
Intent Recognition for Prosthetic Arms

UNDERGRADUATE THESIS

*Submitted in partial fulfillment of the requirements of
BITS F421T Thesis*

By

Samaksh JUDSON
ID No. 2019A4TS0278P

Under the supervision of:

Dr. Dibakar SEN
&
Dr. Venkatesh K. P. RAO



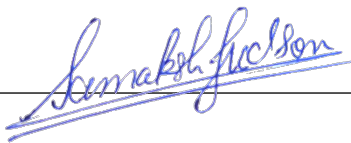
BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI, PILANI CAMPUS
December 2022

Declaration of Authorship

I, Samaksh JUDSON, declare that this Undergraduate Thesis titled, 'Intent Recognition for Prosthetic Arms' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: _____



Date: December 9th, 2022.

“The worthwhile problems are the ones you can really solve or help solve, the ones you can really contribute something to. . . . No problem is too small or too trivial if we can really do something about it.”

Richard Feynman

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI, PILANI CAMPUS

Abstract

Bachelor of Engineering (Hons.)

Intent Recognition for Prosthetic Arms

by Samaksh JUDSON

This thesis deals with the development and fabrication of a novel control system for a hybrid-powered prosthetic arm. The documentation follows a chronological sequence of problem identification and solution, all the while providing scientific evidence to back each decision. Research gaps have been identified by performing an extensive literature review of existing technology and dealt with during the progression of the project. The literature ends by highlighting a potential roadmap for further development of the system, which is once again backed by snippets of code and relevant research in the domain...

Acknowledgements

I would like to thank my thesis supervisor, Dr D. Sen, for this amazing opportunity and my co-supervisor, Dr V. K. P. Rao, for their guidance and unwavering faith in me. This endeavour would not have been nearly as successful without their skill and experience in the domains of Robotics, Controls and Design. To them, I owe a debt of gratitude. . . .

Contents

Declaration of Authorship	i
Abstract	iii
Acknowledgements	iv
Contents	v
List of Figures	vii
Abbreviations	viii
1 Introduction	1
1.1 Overview	1
1.2 Motivation	2
1.3 Background Study	3
1.3.1 Literature Review	3
1.3.2 Analysis of Human Movement	5
1.3.3 Preexisting Set-Up	5
2 Characterization of FSR	8
2.1 The Force Sensitive Resistor	8
2.2 FSR Circuit	9
2.2.1 Purpose	9
2.2.2 Task	10
2.3 FSR Buffered Circuit	10
2.3.1 Purpose	10
2.3.2 Task	11
2.4 Feedback-based Linear Actuator	12
2.4.1 Purpose	12
2.4.2 Task	12
2.5 Cycle-Time Analysis of Resistance	14
2.5.1 Purpose	14
2.5.2 Task	14
2.6 Sampling Window	15
2.6.1 Purpose	15

2.6.2 Task	15
2.7 Results	16
3 Control Scheme	18
3.1 Dynamic Servo Response	18
3.1.1 Purpose	18
3.1.2 Task	19
3.2 Implementation	21
4 Future Work	23
5 Conclusion	25
A Microcontroller Code	26
A.1 FSR Cycle Time	26
A.2 FSR Sampling Window	28
A.3 Code to potentially reduce fluctuations	30
B MATLAB Code	32
B.1 FSR Data Storage and Analysis	32
B.2 Servo Response Code	33

List of Figures

1.1	Various kinds of Sensor Data	2
1.2	PURAK Prosthetic Arm	3
1.3	Quaternion Analysis of Tracker Data	5
1.4	An Arduino Board	6
2.1	Components of an FSR	8
2.2	FSR Resistance	9
2.3	Voltage Divider Circuit	10
2.4	An Operational Amplifier	11
2.5	The Op-Amp IC	11
2.6	New Voltage Divider Circuit	12
2.7	Servo Driven Linear Actuator	13
2.8	Resistance of FSR through a complete Loading-Unloading cycle	14
2.9	Depiction of a Sampling Window	16
2.10	Hysteresis Graph for FSR	17
3.1	Linearly Transferred Servo Response	19
3.2	Amended Servo Response	20
3.3	Motor and Motor Mount	21
3.4	L298 Motor Controller	21
3.5	Depiction of an H Bridge	22
4.1	Selective Delay Control Scheme	23
4.2	Depiction of the relationship between Jitter and Latency	24

Abbreviations

LDR	L ight D ependent R esistor
LDA	L inear D iscriminant A nalysis
DBN	D ynamic B ayesian N etwork
FSR	F orce S ensitive R esistor
SOM	S elf O rganizing M aps
ADL	A ctivities of D aily L iving
TTL	T ransistor T ransistor L ogic
SFS	S equential F eature S election
IMU	I nertial M easurement U nit
EMG	E lectro M yo G raphy
Op Amp	O perational A mplifier
IC	I ntegrated C ircuit

Chapter 1

Introduction

1.1 Overview

A prosthesis or a prosthetic device plays an integral role in the rehabilitation of an individual who has suffered from the loss or has undergone an amputation of a limb or an extremity. Whether the loss occurs due to an accident or due to a congenital defect, such as partial limb deficiencies in children, a prosthetic apparatus helps to restore mobility and independence in managing daily activities. But as the technology in this field stands today, amputations are not easy to overcome due to drastic changes in capability and reduction in independence. This is primarily due to the kinematics of human limbs being challenging to analyse and replicate with an accuracy that would satisfy an individual looking for a replacement - at least when there is a cost cap involved.

When elucidating this issue while considering the plight of a Transradial Amputee, tasks classified under ‘Activities of Daily Living’ or ADLs involving putting on clothes, grasping/interacting with objects of varying rigidity, roughness such as a bottle of water or silverware all demand a high level of dexterity and flexibility in terms of strength.

Currently, four broad categories of prosthetic arms are used to restore certain degrees of capability to the user: body-powered, extrinsically powered, passive and a combination of the former two types. While they all have their advantages and disadvantages, a comprehensive review of all technology and designs that come under these categories shows that the qualities of a body-powered and an extrinsically powered prosthetic arm, when combined, serve to mitigate the dearth of control most effectively.

In order to most effectively serve as a viable arm replacement, the prosthetic must mimic a human arm when it comes to performing common tasks or ADLs. As discussed earlier, this is a computationally heavy task and requires predisposition on the part of the powered prosthetic to function accurately in tandem with the rest of the user's will. This feature involving the prediction of the user's 'intent' is most commonly referred to as intent recognition and can be seen as the cornerstone in the development of intelligent prosthetic devices and has several unique approaches to it.

1.2 Motivation

From all recent literature pertaining to the topic of intent recognition in prosthetic arms, one can surmise that a combination of inputs involving electromyography (EMG) signals, kinetic and kinematics sensors (such as load cells and inertial measurement units or IMUs), photographic inputs and data collected by other, more intricate means are measured by the powered prosthetic arm and run through highly computational deep learning models such as recursive neural networks or deep reinforcement learning models.

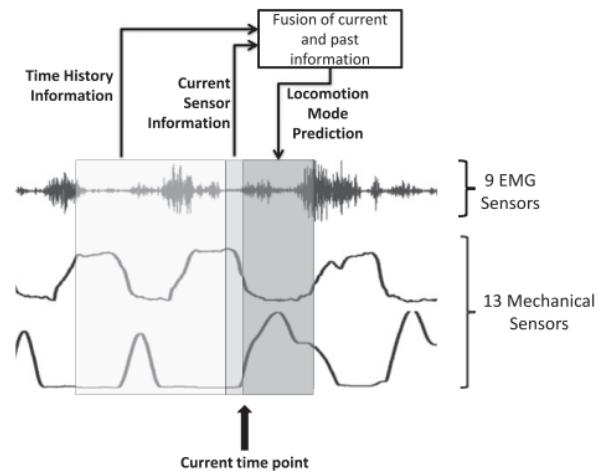


FIGURE 1.1: Various kinds of Sensor Data

While accurate, a significant drawback of these models is the post-processing time. For a powered prosthetic to integrate seamlessly with the user's intent, the decision must be computed sufficiently well in advance, but due to an abundance of input from the multitude of sensors present on the device, the response is noticeably delayed, making predictive models more of a hindrance to integration. While papers mention selection algorithms (such as SFS and mRMR) and optimised datasets with the help of self-organising maps (SOMs), there still exists a major room for improvement in these areas.

Another drawback of the aforementioned models is the increasing level of specificity required for them to perform even the most basic functions. While a number of characteristics of each prosthetic are unique to the user, the very framework of these algorithms must be modified



FIGURE 1.2: PURAK Prosthetic Arm

based on the unique datasets of each user in order to refit and retrain the device sufficiently. Both these reasons, along with the numerous sensors, materials and devices that support the methodology, contribute to a spike in the cost of the product, making it inaccessible to most of the commonwealth, especially in developing nations.

Therefore, the following study entails the design and control scheme for a sensor system in a proprioceptive prosthetic arm created at a lab in IISc Bangalore. The device, with fully articulated fingers, employs a single motor-driven under-actuated mechanism to adaptively grasp arbitrary shapes and initially made use of the residual limb for the wrist rotation using a myo-mechanical control interface. However, the mechanism for wrist rotation to orient the arm along the surface of an object preceding the grasping motion proved to be insufficient and has been completely reworked and revisualised following the design process discussed in detail below.

1.3 Background Study

1.3.1 Literature Review

A review of the literature was carried out and summarised to gain a more coherent understanding of the domain of intent recognition and of the existing research and the prevalent ideology. Findings from this review will be referenced throughout the report as various traits of the novel control scheme and its sensor system are developed. Some of the major takeaways from the literature review are given here.

The papers by Zhang et al and Young et al explain in detail the implementation of intent recognition in lower limb prostheses for transfemoral amputees. The the value of several data

sources for recognising user intent was examined and a beneficial collection of data sources for the volitional control of prosthetic legs was discovered. Then, using three source selection algorithms, data sources were ranked according to their utility for recognising user intent and it was observed that choosing a smaller set of data sources guaranteed accurate recognition of the user's intended job. Four patients with transfemoral amputations were analysed to corroborate results. The latter sourced these findings and made use of Dynamic Bayesian Networks (DBN) and Linear Discriminant Analysis (LDA) to predict the walking gait based on the user-specific training of the model.

The work of K.T. Luhandjula et al. works on intent recognition in a different manner, where the face in motion is represented using an intention curve rather than inferring intention as they proposed using the position of the face on a single frame. This work's major contribution is then categorised for intent recognition using a decision rule. Therefore, facial movements are regarded as an input to aid in locomotory tasks for patients suffering from a lack of lower body mobility.

On the other hand, literature by Z. Yu, M. Lee relied on more computational models to predict the intent of the user. In their paper, a dynamic classification model for intention recognition was proposed. It was based on a deep structure supervised MTRNN model. It was established that the deep supervised MTRNN model outperformed the single supervised MTRNN model in terms of efficiency and robustness. Their approach could finally identify an accurate intent by rejecting false intentions throughout the earlier stages, even though it failed to instantly distinguish human intention. In conclusion, the deep supervised MTRNN model presents a potential approach to identify human intention based on classification of human motion.

Olaya's work served to discern intent through a real-time myoelectric algorithm that improves the categorization of upper limb movement, taking into account changes in the activity level of each muscle for a specific motion between each person; this work advocated the use of wearable sensors to enhance detecting motion intention. A method for pattern recognition using IMU data and EMG has been created. The proposed myoelectric control method included both EMG data and kinematics information (angular velocity) of upper-limb joints, in contrast to conventional EMG-based pattern recognition techniques found in the literature, to compensate for EMG pattern changes caused by changes in upper-limb position.

1.3.2 Analysis of Human Movement

Although there have been tremendous technological advancements over the past forty years, the question of how to handle upper limb prosthetics naturally and without effort is still unresolved. Myoelectric prostheses that are sold commercially have a limited range of motion (DoF), mostly because there aren't any trustworthy independent control signals coming from the human body. Therefore, commercial hand prostheses only offer manually adjusted passive wrists or actuated rotators controlled by (unnatural) sequential control techniques, despite the vital function that an actuated wrist could play in a transradial prosthesis in terms of preventing compensatory movements. In order to further analyse consistent postural synergies present among all parts of the human arm for nearly all of the investigated everyday activities, we used OptiTrack software on a subject who performed various ADLs and analysed via its tracking coordinate storage system, the relationship between the elbow, shoulder and wrist along with wrist orientation this may help open the door to autonomously controlling the wrist rotator unit in transradial prosthesis, which would enhance the amputee's dexterity and fluidity.

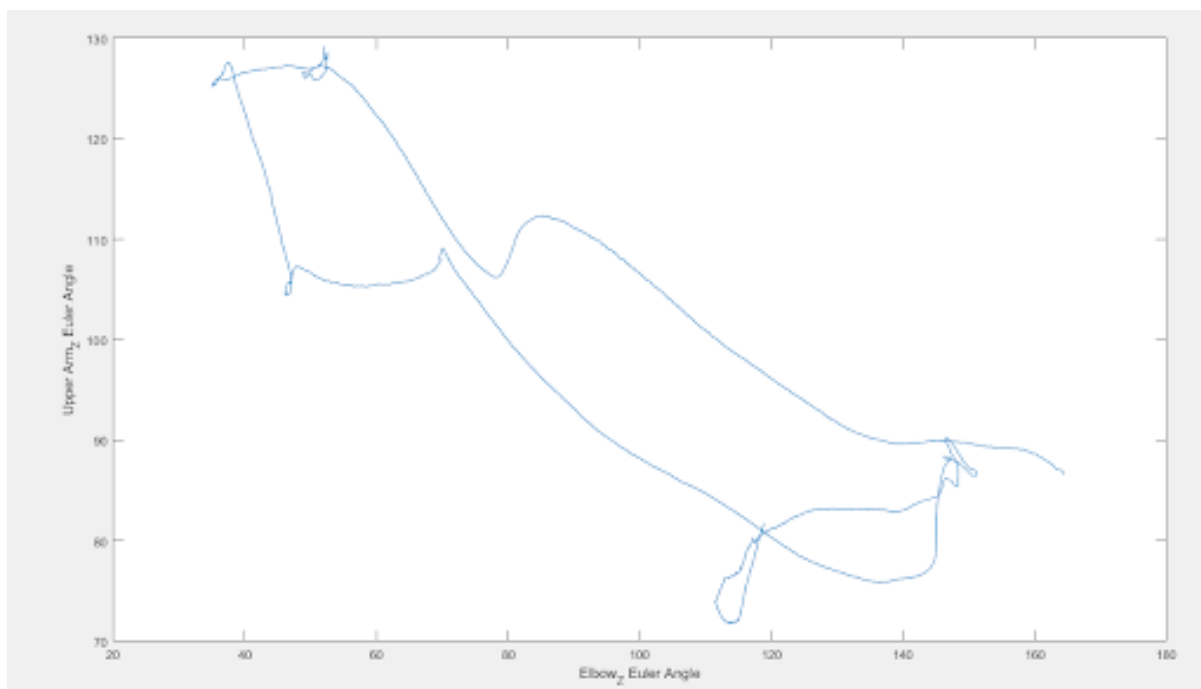


FIGURE 1.3: Quaternion Analysis of Tracker Data

1.3.3 Preexisting Set-Up

In order to facilitate wrist movements of pronation and supination based on user intent, the design makes use of two Force Sensitive Resistors (FSR) as sensors. These are placed at the base

of the digitus secundus and the digitus minimus on the prosthetic. This area would commonly be referred to as the head of the metacarpal in a human hand. The purpose of these sensors was to detect not only the presence but also the intensity of contact with a surface. This way, when the user would like to grasp an object, the device would discern the severity of their intentions simply if the object was to come in contact with the palm area.

The original test setup for the Force Sensitive Resistors (FSR) consisted of two FSRs mounted onto a vertical platform. Force would be applied onto the resistors via a flat plate (to ensure even distribution of force) connected to springs. The springs would, in turn, be connected to semi-spherical rigid structures resembling buttons which would indirectly impinge upon the FSRs by loading the springs. These were to be pressed via ACTUONIX L16 feed-forward linear actuators, which operated on a 12V PWM input.

The platform containing the FSRs could be made to rotate using an MG995 Servo motor about an axis parallel to the vertical platform and passing between both FSRs such that rotation would also change the degree of loading of springs on both FSRs.

The purpose of this setup was to design a system where the platform would rotate if force was applied onto the FSRs (by pressing a button) so as to equalise the force applied on both FSRs. This test setup helped simulate real-life conditions of sensor activation and was used to refine the circuits, control algorithm and the physical build of the proposed system. Fallacies were recognised in the setup, and a few components were amended and replaced over the course of the project. One major addition was that of the Arduino microcontroller for the purpose of data collection and computation in real-time.

An ATMEL AVR microcontroller serves as the foundation for the Arduino board. Microcontrollers are integrated circuits that you may programme using the programming language available in the Arduino IDE environment. These instructions can then be stored on the microcontrollers, enabling the user to write programmes that communicate with the circuits on the board using these commands. Furthermore, it has input/output and communication ports with which one may connect various kinds of peripherals on the board. The information



FIGURE 1.4: An Arduino Board

of these peripherals that can be connected will be transferred to the microcontroller, which will process the relevant data as per its programming.

Chapter 2

Characterization of FSR

2.1 The Force Sensitive Resistor

A conductive polymer makes up force-sensing resistors, which change resistance in a predictable way in response to the application of force to their surface. Typically, they are provided in the form of a polymer sheet or an ink that may be screen printed on. Particles suspended in a matrix, both electrically conducting and non-conducting, make up the sensing film. The sub-micrometer-sized particles are designed to boost surface durability, improve mechanical qualities, and lessen temperature dependence. Particles on the sensing film's surface that come into contact with the

conducting electrodes change the film's resistance when a force is applied to the surface as they vary the passage of electrons from one electrode to another.

Force-sensing resistors, like all resistive-based sensors, may function satisfactorily in somewhat hostile settings and call for a reasonably simple interface. The benefits of FSRs over other force sensors include their small size (usually less than 0.5 mm thick), low cost, and strong shock resistance. Their low precision is a drawback; measurement results can vary by 10 percent or

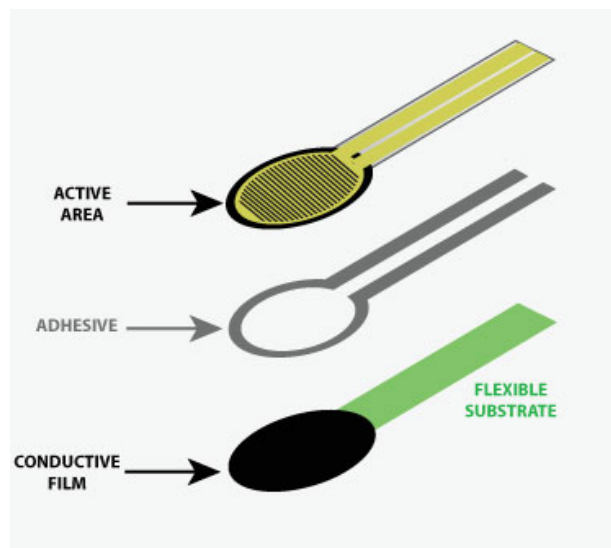


FIGURE 2.1: Components of an FSR

more. Force-sensing capacitors have better sensitivity and long-term stability, but their driving circuits are more intricate.

Percolation and quantum tunnelling are the two main operating concepts of force-sensing resistors. In the conductive polymer, both processes actually take place concurrently, but depending on particle concentration, one phenomenon predominates over the other. The filler volume fraction is another name for particle concentration in literature. New mechanistic explanations that are based on the property of contact resistance that occurs between the sensor electrodes and the conductive polymer have recently been developed to explain the performance of force-sensing resistors.

In two different ways, contact resistance is crucial to the current conduction of force-sensing

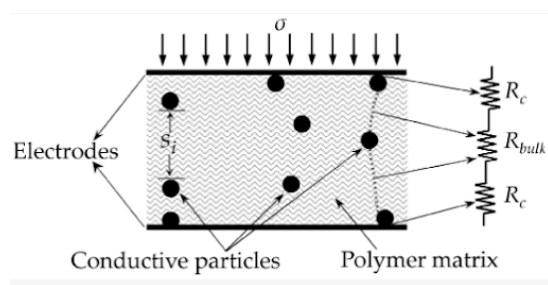


FIGURE 2.2: FSR Resistance

resistors. First, a plastic deformation between the sensor electrodes and the polymer particles takes place for a given applied stress or strain, lowering the contact resistance. Second, as the polymer surface is subjected to incremental stresses, it flattens down and creates more contact pathways, increasing the effective area for current conduction. The surface of the polymer is smooth at the macroscopic

level. The conductive polymer, however, appears erratic under a scanning electron microscope due to agglomerations of the polymeric binder.

2.2 FSR Circuit

2.2.1 Purpose

There is currently no complete model that can anticipate all of the non-linearities found in force-sensing resistors. The conductive polymer's numerous processes turn out to be too complex to be understood simultaneously; this is characteristic of systems covered by condensed matter physics. To either the percolation theory or the equations governing quantum tunnelling over a rectangular potential barrier, force-sensing resistors' experimental performance may be roughly compared in most circumstances. Therefore, it was essential to map the change in resistance of

each sensor with respect to the force applied. This was the purpose for the sensor characterization, and details as to how it was achieved can be found below.

2.2.2 Task

Since the force on the FSR was applied by a spring, a linear relationship between the force applied on the spring and its deflection - characterised by its loading and unloading was assumed, and an attempt was made to graph the relevant data. In order to achieve this, the FSR was incorporated into a Voltage Divider Circuit that made use of a known resistance value, a constant voltage source and an ARDUINO microcontroller programmed to continuously measure the voltage dissipated by the FSR via analog pins located on the circuit board.

The microcontroller was programmed using the Arduino IDE software, and voltage dissipated by the FSR was transmitted in real-time as serial data to the computer. This was then used to compute the resistance of the FSR. Manually applying force to the FSR resulted in visible resistance fluctuations, but the data was far from accurate. Therefore, the next task entails attempts to alleviate this problem.

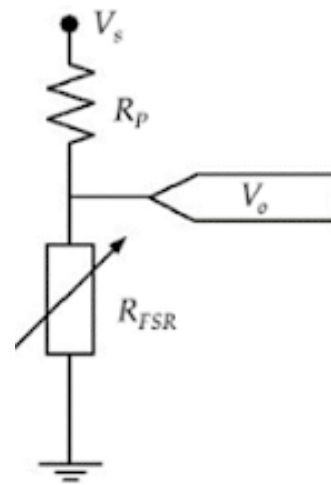


FIGURE 2.3: Voltage Divider Circuit

2.3 FSR Buffered Circuit

2.3.1 Purpose

The cause for inaccurate signals was identified as noise or minor aberrations which were picked up by the microcontroller while measuring the voltage dissipated by the FSR. Since the Voltage Divider Circuit was powered by the Arduino - which can only facilitate up to 5 volts of electricity - the magnitude of change in voltage measured was so minute that even comparatively minor disturbances caused significant fluctuation.

2.3.2 Task

An Operational Amplifier (Op-Amp) was used to amplify the voltage change and reduce the effect of noise. Moreover, the op-amp was incorporated as part of a closed-loop circuit so as to buffer the signal and curb fluctuations further. The new circuit was designed so as to maintain the same formula for exacting the FSR resistance from voltage dissipated for computational convenience.

In essence, an operational amplifier is a three-terminal apparatus with two high-impedance inputs. The Inverting Input is one of the inputs, and it is identified by a "minus" or negative sign ($-$). The second input is referred to as the Non-inverting Input and is denoted by a plus sign ($+$).

The operational amplifier's output port is represented by a third terminal, which can source or sink a voltage or current. In a linear operational amplifier, the output signal is the amplification factor, or amplifier gain (A), multiplied by the magnitude of the input signal. There are four main types of operational amplifier gain, depending on the nature of the input and output signals. A device with certain unique properties, such as infinite open-loop gain, infinite input resistance, zero output resistance, infinite bandwidth and zero offsets, is considered to be an "ideal" or perfect operational amplifier (the output is exactly zero when the input is zero).

The device used in this case is an LM324 integrated circuit which functions as a high gain voltage amplifier or a comparator and consists of four op-amps which can be operated independently while sharing a power supply. This is useful as the sensor system requires output from two FSRs, each of which requires its own voltage divider circuit. In the LM324, the output turns to zero, indicating that no current is flowing when power is provided to the non-inverting terminal that is less than the op-inverting amp's voltage. Because the non-inverting terminal is greater than the inverting terminal is a known condition. Here, the "+" symbol denotes a non-inverting terminal, while the "-" sign denotes an inverting terminal.

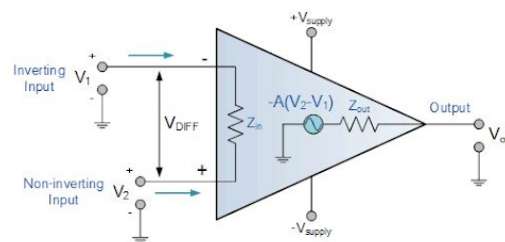


FIGURE 2.4: An Operational Amplifier

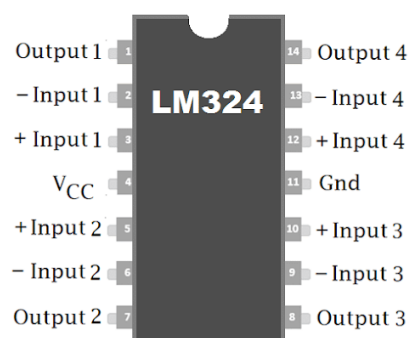


FIGURE 2.5: The Op-Amp IC

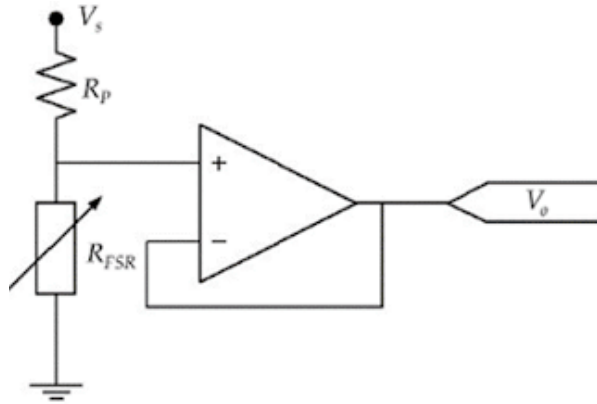


FIGURE 2.6: New Voltage Divider Circuit

Using these properties, the voltage divider system connects the output terminal and inverting input together to form a feedback loop. When the voltage dissipated by the FSR changes, the voltage at the non-inverting input will also change, allowing the op-amp output to vary accordingly until the voltage at the inverting input catches up to the change.

2.4 Feedback-based Linear Actuator

2.4.1 Purpose

While the modifications to the circuit proved to be much more effective in preventing interference of unwanted signals/disturbances. Up till this point, the FSR was being manipulated purely via manual means. There was no controlled input which could be analysed along with the variation in resistance being caused. The aforementioned linear actuators that were part of the previous set-up were both feedforward and relied on open loop control. This made them imprecise and unable to be measured. The task at hand serves to assuage this requirement.

2.4.2 Task

For a linear actuator to produce small, precise movements that were based on a closed-loop control, a rack-and-pinion-based linear actuator design was created. This design was operated with the help of an SG90 Tower Pro miniature servo motor connected to an independent power source and controlled using the microcontroller as a feedback device.

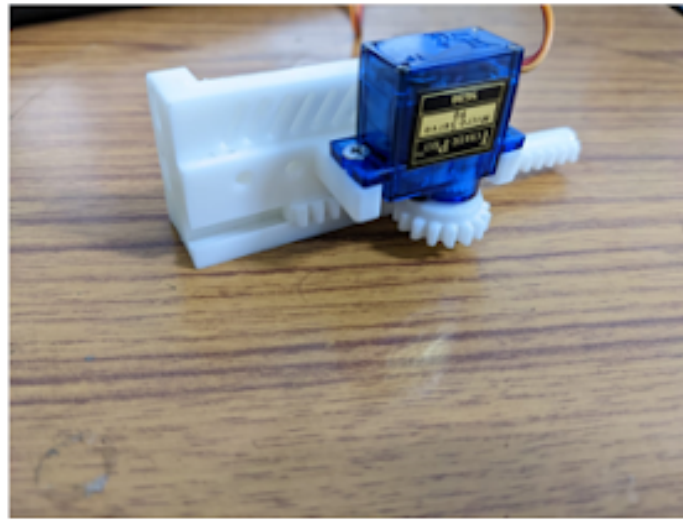
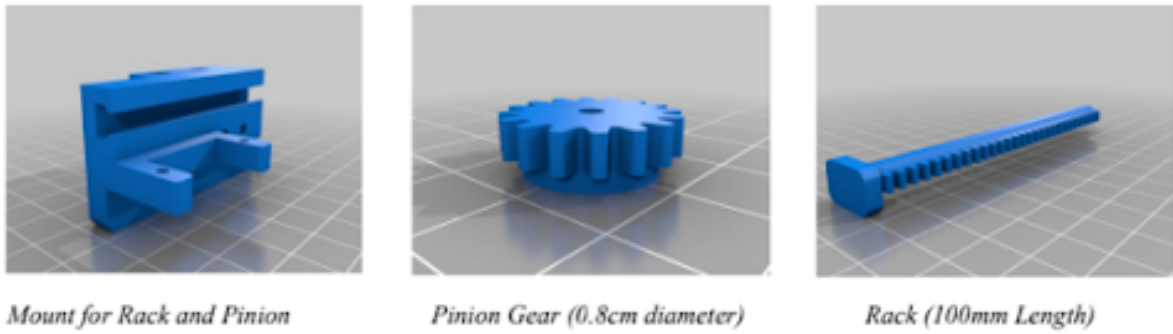


FIGURE 2.7: Servo Driven Linear Actuator

A servo motor generates torque and velocity dependent on the current and voltage input. A servo motor operates as a component of a closed-loop system, delivering torque and velocity in response to commands from a servo controller. The loop is closed by a feedback device. The feedback device gives current, velocity, or position information to the servo controller, which modifies the motor's behaviour in accordance with the requested parameters.

The rest of the linear actuator consisted of three major components - the mount for the motor, the rack and the pinion. All of these were modelled using SolidWorks, and concepts of generative design were used to remove excess material before 3D printing the parts. The pinion gear had a diameter of 0.8cm, and the rack was 7.5cm in length. These measurements made it so that a one-degree increment (minimum servo increment) caused an actuation of 0.12mm. This value was sufficiently small to ensure multiple iterations in the loading and unloading phases of the spring and was large enough so that each iteration could be observed.

2.5 Cycle-Time Analysis of Resistance

2.5.1 Purpose

Since the microcontroller being used was equipped with an internal clock feature, this segment of the experiment observes the use of this function, coupled with the timestep based loop function which is the basis of all microcontroller code. After creating a feedback based actuator, an autonomous algorithm was needed to treat time as an independent variable while controlling the servo motor driving the actuator and measuring the voltage signals relevant to the analysis.

2.5.2 Task

Since the microcontroller being used was equipped with an internal clock feature, this segment of the experiment observes the use of this function, coupled with the timestep-based loop function which is the basis of all microcontroller code. After creating a feedback-based actuator, an autonomous algorithm was needed to treat time as an independent variable while controlling the servo motor driving the actuator and measuring the voltage signals relevant to the analysis. The microcontroller was programmed to increment the servo motor driving the linear actuator

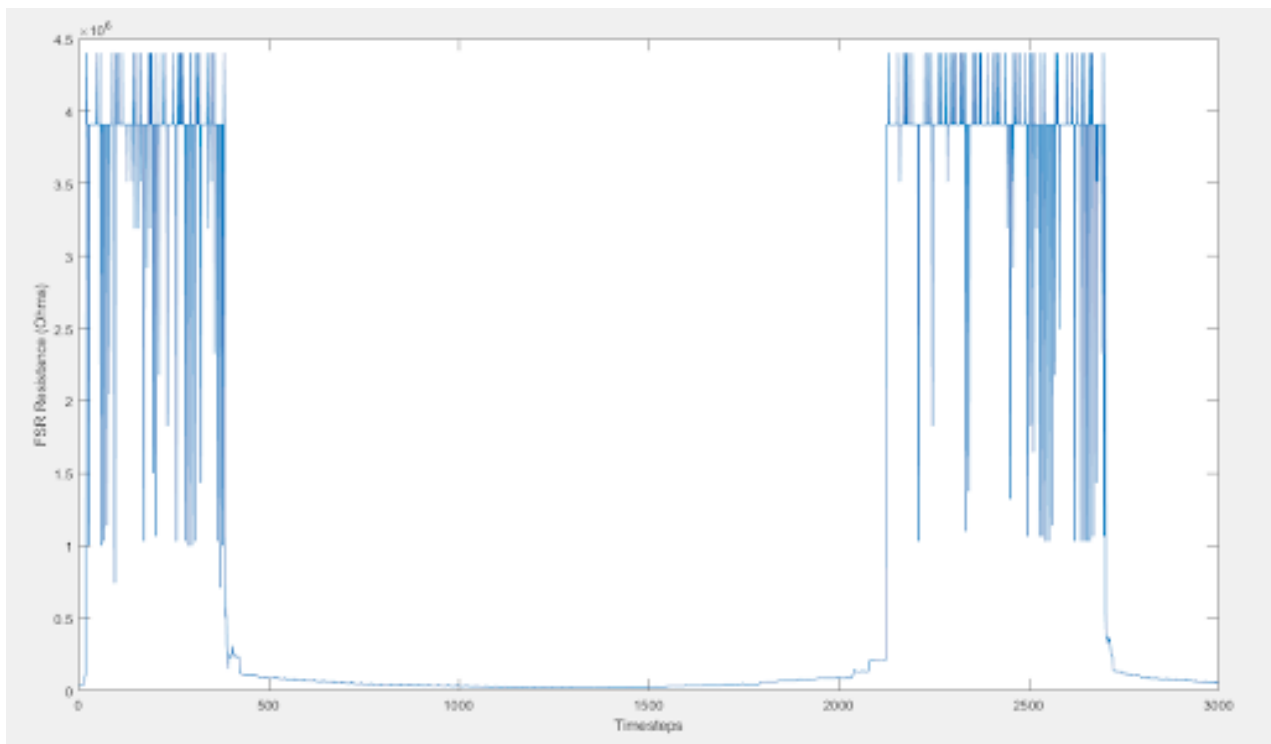


FIGURE 2.8: Resistance of FSR through a complete Loading-Unloading cycle

at regular time intervals. These iterations of periodic increase and incremental loading of the spring connected to the FSR would go on until the resistance of the FSR would hit a plateau, beyond which the servo motor would start retracting the linear actuator at similar intervals until the spring was fully unloaded. A figure comprising the entire loading/unloading cycle of the FSR would include the highest and lowest resistances that the FSR was capable of having. This data, while still somewhat noisy during the idle stages of the FSR, was important while adopting a scale to characterise FSR resistance with respect to the application of force. The relevant code can be found at [A.1](#).

2.6 Sampling Window

2.6.1 Purpose

In order to further smoothen the signal and increase its accuracy, a post-processing technique was also applied to the signal. Since the computation was to be done in the microcontroller itself, care was taken to maximise its efficiency while keeping the computation time to a minimum. This code was originally written and tested in MATLAB but then adapted to the Arduino IDE to ensure that MATLAB was only used for analysis purposes and not any real-time signal processing /generation.

2.6.2 Task

A window function is a mathematical function that is zero-valued outside of a selected interval, typically symmetric at the middle of the interval, typically around a maximum in the middle, and typically tapering away from the middle. It is used in signal processing and statistics. Mathematically, when a window function "multiplies" another function, waveform, or data sequence, the result is similarly zero-valued outside the interval; all that is left is the overlapped region, or "view through the window." Alternatively, and in reality, the window's data segment is isolated first, and then just that data is multiplied by the values of the window function. Thus, the primary goal of window functions is tapering rather than segmentation.

Determining transient occurrences and averaging frequency spectra over time are two reasons to look at portions of a longer function. Each application's requirements, such as those for time and frequency resolution, dictate the length of the segments.

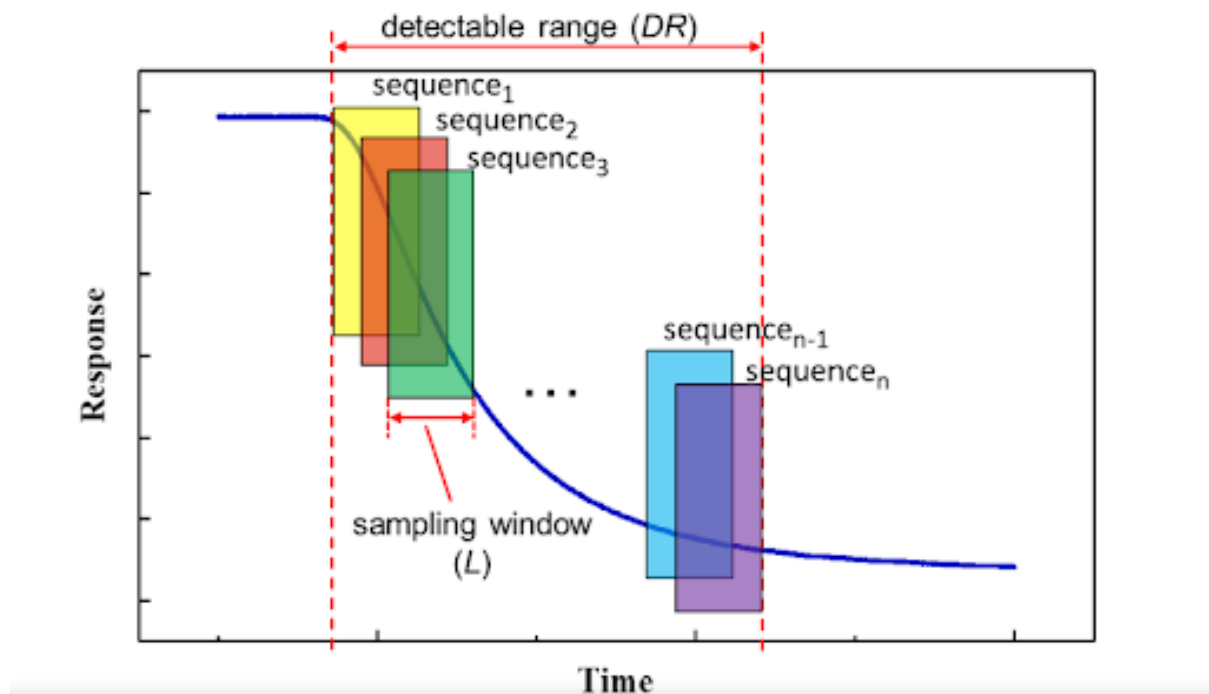


FIGURE 2.9: Depiction of a Sampling Window

A similar strategy was adopted for cleaning the signal of jitters wherein a sample time(segment) of 100 milliseconds was identified and all signals produced in this timespan were continuously stored and discarded as they dropped out of the sample time. These stored signals were then computed as part of a moving average to produce their arithmetic mean value. This value was then treated as the output for the purpose of further computation. The relevant code can be found at [A.2](#).

2.7 Results

Since the serial data was only being displayed in real-time on the current software, instead of being displayed, the FSR resistance values and the servo control signal data were routed via a MATLAB software and stored in dynamically allocated memory for analysis. The program was equipped with conditional statements, which ensured sufficient data was accumulated (data points pertaining to at least one complete loading and unloading cycle) before terminating the collection process. Throughout this process, the signal processing technique discussed earlier was being applied in real-time, leading to the collection of relatively less noisy data.

Once the algorithm had run its course and collection was complete, data pertaining to the control signal of the servo motor was used for scaling the linear actuator, which led to the

derivation of the degree of spring loading and unloading. Since this value was collected in real-time, the timestamp of data collection of the servo control signals to the FSR resistance values was compared, and a relation was graphed on a semilog scale owing to an exponential drop in resistance values upon FSR activation. This made it easier to observe changes in the lower range of values which was the primary area of interest. Moreover, multiple loading/unloading cycles were run, and the resistance of the FSR for each cycle was plotted simultaneously as shown in the figure below. This was done to examine the effects of creep, sensor reliability, hysteresis, and the upper and lower limits of resistance of the FSR. The MATLAB code can be found here [B.1](#).

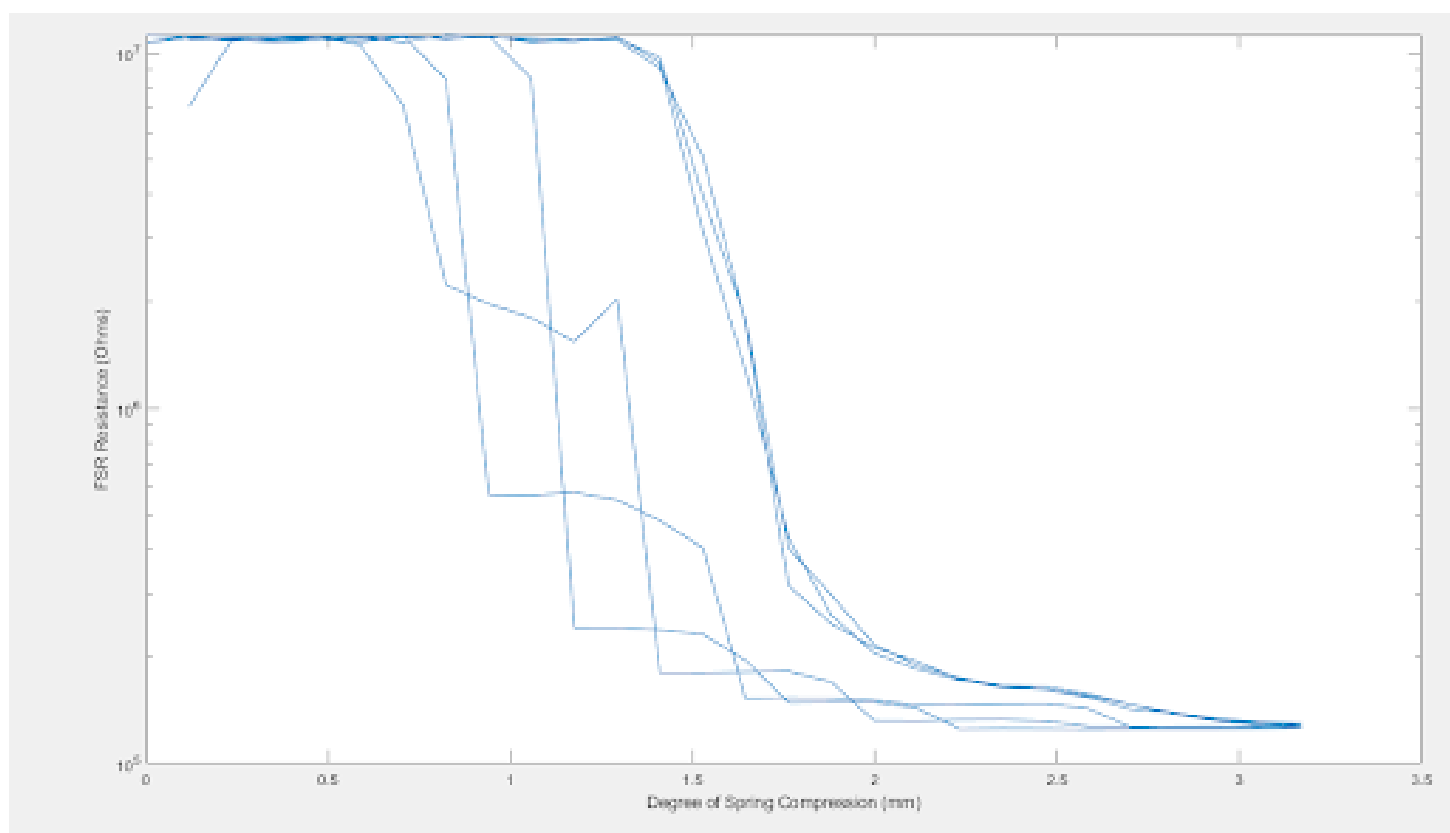


FIGURE 2.10: Hysteresis Graph for FSR

Chapter 3

Control Scheme

So far, the force sensitive resistor based sensors have been effectively characterised, and accurate relations have been derived to map to and predict their change in resistance. This chapter focuses on using this derived data as input and modelling a system that dynamically responds to it. It is owing to these capabilities that the powered prosthetic device can be called smart in terms of discerning intent, thus forming the basis for intent recognition. This part of the project observes a relatively similar set-up of the FSR and the voltage divider circuit. Since the FSR response has already been recorded, the feedback-driven linear actuator has been replaced by a standalone servo motor which acts as an output display for the response of the sensor system when the FSRs are activated. A later part of this section also includes implementation of the developed algorithm on the prosthetic arm. The arm is driven by a 12 DC motor connected to a customised gearbox to maximise torque for facilitating wrist rotation. The motor is driven by an L298N dual H-bridge motor driver module. Details about which have been discussed later in the document.

3.1 Dynamic Servo Response

3.1.1 Purpose

While the user of the prosthetic may be familiar with the pose of the object that they would like to interact with. This model of the prosthetic arm is not equipped with conventional sensors or neural network-based pose estimation methods. It relies purely on the degree of sensor activation

and user perception to predict intent. This implies that when the user is in the process of attempting to grasp an object, the speed and intensity of their action will dictate the severity of their intent. These attributes can be measured using the proposed sensors and should be reflected in the response from the system. The following task aims to quantize said response based on input from the sensors.

3.1.2 Task

An SG90 Tower Pro servo motor was connected to a 5V power source and controlled via analog output from the microcontroller. The servo motor shaft was fitted with a dial, and the resting position was set to 90 degrees of servo rotation with 0 and 180(or, in the case of this servo, 185), indicating maximum activation of the FSR on their respective end.

Due to the exponential change in FSR resistance upon activation, linearly translating the change in FSR resistance to servo position would result in a huge jump in the magnitude of the response and the later stages - denoting the degree of activation - would have little to no observable response. When the mapped servo response demonstrated earlier was linearly converting to servo control signals, here is what the resultant response curve looked like.

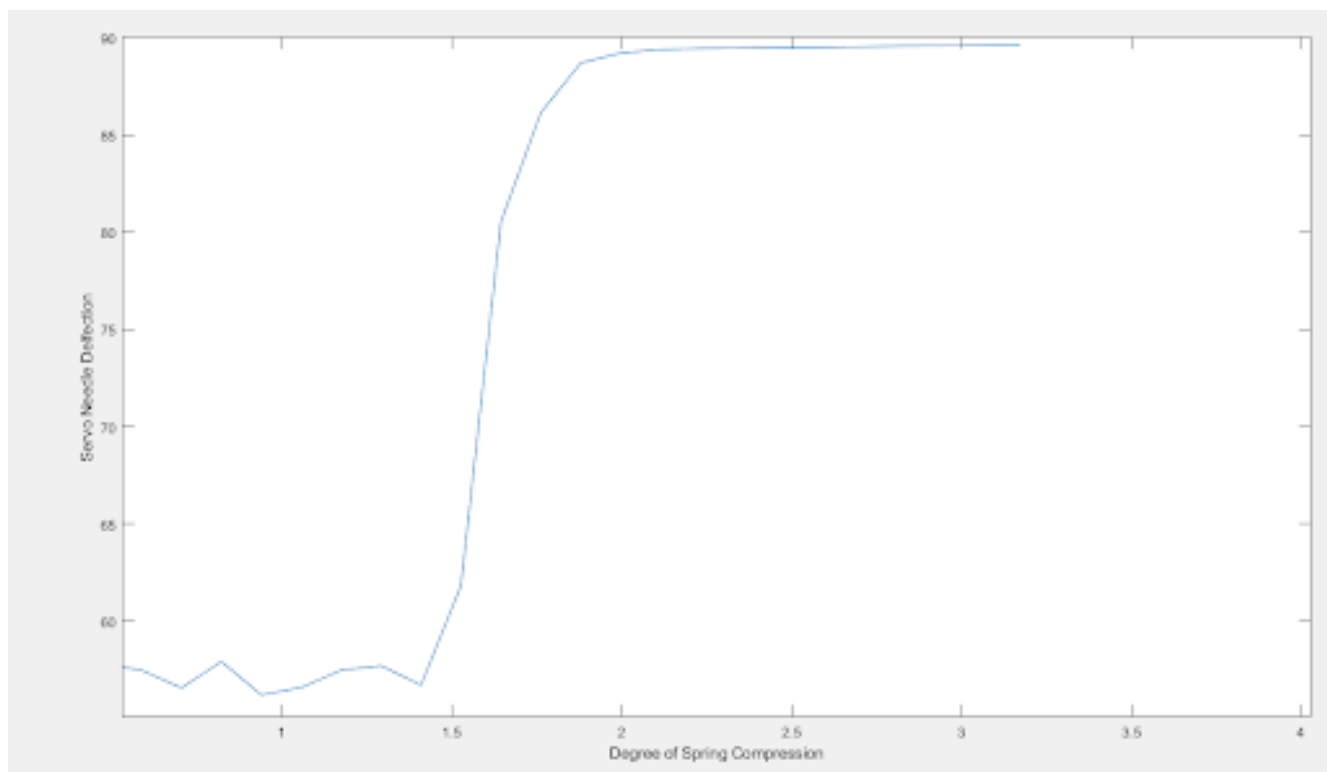


FIGURE 3.1: Linearly Transferred Servo Response

This response essentially acts as a toggle switch between active and inactive FSR states and would not only be infeasible and unsafe to implement in a powered prosthetic, but it also defeats the purpose of predicting and discerning intent using such an elaborate sensor system. Therefore, to ameliorate this, the microcontroller code was amended to incorporate a calibration function wherein the user will be required to fully activate both FSRs by completely pressing the buttons on either side for a brief period of time.

The highest and lowest thresholds of both resistors would be stored from here on and would constantly be updated in the off-chance that the calibration function was not carried out properly. The lowest stored value would then be multiplied to an inverse of the sample value to create a curve where the responsiveness would increase with activation. This would compensate for the decreasing magnitude of change as loading progresses. This is what the new transfer curve resembles after the amendment. The code pertaining to generation and analysis can be viewed [here](#) [B.2](#).

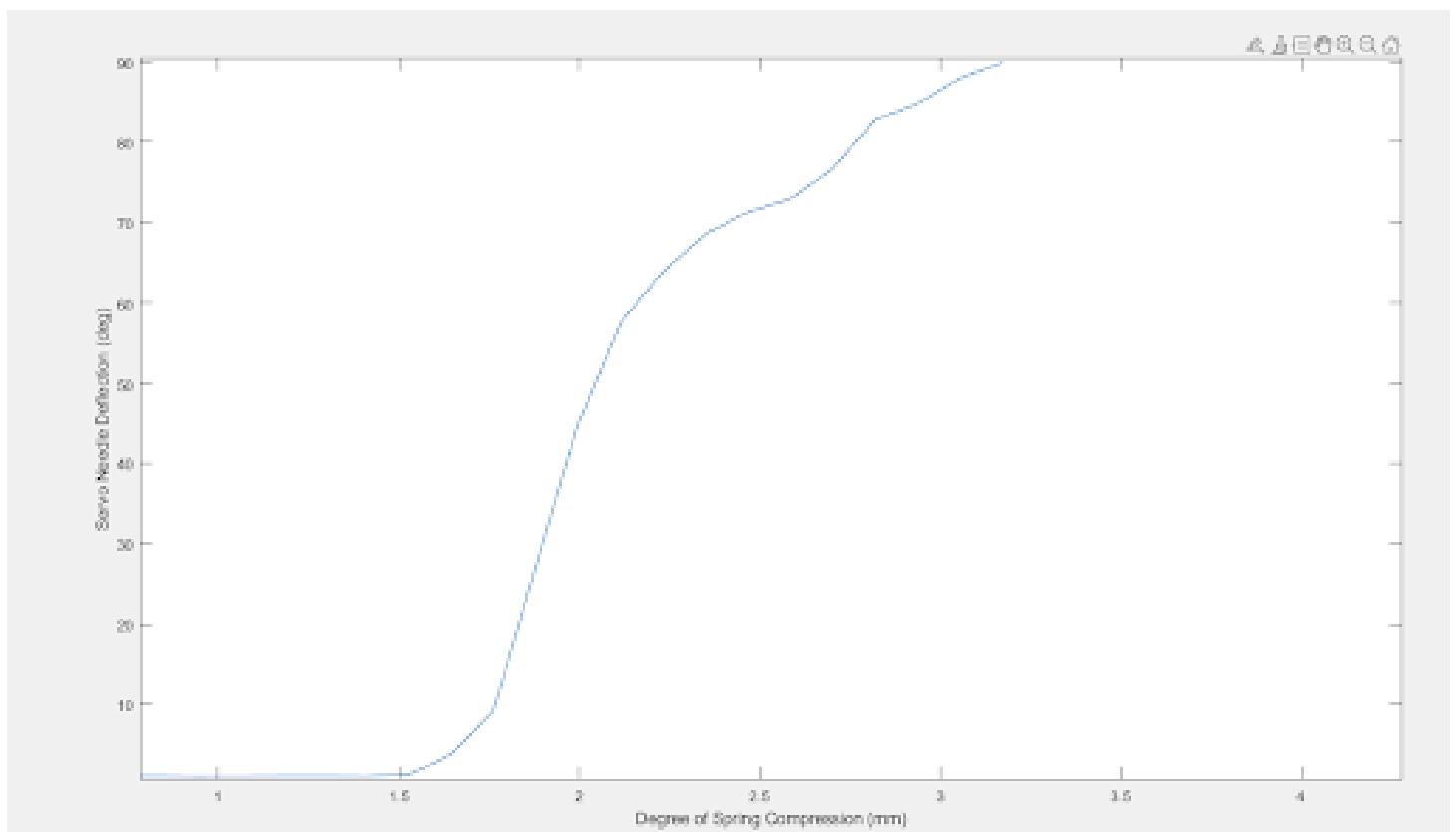


FIGURE 3.2: Amended Servo Response

3.2 Implementation

The prosthetic arm was fitted with a 3D printed mount for the motor along with ball bearings to facilitate smooth rotation. The intended motion was carried out via a 12V DC Motor fitted with a gearbox which was also specifically manufactured for increasing the torque to sufficiently support the load of the entire device. The motor was driven using an independent power source routed through an L298N motor controller.



FIGURE 3.3: Motor and Motor Mount

The L298N is a twin full-bridge motor driver integrated circuit (IC) capable of handling relatively higher levels of voltage and current. It manages inductive loads like relays, solenoids, DC motors, and stepper motors and accepts common TTL logic levels (Control Logic). The integrated circuit has 15 pins. The operational voltage range for the L298 is +5 to +46V, and the maximum current that may pass through each output is 3A. Two enable inputs on this IC are available for independently enabling and disabling the device regardless of input signals.

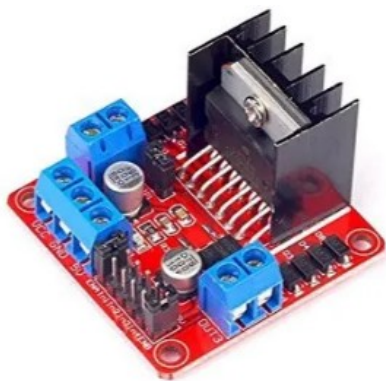


FIGURE 3.4: L298 Motor Controller

The module's IC is connected to a black heat sink. The heat produced by an electronic or mechanical device is transferred to a fluid medium, frequently air or a liquid coolant, by a passive heat exchanger commonly referred to as the heat sink.

The H bridge in the motor controller helps to reverse the direction of motor movement and switch between supination and pronation of the arm. The controller also requires a control signal from the microcontroller, which helps it to vary the speed of

rotation by changing the voltage supplied to the DC motor. Utilising this, the servo response algorithm is adapted to vary voltage via analog output from the microcontroller, causing a change in the speed of motor rotation. This quality aims to imitate natural human response

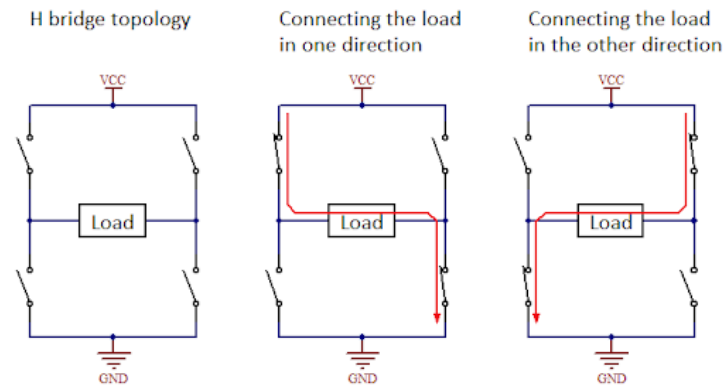


FIGURE 3.5: Depiction of an H Bridge

by dynamically changing the rate of execution of its function based on the urgency that its algorithm is cognizant of.

Chapter 4

Future Work

The system is still far from perfect, however, and one major research gap that has been identified with its ability to integrate with a biological system is that unintentional, undesirable and unfavourable actions may be committed due to either unfamiliarity or inexperience on behalf of the user. The following task aims to minimise such fastidious occurrences by introducing another parameter to the control scheme. [A.3](#)

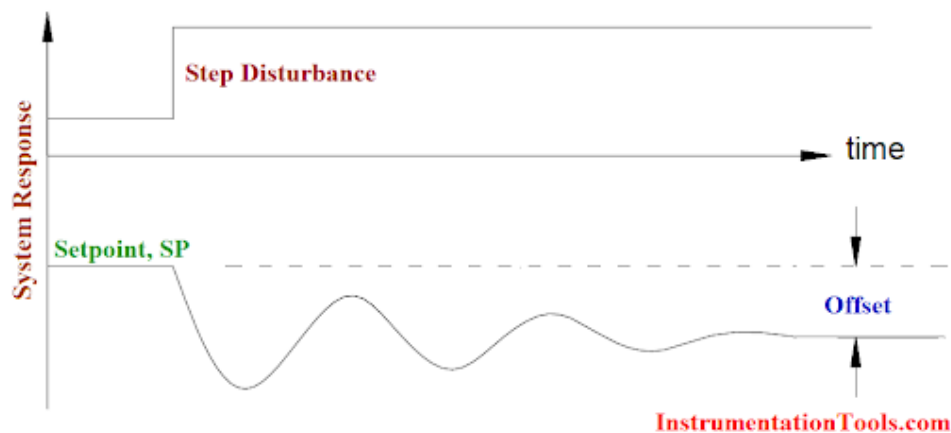


FIGURE 4.1: Selective Delay Control Scheme

Future work may recognize the optimisation of an additional function that was added to the code, which helped prevent aberrant responses from the system when faced with jittery input. While human error may be the primary cause for such a response, it may also be caused due to accidents or malfunctions, and an externally powered response in such conditions would only add to the risk of injury. Therefore this feature also functions as a fail-safe for such scenarios.

The function computes and stores values of previous samples and relates the magnitude of the current sample with previous values, negating steady state error due to hysteresis, creep or another factor that rheologically affects the FSR and also delaying responses to sudden changes just enough to make sure it was caused intentionally beyond which, the system performs with similar efficacy as earlier.

The figure given below indicates, in part, how jittery signals can be nullified and highlights

the relation between jitter and latency, which may be further tuned via practical implementation.

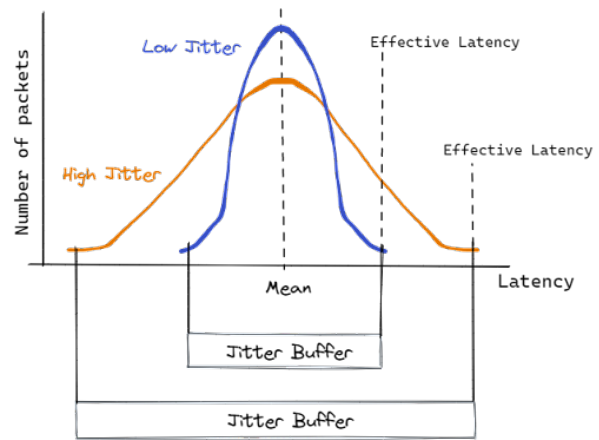


FIGURE 4.2: Depiction of the relationship between Jitter and Latency

Chapter 5

Conclusion

The research has produced documentation of the production and installation of a novel control system for intent recognition, as well as a comprehensive analysis and rationale for each component that has been utilised in the system. Techniques of sensor characterisation, 3-dimensional modelling, data processing and development of electronic circuits are among some that have been used to accomplish this task. This system provides the user of a prosthetic arm with assistance in completing tasks that involve grasping or interacting with an object and increases overall dexterity and perceived familiarity with the object without relying on a large number of sensors or other electronic equipment, which is largely in contrast to other prosthetics. A detailed analysis of the prevalent technology in this field has revealed that most have chosen to leave behind feasibility in pursuit of perfection without considering the idea of achieving both simultaneously.

The body of research then goes on to elaborate on the overarching idea that underpins the implementation of such methods, which is accompanied by the development of a computationally efficient strategy that is based on a novel concept that reduces the risk of suffering any consequences due to an error made on the part of the user. In conclusion, despite the significant advancements in prosthetic technology over the years, this research highlights the need for hand control to be improved by using a limited resource base to increase feasibility in order to ensure ease in performing everyday chores, all the while restoring the amputee's sense of engagement with the world and provides a unique, practical approach to do so.

Appendix A

Microcontroller Code

A.1 FSR Cycle Time

```
#include<Servo.h>

unsigned long  t = 0,temp,f;
int var = 1;
unsigned long timeLag = 4000000;
Servo s;
float val = 0;
float big;
float V,FSR1,FSR2,FSR3;
int upperLimit = 27;
int lowerLimit = 0;
void setup() {
    //pinMode(11,OUTPUT);
    pinMode(A0,INPUT);
    s.attach(11);
    Serial.begin(9600);
}

void loop() {
    //analogWrite(11,1023);
    big = FSR1;
```

```
//V = analogRead(A0);
V = analogRead(A0)*5/1023.;
FSR1 = (3.78/V-1)*46000.;
//FSR1 = 15000./(3.3/V-1);
if(abs(FSR2-FSR1)<big)
{
    FSR2 = FSR1;
}
Serial.println(FSR1);
s.write(val);
temp = micros();
f = temp - t;
switch (var)
{

case 1:
if(val < upperLimit && f>timeLag)
{
    val = val +1;
    t = temp;
    if(val == upperLimit)
    {
        var = 2;
    }
}
break;

case 2:
    if(val > lowerLimit && f > timeLag)
{
    val = val - 1;
    t = temp;
    if(val == lowerLimit)
    {
```

```
        var = 1;
    }
}
    break;
}
}
// if(Serial.parseInt()==1)
//{
//  val = val + 5;
//  Serial.print("Angle is ");
//  Serial.println(val);
//}
```

A.2 FSR Sampling Window

```
#include<Servo.h>
unsigned long t = 0,temp,f;
int var = 1;
float data[10];
int count = 0;
float avg;
unsigned long timeLag = 800000;
Servo s;
float val = 1;
float big;
float V,FSR1,FSR2,FSR3;
int upperLimit = 27;
int lowerLimit = 0;
void setup() {
    //pinMode(11,OUTPUT);
    pinMode(A0,INPUT);
    s.attach(11);
    Serial.begin(9600);
```

```
}

void loop() {
    //analogWrite(11,1023);
    //big = FSR1/2;
    //V = analogRead(A0);
    V = analogRead(A0)*5/1023.;
    FSR1 = (3.78/V-1)*46000.;
    if(FSR1>10000000)
    {
        FSR1 = 10000000;
    }
    data[count] = FSR1;
    count = count +1;
    if(count>9)
    {
        count = 0;
    }
    avg = (data[0]+data[1]+data[2]+data[3]+data[4]+data[5]+data[6]+data[7]+data[8]+data[9])/10;

    //if(abs(FSR2-FSR1)<big)
    // {
    //   FSR2 = FSR1;
    //}

    Serial.print(avg);
    Serial.print(",");
    Serial.println(val);
    s.write(val);
    temp = micros();
    f = temp - t;
    switch (var)
    {
```

```
case 1:
if(val < upperLimit && f>timeLag)
{
    val = val +1;
    t = temp;
    if(val == upperLimit)
    {
        var = 2;
//    delay(1000);
    }
}
break;

case 2:
    if(val > lowerLimit && f > timeLag)
{
    val = val - 1;
    t = temp;
    if(val == lowerLimit)
    {
        var = 1;
    }
}
break;
}
```

A.3 Code to potentially reduce fluctuations

```
double computeFluc(double inp){
    currentTime = millis();
    elapsedTime = (double)(currentTime - previousTime);
    error = setPoint - inp;
```

```
    cumError += error * elapsedTime;
    potError = (error - prevError)/elapsedTime;

    double out = kp*error + ki*cumError + kd*prevError ;           // output

    lastError = error;
    previousTime = currentTime;

    return out;
}
```


Appendix B

MATLAB Code

B.1 FSR Data Storage and Analysis

```
clc;
delete(instrfind);
s1 = serial('COM5','BaudRate',9600,'DataBits',8);
b = [];
a = [];
d = [];
sum = 0;
counter = 0;
overall_count = 0;
avg = [];
temp = 1;
fopen(s1);
for i=1:1:50000
    s = fscanf(s1);
    c = strsplit(s,',');
    %%d = cell2mat(c);
    a = [a,str2num(cell2mat(c(2)))];
    b = [b,str2num(cell2mat(c(1)))];
    if i > 1 && a(i) == a(i-1)
        if str2num(cell2mat(c(1)))
```

```
        sum = sum + b(i-1);
        counter = counter +1;
        avg(temp) = sum/counter;
    end
end

if i>1 && a(i)~=a(i-1)
    d = [d,a(i-1)*0.1175];

    if a(i) == 0
        overall_count = overall_count+1;
    end

    if overall_count == 3
        break;
    end

    sum = 0;
    temp = temp +1;
    counter = 0;
    avg = [avg,0];
end
end
semilogy(d,avg);
fclose(s1);
```

B.2 Servo Response Code

```
h = openfig('hysterisis_moving_avg_3_plots');
h = findobj(gca,'Type','line');
x = get(h,'Xdata');
```

```
y = get(h,'Ydata');
y = [0,y];
x = [x(1),x];
avg = zeros(1,28);
dist = x(1:28);
new = [];
for i=1:length(y)
    b = rem(i,54);

    if(b>27)
        break
    end
    b = b+1;
    avg(b) = avg(b) + (y(i))/3;
end
low = avg(length(avg));
for i = 1:length(avg)
    %new =[new,((low/avg(i))^1)*90];
    new =[new,(1-avg(i)/10000000)*90];
end
    plot(dist,new);
```