Development of a Parallel Robot Manipulator

by

Senarath Mudalige Don Mirantha Rangara Bernadeen Jayathilaka

A capstone project submitted in partial fulfillment of the requirements for the degree of Bachelor of Science in
Mechatronics Engineering

Examination Committee: Dr. Manukid Parnichkun (Chairperson)
Dr. Erik L.J.Bohez
Dr. Than Lin

Nationality: Sri Lankan

Asian Institute of Technology
School of Engineering Technology
Thailand
May 2015

Acknowledgement

First of all I want to praise God for giving me strength in well-being that was very much needed to complete this study and my parents for their constant attention and care all throughout.

I intend to express my sincere thanks to Dr.Manukid Parnichkun, my adviser and chairperson in this study, for sharing his expertise and valuable guidance during the study. I'm also grateful to the professors in my capstone committee and all the other faculty members for their help and support. Last but not least, I express my gratitude to all my friends who directly or indirectly provided their valuable assistance during this venture.

Abstract

Parallel manipulation can offer a number of advantages over conventional serial manipulation with respect to optimising operational speed while retaining high accuracy in positioning. In this study a parallel robot manipulator was developed to achieve precise end effector positioning in a three-dimensional workspace. The mechanical assembly, the kinematics analysis and the control programme were built and composed from scratch and several applications were executed in the end to test the operational capabilities of the robot.

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Background

The use of robot manipulators in both industrial and domestic environments has become extensively popular during the recent past. Factors such as efficiency, consistency, accuracy and long term high productivity are driving their usage forward very rapidly. Almost all of the giant scale production lines in the world today are consisted of some degree of automation.

The current commercial robotic manipulators can be classified into two main categories, namely, serial manipulators and parallel manipulators. Serial manipulators are the most common industrial robots. They are designed as a series of links connected by motor-actuated joints that extend from a base to an end effector. Often they have an anthropomorphic arm structure described as having a ”shoulder”, an ”elbow”, and a ”wrist”. Whereas in parallel manipulators, the end effector of this linkage is connected to its base by a number of separate and independent linkages working in parallel.

Most of the serial manipulators in the industry today face the challenge of maintaining high accuracy with optimal working speeds. Because of the ability of parallel robots to execute a task with higher accuracy and speed compared to a serial manipulator, the application of these is becoming quite significant. So in this study the development of a working model of a parallel robot manipulator will be done which will have a wide range of applications in the industry.

1.2 Problem Statement

- The wide use of industrial robot manipulators is often faced with the challenge of optimum efficiency along with accuracy. At these instances the application of parallel manipulators will resolve this challenge up to a considerable extent.
- It was intended to improve precise positioning by accurate inverse kinematics and practically compensating for the singularities of the manipulator.
- As the study progresses some short comings of the actuators used to build the prototype might surface. These are planned to be addressed on a practical basis.

1.3 Objectives

1. Development of a working prototype of a parallel manipulator.
2. Application of accurate inverse kinematics to the system to obtain precise end effector positioning.
3. Proper execution of control algorithms to define the path of the robot.

1.4 Limitations and Scope

1. The prototype will consist of only three degrees of freedom that will limit the work space to a particular range.
2. Some of the physical parameter such as friction between surfaces and signal delays will not be considered in the calculations, but these will be addressed practically.
3. The operation speed will be limited according to the used motor specifications.

Chapter 2

Literature Review

2.1 Information

The invention of parallel manipulators under theoretical basis runs decades back but the emergence of these on industrial scale comes forth year later. Diverging from the conventional serial manipulators, that possessed large workspaces and dexterous maneuverability, parallel manipulators were able to distinguishable abilities due to there design aspects. Serial manipulators lacked precision and contained many other problems due to bending at high load and vibrations occurring at the joints. On the other hand these parallel manipulators offered enhanced capabilities with regard to precision and speed. A desirable comparison between serial and parallel manipulators was presented in Y. D. Patel's and P. M. George's paper *Parallel Manipulators ApplicationsA Survey*. A summary is given below in figure 2.1. According to their study, parallel kinematic manipulators offer several advantages over their serial counterparts for certain applications. Among the advantages are greater load carrying capacities as total load can be shared by number of parallel links connected to fixed base, low inertia, higher structural stiffness, reduced sensitivity to certain errors, easy controlling and built-in redundancy but smaller and less dexterous workspace due to link interference, physical constraints of universal and spherical joints and range of motion of actuators and suffer from platform singularities. The abundant use of multi DOF spherical and universal joints in parallel manipulator not only simplify the kinematics, but they also make sure that the legs in the Stewart platforms [1] experience only compressive or tensile loads, but no shear forces or bending and torsion moments. This reduces the deformation of the platform, even under high loads. The fully parallel designs of robots have all actuators in or near the base, which results in a very low inertia of the part of the robot that has actually to be moved. Hence, a higher bandwidth can be achieved with the same actuation power.

It is seen that for a parallel kinematic mechanism, the kinematic equations will be considerably more complex due to the closed kinematic loops, than for an open (serial) kinematic structure. In contrast to serial manipulators, there can be presence of unactuated or passive joints. The presence of passive joints and multi DOF joints makes the kinematic analysis very different from kinematic analysis of serial manipulators. Direct kinematics is much harder and involves elimination of passive joint variables in parallel kinematics, although inverse kinematics would be much simpler in parallel mechanisms. The application of both forward and inverse kinematics on a Stewart parallel mechanism was presented in the paper *The forward and inverse kinematics problems for Stewart parallel mechanism* by Domagoj Jakobovic and Leonardo Jelenkovic which concludes that inverse kinematics analysis is much more simpler on a parallel mechanisms since the forwards analysis can produce multiple solutions for a given configuration.

Parallel robots are intrinsically more accurate than serial robots because their errors are averaged instead of added cumulatively due to many parallel links as well as closed loop architecture, in turn making them feasible for many accuracy driven applications.

[1] A Stewart platform is a type of parallel robot that incorporates six prismatic actuators, commonly hydraulic jacks. These actuators are mounted in pairs to the mechanism's base, crossing over to three mounting points on a top plate. Devices placed on the top plate can be moved in the six degrees of freedom in which it is possible for a freely-suspended body to move. These are the three linear movements x, y, z (lateral, longitudinal and vertical), and the three rotations pitch, roll, and yaw.

	Parallel manipulator	Serial manipulator
Type of manipulators	Closed loop	Open loop
End effectors	Platform	Gripper
Natural description	In Cartesian space	In joint space
Location of actuators	Near the immobile base	On the links
Inertia forces & stiffness	Less and high respectively	High and less respectively
Design considerations	Structure, workspace considerations, singularities, link interference	Strength and stiffness considerations, vibration characteristics
Preferred property	Stiffness	Dexterity
Use of direct kinematics	Difficult and complex	Straightforward and unique
Use of inverse kinematics	Straightforward and unique	Complicated
Singularity	Static	Kinematic
Direct force transformation	Well defined and unique	Not well defined; may be non-existent, unique or infinite
Preferred application	Precise positioning	Gross motion

Figure 2.1: Comparison between serial and parallel manipulators.

The potential of high industrial use of parallel robots particularly in areas like high speed pick-and-place is quite evident by the increasing production of this sort by leading industrial robot manufacturers such as ABB and FANUC. ABB's robot *FlexPicker* and FANUC's robot *M-2iA Delta Robot* are good examples of it.

The applications of parallel robots as depicted in numerous sources cover a wide range. Areas covering, hexapod (walking mechanisms), delata robots, space and satellites, mining machines, vehicle suspension, aircraft simulation, telescope positioning and pointing devices, cable actuated cameras and many applications in the medical field are some of the existing applications in both commercial and research levels. Added to that the current interest in automated painting, writing and 3-dimensional printing as seen the light a these parallel robot mechanisms immensely.

Thus the potential of parallel robots mechanisms is quite significant and this study is intended to investigate deeper into this sector and produce more effective and precise results.

Since parallel robots are developed in a number of different designs it was important to select a design for this prototype so that it could be built using the resources available. Searching across many sources of information, Jonathan Keep's design of a robot commonly known as a linear delta robot, which was developed for 3D printing using ceramic was drawn into consideration. In his blog the information of his project of the linear robot is presented along with the materials and tools he had used in the process. The most significant feature about this design was that it was simple and effective. The design was attractive and steady. For actuation, stepper motors were used along with an Arduino stepper driver circuit. This was one thing that didn't fit the prototype design of this study since the actuation was intended to obtain through DC motors and encoders while the Arduino control code had to be developed from scratch. The end effector of Jonathan Keep's design was holding a nozzel for the ceramic molten material to flow through and this was something which was unnecessary for this study. Overall this source was very much used in developing the design features of this study and hence acknowledged.

The same design had been executed in different names, prominently used for applications of 3D

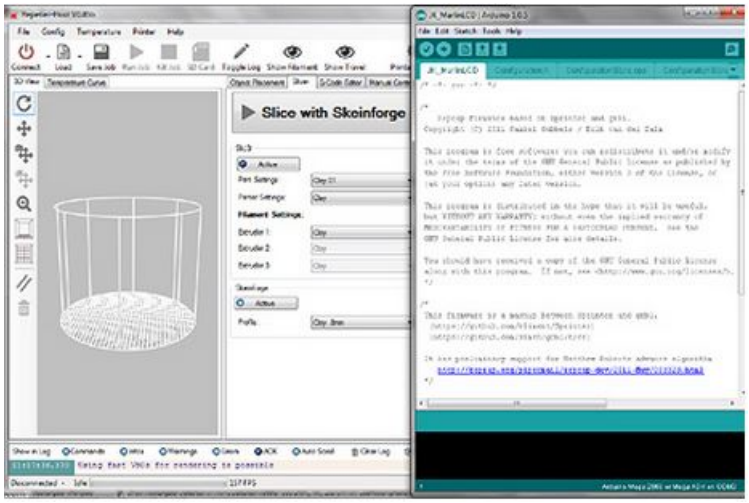

Figure 2.2: Marlin firmware interface.

printing, such as Rostock, Kossel, Cerberus, cOssel, Cherry Pi, Delta-Pi etc. It was found that all most all of these models uses a firmware platform known as 'Marlin' shown in figure 2.2. The firmware was used for configuration and provide a G code for the movement of the end effector of the robot printer. The design can be drawn in the workspace given in the software and the code can be executed. Even though this was an impressive advancement, in this study the control part was decided to be developed from scratch.

When it comes to the controlling part of the robot in order to obtain precise end effector positioning, the kinematics analysis was vital. Several literature sources were found to address the analysis of similar robot designs. A study done by Zhongfei Wang, Guan Wang, Shiming Ji, Yuehua Wan and Qiaoling Yuan stated means to determine a set of optimal design parameters of a Parallel robot whose workspace is possibly equal to a prescribed cuboid dexterous. The concepts were found to be relevant but not much applicable for the case in this study.

Going through the article by Steve Graves named as *Delta Robot Kinematics* provided a much more simpler and an effective solution for the application of inverse kinematics for a robot of similar fashion. Figure 2.3 shows a diagram used in this paper for the nomenclature of parameters. Although it has a number of references to be applied along with the Marlin firmware, the analysis was found to be very useful in gaining an idea on the application of proper inverse kinematics to the design. The paper also provided a break down of the forward kinematics analysis for the system but was found to be an approximate solution, mostly since the application of forward kinematics for parallel robots are theoretically very challenging. But the information on the inverse kinematics analysis was very useful in the development of the kinematics analysis of the prototype in this study.

It was planned to use DC motors for the actuation of the robot prototype in this study. Therefore, encoders had to be fixed so that the control programme could keep track of the number of rotations performed by the motors in turn calculating the position of the runners acting as the feedback of the system. A Electro-Mechanical Machine Design Lab instruction sheet on Encoders compiled by H.H.Manh was found to be of valuable information on the structure and operation of encoders. According to the information provided by the paper there are main two types of encoders namely incremental and absolute encoders. The incremental encoders provide the number of rotations relative to previously obtained reference position where the absolute encoders can provide exact positions

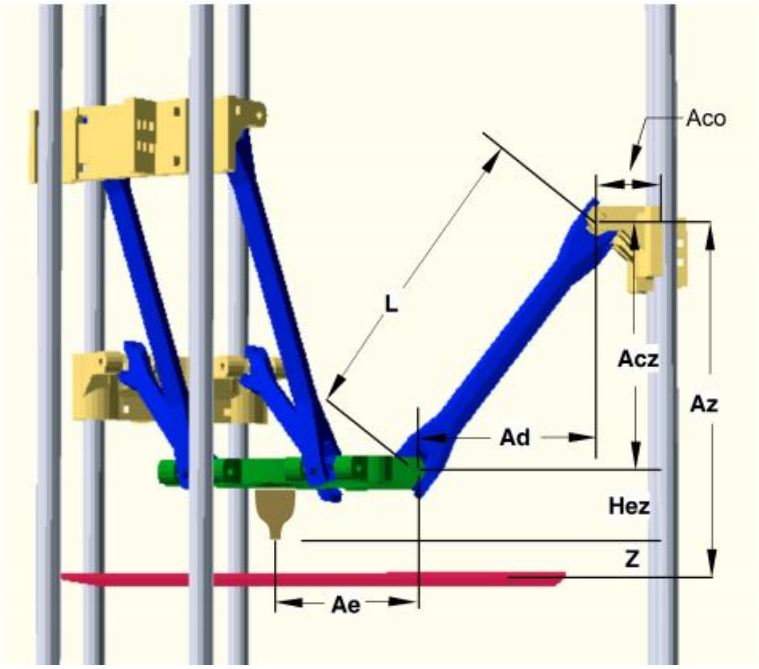

Figure 2.3: Marlin firmware interface.

caliberated along a given distance. Basically the incremental encoders would loose the stored value if the power goes off but not the absolute encoders.

The encoders could also be categorized according to the technology used to detect the rotation and send pulses. These were mainly two types known as magnetic and optical encoders. For this study incremental magnetic encoders were purchased. Also in the control programme, the most efficient way of precisely measuring the amount of rotation was by using interrupts in the code. This was also being elaborated in the lab sheet and was found to be very useful when developing the code. Figure 2.4 shows the pattern of detecting a clockwise rotation by the encoder pulses.

Developing the control programme with using the feedback from the encoders as to compensate the error in position control was significant in the precision control of the robot. For this the best controller was chosen to be the PID (Proportional, Integral and Derivative) controller. So knowledge regarding this was first obtained by studying the subject material on classic control in the Automatic Control module conducted by Dr.Manukid Parnichkun. The study presented the basic theory of PID control along with the approaches taken in computing the output parameter compensating the error in the system.

The PID controller works by providing three gain coefficients, K_p, K_i and K_d that respond to different parts of a system. To understand how they are used, the error $e(t)$ is defined as the difference between the desired value and the actual value. The output $u(t)$ from the PID loop is:

$$u(t) = K_p e(t) + K_i \int_0^t e(t)\delta t + K_d \frac{d}{dt} e(t) \qquad \text{(Equation 2.1)}$$

When working with discrete parameters the same equation can be presented as follows;

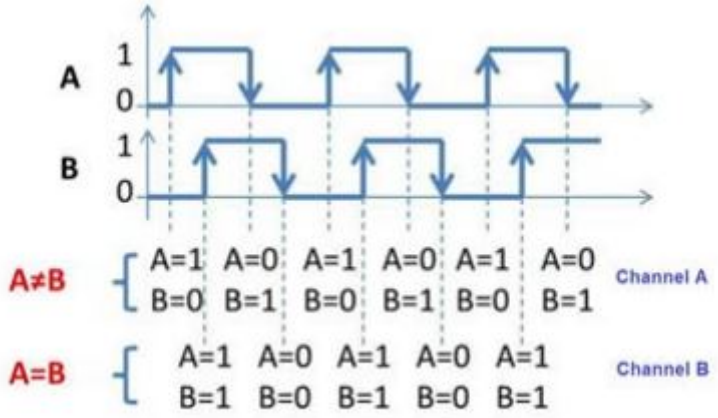

Figure 2.4: Clockwise rotation of an encoder.

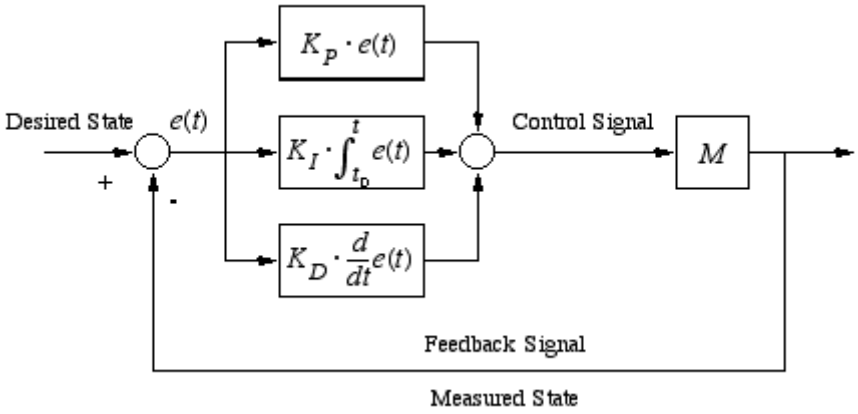

Figure 2.5: PID block diagram.

$$u(k) = K_p e(k) + K_i \sum_{0}^{k} e(k)\Delta t + K_d \frac{\Delta e(k)}{\Delta t} \qquad \text{(Equation 2.2)}$$

When applied to a system the PID works as shown in the block diagram in figure 2.5. This principle is executed in controlling the velocity of the motors in order to gain precise positioning of the runners along each column.

Since Arduino was used as the control platform, the PID library developed by Brett Beauregard was found to be useful. Even though the library consisted a few drawbacks it was managed to be implemented according to the requirements of the control programme. The functions PID() and Compute() was executed from the library in order to output the PWM value for the motor to run to the given position. Also in the separate project blog the author of the library presents means of improving the compute function to overcome the drawbacks. This was also studied to make the final

application effective.

2.2 Summery

The information presented in the Literature review starts with a comparison between conventional serial manipulators versus parallel manipulators. The competitive advantages of parallel robots were noted with regard to precision and speed. Several design of existing parallel robots were noted and technologies used in them. Developing the prototype for this study references were made to studies done in kinematic analysis of similar robot designs. The operation principles of encoders were studied in the actuation of the robot and finally for the control programme, the proper execution of the PID controller was studied from the resources.

Chapter 3

Mechanical Assembly

3.1 System Overview

The prototype of the parallel robot would contain three degrees of freedom. The three links would be designed to have a linear motion along three rails, located in a triangular configuration. The end-effector will be fixed to the other ends of the three links, fixed with hollow ball joints to provide free motion. The defined positions of the three links along the rails at a given moment will define the position of the end-effector.

3.2 Parts Catalog

The main parts of the robot were carved out of wood including the top assembly, the base and the runners. For the vertical shafts, stainless steel bars were used. The links between the runners and the end-effector were made using aluminium bars with hollow ball joints. The main design features are shown in figure 3.1 and the details are presented below.

3.3 Top Assembly

The top assembly comprises of two parts. This was done in order to provide mount space for the motors. The first part is shown in figure 3.2. And the motor mount is shown in figure 3.3. These two part together assemble the top portion of the robot.

3.4 Base

The base of the robot is carved with heavy wood in order to provide the stability for the structure. The dimensions and the design is shown in figure 3.4.

3.5 Vertical Shafts

Stainless steel bars of 12mm in diameter were used as the vertical shafts that facilitated the runners to move up and down. Stainless steel was used since the smoothness supports effective movements in the runners.

3.6 Runners

The bearings were lathed out of Nylon bars. Nylon was used because of the easiness to machine. A rectangular block was made out of wood to connect the two bearings along each couple of shafts.

3.7 Links to the end effector with ball and socket bearings

The parallel links were made out of Aluminium bars because of their light weight and the two ends were fixed with sockets lathed out of Nylon. These are shown in figure 3.5. These facilitate motion in all directions making them universal joints.

3.8 Final mechanical Assembly

The parts were all assembled together to a final mechanical assembly. The runners generated a small amount of friction on the vertical bars so they were further smoothed on the inner area. The rest of the friction generated was expected to overcome by the torque power of the motors. The final assembly of the mechanical structure is show in figure 3.6.

The runners were held in place with the bushes using zip ties. This way was convenient and was able to provide a tight grip. The universal joints provided good movement after assembly as expected. A small amount of tension was expected when setting up the columns in line with the holes drilled on to the base and the top assembly, but the final assembly was steady and according to the model design.

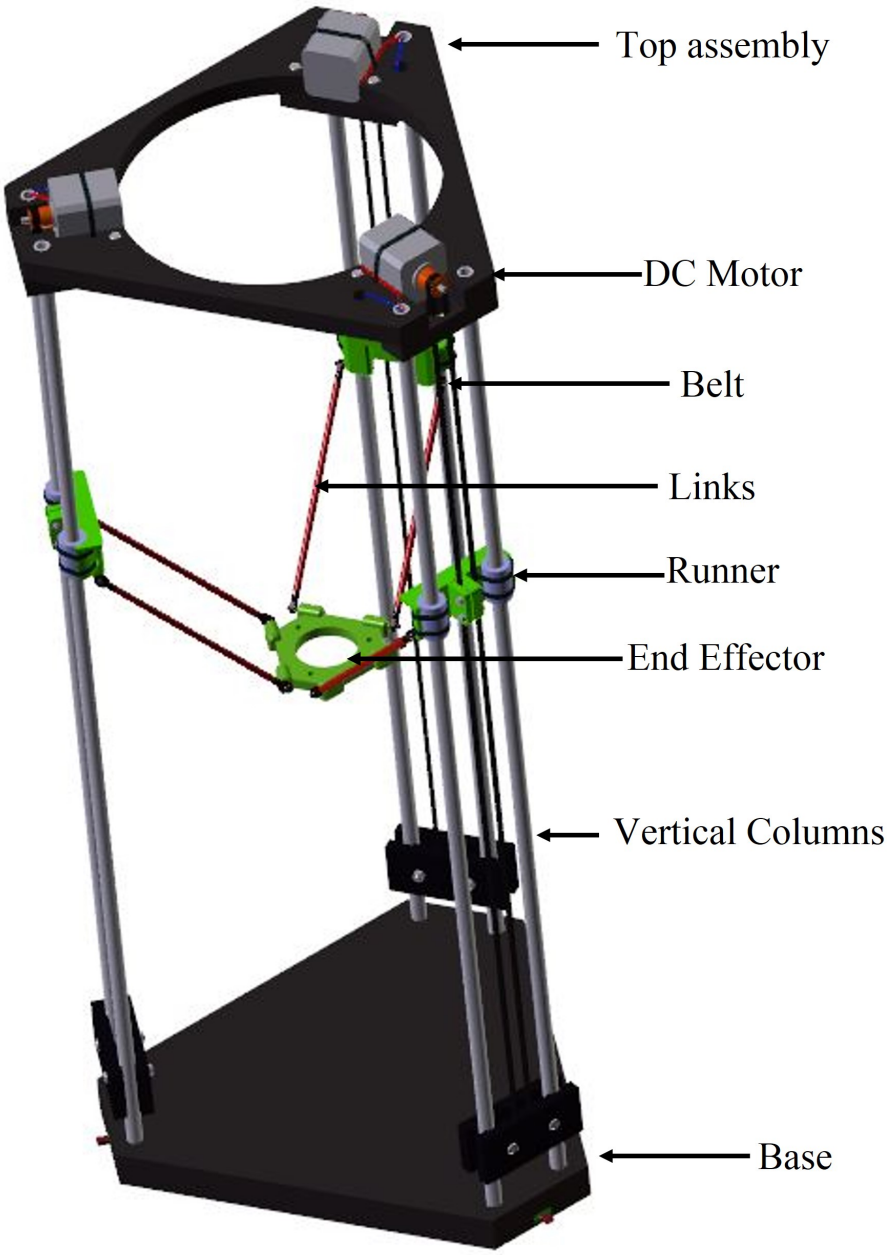

Figure 3.1: Main design features.

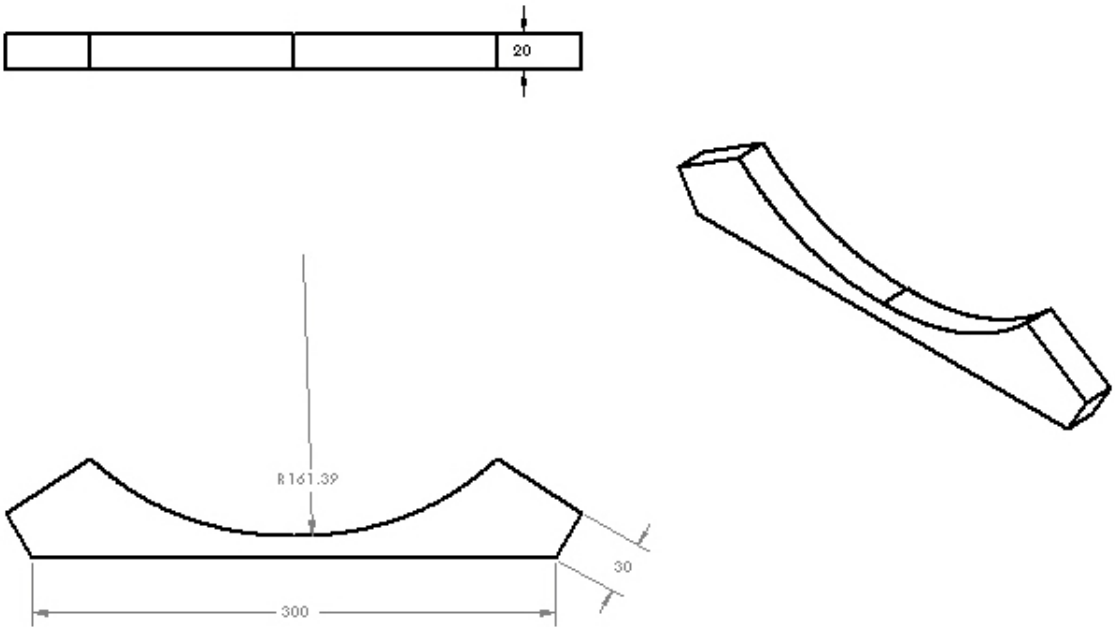

Figure 3.2: The first part of the top assembly.

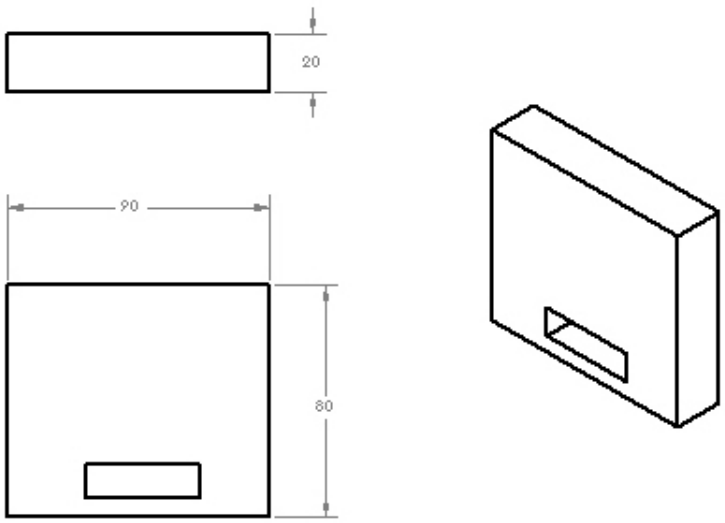

Figure 3.3: The motor mount of the top assembly.

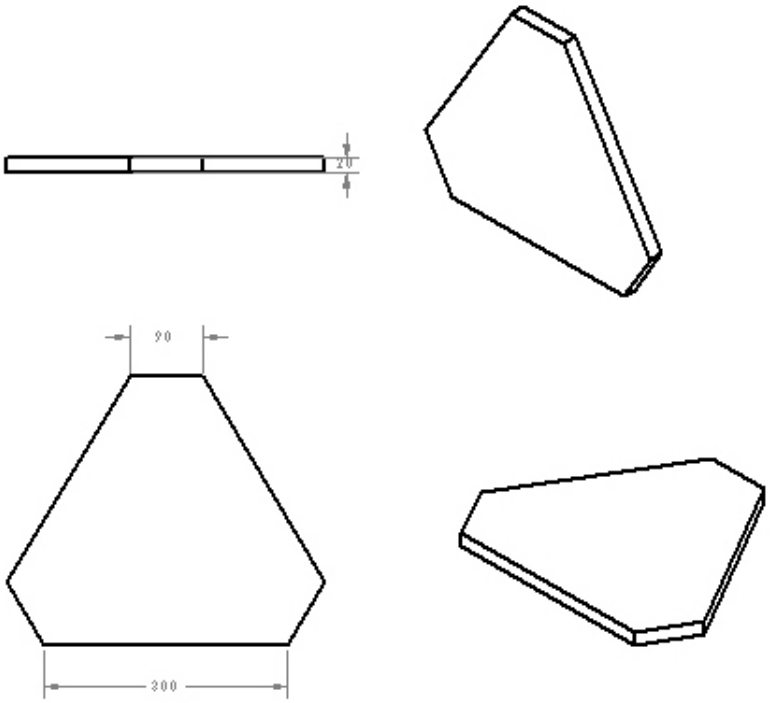

Figure 3.4: The base part of the structure.

Figure 3.5: The six links and the end effector.

Figure 3.6: The final mechanical assembly.

Chapter 4

Actuation and System Control

4.1 Motors

The motors used for the robot were three DC motors of 20 revolutions per minute(rpm), with a 12V power input. Motors were selected considering the estimated torque needed to overcome the friction generated in the runners. The motors were mounted using aluminium holders in order to provide stiffness of the structure when actuating. A mounted motor are shown in figure 4.1.

4.2 Encoders

Three rotary encoders were used to determine the position of the motors. These were incremental encoders with a resolution of 300 pulses per revolution, requiring a reference position to calculate the current position. Figure 4.2 shows the encoders mounted using the same aluminium mounts as the motors. The encoders were mounted on the base concerning the effectiveness of the design. The motors and the encoders were connected using belts that were connected to the runners.

4.3 Motor Drivers

The L298N H-Bridge motor driver module was used to drive the motors of the robot. Two modules were needed since one module facilitates only two motor inputs. This is shown in figure 4.3.

4.4 Micro-controller

The Micro-controller used in the robot was the Arduino Mega 2560 module consisting the ATmega2560 controller. This facilitating six interrupt pins made it ideal for the robot since all the six encoder inputs could be given interrupts to check the position precisely with minimum errors.

But when developing the robot for applications this the Arduino Mega controller that was initially planned to use gave hardware limitations. Because of this reason an Arduino Due module was used later, shown in figure 4.4, which was more capable and faster. Replacing the Mega with the Due module didn't cause much problem since the dimensions of the two boards were almost the same and both of them provided the same number of pins. The significant advantage with the Due was that any of it's pins could be defined as an interrupt. Also it was much more capable of handling several operations such as running the interrupts and executing the servo library simultaneously. This made it perfect for the intended robot operations.

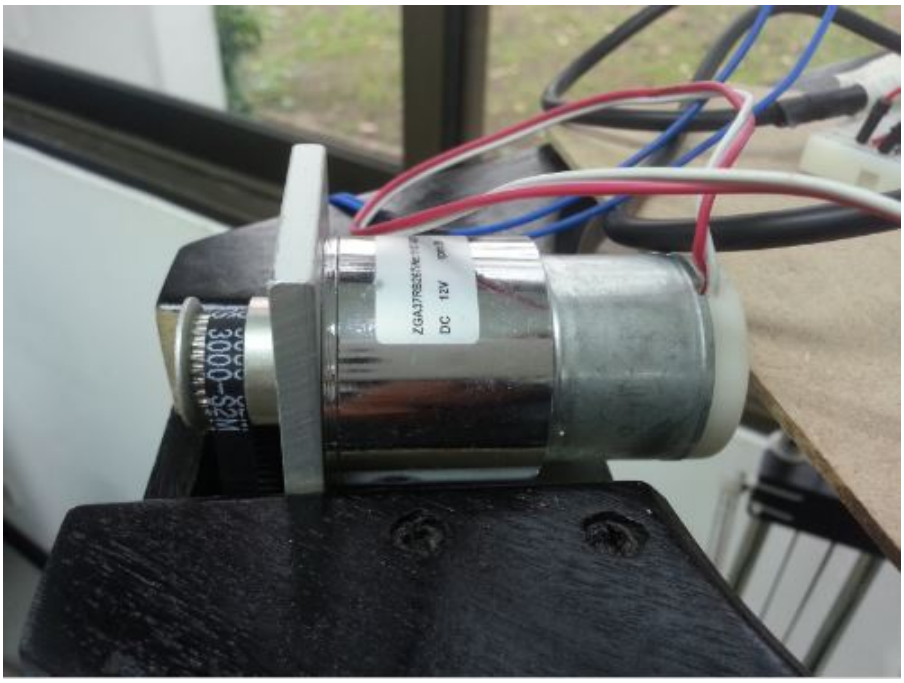

Figure 4.1: Motor assembly.

4.5 Limit Switches

Limit switches were fixed in order to provide a reference for the encoders. The programme had to be written so that at the beginning of every execution cycle the limit switches had to be pressed and the current position made zero.

A limit switch is made up of three pins as shown in figure 4.5. Here the normally closed (NC) pin and the com pin was used with the com pin connected to ground and the NC pin connected to the board input. This method was the safest since there's no input of current into the board and at the same time the contact could be detected.

Figure 4.2: Encoder assembly.

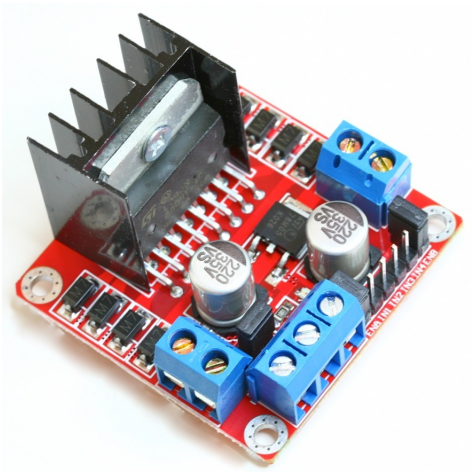

Figure 4.3: Motor Driver Module.

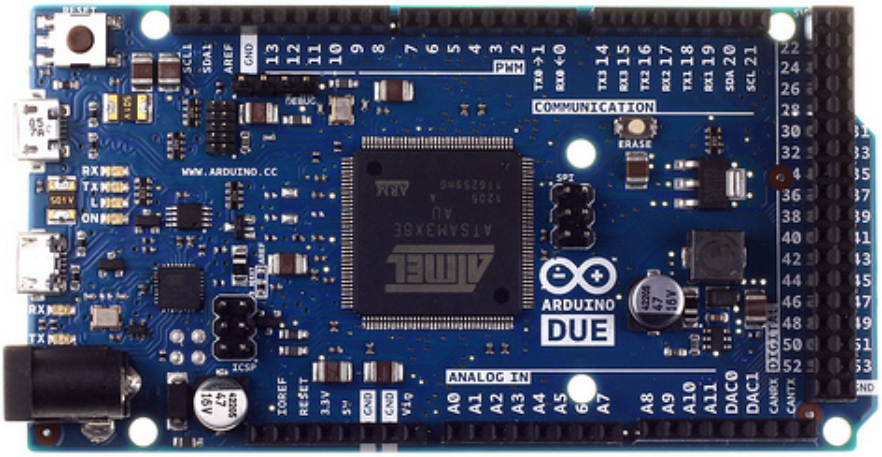

Figure 4.4: The micro-controller module.

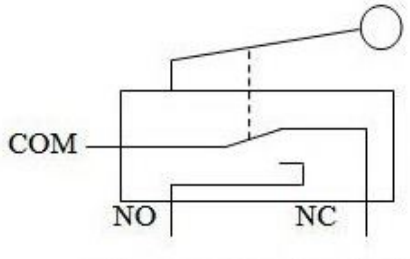

Figure 4.5: Limit Switch diagram.

Chapter 5

Kinematics Analysis

When building a control algorithm to move the end effector to the desired position in the workspace, the kinematics analysis is vital in order to define the points in the workspace. This was done using simple geometric relationships and several assumptions were made in order to simplify the problem as much as possible while retaining the validity of the results.

5.1 Naming Conventions and Assumptions

Figure 5.1 presents a three-dimensional model of a similar linear delta robot[1] and it was used as a reference.

The three columns were named as A, B and C and they were considered to be sitting on the vertices of an equilateral triangle so that the distances between the columns were considered to be equal with a 120deg angle between each other from the origin. (Origin being the mid point of the triangular base.) Each runner is connected to two parallel links which are considered to be equal in size and perfectly parallel.

The bottom surface was called as the bed and the plane on which the end effector lies was called the end effector plane. It is very important to consider that the end effector always lies on the plane flat without any change in its horizontal orientation. In other words, the end effector is considered to be always parallel to the bed. This simplifies the kinematics. It was seen that even in the practical scenario the links were able to maintain the end effector as such.

It was seen that even though the workspace consists of three dimensions, X, Y, and Z this can be simplified in the control of the robot since the Z axis merely can be controlled by moving the three runner by equal lengths along the columns. Therefore what should be considered to determine a point can be deduced to the position in the XY plane. Hence, the bed of the robot was named as the XY plane with the origin at the middle and the positive Y direction going through the A column. This nomenclature is presented in figure 5.2.

Another prominent assumption made in this kinematics analysis is that the pairs of columns and the pairs of links were considered to be single rods. These single rods were assumed to pass trough the mid points of each pair and is clearly presented in figure 5.3.[2] Also the diameter of the end effector is neglected so the links were considered to meet at the effector end.

Based on the above mentioned assumptions, inverse kinematics was applied to the robot.

---

[1]This was obtained from Jonathan Keep's blog on his model of the delta robot used for 3D printing.

[2]This figure was obtained from Steve Graves's paper named "Delta Robot Kinematics".

Figure 5.1: Model Design.

5.2 Inverse Kinematics

The range of movements of the links were observed and it was attempted to apply several geometric and trigonometric relations to the movement paths. Note that equations were derived only for one side since that same procedure can be used for the other two links in a similar manner.

Lets consider the column A and name the parameters in the XZ plane as shown in figure 5.4.

The elevation of the end effector from the bed is the 'Z' parameter of the system. The length of the link is named as 'L', the perpendicular distance from the column to the end effector as 'Ar' and the height from the end effector plane to the point where the link is connected to the column as 'Az'. It can be seen that a right angled triangle is formed with the sides 'L', 'Ar' and 'Az'.

From the motion of the universal joints we can see that a single link allows the end effector to move in a circular arc with the centre at the column. So considering the general equation of a circle we can right an equation as;

$$(X - Acx)^2 + (Y - Acy)^2 = Ar^2 \qquad \text{(Equation 5.1)}$$

Here 'Acx' and 'Acy' represent the X and Y coordinates of the point at which the end effector plane cuts the column, respectively. Looking at the right angled triangle we can say that;

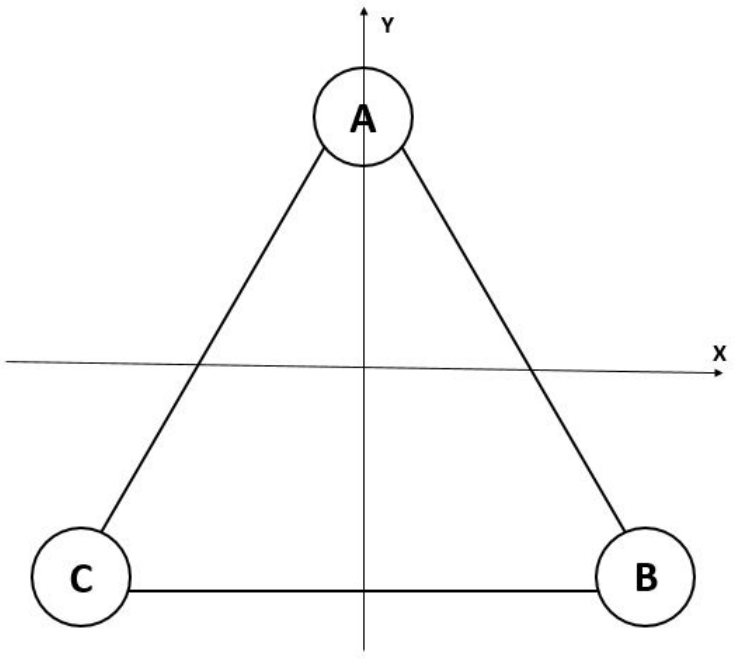

Figure 5.2: Axis Declaration.

$$Ar^2 = L^2 - Az^2 \quad \text{(Equation 5.2)}$$

Substituting the above relation in to the first equation yields;

$$(X - Acx)^2 + (Y - Acy)^2 = L^2 - Az^2 \quad \text{(Equation 5.3)}$$

$$Az^2 = L^2 - (X - Acx)^2 - (Y - Acy)^2 \quad \text{(Equation 5.4)}$$

Now if we measure the full height of the column from the top platform to the bed as 'AH', which are the maximum and the minimum points that the runner can move, and the height from the point where the link is attached to the column to the top platform as 'AE', we can say that;

$$Az = AH - AE - Z \quad \text{(Equation 5.5)}$$

Then;

$$(AH - AE - Z)^2 = L^2 - (X - Acx)^2 - (Y - Acy)^2 \quad \text{(Equation 5.6)}$$

$$AH - AE - Z = \sqrt{L^2 - (X - Acx)^2 - (Y - Acy)^2} \quad \text{(Equation 5.7)}$$

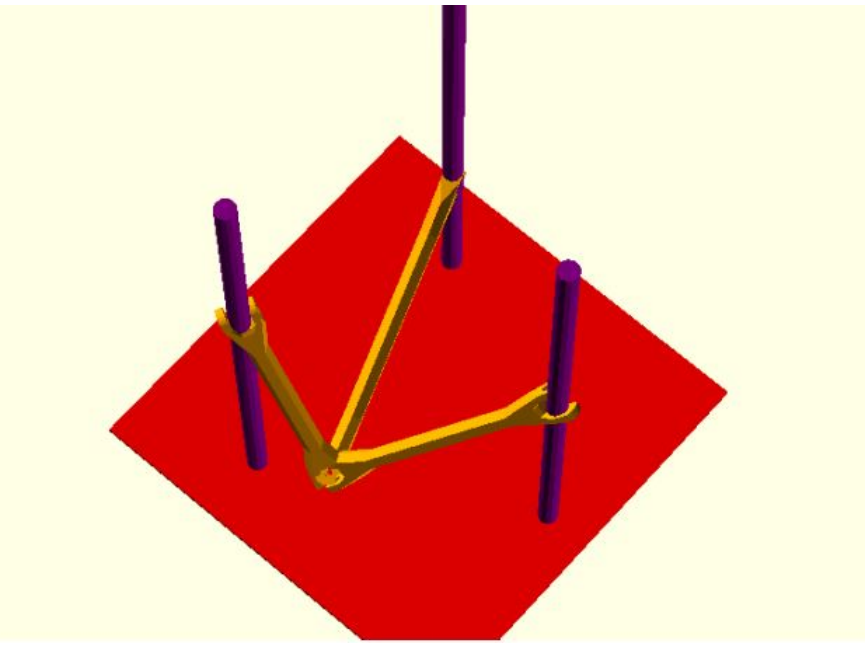

Figure 5.3: Assumed columns and links.

Now we have the result;

$$AE = AH - \sqrt{L^2 - (X - Acx)^2 - (Y - Acy)^2} - Z \qquad \text{(Equation 5.8)}$$

The above result is very important since it yields the height that runner should move downwards from the top platform in order to come to a given point in the XYZ space. Now the above result can be similarly applied to the other to columns as well. The similar symbols will represent the same parameter in the respective columns.

$$BE = BH - \sqrt{L^2 - (X - Bcx)^2 - (Y - Bcy)^2} - Z \qquad \text{(Equation 5.9)}$$

$$CE = CH - \sqrt{L^2 - (X - Ccx)^2 - (Y - Ccy)^2} - Z \qquad \text{(Equation 5.10)}$$

The control programme of the robot will make the runner move upwards at the setup making the encoder values zero at the beginning. This will be explained in detail in the following chapter. Hence the encoder starts measuring the pulses from zero relative to the top platform. This is where the above result come into play.

When relationships between 'AE', 'BE' and 'CE' lengths and particular encoder readings are built, an algorithm could be developed to control the precise positioning of the end effector in the workspace.

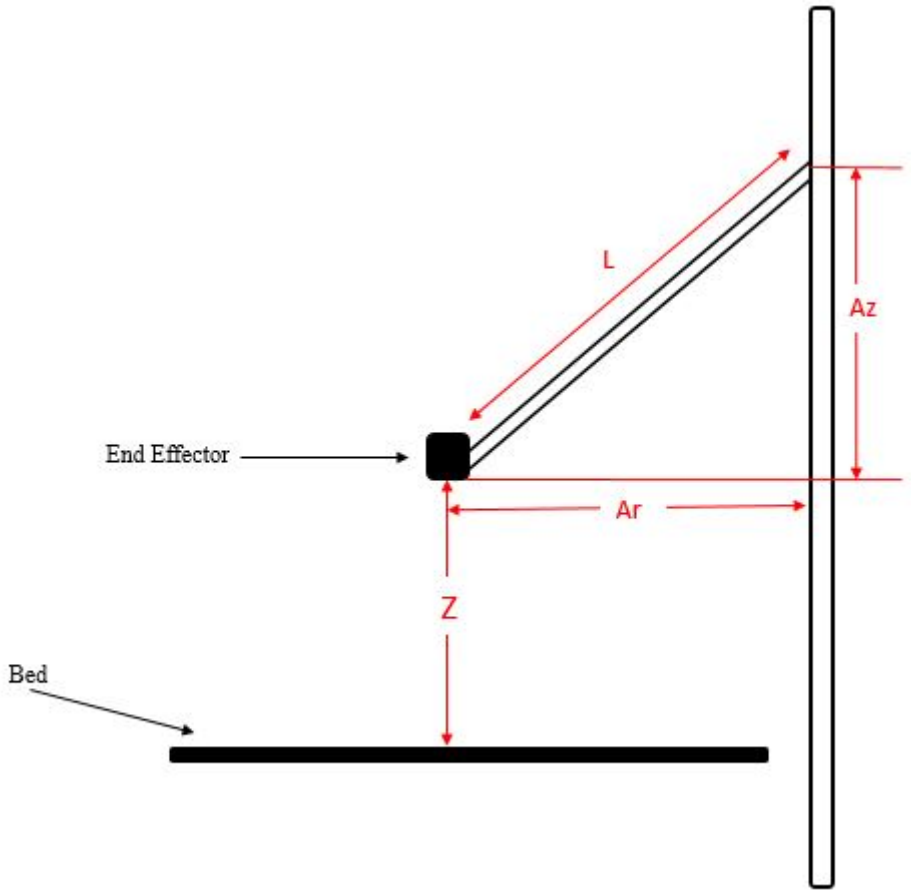

Figure 5.4: Column A parameters.

5.3 Further Calculations

The kinematics analysis was developed further more to the extent of considering the pitch and yaw movements of the end effector platform. This analysis goes beyond the assumption of the end effector being perfectly horizontal so that the angles at the joints were taken into consideration. These motions will further enhance the versatility of the robot manipulator.

The pitch and yaw motions were intended to be attained using a mechanical assembly of two links on the existing end effector as shown in figure 5.5 and 5.6. The links are fixed using two rotary joints and can be actuated using two motors. The two variable angles were named as $\theta 3$ and $\theta 4$. By altering $\theta 3$ the yaw motion can be attained while $\theta 4$ facilitates the pitch of the end effector. These two links could be considered as a separate system for the kinematics analysis. The kinematics model was developed using MATLAB for the end effector assembly as shown in figure 5.7. The DenavitHartenberg (DH) parameters used to simulate the model are given below. Here;

a_i = distance between axes z_i and $z_i + 1$, and measured along the axis x_i.
α_i = angle between axes z_i and $z_i + 1$, and measured in a plane normal to x_i.
d_i : distance from origin to the intersection of the axis $x_i + 1$ with z_i, and measured along the axis z_i.
θ_i : angle from x_i to $x_i + 1$, and measured in a plane normal to z_i.

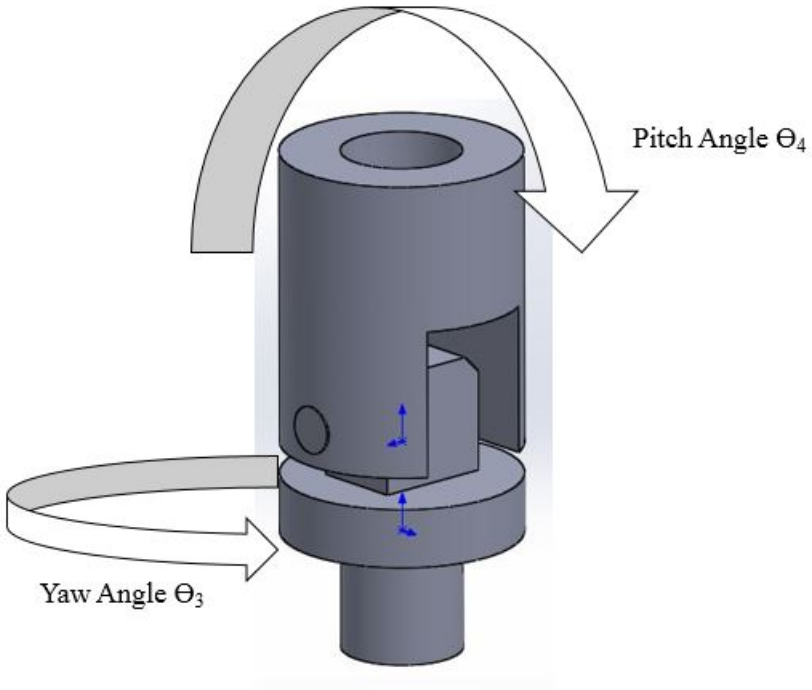

Figure 5.5: End effector assembly.

Looking at the model, two equations were composed applying inverse kinematics using the geometrical relationships to obtain the values for the angles $\theta 3$ and $\theta 4$ as follows.

$$\theta_3 = atan2(X, Y) \quad \text{(Equation 5.11)}$$

$$D = \frac{X^2 + Y^2 + Z^2 - a_1^2}{2a_1}$$

$$\theta_4 = atan2(D, +/- \sqrt{1 - D^2}) \quad \text{(Equation 5.12)}$$

Using the above results the robot can be given a coordinate for in the XYZ plane so that the pitch and yaw angles can be calculated. These angles together with the column parameters calculated in the above sections will provide the robot 5 DOF. But for the applications presented in the next chapters, only 3 DOF are used according to the developed prototype. Improving the mechanism for 5 DOF is presented in future developments.

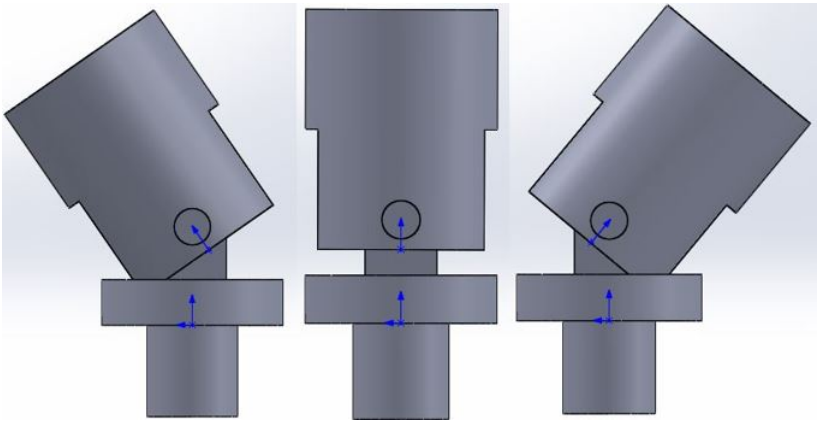

Figure 5.6: End effector movements.

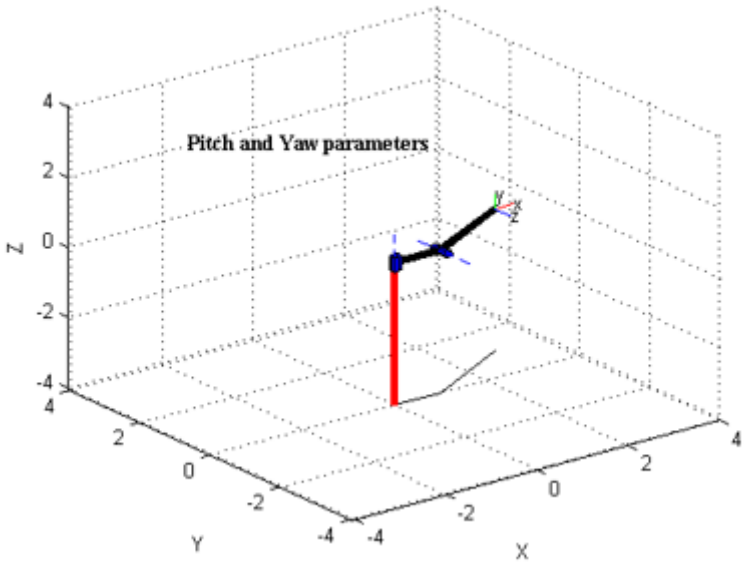

Figure 5.7: Kinematics model of the end effector.

Link	a	α	d	ϴ
3	a_1	90°	0	Θ_3*
4	a_2	0	0	Θ_4*

Chapter 6

Control Algorithm and Programme

6.1 Experimental Data

In order to build an algorithm on the derived inverse kinematics equations, respective measurements of the physical attributes were obtained. (All units in cm)

AH = 52 Acx = 0 Acy = 15
BH = 52 Bcx = 13.5 Bcy = -8
CH = 51 Ccx = -13 Ccy = -8.6
L = 23

In order to obtained the encoder values with respect to the length parameter obtained from the equations, length travelled by the runners for a given encoder value was acquired.

For 1000 pulses, all three columns travelled 16.3 units. Therefore to move one unit the encoder ratio is given by,

$$EncoderRatio(ER) = (10000/16.3) \qquad \text{(Equation 6.1)}$$

6.2 Development of the Algorithm

At this point the main purpose of the algorithm was to be able to feed in the accurate encoder parameters derived from the inverse kinematics equations. The steps could be presented as follows;

- Obtain the user provided X, Y and Z coordinates
- Input the coordinates into the inverse kinematics equations and obtain AE, BE and CE
- Multiply AE, BE and CE values with ER to obtain the encoder parameters
- Return the encoder parameter to be fed into the PID controller as the set point.

These were written as separate return functions for each of the columns in the Arduino code as shown in figure 6.1.

6.3 Development of the Programme

The programme to control the parallel robot was started with testing manipulation of a single column. The idea was to develop the programme for exact positioning of the runner along the column with accurate acquisition of encoder values, direction of the motor and PID parameters, so that the same procedure could be applied to the other two column and eventually make them work together. Limit switch was employed to make the encoder reading zero at the beginning of each cycle, interrupts were

```
double AlgoA(double X, double Y, double Z) {

  double AE; double L = 23;
  double Setpoint;
  double AH = 52;       double Acx = 0;       double Acy = 29;

  double ER = (10000 / 16.3);

  AE = AH - sqrt(pow(L, 2) + Acx + Acy - (X + Y)) - Z;

  Setpoint = ER * AE;
  Serial.println(Setpoint);

  return Setpoint;

}
```

Figure 6.1: Execution of the algorithm in Column A.

used to track the encoder values and the set point was given manually. The preliminary flow of the code is shown in figure 6.2.

After the initial code was a success, it was implemented to the other two motors and all the motors were programmed to work together. It was important that the motors had to wait until all three encoders were made zero at the top and then start moving together since when the end effector and the links were connected they could move into singularities unless the runners were always keeping the link above the level of the end effector plane.

The tuning of PID gains was given considerable time and concern since the stability of the entire system was dependant on correct positioning of the runners. First some random gain values were assigned to check the code before they were fine-tuned. At the initial stages there were errors in the positioning and the PID gains gave oscillations to the motors. These errors were noted and plotted in order to determine what gains should be altered. Figure 6.3 shows the set point versus the output fluctuation in the column A runner.

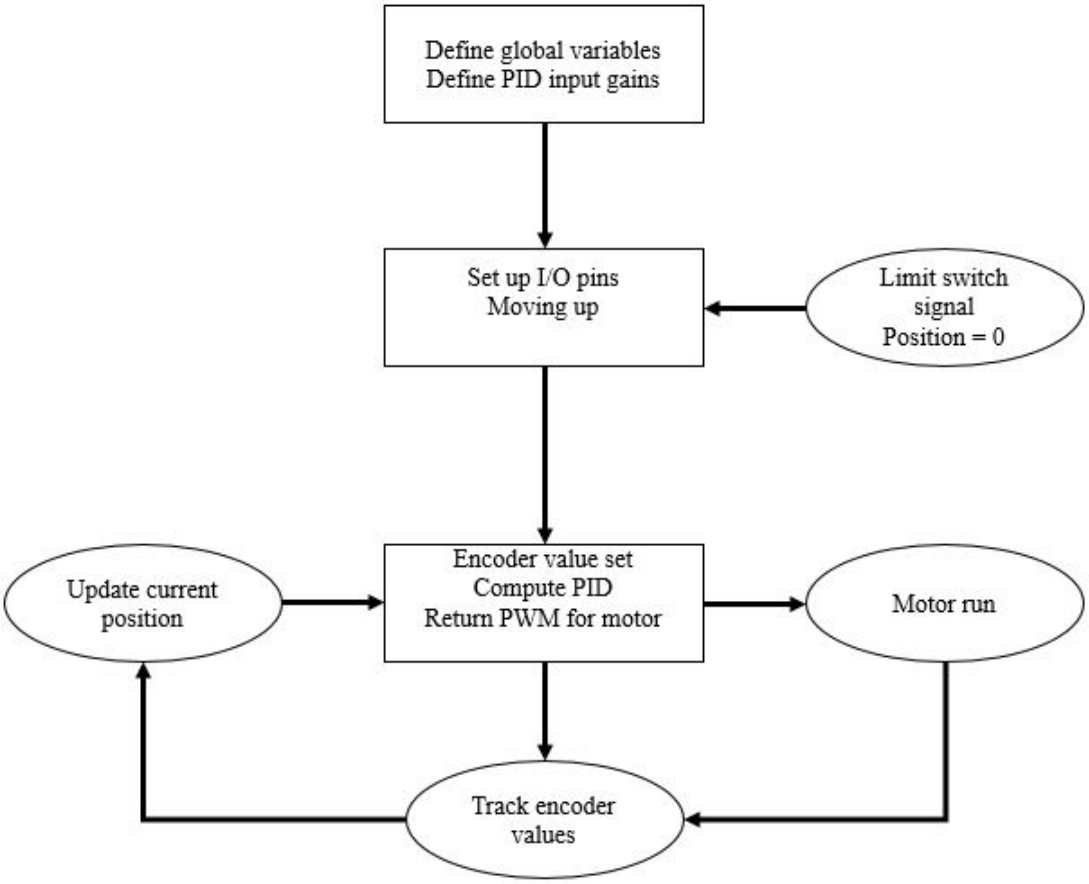

Figure 6.2: Preliminary flow of the code.

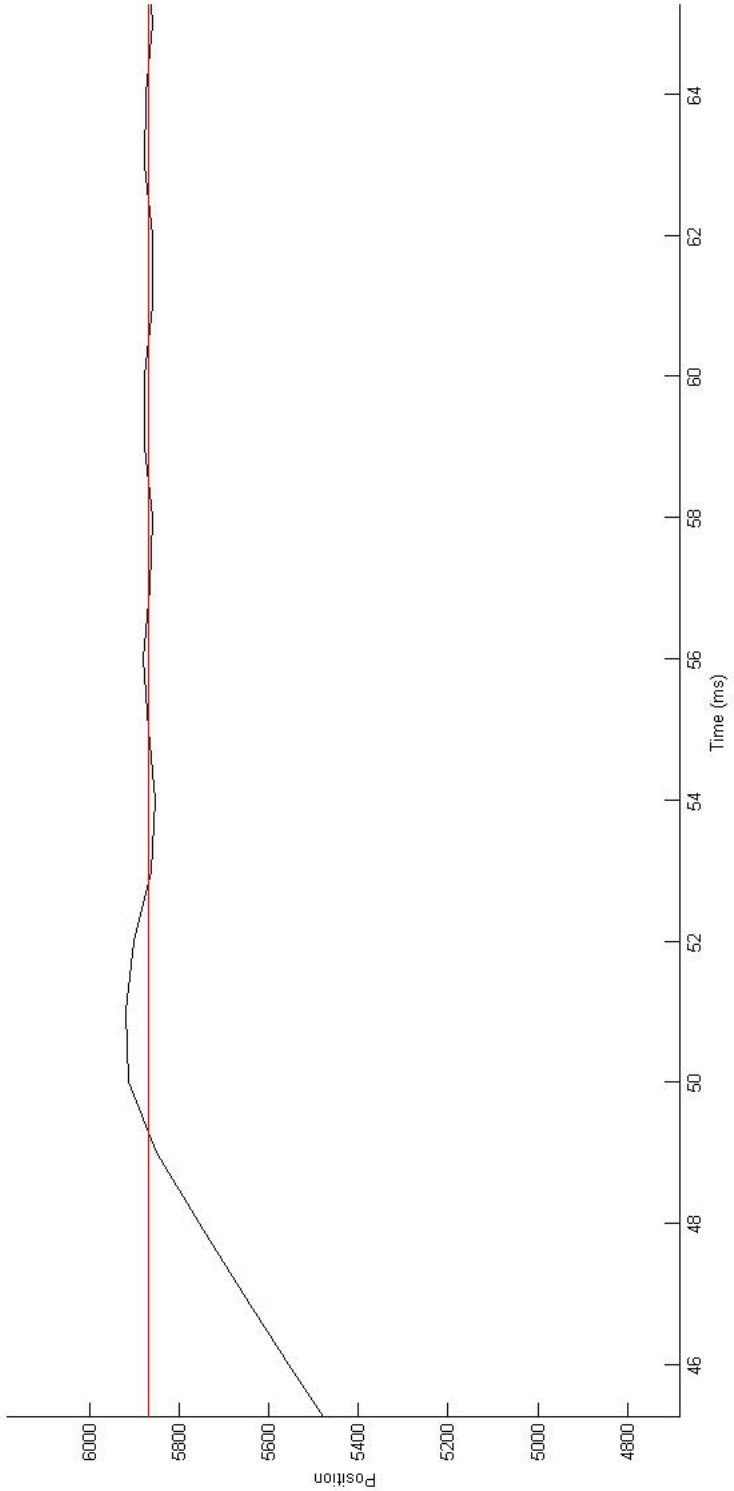

Figure 6.3: Set point versus the output fluctuation in the column A runner.

This condition was addressed by trial and error method. First the Ki and Kd values were made zero and Kp was adjusted until the system is balanced. Then Kd value was increased gradually until there was no oscillation in the motors and less error in positioning. Ki value was given a very small number to compensate the accumulation of error. In the final application code the gains were set as follows.

$$Kp = 0.5$$
$$Ki = 0.001$$
$$Kd = 0.3$$

In order to get an application out of the robot, it was needed to develop the programme in order to move to accurate positions in one cycle. This was the next step forward.

6.4 Programme Enhancement

For the end effector to move according to a given array of coordinates, the control programme is enhanced. The add-on features are presented in figure 6.4. This part is added to the previously developed code and the was found to work considerably fine.

The coordinate values were given in three arrays, namely, x, y and z. They were ordered and the a coordinate at one time is read by the appropriate array position of each value in each of the three arrays. But an issue arose when dealing with how to shift to the other position and continue the instructions since the encoder values didn't yield the exact number that was fed into. So a tolerance of 200 was provided, meaning that if the position of the runner stops between +/-200 from the input value, the instruction will be considered fulfilled and the next coordinate will be executed. This procedure was found to work considerably good.

When enhancing the code to fit a particular application like pick and place, the code had to be more modified since an extra component was added, which was the gripper consisting a servo motor, so that in between some coordinates the shift has to be stopped and the instruction to either grip or release had to be given. This was attained by using another array called the commandType, which was given values of 1, 2 and 3 where they meant continue to the other coordinate, grip and release respectively. So in each loop cycle this number is read at first to see if the instructions should increment to the next coordinate or should it send the signal to the gripper. This method worked successfully.

The most significant problem that came up here was that the Arduino Mega micro controller that was being used gave wrong computations when the servo motor instructions were added to the code. This was later found to be an error that occur with the hardware of the module since the timers were overwritten by the instructions. So then the micro controller module was changed to be an Arduino Due, that had a faster ARM processor which suited the application properly.

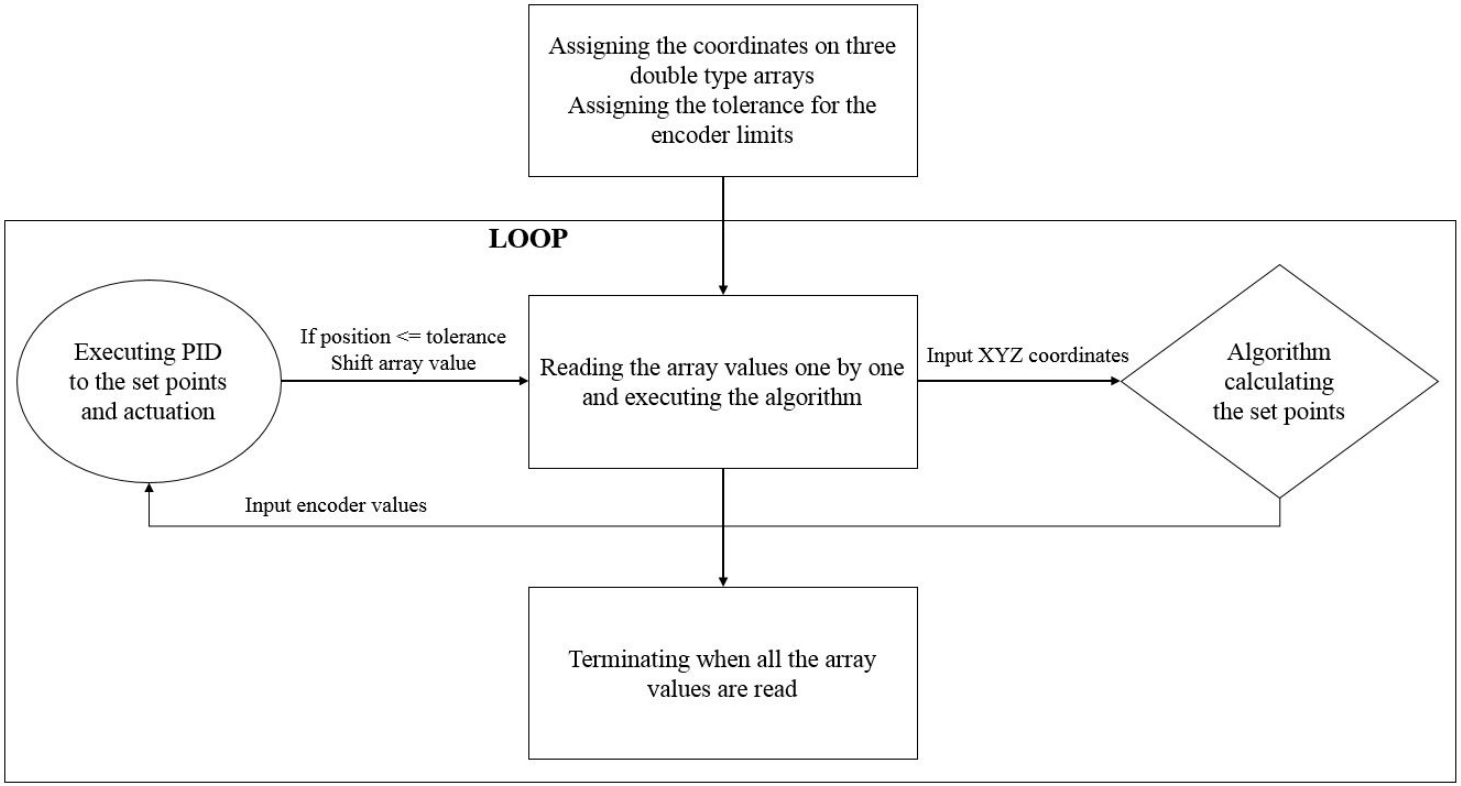

Figure 6.4: Programme enhancement flow to input the coordinates.

Chapter 7

Results and Applications

The final assembly of the entire system consisting all the components is shown in figure 7.1 and figure 7.2. The assembly was robust and all the electronic components worked as expected.

7.1 Test for accuracy

When a coordinate in the XYZ plane is defined for the end effector to reach, the corresponding encoder values were calculated by the algorithm and executed. The PID controller provides the PWM values for the motors to reach the calculated encoder values and the precision of the end effector positioning depends on how accurately the output encoder values match the calculated values. This was tested and the the errors were noted.

Given coordinate: (X,Y,Z) = (0,0,10)

Substituting the coordinate values along with the other parameters in equations 5.8, 5.9 and 5.10 the lengths were calculated and the set points were obtained by multiplying them with the encoder ratio given in equation 6.1. The Matlab code used for the calculation is given below.

```
AH = 52; Acx = 0; Acy = 15; L = 23; ER = (10000 / 16.3);
BH = 52; Bcx = 13.5; Bcy = -8;
CH = 51; Ccx = -13; Ccy = -8.6;

X = 0; Y = 0; Z = 10;

format long g

AE = AH - sqrt((L^2) - ((X - Acx)^ 2) - ((Y - Acy)^ 2)) - Z
Setpoint_A = ER * AE


BE = BH - sqrt((L^ 2) - ((X - Bcx)^ 2) - ((Y - Bcy)^ 2)) - Z
Setpoint_B = ER * BE

CE = CH - sqrt((L^ 2) - ((X - Ccx)^ 2) - ((Y - Ccy)^ 2)) - Z
Setpoint_C = ER * CE
```

The awswers were as follows;

AE = 24.56
Setpoint A = 15070

BE = 25.18
Setpoint B = 15451

CE = 24.08
Setpoint C = 14777

The coordinates were fed into the control programme and the set points reached by the robot were obtained.

Motor A position = 14430
Motor B position = 15350
Motor C position = 14146

$$ErrorPercentage = \frac{(Calculatedsetpoint - Reachedmotorposition)}{Calculatedsetpoint} * 100\% \quad \text{(Equation 7.1)}$$

Substituting the values in equation 7.1, the error percentages of each column were as follows;

Error in positioning of column A = 4.25%
Error in positioning of column B = 0.65%
Error in positioning of column C = 4.27%

We can see that the errors in column A and C are quite significant compared to the very low error in column B. It was concluded that this was because of the friction of the runners along column A and C if higher compared to that of column B. In fact, this was noticed in the first mechanical assembly and it was attempted to reduce the friction by smoothing the inner surfaces of the runners. But still those runner generated a small amount of friction during operation but the error percentages were concluded to be acceptable compared to the entire performance of the robot positioning.

Next step, the exact position of the end effector reached during operation was compared with the given coordinates. Here five independent coordinates were given for the robot to move and the positions were measure and the results were plotted. Figure 7.3 to 7.7 presents the plotted results and the errors in each axis were calculated as Ex, Ey and Ez.

Obtaining the error values from the calculations the mean square error can be found for the entire system with respect to each axis. The general equation for calculating mean square is given as;

$$S = \frac{1}{n}\sum_{i=1}^{n} E_i \quad \text{(Equation 7.2)}$$

Where Ex = error respected expected amount and n = number of test samples.

Substituting the error values for each axis in the equation 7.2, we can obtain the following mean square errors.

Sx = ((-0.7)^2 + $(-1.5)^2$ + $(0.5)^2$ + $(0.5)^2$ + $(-0.1)^2$)5 = 0.65

Sy = ((1.5)^2 + $(-0.5)^2$ + $(1)^2$ + $(-0.5)^2$ + $(-1)^2$)/5 = 0.95

Sz = ((0.5)^2 + $(0)^2$ + $(0)^2$ + $(0)^2$ + $(0)^2$)/5 = 0.05

It can be seen that the error in the Z plane, compared to X and Y planes, is very low. The overall effect of the above calculated error was addressed and dealt with when performing applications.

The maximum workspace volume that the end effector can move is shown in figure 7.8. This was simulated executing the forward kinematics analysis presented in Steve Graves's article on Delta robot kinematics. Also as explained earlier, since a link can move in an arc independent from the other links, the effective workspace in XY plane was found by visualizing the possible end effector paths of the three independent links. The area where the three paths overlaps can be define as the effective 2D workspace.

7.2 Applications

In order to demonstrate the capabilities of the robot, it was programmed to perform two distinct applications of drawing and pick and place. The results were recorded and the video files were provided with the presentation.

First the robot was checked for a drawing application. A paint brush was fixed on to the end effector and coordinates were given for movement with the correct elevation from the base. It was attempted to draw a square with the two diagonals inside and the result is shown in figure 7.10. The result was found to be considerably accurate.

Secondly the robot was set up for a pick and place application and the end effector was fitted with a gripper component with a servo motor as shown in figure 7.11. The specific coordinates were provided in the code itself and objects were placed on the base at those locations. Since the goal was to test the accuracy of positioning, the gripper module was not further developed. The programme was written in order for the robot to pick the objects at three different locations on the workspace and place them on top of each other at the center. The result was considerably satisfactory after several trials. The full control code can be found at the website *github.com /miranthajayatilake/ Parallel_robot/tree/master.*

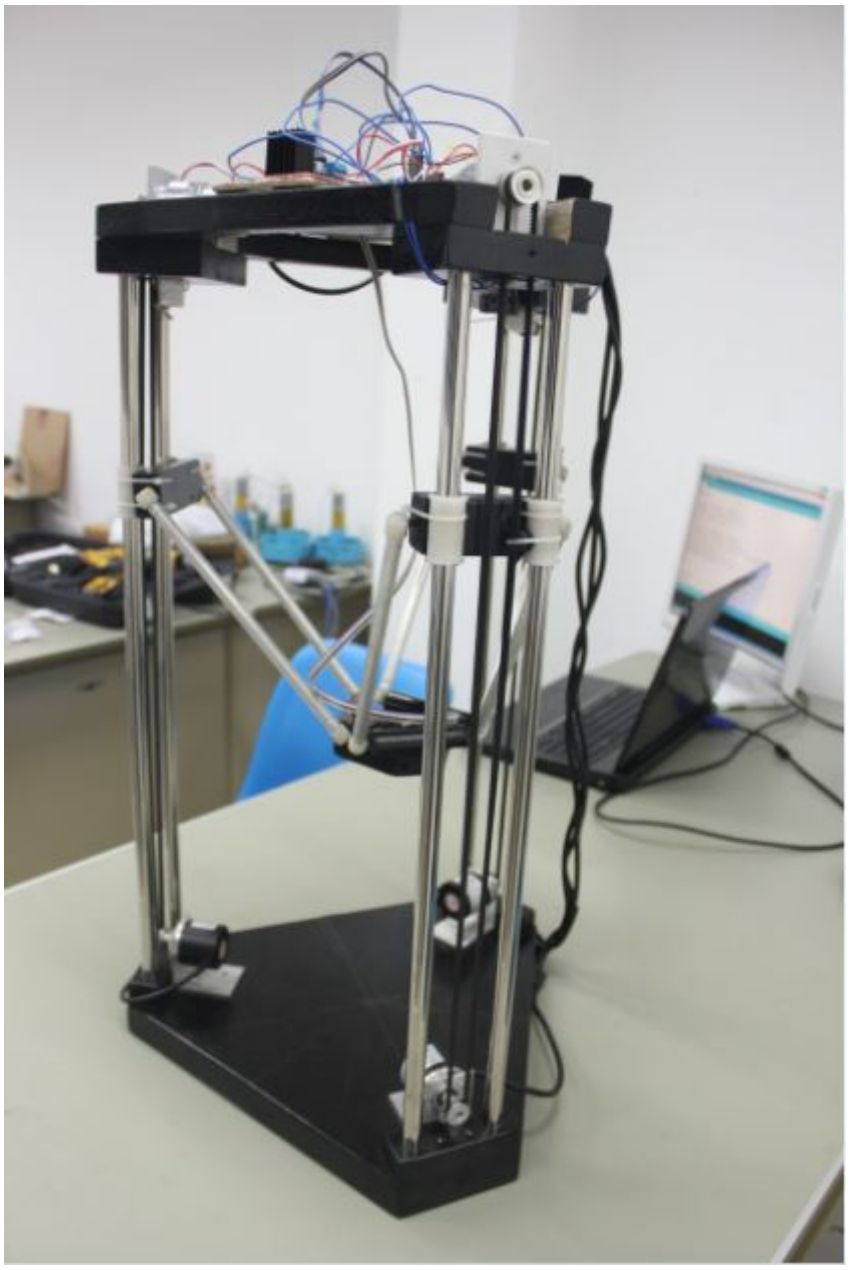

Figure 7.1: Final assembly of the robot.

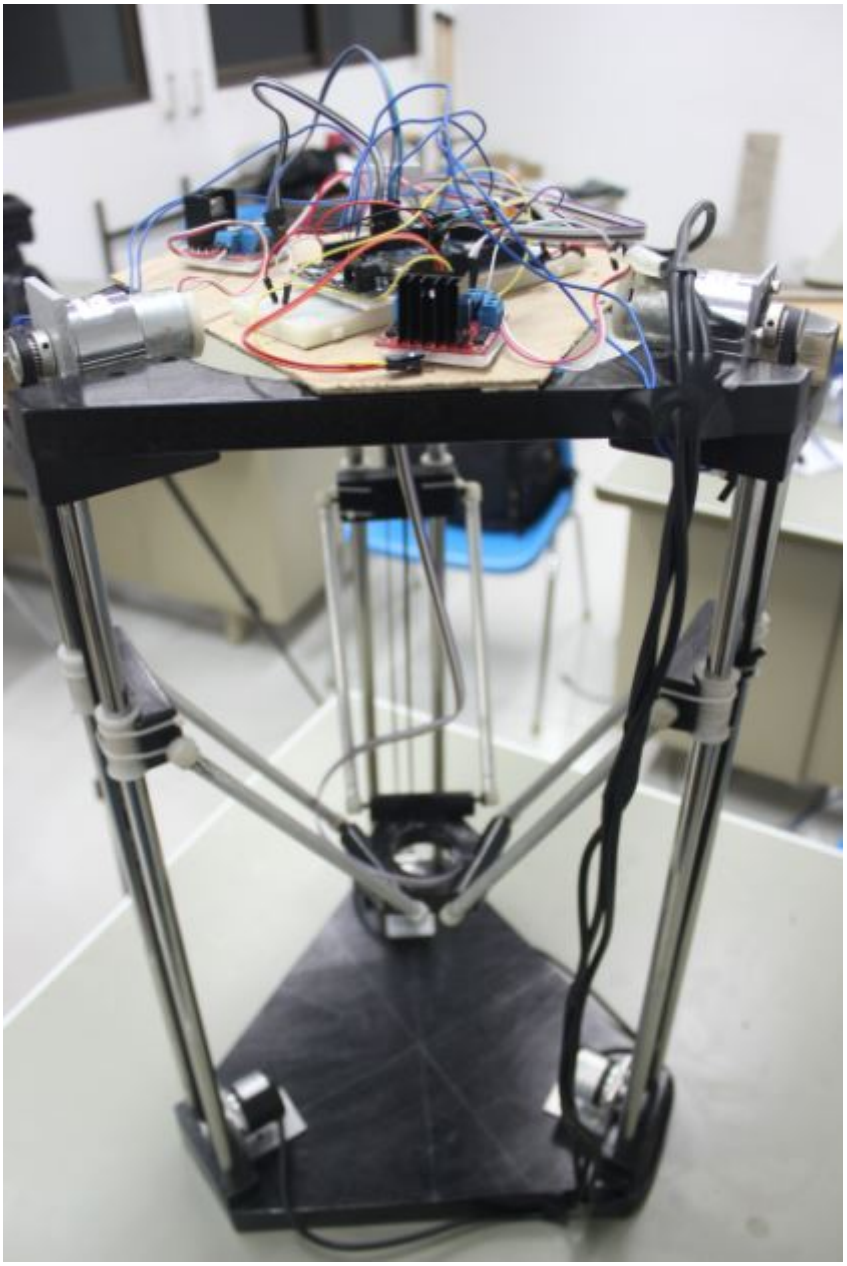

Figure 7.2: Final assembly of the robot.

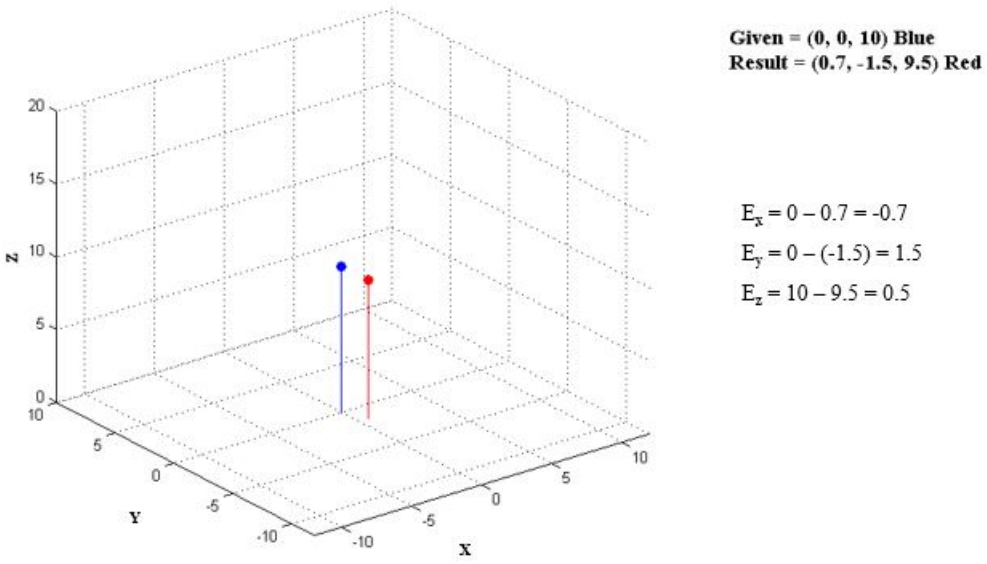

Figure 7.3: First test position.

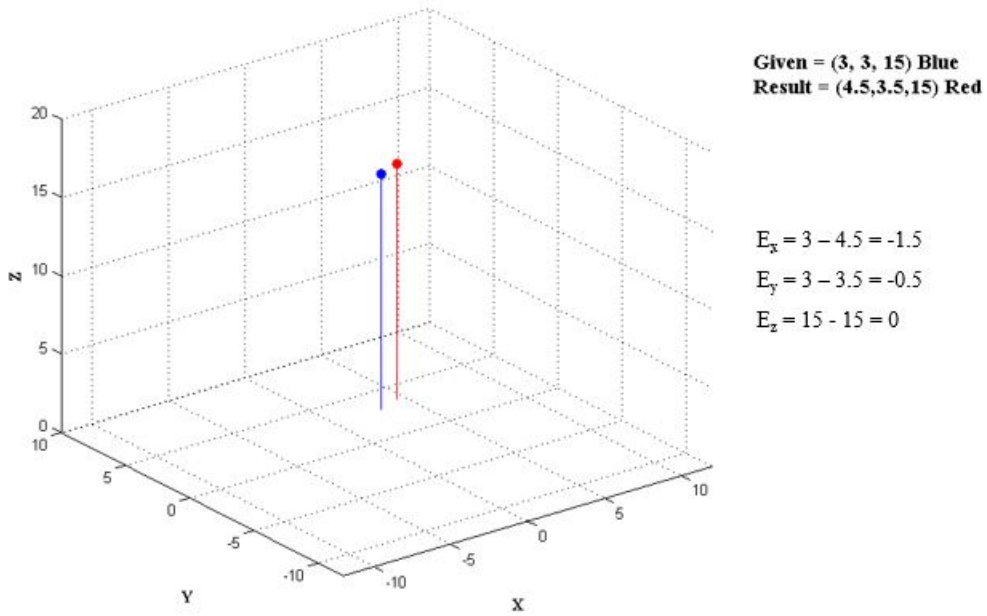

Figure 7.4: Second test position.

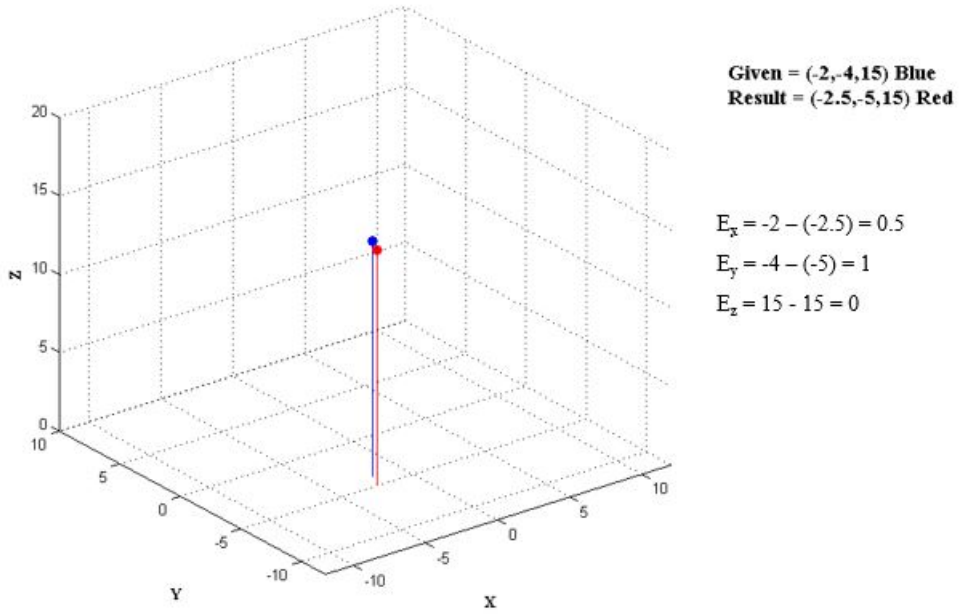

Figure 7.5: Third test position.

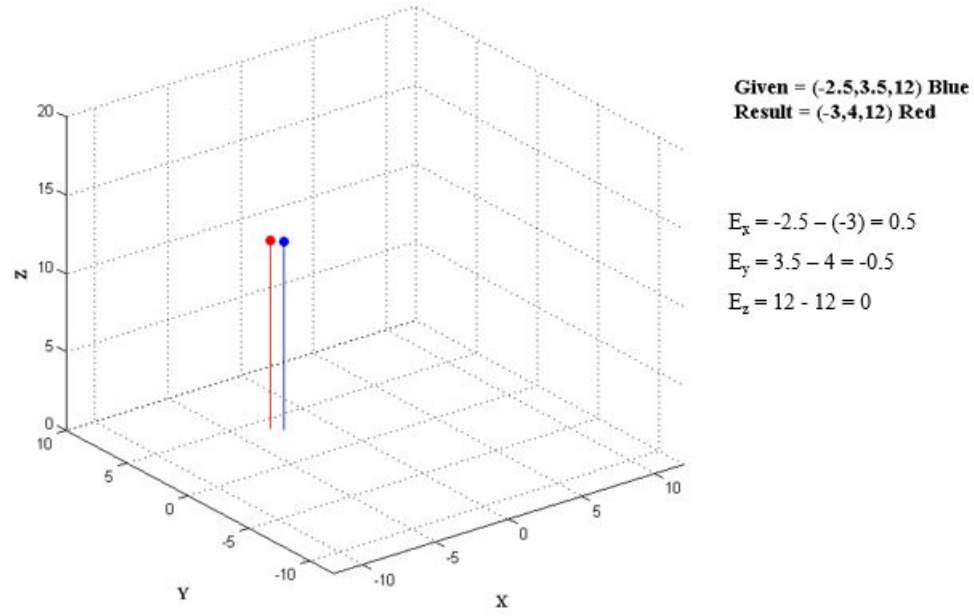

Figure 7.6: Fourth test position.

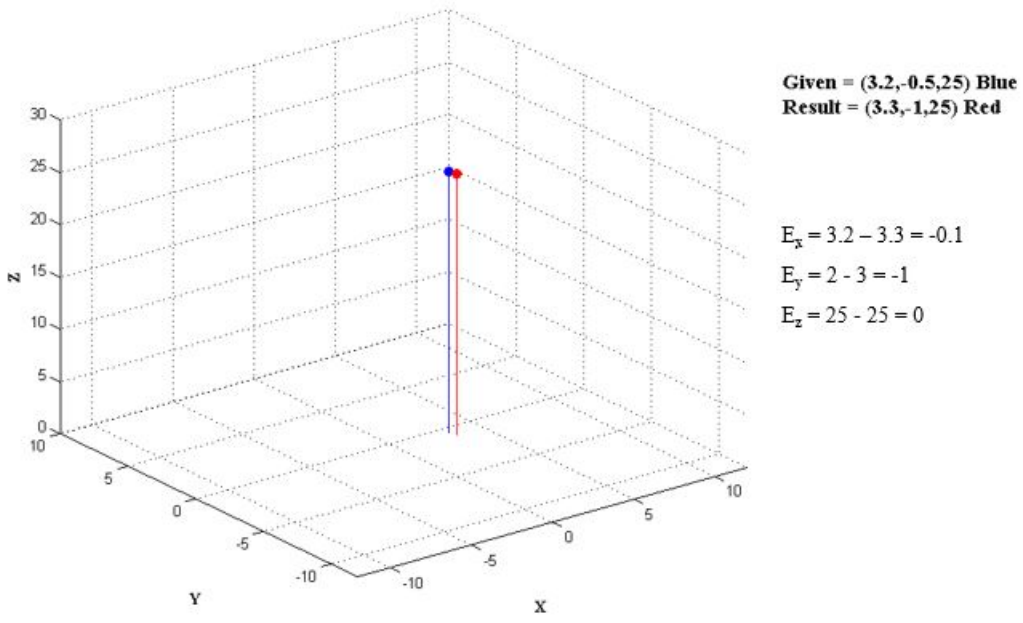

Figure 7.7: Fifth test position.

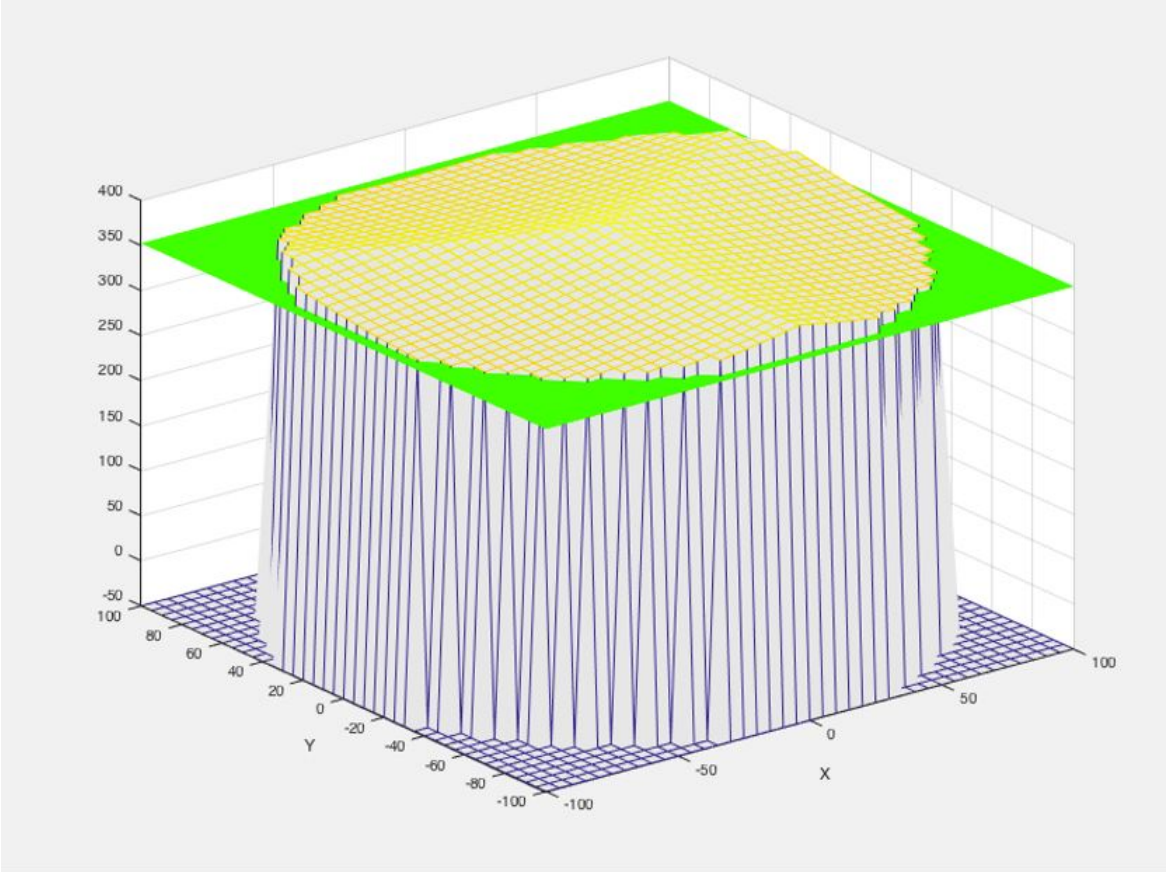

Figure 7.8: Maximum workspace volume of the end effector.

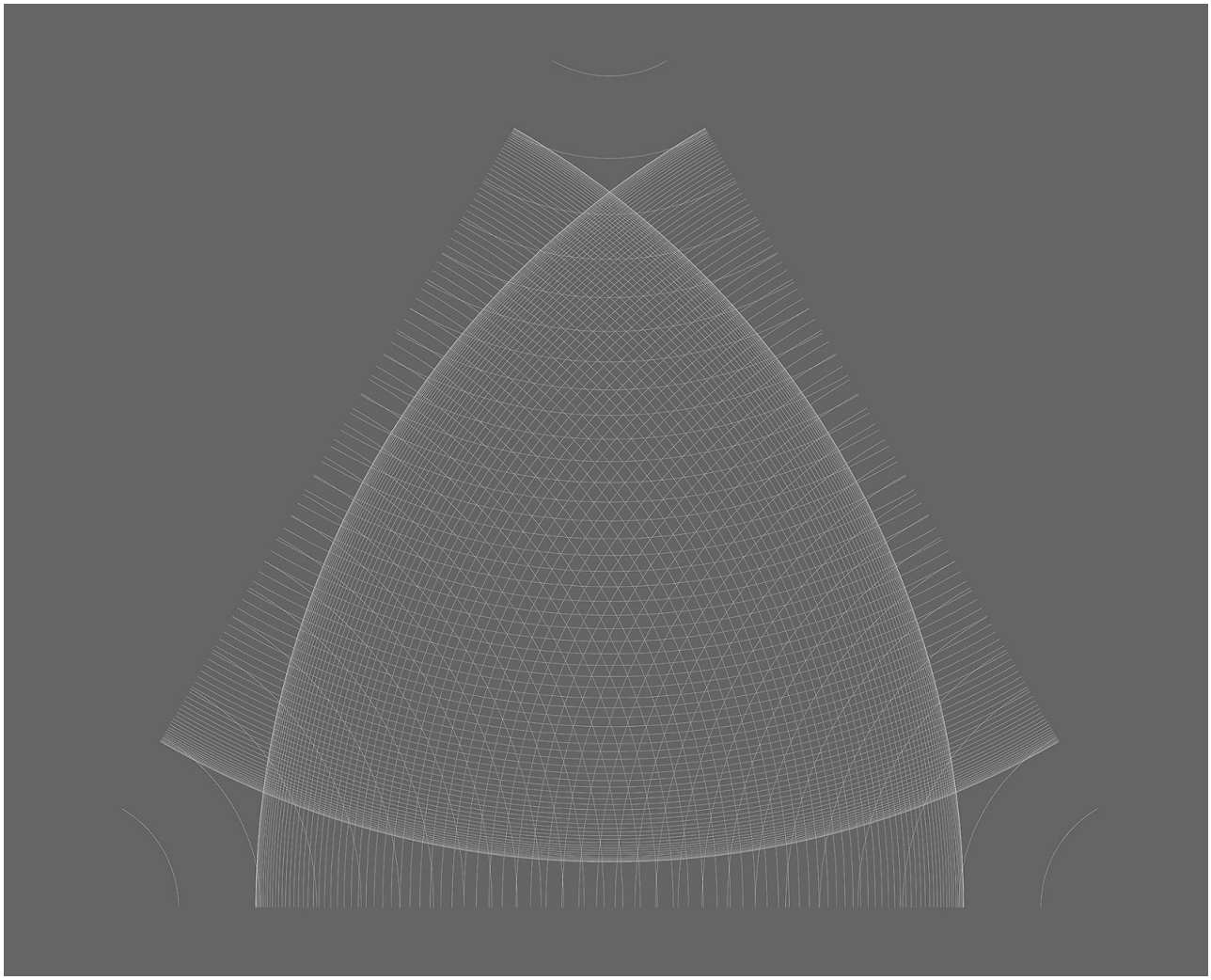

Figure 7.9: Effective workspace in XY plane.

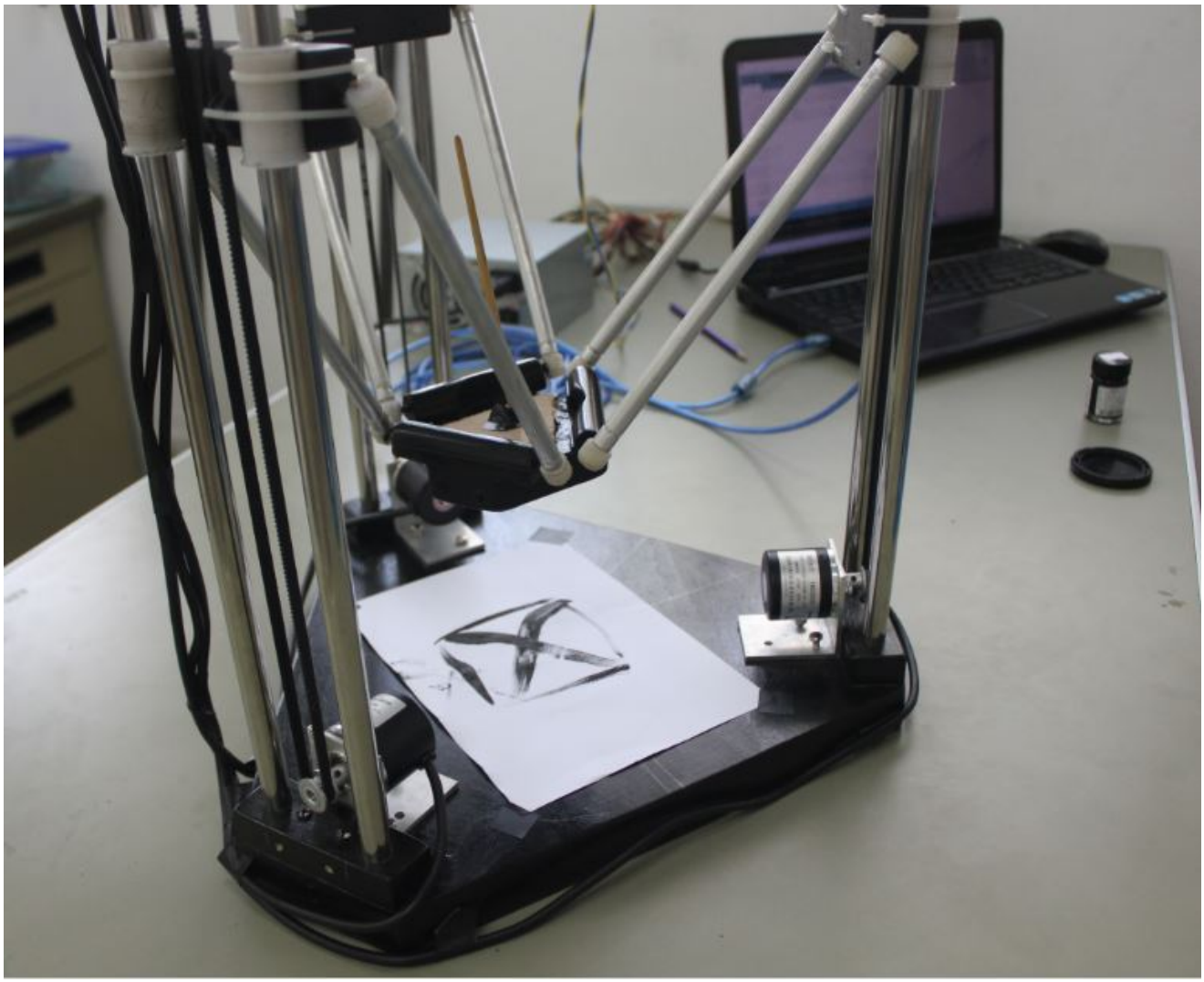

Figure 7.10: Result of the drawing application.

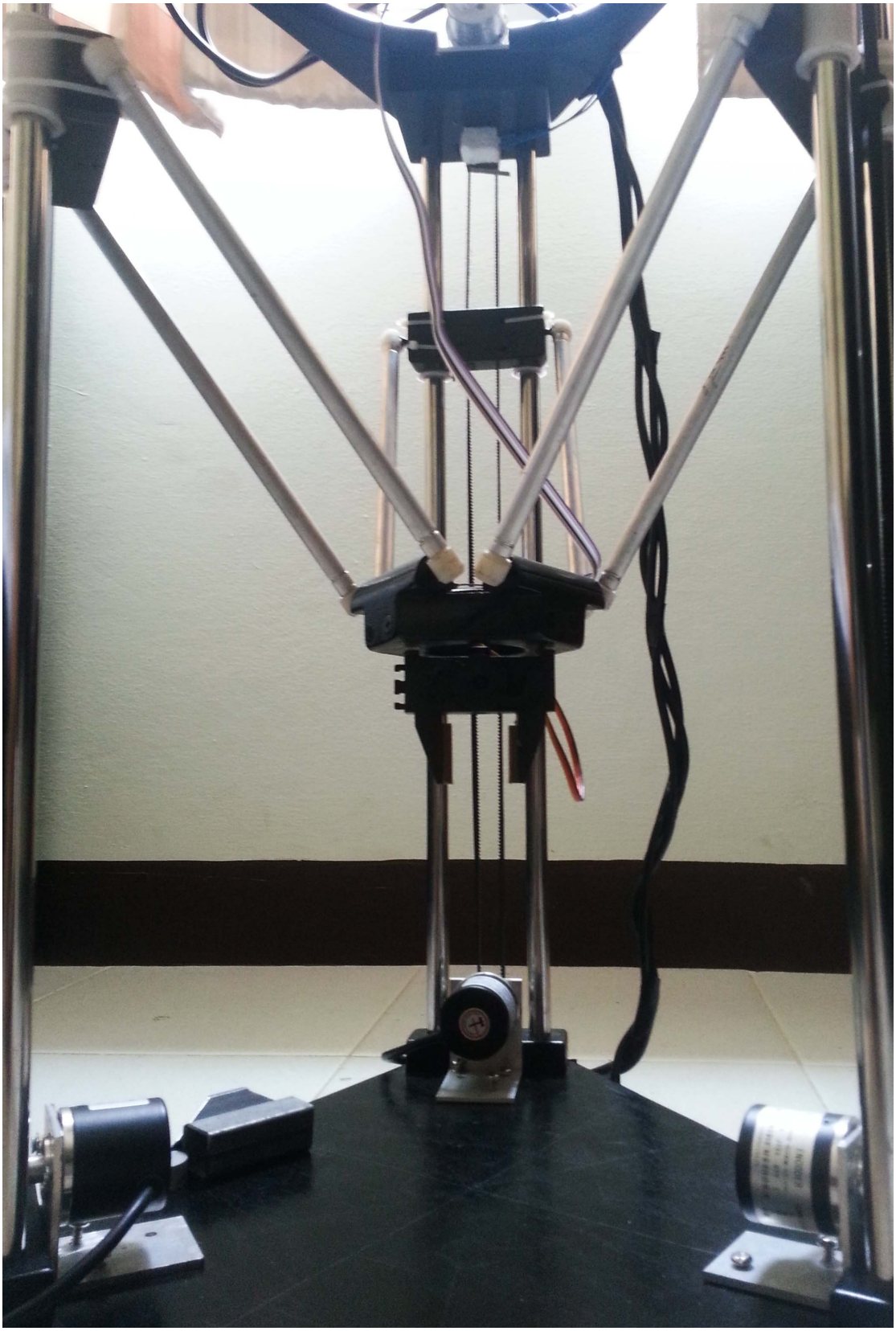

Figure 7.11: Pick and place assembly.

7.3 Conclusion

It was seen that the small errors developed in the columns were not that significant in the overall positioning of the robot since the performance in the above applications was satisfactory. The project was completed within the time plan that was provided in the proposal. The fore-casted budget was exceeded but the objectives were attained.

7.4 Future possible developments

The speed versus the accuracy of this prototype can certainly be enhanced by using higher RPM rated motors along with proper linear bearings and a faster control platform. Also the 5 DOF assembly presented in chapter 5 can be executed providing more versatility for the robot.

A Graphical-User-Interface (GUI) can be developed to input the coordinates for the robot to reach in order to execute some predefined task. Moreover an interface can be developed to sketch a path for the robot to move so that by generating a G-code through it the user will be able to define a specific path that the end effector could follow. This will have higher level applications such as 3D printing if a nozzle is fixed at the end effector to provide an in-feed of 3D printing material, the path could be defined.

A parallel robot originally comes with a higher advantage of high operating speeds along with high accuracy because of its design attributes, so this prototype can be further developed to perform high speed precision tasks.

References

Arduino. (2012). *PID Library*. (Information available at http://playground.arduino.cc/Code/ PIDLibrary)

Brett Beauregard. (2011). *Improving the Beginners PID*. (Information available at http:// brettbeauregard.com/blog/2011/04/ improving-the-beginners-pid-introduction/)

Farid Golnaraghi and Benjamin C. Kuo. (2010). Automatic Control Systems. Ninth Edition. Simon Fraser University and University of Illinois at Urbana-Champaign.

H.H.Manh. (2014). *Encoders*. Electro-Mechanical Machine Design Labs. Mechatronics, AIT Thailand.

Instructables. (2014). *Arduino Modules - L298N Dual H-Bridge Motor Controller*. (Information available at http:// www.instructables.com/id/ Arduino-Modules-L298N-Dual-H-Bridge-Motor-Controll/?ALLSTEPS)

Mecademic article resources. (2014). *What is a parallel robot?*. (Information available at http:// www.mecademic.com/Whatis-a-parallel-robot.html)

Steve Graves. (2012). *Delta Robot Kinematics*. Johann C. Rocholl. (Rostock) Style

Wikibook. (2012). *Latex*. (Information available at http://en.wikibooks.org/wiki/ LaTeX)

Wikipedia. (2014). *Parallel Robots*. (Information available at http:// en.wikipedia.org/wiki/Parallel manipulator)

Y. Patel and P. George. (2012). *Parallel Manipulators Applications - A Survey*. Modern Mechanical Engineering. Vol 2. pg 57-64

Zhongfei Wang, Guan Wang, Shiming Ji, Yuehua Wan and Qiaoling Yuan. (2007). *Optimal Design of a Linear Delta Robot for the Prescribed Cuboid Dexterous Workspace*. The MOE Key Laboratory of Mechanical Manufacture and Automation Zhejiang University of Technology Hangzhou 310014, China.