

# **Haptic robotic device for tele-surgery**

## **Senior design project report**

**BY**

**Maha Arshad**

**Muhammad Afaq**

**Muhammad Fahad Saeed**

**Taha Mahmood**

**Supervised by**

**Sir Abid Imran**



**Faculty of Mechanical Engineering**

**GIK Institute of Engineering Sciences & Technology**

**May 2021**

# **Haptic robotic device for tele-surgery**

## **Senior design project report**

**BY**

<b>Maha Arshad</b>	<b>2017187</b>
<b>Muhammad Afaq</b>	<b>2017229</b>
<b>Muhammad Fahad Saeed</b>	<b>2017256</b>
<b>Taha Mahmood</b>	<b>2017472</b>

**Supervised by**

**Sir Abid Imran**

submitted in partial fulfillment of the requirements for the degree  
of bachelor of science

**Faculty of Mechanical Engineering**

**GIK Institute of Engineering Sciences & Technology**

**May 2021**



# Faculty of Mechanical Engineering

GIK INSTITUTE OF ENGINEERING SCIENCES & TECHNOLOGY

iii

## Senior Design Project Status/Completion Certificate

**Group No: 25**

**Title: Haptic robotic device for tele-surgery.**

This is to certify that senior year design project has satisfied the following:

- (i) The design part of the project is completed to a sufficient level. Specifically, the project has achieved.

a.	
b.	
c.	

- (ii) The students have engaged in weekly meetings and demonstrated gradual progression of work, meeting the 6CH requirement.
- (iii) The defined Scope / Objectives are;

Sr.#	Objectives	KPI (min. 50%) achieved	
		Advisor	Ext. Examiner
a.			
b.			
c.			
d.			
e.	Quality of Report (Technical content, breadth/depth)		

- (iv) Based on the above score, the project stands \_\_\_\_\_. (complete/incomplete)
- (v) I (Advisor) understand that the report may be subjected to external review.

\_\_\_\_\_  
**Advisor**

\_\_\_\_\_  
**External Examiner**

(To be filled by FYP Coordinator)		Good	Average	Poor
(vi)	Overall structure of report is in-line with the provided FME guidelines.			
(vii)	Students followed the given deadlines by SDP committee			

\_\_\_\_\_  
**FYP Coordinator**

\_\_\_\_\_  
**Dean FME**



# Faculty of Mechanical Engineering

GIK INSTITUTE OF ENGINEERING SCIENCES & TECHNOLOGY

iv

## **FYP Mapping with Complex Engineering Problem Attributes**

**Group No: 25**

**Title:** Haptic robotic device for tele-surgery.

**Problem Statement:** In this project, the User will be able to control the robotic arm (slave) using a remotely operating controller (master) and adjusting the end-effector to a specific coordinates while experiencing a uni-directional Haptic feedback on the User.

### **CEP Attributes Mapping:**

(Sr.#1 is mandatory and at least one from the remaining 2~9).

S. No	Attribute	Justification
1	Preamble In-depth engineering knowledge	
2	Range of conflicting requirements	
3	Depth of analysis required	
4	Depth of knowledge required	
5	Familiarity of issues	
6	Extent of applicable codes	
7	Extent of stakeholder involvement and level of conflicting requirement	
8	Consequences	
9	Interdependence	

Advisor: \_\_\_\_\_

Signature: \_\_\_\_\_

## ABSTRACT

The ever-growing industry of virtual reality compels us to dive more and more into haptics. The need further arises during medical procedures where surgeon's sense of touch is eliminated during robotic procedures. The sense of touch is essential for humans to grasp complete view of the environment, therefore our aim is to manufacture a low-cost device providing haptic feedback, mainly for surgical procedures. The device would be used by the surgeon to control the robotic arm (PUMA 560), and allow him to precisely experience the forces acting on the robotic arm in the surgical environment. This report consists of a comprehensive kinematic analysis of the PUMA 560, along with its trajectory generation, furthered by the development of a Graphical User Interface for controlling the position and orientation of the end effector of the PUMA 560. A model for the one degree of freedom haptic device is proposed, and relevant calculations for its displacement and force are performed. By using a force-sensitive resistor on the end effector, forces on the end effector can be measured, which are expected to be transmitted precisely to haptic device being fabricated by controlling the motor current. The final aim of this project is to establish remote control, which would allow the surgeon to control the robotic arm at a remote location using the haptic device through a series of protocols. The need of haptic tele-surgery became apparent in the recent times, especially during the outbreak of COVID, when doctors could not operate patients without the risk of getting infected. Haptic feedback, if incorporated, would allow robotic surgical procedures to be performed with a higher precision than that possible through manual ones.

## Table of Contents

Abstract.....	v
Table of contents .....	vi
Nomenclature .....	viii
List of Figures .....	ix
List of Tables .....	xi
 Chapter 1	 1
1.1    Background and Motivation	1
1.2    Problem Statement	1
1.3    Scope of the Work	2
1.4    Expected Outcome	2
1.5    Report Outline	2
1.6    Project Schedule and Timeline	3
1.7    Individual and team contribution	4
Chapter 2	5
2.1    Literature Review	5
2.1.1    Control of the PUMA 560	5
2.1.2    Design of Haptic Master Controller	6
2.1.3    Remote Control	7
2.2    Inferences Drawn out of Literature.	7
2.2.1    Critical Analysis	7
2.2.2    Most Appropriate Approach	8
2.3    Summary	8
Chapter 3	9
3.1    Design Methodology	9
3.2    Mathematical Model:	10
3.2.1    Forward Kinematics:	10
3.2.2    Inverse Kinematics:	11
3.2.3    Trajectories:	16
3.2.4    Haptic kinematics and dynamics	17

3.3	Geometric Modeling and Design	21
3.4	Analysis	28
3.4.1	Haptic controller analysis	28
3.4.2	Environmental and Social Impact	30
3.5	Summary	30
Chapter 4		31
4.1	Development Process	31
4.2	Integration and Instrumentation	41
4.2.1	Transmission	41
4.2.2	Actuator	44
4.2.3	Sensors	46
4.3	Testing/Experimental Procedures	50
4.3.1	Signal Processing	50
4.4	Summary	51
Chapter 5 Results		52
5.1	Results	52
5.1.1	Robot Control	52
5.1.2	Haptic Controller Input	53
5.1.3	Haptic Feedback Mechanism	54
5.2	Analysis and Discussions	56
5.3	Summary	58
5.3.1	Complete Uni-directional Haptic Device Prototype	58
Chapter 6		60
Impact and Economic Analysis		60
6.1	Social Impact	60
6.2	Environmental impact	60
6.3	Sustainable development goals (SDGs)	61
6.4	Hazard Identification and Safety Measures	62
6.5	Summary	63
Chapter 7 Conclusion and Future Recommendation		64

7.1	Summary	64
7.2	Recommendations:	64
	References	66
	Appendices	68



## **Nomenclature**

PUMA	Programmable Universal Machine for Assembly
3D	Three Dimensional
GUI	Graphical User Interface
CAD	Computer Aided Design
DOF	Degrees of Freedom
IEEE	Institute of Electrical and Electronics Engineers
D-H	Denavit-Hartenberg
POE	Product of Exponentials
DC	Direct Current
LED	Light Emitting Diode
FSR	Force Sensing Resistor
IOT	Internet of Things

## List of Figures

**Fig 3-1:** Design Process Flow

**Fig 3-2:** Kinematic chain parameters for PUMA 560 [17]

**Fig 3-3:** Four solutions of the PUMA 560 [18]

**Fig 3-4:** PUMA 560 on CoppeliaSim

**Fig 3-5:** Plots of end effector's displacement 10 cm back and forth in positive x direction

**Fig 3-6:** Plots of a quintic time scaling [19]

**Fig 3-7:** Haptic Controller Diagram [20]

**Fig 3-8:** Free-body diagram [20]

**Fig 3-9:** Assembly of the Haptic Feedback Device

**Fig 3-10:** Paddle (all dimensions in mm)

**Fig 3-11:** Pulley (all dimensions in mm)

**Fig 3-12:** Front Plate (all dimensions in mm)

**Fig 3-13:** Baseplate (all dimensions in mm)

**Fig 3-14:** Support (all dimensions in mm)

**Fig 3-15:** Swing Bar (all dimensions in mm)

**Fig 4-1:** Orientation of PUMA 560 in "ready" position [21]

**Fig 4-2:** Tensor Algebra

**Fig 4-3:** 1080p USB Webcam

**Fig 4-4:** Remote Control Flowchart

**Fig 4-5:** ThingSpeak diagram

**Fig 4-6:** Circuit Diagram

**Fig 4-7:** Graphical User Interface controlling the position and orientation of end-effector

**Fig 4-8:** Graphical User Interface for each Motor Control

**Fig 4-9:** Capstan Drives and Belts [22]

**Fig 4-10:** Gear System [23]

**Fig 4-11:** Stepper Motor [24]

**Fig 4-12:** DC Motors [25]

**Fig 4-13:** Servo Motors [26]

**Fig 4-14:** OMRON Incremental Encoder

**Fig 4-15:** Magnetoresistive angle sensor [28]

**Fig 4-16:** Hall-Effect Sensor [29]

**Fig 4-17:** Potentiometers [30]

**Fig 4-18:** Force Sensing Resistor build and change in its properties due to force [31] [32]

**Fig 4-19:** Signal Flow for forward and feedback loop

**Fig 5-1:** Joint angles for PUMA 560 moving 20 cm in positive x direction.

**Fig 5-2:** OMRON E6B2 coupled with a screw shaft

**Fig 5-3:** Encoder Readings displayed on Serial Monitor

**Fig 5-4:** Instantaneous Encoder Readings displayed on Serial Plotter

**Fig 5-5:** FSR readings displayed on Serial Monitor

**Fig 5-6:** Plot of force readings on FSR readings against Motor holding current

**Fig 5-7:** Motor Control Block Diagram

**Fig 5-8:** Controller Input Block Diagram

**Fig 5-9:** Plot of force reading on FSR reading against feedback force.

**Fig 5-10:** Haptic Device Block Diagram

**Fig 5-11:** Haptic Feedback Device (Side View)

**Fig 5-12:** Haptic Feedback Device (Isometric View)

## **List of Tables**

**Table 1-1:** Gantt chart

**Table 1-2:** Individual Contribution

**Table 4-1:** Comparison of different transmission system options

**Table 4-2:** Comparison of different actuator options

**Table 4-3:** Comparison of the Sensor Options

**Table 6-1:** Benefits Of Commercializing robotic tele-surgery

**Table 6-2:** Sustainable Development Goals

**Table 6-3:** Potential Hazards and Safety Measures

# Chapter 1

## Introduction

### 1.1 Background and Motivation

The earliest sense to be developed in human embryology is the sense of touch and it is deemed to be of critical importance in clinical practice. With the advent of recent high bandwidth, reliable and accessible internet technology, medical services are witnessing a paradigm shift. Patients can now gain access to clinical facilities without the physical presence of medical professionals in their vicinity. One crucial example of this is tele-surgery where the surgeon will be able to perform surgery on a patient who is present at a remote location using robots that will replicate precise hand movements performed by the surgeon. While performing tele-surgery, the sense of sight of the surgeon is engaged as they can view the surgery through a screen. However, the important sense of touch is not engaged in tele-surgery currently. The motivation of this project is to improve the quality and reliability of tele-surgery by engaging the sense of touch in tele-surgical operations.

This can be achieved by the use of Haptic (Greek word meaning touch) Devices that implement a force feedback on the user's hand. Therefore, this project aims to initiate the incorporation of Haptic Force Feedback into tele-surgery. The result of this will culminate into a Haptic Robot Controller capable of Uniaxial Force Feedback of the force experienced by the end effector of the controlled robot.

### 1.2 Problem Statement

Current commercially available tele-surgery devices do not engage the sense of touch of the surgeon. The sense of touch is critically important for clinical practice. The purpose of this project is to initiate the incorporation of sensory touch into tele-surgery.

The goal of this project is to develop a Haptic Force Feedback Device. The primary goal is to implement uniaxial force feedback on the controller such that when there is a force experienced by the End Effector in the z-axis, as registered by the Force Sensitive Resistor, the same force is experienced by the user. Consequently, when the surgeon exerts a force on the paddle of the haptic device, the same force should be experienced by the end effector.

The exact position of the robot in 3D space will be input using a graphical user interface. Therefore, a major part of this project is to implement Forward and Inverse kinematics of the PUMA 560 robots such that its joint angles can be computed for it to approach the position input on the graphical interface. While the z-axis motion is controlled through the haptic device, the x and y axis are controlled using the graphical user interface.

The resulting product can serve as a basis for further research and industrial development into the field of tele-surgery as it will immensely improve the quality and reliability of the surgery.

### **1.3 Scope of the Work**

Over the span of 2 semesters, a range of allocated tasks are to be performed. Research into the design aspects of a haptic feedback device and its evaluation with respect to the application is a major part of the project. The project also largely comprises study of robotics theory, its implementation and programming. The Design of the Haptic Feedback device and control of the PUMA 560 robots are worked upon simultaneously on the course of the project.

The following specific tasks are worked upon over the course of the project.

- Design and fabrication of Haptic Force Feedback Device.
- Application of algorithms for 3D space movement of PUMA robot using GUI.
- Uniaxial movement of PUMA 560 using haptic controller in conjunction with GUI.

### **1.4 Expected Outcome**

- The user will be able to experience the uniaxial force on their hand as experienced by robot end effector.
- Smooth 3D movement of PUMA 560 robots to the positions specified by GUI and the controller.
- A robust working product that can be used for further research and development in tele-surgery industry.

### **1.5 Report Outline**

This report shall contain the following chapters.

Chapter 1 - Introduction: This chapter establishes a background and a motivation for the entire project along with the summary of what the group aims to achieve.

Chapter 2 - Literature Review: An overview of the previous research consulted about the current topic and how it influences the project.

Chapter 3 - Design and Analysis: Design methodology. Governing Equations. MATLAB code. Geometric Modeling.

Chapter 4 - Development process, Components selection

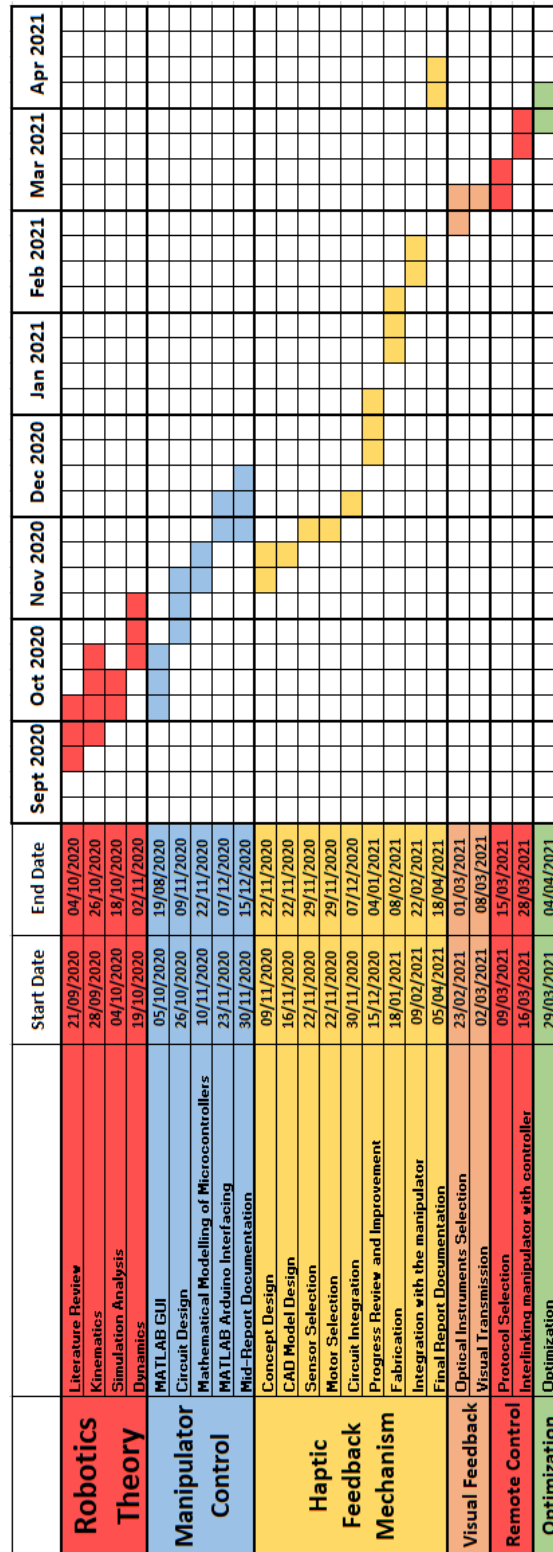
Chapter 5 - Results and Discussion

Chapter 6 - Social and Environmental Impact, SDGs satisfied, Hazard identification

Chapter 7 - Summary of the project and future recommendations

## 1.6 Project Schedule and Timeline

Table 1-1: Gantt chart



## 1.7 Individual and team contribution

Team division depended on each individual's subject of expertise and relative experience so that specialized work and productivity could be accomplished. The obligations of every individual member can be seen in Fig 1-2.

**Table 1-2:** Individual Contribution

Name	Reg No	Task
Maha Arshad	2017187	Haptic Analysis, Signal Processing, Haptic Device Assembly and Working, Remote Control
Muhammad Afaq	2017229	Mathematical Modelling, Signal Processing, Haptic Device Assembly and Working, Remote Control
Muhammad Fahad Saeed	2017256	CAD Model, Control Systems, Haptic Device Assembly and Working, Image Processing
Taha Mahmood	2017472	Graphical User Interface, Control System, Haptic Device Assembly and Working, Image Processing



## Chapter 2

### Literature Review

#### 2.1 Literature Review

For a considerable amount of time, efforts have been made to implement the concept of robotics in surgery. The first surgical robot, the PUMA 560, was used in 1985 to insert a needle in the brain for biopsy [1], a procedure which was previously subjected to errors due to hand tremors. In 1988, the PROBOT was used for prostate surgery, which allowed the surgeon to specify the volume of the prostate to be cut, and automatically cut it without further intervention. In 1992, the ROBODOC was used to assist a hip replacement surgery by drilling a cavity in the patient's femur with high precision [2].

In 1999, the DaVinci surgical system was introduced, which has been upgraded in the recent years to perform various surgical procedures. It consists of three main components, the surgeon console (master controller), the surgical cart (slave), and the vision cart. The surgeon sitting at the console views a clear 3D image of the field using two eyeholes. The system scales, filters and translates the movements of the surgeon's hand on the master control to four interactive robotic arms on the patient's cart. The vision cart monitors and displays faults in the instrument, thereby easing the maintenance procedure. It is the most advanced surgical system available today, however, it has one limitation: the force felt by the robotic arm is not felt by the surgeon.

Although increasing use of robots in surgical environments has been witnessed in the recent times, especially during the outbreak COVID-19, the introduction of haptic feedback in robotic surgery has not been fully explored yet. To fully understand the scope of developing a haptic device for tele-surgery, research was conducted in three major domains, namely:

- Control of the PUMA 560
- Design of Haptic Master Controller
- Remote Control

##### 2.1.1 Control of the PUMA 560

The first step in the designing a master controller is to establish a concrete understanding of the working principles of the slave. In our case, the PUMA 560 was used as the slave, and hence in-depth research was done to complete the kinematic and dynamic analysis, as well as to establish a virtual environment for the robot simulation.

To find the desired position of the end effector based on the joint's angles, we have to perform the forward kinematics on the PUMA 560 by manipulating the 6 DOF kinematic chain [6].

The inverse kinematic analysis can be performed by taking the position of the end effector as the input and calculating the joint angles through it [8].

### 2.1.2 Design of Haptic Master Controller

Haptics is a technology that simulates an experience of touch by applying forces and vibrations to the user [4]. Haptic devices can be split into two major categories - kinesthetic devices, and tactile devices. Kinesthetic devices apply forces to the user, usually through a grounded motor. Kinesthetic sensations are felt in the muscles, tendons and joints, while tactile devices simulate the skin to create contact sensations. Hybrid haptic devices also exist, which are aimed at combining tactile and kinesthetic feedback.

Kinesthetic haptic devices are further divided into two categories, impedance type and admittance type. In impedance type haptic devices [3], the forces experienced by the virtual environment, or the slave in our case, are transmitted to the kinesthetic haptic device with high precision, and are experienced by the end user. In case of admittance type haptic devices, the force on the slave is taken as the input, and the position of the haptic device is calculated as a function of that force, so the user just feels the motion of the device to the calculated position, and does not feel the force experienced by the slave.

The devices can further be divided into three types [12]:

- World-Grounded kinesthetic devices: These devices typically resemble a joystick, in which the user grasps a handle similar to a stylus or thimble. Through a motor connected to the handle, the user experiences a force. Such devices may possess one or more degrees of freedom, depending on the requirements.
- Body-grounded kinesthetic devices: These devices are grounded on the hand and provide force feedback to fingertips using cables.
- Wearable tactile devices: In such a device, individual fingers are grounded using motors, and the device deforms the skin in shear. They are usually attached to fingertips due to the high quantities of mechanoreceptors present. They allow the users to experience normal forces to the skin, as well as lateral translation.

To allow the haptic devices to experience a force feedback, force sensors need to be attached to the robotic arm. Although there may be various possible locations for the placement of these sensors, the ideal location is at the tip of the instrument inside the patient, that is, the end effector [7]. The sensors cannot be placed on instruments outside the patient's body, since they need to precisely measure the forces resulting from tissue interactions, and placing the sensor outside the body would lead to the undesirable forces being sensed on it. Placing the sensors on the instrument shafts is also not a feasible option, since forces on these shafts experience high variations, and are usually greater in magnitude than the interaction forces of the end effector with the body.

In the past, efforts have been made to incorporate haptic feedback in robotic minimally invasive surgery. A laparoscopic grasper was developed by MacFarlane, et al. [9], which utilized force feedback to enable surgeons to distinguish between the stiffness of tissues through a haptic interface, which proved to be inefficient compared to a gloved human hand. Deml, et al. [10] showed that although a haptic feedback reduced unintentional injuries during a dissection procedure, it increased the time consumed during the operating procedure. Wagner, et al. [11] analyzed a dissection task with video feedback, but in the absence of force feedback, and concluded that a much higher magnitude of forces was applied, leading to tissue damage.

### 2.1.3 Remote Control

Teleoperation is the procedure of operating a system that is at a remote location from the user. It consists of the two sites, the local site that includes the user, the haptic device, the monitor, and other tools required, while the remote site includes the robotic arm, sensors and controllers which facilitate interaction with the surgical environment. Telepresence is a concept that allows the user to perceive the remote environment through haptic feedback, without actually being present there. One approach discussed by Paulo, et al. [13] is using a robot operating system (ROS) for the robotic arm server, and the Unity platform for the haptic interface. An IEEE 802.11 wireless local area network can be used to establish communication between the robotic arm server and haptic interface server by making use of the WebSocket protocol, which is compatible with both the systems.

Zhangfang Ju, et al. [14] describes the use of a SensAble Omni Haptic Device to control a 7 degree of freedom robotic arm of the Baxter Robot. After calculating the co-ordinates of the end-effector of the master and slave respectively using the principles of forward and inverse kinematics, workspace mapping is performed to allow these to overlap by utilizing the principles of closed loop inverse kinematics. The haptic device is connected to a computer using the IEEE 1394 interface, another computer in the remote site is connected to the robot using an Ethernet cable. These two computers communicate via the internet, and allow data from the force sensor, kinematic sensor and camera at the remote site to be precisely transmitted to the master computer. Thus, it can be seen that two devices at separate locations can transmit data with high accuracy and precision, allowing efficient communication between the master and slave devices.

## 2.2 Inferences Drawn out of Literature.

### 2.2.1 Critical Analysis

Incorporating haptic feedback in surgical procedures is not a simple task. A.M. Okamura, et al. [7] discusses the position of the force sensor on the end effector, however, the major concern is the biocompatibility of these sensors, so that they do not give rise to harmful immune responses from the patient. They must also withstand harsh sterilization processes, and be

insensitive to the changing temperatures due to interaction with warm sensors inside the human body. Additionally, the prices of these sensors display an upward trend with higher degrees of precision, thereby making the overall system very expensive.

Majority of the devices that provide force feedback, but there is usually no programmable tactile feedback on these devices. Since surgical procedures require high levels of precision, force transmission from the robot to the haptic device needs to be very accurate, however, to achieve that, haptic devices with low mass and inertia need to be developed. This is a challenging task, since frictional losses between the joints can only be minimized, but not completely eliminated.

### 2.2.2 Most Appropriate Approach

After an in-depth analysis of the various possible alternatives, the most appropriate approach would be the fabrication of a one degree of freedom kinesthetic haptic device, resembling a manipulandum. This device would allow the user to move the end-effector of the robotic arm in the z-axis, while control in the x and y-axis would be established through touch-screen. A force sensitive resistor would be positioned at the end-effector, which would precisely measure the force experienced at the interface of the robotic arm with the patient's body, and this data would be transmitted to the motor on the haptic device, thereby allowing the user to experience force feedback. Although wearable haptic devices allow a much greater degree of maneuverability and control, it was beyond the scope of our final year project, and hence additional complications arising from complex devices were avoided. As far as remote control is concerned, research is still being carried out to decide upon the best strategy for performing tele-operations.

## 2.3 Summary

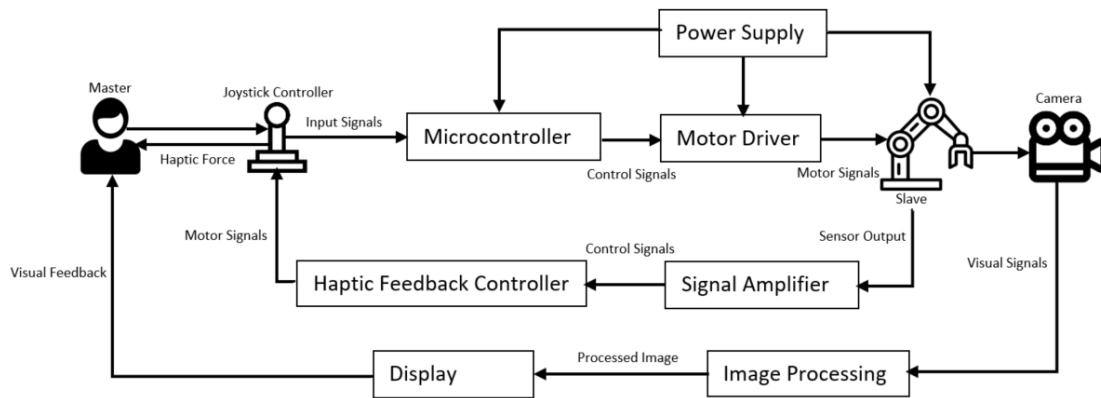
This literature review gave us an understanding of the mathematical modelling, programming and simulation of the robotic arm. It also helped us to distinguish between the different types of haptic devices, thereby allowed us to understand the advantages and disadvantages associated with each, and selecting the most appropriate device based on our design needs. It also enabled us to decide on the motors and sensors for the device, and gave us an insight into establishing remote control for tele-surgery.

## Chapter 3

### Design and Analysis

#### 3.1 Design Methodology

The design methodology that we have utilized for this project involves high level of systems control. The block diagram for the system can be shown in Fig 3-1 below. In order to control the slave (robotic arm) using the master controller (haptic device), we broke down the entire control operations into 3 smaller tasks and handled each task in a sequence. These tasks are:



**Fig 3-1:** Design Process Flow

- Designing the Robotic Control System

In this task, we will be designing the robotic control system which will allow us to control the end-effector position and orientation. This will be done by performing forward and inverse kinematics on the robotic arm, which will allow us to develop a relationship between the joint angles of the robotic arm with the end-effector position and orientation.

- Fabrication of Haptic Device

Here, we will design and fabricate a uni-directional manipulandum which will transmit a haptic force on the user, based on the forces applied on the robotic arm end-effector. The magnitude of the force applied on the haptic controller will be determined by the magnitude of the stress on the end-effector. To simulate this effect, the parameters of the haptic device need to be configured accordingly. This will be done by performing analysis on the controller geometry, as well as consult some previous literature review.

- Remote Control and Image Processing

The main purpose of our project is to create a tele-surgery device which will allow us to perform surgery in a remotely controlled environment. Hence, we need to ensure in our design

methodology that the robotic arm can be controlled remotely using a controller. This will be done by using the concept of Internet of Things (IOT).

Image Processing involves using an optical device to capture and transmit the images from the end effector to the user. These images provide a visual feedback to the user, allowing the user to control the end-effector with greater precision. Furthermore, the image captured will be processed using various image processing software allowing them to perform the tele-surgery with better efficiency.

## 3.2 Mathematical Model:

### 3.2.1 Forward Kinematics:

For a robot, forward kinematics refers to calculation of its end effector's position and orientation from joint coordinates. The forward kinematics is mostly done by attaching reference frames to each link according to Denavit-Hartenberg (D-H) convention. For an arbitrary link, the transformation matrix  $T_{i-1,i}$  is defined using four link parameters:

- The length of mutually perpendicular line between two joints, also known as link length, denoted by  $a_{i-1}$
- The link twist  $\alpha_{i-1}$  is the angle of axis  $i$  with respect to axis  $i-1$
- The link offset  $d_i$
- The joint angle  $\theta_i$

For revolute joints, all parameters except joint angles  $\theta_i$  are constants. The forward kinematics can be written as a product of these transformation matrices defined by D-H parameters. The D-H parameters of PUMA 560 are expressed in table 3-1 [15].

**Table 3-1:** Denavit-Hartenberg (D-H) Table

$i$	$\alpha_{i-1}/^\circ$	$a_{i-1}/m$	$d_i/m$	$\theta_i$
<b>1</b>	90	0	0	$\theta_1$
<b>2</b>	0	0.4318	0	$\theta_2$
<b>3</b>	-90	0.0203	0.15005	$\theta_3$
<b>4</b>	90	0	0.4318	$\theta_4$
<b>5</b>	-90	0	0	$\theta_5$
<b>6</b>	0	0	0	$\theta_6$

A newer, better, and more practical approach for solving forward kinematics is the product of exponentials (POE) formula [16]. In this approach, we define home position  $M$  of a robot end-effector when all joint angles are zero. Then we define a screw axis  $B$  for each joint of the robot, specify joint angles and input these into product of exponentials formula to find end-effector's configuration i.e.: position and orientation.

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & -1 & 0 & -1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ -0.15 & -4.318 & -4.318 & 0 & 0 & 0 \\ 0.452 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.452 & 0 & 0 & 0 & 0 \end{bmatrix} \quad M = \begin{bmatrix} 1 & 0 & 0 & 0.452 \\ 0 & 1 & 0 & -0.15 \\ 0 & 0 & 1 & 0.4318 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The columns of matrix  $B$  each represents screw axis of a joint in end-effector's frame. The first three terms in each represent the direction of angular velocity  $\omega$  of the joint and the last three terms represents its linear velocity  $v$ . Each screw axis is converted into  $se(3)$  matrix form and multiplied with its respective joint angle before taking its matrix exponential. The product of all these exponentials along with end effector's configuration  $M$  at home position is called product of exponentials formula. The MATLAB algorithm is shown in appendix A, the governing equation is shown below:

$$T = M e^{[S_1]\theta_1} e^{[S_2]\theta_2} e^{[S_3]\theta_3} e^{[S_4]\theta_4} e^{[S_5]\theta_5} e^{[S_6]\theta_6} \quad (3.1)$$

### 3.2.2 Inverse Kinematics:

It is the calculation of robot's joint angles given a specific orientation and position of end effector. It can be solved analytically; this method typically takes advantage of geometric insight into the problem, and the particular structure of the robot. To reach an end effector's position  $(p_1, p_2, p_3)$  relative to base frame, the effect of robot parameters like link lengths  $a_i$  and link offsets  $d_i$  needs to be considered [17]. This geometric approach utilizes planar projection robot link lengths and angles to inversely map coordinates to joint angles  $(\theta_1, \theta_2, \theta_3)$ . To calculate these, we consider some variables:

$$r = \sqrt{p_x^2 + p_z^2 - a_1^2} \quad (3.3)$$

$$\alpha_1 = \tan^{-1} \frac{a_1}{r} \quad (3.2)$$

$$\phi_1 = \tan^{-1} \frac{p_x}{p_z} \quad (3.4)$$

The first joint angle can be computed as:

$$\theta_1 = (\phi_1 - \alpha_1) - \left(\frac{\pi}{2}\right) \quad (3.5)$$

To calculate the second joint angle, the geometric relations are:

$$R = \sqrt{p_x^2 + p_z^2 + (p_y - l_1)^2 - a_1^2} \quad (3.6)$$

$$\alpha_2 = \tan^{-1} \frac{(p_y - l_1)}{r} \quad (3.7)$$

$$\cos \beta_2 = \frac{l_2^2 + R^2 - (l_3^2 + a_2^2)}{2 * l_2 * R} \quad (3.8)$$

$$\sin \beta_2 = \pm \sqrt{1 - \cos^2 \beta_2} \quad (3.9)$$

$$\beta_2 = \tan^{-1} \frac{\sin \beta_2}{\cos \beta_2} \quad (3.10)$$

The joint angle equates:

$$\theta_2 = \alpha_2 + \beta_2 \quad (3.11)$$

The third joint angle is calculated with these auxiliary variables:

$$\cos \phi_2 = \frac{l_2^2 + (l_3^2 + a_2^2) - R^2}{2 * l_2 * \sqrt{l_3^2 + a_2^2}} \quad (3.11)$$

$$\sin \phi_3 = \pm \sqrt{1 - \cos^2 \phi_3} \quad (3.12)$$

$$\phi_3 = \tan^{-1} \frac{\sin \phi_3}{\cos \phi_3} \quad (3.13)$$

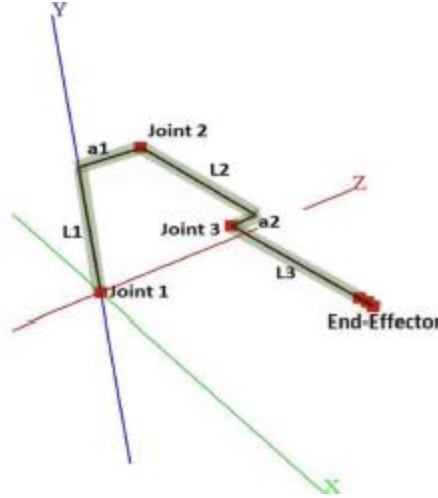
$$\beta_3 = \tan^{-1} \frac{l_3}{a_2} \quad (3.14)$$

The third joint angle can finally be calculated as:

$$\theta_3 = \left(\frac{\pi}{2}\right) - (\phi_3 - \beta_3) \quad (3.15)$$

These three joint angles approximate the position of the end effector as shown in Fig 3-2. Expanding to more DOF will make these calculations more complex and difficult to compute using this method.





**Fig 3-2:** Kinematic chain parameters for PUMA 560 [17]

We can also use Newton-Raphson numerical iterative method to solve the inverse kinematics. This approach requires an initial guess at a solution, then iteratively drives the initial guess toward a solution. The initial guess can be any set of joint angles  $\theta$ , we can then set error limits  $(\epsilon_\omega, \epsilon_v)$  to body twist. The algorithm iterates until the desired configuration  $T_{sd}$  is reached and outputs the joint angles at that configuration. The algorithm for Newton-Raphson inverse kinematic algorithm is as follows:

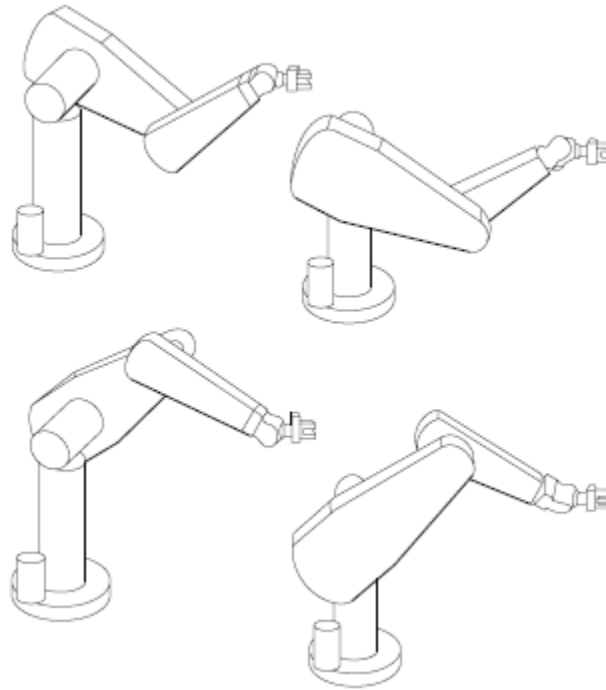
- a) Initialization: Given  $T_{sd}$  and an initial guess  $\theta^0 \in \mathbb{R}^n$ , set  $i = 0$ .
- b) Set  $[v_b] = \log(T_{sb}^{-1}(\theta^i)T_{sd})$ . While  $\|\omega_b\| > \epsilon_\omega$  or  $\|v_b\| > \epsilon_v$  for small  $\epsilon_\omega, \epsilon_v$ :
  - 1) Set  $\theta^{i+1} = \theta^i + J_b^\dagger(\theta^i)v_b$ .
  - 2) Increment  $i$ .

Where,

- $T_{sb}$  is the configuration matrix of current position i.e., at initial guess.
- $v_b$  is body twist vector consisting of  $\omega_b$  and  $v_b$ .
- $J_b^\dagger(\theta^i)$  is the pseudo-inverse of body Jacobian matrix at a given set of joint angles.

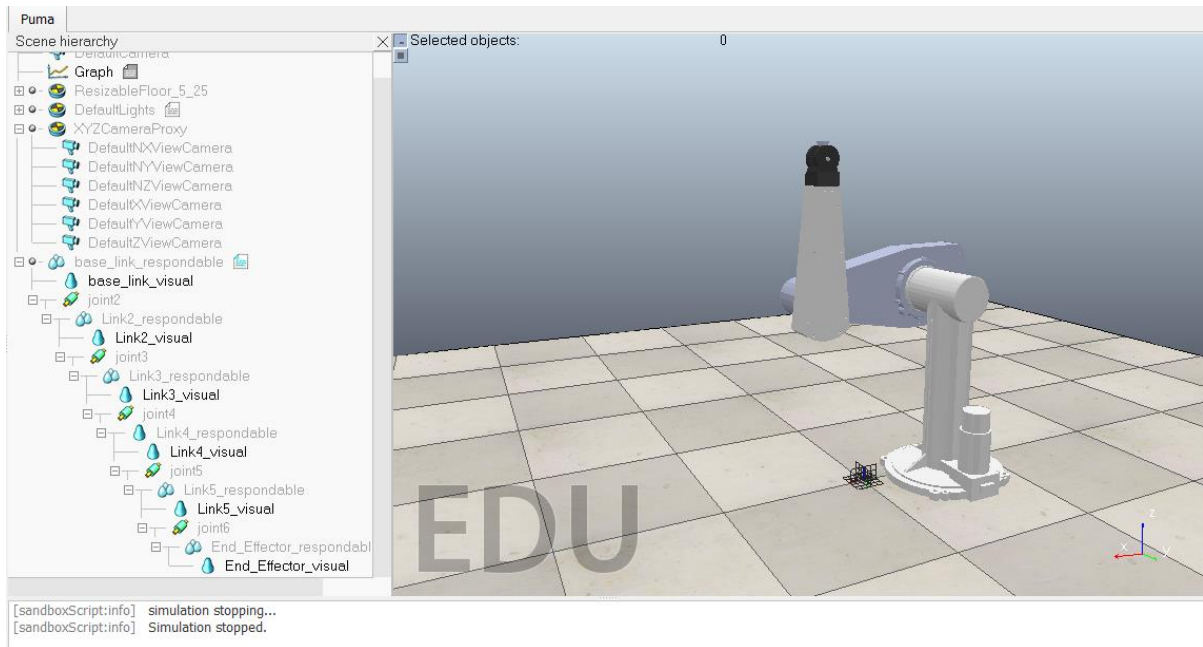
For this numerical inverse kinematics algorithm to converge to the desired solution, it is preferable that the initial guess  $\theta^0$  should be sufficiently close to the desired joint configuration  $\theta_d$ . This is done by starting the robot at a position where both joint angles and end-effector configuration are known and ensuring that the end-effector configuration  $T_{sd}$  changes slowly relative to the frequency of inverse kinematics calculations. Now, for the rest of the robot's run, set the calculated  $\theta_d$  as the initial guess  $\theta^0$  for the new  $T_{sd}$  at the next timestep until the solution converges.

The PUMA robot can reach certain configurations with 8 different solutions [18]. The Fig 3-3 shows different configurations of PUMA 560 robot at the same end effector configuration. The Moore-Penrose pseudoinverse  $J^+$  converges to the solution that is closest to robot's current configuration. If there exists a solution, then using  $J^+$  converges to  $\Delta\theta$  that has smallest 2-norm (the square root sum of the squares of vector elements of  $\Delta\theta$ ) among all solution. If there is no solution  $\Delta\theta$  that exists, the robot is at singularity or has fewer joints  $n$  than end-effector coordinates  $m$ , then the  $J^+$  satisfies conditions as closely as possible in 2-norm sense. The appendix B shows MATLAB code for solving inverse kinematics by Newton-Raphson iterative method.



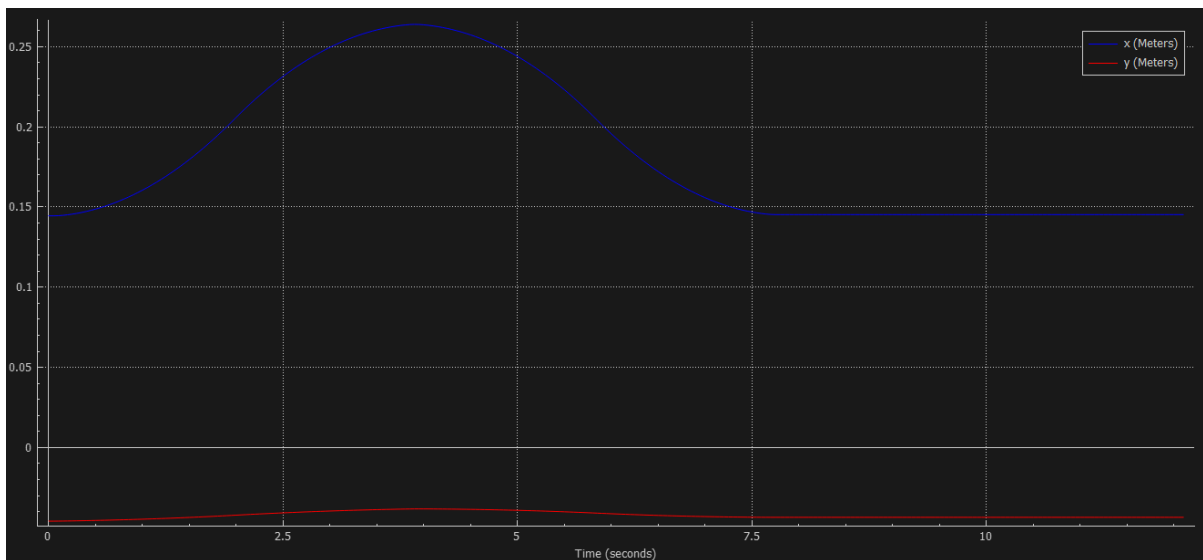
**Fig 3-3:** Four solutions of the PUMA 560 [18]

The best approach is to use both methods to find the kinematic inverse solution of the robot. Initially we can use the analytical approach to find three joint angles to reach robot in the desired position. Then this solution can be input in the Iterative algorithm to find the exact joint angles at desired position and orientation. Since input values for iterative algorithm should be a close to the solution, the analytic solution is the optimal consideration to that.



**Fig 3-4:** PUMA 560 on CoppeliaSim

We tested our code in Appendix B on CoppeliaSim (Virtual Robot Experimentation Platform). We created a Universal Robot Description Format (udrf) for PUMA 560 as shown in Fig 3-4 and input the results from our inverse kinematics algorithm to see if our calculations were correct. We then directed the robot to move 10 cm on positive x-axis, and then 10 cm back to its initial home position. Fig 3-5 shows how the end effector moves 10 cm back and forth with joint angles provided by inverse kinematics using Appendix B. It also shows that the change in y direction is negligible between start and end positions as we intended it to be.



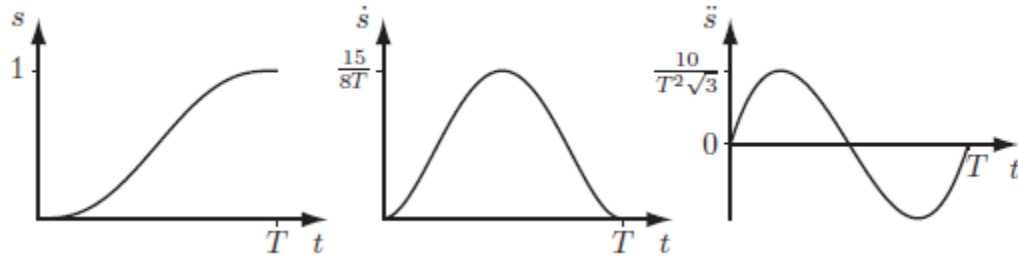
**Fig 3-5:** Plots of end effector's displacement 10 cm back and forth in positive x direction

### 3.2.3 Trajectories:

After kinematics is done, we need to define proper trajectories for our robot to follow. We define a scalar path parameter  $s$ , starting from 0 at the start of the path and ending at 1, at the end of the path at an arbitrary time  $T$ . Time scaling  $s(t)$  assigns a value  $s \in [0, 1]$  to each time throughout the trajectory period  $t \in [0, T]$ . We chose quintic time scaling for our path and after solving for boundary conditions, the time scaling was found to be:

$$s(t) = 10 * \frac{t}{T} - 15 * \left(\frac{t}{T}\right)^3 + 6 * \left(\frac{t}{T}\right)^5 \quad (3.16)$$

We used fifth order polynomial for time scaling because we can use six terminal position, velocity, and acceleration constraints to solve for a uniquely for the polynomial, which yields smoother motion. The issue with the use of third order polynomial was that the robot is asked to make a discontinuous jump in acceleration at starting and end point which implies infinite jerk.



**Fig 3-6:** Plots of a quintic time scaling [19]

Since a trajectory consists of both time scaling and path, for path we can either use straight line path, or a screw path. For a straight-line path, using inverse dynamics algorithm to calculate joint angles at start and end and by using appropriate time scaling which in our scale is fifth order polynomial (quintic) time scaling, we can simply use these equations to compute the point-to-point straight-line trajectory:

$$\theta(t) = \theta_{start} + \left(10 * \frac{t}{T} - 15 * \left(\frac{t}{T}\right)^3 + 6 * \left(\frac{t}{T}\right)^5\right) (\theta_{end} - \theta_{start}) \quad (3.17)$$

$$\dot{\theta}(t) = \left(\frac{10}{T} - 30 * \frac{t^2}{T^3} + 30 * \frac{t^4}{T^5}\right) (\theta_{end} - \theta_{start}) \quad (3.18)$$

$$\ddot{\theta}(t) = \left(-60 * \frac{t}{T^3} + 120 * \frac{t^3}{T^5}\right) (\theta_{end} - \theta_{start}) \quad (3.19)$$

The screw path has constant twist. The end effector does not generally follow straight line in Cartesian space using this since it follows screw motion. The path therefore can be written in the form of transformation matrix  $X$ :

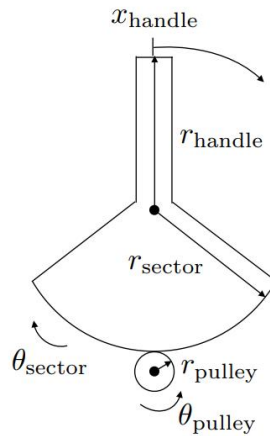
$$X(s) = X_{start} * \exp (\log (X_{end}^{-1} X_{end})s) \quad (3.20)$$

The transformation matrix at each time step can be converted into joint angles using inverse kinematics algorithm hence provide trajectory of robot motion. The MATLAB code for screw trajectory generation is provided in Appendix C. We need to input the starting and ending transformation matrices, the total time for robot motion, number of points in discrete representation of trajectory and the order of polynomial as input to generate trajectory i.e., output in terms of several transformation matrices along time. The decision to use either straight line Cartesian, or screw trajectory will be made after testing both trajectories on the robot.

#### 3.2.4 Haptic kinematics and dynamics

After designing the manipulandum, the kinematics of the haptic device needed to be analyzed. This was done by dividing the components into separate parts. Since, the motion and the torques are generated with respect to a single axis only, we need to consider a single dimension while analyzing the forward kinematics.

The design of our haptic can be simplified into 3 components; the handle, the sector and the pulley attached to the motor. The handle and the sector are pivoted at a point between the handle and the sector. Meanwhile, the bottom surface of the pulley with the friction belt attached to it is in contact with the pulley. Hence, when the pulley rotates at the pivot, the sector also rotates with equal displacement.



**Fig 3-7:** Haptic Controller Diagram [20]

## Displacement

The pulley outer surface is in contact with the friction belt of the sector outer surface. If both surfaces are in contact with no slippage between them, then the angular displacement covered by the pulley and the sector will be equal.

$$s_{pulley} = s_{sector} \quad (3.21)$$

$$r_{pulley} \times \theta_{pulley} = r_{sector} \times \theta_{sector} \quad (3.22)$$

$$\theta_{sector} = \frac{r_{pulley} \times \theta_{pulley}}{r_{sector}} \quad (3.23)$$

Similarly, since the handle and pulley are pivoted at the center, hence the angular displacement covered by the pulley and the sector will be the same too.

$$s_{handle} = s_{sector}$$

$$r_{handle} \times \theta_{handle} = r_{sector} \times \theta_{sector} \quad (3.24)$$

However, we will consider the angular displacement of the sector as  $x_{handle}$  along the handle. Such that

$$x_{handle} = r_{handle} \times \theta_{handle} \quad (3.25)$$

Hence,

$$x_{handle} = r_{handle} \times \theta_{sector} \quad (3.26)$$

Substituting Eq 3.24 in the above equation

$$x_{handle} = r_{handle} \times \frac{r_{pulley} \times \theta_{pulley}}{r_{sector}} \quad (3.27)$$

$$x_{handle} = \frac{r_{pulley} r_{handle}}{r_{sector}} \times \theta_{pulley} \quad (3.28)$$

Using simple analysis, we were able to derive a relationship between the angular displacements of the pulley to the angular displacement of the handle.

## Kinematics

Now, we will derivate the above equations with respect to time to find the relationship between the motion of pulley to the handles.

$$s = r\theta \quad (3.29)$$

Differentiating with respect to time

$$(3.30)$$

$$(3.31)$$

$$\frac{ds}{dt} = \frac{d(r\theta)}{dt}$$

$$\frac{ds}{dt} = r \frac{d\theta}{dt} + \frac{dr}{dt} \theta$$

Since the value of  $r$  will remain constant, hence  $\frac{dr}{dt} = 0$ . and  $v = \frac{ds}{dt}$

Hence,

$$v = r \frac{d\theta}{dt} \quad (3.32)$$

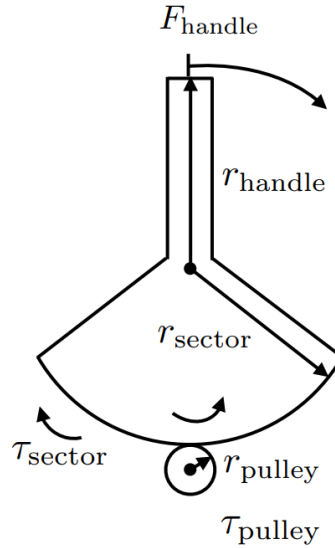
Performing similar analysis on the displacement equation of the haptic device

$$v_{handle} = \frac{r_{pulley} r_{handle}}{r_{sector}} \times \frac{d\theta_{pulley}}{dt} \quad (3.33)$$

$$v_{handle} = \frac{r_{pulley} r_{handle}}{r_{sector}} \times \frac{d\theta_{pulley}}{dt} \quad (3.34)$$

$$v_{handle} = \frac{r_{pulley} r_{handle}}{r_{sector}} \omega_{pulley} \quad (3.35)$$

## Statics



**Fig 3-8:** Free-body diagram [20]

In force torque relationship,

$$\tau = Fr \quad (3.36)$$

Where  $r$  is the pivot distance from the force applied to the center where the torque is produced.

According to Newton's 3<sup>rd</sup> law of motion, the force applied by the pulley on sector will be equal to the force applied by the sector on the pulley.

$$F_{pulley} = F_{sector} \quad (3.37)$$

$$\frac{\tau_{pulley}}{r_{pulley}} = \frac{\tau_{sector}}{r_{sector}} \quad (3.38)$$

Similarly,

$$F_{handle} = \frac{\tau_{sector}}{r_{handle}} \quad (3.39)$$

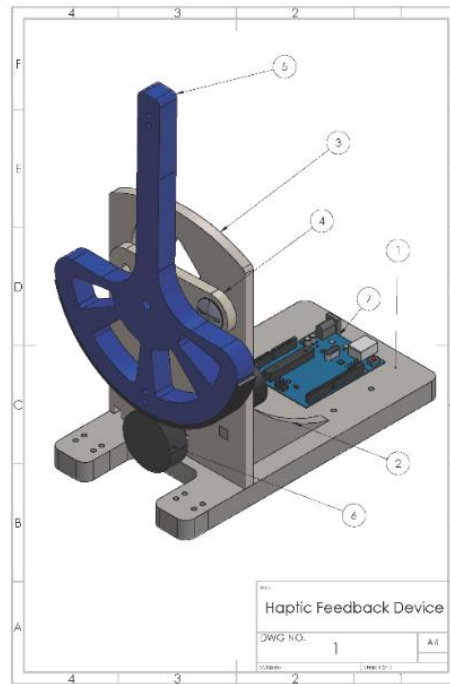
$$F_{handle} = \frac{r_{sector}}{r_{handle}r_{pulley}} \tau_{pulley} \quad (3.40)$$

With this equation, we will be able to find a relationship between the torques generated by the motor to the force produced on the handle.



### 3.3 Geometric Modeling and Design

The haptic device controller is envisioned to be compact, capable and easy to assemble. Individual parts are to be fabricated from materials easily available such that they could be bought easily and manufactured promptly.

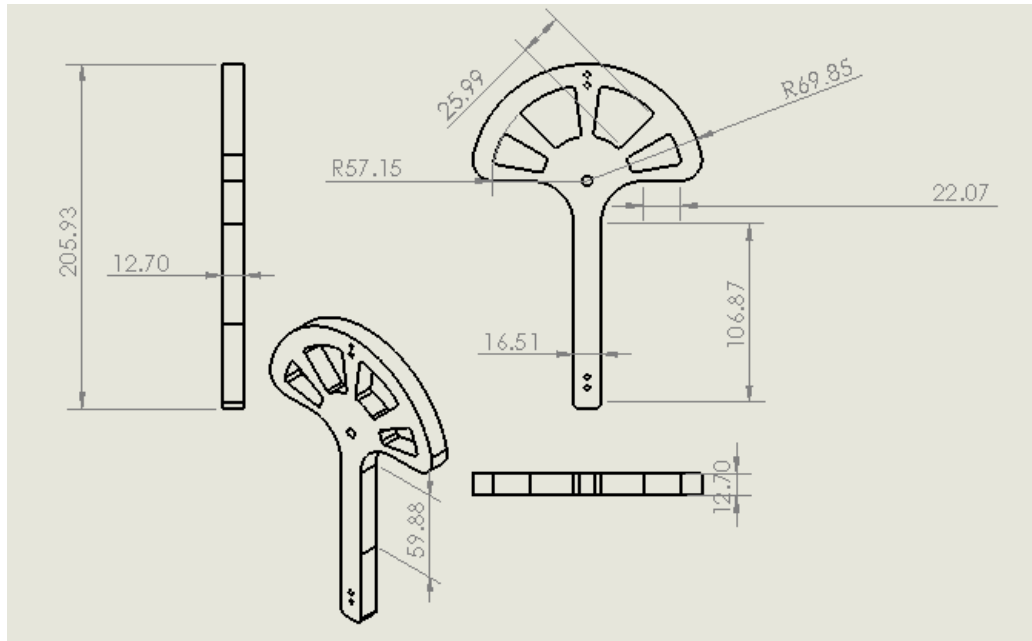


**Fig 3-9:** Assembly of the Haptic Feedback Device

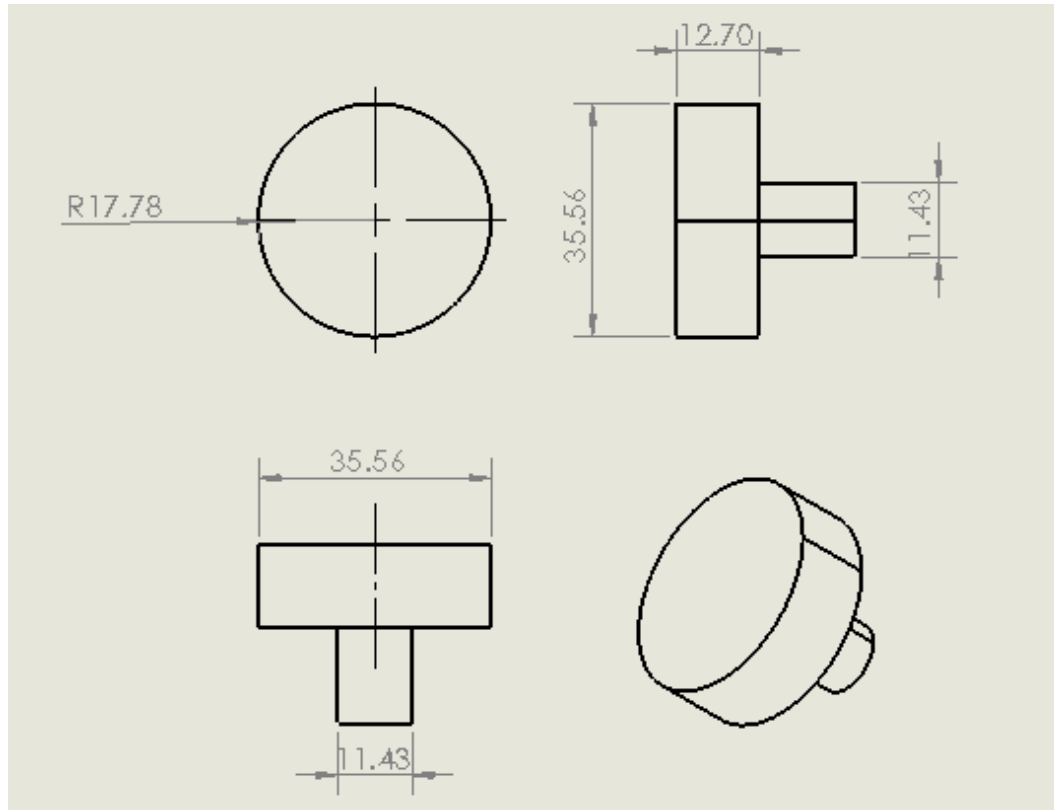
The Assembly can be seen in Fig 3-9. The belt is attached to the handle using adhesive and makes contact with the wheel. The wheel is attached to the motor, and transmits a torque from the motor to the paddle. The exact dimensions of the wheel can be seen in Fig 3-11. The part of the wheel with the smaller radius will fit through the front plate and will be attached to the motor. The balloon label for the haptic device has been compiled to form a list of parts shown in Table 3-2.

**Table 3-2:** List of parts

Item No.	Part Number	Description	Quantity
1	Base Plate	Base structure to support the assembly.	1
2	Support	Sturdy attachment of front plate to base plate	2
3	Front Plate	Structure to support wheel ad paddle	1
4	Thin Plate	Sturdy Attachment of paddle to front wheel	1
5	Paddle	Takes input from the user and initiates a force feedback	1
6	Wheel	Applies a force feedback on wheel	1
7	Arduino	Programming the circuit	1

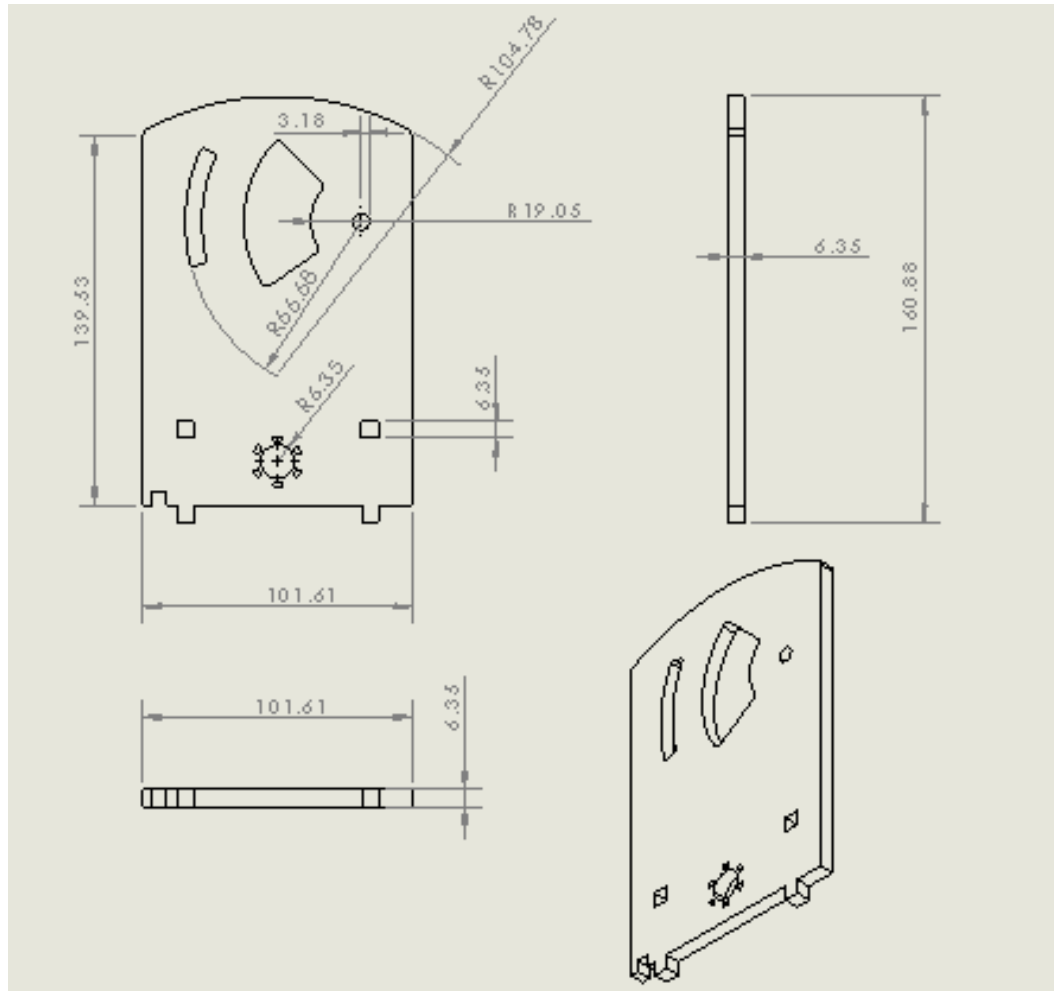
**Paddle:****Fig 3-10:** Paddle (all dimensions in mm)

The paddle can be seen in Fig 3-10. The tip of the paddle is where the user will exert an input and experience a force feedback. The paddle is pivoted at the center hole as can be seen in the Figure. There are empty spaces within the paddle in order to reduce the mass of the paddle. The semi circular base of the paddle will come into contact with the wheel (shown later in the section) such that the friction between the bottom of the paddle and the wheel will cause a torque on the paddle and this will result in the user experiencing the force feedback.

**Wheel:**

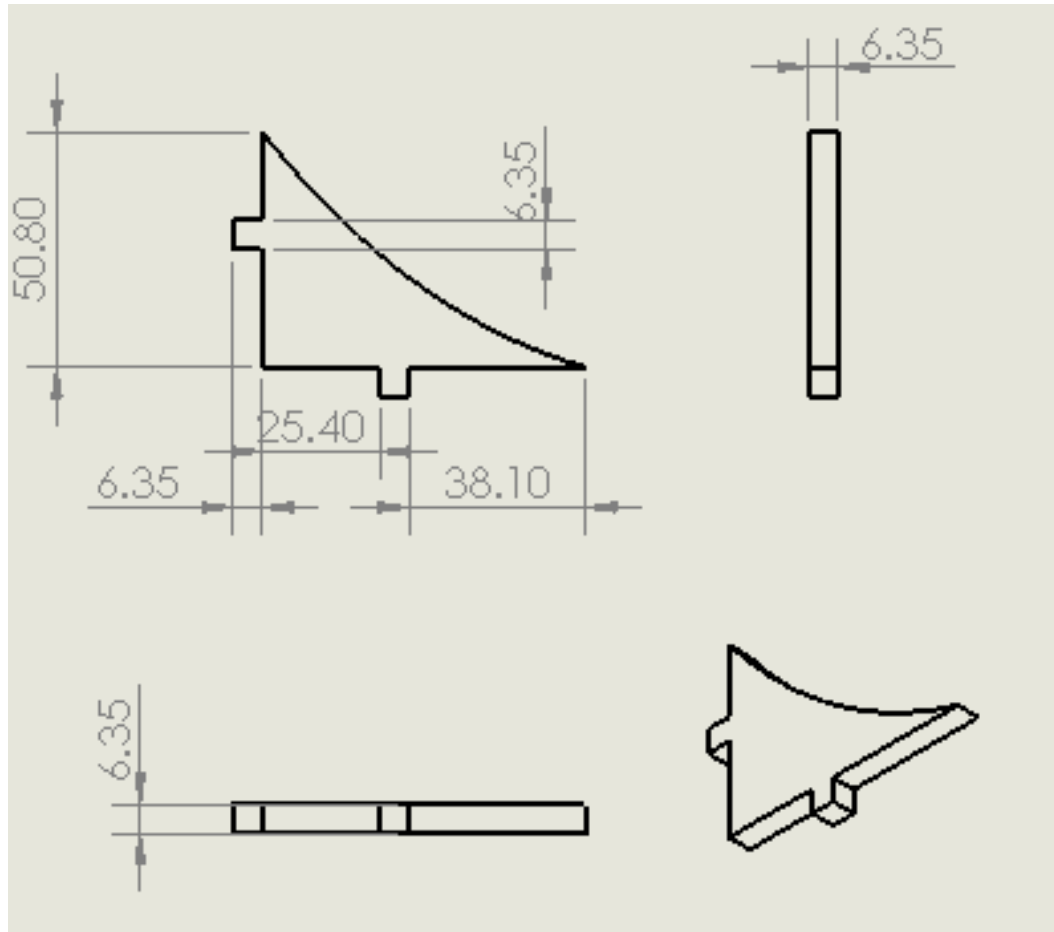
**Fig 3-11:** Pulley (all dimensions in mm)

The wheel is attached to the motor and transmits torque from the motor to the paddle. The exact dimensions of the wheel can be seen in Fig 3-11. The part of the wheel with the smaller radius will fit through the front plate and will be attached to the motor.

**Front Plate:****Fig 3-12:** Front Plate (all dimensions in mm)

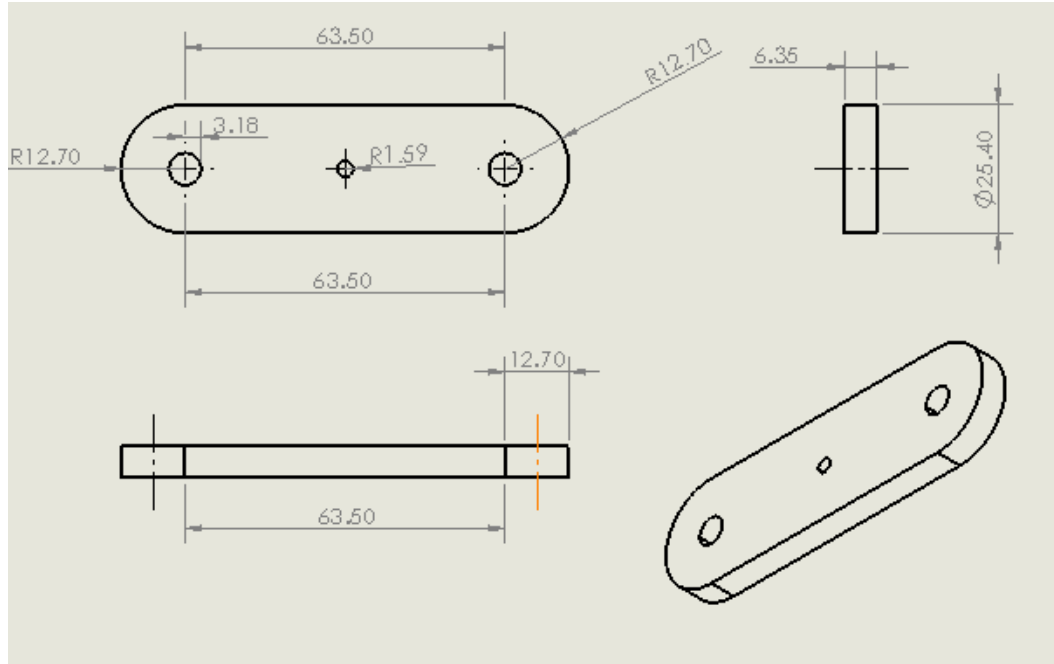
The front plate and its dimensions can be seen in Fig 3-12, the front plate will act as a structure which will hold the critical components of the haptic feedback device. The hole at the bottom is for the wheel and the spaces on the upper half are made in order to attach the paddle to the front plate. The front plate will be attached to another structure called the baseplate by the help of supports and adhesives. The front plate can be fabricated with acrylics hence it will have less weight and there will be ease in manufacturing.



**Support:****Fig 3-14:** Support (all dimensions in mm)

The supports can be seen in the Fig 3-14. They will be attached to the front plate with the help of slots and adhesives. They will be attached to the base plate on the other end.

### Swing Bar:



**Fig 3-6:** Swing Bar (all dimensions in mm)

The bar can be seen in Fig 3-15. It will be screwed to the front plane using the side holes, and the pedal will be attached to it through the center hole.

## 3.4 Analysis

### 3.4.1 Haptic controller analysis

In our CAD Model for haptic feedback, the values of these parameters are: -

$$r_{handle} = 136 \text{ mm} = 0.136 \text{ m}$$

$$r_{sector} = 70 \text{ mm} = 0.070 \text{ m}$$

$$r_{pulley} = 17.78 \text{ mm} = 0.01778 \text{ m}$$

Similarly, the angle of the sector in the CAD model is: -

$$\theta_{pulley} = 159.24^\circ = 2.78 \text{ rad}$$

### Displacement

Inputting these parameters in the displacement equation

$$x_{handle} = \frac{r_{pulley} r_{handle}}{r_{sector}} \times \theta_{pulley} \quad (3.41)$$



$$x_{handle} = \frac{0.136 \times 0.01778}{0.070} \times 2.78 \quad (3.42)$$

$$x_{handle} = 0.096m = 9.6 \text{ cm}$$

It can be analyzed that the range of the handle from one end to the opposite end is 9.6 cm. This means that the handle has a range of motion from -4.8 cm to 4.8 cm along the curvilinear path.

### Kinematics

Inputting these parameters in the kinematics equation

$$v_{handle} = \frac{r_{pulley} r_{handle}}{r_{sector}} \omega_{pulley} \quad (3.43)$$

$$v_{handle} = \frac{0.136 \times 0.01778}{0.070} \omega_{pulley} \quad (3.44)$$

$$v_{handle} = 0.0345 \omega_{pulley}$$

The equation mentioned above correlates the angular velocity of pulley to the linear velocity along the curvilinear path of the controller handle.

### Torque

Inputting these parameters in the torque equation

$$F_{handle} = \frac{r_{sector}}{r_{handle} r_{pulley}} \tau_{pulley} \quad (3.45)$$

$$F_{handle} = \frac{0.070}{0.136 \times 0.01778} \times \tau_{pulley} \quad (3.46)$$

$$F_{handle} = 28.948 \times \tau_{pulley}$$

In our haptic design, the joystick handle will experience the force between 0 N to the maximum of 7.5 N. Since, haptic senses have a very high precision of 0.06 N. The range of these forces will be significant or the user to experience a force feedback.

If the maximum force of 7.5 N is applied on the handle, then.

$$F_{handle} = 28.948 \times \tau_{pulley} \quad (3.47)$$

$$\tau_{pulley} = \frac{F_{handle}}{28.948} \quad (3.48)$$

$$\tau_{pulley} = \frac{7.5}{28.947} = 0.25908 \text{ N.m}$$

The motor installed in the pulley will have to apply the maximum torque of 0.25908 Nm or 2640 g-cm.

#### 3.4.2 Environmental and Social Impact

The proposed design has wide ranging social and environmental implications. Firstly, it will considerably improve the quality of surgical facilities that can be provided in 3rd world areas and remote countries. This is because a number of countries do not have certain specialized surgery options and the haptic design can facilitate remote surgery.

Furthermore, it will allow for more mobility for people who require specialized medical care because they would not have to rely on being physically restrained to an area where they could be instantly provided medical services. Facilitating tele-surgery through haptics can also be highly beneficial for health workers, especially in situations like the current scenario where health workers risk contracting the COVID-19 virus due to their work involving close proximity to the infected people. This need to be present close will be eliminated and health workers would not have to work in a hazardous workplace.

The manufacturing of the haptic device does not consume massive power to manufacture and therefore the manufacturing and fabrication process saves energy and reduces the CO<sub>2</sub> emitted during electrical power production. Furthermore, it requires minimal electricity to operate as well so it has a very minute carbon footprint. If patients would not have to board flights to get specialized medical service, it will save the fuel used to operate airplanes and therefore benefit the environment that way. Aircrafts consume a considerable amount of fuel burning which could greatly contribute to global climate change, this facility can reduce the burning of fuel and benefit the environment.

### 3.5 Summary

The design and analysis part of the project involved a system for controlling the robotic arm and the design and development of a haptic feedback device. Robotic control was achieved through evaluation of the forward kinematics of the robot using (D-H) convention and Product of Exponentials (POE) formula. The inverse kinematics was solved utilizing planar projection robot link lengths and angles to inversely map coordinates to joint angles and iterate them using Numerical iterative Newton-Raphson algorithm to obtain desired orientation and configuration. The Force-Torque relationship on the proposed haptic design was studied to obtain the torque relations on the pulley.

The haptic geometric model was based on rapid assembly and ease of manufacturability. Using the torque relations, the exact torque required at the pulley was evaluated using the geometric model. The vast social and environmental advantages of the haptic device such as physical mobility and lack of jet fuel burning are also explained in the section.

## Chapter 4

### Physical Model Development and Testing

#### 4.1 Development Process

Once the design is finalized, we will start the development process of the project. We will start by manufacturing of the haptic device. Furthermore, we will also design the control system of the PUMA 560. We also need to design a Graphical User Interface which will allow the user to control the robotic arm using an interface.

##### **Fabrication**

To fabricate the haptic device, we will be utilizing the 3D printer to manufacture each part individually. The 3D printer utilized for the fabrication of the device is available on commercial market across the nation as well as the institute. To fabricate the part, we will upload the STL part to the printer program, and commence the operation. The fabrication of the entire model can take up to several hours, depending on the speed of the operation.

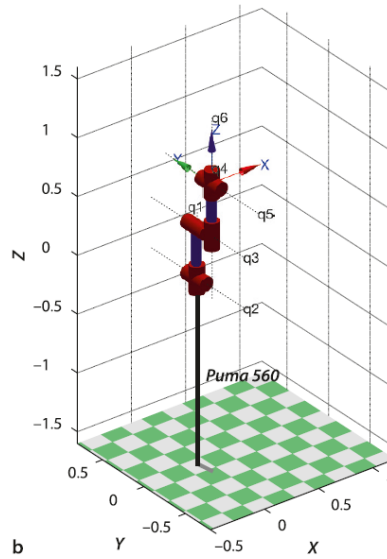
##### **Control System**

Programmable Universal Machine for Assembly 560 or simply PUMA 560 is an industrial robotic arm developed by Unimation. The design of PUMA 560 was inspired by a human arm where it contains 6 axis arms with motors in which 3 of its axes control the rotary arm while the other 3 axes control the spherical wrist. In short, it is a robotic arm which contains 6 revolute joints with each of its axis. The angular position of each individual joints controls the end effector position and orientation.

In order to successfully control robot's end effector position and velocity, we had to simultaneously control each motor attached to the axis of rotation. A program had to be written which allows us to send signals to each joint simultaneously. To do so, the Peter Corke's Robotic Toolbox was used.

The Peter Corke's Robotic Toolbox is a toolbox which allows to control and simulate classical arm-type robotics. This toolbox also provides with numerous functions which allows us to understand and study the kinematics and dynamics of the robotic arm.

This allows us to simulate control of all 6 revolute joints in the PUMA 560 robotic arm. Initially, the robotic arm is set to the "ready" position. It is the position where all the arm is straight and vertical, and it is defined by the angles  $(0, \frac{\pi}{2}, -\frac{\pi}{2}, 0, 0, 0)$  where each element highlights the angle of each robotic arm. Fig 4-1 below simulates the "ready" pose. We can simulate robotic arm movement from this position to any desirable position in the robot's workspace.

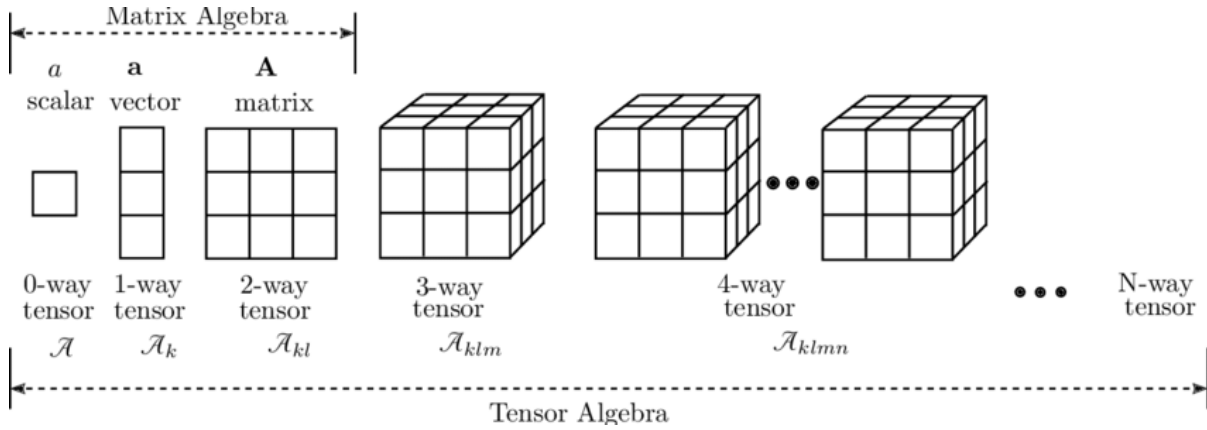


**Fig 4-1:** Orientation of PUMA 560 in "ready" position [21]

### Image Processing

Image Processing consists of processing digital images based on the requirements using an algorithm. Such algorithms are implemented to fine-tune the visuals of the image, such as removing the picture noise and distortion. A two-dimensional image can be enhanced to multiple dimensional images, enhancing many features of the particular image. This is done by manipulating the tensors of the image, where each tensor of the image represents the color scheme of a single pixel in the picture. The tensor algebra can be depicted in Fig 4-2.

MATLAB Image Processing Toolbox provides us with various algorithms and workflow applications for image processing, analysis, and visualization. Furthermore, it allows us to automate image workflow, allowing us to process image data sets. In our project, the MATLAB Image Processing Toolbox was utilized to process the image captured from the image capturing device.



**Fig 4-2:** Tensor Algebra

A USB port 1024P Webcam was used as the image capturing device in our project, as shown in Fig 4-3. The Webcam orientation is set at a particular angle which allows us to capture the visuals of the entire robotic arm configuration. Furthermore, the image is captured at the rate of 30 frames per second, allowing for the movement of the robotic arm to be observed with high quality and smooth transition. To connect the USB Webcam with the User Interface, a USB Webcam MATLAB library was used. The library initiates a serial connection between the Webcam and the MATLAB code. The image is then uploaded in MATLAB in a series of image tensors manipulated through image processing techniques.



**Fig 4-3:** 1080p USB Webcam

To facilitate the user, a series of features were installed on the image. One of the features that was incorporated is the force detection feature. Once the end effector has contacted an object, the image's outline turns red, notifying the user about the contact. Furthermore, the value of force that is felt on the end effector is also displayed in the image. If excessive force is applied to the particular organ, the user is immediately made aware of this using a series of emergency

noises. The implementation of this feature can ensure the safety of the operation. Furthermore, it also improves the surgeon's experience with the robotic arm.

Numerous features can be implemented on the robotic arm using image processing. For example, in the future, deep learning can be utilized the image and allow us to process a large amount of image data. This can be extended, allowing us to process the image to identify the organ and blood vessels. Alternatively, discrepancies in the motion of the robotic arm can also be reduced using such techniques.

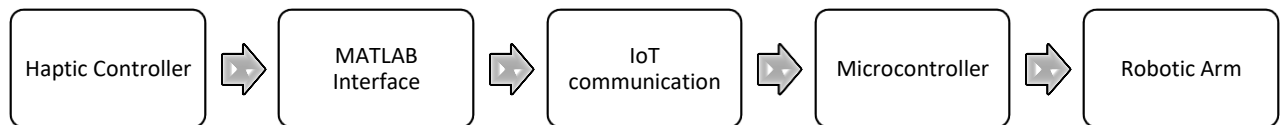
With Image Processing, the visuals of the robotic arm motion were captured at the rate of 30 fps, the users understood the movement of the robotic arm quite clearly. However, due to the poor quality of the camera, the resolution of the image was compromised. This drawback can easily be overcome by installing a high-quality visual image capturing device with a better fps rate. Similarly, replacing the 60Hz monitor with a 144Hz monitor can also improve the quality and framerate of the motion.

Furthermore, the force detection feature that was installed worked quite effectively. MATLAB was able to update the value of force on the screen after every 1 microsecond. However, a delay of a few seconds was noticed when the loop was executed. This was due to the communication delay between the MATLAB and the Arduino microcontroller.

### Remote Control

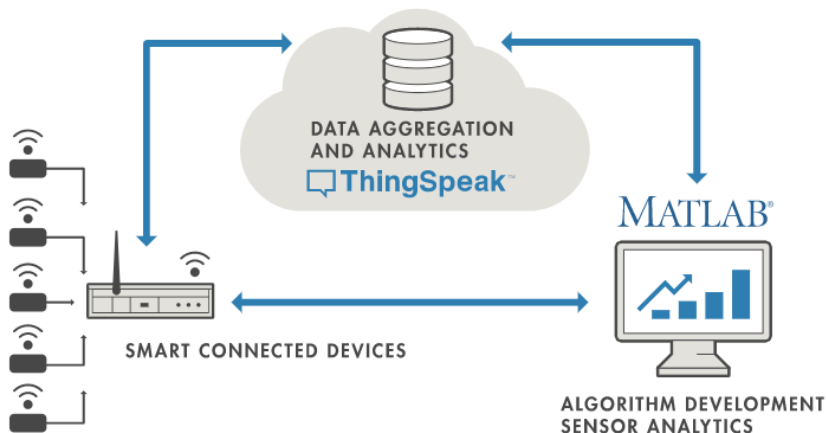
To operate the robotic arm remotely, IoT was utilized to ensure a remote connection between the master controller and the slave robot. Internet of Things (IoT) is a concept of communications between multiple devices equipped with sensors and actuators for convenience and economic benefits. Furthermore, IoT allows us to transfer a mass of data from one data point of the infrastructure to another.

Using this technology, the user interface of our control system can communicate effectively with the robotic arm through the microcontroller. However, a communication infrastructure between the master controller and the slave robotic arm was designed to do so. The infrastructure is highlighted in Fig 4-4 below.



**Fig 4-4:** Remote Control Flowchart

In our project, an open-source Internet of Things application is used to implement IoT serial communication, known as ThingSpeak. ThingSpeak is an open-source Internet of Things application and API to store and retrieve data from things using the HTTP and MQTT protocol over the Internet or via a Local Area Network. In addition, cloud-to-cloud integrations with The Things Network, Senet, the Libelium Meshlium gateway, and Particle.io enable sensor data to reach ThingSpeak over LoRaWAN® and 4G/3G cellular connections. Furthermore, ThingSpeak can be integrated with both MATLAB and Arduino to establish a communication channel between the ends. This is shown in Fig 4-5 given below.

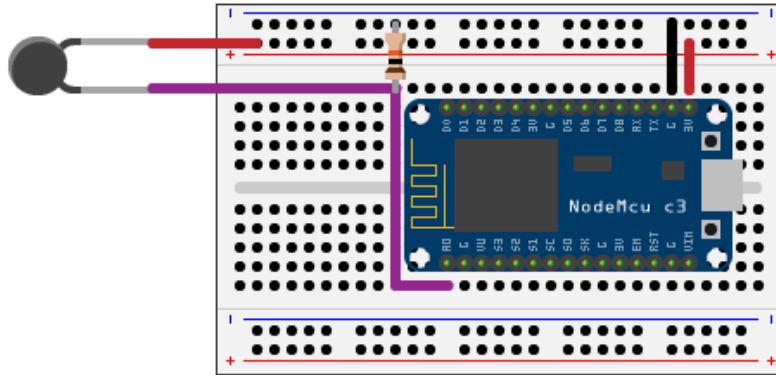


**Fig 4-5: ThingSpeak diagram**

The device can be easily configured through this application to send data and retrieve data through various IoT protocols. In our project, this application is used to configure MATLAB to retrieve data from the user interface and then send the data through ThingSpeak to the Arduino microcontroller via the WLAN protocol. With this, a communication protocol can be established between the master controller and the slave robotic arm.

To get started with ThingSpeak, first, we need to create a channel. To do so, we click on 'New Channel' and then input the required data in the designated fields. Hence, the Channel ID, the required data field, and the API read key are provided as inputs. With this, the channel between the MATLAB interface and the Arduino is designated.

Since we are using the WLAN protocol for communication, Wi-Fi is used to send the data. To do so, the Arduino microcontroller is connected to the Wi-Fi module ESP8266. This is done using the ESP-01 adapter. With this connection, the Arduino is compatible with the Wi-Fi system, and the Wi-Fi credentials are added to the microcontroller using the Simulink.



**Fig 4-6:** Circuit Diagram

Once the appropriate connection is fastened, the IoT communication is initiated in the software. This is done by adding read API key and write API key, channel numbers, and wireless network connection information. Once this is done, the device is connected to the wireless network using the *connectWiFi* function. Now the series of bytes can be sent through the Wi-Fi connection using the channel that has been designated.

Furthermore, this technique can be extended to visualize the data through the IoT protocol. Data can be explored using interactive visualizations such as an area plot, line plot, or scatter plot in static visualizations using other MATLAB plots. However, MATLAB in-built functions will be employed to create the visual graphs. However, once this project is implemented on a broader scale, data visualization through ThingSpeak will be employed to view data in real-time.

### Graphical User Interface

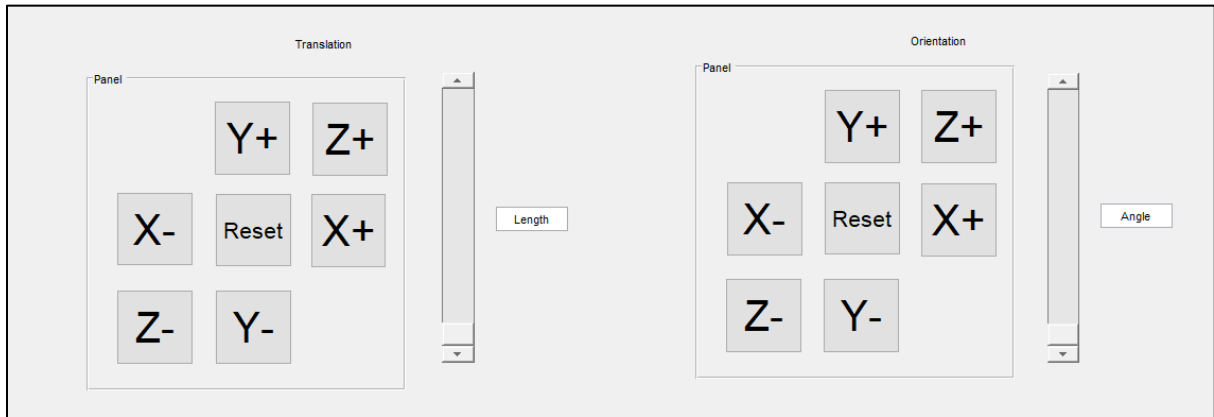
After designing the control system, we intend to use the code to control the end effector on each particular axis and orientation. This was done by streamlining the process. First, we created a motor function which sensed the initial and final position of the robotic arm and then the rotated each motor with the appropriate angle at the axis.

This function allows to control each motor specifically and rotate the specific angle using the rotary motors. Later, this was incorporated on the graphical user interface. The graphical user interface allowed us to control the transitional and rotational motion of the robot's end effector. This was done by using the forward and inverse kinematics shown earlier, and using functions provided in Peter Corke library.

First, by calling `mdl_puma560` command from the library, we load Model PUMA onto the code. Then it requires an input from the user who selects the magnitude of displacement, and the direction in which the robotic arm is supposed to move to. Using the user's input, it calculates the robotic arm initial and final position and then performs inverse kinematics on



the motion. This outputs 2 sets of 6 angles of stepper motor where the initial and final position are specified.

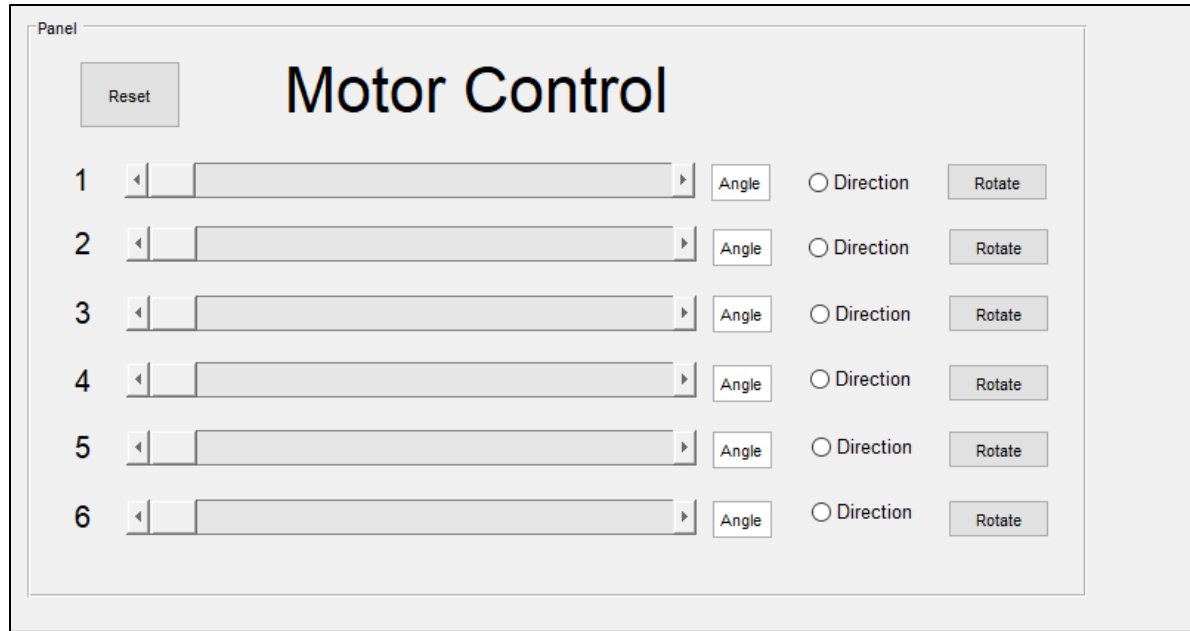


**Fig 4-7:** Graphical User Interface controlling the position and orientation of end-effector

For example, if we select the Z+ button on the translation panel in Fig 4-7, then the code in Appendix D is executed.

With this code, first the code inputs the value set on the slider of the transitional bar. Then, it detects the previous position of the end effector of the robot using the previous calculations. Then after using inverse kinematics used in the `tmove` function, it outputs the final configuration of the angle of each robotic arm. Then using the motor function mentioned above, it incorporates the initial and final position. Then using motor control and serial communication, the end-effector translates from one position to another position. This way the end effector position and orientation can be controlled.

With the previous code, we were able to control the end-effector based on its own axis and orientation. We were able to direct the movement of the end-effector to a particular direction according to the axis defined on the end-effector. Hence, we also came up with another code which allowed us to control the stepper motor on each particular axis individually using the GUI in Fig 4-8. This means instead of performing inverse kinematics and control the end effector, we rotate the stepper motor of each individual link separately. We created a Graphical User Interface for this motion on MATLAB. The code is attached in Appendix E.



**Fig 4-8:** Graphical User Interface for each Motor Control

For each motor axis, we created an individual slider through which we control the particular axis. We select the angle by sliding the slider to a particular position, and then select the direction by clicking on the direction button. After clicking Rotate, the motor axis moves the particular angle on the axis with the angle specified on the slider.

Once we press the button, for example in this case the motor 1 button, once again the motor control and serial communication is utilized to move the end-effector from one position to another position. With this, we were able to control the motor angles and direction of rotation.

The user can adopt both methods to control the end effector of the robotic arm.

## Motor Control

One of the challenges with regards to the project was to operate multiple stepper motors using a microcontroller. This maneuver was required because the PUMA 560 operated with stepper motor on each joint. In order to have the robot achieve the required position at the end effector, it was required that each motor would rotate a specified angle simultaneously for each joint using Inverse Kinematics.

A number of options were considered to achieve this maneuver on the robot. First we considered using a separate Arduino Microcontroller to operate each motor. It was proposed that separate Arduino boards would all be connected using the Master Slave System. However, this approach was not carried forward because it was economically inefficient to achieve the operation using multiple boards.

The challenge with operating multiple stepper motors is that the Arduino Microcontroller is not capable of generating multiple pulses at the same instance. Therefore, operating multiple stepper motors generating simultaneously requires some algorithm such that simultaneous motion could be achieved. Using this approach, the robot could be operated using the same Microcontroller and therefore it is a more efficient solution. Multiple stepper motors were declared as follows:

```
AccelStepper stepper1; // Defaults to AccelStepper::FULL4WIRE (4 pins) on
2, 3, 4, 5

AccelStepper stepper2(AccelStepper::FULL4WIRE, 6, 7, 8, 9);

AccelStepper stepper3(AccelStepper::FULL4WIRE, 10, 11, 22, 24);
```

Between each HIGH and LOW pulse provided to the stepper motor to achieve its required motion, there is a specified delay; this delay determines the speed of the stepper motor rotation. During this delay, the microcontroller is idle. Therefore, during each delay which comprises of microseconds, the controller can be used to send pulses to other stepper motors achieving simultaneous movement.

Using the AccelStepper library, the Arduino Microcontroller was able to achieve the simultaneous motion of robot. This library uses the principle described previously to achieve the smooth and simultaneous motion of the robot by sending pulses between time delays to achieve simultaneous motor rotation.

The code can be found in Appendix F. In this code, each motor is commanded to rotate with a specific speed and acceleration to rotate to a specific point. The amount of rotation required by the robot was obtained by having each joint achieve a certain amount of rotation, and then calibrating to rotate each motor according to the angle computed by inverse kinematics. The

commands as given below take input from the serial port initiated in MATLAB and rotate the motors accordingly:

```
void loop() {
  if (Serial.available() > 0)
  {
    Serial.readBytes(b, 6);

    if (b[0] != 0) {          //switch on the led
      stepper1.moveTo((38000/90)*b[0]);
      stepper1.run();
    }
  }
}
```

### **Haptic Feedback Loop**

The Feedback loop for Haptic Device can be found in Appendix G. The haptic device has a paddle that is in contact with the wheel. The torque and rotation on the wheel is determined by the servo motor attached to the wheel. The feedback loop is programmed using the Arduino. The goal is to have a force applied on the end effector of the PUMA 560 robot, and replicate the same force on the Haptic Device Paddle.

The force on the end effector is registered using FSR. FSR is basically a force sensitive resistor whose resistance varies with the force applied to it. A potentiometer circuit was set up to record input from the end effector. The voltage as it varies across the end effector is recorded into the Arduino. The input into Arduino can be manipulated to give an output force on the haptic paddle. Input is taken from FSR using following commands:

```
int pressureAnalogPin = A0; //pin where our pressure pad is located.
int pressureReading; //variable for storing our reading
pressureReading = analogRead(pressureAnalogPin);
```

Using if statement and continuous loop, we allow the haptic device to freely move around when there is no force on the robot end effector. When the force applied on the FSR is 0, the loop is exited, and no torque is applied by the motor. If there is a force applied on the FSR, the motor will rotate accordingly, imposing a torque on the haptic device paddle. The While and if conditions are given by the following commands.

```
if (pressureReading > 50 ) {
  while (pressureReading > 50 ) {
```

The rotating speed of the motor can be varied via analogue input onto the motor driver. Controlling the speed would vary the torque on the wheel. The analogue input to the servo motor was calibrated such that the current provided to the motor varied accordingly to the force experienced by the end effector and hence creating a force feedback on the haptic paddle. The calibration is given as:

```
int calibrate=(pressureReading/500)*(120);
```

```
analogWrite(A1, calibrate);
```

## Serial Communication

The input is taken from the user using GUI, and computing individual joint angles to reach the final position is a task completed in MATLAB. However, the functioning of individual motors has to be done using Arduino software. This is because Arduino operates on a faster command cycle and can enable smooth motion on the motors. The problem posed is integrating the MATLAB and the Arduino Environments.

This can be done through serial communication through both the interfaces. A serial port comprises of a continuous stream of data transferred from one end to the next. A serial port is defined in MATLAB using a Baud rate of 9600 and specifying the correct Communication port. The array of 6 angles that are supposed to be rotated for each motor is then passed on to Arduino board. Serial Port is defined, and data is input to it in the code

Once the Arduino board receives the data serially, it needs to parse the data into a form in which it can be interpreted. Therefore, parsing commands are implemented such that the stream of data can be sorted, and the useful elements can be acquired and assigned to the appropriate variables. Once angle rotation for each motor is obtained, the robot can move to the specified position. The code can be seen in Appendix H.

## 4.2 Integration and Instrumentation

### 4.2.1 Transmission

To design the feedback device, we need to first design the transmission of the kinesthetic device. The purpose of the transmission is to transfer the force from the actuator to the end user. The actuator applies a torque on the transmission system. The transmission system then outputs the torque to the handle of the device which then applies the force on the end user. Based on its design, the transmission device can also be used to amplify and reduce the force felt by the user given that the torque is same from both ends.

$$\tau_{actuator} = \tau_{handle} \quad (4.1)$$

$$F_{actuator} \times r_{actuator} = F_{handle} \times r_{handle} \quad (4.2)$$

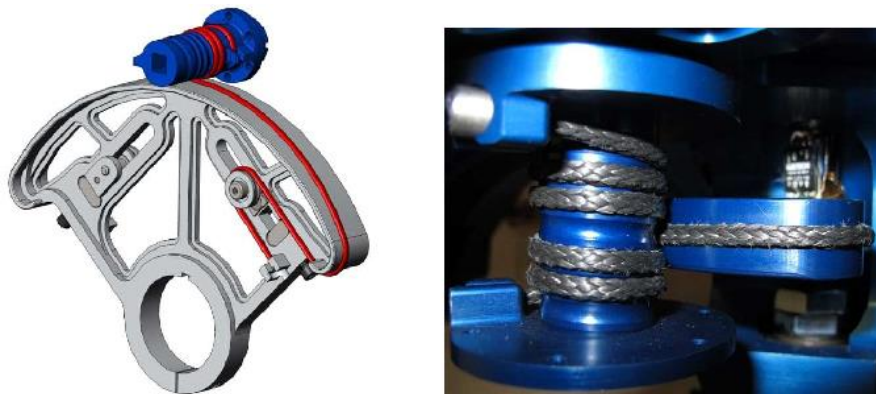
$$F_{handle} = F_{actuator} \times \frac{r_{actuator}}{r_{handle}} \quad (4.3)$$

Based on the ratio of  $\frac{r_{actuator}}{r_{handle}}$ , we can amplify and reduce the force transmitted.

There are several types of transmission systems that can be used.

## Capstan Drive

Capstan drives are rotary transmission elements widely used in robot mechanisms because of their low inertia, low backlash, high stiffness and simple implementation. The cable in a capstan drive is typically wrapped around the input and output drums in a figure-eight pattern to transmit the power.



**Fig 4-9:** Capstan Drives and Belts [22]

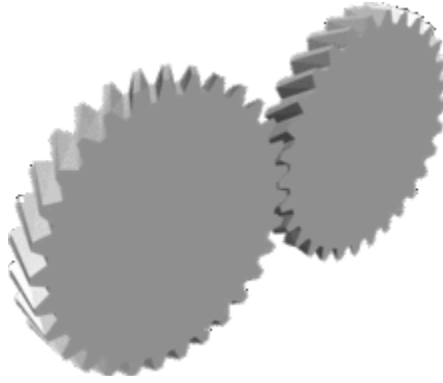
The advantages of capstan drives are that they can provide speed reduction with reduced weight, and enhanced design flexibility compared to the traditional gear system. These drives also offer low inertia, low friction and high stiffness, which can offer better impedance control. However, the main disadvantage of a capstan drive is that the life cycle of the ropes is significantly low despite their high strength. The synthetic ropes need replacing after several operations which can significantly increase the maintenance costs. [22]

## Direct drive

In this case, the motor spindle is directly attached to the output and in our case, the handle. The spindle of the motor is attached to the drive using a permanent or temporary joint. However, the parts need to be geometrically aligned with one another and a slight discrepancy can significantly reduce the efficiency of the design.

## Gear System

A gear is a circular device which has teeth engraved in the outer edges. These teeth are meshed with the teeth of another gear. Rotating one gear causes the entire gear system to rotate, transmitting torque from one end to another. In our device, we can assemble a gear train in the haptic device where the first gear is attached to the actuator which is meshed with the gears connected to the handle. There are several types of gear system which can offer numerous options in the device. For example, using helical gear instead of the spur gears can ensure a smoother operation.



**Fig 4-10:** Gear System [23]

The main advantages of using the gear system compared to other transmission system that it is relatively cheaper and has a longer life cycle compared to others. Furthermore, there are relatively no energy losses with this system, and it can easily be dismounted and replaced. The main disadvantage is that higher inertial force is required to rotate the gear system. This can cause a loss of precision in the haptic device.

### **Friction belt**

The friction drive is a transmission system which utilizes the friction force between the two wheels to transfer torque from the input to the output. The two surfaces travel in a rolling motion against one another when they come into contact with each other. These surfaces are usually covered with a high friction material such as rubber or leather.

The major disadvantage in this system is that it relies on the frictional forces for power transmission. The frictional forces can cause significant wear and tear in the friction belt which can significantly reduce the efficiency of the transmission system. Furthermore, the belt should be placed at an optimum position otherwise it can cause the friction force to be either too much or too less.

**Table 4-1:** Comparison of different transmission system options

<b>Capstan Drive</b>	<b>Gear System</b>	<b>Friction Belt</b>
Offers low friction	High friction between teeth	High friction required for driving
Offers low inertia between input & output	Significant inertia between gear teeth	Significant inertia between friction belt
Short Life Cycle	Long Life Cycle	Long Life Cycle
High Cost	Low Cost	Low Cost

After evaluating all options for the transmission system, we have decided to use the friction belt for the transmission between two wheels. The reason behind is that the friction belt

provides us with a higher resolution of rotation compared to the gear system. Furthermore, it is much more cost effective compared to other alternatives and has a long product life cycle.

#### 4.2.2 Actuator

##### **Stepper Motors**

It is an electric motor which rotates about its axis by constantly changing the polarity of the outer magnetic core. Unlike, other motors, stepper motors offer step angle revolution which ensures a precise and stable rotation. This is done by converting a series of input pulses into precise increment in shaft's rotation position. Each pulse causes the shaft to rotate a fixed angle.



**Fig 4-9:** Stepper Motor [24]

The main advantages of using the stepper motor that it can provide very high precision, and a relatively large torque compared to the DC motor. Furthermore, it is very reliable since it has no contact brushes in the motor. It also does not require any position feedback. The disadvantages of using the stepper motor is that it requires a significant amount of power to run. The torque also drops off at relatively high speed.

##### **DC Motors**

A Direct Current Motor is an electrical motor which has a stationary set of magnets in the stator, and an armature. The coil of wire generates an electromagnetic field aligned with the center of the coil. The intensity and direction of the magnetic field can be controlled using magnitude of the current flowing through the coil. The current is transmitted through brushes to armature, and is constantly switched so that the armature magnetic field remains fixed.





**Fig 4-10: DC Motors [25]**

The advantage of a DC motor is that it can provide with precise control over the speed and torque of the motor. Furthermore, it has a relatively simple design hence it is a cheaper option compared to other alternatives. However, it requires constant maintenance due to carbon brushes, and it is highly vulnerable to dust and other influences of the environment.

#### **Servo Motor.**

A servo motor is a rotary actuator which ensures very precise control of rotary position. This actuator consists of a sensor position feedback where sends a feedback to the user about the servo's position and speed.



**Fig 4-11: Servo Motors [26]**

The main advantages of using a servo motor are that it provides a feedback to the control system. This can allow the servo motor to reset and recalibrate its position after it has completed its operation. It also requires low torque to overcome its inertial torque, and can be used under low current and voltage too. However, it has lower responsiveness compared to stepper and DC motor.

**Table 4-2:** Comparison of different actuator options

Stepper Motor	DC Motor	Servo Motor
Can provide large torque	Only provides low-medium torque	Only provides low torque
High precision with position and speed	High precision with speed	High precision with position and speed
High holding torque	Low holding torque	Low holding torque
No position feedback	No position feedback	Position feedback
High electrical power required	Low power required	Low power required
High cost	Low cost	High cost

After considering several options, for our haptic design we will be using the servo motor to drive the position feedback system. For our project, we require high precision over speed and low holding torques, which are provided by servo motors.

#### 4.2.3 Sensors

We also need to install sensors in our kinesthetic device to detect the rotational speed of the motor spindle, as well as the angular position of the mechanism. The rotational speed of the device can be used to detect the torque applied on the end user. Hence, we would study various sensors suitable for the haptic device.

##### **Incremental Rotary Encoder:**

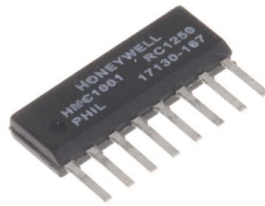
An incremental encoder converts rotary motion of a shaft into an analog or digital output that represents the position of the shaft or its motion. They convert mechanical displacements to electrical signals and then process these signals to an output form. It can also be used for determining the motor speed. The OMRON E6B2 uses optical sensors to detect rotation and compute it into processable signal.

**Fig 4-12:** OMRON Incremental Encoder [27]

The main advantage of incremental encoders is that they offer a high precision and resolution, but they are costly and thus not suitable in case of budget constraints.

### **Magneto-resistive angle sensors**

Magneto-resistive sensors change their electrical resistance when an external magnetic field is applied. The resistance increases or decreases depending on the orientation of the field lines about the direction of current flow. The magneto-resistive sensors can be integrated within the Wheatstone bridge circuit which detects the small fluctuations in the resistance, and outputs a readable voltage.



**Fig 4-13:** Magneto-resistive angle sensor [28]

The main advantage of these sensors is that they operate under very little electrical power. Since no electrical contact is required, they can operate under a large air gap too. The magneto-resistive effect is particularly attractive for the use in harsh environments. Furthermore, they can also operate in extreme temperature ranges, extreme pressure and external force. However, the major drawback that arises with Magneto-resistive angle sensors is that they have reduced accuracy compared to other devices.

### **Hall-Effect Sensors**

A Hall Effect Sensor is a sensor which is used to measure the intensity of the magnetic field. The magnitude of the voltage generated by the Hall Effect Sensor is directly proportional to the intensity of the magnetic field passing through it. The Hall Effect Sensors can then be calibrated to give either linear or digital output. In our device, since we are measuring the rotational speed of the input shaft, we are going to use the Hall Effect Sensor with the digital output.



**Fig 4-14:** Hall-Effect Sensor [29]

The main advantage of Hall-Effect Sensor is that it is not affected by the ambient conditions, such as dust, humidity, and mechanical vibration. They also do not have contact with moving mechanical parts, making the sensors durable, and ensures a long product life. The major disadvantage of the Hall-Effect Sensor is that it is not capable of measuring a current flow at a distance greater than 10 cm. Furthermore, a significant rise in temperature can affect the electrical resistance of the sensor, resulting in a reduced sensitivity.

### **Potentiometer**

A potentiometer is a variable resistor with a sliding contact with the belt of the high resistance material. A potentiometer can act as a voltage divider where the resistance between the terminals varies with the contact in forces. The potentiometer is then attached to a shaft which rotates along the resistance belt, and hence the resistance of the potentiometer varies based on the position of the shaft. The potentiometer can be configured to detect the position of the shaft and the input velocity.



**Fig 4-15:** Potentiometers [30]

The major advantage of using a potentiometer is that it is cheap and easy to use. Furthermore, it has an infinite resolution and can give a highly accurate reading. It is extremely useful when measuring large displacement and can give a significant output with no requirement of the voltage amplification. The major disadvantage is that it requires a large torque to overcome the

friction and inertia in the potentiometer. Furthermore, it is subjected to wear and tear which can significantly reduce the product life cycle.

**Table 4-3:** Comparison of the Sensor Options

<b>Incremental Encoders</b>	<b>Magnetoresistive sensors</b>	<b>Hall-Effect Sensors</b>	<b>Potentiometer</b>
High precision	Low precision	Low precision	High precision
High resolution	Low resolution	Low resolution	High resolution
Sensitive to harsh environment	Can operate under harsh environment	Can operate under harsh environment	Can operate under harsh environment
Low torque required	Low torque required	Low torque required	High torque required
High product life	High product life	High product life	Low product life
High-cost device	Low-cost device	Low-cost device	Low-cost device

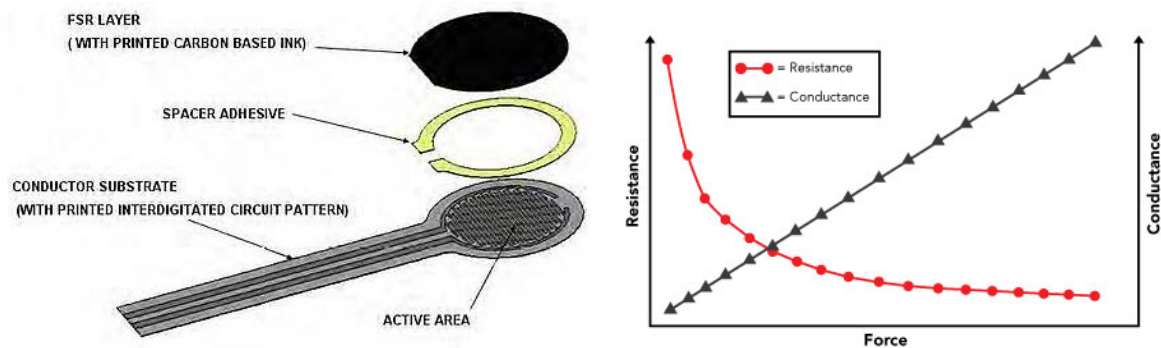
Finally, the sensor we will be using in our haptic device is the incremental rotary encoder. Although, they are costly device compared to other alternatives, they provide us with the necessary precision required for the project.

### 4.3 Testing/Experimental Procedures

#### 4.3.1 Signal Processing

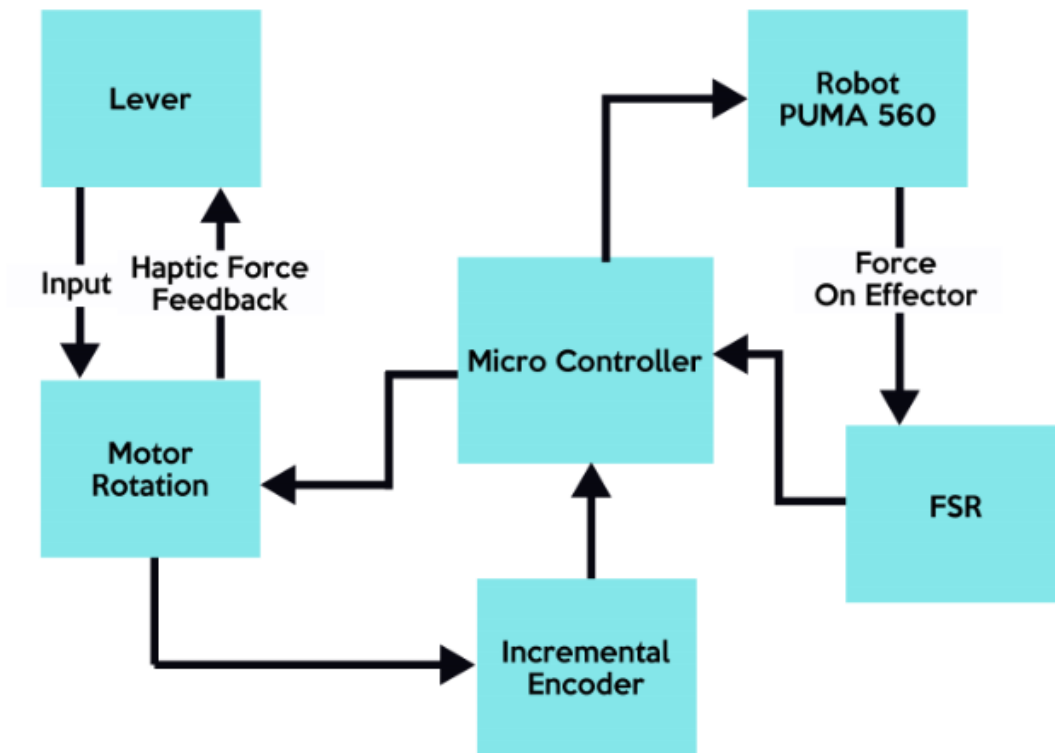
The purpose of joystick is to provide uni-axial robot control, and to provide force feedback to the user in that direction. The movement of robot should be synchronous to the movement of lever arm for proper working of the mechanism. To capture to movement of lever arm, we need to devise a mechanism to precise measure its rotation. We will attach an incremental encoder next to the servo motor to compute its rotation and provide live input to the robot through a micro controller. The input from the encoder can be calibrated to the robot's movement depending on the required sensitivity and range of joystick's movement. We can also use encoded DC motors with built-in encoders for this purpose which will make our model more compact. This however depends on availability of high sensitivity encoded motors in the market as they are difficult to acquire. This is how we control robot through an indirect connection with the motor.

The feedback, however, would be directly provided by the motor. The force sensing resistor FSR will be attached to the robot's end-effector. This sensor will directly compute the force acting on it as the decrease their electrical resistance when force or pressure directly acts on it. The typical flexible FSR is shown in Fig 4-18, which consists of three layers. As more pressure or force is applied to it, the contact force of the conductive material with wires increases, hence more electricity flows through them, allowing us to detect changes in force due to change in electrical flow. This relation is also shown on the graph in Fig 4-18.



**Fig 4-18:** Force Sensing Resistor build and change in its properties due to force [31] [32]

The force sensor upon detecting force will communicate data to the micro controller which will be connected to the motor attached mechanically to the lever of the joystick. The motor will be calibrated according to the amount of force sensitivity required and increase its holding torque accordingly. Since holding torque opposes motion, motion of lever arm, which was free to move earlier, is restricted. This provides us the necessary haptic or force feedback. The whole process is shown on the block diagram in Fig 4-19.



**Fig 4-19:** Signal Flow for forward and feedback loop

#### 4.4 Summary

The MATLAB code for robot control employing a graphical user interface and forward/inverse kinematics and motor control is explained. The different mechanisms of force/torque transmission at critical locations such as the pulley are evaluated in this section. Different sensors and actuators are considered for incorporation into the device based on the criteria best suited for a haptic feedback application. The Signal Processing procedure of the haptic device is explained as well.

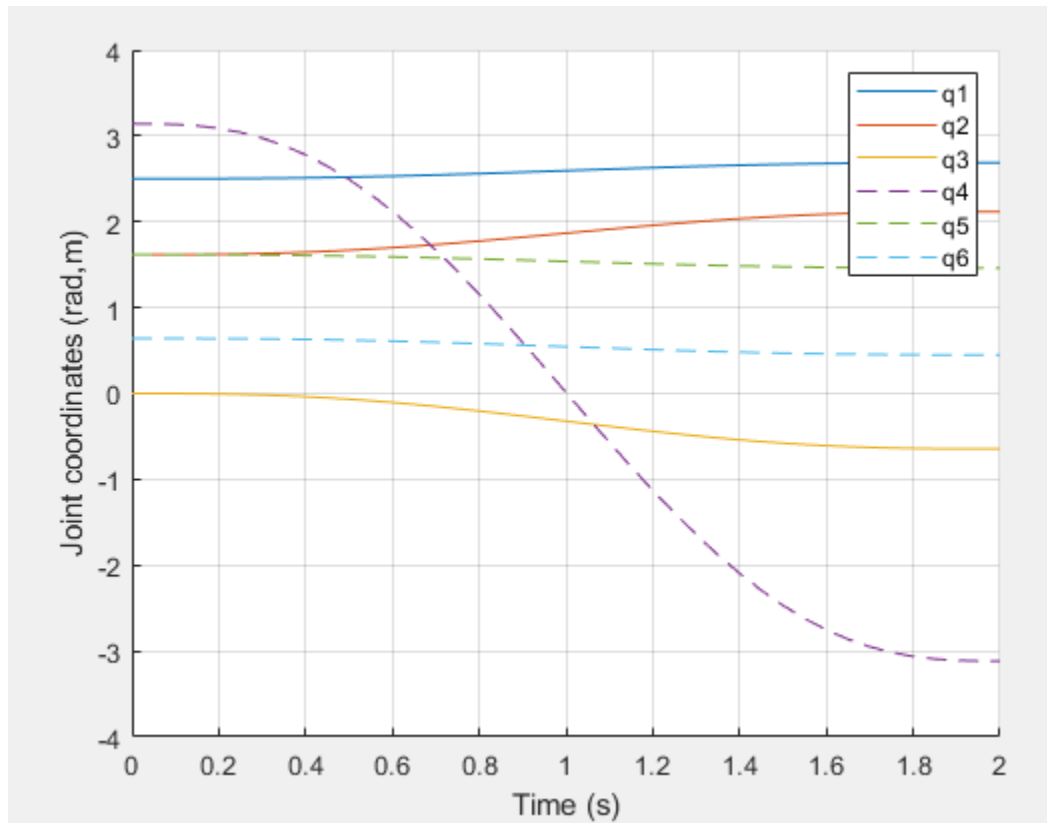
## Chapter 5 Results

### 5.1 Results

#### 5.1.1 Robot Control

For robot control, the input regarding the position of the end effector is taken from the user. The GUI serves as an interface into which this final position is fed. The individual joint rotations are calculated. Using an Arduino, each stepper motor is rotated through the required angles. Rotating the stepper motor of each joint alters the position of the PUMA 560 robot, and as an output the desired end effector position is obtained.

Fig 5-1 shows the joint angles with respect to time for a straight-line trajectory for the robot moving 20 cm in positive x-axis from home position. The joint motion is computed using quintic (fifth-order polynomial) time scaling. This is the trajectory simulation based on our mathematical calculations.



**Fig 5-1:** Joint angles for PUMA 560 moving 20 cm in positive x direction.



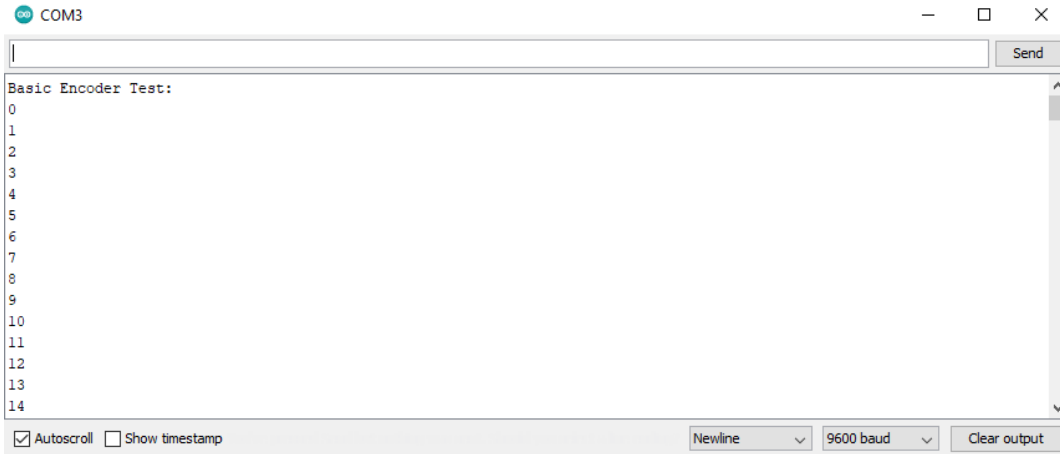
### 5.1.2 Haptic Controller Input

The wheel of the haptic device accounts for the rotary movement of the lever. The idea is to measure rotation and calibrate to into the linear motion of the robot. To measure the pure rotary movement of the wheel, we used an incremental encoder. The rotary incremental encoder is attached to the wheel through a shaft and a coupling. The other end of the coupling is attached to encoder shaft therefore when wheel rotates due to level motion, the encoder shaft also rotates as a result.



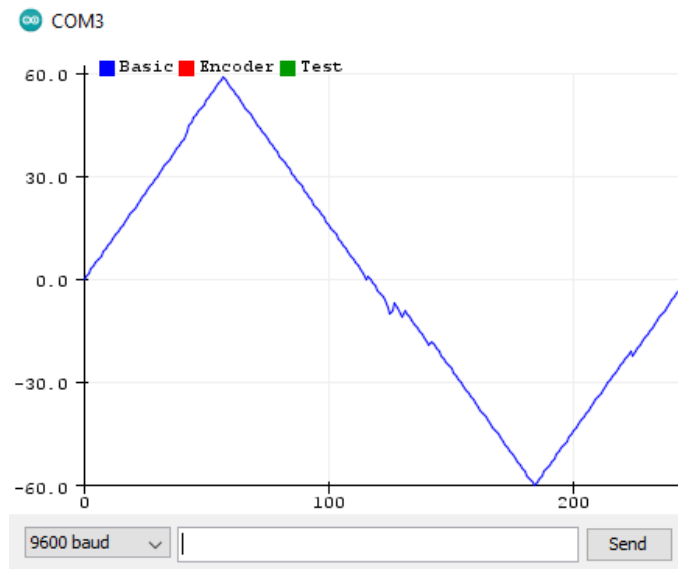
**Fig 5-2:** OMRON E6B2 coupled with a screw shaft.

This rotation is computed by OMRON E6B2 cwz6c incremental encoder shown in Fig 5-2, which attached to an Arduino. The serial monitor displays the rotation of the encoder and we can see in Fig 5-3 that our encoder readings are accurate to the nearest degree. We used Encoder library in Arduino to get input readings; we can send these readings to MATLAB through serial communication which will convert this input to a linear distance through a calibration formula. MATLAB will then perform inverse kinematics to get the necessary joint angles to move the robot in the desired direction. The Arduino code for getting encoder input is shown in the Appendix I.



**Fig 5-3:** Encoder Readings displayed on Serial Monitor

The plot in Fig 5-4, shows the rotation of the wheel in 60 degrees clockwise direction and then 120 degrees in anticlockwise then 60 degrees again clockwise back to initial position. We used serial plotter to show the instantaneous rotation of the wheel. Since the rotation measurement is instantaneous, the motion of the robotic arm would be instantaneous contributing to the accuracy of the system.

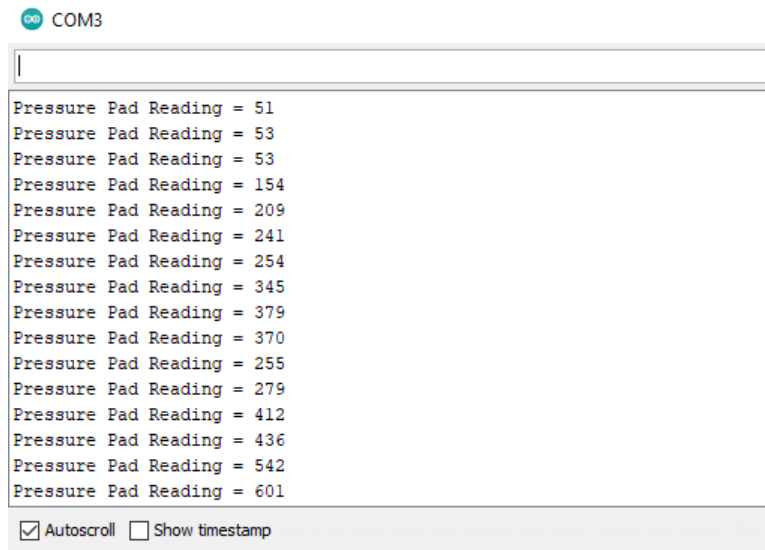


**Fig 5-4:** Instantaneous Encoder Readings displayed on Serial Plotter.

### 5.1.3 Haptic Feedback Mechanism

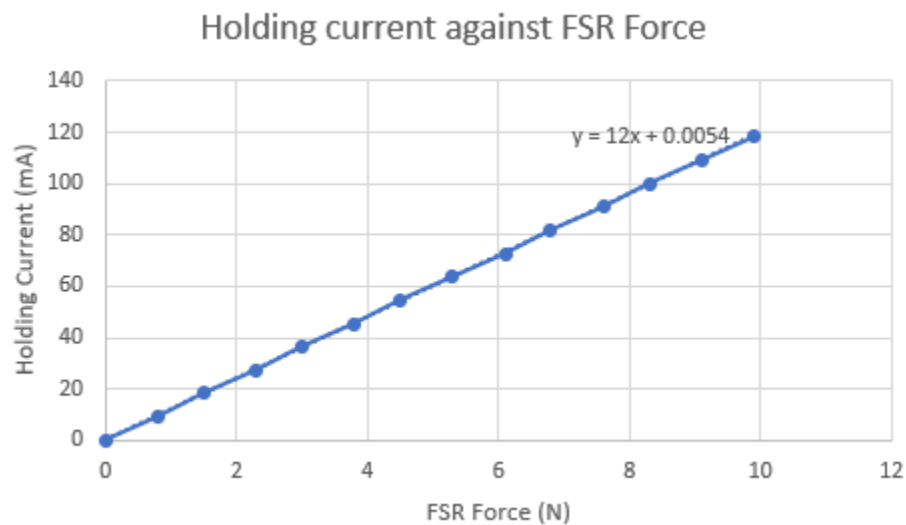
We can see the instantaneous FSR readings on the serial monitor shown in Fig 5-5. The value of input increase with the increase in force of FSR, we can calibrate thus force reading to motor current. The motor we are using needs more than 130 pwm readings to move, we calibrated FSR readings into a linear trend ranging from 0-120 pwm; the motor will not rotate but current

is being provided to it making it harder to move as current provides holding torque. This will make lever harder to move, hence providing the desired haptic feedback.



**Fig 5-5:** FSR readings displayed on Serial Monitor.

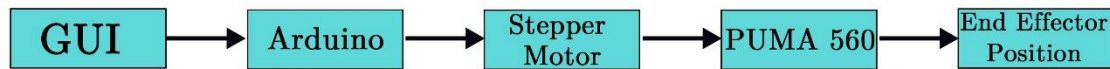
The linear trend of FSR readings and resulting motor holding current is determined by a calibration formula and is shown in Fig 5-6. The plot shows FSR readings on horizontal axis and resulting motor current on vertical axis. Since maximum force FSR can detect is 10 Newton therefore we calibrated analogue readings on the sensor accordingly. The significance of the plot is that it shows how change in force sensor readings affects holding current of the motor. This shows that as the force on the sensor increases, the haptic feedback of the device increases.



**Fig 5-6:** Plot of force readings on FSR readings against Motor holding current.

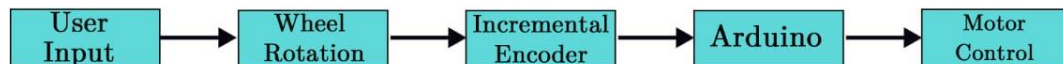
## 5.2 Analysis and Discussions

The motor control can be summarized into the following block diagram in Fig 5-7. The signal flow required to rotate motors and orientate robot at a specific configuration is complete. We can move our robot at any position within its working space. The maximum speed and acceleration of joint motion can also be controlled, allowing formation of trajectories.



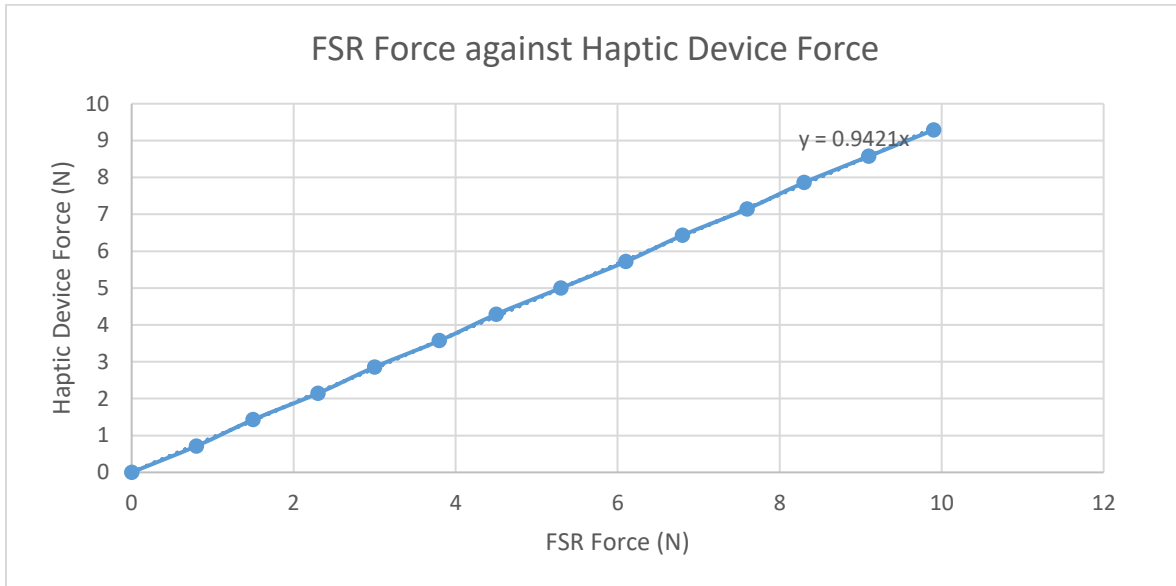
**Fig 5-7:** Motor Control Block Diagram

The haptic device takes input directly from the user. The user moves the paddle, hoping to achieve an end effector's motion in desired axis. The paddle rotates wheel which is connected to a rotary incremental encoder through a shaft. The encoder, connected to an Arduino, measures rotation and sends it as an input to MATLAB. The MATLAB code is calibrated to convert rotation to linear displacement. The linear displacement will then be converted to rotary motion of robot's joints through inverse kinematics hence we have inputs for the execution of motor control loop which will move robot's end effector in the specified direction. The block diagram in Fig 5-8 shows the complete process.



**Fig 5-8:** Controller Input Block Diagram

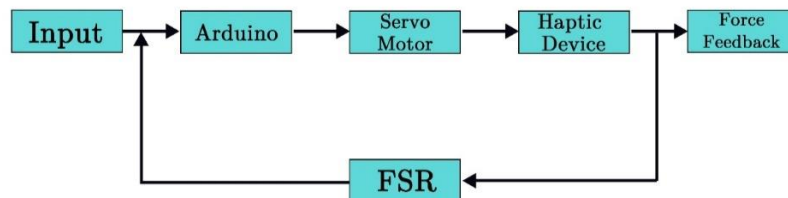
The graph shown in Fig 5-9 between the force felt on the FSR against the force experienced on the used by the haptic device. This is to determine the precision of the force transmitted to the user through the haptic device.



**Fig 5-9:** Plot of force reading on FSR reading against feedback force.

As seen from the graph, the gradient of the trend line between the FSR force against the feedback force is 0.9421. This is almost close to the gradient to one hence ensuring the force felt on the FSR is almost equal to the force transmitted to the Haptic Device handle. The error can be accounted for by having precise control over the motor constant of the servo motor attached.

The feedback mechanism summarizes into the following. The user inputs the required position of the end effector through haptic controller in the desired direction. Once the input is computed using forward loop, the robot moves and when a force is experienced by the end effector, the Force Sensitive Resistor (FSR) transmits the magnitude of the force experienced to the Microcontroller. The microcontroller commands the motor to operate at a specific holding torque whose intensity is based on the force experienced. Using this torque, a force feedback can be experienced at the haptic paddle. The process is shown in Fig 5-10.

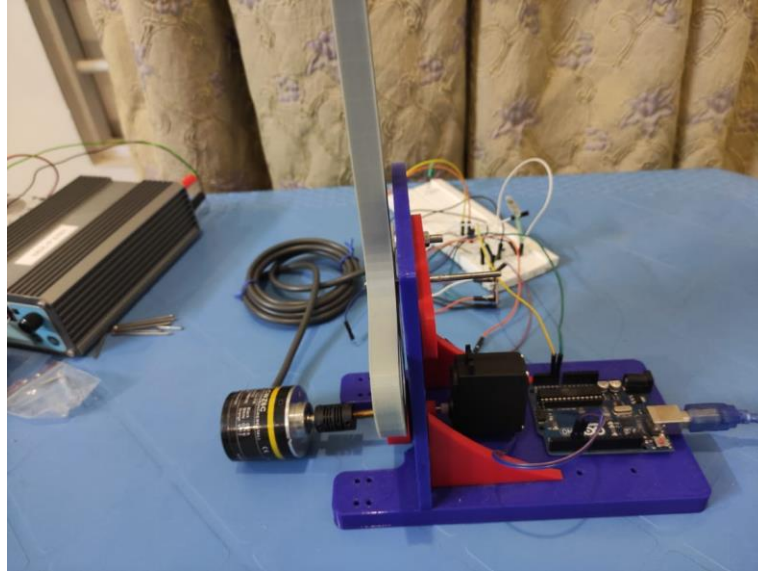


**Fig 5-10:** Feedback Loop Block Diagram

## 5.3 Summary

### 5.3.1 Complete Uni-directional Haptic Device Prototype

The prototype haptic device can be seen in Fig 5-11.



**Fig 5-11:** Haptic Feedback Device (Side View)

As can be seen in the Fig, the device components are supported on top of the horizontal base plate and vertical plate. The vertical plate is attached to the horizontal plate with the help of supports. The main component of the device is the haptic device paddle, this is the paddle where the user will apply a force to control the motion of the robot in desired axis and experience a haptic force feedback. The paddle is in contact with the wheel upon which both the encoder and the servo are mounted.

As a force is applied by the user on the paddle, the wheel in contact is rotated. The angle of rotation is recorded by the encoder and the input from the encoder is taken by the Arduino in order to move the robot in Z axis. Once the end effector experiences a force in the Z direction the input from sensor is input to the Arduino and the servo mounted on the wheel applies a torque on the wheel which is transmitted to the paddle and is finally experienced by the user who is operating the controller. The Fig 5-12 shows isometric view of the prototype.



**Fig 5-12:** Haptic Feedback Device (Isometric View)

## Chapter 6

### Impact and Economic Analysis

#### 6.1 Social Impact

The implementation of robot assisted surgery has always been a challenge for Pakistan due to the limited resources available in the country. Despite the advancements being made in this field at a global level, Pakistan only has one center that performs robot assisted surgery. This is because robotic platforms require a significantly high capital investment, along with high maintenance costs, and Pakistan lacks the funds, and the expertise for it.

The benefits of commercializing robotic tele-surgery with haptic feedback can be highlighted in Table 6-1.

**Table 6-0-1: Benefits of Commercializing robotic tele-surgery**

Benefit to the Hospital	Benefit to the Surgeon	Benefit to the Patient
More patients catered as the recovery time is lesser	Improved tissue manipulation	Smaller incisions
Positive reputation due to precision in surgeries	Reduced breaking of sutures	Lesser pain
Funding by the government to support	Cater to more patients as remote operations possible	Quick recovery
	Avoid risk of infection	Treated well even if the disease is contagious

#### 6.2 Environmental impact

The project does not have a significantly negative impact on the environment; however, the components of the haptic device need to be 3D printed, and 3D printers have a high energy consumption compared to CNC, and other machining tools. Since Pakistan mainly uses fossil fuels for electricity generation, the high-power consumption during 3D printing is linked to emission of greenhouse gases during the burning of fossil fuels, which causes air pollution, global warming and environmental degradation.



### 6.3 Sustainable development goals (SDGs)

Our project is in compliance with the Sustainable Development Goals by the UN as mentioned in Table 6-2 below:


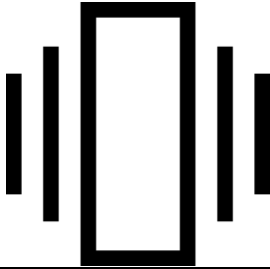
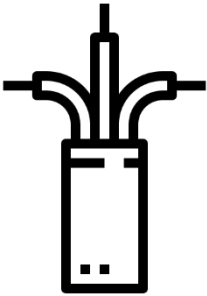

**Table 6-0-2: Sustainable Development Goals**

S. No	SDG	Adherence of FYP to SDG
1.	No Poverty	
2.	Zero Hunger	
3.	Good Health and Well-Being	The project would improve the surgical procedures by leading to a greater precision during surgeries, thereby reducing the pain experienced by patients, and facilitate speedy recoveries.
4.	Quality Education	
5.	Gender Equality	
6.	Clean Water and Sanitation	
7.	Affordable and Clean Energy	
8.	Decent Work and Economic Growth	
9.	Industry, Innovation and Infrastructure	Innovation would be fostered by the project because even the latest Da Vinci surgical systems lack the sense of touch. This project would provide haptic feedback during surgeries which would increase the feeling of tele-presence among the surgeons and make the surgical procedure more precise.
10.	Reduced Inequalities	
11.	Sustainable Cities and Communities	
12.	Responsible Consumption and Production	
13.	Climate Action	
14.	Life Below Water	
15.	Life on Land	
16.	Peace, Justice and Strong Institutions	
17.	Partnerships	Partnerships with the government can be established to provide funding to facilitate a commercialize model of this concept.

## 6.4 Hazard Identification and Safety Measures

Table 6-3 below identifies the potential hazards associated with the equipment, along with the safety measures to prevent those hazards.

**Table 6-3:** Potential Hazards and Safety Measures

Symbol	Hazard	Safety Measures
	<p><b>Water Damage</b></p> <p>Exposure of the electrical components to water can damage the equipment and lead to its failure, as well as causing a shock hazard.</p>	<p>Make sure that there are no fluids near the surgical equipment.</p>
	<p><b>Mechanical Vibrations</b></p> <p>The haptic device may suffer from vibrations due to the motion of the servo motor mounted on it. This vibration may cause mechanical failures.</p>	<p>The motor should be tightly mounted on the haptic device, and shock absorbers should be used.</p>
	<p><b>Short Circuiting</b></p> <p>Exposure to bare windings can lead to electrocution.</p>	<p>All the wires should be properly insulated, and repaired immediately if damaged.</p>
	<p><b>Overheating</b></p> <p>If exceptionally high currents are supplied to the servo and stepper motors, the motors may overheat and suffer from damage.</p>	<p>The current supplied to the motors shall be lower than their maximum current rating, and the motors should be replaced periodically after extended periods of use.</p>

## **6.5 Summary**

This chapter summarizes the social benefits of robotic tele-surgery with haptic feedback as well as the economic and environmental impact. The hazards associated with the setup are identified, and safety measures proposed. The concept of haptics is innovative and can prove to be extremely beneficial in the robotic surgery domain if its maximum capacity is exploited, which requires intense research and abundant financial resources.

## Chapter 7

### Conclusion and Future Recommendation

#### 7.1 Summary

The aim of this project was to build a haptic feedback controller capable of remotely controlling the PUMA 560 robot. The primary objectives were to have the user of the haptic device control the robot and experience the same uniaxial force as experienced by the robot end effector. Further objectives include the control of the PUMA 560 using GUI and controller from a remote location.

The haptic feedback device was 3D printed with the reference of the cad model and was operated using servo motors. The PUMA 560 robot was operated using inverse kinematics computed using MATLAB, and the Arduino microcontroller was used to obtain the required motion from the robot and the force feedback on the haptic controller.

The group was able to integrate the MATLAB and Arduino IDEs to control the position of the end effector of the PUMA 560 robot. Furthermore, Image recognition was enabled, and the group was able to send commands remotely in order to establish remote control of the system. This remote control was achieved by means of a GUI and the haptic feedback control which also produces a force feedback on the controller.

The major causes of the problem arose with the control of the PUMA 560 robot and synchronizing the movements of different joints. This was solved by operating different stepper motors using the same microcontroller. Furthermore, due to lack of access to the lab owing to COVID-19 restrictions, the final synchronized model could not be established for the project forward loop, however the individual components were made operational.

#### 7.2 Recommendations:

The Haptic feedback device can be improved by adding force feedback in more axis. A uniaxial force feedback was established in the current project. The controller can be improved by adding force feedback along all 6 degrees of motion. The controller should be able to manipulate the end effector in all 6 Degree of Freedom and be able to experience force feedback in all 6 directions. This will greatly aid tele-surgery operations.

Image Recognition using deep learning algorithms can be implemented to help the surgeon performing tele-surgery by recognizing different organs and operation sights displayed on the

camera. This will help the surgeon with viewing the tele-surgery operation being performed by giving visual aid.

The PUMA 560 Device is fairly traditional, and it is not advisable to perform tele-surgery using such a robot. Therefore, haptic device needs to be integrated with robots having better accuracy and precision in order to safely perform the tele-surgery operation, because such operations require a high accuracy and precision in order to function well.

## References

- [1] Okamura, A. M. (2009). *Haptic feedback in robot-assisted minimally invasive surgery. Current Opinion in Urology*, 19(1), 102–107. doi:10.1097/mou.0b013e32831a478c
- [2] Peter Kyberd. (1992). *Technology: Robodoc carves a place in medical history*
- [3] Dey, U., & Cheruvu S., K. (2020). *A web-based integrated GUI for 3D modeling, kinematic study, and control of robotic manipulators. Computer Applications in Engineering Education*. doi:10.1002/cae.22282
- [4] Luis M. Bergasa, Joshua Puerta. (2018), *Haptics, Handbook of Robotic and Image-Guided Surgery*
- [5] C. Parthiban and M. R. Zinn, "A simplified approach to admittance-type haptic device impedance evaluation," *2014 IEEE Haptics Symposium (HAPTICS)*, Houston, TX, 2014, pp. 587-590, doi: 10.1109/HAPTICS.2014.6775521.
- [6] M. Medina, A. Benitez, I. Huitzil, J. De La Calleja and A. Casiano-Ramos, "A 3D simulation environment for kinematic task of the PUMA 560 robot," in Electronics, Communications, and Computers, International Conference on, San Andres Cholula, Puebla, Mexico, 2011
- [7] A.M. Okamura, (2009)," Haptics for Robot-Assisted Minimally Invasive Surgery, Robotics Research, 2011, Volume 66, ISBN : 978-3-642-14742-5
- [8] Dey, U., & Cheruvu S., K. (2020). A web-based integrated GUI for 3D modeling, kinematic study, and control of robotic manipulators. *Computer Applications in Engineering Education*. doi:10.1002/cae.22282
- [9] MacFarlane, M., Rosen, J., Hannaford, B., Pellegrini, C., Sinanan, M.: Force feedback grasper helps restore the sense of touch in minimally invasive surgery. *Journal of Gastrointestinal Surgery* 3(3), 278–285 (1999)
- [10] Deml, B., Ortmaier, T., Seibold, U.: The touch and feel in minimally invasive surgery. In: IEEE International Workshop on Haptic Audio Visual Environments and their Applications, pp. 33–38 (2005)
- [11] Wagner, C.R., Stylopoulos, N., Jackson, P.G., Howe, R.D.: The benefit of force feedback in surgery: Examination of blunt dissection. *Presence: Teleoperators and Virtual Environments* 16(3), 252–262 (2007)
- [12] Tzemanaki, A., Al, G. A., Melhuish, C., & Dogramadzi, S. (2018). Design of a Wearable Fingertip Haptic Device for Remote Palpation: Characterisation and Interface with a Virtual Environment. *Frontiers in Robotics and AI*, 5. doi:10.3389/frobt.2018.00062
- [13] Rezeck, P., Frade, B., Soares, J., Pinto, L., Cadar, F., Azpurua, H., ... M. Campos, M. F. (2018). *Framework for Haptic Teleoperation of a Remote Robotic Arm Device. 2018 Latin American Robotic Symposium, 2018 Brazilian Symposium on Robotics (SBR) and 2018 Workshop on Robotics in Education (WRE)*. doi:10.1109/lars/sbr/wre.2018.00039
- [14] Ju, Z., Yang, C., Li, Z., Cheng, L., & Ma, H. (2014). *Teleoperation of humanoid baxter robot using haptic feedback. 2014 International Conference on Multisensor Fusion and Information Integration for Intelligent Systems (MFI)*.
- [15] Aggarwal, Luv & Aggarwal, Kush & Urbanic, Ruth Jill. (2014). *Use of Artificial Neural Networks for the Development of an Inverse Kinematic Solution and Visual Identification of Singularity Zone(s)*. *Procedia* 107.
- [16] Kevin M. Lynch and Frank C. Park. 2017. *Modern Robotics: Mechanics, Planning, and Control (1st. ed.)*. Cambridge University Press, USA.

- [17] Benitez Ruiz, Antonio & Huitzil, Ignacio & Casiano, Azgad & Medina, María & de la Calleja, Jorge. (2011). *A 3D simulation environment for kinematic task of the PUMA 560 robot*. CONIELECOMP 2011 - 21st International Conference on Electronics Communications and Computers, Proceedings.
- [18] John J.Craig, *Introduction to Robotics: Mechanics and Control, Third Edition*. Pearson Education, 2005
- [19]
- [20] Allison Okamura. *Design and Control of Haptic Systems*, Stanford University, 2020
- [21] Peter Corke, *Robotics, Vision and Control Fundamental Algorithms in MATLAB®*, 2017, Springer Tracts in Advanced Robotics
- [22] A. Mazumdar et al., *Synthetic Fiber Capstan Drives for Highly Efficient, Torque Controlled, Robotic Applications*, in IEEE Robotics and Automation Letters
- [23] Sader, *3D animation representing helicoidal gears*.
- [24] <https://www.ato.com/nema-17-stepper-motor-6v-0-8a-1-8-degree-2-phase-4-wires>
- [25] <https://www.indiamart.com/proddetail/dc-motor-2120393155.html>
- [26] <https://www.jsumo.com/mg996r-servo-motor-digital>
- [27] <https://instrumentationtools.com/encoder-working-principle/>
- [28] <https://no.rs-online.com/web/p/hall-effect-sensor-ics/1698128/>
- [29] <https://www.addicore.com/SS49E-Linear-Hall-Sensor-p/ad316.html>
- [30] <https://www.sparkfun.com/products/14624>
- [31] <https://www.interlinkelectronics.com/force-sensing-resistor>
- [32] <https://www.tekscan.com/blog/flexiforce/how-does-force-sensing-resistor-fsr-work>

## Appendices

### Appendix A: Forward Kinematics Algorithm

```

1 -   clc
2 -   a2=0.4318;
3 -   a3=0.0203;
4 -   d2=0;
5 -   d3=0.15005;
6 -   d4=0.4318;
7 -   d6=0;
8
9 -   M = [1 0 0 a2+a3;
10         0 1 0 d2-d3;
11         0 0 1 d4+d6;
12         0 0 0 1];
13
14 -   Blist = [0 0 0 0 0 0;
15             0 -1 -1 0 -1 0;
16             1 0 0 1 0 1;
17             d3-d2 -(d4+d6) -(d4+d6) 0 d6 0;
18             a2+a3 0 0 0 0 0;
19             0 (a3+a2) 0 0 0 0];
20 -   t1 = input('input joint angle 1 :');
21 -   t2 = input('input joint angle 2 :');
22 -   t3 = input('input joint angle 3 :');
23 -   t4 = input('input joint angle 4 :');
24 -   t5 = input('input joint angle 5 :');
25 -   t6 = input('input joint angle 6 :');
26 -   thetalist = [t1; t2; t3; t4; t5; t6];
27
28 -   Tsd=FKinBody(M,Blist,thetalist)

30 -   function T = FKinBody(M, Blist, thetalist)
31 -       T = M;
32 -       for i = 1: size(thetalist)
33 -           T = T * MatrixExp6(VecTose3(Blist(:, i) * thetalist(i)));
34 -       end
35 -   end

```



```

37 - function T = MatrixExp6(se3mat)
38 -     omgtheta = so3ToVec(se3mat(1: 3, 1: 3));
39 -     if NearZero(norm(omgtheta))
40 -         T = [eye(3), se3mat(1: 3, 4); 0, 0, 0, 1];
41 -     else
42 -         [omghat, theta] = AxisAng3(omgtheta);
43 -         omgmat = se3mat(1: 3, 1: 3) / theta;
44 -         T = [MatrixExp3(se3mat(1: 3, 1: 3)), ...
45 -             (eye(3) * theta + (1 - cos(theta)) * omgmat ...
46 -             + (theta - sin(theta)) * omgmat * omgmat) ...
47 -             * se3mat(1: 3, 4) / theta;
48 -             0, 0, 0, 1];
49 -     end
50 - end
51
52 - function R = MatrixExp3(so3mat)
53 -     omgtheta = so3ToVec(so3mat);
54 -     if NearZero(norm(omgtheta))
55 -         R = eye(3);
56 -     else
57 -         [omghat, theta] = AxisAng3(omgtheta);
58 -         omgmat = so3mat / theta;
59 -         R = eye(3) + sin(theta) * omgmat + (1 - cos(theta)) * omgmat * omgmat;
60 -     end
61 - end
62
63 - function omg = so3ToVec(so3mat)
64 -     omg = [so3mat(3, 2); so3mat(1, 3); so3mat(2, 1)];
65 - end
66
67 - function se3mat = VecTose3(V)
68 -     se3mat = [VecToso3(V(1: 3)), V(4: 6); 0, 0, 0, 0];
69 - end
70
71 - function so3mat = VecToso3(omg)
72 -     so3mat = [0, -omg(3), omg(2); omg(3), 0, -omg(1); -omg(2), omg(1), 0];
73 - end
74
75 - function judge = NearZero(near)
76 -     judge = norm(near) < 1e-6;
77 - end
78
79 - function [omghat, theta] = AxisAng3(expc3)
80 -     theta = norm(expc3);
81 -     omghat = expc3 / theta;
82 - end

```

## Appendix B: Inverse Kinematics Iterative Algorithm:

```

1 -   clc
2 -   a2=0.4318;
3 -   a3=0.0203;
4 -   d2=0;
5 -   d3=0.15005;
6 -   d4=0.4318;
7 -   d6=0;
8
9 -   M = [1 0 0 a2+a3;
10         0 1 0 d2-d3;
11         0 0 1 d4+d6;
12         0 0 0 1];
13
14 -   Blist = [0 0 0 0 0 0;
15             0 -1 -1 0 -1 0;
16             1 0 0 1 0 1;
17             d3-d2 -(d4+d6) -(d4+d6) 0 d6 0;
18             a2+a3 0 0 0 0 0;
19             0 (a3+a2) 0 0 0 0];
20
21 -   Tsd=[0 1 0 -0.5;
22         0 0 -1 0.1;
23         -1 0 0 0.1;
24         0 0 0 1];
25
26 -   Theta = [0.2;0.3;0.1;0;.1;0.1];
27 -   ew=0.001;
28 -   ev=0.0001;
29
30 -   [Theta d,success] = IKinBody(Blist,M,Tsd,Theta,ew,ev);

```

---

```

33 -   function [thetalist, success] = IKinBody(Blist, M, T, thetalist0, eomg, ev)
34 -   thetalist = thetalist0;
35 -   i = 0;
36 -   maxiterations = 20;
37 -   Vb = se3ToVec(MatrixLog6(TransInv(FKinBody(M, Blist, thetalist)) * T));
38 -   err = norm(Vb(1: 3)) > eomg || norm(Vb(4: 6)) > ev;
39 -   while err && i < maxiterations
40 -       thetalist = thetalist + pinv(JacobianBody(Blist, thetalist)) * Vb;
41 -       i = i + 1;
42 -       Vb = se3ToVec(MatrixLog6(TransInv(FKinBody(M, Blist, thetalist)) * T));
43 -       err = norm(Vb(1: 3)) > eomg || norm(Vb(4: 6)) > ev;
44 -   end
45 -   success = ~ err;
46 -   end
47
48 -   function V = se3ToVec(se3mat)
49 -   V = [se3mat(3, 2); se3mat(1, 3); se3mat(2, 1); se3mat(1: 3, 4)];
50 -   end

```

```

52 - function expmat = MatrixLog6(T)
53 -     [R, p] = TransToRp(T);
54 -     omgmat = MatrixLog3(R);
55 -     if isequal(omgmat, zeros(3))
56 -         expmat = [zeros(3), T(1: 3, 4); 0, 0, 0, 0];
57 -     else
58 -         theta = acos((trace(R) - 1) / 2);
59 -         expmat = [omgmat, (eye(3) - omgmat / 2 ...
60 -             + (1 / theta - cot(theta / 2) / 2) ...
61 -             * omgmat * omgmat / theta) * p;
62 -             0, 0, 0, 0];
63 -     end
64 - end

66 - function so3mat = MatrixLog3(R)
67 -     acosinput = (trace(R) - 1) / 2;
68 -     if acosinput >= 1
69 -         so3mat = zeros(3);
70 -     elseif acosinput <= -1
71 -         if ~NearZero(1 + R(3, 3))
72 -             omg = (1 / sqrt(2 * (1 + R(3, 3)))) ...
73 -                 * [R(1, 3); R(2, 3); 1 + R(3, 3)];
74 -         elseif ~NearZero(1 + R(2, 2))
75 -             omg = (1 / sqrt(2 * (1 + R(2, 2)))) ...
76 -                 * [R(1, 2); 1 + R(2, 2); R(3, 2)];
77 -         else
78 -             omg = (1 / sqrt(2 * (1 + R(1, 1)))) ...
79 -                 * [1 + R(1, 1); R(2, 1); R(3, 1)];
80 -         end
81 -         so3mat = VecToso3(pi * omg);
82 -     else
83 -         theta = acos(acosinput);
84 -         so3mat = theta * (1 / (2 * sin(theta))) * (R - R');
85 -     end
86 - end

87 -
88 - function T = RpToTrans(R, p)
89 -     T = [R, p; 0, 0, 0, 1];
90 - end

```

```

92 - function Jb = JacobianBody(Blist, thetalist)
93 -     Jb = Blist;
94 -     T = eye(4);
95 -     for i = length(thetalist) - 1: -1: 1
96 -         T = T * MatrixExp6(VecTose3(-1 * Blist(:, i + 1) * thetalist(i + 1)));
97 -         Jb(:, i) = Adjoint(T) * Blist(:, i);
98 -     end
99 - end
100
101 - function T = FKinBody(M, Blist, thetalist)
102 -     T = M;
103 -     for i = 1: size(thetalist)
104 -         T = T * MatrixExp6(VecTose3(Blist(:, i) * thetalist(i)));
105 -     end
106 - end
107
108 - function T = MatrixExp6(se3mat)
109 -     omgtheta = so3ToVec(se3mat(1: 3, 1: 3));
110 -     if NearZero(norm(omgtheta))
111 -         T = [eye(3), se3mat(1: 3, 4); 0, 0, 0, 1];
112 -     else
113 -         [omghat, theta] = AxisAng3(omgtheta);
114 -         omgmat = se3mat(1: 3, 1: 3) / theta;
115 -         T = [MatrixExp3(se3mat(1: 3, 1: 3)), ...
116 -             (eye(3) * theta + (1 - cos(theta)) * omgmat ...
117 -              + (theta - sin(theta)) * omgmat * omgmat) ...
118 -             * se3mat(1: 3, 4) / theta;
119 -             0, 0, 0, 1];
120 -     end
121 - end

```

```

123 - function R = MatrixExp3(so3mat)
124 -     omgtheta = so3ToVec(so3mat);
125 -     if NearZero(norm(omgtheta))
126 -         R = eye(3);
127 -     else
128 -         [omghat, theta] = AxisAng3(omgtheta);
129 -         omgmat = so3mat / theta;
130 -         R = eye(3) + sin(theta) * omgmat + (1 - cos(theta)) * omgmat * omgmat;
131 -     end
132 - end
133
134 - function omg = so3ToVec(so3mat)
135 -     omg = [so3mat(3, 2); so3mat(1, 3); so3mat(2, 1)];
136 - end
137
138 - function se3mat = VecTose3(V)
139 -     se3mat = [VecToso3(V(1: 3)), V(4: 6); 0, 0, 0, 0];
140 - end
141
142 - function so3mat = VecToso3(omg)
143 -     so3mat = [0, -omg(3), omg(2); omg(3), 0, -omg(1); -omg(2), omg(1), 0];
144 - end
145
146 - function judge = NearZero(near)
147 -     judge = norm(near) < 1e-6;
148 - end
149
150 - function [omghat, theta] = AxisAng3(expc3)
151 -     theta = norm(expc3);
152 -     omghat = expc3 / theta;
153 - end
154
155 - function invT = TransInv(T)
156 -     [R, p] = TransToRp(T);
157 -     invT = [transpose(R), -transpose(R) * p; 0, 0, 0, 1];
158 - end
159
160 - function [R, p] = TransToRp(T)
161 -     R = T(1: 3, 1: 3);
162 -     p = T(1: 3, 4);
163 - end
164
165 - function AdT = Adjoint(T)
166 -     [R, p] = TransToRp(T);
167 -     AdT = [R, zeros(3); VecToso3(p) * R, R];
168 - end

```

## Appendix C: Trajectory Generation

```

1  function traj = ScrewTrajectory(Xstart, Xend, Tf, N, method)
2      timegap = Tf / (N - 1);
3      traj = cell(1, N);
4      for i = 1: N
5          if method == 3
6              s = CubicTimeScaling(Tf, timegap * (i - 1));
7          else
8              s = QuinticTimeScaling(Tf, timegap * (i - 1));
9          end
10         traj{i} = Xstart * MatrixExp6(MatrixLog6(TransInv(Xstart) * Xend) * s);
11     end
12 end
13
14 function s = QuinticTimeScaling(Tf, t)
15     s = 10 * (t / Tf) ^ 3 - 15 * (t / Tf) ^ 4 + 6 * (t / Tf) ^ 5;
16 end

```

## Appendix D: Graphical User Interface based on end-effector axis

The following code will run once the Z+ button is pressed in the GUI.

```

243 function Z_plus_Callback(hObject, eventdata, handles)
244 % hObject    handle to Z_plus (see GCBO)
245 % eventdata  reserved - to be defined in a future version of MATLAB
246 % handles    structure with handles and user data (see GUIDATA)
247 -
248 -     global T;
249 -     T_initial = T;
250 -     direction = 'z';
251 -     increment = get(handles.length, 'string');
252 -     disp(increment);
253 -     increment2 = str2num(increment)
254 -     class(increment2);
255 -     T_final = tmove(direction, increment2, T_initial);
256 -     for motor_number = 1:6
257 -         initial_angle = T_initial(motor_number);
258 -         final_angle = T_final(motor_number);
259 -         motor(motor_number, final_position, initial_position);
260 -     end
261 -     T = T_final;

```

## Appendix E: Graphical User Interface based on motor axis

The following code will run once the motor 1 button is pressed in the GUI

```

140 % --- Executes on button press in rotate_1.
141 function rotate_1_Callback(hObject, eventdata, handles)
142 % hObject      handle to rotate_1 (see GCBO)
143 % eventdata    reserved - to be defined in a future version of MATLAB
144 % handles      structure with handles and user data (see GUIDATA)
145 - global T;
146 - global direction_1;
147 - initial_position = T(1)
148 - motor_number = 1;
149 - increment = get(handles.Angle_1,'string');
150 - increment = str2num(increment)
151 - if direction_1 == 1;
152 -     final_position = initial_position + increment;
153 - else
154 -     final_position = initial_position - increment;
155 - end
156 - motor(motor_number,final_position,initial_position);
157 - T(1) = final_position;
158 - disp(T);

```

## Appendix F: Motor Control Code

```
#include <AccelStepper.h>

// Define some steppers and the pins the will use
AccelStepper stepper1(AccelStepper::FULL4WIRE, 2, 3, 4, 5);
AccelStepper stepper2(AccelStepper::FULL4WIRE, 6, 7, 8, 9);
AccelStepper stepper3(AccelStepper::FULL4WIRE, 10, 11, 22, 24);

double doi1;
double doi2;
double doi3;

void setup()
{
    Serial.begin(9600);
    while(!Serial.available()) {}
    String temp;
    temp = Serial.read();
    Serial.print(temp);
    doi1 = double((temp.substring(0,1)).toDouble());
    doi2 = double((temp.substring(3)).toDouble());
    doi3 = double((temp.substring(16)).toDouble());
    stepper1.setMaxSpeed(800.0);
    stepper1.setAcceleration(100.0);
    stepper1.moveTo((38000/90)*doi1);

    stepper2.setMaxSpeed(800.0);
    stepper2.setAcceleration(100.0);
    stepper2.moveTo(22000*doi2/90);

    stepper3.setMaxSpeed(600.0);
    stepper3.setAcceleration(100.0);
    stepper3.moveTo(10000*doi3/90);
}

void loop()
{
    stepper1.run();
    stepper2.run();
    stepper3.run();
}
```

---



## Appendix G: Haptic Feedback Control

This code is used to control the feedback on the haptic device.

```
int pressureAnalogPin = A0; //pin where our pressure pad is located.
int pressureReading; //variable for storing our reading
#include <Servo.h>
Servo myservo;
int pos = 0;

void setup(void) {
  Serial.begin(9600);
  myservo.attach(9);
}
void loop(void) {
  pressureReading = analogRead(pressureAnalogPin);

  if (pressureReading > 50 ) {
    while (pressureReading > 50 ) {
      pressureReading = analogRead(pressureAnalogPin);
      float kek=(pressureReading/500)*(120);
      analogWrite(A1, kek);
      Serial.print("Pressure Pad Reading = ");
      Serial.println(pressureReading);
    }
  }

  analogWrite(A1, 0);
  myservo.write(180);
}
```

## Appendix H: Serial Communication Code

This code is used to initiate the serial communication between Arduino and MATLAB

```
1 - |s=serial('COM4','BAUDRATE',9600);    %to create the serial port in MATLAB
2 - fopen(s);                            %open the serial port
3 - fwrite(s,T_rotate);                  % writes the binary data A to the device connected to the serial port object, s.
4 - fclose(instrfind);
```

## Appendix I: Encoder Rotation Measurement

This code is tasks incremental encoder to measure rotation and is utilized in the forward loop of the haptic device

```
#include <Encoder.h>

// Change these two numbers to the pins connected to your encoder.
//   Best Performance: both pins have interrupt capability
//   Good Performance: only the first pin has interrupt capability
//   Low Performance:  neither pin has interrupt capability
Encoder myEnc(5, 6);
//   avoid using pins with LEDs attached

void setup() {
  Serial.begin(9600);
  Serial.println("Basic Encoder Test:");
}

long oldPosition = -999;

void loop() {
  long newPosition = myEnc.read();
  if (newPosition != oldPosition) {
    oldPosition = newPosition;
    Serial.println(newPosition);
  }
}
```