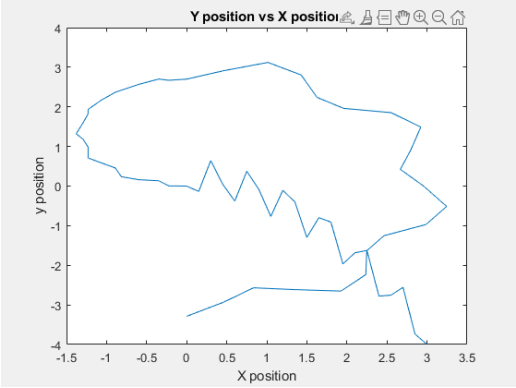
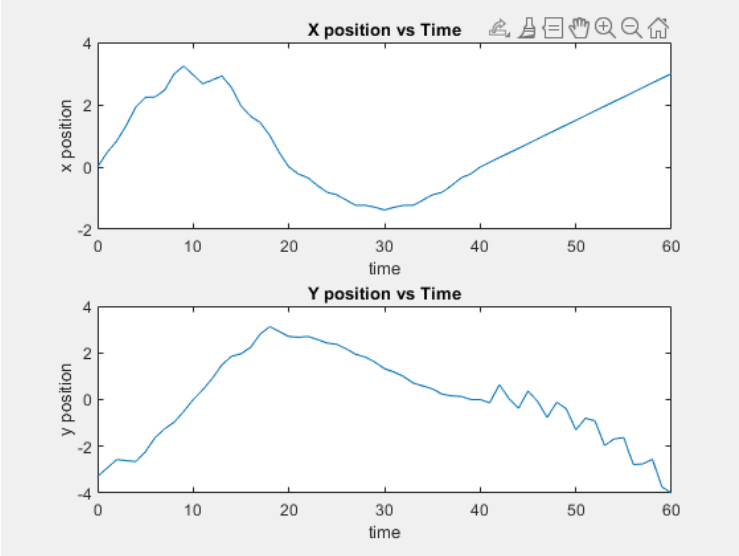
Prarthona Paul December 9, 2020

MATLAB Report: Robot Movement and Control

This report in centred around a project that looks at the movement of a robot, one much like the robotic vacuum that is used to clean houses. The project analyzes the movement of the robot in a 2D space, along the x-y plane using MATLAB. The robot can only collect data at finitely differently amounts and at equally spaced time intervals. In another words, the robot cannot take data at infinitesimally small time intervals, so the data needs to be interpolated in order to make any assumptions in between the intervals.

We are given the position of the robot in space using the x and y coordinate of the robot's position at 1 second increments starting at 0 second and ending at 61 seconds. The MATLAB file analyses the position of the robot in the 2D space by relating the position with time as well as relating the position along the x and the y axis against each other. Plotting the x and y position of the robot against time on a 2D plane, we have,





Since the position of the robot at different time points is given, any calculations will have to be made based on the position at a given time. Therefore, any estimation about the velocity and the

acceleration of the robot are to be made using the position of the robot with respect to time and its first and second order derivatives. Since we do not have an actual function for the position with respect to time, we will use first principles of calculus to determine the slope of the secant line from point S(t) to S(t+1), where S(t) is the position function of the robot as a function of time in seconds. So,

$$V(t) \cong \frac{S(t+h) - S(t)}{h}$$

Additionally, since the robot moves along the x and the y axis, the position vector will have an x and a y component, where,

$$\mathrm{S(t)} = \sqrt{S_x(\mathrm{t})^2 + S_y(\mathrm{t})^2}$$

Similarly, the velocity will also have a horizontal (assuming x is horizontal) component and a vertical component (assuming y is vertical). In this case,

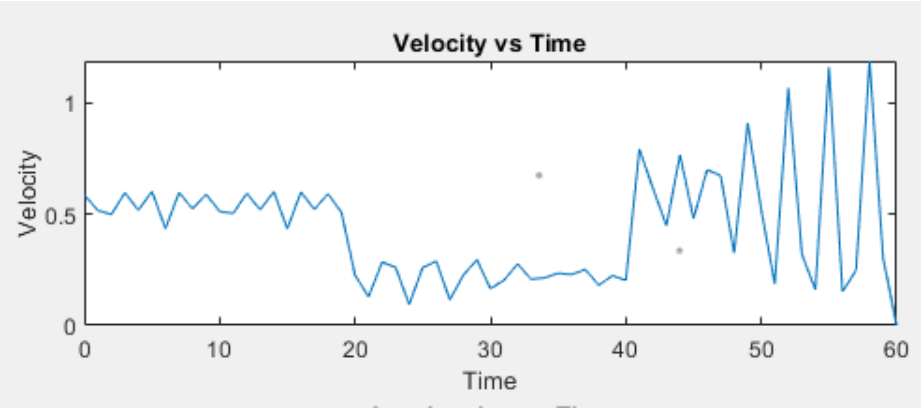
$$\mathrm{V(t)} = \sqrt{V_x(\mathrm{t})^2 + V_y(\mathrm{t})^2}$$

Recall, that the velocity function is the derivative of the position function. So, using that idea and the first principal calculus.

$$\mathrm{V(t)} = \sqrt{V_x(\mathrm{t})^2 + V_y(\mathrm{t})^2} = \sqrt{\left(\frac{S_x(\mathrm{t+1}) - S_x(\mathrm{t})}{1}\right)^2 + \left(\frac{S_y(\mathrm{t+1})^2 - S_y(\mathrm{t})^2}{1}\right)^2}$$

$$\left(S_x(\mathrm{t+1}) - S_x(\mathrm{t})\right)^2 + \left(S_y(\mathrm{t+1}) - S_y(\mathrm{t})\right)^2$$

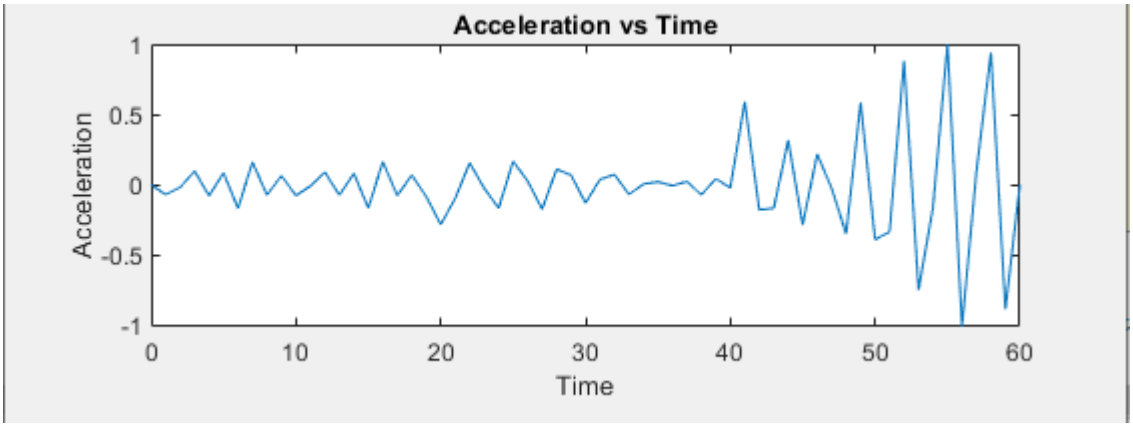
Where h = 1. Using this in a for loop, the velocity of the robot can be calculated from t = 0 seconds to t = 60 seconds. Notice how the position was gives from 0 second to 61 seconds and the velocity is calculated from 0 seconds to 60 seconds. This is because the difference between two consecutive terms can only be calculated n-1 times when there are n terms. Using this data, the velocity of the Robot can be plotted against time as seen above.



The same method can be used to calculate the acceleration of the robot since the acceleration of an object as function of time is the derivative of the velocity function, or the second derivative of the position function. Here, the first principal will be $A(t) = \frac{v(t+1)-v(t)}{1}$. Which simplifies to

$$A(t) = v(t+1) - v(t)$$

As seen before, the acceleration graph has 59 points from 0 second to 59 seconds for reasons explained above. Plotting this data on MATLAB against time gives,

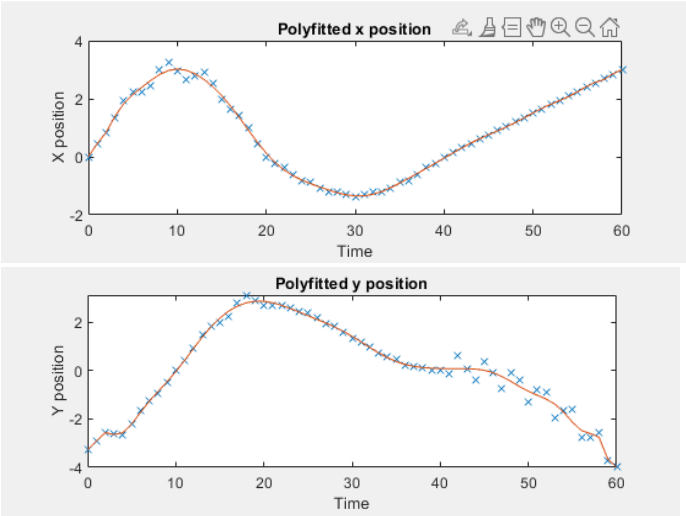


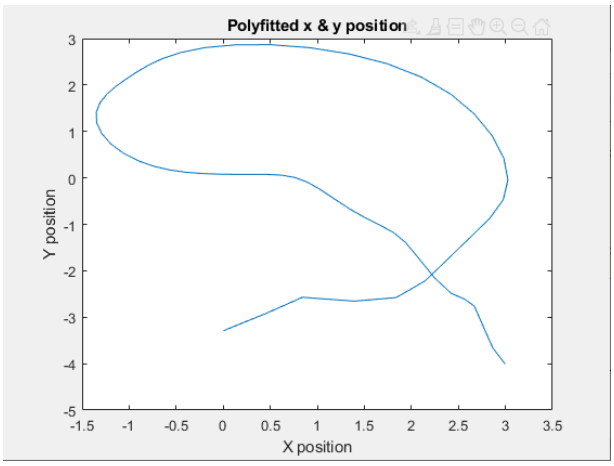
Part B: Polynomial Curve Fitting

Since the data is raw data, there are several factors that cause imperfections in the data including small bumps on the floor, change in friction, any uncertainty in data and more. As a result, using polynomial curve fitting to make the curve smoother gives a much clearer sense of the data and gives a set that can be used much more efficiently.

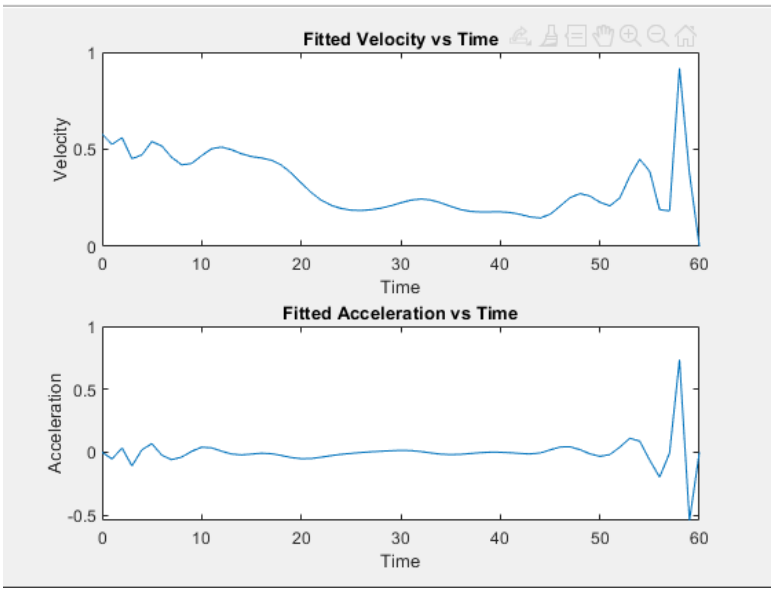
In MATLAB this can be done using the polyfit and the polyval function, which take a set of data and a given degree of polynomial and finds an equation that best fits the data, producing a curve of best fit. To minimize error bounds on the data, the position data will be fitted to a polynomial and will be differentiated to find the velocity function and the acceleration function.

After trying several different degrees, the data is fitted to a degree 20 polynomial and the x and y position values are plotted against time as done previously. This gives,





Using the equations for velocity and acceleration equations from above, we get the following.



Part C: Path control

To model the circular path of a robot that starts and ends (x, y) = (0,0), and has a diameter of 2.4 m, we can model this using the equation $x^2 + y^2 = 5.76$, where x and y are the distance of the robot from the x and the y axis, respectively.

Using implicit differentiation, the velocity and the acceleration of the robot can be determined. The first derivative is the velocity of the robot with respect to time and the acceleration is the second derivative of the robot with respect to time, or the derivative of the velocity with respect to time.