****

**MEEM 4707: Autonomous system**

**Spring, 2024**

**Lab – 3**

**By**

**Colton Kreischer**

**Problem 1.** What will be the last printed value of “count” before the while loop gets terminated?

|  |
| --- |
| rate=rospy.Rate(5)  Stop = False  count = 0  while ~Stop:  count = count+1  time = count/5  if time < 2  print(count)  else  Stop = True |

**While the code would not run as is (missing import, missing colons, misuse of tilde as not operator), the loop would theoretically continue until “1.8” is printed, when count is equal to 9.**

**Problem 2.** This file is named “controller\_lab3\_cw.py” and is present in the package named turtlebot3\_gazebo\_lab\scripts

**Text, letter

Description automatically generated**

In this code, “rate=rospy.Rate(20)” means the while loop will run 20 times per second. Also, the count will be added every iteration if you put “count=count+1” in the while loop as you did in problem 1.

1. If your velocity is 0.2m/s, rate=rospy.Rate(20) and count=100; where would the turtlebot be located?

***(100) / (20 Hz) = 5 s***

***(5s)(0.2 m/s) = 1.0 m***

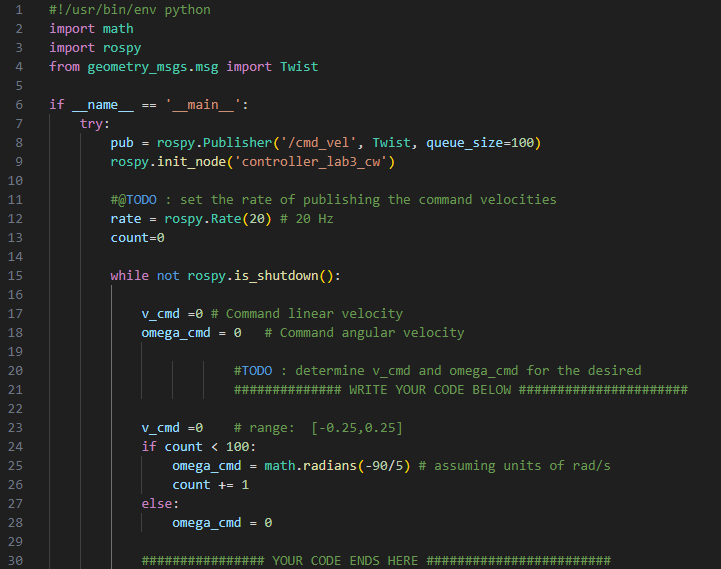
**After 100 counts (5 s), the robot would be 1 m from where it started at count=0.**

1. Try to make the turtlebot move straight forward 1m. (\*Show us your code)

A computer screen shot of a program code

Description automatically generated

1. Spin the robot clockwise until it reaches 90 degrees on the x-axis using “if / else” conditions in Gazebo. (\*Show us your code)



1. Repeat (2) and (3) to drive a square path. Record the trajectory of the robot in Gazebo and plot the trajectory. **Note: Your robot will not strictly follow square paths on the floor. They are for reference only. Do not try to make it run perfect square.** (\*Show us your code and the resultant path)

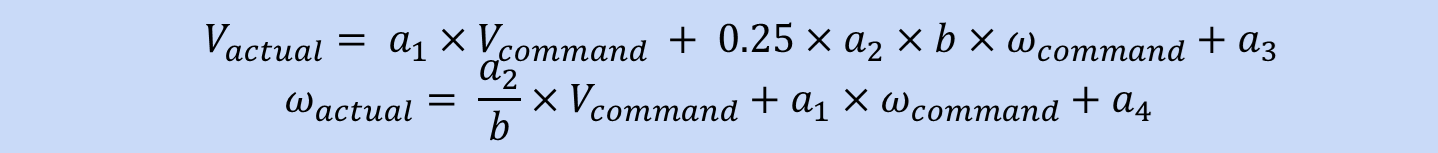
**A screenshot of a computer program

Description automatically generated**

**A graph of a line

Description automatically generated**

**Problem 3.** You can calibrate the robot’s performance based on the relationship between the command and actual velocities as below. Note that represents the distance between the wheels here.

****

**Calculate using path trajectory (the content of csv file) that you made in Problem 2.** Note: Refer to pages 17&18 Simulation2. Use the values obtained in Gazebo environment (You have 4 variables and 4 equations)

A graph of a line graph

Description automatically generatedA graph of a number of signals

Description automatically generated with medium confidence

A screenshot of a computer code

Description automatically generated

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Path | V (real, avg) | Omega (real, avg) | V (commanded) | Omega (commanded) |
| Straight | 0.1894 | -4.2364e-04 | 0.2 | 0 |
| Turn | 0.0121 | -0.2877 | 0 | -0.31416 |

0.1894 = (0.2)(a1) + (a3)

-4.2364e-04 = (1 / 0.287)(0.2)(a2) + (a4)

0.0121 = (0.25)(0.287)( -0.31416)(a2) + (a3)

-0.2877 = (-0.31416)(a1) + (a4)

a1 = 0.8850

a2 = 0.01326

a3 = 0.01240

a4 = -0.009667