Robotics 1: Tutorial #3 Submission

Completed by: Jenna Kelly Luchak
CID: 01429938

November 21, 2017

Question 1

The pictures presented in Figure 1 below are diagrams of the Baxter Robot arm. The Baxter robot possesses seven degrees of freedom; each joint is labeled for added reference and clarity to this report.

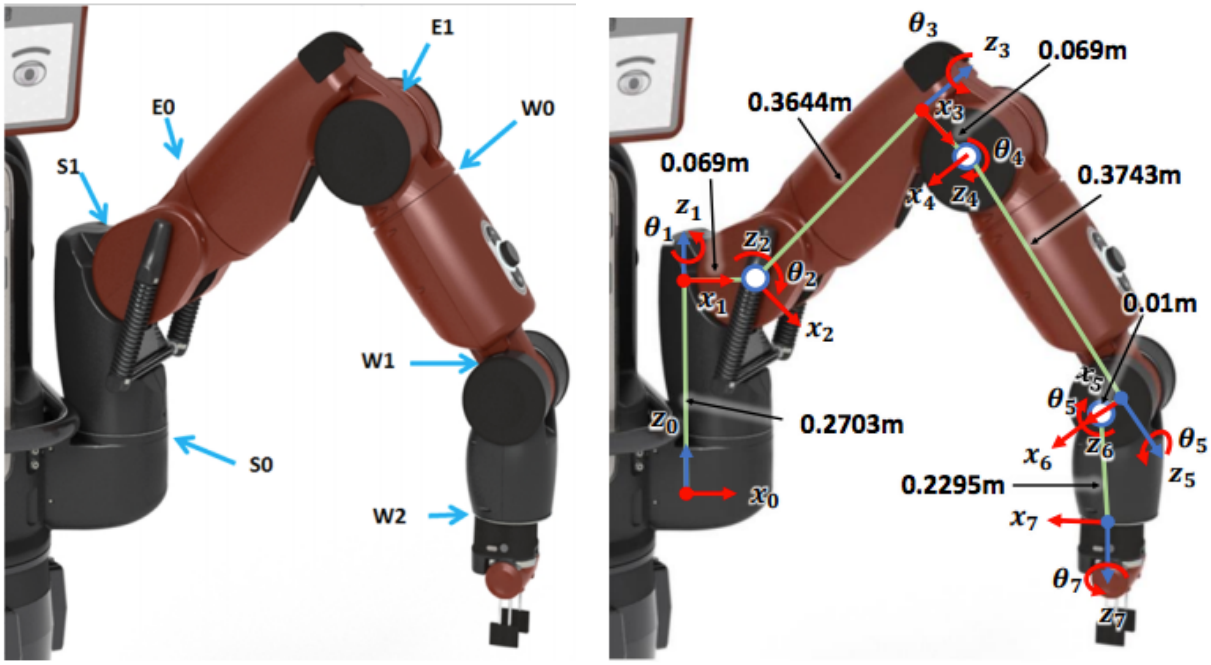


Figure 1: [Left] A diagram of the 7 degrees of freedom of the Baxter robot arm. All seven joints are labeled accordingly. [Right] The reference frames of the Baxter robot arm, where θ_7 is the end-effector.

The point of this assignment was to examine the joint angle profiles and end-effector speeds and position of the robot. Initial joint configuration data was given for this assignment and can be seen in Table 1 below.

Table 1: Angle Ranges and D-H Data of Baxter Robot Arm

Joint Label	Angle Ranges [Rad]		Joint number i	a_i [m]	α_i [Rad]	d_i [m]	Joint Angle θ_i [Rad]
	Min	Max					
S0	-1.7016	1.7016	1	0	0	0.2703	θ_1
S1	-2.147	1.047	2	0.069	-1.571	0	θ_2
E0	-3.0541	3.0541	3	0	1.571	0.3644	θ_3
E1	-0.05	2.618	4	0.069	-1.571	0	θ_4
W0	-3.059	3.059	5	0	1.571	0.3743	θ_5
W1	-1.5707	2.094	6	0.01	-1.571	0	θ_6
W2	-3.059	3.059	7	0	1.571	0.2295	θ_7

a) The homogenous transformation matrix, the transformation from joint S0 to W2, was solved using the data given in Table 1. A MatLab script, located in the Appendix of this report, details the rules used to compute the transformation

matrix. The homogenous transformation matrix was computed as a product of seven adjacent joint pairing transformation matrices.

b) The Jacobian matrix is a 3 x 7 matrix, dependent on the seven joint angles. The Jacobian was computed by differentiating the x, y and z components of the position vector with respect to the seven joint angles of the robot arm. The position vector was defined by taking the first three rows of the last column of the transformation matrix computed in question 1a. For the step-by-step details of this calculation see the script file in the Appendix of this report.

c)/d) The joint angle profiles, end-effector speed, and end-effector position of the Baxter robot arm were determined using the Jacobian matrix and a series of equations, detailed in the Appendix. A plot of the end-effector position profile, question 1c, is shown in Figure 2 below. While, plots of the joint angle profiles and end effector speed profiles, question 1d, can be seen in Figure 3 below Please note that the sampling size used in this assignment was 0.002 seconds for a total time span of 4 seconds.

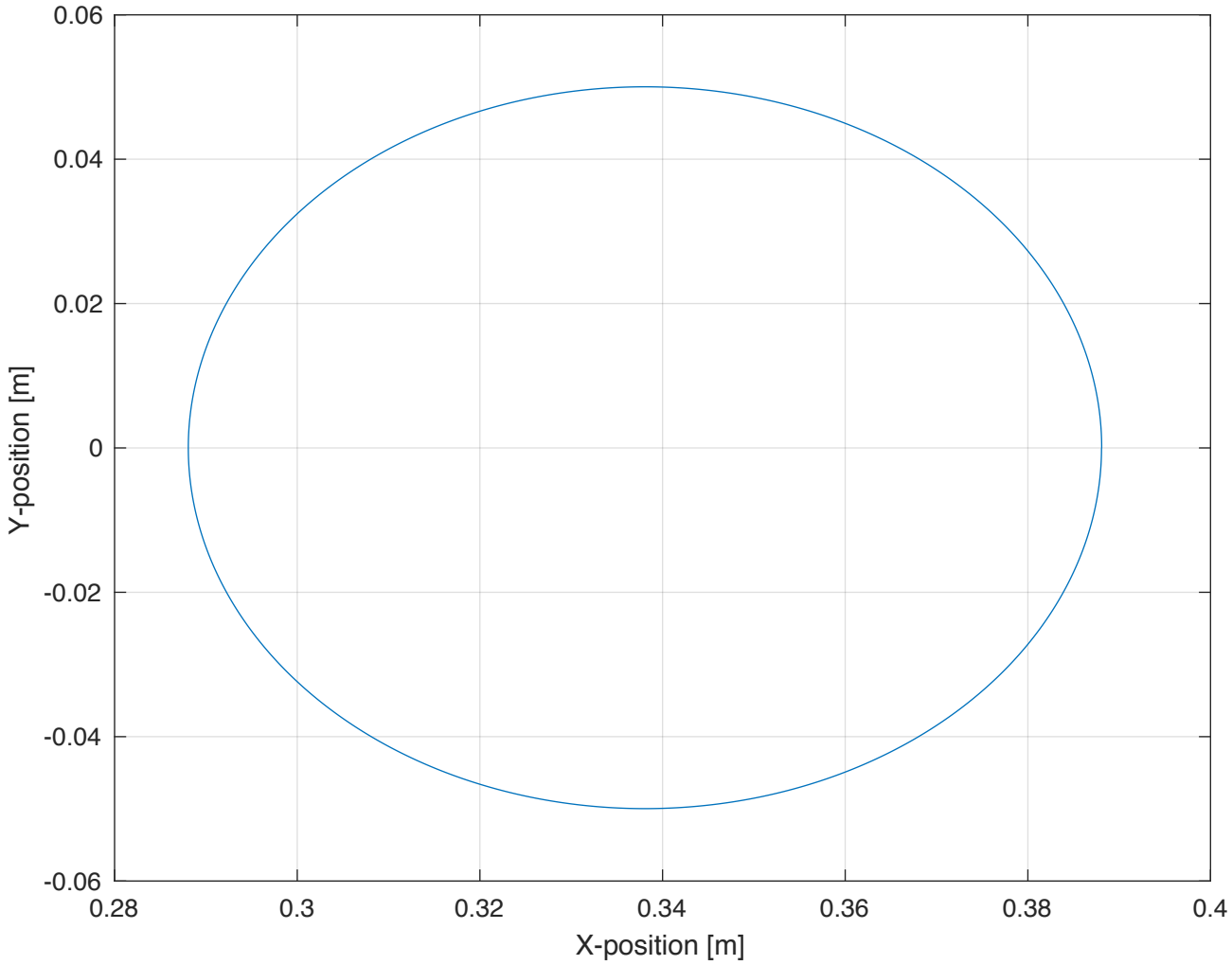


Figure 2: A plot of the Baxter robot arms end-effector position in the xy plane.

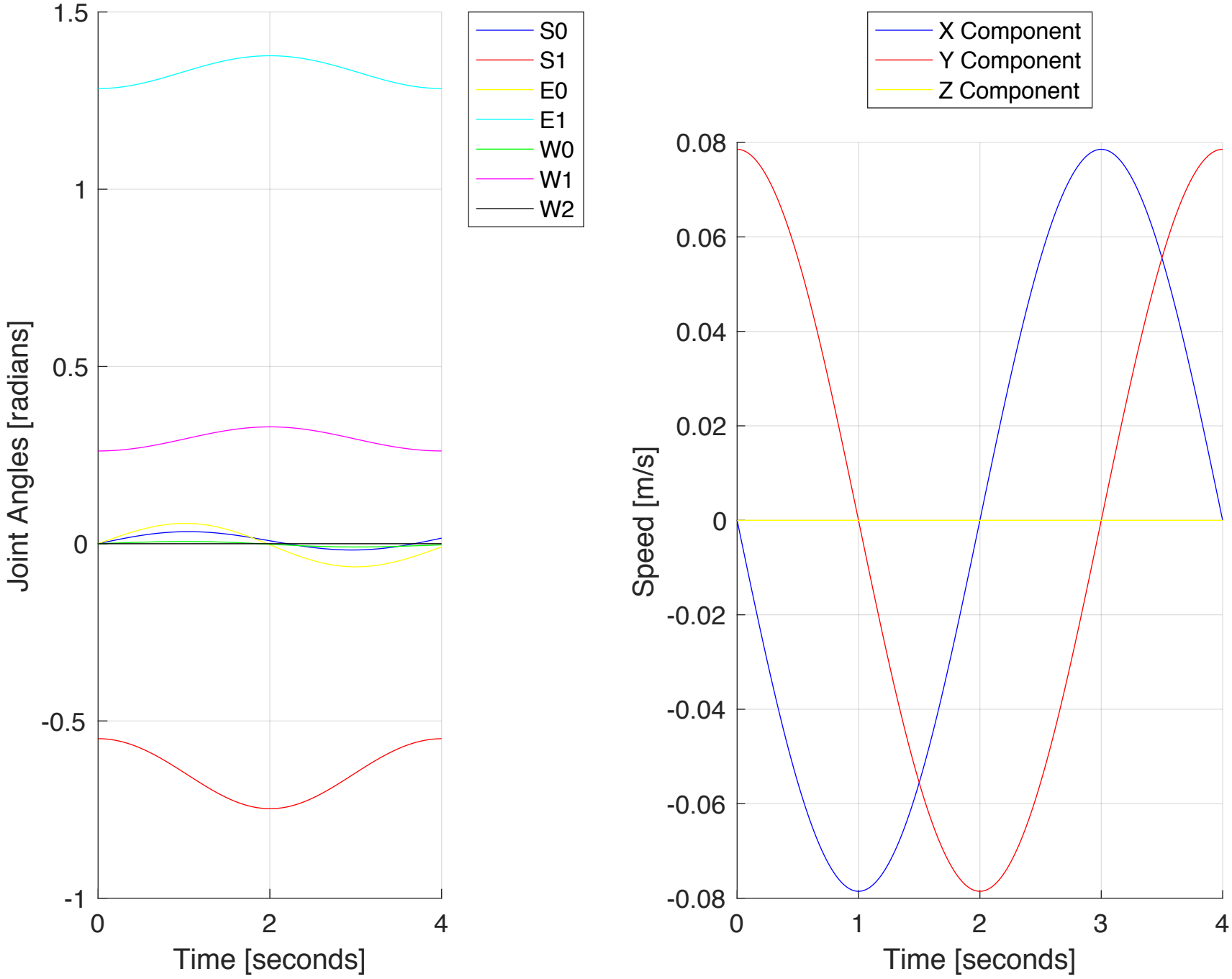


Figure 3: [Left] A plot of the Baxter robot arms seven joint angle profiles with respect to time. [Right] A plot of the end-effector speed, dissected into x, y and z components with respect to time.

As can be seen by the position plot in Figure 2, the end-effector moves in a circular pattern of uniform radii 0.05m.

As can be seen by the joint angle profiles in Figure 3, all of the joints demonstrate a sinusoidal pattern of minimal amplitude, with the exception of joint W2, which remains constant at zero radians. Observably, S1 has the largest amplitude. The speed profiles, in Figure 3 as well, show that the x and y components of speed exhibit a sinusoidal pattern of similar amplitude and varying phase, while the z-component remains constant at zero.

The information demonstrated in figures 2 and 3 show that the end-effector is not moving in the z-direction and moves with circular precision in the xy plane. This is expected, because in a robot, the end-effector will be used to control a tool or instrument and precision is needed for that task.

e) The initial joint angle configuration was provided in this assignment and can be seen in equation 1, where as I chose the new joint configuration as follows in equation 2.

$$q_i = LowerBound + 0.5 * Range \quad (1)$$

$$q_i = LowerBound + 1.0 * Range \quad (2)$$

Where the lower bound refers to the minimum joint angle and the range is the maximum joint angle subtracted by it's corresponding minimum value, as seen in Table 1 above.

A figure showing the joint angle profiles of the original and new initial configurations are presented side by side in Figure 4 below.

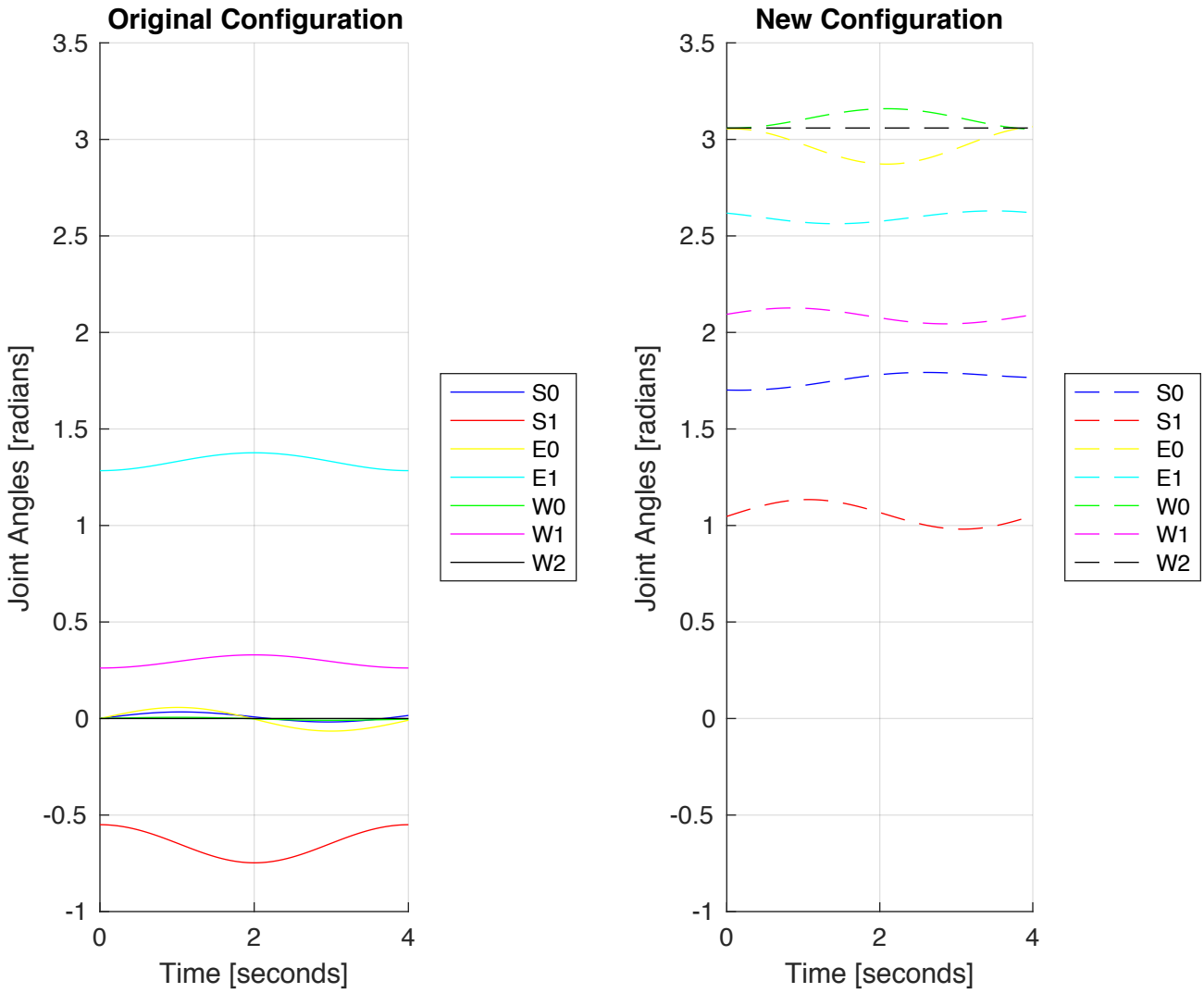


Figure 4: [Left] A plot of the joint angular profile with respect to time of the Original initial configuration. [Right] A plot of the joint angular profile with respect to time of the New initial configuration.

The new, initial configuration differs from the original configuration by an addition term of 0.5*range. Therefore, when both angular profiles are plotted side-by-side, different joint sensitivities can be observed.

For example, observe the differences in joint order from minimum to maximum. In the original configuration, joint S1 had the lowest angular values, and then S0, E0, W0 and W2 had the next highest values, followed by W1 and lastly E1 had the largest value. However, in the new configuration, the order appears slightly different. Joint S1 is still the minimum, however, E1 is no longer the largest, and instead, E0, W0 and W2 supersede it. This suggests that joints E0, W0 and W2 have higher sensitivities than the other joints.

In order to properly explain joint sensitivities, a quantitative look at the angular differences between configurations should be explored. Figure 5 below presents a bar graph detailing the average and maximum calculated differences between each joints configurations angular profile.

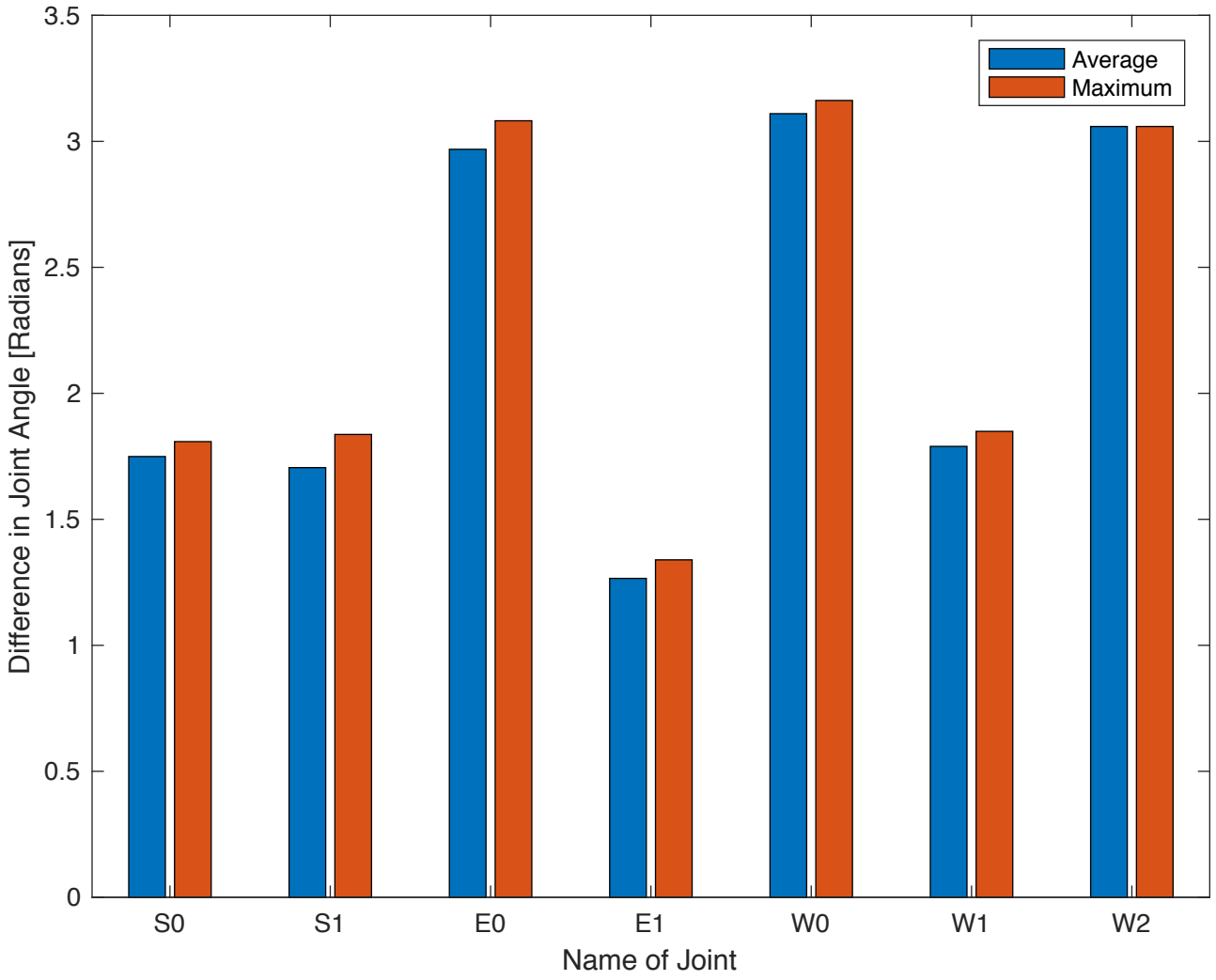


Figure 5: A plot of bar graphs documenting the average and maximum angular differences between the Original and New configuration angular profiles.

Based on the information from Figure 5, the group of joints S0, S1 and W1 as well as the group of E0, W0 and W2 have similar bar heights. This shows that the joints in each group have similar sensitivities. Joint E1 has the lowest sensitivity to changes in initial configurations, as it has the smallest quantitative difference. The joints E0, W0 and W2 tie for the highest sensitivity to changes in initial configuration, where as joints S0, S1 and W1 tie for the second highest sensitivity.

Figures 4 and 5 both suggest that joints E0, W0 and W2 have the highest sensitivities to changes in initial configurations.

Now that we have analyzed how the angular profiles are affected when a new initial configuration is introduced, lets look at the speed and position profiles. Figure 6 and 7 below present plots of the end-effector speed with respect to time and the end-effector position, respectively.

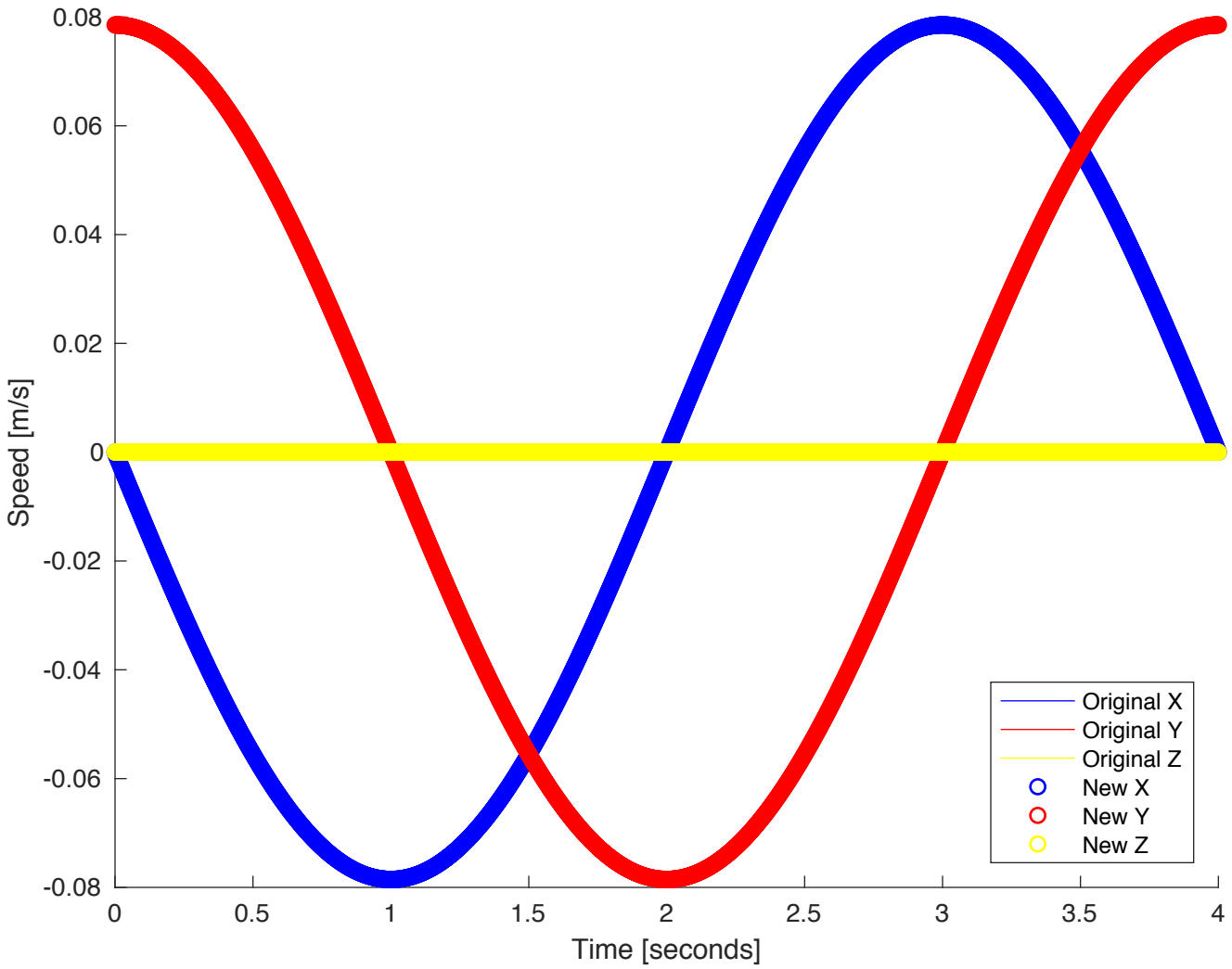
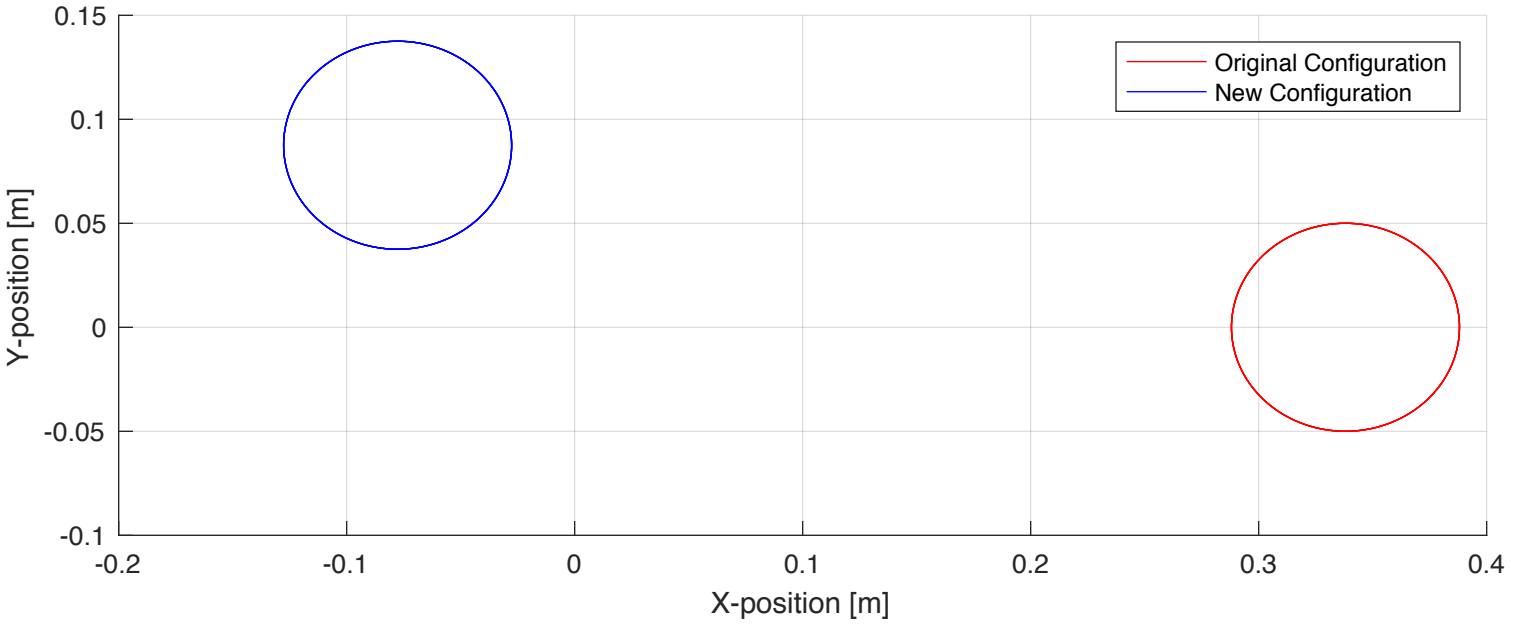
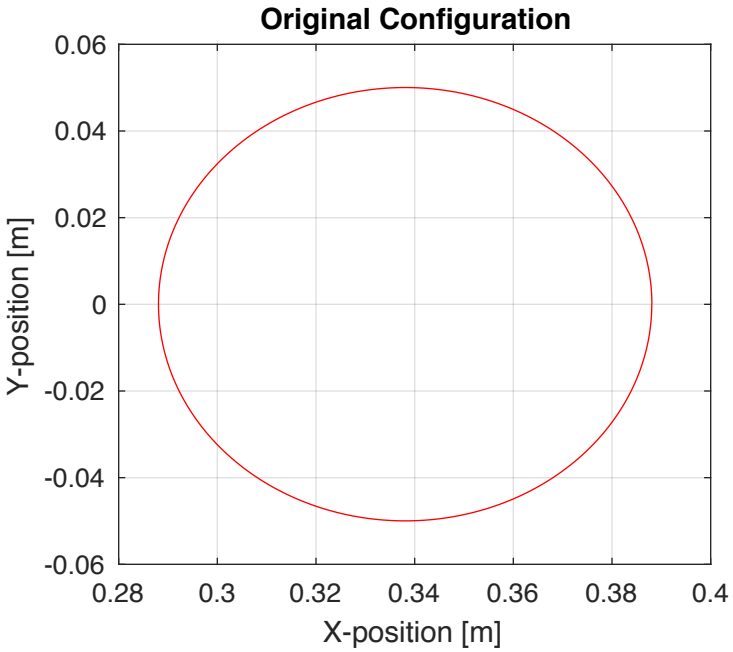


Figure 6: A plot of the end-effector speed dissected into x, y and z components with respect to time for the Original and New initial configurations.





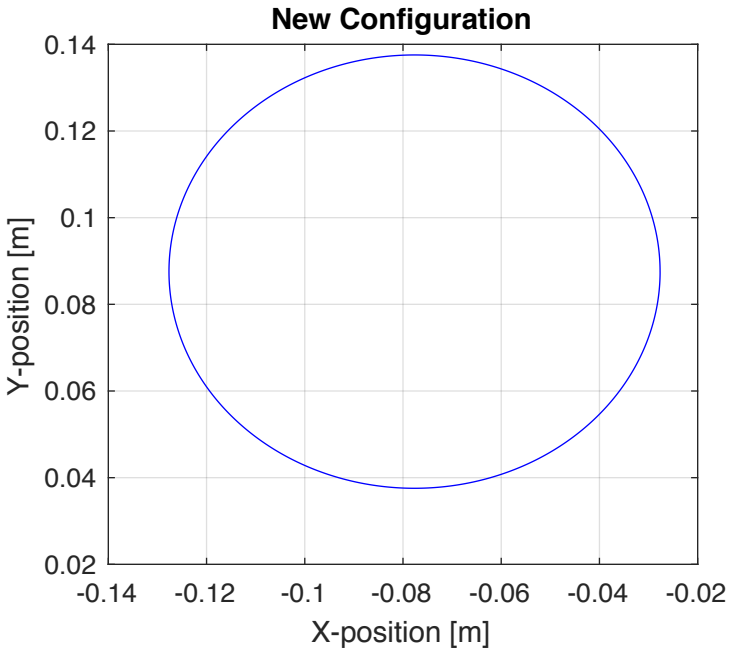


Figure 7: [Top] A plot of the end-effector position in the xy plane of the Original and New initial configurations. [Bottom Left] A plot of the end-effector position in the xy plane of the Original initial configuration. [Bottom Right] A plot of the end-effector position in the xy plane of the New initial configuration.

Figure 6, shows that changing the initial configuration have no affects on the end-effector speed, as the plots for both the original and new configurations are equivalent. However, Figure 7 shows that changing the initial configuration changes the resulting end-effector position, but it does not affect the size or shape of the tracer, as it is still a circle of radii 0.05m. These results are expected as the end-effector joint, W2 was found to have a high sensitivity, therefore, since the joints angles were affected, the position should be altered in space.

Question 2/Question 3

My answers for question 1C (Figure 2) and question 1D (Figure 3) were checked against the figures given in this assignment package using Plot Digitizer. Plot Digitizer is an application that allows you to trace a plot from any figure and export it into an excel file of data. I took this export and plotted it over my graphs on MatLab. The results are as follows below. My check for question 1C is in Figure 8 and my check for question 1D are in Figures 9 and 10.

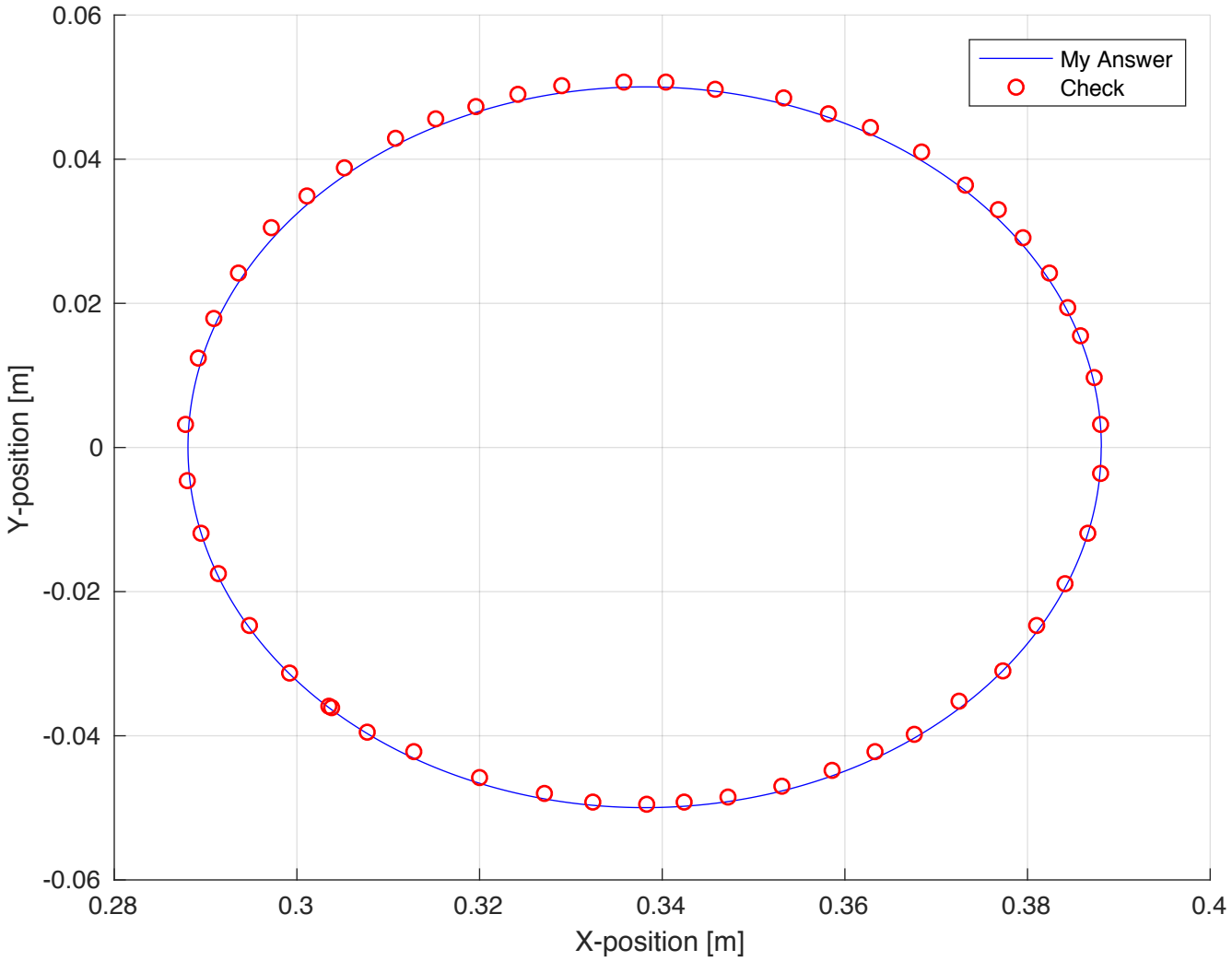


Figure 8: The correct end-effector position profiles plotted with my end-effector position result from MatLab.

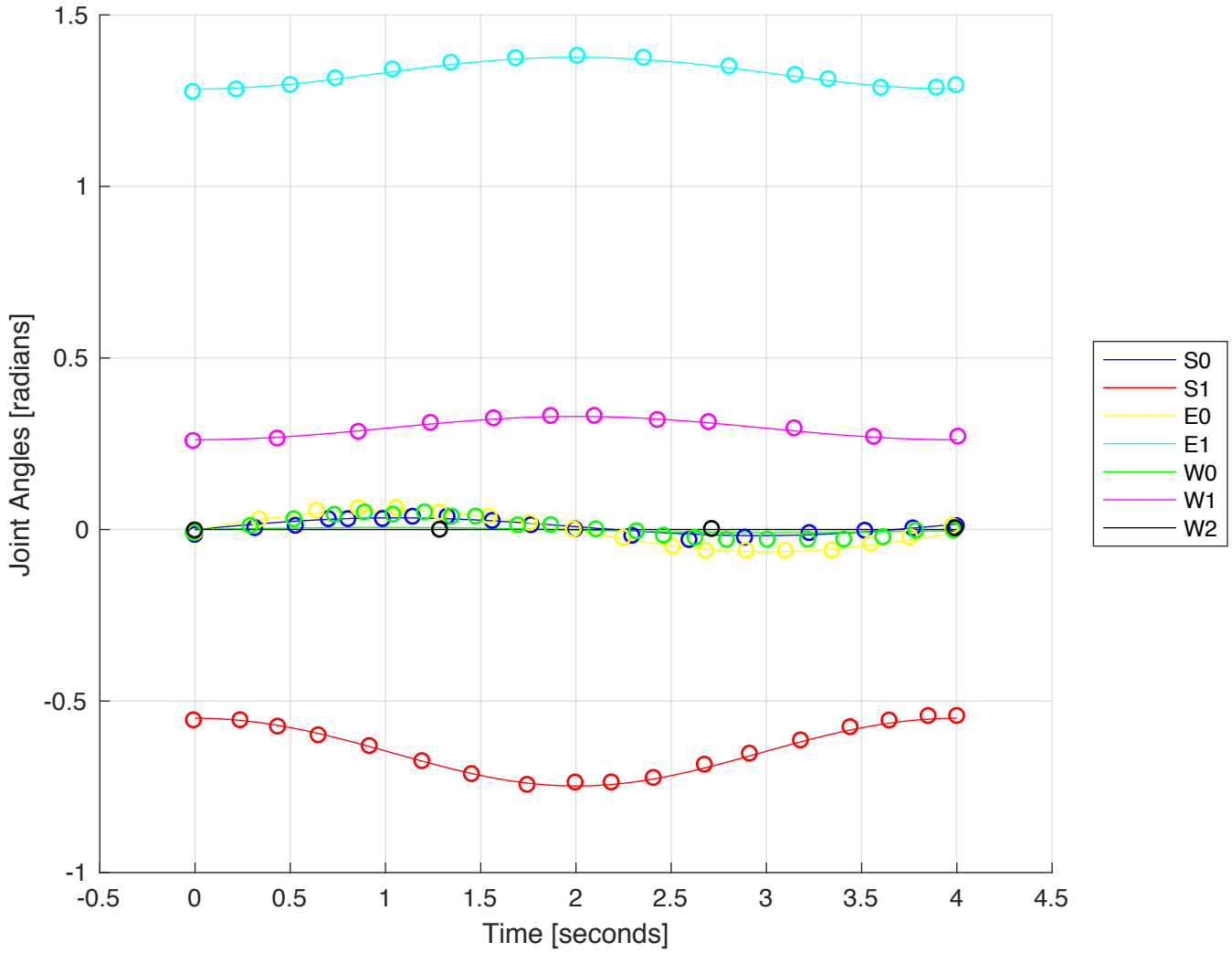


Figure 9: The correct angular joint profiles (circular data points) with my MatLab result (solid lines).

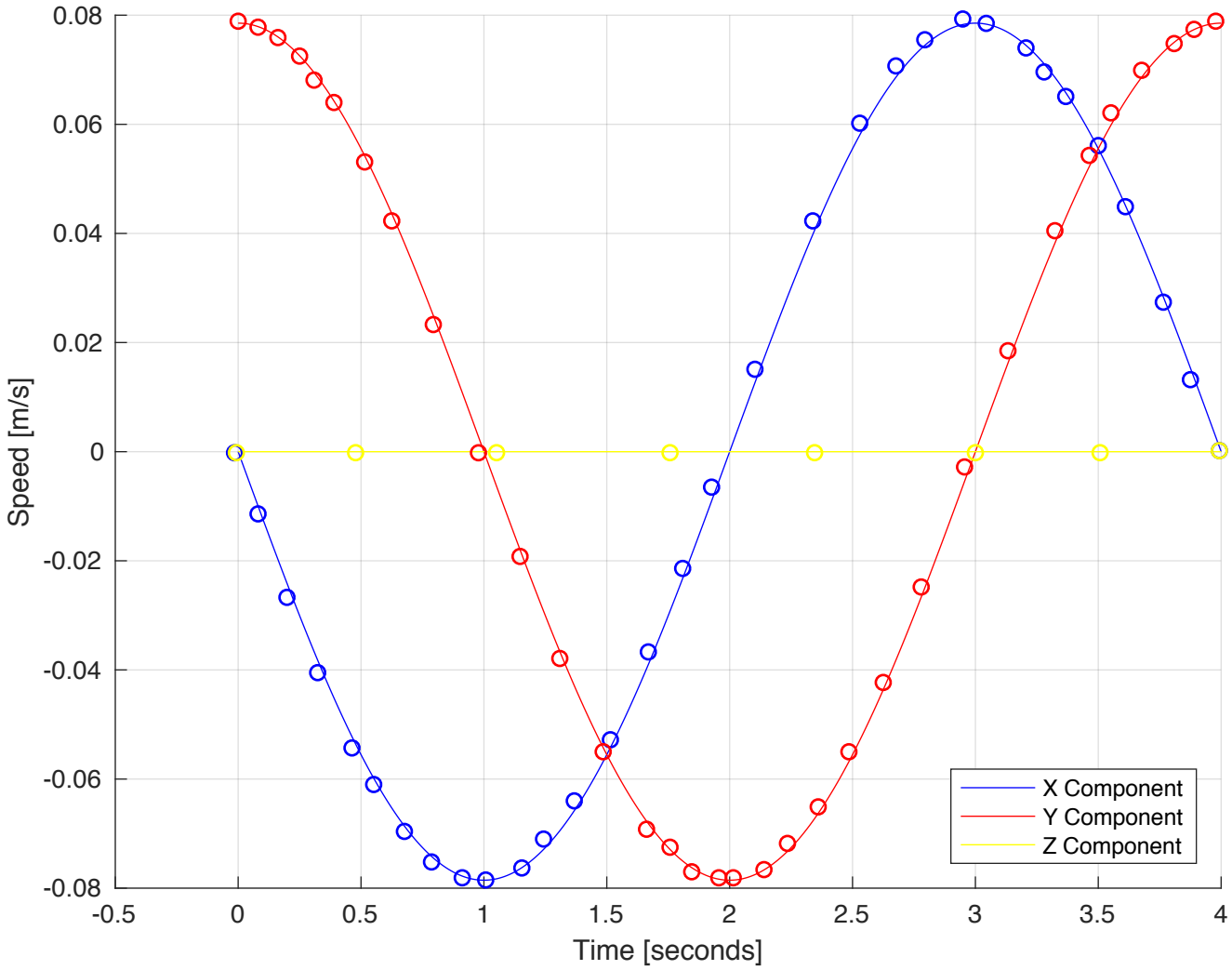


Figure 10: The correct end-effector speed profile (circular data points) with my MatLab result (solid lines).

Based on the given plots from Figures 8, 9 and 10, my answers from questions 1C and 1D match the given plots exceptionally well. They are therefore acceptable answers.

Appendix

The MatLab code for this assignment can be seen as follows.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This script was prepared by:
% Jenna Luchak
% CID: 01429938
% For Robotics 1: Tutorial #3
% November 21, 2017
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Clear Workspace and Command Window

clc;
clear;

%% Define the time constraints of the project

ts=0.002; % Time step in seconds
time=4; % Total time duration in seconds
t=0:ts:time; % Time Vector

%% Define the symbols of the script

syms theta_1 theta_2 theta_3 theta_4 theta_5 theta_6 theta_7;

%% Define the Angle Ranges of the Baxter Robot
% i.e. [Min ; Max] all values in radians

S_0 = [-1.7016 1.7016];
S_1 = [-2.147 1.047];
E_0 = [-3.0541 3.0541];
E_1 = [-0.05 2.618];
W_0 = [-3.059 3.059];
W_1 = [-1.5707 2.094];
W_2 = [-3.059 3.059];

% Combine all joint angle ranges into one matrix
angle_ranges = [S_0;S_1;E_0;E_1;W_0;W_1;W_2];

%% Define the D-H table data for the Baxter Robot arm

a = [0;0.069;0;0.069;0;0.01;0]; % Distances b/w points i-1 & i on X_i-1
axis [m]
alpha = [0;-1.571;1.571;-1.571;1.571;-1.571;1.571]; % Angles b/w points i-1
& i on X_i-1 axis [rad]
d = [0.2703;0;0.3644;0;0.3743;0;0.2295]; % Distances b/w points i-1 & i on
Z_i axis [m]
theta=[theta_1 theta_2 theta_3 theta_4 theta_5 theta_6 theta_7]; % Angles
b/w points i-1 & i on Z_i axis [m]

%% Question 1A
% Compute the Homogenous Transformation Matrix for the Baxter Arm
% from S_0 to W_2
% i.e. Calculate each adjacent joint pairings transformation matrix, store
% them into a structure array and then multiply each element together.

% Step1: Create a for loop to print each joint pairings transformaiton
% matrix into a 1x7 structure array
```

```
for i=1:1:7
    T(i).a=[   cos(theta(i))                -sin(theta(i))             0
a(i);
    sin(theta(i))*cos(alpha(i)) cos(theta(i))*cos(alpha(i)) -sin(alpha(i))
-sin(alpha(i))*d(i);
    sin(theta(i))*sin(alpha(i)) cos(theta(i))*sin(alpha(i))  cos(alpha(i))
cos(alpha(i))*d(i);
               0                            0                     0
1];
end

% Step 2: Multiply each component of the structure array together to obtain
% the homogenous transformation matrix

Transform = T(1).a*T(2).a*T(3).a*T(4).a*T(5).a*T(6).a*T(7).a;

%% Question 1B: Compute the Jacobian Matrix
% Define the position vector as the first three rows and last column of the
% previously computed Transformation matrix.

p=Transform(1:3,4); % position vector [x,y,z]

% Jacobian matrix is calculated using the position vector and Matlab's
built
% in Jacobian function with respect to the seven variables theta
Jacobian = jacobian(p,theta);

%% Define the desired end-effector position and speed profiles

desired_x = 0.05*cos(0.5*pi()*t); % x component of position
desired_y = 0.05*sin(0.5*pi()*t); % y component of position
desired_z = 0*t; % z component of position

desired_x_dot = -0.05*0.5*pi()*sin(0.5*pi()*t); % x component of speed
desired_y_dot = 0.05*0.5*pi()*cos(0.5*pi()*t); % y component of speed
desired_z_dot = 0*t; % z component of speed

% Combine all speed components into one desired velocity vector
desired_p_dot=[desired_x_dot;desired_y_dot;desired_z_dot];

%% Calculate the initial joint angle configurations of the Baxter Robot Arm

q=zeros(7,time/ts+1); % Empty matrix for joint angles
q_dot=zeros(7,time/ts+1); % Empty matrix for derivative of joint angles

for c=1:2
    % This outer "for" loop is used to repeat the calculations of all of
    % the internal equations for the number of configurations specified.

for i=1:1:7
    % This inner "for" loop is used to calculate the initial joint angles
    if c==1
    q(i,1)=angle_ranges(i,1)+(0.5*(angle_ranges(i,2)-angle_ranges(i,1))); %
Original configuration
    else
        q(i,1)=angle_ranges(i,1)+(1*(angle_ranges(i,2)-angle_ranges(i,1)));
% New configuration
    end
end
```

```
%% Calculate the initial speed and position values of the Baxter robot arm

% Redefine the symbols as the initial joint angles numeric values
theta_1 = q(1,1);
theta_2 = q(2,1);
theta_3 = q(3,1);
theta_4 = q(4,1);
theta_5 = q(5,1);
theta_6 = q(6,1);
theta_7 = q(7,1);

% Substitue the redefined theta symbols into the Jacobian
JJ_0 =subs(Jacobian);
JJJ_0=double(JJ_0); % convert the jacobian value from sym to double

% Substitute the redefined theta symbols into the position vector
PP(:,1) = subs(p);
PPP(:,1) = double(PP(:,1)); % convert the position value from sym to double

% Calculate the Psuedo Inverse of the Jacobian
Inv_Jac_0 = pinv(JJJ_0);

% Compute the initial speed and angular velocity
q_dot(:,1)=Inv_Jac_0*desired_p_dot(:,1); % angular velocity [rad/s]
speed(:,1) = JJJ_0*q_dot(:,1); % speed [m/s]

%% Calculate the entire joint angle profile, speed profile and position
% i.e. use a "for" loop to determine the actual joint angles and
% angular velocities at every time step up to the specified time.

for i=2:1:(time/ts+1)

% Update joint angle matrix
    q(:,i)=q(:,i-1)+q_dot(:,i-1)*ts;

% Redefine the symbols as numeric values
    theta_1 = q(1,i);
    theta_2 = q(2,i);
    theta_3 = q(3,i);
    theta_4 = q(4,i);
    theta_5 = q(5,i);
    theta_6 = q(6,i);
    theta_7 = q(7,i);

% Substitue the redefined symbols into the Jacobian equation
    JJ = subs(Jacobian);
    JJJ = double(JJ); % convert the jacobian value from sym to double
    PP(:,i) = subs(p);
    PPP(:,i) = double(PP(:,i)); % convert the position value from sym to
double

% Calculate the Psuedo Inverse of the Jacobian
    Inv_Jac = pinv(JJJ);

% Compute the angular velocity of the robot
    q_dot(:,i)=Inv_Jac*desired_p_dot(:,i);

% Compute the end-effector speed of the robot
    speed(:,i) = JJJ*q_dot(:,i);
```

```
% Save each configuration into a structure array
    joint_angle(c).a = q; % Joint angle Profile
    S(c).a = speed; % Speed Profile
    position(c).a = PPP; % Position Profile
end
end

%% Redefine elements of structure array into seperate matrices
% Configuration 1 elements
position_1 = position(1).a; % Position Profile
q_1 = joint_angle(1).a; % Joint Angle Profile
speed_1 = S(1).a; % Speed Profile

% Configuration 2 elements
position_2 = position(2).a; %Position Profile
q_2 = joint_angle(2).a; % Joint Angle Profile
speed_2 = S(2).a; % Speed Profile

%% Question 1C
% Print a plot of the end-effector position using the original joint angle
% configuration

figure(1);
plot(position_1(1,:),position_1(2,:));
grid on
xlabel('X-position [m]'); % x axis title
ylabel('Y-position [m]'); % y axis title

%% Question 1D
% Print a plot of the Joint angle profiles and end-effector speeds with
% respect to time and the original joint angle configuration

figure(2);
% Joint Angle Profile
subplot(1,2,1)
grid on
hold on
plot(t,q_1(1,:),'b'); % S_0
plot(t,q_1(2,:),'r'); % S_1
plot(t,q_1(3,:),'y'); % E_0
plot(t,q_1(4,:),'c'); % E_1
plot(t,q_1(5,:),'g'); % W_0
plot(t,q_1(6,:),'m'); % W_1
plot(t,q_1(7,:),'k'); % W_2
legend('S0','S1','E0','E1','W0','W1','W2','location','bestoutside');
xlabel('Time [seconds]'); % x axis title
ylabel('Joint Angles [radians]'); % y axis title
hold off

% Speed Profile
subplot(1,2,2)
grid on
hold on
plot(t,speed_1(1,:),'b'); % x component of end effector speed
plot(t,speed_1(2,:),'r'); % y component of end effector speed
plot(t,speed_1(3,:),'y'); % z component of end effector speed
legend('X Component','Y Component','Z
Component','location','northoutside');
xlabel('Time [seconds]'); % x axis title
ylabel('Speed [m/s]'); % y axis title
hold off
```

```
%% Question 1E
% Re-Plot the joint angle profiles and end-effector speeds, for both the
% original as well as a new joint angle configuration

%% Joint Angle Profiles for Question 1E

figure(3)
subplot(1,2,1)
grid on
hold on
% Plot the original configurations joint angle profiles
plot(t,q_1(1,:),'b'); % S_0
plot(t,q_1(2,:),'r'); % S_1
plot(t,q_1(3,:),'y'); % E_0
plot(t,q_1(4,:),'c'); % E_1
plot(t,q_1(5,:),'g'); % W_0
plot(t,q_1(6,:),'m'); % W_1
plot(t,q_1(7,:),'k'); % W_2
title('Original Configuration');
legend('S0','S1','E0','E1','W0','W1','W2','location','eastoutside');
xlabel('Time [seconds]'); % x axis title
ylabel('Joint Angles [radians]'); % y axis title
axis([0 4 -1.0 3.5]);

% Plot the new configuration joint angle profiles
subplot(1,2,2)
hold on
grid on
plot(t,q_2(1,:),'--b'); % S_0
plot(t,q_2(2,:),'--r'); % S_1
plot(t,q_2(3,:),'--y'); % E_0
plot(t,q_2(4,:),'--c'); % E_1
plot(t,q_2(5,:),'--g'); % W_0
plot(t,q_2(6,:),'--m'); % W_1
plot(t,q_2(7,:),'--k'); % W_2
title('New Configuration');
legend('S0','S1','E0','E1','W0','W1','W2','location','eastoutside');
xlabel('Time [seconds]'); % x axis title
ylabel('Joint Angles [radians]'); % y axis title
axis([0 4 -1.0 3.5]);

%% Position Profile plots for Question 1E

figure(4)
% Both configurations on one plot
subplot(2,2,1:2)
hold on
plot(position_1(1,:),position_1(2,:),'r'); % Original Configuration
plot(position_2(1,:),position_2(2,:),'b'); % New configuration
grid on
axis([-0.2 0.4 -0.1 0.15]);
xlabel('X-position [m]'); % x axis title
ylabel('Y-position [m]'); % y axis title
legend('Original Configuration','New
Configuration','location','northeast');

% Plot each configuration one at a time for clarity
% Original
subplot(2,2,3)
plot(position_1(1,:),position_1(2,:),'r'); % Original Configuration
```

```
grid on
xlabel('X-position [m]'); % x axis title
ylabel('Y-position [m]'); % y axis title
title('Original Configuration');
% New
subplot(2,2,4)
plot(position_2(1,:),position_2(2,:),'b'); % New configuration
grid on
title('New Configuration');
xlabel('X-position [m]'); % x axis title
ylabel('Y-position [m]'); % y axis title


%% Speed Profile plots for Question 1E

figure(5)
hold on
% Plot the original configurations end-effector speeds on figure 3
plot(t,speed_1(1,:),'b'); % x component of end effector speed
plot(t,speed_1(2,:),'r'); % y component of end effector speed
plot(t,speed_1(3,:),'y'); % z component of end effector speed

% Plot the new configurations end-effector speeds on figure 3
plot(t,speed_2(1,:),'ob'); % x component of end effector speed
plot(t,speed_2(2,:),'or'); % y component of end effector speed
plot(t,speed_2(3,:),'oy'); % z component of end effector speed

legend('Original X','Original Y','Original Z','New X','New Y','New
Z','location','southeast');
xlabel('Time [seconds]'); % x axis title
ylabel('Speed [m/s]'); % y axis title

%% Joint Angle differences of configurations for Question 1E

differences = abs(q_1-q_2); % 7 x 21 matrix
A = mean(differences');
M = max(differences');

figure(6)
category = {'S0';'S1';'E0';'E1';'W0';'W1';'W2'};
y = [A(1) M(1); A(2) M(2); A(3) M(3); A(4) M(4); A(5) M(5); A(6) M(6); A(7)
M(7)];
bar(y);
xticklabels(category) % This will set labels to be used for each tick of
the x-axis
xticks(1:1:length(category)) % This will set how many ticks you want on the
x-axis.

legend('Average','Maximum');
ylabel('Difference in Joint Angle [Radians]');
xlabel('Name of Joint');

%% Question 2
% A plot of the desired end-effector position circle and the actual
% end-effector position shape with respect to the ORIGINAL joint
% angle configuration

figure(7)
hold on
grid on
% My Answer to Question 1c
```

```
plot(position_1(1,:),position_1(2,:),'b'); % Plot my position result

% Data given from figure in class assignment: Obtained by plot digitizer
X_position=[0.3035  0.2992  0.2948  0.2914  0.2895  0.2880  0.2878  0.2892
0.2909  0.2936  0.2972  0.3011  0.3052  0.3108  0.3152  0.3196  0.3242
0.3290  0.3358  0.3404  0.3458  0.3533  0.3582  0.3628  0.3684  0.3732
0.3768  0.3795  0.3824  0.3844  0.3858  0.3873  0.3880  0.3880  0.3866
0.3841  0.3810  0.3773  0.3725  0.3676  0.3633  0.3586  0.3531  0.3472
0.3424  0.3383  0.3324  0.3271  0.3200  0.3128  0.3077  0.3038];
Y_position=[-0.0359 -0.0313 -0.0247 -0.0175 -0.0119 -0.0046 0.0032  0.0124
0.0179  0.0242  0.0305  0.0349  0.0388  0.0429  0.0456  0.0473  0.0490
0.0502  0.0507  0.0507  0.0497  0.0485  0.0463  0.0444  0.0410  0.0364
0.0330  0.0291  0.0242  0.0194  0.0155  0.0097  0.0032  -0.0036 -0.0119 -
0.0189 -0.0247 -0.0310 -0.0352 -0.0398 -0.0422 -0.0448 -0.0470 -0.0485 -
0.0492 -0.0495 -0.0492 -0.0480 -0.0458 -0.0422 -0.0395 -0.0361];

plot(X_position,Y_position,'or'); % Plot data given in Assignment
xlabel('X-position [m]');
ylabel('Y-position [m]');
legend('My Answer','Check');
hold off

%% Question 3
% A plot of the desired angular profiles and the actual angular profiles
% with respect to the ORIGINAL joint angle configuration.

figure(8)
hold on
grid on
% Plot my Answer to question 1d
plot(t,q_1(1,:),'b'); % S_0
plot(t,q_1(2,:),'r'); % S_1
plot(t,q_1(3,:),'y'); % E_0
plot(t,q_1(4,:),'c'); % E_1
plot(t,q_1(5,:),'g'); % W_0
plot(t,q_1(6,:),'m'); % W_1
plot(t,q_1(7,:),'k'); % W_2

% Data given from figure in class assignment: obtained by plot digitizer
Time_E0=[-0.00242754 0.336728 0.636468  0.857359 1.0546 1.28342 1.54381
1.75687 1.98571 2.24613 2.50655 2.68015 2.89316 3.09829 3.34287 3.54795
3.75303 3.97386];
Angle_E0=[-0.00763359   0.0310739 0.056996 0.0637086 0.0640223 0.0516635
0.0393549 0.0206097 0.00188958 -0.0231417 -0.0481729 -0.0606195 -0.0602808
-0.0599545 -0.0595656 -0.0401554 -0.0207452 0.0114127];

Time_E1=[-0.0134604 0.21532 0.499311 0.73595 1.03569 1.34333 1.68255
2.00601 2.35316 2.80293 3.15013 3.32373 3.59992 3.89183 3.99438];
Angle_E1=[1.27734   1.28407 1.29724 1.3167 1.34262 1.3622 1.37546 1.38233
1.37653 1.35179 1.3269 1.31446 1.28945 1.28991 1.29644];

Time_S0=[-0.00241198 0.313122 0.526123 0.699646 0.80221 0.983669 1.14144
1.3229 1.55962 1.76478 1.9936 2.29345 2.59329  2.88518 3.2244  3.5163
3.76875 3.99753];
Angle_S0=[-0.0139949 0.00559094 0.012291 0.031651 0.0318141 0.0321027
0.038715 0.0390036 0.0266573 0.0142609 0.00190212 -0.0167051  -0.0289509  -
0.0221254  -0.00886319 -0.00203763 0.0047252 0.0114504];

Time_S1=[-0.0089788 0.235596 0.432881 0.64596 0.914281 1.19052 1.45097
1.74296 1.99541 2.18476 2.40563 2.67378 2.91039 3.17854  3.4388 3.64388
3.84898 3.99888];
```

```
Angle_S1=[-0.55472 -0.554331 -0.573101 -0.598208 -0.629588 -0.673678 -
0.711432 -0.742774 -0.736011 -0.73571 -0.722636 -0.684042 -0.651859 -
0.613264 -0.574682 -0.555272 -0.542223 -0.541985];

Time_W0=[-0.0103    0.2894  0.5182  0.7312  0.8889  1.0389  1.2045  1.3466
1.4728  1.6938  1.8673  2.1041  2.3171  2.4591  2.6248  2.7905  3.0035
3.2166  3.4059  3.6110  3.7845  3.9818];
Angle_W0= [-0.0076  0.0119  0.0314  0.0444  0.0510  0.0449  0.0515  0.0390
0.0392  0.0141  0.0144  0.0021  -0.0039 -0.0164 -0.0225 -0.0286 -0.0283 -
0.0280 -0.0277 -0.0210 -0.0016 -0.0013];

Time_W1=[-0.0109706 0.430827    0.856814    1.23545 1.56678 1.86656 2.09536
2.42675 2.69501 3.14476 3.56296 4.00478];
Angle_W1=[0.259529  0.266593    0.286355    0.312402    0.325652    0.33249
0.332854    0.320658    0.314724    0.296355    0.271575    0.272277];

Time_W2=[-0.0024431 1.28355 2.71155 3.98965];
Angle_W2=[-0.00127226   0.000772894 0.0030439   0.00507651];

plot(Time_S0,Angle_S0,'ob'); % S_0
plot(Time_S1,Angle_S1,'or'); % S_1
plot(Time_E0,Angle_E0,'oy'); % E_0
plot(Time_E1,Angle_E1,'oc'); % E_1
plot(Time_W0,Angle_W0,'og'); % W_0
plot(Time_W1,Angle_W1,'om'); % W_1
plot(Time_W2,Angle_W2,'ok'); % W_2
hold off
legend('S0','S1','E0','E1','W0','W1','W2','location','eastoutside');
xlabel('Time [seconds]'); % x axis title
ylabel('Joint Angles [radians]'); % y axis title

% A plot of the desired speed profiles and the actual speed profiles
% with respect to the ORIGINAL joint angle configuration.
figure(9)
hold on
grid on
plot(t,speed_1(1,:),'b'); % x component of end effector speed
plot(t,speed_1(2,:),'r'); % y component of end effector speed
plot(t,speed_1(3,:),'y'); % z component of end effector speed

% Data given from figure in class assignment: obtained by plot digitizer
Time_X=[-0.0147 0.0809  0.1985  0.3235  0.4632  0.5515  0.6765  0.7868
0.9118  1.0074  1.1544  1.2427  1.3677  1.5147  1.6691  1.8088  1.9265
2.1029  2.3382  2.5294  2.6765  2.7941  2.9485  3.0441  3.2059  3.2794
3.3677  3.5000  3.6103  3.7647  3.8750  3.9927];
Speed_X=[-0.0002 -0.0114 -0.0267 -0.0405 -0.0543 -0.0610 -0.0696 -0.0752 -
0.0781 -0.0785 -0.0763    -0.0710 -0.0640 -0.0528 -0.0367 -0.0214 -0.0065
0.0151  0.0423  0.0602  0.0707  0.0755  0.0793  0.0785  0.0740  0.0696
0.0651  0.0561  0.0449  0.0274  0.0132  0.0002];

Time_Y=[0.0000  0.0809  0.1618  0.2500  0.3088  0.3897  0.5147  0.6250
0.7941  0.9779  1.1471  1.3088  1.4853  1.6618  1.7574  1.8456  1.9559
2.0147  2.1397  2.2353  2.3603  2.4853  2.6250  2.7794  2.9559  3.1324
3.3235  3.4632  3.5515  3.6765  3.8088  3.8897  3.9779];
Speed_Y=[0.0789 0.0778  0.0759  0.0725  0.0681  0.0640  0.0531  0.0423
0.0233  -0.0002 -0.0192 -0.0379 -0.0550 -0.0692 -0.0725 -0.0770 -0.0781 -
0.0781 -0.0766 -0.0718 -0.0651 -0.0550 -0.0423 -0.0248 -0.0028 0.0185
0.0405  0.0543  0.0621  0.0699  0.0748  0.0774  0.0789];

Time_Z=[-0.00735294 0.477941    1.05147 1.75735 2.34559 3   3.50735
3.99265];
```

```
Speed_Z=[-0.00018648    -0.00018648 -0.00018648 -0.00018648 -0.00018648 -
0.00018648 -0.00018648 0.00018648];

plot(Time_X,Speed_X,'ob'); % x component of end effector speed
plot(Time_Y,Speed_Y,'or'); % y component of end effector speed
plot(Time_Z,Speed_Z,'oy'); % z component of end effector speed
hold off
legend('X Component','Y Component','Z Component','location','southeast');
xlabel('Time [seconds]'); % x axis title
ylabel('Speed [m/s]'); % y axis title

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% End of Script
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```