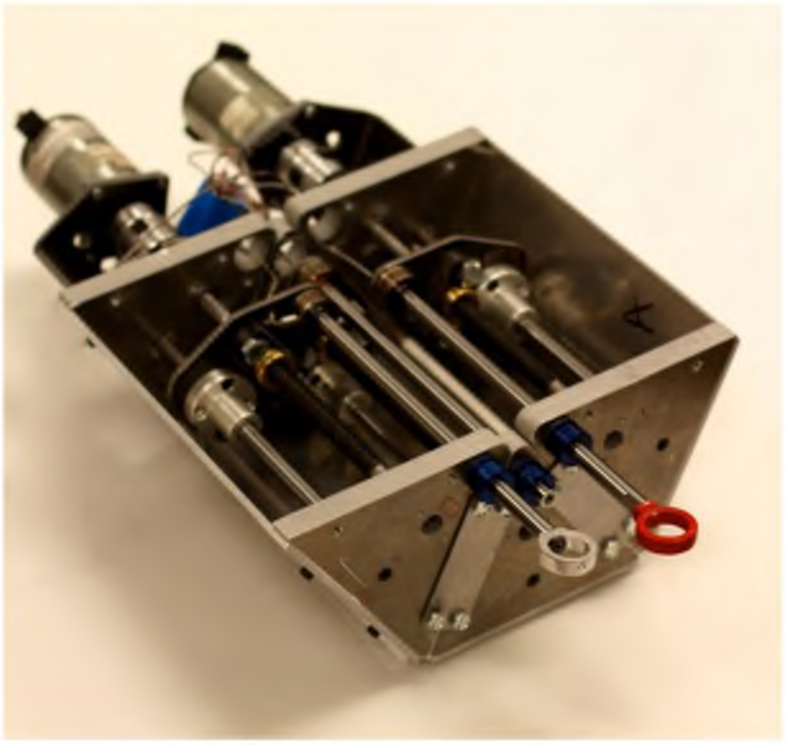


Figure 8: Three devices arranged for minimal finger separation distance; photo courtesy Thomas Taylor

4. HARDWARE

A more thorough discussion of the important components is given in this section. These were parts that were selected and purchased, rather than designed and manufactured specifically for this project.

4.1. Drive system

Due to their interdependence, the motor and lead screw (and its corresponding nut) had to be chosen together to meet the velocity and force output criteria.

The velocity requirement was 150 mm/s (6 in/sec). Since this requirement involves both the motor and the lead screw, both had to be considered as a combination. Screws of different leads were considered along with their correlation to the maximum motor speed. Lead is a measurement of how far the nut will move along a lead screw for one rotation. See Table 1 for the comparison of common leads and the calculated rotational speed of the motor to meet the linear velocity requirement. The lead which was finally selected is highlighted.

Data were taken from a lead screw manufacturer's website (Nook Industries n.d.) to calculate the torque required of the motor to provide the minimum required linear force of 5 pounds (22 Newtons). The lead screw characteristics affecting the torque required of the motor are nut material (plastic or bronze) and lead. Discussion with a company representative and Dr. Meek produced a recommendation of a multiplier of 3 to account for the loss of torque to the friction of the lead screw bearings and linear bearings. Torque calculations for a plastic nut are in Table 2, with a bronze nut in Table 3. The lead which was finally selected is highlighted in Table 3. This torque value was then used to select a motor.

4.1.1. Lead screw and nut

The lead screw used is a Nook Industries 3/8”-2 screw (Nook Industries n.d.). Its lead is 0.5 inches (13mm), meaning that the nut advances linearly half an inch for each complete rotation of the screw. The company representative quoted a screw efficiency above 50% as the threshold for a linear load on the nut itself to cause the screw to turn (Nook Industries n.d.), which is desirable for the device’s application. With the bronze nut, the screw’s efficiency is rated at 74% and can withstand a linear load of 700 lbs (3114N) while moving. The nut’s rated 0.009” (0.2mm) lash was so small as to not be detectable by feel.

An important aspect to consider in a lead screw selection is the PV value, the product of Pressure and Velocity that the nut sees. The selected manufacturer quoted 10000 (no units given) as the maximum PV value for a plastic nut, and 30000 for bronze. Using the manufacturer’s website’s PV calculator, the values in the following table were gathered, assuming a load of 10 pounds (44.5N). All of the PV values were well below the safe threshold for either a plastic or bronze nut. These PV values are shown in Table 4. A bronze nut was chosen based on a company representative’s suggestion that the plastic would loosen up due to wear sooner than a bronze nut, decreasing the accuracy of the position readings.

Another calculator from the website found the maximum recommended RPM for the lead screw. For the configuration of this project, the calculator gave a value of 31759 RPM, which is an order of magnitude larger than the selected motor’s maximum speed.

With the powerful motor and the high strength rating of the nut, the system could experience large forces if the motor were unable to be turned off quickly or driven in the wrong

Table 1: RPM required of motor to reach velocity goal

Lead	Rotations to get 6 inches	RPM
0.2	30	1800
0.25	24	1440
0.5	12	720
1	6	360

Table 2: Torque calculations for motor selection, plastic nut

	Plastic nut				
	Torque for 1lb force		With 3x friction		
Lead	in-lb	5 lb force	factor	in-oz	Efficiency
0.2	0.05	0.25	0.75	12	64%
0.25	0.057	0.285	0.855	13.68	69%
0.5	0.101	0.505	1.515	24.24	79%
1	0.199	0.995	2.985	47.76	80%

Table 3: Torque calculations for motor selection, bronze nut

	Bronze nut				
	Torque for 1lb force		With 3x friction		
Lead	in-lb	5 lb force	factor	in-oz	Efficiency
0.2	0.054	0.27	0.81	12.96	59%
0.25	0.063	0.315	0.945	15.12	63%
0.5	0.107	0.535	1.605	25.68	79%
1	NA				

direction. This was a common occurrence during the debugging phase. For safety and to prevent expensive damage to the equipment, the threads at the end of each lead screw were removed (Figure 9). This allowed the nut to “fall off” the end of the lead screw and the screw could turn indefinitely without causing damage. This proved invaluable during both hardware and software testing, preventing damage.

4.1.2. 12V DC motor

The electric motor used is a 12 volt direct current model (Herbach and Rademan n.d.). It can run in either direction, and is mounted directly to the lead screw shaft through a coupler, with no gears in between. This allows for more direct control of the lead screw due to a complete lack of gear backlash. However, geared electric motors normally generate more torque, so an ungeared motor with sufficient torque had to be selected. The torque capabilities of the motor had to be balanced with its potential speed in order for the device to be able to move quickly enough. This motor is rated to 3000 rpm and 26 in/oz (0.18 N/m) of torque at no load, and 10.6 in/oz (0.07N-m) of torque at full load.

4.2. Optical encoder

The optical encoder used is a US Digital E5 model with 500 counts per revolution (US Digital n.d.). Combined with the 0.5 inches (13mm) spiral rate of the lead screw, this allows for location data to be accurate to .001 inches (0.03mm), ignoring lash in the mechanical system.

The lash in the screw is rated to 0.009” (0.2mm), so this is the expected accuracy of the encoder’s position sensing. A model with a pass-through hole was chosen to allow the encoder to be mounted in between the threaded part of the lead screw and the motor coupler. The disk is placed directly on the lead screw shaft and held in place with a set screw. This allowed

Table 4: PV values for screws of different leads

Lead	RPM	PV Value
0.2	1800	6290
0.25	1440	5032
0.5	720	2516
1	360	1258

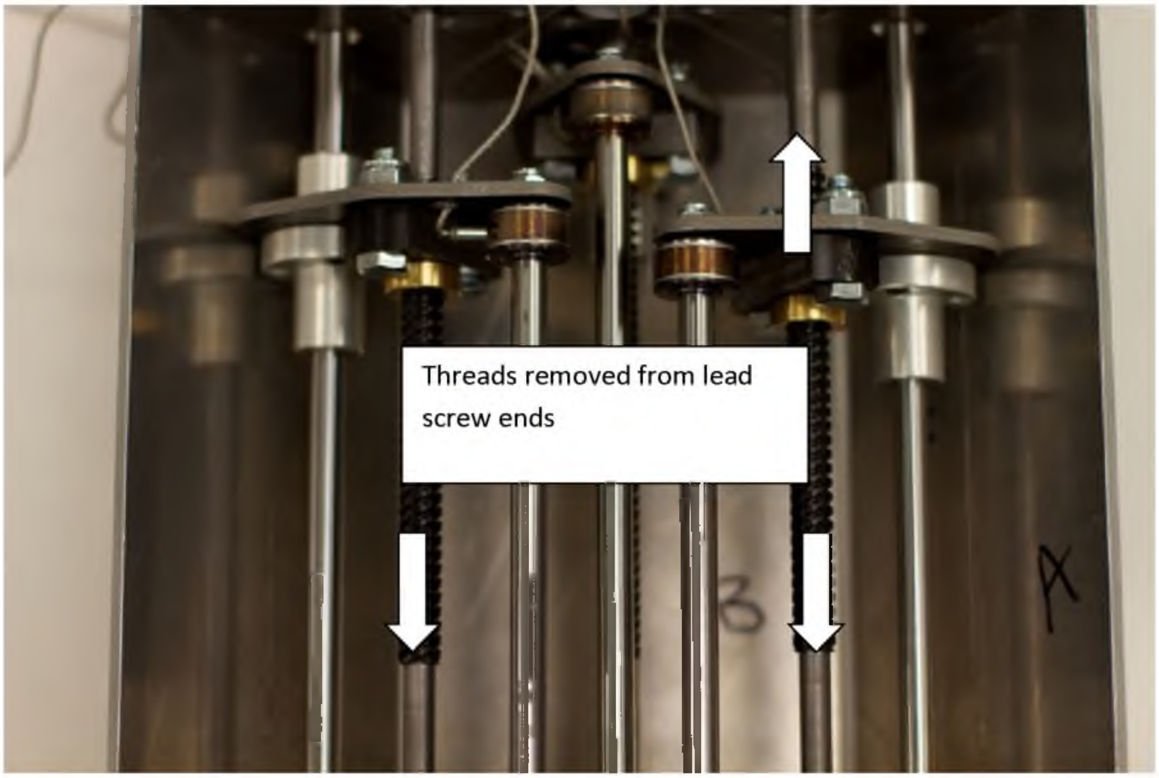


Figure 9: Threads removed from lead screw ends to prevent damage; photo courtesy Thomas Taylor

the motor to sit on the end of the lead screw and eliminated gear lash with a direct drive coupler.

Its electrical output has two channels, allowing for determination of rotational direction. It also has an index channel to easily count the number of full revolutions, which can be useful in avoiding memory overrun in the controlling computer or microchip.

The position calculated with the encoder's readings was compared with a measurement of the physical displacement of the system, using a dial indicator. A distance of one inch was measured in either direction because that was the limit of the dial indicator. Correlation was excellent, showing a measured accuracy greater than the 0.009” (0.2mm) expected from the limitations of the lead screw. The results are shown in Figure 10.

4.3. Force sensor load cell

The force sensor used in the device is a Honeywell Model 31 miniature load cell, 10 pound (44.5N) rating (Honeywell n.d.). It is capable of measuring forces in both tension and compression, which not all load cells are. Since the wrist is to be constrained for the test, this load rating met the design criteria for force and sensitivity. The voltage changes from the load cell are quite small, requiring the signal processing described in Section 6.6 Load Cell Signal Processing.

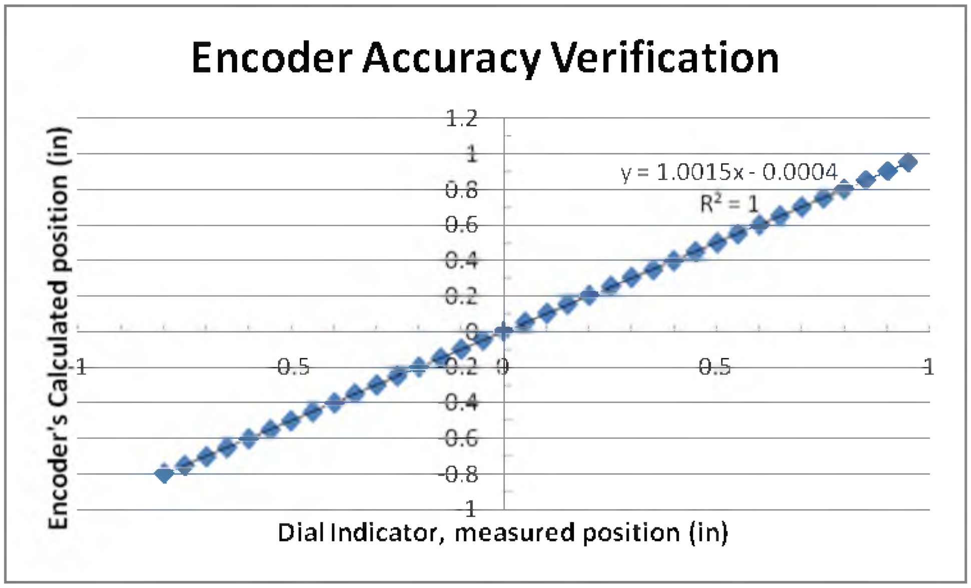


Figure 10: Encoder's position reading vs physically measured position

As discussed previously, this model of load cell is delicate if the force is not applied along the single, designated axis. Damage to this instrument was prevented by limiting the angle at which the testing shaft approaches the force sensor. In the axial direction, the load cell is rated to 50% above its nominal rating, or 15 lbf (67N).

4.4. Linear bearings

The linear bearings used in the device are Pacific Bearing Company Teflon-lined dry bearings (PBC Linear n.d.), sliding along hardened and polished steel shafting. They have reduced friction to maximize the smooth motion of the system, and need no lubrication, which lessens maintenance requirements. The lower bearing is double-width to counteract the moment on the nut when a force is applied on the testing shaft.

The diameter of the shafting was reduced for the second prototype, from 3/8" to 8mm nominal diameter. This decrease in size also reduced weight that the motor had to move to respond to the command inputs, with the intention of speeding up the response of the system.

4.5. Servo drive cards/power supply

Early testing with an h-bridge showed that the motors could draw 3 amps of current for heavy loads of short duration. They would heat up quickly but were not damaged if the high current flow was for a short period. However, the heat generated in the h-bridge itself was excessive and troublesome if the device was used continuously for longer periods of time. A more capable solution was sought.

The AMC BD15A8 analog servo drive (see Figure 11) was chosen for its ability to handle the motor's current and voltage requirements easily, its use of a PWM input, and common

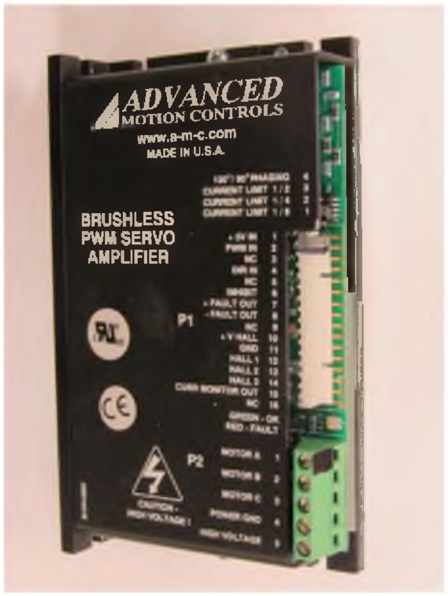


Figure 11: The AMC BD15A8 analog servo drive

usage of AMC's other servo drives in the university lab leading to familiarity with their integration (Advanced Motion Controls n.d.).

Supplying the servo drives with enough current was difficult. The AMC PS4X3H24 power supply (see Figure 12) was chosen because of its ability to supply 12 amps (3 for each motor) and the convenient mounting location for the four servo drives on the same plate as the power supply (Advanced Motion Controls n.d.). This brand of power supply has had issues with supplying clean DC power in other applications in the lab, but this particular model proved reliably stable when tested on several occasions.

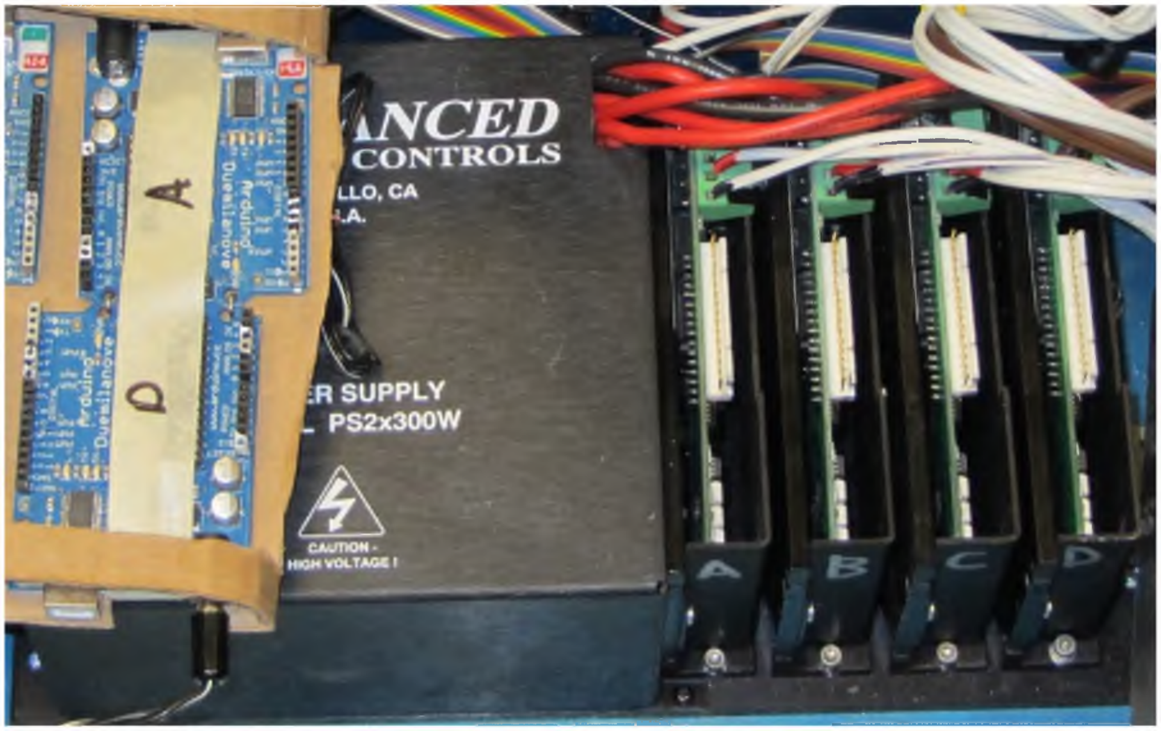


Figure 12: The AMC PS4X3H24 power supply

5. CONTROL BOX AND USER INTERFACE

All data reading outputs, user-controllable inputs, and control switches are on the front panel. All cables running directly to and from the devices attach to the back panel of the box. Power inputs (both AC and DC) also connect to the back of the box. All switches on the control box select a mode or function by moving the end of the switch towards the label stating the desired mode.

An important consideration for the control box was the organization and delineation of the features. For this reason, connections with a similar purpose were grouped together. Markings were used to separate sections of similar features. Labels were placed to describe features easily.

5.1. Control box front panel features

The control box's front panel is shown in Figure 13, and the numbered components are discussed afterwards. The labels on the front panel itself are described by the numbers in the corresponding list. Box features are listed in the order corresponding to the labels in Figure 13.

1. Run mode indicator LED: When illuminated, indicates that the microcontrollers are running the control code.
2. Data outputs: Connect BNC cables to read force and displacement data from the four devices. Each device has one output for the force reading (top row) and two outputs for the two encoder channels. For information on how to read and interpret data gathered from these sources, refer to Section 9.2.

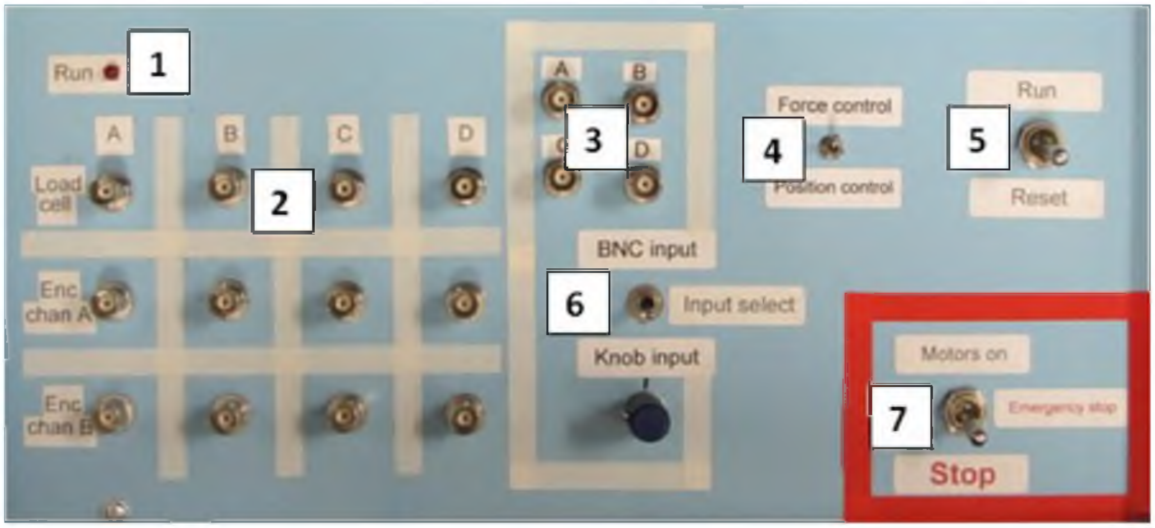


Figure 13: Control box front panel

3. Desired force inputs: BNC jacks take in a voltage from 0-5V and tell the microcontrollers what force to apply to the finger. Each of the four devices has its own high-impedance input. For information on what input voltage will give what force or position, refer to Section 9.2.
4. Operation mode switch: This switch selects the program the devices will run. If Force control is selected, the devices will run a program that allows the devices to follow a force input chosen by the Input select switch (#6). The Position control mode moves the devices to a location determined by the Input select switch (#6). The position of this Mode switch is only detected once, when the device is put back to Run mode from Reset mode (#5). In other words, a change in this switch can only have an effect on the system if the system is put in Reset mode during or after the change in program. This is due to safety issues related to the device suddenly snapping to a new position when switched from Force control mode.

5. Run/Reset switch: This switch tells the microcontrollers to start running their selected program, or to wait in idle mode. This switch can serve as a stop switch for non-emergencies as it causes the microcontrollers to stop sending movement commands to the devices.
6. Input select switch: Selects the source of input for the devices. When in Force control mode, the devices will seek a force based on the selected input. In Position control mode, the devices will move to or hold a position based on the selected input. Moving the switch to the upper position enables the BNC inputs for each separate device (For information on what input voltage will give what force or position, refer to Section 9.2. Moving the switch to the lower position uses the input of the potentiometer (pot knob) for all four devices. When the pot is used, all devices receive the same input signal, which is useful for setting the devices to a desired starting position. When the knob is set to vertical (or 12 o'clock) in Force control mode, the devices will move to where there is no force placed on the rings, and moving the knob to one side of center tells the devices to seek a force in one direction. Putting the knob to the vertical position makes it easy to set up the devices for a test, as the motors will assist the motion. When the knob is set to vertical (or 12 o'clock) in Position control mode, the devices will move to their initial position, where they were when the Reset switch (#5) was set to Run. Unlike the Operation mode switch, the input chosen by this switch is continually monitored, and can be changed without resetting the devices. Be careful when switching the input while the device is running, as a sudden change in input can cause the devices to move suddenly.
7. Emergency stop switch: This switch completely cuts power from the devices' motors, preventing them from moving. It is marked clearly in red and set up so that the more

natural motion of hurriedly hitting a switch downwards will cut power to the motors. Note that use of this switch should be accompanied by a toggle of the Run/Reset switch (#5) because the microcontrollers are still calculating motion for the devices while the Emergency stop switch is engaged, and returning power to the motors may result in a sudden jump of the finger rings. There is also the possibility of installing a remote Emergency stop switch so that it could be closer to the operator.

5.2. <u>Control box rear panel features</u>

The control box's rear panel is shown in Figure 14, and the numbered components are discussed afterwards.

Box features are listed in the order corresponding to the labels in Figure 14.

8. Motor power outputs: These banana cable jacks are the output to the devices' motor power cable. Each device A-D requires both a red and black cable to be connected. Be sure to match the red plugs to red jacks and black plugs to black jacks or the devices will run backwards and not behave correctly.
9. Encoder signal input: This input takes in the signals from the devices' encoders as well as sending necessary power to the encoders.
10. Load cell signal input: This input powers and receives signals from the devices' load cells. Although this input has the same shape and size as #9, they are not interchangeable to prevent damaging the devices by supplying power incorrectly.
11. DC power inputs: These jacks take in power from a regulated power supply for the control circuitry. 5V and ±15V are necessary for the circuit board, and 9V supplies power to the Arduino. The ground for both of these supplies should be the same, which is most easily achieved by connecting the black supplies of the 5V and 9V together with the

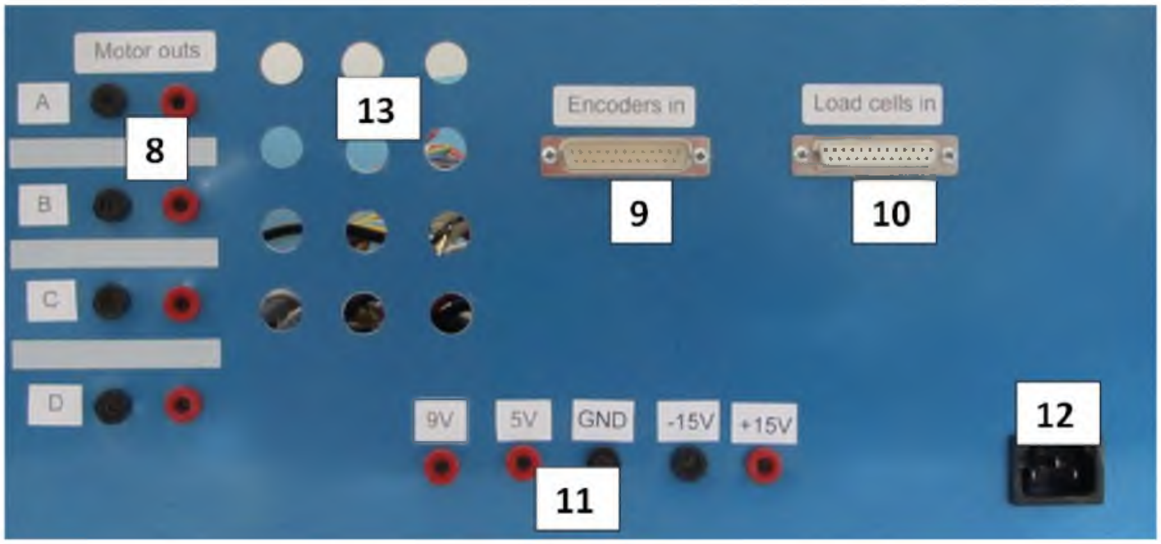


Figure 14: Control box back panel

common channel of the ±15V supply with banana cables outside the control box.

12. AC power input: This jack takes in 120V AC power to run the motors. The cable is the same as that used by most desktop computers. The jack is located far from other components to minimize 60 Hz noise transfer.
13. Ventilation holes: Allow air to enter and exit the control box so heat from electronics will not build up excessively. No overheating issues were ever encountered in testing, but these holes should still be kept clear for ventilation and cooling.

6. CONTROL BOX INTERNALS

The control box is designed so that it is unnecessary to open it for normal use. Changing elements or code could result in the need for extensive testing to restore functionality to the system. However, should the need arise to change the internal parts (such as replacing a burned-out op-amp), or for modification to the system, all components are documented in this section.

6.1. Location of components

The internal components are carefully arranged for electrical and space considerations. An internal view is shown in Figure 15.

The box interior with all components in place and connected is shown in Figure 16. Circuit board, shielding, and signal cables have been removed to show the components underneath better.

6.2. Instrumentation amplifiers

The first stage of amplification for the load cell signal is an Electronic Innovations Corp EL-1040 Instrumentation amplifier (Figure 17) (Electronic Innovation Corp (EIC) n.d.).

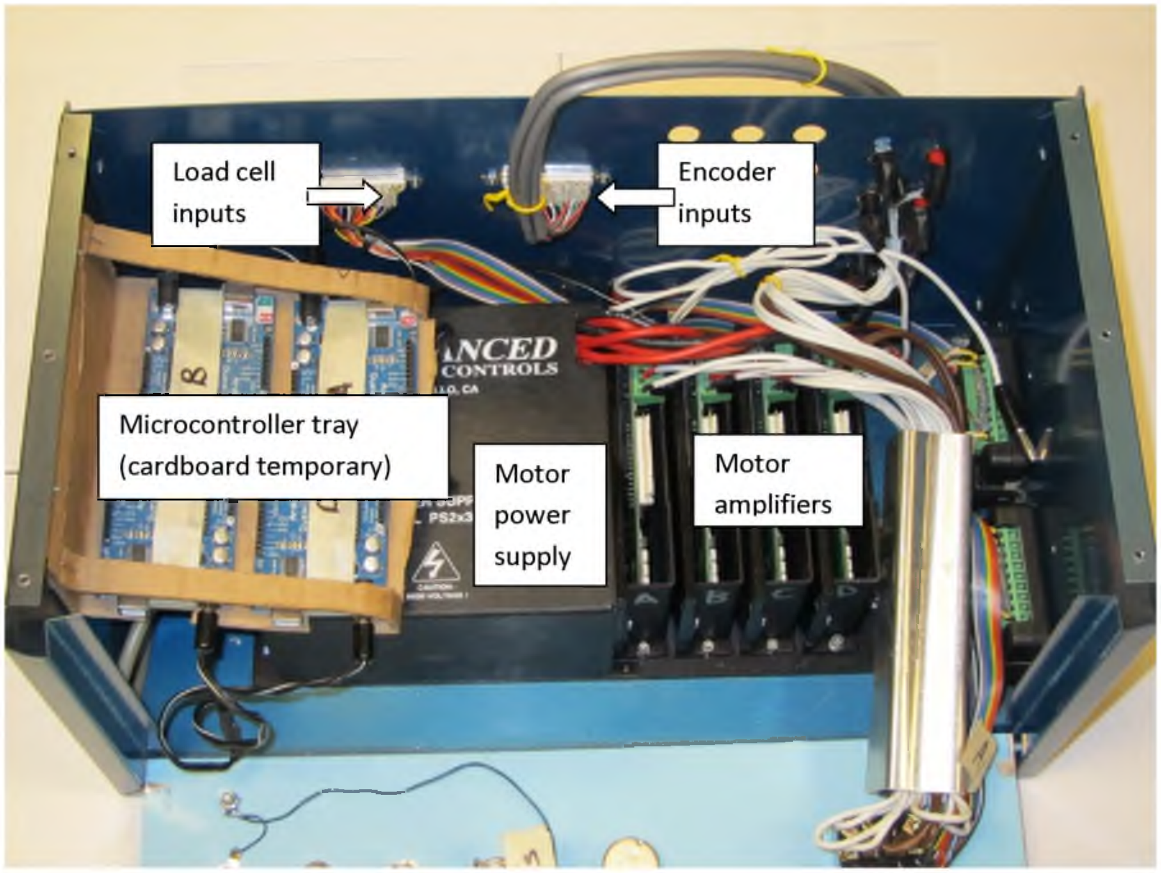


Figure 15: Internal view with circuit board, shielding, and signal cables having been removed

This amplifier is set to amplify the very small load cell signals by a factor of 1000. Note that each amplifier serves two load cell signals. These amplifiers were selected due to a positive result in other university labs, as well as their compact form factor.

The mounting location for these amplifiers is shown in Figure 18.

The signal wires that connect to the EL-1040 are all connected to headers so that they stay in the correct order. Each header is marked with labels corresponding to the labels on the EL-1040 so that they are easily oriented when connecting.

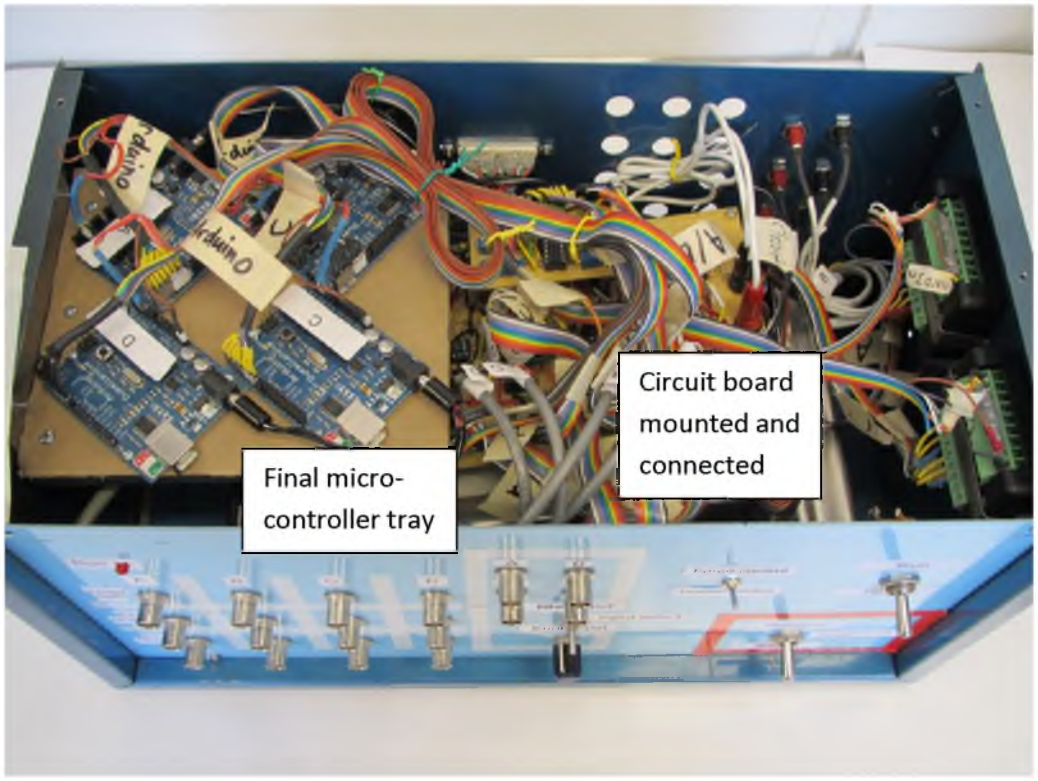

Circuit board mounted and connected
Final micro-controller tray

Figure 16: Internal view showing all components present and connected

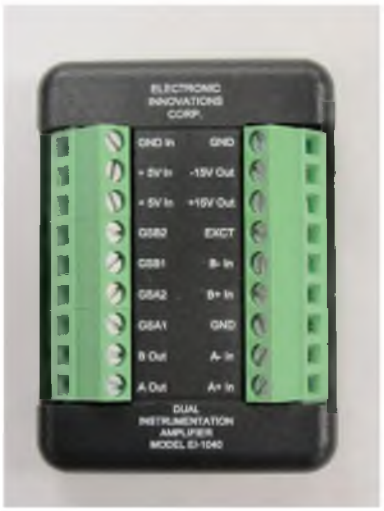


Figure 17: Close up of EL-1040 Instrumentation amplifier

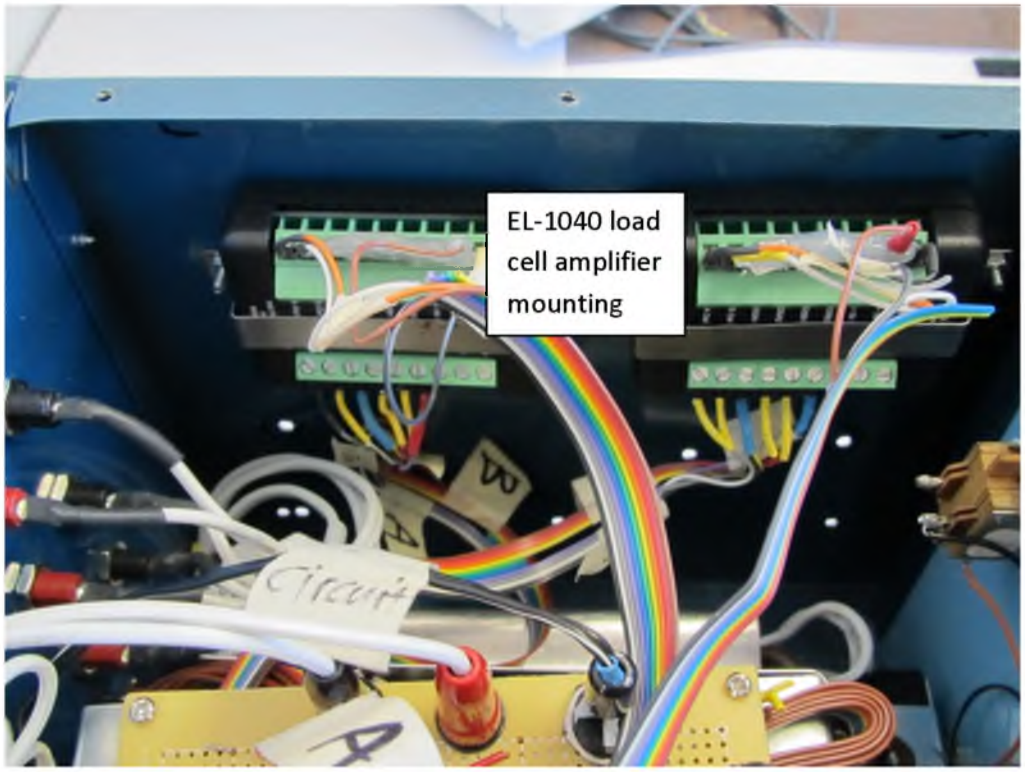


Figure 18: Mounting location of EL-1040's inside box

6.3. Internal shielding

The devices inside that are more likely to introduce electrical noise into other components have been shielded (see Figure 19). This includes a shielding cover for the large power supply, and a cover for the motor power cables where they run close to other circuitry.

Also, the 120VAC cable powering the motor supply is on the far end from other cables to minimize introduction of 60Hz noise into the DC components.

6.4. Microcontrollers

The final microcontroller chosen is an Arduino Duemilanove microcontroller with ATMega328 chip (Figure 20) (Arduino IDE n.d.). It was chosen for its ease of use, functionality, and wide user base. Much of the initial microcontroller work was done using a Microchip

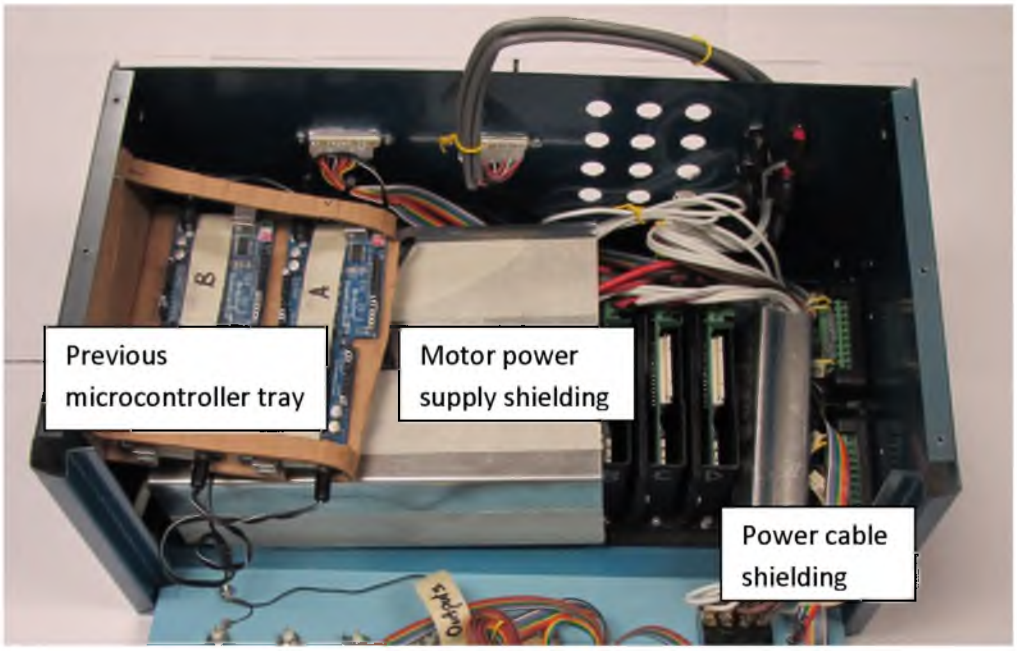


Figure 19: Internal view showing shielding over power source and PWM power cables

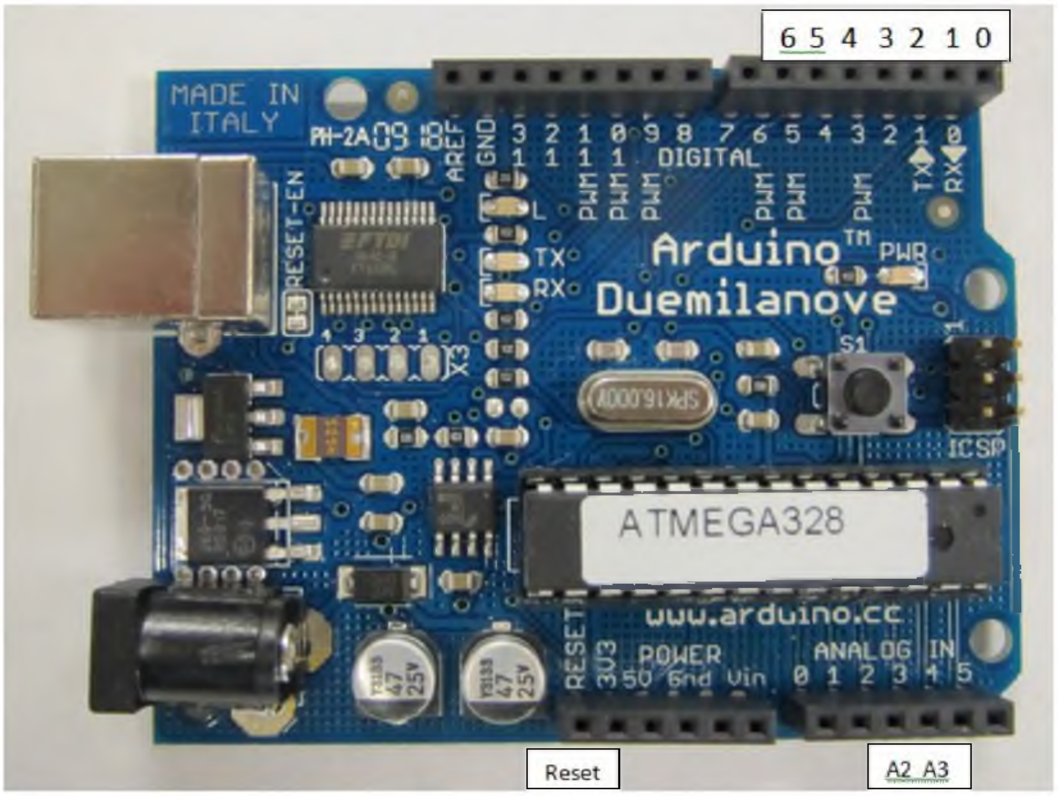


Figure 20: Arduino Duemilanove microcontroller with ATMega328 chip

dsPIC30F4011 microchip (Microchip n.d.). This was initially chosen due to familiarity gained in lecture and lab work for the ME 6960 Advanced Mechatronics class, but was replaced with the Arduino due to more universal familiarity and a wider user base on campus.

As configured, each microcontroller uses two analog sensors (load cell and BNC input reading) and three digital sensors (2 encoder channels and mode sensor). Each has two digital outputs, a PWM signal and a direction signal that go to the servo drive.

The pins of the Arduino that have electrical connection to the circuit board are described below, and are labeled in Figure 20.

0. Rx – receive pin, no programmed use. Note that this pin must be disconnected in order to upload new code to the Arduino.
1. Tx – transmit pin, no programmed use. Note that this pin must be disconnected in order to upload new code to the Arduino.
2. Encoder Channel A
3. Encoder Channel B
4. Detects position of mode select switch, determines control program
5. PWM signal out to servo drive
6. Direction signal to servo drive

A2. Load cell reading

A3. Input reading

Reset. Puts the microcontroller in reset (idle) mode, where no program is running.

The four microcontrollers used in the device are mounted together on a tray for easy access and connection (see Figure 21). Note the colored pins for easy connection of cables and correct orientation, important for safety of components.

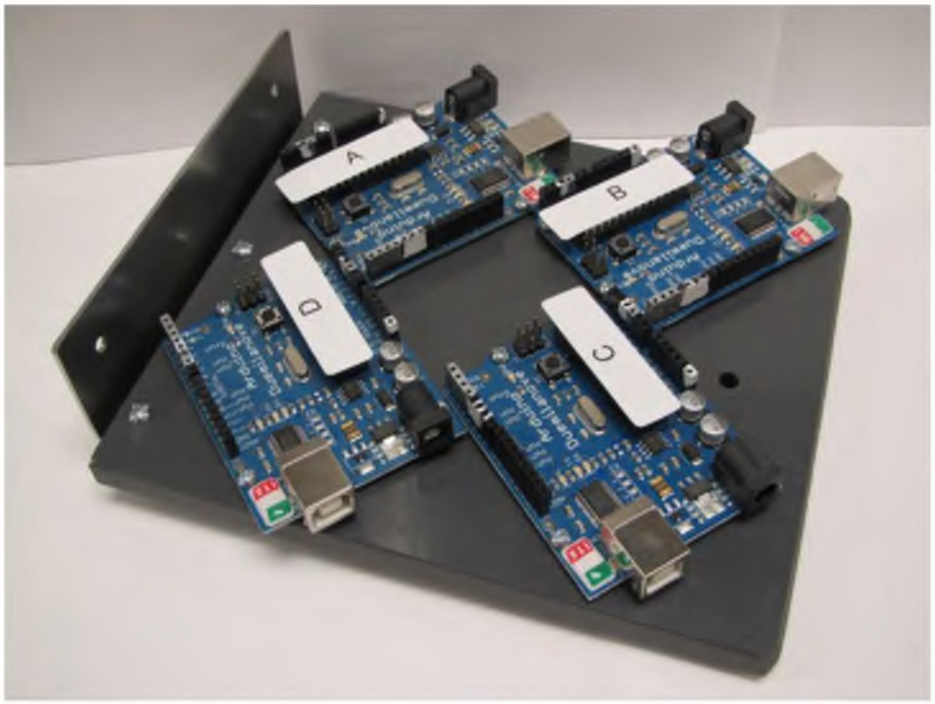


Figure 21: Nonconductive tray for microcontrollers. Note colored pin sockets for correct cable connection/orientation.

6.5. <u>Circuit board</u>

The circuit board shown in Figure 22 serves as a central location for all signals to gather. The inputs from the load cells, encoders, and front panel all connect to the circuit board and receive their 5V power here. The outputs to the front panel and the servo drives come from here. All signals in or out of the Arduino are gathered here as well. The circuit also houses the three op-amp stages for the load cell signal conditioning. All cables connecting here are marked with colored ends to make their orientation to the board clear. This is important to prevent potential damage caused by incorrect power connections.

For a complete diagram showing every individual wire's connection for each cable input, as well as the connections to the op-amps, please see APPENDIX A: LABELED CIRCUIT PHOTOS.

6.6. Load cell signal processing

Since the force sensing load cell has such a small change in signal, it needed to be amplified. It also needed to be modified further to correspond to the Arduino's analog range. The amplification circuitry diagram is shown in Figure 23.

In Figure 23, a signal generator represents the load cell. Its first step to become a more useful signal is the EL-1040 Instrumentation Amplifier, set to a gain of 1000.This gives the load cells a much larger output range, approximately -7 to 7V and centered close to zero for load cells A,B, and C. The load cell for device D had a lower output range with more bias, although the company tech support representative said its impedance readings were in tolerance. Since the load cells did not amplify equally, the signal then passes through an adjustable-gain amplifier circuit. Each signal was individually tuned so that all the outputs were a similar range, within 0.01V change for a given load. At this stage, the signal is sent to the front panel jacks to be read when it is largest.

The remaining stages optimize the signals for the Arduino 0-5V analog-to-digital converter (ADC). Before these stages, the signal still centered on zero volts, fluctuating above or below depending on the direction of the force applied. This possibility of negative voltage being applied to the microchip was a problem as the chip is not rated for it. This ADC can only read from 0 to 5V, so the signal needs to be shrunk and shifted to the center of the range. To solve this problem, a summing amplifier is used, adding an adjustable voltage to the signal. 2.59V was found to be a truer center than the expected 2.5V by reading the data output in a print statement. Each load cell's signal is individually adjusted to the 2.59V center value because each one is centered on a slightly different value. The summer amp inverts the signal, so it is followed by an inverting amplifier which also reduces the range of the signal so more of the range can be detected by the Arduino.

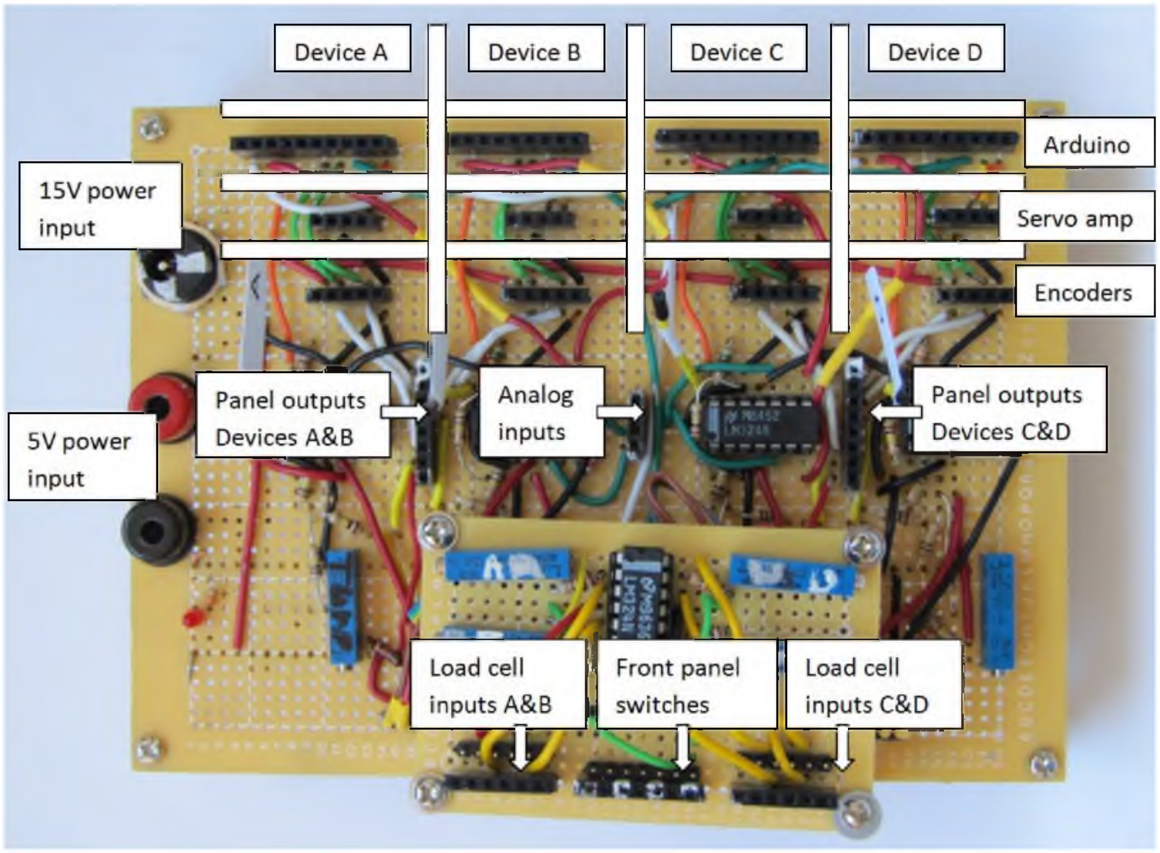


Figure 22: Top view of circuit with labeled connection spots

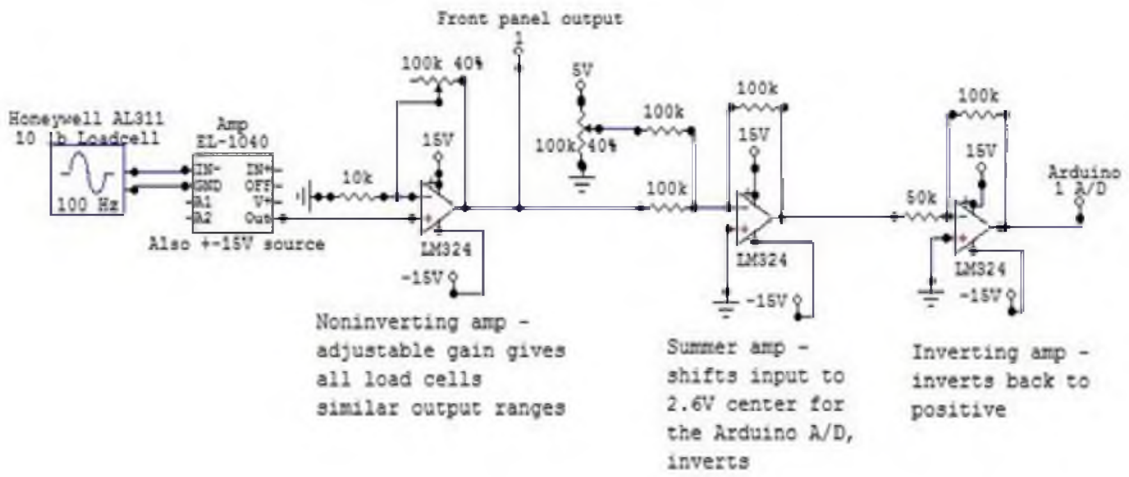


Figure 23: Signal processing sequence for each load cell

The range is reduced in order for the Arduino to read with better resolution, resulting in that the limits of the A/D correspond to approximately -7 to 7 pounds (-31 to 31 Newtons). It was a difficult design decision striking a balance between making the signal have a larger range to take advantage of the limited resolution of the A/D converter, but still leave the signal small enough that a useful range of force can be detected.

The two places in which the circuits are adjustable are shown in Figure 24. Neither should need adjustment. The first amplification should be adjusted with care as calibrating for all four load cells is a time-intensive iterative process. The second is easily measured at the marked locations and adjusting it does not affect other parts of the circuit.

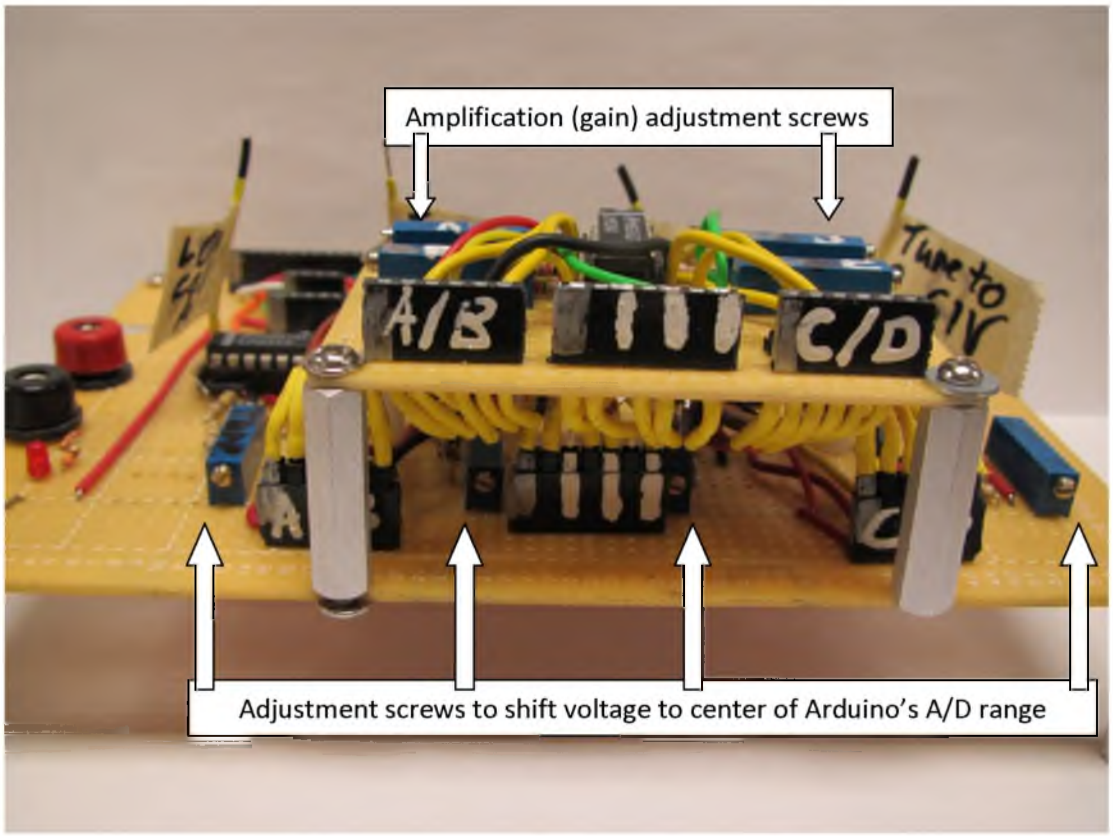

Amplification (gain) adjustment screws
A/B
C/D
Tune to 5V
Adjustment screws to shift voltage to center of Arduino's A/D range

Figure 24: Location of circuitry adjustment screws on circuit board

7. SOFTWARE DESIGN

7.1. Controller design

Each of the two modes discussed in Section 5.1 uses a Proportional control (P control) scheme to run the devices. P controllers use error between expected results and actual results to drive and continually correct the output. The different basis for error for each mode is contained in the name of each mode, i.e. Position control the actual position to the desired, and any difference is error.

Both modes have a scaling start up feature for safety. This mode limits the output of the motor for the first second of function to start more smoothly. Further discussion of this feature is given with an example in Section 8.2.

7.1.1. Force control

The Force control program only uses the load cell readings to determine the behavior of the system. As the user pushes and pulls on the finger ring, the motors assist in moving the ring to make the force on the load cell match the given input. A target load is given to the controller through the front panel inputs, either the potentiometer or the BNC inputs. The difference between the target load and the actual load is calculated to give the error. This error is multiplied by a proportional constant (gain) to give the desirable size characteristics before the error is converted into a physical motion in the motor.

The Force control program senses the force on the ring, reads the input (desired) force, and updates the signal controlling the motor at a frequency of 2.9kHz, or about every 3.4 milliseconds.

Steady-state error was found to be a near nonissue, so no integral gain was added. The load cell readings had natural fluctuations due to their high gain, and these changes proved too noisy for derivative gain, causing instability.

The largest limiting factor with this controller came from the hardware and how the load cells were read. The load cells used for force sensing were very sensitive and any small forces or moments on the cell would be registered as a force on the ring. The shaft attached to the ring passes through a low friction bearing, but the friction from the bearing was still significant enough to give incorrect readings to the controller. Because the controller was aiming for zero force and a responsive system, even slight incorrect readings would cause the motors to kick in. This had the effect of the ring having slight jitters on its own without any input from the user. The noise in the load cell signal also contributed to this slight instability. Iterative tuning of the proportional gain reduced these jitters but maintained the response of the system.

7.1.2. Position control

This mode uses only the encoder readings to determine the behavior of the system. The control scheme for this mode was relatively simple. Position of the ring was read by the encoder and compared with the expected position to determine the relative error. This error was then used in a P controller feedback loop to give appropriate output from the motors. Once the proportional gain was sufficient, it was found that any derivative control added shakiness to the system. Low steady-state error made integral gain unnecessary. There was a slight shakiness inherent in the Position control due to the slight fluctuations in the A/D reading of the position input and the magnification of these fluctuations when scaling up the input to provide a wider range of position travel.

The Position control program reads the input (desired) position and updates the signal controlling the motor at a frequency of 4.6kHz, or about every 2.2 milliseconds. The Position control program runs about 35% faster than the Force control program because it has one less analog-to-digital conversion, which saves time in the calculations. This operating frequency drops as low as an observed 3.9kHz when the finger ring is moving very quickly due to the encoder reading process occurring more often.

Because the position of the encoder is sampled at a certain frequency that is limited by the speed of the microcontroller and the code that is being run, it is possible that quickly-changing input signals produce severe aliasing. Simple tests using a sine wave input showed the device maintained full functionality over the entire length of motion up to 3Hz, above which the device usually became unstable.

7.2. Code

The software was programmed in the Arduino language (based on C/C++ (Arduino n.d.)) for sensor reading and control outputs. It was optimized to run quickly, which gives more consistent and smooth behavior of the system.

The code used a regularly timed interrupt to recalculate the motor outputs. The chip's PWM (Pulse Width Modulation) module was implemented to control the motors. Depending on the direction and speed desired, the PWM signal was modified and the direction signal was updated. The encoder function was written using input capture interrupts to keep track of the pulses and determine direction.

The actual code used is shown in APPENDIX B: FULL MICROCONTROLLER CODE.

8. SAFETY

Safety is an important consideration in any design, and the prototype would be a failure if it caused injury to the user. Listed in this section are some design decisions which took safety into consideration.

8.1. Physical considerations

The front of the device is configured so that the test subject is not exposed to the dangerous moving internal parts of the devices. The tops of the devices are protected by covers, which also protect the test subject from the moving parts, although the test subject is to have the wrist constrained, limiting ability to reach this area anyway. All signal cables emerge only from the back of the devices, far from the test subject, and are bound together and covered with protective tubing in a single package out to the control box.

At opposite ends of their travel, the distance between the rings of two devices exceeds the distance that can be spread between two fingers. This means that it is possible to have the motors move in opposite directions and injure the fingers of the user. To remove this possibility, collar clamps are placed on the shafts during testing so that the device physically cannot move far enough to cause injury.

8.2. Controller/software considerations

The controllers are designed so that changing between Force control and Position control requires a reset of the system. This was found to be important because the input used to give a certain force, for example, could cause the device to move violently to a different position if the modes were changed.

The software includes a “start up” time for when it is initially turned on during which the motors are not given full power. Testing showed that violent motion of the devices would occur if the input called for was far from the current state, in either Position control or Force control. As an example, if the device was started (making its current position zero) and the input called for the device to already be at a position of 50mm, it will move very quickly to that position, probably before the user is ready for such a motion. To counteract this possibility, a “start up” feature was included in the code in which the input is scaled up to the full value over the first second of operation time. In effect, this makes the devices start slow and move gently to the input, even if it was far from the current position/force.

8.3. Operator considerations

In the event of needing to quickly shut off the device, the power switch to the motors was set up so that it turned off by moving to the down position, which is a more natural motion for quickly reaching out to slap downwards, as in the event of a panic stop. This emergency stop directly disconnects the power to the motors for improved safety over switches affecting their changes through code. Continued motion after the emergency switch was flipped due to momentum was a nonissue due to the friction of the system. Also, the emergency stop is the only switch highlighted with bright red markings to catch the eye more easily.

9. CALIBRATION DATA FOR DATA I/O

Contained in this section are the data necessary for interpreting the voltages given by the sensors to convert them into physical values (force or position). Also contained here are the equations to convert the desired input values (force or position) to the input voltages required by the microcontrollers.

9.1. Calibration data for control box outputs

Both force and position readings use a coordinate system in which positive is towards the finger ring, and negative towards the motor. Thus, if the finger ring is extending out farther from the device, moving away from the motor end, the position count will increase. A tensile force on the finger ring which would make the device move in a positive direction is also a positive force, with the force voltage increasing above its steady state value.

The force readings from the top row of BNC jacks are an analog voltage ranging from -10V to +10V. The equations necessary to convert the voltages into force values are given in Table 5. Note the error value of ±1.2N on the force readings. The tests performed to determine this error value are given in Section 10.2.

The bottom two rows of BNC jacks are digital output (TTL) from the encoder readings. At any time, each channel is either 0 or 5V, and the changes from either state are the significant events. Each change in voltage represents a rotation of 1/2000th of an inch, or 0.0005" (.013mm), with an error of ±0.009" (0.2mm) primarily due to lash in the lead screw.

Table 5: Equations to convert each device's load cell voltage reading into a force

Device label	**BNC output calibration equation F (Newtons) = m*V (Voltage) + b**
A	F = 6.98*V +22.58 ±1.2N
B	F = 6.98*V +8.93 ±1.2N
C	F = 7.12*V -3.02 ±1.2N
D	F = 6.83*V - 9.31 ±1.2N

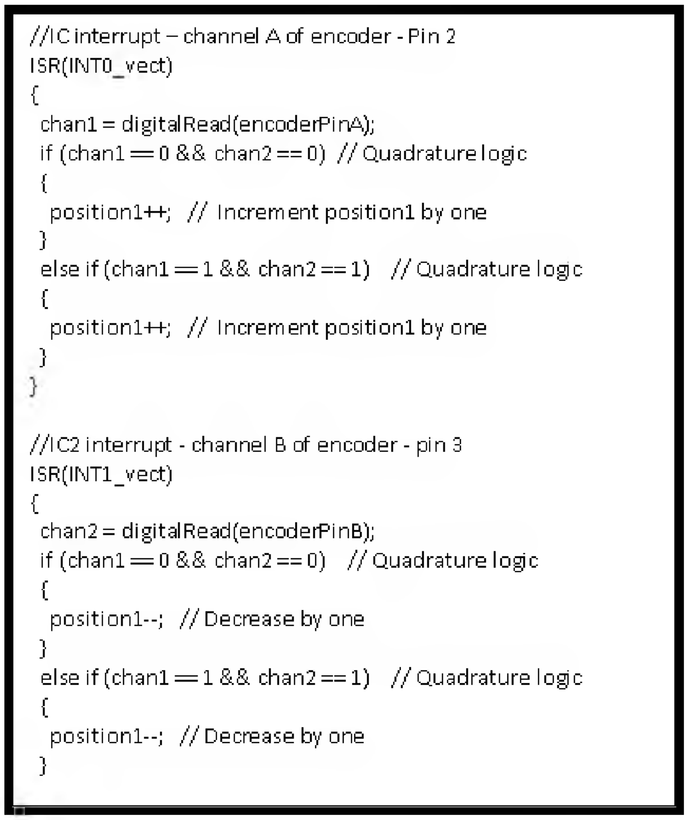
To determine the direction of motion, both channels need to be used together with the following logic:

1. When either channel changes state, the digital states of both channels are compared.
2. If a change is sensed in Channel A and this new digital state of A matches that of Channel B, then the position count increases by 0.0005" in a counter-clockwise direction (when viewed from the motor end), which equates to pulling force on the finger ring, with the finger ring shaft extending farther out of the front plate.

For example, if Chan. A changes to a high (5V) signal, and Chan. B is also high, the lead screw is turning counter-clockwise (when viewed from the motor end) and the finger ring is extending farther out from the frame.

3. If a change is sensed in Channel B and this new digital state of B matches that of Channel A, then the position count decreases by 0.0005" in a counter-clockwise direction (when viewed from the motor end), which equates to pushing force on the finger ring, with the finger ring shaft retracting farther into the frame.

The actual code (C++) used in the microcontrollers to determine position relative to the starting point is given in Figure 25.



```
//IC interrupt – channel A of encoder - Pin 2
ISR(INT0_vect)
{
 chan1 = digitalRead(encoderPinA);
 if (chan1 == 0 && chan2 == 0)  // Quadrature logic
 {
  position1++;  // Increment position1 by one
 }
 else if (chan1 == 1 && chan2 == 1)   // Quadrature logic
 {
  position1++;  // Increment position1 by one
 }
}

//IC2 interrupt - channel B of encoder - pin 3
ISR(INT1_vect)
{
 chan2 = digitalRead(encoderPinB);
 if (chan1 == 0 && chan2 == 0)   // Quadrature logic
 {
  position1--;  // Decrease by one
 }
 else if (chan1 == 1 && chan2 == 1)   // Quadrature logic
 {
  position1--;  // Decrease by one
 }
```

Figure 25: C++ Code used for relative positioning in microcontrollers

9.2. Calibration data for control box inputs

When run in Force control mode with the Input BNC's selected as the input (instead of the knob), an input voltage will tell the devices what force to seek, or apply to a stationary object in the finger ring. The equations used to know what force will be returned for a given input voltage are shown in Table 6. It is important to note that the force input only accepts a range of 0-5V (zero to five volts). Outside this range could cause unstable behavior in the devices or damage them.

When run in Position control mode with the Input BNC's selected as the input (instead of the knob), an input voltage will tell the devices what position to go to. The equations used to know what position corresponds to a given input voltage are shown in Table 7. It is important to

Table 6: Equations to correlate a voltage input to a force output

Device label	**Force input calibration equation V_{in} (Voltage) = m*$F_{desired}$ (Newtons) + b**
A	$V_{in} = .0882*F_{desired}+2.56$
B	$V_{in} = .0867*F_{desired}+2.56$
C	$V_{in} = .0872*F_{desired}+2.56$
D	$V_{in} = .0941*F_{desired}+2.56$

Table 7: Equations to correlate a voltage input to a position output

Device label	**Position input calibration equation V_{in} (Voltage) = m*$Pos_{desired}$ (inches) + b**
A	$V_{in} = 2.4357*P_{desired} + 2.6832$
B	$V_{in} = 2.4843*P_{desired} + 2.6814$
C	$V_{in} = 2.4804*P_{desired} + 2.6837$
D	$V_{in} = 2.4536*P_{desired} + 2.7064$

note that the position input only accepts a range of 0-5V (zero to five volts). Outside this range could cause unstable behavior in the devices or damage them.

Refer to Section 10.3.2 for issues related to Position control accuracy and an alternate method for controlling position.

It should be noted that later testing showed difficulty with the Position control (see 10.3.2), rendering Table 7 inaccurate. Please see Section 10.4 for the correct input method.

10.CHARACTERIZATION RESULTS

This section describes the tests performed to evaluate and characterize the performance of the completed system.

10.1. PWM noise on the Arduino

Towards the end of testing, electrical noise issues were detected when using the potentiometer as an input. It affected the load cell readings and input readings, which in turn affected the system behavior, especially under Force control. This affected both the reading of the load cell value and the front panel input value. The effect of the PWM noise on the load cell reading is shown in Figure 26.

The noise was greater when the error was positive than negative, leading to a decreased noise coupling when the motor needed to turn in one direction over the other. See Figure 27 for the decreased noise in the other direction.

Many options were explored to eliminate this noise. Analog filters of the PWM signal were able to clean it up to a more regular square wave, but the contamination had already taken place before the filter. All the separate analog pins were utilized, and the other PWM pins were tried, but nothing changed. The PWM frequency was altered but the noise remained. The wires carrying the signals were separated and many individual Arduino boards were tried, including a newer model (the Uno). Proving too difficult to remove, the effect of the noise was minimized with a digital filter in the code itself. Empirical testing showed best results with τ=0.15 and

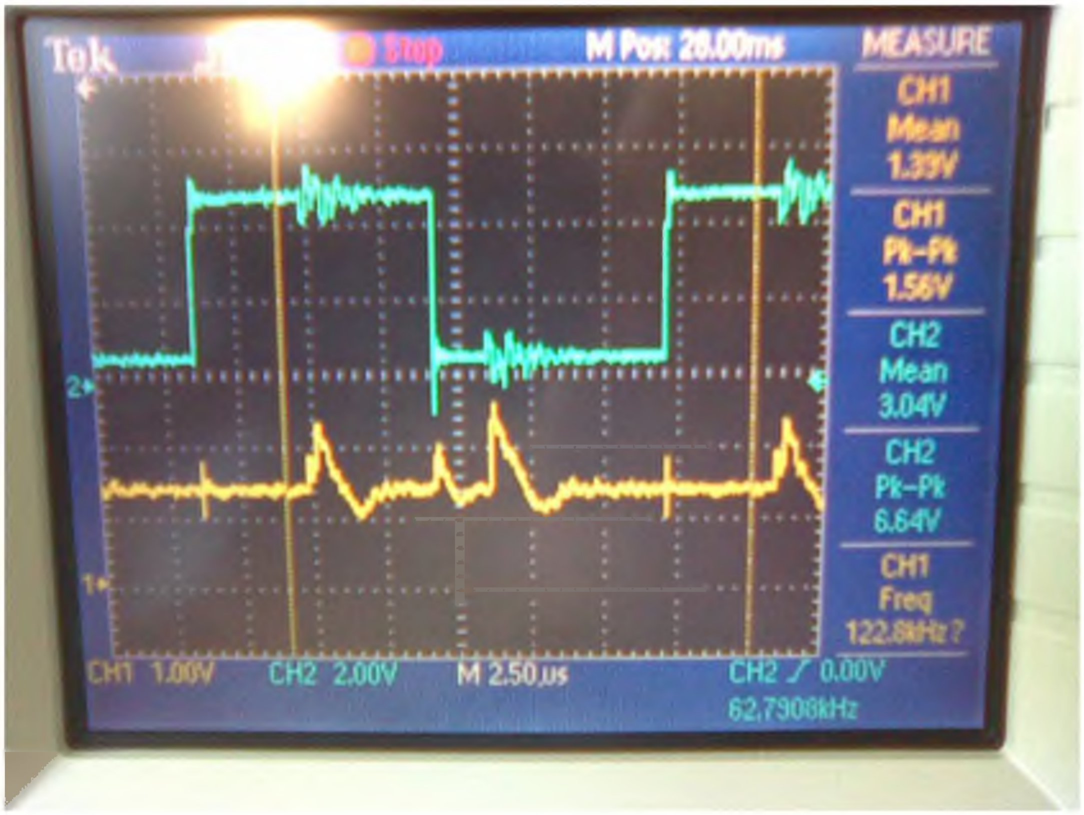


Figure 26: Noise coupled from the PWM signal (blue line) to the load cell signal (yellow line)

f_c=0.94. This reduced the effect of the noise in the system's physical behavior to where it was unnoticeable to the test subject, returning the usefulness of the input knob for demos and quick hardware tests. As the noise was only present when using the input knob, it will not affect the performance of the system under expected use, where each device has an individual BNC input. The digital filter had no noticeable harmful effect on the behavior when running on a clean input signal, as from a signal generator.

10.2. <u>Accuracy of force readings</u>

In order for accurate force measurements to be taken, the limitations of the force readings needed to be known.

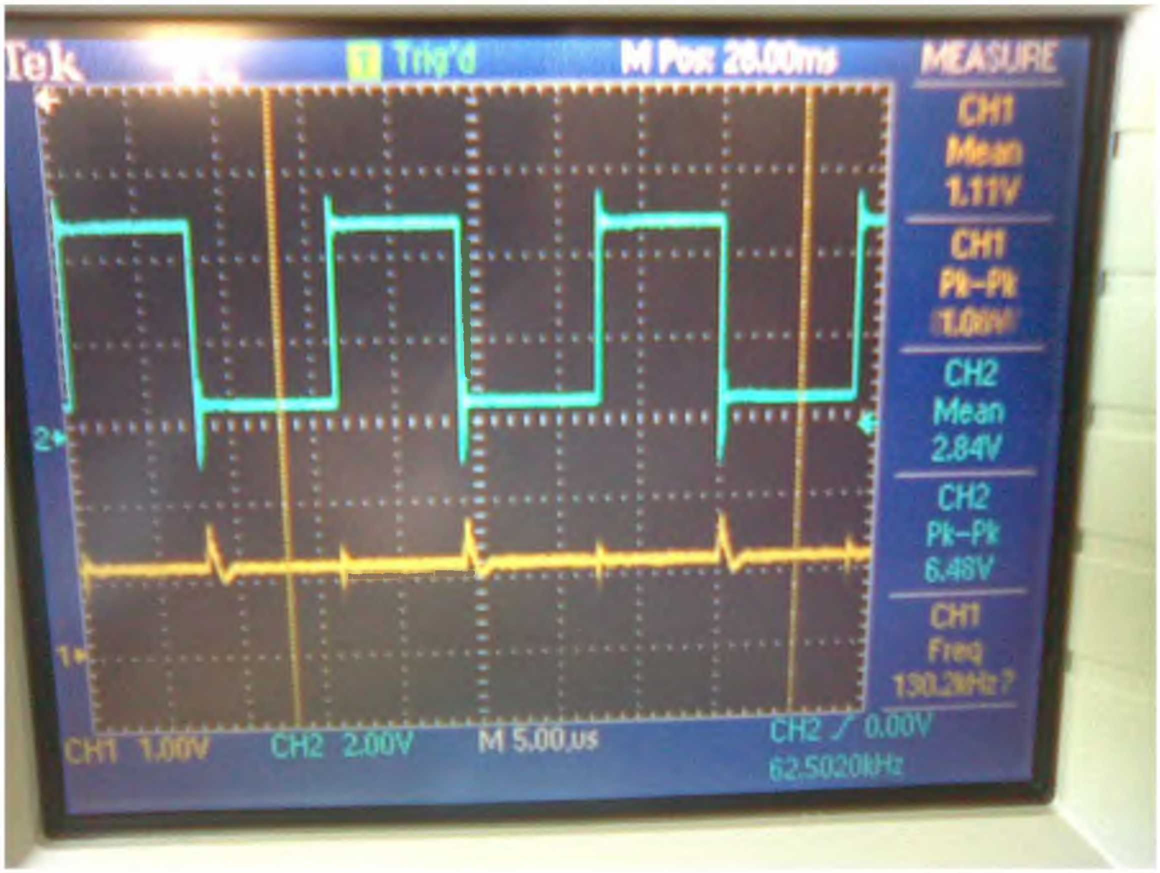


Figure 27: Decreased noise when running in the opposite direction

10.2.1. Effect of linear bearing on force readings

As previously mentioned, the load cell needed to be protected from off-axis forces as well as from tampering by the test subject. This led to the design decision of having a linear bearing in between the load cell and the finger ring, which affects the accuracy of the force readings due to the bearing's friction on the shaft. In order to know the margin of error of the force readings, the finger ring was pushed and pulled, and the static force value measured and compared to the no-load value of the load cell (with the shaft disconnected). The results of these tests are shown in Figure 28. The resulting recommendation is to consider the force readings to have an average error of ±0.3N (0.07lbf) with a maximum error of ±0.9N (0.2lbf). Therefore, in any calculations, it

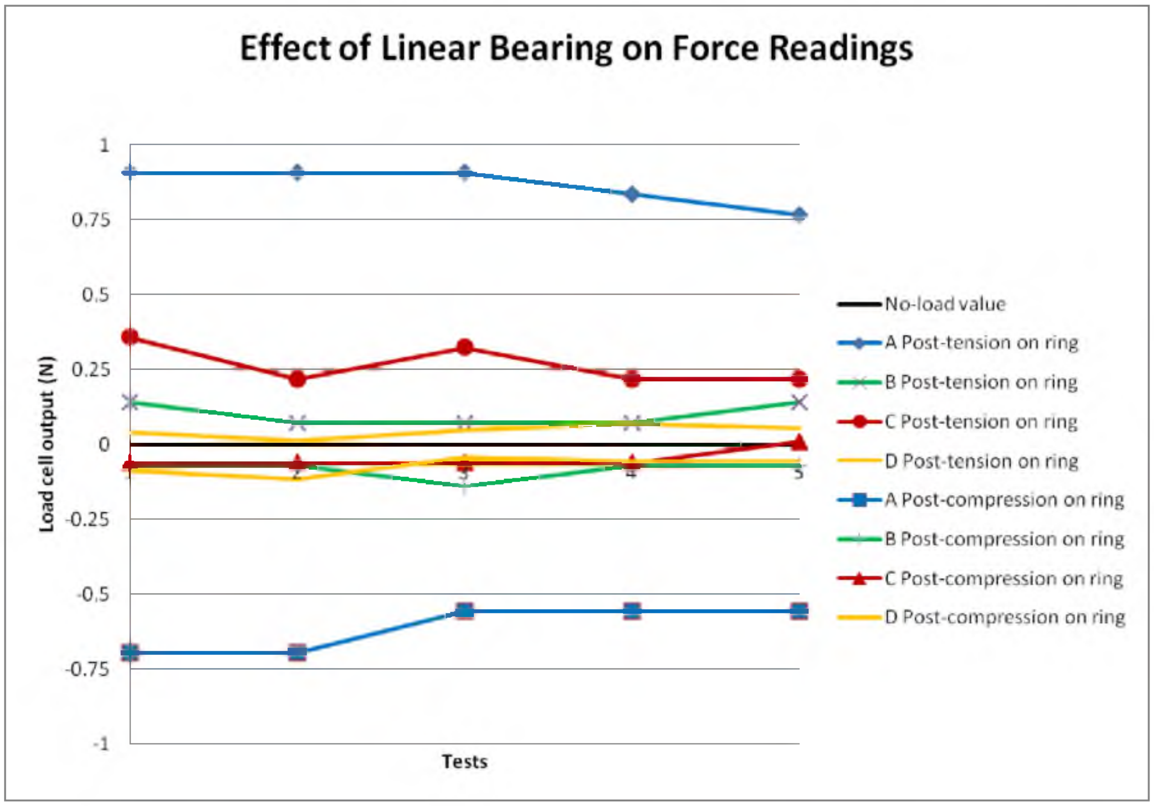


Figure 28: Measurement of friction's effect on force readings

is recommended that the error due to the linear bearing be considered ±1N (0.22lbf). These readings only account for the error caused by the load cell in a static situation, but as the coefficient of static friction is usually larger than its kinetic counterpart (Nave n.d.), the force readings should be less affected by the linear bearing while in motion.

10.2.2. Effect of system noise and motion on force readings

Other factors to consider in estimating the accuracy of the force readings are the variations in the load cell readings due to the high amplification and the slight vibrations in the system during use. The effect these individual factors have on the load cell readings would be difficult to separate, and as they are all occurring continually and simultaneously, they may be considered as a single effect. The results of a Position control test show this effect on the force reading (see Figure 29 and Figure 30). The test was designed to move the ring a certain distance, hold that position, then return to the original position. More details about this type of testing are given in Section 10.3. This type of test was chosen because there was no restraint or force placed on the finger ring, so the levels of accuracy due to noise would be inherent in the system itself.

The two separate areas outlined in Figure 30 show the force readings while the system is in motion and while it is stationary. Note the difference in force amplitude when it is moving; small vibrations from motion cause more variation in the force signal. In order to quantify the variation, the standard deviations for stationary and moving force readings were calculated from six tests similar to that shown in Figure 29. A close approximation to the normal distribution was confirmed. The standard deviations are given in Table 8.

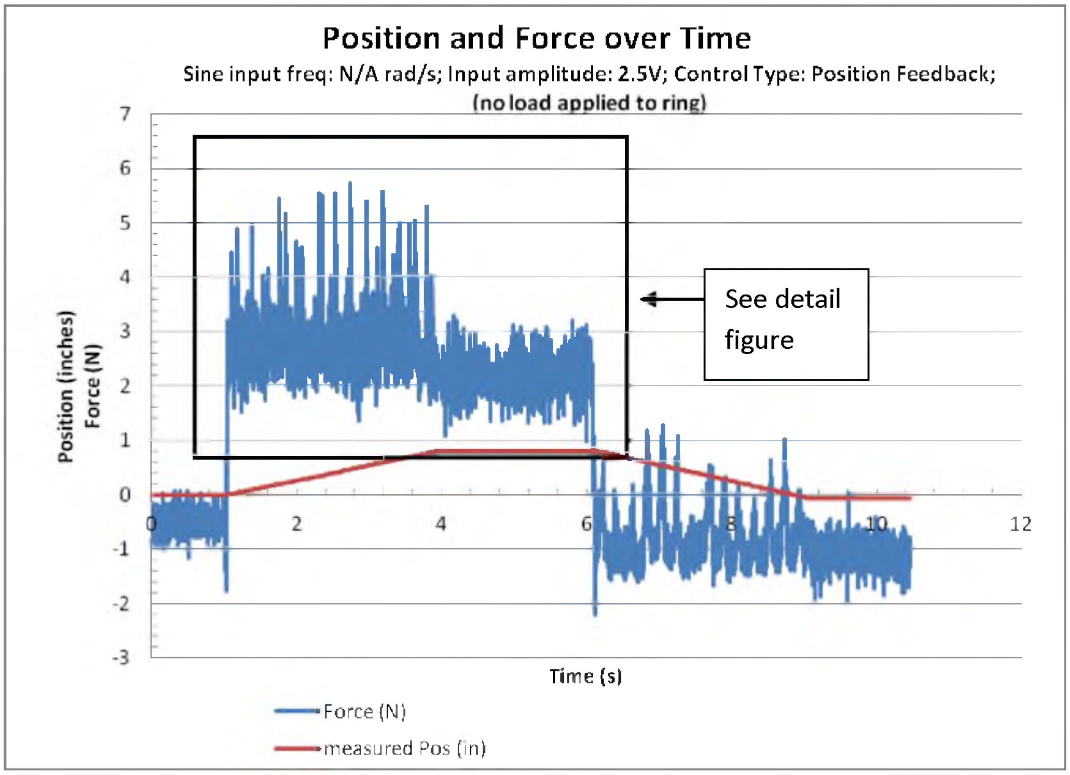


Figure 29: Example of the noise on force readings

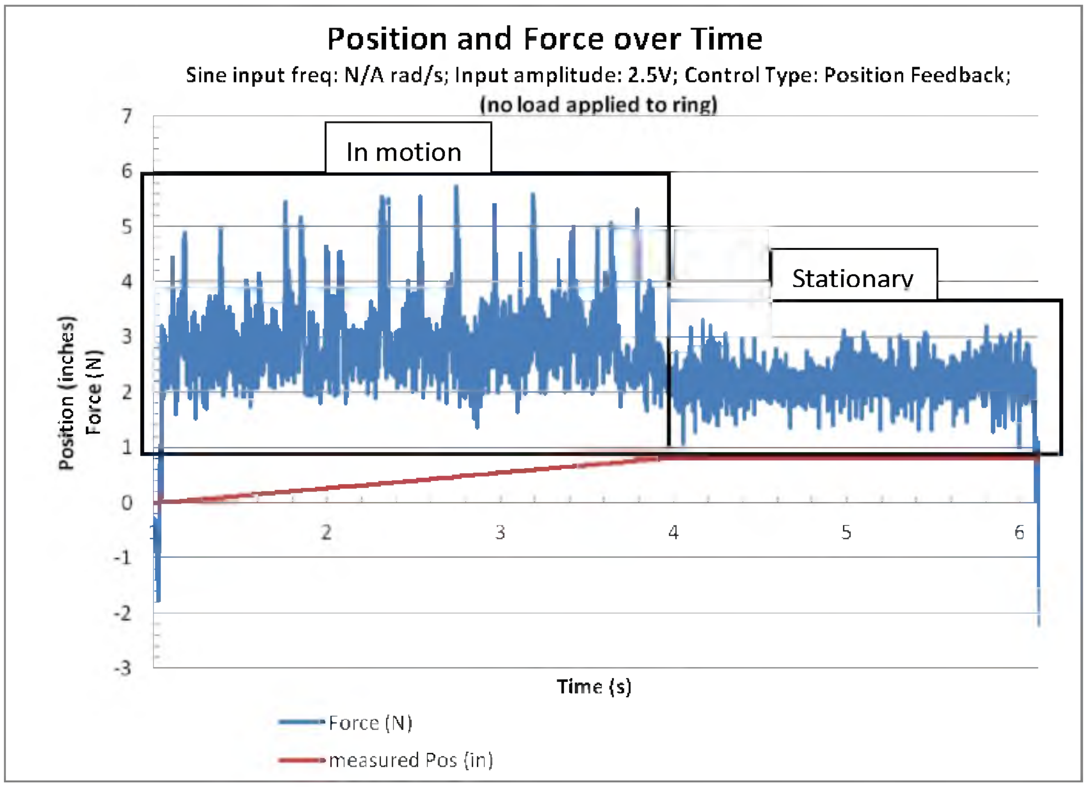


Figure 30: Example of the noise on force readings (detail)

Table 8: Statistical analysis of force readings

Test number:	1	2	3	4	5	6	**Average σ**
In motion σ:	0.417	0.6	0.543	0.731	0.619	0.604	0.586
Stationary σ:	0.219	0.39	0.386	0.442	0.252	0.213	0.317
				68% of force readings within ±0.6N			
				95% of force readings within ±1.2N			

Since few tests will be stationary for very long, the average standard deviation for moving tests was chosen and rounded up to 0.6N. In order to account for a high percentage (95%) of the force readings, an error value of 2***σ** was chosen, leading to a recommended error of ±1.2N on the force readings. This is not much larger than the static error induced by the linear bearing, and indeed the bearing is certainly a contributor to this error.

10.3. Bandwidth testing

In order to determine the limits of the response of the system, the bandwidth was tested using a sine-wave input of increasing frequency but constant amplitude. The input frequency did not change within a test, meaning that each individual test was run at a constant frequency, then a new test was begun at the next frequency. The system response was tested for both Force and Position control. All data were gathered using a dSpace 1104 board sampling at 1kHz. The model used to control the dSpace board is shown in Figure 31.

The system is only rated to 2.5V input amplitude due to the limitations of the Arduino ADCs. A 2V input amplitude was chosen for the force bandwidth test because it corresponds closely to the customer requirement of being able to apply 5 pounds of force (22N). The 2V input rating for the position bandwidth test corresponds to the approximately 2 inch (50mm) motion of a small finger's range. These input amplitude values were chosen for their similarity to expected testing conditions. A few sample tests at higher or lower amplitude showed that the maximum functional speed was inversely related to the size of the input amplitude. This is to be expected, since the momentum of the moving parts is related to the speed with which they are moving and since it is tied to the power of the drive system and control logic.

The bandwidth is shown in both inches and dB. The values were converted to dB using Equation 1.

$$Level_{dB} = 20\log_{10}\left(\frac{L_{\text{measured}}}{L_{\text{expected}}}\right) \tag{1}$$

10.3.1. Force control bandwidth

In order to test the Force control response of the system, a consistent method of applying force to the load cell was necessary. For this purpose, shaft collars (clamps made for cylindrical

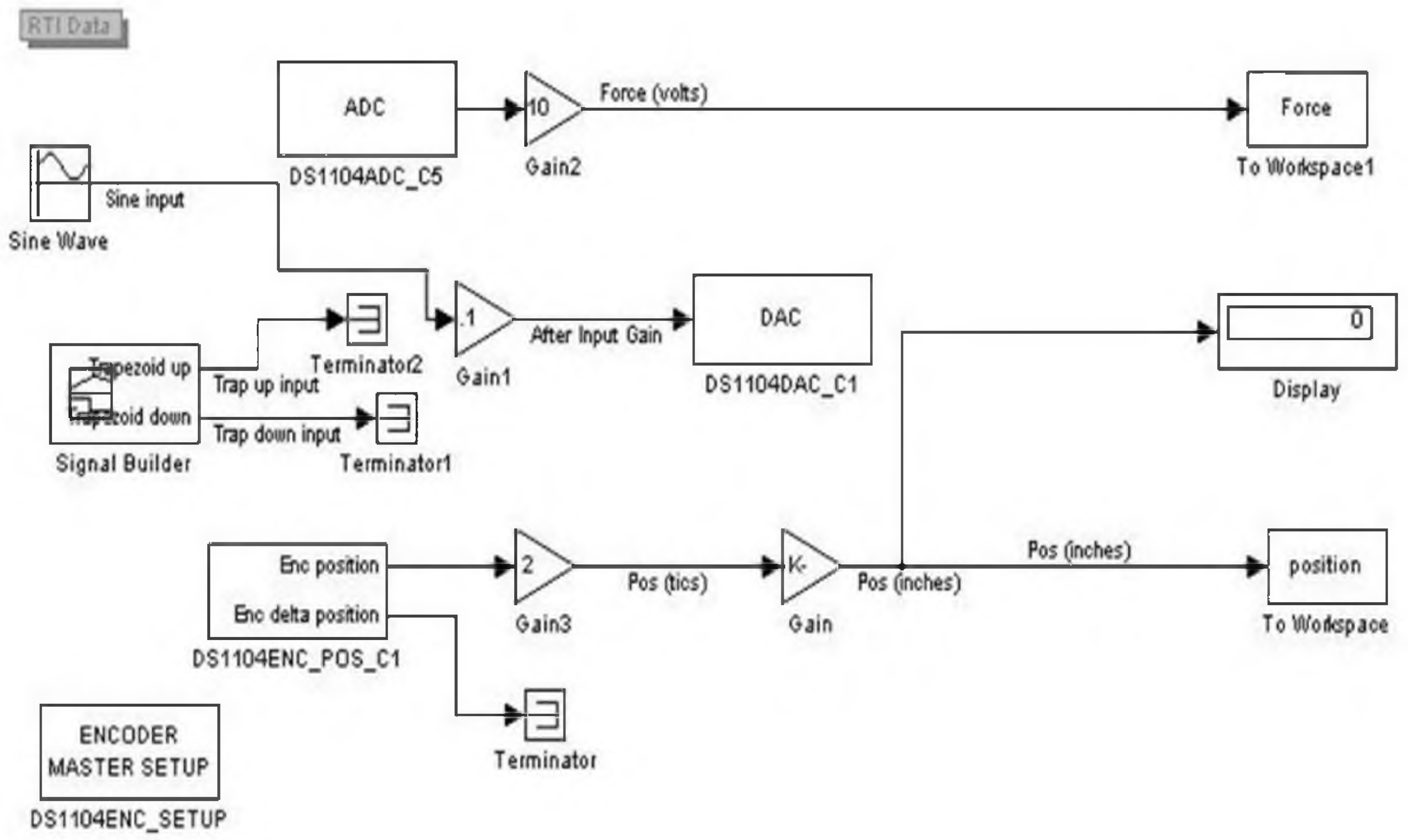


Figure 31: Model used to control dSpace board for testing

pieces) were used on both sides of the linear bearing with foam blocks in between (see Figure 32). Initially, the shaft collars were placed alone against the metal linear bearing, but this created intense vibrations, so a compliant layer of foam was added in between them. This modification gave the system some ability to move towards the desired force, which reduced vibration.

The bandwidth plot for Force control mode is shown in Figure 33. The same plot converted to dB (using Equation 1) is shown in Figure 34. The system followed the input well at lower speeds, although it did not quite reach the input value in one direction (Figure 35). The general decrease in performance as the input speed increases is to be expected as the physical limitations of the system hinder its keeping up with the input signal. Once the input frequency reaches 5.4Hz (34 rad/s), the force readings are barely recognizable as following the input and there is a lot of vibration in the system (Figure 36). Testing beyond 5.4Hz was not performed due to the increasingly high forces of impact and risk of damaging the load cell. Note that the increasing vibration coupled with the general noise on the force signal made the estimation of the peak force difficult. For each data point, several maximums were estimated and their average was found.

For Force control at the maximum desired force (5 lb/22 N), it is recommended that the frequency of the input to the system not surpass 32 rad/sec. However, this is likely to be far faster than necessary for almost any tests.

10.3.2. Position control bandwidth

A similar series of tests were begun to determine the bandwidth for Position control. However, tests quickly showed that the system would gradually drift from its initial position. The center point for the sine-wave oscillation would slowly move farther out, away from the motor.

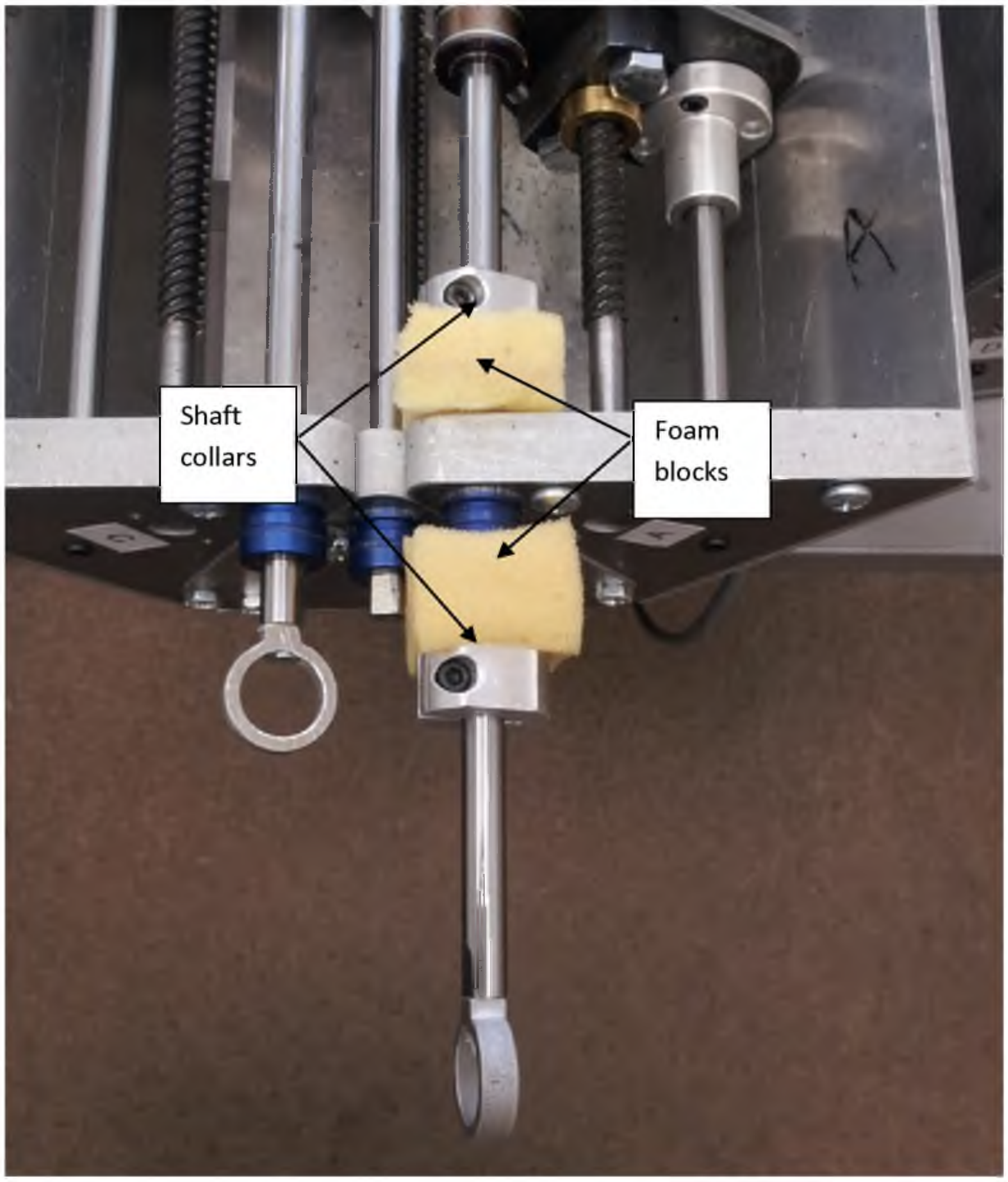


Figure 32: Setup for testing Force control bandwidth

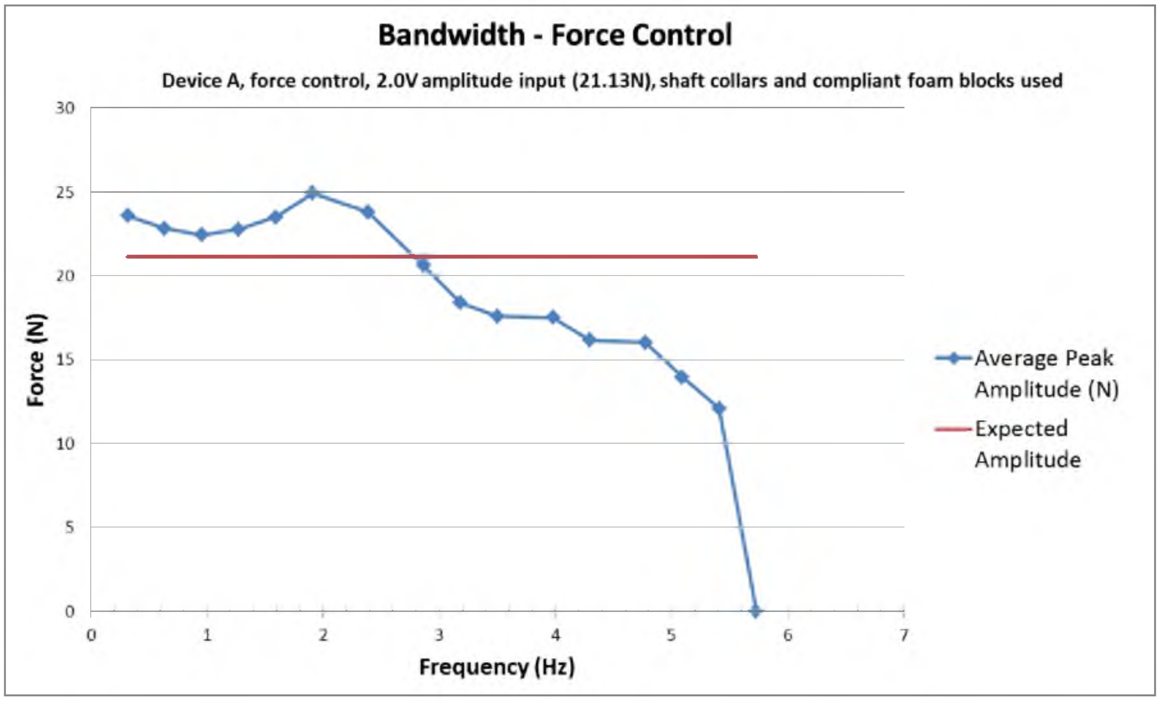


Figure 33: Force control bandwidth testing results – Newtons

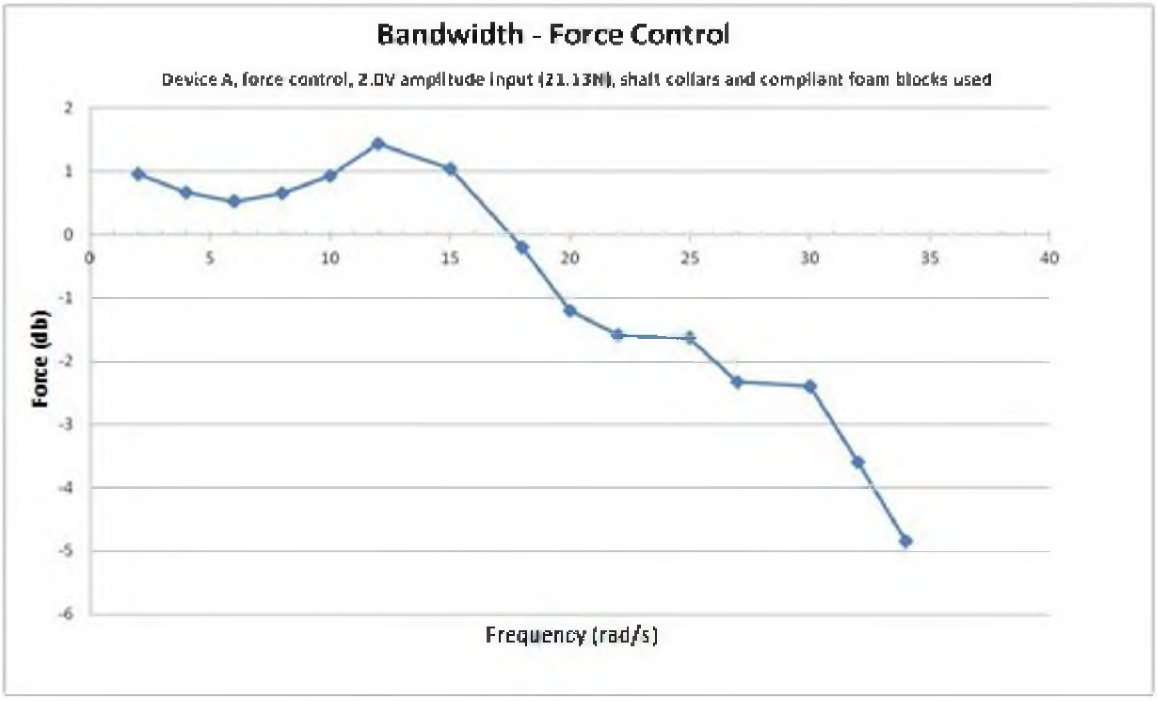


Figure 34: Force control bandwidth testing results - dB

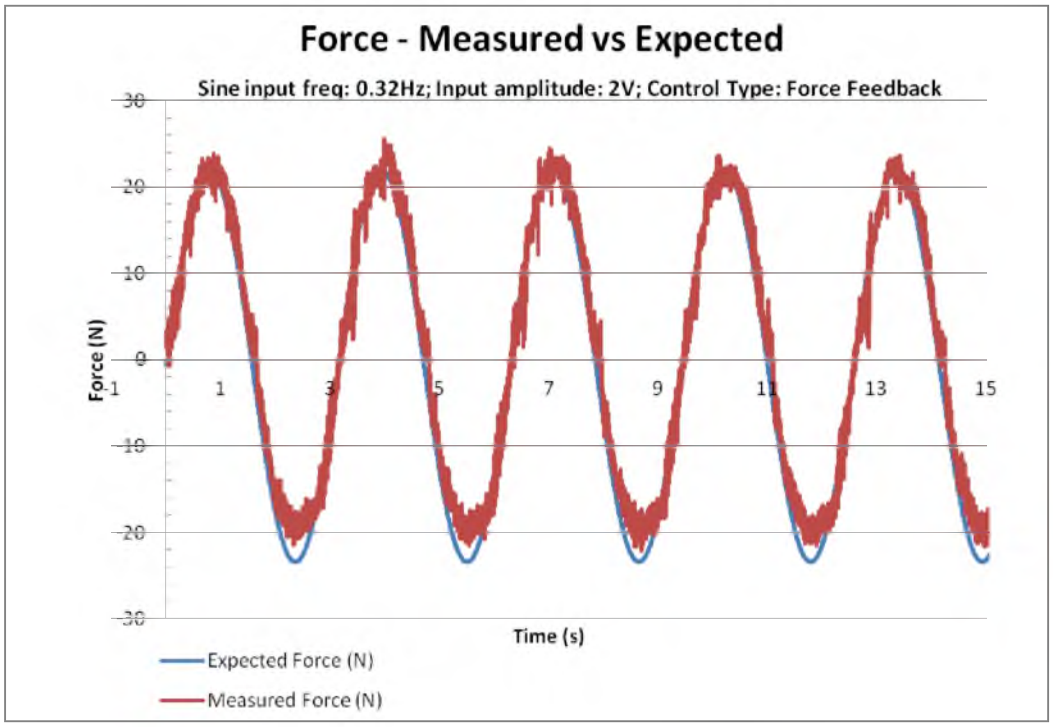


Figure 35: Measured force vs expected, 2 rad/s input

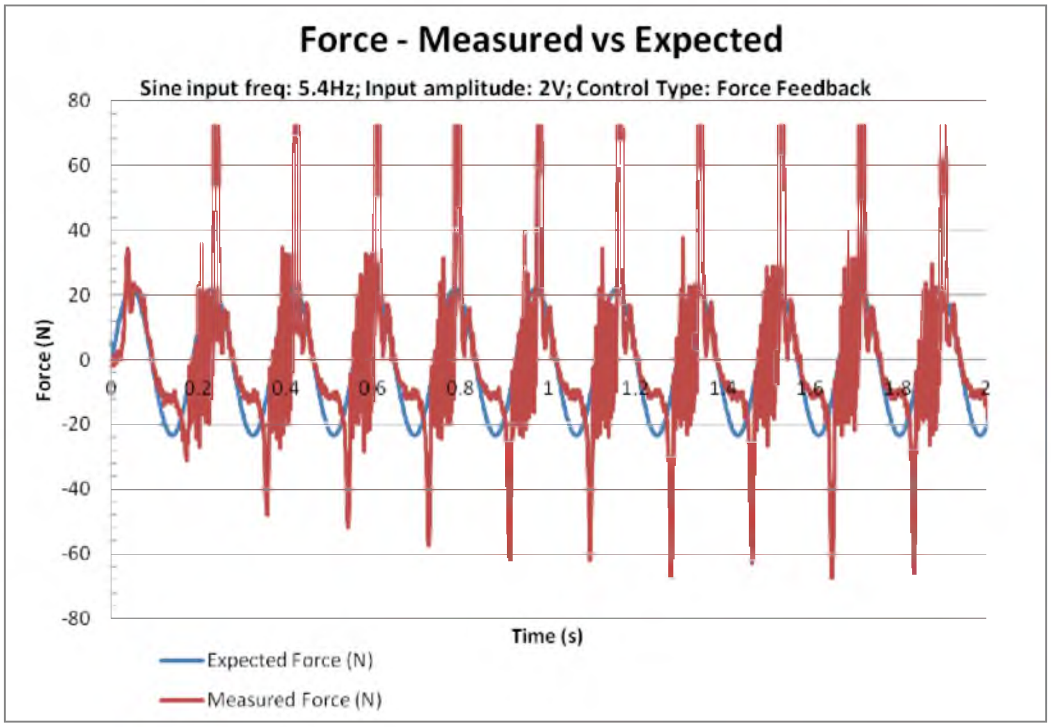


Figure 36: Measured force vs expected, 34 rad/s input

This can be seen in Figure 37. This error was encountered at all input speeds, and in all four of the devices.

It should be noted here that the system was unable to run at all with an input frequency above 15 rad/s. The momentum of the moving parts immediately carried them off onto the removed threads part of the lead screw instead of returning in the other direction.

The position error values were found at three different input frequencies. The results are shown in Table 9. The result for sine-wave inputs was that the system moved out farther every time than it moved back in.

In order to further investigate the problem, a stepped-ramp input was used (see Figure 38). These tests showed the same directional bias seen in the sine-wave input analysis.

The error was initially assumed to be a problem with the Arduino code. The USB output of the Arduino was used to read the variables as they changed. The ADC read in the correct values from the input and the position in which the Arduino believed itself to be corresponded with the input. The code was inspected and optimized so that its speed would not be a liability in reading the position correctly. The code's function matched several versions found online which were rated well by experienced users. Local individuals with more Arduino experience were asked for input but none were able to identify the problem. It was concluded that the Arduino was missing counts of the encoder somewhere, and that a much higher level of expertise would be necessary to resolve the issue, so an alternate control method was found. The position mode could still prove useful for stationary tests, as it only lost its position while moving.

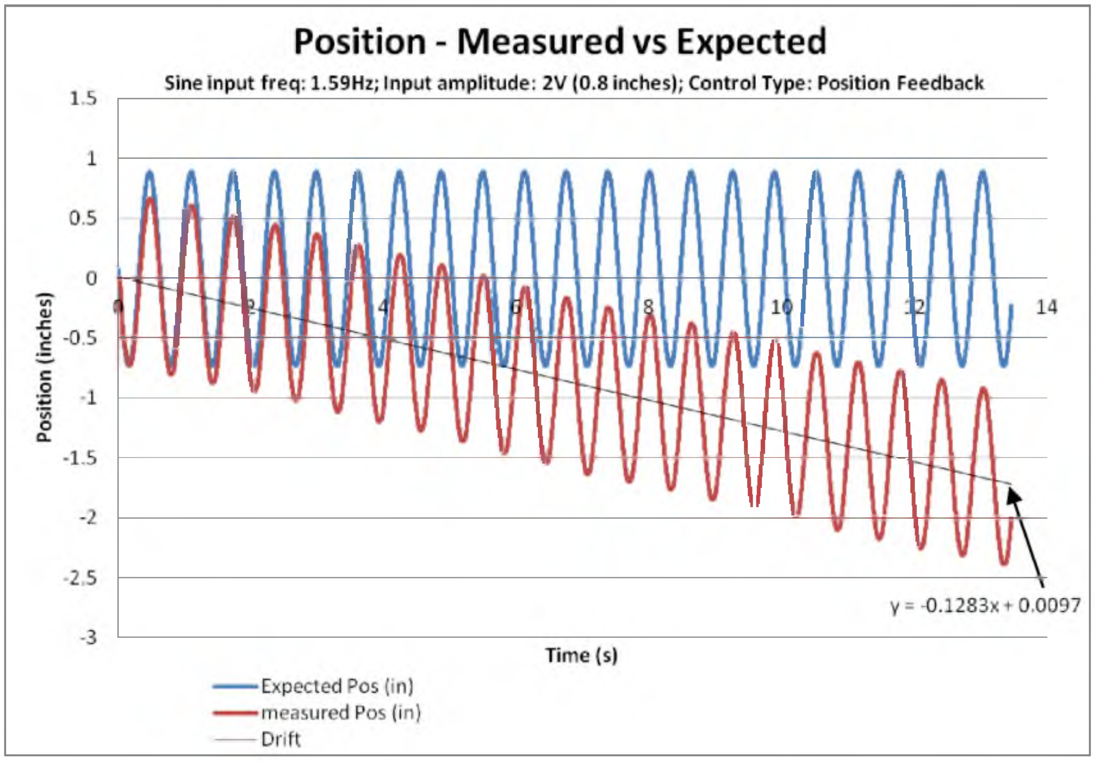


Figure 37: Typical example of drift in position testing

Table 9: Position control error data

		5 rad/s		8 rad/s		12 rad/s	
Cycle	Peak	Positions (in)	Abs Δ	Positions (in)	Abs Δ	Positions (in)	Abs Δ
1	Low	-0.6995		-0.705		-0.73725	
1	High	0.6585	1.358	0.6795	1.3845	0.67825	1.4155
2	Low	-0.79225	1.45075	-0.784	1.4635	-0.802	1.48025
2	High	0.571	1.36325	0.59925	1.38325	0.60525	1.40725
3	Low	-0.8865	1.4575	-0.87875	1.478	-0.87025	1.4755
3	High	0.471	1.3575	0.50975	1.3885	0.5395	1.40975
4	Low	-0.985	1.456	-0.96425	1.474	-0.9335	1.473
4	High	0.3695	1.3545	0.4245	1.38875	0.467	1.4005
5	Low	-1.088	1.4575	-1.0505	1.475	-1.0163	1.4833
5	High	0.285	1.373	0.33125	1.38175	0.38025	1.39655
6	Low	-1.1643	1.4493	-1.1295	1.46075	-1.1043	1.48455
		Average in:	1.36125		1.38535		1.40591
		Average out:	1.45421		1.47025		1.47932
Δ between averages:			0.09296		0.0849		0.07341

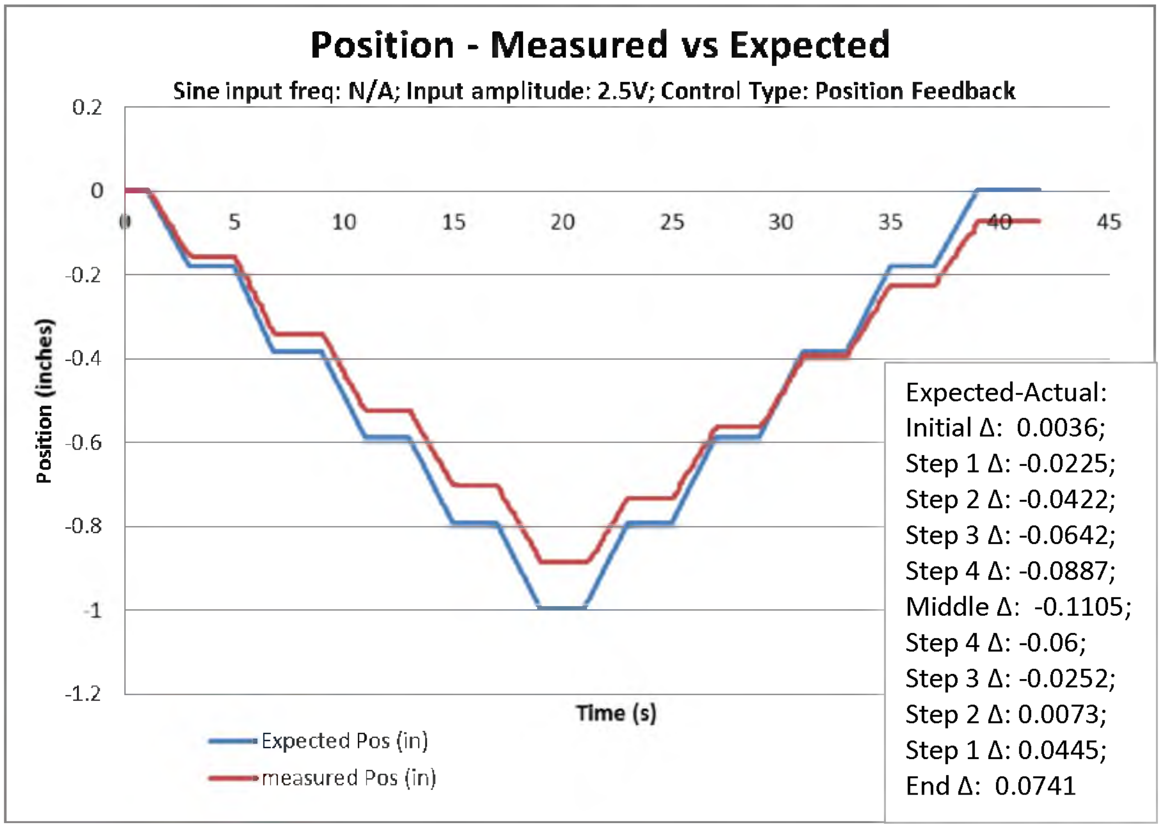


Figure 38: Stepped-ramp input showing ineffective Position control

10.4. <u>External Position control using force</u>

An effective alternate to the Position control was found through modifying the input to the Force mode, which had been shown effective and functional. Note that this required no change to the Arduino code or any other component. This mode was called External Position control using Force, abbreviated as Position/Force control for convenience. The method was to compare the position of the system to the desired position, then send the Arduino a command to apply a force which moves it in the correct direction. The Force control mode on the Arduinos was used for this process, unchanged from Section 10.3.1. The actual model used with the dSpace board is shown in Figure 39. Note that all position calculations and changes were made on the controlling computer, external to the Arduino calculations. Tests showed this method

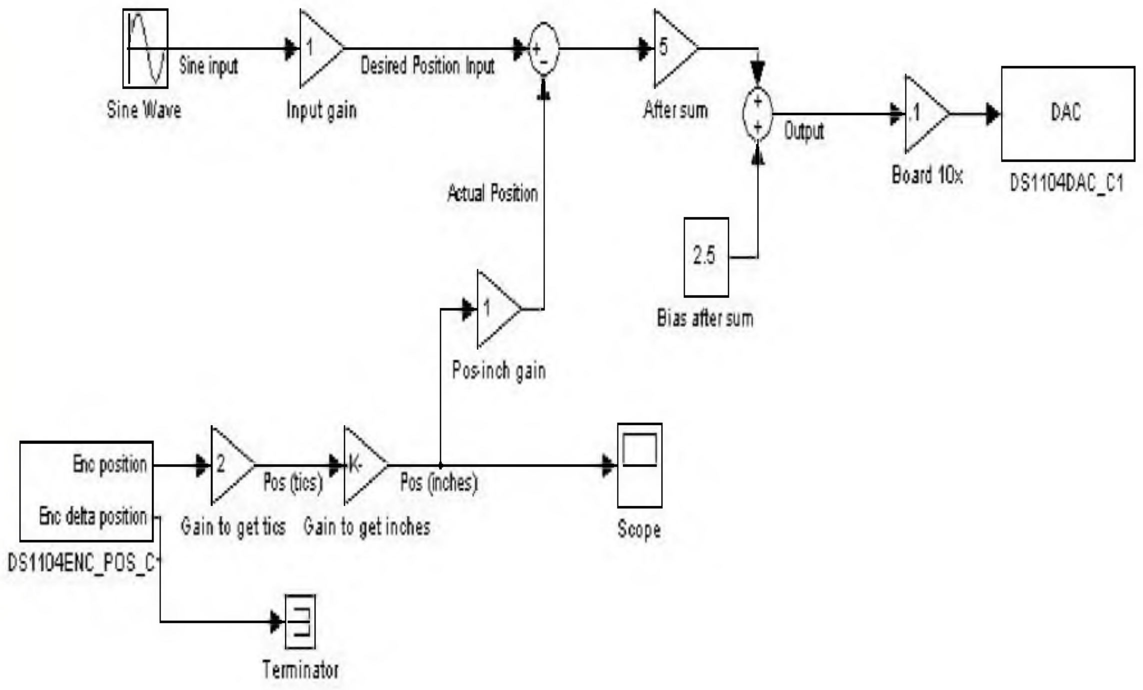


Figure 39: Matlab model for running Position Control through Force

was far more effective at controlling position. No drift was detected in tests lasting 100 seconds, whereas before, it was noticeable in less than 10 seconds. A sample test comparable to Figure 37 (approx 1 inch amplitude, 10 rad/s) is shown in Figure 40. The same input given in Figure 38 was used again, with vastly improved results (see Figure 41).

Note that, as configured in Figure 39, the voltage given to the Arduino as an input corresponds directly to position. In equation form, this means V_{in} (Voltage) = $Pos_{desired}$ (inches). In this situation, the 0-5V input range of the Arduino means the voltage should stay between -2.5 and 2.5V, yielding a motion range of ±2.5 inches (63.5mm).

The bandwidth plot for Position/Force control is shown in Figure 42. The same plot converted to dB (using Equation 1) is shown in Figure 43.

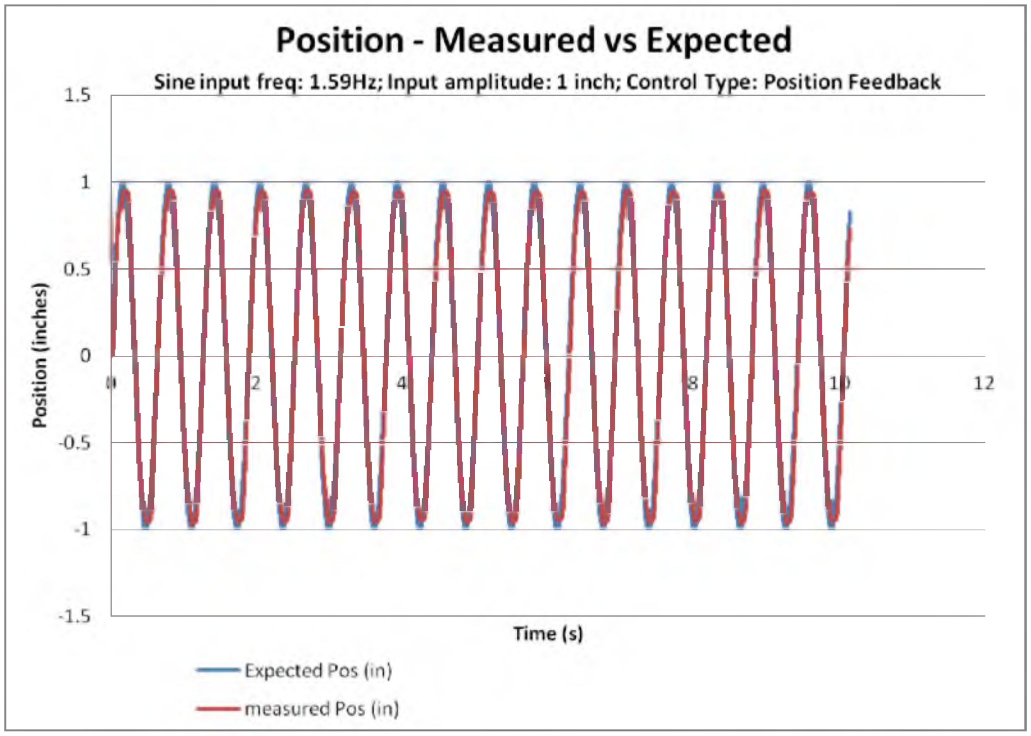
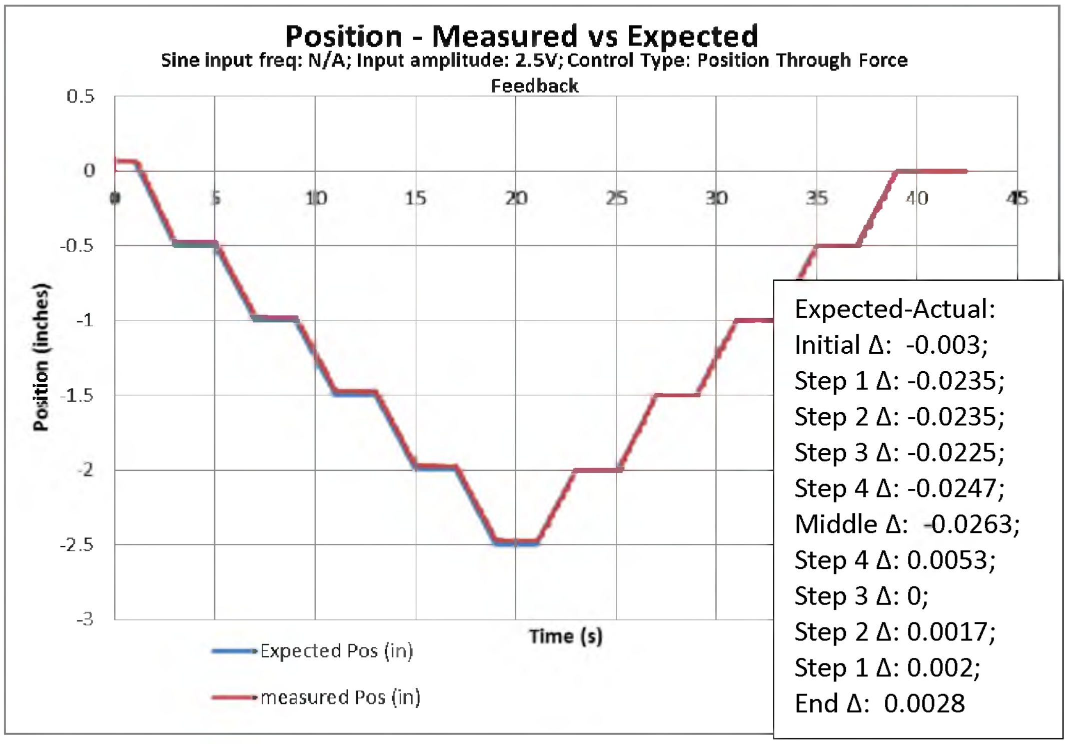


Figure 40: Improved results from controlling position through force



Expected-Actual:
Initial Δ: -0.003;
Step 1 Δ: -0.0235;
Step 2 Δ: -0.0235;
Step 3 Δ: -0.0225;
Step 4 Δ: -0.0247;
Middle Δ: -0.0263;
Step 4 Δ: 0.0053;
Step 3 Δ: 0;
Step 2 Δ: 0.0017;
Step 1 Δ: 0.002;
End Δ: 0.0028

Figure 41: Stepped ramp input using Position/Force control

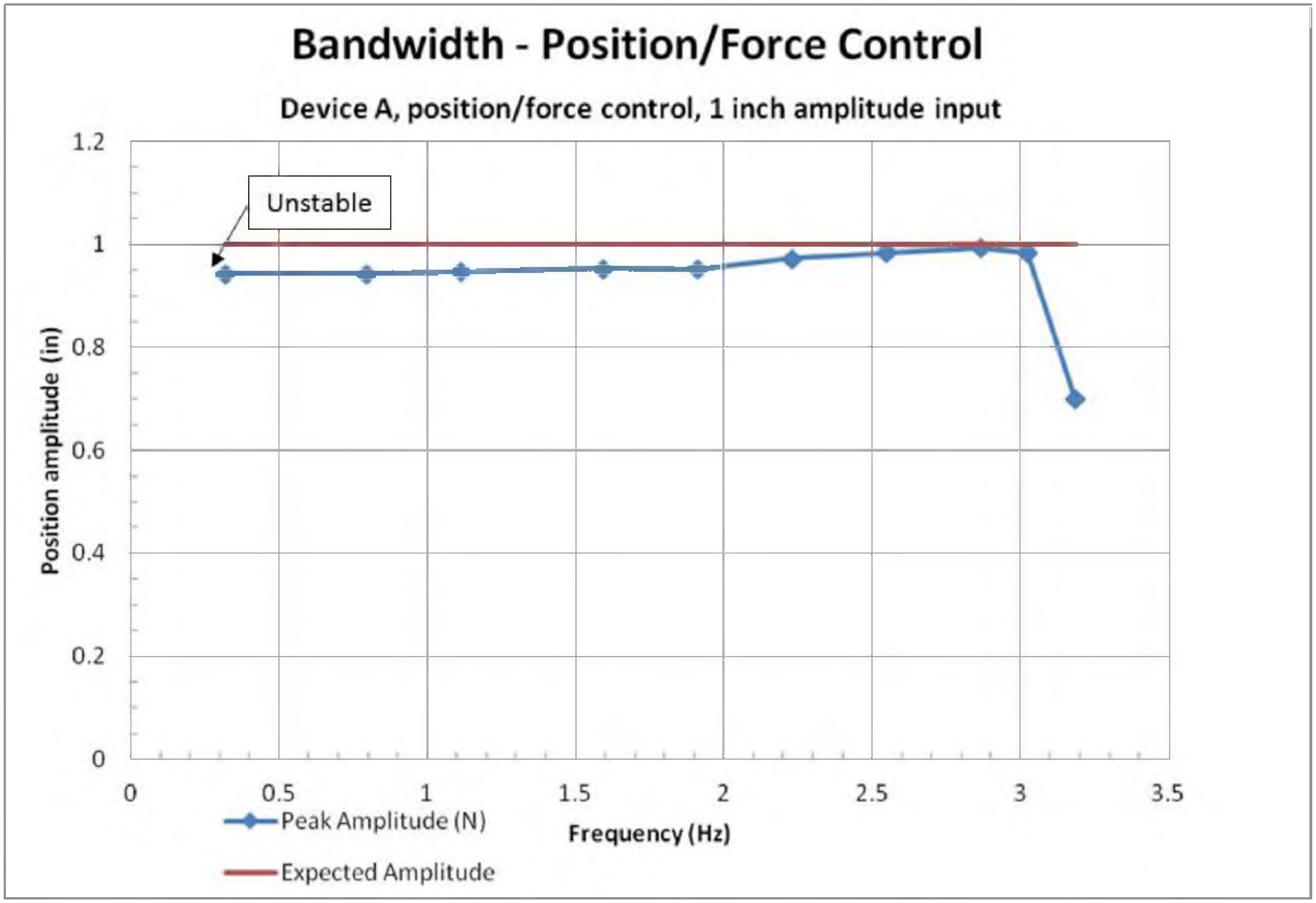


Figure 42: Position/Force control bandwidth testing results – inches

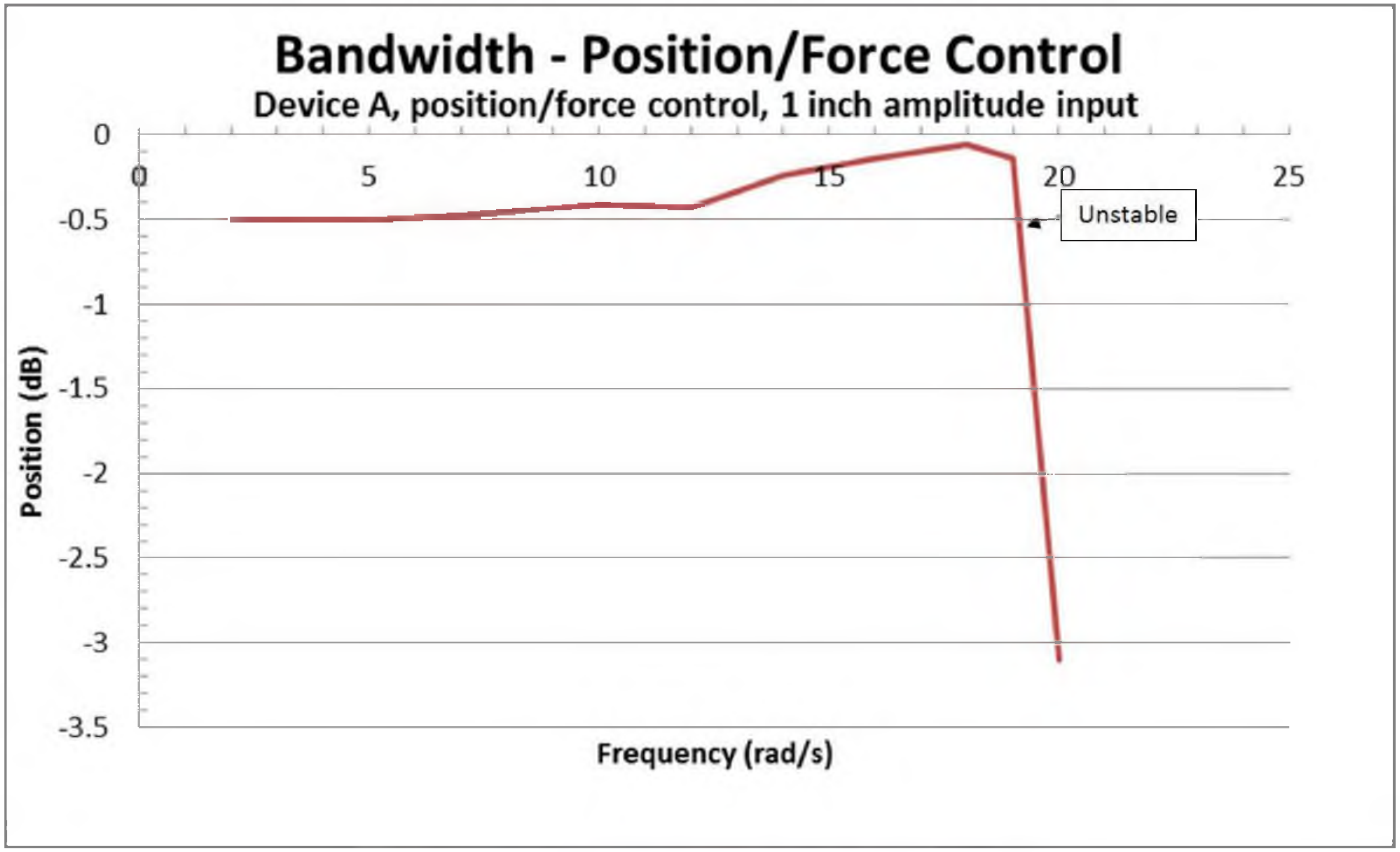


Figure 43: Position/Force control bandwidth testing results – dB

These tests showed the same instability at higher speeds as the earlier Position control tests, where the moving parts were unable to reverse direction and so continued off the threaded part of the lead screw. Note that the last stable input frequency for Position control was 15 rad/s sec, and this value increased to 19 rad/s for Position/Force control with a slightly higher amplitude.

The data for the last stable test (at 19 rad/s) were analyzed and the highest speed calculated was almost 19 inches/s (480mm/s). This value is more than three times the required speed for function, while still remaining stable. Exceeding the speed goal by such a large margin was encouraging.

10.5. Accuracy of position readings to input

With the Position control functioning through an external loop, it was possible to measure the accuracy of the position measurements compared to the input (expected) value. For all Position/Force control tests, the error between expected position and true position was calculated. The absolute value of the error was averaged over the entire test, and this average is shown in Table 10. Of interest in these position error values is the increasing error as the sine-wave frequency increases. This is to be expected as the input is changing more quickly, allowing more difference in the desired velocity the next time the Arduino checks the input. Also it is important to note that the Stepped-Ramp Tests (Test 2 is shown in Figure 41) had an order of magnitude better correlation than the faster sine input tests. This can be attributed to the nature of the test; slow velocities and stops at certain positions allowed the Arduino to keep up with the desired input.

The error for all tests was averaged and found to be 2.5mm (0.1in). This value is justifiable for the recommended error on all position measurements because tests at the high input

Table 10: Average position error for all tests

Input Type (1 inch amplitude)	Average Position Error (inches)
Sine 0.318 Hz	0.078
Sine 0.637 Hz	0.098
Sine 1.114 Hz	0.107
Sine 1.592 Hz	0.124
Sine 1.91 Hz	0.135
Sine 2.228 Hz	0.147
Sine 2.546 Hz	0.16
Sine 2.865 Hz	0.171
Sine 3.024 Hz	0.177
Stepped-Ramp test 1	0.054
Stepped-Ramp test 2	0.017
Stepped-Ramp test 3	0.014
Stepped-Ramp test 4	0.017
Average of all tests	**0.0999**

speeds are very unlikely in the intended purpose of the system. Input speeds in actual use are more likely to be on the slower end of the sine-wave tests or similar to the stepped-ramp tests, where the error is close to or less than the recommended error of ±2.5mm on the position readings.

11.TESTING RESULTS

This experiment was performed with the purpose of showing the equipment's ability to provide force stimuli to the fingers and quantify their biomechanical behavior. Surface EMG data from the primary muscles controlling the thumb, index, and middle fingers were also recorded simultaneously, to show the device's ability to relate EMG signals to the motion of the fingers. As discussed previously, this correlation has important implications for the improvement of multiple DOF prosthetic control. Furthermore, the suitability of such data for individual control of the fingers in a prosthetic will be shown.

11.1. <u>Test procedure</u>

The entire test protocol was written and tested prior to the experiment. All required equipment was listed, with labeled pictures of complicated wiring schemes.

For this experiment, the subject was instructed to keep all fingers as still as possible, resisting whatever force was applied. This isometric test was chosen due to its simplicity and ease of instruction to any subject. Only forces that pulled on the fingertips in a hand-opening motion were applied, never pushing down on the fingernail in a hand-closing motion. Each finger was tested alone at 1, 2, and 3 lbf. All combinations of the three fingers were tested at those same force levels, with every active finger receiving the same force level. For consistency in description, each individual combination of active fingers and force level is referred to as a test, and all these tests are called the experiment.

The force profile applied is shown in Figure 44. This profile was repeated five times at the same levels and fingers to form one test.

11.1.1. EMG preparation

EMG locations were carefully located on two subjects' arms multiple times using a Myolab II EMG amplifier. For each of the three fingers, the site which generated the largest and most independent signal was marked. Complete isolation of a finger's EMG was not possible, but the most independent signal location was found. For the index and middle fingers, these sites corresponded to different parts of the flexor digitorum profundus muscle. For the thumb, this corresponded to the flexor pollicis longus muscle. Silver/silver chloride electrolyte gel and EKG electrodes were then applied to each location. See Figure 45 for labeled muscle electrode locations. This process was practiced multiple times, and was performed before traveling to the other lab for the actual testing

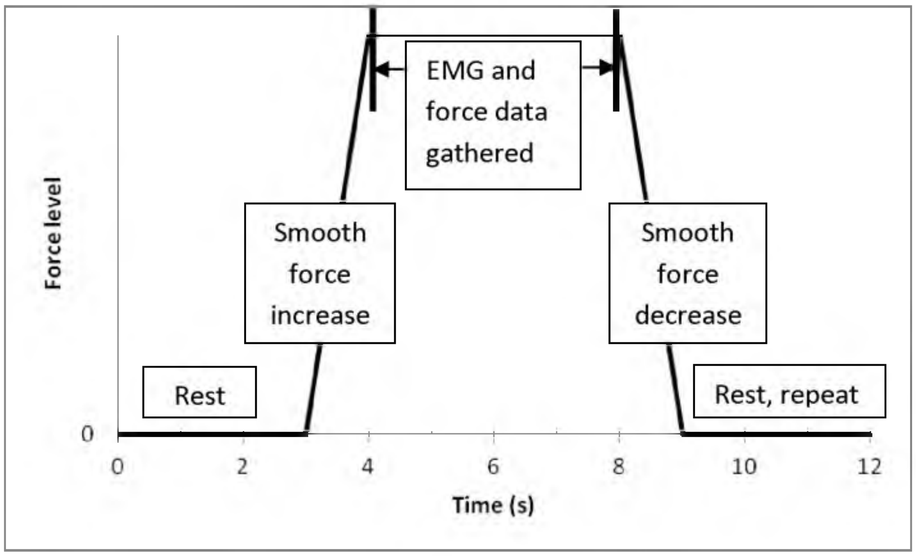


Figure 44: Force profile for tests

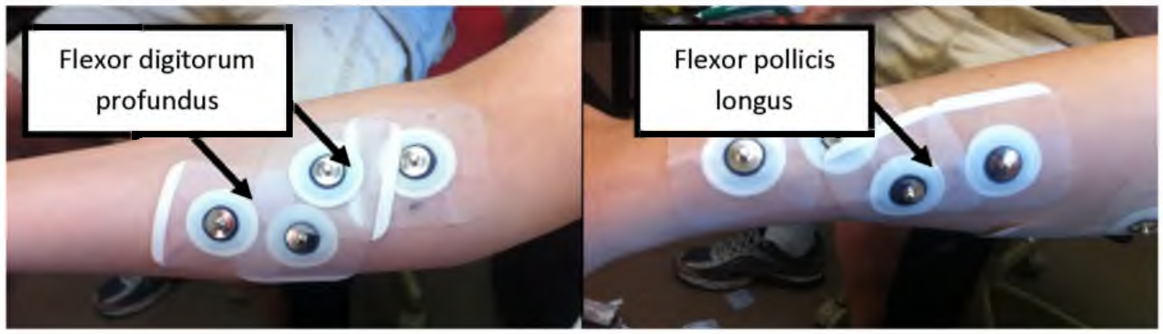


Figure 45: Arm in process of EMG electrode placement

11.1.2. Testing

Upon arriving at the testing location, procedure was followed to set up the EMG amplification, data recording hardware, and recording software. Upon initial results, EMG amplification was increased slightly for more signal change. Also, the EMG data were found to have noise that was not present in the location where the preparation was done. This was remedied as well as possible on location using the built-in filters on the EMG amplifiers.

See Figure 46 for a picture of test setup, arm position, and hardware interface with the test subject. Note that this picture was taken during setup; not all electrodes had been connected yet.

Each combination of fingers at three force levels was tested, at 1, 2, and 3 lbf. The experiment was performed such that each force level and combination was achieved five times for diversity of sampling, as described previously. Care was taken to randomize the order of the tests so as to not cause fatigue on any single finger. Testing was completed without incident or injury.

After the first subject completed testing, the process was repeated for the second subject.

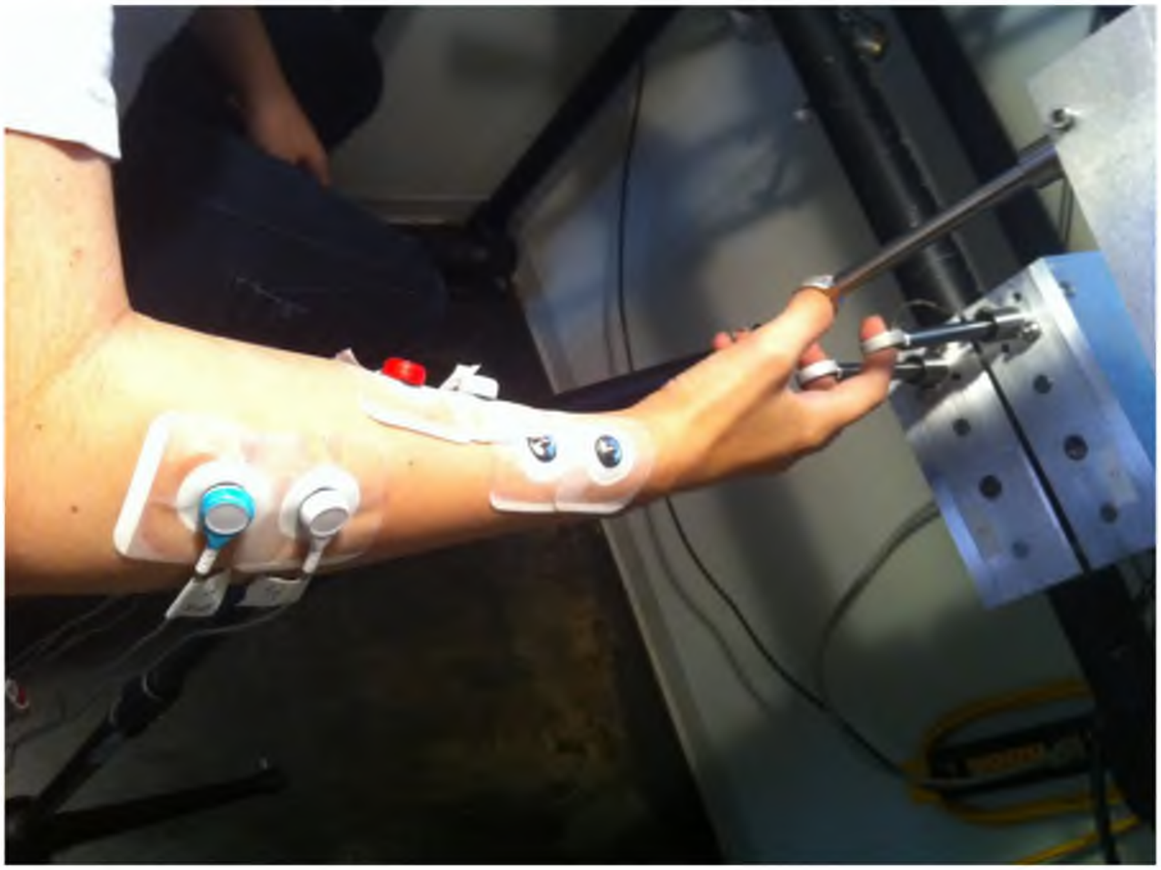


Figure 46: Test subject during setup, with force application/measurement hardware

11.1.3. Equipment used

The experiment described here was performed on a more customized iteration of the device described previously, with different force sensors and amplifiers, and a PC-based controller that was more integrated with existing neural test equipment. This PC-based control had some limitations, such as inability to output position, only applying the same force level to any tested fingers (rather than the completely independent control of the previous version), and only thumb, index, and middle fingers being tested.

Given here is a general list of equipment used for testing, in addition to the device and its control PC.

- Surface EKG electrodes and silver/silver chloride electrolytic gel

- Myolab II EMG preamplifier
- 4 channel EMG preamplifier
- Circuit board with precision rectifiers
- DC power supply
- PC with dSpace, Simulink, and connected DS1103 controller board sampling at 1kHz

11.2. Testing results

Upon review, it was found that the EMG results from the second subject were excessively noisy; a combination of low EMG signals and excessive interference from nearby electrical devices made changes in the signal indiscernible. This result was confusing, as the second subject had simultaneously undergone the same process of identifying optimal surface EMG locations and was in the same testing area as the first subject. This noise, while worse with the second subject, was present in both datasets. It is likely that the seemingly high ambient interference was exacerbated by the long wires for the electrodes before amplification.

11.2.1. Graphical results

Each individual force level was graphed and analyzed. A sample graph, in which each of the three fingers experienced 2lbf, is shown in Figure 47.

The noise on the EMG signals that was previously discussed was filtered for visual clarity in the graphs. A centered moving-average filter with time samples of 10ms, 25ms, and 50 ms were tested on all data sets, with 25ms having the best balance between maintaining detail and removing noise. See Figure 48 as an example of the improved clarity using this filter.

Note the correspondence of the EMG signals' rise with the force measurements. This is the expected result. The middle finger and thumb showed a stronger correlation for all force

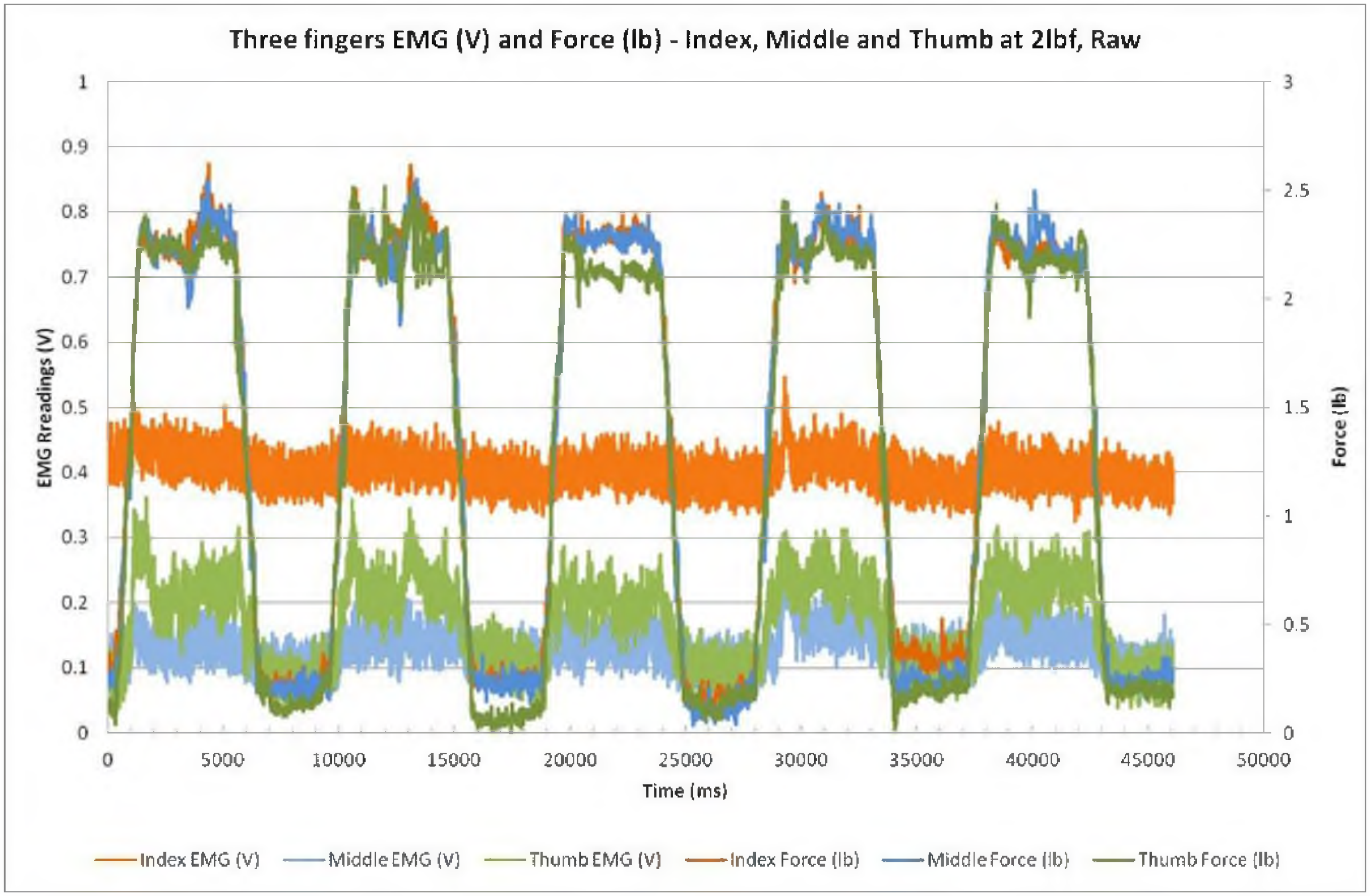


Figure 47: Sample graph, three finger test at 2lbf

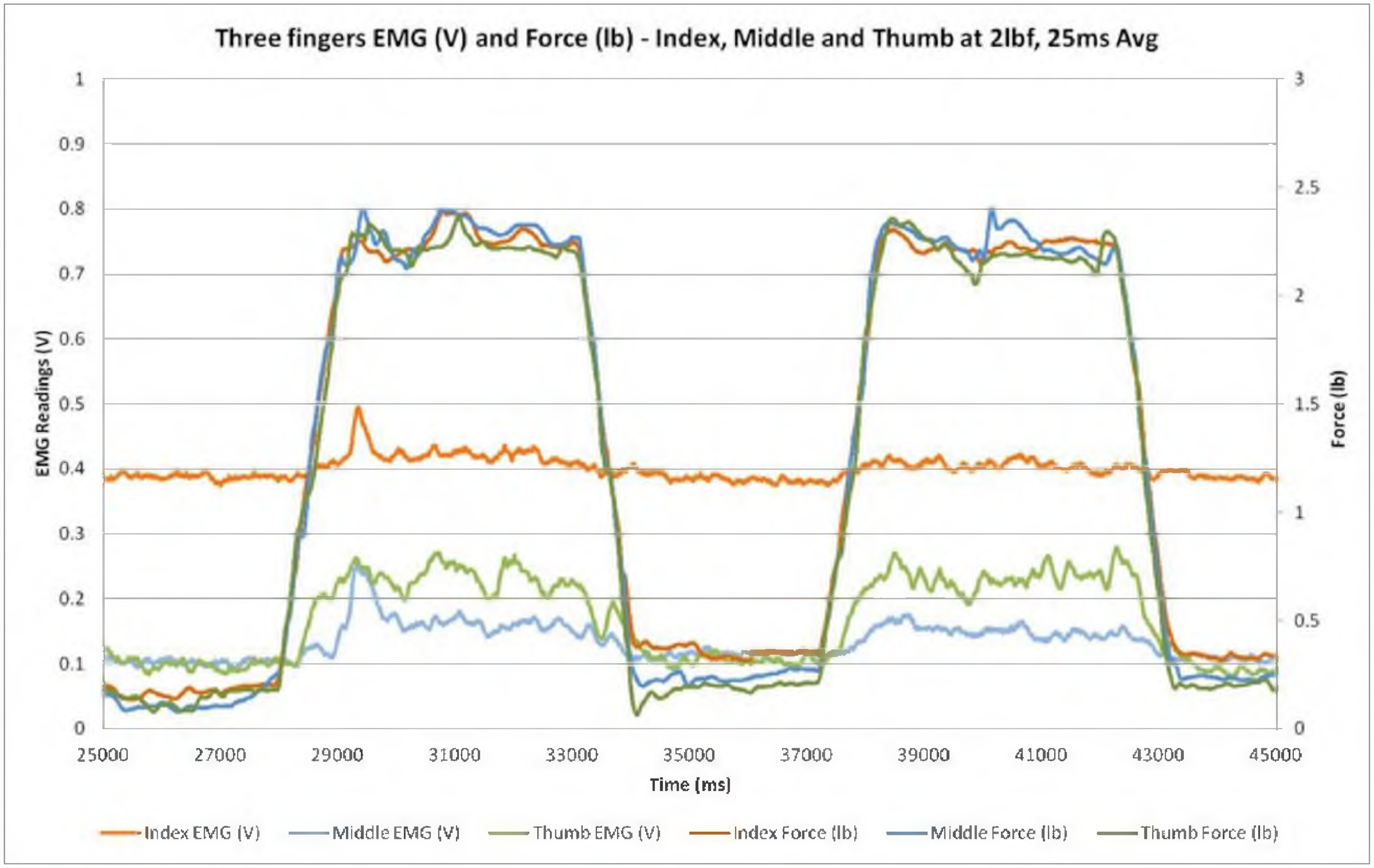


Figure 48: Detail of Figure 47, with filter applied

levels, most likely due to better electrode placement. The index finger shows a slight increase during the data-gathering period, but not as significant as the other two.

11.2.2. Force prediction capability

While a full data analysis for prosthetic control is beyond the scope of this thesis, the beginning steps of finding the relation between the EMG and force signals was performed.

The linear model of EMG and force signals was used for the force prediction, and is given in Equation 2.

$$[Force]_{3\times n} = [A] * [EMG]_{3\times n}, where\ Force\ and\ EMG\ are\ matrices: \begin{vmatrix} Index: & test\ 1 & test\ 2 & \dots & test\ n \\ Middle: & test\ 1 & test\ 2 & \dots & test\ n \\ Thumb: & test\ 1 & test\ 2 & \dots & test\ n \end{vmatrix}$$

$$A = FE^T(EE^T)^{-1} \tag{2}$$

The data used in this section used averaged values taken from the raw data. The mean of each recorded input was taken during the peak time, the flat top section labeled "EMG and force data gathered" in Figure 44. Each input had five averaged values per force level, one per force ramp-up. These five values were averaged into a single value used here. For example, the average of all five peaks' average for the index finger's force during one test was put in the Index row as the "test 1" value in **Error! Reference source not found.**.

This equation was populated with data for the first subject and evaluated. The A regression matrix for this set of data is shown in Table 11.

With further but similar tests that would serve to refine the values, reduce variability of the EMG placement, and increase the sample size, this regression matrix could be used in real-time control of a multiple DOF prosthetic hand by multiplying the incoming EMG signals by the A matrix to predict force.

To show the suitability of even this preliminary testing for prosthetic control, this process was used to predict the force values. Ideally this would be evaluated using the testing

Table 11: The A matrix, relating EMG and force

0.5976	7.6443	-1.2293
0.5337	1.43	4.448
0.6171	5.5899	-0.1314

data described above to predict the force values of a separate set of data in which the subject was asked to perform simple tasks, motions representative of what a prosthetic user would want to do. Two example task motions would be moving all fingers in a grasping motion, and moving one finger as if pushing a button.

Due to limited testing time, no task-oriented data set was gathered for true comparison of expected versus real results. However, the regression matrix gained from all tests was applied to each individual test separately. The results of one test are shown in Figure 49. In this test, the index and middle fingers were tested at 2lbs, and the thumb at zero applied force. The shape of the force profile is distinct in all the fingers, but especially in the middle finger. The predicted force for when zero force was applied tends to run high, but the trapezoidal force trend is visible. This error in predicting zero force has two likely sources. First is the lack of tests with the fingers receiving force in the other direction, which limits the dataset. The second is due to the method of finding the A regression matrix: only values from the peak of each force profile were used. This results in the exclusion of both a near-zero baseline force and its corresponding EMG activity from the A matrix. A full battery of tests designed to populate the A matrix for the purpose of personalized prosthetic control would include zero force steady-state data. This would allow for more accurate prediction of force values.

In the test shown in Figure 49, no force was actively applied to the thumb, but the thumb itself still applied a force, synchronized with the fingers. This is likely due to the movement of the rest of the hand more than any muscular motion of the thumb, but it is still

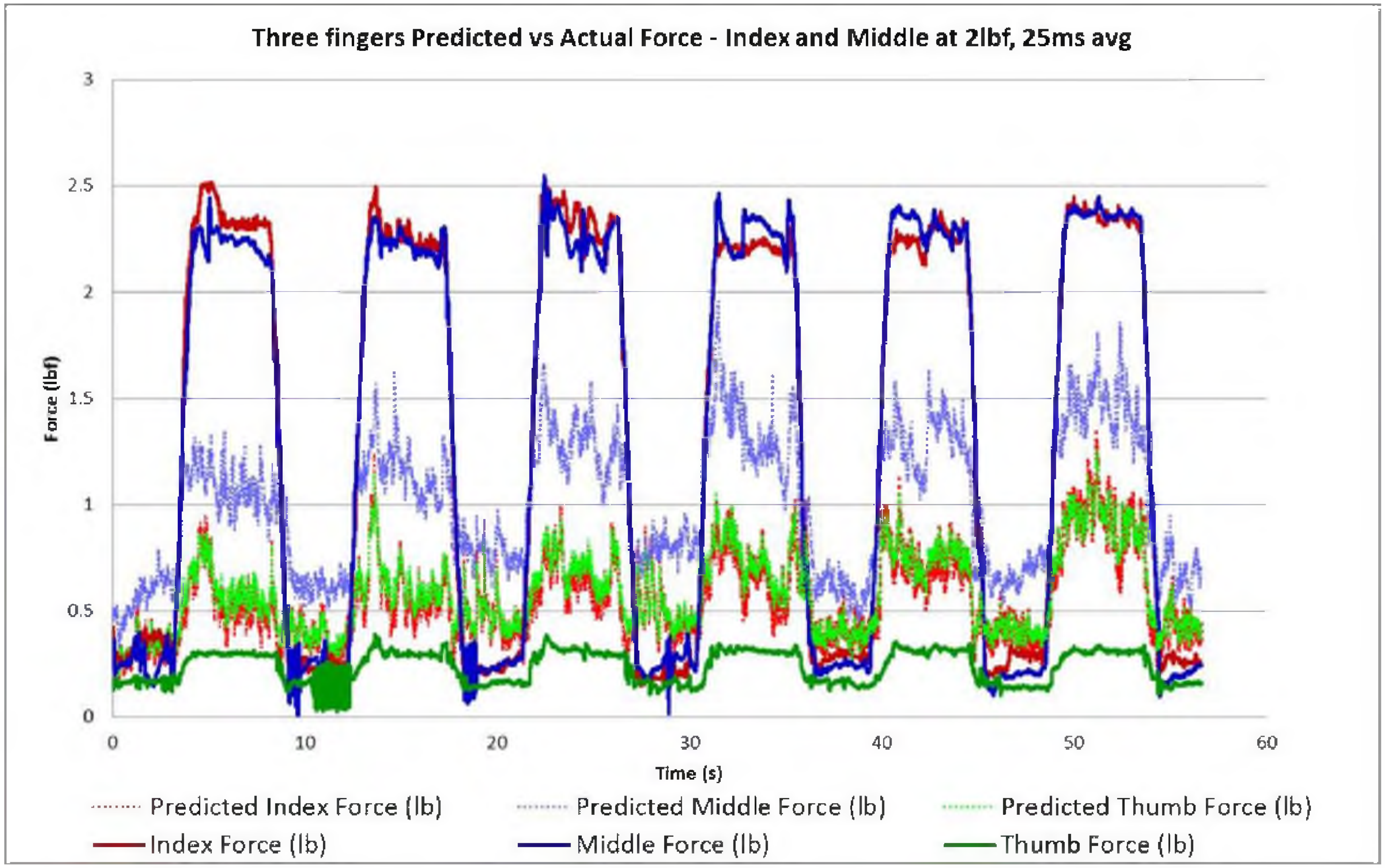


Figure 49: Individual finger force predictions compared to measured force for a single test

interesting to note the force prediction's approximation of the shape. It is likely that the thumb EMG values that caused this pattern in the predicted force come from cross-talk between the electrodes, but they could partially be a result of co-contraction of the muscles. This also would account for the slight force activity in the thumb.

Overall, the accuracy of this prediction is not sufficient for prosthetic control, primarily due to the amount of steady-state error in force levels. However, the preliminary tests show similar force profile shapes.

To show more of the force prediction results simultaneously, all five average force values during each test's data-gathering periods were averaged and compared to the average force prediction based on the EMG readings during the same periods. The averages of each finger and its prediction are shown in Figure 50, Figure 51, and Figure 52, broken up into tests of 1lbf, 2lbf, and 3lbf, respectively.

In Figure 50, Figure 51, and Figure 52, each finger's predicted and measured forces are shown next to each other, with the measured force in a darker hue on the right of the predicted. The prediction is shown next to the measured force for each of the tests, grouped along the bottom. Above each predicted force value is the standard error percentage relative to the measured force value, found using Equation 3. The average standard error for all tests was 21%.

$$Standard\ error = \sqrt{\frac{\Sigma\left(F_{measured} - F_{predicted}\right)^2}{n-1}} \qquad (3)$$

These graphs quantify the previous observation that the steady state force predictions need a more robust A regression matrix. A higher signal-to-noise ratio in the EMG readings would help make the predictions more accurate as to steady-state force level. A regression with a constant added was also performed, but did not help with steady-state error.

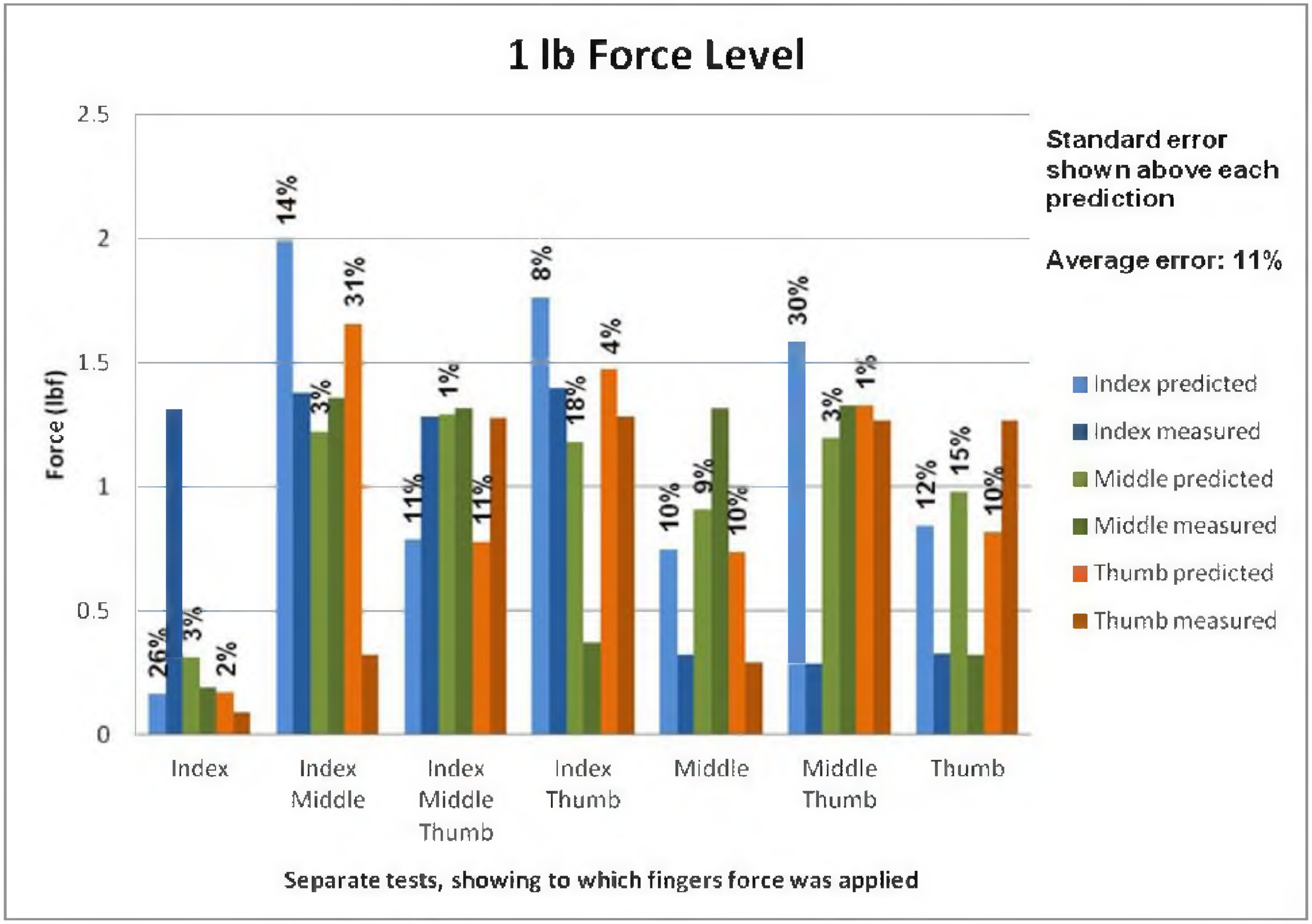


Figure 50: Comparison of measured and predicted forces, 1lbf tests

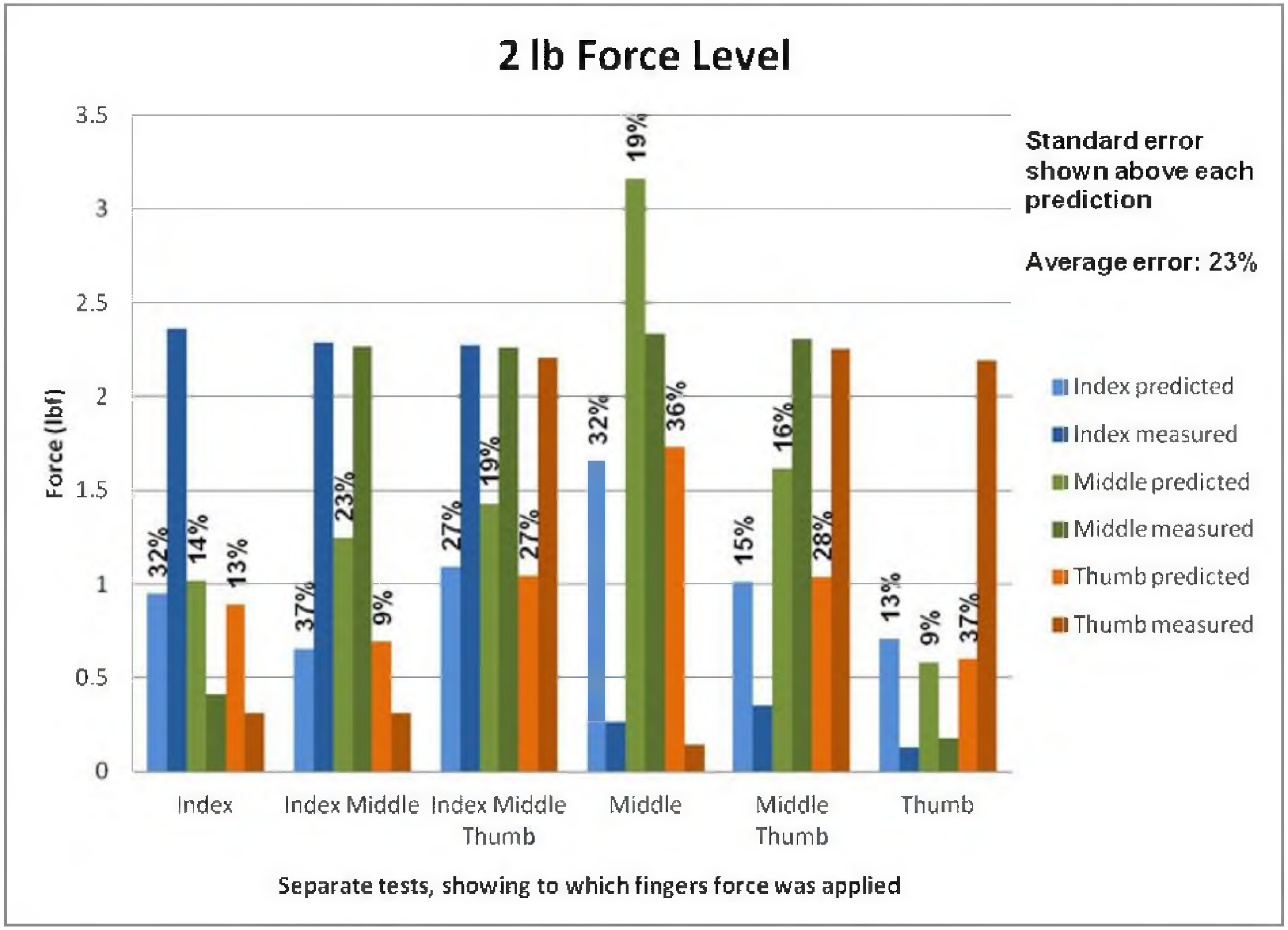


Figure 51: Comparison of measured and predicted forces, 2lbf tests

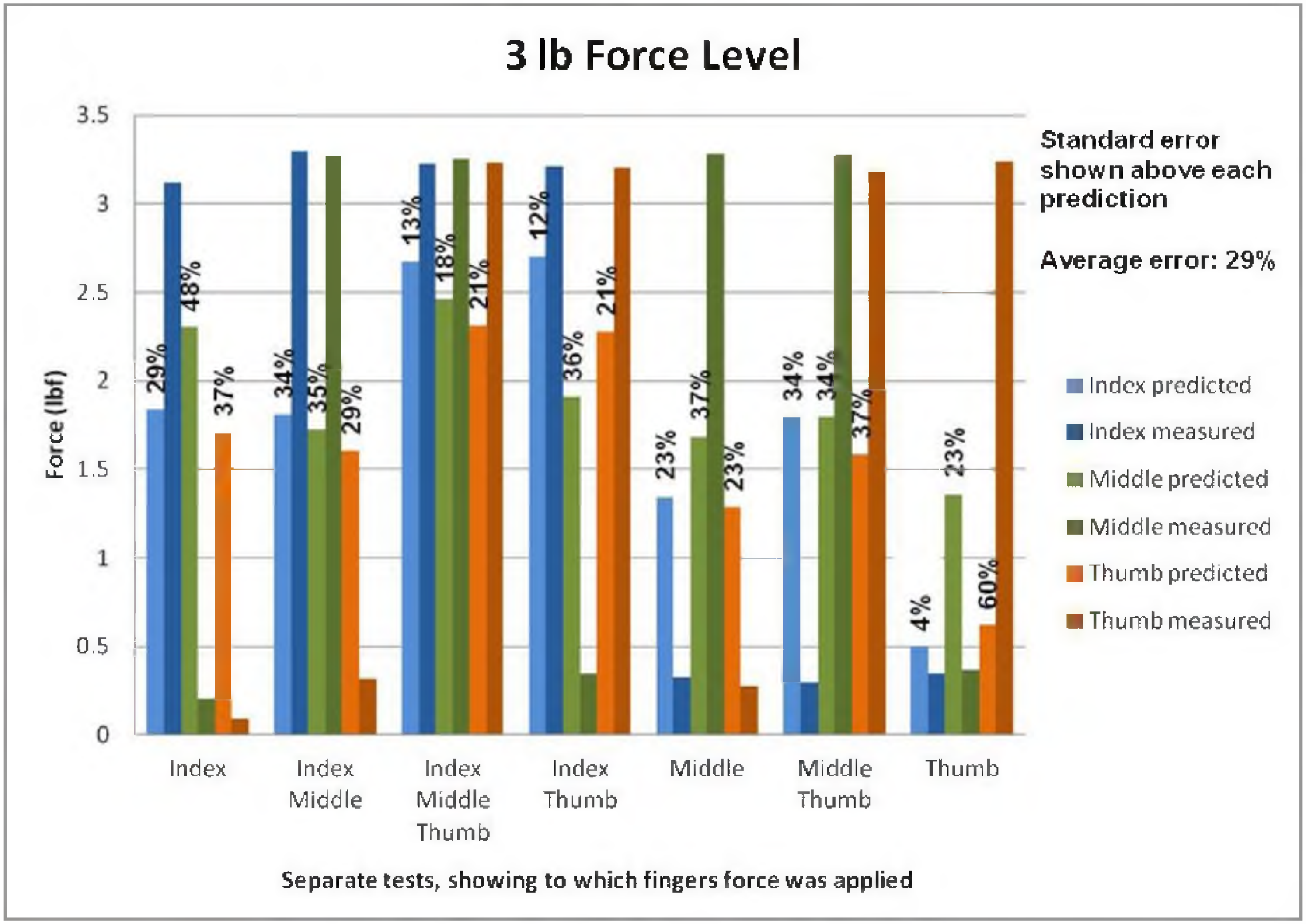


Figure 52: Comparison of measured and predicted forces, 3lbf tests

Some of the lowest error percentages occurred when all fingers were engaged. This validates the already standard method of controlling grasping speed or force in a single DOF prosthetic using EMG control; more muscles in the forearm performing a similar contraction makes for clearer readings. The model had more difficulty predicting the forces of fingers not receiving force, often over-predicting them. Possible contributing factors to this are cross-talk in the electrodes and co-contraction of the muscles. A more targeted form of electrode coupled with improved shielding from the ambient electromagnetic noise would help to correct this type of error.

11.3. Numerical analysis

After seeing the need for more independent EMG readings, the correlation matrices for EMG and force were found in order to see the quality of the data; see Table 12 and Table 13. Ideally, these would be equal to the identity matrix; values off the diagonal show more independence if closer to zero.

The force values are very independent from each other. This is to be expected: the force sensors are mechanically very isolated from each other. Also as expected, the EMG signals do not show the same high level of independence. This correlates with the previous findings from Figure 50, Figure 51, and Figure 52. The 0.6318 value between the thumb and middle finger is especially high, and helps explain why those two fingers tended to follow each other throughout the tests. This interdependence is partly because the muscles of the forearm often do not activate with complete independence (Schieber 1991). In addition, the surface EMG sensors used are not capable of isolating the deep muscle flexors associated with the fingers. This correlation should be improved for wired EMG sensors or IMES, which would target the specific muscles more accurately.

Table 12: Normalized correlation of EMG values

		EMG signals		
		Index	Middle	Thumb
EMG signals	Index	1	0.2054	0.166
	Middle	0.2054	1	0.6318
	Thumb	0.166	0.6318	1

Table 13: Normalized correlation of force values

		Force signals		
		Index	Middle	Thumb
Force signals	Index	1	0.058	-0.021
	Middle	0.058	1	0.0719
	Thumb	-0.021	0.0719	1

12. SUMMARY

This thesis has described the design, components, and function of hardware capable of measuring the individual forces and displacements of the fingers for correlation with the EMG signals.

The result is a functional device which fulfills the original goals. The required specifications were met or exceeded, as shown in Table 14. The recommended error for the accuracy of the sensors' readings are shown in Table 15.

More importantly, the ability to estimate individual finger forces through EMG readings was demonstrated at a minimal level. Furthermore, the preliminary tests show that individual finger force profile shapes can be derived from the recorded EMG data if further improvements are made to the system.

12.1. <u>Future work</u>

The results of this research have shown ways that the system could be improved in a future iteration. Several of these are related to the Arduino board and its limitations, while others were learned from experience with the current device. Direction for future testing of EMG signals for multiple DOF prosthetics is also provided.

12.1.1. Control and hardware

First and foremost, the shortcomings of the Arduino board could be remedied with the implementation of a PC-based control like that which was originally used for the first prototype.

Table 14: Original specifications and results

	Goal	Actual
Force Measurement (min):	45 N	67N
Velocity (min):	150 mm/s	480 mm/s
Travel (min):	100 mm	121mm
Applied force(min):	22 N	27N
Finger separation (max):	25mm	21mm

Table 15: Accuracy of measured data

Recommended Error for Measured Data	
Force	Position
±1.2N	±2.5mm

This would eliminate the failure of the Arduino board to correctly maintain the position, and allow for more intuitive Position control. This would in turn necessitate reworking of the code to account for any variations in output from a different controller, as well as a likely retuning of the control gains, but the gains through this change would be worth the effort.

A more streamlined version of the circuit board could improve access to the adjustment potentiometers, and assist in debugging. It is particularly recommended that the three stages of the load cell processing circuit be reordered so that the balancing circuit comes first. This would allow for the same calibration curve to be used for all force data read from the front panel. It could also improve the layout of the cable connections, which would help organize the inside of the control box. Planning and having a printed circuit board (PCB) commercially made is recommended for simplicity and function, as well as reducing the time-consuming and difficult task of making the circuits by hand. Commercially available solutions for simultaneous amplification and offsetting should be investigated.

The source of the noise associated with the input knob could be better understood. This may also be an issue that can only be resolved by replacing the Arduino with a microcontroller that has better isolated functions.

As noted in the Design section, the three shafts need to be very parallel for smooth motion. A future version could make the shafts easier and more natural to align, which would aid in reassembly after cleaning or inspection.

Testing showed that the system has a slight directional bias in smoothness in the lead screw. This effect is only noticeable at very low speeds. The motion is smoother as the finger ring extends, with a more jerky motion as it retracts (see slight steps in Figure 41 along the right side). It may be possible to reduce or eliminate this effect by using a different drive system with less natural directional bias. Possibilities include a ball screw or a linear actuator.

Although the chosen DC motors met the requirements well and were very inexpensive, their power connection tabs can break off and were difficult to repair. Other motors with a more protected connection would prevent this.

12.1.2. Testing and application to prosthetic control

As has been mentioned previously, more isolated EMG signals would help with the force prediction accuracy. There is a limit to how isolated the signals can be due to muscular interactions, but they could be improved greatly with a more targeted electrode type (wired or IMES are examples). Improved shielding from electromagnetic noise would also help improve the predictions by giving a cleaner signal.

A more constrained shoulder joint would help to isolate the subject's hand muscles more reliably.

More data sets are needed to determine a robust A regression matrix. Repeating the same tests again would strengthen the relations found for the linear model by varying the EMG reading locations as would be typical of a prosthetic user regularly donning an electrode device. This would include baseline level analysis for better accuracy on predictions that should be zero force. Also beneficial would be tests in which each of the fingers received individual force levels for more rich data, rather than the same force value applied to any fingers involved. This would be a simple step, as the device was already configured for these types of independent inputs. It would also be possible to increase the maximum level of force in a test, as the subjects reported that the 1lbf level was so low as to be difficult to discern and the 3lbf level was not too straining. Increased force levels would require larger resting periods to avoid muscle fatigue.

An important part of the expanded data set is the gathering of data that are not part of a formal test. The subject would be instructed to move the fingers in common motions such as a grasp or a two finger pinch. These data sets would allow for comparisons of predicted force in a more natural activity than the trapezoidal force profile used here.

Finally, the ultimate step would be to apply these force predictions to a multiple DOF prosthetic hand with the goal of improving their intuitiveness and usefulness to an amputee.

12.2. Conclusion

In order to create more natural motion in prosthetic devices, the relation between the oft-used EMG signals and the mechanical motion of the limb needs to be better understood. This relationship grows in complexity with multiple fingers and complicated motions. This thesis presents a device designed to measure the force, position, and velocity of each finger independently for comparison with EMG data. Data from a representative test were presented, and a method of predicting the force of each finger was shown, with the resulting prediction generally following the measured force profile. This design will allow improved data to be independently gathered from more fingers, representing more complex motions of the hand, helping the EMG-motion relation to be comprehended more fully.

APPENDIX A:

LABELED CIRCUIT PHOTOS

Large, detailed diagrams of the actual circuit board with all its connections are shown in this appendix.

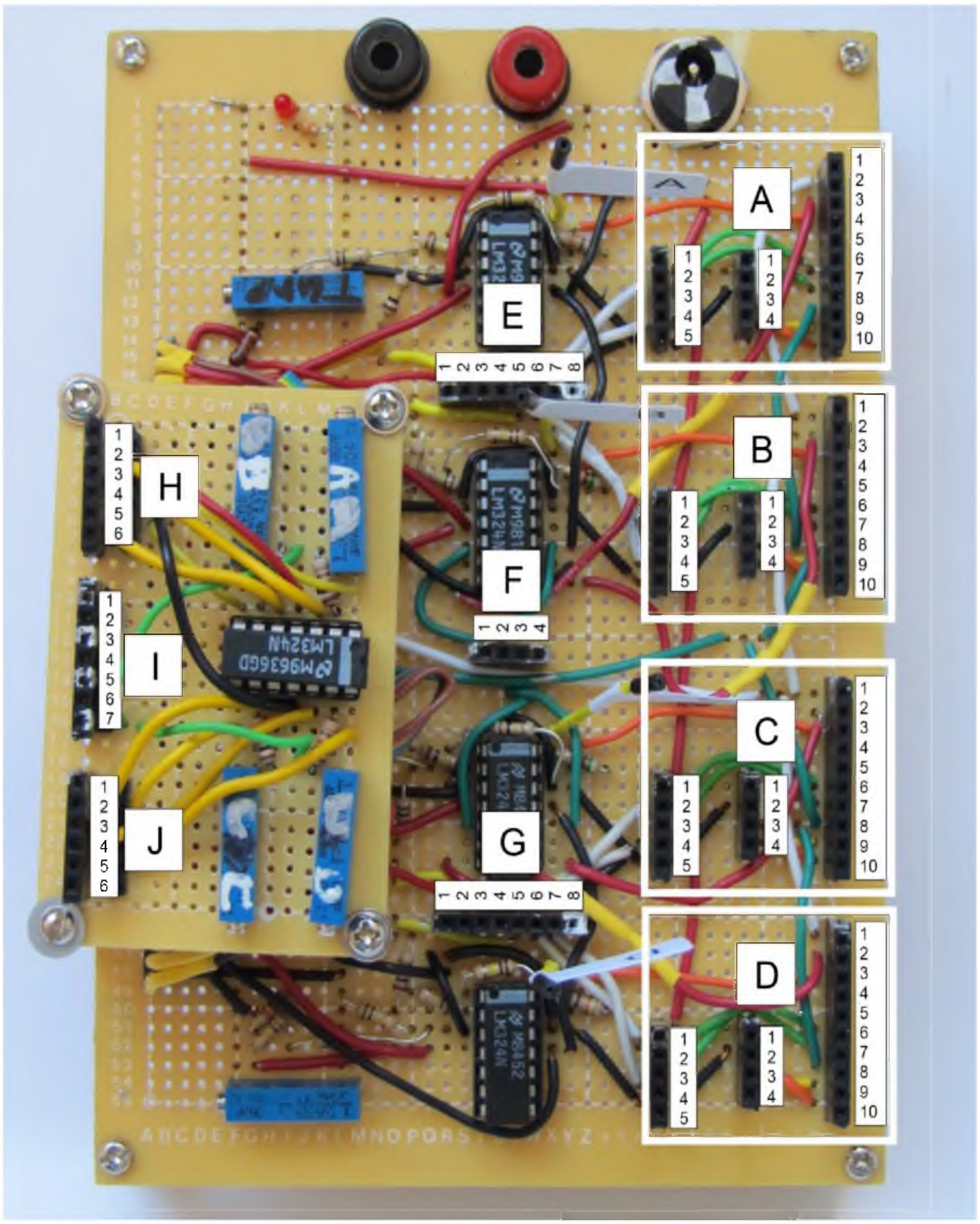


Figure 53: Circuit with all connectors labeled

Table 16: Descriptions of labeled connectors in Figure 53

	A, B, C and D			E: Front panel outputs	F: Front panel inputs	G: Front panel outputs	H: Loadcells A and B	I: Front panel switches	J: Loadcells C and D
Pin	Encoder	Servo drive	Arduino						
1	5V	5V	Reset switch	5V	A	5V	5V	5V	5V
2	Chan A	GND	Analog input	Load cell A	B	Load cell C	+15V	Reset switch	+15V
3	Index	Direction	Loadcell in	Enc A Chan A	C	Enc C Chan A	Load cell A	none	Load cell C
4	Chan B	PWM	none	Enc A Chan B	D	Enc C Chan B	-15V	none	-15V
5	GND		none	Load cell B		Load cell D	Load Cell B	Mode switch	Load Cell D
6			Enc Chan A	Enc B Chan A		Enc D Chan A	GND	none	GND
7			Enc Chan B	Enc B Chan B		Enc D Chan B		none	
8			Mode switch	GND		GND			
9			PWM out						
10			Motor dir out						

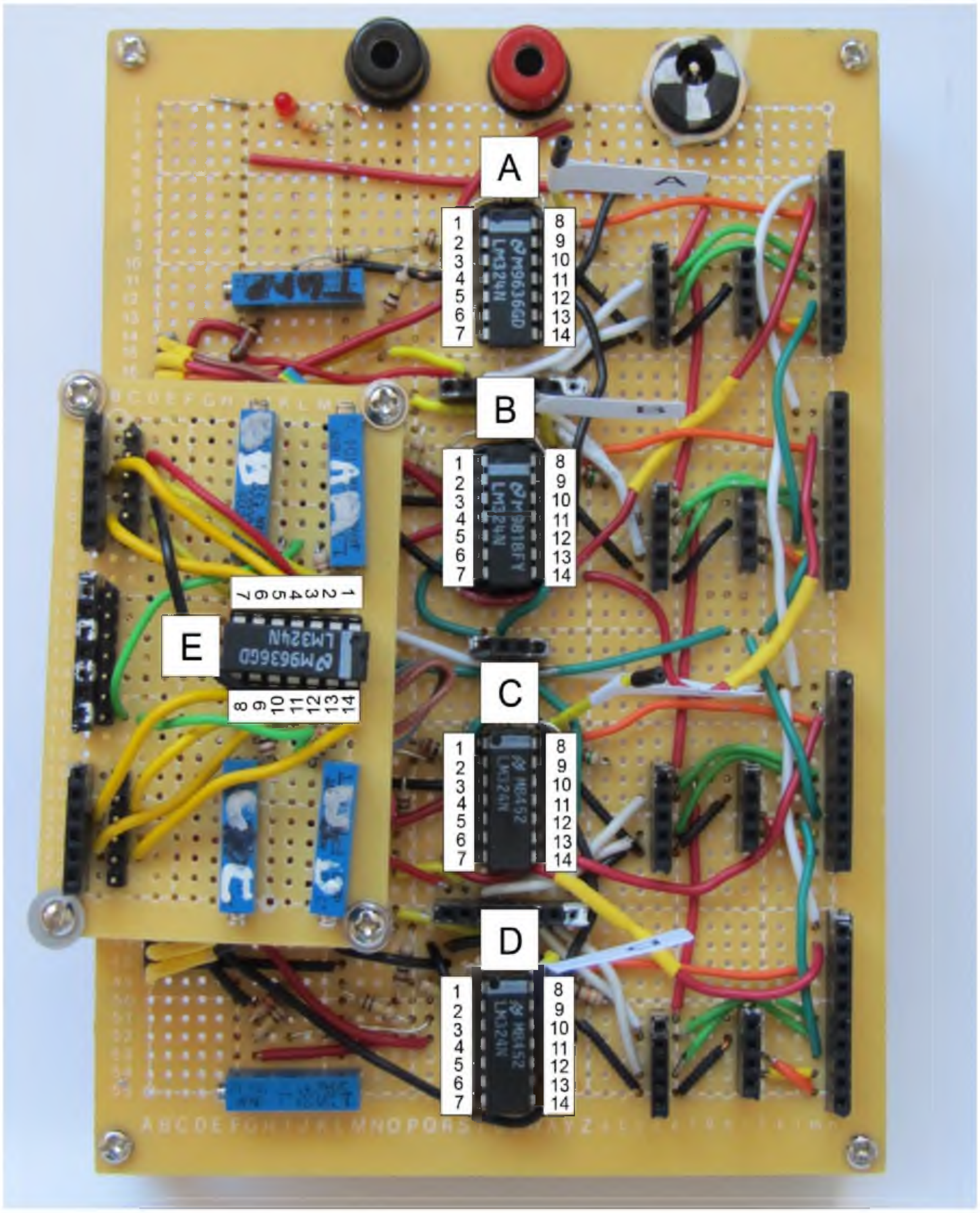


Figure 54: Circuit with all op-amp connections labeled

Table 17: Descriptions of labeled op-amp connections in Figure 54

Pin	Op Amp A	Op Amp B	Op Amp C	Op Amp D	Op Amp E
1	Vout: To inverting	Vout: To inverting	Vout: To inverting	Vout: To inverting	Vout: To Op Amp A
2	V(-)in: Summer amp	V(-)in: Summer amp	V(-)in: Summer amp	V(-)in: Summer amp	V(-)in: Gain resistor pot
3	V(+)in: Summer amp	V(+)in: Summer amp	V(+)in: Summer amp	V(+)in: Summer amp	V(+)in: Input A from EL-1040
4	V(+)s: +15V	V(+)s: +15V	V(+)s: +15V	V(+)s: +15V	V(+)s: +15V
5	none	V(+)in: A analog in	V(+)in: D analog in	none	V(+)in: Input B from EL-1040
6	none	V(-)in: A Voltage follower	V(-)in: D Voltage follower	none	V(-)in: Gain resistor pot
7	none	Vout: To Arduino A	Vout: To Arduino D	none	Vout: To Op Amp B
8	Vout: To Arduino	Vout: To Arduino	Vout: To Arduino	Vout: To Arduino	Vout: To Op Amp C
9	V(-)in: Inverting amp	V(-)in: Inverting amp	V(-)in: Inverting amp	V(-)in: Inverting amp	V(-)in: Gain resistor pot
10	V(+)in: Inverting amp	V(+)in: Inverting amp	V(+)in: Inverting amp	V(+)in: Inverting amp	V(+)in: Input C from EL-1040
11	V(-)s: -15V	V(-)s: -15V	V(-)s: -15V	V(-)s: -15V	V(-)s: -15V
12	none	V(+)in: B analog in	V(+)in: C analog in	none	V(+)in: Input D from EL-1040
13	none	V(-)in: B Voltage follower	V(-)in: C Voltage follower	none	V(-)in: Gain resistor pot
14	none	Vout: To Arduino B	Vout: To Arduino C	none	Vout: To Op Amp D

APPENDIX B:

FULL MICROCONTROLLER CODE

The actual code used by the Arduinos to control each device is shown here:

```
//Position and Force control code

/* Note: All four devices have different gains in Force control mode in order to run more smoothly. If uploading new code to   all Arduinos,
 remember to put in the correct gain by changing the value of the pgainForce variable in the Global variables section (two sections down).
 It looks like:

 unsigned int pgainForce = X;

 For each device, input a different value:

 Device A pgainForce: 12
 Device B pgainForce: 15
 Device C pgainForce: 11
 Device D pgainForce: 14

 */

//#include stuff
#include <C:\Users\Brian\Desktop\Arduino\hardware\tools\avr\avr\include\avr\interrupt.h> /* To compile code, you'll have to locate these files on your own computer and */
#include <C:\Users\Brian\Desktop\Arduino\hardware\tools\avr\avr\include\avr\io.h>       /*type in the correct file locations. It will be in a similar folder. */
#define encoderPinA  2
#define encoderPinB  3
#define modeSelect  4
#define PWMout 5
#define motordirection 6
#define loadcell 16
#define pot 17
#define LEDPin 13


//********************************** Global variables *****************************
unsigned int pgainForce = 11; /* This gain needs to be a different value depending upon the selected device, see note above. */
unsigned int pgainPos = 15; // Stays at 15 for all devices.
double effort1 = 0;
double effort2 = 0;
double effort2prev = 0;
double forceerror1 = 0;
long poserror1 = 0;
long position1 = 0;
int force1 = 0;
int forcegoal = 512;
long posgoal = 0;
int forceinit = 512;
int speed1 = 0;
float startcounter = 0.0;
int chan1 = 0;
int chan2 = 0;
char direction = 0;
int potread = 0;
int mode = 0;

//______________________________________________________________________________
//Interrupts        ******************************************************************************
//______________________________________________________________________________

//  Two input capture interrupts as edge finders for encoder

//IC interrupt – channel A of encoder - Pin 2
ISR(INT0_vect)
{
  chan1 = digitalRead(encoderPinA);
  if (chan1 == 0 && chan2 == 0)  // Quadrature logic
  {
    position1++;   //  Increment position1 by one

  }
  else if (chan1 == 1 && chan2 == 1)    // Quadrature logic
  {
    position1++;   //  Increment position1 by one
  }
}

//IC2 interrupt - channel B of encoder - pin 3
ISR(INT1_vect)
{
  chan2 = digitalRead(encoderPinB);
  if (chan1 == 0 && chan2 == 0)    // Quadrature logic
  {
    position1--;   // Decrease by one
  }
  else if (chan1 == 1 && chan2 == 1)    // Quadrature logic
  {
    position1--;   // Decrease by one
  }
}

// Timer buffer overrun interrupt – for regularly timed motor updates

ISR(TIMER2_OVF_vect) {

  if (mode == HIGH)    // If in Position control mode
  {
    interrupts();
    potread = analogRead(pot);
    digitalWrite(LEDPin, HIGH);
    posgoal = (potread-512)*4;
    poserror1 = position1 - posgoal;

    effort1 = pgainPos*poserror1;

    invertedmotor(effort1*startcounter);
    digitalWrite(LEDPin, LOW);
    if (startcounter < 0.9999)
    {/* startcounter helps the devices start more smoothly if their initial input goal is not close to where they are, but scaling up the
effort over about 1 sec */
      startcounter += 0.0005;
    }

    TCNT2 = 250; /* This number has to count to 255 outside the interrupt before the next interrupt starts. 250 worked well for
running speed, 100 is better for debugging with prints. */
  }

  if (mode == LOW)    // If in Force control mode
  {

    //interrupts();
    force1 = analogRead(loadcell);
    digitalWrite(LEDPin, HIGH);
    forcegoal = analogRead(pot)-512;
    forceerror1 = forcegoal-(force1-forceinit);
    effort1 = pgainForce*forceerror1;

    // Digital Filter, 8/1
    effort2 = effort1*0.0002362 + 0.9998*effort2prev;
    effort2prev = effort2;

    invertedmotor(effort1*startcounter);
    digitalWrite(LEDPin, LOW);
    if (startcounter < 0.9999)
    {/* startcounter helps the devices start more smoothly if their initial input goal is not close to where they are, but scaling up the
effort over about 1 sec */
      startcounter += 0.0005;
    }
    TCNT2 = 250; /* This number has to count to 255 outside the interrupt before the next interrupt starts. 250 worked well. */
  }
}

// Motor control function
/* This file controls the BD15A8 servo drive, which needs an inverted PWM when using the optically isolated inputs */
void invertedmotor(long speed)
{
  direction = 1;       // 1 is CW, 0 is CCW
  if (speed < 0)
  {
    direction = 0;
    speed = speed * (-1);
  }

  if (direction == 1)  // Sets direction pin
  {
    digitalWrite(motordirection,HIGH);
  }
  else
  {
    digitalWrite(motordirection,LOW);
  }

  // clamp speed between 0 and 10000 which is 100%
  // duty cycle
  if    (speed > 10000)
  {
    speed = 10000;
  }
  speed1 = 255-(speed*255/10000);  /* Scales the motor control effort to the required 8 bit number for a PWM output, and inverts
PWM */
  analogWrite(PWMout,speed1);
```

```
}

// From Mitch 6/29

void setPwmFrequency(int pin, int divisor) {
 byte mode;
 if(pin == 5 || pin == 6 || pin == 9 || pin == 10) {
   switch(divisor) {
   case 1:
     mode = 0x01;
     break;
   case 8:
     mode = 0x02;
     break;
   case 64:
     mode = 0x03;
     break;
   case 256:
     mode = 0x04;
     break;
   case 1024:
     mode = 0x05;
     break;
   default:
     return;
   }
   if(pin == 5 || pin == 6) {
     TCCR0B = TCCR0B & 0b11111000 | mode;
   }
   else {
     TCCR1B = TCCR1B & 0b11111000 | mode;
   }
 }
 else if(pin == 3 || pin == 11) {
   switch(divisor) {
   case 1:
     mode = 0x01;
     break;
   case 8:
     mode = 0x02;
     break;
   case 32:
     mode = 0x03;
     break;
   case 64:
     mode = 0x04;
     break;
   case 128:
     mode = 0x05;
     break;
   case 256:
     mode = 0x06;
     break;
   case 1024:
     mode = 0x7;
     break;
   default:
     return;
   }
   TCCR2B = TCCR2B & 0b11111000 | mode;
 }
}

//_________________________________________________________________________________
// Loops
//_________________________________________________________________________________
```

```
// Setup loop for inititalization
void setup()
{
  noInterrupts();
  //  Serial.begin(9600);
  //  Serial.println("Initializing...");
  analogWrite(PWMout,255);
  pinMode(PWMout, OUTPUT); //PWM output
  pinMode(motordirection, OUTPUT); //motor direction output 1
  pinMode(loadcell, INPUT); //Load cell analog - Pin 14, A0
  pinMode(pot, INPUT); //Pot analog - Pin 15, A1
  pinMode(encoderPinA, INPUT);    // Encoder Chan A - Pin D2
  pinMode(encoderPinB, INPUT);    // Encoder Chan B - Pin D3
  pinMode(modeSelect, INPUT);    // Switch  - Pin D4
  digitalWrite(modeSelect, HIGH);  // Turn on a pull-up resistor for mode pin
  pinMode(LEDPin, OUTPUT); // Make onboard LED an output

  // Codes to set up encoder interrupts
  EIMSK |= ( 1 << INT0); // for arduino-pin2
  EIMSK |= ( 1 << INT1); // for arduino-pin3
  TIMSK2 |= (1<<TOIE2); // set interrupts=enabled
  EICRA |= ( 1 << ISC00); // on int0 CHANGE
  EICRA |= ( 0 << ISC01); //
  EICRA |= ( 1 << ISC10); // on int1 CHANGE
  EICRA |= ( 0 << ISC11); //

  // From Mitch 6/29 email
  setPwmFrequency(5, 1);  //*setPwmFrequency(int pin, int divisor); The base frequency for pins 5 and 6 is 62500 Hz. - Divisors: pins
5, 6, 9 and 10 are: 1,
  , 64, 256, and 1024. */

  // Codes to set up Timer2 Overflow Interrupt
  TCCR2A = 0;
  TCCR2B = 1<<CS22 | 0<<CS21 | 0<<CS20;  /* Page 162 of AVR datasheet gives prescalers for timer */
  TIMSK2 = 1<<TOIE2;

  mode = digitalRead(modeSelect);

  analogWrite(PWMout,255);
  position1 = 0;
  analogWrite(PWMout,255);
  interrupts();
}
```

REFERENCES

Advanced Motion Controls. "Analog Servo Drive BD15A8." *Advanced Motion Controls.* n.d. http://www.a-m-c.com/download/datasheet/bd15a8.pdf (accessed January 4, 2010).

—. "PS2X3 and PS4X3 Series." *Advanced Motion Controls.* n.d. http://www.a-m-c.com/download/datasheet/ps2x3h24.pdf (accessed January 4, 2010).

Almström, Christian, and Peter Herberts. "Control of multifunctional prosthetic hands using pattern recognition technique." (International Journal of Rehabilitation Research) 1, no. 3 (1978).

Andersen, Richard A., Eun Jung Hwang, and Grant H. Mulliken. "Cognitive Neural Prosthetics." (Annual Review of Psychology) 61 (2009).

Arduino. *Arduino Language Reference.* n.d. http://arduino.cc/en/Reference/HomePage (accessed June 15, 2010).

Arduino IDE. *Arduino Duemilanove.* n.d. http://www.arduino.cc/en/Main/ArduinoBoardDuemilanove (accessed June 15, 2010).

Baker, Justin J., Erik Scheme, Kevin Englehart, Douglas T. Hutchinson, and Bradley Greger. "Continuous Detection and Decoding of Dexterous Finger Flexions With Implantable MyoElectric Sensors." *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 18, no. 4 (August 2010): 424-432.

Baker, Justin J., et al. "Decoding Individuated Finger Flexions with Implantable MyoElectric Sensors." *EMBS Conference.* Vancouver, British Columbia, Canada: IEEE, 2008. 193-196.

Baker, Justin, William Bishop, Spencer Kellis, Todd Levy, Paul House, and Bradley Greger. "Multi-scale Recordings for Neuroprosthetic Control of Finger Movements." Minneapolis: IEEE EMBS, 2009.

BeBionic. *The Hand Grip Patterns.* n.d. http://www.bebionic.com/grip-patterns/ (accessed 11 2011).

Electronic Innovation Corp (EIC). *Electronic Innovation Corp (EIC).* n.d. http://www.designcircuit.com/ (accessed August 3, 2009).

Farrell, Todd R., and Richard F. ff. Weir. "A Comparison of the Effects of Electrode Implantation and Targeting on Pattern Classification Accuracy for Prosthesis Control." IEEE Transactions on Biomedical Engineering, 2008.

Herbach and Rademan. *3000 RPM Reversible DC PM Field 12 VDC.* n.d. http://www.herbach.com/Merchant2/merchant.mv?Screen=PROD&Store_Code=HAR&Product_Code=TM00MTR4415&Category_Code=MTR (accessed January 4, 2010).

Honeywell. *Miniature Load Cells.* n.d. http://content.honeywell.com/sensing/sensotec/loadcell.asp?category=mini (accessed January 4, 2010).

Huang, He, Fan Zhang, Yan L Sun, and Haibo He. "Design of a robust EMG sensing interface for pattern classification." (Journal of Neural Engineering) 7, no. 056005 (2010).

Meek, Sanford G., Stephen C. Jacobsen, and R. Rees Fullmer. "Control Inputs to a Multiple Degree of Freedom Artificial Arm." *IEEE Frontiers of Engineering and Computing in Health Care*, 1984: 12.6.1-5.

Micera, Silvestro, Jacopo Carpaneto, and Stanisa Raspopovic. "Control of Hand Prostheses Using Peripheral Information." IEEE Reviews in Biomedical Engineering, 2010.

Microchip. *dsPIC30F4011.* n.d. http://www.microchip.com/wwwproducts/Devices.aspx?dDocName=en010337 (accessed February 2010).

Musallam, S., B. D. Corneil, B. Greger, H. Scherberger, and R. A. Anderson. "Cognitive Control Signals for Neural Prosthetics." *Science* 305 (2004): 258-262.

Nave, Carl R. *Hyperphysics.* n.d. http://hyperphysics.phy-astr.gsu.edu/hbase/frict2.html (accessed November 16, 2011).

Nook Industries. *3/8-2 Inch Acme Screw Assemblies.* n.d. http://www.nookindustries.com/acme/AcmeInchInfo.cfm?id=8 (accessed January 4, 2010).

—. *Acme & Lead Screw Assembly Glossary and Technical Data.* n.d. http://www.nookindustries.com/acme/AcmeGlossary.cfm#Backdriving (accessed January 4, 2010).

Ortmann, Valerij, and Boris Kh. Baziyan. "Intracortical Neural Interface for Prosthetic Applications." Lyon, France: IEEE EMBS, 2007.

Ottobock. *SensorHand Speed.* n.d. http://www.ottobockus.com/cps/rde/xchg/ob_us_en/hs.xsl/6953.html?id=15060#t15060 (accessed 11 2011).

PBC Linear. *FM Metric Closed Linear Plain Bearing.* n.d. http://www.pacific-bearing.com/ISOMetricClosedPlaneBearing-FM.aspx (accessed January 4, 2010).

Reaz, M. B. I., M. S. Hussain, and F. Mohd-Yasin. "Techniques of EMG signal analysis: detection, processing, classification and applications." (Biological Procedures Online) 8, no. 1 (2006).

Scheme, Erik, and Kevin Englehart. "Electromyogram pattern recognition for control of powered upper-limb prostheses: state of the art and challenges for clinical use." (Journal of Rehabilitation Research and Development) 48, no. 6 (2011).

Schieber, Marc H. "Individuated Finger Movements of Rhesus Monkeys: A Means of Quantifying the Independence of the Digits." (Journal of Neurophysiology) 65, no. 6 (1991).

Sears, Harold H., and Julie Shaperman. "Proportional Myoelectric Hand Control: An Evaluation." (American Journal of Physical Medicine and Rehabilitation) 70, no. 1 (1991).

Smith, Ryan J., Francesco Tenore, David Huberdeau, Ralph Etienne-Cummings, and Nitish V. Thakor. "Continuous Decoding of Finger Position from Surface EMG Signals for the Control of Powered Prostheses." Vancouver, British Columbia: IEEE EMBS Conference, 2008.

Sylvester, Nicholas D. "Friction Perception and Tactile Feedback." (University of Utah) MS Thesis (2006).

Tenore, Francesco, Ander Ramos, Amir Fahmy, Soumyadipta Acharya, Ralph Etienne-Cummings, and Nitish V. Thakor. "Towards the Control of Individual Fingers of a Prosthetic Hand Using Surface EMG Signals." Cite Internationale, Lyon, France: IEEE EMBS, 2007.

US Digital. *E5 Optical Kit Encoder.* n.d. http://usdigital.com/products/encoders/incremental/rotary/kit/e5/ (accessed January 4, 2010).

Zhou, Ping, et al. "Decoding a New Neural-Machine Interface for Control of Artificial Limbs." (Journal of Neurophysiology) 98 (August 2007): 2974-2982.