1.1 : DH Parameters

i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	d_1	θ_1
2	0	a_1	0	$\theta_2 + 90°$
3	0	a_2	$-d_3$	0
4	0	0	$-d_4$	θ_4

SCARA specs:

$a_1 = .325\ m$

$a_2 = .225\ m$

$d_1 = .416\ m$

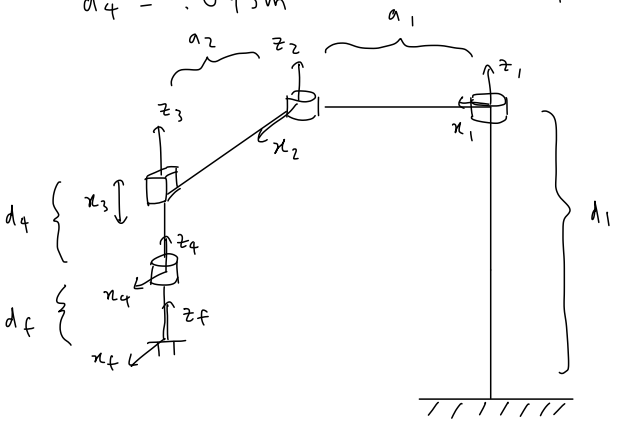
$d_4 = .093\ m$

$-170° \le \theta_1 \le 170°$

$-145° \le \theta_2 \le 145°$

$0 \le d_3 \le .150\ m$

$-360° \le d_4 \le 360°$



1.2 : FK & IK

Forward Kinematics

$$T_0^f = \begin{bmatrix} -s(\theta_1+\theta_2+\theta_4) & -c(\theta_1+\theta_2+\theta_4) & 0 & .325\,c\theta_1 - .225\,s(\theta_1+\theta_2) \\ c(\theta_1+\theta_2+\theta_4) & -s(\theta_1+\theta_2+\theta_4) & 0 & .325\,s\theta_1 + .225\,c(\theta_1+\theta_2) \\ 0 & 0 & 1 & .323 - d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Default Position (Homogeneous Transformation)

$\theta_1 = 0,\ \theta_2 = 0,\ d_3 = 0.12,\ \theta_4 = 0$

$$T_0^f = \begin{bmatrix} 0 & -1 & 0 & .325 \\ 1 & 0 & 0 & .225 \\ 0 & 0 & 1 & .203 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Inverse Kinematics

Use z-axis w/ d_3.

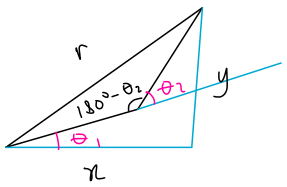
Find : $\theta_1, \theta_2, d_3, \theta_4$

Given : (x, y, z, θ_r)

position ← → orientation

$$z = .323 - d_3$$

$$\boxed{d_3 = .323 - z}$$



$x^2 + y^2 = r^2$, $L_1 = .325$, $L_2 = .325$

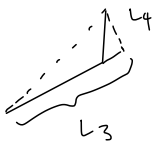
$$r^2 = L_1^2 + L_2^2 - 2L_1L_2\,c(180° - \theta_2)$$

$$c\theta_2 = -c(180° - \theta_2)$$

$$c\theta_2 = \frac{r^2 - L_1^2 - L_2^2}{2L_1L_2}$$

$$s\theta_2 = \pm\sqrt{1 - c\theta_2^2}$$

$$\theta_2 = \text{Atan2}(s\theta_2, c\theta_2)$$

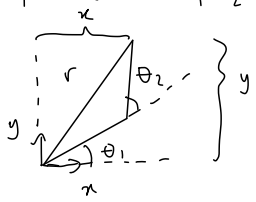


$$L_3 = L_1 + c\theta_2 L_2$$

$$L_4 = s\theta_2 L_2$$

$$\theta_1 = \text{Atan2}(y, x) - \text{Atan2}(L_4, L_3)$$

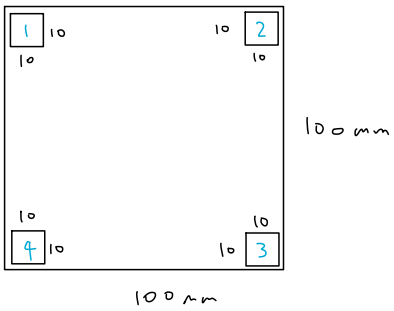
$$r^2 = a_1^2 + a_2^2 - 2a_1 a_2 \cos(180° - \theta_2)$$



$$\theta_4 = \theta_r - 90° - \theta_1 - \theta_2$$

1.3 Key Positions

PCB & Chip Dimensions



$\{0\}$: $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ base frame

$\{f\}$: $\begin{bmatrix} 0 & -1 & 0 & .325 \\ 1 & 0 & 0 & .225 \\ 0 & 0 & 1 & .180 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ feeder frame center of top of chip

Goal #1: $\begin{bmatrix} 1 & 0 & 0 & .280 \\ 0 & 1 & 0 & .270 \\ 0 & 0 & 1 & .180 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ rotate about z-axis $\begin{bmatrix} c\theta & -s\theta & 0 \\ s\theta & c\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Goal #2: $\begin{bmatrix} 0 & -1 & 0 & .280 \\ 1 & 0 & 0 & .330 \\ 0 & 0 & 1 & .180 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ ← +90° relative to #1

Goal #3: $\begin{bmatrix} -1 & 0 & 0 & .370 \\ 0 & -1 & 0 & .330 \\ 0 & 0 & 1 & .180 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ ← +90° relative to #2

Goal #4: $\begin{bmatrix} 0 & 1 & 0 & .370 \\ -1 & 0 & 0 & .240 \\ 0 & 0 & 1 & .180 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ ← +90° relative to #3

1.4 Via Points

Z-height add +10mm for clearance to avoid collisions.

a) feeder

$$\begin{bmatrix} 0 & -1 & 0 & .325 \\ 1 & 0 & 0 & .225 \\ 0 & 0 & 1 & .190 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

b) #1

$$\begin{bmatrix} 1 & 0 & 0 & .280 \\ 0 & 1 & 0 & .240 \\ 0 & 0 & 1 & .190 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

c) #2

$$\begin{bmatrix} 0 & -1 & 0 & .280 \\ 1 & 0 & 0 & .330 \\ 0 & 0 & 1 & .190 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

d) #3

$$\begin{bmatrix} -1 & 0 & 0 & .370 \\ 0 & -1 & 0 & .330 \\ 0 & 0 & 1 & .190 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

e) #4

$$\begin{bmatrix} 0 & 1 & 0 & .370 \\ -1 & 0 & 0 & .240 \\ 0 & 0 & 1 & .190 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

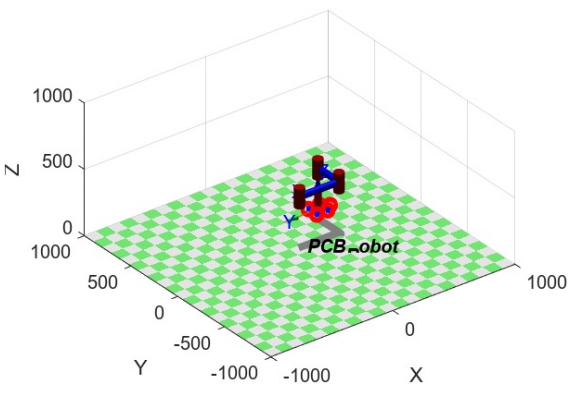
Trajectory sequence via points:

f → a → b → #1 → b → a → f → a → c →
#2 → c → a → f → a → d → #3 → d → a →
f → a → e → #4 → e

Let robot wait ~1 sec for its gripper to open & close after going to f or goal.

Start & End at "home" position:

$$\begin{bmatrix} 0 & -1 & 0 & .325 \\ 1 & 0 & 0 & .225 \\ 0 & 0 & 1 & .200 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Contents


```
% Jacob Sayono
% 505368811

% MAE C163B
% Project 2
```

Part 1:

Forward Kinematics Inverse Kinematics Matlab Robotics Toolbox

```
close; clear; clc;

% lengths (m)
a1 = 0.325;
a2 = 0.225;
d1 = 0.416;
d4 = 0.093;

% key positions
home = [0, -1, 0, .325;
        1, 0, 0, .225;
        0, 0, 1, .203;
        0, 0, 0, 1];
feeder = [0, -1, 0, .325;
          1, 0, 0, .225;
          0, 0, 1, .180;
          0, 0, 0, 1];
goal1 = [1, 0, 0, .280;
         0, 1, 0, .240;
         0, 0, 1, .180;
         0, 0, 0, 1];
goal2 = [0, -1, 0, .280;
         1, 0, 0, .330;
         0, 0, 1, .180;
         0, 0, 0, 1];
goal3 = [-1, 0, 0, .370;
         -1, 0, 0, .240;
          0, 0, 1, .180;
          0, 0, 0, 1];
goal4 = [0, 1, 0, .370;
        -1, 0, 0, .240;
         0, 0, 1, .180;
         0, 0, 0, 1];

% via points
a = [0, -1, 0, .325;
     1, 0, 0, .225;
     0, 0, 1, .190;
     0, 0, 0, 1];
b = [1, 0, 0, .280;
     0, 1, 0, .240;
     0, 0, 1, .190;
     0, 0, 0, 1];
c = [0, -1, 0, .280;
     1, 0, 0, .330;
     0, 0, 1, .190;
     0, 0, 0, 1];
d = [-1, 0, 0, .370;
     0, -1, 0, .330;
     0, 0, 1, .190;
     0, 0, 0, 1];
e = [0, 1, 0, .370;
     -1, 0, 0, .240;
     0, 0, 1, .190;
     0, 0, 0, 1];
```

```
% test forward kinematics
T = fk(0, 0, .100, 0);

% test inverse kinematics
[~, t2, d3, t4] = ik(a);

% visualize robot
L1 = Revolute('alpha', 0, 'a', 0, 'd', d1, 'qlim', [-170 170]*pi/180);
L2 = Revolute('alpha', 0, 'a', a1, 'd', 0, 'qlim', [-145 145]*pi/180);
L3 = Prismatic('alpha', pi, 'a', a2, 'theta', 0, 'qlim', [0 .15]);
L4 = Revolute('alpha', 0, 'a', 0, 'd', d4, 'qlim', [-360 360]*pi/180);
tool = transpose([0, 0, 0]);
PCB_Robot = SerialLink([L1 L2 L3 L4], 'name', 'PCB_Robot', 'tool', tool);

% now convert waypoints into joint space values and interpolate
trajectory_sequence = [a', a, b, b', b, a, a', a, c, c', c, a, a', a, d, d', d, a, a', a, e, e', e];
time_between_via = [4 0.2 0.2 4 0.2 0.2 4 0.2 0.2 4 0.2 0.2 4 0.2 0.2 4 0.2 0.2 4 0.2 0.2 4 0.2 0.2 4 0.2 0.2 4 0.2 0.2 4];
```

Part 2:

Joint Space Trajectory Generation (Linear w/ Parabolic Blend) Joing Angle, Velocity, Acceleration Simulate Robot and Animate Trajectory

```
% convert via points to joint space with elbow up
vp_js = zeros(4,4,6);
for i = 1:6
    vp_js(:,i+1) = IK(zeros(4,4,i));
end

% initial position and final position of end effector
vp_js(:,1) = [0 pi/2 0 -140];  % initial
vp_js(:,26) = [0 pi/2 0 -140];  % final

% time component between via points
t1 = 0;
for i = 1:6
    [q,qd,qdd,t]=traj_linear_w_parabolic_blend_vector(t1, t1+time_between_via(i), JS_via(:,i), JS_via(:,i+1), des_qdd, K*time_between_via(i)) ;
     if(i<25)
         q(:,end)=[];
         qd(:,end)=[];
         qdd(:,end)=[];
         t(end)=[];
     end
    theta1=[q_all q];
    theta2=[qd_all qd];
    theta3=[qdd_all qdd];
    if i>1
           sum = sum + time_between_via(i-1);
    end
    t = t + sum;
end

% joint acceleration, velocity, position vs time
q_all(1:3,:) = (180/pi)*q_all(1:3,:);
qd_all(1:3,:) = (180/pi)*qd_all(1:3,:);
qdd_all(1:3,:) = (180/pi)*qdd_all(1:3,:);

n = 4 * 9 * 0.2 * 16 * K - 24;

% plot graphs for theta1, theta2, theta4, and d3
titles = {'Joint Angle vs Time', 'Joint Speed vs Time', 'Joint Acceleration vs Time', ...
          'Joint Position vs Time', 'Joint Speed vs Time', 'Joint Acceleration vs Time'};
ylabels = {'Joint Angle [degree]', 'Joint Speed [degree/s]', 'Joint Acceleration [degree/s^2]', ...
           'Joint Position [mm]', 'Joint Speed [mm/s]', 'Joint Acceleration [mm/s^2]'};
data = {q_all, qd_all, qdd_all, q_all, qd_all, qdd_all};
indices = {[1, 2, 3], [1, 2, 3], [1, 2, 3], 4, 4, 4};

for i = 1:6
    figure()
    plot(t_all(1:N), data{i}(indices{i},1:N));
    legend('q1', 'q2', 'q3','d4','location', 'best');
    title(titles{i});
    xlabel('Time (s)'); ylabel(ylabels{i});
    xlim([0 t_all(N)]);
    grid on;
    hold off
```

```
end

% extract via points into simulation
viapoints=zeros(3,26);
q1=[0, pi/2, 0, -140];
T_initial_final = RH3FRH55.fkine(q1);
viapoints(:,1)=T_initial_final.t;
viapoints(:,26)=T_initial_final.t;

for i=1:24
    viapoints(:,i+1)=T_O_all_v(1:3,4,i);
end

% Extract xyz positions of each intermediary point
xyz_jointTraj = zeros(3,length(q_all));
for i = 1:length(q_all)
    T_all_points = RH3FRH55.fkine(q_all(:,i));
    xyz_jointTraj(:,i) = T_all_points.t;
end

% Plot robot with viapoints and xyz trajectories
figure()
PCB_Robot.plot(q1, 'jointdiam', 1.5,'workspace',[-1000,1000,-1000,1000,0,1000]);
hold on;
plot3(viapoints(1,:), viapoints(2,:), viapoints(3,:), 'ro', 'LineWidth', 2); hold on;
plot3(xyz_jointTraj(1,:), xyz_jointTraj(2,:), xyz_jointTraj(3,:), 'b.-');
grid on;
```

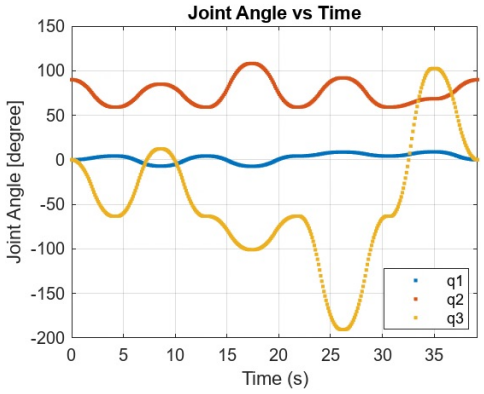
Function Definitions

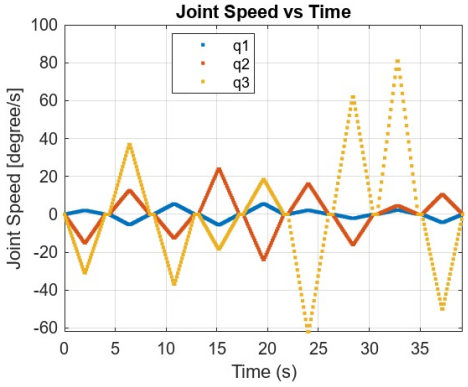
```
% forward kinematics function
function T = fk(theta1, theta2, d3, theta4)
    a1 = 0.325;
    a2 = 0.225;
    d1 = 0.416;
    d4 = 0.093;

    T01 = mat_from_DH(0, 0, d1, theta1);
    T12 = mat_from_DH(0, a1, 0, theta2 + 90);
    T23 = mat_from_DH(0, a2, -d3, 0);
    T34 = mat_from_DH(0, 0, -d4, theta4);

    T = T01*T12*T23*T34;
end

% matrix function given DH parameters
function matrix = mat_from_DH(alpha_iminus1, a_iminus1, d_i, theta_i)
    matrix = [cosd(theta_i) -sind(theta_i) 0 a_iminus1;
              sind(theta_i)*cosd(alpha_iminus1) cosd(theta_i)*cosd(alpha_iminus1) -sind(alpha_iminus1) -sind(alpha_iminus1)*d_i;
              sind(theta_i)*sind(alpha_iminus1) cosd(theta_i)*sind(alpha_iminus1) cosd(alpha_iminus1) cosd(alpha_iminus1)*d_i;
              0 0 0 1];
end

% inverse kinematics function
function [theta1, theta2, d3, theta4] = ik(T)
    [x, y, z, theta] = matrix_to_position(T);

    a1 = 0.325;
    a2 = 0.225;
    d3_val = .323-z;

    c_theta_2 = (x^2+y^2-a1^2-a2^2)/(2*a1*a2);
    s_theta_2_pos = sqrt(1-c_theta_2^2);
    s_theta_2_neg = -sqrt(1-c_theta_2^2);

    theta_2_val1 = atan2(s_theta_2_pos,c_theta_2);
    theta_2_val2 = atan2(s_theta_2_neg,c_theta_2);

    L3_1 = a1+cos(theta_2_val1)*a2;
    L3_2 = a1+cos(theta_2_val2)*a2;
    L4_1 = sin(theta_2_val1)*a2;
    L4_2 = sin(theta_2_val2)*a2;

    theta_1_val1 = atan2(y,x) - atan2(L4_1,L3_1);
    theta_1_val2 = atan2(y,x) - atan2(L4_2,L3_2);
```

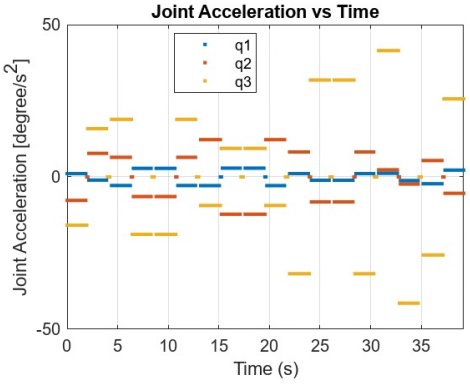
```
    theta_4_val1 = theta - pi/4 - theta_1_val1 - theta_2_val1;
    theta_4_val2 = theta - pi/4 - theta_1_val2 - theta_2_val2;

    if (d3_val < 0 || d3_val > 0.150)
        d3 = -1;
    else
        d3 = d3_val;
    end

    if (check_angles(theta_1_val1, theta_2_val1, theta_4_val1) == 1)
        theta1 = theta_1_val1;
        theta2 = theta_2_val1;
        theta4 = theta_4_val1;
        return
    elseif (check_angles(theta_1_val2, theta_2_val2, theta_4_val2) == 1)
        theta1 = theta_1_val2;
        theta2 = theta_2_val2;
        theta4 = theta_4_val2;
        return
    else
        theta1 = -1;
        theta2 = -1;
        theta4 = -1;
        return
    end
end

% check angles function
function check = check_angles(t1, t2, t4)
    check = -1;
    if (t1 < -170 || t1 > 170)
        return
    elseif (t2 < -145 || t2 > 145)
        return
    elseif (t4 < -360 || t4 > 360)
        return
    else
        check = 1;
        return
    end
end

% euler angles function
function [x, y, z, theta] = matrix_to_position(T)
    x = T(1,4);
    y = T(2,4);
    z = T(3,4);
    euler_angles = rotm2eul(T(1:3,1:3));
    theta = euler_angles(1);
end

% trajectory generation (provided by TA)
function [q, qd, qdd, t] = traj_linear_w_parabolic_blend_vector(t1, t2, q1, q2, des_qdd, n_intervals)
    q = zeros(length(q1), n_intervals); qd = q; qdd = q; t = q;
    for i = 1:length(q1)
        [q(i,:), qd(i,:), qdd(i,:), t] = traj_linear_w_parabolic_blend_scalar(t1, t2, q1(i), q2(i), des_qdd(i), n_intervals);
    end
end

function [q, qd, qdd, t] = traj_linear_w_parabolic_blend_scalar(t1, t2, q1, q2, ~, n_intervals)
    t = linspace(t1, t2, n_intervals);
    des_qdd = 4*(abs(q1-q2))/(t2-t1)^2+0.0001;
    tb = 0.5*(t2-t1) - 0.5*sqrt(des_qdd^2*(t2-t1)^2 - 4*abs(des_qdd)*abs(q2-q1))/abs(des_qdd);

    if q2 > q1
        % First parabolic region
        constraints = [q1; 0; des_qdd];
        relationships = [1, t1, t1^2;
                         0,  1, 2*t1;
                         0,  0,    2];
        ab1 = (relationships\constraints)';

        % Second parabolic region
        constraints = [q2; 0; -des_qdd];
        relationships = [1, t2, t2^2;
```

```
                        0,  1, 2*t2;
                        0,  0,    2];
        ab2 = (relationships\constraints)';
    else
        % First parabolic region
        constraints = [q1; 0; -des_qdd];
        relationships = [1, t1, t1^2;
                        0,  1, 2*t1;
                        0,  0,    2];
        ab1 = (relationships\constraints)';

        % Second parabolic region
        constraints = [q2; 0;  des_qdd];
        relationships = [1, t2, t2^2;
                        0,  1, 2*t2;
                        0,  0,    2];
        ab2 = (relationships\constraints)';
    end

    % Linear region
    q1b = ab1(1) + ab1(2)*(t1+tb) + ab1(3)*(t1+tb)^2;
    q2b = ab2(1) + ab2(2)*(t2-tb) + ab2(3)*(t2-tb)^2;
    constraints =   [q1b; q2b];
    relationships = [1 tb+t1; 1 t2-tb];
    al = (relationships\constraints)';

    % Outputs
    t11 = t((t1<=t) & (t<=t1+tb)); % first parabolic region
        a0 = ab1(1); a1 = ab1(2); a2 = ab1(3);
        q   = a0 + a1*t11 + a2*t11.^2;
        qd  =      a1   + 2*a2*t11;
        qdd =             2*a2*ones(size(t11));

    t22 = t((t1+tb<t) & (t<t2-tb)); % linear region
        a0 = al(1); a1 = al(2);
        q   = [q, a0 + a1*t22];
        qd  = [qd, a1.*ones(size(t22))];
        qdd = [qdd, zeros(size(t22))];

    t33 = t((t2-tb<=t) & (t<=t2)); % second parabolic region
        a0 = ab2(1); a1 = ab2(2); a2 = ab2(3);
        q   = [q, a0 + a1*t33 + a2*t33.^2];
        qd  = [qd,     a1   + 2*a2*t33];
        qdd = [qdd,            2*a2*ones(size(t33))];
end
```

Published with MATLAB® R2022b







```
% Jacob Sayono
% 505368811

% MAE C163B
% Project 2

close; clear; clc;

% lengths (m)
a1 = 0.325;
a2 = 0.225;
d1 = 0.416;
d4 = 0.093;

% key positions
home = [0, -1, 0, .325;
        1, 0, 0, .225;
        0, 0, 1, .203;
        0, 0, 0, 1];
feeder = [0, -1, 0, .325;
          1, 0, 0, .225;
          0, 0, 1, .180;
          0, 0, 0, 1];
goal1 = [1, 0, 0, .280;
         0, 1, 0, .240;
         0, 0, 1, .180;
         0, 0, 0, 1];
goal2 = [0, -1, 0, .280;
         1, 0, 0, .330;
         0, 0, 1, .180;
         0, 0, 0, 1];
goal3 = [-1, 0, 0, .370;
         -1, 0, 0, .240;
          0, 0, 1, .180;
          0, 0, 0, 1];
goal4 = [0, 1, 0, .370;
        -1, 0, 0, .240;
         0, 0, 1, .180;
         0, 0, 0, 1];

% via points
a = [0, -1, 0, .325;
     1, 0, 0, .225;
     0, 0, 1, .190;
     0, 0, 0, 1];
b = [1, 0, 0, .280;
     0, 1, 0, .240;
     0, 0, 1, .190;
     0, 0, 0, 1];
c = [0, -1, 0, .280;
     1, 0, 0, .330;
     0, 0, 1, .190;
     0, 0, 0, 1];
d = [-1, 0, 0, .370;
     0, -1, 0, .330;
     0, 0, 1, .190;
     0, 0, 0, 1];
e = [0, 1, 0, .370;
     -1, 0, 0, .240;
     0, 0, 1, .190;
     0, 0, 0, 1];

% forward kinematics
T = FK(0, 0, .100, 0)

% inverse kinematics
[t1, t2, d3, t4] = IK(a)
```

```
% visualize robot
L1 = Revolute('alpha', 0, 'a', 0, 'd', d1, 'qlim', [-170 170]*pi/180);
L2 = Revolute('alpha', 0, 'a', a1, 'd', 0, 'qlim', [-145 145]*pi/180);
L3 = Prismatic('alpha', pi, 'a', a2, 'theta', 0, 'qlim', [0 .15]);
L4 = Revolute('alpha', 0, 'a', 0, 'd', d4, 'qlim', [-360 360]*pi/180);
tool = trans1(0, 0, 0);
PCB_Builder = SerialLink([L1 L2 L3 L4], 'name', 'PCB_Builder', 'tool', tool);

% convert waypoints into joint space values and interpolate

% forward kinematics function
function T = FK(theta1, theta2, d3, theta4)
    a1 = 0.325;
    a2 = 0.225;
    d1 = 0.416;
    d4 = 0.093;

    T01 = mat_from_DH(0, 0, d1, theta1);
    T12 = mat_from_DH(0, a1, 0, theta2 + 90);
    T23 = mat_from_DH(0, a2, -d3, 0);
    T34 = mat_from_DH(0, 0, -d4, theta4);

    T = T01*T12*T23*T34;
end

% matrix function given DH parameters
function matrix = mat_from_DH(alpha_iminus1, a_iminus1, d_i, theta_i)
    matrix = [cosd(theta_i) -sind(theta_i) 0 a_iminus1;
              sind(theta_i)*cosd(alpha_iminus1) cosd(theta_i)*cosd(alpha_iminus1) -sind(alpha_iminus1) -sind(alpha_iminus1)*d_i;
              sind(theta_i)*sind(alpha_iminus1) cosd(theta_i)*sind(alpha_iminus1) cosd(alpha_iminus1) cosd(alpha_iminus1)*d_i;
              0 0 0 1];
end

% inverse kinematics function
function [theta1, theta2, d3, theta4] = IK(T)
    [x, y, z, theta] = mat_to_pos(T);

    a1 = 0.325;
    a2 = 0.225;
    d3_min = 0;
    d3_max = 0.150;
    d3_val = .323-z;

    c_theta_2 = (x^2+y^2-a1^2-a2^2)/(2*a1*a2);
    s_theta_2_pos = sqrt(1-c_theta_2^2);
    s_theta_2_neg = -sqrt(1-c_theta_2^2);

    theta_2_val1 = atan2(s_theta_2_pos,c_theta_2);
    theta_2_val2 = atan2(s_theta_2_neg,c_theta_2);

    L3_1 = a1+cos(theta_2_val1)*a2;
    L3_2 = a1+cos(theta_2_val2)*a2;
    L4_1 = sin(theta_2_val1)*a2;
    L4_2 = sin(theta_2_val2)*a2;

    theta_1_val1 = atan2(y,x) - atan2(L4_1,L3_1);
    theta_1_val2 = atan2(y,x) - atan2(L4_2,L3_2);

    theta_4_val1 = theta - pi/4 - theta_1_val1 - theta_2_val1;
    theta_4_val2 = theta - pi/4 - theta_1_val2 - theta_2_val2;

    if (d3_val < d3_min || d3_val > d3_max)
        d3 = -1;
    else
        d3 = d3_val;
    end

    if (check_angles(theta_1_val1, theta_2_val1, theta_4_val1) == 1)
```

```
        theta1 = theta_1_val1;
        theta2 = theta_2_val1;
        theta4 = theta_4_val1;
        return
    elseif (check_angles(theta_1_val2, theta_2_val2, theta_4_val2) == 1)
        theta1 = theta_1_val2;
        theta2 = theta_2_val2;
        theta4 = theta_4_val2;
        return
    else
        theta1 = -1;
        theta2 = -1;
        theta4 = -1;
        return
    end
end

% check angles function
function check = check_angles(t1_test, t2_test, t4_test)
    t1_min = -170;
    t1_max = 170;
    t2_min = -145;
    t2_max = 145;
    t4_min = -360;
    t4_max = 360;

    check = -1;
    if (t1_test < t1_min || t1_test > t1_max)
        return
    elseif (t2_test < t2_min || t2_test > t2_max)
        return
    elseif (t4_test < t4_min || t4_test > t4_max)
        return
    else
        check = 1;
        return
    end
end

% euler angles function
function [x, y, z, theta] = mat_to_pos(T)
    x = T(1,4);
    y = T(2,4);
    z = T(3,4);
    eul_angles = rotm2eul(T(1:3,1:3));
    theta = eul_angles(1);
end
```

Published with MATLAB® R2022b

Jacob Sayono

505368811

MAE C163B Midterm

Foward Kinematics

$$ {}^{i-1}_{i}T = \begin{bmatrix} c\theta_i & -s\theta_i & 0 & a_{i-1} \\ s\theta_i c\alpha_{i-1} & c\theta_i c\alpha_{i-1} & -s\alpha_{i-1} & -s\alpha_{i-1} d_i \\ s\theta_i s\alpha_{i-1} & c\theta_i s\alpha_{i-1} & c\alpha_{i-1} & c\alpha_{i-1} d \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	0	θ_1
2	$-90°$	0	0	θ_2
3	0	a_2	d_3	θ_3
4	$-90°$	a_3	d_4	θ_4
5	$90°$	0	0	θ_5
6	$-90°$	0	0	θ_6

$$ {}^{0}_{1}T = \begin{bmatrix} c\theta_1 & -s\theta_1 & 0 & 0 \\ s\theta_1 & c\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{1}_{2}T = \begin{bmatrix} c\theta_2 & -s\theta_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -s\theta_2 & -c\theta_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{2}_{3}T = \begin{bmatrix} c\theta_3 & -s\theta_3 & 0 & a_2 \\ s\theta_3 & c\theta_3 & 0 & 0 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{3}_{4}T = \begin{bmatrix} c\theta_4 & -s\theta_4 & 0 & a_3 \\ 0 & 0 & 1 & d_4 \\ -s\theta_4 & -c\theta_4 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{4}_{5}T = \begin{bmatrix} c\theta_5 & -s\theta_5 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ s\theta_5 & c\theta_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{5}_{6}T = \begin{bmatrix} c\theta_6 & -s\theta_6 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -s\theta_6 & -c\theta_6 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$a_2 = 0.4318$

$a_3 = 0.0190$

$d_3 = 0.1254$

$d_4 = 0.4318$

$$ {}^{b}_{t}T = \begin{bmatrix} c\theta_t & -s\theta_t & 0 & -0.1 \\ s\theta_t & c\theta_t & 0 & 0 \\ 0 & 0 & 1 & 0.08 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$\alpha = 0$

$a = -0.1$

$d = 0.08$

$\theta = \theta_t$

Inverse Kinematics

$$P_x = a_2 c_2 + a_3 c_{23} - d_4 c_{23}$$

$$P_y = d_3$$

$$P_z = -a_3 s_{23} - a_2 s_2 - d_4 c_{23}$$

$$p_x = \rho \cos\phi$$

$$p_y = \rho \sin\phi$$

$$\rho = \sqrt{p_x^2 + p_y^2}$$

$$\phi = A\tan2(p_y, p_x)$$

$$\theta_1 = A\tan2(p_y, p_x) - A\tan2\left(\frac{d_3}{\rho}, \pm\sqrt{1 - \frac{d_3^2}{\rho^2}}\right)$$

$$k = \frac{1}{2a_2}\left(p_x^2 + p_y^2 + p_z^2 - a_2^2 - a_3^2 - d_3^2 - a_4^2\right)$$

$$\theta_3 = A\tan2\left[(-a_3 - a_2 c_3)p_z + (c_1 p_x + s_1 p_y)(a_2 s_3 - d_4),\ (a_2 s_3 - d_4)p_z - (-a_3 - c_{12} c_3)(c_1 p_x + s_1 p_y)\right]$$

$$\theta_2 = \theta_{23} - \theta_3$$

$$\theta_4 = A\tan2\left(-r_{13} s_1 + r_{23} c_1,\ -r_{13} c_1 c_{23} - r_{23} s_1 c_{23} + s_{23} r_{33}\right)$$

$$-s_5 = r_{13}(c_1 c_{23} c_4 + s_1 s_4) + r_{23}(s_1 s_{23} c_4 - c_1 s_4) - r_{33}(s_{23} c_4)$$

$$c_5 = r_{13}(-c_1 s_{23}) + r_{23}(-s_1 s_{23}) + r_{33}(-c_{23})$$

$$\theta_5 = A\tan2(s_5, c_5)$$

$$c_6 = r_{11}(c_1 c_{23} c_4 c_5 + s_1 s_4 c_5 - c_1 s_5 s_{23}) + r_{21}(c_4 c_5 s_1 c_{23} - s_1 s_5 s_{23} - c_1 c_5 s_4) + r_{31}(-c_{23} s_5 - c_4 c_5 s_{23})$$

$$s_6 = r_{11}(c_4 s_1 - c_1 s_4 s_{23}) - (c_1 c_4 + s_1 s_4 c_{23}) r_{21} + s_{23} s_4 r_{31}$$

$$\theta_6 = A\tan2(s_6, c_6)$$

Contents

Jacob Sayono

```
% 505368811
% MAE C163B
% Midterm
```

Part 1 - Kinematic Analysis

```
close; clear; clc

% DH PARAMETERS
syms th1 th2 th3 th4 th5 th6

% Modified DH parameters
alpha = [0, -pi/2, 0, -pi/2, pi/2, -pi/2];
a = [0, 0, 0.4318, 0.019, 0, 0];
d = [0, 0, 0.1254, 0.4318, 0, 0];
th = [th1, th2, th3, th4, th5, th6];


% FORWARD KINEMATICS
syms theta1 theta2 theta3 theta4 theta5 theta6 theta_t

T01 = [cos(theta1) -sin(theta1) 0 0;
sin(theta1) cos(theta1) 0 0;
0 0 1 0;
0 0 0 1];

T12 = [cos(theta2) -sin(theta2) 0 0;
0 0 1 0;
-sin(theta2) -cos(theta2) 0 0;
0 0 0 1];

T23 = [cos(theta3) -sin(theta3) 0 0.4318;
sin(theta3) cos(theta3) 0 0;
0 0 1 0.1254;
0 0 0 1];

T34 = [cos(theta4) -sin(theta4) 0 0.019;
0 0 1 0.4318;
-sin(theta4) -cos(theta4) 0 0;
0 0 0 1];

T45 = [cos(theta5) -sin(theta5) 0 0;
0 0 -1 0;
sin(theta5) cos(theta5) 0 0;
0 0 0 1];

T56 = [cos(theta6) -sin(theta6) 0 0;
0 0 1 0;
-sin(theta6) -cos(theta6) 0 0;
0 0 0 1];

T6t = [cos(theta_t) -sin(theta_t) 0 -0.1;
sin(theta_t) cos(theta_t) 0 0;
0 0 1 0.08;
0 0 0 1];

% find forward kinematics
T = T01 * T12 * T23 * T34 * T45 * T56;
T0t = T * T6t;
Tinv6t = inv(T6t);

% set some random joint angles in radians
q = [0.2, 0.3, -0.5, 0.4, 0.1, 0.8];

% check forward kinematics function
T = fk(q)

% separate parameters from matrix
position = T(1:3, 4)
orientation = tform2eul(T, 'XYZ')

% INVERSE KINEMATICS
[T1, T2, T3, T4, T5, T6, Tt] = IK(T)
```

```
% create table
PUMA_ik = SerialLink(L, 'name', 'PUMA_INVERSE_KINEMATICS')
```

Part 2 - Simulation

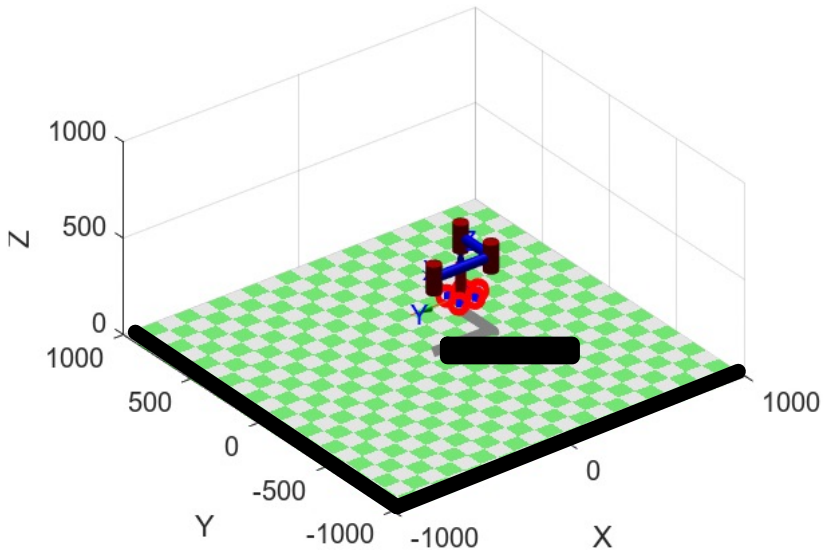
```
% FORWARD KINEMATICS

% DH parameters for Puma560
L1 = Link('revolute', 'alpha', 0, 'a', 0, 'd', 0, 'modified');
L2 = Link('revolute', 'alpha', -pi/2, 'a', 0, 'd', 0, 'modified');
L3 = Link('revolute', 'alpha', 0, 'a', 0.4318, 'd', 0.1254, 'modified');
L4 = Link('revolute', 'alpha', -pi/2, 'a', 0.019, 'd', 0.4318, 'modified');
L5 = Link('revolute', 'alpha', pi/2, 'a', 0, 'd', 0, 'modified');
L6 = Link('revolute', 'alpha', -pi/2, 'a', 0, 'd', 0, 'modified');
% L7 = Link('revolute', 'alpha', 0, 'a', -0.1, 'd', .008, 'modified');
tool = transl(0, -0.1, .008);

% create robot using the DH parameters
Puma560 = SerialLink([L1 L2 L3 L4 L5 L6], 'name', 'Puma 560', 'tool', tool);

% calculate the forward kinematics for the given joint angles
T = Puma560.fkine(q)


% INVERSE KINEMATICS

% set the desired end-effector position and orientation
position = [0.5, -0.3, 0.4];
orientation = [pi/2, 0, pi/2];

% convert the orientation to a rotation matrix
R = eul2r(orientation, 'XYZ');

% calculate the inverse kinematics for the desired pose
q = Puma560.ikine(transl(position) * rpy2tr(orientation), 'mask', [1 1 1 0 0 0])

% calculate the forward kinematics of the resulting joint angles to verify the solution
T = Puma560.fkine(q)

% visualize robot
figure(1)
Puma560.plot([T1, T2, T3, T4, T5, T6, Tt],'workspace',[-500,500,-500,500,0,500])
```

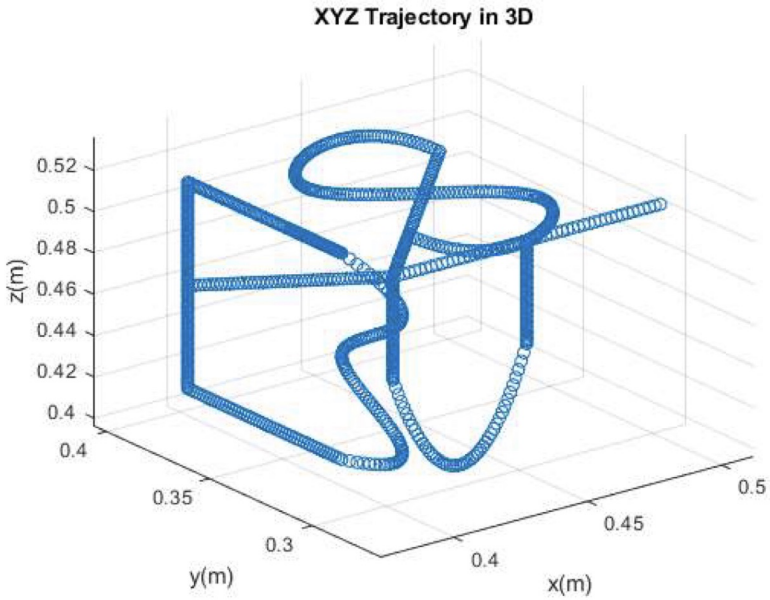
Function Definitions

```
% matrix function given DH parameters
function matrix = mat_from_DH(alpha_iminus1, a_iminus1, d_i, theta_i)
    matrix = [cosd(theta_i) -sind(theta_i) 0 a_iminus1;
              sind(theta_i)*cosd(alpha_iminus1) cosd(theta_i)*cosd(alpha_iminus1) -sind(alpha_iminus1) -sind(alpha_iminus1)*d_i;
              sind(theta_i)*sind(alpha_iminus1) cosd(theta_i)*sind(alpha_iminus1) cosd(alpha_iminus1) cosd(alpha_iminus1)*d_i;
              0 0 0 1];
end

% forward kinematics
function T = fk(theta1, theta2, theta3, theta4, theta5, theta6, thetat)
    % joint lengths
    a2 = 0.4318;
    a3 = 0.0191;
    d3 = 0.1254;
    d4 = 0.4318;

    T01 = mat_from_DH(0, 0, 0, theta1);
    T12 = mat_from_DH(-90, 0, 0, theta2);
    T23 = mat_from_DH(0, a2, d3, theta3);
    T34 = mat_from_DH(-90, a3, d4, theta4);
    T45 = mat_from_DH(90, 0, 0, theta5);
    T56 = mat_from_DH(-90, 0, 0, theta6);
    T6t = mat_from_DH(0, -0.1, 0.08, thetat);

    T = T01*T12*T23*T34*T45*T56*T6t;

end

% inverse kinematics
function [t1, t2, t3, t4, t5, t6] = ik(T_location)
    % joint lengths
    a2 = 0.4318;
    a3 = 0.0191;
    d3 = 0.1254;
    d4 = 0.4318;

    % transformation matrices
    T_Gto6 = [1 0 0 0; 0 1 0 0; 0 0 1 0.05625; 0 0 0 1];
    T_TtoG = [1 0 0 -0.1; 0 1 0 0; 0 0 1 0.08; 0 0 0 1];
    T_input = T_location * inverse(T_TtoG) * inverse(T_Gto6);
```

```
    % extract variables
    r = T_input(1:3,1:3);
    p = T_input(1:3,4);
    px = p(1);
    py = p(2);
    pz = p(3);

    % compute inverse kinematics
    t1 = atan2(py, px);
    t3 = atan2(a3, -d4) - atan2(sqrt(1 - ((px^2 + py^2 + pz^2 - a2^2 - a3^2 - d3^2 - d4^2)/(2*a2))^2), (px^2 + py^2 + pz^2 - a2^2 - a3^2 - d3^2 - d4^2)/(2*a2));
    t23 = atan2((a3 + a2*cos(t3))*pz - (cos(t1)*px + sin(t1)*py)*(a2*sin(t3) - d4), (a3 + a2*cos(t3))*(cos(t1)*px + sin(t1)*py) + (a2*sin(t3) - d4)*pz);
    t2 = t23 - t3;
    t4 = atan2(r(3,1)*sin(t1) - r(3,2)*cos(t1), -r(1,3)*sin(t1)*cos(t2+t3) + r(2,3)*cos(t1)*cos(t2+t3) - r(3,3)*sin(t2+t3));
    t5 = atan2(sqrt(1 - (r(1,3)*sin(t1)*cos(t2+t3) - r(2,3)*cos(t1)*cos(t2+t3) + r(3,3)*sin(t2+t3))^2), r(1,3)*cos(t1)*cos(t2+t3) + r(2,3)*sin(t1)*cos(t2+t3) + r(3,
    t6 = atan2(-r(1,2)*sin(t1)*cos(t4) - r(2,2)*cos(t1)*cos(t4) + r(3,2)*sin(t4), r(1,1)*sin(t1)*cos(t4) + r(2,1)*cos(t1)*cos(t4) - r(3,1)*sin(t4)*cos(t2+t3));
end
```

Published with MATLAB® R2022b





Jacob Sayono

MAE C163B Midterm Report PART 3

Contents

Forward Kinematics

```
% Symbolic Expressions

syms t1 t2 t3 t4 t5 t6 a2 a3 d2 d3 d4

DH = [      0        0       0       t1;     %alpha, a, d, theta
            -pi/2     0       d2      t2;
            0       a2      d3      t3;
            pi/2    a3      d4      t4;
            -pi/2   0       0       t5;
            pi/2    0       0       t6
        ]

T_01 = transformationMatrix(DH(1,:));
T_12 = transformationMatrix(DH(2,:));
T_23 = transformationMatrix(DH(3,:));
T_34 = transformationMatrix(DH(4,:));
T_45 = transformationMatrix(DH(5,:));
T_56 = transformationMatrix(DH(6,:));

T_06 = T_01*T_12*T_23*T_34*T_45*T_56;
T_06 = simplify(T_06)



% Numercial Expressions


DH = [      0        0       0       t1;     %alpha, a, d, theta
            -pi/2     0       0      t2;
            0       0.4318      -0.0934      t3;
            pi/2    -0.0203      0.4331      t4;
            -pi/2   0       0       t5;
            pi/2    0       0       t6
        ]

T_01 = transformationMatrix(DH(1,:));
T_12 = transformationMatrix(DH(2,:));
T_23 = transformationMatrix(DH(3,:));
T_34 = transformationMatrix(DH(4,:));
T_45 = transformationMatrix(DH(5,:));
T_56 = transformationMatrix(DH(6,:));
T_6T = [    1       0       0       -0.1;
            0       1       0       0;
            0       0       1       0.13625;
            0       0       0       1];

T_06 = T_01*T_12*T_23*T_34*T_45*T_56;
T_06 = vpa(simplify(T_06))
T_0T = T_06*T_6T;
T_0T = vpa(simplify(T_0T))
T_T6 = (T_6T)^(-1)
```

```
DH =

[    0,  0,  0, t1]
[-pi/2,  0, d2, t2]
[    0, a2, d3, t3]
[ pi/2, a3, d4, t4]
[-pi/2,  0,  0, t5]
[ pi/2,  0,  0, t6]


T_06 =

[- sin(t6)*(cos(t4)*sin(t1) - sin(t4)*(cos(t1)*sin(t2)*sin(t3) - cos(t1)*cos(t2)*cos(t3))) - cos(t6)*(cos(t5)*(sin(t1)*sin(t4) + cos(t4)*(cos(t1)*sin(t2)*sin(t3) -
[  sin(t6)*(cos(t1)*cos(t4) + sin(t4)*(sin(t1)*sin(t2)*sin(t3) - cos(t2)*cos(t3)*sin(t1))) + cos(t6)*(cos(t5)*(cos(t1)*sin(t4) - cos(t4)*(sin(t1)*sin(t2)*sin(t3) -
[                                                                                                                                                                sin(
[
```

```
DH =

[    0,          0,          0, t1]
[-pi/2,          0,          0, t2]
[    0,  2159/5000,  -467/5000, t3]
[ pi/2, -203/10000, 4331/10000, t4]
[-pi/2,          0,          0, t5]
[ pi/2,          0,          0, t6]


T_06 =

[- 1.0*cos(t6)*(cos(t5)*(sin(t1)*sin(t4) + cos(t4)*(cos(t1)*sin(t2)*sin(t3) - 1.0*cos(t1)*cos(t2)*cos(t3))) + sin(t5)*(cos(t1)*cos(t2)*sin(t3) + cos(t1)*cos(t3)*sin
[      sin(t6)*(cos(t1)*cos(t4) + sin(t4)*(sin(t1)*sin(t2)*sin(t3) - 1.0*cos(t2)*cos(t3)*sin(t1))) - cos(t6)*(1.0*sin(t5)*(cos(t2)*sin(t1)*sin(t3) + cos(t3)*sin(t1)
[
[


T_0T =

[  - 1.0*cos(t6)*(cos(t5)*(sin(t1)*sin(t4) + cos(t4)*(cos(t1)*sin(t2)*sin(t3) - 1.0*cos(t1)*cos(t2)*cos(t3))) + sin(t5)*(cos(t1)*cos(t2)*sin(t3) + cos(t1)*cos(t3)*s
[sin(t6)*(cos(t1)*cos(t4) + sin(t4)*(sin(t1)*sin(t2)*sin(t3) - 1.0*cos(t2)*cos(t3)*sin(t1))) - 1.0*cos(t6)*(1.0*sin(t5)*(cos(t2)*sin(t1)*sin(t3) + cos(t3)*sin(t1)*s
[
[


T_T6 =

    1.0000         0         0    0.1000
         0    1.0000         0         0
         0         0    1.0000   -0.1363
         0         0         0    1.0000
```

Inverse Kinematics

```
% Numercial Expressions

% DH = [      0        0       0       t1;     %alpha, a, d, theta
%             -pi/2     0       0.2435      t2;
%             0       0.4318      -0.0934      t3;
%             pi/2    -0.0203      0.4331      t4;
%             -pi/2   0       0       t5;
%             pi/2    0       0       t6
%         ];

syms t1 t2 t3 t4 t5 t6 a2 a3 d2 d3 d4

DH = [      0        0       0       t1;     %alpha, a, d, theta
            -pi/2     0       d2      t2;
            0       a2      d3      t3;
            pi/2    a3      d4      t4;
            -pi/2   0       0       t5;
            pi/2    0       0       t6
        ]

T_01 = transformationMatrix(DH(1,:));
T_12 = transformationMatrix(DH(2,:));
T_23 = transformationMatrix(DH(3,:));
T_34 = transformationMatrix(DH(4,:));
T_45 = transformationMatrix(DH(5,:));
T_56 = transformationMatrix(DH(6,:));

T_06 = T_01*T_12*T_23*T_34*T_45*T_56;
vpa(simplify(combine(T_06)), 5)


% T_04 = T_01*T_12*T_23*T_34;
% T_04 = vpa(simplify(T_04), 5)

syms R11 R12 R13 R21 R22 R23 R31 R32 R33 Px Py Pz

Goal = [R11 R12 R13 Px;
        R21 R22 R23 Py;
        R31 R32 R33 Pz;
        0   0   0   1];



T_01_inverse = simplify(T_01^-1);

Step_1_LHS = T_01_inverse * Goal
Step_1_RHS = vpa(simplify(T_12*T_23*T_34),5)
```

```
T_02_inverse = simplify(T_12^-1)*T_01_inverse;
Step_2_LHS = (T_02_inverse * Goal)
Step_2_RHS = (simplify(T_23*T_34))

T_03 = T_01*T_12*T_23
T_03_inverse = simplify(T_03^-1)
Step_3_LHS = (T_03_inverse * Goal)
Step_3_RHS = (simplify(T_34*T_45*T_56))

T_04 = T_01*T_12*T_23*T_34
T_04_inverse = simplify(T_04^-1)
Step_4_LHS = (T_04_inverse * Goal)
Step_4_RHS = (simplify(T_45*T_56))

T_05 = T_01*T_12*T_23*T_34*T_45
T_05_inverse = simplify(T_05^-1)
Step_5_LHS = (T_05_inverse * Goal)
Step_5_RHS = (simplify(T_56))

% test IK
DH = [      0        0       0       pi/2;     %alpha, a, d, theta
            -pi/2     0       0.2435        -pi/2;
            0       0.4318      -0.0934      0;
            pi/2    -0.0203      0.4331      0;
            -pi/2   0       0      pi/3;
            pi/2    0       0       0
       ]

T_01 = transformationMatrix(DH(1,:));
T_12 = transformationMatrix(DH(2,:));
T_23 = transformationMatrix(DH(3,:));
T_34 = transformationMatrix(DH(4,:));
T_45 = transformationMatrix(DH(5,:));
T_56 = transformationMatrix(DH(6,:));


T_06 = T_01*T_12*T_23*T_34*T_45*T_56

theta = IKPuma(T_06, 0.4318, -0.0203, 0.2435, -0.0934, 0.4331)
```

```
DH =

[    0,  0,  0, t1]
[-pi/2,  0, d2, t2]
[    0, a2, d3, t3]
[ pi/2, a3, d4, t4]
[-pi/2,  0,  0, t5]
[ pi/2,  0,  0, t6]


ans =

[- 1.0*cos(t6)*(cos(t5)*(sin(t1)*sin(t4) + cos(t4)*(cos(t1)*sin(t2)*sin(t3) - 1.0*cos(t1)*cos(t2)*cos(t3))) + sin(t5)*(cos(t1)*cos(t2)*sin(t3) + cos(t1)*cos(t3)*sin
[      sin(t6)*(cos(t1)*cos(t4) + sin(t4)*(sin(t1)*sin(t2)*sin(t3) - 1.0*cos(t2)*cos(t3)*sin(t1))) - cos(t6)*(1.0*sin(t5)*(cos(t2)*sin(t1)*sin(t3) + cos(t3)*sin(t1)
[
[


Step_1_LHS =

[R11*cos(t1) + R21*sin(t1), R12*cos(t1) + R22*sin(t1), R13*cos(t1) + R23*sin(t1), Px*cos(t1) + Py*sin(t1)]
[R21*cos(t1) - R11*sin(t1), R22*cos(t1) - R12*sin(t1), R23*cos(t1) - R13*sin(t1), Py*cos(t1) - Px*sin(t1)]
[                      R31,                       R32,                       R33,                     Pz]
[                        0,                         0,                         0,                      1]


Step_1_RHS =

[     cos(t2 + t3)*cos(t4), -1.0*cos(t2 + t3)*sin(t4), sin(t2 + t3),         a3*cos(t2 + t3) + d4*sin(t2 + t3) + a2*cos(t2)]
[                  sin(t4),                  cos(t4),            0,                                          d2 + d3]
[-1.0*sin(t2 + t3)*cos(t4),      sin(t2 + t3)*sin(t4), cos(t2 + t3), d4*cos(t2 + t3) - 1.0*a3*sin(t2 + t3) - 1.0*a2*sin(t2)]
[                        0,                         0,            0,                                                 1.0]


Step_2_LHS =

[  R11*cos(t1)*cos(t2) - R31*sin(t2) + R21*cos(t2)*sin(t1),   R12*cos(t1)*cos(t2) - R32*sin(t2) + R22*cos(t2)*sin(t1),   R13*cos(t1)*cos(t2) - R33*sin(t2) + R23*cos
[- R31*cos(t2) - R11*cos(t1)*sin(t2) - R21*sin(t1)*sin(t2), - R32*cos(t2) - R12*cos(t1)*sin(t2) - R22*sin(t1)*sin(t2), - R33*cos(t2) - R13*cos(t1)*sin(t2) - R23*sin
[                              R21*cos(t1) - R11*sin(t1),                                 R22*cos(t1) - R12*sin(t1),                                 R23*cos(t1) -
[                                                      0,                                                         0,


Step_2_RHS =
```

```
[cos(t3)*cos(t4), -cos(t3)*sin(t4),  sin(t3), a2 + a3*cos(t3) + d4*sin(t3)]
[cos(t4)*sin(t3), -sin(t3)*sin(t4), -cos(t3),      a3*sin(t3) - d4*cos(t3)]
[        sin(t4),          cos(t4),        0,                          d3]
[              0,                0,        0,                           1]


T_03 =

[cos(t1)*cos(t2)*cos(t3) - cos(t1)*sin(t2)*sin(t3), - cos(t1)*cos(t2)*sin(t3) - cos(t1)*cos(t3)*sin(t2), -sin(t1), a2*cos(t1)*cos(t2) - d3*sin(t1) - d2*sin(t1)]
[cos(t2)*cos(t3)*sin(t1) - sin(t1)*sin(t2)*sin(t3), - cos(t2)*sin(t1)*sin(t3) - cos(t3)*sin(t1)*sin(t2),  cos(t1), d2*cos(t1) + d3*cos(t1) + a2*cos(t2)*sin(t1)]
[              - cos(t2)*sin(t3) - cos(t3)*sin(t2),                  sin(t2)*sin(t3) - cos(t2)*cos(t3),        0,                                  -a2*sin(t2)]
[                                                0,                                                  0,        0,                                           1]


T_03_inverse =

[ cos(t2 + t3)*cos(t1),  cos(t2 + t3)*sin(t1), -sin(t2 + t3), -a2*cos(t3)]
[-sin(t2 + t3)*cos(t1), -sin(t2 + t3)*sin(t1), -cos(t2 + t3),  a2*sin(t3)]
[             -sin(t1),               cos(t1),             0,   - d2 - d3]
[                    0,                     0,             0,           1]


Step_3_LHS =

[  R11*cos(t2 + t3)*cos(t1) - R31*sin(t2 + t3) + R21*cos(t2 + t3)*sin(t1),   R12*cos(t2 + t3)*cos(t1) - R32*sin(t2 + t3) + R22*cos(t2 + t3)*sin(t1),   R13*cos(t2 +
[- R31*cos(t2 + t3) - R11*sin(t2 + t3)*cos(t1) - R21*sin(t2 + t3)*sin(t1), - R32*cos(t2 + t3) - R12*sin(t2 + t3)*cos(t1) - R22*sin(t2 + t3)*sin(t1), - R33*cos(t2 +
[                                           R21*cos(t1) - R11*sin(t1),                                                  R22*cos(t1) - R12*sin(t1),
[                                                                  0,                                                                          0,


Step_3_RHS =

[cos(t4)*cos(t5)*cos(t6) - sin(t4)*sin(t6), - cos(t6)*sin(t4) - cos(t4)*cos(t5)*sin(t6), cos(t4)*sin(t5),  a3]
[                       cos(t6)*sin(t5),                          -sin(t5)*sin(t6),        -cos(t5), -d4]
[cos(t4)*sin(t6) + cos(t5)*cos(t6)*sin(t4),   cos(t4)*cos(t6) - cos(t5)*sin(t4)*sin(t6), sin(t4)*sin(t5),   0]
[                                       0,                                        0,              0,   1]


T_04 =

[- sin(t1)*sin(t4) - cos(t4)*(cos(t1)*sin(t2)*sin(t3) - cos(t1)*cos(t2)*cos(t3)), sin(t4)*(cos(t1)*sin(t2)*sin(t3) - cos(t1)*cos(t2)*cos(t3)) - cos(t4)*sin(t1), cos
[  cos(t1)*sin(t4) - cos(t4)*(sin(t1)*sin(t2)*sin(t3) - cos(t2)*cos(t3)*sin(t1)), cos(t1)*cos(t4) + sin(t4)*(sin(t1)*sin(t2)*sin(t3) - cos(t2)*cos(t3)*sin(t1)), cos
[                                  -cos(t4)*(cos(t2)*sin(t3) + cos(t3)*sin(t2)),                                  sin(t4)*(cos(t2)*sin(t3) + cos(t3)*sin(t2)),
[                                                                             0,                                                                            0,


T_04_inverse =

[cos(t1)*cos(t2)*cos(t3)*cos(t4) - sin(t1)*sin(t4) - cos(t1)*cos(t4)*sin(t2)*sin(t3), cos(t1)*sin(t4) + cos(t2)*cos(t3)*cos(t4)*sin(t1) - cos(t4)*sin(t1)*sin(t2)*si
[cos(t1)*sin(t2)*sin(t3)*sin(t4) - cos(t1)*cos(t2)*cos(t3)*sin(t4) - cos(t4)*sin(t1), cos(t1)*cos(t4) - cos(t2)*cos(t3)*sin(t1)*sin(t4) + sin(t1)*sin(t2)*sin(t3)*si
[                                                  sin(t2 + t3)*cos(t1),                                                                 sin(t2 + t3)*si
[                                                                     0,


Step_4_LHS =

[R21*(cos(t1)*sin(t4) + cos(t2)*cos(t3)*cos(t4)*sin(t1) - cos(t4)*sin(t1)*sin(t2)*sin(t3)) - R11*(sin(t1)*sin(t4) - cos(t1)*cos(t2)*cos(t3)*cos(t4) + cos(t1)*cos(t4
[R21*(cos(t1)*cos(t4) - cos(t2)*cos(t3)*sin(t1)*sin(t4) + sin(t1)*sin(t2)*sin(t3)*sin(t4)) - R11*(cos(t4)*sin(t1) + cos(t1)*cos(t2)*cos(t3)*sin(t4) - cos(t1)*sin(t2
[                                                                                                                                                  R31*cos(t2 + t3) + R11*si
[


Step_4_RHS =

[ cos(t5)*cos(t6), -cos(t5)*sin(t6), sin(t5), 0]
[         sin(t6),          cos(t6),       0, 0]
[-cos(t6)*sin(t5),  sin(t5)*sin(t6), cos(t5), 0]
[               0,                0,       0, 1]


T_05 =

[- cos(t5)*(sin(t1)*sin(t4) + cos(t4)*(cos(t1)*sin(t2)*sin(t3) - cos(t1)*cos(t2)*cos(t3))) - sin(t5)*(cos(t1)*cos(t2)*sin(t3) + cos(t1)*cos(t3)*sin(t2)),   sin(t5)*
[  cos(t5)*(cos(t1)*sin(t4) - cos(t4)*(sin(t1)*sin(t2)*sin(t3) - cos(t2)*cos(t3)*sin(t1))) - sin(t5)*(cos(t2)*sin(t1)*sin(t3) + cos(t3)*sin(t1)*sin(t2)), - sin(t5)*
[                                                 - sin(t5)*(cos(t2)*cos(t3) - sin(t2)*sin(t3)) - cos(t4)*cos(t5)*(cos(t2)*sin(t3) + cos(t3)*sin(t2)),
[                                                                                                                                                       0,


T_05_inverse =

[cos(t1)*cos(t2)*cos(t3)*cos(t4)*cos(t5) - cos(t1)*cos(t2)*sin(t3)*sin(t5) - cos(t1)*cos(t3)*sin(t2)*sin(t5) - cos(t5)*sin(t1)*sin(t4) - cos(t1)*cos(t4)*cos(t5)*sin
[sin(t1)*sin(t4)*sin(t5) - cos(t1)*cos(t2)*cos(t5)*sin(t3) - cos(t1)*cos(t3)*cos(t5)*sin(t2) - cos(t1)*cos(t2)*cos(t3)*cos(t4)*sin(t5) + cos(t1)*cos(t4)*sin(t2)*sin
[                                                                                                cos(t1)*sin(t2)*sin(t3)*sin(t4) - cos(t1)*cos(t2)*cos(t3)*sin(t4) - cos
[


Step_5_LHS =
```

```
[- R11*(cos(t5)*sin(t1)*sin(t4) + cos(t1)*cos(t2)*sin(t3)*sin(t5) + cos(t1)*cos(t3)*sin(t2)*sin(t5) - cos(t1)*cos(t2)*cos(t3)*cos(t4)*cos(t5) + cos(t1)*cos(t4)*cos(
[  R31*(cos(t5)*sin(t2)*sin(t3) - cos(t2)*cos(t3)*cos(t5) + cos(t2)*cos(t4)*sin(t3)*sin(t5) + cos(t3)*cos(t4)*sin(t2)*sin(t5)) - R21*(cos(t1)*sin(t4)*sin(t5) + cos(
[
[


Step_5_RHS =

[cos(t6), -sin(t6),  0, 0]
[      0,        0, -1, 0]
[sin(t6),  cos(t6),  0, 0]
[      0,        0,  0, 1]


DH =

         0         0         0    1.5708
   -1.5708         0    0.2435   -1.5708
         0    0.4318   -0.0934         0
    1.5708   -0.0203    0.4331         0
   -1.5708         0         0    1.0472
    1.5708         0         0         0


T_06 =

    0.0000   -1.0000   -0.0000   -0.1501
    0.8660    0.0000   -0.5000   -0.4331
    0.5000    0.0000    0.8660    0.4115
         0         0         0    1.0000


theta =

    4.0452   -1.5708   -3.0479   -0.3315   -1.2556    0.7050
    4.0452    0.0511         0    0.7268    0.4844   -0.0680
    1.5708    3.0905   -3.0479         0   -0.5661         0
    1.5708   -1.5708         0         0    1.0472         0
```

Trajectory Generation (Joint Space)

```
clear all
close all

% Numercial Expressions

DH = [      0        0       0       0;     %alpha, a, d, theta
            -pi/2     0       0.2435      0;
            0       0.4318      -0.0934      pi;
            pi/2    -0.0203      0.4331      0;
            -pi/2   0       0       0;
            pi/2    0       0       0
        ]

T_01 = transformationMatrix(DH(1,:));
T_12 = transformationMatrix(DH(2,:));
T_23 = transformationMatrix(DH(3,:));
T_34 = transformationMatrix(DH(4,:));
T_45 = transformationMatrix(DH(5,:));
T_56 = transformationMatrix(DH(6,:));
T_6T = [    1       0       0       -0.1;
            0       1       0       0;
            0       0       1       0.13625;
            0       0       0       1]

T_06 = T_01*T_12*T_23*T_34*T_45*T_56
T_0T = T_06*T_6T
T_T6 = (T_6T)^(-1)

% Intial Configuration
T_initial = FKPuma([0 0 pi 0 0 0])

 FKPuma([0 0 0 0 pi/2 0])

T0_temp = FKPuma([0 -pi/2 3.5*pi/4 0 0 0])
[R0_temp, P0_temp] = tr2rt(T0_temp);
R0 = [  0 1 0;
        0 0 1;
        1 0 0];
T0 = rt2tr(R0, P0_temp)

IKPuma(T0, 0.4318, -0.0203, 0.2435, -0.0934, 0.4331)

% This the common corner point of the cube. This will be considered
```

```
% origin to find other points
T0_corner = T0*T_6T


[R0_corner, P0_corner] = tr2rt(T0_corner);
P_now = [0; 0; 0];
theta1 = IKPuma2(P_now, P0_corner, 0.4318, -0.0203, 0.2435, -0.0934, 0.4331, 1)

P = [];
positionError = [];
orientationError = [];

% First Letter (S)
Sxy = [ 0.075, 0.075;
        0.05, 0.1;
        0.025, 0.075;
        0.05, 0.05;
        0.075, 0.025;
        0.05, 0;
        0.025, 0.025];

ax_lim_s = [-0.02 0.12 -0.02 0.12];

[sx, sy] = plotLetter(Sxy, 200, ax_lim_s);

% Phase 1: Corner Point to Start of First Letter Transition 1
Tr1xy = [0, 0;
         sx(1), sy(1)];
[tr1x, tr1y] = plotLines(Tr1xy, 50);

theta1 = [];
theta0 = [0 0 pi 0 0 0];
theta_corner = IKPuma3([0; 0; 0], P0_corner, 0.4318, -0.0203, 0.2435, -0.0934, 0.4331, 1)
theta1 = [theta1; JSTrajectory2(theta0, theta_corner, 50)];

for i=1:length(tr1x)
    Px = tr1x(i); Py = tr1y(i); Pz = 0;
    P_now = [Px; Py; Pz];
    theta_now = IKPuma3(P_now, P0_corner, 0.4318, -0.0203, 0.2435, -0.0934, 0.4331, 1);
    theta1 = [theta1; theta_now];
    P = [P P_now];
    [positionErrorNow, orientationErrorNow] = calculatePositionError(theta_now, P_now, P0_corner, 1);
    positionError = [positionError, positionErrorNow];
    orientationError = [orientationError, orientationErrorNow];
end

figure
time1 = 10;
time = plotThetas(theta1, time1, 1);

figure
plotxyz(theta1, time1, 1);

% Phase 2 : Trace the first letter
theta2 = [];
for i=1:length(sx)
    Px = sx(i); Py = sy(i); Pz = 0;
    P_now = [Px; Py; Pz];
    theta_now = IKPuma3(P_now, P0_corner, 0.4318, -0.0203, 0.2435, -0.0934, 0.4331, 1);
    theta2 = [theta2; theta_now];
    P = [P P_now];
    [positionErrorNow, orientationErrorNow] = calculatePositionError(theta_now, P_now, P0_corner, 1);
    positionError = [positionError, positionErrorNow];
    orientationError = [orientationError, orientationErrorNow];
end

% Plot all joint angles of phase 2
figure
time2 = 20;
time_prev = time(end);
time = [time, plotThetas(theta2, time2, 2) + time_prev];

figure
plotxyz(theta2, time2, 2);

% Second Letter: B
Bxy_straight_1 = [ 0.1, -0.05;
                   0.1, 0;
                   0.05, 0;
                   0.025, 0];
Bxy_spline = [   0.025, 0
                 0, -0.025;
                 0.025, -0.05;
                 0, -0.075;
                 0.025, -0.1];
Bxy_straight_2 = [ 0.025, -0.1;
                   0.1, -0.1;
```

```
                  0.1, -0.05];


figure
ax_lim_b = [-0.02 0.12 -0.12 0.02];

[bx_1, by_1] = plotLines(Bxy_straight_1, 50);
[bx_2, by_2] = plotLetter(Bxy_spline, 100, ax_lim_b);
[bx_3, by_3] = plotLines(Bxy_straight_2, 50);
bx = [bx_1, bx_2, bx_3];
by = [by_1, by_2, by_3];
figure
plotScript(bx, by, ax_lim_b);

%Phase 3: Transition from letter S to B
Tr3xy1 = [sx(end), sy(end);
          0, 0];
[tr3x1, tr3y1] = plotLines(Tr3xy1, 50);

theta3 = [];
for i=1:length(tr3x1)
    Px = tr3x1(i); Py = tr3y1(i); Pz = 0;
    P_now = [Px; Py; Pz];
    theta_now = IKPuma3(P_now, P0_corner, 0.4318, -0.0203, 0.2435, -0.0934, 0.4331, 1);
    theta3 = [theta3; theta_now];
    P = [P P_now];
    [positionErrorNow, orientationErrorNow] = calculatePositionError(theta_now, P_now, P0_corner, 1);
    positionError = [positionError, positionErrorNow];
    orientationError = [orientationError, orientationErrorNow];
end

theta3 = [theta3; JSTrajectory1(1, 2, P0_corner, 50)];

figure
plotEulerAngles(1,2, 10/3);
% Euler angles are plotted only where euler angles are changing. Rest
% everywhere they are 0.

Tr3xy2 = [0, 0;
          bx(1), by(1)];
[tr3x2, tr3y2] = plotLines(Tr3xy2, 50);

for i=1:length(tr3x2)
    Px = 0; Py = tr3x2(i) ; Pz = tr3y2(i);
    P_now = [Px; Py; Pz];
    theta_now = IKPuma3(P_now, P0_corner, 0.4318, -0.0203, 0.2435, -0.0934, 0.4331, 2);
    theta3 = [theta3; theta_now];
    P = [P P_now];
    [positionErrorNow, orientationErrorNow] = calculatePositionError(theta_now, P_now, P0_corner, 2);
    positionError = [positionError, positionErrorNow];
    orientationError = [orientationError, orientationErrorNow];
end

figure
time_prev = time(end);
time = [time, plotThetas(theta3, 10, 3)+time_prev];

figure
plotxyz(theta3, 10, 3);

% Phase 4 : Trace the second letter B
theta4 = [];
for i=1:length(bx)
    Px = 0; Py = bx(i); Pz = by(i);
    P_now = [Px; Py; Pz];
    theta_now = IKPuma3(P_now, P0_corner, 0.4318, -0.0203, 0.2435, -0.0934, 0.4331, 2);
    theta4 = [theta4; theta_now];
    P = [P P_now];
    [positionErrorNow, orientationErrorNow] = calculatePositionError(theta_now, P_now, P0_corner, 2);
    positionError = [positionError, positionErrorNow];
    orientationError = [orientationError, orientationErrorNow];
end

figure
time_prev = time(end);
time = [time, plotThetas(theta4, 20, 4)+time_prev];

figure
plotxyz(theta4, 20, 4);

% Third letter: J
Jxy_spline = [0, -0.05;
              0.025, -0.1;
              0.05, -0.05];

Jxy_straight_1 = [0.05, -0.05;
              0.05, 0];
```

```
Jxy_straight_2 = [0.1, 0;
                  0, 0];


figure
ax_lim_p = [-0.02 0.12 -0.12 0.02];

[px_1, py_1] = plotLetter(Jxy_spline, 50, ax_lim_p);
[px_2, py_2] = plotLines(Jxy_straight_1, 50);
[px_3, py_3] = plotLines(Jxy_straight_2, 50);
px = [px_1, px_2, px_3];
py = [py_1, py_2, py_3];
figure
plotScript(px, py, ax_lim_p);

%Phase 5: Transition from letter B to J
Tr5xy1 = [bx(end), by(end);
          0, 0];
[tr5x1, tr5y1] = plotLines(Tr5xy1, 50);

theta5 = [];
for i=1:length(tr5x1)
    Px = 0; Py = tr5x1(i) ; Pz = tr5y1(i);
    P_now = [Px; Py; Pz];
    theta_now = IKPuma3(P_now, P0_corner, 0.4318, -0.0203, 0.2435, -0.0934, 0.4331, 2);
    theta5 = [theta5; theta_now];
    P = [P P_now];
    [positionErrorNow, orientationErrorNow] = calculatePositionError(theta_now, P_now, P0_corner, 2);
    positionError = [positionError, positionErrorNow];
    orientationError = [orientationError, orientationErrorNow];
end

theta5 = [theta5; JSTrajectory1(2, 3, P0_corner, 50)];

figure
plotEulerAngles(2,3, 10/3);

Tr5xy2 = [0, 0;
          px(1), py(1)];
[tr5x2, tr5y2] = plotLines(Tr5xy2, 50);

for i=1:length(tr5x2)
    Px = tr5x2(i); Py = 0; Pz = tr5y2(i);
    P_now = [Px; Py; Pz];
    theta_now = IKPuma3(P_now, P0_corner, 0.4318, -0.0203, 0.2435, -0.0934, 0.4331, 3);
    theta5 = [theta5; theta_now];
    P = [P P_now];
    [positionErrorNow, orientationErrorNow] = calculatePositionError(theta_now, P_now, P0_corner, 3);
    positionError = [positionError, positionErrorNow];
    orientationError = [orientationError, orientationErrorNow];
end

figure
time_prev = time(end);
time = [time, plotThetas(theta5, 10, 5)+time_prev];

figure
plotxyz(theta5, 10, 5);

% Phase 6 : Trace the third letter J
theta6 = [];
for i=1:length(px)
    Px = px(i); Py = 0; Pz = py(i);
    P_now = [Px; Py; Pz];
    theta_now = IKPuma3(P_now, P0_corner, 0.4318, -0.0203, 0.2435, -0.0934, 0.4331, 3);
    theta6 = [theta6; theta_now];
    P = [P P_now];
    [positionErrorNow, orientationErrorNow] = calculatePositionError(theta_now, P_now, P0_corner, 3);
    positionError = [positionError, positionErrorNow];
    orientationError = [orientationError, orientationErrorNow];
end

figure
time_prev = time(end);
time = [time, plotThetas(theta6, 20, 6)+time_prev];

figure
plotxyz(theta6, 20, 6);

% Phase 7: Letter J to the Corner
Tr7xy1 = [px(end), py(end);
          0, 0];
[tr7x1, tr7y1] = plotLines(Tr7xy1, 50);

theta7 = [];
for i=1:length(tr7x1)
```

```
    Px = tr7x1(i); Py = 0; Pz = tr7y1(i);
    P_now = [Px; Py; Pz];
    theta_now = IKPuma3(P_now, P0_corner, 0.4318, -0.0203, 0.2435, -0.0934, 0.4331, 3);
    theta7 = [theta7; theta_now];
    P = [P P_now];
    [positionErrorNow, orientationErrorNow] = calculatePositionError(theta_now, P_now, P0_corner, 3);
    positionError = [positionError, positionErrorNow];
    orientationError = [orientationError, orientationErrorNow];
end

figure
theta7 = [theta7; JSTrajectory1(3, 1, P0_corner, 50)];

theta7 = [theta7; JSTrajectory2(theta_corner, theta0, 50)];

time_prev = time(end);
time = [time, plotThetas(theta7, 10, 7)+time_prev];

figure
plotxyz(theta7, 10, 7);

figure
P(1,:) = P(1,:)+P0_corner(1);
P(2,:) = P(2,:)+P0_corner(2);
P(3,:) = P(3,:)+P0_corner(3);
scatter3(P(1,:),P(2,:),P(3,:))
axis([-0.02+P0_corner(1) 0.12+P0_corner(1) -0.02+P0_corner(2) 0.12+P0_corner(2) -0.12+P0_corner(3) 0.02+P0_corner(3)])
xlabel('x(m)')
ylabel('y(m)')
zlabel('z(m)')
title('XYZ Trajectory in 3D')

figure
plot(positionError)
ylabel('Position Error')

figure
plot(orientationError)
ylabel('Orientation Error')
```

```
DH =

         0         0         0         0
   -1.5708         0    0.2435         0
         0    0.4318   -0.0934    3.1416
    1.5708   -0.0203    0.4331         0
   -1.5708         0         0         0
    1.5708         0         0         0


T_6T =

    1.0000         0         0   -0.1000
         0    1.0000         0         0
         0         0    1.0000    0.1363
         0         0         0    1.0000


T_06 =

   -1.0000   -0.0000    0.0000    0.4521
    0.0000    1.0000    0.0000    0.1501
   -0.0000    0.0000   -1.0000   -0.4331
         0         0         0    1.0000


T_0T =

   -1.0000   -0.0000    0.0000    0.5521
    0.0000    1.0000    0.0000    0.1501
   -0.0000    0.0000   -1.0000   -0.5694
         0         0         0    1.0000


T_T6 =

    1.0000         0         0    0.1000
         0    1.0000         0         0
         0         0    1.0000   -0.1363
         0         0         0    1.0000
```

User Functions

```
function [T_06] = FKPuma(theta)

    % Numercial Expressions

        DH = [      0        0       0       theta(1);     %alpha, a, d, theta
                    -pi/2     0       0.2435      theta(2);
                    0       0.4318      -0.0934      theta(3);
                    pi/2    -0.0203      0.4331      theta(4);
                    -pi/2   0       0       theta(5);
                    pi/2    0       0       theta(6)
                ];

        T_01 = transformationMatrix(DH(1,:));
        T_12 = transformationMatrix(DH(2,:));
        T_23 = transformationMatrix(DH(3,:));
        T_34 = transformationMatrix(DH(4,:));
        T_45 = transformationMatrix(DH(5,:));
        T_56 = transformationMatrix(DH(6,:));

        T_06 = T_01*T_12*T_23*T_34*T_45*T_56;


end

function [theta] = IKPuma(M, a2, a3, d2, d3, d4 )

    Px = M(1,4); Py = M(2,4); Pz = M(3,4);

    R11 = M(1,1); R12 = M(1,2); R13 = M(1,3);
    R21 = M(2,1); R22 = M(2,2); R23 = M(2,3);
    R31 = M(3,1); R32 = M(3,2); R33 = M(3,3);
    % theta1
    t1s1 = atan2((Px^2 + Py^2 - (d2 + d3)^2)^0.5, d2 + d3) + atan2(-Px,Py);
    t1s2 = atan2(-(Px^2 + Py^2 - (d2 + d3)^2)^0.5, d2 + d3) + atan2(-Px,Py);

    %theta3 with t1 as t1s1
    t1 = t1s1;
    K3 = ((Px*cos(t1)+Py*sin(t1))^2 + Pz^2 - (a2^2 + a3^2 + d4^2))/(2*a2);
    t3s1 = atan2((a3^2 + d4^2 - K3^2)^0.5, K3) + atan2(d4, a3);
    t3s2 = atan2(-(a3^2 + d4^2 - K3^2)^0.5, K3) + atan2(d4, a3);
    t3s1 = wrapToPi(t3s1);
    t3s2 = wrapToPi(t3s2);

    %theta2 with t1s1 and t3s1
    t1 = t1s1;
    t3 = t3s1;
    A = Px*cos(t1)+Py*sin(t1);  C = -Pz;    D = a2 + a3*cos(t3) + d4*sin(t3);
    E = -Pz;                F = -A;         G = a3*sin(t3) - d4*cos(t3);
    t2s1 = atan2(D*E-A*G, C*G-D*F);

    %theta2 with t1s1 and t3s2
    t1 = t1s1;
    t3 = t3s2;
    A = Px*cos(t1)+Py*sin(t1);  C = -Pz;    D = a2 + a3*cos(t3) + d4*sin(t3);
    E = -Pz;                F = -A;         G = a3*sin(t3) - d4*cos(t3);
    t2s2 = atan2(D*E-A*G, C*G-D*F);

    %theta3 with t1 as t1s2
    t1 = t1s2;
    K3 = ((Px*cos(t1)+Py*sin(t1))^2 + Pz^2 - (a2^2 + a3^2 + d4^2))/(2*a2);
    t3s3 = atan2((a3^2 + d4^2 - K3^2)^0.5, K3) + atan2(d4, a3);
    t3s4 = atan2(-(a3^2 + d4^2 - K3^2)^0.5, K3) + atan2(d4, a3);
    t3s3 = wrapToPi(t3s3);
    t3s4 = wrapToPi(t3s4);

    %theta2 with t1s2 and t3s3
    t1 = t1s2;
    t3 = t3s3;
    A = Px*cos(t1)+Py*sin(t1);  C = -Pz;    D = a2 + a3*cos(t3) + d4*sin(t3);
    E = -Pz;                F = -A;         G = a3*sin(t3) - d4*cos(t3);
    t2s3 = atan2(D*E-A*G, C*G-D*F);

    %theta2 with t1s2 and t3s4
    t1 = t1s2;
    t3 = t3s4;
    A = Px*cos(t1)+Py*sin(t1);  C = -Pz;    D = a2 + a3*cos(t3) + d4*sin(t3);
    E = -Pz;                F = -A;         G = a3*sin(t3) - d4*cos(t3);
    t2s4 = atan2(D*E-A*G, C*G-D*F);

    theta123 = [t1s1, t2s1, t3s1;
            t1s1, t2s2, t3s2;
            t1s2, t2s3, t3s3;
            t1s2, t2s4, t3s4];

    %theta4 with t1s1, t2s1, t3s1
    t1 = t1s1;  t2 = t2s1;  t3 = t3s1;
```

```
    t4s1 = atan2(R23*cos(t1)-R13*sin(t1), R23*sin(t1)*cos(t2+t3) - R13*cos(t1)*cos(t2+t3) - R33*sin(t2+t3));
    t4s1 = wrapToHalfPi(t4s1);
    t4 = t4s1;
    t5s1 = atan2((R31*cos(t2+t3) + R21*sin(t1)*sin(t2+t3) + R11*cos(t1)*sin(t2+t3)), R31*sin(t2+t3)*cos(t4) - R21*(cos(t1)*sin(t4)...
        + sin(t1)*cos(t4)*cos(t2+t3)) - R11*(-sin(t1)*sin(t4) + cos(t1)*cos(t4)*cos(t2+t3)) );
    t6s1 = atan2((R21*cos(t1)*cos(t4) - R21*sin(t1)*sin(t4)*cos(t2+t3) - R11*sin(t1)*cos(t4) - R11*cos(t1)*sin(t4)*cos(t2+t3) + R31*sin(t2+t3)*sin(t4)),...
        (R22*cos(t1)*cos(t4) - R22*sin(t1)*sin(t4)*cos(t2+t3) - R12*sin(t1)*cos(t4) - R12*cos(t1)*sin(t4)*cos(t2+t3) + R32*sin(t2+t3)*sin(t4))  );


    t5s1 = wrapToHalfPi(t5s1);
    t6s1 = wrapToHalfPi(t6s1);

    %theta4 with t1s1, t2s2, t3s2
    t1 = t1s1; t2 = t2s2; t3 = t3s2;
    t4s2 = atan2(R23*cos(t1)-R13*sin(t1), R23*sin(t1)*cos(t2+t3) - R13*cos(t1)*cos(t2+t3) - R33*sin(t2+t3));
    t4s2 = wrapToHalfPi(t4s2);
    t4 = t4s2;
    t5s2 = atan2((R31*cos(t2+t3) + R21*sin(t1)*sin(t2+t3) + R11*cos(t1)*sin(t2+t3)), R31*sin(t2+t3)*cos(t4) - R21*(cos(t1)*sin(t4)...
        + sin(t1)*cos(t4)*cos(t2+t3)) - R11*(-sin(t1)*sin(t4) + cos(t1)*cos(t4)*cos(t2+t3)) );
    t6s2 = atan2((R21*cos(t1)*cos(t4) - R21*sin(t1)*sin(t4)*cos(t2+t3) - R11*sin(t1)*cos(t4) - R11*cos(t1)*sin(t4)*cos(t2+t3) + R31*sin(t2+t3)*sin(t4)),...
        (R22*cos(t1)*cos(t4) - R22*sin(t1)*sin(t4)*cos(t2+t3) - R12*sin(t1)*cos(t4) - R12*cos(t1)*sin(t4)*cos(t2+t3) + R32*sin(t2+t3)*sin(t4))  );


    t5s2 = wrapToHalfPi(t5s2);
    t6s2 = wrapToHalfPi(t6s2);


    %theta4 with t1s2, t2s3, t3s3
    t1 = t1s2; t2 = t2s3; t3 = t3s3;
    t4s3 = atan2(R23*cos(t1)-R13*sin(t1), R23*sin(t1)*cos(t2+t3) - R13*cos(t1)*cos(t2+t3) - R33*sin(t2+t3));
    t4s3 = wrapToHalfPi(t4s3);
    t4 = t4s3;
    t5s3 = atan2((R31*cos(t2+t3) + R21*sin(t1)*sin(t2+t3) + R11*cos(t1)*sin(t2+t3)), R31*sin(t2+t3)*cos(t4) - R21*(cos(t1)*sin(t4)...
        + sin(t1)*cos(t4)*cos(t2+t3)) - R11*(-sin(t1)*sin(t4) + cos(t1)*cos(t4)*cos(t2+t3)) );
    t6s3 = atan2((R21*cos(t1)*cos(t4) - R21*sin(t1)*sin(t4)*cos(t2+t3) - R11*sin(t1)*cos(t4) - R11*cos(t1)*sin(t4)*cos(t2+t3) + R31*sin(t2+t3)*sin(t4)),...
        (R22*cos(t1)*cos(t4) - R22*sin(t1)*sin(t4)*cos(t2+t3) - R12*sin(t1)*cos(t4) - R12*cos(t1)*sin(t4)*cos(t2+t3) + R32*sin(t2+t3)*sin(t4))  );


    t5s3 = wrapToHalfPi(t5s3);
    t6s3 = wrapToHalfPi(t6s3);


    %theta4 with t1s2, t2s4, t3s4
    t1 = t1s2; t2 = t2s4; t3 = t3s4;
    t4s4 = atan2(R23*cos(t1)-R13*sin(t1), R23*sin(t1)*cos(t2+t3) - R13*cos(t1)*cos(t2+t3) - R33*sin(t2+t3));
    t4s4 = wrapToHalfPi(t4s4);
    t4 = t4s4;
    t5s4 = atan2((R31*cos(t2+t3) + R21*sin(t1)*sin(t2+t3) + R11*cos(t1)*sin(t2+t3)), (R31*sin(t2+t3)*cos(t4) - R21*(cos(t1)*sin(t4)...
        + sin(t1)*cos(t4)*cos(t2+t3)) - R11*(-sin(t1)*sin(t4) + cos(t1)*cos(t4)*cos(t2+t3))) );
    t6s4 = atan2((R21*cos(t1)*cos(t4) - R21*sin(t1)*sin(t4)*cos(t2+t3) - R11*sin(t1)*cos(t4) - R11*cos(t1)*sin(t4)*cos(t2+t3) + R31*sin(t2+t3)*sin(t4)),...
        (R22*cos(t1)*cos(t4) - R22*sin(t1)*sin(t4)*cos(t2+t3) - R12*sin(t1)*cos(t4) - R12*cos(t1)*sin(t4)*cos(t2+t3) + R32*sin(t2+t3)*sin(t4))  );


    t5s4 = wrapToHalfPi(t5s4);
    t6s4 = wrapToHalfPi(t6s4);

    theta = [t1s1, t2s1, t3s1, t4s1, t5s1, t6s1;
            t1s1, t2s2, t3s2, t4s2, t5s2, t6s2;
            t1s2, t2s3, t3s3, t4s3, t5s3, t6s3;
            t1s2, t2s4, t3s4, t4s4, t5s4, t6s4];

end

function [theta] = IKPuma2(P, P0_corner, a2, a3, d2, d3, d4, O )

    P = P0_corner + P;

    if (O==1)
        M = [   0   1   0   P(1);
                0   0   1   P(2);
                1   0   0   P(3)
                0   0   0   1];
    elseif (O==2)
        M = [   -1   0   0   P(1);
                0   1   0   P(2);
                0   0   -1   P(3)
                0   0   0   1];
    elseif (O==3)
        M = [   0   0   -1   P(1);
                -1   0   0   P(2);
                0   1   0   P(3)
                0   0   0   1];
    end

    T_6T = [    1       0       0       -0.1;
            0       1       0       0;
            0       0       1       0.13625;
```

```
        0       0       0       1];


    M = M*(T_6T^(-1));

    Px = M(1,4); Py = M(2,4); Pz = M(3,4);

    R11 = M(1,1); R12 = M(1,2); R13 = M(1,3);
    R21 = M(2,1); R22 = M(2,2); R23 = M(2,3);
    R31 = M(3,1); R32 = M(3,2); R33 = M(3,3);



    % theta1
    t1s1 = atan2((Px^2 + Py^2 - (d2 + d3)^2)^0.5, d2 + d3) + atan2(-Px,Py);
    t1s2 = atan2(-(Px^2 + Py^2 - (d2 + d3)^2)^0.5, d2 + d3) + atan2(-Px,Py);
    if(abs(t1s1) <= abs(t1s2))
        t1 = t1s1;
    else
        t1 = t1s2;
    end

    % theta3
    K3 = ((Px*cos(t1)+Py*sin(t1))^2 + Pz^2 - (a2^2 + a3^2 + d4^2))/(2*a2);
    t3s1 = atan2((a3^2 + d4^2 - K3^2)^0.5, K3) + atan2(d4, a3);
    t3s2 = atan2(-(a3^2 + d4^2 - K3^2)^0.5, K3) + atan2(d4, a3);
    t3s1 = wrapToPi(t3s1);
    t3s2 = wrapToPi(t3s2);
    if (abs(t3s1)>=pi/2)
        t3 = t3s1;
    else
        t3 = t3s2;
    end

    % theta2
    A = Px*cos(t1)+Py*sin(t1);  C = -Pz;    D = a2 + a3*cos(t3) + d4*sin(t3);
    E = -Pz;                F = -A;         G = a3*sin(t3) - d4*cos(t3);
    t2 = atan2(D*E-A*G, C*G-D*F);



    %theta4, theta5, theta6
    t4 = atan2(R23*cos(t1)-R13*sin(t1), R23*sin(t1)*cos(t2+t3) - R13*cos(t1)*cos(t2+t3) - R33*sin(t2+t3));
    t4 = wrapToHalfPi(t4);
    t5 = atan2((R31*cos(t2+t3) + R21*sin(t1)*sin(t2+t3) + R11*cos(t1)*sin(t2+t3)), R31*sin(t2+t3)*cos(t4) - R21*(cos(t1)*sin(t4)...
        + sin(t1)*cos(t4)*cos(t2+t3)) - R11*(-sin(t1)*sin(t4) + cos(t1)*cos(t4)*cos(t2+t3)) );
    t6 = atan2((R21*cos(t1)*cos(t4) - R21*sin(t1)*sin(t4)*cos(t2+t3) - R11*sin(t1)*cos(t4) - R11*cos(t1)*sin(t4)*cos(t2+t3) + R31*sin(t2+t3)*sin(t4)),...
        (R22*cos(t1)*cos(t4) - R22*sin(t1)*sin(t4)*cos(t2+t3) - R12*sin(t1)*cos(t4) - R12*cos(t1)*sin(t4)*cos(t2+t3) + R32*sin(t2+t3)*sin(t4))  );


    t5 = wrapToHalfPi(t5);
    t6 = wrapToHalfPi(t6);


    theta = [t1, t2, t3, t4, t5, t6];

end

function [theta] = IKPuma3(P, P0_corner, a2, a3, d2, d3, d4, O )

    P = P0_corner + P;

    if (O==1)
        M = [   0   1   0   P(1);
                0   0   1   P(2);
                1   0   0   P(3)
                0   0   0   1];
    elseif (O==2)
        M = [   -1   0   0   P(1);
                0   1   0   P(2);
                0   0   -1   P(3)
                0   0   0   1];
    elseif (O==3)
        M = [   0   0   -1   P(1);
                -1   0   0   P(2);
                0   1   0   P(3)
                0   0   0   1];
    end

    T_6T = [    1       0       0       -0.1;
            0       1       0       0;
            0       0       1       0.13625;
            0       0       0       1];


    M = M*(T_6T^(-1));
```

```
    Px = M(1,4); Py = M(2,4); Pz = M(3,4);

    R11 = M(1,1); R12 = M(1,2); R13 = M(1,3);
    R21 = M(2,1); R22 = M(2,2); R23 = M(2,3);
    R31 = M(3,1); R32 = M(3,2); R33 = M(3,3);



    % theta1
    t1s1 = atan2((Px^2 + Py^2 - (d2 + d3)^2)^0.5, d2 + d3) + atan2(-Px,Py);
    t1s2 = atan2(-(Px^2 + Py^2 - (d2 + d3)^2)^0.5, d2 + d3) + atan2(-Px,Py);
    if(abs(t1s1) <= abs(t1s2))
        t1 = t1s1;
    else
        t1 = t1s2;
    end

    % theta3
    K3 = ((Px*cos(t1)+Py*sin(t1))^2 + Pz^2 - (a2^2 + a3^2 + d4^2))/(2*a2);
    t3s1 = atan2((a3^2 + d4^2 - K3^2)^0.5, K3) + atan2(d4, a3);
    t3s2 = atan2(-(a3^2 + d4^2 - K3^2)^0.5, K3) + atan2(d4, a3);
    t3s1 = wrapToPi(t3s1);
    t3s2 = wrapToPi(t3s2);
    if (abs(t3s1)>=pi/2)
        t3 = t3s1;
    else
        t3 = t3s2;
    end

    % theta2
    A = Px*cos(t1)+Py*sin(t1);  C = -Pz;    D = a2 + a3*cos(t3) + d4*sin(t3);
    E = -Pz;                F = -A;         G = a3*sin(t3) - d4*cos(t3);
    t2 = atan2(D*E-A*G, C*G-D*F);



    %theta4, theta5, theta6
    t4 = atan2(R23*cos(t1)-R13*sin(t1), R23*sin(t1)*cos(t2+t3) - R13*cos(t1)*cos(t2+t3) - R33*sin(t2+t3));
%     t4 = wrapToHalfPi(t4);
    t5 = atan2((R31*cos(t2+t3) + R21*sin(t1)*sin(t2+t3) + R11*cos(t1)*sin(t2+t3)), R31*sin(t2+t3)*cos(t4) - R21*(cos(t1)*sin(t4)...
        + sin(t1)*cos(t4)*cos(t2+t3)) - R11*(-sin(t1)*sin(t4) + cos(t1)*cos(t4)*cos(t2+t3)) );
    t6 = atan2((R21*cos(t1)*cos(t4) - R21*sin(t1)*sin(t4)*cos(t2+t3) - R11*sin(t1)*cos(t4) - R11*cos(t1)*sin(t4)*cos(t2+t3) + R31*sin(t2+t3)*sin(t4)),...
        (R22*cos(t1)*cos(t4) - R22*sin(t1)*sin(t4)*cos(t2+t3) - R12*sin(t1)*cos(t4) - R12*cos(t1)*sin(t4)*cos(t2+t3) + R32*sin(t2+t3)*sin(t4))  );


%     t5 = wrapToHalfPi(t5);
%     t6 = wrapToHalfPi(t6);


    theta = [t1, t2, t3, t4, t5, t6];

end

function [theta] = IKPuma4(M, a2, a3, d2, d3, d4 )


    T_6T = [    1       0       0       -0.1;
            0       1       0       0;
            0       0       1       0.13625;
            0       0       0       1];


    M = M*(T_6T^(-1));

    Px = M(1,4); Py = M(2,4); Pz = M(3,4);

    R11 = M(1,1); R12 = M(1,2); R13 = M(1,3);
    R21 = M(2,1); R22 = M(2,2); R23 = M(2,3);
    R31 = M(3,1); R32 = M(3,2); R33 = M(3,3);



    % theta1
    t1s1 = atan2((Px^2 + Py^2 - (d2 + d3)^2)^0.5, d2 + d3) + atan2(-Px,Py);
    t1s2 = atan2(-(Px^2 + Py^2 - (d2 + d3)^2)^0.5, d2 + d3) + atan2(-Px,Py);
    if(abs(t1s1) <= abs(t1s2))
        t1 = t1s1;
    else
        t1 = t1s2;
    end

    % theta3
    K3 = ((Px*cos(t1)+Py*sin(t1))^2 + Pz^2 - (a2^2 + a3^2 + d4^2))/(2*a2);
    t3s1 = atan2((a3^2 + d4^2 - K3^2)^0.5, K3) + atan2(d4, a3);
    t3s2 = atan2(-(a3^2 + d4^2 - K3^2)^0.5, K3) + atan2(d4, a3);
    t3s1 = wrapToPi(t3s1);
    t3s2 = wrapToPi(t3s2);
```

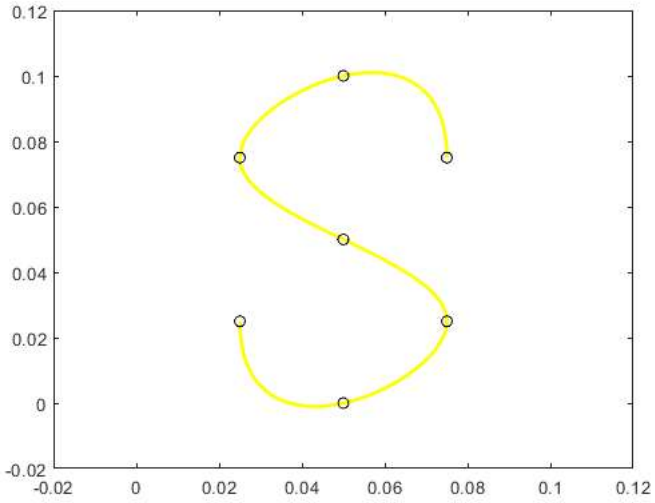
```
    if (abs(t3s1)>=pi/2)
        t3 = t3s1;
    else
        t3 = t3s2;
    end

    % theta2
    A = Px*cos(t1)+Py*sin(t1);  C = -Pz;    D = a2 + a3*cos(t3) + d4*sin(t3);
    E = -Pz;                F = -A;         G = a3*sin(t3) - d4*cos(t3);
    t2 = atan2(D*E-A*G, C*G-D*F);



    %theta4, theta5, theta6
    t4 = atan2(R23*cos(t1)-R13*sin(t1), R23*sin(t1)*cos(t2+t3) - R13*cos(t1)*cos(t2+t3) - R33*sin(t2+t3));
%     t4 = wrapToHalfPi(t4);
    t5 = atan2((R31*cos(t2+t3) + R21*sin(t1)*sin(t2+t3) + R11*cos(t1)*sin(t2+t3)), R31*sin(t2+t3)*cos(t4) - R21*(cos(t1)*sin(t4)...
        + sin(t1)*cos(t4)*cos(t2+t3)) - R11*(-sin(t1)*sin(t4) + cos(t1)*cos(t4)*cos(t2+t3)) );
    t6 = atan2((R21*cos(t1)*cos(t4) - R21*sin(t1)*sin(t4)*cos(t2+t3) - R11*sin(t1)*cos(t4) - R11*cos(t1)*sin(t4)*cos(t2+t3) + R31*sin(t2+t3)*sin(t4)),...
        (R22*cos(t1)*cos(t4) - R22*sin(t1)*sin(t4)*cos(t2+t3) - R12*sin(t1)*cos(t4) - R12*cos(t1)*sin(t4)*cos(t2+t3) + R32*sin(t2+t3)*sin(t4))  );


%     t5 = wrapToHalfPi(t5);
%     t6 = wrapToHalfPi(t6);


    theta = [t1, t2, t3, t4, t5, t6];

end

function [theta] = JSTrajectory1(O1, O2, P0_corner, n)

     thetaA = IKPuma3([0 0 0], P0_corner, 0.4318, -0.0203, 0.2435, -0.0934, 0.4331, O1);
     thetaB = IKPuma3([0 0 0], P0_corner, 0.4318, -0.0203, 0.2435, -0.0934, 0.4331, O2);


%      for i=1:6
%          theta(:,i) = linspace(thetaA(i), thetaB(i), n);
%      end
%     vmax = [8; 10; 10; 5; 5; 5];
    [m10, m11, m12, m13] = cubicTrajectory(thetaA', thetaB', n);
    t1 = linspace(0, n, n);
    theta = m10 + m11*t1 + m12*t1.^2 + m13*t1.^3;
    theta = theta';
end

function [theta] = JSTrajectory2(thetaA, thetaB, n)
%     for i=1:6
%          theta(:,i) = linspace(thetaA(i), thetaB(i), n);
%     end
    [m10, m11, m12, m13] = cubicTrajectory(thetaA', thetaB', n);
    t1 = linspace(0, n, n);
    theta = m10 + m11*t1 + m12*t1.^2 + m13*t1.^3;
    theta = theta';
end

function [t] = plotEulerAngles(O1,O2, time)
    t = linspace(0, time, 50);
    if O1 == 1 && O2 ==2
        alpha = (pi/50)*ones(1, 50);
        beta = 0;
        gamma = ((pi/2)/50)*ones(1, 50);
    end
    if O1 ==2 && O2 == 3
        alpha = ((-pi/2)/50)*ones(1, 50);
        beta = 0;
        gamma = ((-pi/2)/50)*ones(1, 50);
    end

    figure
    hold on
    plot(t, alpha);
    plot(t, beta);
    plot(t, gamma);
    hold off
    legend('$\alpha$','$\beta$','$\gamma$', 'interpreter', 'latex')
    xlabel('Time(s)')
    ylabel('Euler Angles (rad)')
end

function [lx, ly] = plotLetter(Lxy, n, ax_lim )
    xxa=Lxy(:,1);
    yya=Lxy(:,2);
    distF=[0 ;sqrt(sum(diff([xxa,yya]).^2,2))];
    distFSum=cumsum(distF);
    t = linspace(min(distFSum),max(distFSum),n);
    lx=spline(distFSum,xxa,t);
```

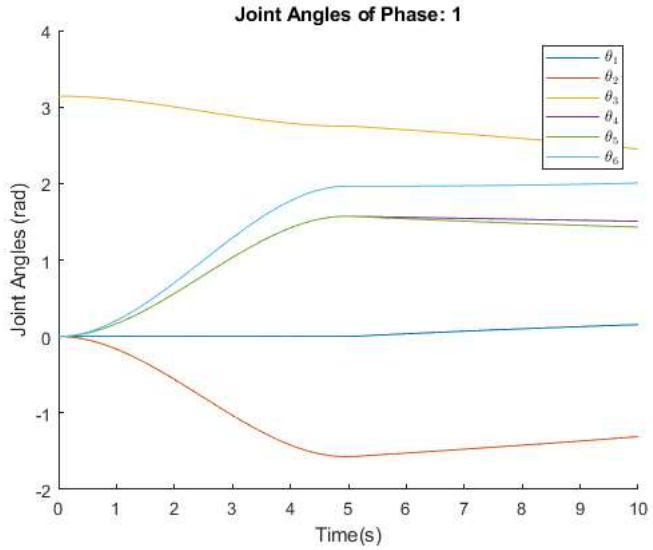
```
    ly=spline(distFSum,yya,t);
    plot(lx,ly,'y','LineWidth',2)
    hold on
    plot(xxa,yya,'ko')
    axis(ax_lim)
    hold off
end

function [lx, ly] = plotLines(Lxy, n)
    xxa=Lxy(:,1);
    yya=Lxy(:,2);
    lx = [];
    ly = [];
    for i=1:length(xxa)-1
        xxa_intermediate = linspace(xxa(i), xxa(i+1), n);
        yya_intermediate = linspace(yya(i), yya(i+1), n);
        lx = [lx, xxa_intermediate];
        ly = [ly, yya_intermediate];
    end
end

function [done] = plotScript(lx, ly, ax_lim)
    plot(lx,ly,'y','LineWidth',2)
    axis(ax_lim)
    done= 1;
end

function [t] = plotThetas(theta, time, phase)

t = linspace(0, time, length(theta(:,1)));
figure
hold on
plot(t, theta(:, 1));
plot(t, theta(:, 2));
plot(t, theta(: ,3));
plot(t, theta(: ,4));
plot(t, theta(:, 5));
plot(t, theta(:, 6));
hold off
legend('$\theta_1$','$\theta_2$','$\theta_3$','$\theta_4$','$\theta_5$','$\theta_6$', 'interpreter', 'latex')
xlabel('Time(s)')
ylabel('Joint Angles (rad)')
title(sprintf('Joint Angles of Phase: %d', phase) )
fprintf('The amount of via points used in this Phase is: %d', length(theta(:,1)))
end

function [t] = plotxyz(theta, time, phase)
    t = linspace(0, time, length(theta(:,1)));
     T_6T = [    1       0       0       -0.1;
            0       1       0       0;
            0       0       1       0.13625;
            0       0       0       1];
     x = []; y = []; z = [];
    for i=1:length(theta(:,1))
        T_actual = FKPuma(theta(i,:));
%         T_tool = T_actual*T_6T;
%         T_tool = T_actual;
        T_tool = T_actual;
        T_tool(1,4) = T_tool(1,4) + T_6T(1,4);
        T_tool(2,4) = T_tool(2,4) + T_6T(2,4);
        T_tool(3,4) = T_tool(3,4) + T_6T(3,4);
        [R_tool, P_tool] = tr2rt(T_tool);
        x = [x, P_tool(1)];
        y = [y, P_tool(2)];
        z = [z, P_tool(3)];
    end

    figure
    hold on
    plot(t, x);
    plot(t, y);
    plot(t, z);
    hold off
    legend('x','y','z')
    xlabel('Time(s)')
    ylabel('x, y, z (m)')
    title(sprintf('X,Y, Z Positions of Tool Tip of Phase: %d', phase) )
end

function [posError, orientError] = calculatePositionError(theta, P_required, P0_corner, O)
    T_actual = FKPuma(theta);
    T_6T = [    1       0       0       -0.1;
            0       1       0       0;
            0       0       1       0.13625;
            0       0       0       1];

    P = P0_corner + P_required;
```

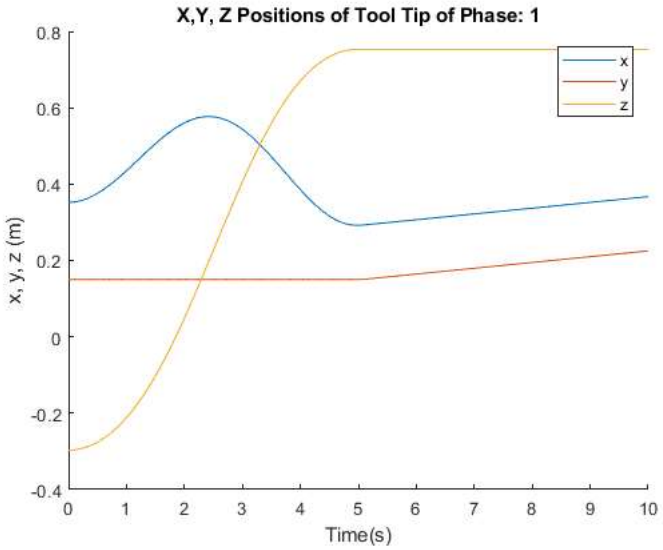
```
    if (O==1)
        M = [   0   1   0   P(1);
                0   0   1   P(2);
                1   0   0   P(3)
                0   0   0   1];
    elseif (O==2)
        M = [   -1   0   0   P(1);
                0   1   0   P(2);
                0   0   -1   P(3)
                0   0   0   1];
    elseif (O==3)
        M = [   0   0   -1   P(1);
                -1   0   0   P(2);
                0   1   0   P(3)
                0   0   0   1];
    end



    T_required = M*(T_6T^(-1));

    [R_required, P_required] = tr2rt(T_required);
    [R_actual, P_actual] = tr2rt(T_actual);
    posError = sqrt(sum((P_actual - P_required) .^ 2));
    orientError = sqrt(sum((R_actual - R_required) .^ 2, 'all'));
end

function [m0, m1, m2, m3] = cubicTrajectory(theta0, theta1, n)

    tf = n;
    m0 = theta0;
    m1 = 0;
    m2 = (3/tf^2)*(theta1 - theta0);
    m3 = (-2/tf^3)*(theta1 - theta0);
end

function [T] = transformationMatrix(DH_row)

 T = [cos(DH_row(4))                    -sin(DH_row(4))                    0                  DH_row(2);
      sin(DH_row(4))*cos(DH_row(1))     cos(DH_row(4))*cos(DH_row(1))     -sin(DH_row(1))    -sin(DH_row(1))*DH_row(3);
      sin(DH_row(4))*sin(DH_row(1))     cos(DH_row(4))*sin(DH_row(1))     cos(DH_row(1))     cos(DH_row(1))*DH_row(3);
      0                                 0                                 0                  1];
end

function [wrapped_angle] = wrapToHalfPi(lambda)

    tmp = mod(lambda+pi/2,pi);
    wrapped_angle = tmp+pi*(lambda>0&tmp==0)-pi/2;
end
```

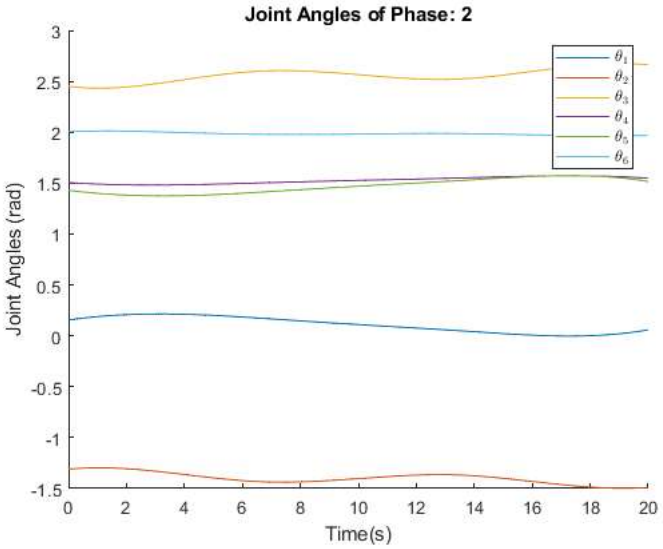
```
T_initial =

   -1.0000   -0.0000    0.0000    0.4521
    0.0000    1.0000    0.0000    0.1501
   -0.0000    0.0000   -1.0000   -0.4331
         0         0         0    1.0000


ans =

    0.0000   -0.0000    1.0000    0.4115
    0.0000    1.0000    0.0000    0.1501
   -1.0000    0.0000    0.0000    0.4331
         0         0         0    1.0000


T0_temp =

    0.3827   -0.0000    0.9239    0.3924
    0.0000    1.0000    0.0000    0.1501
   -0.9239    0.0000    0.3827    0.6163
         0         0         0    1.0000


T0 =

         0    1.0000         0    0.3924
         0         0    1.0000    0.1501
    1.0000         0         0    0.6163
         0         0         0    1.0000


ans =

   -0.0000   -1.5708    2.7489    1.5708    1.5708   -1.1781
```

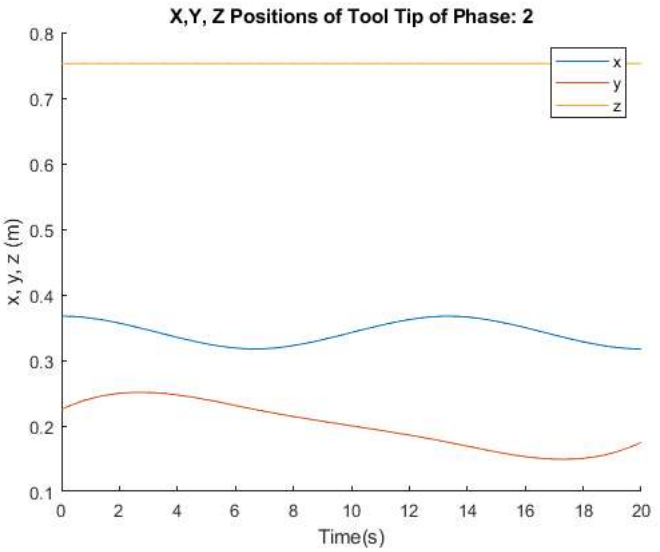
```
   -0.0000   -0.4369    0.4864    1.5708    1.5708   -0.0494
   -2.4108   -2.7047    2.7489    0.8405    1.5413    0.0329
   -2.4108   -1.5708    0.4864    1.1741   -0.9397   -0.9534


T0_corner =

         0    1.0000         0    0.3924
         0         0    1.0000    0.2863
    1.0000         0         0    0.5163
         0         0         0    1.0000


theta1 =

   -0.0000   -1.5708    2.7489    1.5708    1.5708   -1.1781


theta_corner =

   -0.0000   -1.5708    2.7489    1.5708    1.5708    1.9635

The amount of via points used in this Phase is: 100The amount of via points used in this Phase is: 200The amount of via points used in this Phase is: 150The amount
```

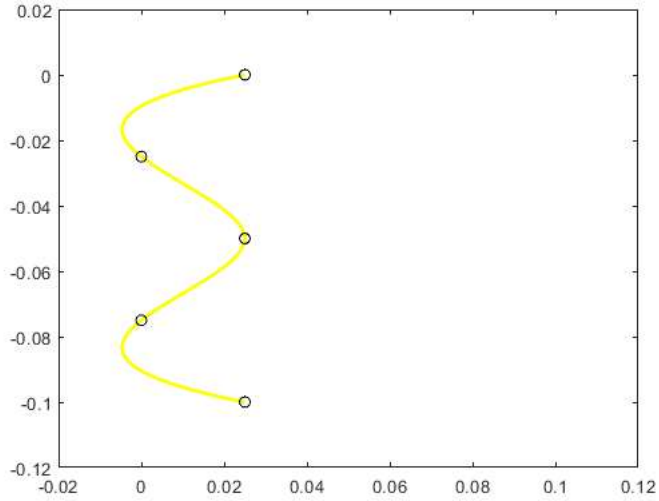


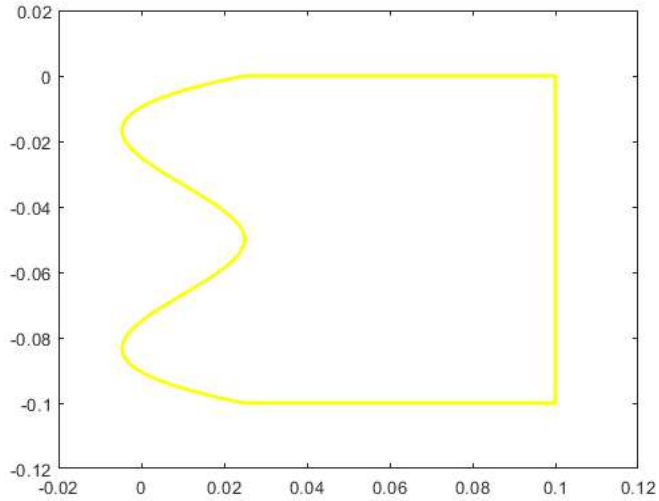

Joint Angles of Phase: 1
Joint Angles (rad)
Time(s)
θ_1
θ_2
θ_3
θ_4
θ_5
θ_6

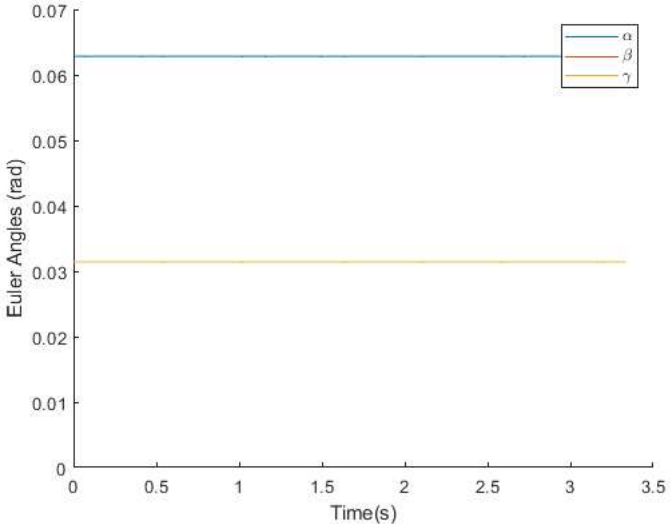

X,Y, Z Positions of Tool Tip of Phase: 1
x, y, z (m)
Time(s)
x
y
z

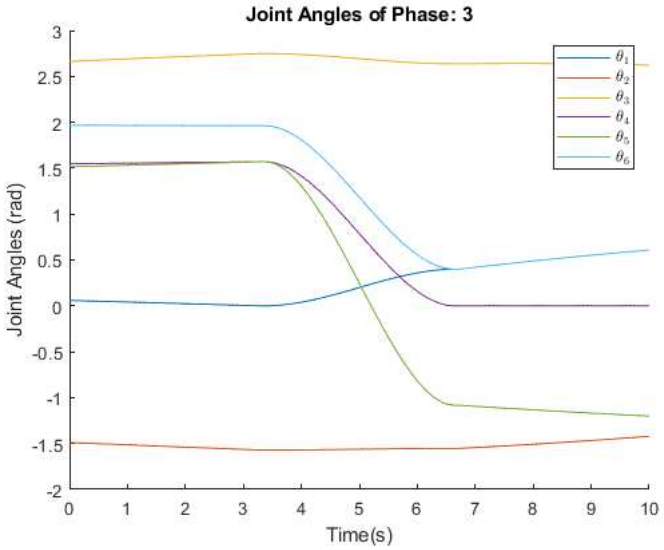

Joint Angles of Phase: 2
Joint Angles (rad)
Time(s)
3
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
0
2
4
6
8
10
12
14
16
18
20
θ_1
θ_2
θ_3
θ_4
θ_5
θ_6

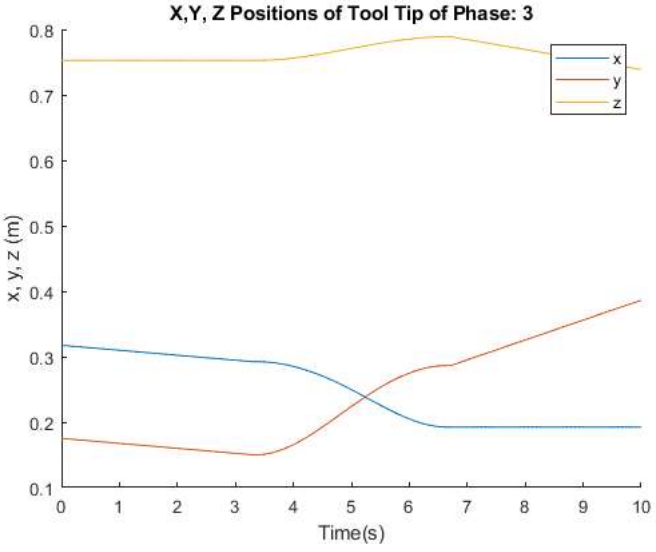

X,Y, Z Positions of Tool Tip of Phase: 2
x
y
z
x, y, z (m)
Time(s)
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
2
4
6
8
10
12
14
16
18
20

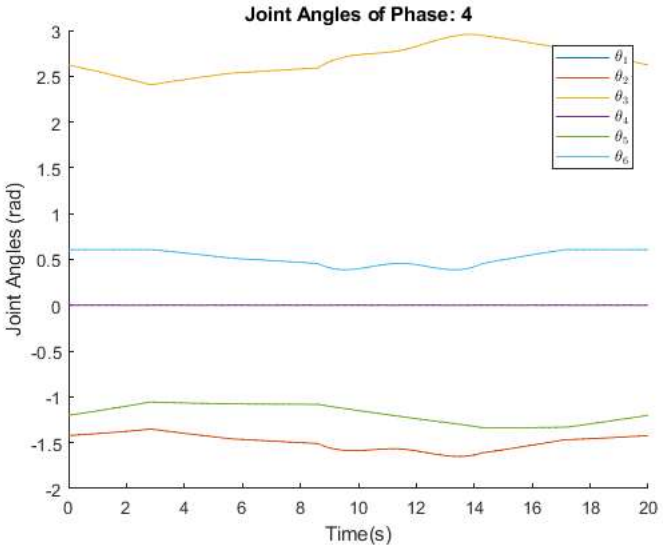

0.02
0
-0.02
-0.04
-0.06
-0.08
-0.1
-0.12
-0.02
0
0.02
0.04
0.06
0.08
0.1
0.12

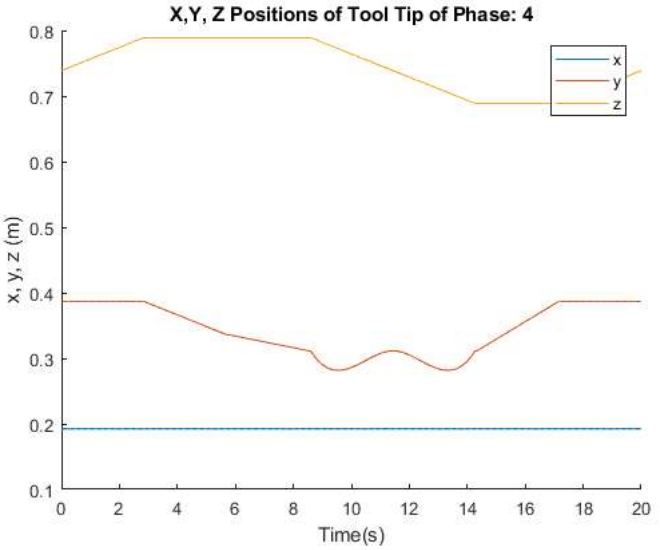

0.02
0
-0.02
-0.04
-0.06
-0.08
-0.1
-0.12
-0.02
0
0.02
0.04
0.06
0.08
0.1
0.12

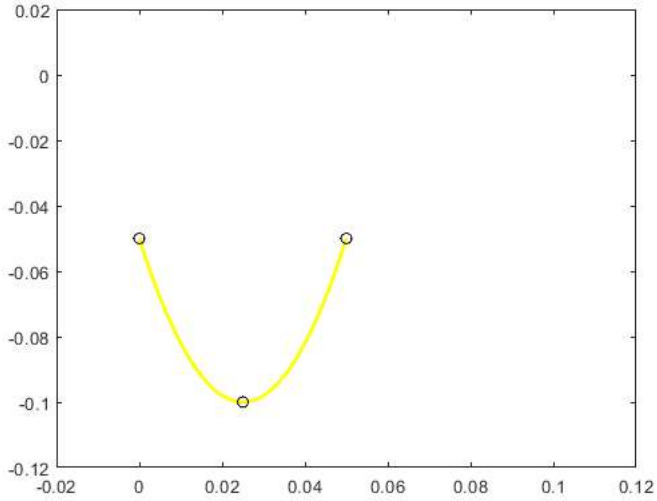

0.07
0.06
0.05
0.04
0.03
0.02
0.01
0
Euler Angles (rad)
0
0.5
1
1.5
2
2.5
3
3.5
Time(s)
α
β
γ

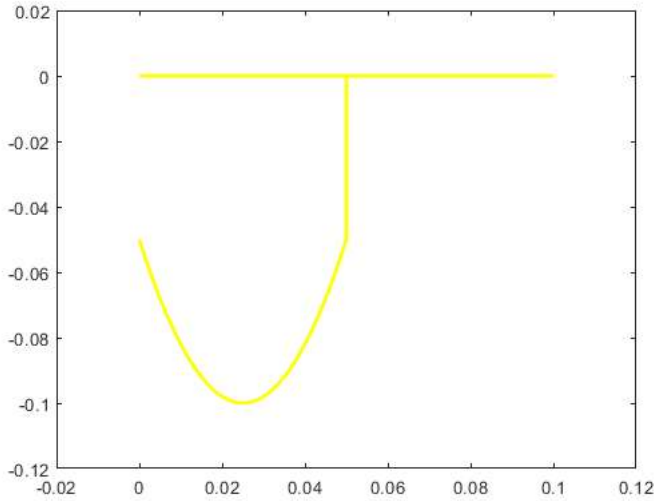

Joint Angles of Phase: 3
Joint Angles (rad)
Time(s)
θ_1
θ_2
θ_3
θ_4
θ_5
θ_6

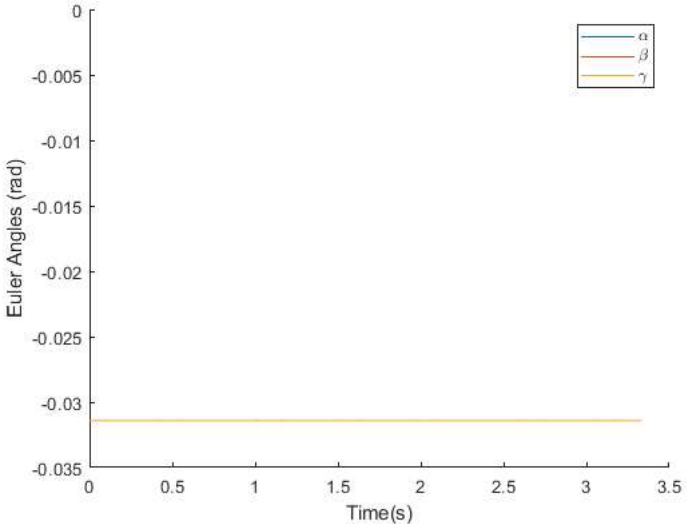

X,Y, Z Positions of Tool Tip of Phase: 3
x, y, z (m)
Time(s)
x
y
z

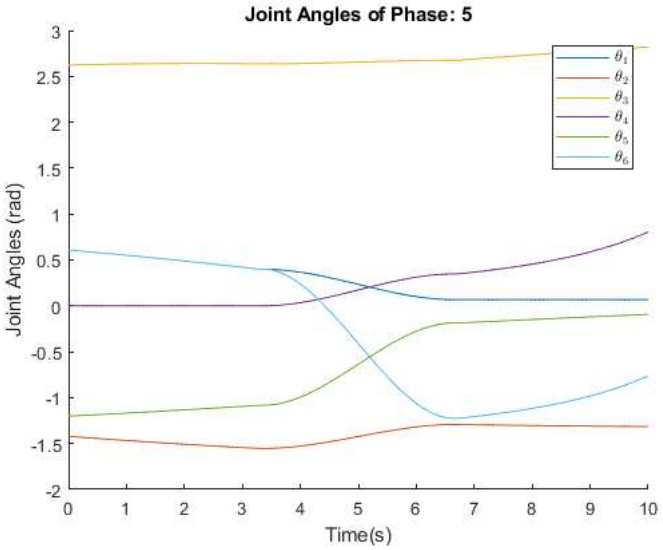

Joint Angles of Phase: 4
Joint Angles (rad)
Time(s)
3
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
0
2
4
6
8
10
12
14
16
18
20
θ_1
θ_2
θ_3
θ_4
θ_5
θ_6

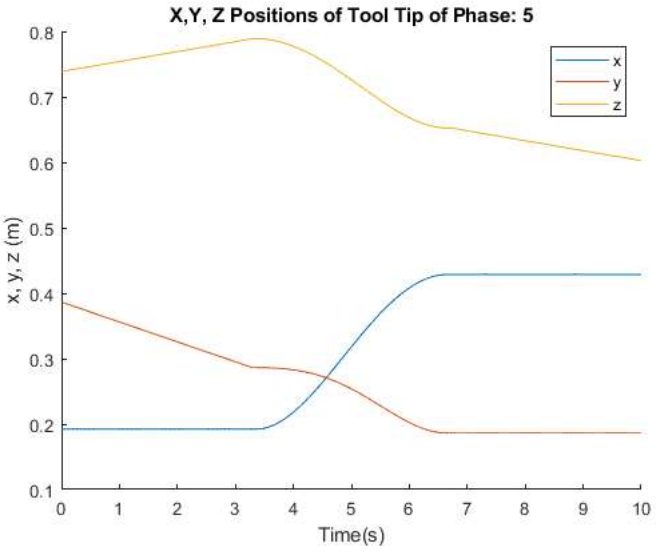

X,Y, Z Positions of Tool Tip of Phase: 4
x
y
z
x, y, z (m)
Time(s)

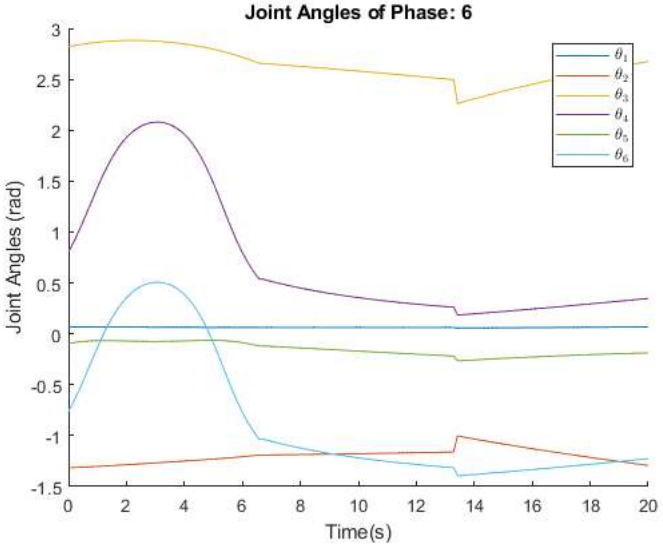


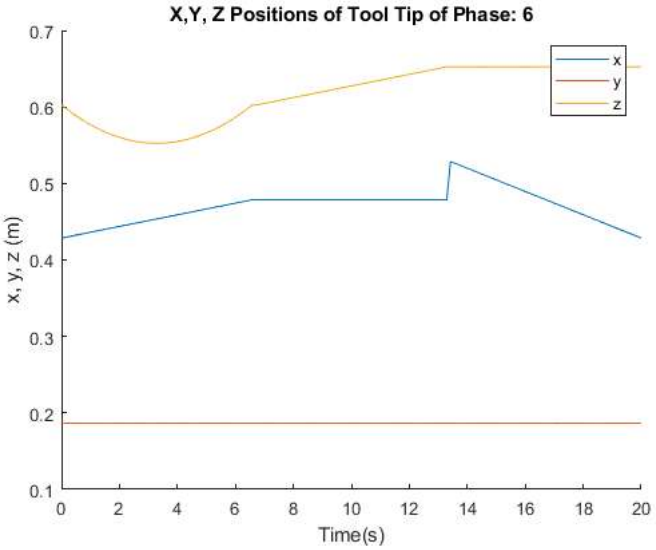


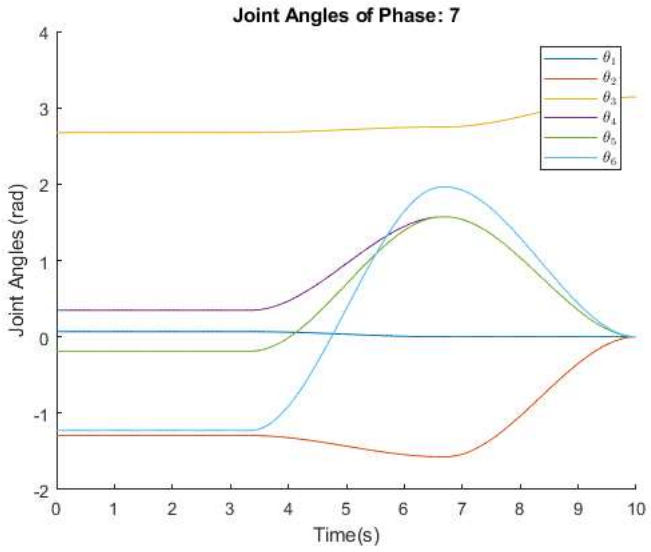

0
-0.005
-0.01
-0.015
-0.02
-0.025
-0.03
-0.035
Euler Angles (rad)
0
0.5
1
1.5
2
2.5
3
3.5
Time(s)
α
β
γ

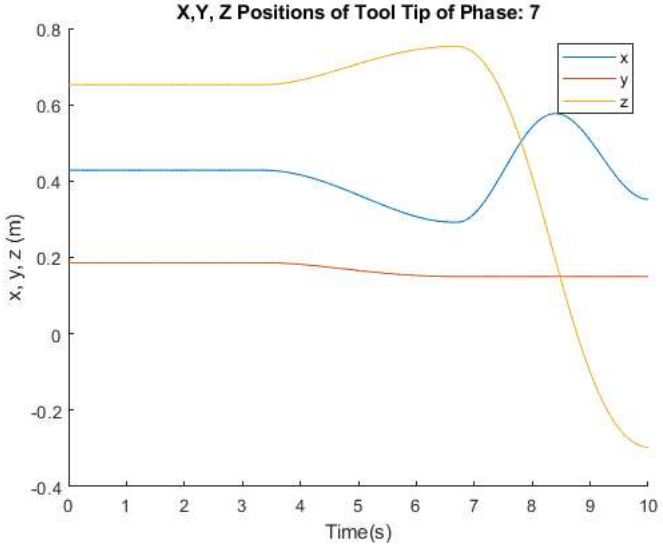

Joint Angles of Phase: 5
Joint Angles (rad)
Time(s)
θ_1
θ_2
θ_3
θ_4
θ_5
θ_6

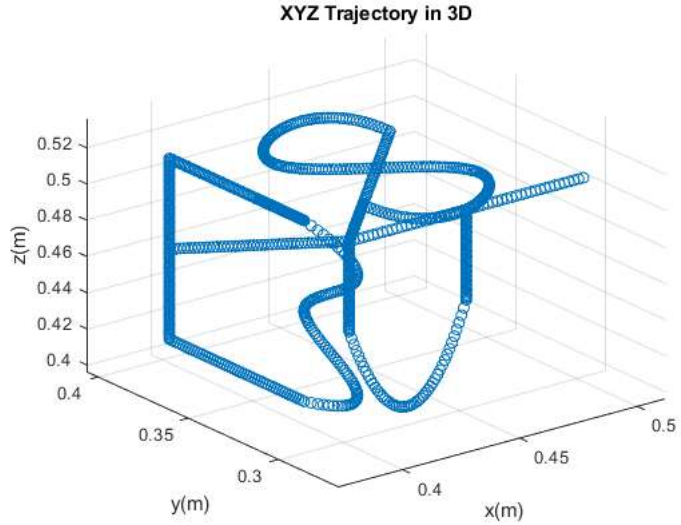

X,Y, Z Positions of Tool Tip of Phase: 5
x, y, z (m)
Time(s)
x
y
z

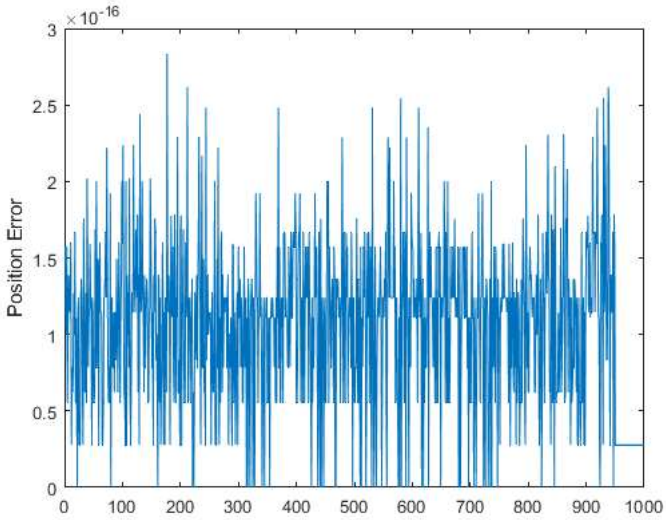

Joint Angles of Phase: 6
Joint Angles (rad)
Time(s)
θ_1
θ_2
θ_3
θ_4
θ_5
θ_6
3
2.5
2
1.5
1
0.5
0
-0.5
-1
-1.5
0
2
4
6
8
10
12
14
16
18
20

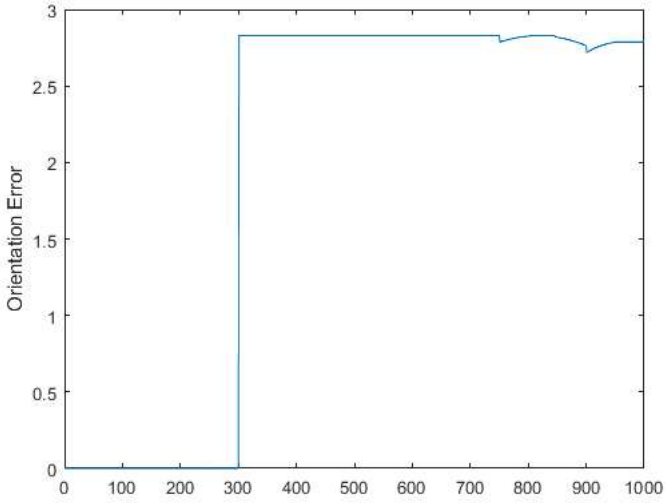

X,Y, Z Positions of Tool Tip of Phase: 6
x
y
z
x, y, z (m)
Time(s)
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
2
4
6
8
10
12
14
16
18
20


Joint Angles of Phase: 7
θ_1
θ_2
θ_3
θ_4
θ_5
θ_6
Joint Angles (rad)
Time(s)
4
3
2
1
0
-1
-2
0
1
2
3
4
5
6
7
8
9
10

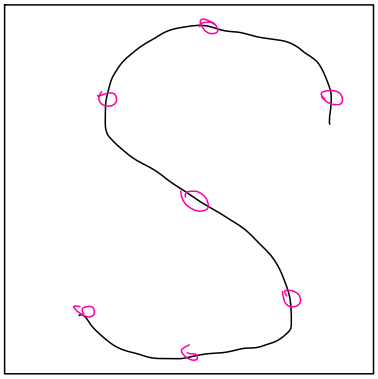

X,Y, Z Positions of Tool Tip of Phase: 7
x
y
z
x, y, z (m)
Time(s)
0.8
0.6
0.4
0.2
0
-0.2
-0.4
0
1
2
3
4
5
6
7
8
9
10

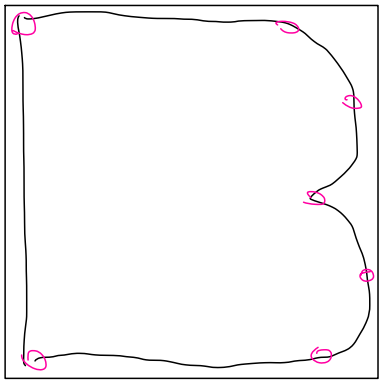

XYZ Trajectory in 3D
z(m)
y(m)
x(m)
0.52
0.5
0.48
0.46
0.44
0.42
0.4
0.4
0.35
0.3
0.4
0.45
0.5

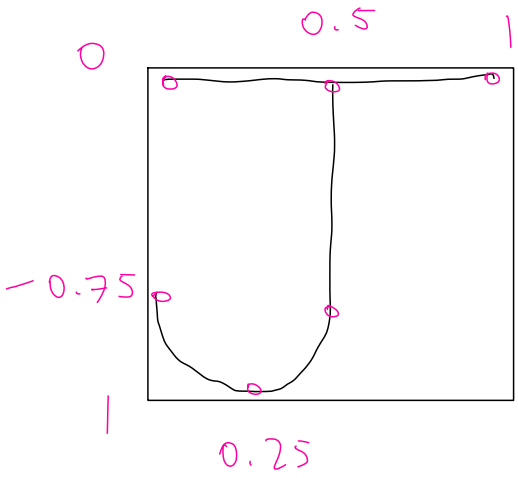




Published with MATLAB® R2022b






0
0.5
1
0.75
1
0.25