World Journal of Engineering Research and Technology
WJERT
www.wjert.org
SJIF Impact Factor: 5.924

EXPERIMENTAL STUDY AND ALGORITHM FOR A GANTRY ROBOT

Abhinanth M.M.*[1] and Ramesh Kumar R.[2]

System Engineer, Infosys, Thiruvananthapuram.

Article Received on 17/08/2020 **Article Revised on 07/09/2020** **Article Accepted on 27/09/2020**

***Corresponding Author**
Abhinanth M.M.
System Engineer, Infosys, Thiruvananthapuram.

ABSTRACT

A scaled version of a gantry robot to cover an envelope of 300 x 300 x150mm is designed to establish the accuracy of end positions prior to the manufacturing of any industrial gantry robot. A desk top 3-D printer is suitably modified for the fabrication of robot and measured translational motions are compared with prediction following well known D-H parameter. A good agreement on the end positions between the test and prediction are illustrated. Arduino code employed to control the real time execution of the gantry robot movement in a semi-automatic mode is provided for the designer.

KEYWORDS: DH Parameter, Gantry robot, Pose, End effector.

INTRODUCTION

Over couple of decades, industrial robots are worldwide implemented in various hazardous and non-hazardous applications. Mostly these are being used for material handling, welding, painting, assembling of parts, packaging, handling hazardous materials, undersea operations. In order to access the proven technology on forward kinematics for large industrial manipulator gantry robots, it is necessary to produce robots of scaled geometry and compare with test data on its end position vectors. Well-known DH approach developed in the year 1955 at North Western University to predict the end position of robot within a coordinate frame space is availed in the present study to predict the end positions analytically.[1] The Denavit-Hartenberg analysis (DH) is presented to build the homogeneous transformations matrices between the robot joint axes.

The various commanding methods can be employed such as through pc with Arduino or Bluetooth control for remote application and accelerometer based on gesture control method.[2- 5] It is well-known that there are two types of kinematic analyses; forward and inverse kinematic analysis, Forward kinematic analysis is concerned with the relationship between the joint angle of the robot manipulator and the position and orientation of the end-effector.[6] The forward kinematic analysis of any robot configuration is simple to do but greater challenge is to analyse the inverse kinematics solution of the robot configuration using the final position of the robot.[7] To perform like a human being, these robots normally are designed with a high center of mass, which makes it challenging to maintain stability while achieving high performance on complex and unpredictable terrain.[8] The kinematic analysis of composite robotic manipulator with three degree-of-freedom (DOF) by moving in 3D spaces.[9] A stationary five axis articulated robot arm with the kinematic modelling which is used for effective robotic manipulation job in its workspace.[10] A modelling process of single link flexible robotic manipulator and equation of motion of the system for clamped boundary condition is solved by FEM.[11] The performance evaluation and computational requirements of the FEM in the simulation of a flexible robotic manipulator.[12] Execution of the motion of robot is carried out by Arduino hardware and associated software. It provides readily available examples to execute a motor individually and by combination. Stepper motors and servo motors are commonly used for the motions. Stepper motor used in the present study works on 12V DC and servo motor works on 5V DC. So SMPS unit with a RAMPS board to supply the power to motors are availed. Once a controlling command for a definite movement to achieve an intermediate end location, it necessary to ascertain the accuracy following an error correction feedback system. However, in this study, after each movement intermediate location is made as a new reference location for the subsequent movement. The specification of motors is given in appendix 3.

In this study, a new gantry robot is fabricated and kinematic analyses of that consist of three translations degrees of freedom are carried out following DH parameter. In order to compare the prediction on end position, gantry robot is commanded with the serial monitor available in Arduino software and end positions are arrived at a set up step by linear movement of the stepper motor. A good agreement is observed between predictions following DH parameter as well as Roboanalyzer with present test results on end positions.

METHODOLOGY

D-H parameter table for gantry robot is defined first and then using these values transformation matrix for each link is established (Table 1). The end effecter matrix is obtained as post multiplication product of these matrices.

A. Fabrication of Gantry Robot

While the drive technologies, work envelope geometries such as spherical, cylindrical and rectangular and also the different motion control methods provide convenient ways to broadly classify robots. There are number of additional characteristics that allow the user to further specify robotic manipulators. The gantry robot is designed to provide a work space of 300 x300 x150(vertical) mm (Fig. 1). The translation movements for 10mm is aimed with a speed of 0.06 seconds. Maximum speed of servo motor used for the gripper is 0.09 seconds for 60^0rotation. The expected accuracy of the unit is 0.012mm.

The guide block is designed to move in X direction which holds two servo motors for one rotation and one picking (Fig.1 (a)). This guide block unit will pick the object from the bed and place the object in another place. The load carrying capacity of the robot is around 200 grams. The guide block is modelled in software and then its fabricated using 3d printing technology. The fabricated guide block is fixed in the place of the extruder unit in the anet a6 printer to obtain the gantry robot setup. The modelled part is shown in Fig. 1(c).

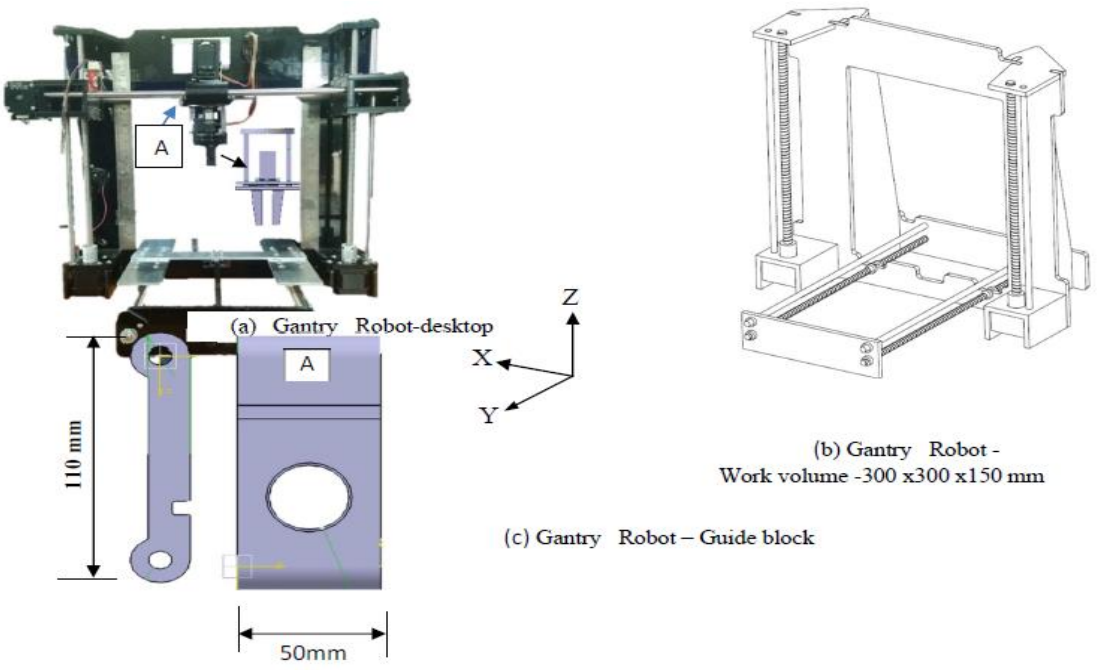

Fig.1. Gantry Robot

B. Design of Gripper Block

The gripper is designed for the point to point motion control in order to pick-and-place object from one space to another in the work volume. For the large scale operation loading and unloading is also done by the point to point motion control method. The gripper unit is also fabricated like guide block after that which is fitted with the robot. A 9 gm servo motor powers up the gripper to pick and drop object from the work space. Gripper totally rotatable in nature by fixing it on the servo.

C. Program Code

The interface between PC and robot is activated using Arduino code which enables the function of actuators. In other words, both stepper and servo motors are given commands. The specification of these motors is given below. All movement command along x, y, z axes also by guide block and gripper are given in Appendix1.

Table 1: D-H Parameters of gantry robot.

Joint	a (Link length)	d (Joint offset)	α (Twist angle)	θ (Joint angle)
1 (X axis)	a1=0	300(JV)	90	0
2 (Y axis)	a2=0	300(JV)	90	0
3 (Z axis)	a3=0	150(JV)	90	0
4 (Guide block)	a4=0	0	0	180(JV)
5 (Gripper)	a5=0	0	0	60(JV)

D. Robot Gripper Block End Position Prediction

Transformation matrix for first link with respect to base frame is given below

$$ {}^{0}T_{1}(\theta, d, a, \alpha) = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 \cos\alpha_1 & \sin\theta_1 \sin\alpha_1 & a_1 \cos\theta_1 \\ \sin\theta_1 & \cos\theta_1 \cos\alpha_1 & -\cos\theta_1 \sin\alpha_1 & a_1 \sin\theta_1 \\ 0 & \sin\alpha_1 & \cos\alpha_1 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1) $$

Transformation matrix for second link with respect to first frame is given below

$$T_2(\theta, d, a, \alpha) = \begin{bmatrix} \cos\theta_2 & -\sin\theta_2\cos\alpha_2 & \sin\theta_2\sin\alpha_2 & a_2\cos\theta_2 \\ \sin\theta_2 & \cos\theta_2\cos\alpha_2 & -\cos\theta_2\sin\alpha_2 & a_2\sin\theta_2 \\ 0 & \sin\alpha_2 & \cos\alpha_2 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

Transformation matrix for third link with respect to second frame is given below

$$^2T_3(\theta, d, a, \alpha) = \begin{bmatrix} \cos\theta_3 & -\sin\theta_3\cos\alpha_3 & \sin\theta_3\sin\alpha_3 & a_3\cos\theta_3 \\ \sin\theta_3 & \cos\theta_3\cos\alpha_3 & -\cos\theta_3\sin\alpha_3 & a_3\sin\theta_3 \\ 0 & \sin\alpha_3 & \cos\alpha_3 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3)$$

The homogenous transformation matrix of the end effector frame with respect to base frame i.e. Transformation matrix T is obtained by the post multiplication of the above individual homogenous transformations.

$$^0T_3(\theta, d, a, \alpha) = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & d_1 + d_2 + d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$

RESULTS AND DISCUSSION

Synthesis of gantry robot by suitably modifying an existing Anet a6 three –D printer is given in Fig. 1. The DH parameter method available in literature is used to predict the end positions of gantry robot of small size for specified motions along the three mutually perpendicular directions and given Table 2(a) - (d). Same Tables also compare the predicted values with measured data with a resolution of 0.012mm. Code for real time execution is generated such that after each command for movement, the new position is set as a reference point for the

next operation. The code is provided in the Appendix 1.

It may be noted from Table 2(a) that for 1mm displacement in X- direction from an initial position of 44.05mm of the gripper block of the robot shown in Fig.1 (a) is expected to move to 45.05 mm by a command "C1X10". However, from actual measurement it is found to be only at 45.03mm with 2% error. Further the commands are given for four more times from each intermediate final position along X-axis. The variation in error is within 1%. Similar comparison between the prediction and test for a movement of 10mm displacement is observed within 0.6%.It may be noted that the maximum absolute deviation is 0.06mm.Test is repeated along Y-axis and the maximum absolute deviation is found to be 0.03mm and percentage deviation with respect to prediction is 2%. The study was not carried out along Z axis movement due to test setup constraint for measurements.

It is concluded that even with step wise command to set the intermediate local end locations as reference position for subsequent movement instead of error correction by feedback system, it is possible to achieve the final end position accurately. The x command is implemented through program as shown below (vide Appendix 1.1)

Table 2. Comparison of translator motions predicted with measured data

(a)Along X-axis for 1mm movement

Trial No.	Command	Initial value	Final value	Error
1	C1X10	44.05	45.03	2%
2	C1X10	45.03	46.04	1%
3	C1X10	46.04	47.04	0%
4	C1X10	47.04	48.03	1%
5	C1X10	48.03	49.02	1%

C1X10 – Command for 1mm displacement in X-direction with zero movement in Y- direction from any initial or intermediate position

(b) Along X-axis for 10mm movement

Trial No.	Command	Initial value	Final value	Error
1	C1X100	27.40	37.38	0.2%
2	C1X100	37.38	47.36	0.2%
3	C1X100	47.36	57.32	0.4%
4	C1X100	57.32	67.37	0.5%
5	C1X100	67.37	77.31	0.6%

C1X100 – Command for 10mm displacement in X-direction with zero movement in Y- direction from any initial or intermediate position

(c)Along Y-axis for 1mm movement

Trial No.	Command	Initial value	final value	Error
1	C1X0Y10	44.05	45.03	2%
2	C1X0Y10	45.03	46.06	3%
3	C1X0Y10	46.06	47.04	2%
4	C1X0Y10	47.04	48.02	2%
5	C1X0Y10	48.02	49.04	2%

C1X0Y10 – Command for 1mm displacement in Y-direction with zero movement in X- direction from any initial or intermediate position

(d) Along Y-axis 10mm movement

Trial No.	Command	Initial value	Final value	Error
1	C1X0Y100	27.40	37.38	0.2%
2	C1X0Y100	37.38	47.37	0.1%
3	C1X0Y100	47.37	57.36	0.1%
4	C1X0Y100	57.38	67.36	0.2%
5	C1X0Y100	67.36	77.34	0.2%

C1X0Y100– Command for 100mm displacement in Y- direction with zero movement in X- direction from any intermediate position

Void C1

{x = Serial.parseInt(); Y = Serial.parseInt()}

When value of X is assigned to X62(6.2 mm movement along X- direction),then "parseInt" accept only numerical value. Similarly C1X10 means, 10mm movement along X direction. Similarly When value of Y is assigned to Y62(6.2 mm movement along Y- direction),then "parseInt" accept only numerical value. Similarly C1X0Y10 means, 10mm movement along Y direction.

The displacement versus trail number graph for x and y direction are given below. There is a small error between actual position and predicted position. The error ranges between -0.025% to + 0.025%.

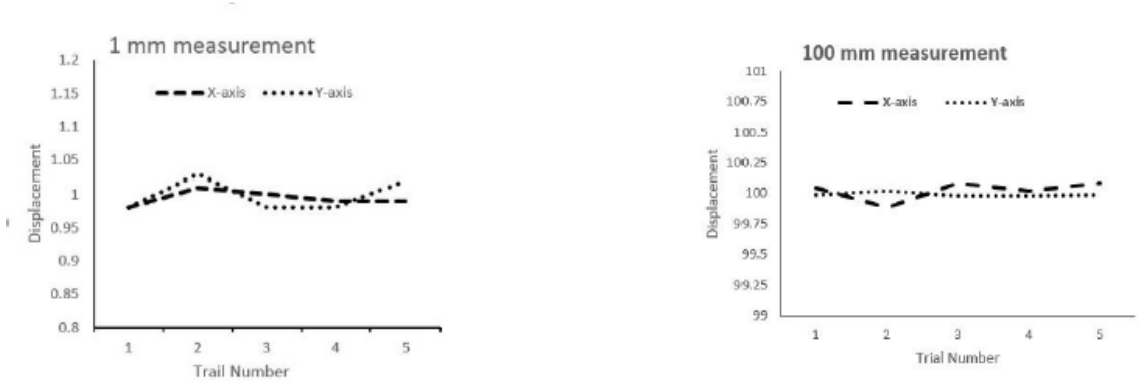

Fig. 2: Graphical representation of 1mm and 100mm displacement in X and Y direction.

CONCLUSION

A well-established technique of DH parameter has been availed for the prediction of end position of scaled gantry robot required for the implementation of industrial manipulator. Details on fabrication of the robot are given for a ready reference. Comparison with measured data on end position with the prediction has shown good agreement within a resolution of 0.012mm. For the real time execution after each command for movement, the new position has been set as a reference point for the next operation for end positions along three mutually perpendicular axes. A generated code is provided in annexure. Synthesize of small scale robot presented will be useful so that large size robot end positions can be confidently envisaged.

REFRENCES

1. Denavit J and Hartenberg R S A kinematic analysis notation for lower pair mechanisms based on matrices .Trans ASME J. ApplMech, 1955; 23: 215-221.
2. Marvroids.C and Lee E Geometric design of 3r robot manipulators for reaching four end effector spatial poses, The international journal of robotics research, 2004; 23(3): 247-254.
3. Xu D, Acosat C A, Gan J Q and Hu H Analysis of the inverse kinematics for a 5 dof manipulator, International Journal of Automation and Computing, 2005; 2: 114-124.
4. Constantin D, Lupose M, Bachu C and Bulgia D Forward kinematics analysis of an industrial robot, New Developments in Mechanics and Mechanical Engineering, 2013; 03: 90-95.
5. Elot E, Deepak B B, Parhi D R and Srinivas J et.al, Design & kinematic analysis of an articulated robotic manipulator", International Journal of Mechanical and Industrial Engineering, 2013; 3: 105-108.
6. Khatamin A Solving kinematics problems of a 6 dof robot manipulator, International Conference Scientific Computing, 2015; 2.
7. Game R U, Davkhare A A, .Pakhale S S and Shinde V B Kinematic analysis of various robot configurations, International Research Journal of Engineering and Technology, 2017; 4: 921-933.
8. Rojas S,He Shen, Griffths H, Ni Li and Zhang L, Motion and gesture compliance control for high performance of a wheeled humanoid robot, International Mechanical Engineering and Exposition, IMECE2017-72337, 2017.
9. Chang Y, Ma S, Wang H and Tan D. A Kinematic modelling method for a wheeled mobile robot, International Conference on Mechatronics and Automation, 2009.

10. Hussien M T, Al-Robaiy M J and Al-Shjary M A. Mathematical Modelling of flexible Robotic Arm using Finite Element Method, 2008.
11. Tokhi M O and Mohamed Z. Finite element approach to dynamic modelling of a flexible robot manipulator: performance evaluation and computational requirements. Communications in Numerical Methods in Engineering, 1999; 15: 669-678.
12. Loudini M, Boukhetala D, Tadjine M and Boumehdi M.A. Application of timoshenko beam theory for deriving motion equations of a lightweight elastic link robot manipulator. International Journal of Automation, Robotics and Automomous System, 2006; 05(2): 11-18.

Appendix I

1.Mega code

```
#include <Servo.h>
#define X_STEP_PIN 54
#define dirPinx 55
#define X_DIR_PIN 55
#define stepPinx 54
#define X_ENABLE_PIN 38
#define X_MIN_PIN 3
#define ex 3
#define X_MAX_PIN 2
#define Y_STEP_PIN A6
#define dirPiny A7
#define Y_DIR_PIN A7
#define stepPiny A6
#define Y_ENABLE_PIN 56
#define Y_MIN_PIN 14
#define ey 14
#define Y_MAX_PIN 15
#define Z_STEP_PIN 46
#define dirPinz 48
#define stepPinz 46
#define Z_DIR_PIN 48
#define Z_ENABLE_PIN 48
#define Z_MIN_PIN 18
#define ez 18
#define Z_MAX_PIN 19
#define FAN_PIN 9
#define P_SERVO_PIN 11
#define R_SERVO_PIN 4
//Servo pins 11 6 5 4
Servo plucker,rotate;
int C = 0, x = 0, y = 0, z = 0, p =
120, r = 90;
void setup() {
//plucker.attach(P_SERVO_PIN);
//rotate.attach(R_SERVO_PIN);
pinMode(X_STEP_PIN , OUTPUT);
pinMode(X_DIR_PIN , OUTPUT);
pinMode(X_ENABLE_PIN , OUTPUT);
pinMode(Y_STEP_PIN , OUTPUT);
pinMode(Y_DIR_PIN , OUTPUT);
pinMode(Y_ENABLE_PIN , OUTPUT);
pinMode(Z_STEP_PIN , OUTPUT);
pinMode(Z_DIR_PIN , OUTPUT);
pinMode(Z_ENABLE_PIN , OUTPUT);
digitalWrite(X_ENABLE_PIN , LOW);
digitalWrite(Y_ENABLE_PIN , LOW);
digitalWrite(Z_ENABLE_PIN , LOW);
pinMode(X_MIN_PIN, INPUT);
pinMode(ex, INPUT);
pinMode(ey, INPUT);
pinMode(ez, INPUT);
pinMode(X_MAX_PIN, INPUT); //X
EndstoppinMode(Y_MIN_PIN, INPUT);
pinMode(Y_MAX_PIN, INPUT); //Y
EndstoppinMode(Z_MIN_PIN, INPUT);
pinMode(Z_MAX_PIN, INPUT); //Z
Endstop
Serial.begin(9600);
//plucker.write(p);
// rotate.write(r);
Serial.print("Ready to work");//
Tells program is is ready
delay(50);}
void loop ()
{ while (!Serial.available())
{ C = Serial.parseInt();
//Serial.println(C);
switch (C)
{case 1 : C1(); break;
case 2 : C2(); break;
case 3 : C3(); break;
```

```
case 4 : C4(); break;
case 5 : C5(); break;
case 6 : C6(); break;
}//Serial.print('F');
Serial.flush();
/* while (Serial.read() != 'L')
{ delay(50);
Serial.print('F');
}*/ }}
```

1.1 Case 1

```
void C1()
{x = Serial.parseInt();
y = Serial.parseInt();
z = Serial.parseInt();
xaxis(x);
delay(10);
yaxis(y);
delay(10);
zaxis(z);
delay(10);}
```

1.2 case 2

```
voidendstop()
{ //Serial.println("endstopx");
digitalWrite(dirPinx,LOW);
while(digitalRead(ex)==HIGH)
{ digitalWrite(stepPinx,HIGH);
delayMicroseconds(1500);
digitalWrite(stepPinx,LOW);
delayMicroseconds(1500); }
digitalWrite(dirPinx,HIGH);
while(digitalRead(ex)==LOW)
{digitalWrite(stepPinx,HIGH);
delayMicroseconds(1500);
digitalWrite(stepPinx,LOW);
delayMicroseconds(1500); }}
```

1.3 Case 3

```
voidendstopy()
{ digitalWrite(dirPiny,HIGH);
while(digitalRead(ey)==HIGH)
{digitalWrite(stepPiny,HIGH);
delayMicroseconds(1500);
digitalWrite(stepPiny,LOW);
delayMicroseconds(1500);
}digitalWrite(dirPiny,LOW);
while(digitalRead(ey)==LOW)
{ digitalWrite(stepPiny,HIGH);
delayMicroseconds(1500);
digitalWrite(stepPiny,LOW);
delayMicroseconds(1500); } }
```

1.4 Case 4

```
voidendstopz()
{digitalWrite(dirPinz,LOW);
while(digitalRead(ez)==LOW)
{ digitalWrite(stepPinz,HIGH);
delayMicroseconds(1500);
digitalWrite(stepPinz,LOW);
delayMicroseconds(1500);}
int n=0;
digitalWrite(dirPinz,HIGH);
while(n<=500)
{ digitalWrite(stepPinz,HIGH);
delayMicroseconds(1500);
digitalWrite(stepPinz,LOW);
delayMicroseconds(1500);
n++;}}
```

1.5 Case 5

```
void C5()
{delay(10);
r = Serial.parseInt();
rotate.attach(R_SERVO_PIN);
delay(10);
rotate.write(r);
delay(1000);
rotate.detach();}
```

1.6 Case 6

```
void C6()
{delay(10);
p = Serial.parseInt();
delay(50);
plucker.attach(P_SERVO_PIN);
delay(10);
plucker.write(p);
delay(1000);
plucker.detach();
}
```

APPENDIX 2

2.1 Programme for X stepper motor

```
voidxaxis(intxs)
{ if (xs> 0)
digitalWrite(X_DIR_PIN , HIGH);
else
digitalWrite(X_DIR_PIN , LOW);
for (inti = 0; i< abs(xs); i++)
{ digitalWrite(X_STEP_PIN , LOW);
delayMicroseconds(1000);
digitalWrite(X_STEP_PIN , HIGH);
delayMicroseconds(1000);} }
```

2.2 Programme for Y stepper motor

```
voidyaxis(intys)
{ if (ys> 0) digitalWrite(Y_DIR_PIN
```

```
, HIGH);
else
digitalWrite(Y_DIR_PIN , LOW);
for (inti = 0; i< abs(ys); i++)
{ digitalWrite(Y_STEP_PIN , LOW);
delayMicroseconds(1000);
digitalWrite(Y_STEP_PIN , HIGH);
delayMicroseconds(1000);}}
```

2.3 Programme for Z stepper

```
voidzaxis(intzs)
{ if (zs> 0) digitalWrite(Z_DIR_PIN
, HIGH);
else {
digitalWrite(Z_DIR_PIN , LOW);}
if (abs(zs) < 501)
{for (inti = 0; i< abs(zs); i++)
{digitalWrite(Z_STEP_PIN , LOW);
delayMicroseconds(1500);
digitalWrite(Z_STEP_PIN , HIGH);
delayMicroseconds(1500);}}
else
{for (inti = 0; i< 250; i++)
{digitalWrite(Z_STEP_PIN , LOW);
delayMicroseconds(1500 - (i * 2));
digitalWrite(Z_STEP_PIN , HIGH);
delayMicroseconds(1500 - (i * 2));}
for (inti = 0; i< (abs(zs) - 500);
i++)
{digitalWrite(Z_STEP_PIN ,
LOW);
delayMicroseconds(1000);
digitalWrite(Z_STEP_PIN , HIGH);
delayMicroseconds(1000);}
for (inti = 0; i< 250; i++)
{digitalWrite(Z_STEP_PIN ,
LOW);
delayMicroseconds(1000 + (i * 2));
digitalWrite(Z_STEP_PIN , HIGH);
delayMicroseconds(1000 + (i * 2));}}
```

2.4 Programme for XY combination

```
voidxycomb(intxs, intys)
{if (xs> 0) digitalWrite(X_DIR_PIN
, HIGH);
Else
{ digitalWrite(X_DIR_PIN , LOW);}
if (ys> 0) digitalWrite(Y_DIR_PIN
, HIGH);
else
{ digitalWrite(Y_DIR_PIN , LOW);}
int extra = abs(xs - ys), rest = 0;
if (abs(xs) == abs(ys)) rest =
abs(xs);
if (abs(xs) > abs(ys))
{ rest = abs(xs) - extra;
if (abs(extra) < 501)
{for (inti = 0; i< abs(extra); i++)
{digitalWrite(X_STEP_PIN , LOW);
delayMicroseconds(1000);
digitalWrite(X_STEP_PIN , HIGH);
delayMicroseconds(1000);}}
else
{ for (inti = 0; i< 250; i++)
{ digitalWrite(X_STEP_PIN , LOW);
delayMicroseconds(1000 - (i * 3));
digitalWrite(X_STEP_PIN , HIGH);
delayMicroseconds(1000 - (i * 3));}
for (inti = 0; i< (abs(extra) - 500);
i++)
{ digitalWrite(X_STEP_PIN , LOW);
delayMicroseconds(250);
digitalWrite(X_STEP_PIN , HIGH);
delayMicroseconds(250);}
for (inti = 0; i< 250; i++)
{digitalWrite(X_STEP_PIN ,
LOW);
delayMicroseconds(250 + (i * 3));
digitalWrite(X_STEP_PIN , HIGH);
delayMicroseconds(250 + (i * 3));}}}
else
{rest = abs(ys) - extra;
if (abs(extra) < 501)
{for (inti = 0; i< abs(extra);
i++)
{ digitalWrite(Y_STEP_PIN , LOW);
delayMicroseconds(1000);
digitalWrite(Y_STEP_PIN , HIGH);
delayMicroseconds(1000); }}
else
{for (inti = 0; i< 250; i++)
{ digitalWrite(Y_STEP_PIN , LOW);
delayMicroseconds(1000 - (i * 3));
digitalWrite(Y_STEP_PIN , HIGH);
delayMicroseconds(1000 - (i * 3));}
for (inti = 0; i< (abs(extra) - 500);
i++)
{ digitalWrite(Y_STEP_PIN , LOW);
delayMicroseconds(250);
digitalWrite(Y_STEP_PIN , HIGH);
delayMicroseconds(250); }
for (inti = 0; i< 250; i++)
```

```
{ digitalWrite(Y_STEP_PIN , LOW);
delayMicroseconds(250 + (i * 3));
digitalWrite(Y_STEP_PIN , HIGH);
delayMicroseconds(250 + (i * 3));}}}
if (rest < 501)
{digitalWrite(X_STEP_PIN , LOW);
digitalWrite(Y_STEP_PIN , LOW);
delayMicroseconds(1000);
digitalWrite(X_STEP_PIN , HIGH);
digitalWrite(Y_STEP_PIN , HIGH);
delayMicroseconds(1000);}
else
{ for (inti = 0; i< 250; i++)
{ digitalWrite(X_STEP_PIN , LOW);
digitalWrite(Y_STEP_PIN , LOW);
delayMicroseconds(1000 - (i * 2));
digitalWrite(X_STEP_PIN , HIGH);
digitalWrite(Y_STEP_PIN , HIGH);
delayMicroseconds(1000 - (i * 2)); }
for (inti = 0; i< (abs(rest) - 500);
i++)
{ digitalWrite(X_STEP_PIN , LOW);
digitalWrite(Y_STEP_PIN , LOW);
delayMicroseconds(500);
digitalWrite(X_STEP_PIN , HIGH);
digitalWrite(Y_STEP_PIN , HIGH);
delayMicroseconds(500);}
for (inti = 0; i< 250; i++)
{ digitalWrite(X_STEP_PIN , LOW);
digitalWrite(Y_STEP_PIN , LOW);
delayMicroseconds(500 + (i * 3));
digitalWrite(X_STEP_PIN , HIGH);
digitalWrite(Y_STEP_PIN , HIGH);
delayMicroseconds(500 + (i * 3));
}}}
```

Appendix 3

1.12g servo motor specification

Operating voltage:-4.5v to 6.0v

Operating current:- 50mA to 180mA

Operating speed:-0.09 sec for 60degree

Torque:- 1.6kg.cm

Weight:-12g

Limit angle:-180 degree

2. 9g servo motor specification

Operating voltage:-4.5v to 6.0v

Operating current:- 50mA to 180mA

Operating speed:-0.12 sec for 60degree

Torque:- 1.6kg.cm

Weight:-9g

Limit angle:-180 degree