CSE360 Workshop 1 Report

Author: Jitong Ding

1.1

1. thermometer: it measures the temperature in the environment. We need to convert it into Celsius or Fahrenheit and save as float or double data type in the computer.

Camera: it measures what does the environment look like. For example, is it humans, cars, houses, pets or any other object in the scene? The image can be saved as a grid of pixels which is an array of Integer.

Radar: It measures the distance between the robotic and the objects near the robotic so it can warn the robotic to stop or change the direction. The data type is double.

Odometer: It measures the speed and time. The data type can be double or float.

Laser scanner (LIDAR): It measures distances by illuminating the target with laser light and measuring the reflection with a sensor. Can be saved into double

Inertial measurement unit (IMU): It measures and reports a body's specific force, angular rate, and sometimes the orientation of the body. Can be saved in double or float.

2. Sensors: GPS, Encoders, proximity sensors, sonar, Bump sensors

3. Actuators: Electric motors, Servo, Artificial muscles, Linear actuator, Solenoids

1.2

Link: https://github.com/djt1998/CSE360/tree/master/workshop1

1.

(1) Wrote out the ellipse equation: $\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1.$

(2) Represented the “X” and “Y” with cos and sin due to main axis: 4: x = 2*cost , y = 1 *sint

(3) Made differential for x and y by t to obtain the move function: x = 2 * (-sin(t)) , y = cos(t)

(4) Researched how python can rotate the image with “matplotlib” package and played with it to achieve to rotate the ellipse.

(5) Found a equations that achieve the rotation without changing the center: xx = cos(pi/6)*x - sin(pi/6)*y, yy = sin(pi/6)*x + cos(pi/6)*y. The x and y here are on the above.

(6) Final control policy is: x = 2 * (-sin(t)), y = cos(t), ux = cos(pi/6)*x - sin(pi/6)*y , uy = sin(pi/6)*x + cos(pi/6)*y

2.

(1) Draw the infinite symbol on the graph

(2) Find the points where the velocity equal to 0 for x axis first and then draw the cost and sint graph

(3) Make differential for x but it is not accurate, so I try many similar form of cost and get cos(t)-cos(2*t)

(4) From the plot, get a similar equation for y and then make differential

(5)Try the similar equation and then find the final answer uy =cos(t)+ cos(2*t)

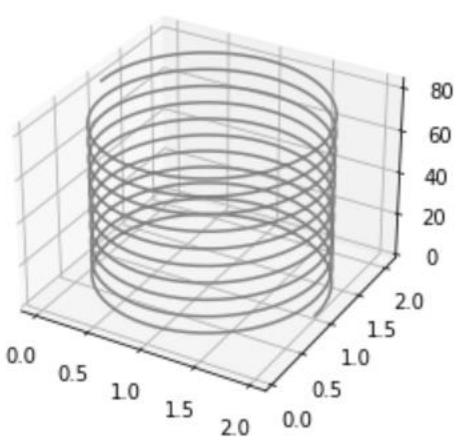
3.

(1) Looked at the notes P12 on the coursesite, and found a position equation $$\boldsymbol{h}(t) = \begin{bmatrix} \cos t \\ \sin t \\ \frac{2}{5\pi} t \end{bmatrix}.$$

(2) Make differential on the h(t) and got [-sint,cost,2/5*pi]. 2/5 *pi is the z axis moving equation.

(3) Changed the control to be 3D and policy is ux = -sin(t), uy = cos(t), uz = ((2 * pi) / 5)

(4) Initialed the array with 3 dimension and altered the plot functions to plot the 3D helix plot.



Graph: ______________________________

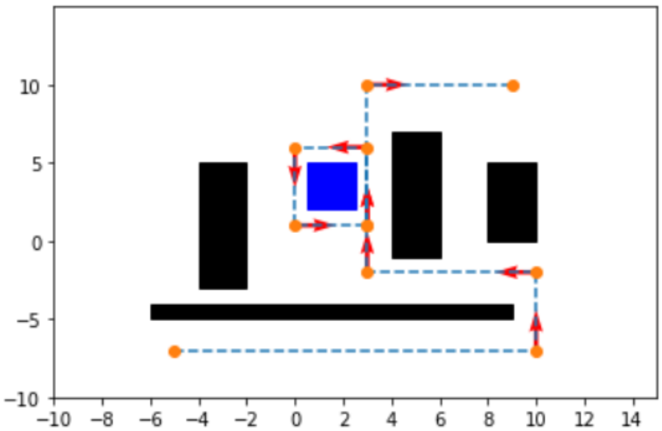
1.3

1.

(1) Find each point of the polylines

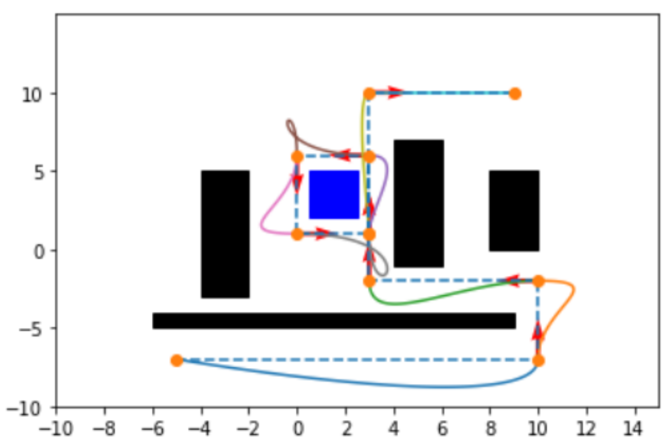
(2) Use the ploy function to connect every point one by one

(3) Define the velocity and time



2

(1) Based on the answer got from last question
(2) Try different velocity and time to avoid collide the objects



3.

(1) Use Professor's code as base and change the trajectory function(piecewise2D function)

(2) Test the function one sphere by one sphere to avoid collide and get the coefficients for humdrums of times for velocity, position and time

(3) For every segmentation, the robotic will move to one point at first and then based on the relative distance between robot and the sphere calculate the trajectory lines.

(4) I used the wrong method for many days and when there is only one hours left I realized I did not implemented the spline.

(4) I only solve the first four spheres and when I tried to solve the fifth one every spheres I made before failed. Thus, I stuck for a long time.

https://drive.google.com/drive/folders/1iYM08Pb5HtdI_7te6aFCJoTT5wWWhPIv?usp=sharing

4.

(1) Use Professor's code as base and change the trajectory function (piecewise2D function)

(2) Test the function one sphere by one sphere to avoid collide and get the coefficients for humdrums of times for the position of different points and sphere.

(3) For every segmentation, the robotic will move to one point at first and then based on the relative distance between robot and the sphere calculate the trajectory lines.

(4) I tried many times to find the right positions, velocity and time for first sphere and after change the plane did not move any more. I did not achieve the goal.

(5) Due to use the wrong method, I did not get enough time to solve this.

(6) Next workshop I need to confirm the code and methods I implemented are correct. I can not make this stupid problem again.

Same link as above.