Robotics Fundamentals coursework

By

Wentao Gao 21070393 Jingzhi Wu 21070401

MSc Robotics Report

Department of Engineering Mathematics

UNIVERSITY OF BRISTOL

&

Department of Engineering Design and Mathematics

UNIVERSITY OF THE WEST OF ENGLAND

January 12, 2022

Coursework Report - Group Cover Page

Coursework: Serial and Parallel Robot Kinematics

Student name and number	PART I.A	PART I.B	PART II.1	PART II.2	PART III
Jingzhi Wu 21070401	√	√	√	√	√
Wentao Gao 21070393	√	√	√	√	√

Note: Students must tick the sections that they have been undertaking or helped with.

Contents

List of Tables

1 Part 1: Kinematics of Lynxmotion Arm

1.1 Forward Kinematics

Forward kinematics is a process which utilize the known angle of joints and length between links to calculate the end position of the end-effector .In this section ,firstly the setting of frame on Lynxmotion arm will be discussed ,then the calculation of DH parameters will be presented .Moreover ,the transition matrix will be presented as well as the workspace will be shown.

1.1.1 DH Representation

Assigning the Coordinate Frames

The following steps were applied to set the coordinate of frames.

1. Assigning base frame (Link 0):
    - $\alpha_0=0$
    - $d_1=0$
    - $a_0=0$
2. Assigning frames and joints :
    - The axis of joint i and the axis of coordinate frame i are parallel, and the direction is defined by Right-handed rules. Coordinate frame i axis X_i corresponds with the common horizontal of joints i and i+1, as well as the direction is still from i to i+1. Correspondingly, axis Y_i could also be defined using Right-handed principles. The intersection point of X_i and Z_i is the origin . The location at which Z_i and Z_i+1 connect is known as the origin. When Z_i and Z_i+1 are orthogonal, the origin will cause $d_i = 0$, indicating that X_i and X_i+1 remain collinear.

3. Assigning End-effector frame :

- The joint of end-effector is a revolute joint, $\vartheta_n = 0$, X_n parallel with X_{n-1} ,therefore $d_n = 0$.

According to the steps above , the Lynxmotion arm model could be generated as the graph below (Figure1.1).

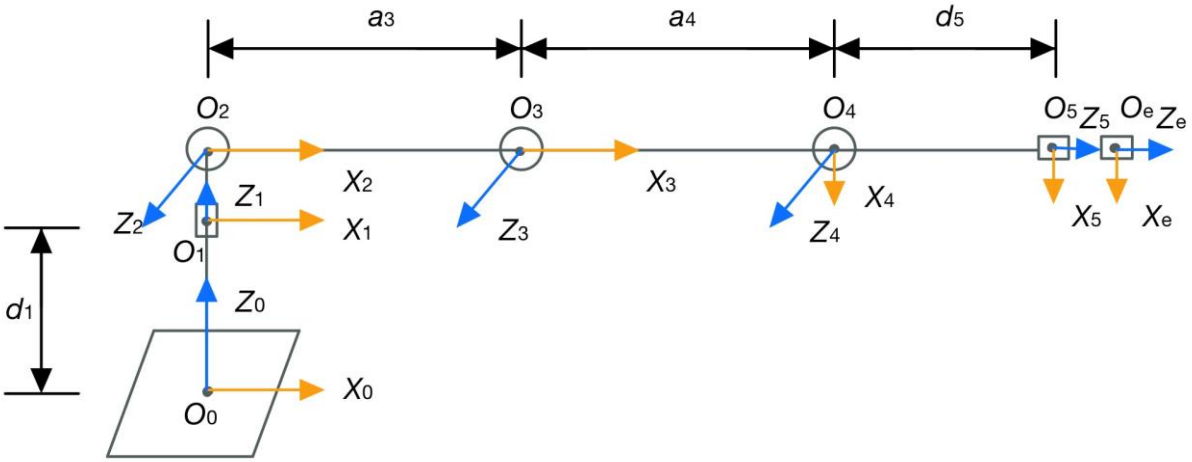


Figure 1.1: Modified D-H Convention Model

D-H Parameters

After setting the coordinates , D-H parameters could be figure out as shown in Table 1.1 below , where we define all the link lengths are 0.1m .Moreover,the description of parameter are stated below.

- a_i:distance along X_{n-1},Z_{n-1} to Z_n
- α_i:rotation about X_{n-1},Z_{n-1} to Z_n
- d_i:distance along Z_n,X_{n-1} to X_n
- ϑ_i:rotation about Z_n,X_{n-1} to X_n

	α_{i-1}	a_{i-1}	ϑ_i	d_i
Link1	0	0	ϑ_1	d_1
Link2	90	0	ϑ_2	0
Link3	0	a_3	ϑ_3	0
Link4	0	a_4	ϑ_4	0
Link5	−90	0	ϑ_5	d_5

Table 1.1: D-H Parameters

D-H Matrix

By utilize the equation (1.1) and equation (1.2) below , the transition matrix from link i to link i+1 were calculated

$$T_i = R(x_{i-1},\alpha_{i-1}) * T(x_{i-1},a_{i-1}) * R(z_i,\vartheta_i) * T(z_i,d_i) \quad (1.1)$$

$$^{i-1}_{i}T(1.2) = \begin{vmatrix} cos\vartheta_i & -sin\vartheta_i & 0 & a_{i-1} \\ sin\vartheta_i cos\alpha_{i-1} & cos\vartheta_i cos\alpha_{i-1} & -sin\alpha_{i-1} & -d_i sin\alpha_{i-1} \\ sin\vartheta_i sin\alpha_{i-1} & cos\vartheta_i sin\alpha_{i-1} & cos\alpha_{i-1} & d_i cos\alpha_{i-1} \\ 0 & 0 & 0 & 1 \end{vmatrix}$$

Then by calculate all $^{i-1}_{i}T$, the position of end-effector could be generated as

$$T = {}^{0}_{5}T = {}^{0}_{1}T * {}^{1}_{2}T * {}^{2}_{3}T * {}^{3}_{4}T * {}^{4}_{5}T \quad (1.3)$$

Matlab simulation

Four groups of joint angles were utilized in the Matlab simulation are shown in the Table1.2 below.

By apply the parameters above , the Lynxmotion AL5B arm under four different position animation can be generated as the Figure1.2 below.The corresponding MATLAB file is *FKFIVEDOF.m*

.

*q*1	*q*2	*q*3	*q*4
55	-30	45	-60
65	-35	50	-55
75	-40	55	-50
85	-45	60	-45

Table 1.2: Joint angles (degree)

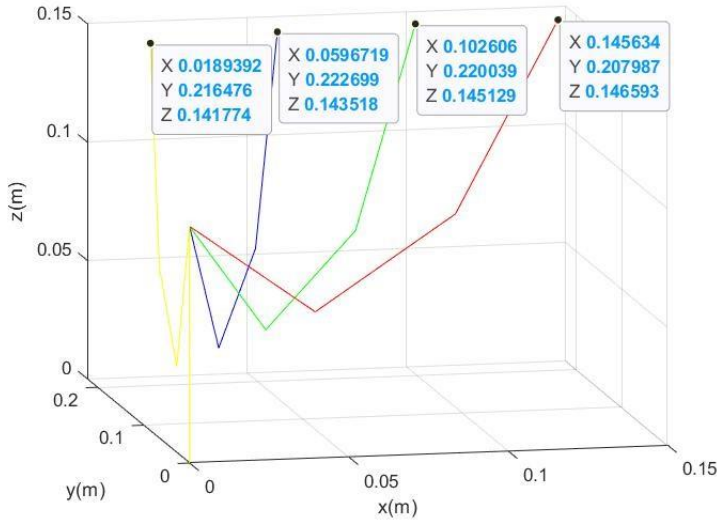


Figure 1.2: Lynxmotion AL5B arm under four conditions

1.1.2 Work space

The workspace is the area that the robot's end-effector can reach via at least single method. The attainable workspace was depicted as a sketch, where the robot could reach by many more than a method.

Since the angle of the end-effector is a fixed revlote joint which would not influence the position of the end-effector ,the angle of the first 4 joints are considered as a range shown in the Table1.3 below.

By calculate forward kinematic of one set of 4 random numbers in the range of angles for 10000 times into the *FKFIVEDOF.m* program as well as print the end-effector position at each time, the end-effector 3D and 2D workspace can be generated as shown in the Figure1.3 1.6 below.The

Joints	Angle ranges(degree)
q1	(0,180)
q2	(0,180)
q3	(0,120)
q4	(-180,0)

Table 1.3: Joint angles variation range (degree)

rationale is that since the number of the sample is large enough , the end-effector position should be evenly distributed in the entire workspace which perfectly described the workspace range.

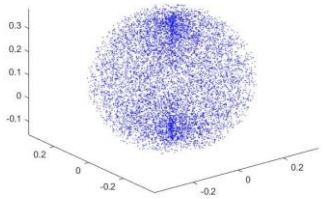


Figure 1.3: 3D workspace

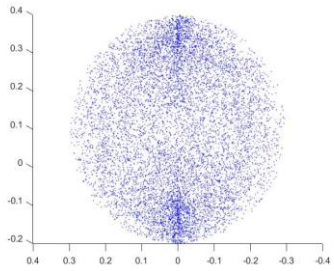
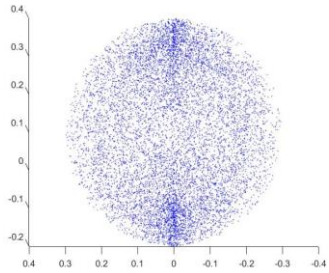


Figure 1.4: 2D workspace in x direction



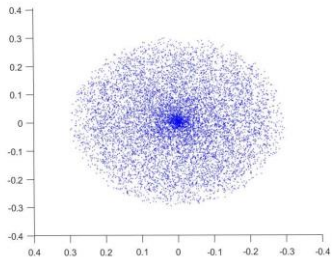


Figure 1.5: 2D workspace in y direction Figure 1.6: 2D workspace in z direction

1.2 Inverse Kinematics

Inverse kinematics is the use of kinematic equations to compute the motion of a robot to reach a certain position. Generally, we have two solutions in inverse kinematics which are Numerical Solution and Analytical Solution. The corresponding MATLAB file is *IK*11*.m* which will mainly introduce the Analytical Solution.

1.2.1 Analytical Approach

We have known informations including position of the end-effector and orientation which is X, Y, Z and phy.

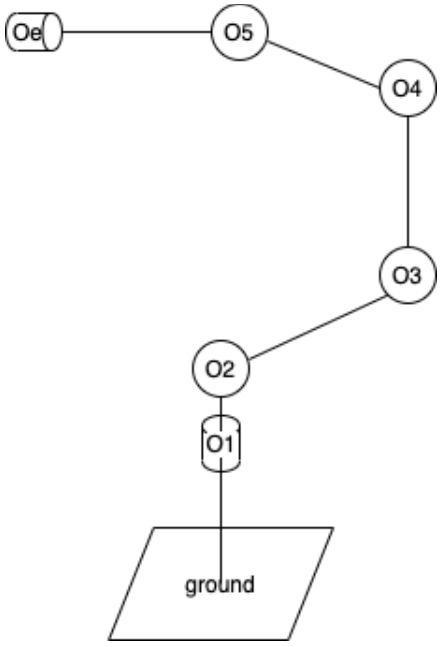


Figure 1.7: Simplified Lynxmotion Arm

Using algebra knowledge, we can get the solution of ϑ step by step.The calculation will base on the figure below.

At first, because the bottom has the height of 0.1m ,our parameter Z should delete 0.1 m.

$$Q(1,1)= atan2(Y,X); \tag{1.4}$$

$$Q(2,1)= atan2(Y,X); \tag{1.5}$$

where Q(i,1) is the rotation angle in vertical direction of orientation and i = 1:2.

Then using kinematics analysis, the coordinate of the end-effector(x, y, z) can be found:

$$x = X - 0.1 * cos(phy) * cos(Q(1,1)) \tag{1.6}$$

$$y = Y - 0.1 * cos(phy) * sin(Q(1,1)) \tag{1.7}$$

$$z = Z - 0.1 * sin(phy) \tag{1.8}$$

Using the law of cosines we will get the Q(1,3) and Q(2,3) which represent the angle of the a4 and d5. In fact, the representation of the angle is unique. However, when using root sign there will be two solutions which contains + or - . in this case, it is possible to have two but no more than two solutions when giving a specific orientation.

$$Q(1,3)= -acos((x^2+y^2+z^2)/(2*0.1^2)-1) \tag{1.9}$$

$$Q(2,3)= acos((x^2+y^2+z^2)/(2*0.1^2)-1) \tag{1.10}$$

In addition,

$$gama = acos((\sqrt{x^2+y^2+z^2})/(2*0.1)) \tag{1.11}$$

Finally, we can get the Q(1,2), Q(2,2) and Q(1,4), Q(2,4).

$$Q(1,2)= atan2(z,\sqrt{x^2+y^2})+gama \tag{1.12}$$

$$Q(2,2)= atan2(z,\sqrt{x^2+y^2})-gama \tag{1.13}$$

$$Q(1,4)= phy-Q(1,2)-Q(1,3)- pi/2 \tag{1.14}$$

$$Q(2,4)= phy-Q(2,2)-Q(2,3)- pi/2 \tag{1.15}$$

1.3 Trajectory and task planning

1.3.1 Task planning

Five positions for the task of the Lynxmotion AL5B arm are set as the Table1.4 below

Positions	x(m)	y(m)	z(m)	ψ(degree)
1	0.18	0.12	0.08	0
2	0.195	0.1325	0.095	0
3	0.21	0.145	0.11	0
4	0.195	0.1575	0.095	0
5	0.18	0.17	0.08	0

Table 1.4: position of task points

Then utilize the inverse kinematic program *partI$_{B1t}$ask$_p$lan.m* , five positions with the animation of Lynxmotion AL5B arm are shown in the Figure1.8 1.12 below

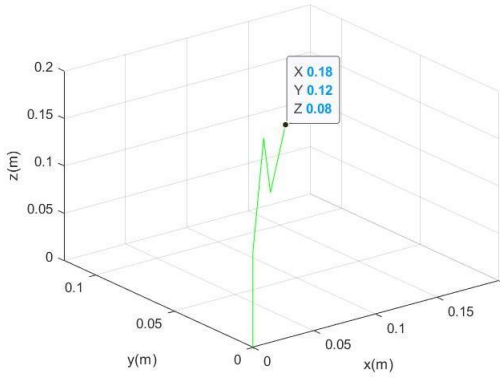


Figure 1.8: position1

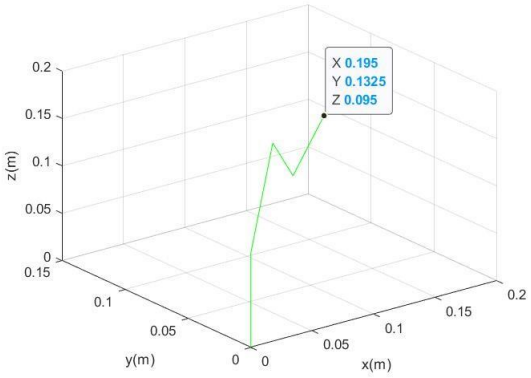


Figure 1.9: position2

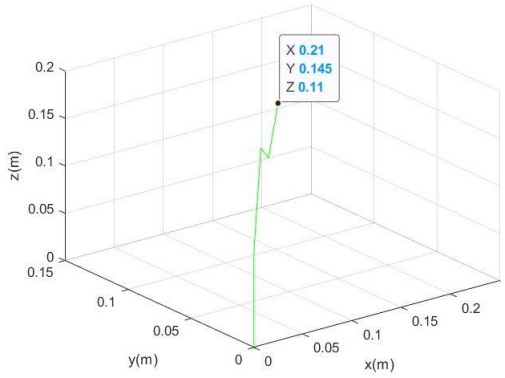


Figure 1.10: position3

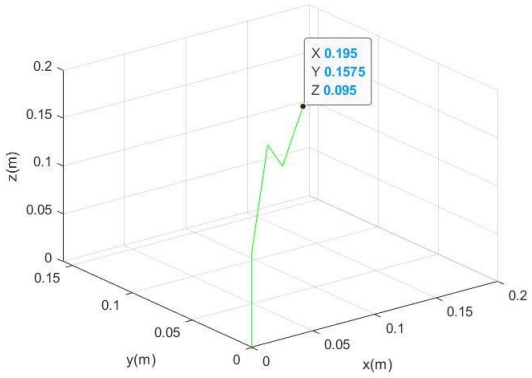


Figure 1.11: position4

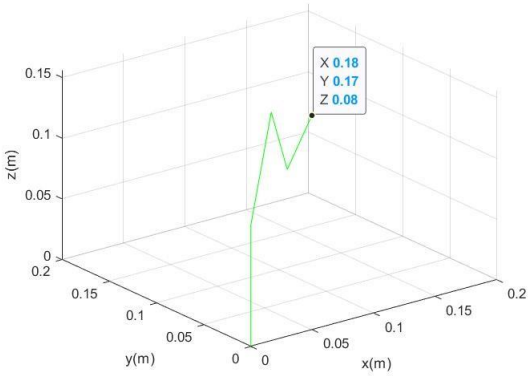


Figure 1.12: position5

1.3.2 Trajectories Implementation

Free motion

The free motion of Lynxmotion AL5B arm animation between 5 points is shown in the Figure1.13 below .The implementation of the free motion is to utilize intermediate points between each two set positions as well as let Lynxmotion AL5B arm pass through all the intermediate points as well as set positions.The corresponding program is *partIB*$_{3}$*ree*$_{m}$*otion.m*

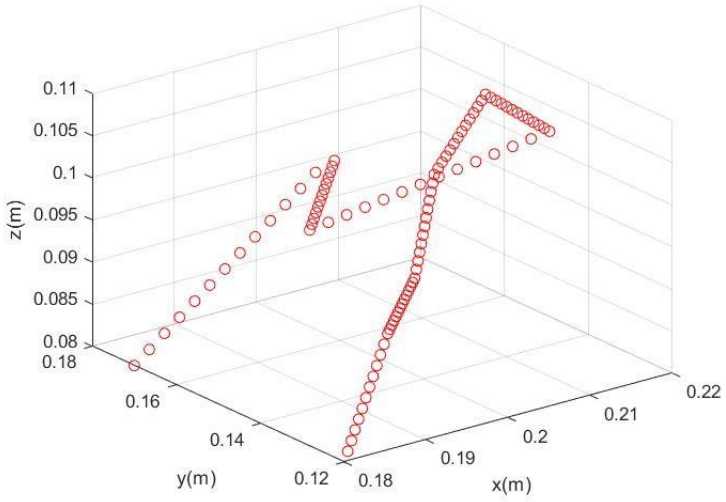


Figure 1.13: Free motion

Straight line trajectory

The straight line trajectory (Figure1.14) is implemented by add two straight line motion between the first 3 points as well as between the last 3 points .To add the straight line motion ,piecewise function had been applied for linear motion of x y and z coordinates.The corresponding program is *partI_B3straightline.m*

Object avoidance trajectory

To achieve the object avoidance trajectory (Figure1.15) ,firstly a ball was added between the position4 and position5. Secondly to avoid the obstacle , two more piecewise functions were ap-

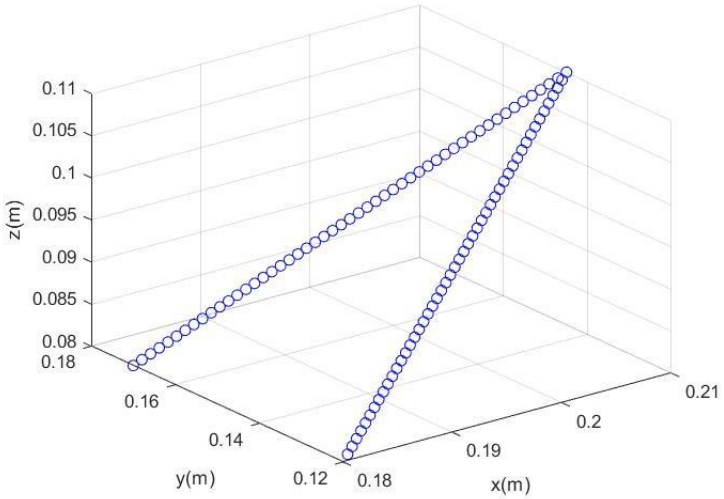


Figure 1.14: straight line trajectory

plied to achieve alternative approach from position4 to the position5.The corresponding program is *partIB3avoidence.m*

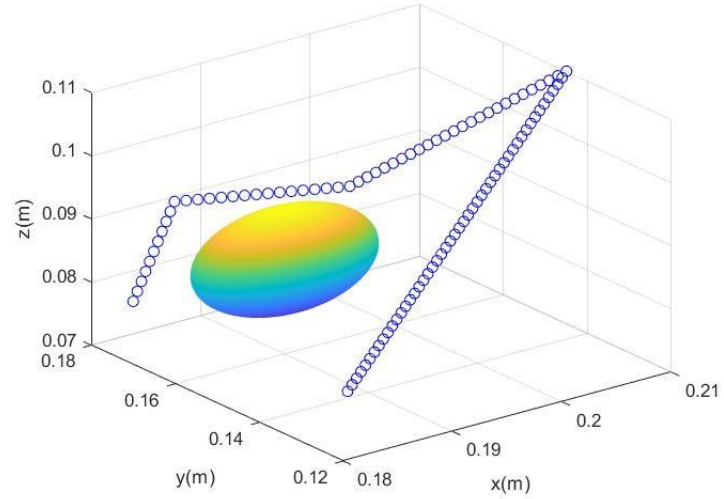


Figure 1.15: object avoidance trajectory

2 Kinematic simulation of Planar Parallel Robot

This part we are focusing on the kinematics simulation of planar parallel robot.

2.1 Inverse Kinematics of Parallel Robot

The parallel robot is made up of a permanent Base Plate, a Moving Platform, six passive revolute joints (M;and PP where i=1:3), and three active revolute joints (PB; where i=1:3). The i th strut of the robot is made up of three joints divided into two sections: upper portion sA and lower portion L. Figure 3.1 depicts the robot's kinematic model in further detail. The related matlab code is shown in *PartII*−1*.m* file.

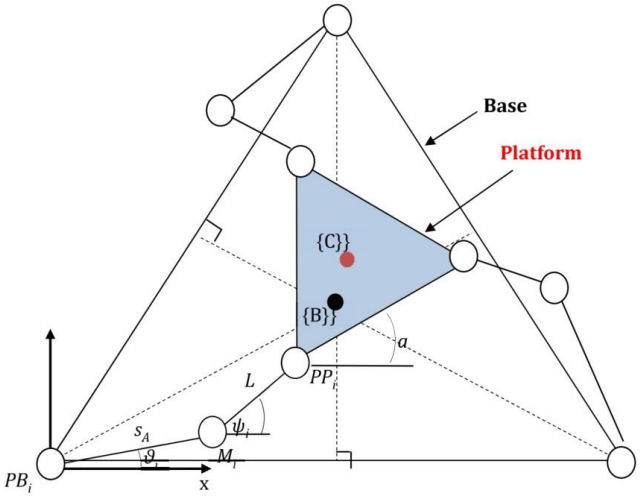


Figure 2.1: Planar Parallel Robot Kinematic Model

The calculation process are shown below:

1. Specify design parameters: Design parameters are shown in Table 2.2.

Parameter-Data Geometry	Value
ra(1,2,3)	170mm
L	130mm
Rplat(joint circle radius)	130mm
Rbase(joint circle radius)	290mm

Table 2.1: Design parameters of planar parallel robot.

2. Specify coordinates(xi, yi) of joints PBi and PPi: The angular joint positions are shown below.

PB1	pi/2
PB2	pi+pi/6
PB3	2pi-pi/6
PP1	pi/2
PP2	pi+pi/6
PP3	2pi-pi/6

Table 2.2: Design parameters of planar parallel robot.

3. Calculate the vector PB to PP : The angular joint positions are shown below.

$$P(1,i) = -R_{plat} * cos((30 + a + 120 * (i-1)) * (pi()/180)) + X_c \quad (2.1)$$

$$P(2,i) = -R_{plat} * sin((30 + a + 120 * (i-1)) * (pi()/180)) + Y_c \quad (2.2)$$

$$B(1,i) = -R_{base} * cos((210 + 120 * (i-1)) * (pi()/180)) \quad (2.3)$$

$$B(2,i) = -R_{base} * sin((210 + 120 * (i-1)) * (pi()/180)) \quad (2.4)$$

where:

- $i = 1 : 3$
- The joint circle radius of the base: $R_{base} = 290$
- The joint circle radius of the platform: $R_{plat} = 130$
- The orientation of the platform a is calculated by alpha which is an input parameter: $a = alpha * pi/180$
- The Cartesian coordinate of the centre of the platform : (X_c, Y_c)

After calculate the Positions, using the above equations we can calculate:

$$PB2PP(1,i) = B(1,i) + P(1,i) \tag{2.5}$$

$$PB2PP(2,i) = B(2,i) + P(2,i) \tag{2.6}$$

4. Set intermediate parameters: Then defining three intermediate parameters to simplify the complexity of the equation:

$$e1(i) = -2 * PB2PP(2,i) * ra(i) \tag{2.7}$$

$$e2(i) = -2 * PB2PP(1,i) * ra(i) \tag{2.8}$$

$$e3(i) = (PB2PP(1,i))^2 + (PB2PP(2,i))^2 + ra(i)^2 - L^2 \tag{2.9}$$

suppose

$$\vartheta = (30 + a + 120 * (i-1)) * (pi()/180) \tag{2.10}$$

using the equations before(2.5, 2.6, 2.7, 2.8, 2.9), we can get the equation :

$$e_1 sin\vartheta + e_2 cos\vartheta + e3 = 0 \tag{2.11}$$

defining: $t = tan(\vartheta/2)$

Then we can get:

$$sin\vartheta = 2t/(1+t^2) \tag{2.12}$$

$$cos\vartheta = (1-t^2)/(1+t^2) \tag{2.13}$$

combining formula (2.12, 2.13) and formula (2.11), we can get:

$$(e_3 - e_2)t^2 + 2e_1 t + e_2 + e_3 = 0 \tag{2.14}$$

the solution of this equation are:

$$t^{1,2} = (-e_1 \pm \sqrt{e_1^2 + e_1^2 - e_1^2})/(e_3 - e_2) \tag{2.15}$$

$$\vartheta = 2atan(t_{1,2}) \tag{2.16}$$

5. Calculate the joint positions :The value of ϑ can be calculated by the equations above.

Then Calculate the joint positions.

$$J(1,i) = -B(1,i) + ra(i) * cos(\vartheta(i)) \tag{2.17}$$

$$J(2,i) = -B(2,i) + ra(i) * sin(\vartheta(i)) \tag{2.18}$$

where $i = 1 : 3$

Then all the parameters needed to simulate are collected. Details are in the matlab *PartII* − 1*.m* file code. Using the code to implementing the process. Choosing four positions randomly, the solutions are shown below in figure(2.2, 2.3).

2.2 Parallel robots workspace for a given orientation.

The workspace is a collection of different end-effector configurations for the robot and has nothing to do with any particular assignment. For this parallel robot, defining the interval of PP_i and PB_i with *limit*. Using this limit, we can make (X_c, Y_c) travel around in the interval where the workspace contains in, and using the loop to find every possible point which will be combined as workspace.

The corresponding matlab code is shown in *PartII*−1*.m* file.

The simulation using MATLAB is shown below:

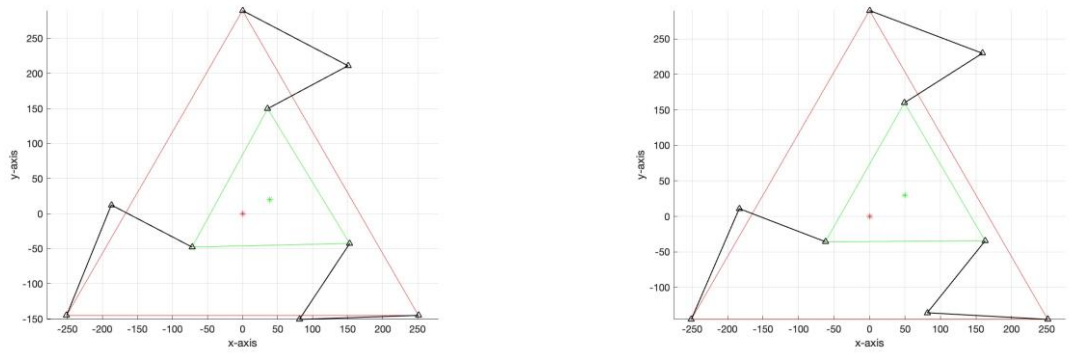


Figure 2.2: X(mm)-Y(mm)-alpha(degree): 39-20-76 (left) and 50-30-18 (right)

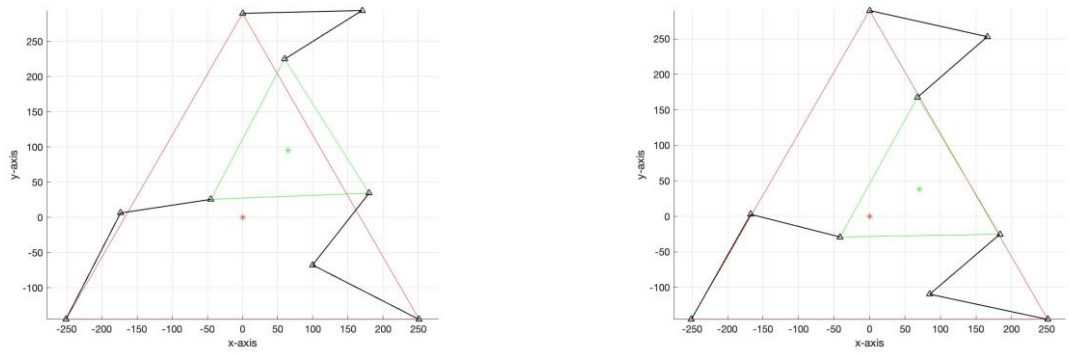


Figure 2.3: X(mm)-Y(mm)-alpha(degree): 65-95-130 (left) and 70-38-57 (right)

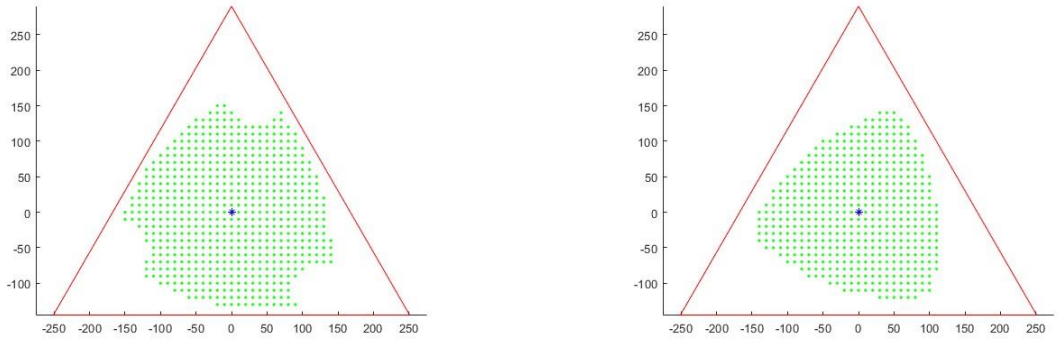


Figure 2.4: alpha(degree of orientation): 15(left) and 30(right)

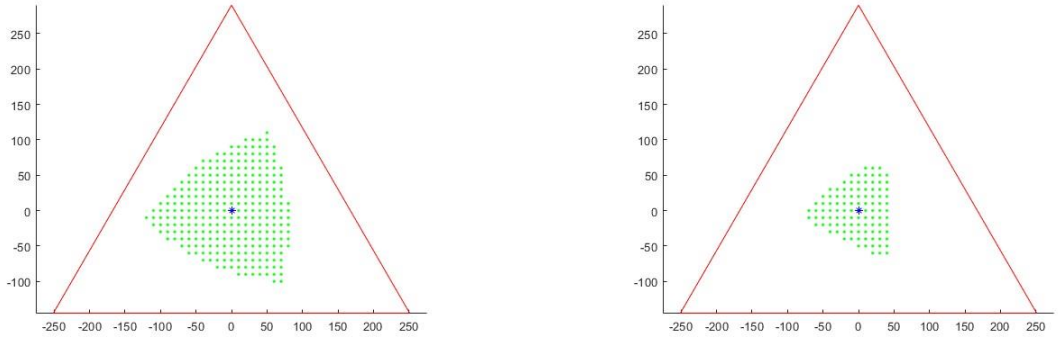


Figure 2.5: alpha(degree of orientation): 45 (left) and 60 (right)

3 Further obstacles avoidance research

To achieve objects avoidance trajectory with more obstacles (Figure3.1) ,firstly one more ball was added between the position1 and position2 and the other additional ball was added between

the position2 and position3. Secondly to avoid the additional obstacles , two more sets of piecewise functions were applied to achieve alternative approach from position1 to the position2 then to position 3.The corresponding program is *part3.m*

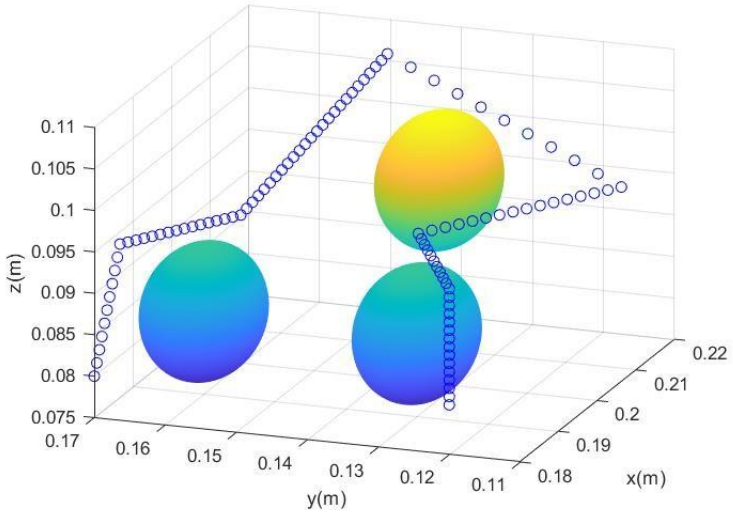


Figure 3.1: obstacles avoidance with more objects

4 Conclusion and discussion

The study findings are divided into three sections, each of which incorporates essential ideas in simulation-based robotic kinematics. We begin with the Denavit Hartenberg convention and use the Denavit Hartenberg transformation to combine Lynxmotion's joint space with Cartesian space and finally generate the workspace. The following stage constructs the robot's end-trajectory effectors using three distinct strategies: polynomial fitting free motion, linear segments, and obstacle avoidance.

Second, parallel robots can be thought of as a collection of serial robots that collaborate to perform a task by sharing an end-effector. Thus, the reverse kinematics of parallel robots can be broken into a few fundamental serial roots. Additionally, when charting a parallel robot's workspace, extra constraints can be added to determine whether or not a point is contained inside the workspace bounds. For instance, we can specify a state that the length of the second arm is 130 inches. If this is the case, the point is positioned within the workplace; otherwise, it is located outside.

Finally, in real applications, our robot will face a more complicated situation. So we design some more obstacles to achieve a further obstacle-avoiding task. we implement an essential principle of robot motion called path planning algorithm in a simulation-based environment. The mobile robot follows the path planed before to avoid the obstacle and tries to be smooth the path to a greater
extent.

In addition ,the free motion could be achieved by make each angle chaning gradually from the previous position angle to the next position angle .Besides ,the free motion could also achieved by three dimensional quadratic system .However due to the limitation of hardware , the three dimensional quadratic system could not achieve in this project,but by limit the motion in 2D plane which means fix x or y or z coordinate of the end-effector ,the curvature motion could be achieved .

Appendix：

FKFIVEDOF

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%% 2022 Jan %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%% WENTAO GAO &&& JINGZHI WU %%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% initialization %%%

clear all close all clc disp('The following code simulates the forward
kinematics of a simple 2DOF') disp('serial manipulator. Figure 2 shows its
movement from start to end position,') disp('Figure 1 shows the location of
its end effector at points of its trajectory') disp('and Figure 3 shows the
maximum potential workspace of its end effector')

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% A series of joint angles %%%

q1 = [55 65 75 85]*sym(pi)/180; q2
= [-30 -35 -40 -45]*sym(pi)/180; q3
= [45 50 55 60]*sym(pi)/180; q4 =
[-60 -55 -50 -45]*sym(pi)/180;

q1_range = [0,180]*sym(pi)/180;
q2_range = [0,180]*sym(pi)/180;
q3_range = [0,120]*sym(pi)/180;
q4_range = [-180,0]*sym(pi)/180;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Links Lengths %%%

l1 = 0.1 ;
l2 = 0 ;
l3 = 0.1 ;
l4 = 0.1 ;
l5 = 0.1 ;
alpha =
[0 ,
sym(pi)/2
, 0 , 0 ,
-
```

```
sym(pi)/2
]; a = [0,
0 , l3 ,
l4 , 0];
theta =
[0 ,0 ,
0 ,0 , 0];
d = [l1 ,
0 , 0 ,
0 , l5];

for i = 1:4

theta(1,1) = q1(1,i);
theta(1,2) = q2(1,i);
theta(1,3) = q3(1,i);
theta(1,4) = q4(1,i);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Trigonometric abbreviations %%%
c1 = cos(theta(1,1)); c2 =
cos(theta(1,2)); c3 =
cos(theta(1,3)); c4 =
cos(theta(1,4)); c5 =
cos(theta(1,5)); c12 = cos(q1+q2);

ca1 = cos(alpha(1));
ca2 = cos(alpha(2));
ca3 = cos(alpha(3));
ca4 = cos(alpha(4));
ca5 = cos(alpha(5));

s1 = sin(theta(1,1));
s2 = sin(theta(1,2));
s3 = sin(theta(1,3));
s4 = sin(theta(1,4));
s5 = sin(theta(1,5));
s12 = sin(q1+q2);

sa1 = sin(alpha(1));
sa2 = sin(alpha(2));
```

```
sa3 = sin(alpha(3));
sa4 = sin(alpha(4));
sa5 = sin(alpha(5));


T1 = [ [ c1, -s1, 0, a(1)]
[ ca1*s1, ca1*c1, -sa1, -d(1)*sa1]
[ sa1*s1, sa1*c1, ca1, d(1)*ca1]
[ 0, 0, 0, 1] ];
T12 = [ [ c2, -s2, 0, a(2)]
[ ca2*s2, ca2*c2, -sa2, -d(2)*sa2]
[ sa2*s2, sa2*c2, ca2, d(2)*ca2]
[ 0, 0, 0, 1] ];
T23 = [ [ c3, -s3, 0, a(3)]
[ ca3*s3, ca3*c3, -sa3, -d(3)*sa3]
[ sa3*s3, sa3*c3, ca3, d(3)*ca3]
[ 0, 0, 0, 1] ];
T34 = [ [ c4, -s4, 0, a(4)]
[ ca4*s4, ca4*c4, -sa4, -d(4)*sa4]
[ sa4*s4, sa4*c4, ca4, d(4)*ca4]
[ 0, 0, 0, 1] ];
T45 = [ [ c5, -s5, 0, a(5)]
[ ca5*s5, ca5*c5, -sa5, -d(5)*sa5]
[ sa5*s5, sa5*c5, ca5, d(5)*ca5]
[ 0, 0, 0, 1] ];

T01 = T1;
T02 = T01*T12;
T03 = T02*T23;
T04 = T03*T34;
T05 = T04*T45;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Tip position %%% frames_x = [0, T01(1,4), T02(1,4), T03(1,4),


T04(1,4), T05(1,4)]'; % x coordinate of frames
```

```
frames_y = [0, T01(2,4), T02(2,4), T03(2,4), T04(2,4), T05(2,4)]'; % y
coordinate of frames frames_z = [0, T01(3,4), T02(3,4), T03(3,4),
T04(3,4), T05(3,4)]'; % z coordinate of frames



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Plot the robotic arm, in 4 different positions %%%
figure (1)
Color = ['r','g','b','y'];

plot3(frames_x ,frames_y ,frames_z,Color(1,i));

xlabel('x(m)');
ylabel('y(m)');
zlabel('z(m)');
grid on; hold
on; end



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Workspace %%%

figure (3)

for j = 1:10000 c1 =
cos(theta(1,1)); c2 =
cos(theta(1,2)); c3 =
cos(theta(1,3)); c4 =
cos(theta(1,4)); c5 =
cos(theta(1,5)); c12
= cos(q1+q2);

ca1 = cos(alpha(1));
ca2 = cos(alpha(2));
ca3 = cos(alpha(3));
ca4 = cos(alpha(4));
ca5 = cos(alpha(5));

s1 = sin(theta(1,1));
s2 = sin(theta(1,2));
```

```
s3 = sin(theta(1,3));
s4 = sin(theta(1,4));
s5 = sin(theta(1,5));
s12 = sin(q1+q2);

sa1 = sin(alpha(1));
sa2 = sin(alpha(2));
sa3 = sin(alpha(3));
sa4 = sin(alpha(4));
sa5 = sin(alpha(5));

theta(1,1) = 0 + 2*pi*rand;
theta(1,2) = 0 + 2*pi*rand;
theta(1,3) = 0+ 2/3*pi*rand;
theta(1,4) = -pi+ pi*rand;
theta(1,5) = 0;

T1 = [ [ c1, -s1, 0, a(1)]
[ ca1*s1, ca1*c1, -sa1, -d(1)*sa1]
[ sa1*s1, sa1*c1, ca1, d(1)*ca1]
[ 0, 0, 0, 1] ];
T12 = [ [ c2, -s2, 0, a(2)]
[ ca2*s2, ca2*c2, -sa2, -d(2)*sa2]
[ sa2*s2, sa2*c2, ca2, d(2)*ca2]
[ 0, 0, 0, 1] ];
T23 = [ [ c3, -s3, 0, a(3)]
[ ca3*s3, ca3*c3, -sa3, -d(3)*sa3]
[ sa3*s3, sa3*c3, ca3, d(3)*ca3]
[ 0, 0, 0, 1] ];
T34 = [ [ c4, -s4, 0, a(4)]
[ ca4*s4, ca4*c4, -sa4, -d(4)*sa4]
[ sa4*s4, sa4*c4, ca4, d(4)*ca4]
[ 0, 0, 0, 1] ];
T45 = [ [ c5, -s5, 0, a(5)]
[ ca5*s5, ca5*c5, -sa5, -d(5)*sa5]
[ sa5*s5, sa5*c5, ca5, d(5)*ca5]
[ 0, 0, 0, 1] ];

T050 = T1*T12*T23*T34*T45;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Plot the workspace of the robot %%%
```

```
X = T050(1,4);
Y = T050(2,4); Z = T050(3,4);
plot3(X,Y,Z,'b.','MarkerSize',1);
hold on;

end
```

IK11

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%% 2022
Jan %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% %%%%%%%%%%%%%%%%%%%% WENTAO GAO
&&& JINGZHI WU %%%%%%%%%%%%%%%%%%%%%%%%%%%% function Q = IK11(X,Y,Z,phy)

Z = Z-0.1 ; %0.1 is the height of the first section

%%% calculate Q1 %%%
Q(1,1)= atan2(Y,X);
Q(2,1)= atan2(Y,X);

x = X-0.1*cos(phy)*cos(Q(1,1));
y = Y-0.1*cos(phy)*sin(Q(1,1));
z = Z - 0.1*sin(phy);

%%% calculate Q3 and gama %%%
Q(1,3) = -acos((x^2+y^2+z^2)/(2*0.1^2)-1);
Q(2,3) = acos((x^2+y^2+z^2)/(2*0.1^2)-1);

gama = acos((sqrt(x^2+y^2+z^2))/(2*0.1));

%%% calculate Q2 by using gama and Q3 %%%
Q(1,2) = atan2(z,sqrt(x^2+y^2))+gama;
Q(2,2) = atan2(z,sqrt(x^2+y^2))-gama;

%%% calculate Q4 by using phy , Q2 and Q3 %%%
Q(1,4) = phy - Q(1,2) - Q(1,3)-pi/2;
Q(2,4) = phy - Q(2,2) - Q(2,3)-pi/2;

%%% prtint two solution %%%
disp('1st solution')
disp([ Q(1,1) Q(1,2) Q(1,3) Q(1,4) ]*180/pi)

disp('2nd solution')
```

```
disp([ Q(2,1) Q(2,2) Q(2,3) Q(2,4) ]*180/pi)
```

partI_B_1_task_plan

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%% 2022 Jan %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%% WENTAO GAO &&& JINGZHI WU %%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% initialization %%%
clear

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% parameters %%%

l1 = 0.1 ;
l2 = 0 ;
l3 = 0.1 ;
l4 = 0.1 ;
l5 = 0.1 ;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% DH table %%%

theta = [0 ,0 , 0 ,0 , 0];
d = [l1 , 0 , 0 , 0 , l5];
alpha = [0 , sym(pi)/2 , 0 , 0 , -sym(pi)/2 ];
a = [0, 0 , l3 , l4 , 0];


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Position planning %%%


%position1
Q = IK11(0.18,0.12,0.08,0);

%position2
%Q = IK11(0.195,0.1325,0.095,0);

%position3
%Q = IK11(0.21,0.145,0.11,0);
```

```
%position4
%Q = IK11(0.195,0.1575,0.095,0);

%position5
%Q = IK11(0.18,0.17,0.08,0);

theta(1) = Q(1,1);
theta(2) = Q(1,2);
theta(3) = Q(1,3);
theta(4) =
Q(1,4); %%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%
%%%%
%%% get essentical sin and cos value %%%

c1 = cos(theta(1));
c2 = cos(theta(2));
c3 = cos(theta(3));
c4 = cos(theta(4));
c5 = cos(theta(5));

ca1 = cos(alpha(1));
ca2 = cos(alpha(2));
ca3 = cos(alpha(3));
ca4 = cos(alpha(4));
ca5 = cos(alpha(5));

s1 = sin(theta(1));
s2 = sin(theta(2));
s3 = sin(theta(3));
s4 = sin(theta(4));
s5 = sin(theta(5));


sa1 = sin(alpha(1));
sa2 = sin(alpha(2));
sa3 = sin(alpha(3));
sa4 = sin(alpha(4));
sa5 = sin(alpha(5));
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% get the transfor matrix of each part of Ly- Arm %%%

T1 = [ [ c1, -s1, 0, a(1)]
[ ca1*s1, ca1*c1, -sa1, -d(1)*sa1]
[ sa1*s1, sa1*c1, ca1, d(1)*ca1]
[ 0, 0, 0, 1] ];
T12 = [ [ c2, -s2, 0, a(2)]
[ ca2*s2, ca2*c2, -sa2, -d(2)*sa2]
[ sa2*s2, sa2*c2, ca2, d(2)*ca2]
[ 0, 0, 0, 1] ];
T23 = [ [ c3, -s3, 0, a(3)]
[ ca3*s3, ca3*c3, -sa3, -d(3)*sa3]
[ sa3*s3, sa3*c3, ca3, d(3)*ca3]
[ 0, 0, 0, 1] ];
T34 = [ [ c4, -s4, 0, a(4)]
[ ca4*s4, ca4*c4, -sa4, -d(4)*sa4]
[ sa4*s4, sa4*c4, ca4, d(4)*ca4]
[ 0, 0, 0, 1] ];
T45 = [ [ c5, -s5, 0, a(5)]
[ ca5*s5, ca5*c5, -sa5, -d(5)*sa5]
[ sa5*s5, sa5*c5, ca5, d(5)*ca5]
[ 0, 0, 0, 1] ];

T010 = T1;
T020 = T010*T12;
T030 = T020*T23;
T040 = T030*T34;
T050 = T040*T45;


frames_x = [0, T010(1,4), T020(1,4), T030(1,4), T040(1,4), T050(1,4)]';
frames_y = [0, T010(2,4), T020(2,4), T030(2,4), T040(2,4), T050(2,4)]';
frames_z = [0, T010(3,4), T020(3,4), T030(3,4), T040(3,4), T050(3,4)]';

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% plot the position %%%

plot3(frames_x ,frames_y ,frames_z,'g');

xlabel('x(m)');
ylabel('y(m)');
```

```
zlabel('z(m)');
grid on; hold
on;
```

partI_B_3_straight_line

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%% 2022 Jan %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%% WENTAO GAO &&& JINGZHI WU %%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% initialization %%%
clear

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% parameters %%%
l1 = 0.1 ;
l2 = 0 ;
l3 = 0.1 ;
l4 = 0.1 ;
l5 = 0.1 ;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% DH table %%%

theta = [0 ,0 , 0 ,0 , 0];
d = [l1 , 0 , 0 , 0 , l5];
alpha = [0 , sym(pi)/2 , 0 , 0 , -sym(pi)/2 ];
a = [0, 0 , l3 , l4 , 0];



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Stright path to 5 points planning %%%

for i = 1:100

if i<=50

X = 0.18+0.0006*i;
Y = 0.12+0.0005*i; Z = 0.08+0.0006*i;
```

```
end

if i>50
X = 0.21-0.0006*(i-50);
Y = 0.145+0.0005*(i-50); Z = 0.11-0.0006*(i-
  50); end

Q = IK11(X,Y,Z,0);


theta(1) = Q(1,1);
theta(2) = Q(1,2);
theta(3) = Q(1,3);
theta(4) =
Q(1,4); %%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%
%%% get essentical sin and cos value %%%

c1 = cos(theta(1));
c2 = cos(theta(2));
c3 = cos(theta(3));
c4 = cos(theta(4));
c5 = cos(theta(5));

ca1 = cos(alpha(1));
ca2 = cos(alpha(2));
ca3 = cos(alpha(3));
ca4 = cos(alpha(4));
ca5 = cos(alpha(5));

s1 = sin(theta(1));
s2 = sin(theta(2));
s3 = sin(theta(3));
s4 = sin(theta(4));
s5 = sin(theta(5));


sa1 = sin(alpha(1));
sa2 = sin(alpha(2));
sa3 = sin(alpha(3));
```

```
sa4 = sin(alpha(4));
sa5 = sin(alpha(5));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% get the transfor matrix of each part of Ly- Arm %%%

T1 = [ [ c1, -s1, 0, a(1)]
[ ca1*s1, ca1*c1, -sa1, -d(1)*sa1]
[ sa1*s1, sa1*c1, ca1, d(1)*ca1]
[ 0, 0, 0, 1] ];
T12 = [ [ c2, -s2, 0, a(2)]
[ ca2*s2, ca2*c2, -sa2, -d(2)*sa2]
[ sa2*s2, sa2*c2, ca2, d(2)*ca2]
[ 0, 0, 0, 1] ];
T23 = [ [ c3, -s3, 0, a(3)]
[ ca3*s3, ca3*c3, -sa3, -d(3)*sa3]
[ sa3*s3, sa3*c3, ca3, d(3)*ca3]
[ 0, 0, 0, 1] ];
T34 = [ [ c4, -s4, 0, a(4)]
[ ca4*s4, ca4*c4, -sa4, -d(4)*sa4]
[ sa4*s4, sa4*c4, ca4, d(4)*ca4]
[ 0, 0, 0, 1] ];
T45 = [ [ c5, -s5, 0, a(5)]
[ ca5*s5, ca5*c5, -sa5, -d(5)*sa5]
[ sa5*s5, sa5*c5, ca5, d(5)*ca5]
[ 0, 0, 0, 1] ];

T010 = T1;
T020 = T010*T12;
T030 = T020*T23;
T040 = T030*T34;
T050 = T040*T45;


frames_x = [0, T010(1,4), T020(1,4), T030(1,4), T040(1,4), T050(1,4)];
frames_y = [0, T010(2,4), T020(2,4), T030(2,4), T040(2,4), T050(2,4)];
frames_z = [0, T010(3,4), T020(3,4), T030(3,4), T040(3,4), T050(3,4)];


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% plot in matlab %%%
```

```
scatter3(frames_x(6),frames_y(6),frames_z(6),'b');
hold on;

arm=plot3(frames_x,frames_y,frames_z,'g');
xlabel('x(m)'); ylabel('y(m)');
zlabel('z(m)'); grid on; pause(0.0001);
delete(arm); end
```

partIB_3_avoidence

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%% 2022 Jan %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%% WENTAO GAO &&& JINGZHI WU %%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% initialization %%%
clear %%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% parameters %%%

l1 = 0.1 ;
l2 = 0 ;
l3 = 0.1 ;
l4 = 0.1 ;
l5 = 0.1 ;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% set up obstacle %%%

[xs,ys,zs]=sphere(900);

Xs=xs*0.008+0.19;
Ys=ys*0.008+0.16;
Zs=zs*0.008+0.085;

mesh(Xs,Ys,Zs);
hold on;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% DH table %%%
```

```
theta = [0 ,0 , 0 ,0 , 0];
d = [l1 , 0 , 0 , 0 , l5];
alpha = [0 , sym(pi)/2 , 0 , 0 , -sym(pi)/2 ];
a = [0, 0 , l3 , l4 , 0];


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Path planning avoid obstacles %%%

for i = 1:100

if i<=50

X = 0.18+0.0006*i;
Y = 0.12+0.0005*i; Z = 0.08+0.0006*i;

end

if (i>50)&&(i<=75) X =
0.21-0.0006*(i-50);
Y = 0.145+0.0005*(i-50);
Z = 0.11-0.0006*(i-50);
end

if (i>75)&&(i<=90) X =
0.195-0.0008*(i-75);
Y = 0.1575+0.0007*(i-75);
Z = 0.095; end if
(i>90)&&(i<=100) X =
0.183-0.0003*(i-90);
Y = 0.168+0.0002*(i-90);
Z = 0.095-0.0015*(i-90);
end


Q = IK11(X,Y,Z,0);


theta(1) = Q(1,1);
theta(2) = Q(1,2);
theta(3) = Q(1,3);
theta(4) = Q(1,4);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% get essentical sin and cos value %%%

c1 = cos(theta(1));
c2 = cos(theta(2));
c3 = cos(theta(3));
c4 = cos(theta(4));
c5 = cos(theta(5));

ca1 = cos(alpha(1));
ca2 = cos(alpha(2));
ca3 = cos(alpha(3));
ca4 = cos(alpha(4));
ca5 = cos(alpha(5));

s1 = sin(theta(1));
s2 = sin(theta(2));
s3 = sin(theta(3));
s4 = sin(theta(4));
s5 = sin(theta(5));


sa1 = sin(alpha(1));
sa2 = sin(alpha(2));
sa3 = sin(alpha(3));
sa4 = sin(alpha(4));
sa5 = sin(alpha(5));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% get the transfor matrix of each part of Ly- Arm %%%

T1 = [ [ c1, -s1, 0, a(1)]
[ ca1*s1, ca1*c1, -sa1, -d(1)*sa1]
[ sa1*s1, sa1*c1, ca1, d(1)*ca1]
[ 0, 0, 0, 1] ];
T12 = [ [ c2, -s2, 0, a(2)]
[ ca2*s2, ca2*c2, -sa2, -d(2)*sa2]
[ sa2*s2, sa2*c2, ca2, d(2)*ca2]
[ 0, 0, 0, 1] ];
T23 = [ [ c3, -s3, 0, a(3)]
[ ca3*s3, ca3*c3, -sa3, -d(3)*sa3]
[ sa3*s3, sa3*c3, ca3, d(3)*ca3]
```

```
[ 0, 0, 0, 1] ];
T34 = [ [ c4, -s4, 0, a(4)]
[ ca4*s4, ca4*c4, -sa4, -d(4)*sa4]
[ sa4*s4, sa4*c4, ca4, d(4)*ca4]
[ 0, 0, 0, 1] ];
T45 = [ [ c5, -s5, 0, a(5)]
[ ca5*s5, ca5*c5, -sa5, -d(5)*sa5]
[ sa5*s5, sa5*c5, ca5, d(5)*ca5]
[ 0, 0, 0, 1] ];

T010 = T1;
T020 = T010*T12;
T030 = T020*T23;
T040 = T030*T34;
T050 = T040*T45;


frames_x = [0, T010(1,4), T020(1,4), T030(1,4), T040(1,4), T050(1,4)];
frames_y = [0, T010(2,4), T020(2,4), T030(2,4), T040(2,4), T050(2,4)];
frames_z = [0, T010(3,4), T020(3,4), T030(3,4), T040(3,4), T050(3,4)];


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% plot in matlab %%%

scatter3(frames_x(6),frames_y(6),frames_z(6),'b');
hold on;

arm=plot3(frames_x,frames_y,frames_z,'g');
xlabel('x(m)'); ylabel('y(m)');
zlabel('z(m)'); grid on; pause(0.0001);
delete(arm); end
```

partIB_3_free_motion

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%% 2022 Jan %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%% WENTAO GAO &&& JINGZHI WU %%%%%%%%%%%%%%%%%%%%%%%%%%
%%% initialization %%%
clear
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% parameters %%%

l1 = 0.1 ;
l2 = 0 ;
l3 = 0.1 ;
l4 = 0.1 ;
l5 = 0.1 ;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% DH table %%%

theta = [0 ,0 , 0 ,0 , 0];
d = [l1 , 0 , 0 , 0 , l5];
alpha = [0 , sym(pi)/2 , 0 , 0 , -sym(pi)/2 ];
a = [0, 0 , l3 , l4 ,
0]; %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Path planning for free motion %%%


for i = 1:100

%start from position 1 to position 2
if i<=12 X = 0.18+0.0008*i;
Y = 0.12+0.0007*i;
Z = 0.08+0.0009*i;

end

if (i>12)&&(i<=25)
X = 0.1896+0.0004153846*(i-12);
Y = 0.1284+0.0003153846*(i-12); Z =
  0.0908+0.0003230769*(i-12);

end

%start from position 2 to position 3
if (i>25)&&(i<=37) X =
0.195+0.0003*(i-25);
Y = 0.1325+0.0002*(i-25);
Z = 0.095+0.0009*(i-25);
```

```
end

if (i>37)&&(i<=50)
X = 0.1986+0.0008769231*(i-37);
Y = 0.1349+0.0007769231*(i-37); Z =
  0.1058+0.0003230769*(i-37); end

%start from position 3 to position 4
if (i>50)&&(i<=62) X =
0.21+0.0006*(i-50);
Y = 0.145-0.0001*(i-50);
Z = 0.11-0.0005*(i-50);
end
if (i>62)&&(i<=75) X =
0.2172-0.001708*(i-62);
Y = 0.1438+0.001054*(i-62); Z
= 0.104-0.0006923077*(i-62);
end

%start from position 4 to position 5
if (i>75)&&(i<=87) X =
0.195+0.0007*(i-75);
Y = 0.1575+0.00088*(i-75);
Z = 0.095+0.0003*(i-75);

end

if (i>87)&&(i<=100) X =
0.2034-0.0018*(i-87);
Y = 0.16894+0.000081538*(i-87);
Z = 0.0989-0.0014538*(i-87);

end
%end at position 5

Q = IK11(X,Y,Z,0);


theta(1) = Q(1,1);
theta(2) = Q(1,2);
theta(3) = Q(1,3);
theta(4) = Q(1,4);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% get essentical sin and cos value %%%


c1 = cos(theta(1));
c2 = cos(theta(2));
c3 = cos(theta(3));
c4 = cos(theta(4));
c5 = cos(theta(5));

ca1 = cos(alpha(1));
ca2 = cos(alpha(2));
ca3 = cos(alpha(3));
ca4 = cos(alpha(4));
ca5 = cos(alpha(5));

s1 = sin(theta(1));
s2 = sin(theta(2));
s3 = sin(theta(3));
s4 = sin(theta(4));
s5 = sin(theta(5));


sa1 = sin(alpha(1));
sa2 = sin(alpha(2));
sa3 = sin(alpha(3));
sa4 = sin(alpha(4));
sa5 = sin(alpha(5));


T1 = [ [ c1, -s1, 0, a(1)]
[ ca1*s1, ca1*c1, -sa1, -d(1)*sa1]
[ sa1*s1, sa1*c1, ca1, d(1)*ca1]
[ 0, 0, 0, 1] ];
T12 = [ [ c2, -s2, 0, a(2)]
[ ca2*s2, ca2*c2, -sa2, -d(2)*sa2]
[ sa2*s2, sa2*c2, ca2, d(2)*ca2]
[ 0, 0, 0, 1] ];
T23 = [ [ c3, -s3, 0, a(3)]
[ ca3*s3, ca3*c3, -sa3, -d(3)*sa3]
[ sa3*s3, sa3*c3, ca3, d(3)*ca3]
[ 0, 0, 0, 1] ];
```

```
T34 = [ [ c4, -s4, 0, a(4)]
[ ca4*s4, ca4*c4, -sa4, -d(4)*sa4]
[ sa4*s4, sa4*c4, ca4, d(4)*ca4]
[ 0, 0, 0, 1] ];
T45 = [ [ c5, -s5, 0, a(5)]
[ ca5*s5, ca5*c5, -sa5, -d(5)*sa5]
[ sa5*s5, sa5*c5, ca5, d(5)*ca5]
[ 0, 0, 0, 1] ];

T010 = T1;
T020 = T010*T12;
T030 = T020*T23;
T040 = T030*T34;
T050 = T040*T45;
frames_x = [0, T010(1,4), T020(1,4), T030(1,4), T040(1,4), T050(1,4)];
frames_y = [0, T010(2,4), T020(2,4), T030(2,4), T040(2,4), T050(2,4)];
frames_z = [0, T010(3,4), T020(3,4), T030(3,4), T040(3,4), T050(3,4)];



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% plot in matlab %%%

scatter3(frames_x(6),frames_y(6),frames_z(6),'r');
hold on;

arm=plot3(frames_x,frames_y,frames_z,'g');
xlabel('x(m)'); ylabel('y(m)');
zlabel('z(m)'); grid on; pause(0.0001);
delete(arm); end
```

PartII_1

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% 2022 Jan %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%% WENTAO GAO &&& JINGZHI WU %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%% initialization %%%
clear all; clc;
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% parameters %%%

for i=1:3
ra(i) =170;
end

L=130;
Rplat=130;
Rbase=290;
PB1=pi/2;
PB2=pi+pi/6;
PB3=2*pi-pi/6;
PP1=pi/2;
PP2=pi+pi/6;
PP3=2*pi-pi/6;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% input the known parameters Xc,Yc,alpha %%%
Xc = input('X(mm) : '); Yc =
input('Y(mm) : '); alpha =
input('alpha(°) : '); a =
alpha*pi/180;

P=[0,0,0;
0,0,0];
B=[0,0,0;
0,0,0];
PB2PP=[0,0,0;
0,0,0];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% calculate the PB2PP %%%
for i=1:3
P(1,i)=-Rplat*cos((30+a+120*(i-1))*(pi()/180))+Xc;
P(2,i)=-Rplat*sin((30+a+120*(i-1))*(pi()/180))+Yc;

B(1,i)=-Rbase*cos((210+120*(i-1))*(pi()/180));
B(2,i)=-Rbase*sin((210+120*(i-1))*(pi()/180));

PB2PP(1,i)=B(1,i)+P(1,i);
PB2PP(2,i)=B(2,i)+P(2,i);
end
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% set intermediate parameters %%%
Theta=[0,0,0]; t=[0,0,0];

e1=[0,0,0];
e2=[0,0,0];
e3=[0,0,0]; for i=1:3
e1(i)=-
2*PB2PP(2,i)*ra(i);
e2(i)=-
2*PB2PP(1,i)*ra(i);
e3(i)=(PB2PP(1,i))^2+(
PB2PP(2,i))^2+ra(i)^2-
L^2;
t(i)=(-e1(i)+sqrt((e1(i))^2+(e2(i))^2-(e3(i))^2))/(e3(i)-e2(i));
Theta(i)=2*atan(t(i)); if e1(i)^2+e2(i)^2-e3(i)^2 < 0 disp('
input error !') return end end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% calculate joint positions %%%
J=[0,0,0;
0,0,0];
for i=1:3
J(1,i)=-B(1,i)+ra(i)*cos(Theta(i));
J(2,i)=-B(2,i)+ra(i)*sin(Theta(i));
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% plot the platform %%% plat=[P(1,:)
P(1,1);P(2,:) P(2,1)];
line(plat(1,:),plat(2,:), 'Color', 'g');
hold on

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% plot the base %%%
base=[-B(1,:) -B(1,1);-B(2,:) -B(2,1)];
line(base(1,:),base(2,:), 'Color', 'r');

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% plot the three links %%%
links1=[-B(1,1) J(1,1) P(1,1);-B(2,1) J(2,1) P(2,1)]; links2=[-
B(1,2) J(1,2) P(1,2);-B(2,2) J(2,2) P(2,2)]; links3=[-B(1,3) J(1,3)
P(1,3);-B(2,3) J(2,3) P(2,3)]; plot(links1(1,:),links1(2,:),'k-
^','linewidth',1); plot(links2(1,:),links2(2,:),'k-
```

```
^','linewidth',1); plot(links3(1,:),links3(2,:),'k-
^','linewidth',1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% plot the points %%%
plot(Xc,Yc,'g*'); plot(0,0,'r*');
xlabel('x-axis') ylabel('y-axis')

axis equal
grid on
hold off

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

PartII_2

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%% 2022 Jan %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%% WENTAO GAO &&& JINGZHI WU %%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%% initialization %%%
clear all; clc;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% parameters %%%

for i=1:3
ra(i) =170;
end

L=130;
Rplat=130;
Rbase=290;

lim=150; %lim=limit
res=10; %res=resolution
Xc=-lim:res:lim;
```

```
Yc=-lim:res:lim;

alpha = input(' angle : ');

P=[0,0,0;
0,0,0];
B=[0,0,0;
0,0,0];
PB2PP=[0,0,0;
0,0,0];
Theta=[0,0,0];
t=[0,0,0];

e1=[0,0,0];
e2=[0,0,0];
e3=[0,0,0];

points=zeros(2,2*lim);

m=1;



for i=1:3
B(1,i)=-Rbase*cos((210+120*(i-1))*(pi()/180));
B(2,i)=-Rbase*sin((210+120*(i-1))*(pi()/180));
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% the loop is to calculate the coordinates of the base and platform
%%% points using the formula and theory mentioned in the coursework sheet
%%% page 7 and 8 %%%



for p=1:length(Yc)
for q=1:length(Xc)
for i=1:3
P(1,i)=-Rplat*cos((30+alpha+120*(i-1))*(pi()/180))+Xc(p);
P(2,i)=-Rplat*sin((30+alpha+120*(i-1))*(pi()/180))+Yc(q);
PB2PP(1,i)=B(1,i)+P(1,i);
PB2PP(2,i)=B(2,i)+P(2,i); e1(i)=-
2*PB2PP(2,i)*ra(i); e2(i)=-2*PB2PP(1,i)*ra(i);
e3(i)=(PB2PP(1,i))^2+(PB2PP(2,i))^2+ra(i)^2-L^2;
```

```
t(i)=(-e1(i)-sqrt((e1(i))^2+(e2(i))^2-(e3(i))^2))/(e3(i)-e2(i));
Theta(i)=2*atan(t(i)); end if isreal(Theta)==1
points(1,m)=Xc(p); points(2,m)=Yc(q);
m=m+1;
end
end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% plot %%%

base=[-B(1,:) -B(1,1);-B(2,:) -B(2,1)];
if points==0 %do nothing else
scatter(points(1,:),points(2,:),'g.');
hold on; end

line(base(1,:),base(2,:), 'Color', 'r');
plot(0,0,'b*'); axis equal;
```

part3

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%% 2022 Jan %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%% WENTAO GAO &&& JINGZHI WU %%%%%%%%%%%%%%%%%%%%%%%%%%
%%% initialization %%%
clear all; clc


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% parameters %%%
l1 = 0.1 ; l2 =
0 ; l3 = 0.1 ; l4
= 0.1 ; l5 =
0.1 ; %%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%
%%%%%%%%
%%% set up three obstacles %%%

[xs,ys,zs]=sphere(900);

Xs=xs*0.008+0.19;
Ys=ys*0.008+0.16;
Zs=zs*0.008+0.085;

mesh(Xs,Ys,Zs);
hold on;

[xs,ys,zs]=sphere(900);

Xs=xs*0.008+0.19;
Ys=ys*0.008+0.13;
Zs=zs*0.008+0.085;

mesh(Xs,Ys,Zs);
hold on;

[xs,ys,zs]=sphere(900);

Xs=xs*0.008+0.205;
Ys=ys*0.008+0.135;
Zs=zs*0.008+0.0975;

mesh(Xs,Ys,Zs);
hold on;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% DH table %%%

theta = [0 ,0 , 0 ,0 , 0];
d = [l1 , 0 , 0 , 0 , l5];
alpha = [0 , sym(pi)/2 , 0 , 0 , -sym(pi)/2 ];
a = [0, 0 , l3 , l4 , 0];


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Path planning avoid obstacle %%%
```

```
for i = 1:100
if (i<=15)
X = 0.18;
Y = 0.12;
Z = 0.08+0.001*i; end

if (i>15)&&(i<=25) X =
0.18+0.0015*(i-15);
Y = 0.12+0.00125*(i-15);
Z = 0.095; end

if (i>25)&&(i<=40) X =
0.195+0.0013*(i-25);
Y = 0.1325-0.0012*(i-25);
Z = 0.095; end if
(i>40)&&(i<=50) X =
0.2145-0.00045*(i-40);
Y = 0.1145+0.00305*(i-40);
Z = 0.095+0.0015*(i-40);
end

if (i>50)&&(i<=75) X =
0.21-0.0006*(i-50);
Y = 0.145+0.0005*(i-50);
Z = 0.11-0.0006*(i-50);
end

if (i>75)&&(i<=90) X =
0.195-0.0008*(i-75);
Y = 0.1575+0.0007*(i-75);
Z = 0.095; end if
(i>90)&&(i<=100) X =
0.183-0.0003*(i-90);
Y = 0.168+0.0002*(i-90);
Z = 0.095-0.0015*(i-90);
end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% using inverse kinematic method to get theta %%%
Q = IK11(X,Y,Z,0);
```

```
theta(1) = Q(1,1);
theta(2) = Q(1,2);
theta(3) = Q(1,3);
theta(4) = Q(1,4);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% get essentical sin and cos value %%%

c1 = cos(theta(1));
c2 = cos(theta(2));
c3 = cos(theta(3));
c4 = cos(theta(4));
c5 = cos(theta(5));

ca1 = cos(alpha(1));
ca2 = cos(alpha(2));
ca3 = cos(alpha(3));
ca4 = cos(alpha(4));
ca5 = cos(alpha(5));

s1 = sin(theta(1));
s2 = sin(theta(2));
s3 = sin(theta(3));
s4 = sin(theta(4));
s5 = sin(theta(5));

sa1 = sin(alpha(1));
sa2 = sin(alpha(2));
sa3 = sin(alpha(3));
sa4 = sin(alpha(4));
sa5 = sin(alpha(5));


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% get the transfor matrix of each part of Ly- Arm %%%

T1 = [ [ c1, -s1, 0, a(1)]
[ ca1*s1, ca1*c1, -sa1, -d(1)*sa1]
[ sa1*s1, sa1*c1, ca1, d(1)*ca1]
[ 0, 0, 0, 1] ];
T12 = [ [ c2, -s2, 0, a(2)]
```

```
[ ca2*s2, ca2*c2, -sa2, -d(2)*sa2]
[ sa2*s2, sa2*c2, ca2, d(2)*ca2]
[ 0, 0, 0, 1] ];
T23 = [ [ c3, -s3, 0, a(3)]
[ ca3*s3, ca3*c3, -sa3, -d(3)*sa3]
[ sa3*s3, sa3*c3, ca3, d(3)*ca3]
[ 0, 0, 0, 1] ];
T34 = [ [ c4, -s4, 0, a(4)]
[ ca4*s4, ca4*c4, -sa4, -d(4)*sa4]
[ sa4*s4, sa4*c4, ca4, d(4)*ca4]
[ 0, 0, 0, 1] ];
T45 = [ [ c5, -s5, 0, a(5)]
[ ca5*s5, ca5*c5, -sa5, -d(5)*sa5]
[ sa5*s5, sa5*c5, ca5, d(5)*ca5]
[ 0, 0, 0, 1] ];

T010 = T1;
T020 = T010*T12;
T030 = T020*T23;
T040 = T030*T34;
T050 = T040*T45;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% The frame x,y,z can be represented %%%

frames_x = [0, T010(1,4), T020(1,4), T030(1,4), T040(1,4), T050(1,4)];
frames_y = [0, T010(2,4), T020(2,4), T030(2,4), T040(2,4), T050(2,4)];
frames_z = [0, T010(3,4), T020(3,4), T030(3,4), T040(3,4), T050(3,4)];


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% plot in matlab %%%

scatter3(frames_x(6),frames_y(6),frames_z(6),'b');
hold on;

arm=plot3(frames_x,frames_y,frames_z,'g');
xlabel('x(m)'); ylabel('y(m)');
zlabel('z(m)'); grid on; pause(0.0001);
delete(arm);
end
```