Abdurrafay Khan 1005215517 khanab56

MAT188: Laboratory #12
Final Project

1. Problem Statement (as given in Project Document):

Consider the data saved in the posted robpos.mat data file. This file contains a matrix with 61 rows and 3 columns, with positions of the robot measured every 1 second during an interval of 1 minute. In each row, the first column represents the timestamp of the measurements (in seconds), the second column represents the robot's x-coordinate in meters at the corresponding time, and the third column represents its y-coordinate in meters.

You are asked to:

a) Use this raw data to visualize a piecewise linear approximation of the position of this robot as it moves through this 2D space, i.e., plot $y(t)$ versus $x(t)$. Determine the values of the robot's velocity and acceleration from this raw data by considering this approach:

If the vehicle's position at time t is given by $[x(t), y(t)]$, an approximation of its speed at time t seconds, t = 0,1, … ,59, is

$$v(t) = \sqrt{v_x^2 + v_y^2} = \sqrt{[x(t+1) - x(t)]^2 + [y(t+1) - y(t)]^2}$$

and an approximation of its acceleration at time t, t = 1,2, … ,59, is

$$a(t) = v(t) - v(t-1)$$

This is because the time increment is one second, i.e., Δt = 1s.

b) Use polynomial curve fitting techniques to create a smoother picture of how the robot has moved through this 2D space and how its velocity and acceleration behave. Explain how this approximate model compares to the original data. Remember that if you know how the velocity and acceleration behave with respect to the x and y directions, then the total speed and acceleration is given as the square root of the sum of the squares, i.e., $a(t) = \sqrt{a_x^2 + a_y^2}$

c) Path control: You have now been asked to be part of a team that has to create a control system that keeps the robot on a circular path. This path starts and ends at (x, y) = (0,0), has a diameter of 2.4 m and takes 1 minute to complete one revolution. In order for your team to know which signals to send to the motors that are controlling the movement in the x and y directions, you have been asked to determine the velocity and acceleration values in each direction that would be required for this kind of movement over one revolution.

2. Real World Scenario (Mathematical Model):

Some mathematical models that would apply to this problem is that fact that the derivative of a position vs time curve is a velocity vs time curve. It is also true that the derivative of a velocity vs time curve is an acceleration vs time curve. These concepts can be applied to find the velocity and acceleration at different values of time using the values in the robpos.mat file. Another fact that may apply is that when you multiply the radius of a circle by cos(θ), the horizontal distance of a point away from the center is given. When you multiply the radius of a circle by sin(θ), the vertical distance of a point away from the center is given. This can be useful when determining position of the robot's circular path and using those values to find their respective velocities and accelerations.

3. Design Parameter

For the purpose of this project, the "for" loop must be used in my code.

4. **Deliverables:**

a. **Script**

```
%Part A
%Plot of X vs Time and Y vs Time
%Creates a matrix with the values from the robpos.mat file
robpos=[[0,2.01312173025967e-16,-3.28767728239894;1,0.463864645274219,-
2.92872610670430;2,0.834487646860826,-2.56828889271968;3,1.33135016588390,-
2.61292182206977;4,1.92589288540993,-2.65076414824229;5,2.23573820007237,-
2.23573820007237;6,2.24800923202624,-1.63327431041574;7,2.46155616877740,-
1.25422551335209;8,2.98662213343857,-0.970412356359334;9,3.24476777033272,-
0.513920727287241;10,2.95700688845688,0;11,2.66687181717537,0.422391000180601
;12,2.79673335461602,0.908713752026275;13,2.92217687187388,1.48892348414376;1
4,2.55225782410012,1.85432385164316;15,1.96093180479913,1.96093180479913;16,1
.62689197131243,2.23922469587414;17,1.42897990487675,2.80453097346686;18,1.01
435848857544,3.12187442108261;19,0.461296079444406,2.91250882030578;20,8.2663
6589424463e-17,2.70000000000000;21,-0.223798260125846,2.67076752748190;22,-
0.348885865299340,2.70221337410530;23,-
0.599319932468110,2.56642604983457;24,-
0.815298685440794,2.42038434756022;25,-
0.892236475057347,2.36695183414633;26,-1.06449383762508,2.17118919537594;27,-
1.23046321889615,1.93528276340654;28,-1.23084001286461,1.82025922654003;29,-
1.29216781488541,1.60239797272234;30,-1.37959589690933,1.32040410309067;31,-
1.29216781488541,1.18002491711372;32,-1.23084001286461,0.985913341727671;33,-
1.23046321889615,0.709508414109763;34,-
1.06449383762508,0.584169014186262;35,-
0.892236475057347,0.457763524942653;36,-
0.815298685440794,0.236038462747866;37,-
0.599319932468110,0.160708434525982;38,-
0.348885865299340,0.134360780108382;39,-
0.223798260125846,0.00400900787502967;40,0,0;41,0.150000000000000,-
0.136117323215344;42,0.300000000000000,0.642870771068388;43,0.450000000000000
,0.0456724218027792;44,0.600000000000000,-
0.377885948459550;45,0.750000000000000,0.373576795444922;46,0.900000000000000
,-0.0832089521889972;47,1.05000000000000,-
0.766044112418487;48,1.20000000000000,-
0.109137158661498;49,1.35000000000000,-
0.397885550819733;50,1.50000000000000,-1.29595896909330;51,1.65000000000000,-
0.797885550819733;52,1.80000000000000,-
0.909137158661498;53,1.95000000000000,-1.96604411241849;54,2.10000000000000,-
1.68320895218900;55,2.25000000000000,-1.62642320455508;56,2.40000000000000,-
2.77788594845955;57,2.55000000000000,-2.75432757819722;58,2.70000000000000,-
2.55712922893161;59,2.85000000000000,-3.73611732321534;60,3,-4]];

t1=robpos(1:61,1);                      %Creates variable t1 that holds the
values for time from 0s-60s
x1=robpos(1:61,2);                      %Creates variable x1 that holds the
values for x position from 0s-60s
y1=robpos(1:61,3);                      %Creates variable y1 that holds the
values for y position from 0s-60s
figure                                  %Creates a new figure to plot "X & Y
vs Time" graph
grid on                                 %Creates grid on graph
hold on                                 %Allows multiple graphs to be plotted
on same figure
plot(t1,x1)                             %Plots "X vs Time"
```

```
plot(t1,y1)                              %Plots "Y vs Time"
legend('X position','Y position');       %Creates legend for the graph
xlabel('Time (s)');                      %Sets label for x axis
ylabel('Position (m)');                  %Sets label for y axis
title('X and Y position vs Time');       %Sets title for the graph

%Plot of X vs Y
figure                                   %Creates a new figure to plot "X & Y
Position" graph
plot(x1,y1)                              %Plots "X and Y Position"
xlabel('X position (m)');                %Sets label for x axis
ylabel('Y position (m)');                %Sets label for y axis
title('X and Y position');               %Sets title for the graph

%Plot of Speed vs Time
figure                                           %Creates a new figure to plot
"Speed vs Time" graph
grid on                                          %Creates grid on graph
t2=robpos(1:60,1);                               %Creates variable t2 that
holds the values for time from 0s-59s
x2=robpos(1:60,2);                               %Creates variable x2 that
holds the values for x position from 0s-59s
y2=robpos(1:60,3);                               %Creates variable y2 that
holds the values for y position from 0s-59s
x3=robpos(2:61,2);                               %Creates variable x3 that
holds the values for x position from 1s-60s
y3=robpos(2:61,3);                               %Creates variable y3 that
holds the values for y position from 1s-60s
v1=(sqrt(((x3-x2).^2)+((y3-y2).^2)));            %Equation for speed of the
robot
plot(t2,v1)                                      %Plots "Speed vs Time"
xlabel('Time (s)');                              %Sets label for x axis
ylabel('Speed (m/s)');                           %Sets label for y axis
title('Speed vs Time');                          %Sets title for the graph

%Plot of Acceleration vs Time
figure                                           %Creates a new figure to plot
"Acceleration vs Time" graph
grid on                                          %Creates grid on graph

t3=robpos(1:59,1);                               %Creates variable t3 that
holds the values for time from 0s-58s
x4=robpos(3:61,2);                               %Creates variable x4 that
holds the values for x position from 2s-60s
y4=robpos(3:61,3);                               %Creates variable y2 that
holds the values for y position from 2s-60s
x5=robpos(2:60,2);                               %Creates variable x5 that
holds the values for x position from 1s-59s
y5=robpos(2:60,3);                               %Creates variable y5 that
holds the values for y position from 1s-59s
v2=(sqrt(((x4-x5).^2)+((y4-y5).^2)));            %Part of the equation for the
acceleration of the robot

x6=robpos(2:60,2);                               %Creates variable x6 that
holds the values for x position from 1s-59s
```

```
y6=robpos(2:60,3);                                %Creates variable y6 that 
holds the values for y position from 1s-59s
x7=robpos(1:59,2);                                %Creates variable x7 that 
holds the values for x position from 0s-58s
y7=robpos(1:59,3);                                %Creates variable y7 that 
holds the values for y position from 0s-58s
v3=(sqrt(((x6-x7).^2)+((y6-y7).^2)));             %Part of the equation for the 
acceleration of the robot

a=v2-v3;                                          %Equation for speed of the 
robot
plot(t3,a)                                        %Plots "Acceleration vs Time"
xlabel('Time (s)');                               %Sets label for x axis
ylabel('Acceleration (m/s^2)');                   %Sets label for y axis
title('Acceleration vs Time');                    %Sets title for the graph



%Part B
%Polyfit for X vs Time and Y vs Time
figure                                            %Creates a new figure to plot 
"X & Y Approximated Position" graph
grid on                                           %Creates grid on graph
hold on                                           %Allows multiple graphs to be 
plotted on same figure
p_xvst=polyfit(t1,x1,10);                         %Produces coefficients for a 
function that would fit the data points the best
xvst=polyval(p_xvst,t1);                          %Generates curve of best fit 
based off of coefficients from the polyfit function
plot(t1,x1);                                      %Plots "X vs Time" (from code 
above)
plot(t1,xvst);                                    %Plots Approximated fit for 
"X vs Time"
p_yvst=polyfit(t1,y1,10);                         %Produces coefficients for a 
function that would fit the data points the best
yvst=polyval(p_yvst,t1);                          %Generates curve of best fit 
based off of coefficients from the polyfit function
plot(t1,y1);                                      %Plots "Y vs Time" (from code 
above)
plot(t1,yvst);                                    %Plots Approximated fit for 
"Y vs Time"

xlabel('Time (s)');                                                   %Sets label 
for x axis
ylabel('Position (m)');                                               %Sets label 
for y axis
legend('X pos.','X pos. approx','Y pos.','Y pos. approx');            %Creates 
legend for the graph
title('X & Y Approximated Position');                                 %Sets title 
for the graph

%Polyfit for Speed vs Time
figure                                            %Creates a new figure to plot 
"Approximated Speed vs Time" graph
grid on                                           %Creates grid on graph
```

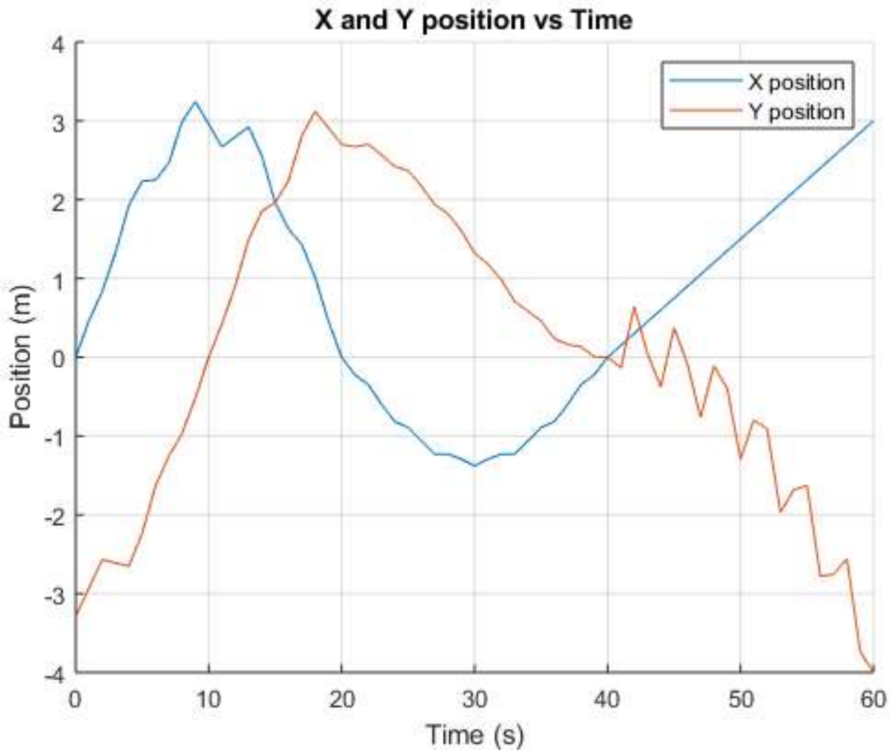
```
hold on                                         %Allows multiple graphs to be
plotted on same figure
p_vvst=polyfit(t2,v1,7);                        %Produces coefficients for a
function that would fit the data points the best
vvst=polyval(p_vvst,t2);                        %Generates curve of best fit
based off of coefficients from the polyfit function
plot(t2,v1);                                    %Plots "Speed vs Time" (from
code above)
plot(t2,vvst);                                  %Plots Approximated fit for
"Speed vs Time"

xlabel('Time (s)');                             %Sets label for x axis
ylabel('Speed (m/s)');                          %Sets label for y axis
legend('V','V approx');                         %Creates legend for the graph
title('Approximated Speed vs Time');            %Sets title for the graph

%Polyfit for Acceleration vs Time
figure                                          %Creates a new figure to plot
"Approximated Acceleration vs Time" graph
grid on                                         %Creates grid on graph
hold on                                         %Allows multiple graphs to be
plotted on same figure
p_avst=polyfit(t3,a,6);                         %Produces coefficients for a
function that would fit the data points the best
avst=polyval(p_avst,t3);                        %Generates curve of best fit
based off of coefficients from the polyfit function
plot(t3,a);                                     %Plots "Acceleration vs Time"
(from code above)
plot(t3,avst);                                  %Plots Approximated fit for
"Acceleration vs Time"

xlabel('Time (s)');                             %Sets label for x axis
ylabel('Acceleration (m/s^2)');                 %Sets label for y axis
legend('A','A approx');                         %Creates legend for the graph
title('Approximated Acceleration vs Time');     %Sets title for the graph



%Part C
%Plot of Circular Path
figure                                          %Creates a new figure to plot
"Circular Path" graph
grid on                                         %Creates grid on graph
for t=1:61                                      %Creates a for loop in which
coordinates of the robots circular points are calculated every second
    xdist(t)=1.2*cos(t*(pi/30));                %Equation that calculates x
coordinate of robot at 1 second intervals multiplied by (pi/30) which is
equivalent to (2pi/60)
    ydist(t)=1.2*sin(t*(pi/30))+1.2;            %Equation that calculates y
coordinate of robot at 1 second intervals multiplied by (pi/30) which is
equivalent to (2pi/60)
end                                             %Ends for loop

plot(xdist,ydist)                               %Plots "Circular Path"
xlabel('X Position (m)');                       %Sets label for x axis
```

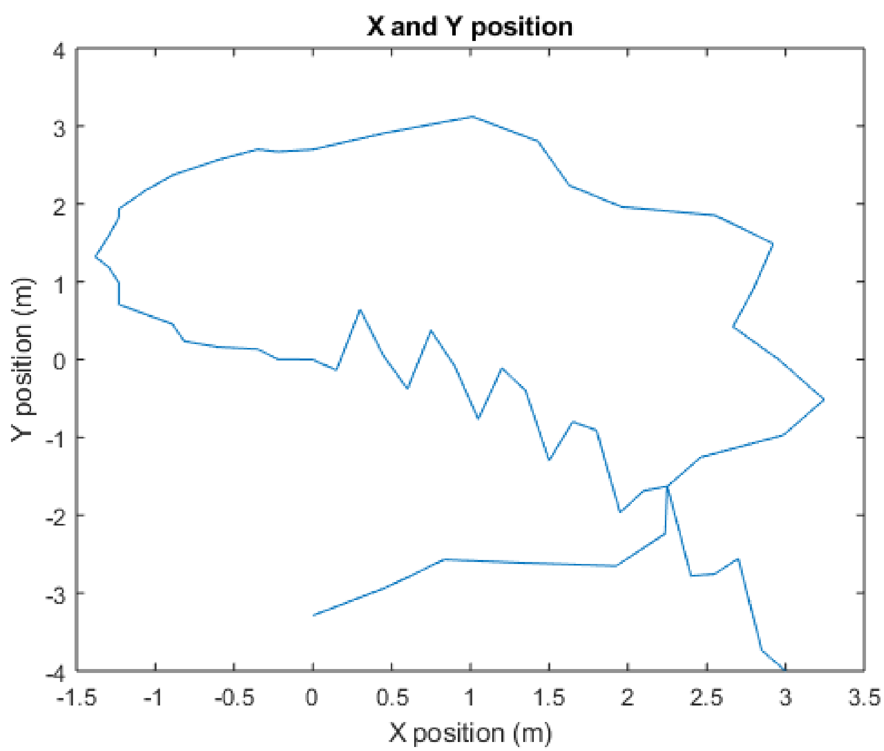
```
ylabel('Y Position (m)');                       %Sets label for y axis
title('Circular Path');                         %Sets title for the graph

%Plot of X Speed of Circular Path
figure                                          %Creates a new figure to plot
"X Speed for Circular Path" graph
grid on                                         %Creates grid on graph
t_dist=0:60;                                    %Domain for the position at x
points
x_circ=(1.2*cos(t_dist*(pi/30)));               %Equation that calculates x
coordinate of robot at 1 second intervals
vx_circ=diff(x_circ);                           %Creates variable to which
the first differences of x_circ are assigned to
t_vel=1:60;                                     %Domain for speed at x points
plot(t_vel,vx_circ)                             %Plots "X Speed for Circular
Path"
xlabel('T (s)');                                %Sets label for x axis
ylabel('V (m/s)');                              %Sets label for y axis
title('X Speed for Circular Path');             %Sets title for the graph

%Plot of Y Speed for Circular Path
figure                                          %Creates a new figure to plot
"Y Speed for Circular Path" graph
grid on                                         %Creates grid on graph
y_circ=(1.2*sin(t_dist*(pi/30)))+1.2;           %Equation that calculates y
coordinate of robot at 1 second intervals
vy_circ=diff(y_circ);                           %Creates variable to which
the first differences of y_circ are assigned to
plot(t_vel,vy_circ)                             %Plots "Y Speed for Circular
Path"
xlabel('T (s)');                                %Sets label for x axis
ylabel('V (m/s)');                              %Sets label for y axis
title('Y Speed for Circular Path');             %Sets title for the graph

%Plot of X Acceleration for Circular Path
figure                                          %Creates a new figure to plot
"X Acceleration for Circular Path" graph
grid on                                         %Creates grid on graph
ax_circ=diff(vx_circ);                          %Creates variable to which
the second differences of x_circ are assigned to
t_acc=2:60;                                     %Domain for acceleration at x
points
plot(t_acc,ax_circ);                            %Plots "X Acceleration for
Circular Path"
xlabel('T (s)');                                %Sets label for x axis
ylabel('A (m/s^2)');                            %Sets label for y axis
title('X Acceleration for Circular Path');      %Sets title for the graph

%Plot of Y Acceleration for Circular Path
figure                                          %Creates a new figure to plot
"Y Acceleration for Circular Path" graph
grid on                                         %Creates grid on graph
ay_circ=diff(vy_circ);                          %Creates variable to which
the second differences of y_circ are assigned to
plot(t_acc,ay_circ);                            %Plots "Y Acceleration for
Circular Path"
```

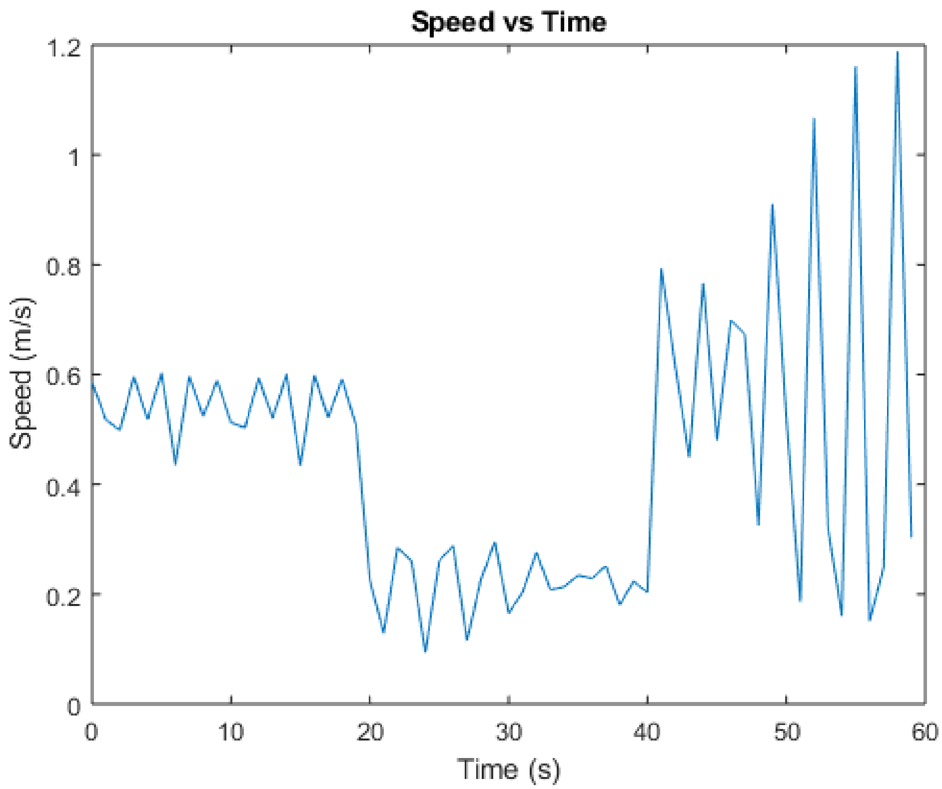
```
xlabel('T (s)');                                %Sets label for x axis
ylabel('A (m/s^2)');                            %Sets label for y axis
title('Y Acceleration for Circular Path');      %Sets title for the graph
```

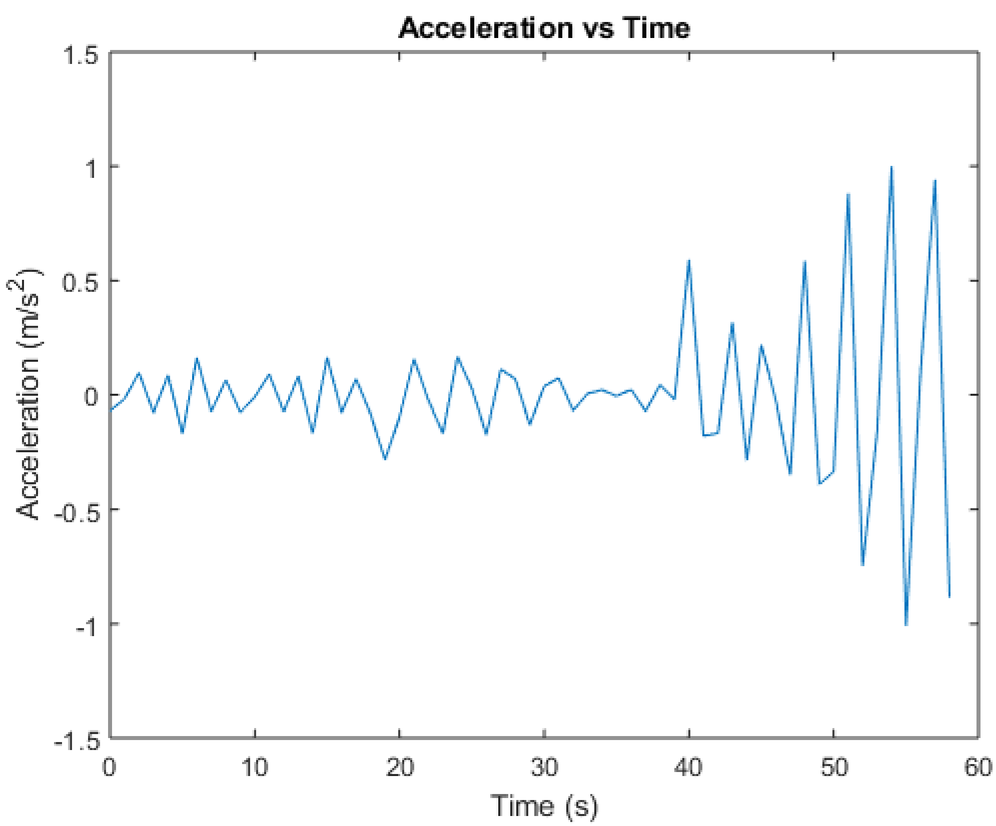
b. Plots:

The plots for part a) include the following:

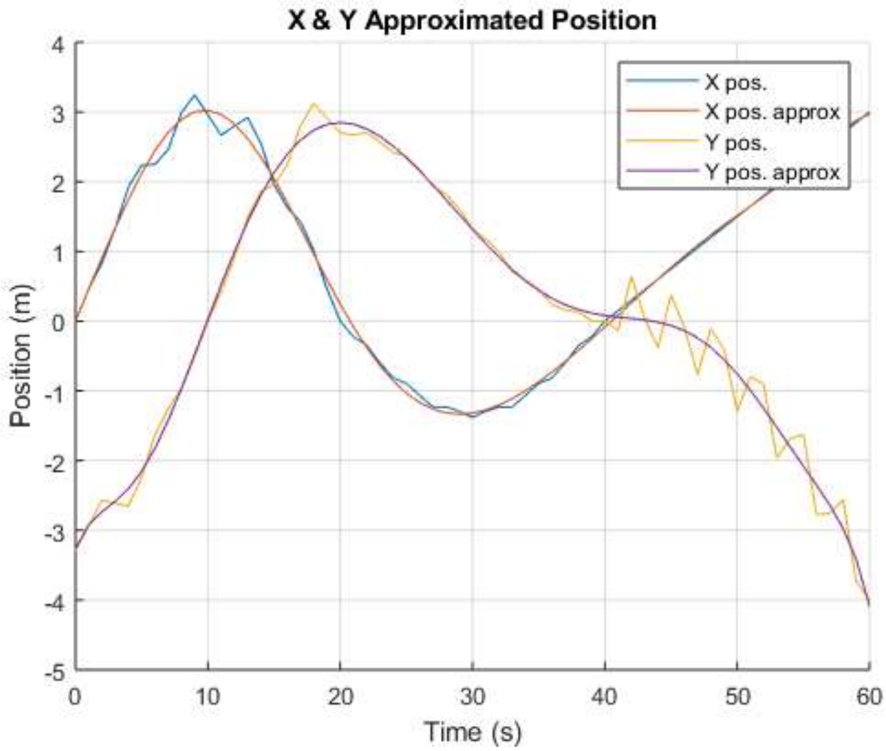


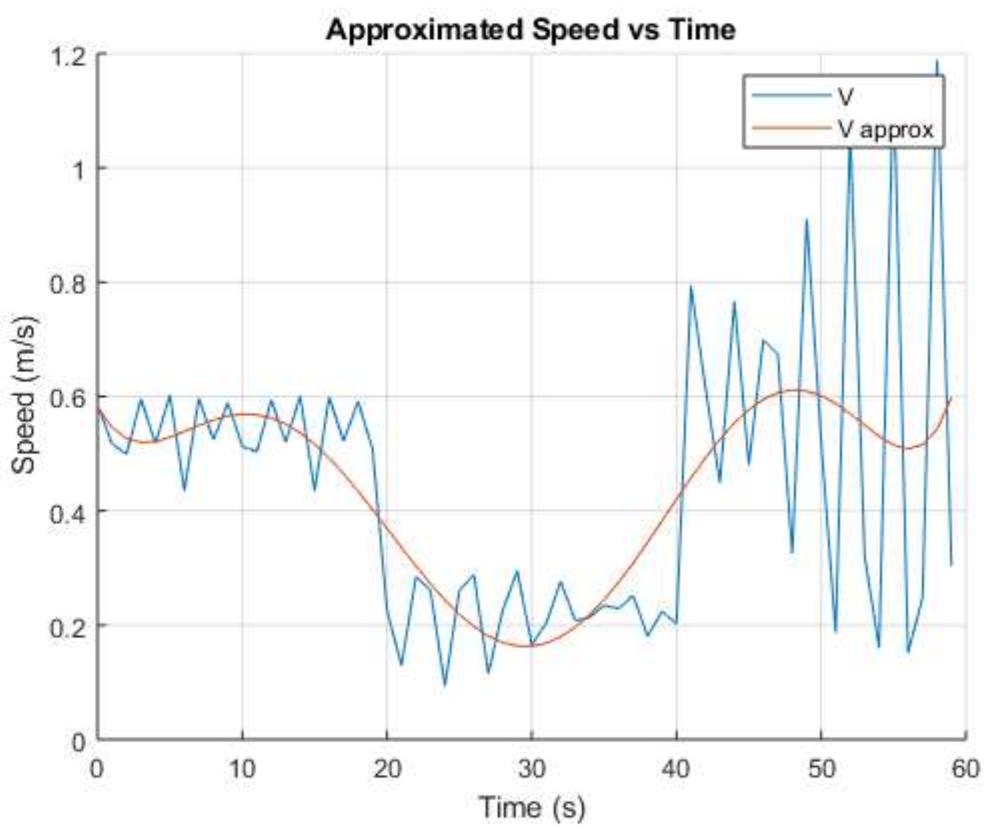


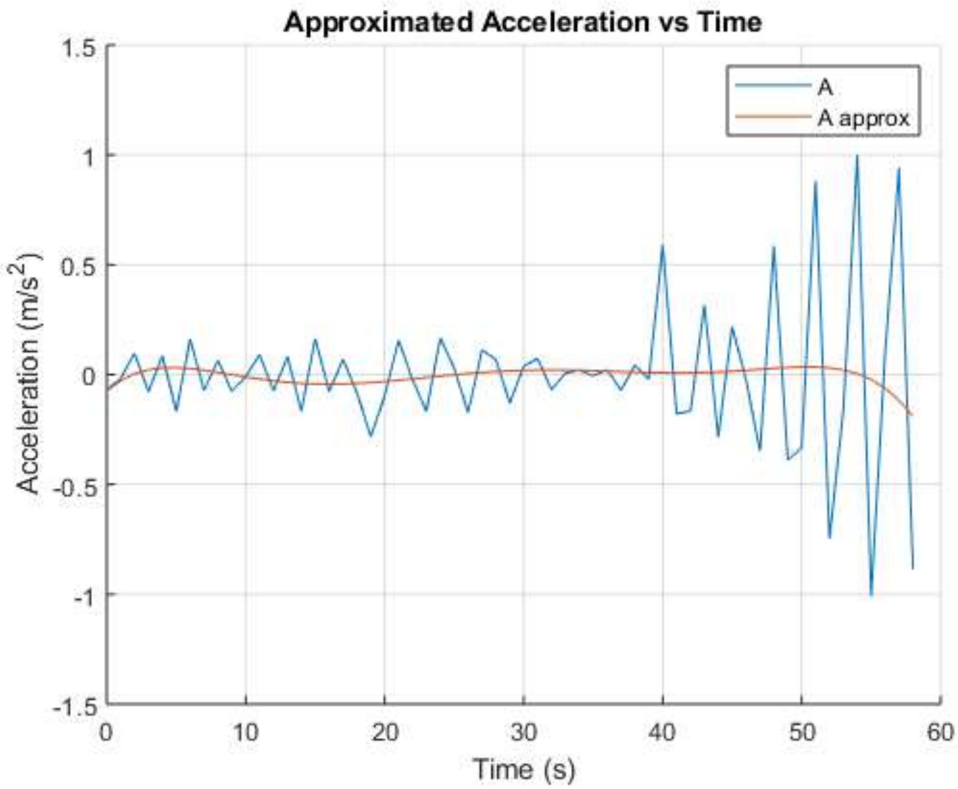

Speed vs Time
Speed (m/s)
Time (s)
1.2
1
0.8
0.6
0.4
0.2
0
0
10
20
30
40
50
60


Acceleration vs Time
Acceleration (m/s^2)
Time (s)
1.5
1
0.5
0
-0.5
-1
-1.5
0
10
20
30
40
50
60

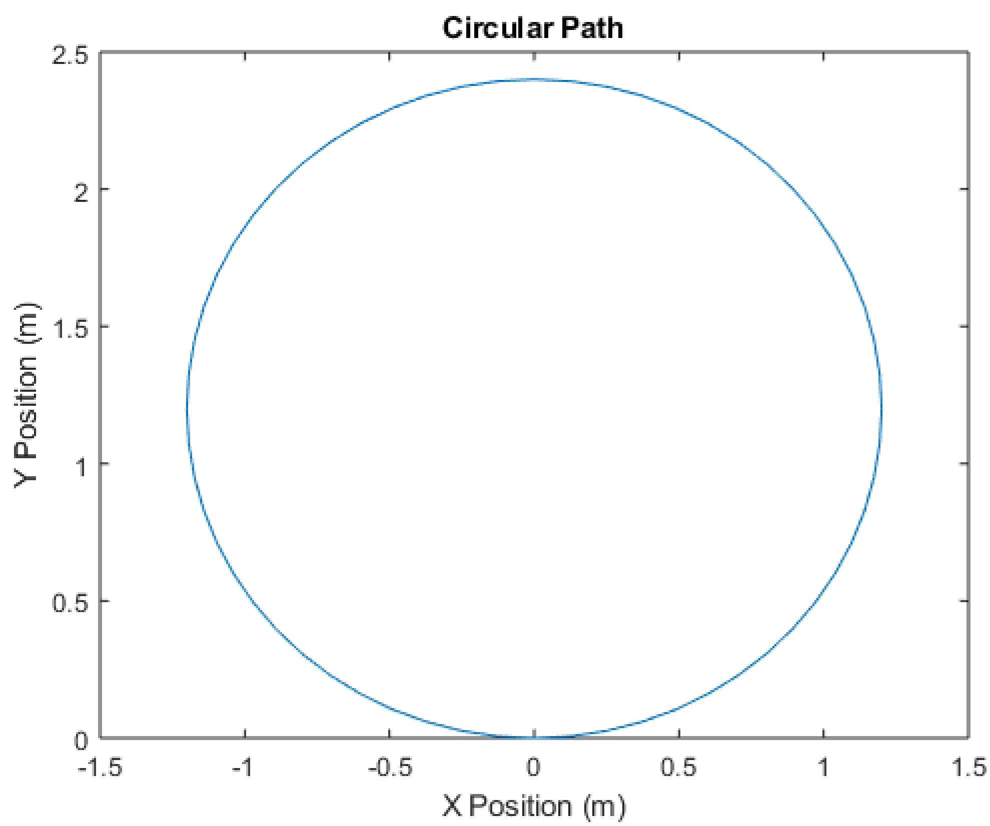
The plots for part b) include the following:

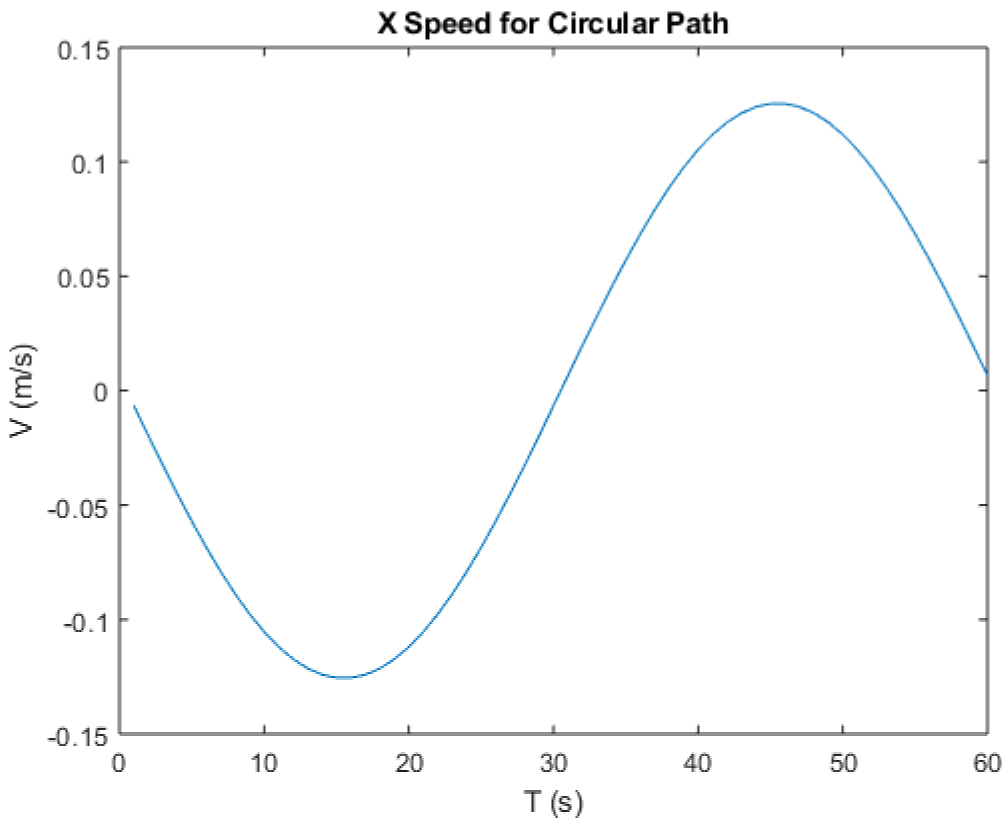


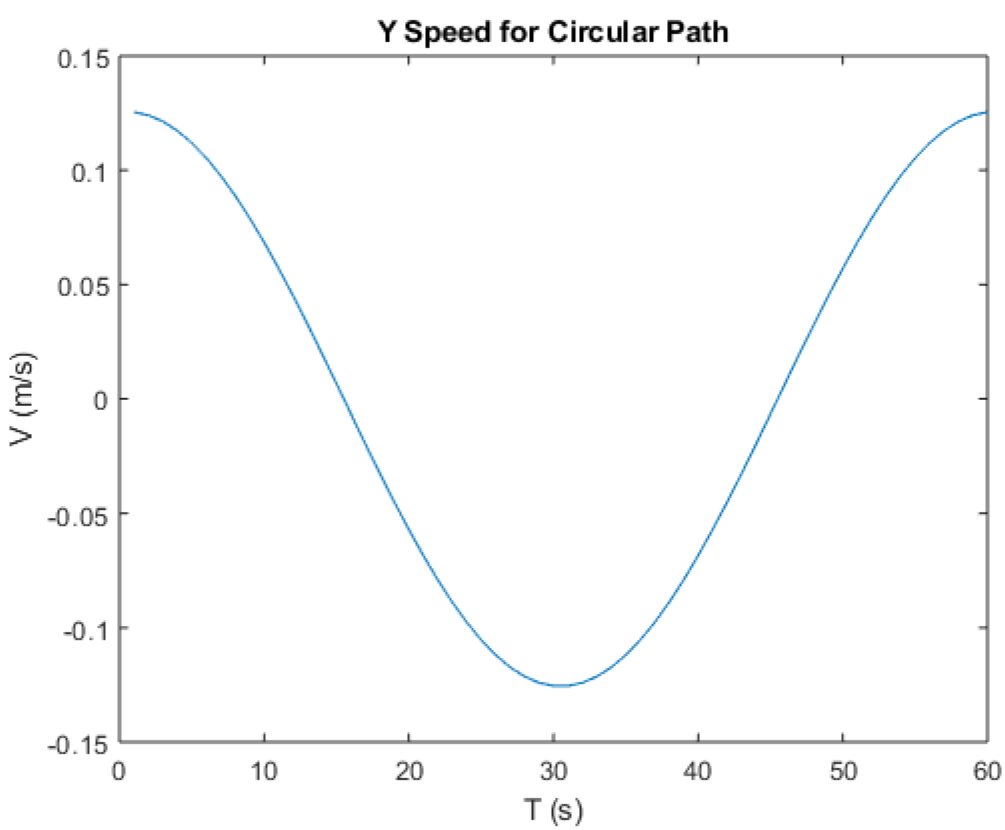


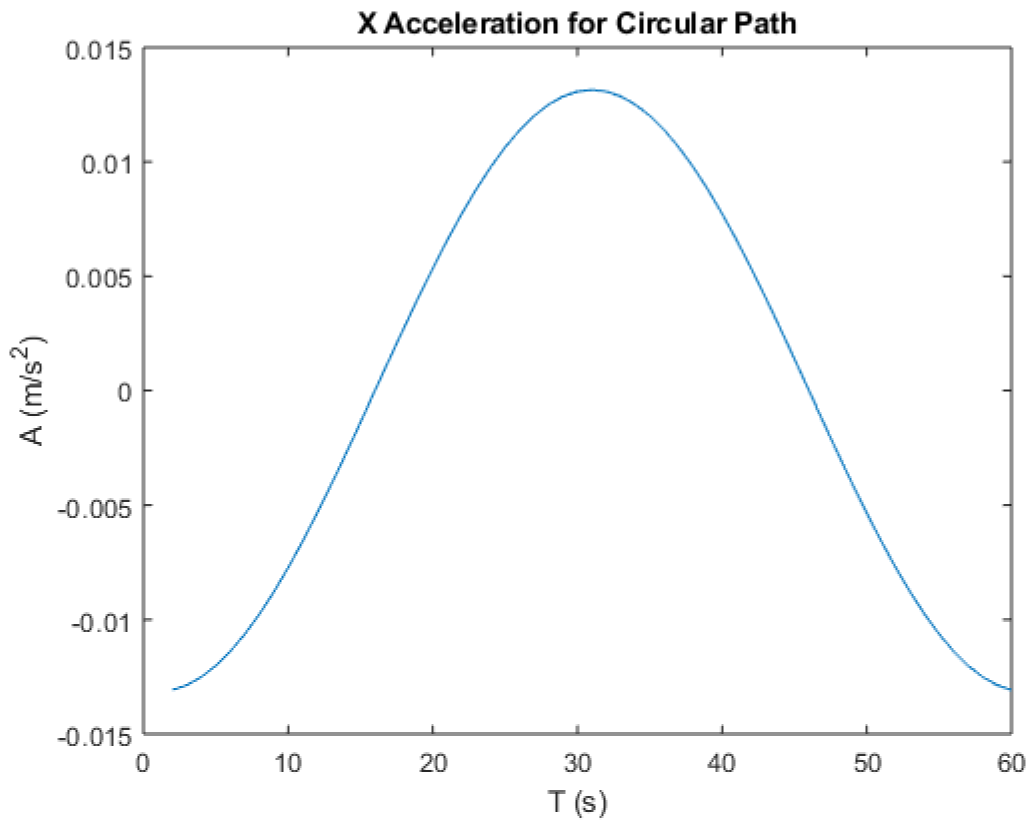


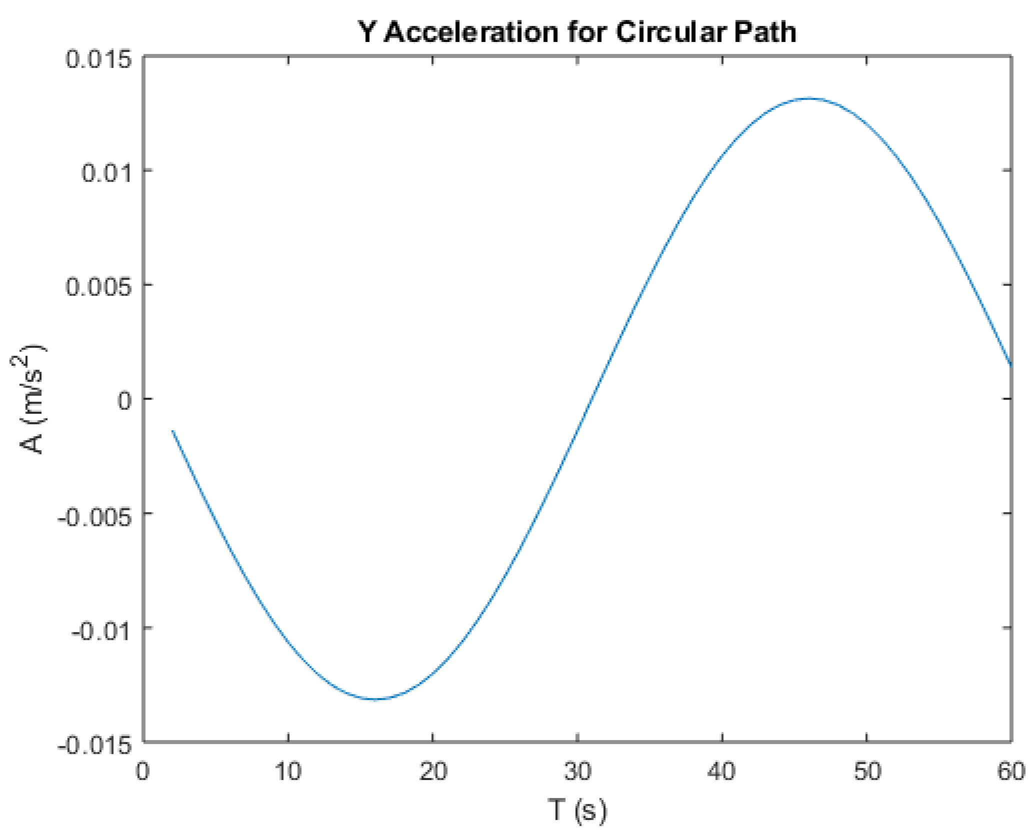
The plots for part c) include the following




X Speed for Circular Path
V (m/s)
T (s)
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
0
10
20
30
40
50
60


Y Speed for Circular Path
V (m/s)
T (s)
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
0
10
20
30
40
50
60


X Acceleration for Circular Path
A (m/s^2)
T (s)
0.015
0.01
0.005
0
-0.005
-0.01
-0.015
0
10
20
30
40
50
60


Y Acceleration for Circular Path
A (m/s^2)
T (s)
0.015
0.01
0.005
0
-0.005
-0.01
-0.015
0
10
20
30
40
50
60

c. Concluding Statement:

The graphs in part a) were constructed using the raw data that was provided. From the data, graphs for the robot's speed and acceleration were easily produced using the given velocity and acceleration equations. Part b) adds curves of best fit to the graphs in part a). In part c), the desired circular path is graphed, as well as the speeds and accelerations that the robot following the circular path has. Using the graphs in part c), it can be easily seen what the speeds and accelerations must be of the robot in the x and y direction so that the robot can follow a circular path. Below is a table of value with the points that are plotted in the graphs above.

Time (s)	X Speed (m/s)	Y Speed (m/s)	X Acceleration (m/s^2)	Y Acceleration (m/s^2)
0	-0.006573726	0.125434156	-0.013075428	-0.001374283
1	-0.019649154	0.124059873	-0.012860148	-0.002733509
2	-0.032509301	0.121326364	-0.012503969	-0.004062786
3	-0.04501327	0.117263578	-0.012010794	-0.00534755
4	-0.057024065	0.111916028	-0.011386027	-0.006573726
5	-0.068410091	0.105342303	-0.010636511	-0.007727878
6	-0.079046603	0.097614425	-0.00977046	-0.008797362
7	-0.088817063	0.088817063	-0.008797362	-0.00977046
8	-0.097614425	0.079046603	-0.007727878	-0.010636511
9	-0.105342303	0.068410091	-0.006573726	-0.011386027
10	-0.111916028	0.057024065	-0.00534755	-0.012010794
11	-0.117263578	0.04501327	-0.004062786	-0.012503969
12	-0.121326364	0.032509301	-0.002733509	-0.012860148
13	-0.124059873	0.019649154	-0.001374283	-0.013075428
14	-0.125434156	0.006573726	2.498E-16	-0.013147451
15	-0.125434156	-0.006573726	0.001374283	-0.013075428
16	-0.124059873	-0.019649154	0.002733509	-0.012860148
17	-0.121326364	-0.032509301	0.004062786	-0.012503969
18	-0.117263578	-0.04501327	0.00534755	-0.012010794
19	-0.111916028	-0.057024065	0.006573726	-0.011386027
20	-0.105342303	-0.068410091	0.007727878	-0.010636511
21	-0.097614425	-0.079046603	0.008797362	-0.00977046
22	-0.088817063	-0.088817063	0.00977046	-0.008797362
23	-0.079046603	-0.097614425	0.010636511	-0.007727878
24	-0.068410091	-0.105342303	0.011386027	-0.006573726
25	-0.057024065	-0.111916028	0.012010794	-0.00534755
26	-0.04501327	-0.117263578	0.012503969	-0.004062786
27	-0.032509301	-0.121326364	0.012860148	-0.002733509
28	-0.019649154	-0.124059873	0.013075428	-0.001374283
29	-0.006573726	-0.125434156	0.013147451	4.44089E-16
30	0.006573726	-0.125434156	0.013075428	0.001374283
31	0.019649154	-0.124059873	0.012860148	0.002733509

32	0.032509301	-0.121326364	0.012503969	0.004062786
33	0.04501327	-0.117263578	0.012010794	0.00534755
34	0.057024065	-0.111916028	0.011386027	0.006573726
35	0.068410091	-0.105342303	0.010636511	0.007727878
36	0.079046603	-0.097614425	0.00977046	0.008797362
37	0.088817063	-0.088817063	0.008797362	0.00977046
38	0.097614425	-0.079046603	0.007727878	0.010636511
39	0.105342303	-0.068410091	0.006573726	0.011386027
40	0.111916028	-0.057024065	0.00534755	0.012010794
41	0.117263578	-0.04501327	0.004062786	0.012503969
42	0.121326364	-0.032509301	0.002733509	0.012860148
43	0.124059873	-0.019649154	0.001374283	0.013075428
44	0.125434156	-0.006573726	-1.05471E-15	0.013147451
45	0.125434156	0.006573726	-0.001374283	0.013075428
46	0.124059873	0.019649154	-0.002733509	0.012860148
47	0.121326364	0.032509301	-0.004062786	0.012503969
48	0.117263578	0.04501327	-0.00534755	0.012010794
49	0.111916028	0.057024065	-0.006573726	0.011386027
50	0.105342303	0.068410091	-0.007727878	0.010636511
51	0.097614425	0.079046603	-0.008797362	0.00977046
52	0.088817063	0.088817063	-0.00977046	0.008797362
53	0.079046603	0.097614425	-0.010636511	0.007727878
54	0.068410091	0.105342303	-0.011386027	0.006573726
55	0.057024065	0.111916028	-0.012010794	0.00534755
56	0.04501327	0.117263578	-0.012503969	0.004062786
57	0.032509301	0.121326364	-0.012860148	0.002733509
58	0.019649154	0.124059873	-0.013075428	0.001374283
59	0.006573726	0.125434156		

d. A Description of your Engineering Problem Solving Process

Understanding the Problem:

What is the unknown? (GOAL)

The primary unknown for this problem are the speeds and accelerations in the x and y directions that would make the robot follow a circular path with a diameter of 2.4m. Secondary unknowns that lead to the primary unknown include the speed of the robot using the raw data and the acceleration of the robot using the raw data.

What relevant information is provided? (KNOWNS)

The x and y position of the robot is given for the duration of 60 seconds. The equation to calculate speed and acceleration is also given. The radius of the desired circular path is given as well.

What is the condition? What fundamental principles are related? (FOUNDATION)
Using fundamental principles, the speed of the robot can be determined using the position values, as the derivative of position is speed. The acceleration of the robot can be determined using the speed values, as the derivative of speed is acceleration.

Draw a diagram (VISUALIZATION)
Since data points are provided, the visuals would simply be the graphs created. Despite, visuals being easily creatable, the idea of how the speed and acceleration graphs can be determined is still important to keep in mind as mentioned above.

Devising a Plan
Consider the unknown. What are the possible connections between the information provided and the unknown.
As mentioned before, for part a), I can use the data points of x and y against time to find speed and acceleration of the robot using the equations provided. For part b), I could use the polyval and polyfit function to create a line of best fit with the graphs made in part a). For part c, I could use a for loop to create a circle and the diff function with the resulting points, I can create a graph that plots the x and y speed's the robot must have to travel a circular path. Using the diff function again with the speed values, I can create a graph that plots the x and y acceleration of the robot as well.

Carrying Out the Plan
Check each step as you carry out the plan. Is it leading you in the right direction?
The script for the code as well as all the produced plots are above in the document. The code is separated into sections that correspond with the different figures that were created for this problem. The first few plots for part a) consisted of a "X vs T" and "Y vs T" position graph, a "Y vs X" position graph, a speed graph, and an acceleration graph. The first 3 graphs display the movement in the x and y direction as well as the cumulative movement of the robot in the x and y direction together. The speed graph displays the speed of the robot at different times while the acceleration graph displays the acceleration of the robot at times. The plots for part b) consist of the same "X vs T" and "Y vs T" position graphs, the speed and acceleration graph with the additional of a curve of best fit. The plots for part c) consist of a circle graph that resembles the desired path the robot must take. The "X Speed for Circular Path" and "Y Speed for Circular Path" display the speeds the robot must have to travel in a circular path. The corresponding acceleration plots display the respective accelerations the robot will have to follow the circular path. Because of the code and plots, it is evident what speeds and accelerations will the robot need to travel in a circular path with diameter 2.4m.

Looking Back
Can you check the result? How about the solution?
The results in the table can be checked and compared to that of the graphs to visualize how the speed and accelerations change with respect to time. The solution can also be checked by analyzing if the plots of the derivative functions resemble that of the proper derivative. For example, if the function is sin(x), the derivative should look along the lines of cos(x).

Is the answer complete/ Have you solved the problem asked?
The problem asked for the speed and acceleration of the robot that would be necessary for the robot to follow a circular path. Following my code, I was able to find the specific values of the robot's speed in the x and y direction as well as the values of the robot's acceleration in the x and y direction that would result in the robot following a circular path. This proves that my code has solved the problem asked.

A brief description any modifications to the existing plan that took place (if applicable). If no modifications took place, comment on why, and what led to this.

No modifications were made to my plan. This is mainly because my plan was clear to follow and act upon from the beginning. It was as simple as plotting data points and plotting a few equations in part a). Part b) just required curves of best fit as I originally planned for. With part c), I used my previous knowledge on the first and second derivatives of a position graph to create that of speed vs time and acceleration vs time. Following this plan resulted in an outcome that solves the original problem.

Describe, concisely, the fundamental scientific and mathematical concepts and/or how to use Matlab for engineering problem solving

The general knowledge that the derivative of position is speed and that the derivative of speed is acceleration helped with plotting the speed and acceleration of the robot's circular path. With such a problem, Matlab became a very valuable tool as computing and solving the problem by hand would be tremendously tedious and has a significant margin for error and miscalculations. The true potential of Matlab became clear to me when plotting the raw data points by indexing them from the table. What seems impossible to do by a human is as simple as a line of code in Matlab. Following this, everything within the code is done easier and more accurate in MatLab compared to what could be done by an average human in a significantly reduced time. This project has opened my eyes to the true potential Matlab has especially for engineering problem solving as many problems are very complex, tedious and sometimes impossible to solve by hand. Matlab is an asset to an engineer who desires to solve problems in a quick efficient and successful way.