PARALLEL-SERIAL MANIPULATOR MANIPULATORS FOR ROBOTIC MACHINING

Harry R. J. Softley-Graham

3rd Year Project Interim Report

Department of Electronic &
Electrical Engineering

UCL

Supervisor: Prof. Chow Yin Lai
Advisor: Prof. Chow Yin Lai

19 December 2022

I have read and understood UCL's and the Department's statements and guidelines concerning plagiarism.

I declare that all material described in this report is my own work except where explicitly and individually indicated in the text. This includes ideas described in the text, figures and computer programs.

This report contains 21 pages (excluding this page and the appendices) and 5866 words.

Signed: [illegible] Date: 19/12/22

(Student)

PARALLEL-SERIAL MANIPULATOR MANIPULATORS FOR ROBOTIC MACHINING

Harry R. J. Softley-Graham

1 Project Description

1.1 Project Background

The field of robotic machining has had explosive development within the past 50 years. Starting with the basic pick and place processes, slowly but surely all ranges of interactions and machining of an object have been fully automated through the use of articulated robotic systems. These robotic systems are typically constructed of a series of rigid linkages, connected via rotary or prismatic joints. The desired end effector is placed at the end of these linkages and is the robotics arms method to interact within its work environment. These interactions can range from picking up and moving objects with the use of a gripper, performing additive manufacturing such as 3D printing, or the most common method, performing the three main machining methods of milling, drilling, and turning.

When building a serial manipulator, there is a trade-off when attempting to improve the qualities of a given manipulator. For example, if the degrees of freedom are increased by adding an extra link, the rigidity of the end effector decreases along with the need for an overall increase in the total cost and robots power consumption to make up for the added weight. The easiest way to recover the rigidity within the end effector within a serial manipulator would be to replace all the joints with ones of a better specification to make up for this extra linkage. However, this project aims to find an extra solution that provides a new avenue within methods to increase a robot's rigidity and manipulation through the use of parallel robotics.

1.2 Project Aims

This project explores, experiments, and refines the use of multiple serial manipulators in a parallel formation to move and interact with a given object. While the main end effector performs its main task of machining, additional end effectors can move, grab, and interact with their surroundings to increase the rigidity of the end effector. The current proposal is to mount two individual end effectors higher up and closer to the wrist of the robot. These effectors would be able to adaptively grab and hold to surrounding structures, providing a closer anchor point for the end effector. This will reduce the total load forces applied to the wrist joint, allowing the machining operations to occur with greater precision.

These additional end effectors will also allow for improved interaction with the object. If not secured to the ground plate, they would be able to grab, rotate and orientate the piece allowing a greater range of motion and degrees of access for the end effector to interact and machine the piece. This would further allow for the robot to act as both a pick and place and machining robot without the need for a tool change or reprogramming.

2 Literature review

2.1 Serial Robotic Stiffness Analysis

To increase the stiffness within robotics, a fundamental understanding of what causes a reduction of stiffness within robotic joints is key. The analysis of joint rigidity and prediction of end effector error has become extremely wide, with many kinematic models reimagined to incorporate and compensate for weaker joints and predict a potential error range for the expected end effector position. [1] [2] This is particularly important within industrial robotics, with precise operations such as machining, as even a slight deviation from the desired position could lead to an overall large propagation in the produced products' tolerance. [3]

Due to the design of serial manipulators, the applied forces range vastly depending on the end effectors' position, orientation, and kinematic approach. [4] As seen within the paper on a *Review of Industrial Robot Stiffness Identification and Modelling* [5], the expected deviation of the robot is down to the ability of the motors to provide the required torque and the initial design of the original robot.

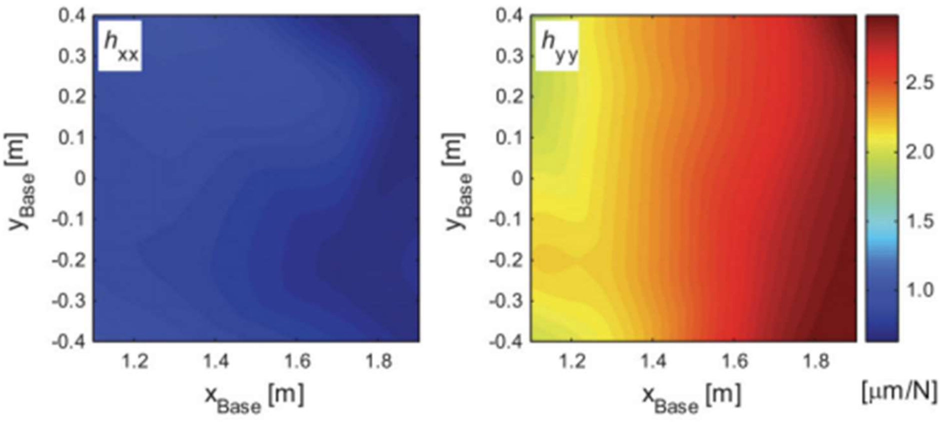


Figure 1 – Expected compliance within the X and Y direction of a 6-DOF Robotic Manipulator [5]

Figure 1 depicts an estimator of the liner stiffness of a KUKA six-axis serial robot, graphing the expected compliance within the robot within each axis as the end effector is placed over specific points of the workspace. As shown within both axis, as the robot extends further from the origin/base of the robot the applied forces and overall torque required to keep the joints stiff significantly increases.

2.2 Practical application of parallel robotics

The main inspiration for this project comes from a paper published by Prof. Chow Lai, David Chavis, and Songlin Ding around a *Transformable parallel-serial manipulator for robotic machining.* [6] This paper explores the application of a seriel-parrallel hybrid manipulator, where the primary end effector is a standard drill and there is a secondary end effector that is an angular gripper. This paper explores the effect a secondary linkage has on the primary end effector rigidity, seeing a significant improvement within the end effector stiffness while the secondary end effector is anchored to a fixed point. However, the main drawbacks came within a few key aspects that lie within the design. The robot found difficulty with the initial accuracy while moving when the second arm is not attached to a fixed body. Secondly, the nature of the robot severely limits its ability. Its 3-DOF significantly restricted its workspace, and in effect made the robot only operate within a 2-dimensional plane, with a single orientation.

Approaches have been taken using Parallel/Delta robotics to create joints with high mobility, along with high rigidity. The construction of joints and linkages through a range of parallel, interconnected linkages, provides each joint to be driven by multiple motors simultaneously and provides each link to have greater degrees of freedom. [7] [8] .

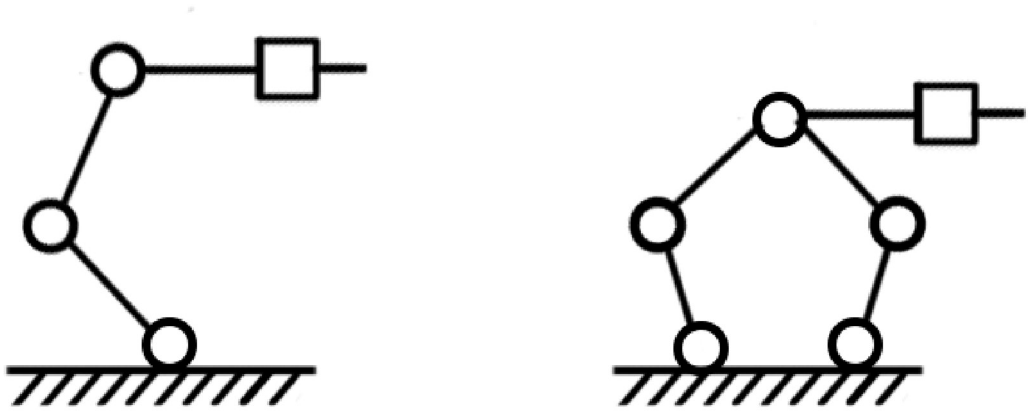


Figure 2 – Example of Serial (left) and parallel (right) robot

The delta robot has a greater number of anchor points and achieves its range of motion through the manipulation of various servos equally angled around a central point. They have seen a vast amount of success within the machine and pick and place, however, compared to a serial manipulator, delta robotics suffer from a few key weaknesses. First is the fact their available work area is extremely limited. Attempts have been made at incorporating delta and serial robotics as seen within various examples outlined within a paper on *State of the Art in Artificial Wrists* [9]. They lay out a range of delta robotic approaches that all contain methods that provide at least 2DOF, however many have size and rigidity issues. Additionally, one large issue within these robots is their functional lifetime, with many of the twisted designs suffering from increased joint degradation and wear, increasing the joint's probability of failure.

A novel approach to this problem is attempting to use compliance error compensation to see an improvement within the rigidity and accuracy of the real end effector position. A software approach that provides pre-calculated error compensation can be seen in the paper on the *Compliance error compensation of a robot end-effector* [10], whereas the addition of *flexible joints controlled through the use of variable stiffness actuators* [11] provides a method to physically improve the end effectors' rigidity without the need for pre-calculated errors, and instead it provides the compensation live as external forces are applied. While there was a significant improvement in the performance of the desired operations, the compo nets within this research are outside this project's budget. However, the addition of precalculated or live compliance error using secondary effector adjustment to support the primary effector would see an improvement within a serial-hybrid manipulator, especially allowing improvements within the manipulator's motion when the second arm is not yet fixed to an anchor point.

The application of parallel, serial manipulators has seen application in other fields of robotics as seen in a paper published in the *Biomedical Robotics* section of *Frontiers in Robotics and AI* [12]. The paper explores the application of replacing serial joints with a parallel structure comprised of pneumatically powered prismatic actuators. This application saw an overall rigidity improvement of 11.8%, with the system using a nonlinear multi-objective optimization method that allowed the optimization of piston extensions to reduce the effective deviation at the end effector. The largest drawback found within the project was the system's overall size and weight, with the parallel additions reducing the overall available workspace and reach of the system, with an overall increase in the size and mass of the robot. These drawbacks are a consistent highlight between mechanical approaches when attempting to increase joint stiffness, hence it's key to apply a work ethic to reduce the overall size and reduce the overall impact within the manipulator's performance with and without the application of parallel robotics.

1 Work performed to date – 3500 words

1.1 Chosen Design

Through inspiration from the original paper, along with the knowledge discovered within the initial research the design that this project will focus on is an improved and expanded iteration of the original parallel series hybrid robotic manipulator. [6]

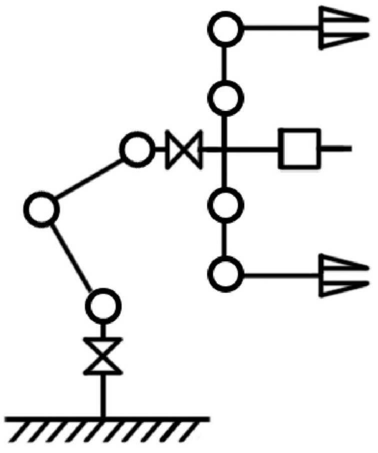


Figure 3-Basic Diagram of robot

This design is comprised of three main robotic structures. The first is the 4DOF serial robotic arm which is where the primary effector, in this case, a drill/mill is mounted. From this structure, the final linkage is where the design splits, and the secondary effectors are attached. These secondary effectors will have robotic grippers attached to the end of them, allowing them to attach and anchor at fixed points within the workspace to form a temporary parallel robot, increasing the primary end effectors' rigidity. These sub-arms will have 3DOF relative to their mounting point, and 7DOF relative to the base mount.

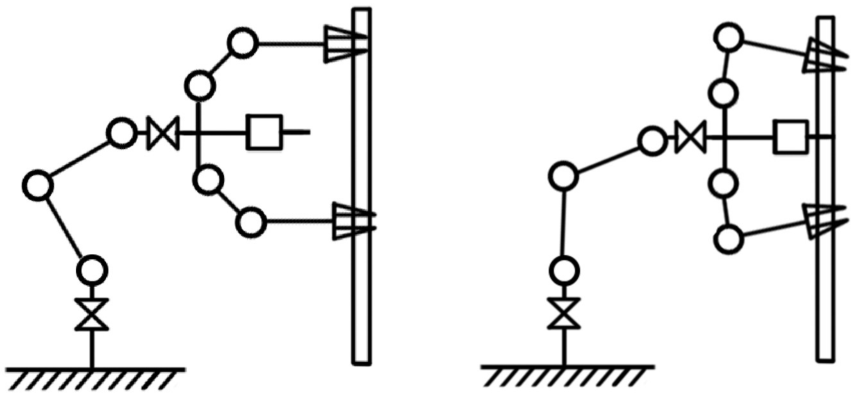


Figure 4 – Sketch of how the robot can perform machining actions.

The increase in the degrees of freedom, along with the multiple arms allows for overall greater mobility along with greater rigidity as support is provided in multiple locations across the robot. When mounted to these points the secondary arms can work in conjunction with the primary effector to position and move the drill, providing a greater drilling force, a reduction in compliance and an overall increase in the rigidity of the system.

1.2 Mechanical Design

With the initial designs of the robot completed, basic CAD designs were first produced, with a full-scale CAD model developed by designing it link by link. Initial rough calculations and dimensions were put together to allow the general design of the robot to be constructed. Each part went through many iterations, with small adjustments throughout the process with the main design features mentioned in this section.

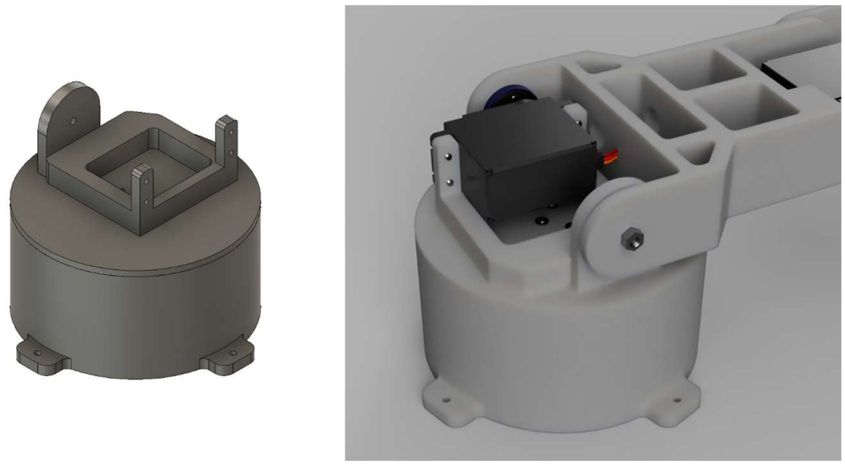


Figure 5 – Model and Render of the robotic base

The base consists of a simple cylindrical design that can be mounted to a flat surface. A high torque motor is mounted internally, attaching a pinning plate. This acts as joint 1 of the robot. The plate contains a basic servo mount that allows for a larger, higher torque motor to be mounted to form the second joint.

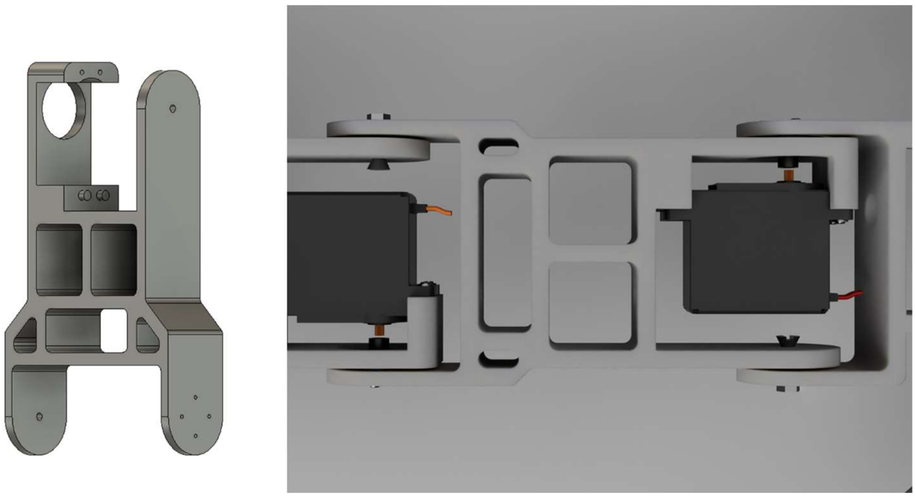


Figure 6- Model and render of the robotic linkage

The basic link design accommodates offer a basic liner rigid joint design. One side attaches to the servo, whereas the other is fastened to an M4 screw. This screw is mounted internally and contains washers between each joint to reduce friction when rotating. The link then contains a servo mount at the end, forming the next join section. A basic support design was produced to reduce the weight of the part without compromising its rigidity, along with small holes for cable management.

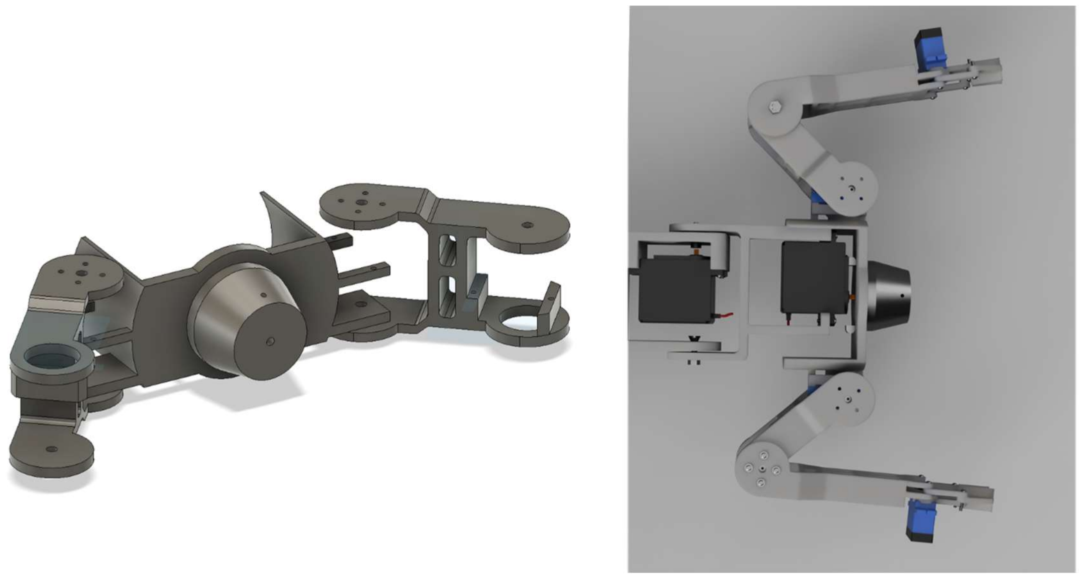


Figure 7- Model and render of the primary end effector

These links continue up to joint 5 where the design incorporates a circular-shaped piece to accommodate multiple servo mounts and a chuck. The servo mounts allow for the secondary serial arms to be attached, where a basic design of a mill chuck is put in place to simulate the space it would take up.

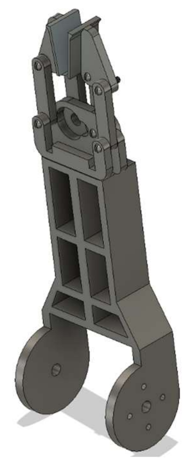
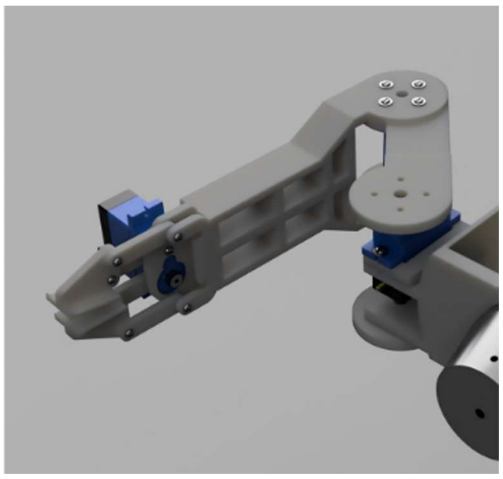


Figure 8 –Model and render of End effector Sub arm

The sub-arms follow a similar design to the primary arm, with a basic parallel gripper design to act as the end effector for this subsystem. The gripper is composed of a small, lightweight design that allows for parallel motion of the jaws that is driven by a small motor driving two gears. The jaws themselves have a flat surface where material can be placed to increase the grip strength of the effector. The overall design is satisfactory; however, a redesign needs to be made to increase the size of the jaws can open up to along with their grip strength as there is a lot of compliance within the current mechanism. This will be redesigned within the second half of the project.

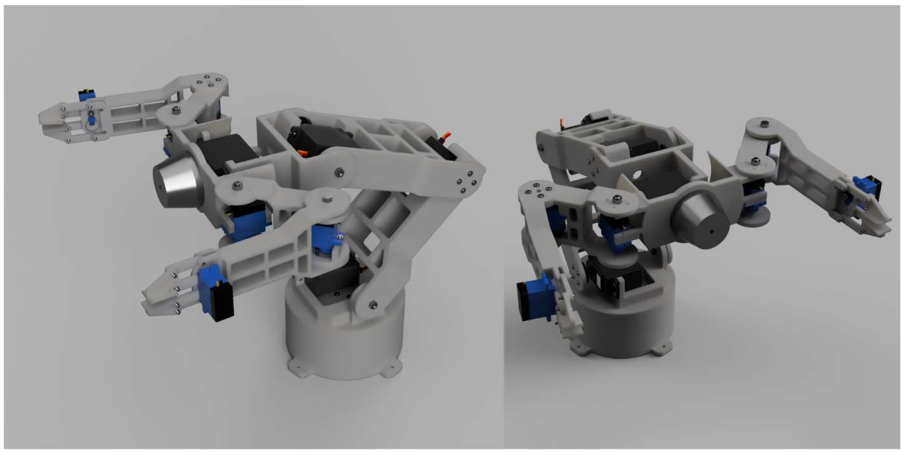


Figure 9-Final renders of the full model

1.3 Manufacturing and Constructing

Initial components were prototyped through the use of a 3D printer, printed within a 2mm lay height. Each of the individual links was printed out, with any key holes drilled out through the use of a hand drill. Each was then secured using a standard M4 screw and washers to reduce the friction between each link.

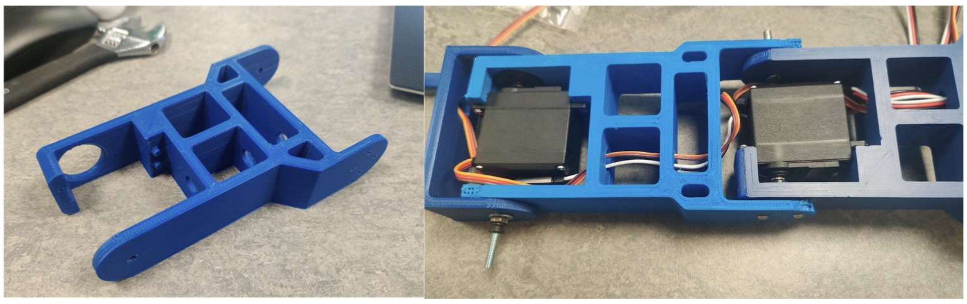


Figure 10 – Example of PLA printed Link (left) and fully constructed link (right)

With all the parts printed and constructed, the first iteration of the main body of the arm was produced. This allowed for the first iteration of the parts to be tested later on through basic simulations of the main kinematics tested against a grid set up against a wall.

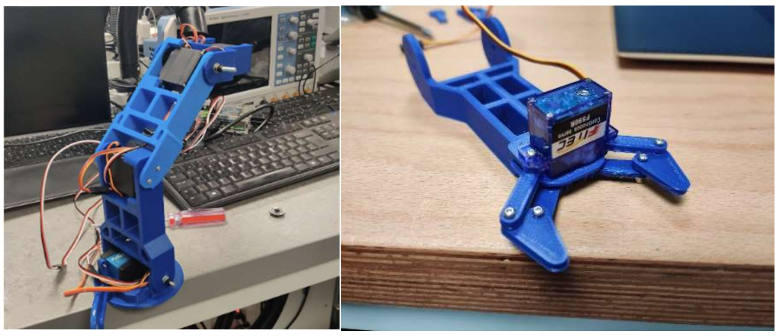


Figure 11- Primary Effector Experimental Set up (left) initial prototype for gripper (right)

While effective, this iteration of the robot with the use of PLA-printed parts proved to have some small mechanical flaws that propagated to larger errors within the accuracy of the kinematics within the robot. The first flaw was found within the overall accuracy of the machine parts. 3D printing suffers from thermal expansion, causing parts to slightly expand as the layers are printed, causing multiple tolerance errors and increased friction between the joints. The second issue was caused by the plasticity of the PLA, causing it to slightly bend under higher loads, particularly around the servo mountings. These caused the linkages to not behave as rigid bodies and causes small errors within each link that propagate to larger errors within the end effectors' position.

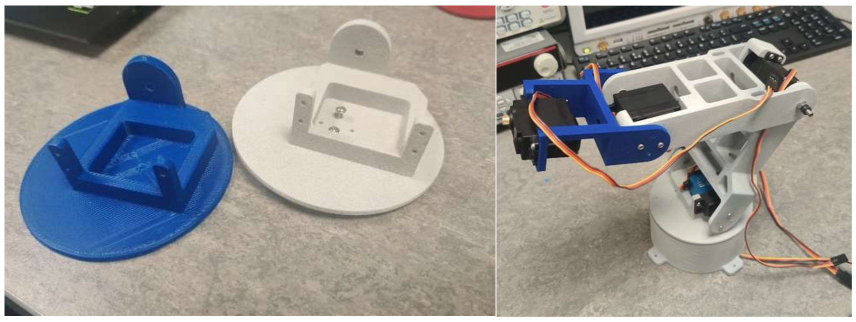


Figure 12- Comparison of PLA and Nylon parts (left) and second iteration of primary effector (right)

The solution to these problems lay within a slight redesign of the parts, along with a change in the manufacturing method of the main body linkages. Through the use of an SLS printer, the parts were manufactured out of solid Nylon. While heavier, the overall accuracy of the parts was far greater with no issues of thermal shrinking or expansion. Additionally, the linkages themselves were far more robust, and do not bend or warp under high loads. The change from PLA to Nylon saw a significant increase in the overall rigidity of the robot, along with significant improvement in the joint's ability to smoothly rotate. The plan is to keep prototype parts made from PLA, with finalized parts being sent to the nylon printer.

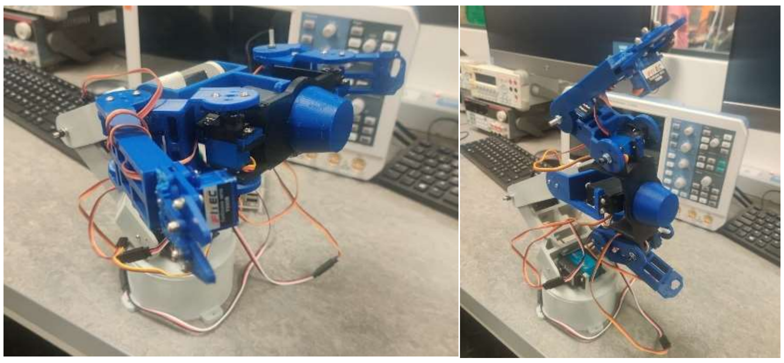


Figure 13 – Fully constructed first iteration showing a mix between nylon and PLA parts.

1.4 Hardware Choices

1.4.1 Microcontroller

The choice of the microcontroller will determine many of the constraints and software approaches this project will take. The two main considerations for this project are an Arduino-driven servo controller and a Raspberry pi driven servo controller. The Arduino offers a cheaper, electronically simpler, easier-to-code avenue to develop the software for this project. Additionally, there are plenty of accurate servo controllers on the market that provide. The second alternative would be the use of a Raspberry Pi. These microcontrollers are far more complex, operate at a higher clock speed, and have far more inbuilt memory allowing them to calculate computationally expensive algorithms quickly and effectively in a shorter period. Within this project, it is expected that due to the nature of the kinematics and trajectory planning, the faster these can be calculated the smother the trajectory profile that will be generated. Additionally, the use of the Raspberry Pi opens the avenue to the use of machine vision and some basic artificial intelligence as it would allow the program to be built in Python. Python is a far more advanced language with a plethora of advanced libraries such as TensorFlow, Skitlearn, and Pytorch that can be used efficiently use for an object, feature, and pattern detection.

Hence due to this, a Raspberry pi 3 B+ has been selected as the main microcontroller within this project. It has been paired with an Adafruit 16-PWM servo hat to drive the arms motors and servos, along with the software being written in python, with a basic UI designed within TKinter and Matplotlib to interact with the arm through very basic software.

1.4.2 Servos

Due to the project's budget limitation, there was a large amount of research and consideration taken into the servos chosen to drive certain joints. A basic philosophy was used whereas the servo got along the robot, the lighter and less driving torque it should have. This will reduce the load on the previous motors and also keep the overall costs down within the robot. Additionally, all the servos operate within the 3.5-6V range, given a constant 5V supply is provided to each servo with a stall current of 2A.

Figure 14 - Table of servos used within the project

Name	**Torque** (at 5v) (kgf/cm)	**Weight** (kg)	**Dimensions** (LxWxH) (mm)	**Supply voltage** (V)	**Operating speed** (° /s)	**Joints Used in**	**Image**
FT5330M	35	0.067	40.7 x 19.7 x 42.9	5-7.4	260	1	
MG996R	11	0.055	40.7 x 19.7 x 42.9	5-7.4	428.6	2,3,4	
MG92B	3.5	0.014	22.8x12x31	5-6.6	750	5,7,	
FS90	1.5	0.009	22.8x12x27	4.8-6	600	6,8,9,10	

1.5 Kinematics Calculations

To get the robot joints to move and effectively work, the kinematics of the robot needs to be defined. Before this, the robot's home plane, axis, and link planes must be initially defined. The current design of the robot is broken down in the following diagram,

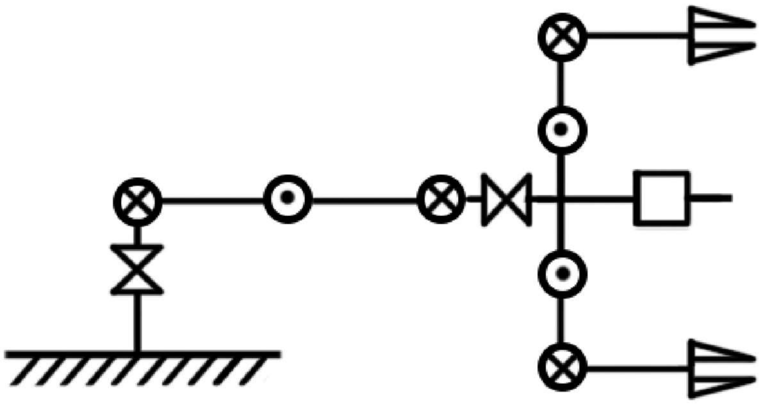


Figure 15 – Diagram of proposed robotic design

However, with some in-built coding fixes to flip the rotation of some of the joints, it simplifies the robot down to an easier-to-deal-with diagram. With this, the forward kinematics can be derived to find the end effector's position given inputted angles, along with the inverse kinematics to find the angles given a set end effector angle.

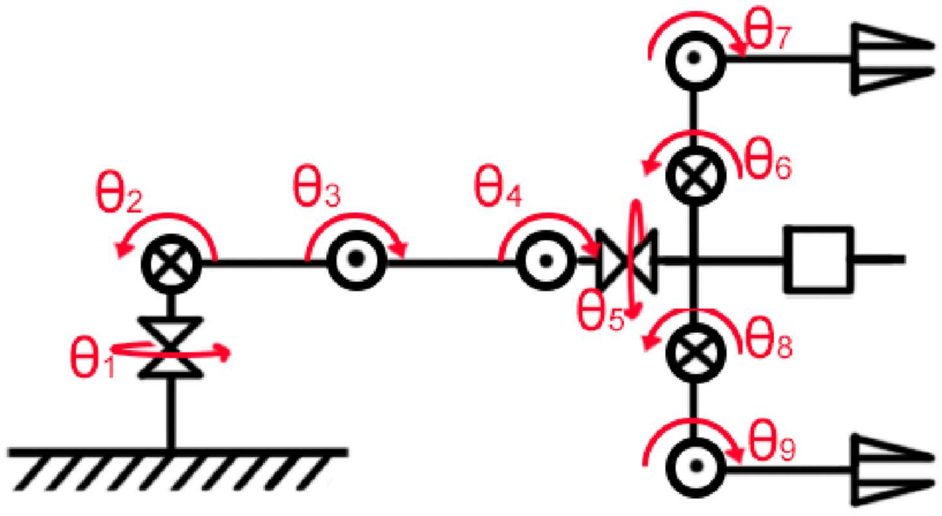


Figure 16- Break down of robot including joint angles

1.5.1 Forward Kinematics

To calculate the forward kinematics, the robot must be broken down into 3 parts with the final result being the superposition of 3 different robots. This would split the robot into the body manipulator with 4-DOF, and two secondary robots for each of the additional manipulators with 3-DOF relative to their mounting point on the main body. The first three transformations are the same for the robots, with the arms with the grippers having slight differences between them depending on their angles. To develop the table the following graph shows the definition for each of the angles and lengths defined for the three sets of DH tables.

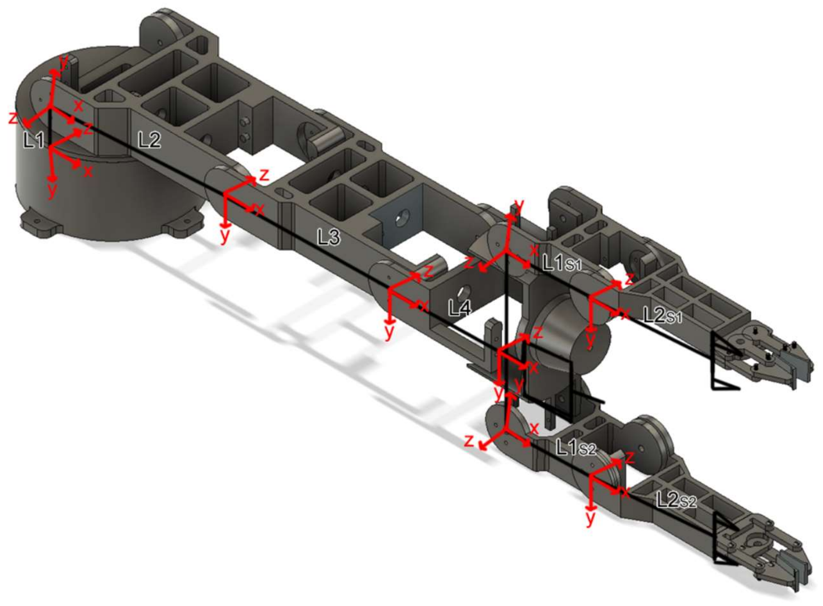


Figure 17 – Robot sketch overlayed with the model, including reference frames for each joint

From this, moving axis by axis the following DH table is derived for the main body of this system,

A	Alpha -1	D-1	Theta
0	0	0	θ_1
L1	90	0	θ_2
L2	180	0	θ_3
L3	0	0	θ_4

With the position of the Drill end effector P_{Drill} located at,

$$ {}^{0}P_{body} = {}^{0}_{1}T_{body} * {}^{1}_{2}T_{body} * {}^{2}_{3}T_{body} * {}^{3}_{4}T_{body} * {}^{4}P_{body} $$

Next, each arm is computed, with the DH tables produced for each secondary effector.

A	Alpha -1	D-1	Theta
L4_Mid	α_1	0	0
0	0	0	θ_6
L1_s1	180	0	θ_7

A	Alpha -1	D-1	Theta
L4_Mid	α_2=180 + α_1	0	0
0	0	0	θ_8
L1_s2	180	0	θ_9

To get the position of the end effector a combination of the main body and sub-arms can be combined to produce the location of an end effector,

$$ {}^{0}P_{Gripper\ 1} = {}^{0}_{1}T_{body} * {}^{1}_{2}T_{body} * {}^{2}_{3}T_{body} * {}^{3}_{4}T_{body} * {}^{0}_{1}T_{arm} * {}^{2}_{1}T_{arm} * {}^{3}_{2}T_{arm} * {}^{3}_{0}P_{Drill} $$

With the end effectors position, as it's a 4-DOF robot, the orientation of the end effector is given by the simple equation of,

$$ \gamma = \theta_3 + \theta_4 - \theta_2 $$

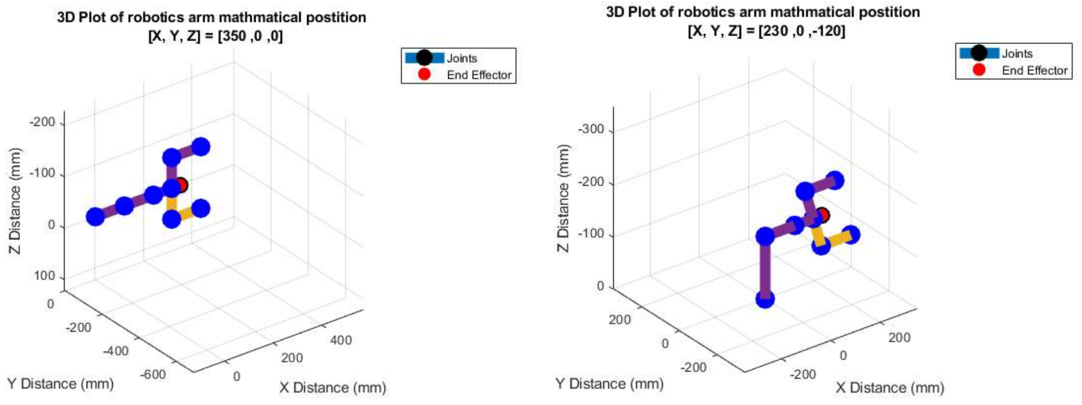


Figure 18 – Plot created using DH table when $\theta_{6/8} = 90, \theta_{7/9} = 90$ *(left) and updated position when* $\theta_2 = 90, \theta_3 = 90\ and\ \theta_5 = 45\ (right)$

1.5.2 Inverse Kinematics – Primary Effector

With the forward kinematics calculated the next step is to calculate the inverse kinematics with a similar approach. Breaking the robot up and calculating it from one effector at a time. This will start with the main end effector's position, which would be the tip of the drill bit. Given the desired end effector position is given by the vector:

$$P_E = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

And that the end effector is given an orientation of γ, which is the total angle away from the X-axis. With this information, the arm can be broken up into a simple geometric problem, where each link can be solved one at a time. This provides the following set of solutions for each of the joint angles defined above for the main actuator.

$$\theta_1 = atand2(Y, X)$$

Next, the position of the second to the last joint is found, this is designated as $P_h = \begin{bmatrix} X_h \\ Y_h \\ Z_h \end{bmatrix}$ and is found as,

$$X_h = P_X - (\cos(\theta_1) * L_4 * \cos(\gamma))$$

$$Y_h = P_Y - (\sin(\theta_1) * L_4 * \cos(\gamma))$$

$$Z_h = P_Z - (L_4 * \sin(\gamma))$$

With these points, the final point angles can be found to be.

$$\beta = atan2\left(Z_h, \sqrt{Y_h^2 + X_h^2}\right)$$

$$\phi = \mathrm{acos}\left(\frac{L_2^2 + Z_h^2 + Y_h^2 + X_h^2 - L_3^2}{2 * L_2 * \sqrt{Z_h + Y_h + X_h}}\right)$$

$$\theta_2 = \beta + \phi$$

$$\sin(\theta_3) = \left(\frac{Z_h^2 + Y_h^2 + X_h^2 - L_3^2 - L_2^2}{2L_3^2 L_2^2}\right)$$

$$\cos(\theta_3) = sqrt(1 - \sin^2(\theta_3))$$

$$\theta_3 = atan2(\sin(\theta_3), \cos(\theta_3))$$

$$\theta_4 = \gamma - (-\theta_2) - \theta_3$$

Within these cases, there is the need for logical solutions as some of the produced angles are calculated with functions that are limited between 0-180 degrees. Some logical reasoning will be needed if the joint limits are increased past the existing 180-degree joint limit.

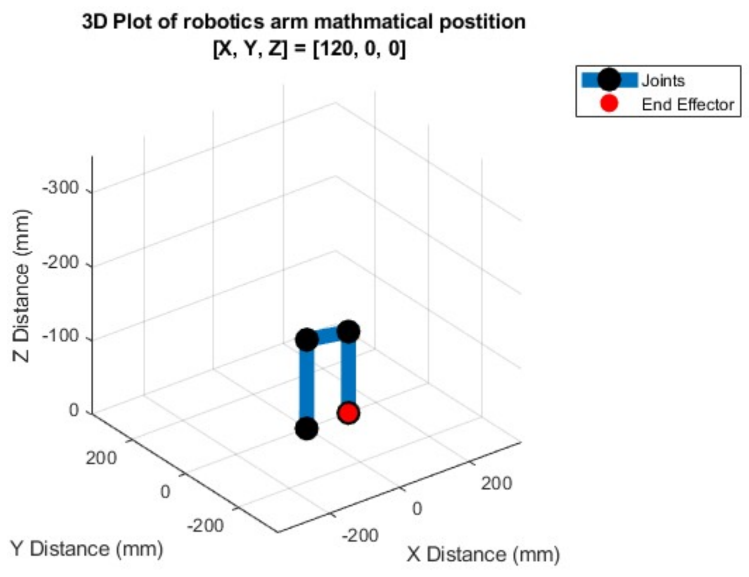


Figure 19 – Plot when the robot is told to the position at location [120. 0 0] with an orientation of 90

1.5.3 Inverse Kinematics – Secondary Effector

To calculate the secondary effector's kinematics, a more in-depth process is needed to be taken due to the relative plane not being the origin. Hence, the joint plane theta 5 is taken as the origin position to calculate these joints. Hence first the coordinates need to be offset and rotated to match this plane. Given an end, the effector is located at the location $P_{SE} = \begin{bmatrix} X_{SE} \\ Y_{SE} \\ Z_{SE} \end{bmatrix}$ and given the location of this rotational joint is given as, $P_{RJ} = \begin{bmatrix} X_{RJ} \\ Y_{RJ} \\ Z_{RJ} \end{bmatrix}$

$$X_{RJ} = P_X - (\cos(\theta_1) * L_4 * \cos(\gamma))$$

$$Y_{RJ} = P_Y - (\sin(\theta_1) * L_4 * \cos(\gamma))$$

$$Z_{RJ} = P_Z - (L_4 * \sin(\gamma))$$

Hence, a fake origin can be set to the point of, $P_{RJ} = \begin{bmatrix} X_{RJ} \\ Y_{RJ} \\ Z_{RJ} \end{bmatrix}$, hence the new relative positions of the effector to PRJ are,

$$P_R = P_{SE} - P_{RJ}$$

Hence a rotational matrix can be applied to place these points in a new plane that is aligned with the orientation. Hence the rotational matrix R is applied,

$$R = \begin{bmatrix} \cos(\theta_1) & -\sin(\theta_1) & 0 \\ \sin(\theta_1) & \cos(\theta_1) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\gamma) & 0 & \sin(\gamma) \\ 0 & 1 & 0 \\ -\sin(\gamma) & 0 & \cos(\gamma) \end{bmatrix}$$

$$P_{SER} = P_R * R = \begin{bmatrix} X_{SER} \\ Y_{SER} \\ Z_{SER} \end{bmatrix}$$

With the points in a new rotational plane, the kinematics can simply be solved one step at a time, hence,

$$\theta_5 = atan2(Y, Z)$$

Next simplifying the system down to a 2D space, a similar method as the main effector can be used to find the final angles,

$$\beta_2 = atan2\left(\sqrt{Y_{SER}^2 + Z_{SER}^2}, X_{SER}^2\right)$$

$$\phi_2 = \text{acos}\left(\frac{L_{6/8}^2 + Z_{SER}^2 + Y_{SER}^2 + X_{SER}^2 - L_3^2}{2 * L_{6/8} * \sqrt{Z_{SER} + Y_{SER} + X_{SER}}}\right)$$

$$\theta_{6/8} = \beta_2 + \phi_2$$

$$\sin(\theta_{7/9}) = \left(\frac{Z_{SER}^2 + Y_{SER}^2 + X_{SER}^2 - L_{7/9}^2 - L_{6/8}^2}{2L_{7/9}^2 L_{6/8}^2}\right)$$

$$\cos(\theta_{7/9}) = sqrt(1 - \sin^2(\theta_{7/9}))$$

$$\theta_{7/9} = atan2(\sin(\theta_{7/9}), \cos(\theta_{7/9}))$$

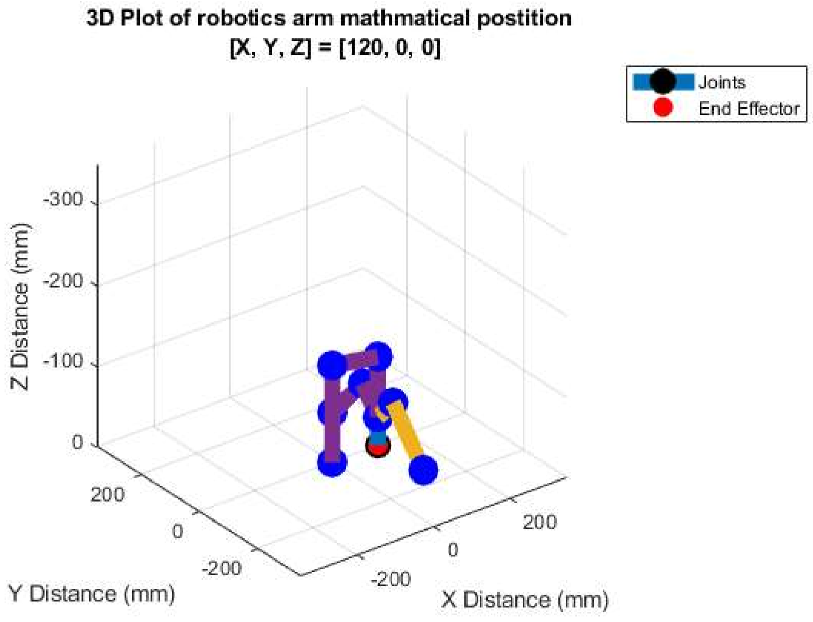


Figure 20 – Location when the primary effector is set at [120, 0, 0] and when secondary effectors are positioned at [120, +/- 160, 0]

1.6 Trajectory and Handling

Similar to the kinematics, to generate and create the desired trajectories the problem has to be decomposed into multiple steps for each of the individual arms and then superimposed to produce the final motion of the robot.

1.6.1 Drill Trajectory

The first, and most important aspect of this robot's trajectory planning is within the smooth trajectory of the primary end effector, the drill. s this effector has 4DOF the trajectory needs to incorporate 4 trajectory profiles that will be generated and executed. Additionally, 3 forms of motion will need to be implemented to execute all the required motions within machining. These are Liner, Nonlinear and circular motion.

Both linear and Non-Linear motion will be executed the same way, through the use of quintic polynomials to generate trajectory profiles for each of the axis and the orientation. Before each motion, these will be pre-generated through the input of the starting and final location of the end effector, along with the desired duration of the motion of the current motion. These are described in the standard quintic form of,

$$f_n(t) = a_0 + a_3t^3 + a_4t^4 + a_5t^5$$

With the constants of the cubic individually calculated as such,

$$a_0 = u_0, a_3 = \frac{10}{t_f^3(u_f - u_0)}, a_4 = -\frac{15}{t_f^4(u_f - u_0)}, a_5 = \frac{6}{t_f^3(u_f - u_0)}$$

Where, u_f and u_0 will be the final and original position/orientation, and t_f is the desired length of the motion of this motion.

In the case of a linear motion, where the kinematics are then calculated for each set and the robot's angles are updated at set intervals. For nonlinear motion, instead of inputting the cartesian coordinates, the two desired joint angles will be inputted instead.

1.6.2 Secondary End Effector Trajectory

Due to the nature of the robot, the secondary End effector's Trajectory will have to be calculated off of the predetermined path of the primary end effector. Within the second end effector, there are 3 Scenarios.

Scenario 1 – Non in use

This scenario is the easiest to deal with. The secondary effectors are placed in their home position and are not used or manipulated. No extra code is needed after they are set in their home position.

Scenario 2 – Not Fixed to an object

Within this scenario, the joint will need to move its end effector's position as the primary effector moves. In effect, the robotic arm is only fixed at the base of the primary manipulator. And is free to move within free space. In this situation the motion will be taken relative to the end effector position of the primary manipulator and hence execute motion relative to its position, allowing for simultaneous motion while the primary end effector is moving to its new location.

Scenario 3 – Fixed to a rigid object

When the second end effector is fixed to a rigid object this will have to be treated as two links fixed to two positions and new angles must be calculated. After the trajectory of the primary effector has been calculated, an algorithm will iterate at each interval and calculate the new angles of the arm per interval with the use of the inverse kinematics. In this case, extra detection will be needed to make sure the desired motion is possible, and skip the step if there are any potential cases of motion that the robot cannot make.

1.7 Coding Implementation

With the initial prototype built, kinematics solved and a basic model for the trajectory of the robot modelled the initial stages of the bulk of the main program can be put together to interface with the motors, and in turn the robot.

1.7.1 – Program Fundamentals

A key aspect of the performance of the robot will be within the optimization of the core programming of the robot. The robot is controlled through a raspberry pi, running Raspbian, a Linux distribution that is optimized for this microcontroller. This has been modified to be compatible with the Adafruit servo controller through inbuilt IC2 us. Furthermore, an external monitor is required to be connected however the firmware to use the Adafruit TFT screen has also been set up for convenience when testing the robot in controlled environments. The program has been written within python with the use of the following libraries:

Tkinter – Provides a basic interface to interact with

Matplotlib – Allow variables to be plotted in real-time, with a similar interface to MATLAB

Numpy – Used to handle matrix operations within python

Math – Handles trigonometric functions

Board/busio/ adafruit_pca9685 – Libraries used to interface with the servos through the adafruit servo hat.

With these imports, various functions and classes are constructed to form a fully object-oriented model, where the robot's movements are saved and stored as an individual object. Each motion updates these variables to keep the code as efficient and computationally simple as possible.

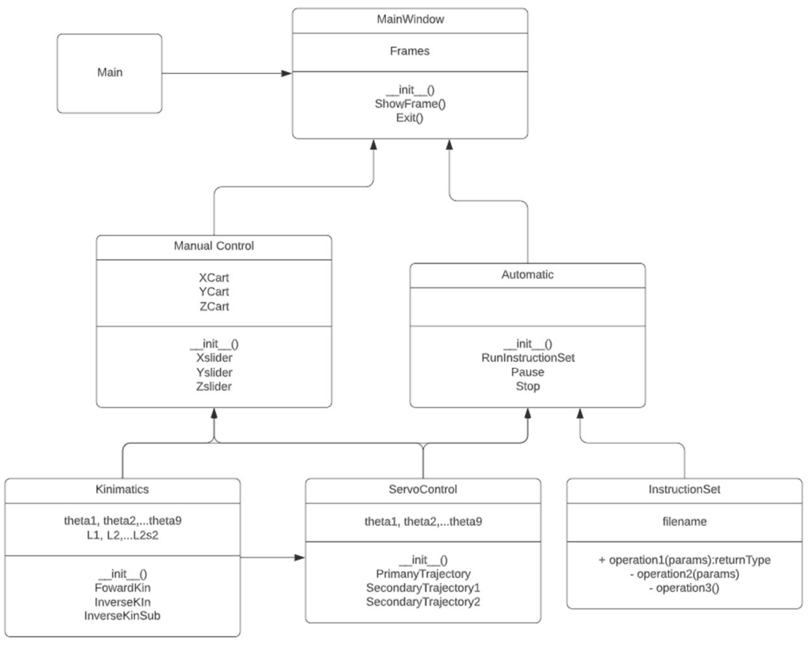
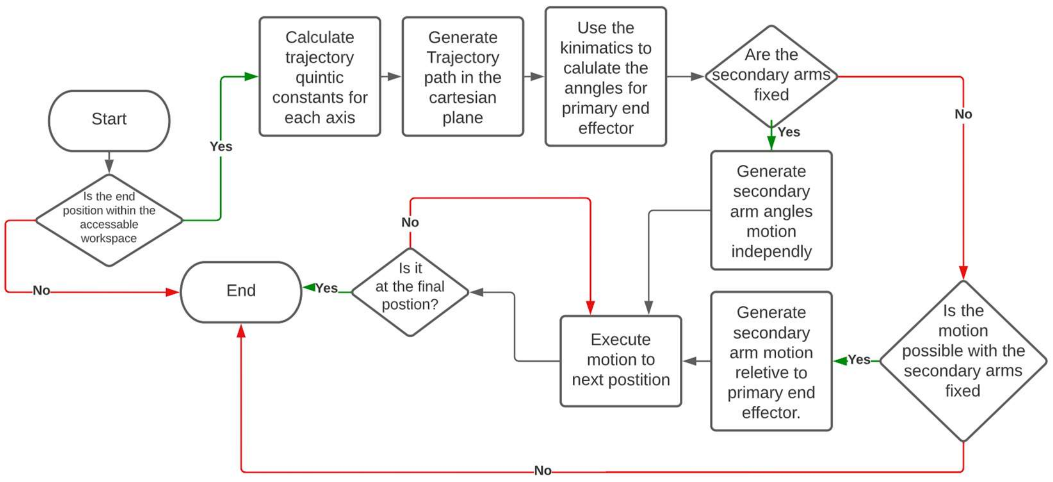


Figure 21 – EML relationship diagram of the key classes within the produced code

1.7.2 – Moving between two set points

The combination of all the kinematics, planning, and logic is needed to instruct the robot. To move between two points the trajectory function is passed the start point, the endpoint, the type of motion and the duration of motion.



Start
Is the end position within the accessable workspace
Yes
No
Calculate trajectory quintic constants for each axis
Generate Trajectory path in the cartesian plane
Use the kinimatics to calulate the anngles for primary end effector
Are the secondary arms fixed
Yes
No
Generate secondary arm angles motion independly
Is the motion possible with the secondary arms fixed
Yes
No
Generate secondary arm motion reletive to primary end effector.
Execute motion to next postition
Is it at the final postion?
No
Yes
End

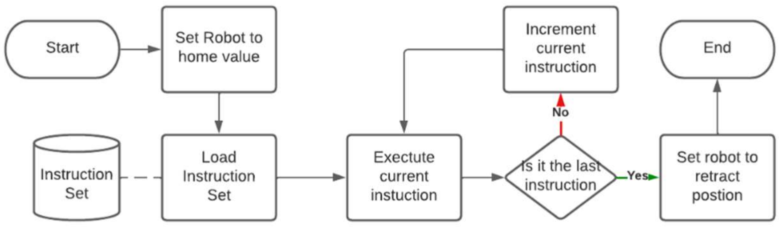
Figure 22 – Flow chart for moving between two points

1.7.3 – Simulating Offline Programming

To simulate how this robot would work with offline programming a basic system has been built allowing instruction sets to be fed to the robot. These are stored in a local CSV file with each line indicating a new instruction. They are stored in the following format,

$$[\text{Motion type},\text{effector},[X_{start},Y_{start},Z_{start}],[X_{end},Y_{end},Z_{end}],t_s]$$

Where motion type is defined if it is nonlinear, linear, or circular encoded as N, L, and C respectively. Effector refers to which effector this instruction set is for, where 1 is the primary and 2/3 are the secondary effectors. Once loaded the basic outline for the method to run the instruction set is seen below, where the execution of the current instruction follows a similar chart to Figure 22.



Start
Set Robot to home value
Instruction Set
Load Instruction Set
Exectute current instuction
Increment current instruction
Is it the last instruction
No
Yes
Set robot to retract postion
End

Figure 23 – Flow chart to execute an instruction set

1.8 Initial testing

While limited in time the initial testing has been positive. The chosen servos can support the load effectively with little to no deviation from the desired position. Each of the robotic links can rotate and position with ease with little friction between the joints. The primary effector can be easily positioned along with the secondary effectors. However, some issues have arisen with sudden jerk motions, or math errors within the kinematic calculations of the sub-arms when placed in specific positions.

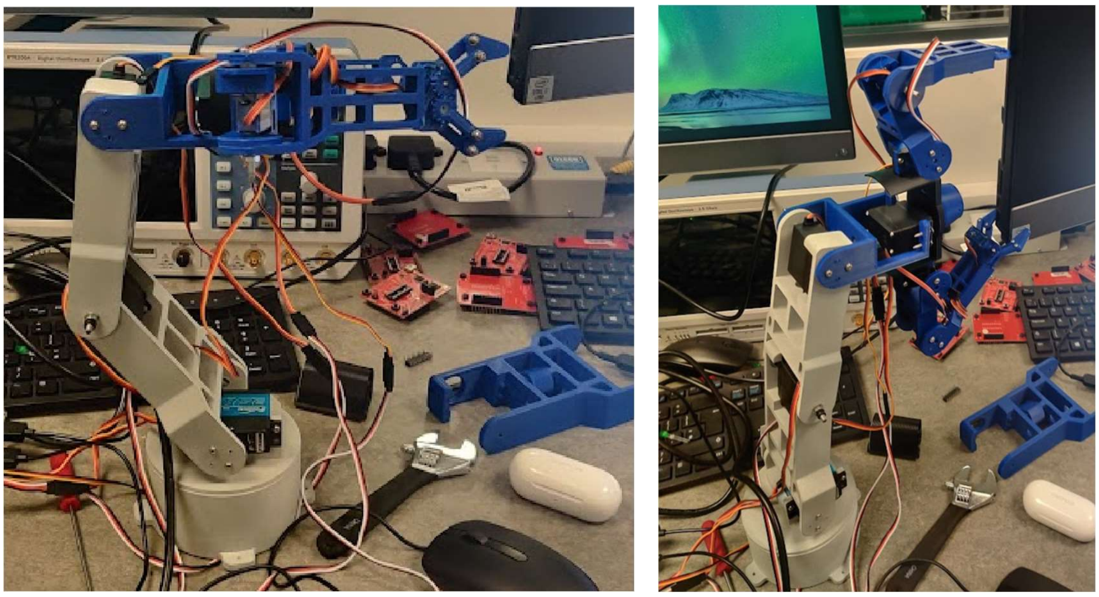


Figure 24 – Initial testing and manipulation of the robot without external aid

As this term has been focused on the design and building of the robot, the next term will focus predominantly on testing, iterating and refining the model. The majority of the programming, along with the fundamental design seems to already provide a significant improvement in the robotics rigidity and mobility when in use.

2 Project Management

2.1 Project Reflection

Overall, compared to the initially laid out schedule within the original project proposal, this project is far ahead of the originally laid out plans. This can be attributed to the roughly equal time distributed for each take, along with an unnecessarily long planning and design stage, which in reality let the manufacturing of components start sooner than later. Currently, a fully functioning arm has been produced, and initial testing of the design has taken place. This means the initially proposed targets; have all been almost fully accomplished within this stage of the project. While the project was a success, the certain task did take a lot longer to complete, these are categorized within the following obstacles,

Delays due to illness - Week 2 and Week 4 saw significant delays in the progress of the project work due to two instances of bed ridding illnesses. This caused a lot of the coding and manufacturing plans to be delayed and pushed back to later in the term.

Delays due to Part ordering - Additional unexpected delays occurred within the manufacturing and ordering time of certain components. Key components, such as the desired high torque servos, took over 3 weeks to ship due to the standard suppliers not having them within the stock. Provided alternatives either were too weak, too large, or too expensive to fit within the project's budget. This is additionally extended due to the current ongoing Royal Mail strikes that delayed the ordering of certain parts even more.

Manufacturing Delays - Finally, there were also large delays in the printing and production of parts. As the robot is being iteratively designed from the ground up and limited number of printers, it took far longer than expected to produce the prototype of the robot.

2.2 Revising the Gantt Chart

With the project already underway and a greater understanding of the time commitment certain aspects of the project take, a revised Gantt chart has been produced that chronicles the currently existing work along with the planned work for the rest of the project into term 2.

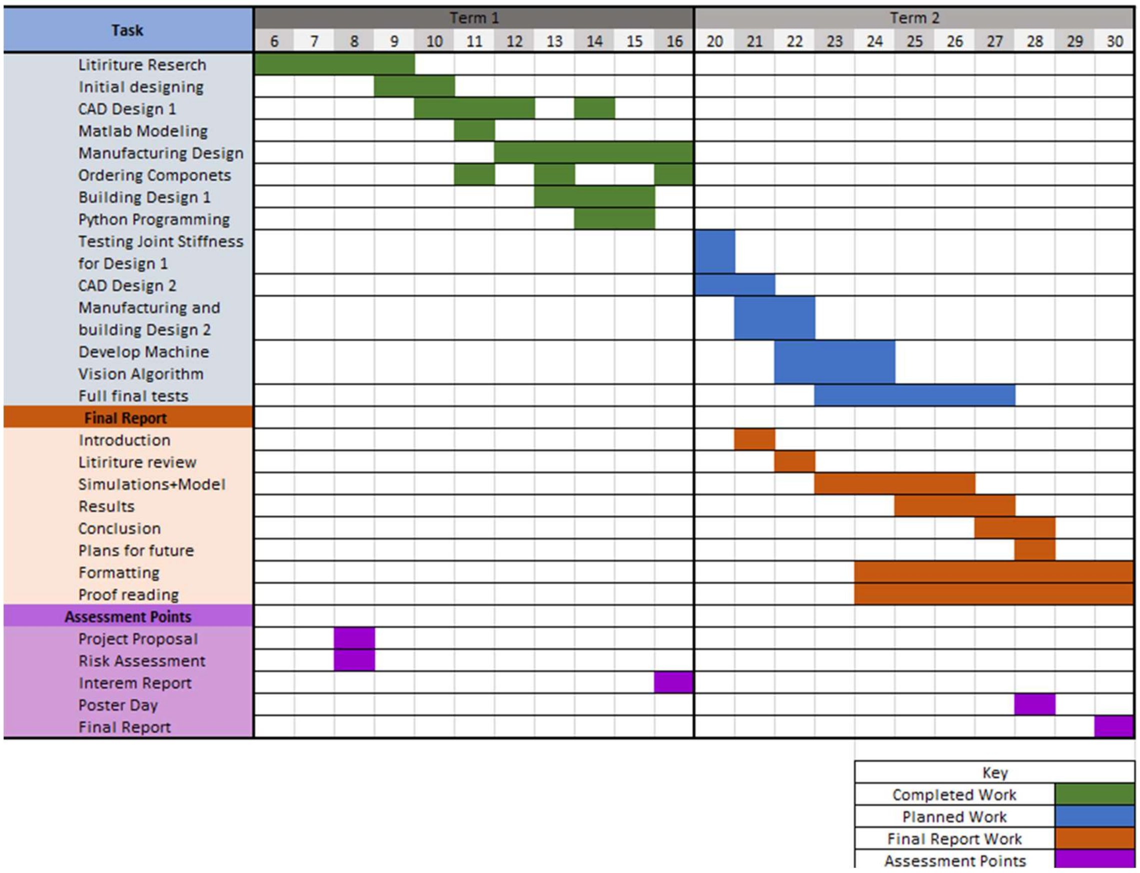


Figure 25 – Revised project Gantt chart

The greatest changes within the current Gannt chart include extra time to modify and manufacture some of the new parts that are not to the current desired standard of the project, allocated time to figure out the machine vision, and a greater timeline for the final report to reduce any potential complications that may arise with writing an extended piece with dyslexia.

3 Future work

The progress of the project has been good, with some of the initial goals stated within the project proposal almost being completed within the objectives set out. With the current progress, the scope of the project has been expanded to three main goals.

3.1 Improve and optimize the mechanical design & Software

The current design, while rigid, does have some physical limitations that limit the ability of the arms to move. These joint limitations mean that the overall workspace of the current design is limited, and a redesign of these parts with the robot's joint mobility in mind would significantly improve the arms workspace and overall ability to manipulate, interact, and machine objects within its workspace. Additionally Reducing the overall load on the processor of the driving microcontroller will provide the main advantage of allowing a faster throughput of calculation and joint adjustments. This will reduce the overall jitter within the motion of the joints, along with improving the accuracy of the motion of joints while moving between two points.

3.2 Test the Joint Stiffness

The next step is to thoroughly test the current, and the improved design against the initial set of specifications. This will require a basic setup that will allow the robot to interact with an "object" that will be covered in pressure sensors. The two methods that are currently planned to measure joint rigidity are the use of a newton meter placed in a fixed position, and the placement of piezoelectric pressure sensors to detect applied force to a single point.

3.3 Introducing machine vision

To expand the scope of the project, the introduction of machine vision would add a new, novel take on the current produced research. The idea would be to mount a camera on a static mount or upon a fixed point near the end effector. From there the camera can capture an image, process it, and identify the object. From there it can calculate an adjustment and hence dynamically move the end effectors to an optimal position concerning the object. This means that the extra end effectors won't need to be pre-programmed for use and will dynamically find what points they can attach to increase the rigidity of a given operation. It would also make this robot far more robust within anomalous situations.

3.4 Final report & Poster planning

This term will also include the two large submissions that need to appropriately plan and delegate. The first is the poster show that will be passively worked through the term. It ties in well with the final report as many of the graphics and graphs that will be used within the poster showcase will also be used within the final report. The latter is the Final report, which has been allocated more time to reduce the chances of mistakes and difficulties that may arise due to dyslexia. Additionally, multiple reviews, spell checks, and review checkpoints have been planned to make sure that the final report is made to a high standard.

4 Reference

[1] A. Raoofian, A. Taghvaeipour and A. Kamali, "On the stiffness analysis of robotic manipulators and calculation of stiffness indices," *Mechanism and Machine Theory, Volume 130,* 2018.

[2] C. Dong, H. Liu, W. Yue and T. Huang, "Stiffness modeling and analysis of a novel 5-DOF hybrid robot," *Mechanism and Machine Theory, Volume 125,* pp. 80-93, 2018.

[3] C. Dumas, S. Caro, C. Mehdi, S. Garnier and B. Furet, "Joint Stiffness Identification of Industrial Serial Robots," *HAL Open Science,* 2011.

[4] J.-Y. K'NEVEZ, M. CHERIF, M. ZAPCIU and A. GERARD, "EXPERIMENTAL CHARACTERIZATION OF ROBOT ARM RIGIDITY," *Proceedings in Manufacturing Systems, Vol. 5,* 2010.

[5] K. Wu, J. Li, H. Zhao and Y. Zhong, "Review of Industrial Robot Stiffness Identification," *Appl. Sci., 12, 8719.,* 2022.

[6] C. Y. Lai, D. E. V. Chavez and S. Ding , "Transformable parallel-serial manipulator for robotic machining," *The International Journal of Advanced Manufacturing Technology,* vol. 97, 2018.

[7] L. Romdhane, "Design and analysis of a hybrid serial-parallel manipulator," *Mechanism and Machine Theory, Volume 34, Issue 7,* 1999.

[8] S. Kumar, H. Wöhrle and J. d. G. Fernández, "A survey on modularity and distributivity in series-parallel hybrid robots," *Institute of Robotics,* 2018.

[9] N. M. Bajaj, A. J. Spiers and . A. M. Dollar, "State of the Art in Artificial Wrists: A Review of Prosthetic and Robotic Wrist Design," *IEEE TRANSACTIONS ON ROBOTICS, Volume 35,* pp. 261-277, 2019.

[10] V. L. Nguyen, C.-H. Kuo and P. T. Lin, "Compliance error compensation of a robot end-effector with joint stiffness uncertainties for milling: An analytical model," *Mechanism and Machine Theory, Volume 170,* 2022.

[11] J. Li, K. Mi and Z. Wu, "Tracking control via switching and learning for a class of uncertain flexible joint robots with variable stiffness actuators," *Neurocomputing, Volume 469,* 2022.

[12] J. Hunt and H. Lee, "Optimizing the Rigid or Compliant Behavior of a Novel Parallel-Actuated Architecture for Exoskeleton Robot Applications.," *Front. Robot. AI 8:596958,* 2021.

5 Appendices

5.1 Appendix 1: Forward Kinematics MATLAB Code

```
clear all
close all


%Set Values of link lengths(mm)
L1 = 0;
L2 = 120;
L3 = 120;
L4 = 110;

Mid_Joint = 75;

L_2_1 = Mid_Joint;
L_2_2 = 0;
L_2_3 = 60;
L_2_4 = 120;

%Desired Angles of each joint
Theta1 = 0;
Theta2 = 0;
Theta3 = 0;
Theta4 = 0;

Theta5 = 0;
Theta6 = 90;
Theta7 = 90;
Theta8 = Theta6;
Theta9 = -Theta7;

%%%-----------Main Arm -------
%Creates DH Tables from provided values
a_i =[0;0;L1;L2;L3];
alpha_i =[0;0;-90;180;0];
d_i = [0;0;0;0;0];
Theta_i = [0;Theta1;Theta2;Theta3;Theta4];

DHTable = horzcat(a_i,alpha_i,d_i,Theta_i);

%Vector for end effector
End_effector = [L4; 0;0; 1];


%Creates transformation matrixs
T_01 = Ken_Transform(1,DHTable);
T_12 = Ken_Transform(2,DHTable);
T_23 = Ken_Transform(3,DHTable);
T_34 = Ken_Transform(4,DHTable);

%Creates single transform matrix T_03
T_03 = T_01*T_12*T_23*T_34;

%Finds cart position of end effector
P_0 = T_03*End_effector
Gamma = -Theta2+Theta3+Theta4;

%Finds the cart position for the other joins
P_1 = T_01*[L1; 0; 0; 1];
P_2 = T_01*T_12*[L2; 0; 0; 1];
P_3 = T_01*T_12*T_23*[L3; 0; 0; 1];
```

```
%%%-----------SubArm -------

%Solve arm joint 1

a_i_extra =     [a_i      ;L_2_1    ;L_2_2    ;L_2_3];
alpha_i_extra = [alpha_i  ;Theta5       ;0      ;180];
d_i_extra =     [d_i;0        ;0        ;0];
Theta_i_extra = [Theta_i       ;0   ;Theta6   ;Theta7];



DHTable_Extra = horzcat(a_i_extra,alpha_i_extra,d_i_extra,Theta_i_extra)

TExtra_01 = Ken_Transform(1,DHTable_Extra);
TExtra_12 = Ken_Transform(2,DHTable_Extra);
TExtra_23 = Ken_Transform(3,DHTable_Extra);
TExtra_34 = Ken_Transform(4,DHTable_Extra);
TExtra_45 = Ken_Transform(5,DHTable_Extra);
TExtra_56 = Ken_Transform(6,DHTable_Extra);
TExtra_67 = Ken_Transform(7,DHTable_Extra);

P_0Extra =
TExtra_01*TExtra_12*TExtra_23*TExtra_34*TExtra_45*TExtra_56*TExtra_67*[L_2_4;0;0;1]
;

P_Extra1 = TExtra_01*[a_i_extra(3);0;0;1];
P_Extra2 = TExtra_01*TExtra_12*[a_i_extra(4);0;0;1];
P_Extra3 = TExtra_01*TExtra_12*TExtra_23*[a_i_extra(5);0;0;1];
P_Extra4 = TExtra_01*TExtra_12*TExtra_23*TExtra_34*[a_i_extra(6);0;0;1];
P_Extra5 = TExtra_01*TExtra_12*TExtra_23*TExtra_34*TExtra_45*[a_i_extra(7);0;0;1];
P_Extra6 =
TExtra_01*TExtra_12*TExtra_23*TExtra_34*TExtra_45*TExtra_56*[a_i_extra(8);0;0;1];

Loc = [P_Extra1 P_Extra2 P_Extra3 P_Extra4 P_Extra5 P_Extra6 P_0Extra]






%Solve arm joint 2

a_i_A2 =     [a_i(1:5)      ;L_2_1    ;L_2_2    ;L_2_3];
alpha_i_A2 = [alpha_i(1:5)  ;180+Theta5       ;0      ;0];
d_i_A2 =     [d_i(1:5)      ;0        ;0        ;0];
Theta_i_A2 = [Theta_i       ;0   ;Theta6   ;Theta7];



DHTable_Extra2 = horzcat(alpha_i_A2,a_i_A2,d_i_A2,Theta_i_A2)

TA2_01 = Ken_Transform(1,DHTable_Extra2);
TA2_12 = Ken_Transform(2,DHTable_Extra2);
TA2_23 = Ken_Transform(3,DHTable_Extra2);
TA2_34 = Ken_Transform(4,DHTable_Extra2);
TA2_45 = Ken_Transform(5,DHTable_Extra2);
TA2_56 = Ken_Transform(6,DHTable_Extra2);
TA2_67 = Ken_Transform(7,DHTable_Extra2);

P_TA2_0 = TA2_01*TA2_12*TA2_23*TA2_34*TA2_45*TA2_56*TA2_67*[L_2_4;0;0;1];

P_TA2_1 = TA2_01*[a_i_A2(3);0;0;1];
P_TA2_2 = TA2_01*TA2_12*[a_i_A2(4);0;0;1];
P_TA2_3 = TA2_01*TA2_12*TA2_23*[a_i_A2(5);0;0;1];
```

```
P_TA2_4 = TA2_01*TA2_12*TA2_23*TA2_34*[a_i_A2(6);0;0;1];
P_TA2_5 = TA2_01*TA2_12*TA2_23*TA2_34*TA2_45*[a_i_A2(7);0;0;1];
P_TA2_6 = TA2_01*TA2_12*TA2_23*TA2_34*TA2_45*TA2_56*[a_i_A2(8);0;0;1];

Loc_A2 = [P_TA2_1 P_TA2_2 P_TA2_3 P_TA2_4 P_TA2_5 P_TA2_6 P_TA2_0]




%Converts all the points in to a 3d Matirx
figure()
X = [0; P_1(1) ; P_2(1); P_3(1);  P_0(1)];
Y = [0; P_1(2) ; P_2(2); P_3(2);  P_0(2)];
Z = [0; P_1(3) ; P_2(3);P_3(3);  P_0(3)];


X_2 = Loc(1,:)
Y_2 = Loc(2,:)
Z_2 = Loc(3,:)

X_3 = Loc_A2(1,:)
Y_3 = Loc_A2(2,:)
Z_3 = Loc_A2(3,:)

%Prints Cart position of end effector
P_0
Gamma
%Creates 3D plot of robot arm
plot3 (X,Y,Z,'-o','LineWidth',8,'MarkerSize',
5,'MarkerFaceColor','k','MarkerEdgeColor','k')
hold on
plot3 (P_0(1),P_0(2),P_0(3),'o','MarkerSize',
9,'MarkerFaceColor','r','MarkerEdgeColor','r')

plot3 (X_2,Y_2,Z_2,'-o','LineWidth',8,'MarkerSize',
7,'MarkerFaceColor','b','MarkerEdgeColor','b')
%plot3 (X_3,Y_3,Z_3,'-o','LineWidth',8,'MarkerSize',
7,'MarkerFaceColor','b','MarkerEdgeColor','b')

title(["3D Plot of robotics arm mathmatical postition "," [X, Y, Z] = ["+P_0(1)+"
,"+P_0(2)+" ,"+P_0(3)+"]"])
grid on
Lim_val = (L1+L2+L3+L4)*1;
ylim([-Lim_val;Lim_val])
xlim([-Lim_val;Lim_val])
zlim([-Lim_val;0])
xlabel("X Distance (mm)")
ylabel("Y Distance (mm)")
zlabel("Z Distance (mm)")
set(gca, 'ZDir','reverse')
legend("Joints","End Effector")

%Function that calculates transformation matrix from a given dh table and
%joint index


function T = Ken_Transform(i,DHTable)
    i = i+1;

    %takes correct values from provided DH tables
    theta_i = DHTable(i,4);
    alpha = DHTable(i,2);
    a_link = DHTable(i,1);
    d_i = DHTable(i,3);

    %creates and returns the correct transformation matrix using the
    %general formula
    T = [cosd(theta_i) -sind(theta_i) 0 a_link;
```

```
        cosd(alpha)*sind(theta_i) cosd(alpha)*cosd(theta_i) -sind(alpha) -
sind(alpha)*d_i;
         sind(alpha)*sind(theta_i) sind(alpha)*cosd(theta_i) cosd(alpha)
cosd(alpha)*d_i;
        0 0 0 1];
end
```

5.2 Appendix 2: Inverse Kinematics MATLAB Code

```
clear all
close all


%Set Values of link lengths(mm)
L1 = 0;
L2 = 120;
L3 = 120;
L4 = 110;

Mid_Joint = 75;

L_2_1 = Mid_Joint;
L_2_2 = 0;
L_2_3 = 60;
L_2_4 = 120;

%Desired Angles of each joint
Theta1 = 0;
Theta2 = 0;
Theta3 = 0;
Theta4 = 0;

Theta5 = 0;
Theta6 = 90;
Theta7 = 90;
Theta8 = Theta6;
Theta9 = -Theta7;

%%%-----------Main Arm -------
%Creates DH Tables from provided values
a_i =[0;0;L1;L2;L3];
alpha_i =[0;0;-90;180;0];
d_i = [0;0;0;0;0];
Theta_i = [0;Theta1;Theta2;Theta3;Theta4];

DHTable = horzcat(a_i,alpha_i,d_i,Theta_i);

%Vector for end effector
End_effector = [L4; 0;0; 1];


%Creates transformation matrixs
T_01 = Ken_Transform(1,DHTable);
T_12 = Ken_Transform(2,DHTable);
T_23 = Ken_Transform(3,DHTable);
T_34 = Ken_Transform(4,DHTable);

%Creates single transform matrix T_03
T_03 = T_01*T_12*T_23*T_34;

%Finds cart position of end effector
P_0 = T_03*End_effector
Gamma = -Theta2+Theta3+Theta4;

%Finds the cart position for the other joins
```

```
P_1 = T_01*[L1; 0; 0; 1];
P_2 = T_01*T_12*[L2; 0; 0; 1];
P_3 = T_01*T_12*T_23*[L3; 0; 0; 1];




%%%-----------SubArm -------

%Solve arm joint 1

a_i_extra =     [a_i      ;L_2_1    ;L_2_2    ;L_2_3];
alpha_i_extra = [alpha_i  ;Theta5       ;0      ;180];
d_i_extra =     [d_i;0        ;0        ;0];
Theta_i_extra = [Theta_i       ;0   ;Theta6   ;Theta7];



DHTable_Extra = horzcat(a_i_extra,alpha_i_extra,d_i_extra,Theta_i_extra)

TExtra_01 = Ken_Transform(1,DHTable_Extra);
TExtra_12 = Ken_Transform(2,DHTable_Extra);
TExtra_23 = Ken_Transform(3,DHTable_Extra);
TExtra_34 = Ken_Transform(4,DHTable_Extra);
TExtra_45 = Ken_Transform(5,DHTable_Extra);
TExtra_56 = Ken_Transform(6,DHTable_Extra);
TExtra_67 = Ken_Transform(7,DHTable_Extra);

P_0Extra =
TExtra_01*TExtra_12*TExtra_23*TExtra_34*TExtra_45*TExtra_56*TExtra_67*[L_2_4;0;0;1]
;

P_Extra1 = TExtra_01*[a_i_extra(3);0;0;1];
P_Extra2 = TExtra_01*TExtra_12*[a_i_extra(4);0;0;1];
P_Extra3 = TExtra_01*TExtra_12*TExtra_23*[a_i_extra(5);0;0;1];
P_Extra4 = TExtra_01*TExtra_12*TExtra_23*TExtra_34*[a_i_extra(6);0;0;1];
P_Extra5 = TExtra_01*TExtra_12*TExtra_23*TExtra_34*TExtra_45*[a_i_extra(7);0;0;1];
P_Extra6 =
TExtra_01*TExtra_12*TExtra_23*TExtra_34*TExtra_45*TExtra_56*[a_i_extra(8);0;0;1];

Loc = [P_Extra1 P_Extra2 P_Extra3 P_Extra4 P_Extra5 P_Extra6 P_0Extra]







%Solve arm joint 2

a_i_A2 =     [a_i(1:5)      ;L_2_1    ;L_2_2    ;L_2_3];
alpha_i_A2 = [alpha_i(1:5)  ;180+Theta5       ;0      ;0];
d_i_A2 =     [d_i(1:5)      ;0        ;0        ;0];
Theta_i_A2 = [Theta_i       ;0   ;Theta6   ;Theta7];



DHTable_Extra2 = horzcat(alpha_i_A2,a_i_A2,d_i_A2,Theta_i_A2)

TA2_01 = Ken_Transform(1,DHTable_Extra2);
TA2_12 = Ken_Transform(2,DHTable_Extra2);
TA2_23 = Ken_Transform(3,DHTable_Extra2);
TA2_34 = Ken_Transform(4,DHTable_Extra2);
TA2_45 = Ken_Transform(5,DHTable_Extra2);
TA2_56 = Ken_Transform(6,DHTable_Extra2);
TA2_67 = Ken_Transform(7,DHTable_Extra2);

P_TA2_0 = TA2_01*TA2_12*TA2_23*TA2_34*TA2_45*TA2_56*TA2_67*[L_2_4;0;0;1];
```

```
P_TA2_1 = TA2_01*[a_i_A2(3);0;0;1];
P_TA2_2 = TA2_01*TA2_12*[a_i_A2(4);0;0;1];
P_TA2_3 = TA2_01*TA2_12*TA2_23*[a_i_A2(5);0;0;1];
P_TA2_4 = TA2_01*TA2_12*TA2_23*TA2_34*[a_i_A2(6);0;0;1];
P_TA2_5 = TA2_01*TA2_12*TA2_23*TA2_34*TA2_45*[a_i_A2(7);0;0;1];
P_TA2_6 = TA2_01*TA2_12*TA2_23*TA2_34*TA2_45*TA2_56*[a_i_A2(8);0;0;1];

Loc_A2 = [P_TA2_1 P_TA2_2 P_TA2_3 P_TA2_4 P_TA2_5 P_TA2_6 P_TA2_0]




%Converts all the points in to a 3d Matirx
figure()
X = [0; P_1(1) ; P_2(1); P_3(1);  P_0(1)];
Y = [0; P_1(2) ; P_2(2); P_3(2);  P_0(2)];
Z = [0; P_1(3) ; P_2(3);P_3(3);  P_0(3)];


X_2 = Loc(1,:)
Y_2 = Loc(2,:)
Z_2 = Loc(3,:)

X_3 = Loc_A2(1,:)
Y_3 = Loc_A2(2,:)
Z_3 = Loc_A2(3,:)

%Prints Cart position of end effector
P_0
Gamma
%Creates 3D plot of robot arm
plot3 (X,Y,Z,'-o','LineWidth',8,'MarkerSize',
5,'MarkerFaceColor','k','MarkerEdgeColor','k')
hold on
plot3 (P_0(1),P_0(2),P_0(3),'o','MarkerSize',
9,'MarkerFaceColor','r','MarkerEdgeColor','r')

plot3 (X_2,Y_2,Z_2,'-o','LineWidth',8,'MarkerSize',
7,'MarkerFaceColor','b','MarkerEdgeColor','b')
%plot3 (X_3,Y_3,Z_3,'-o','LineWidth',8,'MarkerSize',
7,'MarkerFaceColor','b','MarkerEdgeColor','b')

title(["3D Plot of robotics arm mathmatical postition "," [X, Y, Z] = ["+P_0(1)+"
,"+P_0(2)+" ,"+P_0(3)+"]"])
grid on
Lim_val = (L1+L2+L3+L4)*1;
ylim([-Lim_val;Lim_val])
xlim([-Lim_val;Lim_val])
zlim([-Lim_val;0])
xlabel("X Distance (mm)")
ylabel("Y Distance (mm)")
zlabel("Z Distance (mm)")
set(gca, 'ZDir','reverse')
legend("Joints","End Effector")

%Function that calculates transformation matrix from a given dh table and
%joint index


function T = Ken_Transform(i,DHTable)
    i = i+1;

    %takes correct values from provided DH tables
    theta_i = DHTable(i,4);
    alpha = DHTable(i,2);
    a_link = DHTable(i,1);
    d_i = DHTable(i,3);
    %creates and returns the correct transformation matrix using the
    %general formula
    T = [cosd(theta_i) -sind(theta_i) 0 a_link;
        cosd(alpha)*sind(theta_i) cosd(alpha)*cosd(theta_i) -sind(alpha) -
sind(alpha)*d_i;
         sind(alpha)*sind(theta_i) sind(alpha)*cosd(theta_i) cosd(alpha)
cosd(alpha)*d_i;
        0 0 0 1];
end
```