Design and Control of Throat Swab Sampling Robot

Senior Design Project Report

BY

Muhammad Ibraheem 2018288

Muhammad Musa Chughtai 2018309

Asher Junaid 2018084

Supervised By:

Dr. Abid Imran

Faculty of Mechanical Engineering

GIK Institute of Engineering Sciences & Technology

May 2022

Design and Control of Throat Swab Sampling Robot

Senior design project report

BY

Muhammad Ibraheem 2018288

Muhammad Musa Chughtai 2018309

Asher Junaid 2018084

Supervised by
Dr 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 2022

MAY 2022

Senior Design Project Status/Completion Certificate
Group No: 20

Title: Design and Control of Throat Swab Sampling Robot

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			
(vii)	Students followed the given deadlines by SDP committee			

_____

FYP Coordinator

_____

Dean FME

Dean FME

FYP Mapping with Complex Engineering Problem Attributes

Group No:
Title:
Problem Statement:

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

Pandemics pose a gigantic risk to the global economy, making it incredibly essential to have pandemic prevention and control strategies in place. For pandemic control it is essential to have a safe but effective method of collecting test samples from a massive number of people. We have sought to develop a remotely controlled slave robot, capable of remotely and effectively taking the Oropharyngeal Swab sample from a test subject. We mathematically modelled, designed and manufactured a Stewart platform that was connected to and controlled by a joystick and node red GUI via IOT using MQTT protocol. The slave robot has a probe capable of being manipulated in six degrees of freedom inside the workspace, controlled remotely by the master device whose operator can see a live video depicting the probe and the test subject. the probe can extend itself into or retract itself from the oral cavity. By making the Master and Slave systems components of the Internet of Things, we can allow them to be controlled from anywhere.

Keywords: Oropharyngeal, Degree of Freedom (DOF), Inverse Kinematics, IOT, MQTT, PWM

Table of Contents

Nomenclature

Nasopharyngeal (NP) – relating to the nasal cavity

Oropharyngeal (OP) – relating to the oral cavity

Serial/Open Chain Robot

Parallel /Closed chain robot

Joint

Degree of Freedom (DOF)

Universal Joint – rotational 2DOF joint

Prismatic Joint – joint that allows for linear motion

Spherical Joint – rotational 3 DOF joint

Revolute Joint

Coupler

Crank

UPS (Universal Prismatic Spherical) configuration of Stewart Platform

RUS (Revolute Universal Spherical) configuration of Stewart Platform

Inertial Measurement Unit (IMU)

Inverse Kinematics

Forward Kinematics

Transformation Matrix

IOT Internet of Things

MQTT - Message Queuing Telemetry Transport

MQTT Broker

PWM Pulse Width Modulation

Duty Cycle

List of Tables

List of Figures

1 Introduction

1.1 Background and Motivation

According to the WHO the world was not only ill-prepared for pandemics in the run-up to the COVID-19 crisis, such a pandemic was also highly likely, and it had been alerting governments to the fact for more than a decade[1]. This vulnerability arises particularly from the increased interconnectivity of the world. The globalized economy, the ease, and the sheer volume of international travel mean that any pathogen - whether synthesized in a lab or passed on from animals to humans – with a high enough transmissibility, can pose a massive risk to global health and economy. The WHO has therefore stressed the need for pandemic preparedness. This involves not only Research and Development for newer and faster methods of vaccine development but also vastly improving diagnostic capabilities.

Diagnosing a pathogen primarily involves tests and looking at symptoms. While symptoms might not be the most reliable of tools, tests are either difficult to conduct, longer to conclude, or unavailable. To have a superior testing capability that can provide the kind of data which is necessary to trace and inhibit the spread of the disease, governments need to conduct regular mass testing of as big a share of the population as possible. This currently means establishing testing facilities as well as collecting samples of those people on-site, who are unable or unwilling to visit the testing facilities voluntarily. Invariably this involves hiring teams of healthcare workers who staff the testing facilities and collect samples of test subjects. These healthcare workers are putting themselves in harm's way as they are at the risk of being exposed to the pathogen whenever they collect a test sample. Due to increased exposure, healthcare workers have been found to be seven times more likely than non-essential workers to contract severe covid-19[2]. This is especially true of pathogens that infect the respiratory tract or are spread via the respiratory tract. The COVID-19 virus is not the only pandemic disease that can spread through the respiratory systems of humans in an aerosolized form as highlighted in the WHO briefing warning of the spread of an influenza-like virus was [1].

This throat swab sampling robot will collect the oropharyngeal swab sample of a test subject and will be remote-controlled by a healthcare professional. This primary advantage of this is that the healthcare worker does not need to come into proximity with the potentially infected test subject and that the test subject also does not need to face the risk of exposure from the healthcare worker. Moreover, it is much easier to ensure that the room that the device is in is continuously sanitized, the air is being continuously disinfected and recycled, as well as the testing apparatus itself is being prevented from transmitting the pathogen.

With the advent of the Internet of Things (IoT), we can control the robot from any place with an internet connection. This can allow organizations to manage the workforce more efficiently as the workforce will be utilized only where it's required and does not need to be stationed at less frequented testing facilities. This also addresses a central issue with the present testing infrastructure which is finding and funding the human labor necessary. This frees up human resources for places where their need is at present irreplaceable. Moreover, the robot also provides a platform for further automation of the sample collection process, as it is possible to

use video feedback and artificial intelligence to further automate the process of swab collection. At that stage, the human would not need to remotely execute the complete collection of the swab but would only need to command the robot to do so. This would be very cost-effective in countries with high labor costs, allowing the government and organizations to divert funds elsewhere.

We aim to develop a mechanism that can be used to remotely collect the Oropharyngeal swab sample of test subjects. We want to do this because the process of the collection of test samples needs to be safer and more efficient to increase global pandemic preparedness.

1.2 Problem Statement

Currently, the most used method for COVID-19 detection is a polymerase chain reaction (PCR) test which requires a genetic sample acquired through OP or Nasopharyngeal (NP) swab sample, explained in the following:

"Inserting a swab in the mouth through with the help of a tongue depressor to reach the OP (Oropharynx) and collection of the genetic sample".

We need to develop a mechanism capable of being controlled remotely, that firsts insert and then fully retrieves a swab from the throat of the test subject. It needs to be remotely controlled via IoT and provide real-time video feedback to the end-user.
The robot must be reliable and cost-effective as well to prove that it can be adopted enmasse.

1.3 Scope of the work

The project upon completion can provide the medical sector of this world with a robot that can help to remove the risks of spreading of any genetic pathogen that can be emitted from the OP or NP (Oropharynx or the Nasopharynx) while conducting PCR tests remotely.

The following tasks must be conducted thoroughly to complete this project:

- Design, fabrication, and control of the Stewart Platform with suitable actuators.
- Reverse kinematics study of the Stewart Platform for its use in all 6 DOFs in the designated workspace.
- Design, fabrication, and control of end factor insertion and monitoring.
- Integration of control algorithms with GUI for the operation.
- Combined control of the mechanism to imitate proper functioning.
- Image processing to add the ability of AI-guided functioning of the procedure. (Optional)

1.4 Report Outline

- Chapter 1 deals with the Background and Introduction of the overall project along with task division to each group member
- Chapter 2 deals with the Literature review and sensible inference drawn from it. This links our project to meaningful research significance keeping in view the previous research done.

- Chapter 3- Design Methodology includes Mathematical Modelling, MATLAB code, Analysis of Codes and Standards

1.5 Project Schedule

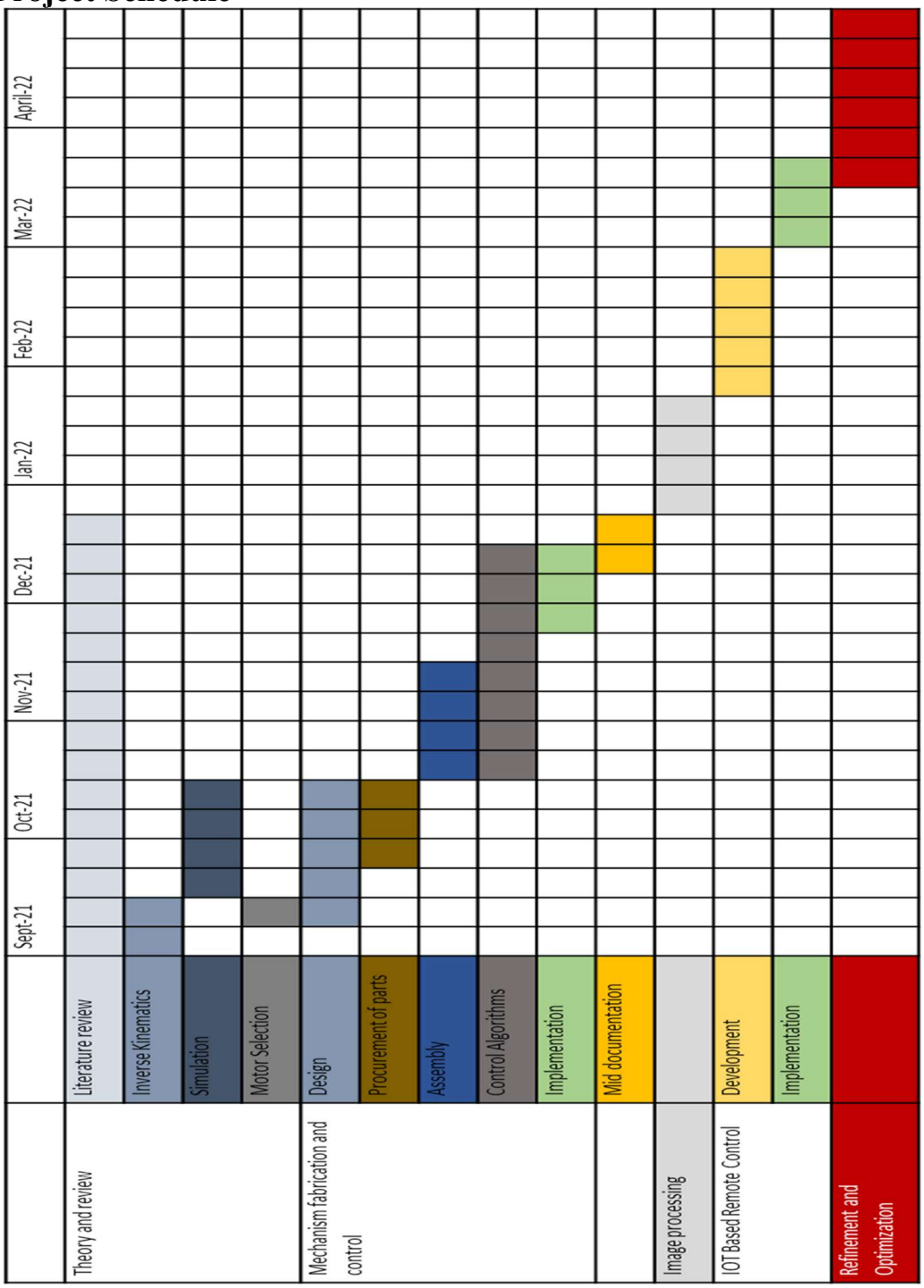


Table 1-1: Gantt Chart

1.6 Individual and team contribution

Table 1-2: Team Contribution

Name	Reg No.	Task
Muhammad Ibraheem	2018288	CAD modelling, mathematical modelling, MATLAB, Control
Muhammad Musa Chughtai	2018309	MATLAB, CAD modelling, mathematical modelling, Assembly
Asher Junaid	2018084	Assembly, MATLAB, CAD modelling,

In the Table above we have mentioned the contributions of each member to the project. Most things were done collaboratively and therefore have been shown as such.

2 Literature Review

2.1 Literature Review

In recent decades, like in most parts of the lives of human beings, robotics was introduced has been introduced to the surgery sector as well. 1988 was marked as the year when the first robot was ever used in this sector. It was used to reduce the uncertainty that was subjected to surgery due to hand tremors. Over time, other major robots that have been used in this sector include the DaVinci surgical system, Xenex, and the Cyerknife. In recent years, the COVID-19 outbreak has claimed the lives of over 5 million people, out of which many have died while tending to the ones that were in the hospital. The virus has been diagnosed to spread from respiratory droplets that emerge from the mouth and the nose of a human being during respiration[3].

An inconsistency can always be seen when doctors have collected OP-swab (oropharyngeal swab) samples from patients. The OP swab sample is collected by the infiltration of the oral cavity and the application of some force and rotation[4]. Among other diseases, Polymerase Chain Reaction (PCR) tests are also used frequently for genealogy, drug tests, and detection of influenza, parainfluenza, asthma, E-Coli, Hepatitis, Human Metapneumovirus, and almost every disease or illness that can be detected from genealogy[5].

To reduce the exposure of doctors to the virus during the intrusions of the oropharyngeal cavity without any chances of error, the use of teleoperated robots has been suggested since the start of the outbreak. Chances of error in terms of collecting PCR swabs may be either because the swab itself has not intruded the OP (oropharynx) enough to collect the sample or it is because that swab has not been applied with enough force or rotated enough when contacting the sample rich site in the OP or NP (nasopharynx). Medical studies do suggest that the samples that are collected for the nasopharynx are better for PCR testing because the doctors can collect much richer samples from there but the difference between the OP and NP swabs was shown to be minimized by the fact that OP can provide better samples if it is used precisely and with greater caution[6].

Hence the conduction of the sample collection process while using a robot was considered as a much favorable way after the need for collecting OP and NP swabs skyrocketed after the emergence of the COVID-19.

The use of robots in medical sciences has been done in the past before but for the COVID-19 specifically, there are only a handful of research papers that contributed significantly. In that handful, there are ones that are that require the operator to apply the force by hand and use the concept of haptic feedback systems to operate the robot. On the other hand, some have been operated by pure signals.

2.1.1 History of Stewart Platforms

The robot that has been called the Stewart platform earlier is also called Gough Platform, 6-UPS robot, 6-6 robot or hexapod. Parallel Robotics by J.-P. MERLET [7] was very helpful in listing down all the research that has ever been done for the Stewart platform. They have

documented the different types of Stewart platforms and how those different types are different from each other. The first step in analyzing any Stewart platform is looking at its configuration. This may be in terms of the series of joints that connect the two platforms. These parallel mechanisms are also classified in terms of the joints that are used in between the six links that are present between the two platforms of the mechanism. The most popular classifications that come into this category can include UPS which stands for Universal-Prismatic-Spherical or RUS which stands for Rotary-Universal-Spherical, but there are other six DOF configurations as well which include the PUS. the type of robot that the Stewart platform is called a parallel mechanism in many kinds of literature such as [7].

2.1.2 Medical History of Swab Sampling with Robots

The transoral robot that is the subject of a research [8] done by Li Changsheng was inserted into the mouth of the test subject and the probe that was inserted was contributing to a couple of roles.

The primary role that the probe mechanism was made for is that the swab can infiltrate the OP cavity and that was fulfilled using a flexible swab which was pushed forward into the mouth of the test subject with the use of flexible shafts and flexible manipulators. The probe mechanism that was built for this robot was a combination of a parallel mechanism and the flexible mechanism, which had the flexible shafts and the manipulator (mentioned above) as its subsystems. The lightweight parallel system that was integrated into this mechanism provides the dexterity that is needed to achieve the precision that is needed for the OP infiltration. Whereas all the flexible parts of the robot are required as compensation for the excess force that can be applied to the test subject. Not including either of these parts from the system would fail one of the biggest constraints that are put onto OP tests; the operator or the robot should not, in any way, harm the test subject.

Another major role that the probe mechanism plays for the robot's proper functioning is that the flexible mechanism initially inserts a tongue depressor along with the swab so that the tongue can be held down to increase the workspace of the robot and simultaneously decrease the chance of any accidents that may happen.

The input for the working of this robot is achieved by a master device that collects the physical commands from an operator (a surgeon) which then actuates motors after processing the data from the feed received from the master device. This data is then used to produce the input of the motors that are combined with thread wires and together it makes the flexible mechanism that inserts the swab in the Oropharynx.

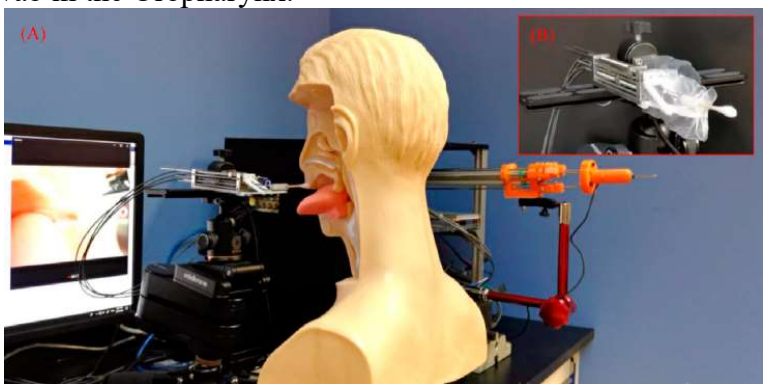


Fig 2-1 Mentioned mechanism shown above in testing stages.

In the picture shown, the probe has not entered the OP yet and the tongue depressor is right behind it, suspended and supported on the lightweight parallel mechanism. This parallel mechanism is positioned on a tripod while it is also given access.

An article [9] written by Seo Joohno showed the study that was done to collect PCR test samples with the help of a robot that is built like a Stewart platform. The Stewart platform, if explained in simple words, has an upper and lower base, of which the lower base is mostly static and is kept so that the upper base can be maneuvered in 6 DOF. The only time that the lower base is also moved is when there is the need of offsetting the position of the upper plates so that it can reach its target area easily. The relative motion between the upper and the lower platforms is done with the help of actuators that are connecting the two. These actuators can be linear actuators, hydraulic or even rotary type (ones that are controlling the upper platform with the help of some motors).
The study that was done in this article was particularly focused on making a Stewart platform that is being controlled with the help of servo motors (much like what is being done in our paper). in this paper, it is mostly referred to as the slave mechanism. The slave mechanism is positioned right next to the test subject so that the probe mechanism, which is supposed to intrude the OP and NP is well aligned. The slave mechanism is controlled from the master mechanism that is based upon the same principle: the same number of actuators being controlled being controlled from the surgeon or the controller.
The actuators that are used in this setup are servo motors. The total number of these actuators is twelve, six of them being used in the slave component and six of them being used in the master component. The motors that are present in the slave component are located on the upper base of the slave mechanism (This is different from what we have opted to do in our project because adjusting the motors on the top platform may provide some ease in terms of 3-6 configuration that is being used in [9] but doing so also suspends the weight of the motors on themselves as well, which in the end adds to the risk factor in terms of the motors failing at any time). The probe on the other hand is controlled by another motor that monitors the distance that a strip (which imitates the swab) travels. The strip is coupled with the motor and their working is similar that of the of a rotational potentiometer. This strip is also controlled by the operator and is actuated with optimum velocity (a value that is not threatening to the test subject) when master mechanism is done with aligning the slave mechanism with the opening of the OP.

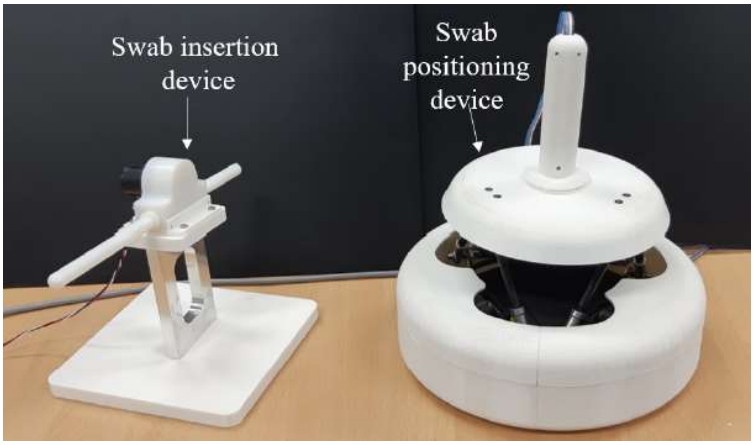


Fig 2-2 Master devices made and used in the cited research.

The two parts of the master device are shown in the diagram above. The Swab positioning device or simply mentioned as the master device above is also based upon a Stewart mechanism but instead of giving any output, this mechanism takes input in terms of haptic feed that is later imitated on the slave mechanism as well. The mechanism on the left (the swab insertion device) is the probe controller device that gets to decide after input and calibration, the amount of intrusion and the speed of it.

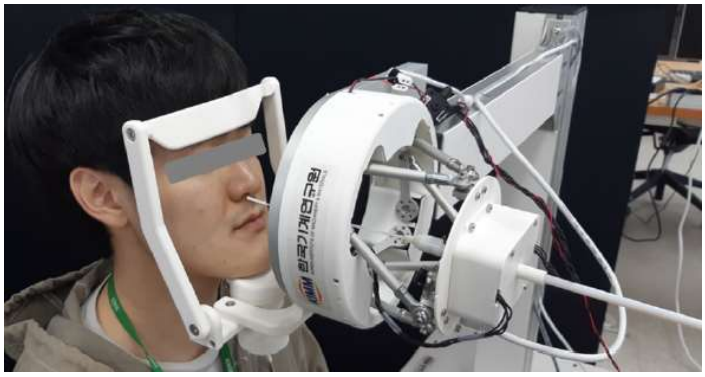


Fig 2-3 Slave mechanism invading the NP of the test subject

In the photograph shown above, the slave mechanism is shown to be performing surgery on a test subject while the test subject's head is placed on a chin holder while also being held in place with a forehead rest. Both are present to provide extra stability to the slave mechanism during the procedure.

2.1.3 Non-Medical application of Stewart Platforms

The research done by Trent Robert Peterson [10] helped us greatly with this project. It mentioned the different configurations of the apparatus and how they make a difference. For the preliminary research, the tire testing and benchmarking machine was used. This machine is used to test newer designs of tires after putting them through a makeshift terrain at different rotational speeds. The makeshift terrain that is used in the mentioned apparatus is the Stewart platform, but the size of its platform is from three to 4 meters in diameter, which is possible because of the high payload capacity of parallel manipulators.
Moreover, the platform has also been widely used for aircraft and racing simulation. In racing simulation environments, the Stewart platform allows the user to be seated in the top platform while the platform changes orientation in synchronization with the race car in the simulated environment or video game.[11] Moreover, it has also be used in flight simulators to emulate the aircraft's six degrees of freedoms.

2.2 Inferences Drawn from the Literature

To support the 6-DOF movement of the upper plates, the design must accommodate 6-DOFs in each of the six links that are being used to join the upper and the lower platforms. These degrees of freedom are necessary if the actuator is rotary and the element of a DOF being added or removed will be done by using different joints. Doing this will also classify the type of Stewart Mechanism one ends up making. In addition to that, to design and assemble a parallel mechanism that is both precise and stable, we must incorporate some sort of rigidity in the

structure, we must preload the joint arrangements that are required. This way the mechanism does not become wobbly when it is being used. Such rigidity comes when the links that are used to join the structure are placed at some angles and not completely perpendicular to the plates.

To achieve rigidity, making both plates of different diameters can also prove to be useful. This way, balanced force coordinates in the radial directions can also exist on the link nodes. This helps a lot in holding the structure together and increasing accuracy altogether.

The upper and lower joints can all be placed at arbitrary angles. They can either be placed at a 60 degrees angle of each other on both plates but that will end up in an absence of the rigidity discussed above. We can choose whichever actuator that we want to use. Every single type of actuator has its own benefits and disadvantages that one can consider before deciding. The actuators include motors, linear actuators (lead screw), hydraulic actuators. Given that the task of the robot requires a lot of sensitivity, we should minimize the critical workspace (the area in which the swab can roam freely while operating) as much as possible.

2.3 Summary

After doing the literature review, we were able to formulate an outlay of how we will proceed with the design process.
First and foremost, we must ensure the configuration of our links, more specifically the type of actuator that we want to use. We opted for six links, each consisting of a servo motor that is actuating a moment arm that is joined with the upper plate with the help of a coupler arm. Between these three nodes, nodes where either two of the above three mentioned parts are joining, there is a joint needed. We opted to go for the RUS 6-6 configuration of the parallel robot, using a rotary joint between the moment arm and the servo motor, a universal joint between both arms and a universal joint between the coupler arm and its respective joint on the upper plate of the platform.
All the links are placed in pairs of 3 consisting of a 30-degree angular distance between the elements of a single pair while the pairs are at an angular distance of 120-degrees of each other. If we consider that the three pairs make a triangle, we placed the top and bottom plate in cross configuration and joint the links in that configuration as well. doing this provided us with better preliminary traction in the robot so we can avoid any wobbling that may occur.
The control of the robot will be done with a PID controller coupled with a motor driver and the probe (end factor) mechanism is supposed to be attached on the top of the upper plate and for the time that we want to operate and test the parallel mechanism on a test subject, we will rotate the mechanism so that vertical axis becomes the horizontal axis and probe mechanism faces the patient.

3 Design And Analysis

3.1 Design Process

We must now move on to the Design and Analysis of the mechanism. We will first design and fabricate the Stewart platform for the swab sampling robot and then implement the control of Stewart platform. Following that, we will begin working on the swab sampling mechanism for the Stewart Platform.

The engineering design process begins by a selection of objectives that the design is ultimately supposed to fulfill. In this case we are looking to design a Stewart platform that has the appropriate dexterity and reachability required to perform the collection of an Oropharyngeal swab sample from the test subject. After studying from literature as well as the constraints of our proposed design we have ascertained that a Stewart platform with the following characteristics would perform the required task. [11]

a) To design a Stewart Platform capable of rotating 25 degrees about the x, y, and z axes of the moving platform's origin.
b) To design a platform capable of translating at least 4 cm along the x, y and z axes of the moving platform of the mechanism.

The next step in the design process is concept generation in which we discover and develop a potential way of solving our problem. After analyzing the initial concept, conducting the necessary modeling, and encountering design constraints, we move on to a more detailed design.

3.1.1 Block Diagram

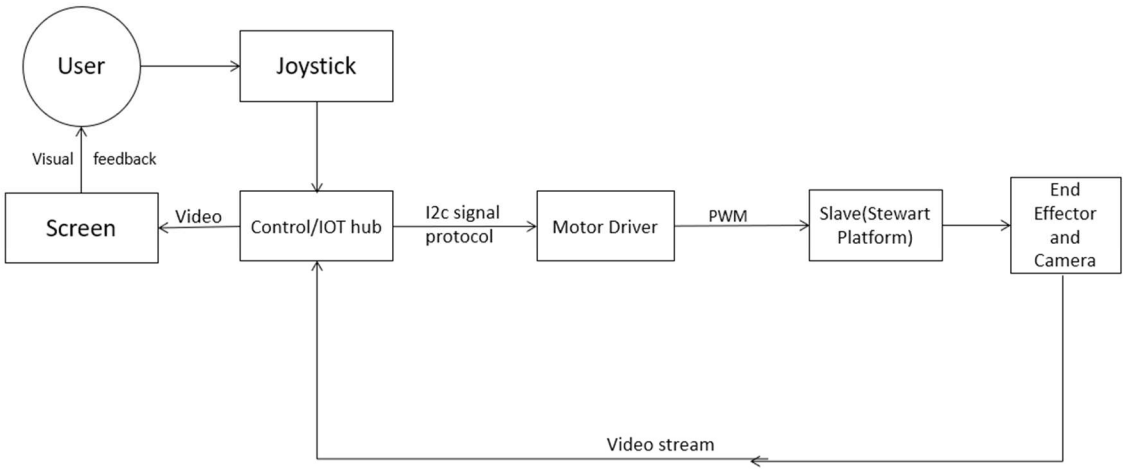


Fig 3-1 Block Diagram of the Stewart Platform

The Block diagram shown in the figure 3-1 is used to depict the flow of information in our system. Our block diagram shows that it's a closed-loop system in which the Plant (Stewart Platform) provides feedback to the controller in the form of data from the IMU which provides the orientation of the moving platform at any instant of time. Moreover, the camera attached at the end effector also provides real-time feedback to the user. The IoT capability allows the

user to access the system from any place at all and still receive the video feedback of the robot's end-effector.

3.1.2 Concept Generation

Fig 3.2 shows the design used by [10] to develop a robotic mechanism to take the nasopharyngeal swab sample of a test subject. That is the example we used to generate our concept which we refined further after analysis of its reachability and dexterity.

The RSS configuration shown in the figure 3-2 introduces a redundancy into the system, which makes it difficult to control the platform as the number of solutions of the inverse kinematics become infinite.[12] Therefore we decided to develop a RUS configuration of the Stewart platform, that means that each of its legs consists of an actuated revolute joint, an unactuated Universal joint and another unactuated Spherical joint.

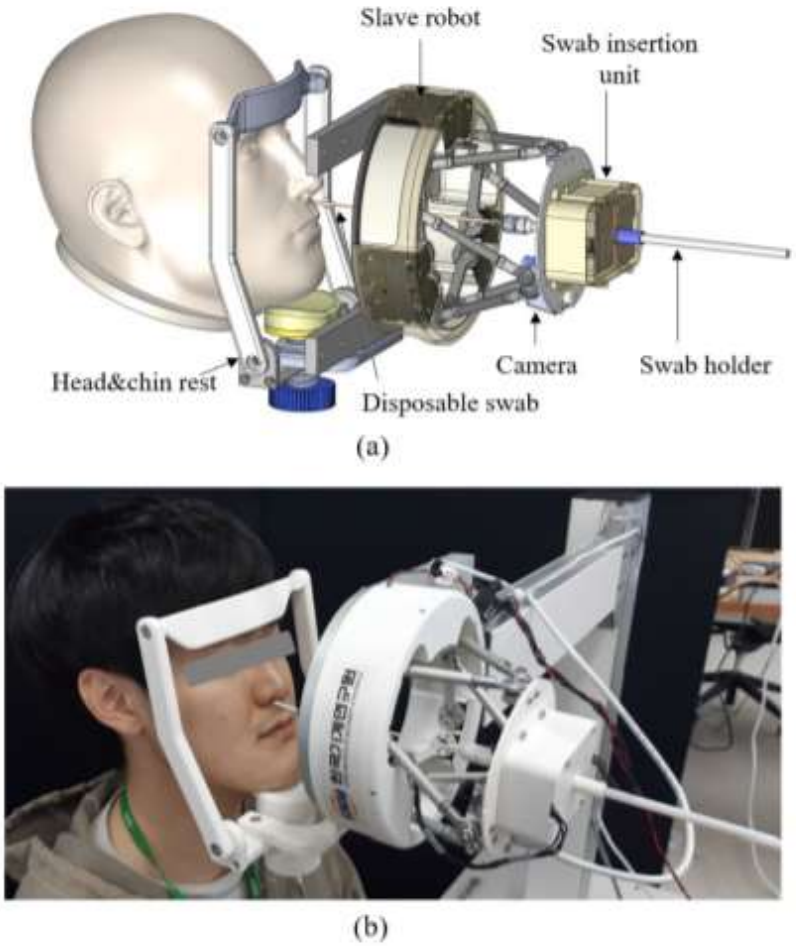


Fig 3-2 Images viewed for concept generation

3.1.3 Configuration and Geometry of the Stewart Platform

We need to first design and control the parallel manipulator on top of which the probe containing the swab sample is to be mounted. The Stewart platform is a parallel manipulator, which has a stationary base platform, a moving platform and six legs connecting the moving

platform to the base. The moving platform has six DOF. That is achieved by changing the lengths of each leg which individually also have 6 DOF.

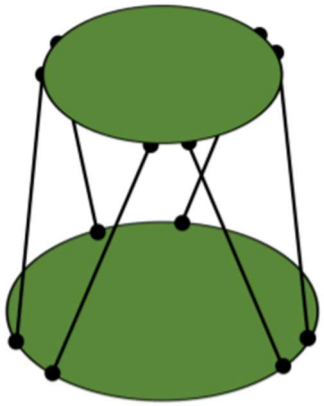


Fig 3-3 A general Stewart Platform

The legs can be of several configurations, UPS (Universal, Prismatic, spherical), PUS (Prismatic, Universal, Spherical), or RUS (Rotary, Universal, Spherical). The most common configurations are UPS, and RUS. In UPS configuration each leg is connected to the base with a universal joint (2DOF), the prismatic joint (e.g., a lead screw) extends or retracts the leg and has 1 DOF, while the link is connected to the moving platform by a 3DOF joint. In total the length has 6DOF, and while this is the most common configuration, the order of the joints can be altered to connect the Spherical joint with the base and the universal joint with the top platform. In the UPS configuration, the Prismatic joint is the only joint which is actuated. In our project we will be using a RUS configuration.[7] In the RUS configuration as well, we are aiming to alter the length of the link vector between the point of attachment at the base and the point of attachment at the platform. This is done by having two links a crank and a coupler, the crank attaches to the point of attachment with the base in a 1DOF rotary joint while the coupler is attached to the platform by a 3DOF spherical joint. They are joined with each other by a 2DOF rotary joint or a universal joint. The mechanism is actuated by six servo motors, each rotating the link attached to the bottom plate.

3.2 Inverse Kinematics

Calculating the joint space variables from the desired orientation or position of the end effector in the task space, is described by the inverse kinematics of that mechanism. And the calculation of the end-effector position int the task space coordinates from on joint space parameters is known as the forward kinematics of a mechanism. For a parallel manipulator, the inverse kinematic equations can be solved analytically while the forward kinematics must be solved numerically and returns multiple solutions, whereas for a robotic or any other serial manipulator the opposite is true.

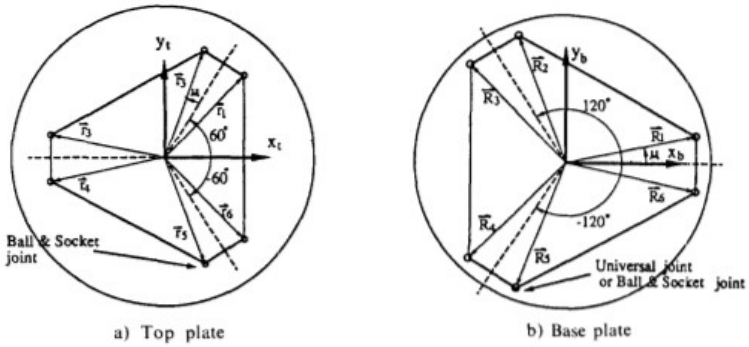
3.2.1 The aim of the Inverse Kinematics:

The inverse kinematics takes the task space coordinates of the end effector and calculates the joint space variables. In simple terms, for our RUS design of the Stewart platform, we need to calculate the angle α that is the rotation of the only actuated joint (as explained in the 3.1.2). Since the revolute joints are actuated but motors, the angle α is equal to the angle that the

motor's shaft needs to rotate for a desired rotation and position of the end effector to be achieved.

3.2.2 Equations

We will first begin by describing the geometry of the general Stewart platform. All the vectors discussed are depicted in fig 3.2. From the origin of the base, there are the axes (x_b, y_b, z_b). And at the center of the moving platform lies the origin of the moving platform from which originate the axes (x_t, y_t, z_t). The position vector of the moving platform with respect to the base origin (x_b, y_b, z_b) is R_t, where the rotary joint in our mechanism is placed. And the moving platform is attached as shown in fig 3.2 with the base by link vectors l_i, $l = (l_1, l_2, l_3, l_4, l_5, l_6)$ where l_i is the ith link that joins the base with the top platform and i as we move counter clockwise. Each of the vectors l_i is attached to the base at a point with the position vector (with respect to the origin of the base) R_i. And each l_i is attached to the moving platform at a point with the position vector (with respect to the moving plate) r_i^T, $r^T = (r_1^T, r_2^T, r_3^T, r_4^T, r_5^T, r_6^T)$, where a 3 DOF joint is placed. [9]



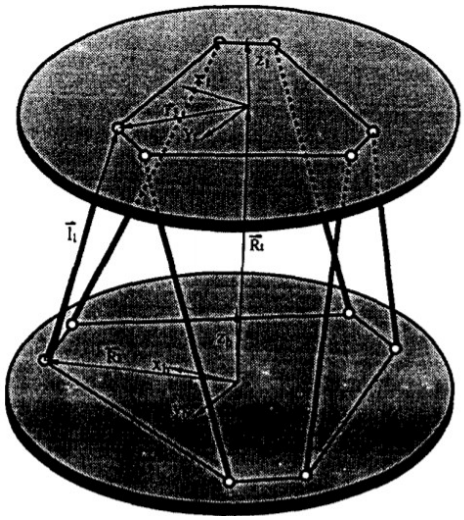


Fig 3-4 a, b, c [12]

To represent the rotation of the top platform about its axis, and to transform the r_i^T to reflect this rotation we multiply the r_i^T vectors with the transformational matrix ${}^B_T R$. The transformation matrix ${}^B_T R$ from the base to the moving platform, consists of the three angles of rotation of the moving platform about its axes.

$$[{}^B_T R] = rot(z,\psi)rot(y,\varphi)rot(x,\theta) \qquad \textit{3-1}$$

$$[{}^B_T R] = \begin{bmatrix} \cos\psi\cos\theta & -\sin\psi\cos\varphi + \cos\psi\sin\theta\sin\varphi & \sin\psi\sin\varphi + \cos\psi\sin\theta\cos\varphi \\ \sin\psi\cos\theta & \cos\psi\cos\varphi + \sin\psi\sin\theta\sin\varphi & -\cos\psi\sin\varphi + \sin\psi\sin\theta\cos\varphi \\ -\sin\theta & \cos\theta\sin\varphi & \cos\theta\cos\varphi \end{bmatrix} \qquad \textit{3-2}$$

When we input into the rotation matrix the values of the angles ψ, φ and θ that represent the rotation of the moving platform about its z, y or x axes respectively, we can find the vector

$$r_i = [{}^B_T R] * r_i^T \qquad \textit{3-3}$$

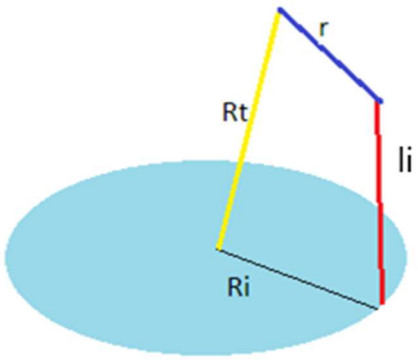


Fig 3-5 Vector sum to find l

Now we can use the vectors shown in Fig 3-4 to calculate the vector representing the vector l_i which is the vector representing the links.

$$l_i = R_t + r_i - R_i \qquad \textit{3-4}$$

Therefore, the magnitude of the link vector l_i is:

$$|l_i| = \sqrt{(R_t + r_i - R_i).(R_t + r_i - R_i)} \qquad \textit{3-5}$$

The equations shown above are for the general case of a Stewart platform, our platform has two connected links in place of each of the link vectors in fig 3.2c. The lower link, the crank, is connected at one end to a servo motor at the base, the position vector of this point is R_i. As shown in the figure 3-6.

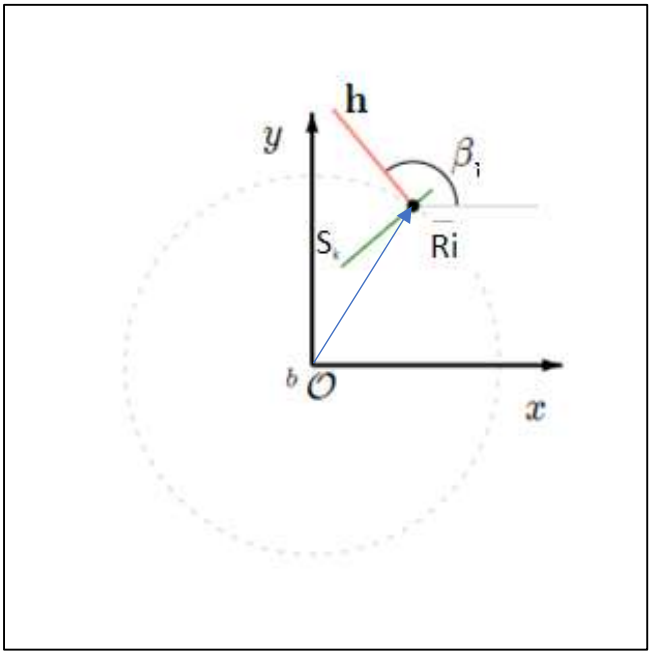


Fig 3-6 A snapshot of the base that shows only one motor and position vector

The angle β_i represents the angle between the crank and the x-axis. This angle is a constant for every motor and is only a result of the orientation of the motor shaft S_i.

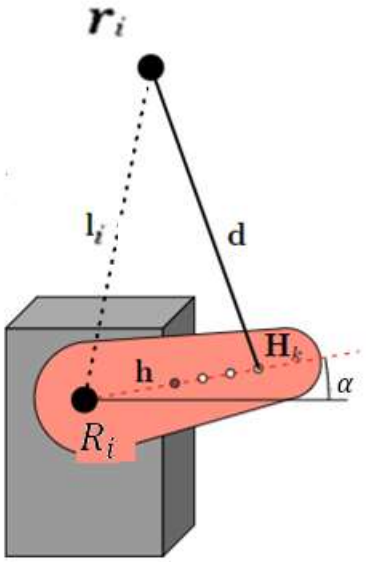


Fig 3-7 servo rotation arm

If h is the length of the crank, then the vector $\overrightarrow{R_\iota H_\iota} = h * \begin{bmatrix} \cos(\alpha) * \sin(\beta) \\ \cos(\alpha) * \cos(\beta) \\ \sin(\alpha) \end{bmatrix}$

$$\overrightarrow{O_\iota H_\iota} = R_i + h * \begin{bmatrix} \cos(\alpha) * \sin(\beta) \\ \cos(\alpha) * \cos(\beta) \\ \sin(\alpha) \end{bmatrix} \quad \textit{3-6}$$

Now taking the square of the norms of link length l_i, crank RH_i, and the coupler $H_i r_i$, we obtain the following simultaneous equations:

$$|h|^2 = (H_i - R_i)^T (H_i - R_i) \quad \textit{3-7}$$

$$|d|^2 = (R_t + r_i - H_i)^T (R_t + r_i - H_i) \quad \textit{3-8}$$

$$|l_i|^2 = (R_t + r_i - R_i)^T (R_t + r_i - R_i) \quad \textit{3-9}$$

By subtracting them we obtain:

$$|l_i|^2 - (|d|^2 - |h|^2) = 2l_i^z |h| \sin\alpha_i + 2|h| \cos(\alpha_i) (l_i^x \cos(\beta_i) + l_i^y \sin(\beta_i)) \quad \textit{3-10}$$

Trigonometric Identity:

$$\mathrm{e}\,.\sin\vartheta + \mathrm{f}.\cos\vartheta = \sqrt{e^2 + f^2}\sin(\vartheta + atan2(f,e)) \qquad \textbf{3-11}$$

By applying the trigonometric identity (3-11), and comparing it with equation 3-10, we get the following:

$$e = 2l_i^z|h| \qquad \textbf{3-12}$$

$$f = 2|h|(l_i^x\cos(\beta_i) + l_i^y\sin(\beta_i)) \qquad \textbf{3-13}$$

$$|l_i|^2 - (|d|^2 - |h|^2) = \sqrt{e^2 + f^2}\sin(\vartheta + atan2(f,e)) \qquad \textbf{3-14}$$

Therefore, since $\alpha_i = \vartheta$.

$$\alpha_i = \sin^{-1}\left(\frac{|l_i|^2 - (|d|^2 - |h|^2)}{\sqrt{f^2 + e^2}}\right) - atan2(f,e) \qquad \textbf{3-15}$$

Thus, we can calculate the required angle α_i at each of the servo motors which was the stated aim of conducting the inverse kinematics analysis of the Stewart Platform. [11]

3.3 CAD Model

The following is a snapshot of the complete assembly of the Stewart platform in solid works:

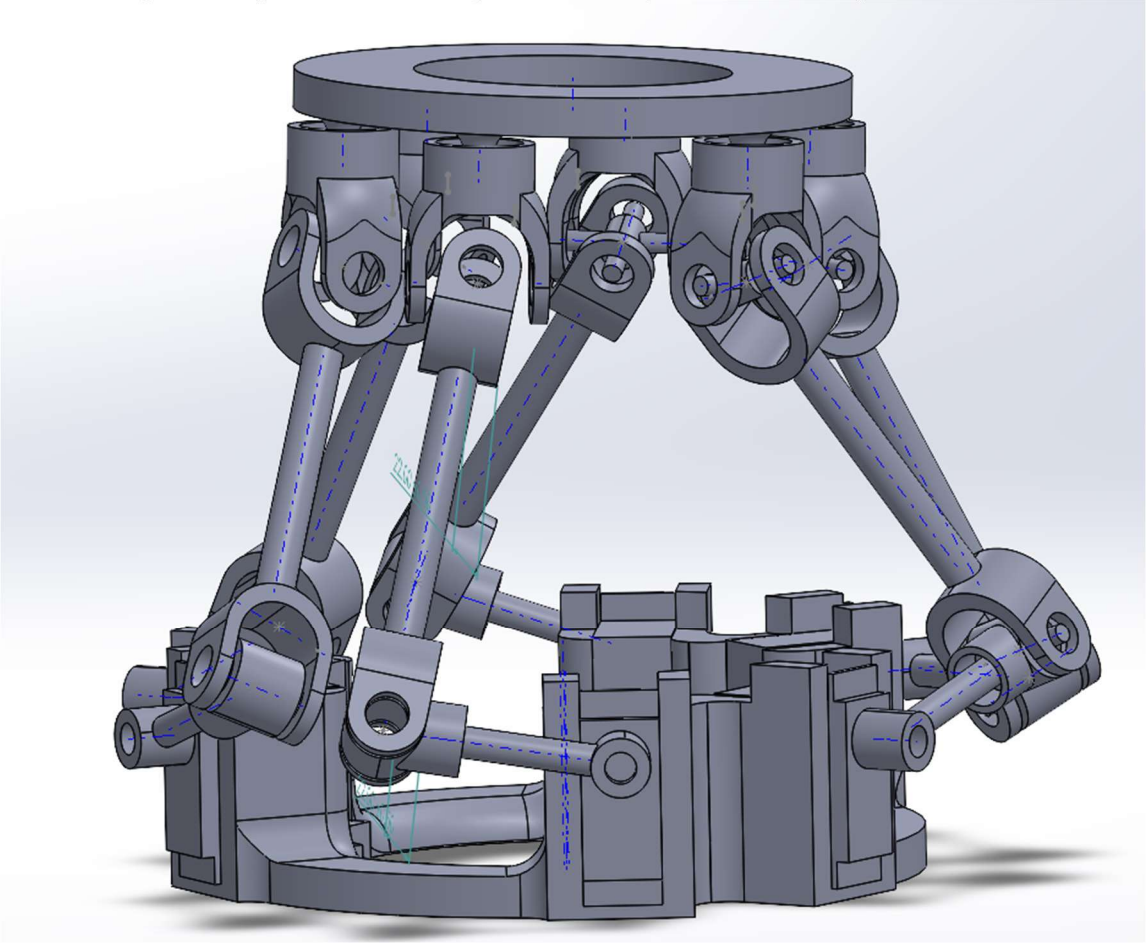
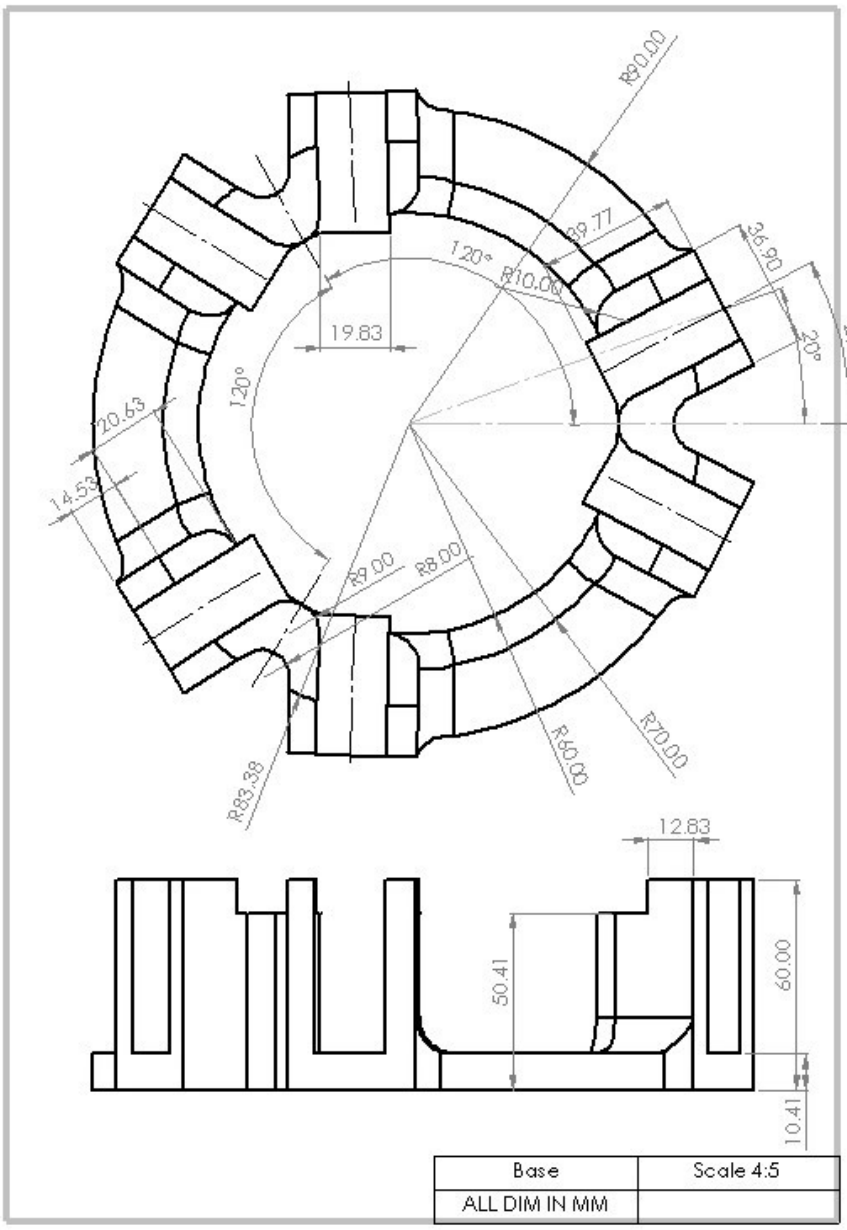


Fig 3-8 Assembly of Stewart Platform on CAD

In this model, the 3DOF freedom joint is a universal joint that has been given an extra degree of freedom by allowing it to rotate about its axis. We achieved that by attaching each yoke of the universal joint to the moving platform with 608-2 RS bearings.
Following are the individual drawings of each of the parts:
(All dimensions are in mm)

3.3.1 Base



Z

Fig 3-9 Base of the Stewart Platform

The above diagram shows the lower base or the static base of the slave mechanism of our Stewart platform. It carries great significance because the locations of each motor and its point of attachment with the crank are crucial for inverse kinematics. All the motors that are supposed to be placed in the base are supposed to be placed at 20 degrees angle to the 120-degree symmetry lines and they are also revolved around the revolution point at an angle of 8 degrees. The aforementioned settings of angles are done to optimize the reverse kinematics from our input points (the motors being the inputs or actuators) that is supposed to be used when making the control algorithms of the Stewart platform.

3.3.2 Upper Yoke of 3DOF joint

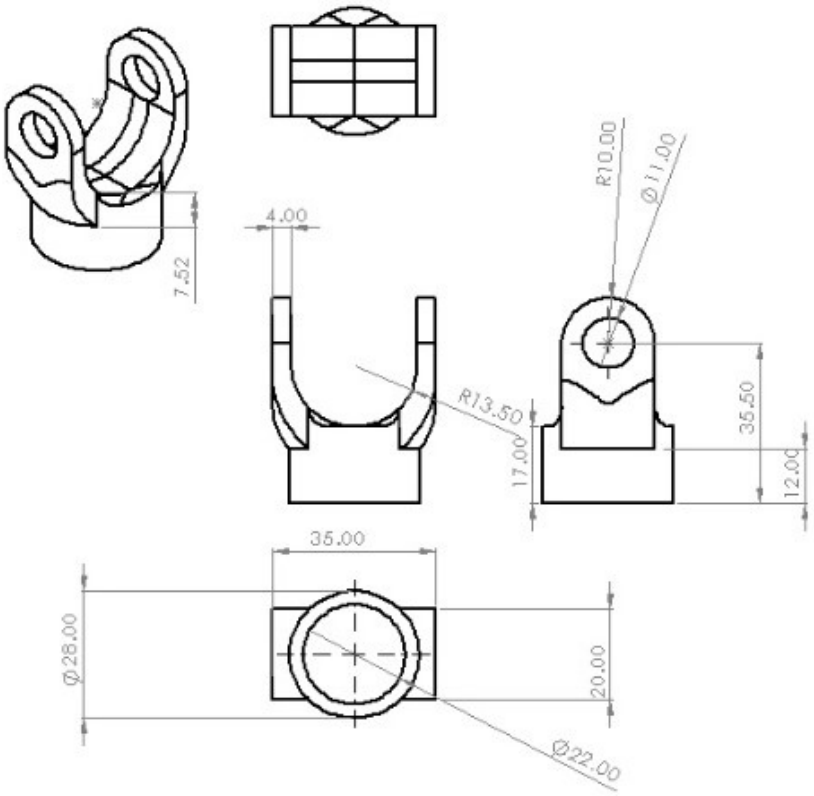


Fig 3-10 3DOF Joint

In the drawing above, the upper yoke is shown. This is the part that is placed between the coupler arm and the upper base as a connection and since there are six links between both platforms, six of these are used. The base of the yoke is coupled with the top platform with the help of a ball bearing allowing the yoke to rotate about its own axis. The legs of the yoke or the parts that are made like prongs are supposed to fit a bearing in between the holes that are present in them. These bearings are then connected to shafts or keys that when connected to a core and subsequently to another yoke, become a 2 DOF universal joint. Therefore, this joint is capable of a total of 3 DOF. The yoke that is supposed to join with the yoke shown above is the upper end of the coupler arm.

3.3.3 Moving Platform

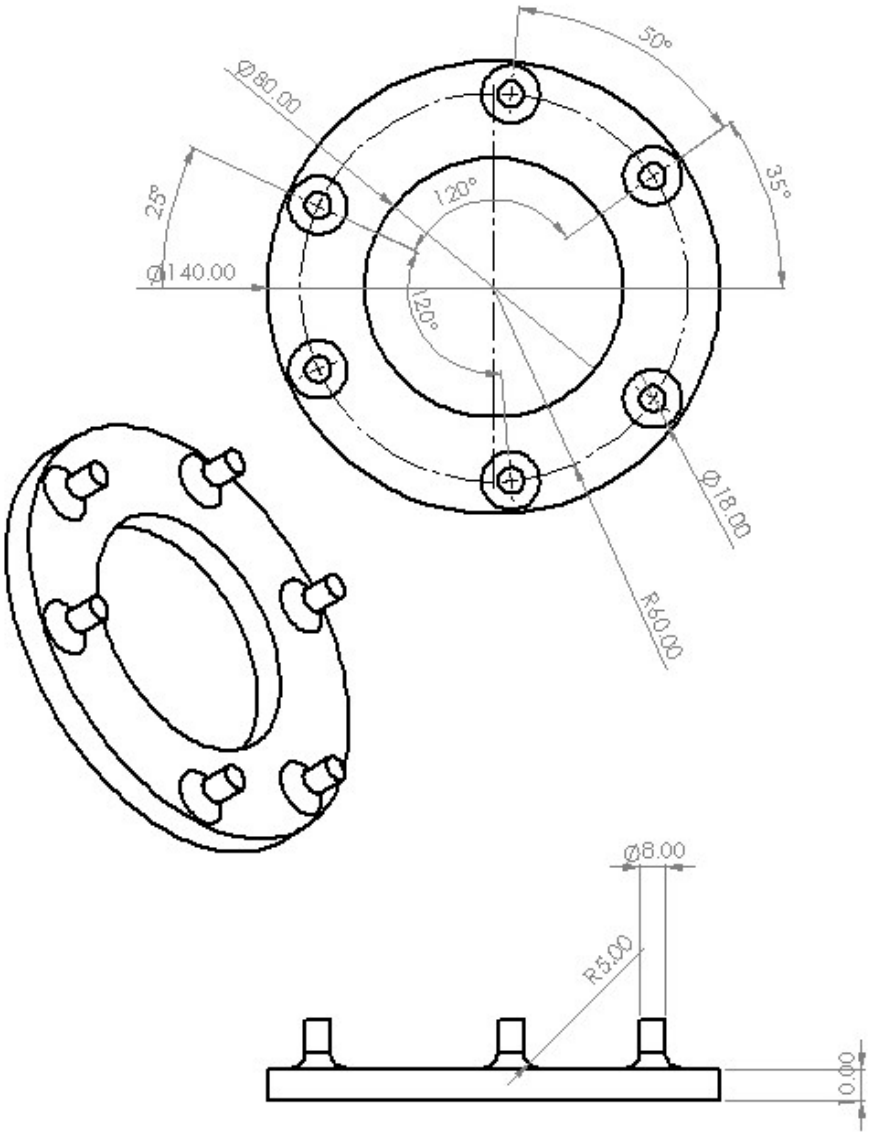


Fig 3-11 Top Platform

The part shown in the diagram above is the moving platform or the upper base of the Stewart platform. This base is supposed to carry the end factor that is to be inserted into the test subjects' mouths after the GUI of the platform is perfected. The pins that are shown in the diagram are the pins that are supposed to support the bearings that would eventually be the foundation of the upper yoke so that they can rotate. The configuration that is shown above is the upper half of the 6-6 configuration, which is the type (or more particularly configuration)

of the Stewart platform that we have made. In this configuration, a pair of pins is symmetrically laid about the 120-degree symmetry line at an angle of 25 degrees.

3.3.4 Lower Arm

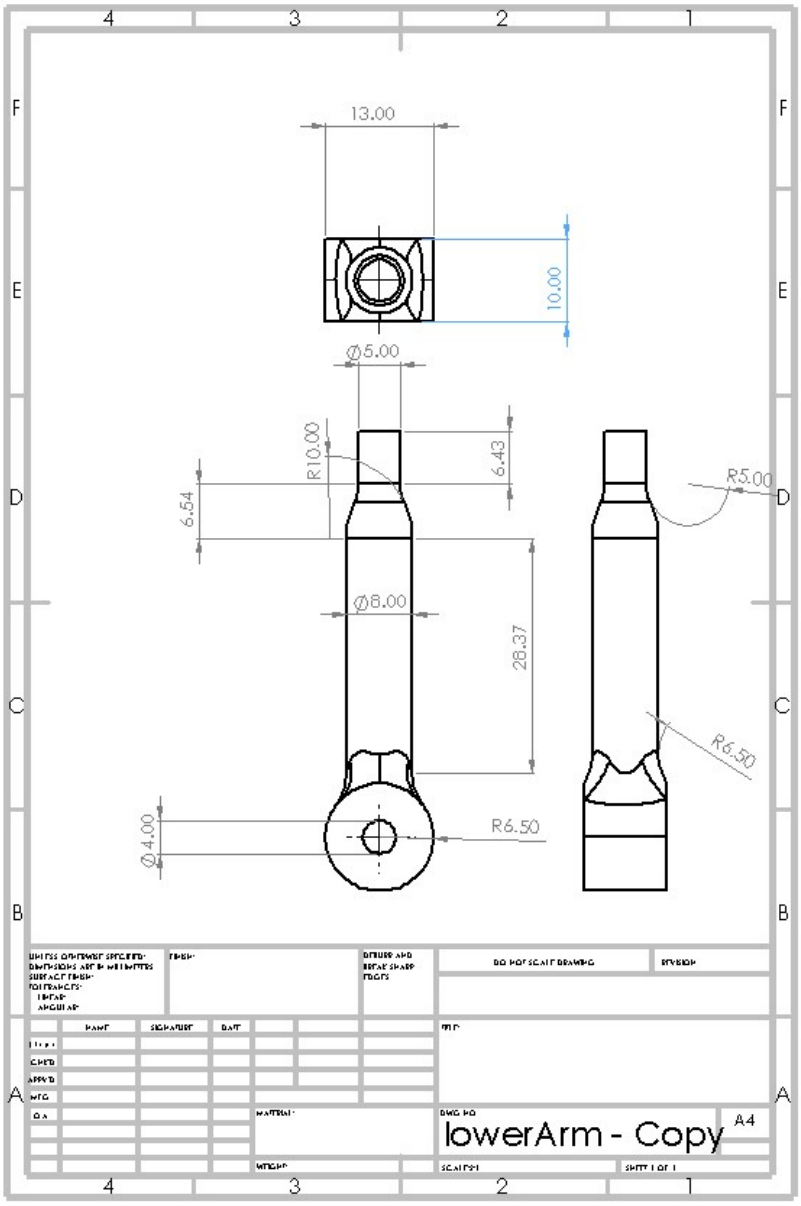


Fig 3-12 Lower Arm

The lower arm, or more functionally called the moment arm of the motor is what is supposed to be placed in between the servo motor (our actuators) and the coupler arm. It is this part that is responsible for the transmission of motion. It plays a very functional role in the assembly since the head (the broad circular part with the hole in between) cannot afford to slip or otherwise it would affect the control of the whole platform. The pin of the coupler (the narrower part) is supposed to fit inside a bearing that will fit into the lower part joint (will be shown below) which, together, will make the lower half of the rotary joint that is to provide 2 DOF to a single link.

3.3.5 Upper Arm

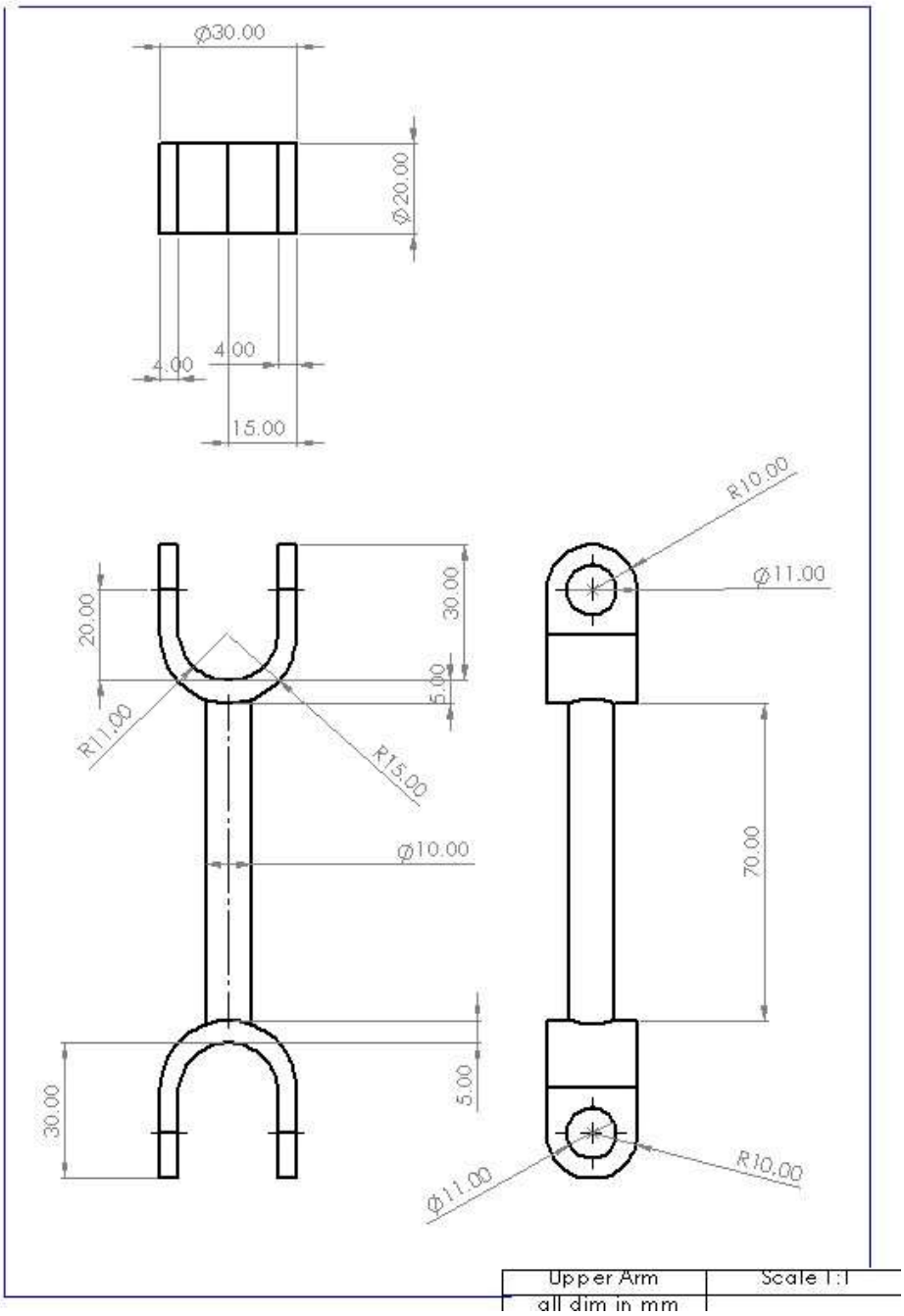


Fig 3-13 Upper arm

The upper arm or more functionally called the coupler in this paper is easily identified as the beam that has yokes or a pair of prongs on both ends. The coupler arm is supposed to connect the moment arm to the upper yoke (that can also be considered as a part of the universal joint which is further connected to the upper platform). On the upper end, with the upper yoke, the coupler arm makes a universal joint that with the help of another bearing acts like a spherical joint hence providing 3 DOF. At the lower end, with the help of the part mentioned as the "lower part joint", the coupler arm creates a rotary joint hence providing 2 DOF.

3.3.6 Lower Part joint

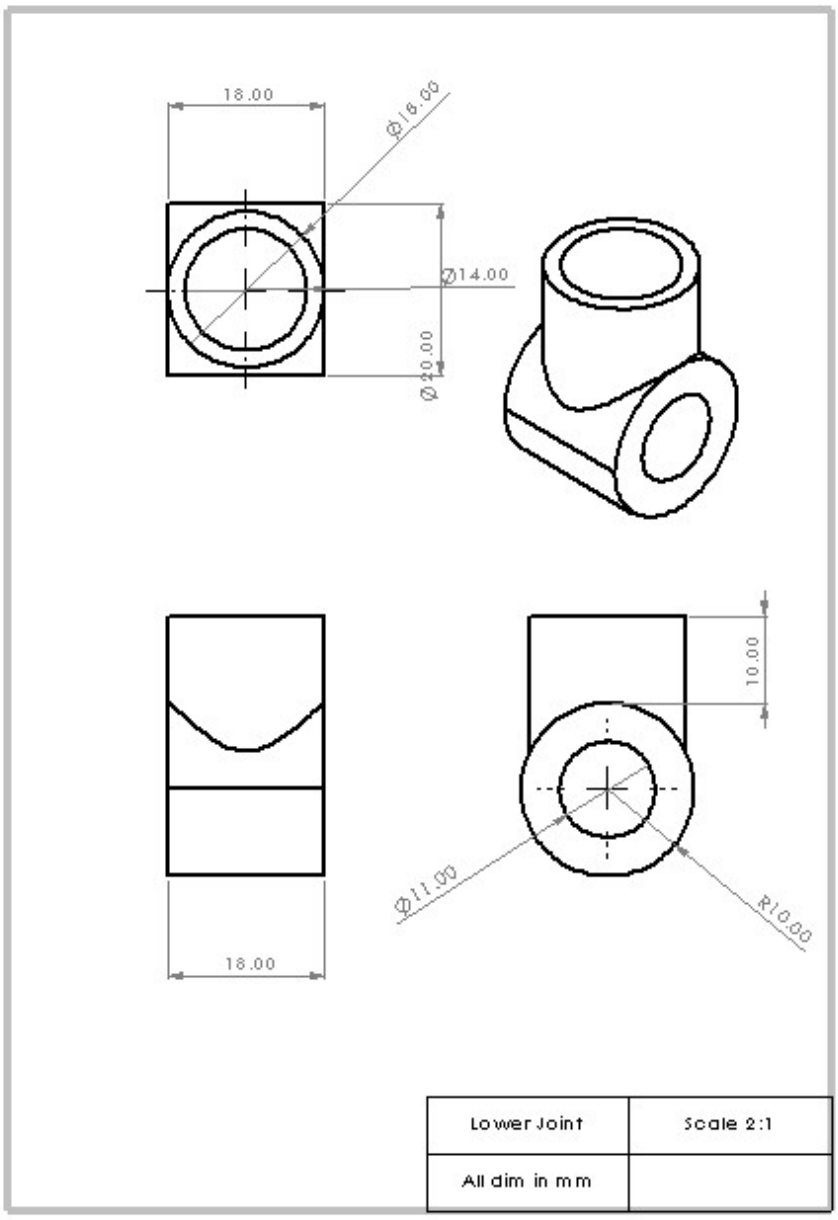


Fig 3-14 Lower part of 2 DOF joint

The lower arm joint plays a vital role in our geometry since it connects the moment arm and the coupler arm to make the revolute joint that is much needed for the Stewart platform to stay sturdy and be controlled. Through one end it fits a bearing and gets attached to the moment arm. Doing this gives the joint the ability to rotate about its own axis and doing this takes away any hardship that is to be faced by the moment arm during rotation. From the other two openings, it fits two bearings and becomes the lower part of the revolute joint that is made with the help of the coupler arm.

3.3.7 Probe Back

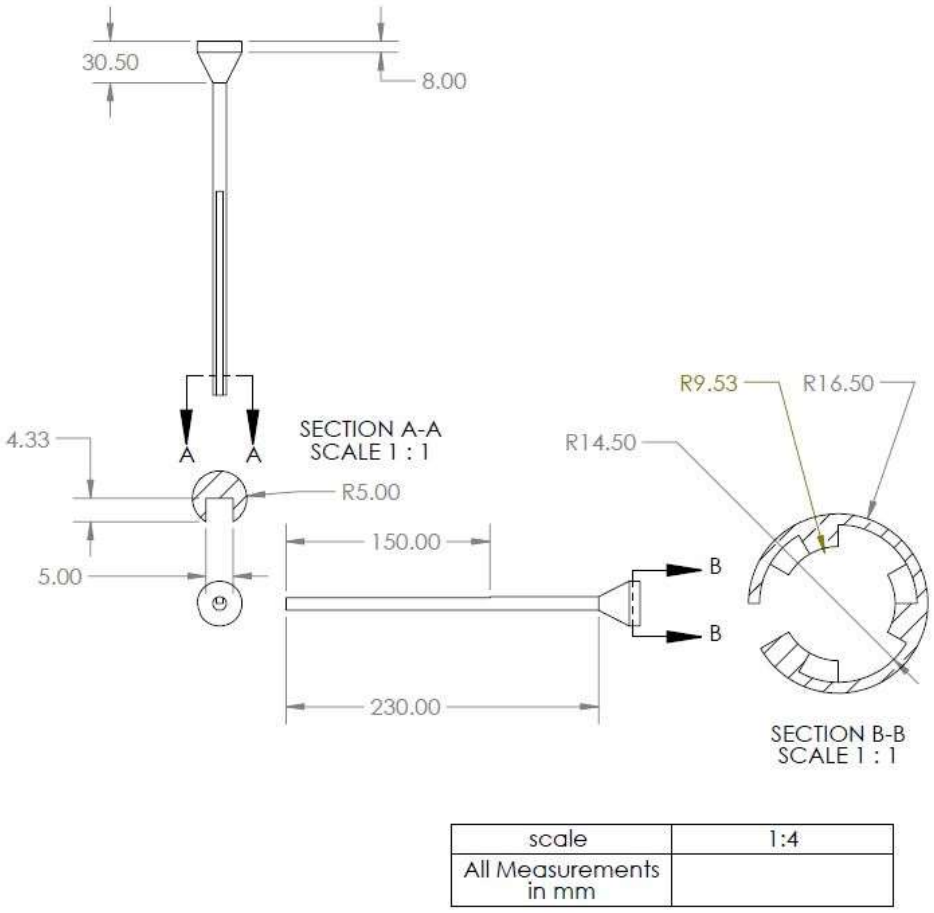


Fig 3-15 Probe back

The probe back is another part that plays a very major role for the robot. The objectives kept in mind while designing the part are as follows:

- The probe back can be utilized to actuate the end factor along with the complete probe assembly.
- The probe back should be able to accommodate a space for a Force Sensing Resistor (FSR) so that it can be used to detect the force that is being applied on the swab.

The front of the probe back has two major geometric features that acknowledge the functionality of the whole part: 3 symmetrical clamps are designed about to assemble the probe completely and around it is a support that is supposed to physically support the clamps from failure. Secondly, a small cylinder has been cut into the frontal surface area of this part to accommodate the FSR. The FSR is supposed to identify the force that is axially being applied on the swab so in case of an emergency, if the force applied is near a magnitude that statistically

identifies as discomforting, the motor can be programmed to immediately recall the end factor, so the patient remains safe.
The shaft of the probe back is designed to be of 10 cm in diameter which is just as wide as needed: the part is thick enough to not break and the part is thin enough to not be much of a load. The rectangular cross section that is cut into the cylinder is supposed to accommodate the aluminum rack which, along with the pinion, powers the probe to move forward and backward.

3.3.8 Probe Head

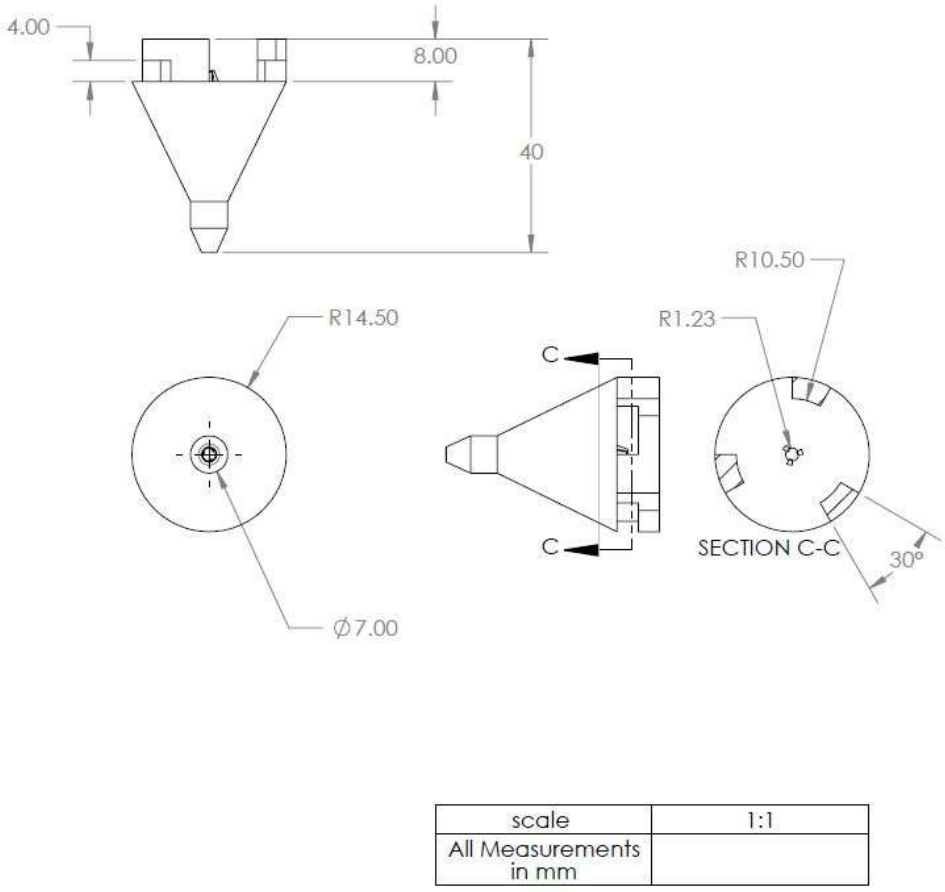


Fig 3-16 Probe Front

The part that is shown in the figure shown above was designed to carry and hold the end factor (swab) in place for when it is inserted into the mouth of the patient.
On the base of the probe head, there are three symmetrically positioned clamps that are used to grasp on to the other half of the probe; '*the probe back*'. From the front, the part of the head that resemble a nozzle is supposed to hold the swab into place and the clamps that are smaller in size placed concentrically in a circle as the bigger clamps are supposed to hold the end factor together for when it physically contacts the oropharynx.
Conclusively, the probe head is supposed to be the part that is nearest to the mouth of the patient (just behind the swab), and it plays the role of keeping the swab as stable as possible.

3.3.9 Probe Base

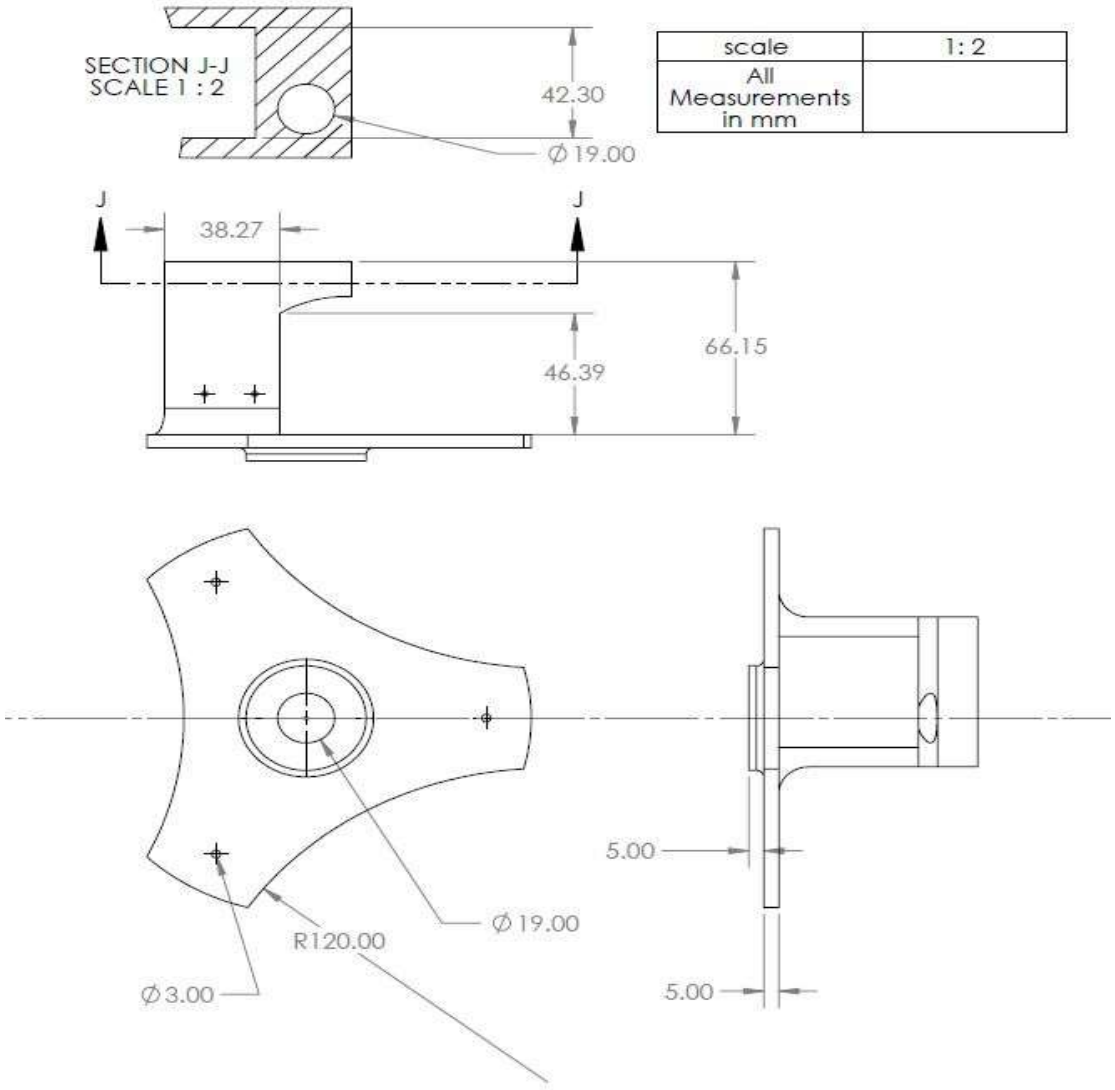


Fig 3-17 Probe base

The base of the probe is a platform that is to be mounted on the top platform that was explained earlier. This platform plays the role of a support for the probe, the linear bushings and the motor that is supposed to power the probe.
This base is based on a triangular design that can be mounted on the top platform with the help of screws for maximum strength. A clamp is positioned on top of it to support the motor which is also contains holes for screws to grip the motor. On the clamp and the center of the triangular base, there are two concentric holes that are supposed to accommodate two linear bushings that will enable smooth motion for the probe.

3.3.10 Gear Train

The gear train that has been used for the probe mechanism acts as a linear actuator and is supposed to enable the linear motion of the probe. The gear train includes two parts:

Gear (Pinion)

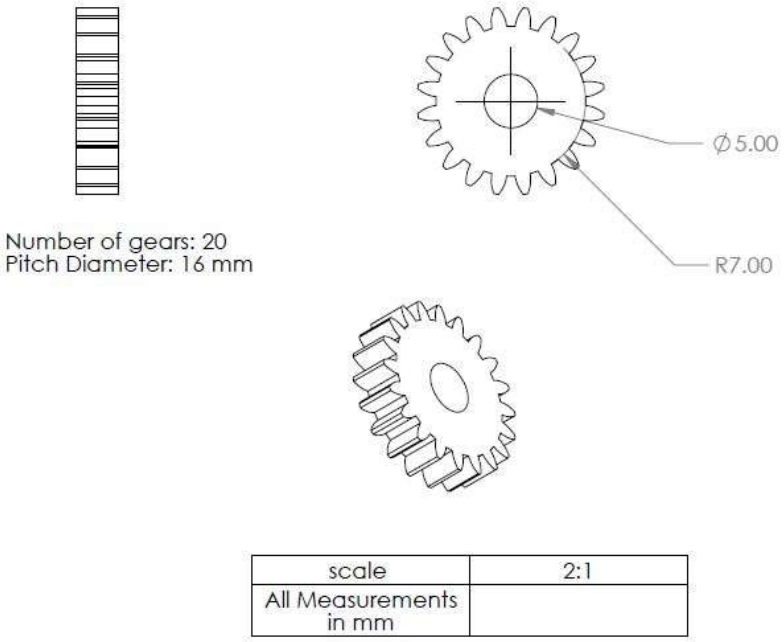


Fig 3-18 Pinion Gear

The gear that is used in this train is fabricated through machining of an aluminum disk of a thickness of 4 millimeters. The module of the gear is 0.8 and the number of teeth is 20. The detailed data for the design of the gear is as follows:

Table 3-1 Pinion Gear

Nominal Shaft Diameter	5 mm
Pitch Diameter	16 mm
Module	0.8
Number of teeth	20
Material	Aluminum
Pressure Angle	20 degrees

Rack

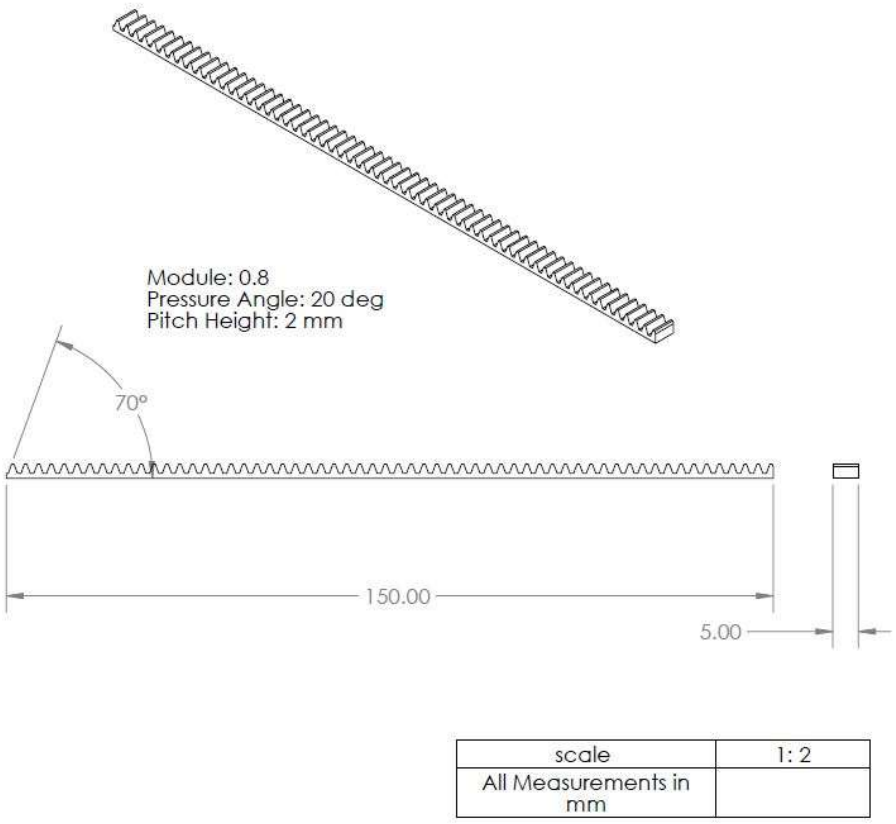


Fig 3-19 Rack

The rack that is fabricated is also made of aluminum. The fabrication of the rack just like the gear cannot be done by 3D printing from PLA. This option was opted because of the lack of tensile strength in 3D printed parts. While keeping the weight of the assembly a constraint as well, especially in case of the rack, the material selected was aluminum.

Table 3-2

Length	150 mm
Module	0.8
Material	Aluminum
Pressure Angle	20 degrees

3.3.11 Linear Bushing

To support the probe that carries the end factor to the oropharynx and to give it smooth motion without any vibrations being produced from PLA-PLA contact, we had to adjust a couple of linear bushings into the assembly that aims to provide the motion of probe. The linear bushing that we chose has the inner diameter that was sufficient to support the probe back. This probe back carried the aluminum rack that, along with the pinion, made the mechanism similar to a linear actuator powered by a NEMA 17. The model number of the linear bushing used is LM10UU.

Table 3-3

	LM10UU
Inner Diameter	10 mm
Outer Diameter	19 mm
Length	29 mm
Bearing type	Linear Bushing

3.3.12 Bearings

In the making of these parallel mechanisms, several joints with various degrees of freedoms were needed. In conventional construction of the Stewart Mechanism in the RUS configuration, usually the motion is transmitted through the help of spherical joint that adds 3 DOF to the individual link in which the joint is being added. Other than that, Revolute joints are use as well and other than that, Universal joints are also used adding 1 and 2 DOF to one link respectively.

In our case, usage of a spherical joint was out ruled due to the complications that can occur due to its assembly, so we opted to make a make-shift spherical joint. This was achieved by integrating a yoke of the universal joint that we are using, that incorporates 2 DOF to the whole link, along with a bearing that is used in such a way that it allows the shaft that is joined to the yoke's key to also rotate about itself. By doing this, we achieved the 3 DOF in our link that we originally gave up due to not using the spherical joint.

Stating that the bearings have single handedly achieved all of the 6 DOFs that we wanted to incorporate into each link is also justified. Four bearings have been used in each 2 DOF Rotary joint that has been made and five of them have been used in all of the make-shift Spherical joint that we have made. This spherical joint, as explained above as well incorporates the parts of a universal joint as well as the bearing that is supposed to allow the shaft to rotate and add the third DOF. Other than these two, revolute joints have also been used in all the links.

Table 3-4: Bearing Data

	608-2 RS	1150 ZZ	605 ZZ
Inner Diameter	8mm	5mm	5mm
Outer Diameter	22mm	11mm	14mm
Bearing type	Ball Bearing	Ball Bearing	Ball Bearing

Table 3-5 : Number of Bearings for each joint

	608-2 RS	1150 ZZ	605 ZZ
3 DOF universal joint	1	4	
2 DOF Rotary Joint		3	1

3.4 MALTAB simulation of the inverse kinematics

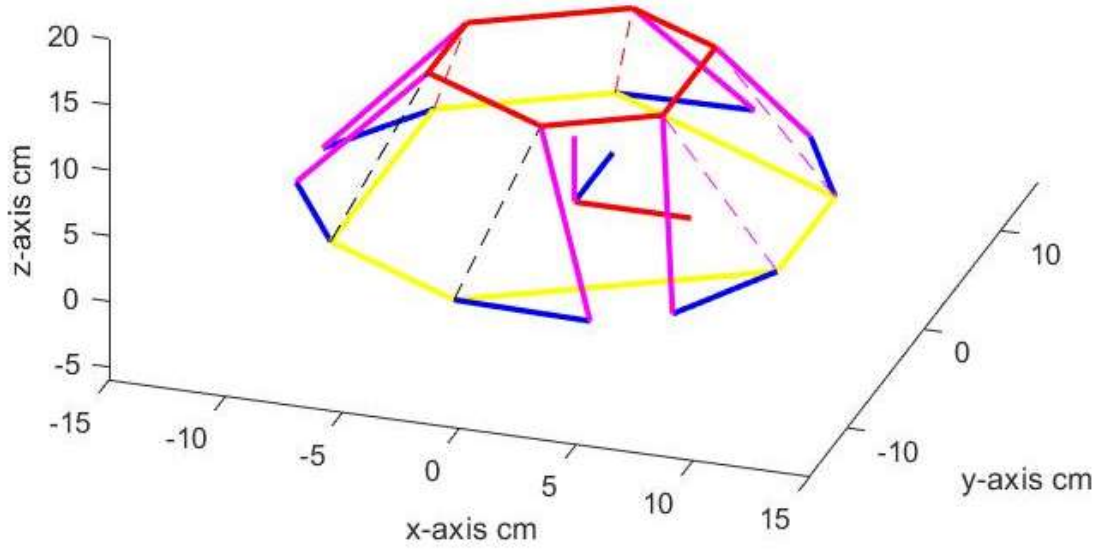


Fig 3-20 Neutral Position

To verify the accuracy of the inverse kinematics equations we created a MATLAB code of the simulation and plotted the Stewart platform. Fig 3.20 shows the platform when the yaw, roll, and pitch of the platform are zero and the platform has not moved from its neutral position at all. At this position, the motors are at a rotation of zero degrees.

Then we provide an angle of rotation about the x-axis of 30 degrees to the platform. To do that we must change the value of the angle associated with rotation about the x-axis in the rotational transformation matrix shown in equation 3-2. The equation 3-1 repeated in the following is being used to remind us of each angle and its associated angle of rotation.

$$[{}^{B}_{T}R] = rot(z,\psi)rot(y,\varphi)rot(x,\theta)$$

Therefore, we must alter the value of the angle θ to rotate the top platform about the x-axis by 30 degrees, as shown in the following table

Table 3-6

Angle	Value-degrees
ψ	0
φ	0
θ	30

However, the values of R_T remained the same, i.e., $R_T = [0,0,10.4]$. As a result, the angles α_i calculated by MATLAB are:

Table 3-7: values of $\boldsymbol{\alpha_i}$

ith leg/joint	Values of α_i/ degrees
1	18.295
2	31.5561
3	11.7137
4	-13.2145
5	-28.8626
6	-15.2762

Fig 3-21 shows the MATLAB plot of this result. We can clearly see the rotation of the top platform about the x axis.

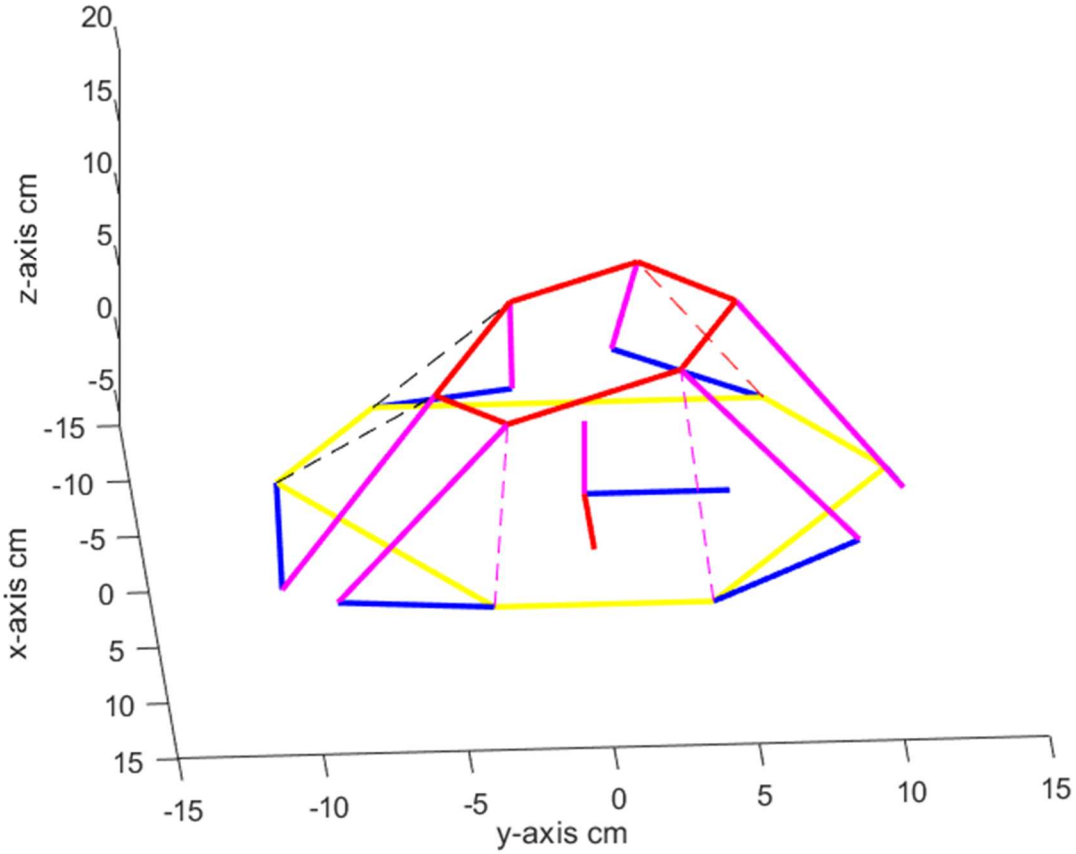


Fig 3-21 Rotation of top platform about x-axis by 30 degrees

To verify our equations for rotation about the z-axis we conduct we used a value of 15 degrees yaw, since that is almost the maximum value or our desired workspace. Fig 3.22 shows the platform after 15-degree rotation about the z-axis (yaw), for which the angles of rotation of the moving platform are listed in the table below while the vector R_T remained the same, i.e., $R_T = [0,0,10.4]$.

Table 3-8

Angle	Value-degrees
ψ	15
φ	0
θ	0

The angles α_i according to the inverse kinematic equations computed by MATLAB are:

Table 3-9

ith leg/joint	Values of α_i/degrees
1	6.7687
2	-2.6593
3	6.7687
4	-2.6593
5	6.7687
6	-2.6593

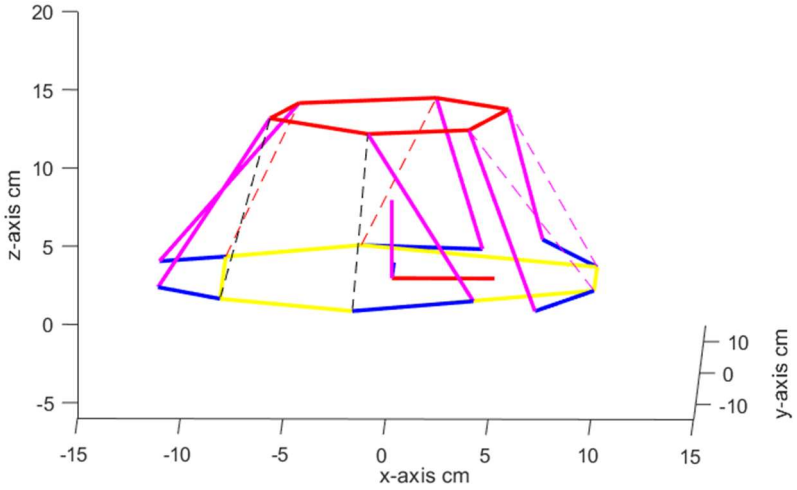


Fig 3-22: 15-degree about z-axis

Fig 3.23 shows the platform after a 3.6 cm translation upwards from its initial position following 10-degree rotation about the z-axis. This means that the vector $R_T = [0,0,14]$, and the angles of the moving platform:

Table 3-10

Angle	Value-degrees
ψ	10
φ	0
θ	0

As a result, the angles α_i are:

Table 3-11

Values of i	Values of α_i/degrees
1	36.7515
2	32.4385
3	36.7515
4	32.4385
5	36.7515
6	32.4385

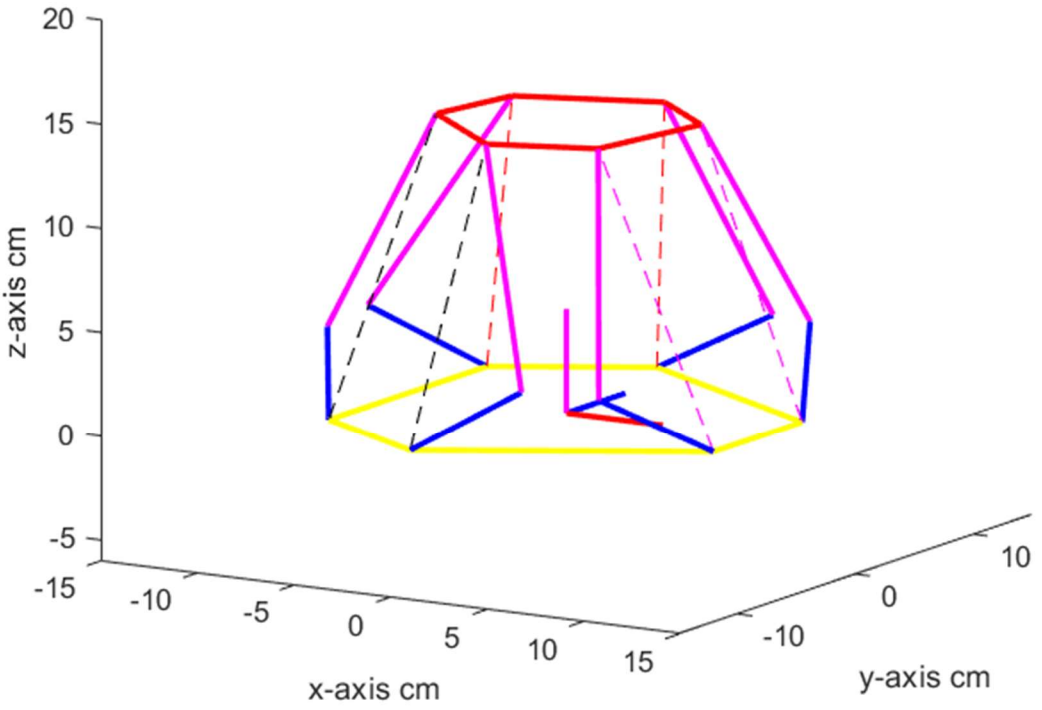


Fig 3-23 Resultant Position

3.5 MSC Adams Simulation of the Inverse kinematics results

After verifying the Inverse kinematics results by plotting them in MATLAB, we used MSC Adams to verify whether our CAD model behaved identically or not. We needed to first make sure that we had not unintentionally introduced a redundancy in the model or not, and second to verify again using MSC Adams whether by using the inverse kinematics we were achieving the desired motion in the CAD model or not. We needed to do this because the MATLAB plot is not enough to check if the model obeys the constraints or not.

We first used the export to Adams option in Solid works to import the complete assembly of our platform to MSC Adams with all its constraints intact. Then we added the force of gravity in the negative z direction as shown in the figure 3-24.

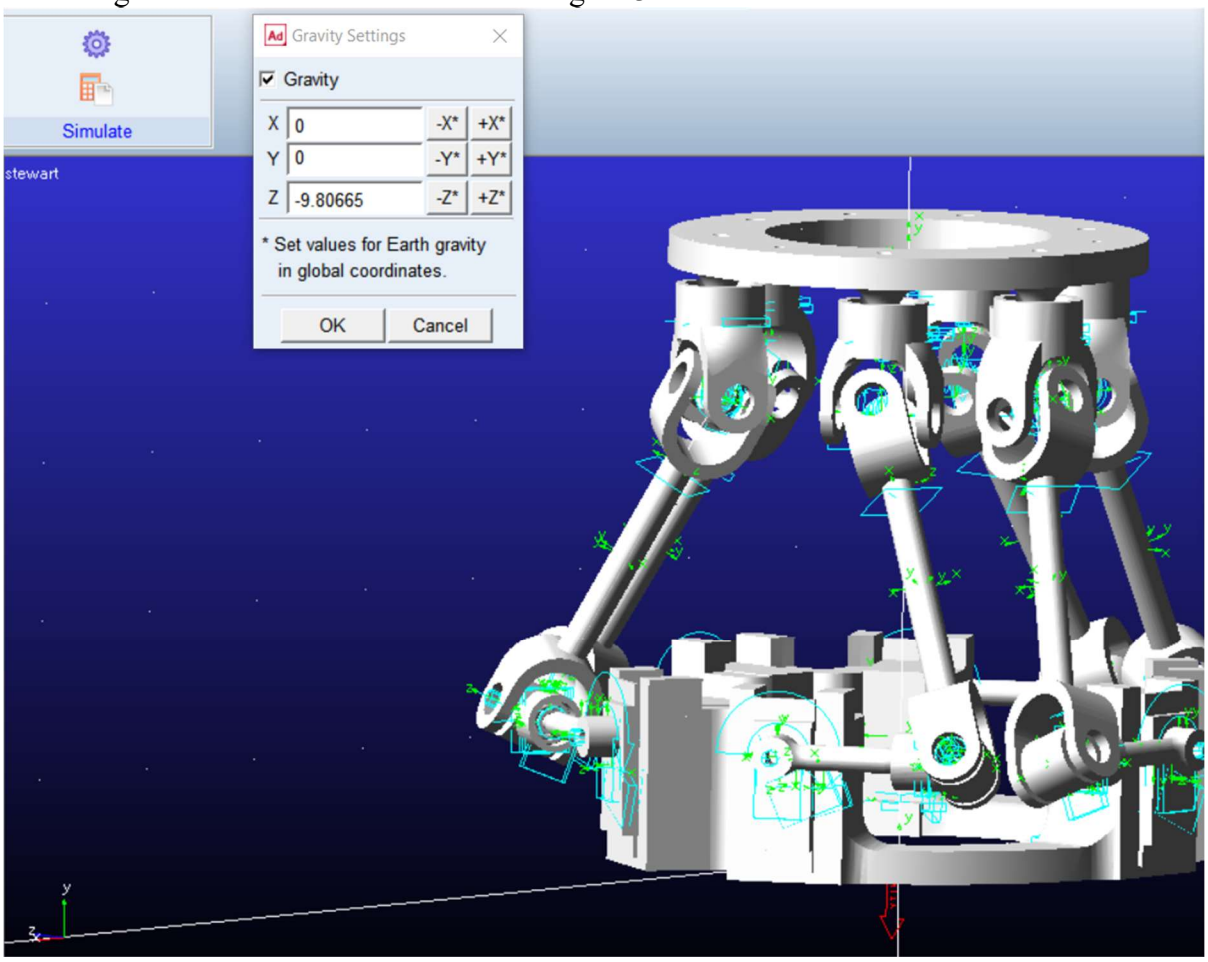


Fig 3-24 MSC Adams Simulation

As the parts will be manufactured by 3d printing using Polylactic Acid(PLA) filament and PLA has a density of 1250 kg/m^3 [15], we used the "geometry and density" setting in the Modify body menu and attributed the density of PLA to each part of the model. Since we are using the "geometry and density setting" (figure 3-25) the mass and moments of inertia are calculated by the software using the geometry of the part in question as shown in figure 3-26.

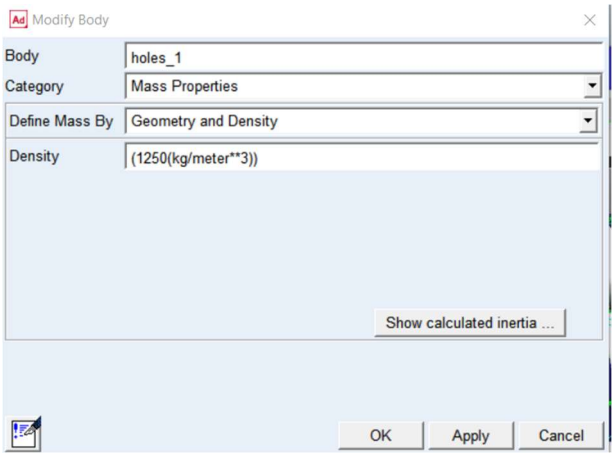


Fig 3-25 Modify Body

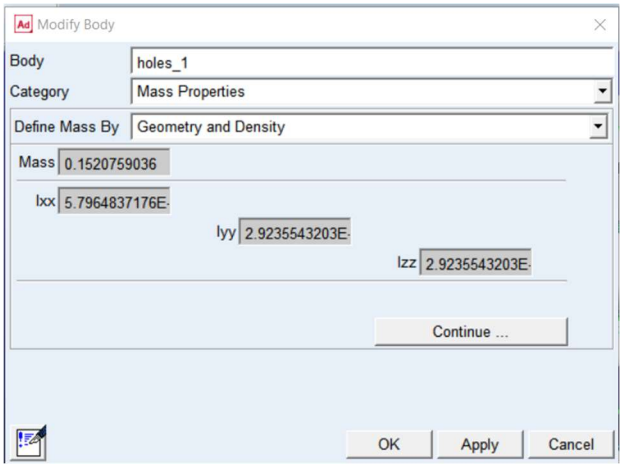


Fig 3-26 Modify Body

Each leg of our model has a RUS (Revolute-Universal-Spherical) configuration, and the model needs to be actuated at all six of the 1 DOF revolute joints in the legs as shown in the figure 3-21. We used a linear function and input the initial and final values of α_i for initial and final values of time. We use the inverse kinematics equations outlined in section 3.2 and coded in MATLAB as shown in appendix A and after calculating the values of α_i for a desired motion of the platform, we input them as the final values.

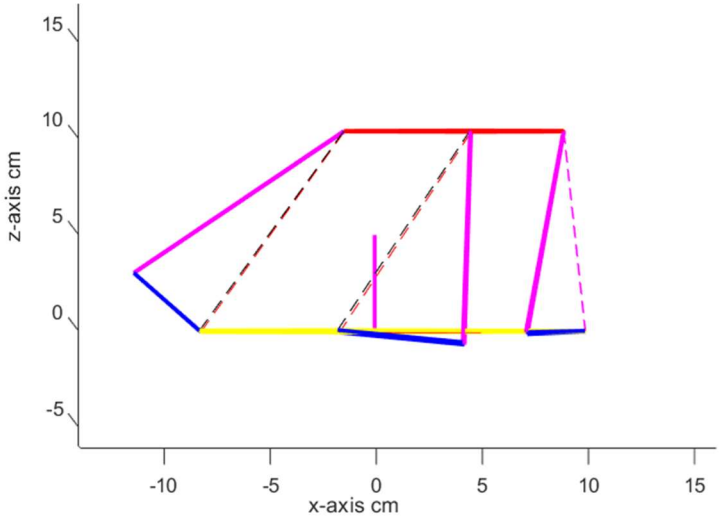
In to verify our code, we used a translation of 4cm from the x-axis while the platform was at its home position. For the initial position the values of all α_i are 0 and for the final position when the platform is at x=4, y=0 and z=10.4 the values of α_i are given in the table 3-12.

Table 3-12 values of α_i for 4cm translation along x-axis

ith leg	α_i/Degrees
1	-0.9932
2	-6.1354
3	30.9315
4	30.9315

5	-6.1354
6	-0.9932

The figures 3-27a and b show the side and top view of the MATLAB plot for the said translation:



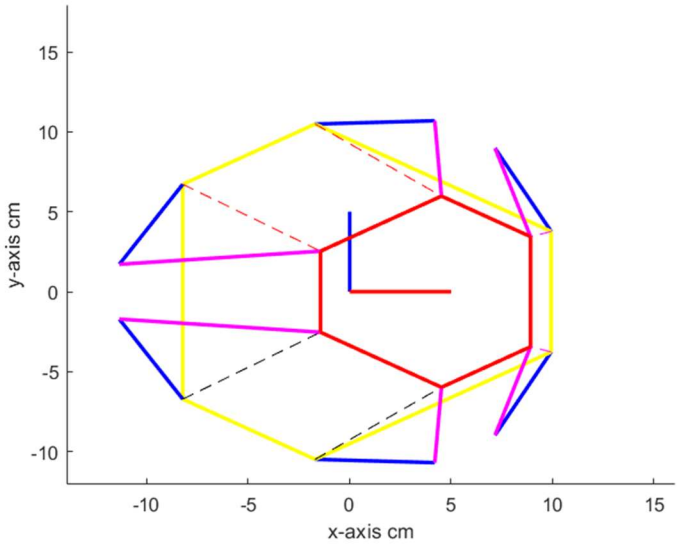


Fig 3-27a, b

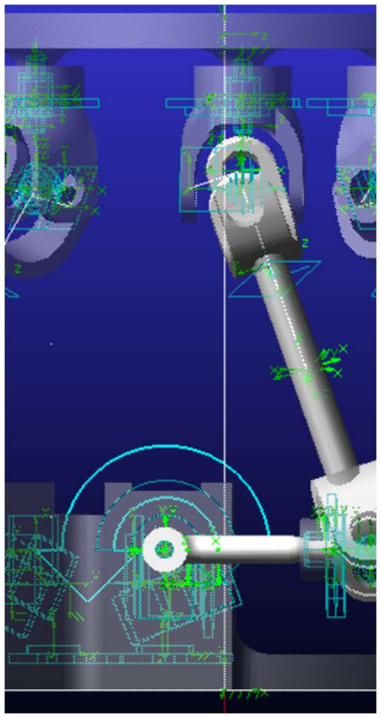


Fig 3-28 Rotational Joint Motion

The values given in the table 11 are then input into the MSC Adams rotational joint motion menu manually, where each joint is provided its respective initial and final value as shown in the figure 3.29 for the first leg. The Rotational Joint Motion is applied to each revolute joint as shown in the figure 3-28.

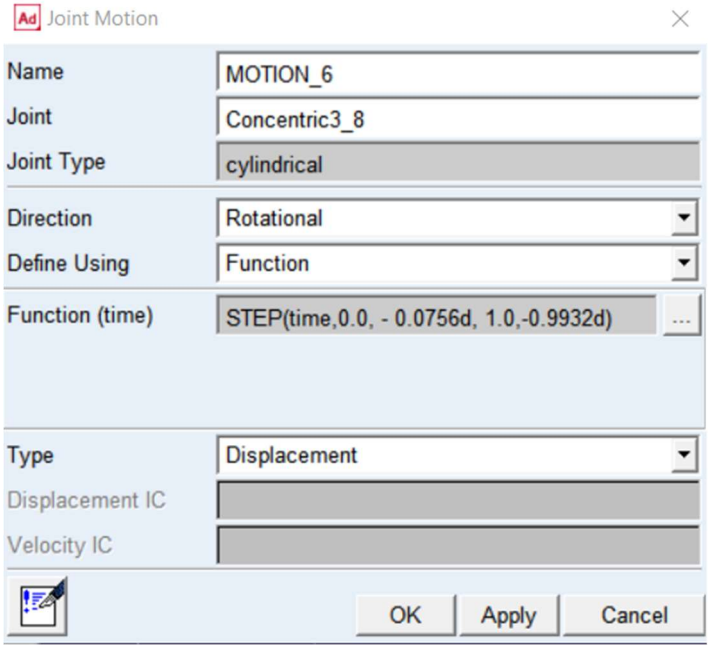


Fig 3-29 Joint Motion Prompt

As a result, the joint of the first leg for which the initial and final angular values are given in the figure 3-29 and is shown in figure3-28 moves in way shown in figure 3-30.

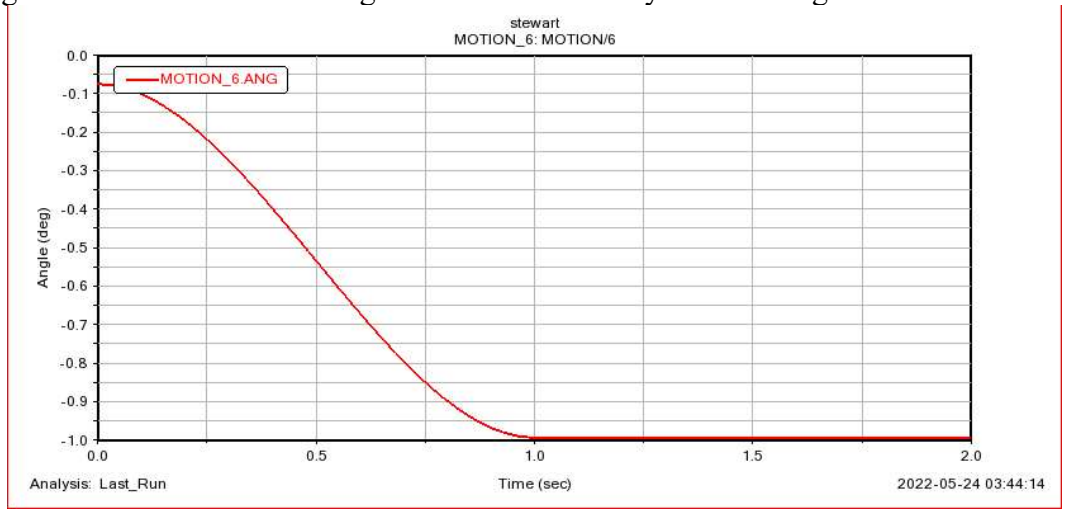


Fig 3-30 Results

The final position of the platform, after the appropriate initial and final conditions from the table 3-12 have been applied to the corresponding legs, should be 4cm in the y direction, 0 cm in the x direction and 10.4 in the z direction. When we run the simulation, we obtain the following results:

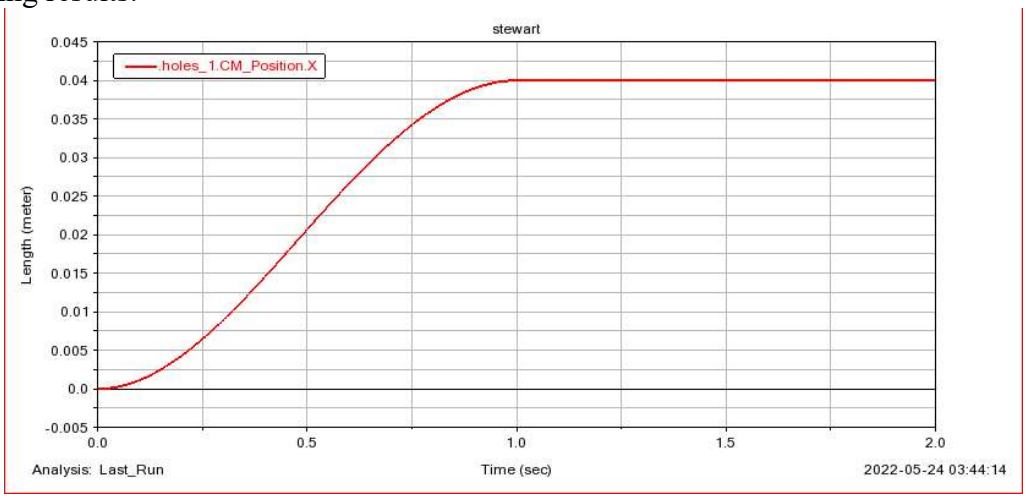


Fig 3-31 4cm translation along x-axis

Where the vertical axis of the fig 3-31 represents the distance of the Centre of Mass of the top platform along the x axis. The corresponding snapshots of the initial and final position of the platform are given in the figures 3-32(a and b) and 3-33(a and b) respectively, where the snapshot a of each figure depicts the top view, and the snapshot b depicts the side view. For brevity's sake we have not include the result of each and each possible motion in this report, however, they can also be simply obtained by calculating the final values of alpha and placing them in the joint motion function in Adams. If we compare the plots in figure 3-27 to the simulation results in figure 3-32, we can see that the results also look very much the same.

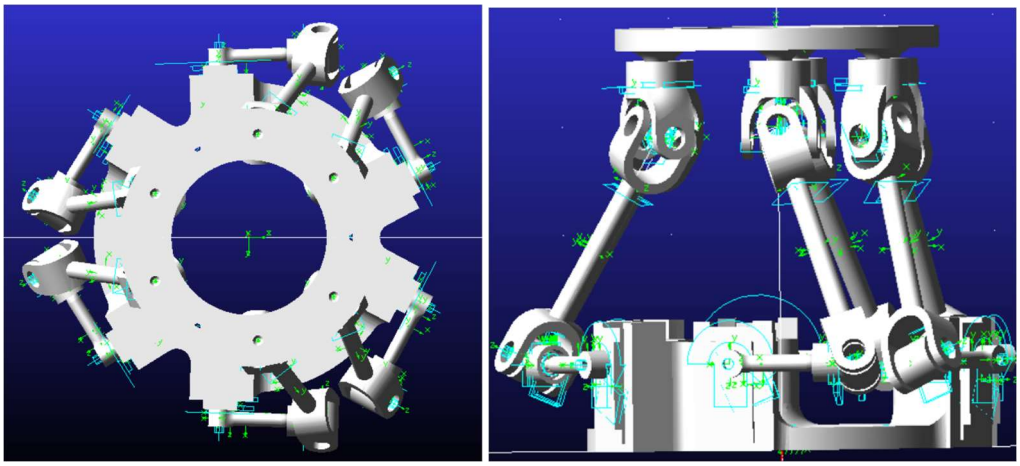


Fig 3-32 a, b

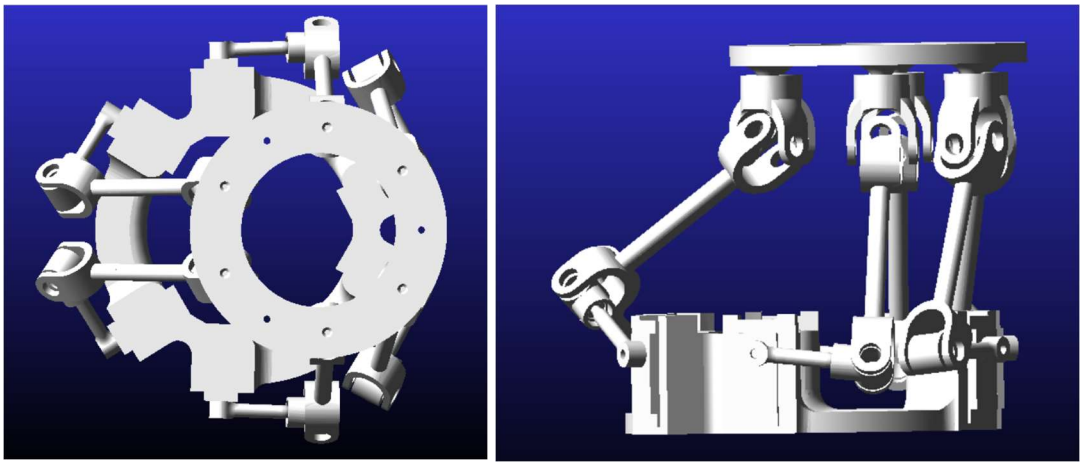


Fig 3-33 a, b

3.6 Motors

We are using Tower Pro 996R motors. The motors need to be selected in such a way that they could support the weight of the platform, therefore the stall torque of the motors should be high enough for the apparatus. Our platform has six legs so six individual motors need to be installed, and the torque will be distributed across the six motors as well.

The MG99R has a stall torque of 11kg/cm according to the manufacturer. 11 kg per cm is equal to 1.0791 Nm of Torque. Our lower arm length is equal to 4cm; therefore, the upper arm needs to exert an orthogonal force of 26 N for the stall torque to be achieved. Moreover, all together it means that the motors can support a weight of 161.865 N which is equal to 16.5 kg. Since our model weighs 660g when 3D printed using PLA as its material, the motors can be seen to be providing more than sufficient torque for our apparatus.

3.7 Summary

We have outlined the inverse kinematic equations of the platform, which are used to calculate the angle of rotation of servo motors. This was achieved by first calculating the link lengths of

the actuators that are being used through the inverse kinematics which was then used to formulate the inverse kinematics for the specific angular positions for the specific orientation required. Moreover, the design requirements have been outlined as well as the CAD model generated from the requirements. This particularly included the preplanned integration of simplistic implementation of inverse kinematics for our ease. Furthermore, the design process also included the designing of the legs which were to have 6 DOF each so that the moving platform could move in 6 DOF as well without any hindrance. The simulation and analysis done were also outlined.

4 Physical Model Development and Instrumentation

4.1 Manufacture of Parts

When we were manufacturing, our requirement was that our manufacturing method should be able to create complex parts and should not be the limiting factor in our design. Subtractive manufacturing has its constraints and produces much more waste as compared to additive manufacturing which can be used to manufacture virtually any design. Due to its flexibility all the parts of the Stewart platform other than the motors and the bearings were manufactured by a type of additive manufacturing called fused deposition modelling – commonly known as 3d printing. The most used materials for 3d printing are Polylactic acid and Acrylonitrile Butadiene Styrene (ABS). Relative to ABS, PLA is more ductile, has greater tensile strength and is much easier to print, due to which PLA is the predominant polymer used in 3d printing. The ABS filament does have a melting point temperature of 200 °C, which is higher than that of PLA (180°C), however, we will only be operating at room temperatures, so a high melting point is not one of our design requirements. Given that we do not require our parts to bear exceedingly high stress loads, or high temperatures, we chose PLA as the material to print with due to the ease with which it can be printed.

Moreover, in addition to selecting the part we also needed to select the infill density of the parts that we printed. Each part had a different infill density based on the estimated load that it would have to bear. The top platform that was initially designed had an infill of 50%, and after it failed, we redesigned the part and increased the infill density to 100%. That means that the part was solid PLA. The base however only had an infill density of 20% as it did not need to bear any significant load and only needed to hold the motors in their designated places. The table 12 lists the final infill density of each part as named in the section 3.3:

Table 4-1

Part	Infill Density %
Base	20
Lower Arm	100
Lower Part Joint	80
Upper Arm	70
Yoke of 3DOF Joint	100
Moving Platform	100

Multiple 3D printers were used throughout the duration of the project since some of our parts failed after being assembled and they were redesigned and re printed after their errors were diagnosed. Most of our parts were 3D printed from 3D printers operated by private businesses and one from the 3D printer that is present in the institute.

The print time of each part varied from a few minutes to even several hours depending on the size, complexity, and the infill density of the part. After all the parts have been printed, the lower arms will be mounted on the shafts of the servo motors and the upper arms will be linked to the other end of the lower arms hence creating a two degree of freedom revolute joint. The configuration has been explained clearly in chapter 3.

The joints: 6 2-DOF revolute and 6 3-DOF universal joints, all were made through a combination of bearings and keys that can be slotted in their preassigned positions in the design.

4.2 Instrumentation and Control

Servo motors need PWM signals to work. These PWM signals can be provided with the help of an Arduino or any other Microcontroller. We are using a Raspberry pi and a PCA 9685 which is a 16 channel PWM driver that connects to the Raspberry Pi via i2c. The reason that we are using a PCA9685 is that a Raspberry Pi 4B only has only one physical PWM output pin while the PCA 9685 has 16 independent PWM pins. The Raspberry pi communicates with the PCA 9685, using the i2c signal protocol, to indicate which PWM pin should send PWM of what duty cycle. And the PCA 9685 then sed PWM signals to the servo motors to turn them at their desired angles. This process is shown in the figure 4-1.

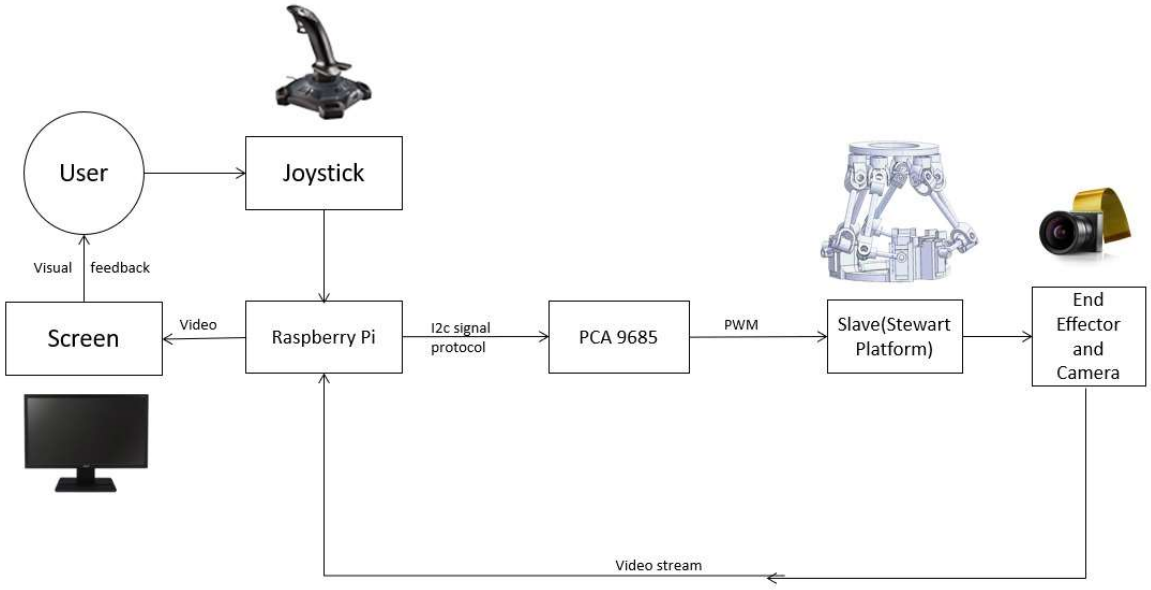


Fig 4-1 Control Loop

Raspberry Pi is superior to an Arduino in our case because it can easily connect to the internet via Wi-Fi or ethernet and has the capability to function as a server for IOT applications in addition to having a dedicated port for a camera. A Raspberry Pi acts as a separate computer, one that has its own operating system.

The operating system of an R-Pi first needs to be installed on an SD card that also plays the role of internal memory of the microcontroller. After the installation, the card can be plugged back into the device and the device is ready to use.

Essentially Raspberry Pi can be used through a screen (connected to a micro-HDMI port) and a separate keyboard and mouse that can be plugged in dedicated USB ports on the Microcontroller. In total, a standard model of a Raspberry Pi 4 B comes with:

i. Two USB 3.0 ports
ii. USB-C port for power supply
iii. 2 Micro HDMI ports that provide display up to 4K resolution
iv. 2-lane MIPI DSI Display port
v. 2-lane MIPI CSI Camera port
vi. A wireless Bluetooth 5.0 connectivity module
vii. 40 GPIO pins. GPIO is short for General Purpose Input-Output
viii. A Gigabyte connection port for a standard RJ-45 headed ethernet cable

Besides the micro-HDMI port, if a Raspberry Pi is connected to a network using either the LAN port on the device or by Wi-Fi, it can be accessed through a remote desktop.
A Raspberry Pi, once accessed, gives the ability to the user to do programming on the microcontroller. In a scenario of controlling a robot, python is preferred. This language can be installed on the microcontroller after connecting the device to the internet. The microcontroller gets the command from the master device, which is to move the robot in a certain manner.

4.3 Internet of Things

IoT, short for Internet of Things is term that refers to things or machines communicating over the internet. In other words, every electronic device that we see or use in any environment can be connected over a network and be used that way. For an example, a light bulb can be turned on from a phone, and the same device can turn on a television or even a stove as they are all communicating with each other over the internet
In the application of this project, the control of the robot can be done over the internet, which will accomplish one of the objectives of this project; 'The robot can be operated over a large distance to minimize any physical exposure'.

4.3.1 MQTT Protocol

To make our robot communicate with the internet we are using the Message Queue Telemetry Transport (MQTT) protocol. The MQTT protocol uses the TCP/IP gateway to publish or subscribe to data from the MQTT broker as shown in the figure.

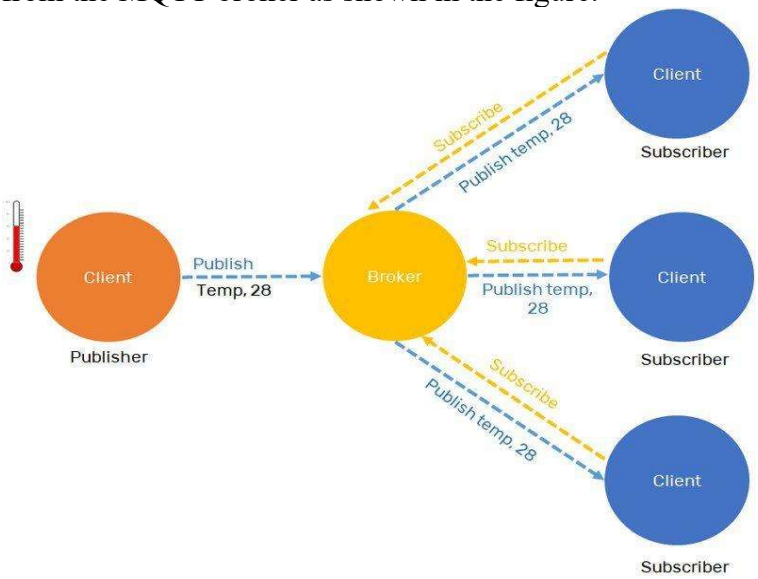


Fig 4-2 MQTT generic structure

We have made our Raspberry Pi an MQTT broker so that it can be connected to any network and begin receiving and publishing data. The crux of the IOT implementation in our project is that when we run our robot control code given in the appendix C, it becomes a subscriber of certain topics from the broker. The broker in this case being the 'localhost' or in other words it becomes a subscriber from the very computer its running on. On the other hand, the joystick becomes a publisher of certain topics, and it publishes information to the MQTT broker which is the raspberry pi. When the raspberry pi receives a "message" from the publisher on a topic that our robot is subscribed to, the robot receives that message. The topics in our cases are x, y, z, roll, pitch, and yaw, When the joystick is rotated about the x axis, it publishes the angle of rotation as a message to the topic "roll" and the broker sends it to the robot which then uses the inverse kinematics function in Appendix B to calculate the joint angles.

Furthermore, we plug a camera into the USB slots of the Pi and publish that to the Broker and make our workstation the subscriber for that topic. Thus, we have the camera streaming video to the workstation from where the person handling the joystick is controlling the slave device.

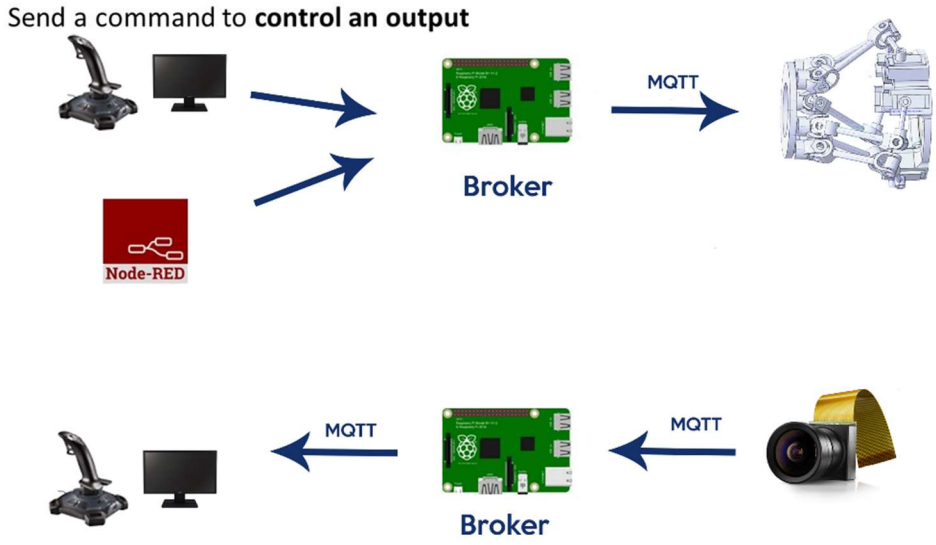


Fig 4-3 MQTT architecture

4.3.2 Joystick

The joystick is Logitech atk3 (figure4-40 which is a 3-axis joystick with 11 buttons. We are using all the rotational axes of the joystick and 7 buttons of the joystick. The joystick is connected to a laptop with a USB cable and interacts with Simulink using the "pilot joystick all" block as shown in the figure 4-5. The code for the initialization of variables is given in the appendix E. The analog output consists of 6 values where the first 3 values indicate the rotational deflections, whereas the last three values are empty. The values are amplified by a factor of 25 and are then separated. They are then published to the MQTT broker that is located on the Raspberry Pi's address using the MATLAB function blocks that are visible in the figure 4-4, the code in the function blocks is given in the Appendix E. The MATLAB function blocks

take the value of rotation as input and publish them by using the MQTT for MATLAB library. The rotations are published under the topics a, b or c. where a corresponds to the rotation about x axis, b to the rotation about y axis, and c to the rotation about z axis. The Robot control code in appendix C is running – on the Raspberry Pi – in a loop subscribed to these topics, so whenever these values are changed the corresponding variable in the robot control code changes. The buttons each correspond to the values 1,2,4,8,16,32 and 1024. When the trigger is pressed it outputs one, when the buttons on top are pressed, they output 2,4,8 and 16 respectively and when the buttons on the side are pressed, they correspond to 32 or 1024. The python code for robot control in Appendix C has if or else statements that decide the action to be performed when one of the buttons is pressed. For example, when the trigger is pressed the robot returns to home position, when the buttons on the sides are pressed the increase or decrease the value of the y coordinate of the moving platform.

Fig 4-4 Logitech atk3 joystick

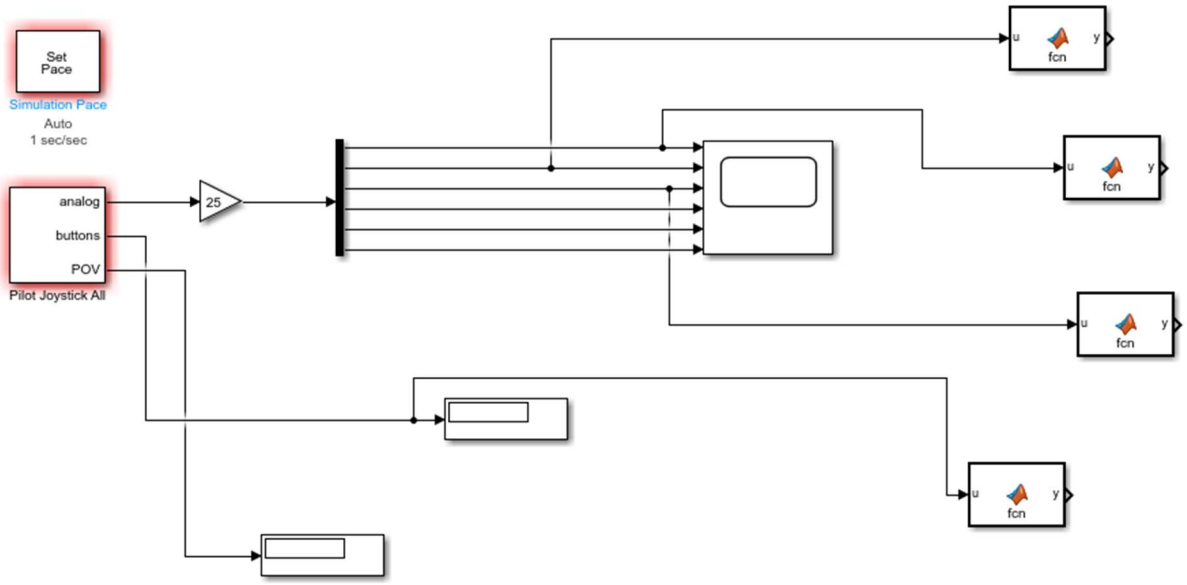


Fig 4-5 Simulink Model to extract and Publish Joystick Data

Frontend and Backend Developing

If viewed from a developer's perspective, everything that we see on any website is the front end of the website and every bit of data that is present in a designated database for that website and the control of that data is called the backend. To enable the communication and control between the master device (the computer that is being controlled by the user) and the microcontroller (which is running on the robot), we need to develop a backend which makes sense.

To develop a frontend and backend easily, IBM made an opensource tool named "Node-red" that can make the flow of data from node to node easily. The application of the inverse kinematics was through the internet was done through making a backend and a front end.

The frontend shown in the figure 4-6 was used to ask the user about how they want to control the platform and the backend (figure 4-7) is what would carry the command in terms of data and convey it to the Raspberry Pi on which the robot control code is running. It is at the final stage (microcontroller) that the data is translated, and the inverse kinematics take place.

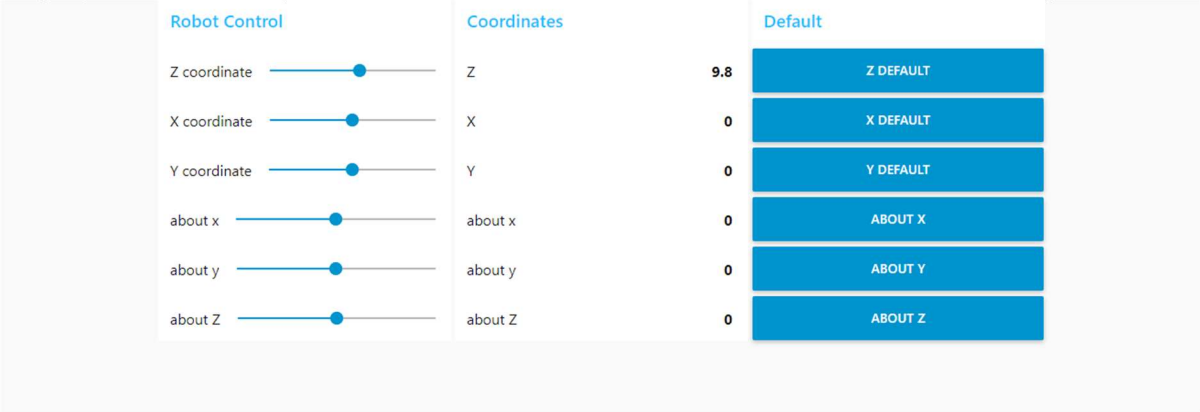


Fig 4-6 User Interface

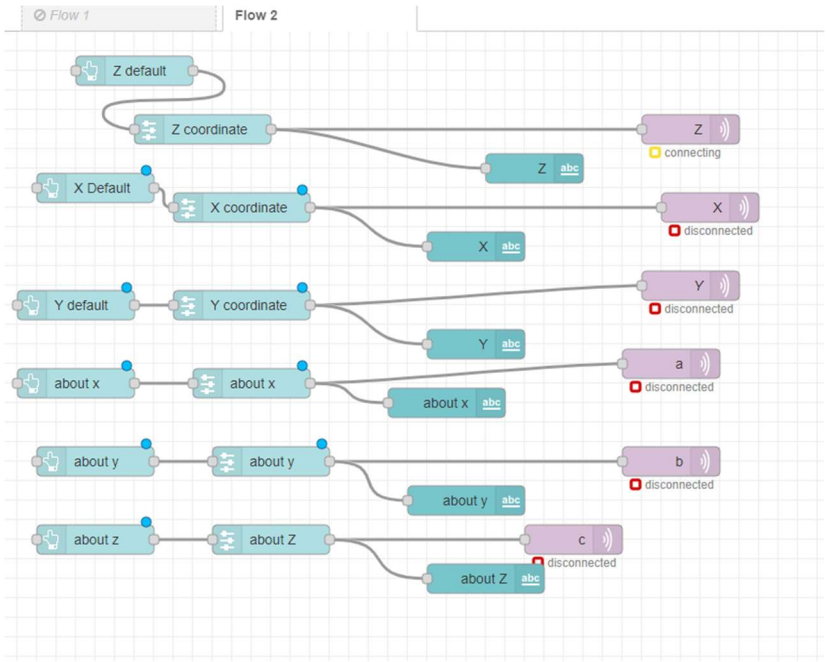


Fig 4-7 Node-Red Flow

4.3.3 Graphical User Interface (GUI) and control

To offer the control to the user, a frontend is developed. For a robot, the graphical user interface (figure 4-5), which is also the front end, would supply all controls that are needed to operate on the robot. For a parallel robot such as the Stewart platform, the motions in all six degrees of freedom can be made separately as well as together. An input is taken from the user from that GUI and is send through the backend to the microcontroller to be processed and used.
An added feature of the GUI in terms of control is also that it provides the user with the option to return to the default positions of the robot at a click of a button. This default position lies at 10.4 cm along the z-axis, 0 along the y axis, and 0 along the x-axis.

4.3.4 Visual Feedback

In any standard medical operating condition, visual feedback is necessary for the surgeon. In our case, one camera supplies feedback of the whole robot and shows how it is moving from a static point of observation. This way the operator can easily see how the Stewart platform is moving with respect to the input that is being provided. the second camera is to be mounted on the end factor that is supposed to invade oral cavity of the patient. This camera provides the visual feedback from an observation point that gives the best feedback of where the end factor has reached.

4.4 Robot Control

The robot control requires the use of inverse kinematics to find the joint angles whenever the user wants to move the platform by using the GUI or the Joystick. Either way the python program (appendix C) running on the Raspberry Pi is programmed to subscribe for the coordinates of movement from the MQTT broker, so that whenever they arrive it calls the inverse kinematics function in appendix B to calculate the values of each joint angle or as they are called in the chapter 3 α_i. The values are then used to find the difference between the current angular position of the servo motor shafts and the desired angular position. The servos are then rotated to their desired positions using PWM signals generated by the PCA 9685 PWM driver, to which all the servos at the base of the Stewart platform are connected as shown in the figure 4-8. The python code uses Paho MQTT client library to subscribe to the broker. In addition to that the addafruit servo library was used to run the servo motors.

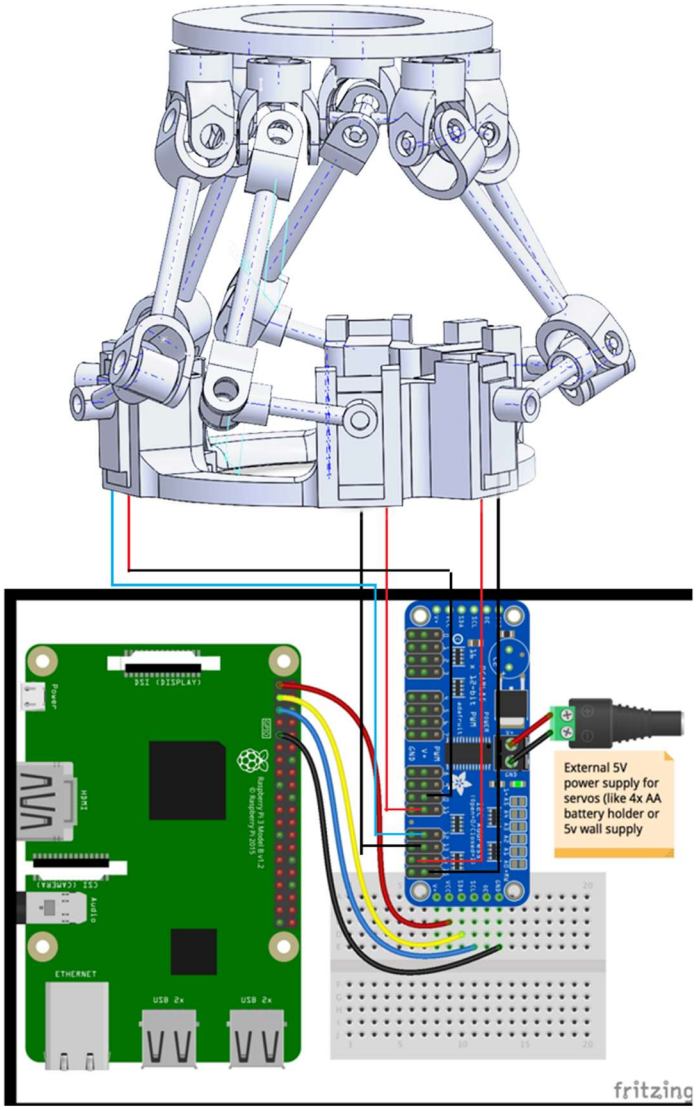


Fig 4-8 Servo Connections

4.5 Testing

Concluding the instrumentation of the project, which included the physical assembly and the integration with the control system, the process of testing it according to the inverse kinematics had to start. This process started out as trial runs which helped us identify key points in the initial design of the project which made the functionality and assembly prone to failure. The two main points from where failure occurred are as follows:

4.5.1 Top platform pins

The first design of the top platform included six pins which were planned to support bearings. These bearings would then support yokes that would be mounted on them to make a 3-DOF joint. According to the configuration for the use of the platform, all these six pins were some

of the few points where load was maximum due to the weight of the object. The problem that we faced almost at once into the testing of the robot was the pins broke off nearly immediately into the test runs. The fact that the part had to be made of PLA to limit the mass of the robot while keeping volume the same also contributed to the failure of the pins.

The failure that occurred was tensile of nature and it mainly occurred due to the layered nature of PLA printed parts. To understand this 'Layered nature', understanding the fundamentals of 3D printing is essential. When any part is printed from a 3D printer, the filament, in our case, PLA, is layered on itself again and again until the desired shape is attained. One could, for the sake of simplicity term the printing as fabricating a mesh. This results into a considerably weak tensile strength of PLA, which was the reason of the failure of the pins of the top part. The tensile failure of the pins looked like cracks propagating parallel to the circumference of the pins.

The failure ended up shortening the pins to the extent of not being able to support the bearing and hence the failure of the parallel robot. The problem was fixed by redesigning this particular part of the robot. The pins were to contain a cylinder cut concentrically to their own geometry and in these cut extrusions, steel rods of the same pitch diameters were inserted. This greatly increased the localized tensile strength since the tensile strengths of the two parts are as:

Table 4-2

	Tensile Strength
Steel	400 MPa
PLA	37 MPa

5 Results and Discussion

5.1 Results

This project concludes three types of results that have been produced so far. They contain the results that were obtained from the mathematical models that were plotted on MATLAB, the results that were attained through geometrical simulations done on MSC ADAMS, and the practical results that were attained after the fabrication and the assembly of the robot while controlling it through IoT.

The MATLAB results were produced after the plotting and understanding of the inverse kinematics on MATLAB. These results were displayed after plotting vectors in a 3D space instead of the infrastructure of the robot. These results turned out to be the first visual understanding of how the project was supposed to operate. The figure 5-1 shows the MATLAB plot of the platform at home position

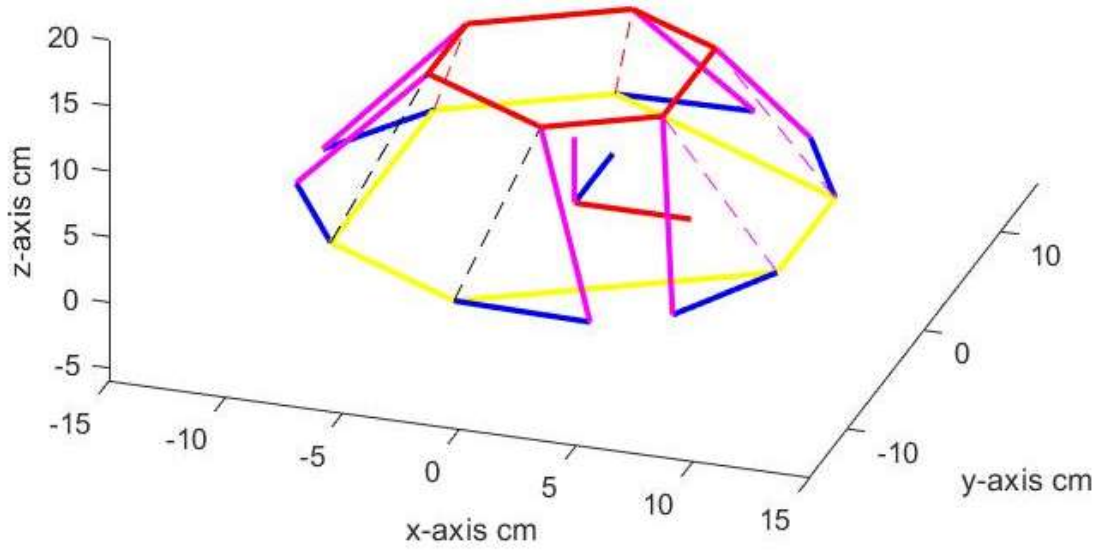


Fig 5-1 MATLAB plot neutral position

After the CAD modeling had been done and the parts were ready to be printed, MSC ADAMS was used to find out if there were any hinderances that would result into unwanted collisions for when the robot would be assembled. This analysis provided us with a result in another as form well. Since the ADMAS simulation was based on the proper CAD model of the assembly, any motion that the simulation would ideally make would also have to occur on the physical model. Hence this analysis was in between the inverse kinematics analysis and the physical model results since the model in the MSC ADMAS simulations was exported from the CAD models, upon which all of the inverse kinematics was based.

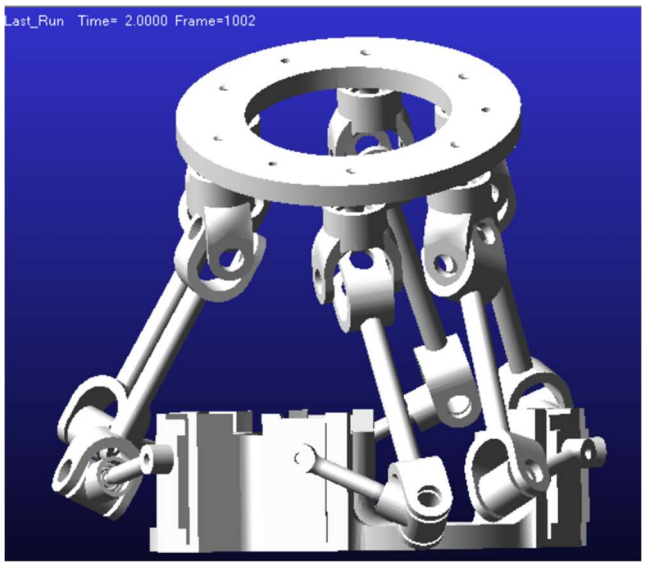


Fig 5-2 MSC Adams rotation about x

After the CAD model was 3D printed and standardized parts were acquired, the assembly had to take place. This would take us to the next stage: controlling the robot through the control system. These would include the joystick and GUI, both of which were actually controlling the robot through IoT. And checking if the actual rotation matches with the rotation given in figure 5-2 and 5-3.

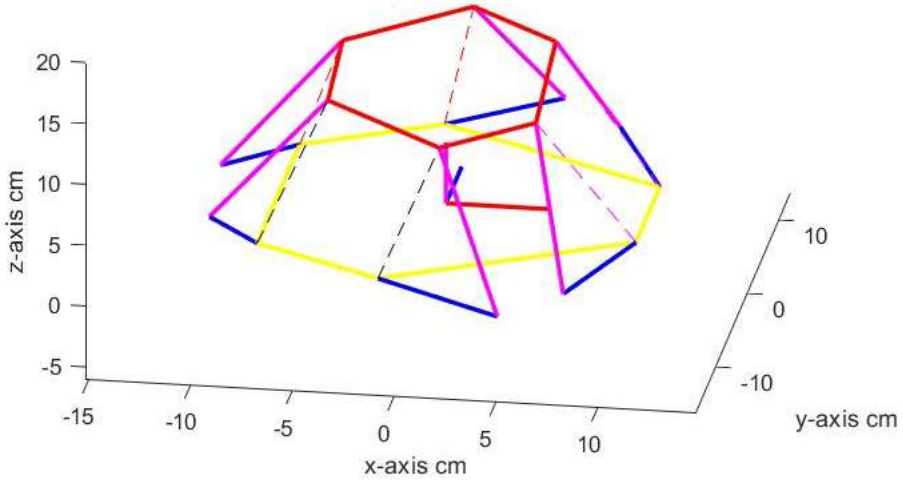


Fig 5-3 MATLAB 25-degree rotation

Fig 5-4 25 degrees rotation about x of joystick

Controlling the robot produced the same satisfactory results as they were expected to. To give an example, the rotation of 25 degrees of the robot about the global x-axis is shown in all 3 results., i.e., figure5-3, figure 5-5 and figure 5-2.

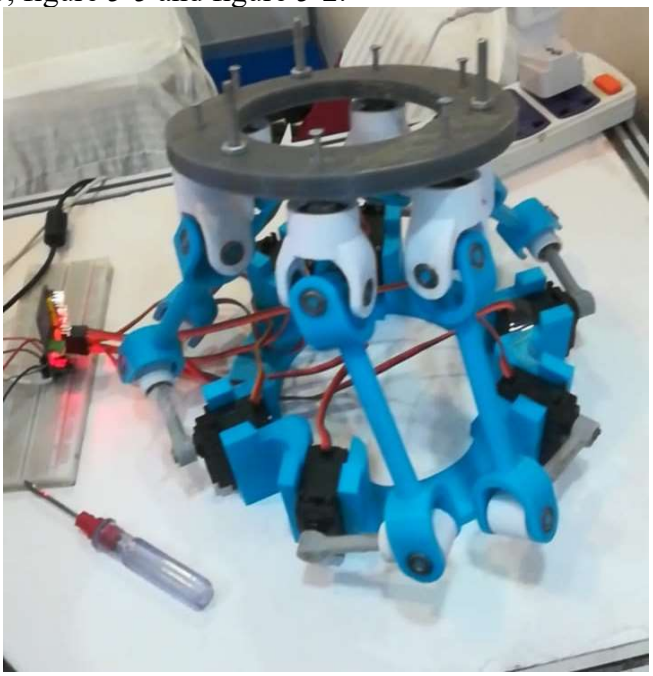


Fig 5-5 25 degrees rotation of the robot

Comparing the diagrams above, we can state that all of the three results was the same result produced through a different analysis. The graphical results that were produced from MATLAB represent the positioning and the movement of the model exactly according to the mathematical model that was acquired from the inverse kinematics. This analysis tells us all the configurations that the robot can shape itself into while only keeping the lengths of the links (vectors plotted in MATLAB) as constraints.
Moving to the results that were attained from MSC ADAMS, they were able to present the analysis with the addition of another constraint: spatial constraints. Spatial constraints refer to the physical constraints that are imposed on the motion of the robot due to the interference of physical parts with each other. This type of analysis is often called collision analysis and is preferred because it provides much realistic results when a domain is put through the inverse kinematics of any motion.
The final result would be the application of control systems on the fabricated version of the robot. The importance of this result is its significance as the moment of truth since it clarifies that if our previous analysis is correct or not.

5.2 Analysis and Discussion

Comparing the results and the quality of these results, we can safely claim that the robot functions near to perfect. That however took a lot of testing and rectification of errors to come about. Initially the robot did not move as planned and upon testing we found that the MSC Adams simulation was also behaving exactly in the same way as the robot for any calculated values of joint angles. Therefore, we concluded that our inverse kinematics function must have been flawed. Only after debugging our code and simulating the results repeatedly were we able to successfully control the robot and were able to verify in the real world that the equations derived in chapter 3 were in fact accurate.

5.3 Summary

We were able to test the robot in the real world, and it to the simulated results in MSc Adams and MATLAB. After rectification of errors, we found the final model to be sufficiently accurate.

6 Impact and Economic Analysis

6.1 Social Impact

Oropharyngeal swabs have been collected in medicine for many purposes. To sum them all up quickly, they are mainly done to collect genetic data that is present in the mucus of human beings. This genetic data can later on be tested to tell many things about the person whose mucus is the specimen. For example, one could tell if the person has any kind of genetic disease or not, which in some cases can also be cancerous. Other than that, respiratory diseases also cause the air coming out of the lungs to leave microorganisms in the mucus, which can be collected to be diagnosed. the applications for which oral swabs are used the most are drug tests and, in recent years, COVID tests.

During the pandemic, the people that were the most exposed to the virus were the ones that were operating on the citizens of nations all around the world. These nations had made getting tested immediately compulsory for the said citizens if they were to show signs of COVID-19 and then quarantine them even if the results were negative. The reason for that was the same reason why all the medical staff that had to carry out these medical procedures had to cover themselves in protective suits and had to sanitize themselves regularly. This reason was 'physical exposure'. Every human being that put themselves near someone that could be infected with the virus ran the risk of exposing themselves to the virus as well.

Our project aimed to do exactly that: take out the risk of physical exposure of the medical staff and the people who were coming in masses to get tested for corona virus. Instead of patients that go in a room one by one and breath on the medical personnel and risking their safety as well as the safety of the medical staff, with the help of this project, those patients would only have to go in a room, place their chins on a machine according to how they are guided and have their oral swab samples collected instantly.

Conclusively, commercializing the robot will make a difference for this particular medical procedure in terms of time, ergonomics, and physical exposure in the following ways:

Table 6-1

For the hospital	For the surgeon	For the patient
The hospital will get a better repute due to the adoption of new technologies.	The surgeon will be safe from exposure from patients.	The patient will have to wait much lesser in lines for checkups.
The hospital will be able to cater to more patients in the same amount of time since the procedure other than the surgery will carried out by robots and not hands.	The surgeon will not have tire very quickly since he will be using only his hands instead of whole limbs.	The patient will be much exposed to risk of infection since he will be spending lesser time near people.

		The patient will not have to worry about shaky hands of the surgeon.
		The patient will not be exposed to the surgeon who could possibly be infected

6.2 Sustainability Analysis

To view the sustainability of this project, we first have to consider the part procurement for the project. This would normally include the method of fabrication, the standardization, and especially the wear and tear of the parts, which is the major reason that requires replacement of parts. All the parts that are needed to make the physical model of the robot and the ones that are needed to operate it are as follows:

- 3D printed parts (Moment arms, bases, and couple)
- Bearings
- Servo motors
- Breadboard
- Microcontroller and motor driver.
- Cameras for visual feedback
- Power supplies

Beginning with the parts that are the least standardized and are specific to the fabrication of only this robot are all made from PLA through a 3D printer. This essentially means that the major parts of the body of the robot can be easily replaced under any circumstances. Moving on to the bearings that are used in the assembly are all procured through their model numbers. The design of the 3D printed parts is made in a way to accommodate the functionality of all those bearings according to their respective dimensions. The brackets for the placements of the servo motors are similarly based on the specific model's dimensions. Conclusively, the physical model of the design can be dealt as a 'plug and play' assembly: the parts are simple to assemble, they just need to fit in their respective places and the robot can be operated.

The parts of the project that are play a role in the instrumentation and control of the robot include the microcontroller, the driver for the motors, and the wires for the configuration. The microcontroller that has been used is a Raspberry-Pi 4 which is widely used in applications such as this. The reason due to which we had to use the driver of the motors is that because the microcontroller only comes with one node that can provide PWM signals which are needed to run the motors. We need to control 6 motors simultaneously for the proper operation of this project and the driver that we have used supports 16 nodes that can supply 16 different PWM signals separately.

This leaves only the power supplies that have to steadily supply a 5 Volt DC at 2 Amperes. This can be achieved by any modern-day power supply that is conventionally sued in robotics labs. To cover expenses in a very tight budget, we have utilized a mobile phone charger that supplies DC power in the same configuration.

Moving on to the pricing of the project, we can conclude that it can be covered in an amount of 25000-30000 PKR. The major portion of the amount is used to purchase the microcontroller for the control of the robot and application of IoT. There is choice for us to operate the robot through the same computer that the master controller is using at the time of operation but to enable the IoT, a separate microcontroller needs to be present on the site of the robot which can be accessed and controlled through the fronted and backend that we have designed so that it can be controlled from the master device that is somewhere else.

6.3 Environmental Impact

Considering the effects of our FYP on the environment, our project is not at all harmful to it while running. The only thing that does contribute during the normal operation of robot is the power supply that the Raspberry Pi uses as a power source which is rated at a miniscule 15 W. Other than that, there is the power supply of the laptop and joystick being used to control the Robot and the power supply of the driver of the servo motors that are being used to actuate the robot which is a maximum of 10 as was measured by us during operation. On the other hand, the power consumed by the 3D printers while fabricating the parts can be contribute to environmental harm in terms of the release of greenhouse gasses if the power supply in Pakistan is traced back to the power plants operating on coal consumption.
On the other hand, using PLA is beneficial from an environmental perspective since PLA is biodegradable. This essentially means that in case the fabricated parts are discarded due to any failure, they will not end up harming the environment.

6.4 Sustainable Development Goals (SDGs)

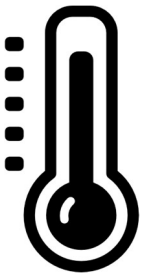
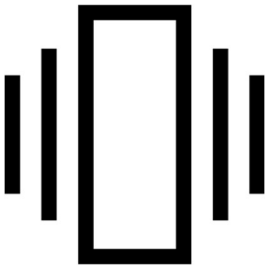
Table 6-2 SDGs Adherence

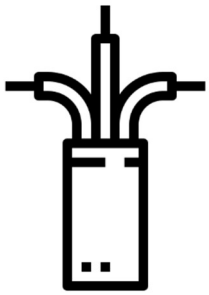
S. No	SDG	Adherence of FYP to SDG
1.	No Poverty	
2.	Zero Hunger	
3.	Good Health and Well-Being	The project is made keeping in mind the safety and health of the medical personnel.
4.	Quality Education	
5.	Gender Equality	
6.	Clean Water and Sanitation	
7.	Affordable and Clean Energy	
8.	Decent Work and Economic Growth	
9.	Industry, Innovation and Infrastructure	A successful implementation of the proposed idea will be helpful in developing a new product.
10.	Reduced Inequalities	
11.	Sustainable Cities and Communities	
12.	Responsible Consumption and Production	Manufacturing of the proposed design is done without material wastage i.e., 3D printing
13.	Climate Action	
14.	Life Below Water	

15.	Life on Land	
16.	Peace, Justice and Strong Institutions	
17.	Partnerships	Making alliance with the Ministry of National Health Services will lead to protecting the medical personnel at grass root level.

6.5 Hazard Identification and Safety Measures

Table 6-3 Hazard Identification and Safety Measures

Symbol	Hazard	Safety Measures
	**Overheating**	The current should not exceed the recommended limit of the motor. In any case, motors should be replaced or cleaned regularly.
	**Mechanical Vibrations**	The moment arms should be tightly bound to the motors. Bearings and keys should be joined strongly. Joints of bearings with pins should be strong as well.
	**Water Damage**	Make sure that there are no fluids near the surgical equipment.

	**Short Circuiting**	Wire repairs and covering is crucial. Short circuits can be cause by uncovered wires.

6.6 Summary

As a conclusion of impact and economic analysis, we can safely claim that our robot can be fabricated in repeatedly for mass production since the fabrication of parts is simple and can be done quickly; through 3D printing while also considering that 3D printing is cheap. Additionally, the cost of instrumentation is not very high as well since all that is needed other than the fabricated PLA parts are standardized bearings, linear bushings, and a motor long with a microcomputer that cost in the same range as well.

Moving on to the environmental analysis, the regular operation of the robot would only require the quite common DC supply of 5 Volts at 2 Amperes of current, which is not much. On the other hand, the mass production of the robot could produce plastic waste, but that will be much better than the plastic waste produced from non-degradable plastics as PLA is degradable.

7 Conclusion and Future Recommendations

7.1 Conclusion

Our project began with studying the configurations of Stewart platform and selecting a suitable one. Moving on to the CAD designing of robot, we had to follow the inverse kinematics or base the inverse kinematic exactly on what the base was being built like. After the physical fabrication and assembly of the robot, we were able to control it thorough IoT, which answers one of the objectives of the project: '*The robot should be designed to be operable through IoT'.* This essentially means that the control of our robot can be done over the internet which guarantees the medical safety of the operator. Other than operating over IoT, the robot also performed precise movements exactly according to the controller's input while visual feedback of the robot was also being provided to the controller. The control of the robot was offered over the virtual GUI as well as through a joystick whose inputs were mapped to the actions of the robot.

Just as the progress on the robot started, the robot was segmented according to its functionality into two parts:

- The Stewart Platform, which aimed to position the probe collection mechanism.
- The probe controlling mechanism, which aimed to carry the end factor (swab) to the oropharyngeal cavity and collect samples.

In terms of completion, we were able to finish the first segment after solving a number of issues with its designs. The second segment, which was relatively simple, was completed to design finalization. This design was conducted according to the safety measures that were needed to be installed and the designs of commercially available end factors.

7.2 Future Recommendations

The problems that we faced while the robot was being made are to be experienced again if research is carried out in similar manner as us: motors are used as actuators with the structure made of PLA.

Our recommendations taken from the knowledge and experience gained over the duration of this research includes the following:

- Reinforcements through steel of all 3D printed points where localized stresses can occur is crucial to avoid failure. Incorporating dedicated slots for these supports in the design process would be ideal.
- Motors that are used as actuators should have the smallest least count possible. This is recommended especially when movements of the robot are smooth and not jerky, which ironically is the primary function that the robot is supposed to provide (smooth movements). Motors with larger least counts cause a delay and induce a wobbly nature to the assembly since the movement that is required by the movement is smaller than the least count of the motor.

References

[1] "WHO/Europe | Influenza - Pandemic preparedness." https://www.euro.who.int/en/health-topics/communicable-diseases/influenza/pandemic-influenza/pandemic-preparedness (accessed Dec. 27, 2021).

[2] M. Mutambudzi *et al.*, "Occupation and risk of severe COVID-19: prospective cohort study of 120 075 UK Biobank participants," *Occup. Environ. Med.*, vol. 78, no. 5, pp. 307–314, May 2021, doi: 10.1136/OEMED-2020-106731.

[3] S. Q. Li *et al.*, "Clinical application of an intelligent oropharyngeal swab robot: Implication for the COVID-19 pandemic," *Eur. Respir. J.*, vol. 56, no. 2, 2020, doi: 10.1183/13993003.01912-2020.

[4] Z. Lei, L. Haixia, Z. Junli, and L. Kang, "Different Methods of COVID-19 Detection," Accessed: Dec. 27, 2021. [Online]. Available: http://www.hsj.gr/Tel:

[5] F. S. Dawood *et al.*, "What Is the Added Benefit of Oropharyngeal Swabs Compared to Nasal Swabs Alone for Respiratory Virus Detection in Hospitalized Children Aged <10 Years?," *J. Infect. Dis.*, vol. 212, no. 10, pp. 1600–1603, Nov. 2015, doi: 10.1093/INFDIS/JIV265.

[6] X. Wang *et al.*, "Comparison of nasopharyngeal and oropharyngeal swabs for SARS-CoV-2 detection in 353 patients received tests with both specimens simultaneously," *Int. J. Infect. Dis.*, vol. 94, pp. 107–109, May 2020, doi: 10.1016/J.IJID.2020.04.023.

[7] J. P. Merlet, *Parallel robots*, vol. 128. 2006.

[8] C. Li, X. Gu, X. Xiao, C. M. Lim, X. Duan, and H. Ren, "A Flexible Transoral Robot Towards COVID-19 Swab Sampling," *Front. Robot. AI*, vol. 8, no. April, pp. 1–8, 2021, doi: 10.3389/frobt.2021.612167.

[9] J. Seo, S. Shim, H. Park, J. Baek, J. H. Cho, and N. H. Kim, "Development of robot-assisted untact swab sampling system for upper respiratory disease," *Appl. Sci.*, vol. 10, no. 21, pp. 1–15, 2020, doi: 10.3390/app10217707.

[10] T. R. Peterson, "[2]Design and implementation of Stewart Platform robot for robotics course laboratory," no. March, 2020.

[11] A. S. Grogan, "A Low Cost, Portable Stewart Platform Study for Flight Simulation and Gaming Simulation." 2020.

[12] J. J. Craig, *Introduction to Robotics: Mechanics and Control, 3rd Edition*. 2004.

[13] W. Kim, K. Huh, B. Yi, and W. Cho, "Optimal Synthesis of a Wrist-Type 6 Degree-of-Freedom Force / Torque Sensor Using Stewart Platform Structure 2 . Kinematic Analysis of FIT Sensor using Stewart Platform Structure Assume that all of prismatic joints of Stewart," vol. 9, no. 4, pp. 462–471, 1995.

[14] R. Eisele, "Inverse Kinematics of a Stewart Platform," 2019. https://www.xarg.org/paper/inverse-kinematics-of-a-stewart-platform/.

[15] "PLA Filament | 3D4Makers | 3D printing." https://www.3d4makers.com/products/pla-filament (accessed May 24, 2022).

Appendix A Inverse Kinematics and Plotting in MATLAB

```
clc
clear

Rt=[0 0 10.4]';
a=0*pi/180;
b=0*pi/180;
c=0*pi/180;
R=10.620943523;
r=6;
%servoangle

%cranklength
d=5.9;
%leglength
s=12;
%Beta
Be=[118, 2, -122, 122, -2, -118 ]'.*pi/180;
%Be=[0 0 0 0 0 0  ]'.*pi/180;

mu=(41.54666139/2)*(pi/180);
R1=[R*cos(mu), R*sin(mu), 0]';
R6=[R*cos(-mu), R*sin(-mu), 0]';
R2=[R*cos((2*pi/3)-mu), R*sin((2*pi/3)-mu), 0]';
R3=[R*cos((2*pi/3)+mu), R*sin((2*pi/3)+mu), 0]';
R4=[R*cos((4*pi/3)-mu), R*sin((4*pi/3)-mu), 0]';
R5=[R*cos((4*pi/3)+mu), R*sin((4*pi/3)+mu), 0]';
Rb=[R1 R2 R3 R4 R5 R6];
mu=25*pi/180;
rt1=[r*cos((pi/3)-mu), r*sin((pi/3)-mu), 0]';
rt2=[r*cos((pi/3)+mu), r*sin((pi/3)+mu), 0]';
rt3=[r*cos(pi-mu), r*sin(pi-mu), 0]';
rt4=[r*cos(pi+mu), r*sin(pi+mu), 0]';
rt5=[r*cos((5*pi/3)-mu), r*sin((5*pi/3)-mu), 0]';
rt6=[r*cos((5*pi/3)+mu), r*sin((5*pi/3)+mu), 0]';
rt=[rt1,rt2,rt3,rt4,rt5,rt6];
rotx=[1 0 0;0 cos(a) -sin(a);0 sin(a) cos(a)];
roty=[cos(b) 0 sin(b); 0 1 0;-sin(b) 0 cos(b)];
rotz=[cos(c) sin(c) 0; -sin(c) cos(c) 0; 0 0 1];
BTR=rotx*roty*rotz;
r=BTR*rt;
Rtp=[Rt Rt Rt Rt Rt Rt];
L=Rtp+r-Rb;

L1=norm(L(:,1));
L2=norm(L(:,2));
L3=norm(L(:,3));
L4=norm(L(:,4));
L5=norm(L(:,5));
L6=norm(L(:,6));
LL=[L1 L2 L3 L4 L5 L6]';

leg1x=[Rb(1,1),Rt(1)+r(1,1)];
leg1y=[Rb(2,1),Rt(2)+r(2,1)];
```

```
leg1z=[Rb(3,1),Rt(3)+r(3,1)];

leg2x=[Rb(1,2),Rt(1)+r(1,2)];
leg2y=[Rb(2,2),Rt(2)+r(2,2)];
leg2z=[Rb(3,2),Rt(3)+r(3,2)];

leg3x=[Rb(1,3),Rt(1)+r(1,3)];
leg3y=[Rb(2,3),Rt(2)+r(2,3)];
leg3z=[Rb(3,3),Rt(3)+r(3,3)];

leg4x=[Rb(1,4),Rt(1)+r(1,4)];
leg4y=[Rb(2,4),Rt(2)+r(2,4)];
leg4z=[Rb(3,4),Rt(3)+r(3,4)];

leg5x=[Rb(1,5),Rt(1)+r(1,5)];
leg5y=[Rb(2,5),Rt(2)+r(2,5)];
leg5z=[Rb(3,5),Rt(3)+r(3,5)];

leg6x=[Rb(1,6),Rt(1)+r(1,6)];
leg6y=[Rb(2,6),Rt(2)+r(2,6)];
leg6z=[Rb(3,6),Rt(3)+r(3,6)];



rho=(s^2)*ones(6,1);
dm=((d^2)*ones(6,1));
gamma=(rho-(LL.^2)-dm)';
X1=[cos(Be');sin(Be');0,0,0,0,0,0];
Y1=[0,0,0,0,0,0;0,0,0,0,0,0;1,1,1,1,1,1];
%(gamma-2*d*dot(X1,L))*(x^2)+(4*d*dot(Y1,L))*(x)+(2*d*dot(X1,L))+gamma
E=-2*d*dot(L,X1);
B=-2*d*dot(L,Y1);

x=(B+sqrt((B.^2)-(gamma.^2)+(E.^2))./(gamma+E);

al=(2*atan(x))';
dal=al*180/pi
%crank vectors

cx=d.*cos(Be);
cy=d.*sin(Be);
cz=d.*sin(al);
cr=[cx';cy';cz'];
A=Rb+cr;


axis([-15 15 -15 15 -6 20]);

p1=line(leg1x,leg1y,leg1z, 'LineWidth',[0.5],'Color','m','LineStyle','--');
p2=line(leg2x,leg2y,leg2z, 'LineWidth',[0.5],'Color','r','LineStyle','--');
p3=line(leg3x,leg3y,leg3z, 'LineWidth',[0.5],'Color','r','LineStyle','--');
p4=line(leg4x,leg4y,leg4z, 'LineWidth',[0.5],'Color','k','LineStyle','--');
p5=line(leg5x,leg5y,leg5z, 'LineWidth',[0.5],'Color','k','LineStyle','--');
p6=line(leg6x,leg6y,leg6z, 'LineWidth',[0.5],'Color','m','LineStyle','--');
```

```
p7=line([Rt(1)+r(1,1),
Rt(1)+r(1,2)],[Rt(2)+r(2,1),Rt(2)+r(2,2)],[Rt(3)+r(3,1),Rt(3)+r(3,2)],
'LineWidth',[2],'Color','r');
p8=line([Rt(1)+r(1,2),
Rt(1)+r(1,3)],[Rt(2)+r(2,2),Rt(2)+r(2,3)],[Rt(3)+r(3,2),Rt(3)+r(3,3)],
'LineWidth',[2],'Color','r');
p9=line([Rt(1)+r(1,3),
Rt(1)+r(1,4)],[Rt(2)+r(2,3),Rt(2)+r(2,4)],[Rt(3)+r(3,3),Rt(3)+r(3,4)],
'LineWidth',[2],'Color','r');
p10=line([Rt(1)+r(1,4),
Rt(1)+r(1,5)],[Rt(2)+r(2,4),Rt(2)+r(2,5)],[Rt(3)+r(3,4),Rt(3)+r(3,5)],
'LineWidth',[2],'Color','r');
p11=line([Rt(1)+r(1,5),
Rt(1)+r(1,6)],[Rt(2)+r(2,5),Rt(2)+r(2,6)],[Rt(3)+r(3,5),Rt(3)+r(3,6)],
'LineWidth',[2],'Color','r');
p12=line([Rt(1)+r(1,6),
Rt(1)+r(1,1)],[Rt(2)+r(2,6),Rt(2)+r(2,1)],[Rt(3)+r(3,6),Rt(3)+r(3,1)],
'LineWidth',[2],'Color','r');

p13=line([Rb(1,1), Rb(1,2)],[Rb(2,1),Rb(2,2)],[Rb(3,1),Rb(3,2)],
'LineWidth',[2],'Color','y');
p14=line([Rb(1,2), Rb(1,3)],[Rb(2,2),Rb(2,3)],[Rb(3,2),Rb(3,3)],
'LineWidth',[2],'Color','y');
p15=line([Rb(1,3), Rb(1,4)],[Rb(2,3),Rb(2,4)],[Rb(3,3),Rb(3,4)],
'LineWidth',[2],'Color','y');
p16=line([Rb(1,4), Rb(1,5)],[Rb(2,4),Rb(2,5)],[Rb(3,4),Rb(3,5)],
'LineWidth',[2],'Color','y');
p17=line([Rb(1,5), Rb(1,6)],[Rb(2,5),Rb(2,6)],[Rb(3,5),Rb(3,6)],
'LineWidth',[2],'Color','y');
p18=line([Rb(1,6), Rb(1,1)],[Rb(2,6),Rb(2,1)],[Rb(3,6),Rb(3,1)],
'LineWidth',[2],'Color','y');

p19=line([Rb(1,1), A(1,1)],[Rb(2,1),A(2,1)],[Rb(3,1),A(3,1)],
'LineWidth',[2],'Color','b');
p20=line([Rb(1,2), A(1,2)],[Rb(2,2),A(2,2)],[Rb(3,2),A(3,2)],
'LineWidth',[2],'Color','b');
p21=line([Rb(1,3), A(1,3)],[Rb(2,3),A(2,3)],[Rb(3,3),A(3,3)],
'LineWidth',[2],'Color','b');
p22=line([Rb(1,4), A(1,4)],[Rb(2,4),A(2,4)],[Rb(3,4),A(3,4)],
'LineWidth',[2],'Color','b');
p23=line([Rb(1,5), A(1,5)],[Rb(2,5),A(2,5)],[Rb(3,5),A(3,5)],
'LineWidth',[2],'Color','b');
p24=line([Rb(1,6), A(1,6)],[Rb(2,6),A(2,6)],[Rb(3,6),A(3,6)],
'LineWidth',[2],'Color','b');

p25=line([A(1,1), Rt(1)+r(1,1)],[A(2,1),Rt(2)+r(2,1)],[A(3,1),Rt(3)+r(3,1)],
'LineWidth',[2],'Color','m');
p26=line([A(1,2), Rt(1)+r(1,2)],[A(2,2),Rt(2)+r(2,2)],[A(3,2),Rt(3)+r(3,2)],
'LineWidth',[2],'Color','m');
p27=line([A(1,3), Rt(1)+r(1,3)],[A(2,3),Rt(2)+r(2,3)],[A(3,3),Rt(3)+r(3,3)],
'LineWidth',[2],'Color','m');
p28=line([A(1,4), Rt(1)+r(1,4)],[A(2,4),Rt(2)+r(2,4)],[A(3,4),Rt(3)+r(3,4)],
'LineWidth',[2],'Color','m');
```

```
p29=line([A(1,5), Rt(1)+r(1,5)],[A(2,5),Rt(2)+r(2,5)],[A(3,5),Rt(3)+r(3,5)],
'LineWidth',[2],'Color','m');
p30=line([A(1,6), Rt(1)+r(1,6)],[A(2,6),Rt(2)+r(2,6)],[A(3,6),Rt(3)+r(3,6)],
'LineWidth',[2],'Color','m');
xlabel("x-axis cm")
ylabel("y-axis cm")
zlabel("z-axis cm")

drawnow


pause()
for i=0:1:7
    Rt=[0 0 10.4]'
    a=0*pi/180;
    b=0*pi/180;
    c=i*pi/180;
    R=10.620943523;
    r=6;
    %servoangle

    %cranklength
    d=5.9;
    %leglength
    s=12;
    %Beta
    Be=[118, 2, -122, 122, -2, -118 ]'.*pi/180;
    %Be=[0 0 0 0 0 0  ]'.*pi/180;

    mu=(41.54666139/2)*(pi/180);
    R1=[R*cos(mu), R*sin(mu), 0]';
    R6=[R*cos(-mu), R*sin(-mu), 0]';
    R2=[R*cos((2*pi/3)-mu), R*sin((2*pi/3)-mu), 0]';
    R3=[R*cos((2*pi/3)+mu), R*sin((2*pi/3)+mu), 0]';
    R4=[R*cos((4*pi/3)-mu), R*sin((4*pi/3)-mu), 0]';
    R5=[R*cos((4*pi/3)+mu), R*sin((4*pi/3)+mu), 0]';
    Rb=[R1 R2 R3 R4 R5 R6];
    mu=25*pi/180;
    rt1=[r*cos((pi/3)-mu), r*sin((pi/3)-mu), 0]';
    rt2=[r*cos((pi/3)+mu), r*sin((pi/3)+mu), 0]';
    rt3=[r*cos(pi-mu), r*sin(pi-mu), 0]';
    rt4=[r*cos(pi+mu), r*sin(pi+mu), 0]';
    rt5=[r*cos((5*pi/3)-mu), r*sin((5*pi/3)-mu), 0]';
    rt6=[r*cos((5*pi/3)+mu), r*sin((5*pi/3)+mu), 0]';
    rt=[rt1,rt2,rt3,rt4,rt5,rt6];
    rotx=[1 0 0;0 cos(a) -sin(a);0 sin(a) cos(a)];
    roty=[cos(b) 0 sin(b); 0 1 0;-sin(b) 0 cos(b)];
    rotz=[cos(c) sin(c) 0; -sin(c) cos(c) 0; 0 0 1];
    BTR=rotx*roty*rotz;
    r=BTR*rt;
    Rtp=[Rt Rt Rt Rt Rt Rt];
    L=Rtp+r-Rb;
    %L1=L(:,1);
    %L2=L(:,2);
    %L3=L(:,3);
```

```
%L4=L(:,4);
%L5=L(:,5);
%L6=L(:,6);
%length1=norm(L1)
%length2=norm(L2)
%length3=norm(L3)
%length4=norm(L4)
%length5=norm(L5)
%length6=norm(L6)
L1=norm(L(:,1));
L2=norm(L(:,2));
L3=norm(L(:,3));
L4=norm(L(:,4));
L5=norm(L(:,5));
L6=norm(L(:,6));
LL=[L1 L2 L3 L4 L5 L6]';

leg1x=[Rb(1,1),Rt(1)+r(1,1)];
leg1y=[Rb(2,1),Rt(2)+r(2,1)];
leg1z=[Rb(3,1),Rt(3)+r(3,1)];

leg2x=[Rb(1,2),Rt(1)+r(1,2)];
leg2y=[Rb(2,2),Rt(2)+r(2,2)];
leg2z=[Rb(3,2),Rt(3)+r(3,2)];

leg3x=[Rb(1,3),Rt(1)+r(1,3)];
leg3y=[Rb(2,3),Rt(2)+r(2,3)];
leg3z=[Rb(3,3),Rt(3)+r(3,3)];

leg4x=[Rb(1,4),Rt(1)+r(1,4)];
leg4y=[Rb(2,4),Rt(2)+r(2,4)];
leg4z=[Rb(3,4),Rt(3)+r(3,4)];

leg5x=[Rb(1,5),Rt(1)+r(1,5)];
leg5y=[Rb(2,5),Rt(2)+r(2,5)];
leg5z=[Rb(3,5),Rt(3)+r(3,5)];

leg6x=[Rb(1,6),Rt(1)+r(1,6)];
leg6y=[Rb(2,6),Rt(2)+r(2,6)];
leg6z=[Rb(3,6),Rt(3)+r(3,6)];

rho=(s^2)*ones(6,1);
dm=((d^2)*ones(6,1));
gamma=(rho-(LL.^2)-dm)';
X1=[cos(Be');sin(Be');0,0,0,0,0,0];
Y1=[0,0,0,0,0,0;0,0,0,0,0,0;1,1,1,1,1,1];
%(gamma-2*d*dot(X1,L))*(x^2)+(4*d*dot(Y1,L))*(x)+(2*d*dot(X1,L))+gamma
E=-2*d*dot(L,X1);
B=-2*d*dot(L,Y1);

x=(B+sqrt((B.^2)-(gamma.^2)+(E.^2)))./(gamma+E);

al=(2*atan(x))';

dal=al*180/pi
```

```
    %crank vectors

    cx=d.*cos(Be);
    cy=d.*sin(Be);
    cz=d.*sin(al);
    cr=[cx';cy';cz'];
    A=Rb+cr;

    set(p1,'X', leg1x, 'Y',leg1y,'Z',leg1z)
    set(p2,'X', leg2x, 'Y',leg2y,'Z',leg2z)
    set(p3,'X', leg3x, 'Y',leg3y,'Z',leg3z)
    set(p4,'X', leg4x, 'Y',leg4y,'Z',leg4z)
    set(p5,'X', leg5x, 'Y',leg5y,'Z',leg5z)
    set(p6,'X', leg6x, 'Y',leg6y,'Z',leg6z)

    set(p7,'X', [Rt(1)+r(1,1), Rt(1)+r(1,2)],
'Y',[Rt(2)+r(2,1),Rt(2)+r(2,2)],'Z',[Rt(3)+r(3,1),Rt(3)+r(3,2)])
    set(p8,'X', [Rt(1)+r(1,2), Rt(1)+r(1,3)],
'Y',[Rt(2)+r(2,2),Rt(2)+r(2,3)],'Z',[Rt(3)+r(3,2),Rt(3)+r(3,3)])
    set(p9,'X', [Rt(1)+r(1,3), Rt(1)+r(1,4)],
'Y',[Rt(2)+r(2,3),Rt(2)+r(2,4)],'Z',[Rt(3)+r(3,3),Rt(3)+r(3,4)])
    set(p10,'X', [Rt(1)+r(1,4), Rt(1)+r(1,5)],
'Y',[Rt(2)+r(2,4),Rt(2)+r(2,5)],'Z',[Rt(3)+r(3,4),Rt(3)+r(3,5)])
    set(p11,'X', [Rt(1)+r(1,5), Rt(1)+r(1,6)],
'Y',[Rt(2)+r(2,5),Rt(2)+r(2,6)],'Z',[Rt(3)+r(3,5),Rt(3)+r(3,6)])
    set(p12,'X', [Rt(1)+r(1,6), Rt(1)+r(1,1)],
'Y',[Rt(2)+r(2,6),Rt(2)+r(2,1)],'Z',[Rt(3)+r(3,6),Rt(3)+r(3,1)])
    set(p13,'X', [Rb(1,1), Rb(1,2)], 'Y',[Rb(2,1),Rb(2,2)],'Z',[Rb(3,1),Rb(3,2)])
    set(p14,'X', [Rb(1,2), Rb(1,3)], 'Y',[Rb(2,2),Rb(2,3)],'Z',[Rb(3,2),Rb(3,3)])
    set(p15,'X', [Rb(1,3), Rb(1,4)], 'Y',[Rb(2,3),Rb(2,4)],'Z',[Rb(3,3),Rb(3,4)])
    set(p16,'X', [Rb(1,4), Rb(1,5)], 'Y',[Rb(2,4),Rb(2,5)],'Z',[Rb(3,4),Rb(3,5)])
    set(p17,'X', [Rb(1,5), Rb(1,6)], 'Y',[Rb(2,5),Rb(2,6)],'Z',[Rb(3,5),Rb(3,6)])
    set(p18,'X', [Rb(1,6), Rb(1,1)], 'Y',[Rb(2,6),Rb(2,1)],'Z',[Rb(3,6),Rb(3,1)])

    set(p19,'X', [Rb(1,1), A(1,1)], 'Y',[Rb(2,1),A(2,1)],'Z',[Rb(3,1),A(3,1)])
    set(p20,'X', [Rb(1,2), A(1,2)], 'Y',[Rb(2,2),A(2,2)],'Z',[Rb(3,2),A(3,2)])
    set(p21,'X', [Rb(1,3), A(1,3)], 'Y',[Rb(2,3),A(2,3)],'Z',[Rb(3,3),A(3,3)])
    set(p22,'X', [Rb(1,4), A(1,4)], 'Y',[Rb(2,4),A(2,4)],'Z',[Rb(3,4),A(3,4)])
    set(p23,'X', [Rb(1,5), A(1,5)], 'Y',[Rb(2,5),A(2,5)],'Z',[Rb(3,5),A(3,5)])
    set(p24,'X', [Rb(1,6), A(1,6)], 'Y',[Rb(2,6),A(2,6)],'Z',[Rb(3,6),A(3,6)])

    set(p25,'X', [A(1,1), Rt(1)+r(1,1)], 'Y',[A(2,1),
Rt(2)+r(2,1)],'Z',[A(3,1),Rt(3)+r(3,1)])
    set(p26,'X', [A(1,2), Rt(1)+r(1,2)], 'Y',[A(2,2),
Rt(2)+r(2,2)],'Z',[A(3,2),Rt(3)+r(3,2)])
    set(p27,'X', [A(1,3), Rt(1)+r(1,3)], 'Y',[A(2,3),
Rt(2)+r(2,3)],'Z',[A(3,3),Rt(3)+r(3,3)])
    set(p28,'X', [A(1,4), Rt(1)+r(1,4)], 'Y',[A(2,4),
Rt(2)+r(2,4)],'Z',[A(3,4),Rt(3)+r(3,4)])
    set(p29,'X', [A(1,5), Rt(1)+r(1,5)], 'Y',[A(2,5),
Rt(2)+r(2,5)],'Z',[A(3,5),Rt(3)+r(3,5)])
    set(p30,'X', [A(1,6), Rt(1)+r(1,6)], 'Y',[A(2,6),
Rt(2)+r(2,6)],'Z',[A(3,6),Rt(3)+r(3,6)])
```

```
    drawnow
    pause(.1)
end
```

Appendix B Inverse Kinematics Function in Python

```
import numpy as np
from numpy import linalg as LA

def IKfunction(a,b,c,x,y,T):
    Rt=np.array([x,y,T])
    pi=np.pi
    a=a*pi/180
    b=b*pi/180
    c=c*pi/180
    R=10.620943523
    r=6
    #cranklength
    d=5.9
    #leglength
    s=12
    #Beta
    Be=np.array([118, 2, -122, 122, -2, -118])
    Be=Be*pi/180
    mu=(41.54666139/2)*(pi/180)
    R1=np.array([R*np.cos(mu), R*np.sin(mu), 0])
    #1.shape=(3,1)
    #1=R1.transpose()
    R6=np.array([R*np.cos(-mu), R*np.sin(-mu), 0])
    #6.shape=(3,1)
    #6=R6.transpose()
    R2=np.array([R*np.cos((2*pi/3)-mu), R*np.sin((2*pi/3)-mu), 0])
    #2.shape=(3,1)
    #2=R2.transpose()
    R3=np.array([R*np.cos((2*pi/3)+mu), R*np.sin((2*pi/3)+mu), 0])
    #3.shape=(3,1)
    #3=R3.transpose()
    R4=np.array([R*np.cos((4*pi/3)-mu), R*np.sin((4*pi/3)-mu), 0])
    #R4.shape=(3,1)
    #4=R4.transpose()
    R5=np.array([R*np.cos((4*pi/3)+mu), R*np.sin((4*pi/3)+mu), 0])
    #R5.shape=(3,1)
    #5=R5.transpose()
    Rb=np.array([ R1, R2, R3, R4, R5, R6])
    Rb=Rb.transpose()
```

```
mu=(50/2)*(pi/180)
rt1=np.array([r*np.cos((pi/3)-mu), r*np.sin((pi/3)-mu), 0])
rt2=np.array([r*np.cos((pi/3)+mu), r*np.sin((pi/3)+mu), 0])
rt3=np.array([r*np.cos(pi-mu), r*np.sin(pi-mu), 0])
rt4=np.array([r*np.cos(pi+mu), r*np.sin(pi+mu), 0])
rt5=np.array([r*np.cos((5*pi/3)-mu), r*np.sin((5*pi/3)-mu), 0])
rt6=np.array([r*np.cos((5*pi/3)+mu), r*np.sin((5*pi/3)+mu), 0])
# rt1.shape=(3,1)
# rt2.shape=(3,1)
# rt3.shape=(3,1)
# rt4.shape=(3,1)
# rt5.shape=(3,1)
# rt6.shape=(3,1)
rt=np.array([rt1,rt2,rt3,rt4,rt5,rt6])
rt=rt.transpose()



rotx=np.array([[1, 0, 0],[0, np.cos(a), -np.sin(a)],[0, np.sin(a), np.cos(a)]])
roty=np.array([[np.cos(b), 0, np.sin(b)],[0, 1, 0],[-np.sin(b), 0, np.cos(b)]])
rotz=np.array([[np.cos(c), np.sin(c), 0],[-np.sin(c), np.cos(c), 0],[ 0, 0, 1]])
BTR=(rotx.dot(roty)).dot(rotz)
r=BTR.dot(rt)


Rtp=np.array([Rt, Rt, Rt, Rt, Rt, Rt])
Rtp=Rtp.transpose()

L=Rtp+r-Rb


LL=LA.norm(L,axis=0)

sm=(np.power(s,2))*(np.ones(6))
dm=(np.power(d,2))*(np.ones(6))
gamma=sm-dm-(np.power(LL,2))
#1D array row vector
X1=np.array([np.cos(Be),np.sin(Be),np.zeros(6)])
Y1=np.array([[0, 0, 0, 0, 0, 0],[0, 0, 0, 0, 0, 0],[1, 1, 1, 1, 1, 1]])

#(gamma-2*d*dot(X1,L))*(x^2)+(4*d*dot(Y1,L))*(x)+(2*d*dot(X1,L))+gamma
E=-2*d*np.multiply(X1,L).sum(axis=0)
#E=2*d*dot(X1,L);
F=-2*d*np.multiply(Y1,L).sum(axis=0)
#F=2*d*dot(Y1,L);
t=np.divide(F,(gamma+E))
```

```
    sqrt=np.sqrt(np.square(E)+np.square(F)-np.square(gamma))  #./(gamma-E)

    u=np.divide(sqrt,(gamma+E))

    al=2*np.arctan(t+u)
    al=al*180/pi
    return al
```

Appendix C Robot Control Code in Python

```
 import time

from board import SCL, SDA
import busio
from adafruit_motor import servo
from adafruit_pca9685 import PCA9685

i2c = busio.I2C(SCL, SDA)
pca = PCA9685(i2c)
pca.frequency = 50
servo1 = servo.Servo(pca.channels[0], min_pulse=525, max_pulse=2325)
servo2 = servo.Servo(pca.channels[1], min_pulse=525, max_pulse=2325)
servo3 = servo.Servo(pca.channels[2], min_pulse=525, max_pulse=2325)
servo4 = servo.Servo(pca.channels[3], min_pulse=525, max_pulse=2325)
servo5 = servo.Servo(pca.channels[4], min_pulse=525, max_pulse=2325)
servo6 = servo.Servo(pca.channels[5], min_pulse=525, max_pulse=2325)

from IKfunction import*

import paho.mqtt.client as mqtt
translate=0;
b=0.0
a=0.0
c=0.0
x=0.0
y=0.0
z=9.8
a1=105.0
a2=87.0
a3=90.0
a4=85.0
a5=80.0
a6=100.0

a1i=a1
```

```
a2i=a2
a3i=a3
a4i=a4
a5i=a5
a6i=a6

pos = np.array([a1-a1i, a2-a2i, a3-a3i, a4-a4i, a5-a5i, a6-a6i])


def on_connect (client, userdata, flags, rc):
    print("connected with result code" + str(rc))
    client.subscribe("X")
    client.subscribe("Y")
    client.subscribe("Z")
    client.subscribe("a")
    client.subscribe("b")
    client.subscribe("c")
    client.subscribe("translation")


def on_message(client, userdata, msg):
    global x
    global y
    global z
    global a
    global b
    global c
    global translate
    if msg.topic=='Z':
        z=float(msg.payload)
    elif msg.topic=='Y':
        y=float(msg.payload)
    elif msg.topic=='X':
        x=float(msg.payload)
    elif msg.topic=='a':
        a=float(msg.payload)
    elif msg.topic=='b':
        b=float(msg.payload)
    elif msg.topic=='c':
        c=float(msg.payload)
    elif msg.topic=='translation':
        translate=float(msg.payload)
    print(msg.topic+"="+msg.payload.decode("utf-8"))

def on_subscribe(client, userdata, mid, granted_qos):
    print ('Subscribed')
```

```
client = mqtt.Client()
client.connect("localhost", 1883, 60)
#client.connect("192.168.100.13",1883)
client.on_connect = on_connect
client.on_subscribe = on_subscribe
client.on_message = on_message
client.loop_start()
while True:
    if translate==4:
        if z<=13:
            z=z+0.1

    elif translate==2:
        if z>=7:
            z=z-0.1


    elif translate==8:
        y=y-0.1
    elif translate==16:
        y=y+0.1
    elif translate==32:
        x=x+0.1
    elif translate==1024:
        x=x-0.1
    elif translate==1:
        b=0.0
        a=0.0
        c=0.0
        x=0.0
        y=0.0
        z=9.8

    al=IKfunction(a,b,c,x,y,z)
    print(al)
    #al=al.astype(int)

    temp=al-pos
    ti=10
    j=temp/ti
    #print(j)
    opos = np.array([a1, a2, a3, a4, a5, a6])
    for i in range(ti):
        a1 = a1+j[0]
        a2 = a2-j[1]
```

```
        a3 = a3+j[2]
        a4 = a4-j[3]
        a5 = a5+j[4]
        a6 = a6-j[5]
        servo1.angle= a1
        servo2.angle= a2
        servo3.angle= a3
        servo4.angle= a4
        servo5.angle= a5
        servo6.angle= a6
        #time.sleep(0.01)

    npos = np.array([a1, a2, a3, a4, a5, a6])
    pos = np.array([(a1-a1i), -(a2-a2i), (a3-a3i), -(a4-a4i), (a5-a5i), -(a6-a6i)])
```

Appendix D MQTT code in Python

```
import paho.mqtt.client as mqtt
import time
from IKfunction import*

def on_connect (client, userdata, flags, rc):
    print("connected with result code" + str(rc))
    client.subscribe("X")
    client.subscribe("Y")
    client.subscribe("Z")


def on_message(client, userdata, msg):
    global th_abort
    global t
    if 'abort' in msg.payload.decode("utf-8"):
        if t.is_alive():
            t.cancel()
        th_abort = True
    else:
        print(msg.payload.decode("utf-8"))

def on_subscribe(client, userdata, mid, granted_qos):
    print ('Subscribed')

t = None
th_abort = False

client = mqtt.Client()
```

```
client.on_connect = on_connect
client.on_subscribe = on_subscribe
client.on_message = on_message
client.connect("localhost", 1883, 60)
client.connect("192.168.1.101",1883)

client.loop_start()
while th_abort == False:
   client.on_message=client.on_message
```

Appendix E MQTT code in MATLAB

For initialization of Simulink Model

```
clc
clear
close all

dt=1/50;
tfinal=1000;
```

For rotation about X axis

```
function y=mymqtta(u)
raspi=mqtt('tcp://192.168.1.101','Port',1883);
topic="a";
message=string(u);
publish(raspi,topic,message)
y=zeros(1,1);
```

For rotation about Y axis

```
function y=mymqttb(u)
raspi=mqtt('tcp://192.168.1.101','Port',1883);
topic="b";
message=string(u);
publish(raspi,topic,message)
y=zeros(1,1);
```

For rotation about Z axis

```
function y=mymqttc(u)
raspi=mqtt('tcp://192.168.1.101','Port',1883);
topic="c";
message=string(u);
publish(raspi,topic,message)
y=zeros(1,1);
```

For all translations:

```
function y=mymqtttranslate(u)
raspi=mqtt('tcp://192.168.1.101','Port',1883);
topic="translation";
```

```
message=string(u);
publish(raspi,topic,message)
y=zeros(1,1);
```