Michigan Technological University

MEEM 4707: Autonomous system

Spring, 2024

Lab - 8

By

Colton Kreischer

Problem 1. In this lab, we will use a P or PI controller to run the robot.

(1) Please describe the P and I functions controlling the robot to the desired position.

The P and I functions calculate a system's response to error from a given setpoint. This is done by taking the some of arbitrary gains kP and KI multiplied by the error and by the integral of all past error, respectively. In this case, the error is the lateral distance from the robot to the target coordinate, relative to the robot's forward direction. Effectively, this operates as a heading-regulating PI loop.

(2) In Gazebo, reach the point (1, 0.5) in the middle of the map using the P or PI controller in your code. Submit the video and the P and I parameters you implemented.
NOTE: If you run "prelab8.py", it will generate "pidcont.py" automatically. Use these two codes to work in this lab. However, you need to tune your parameters in the "pidcont.py" file.

PID gains of (5, 0.5, 0) were used in the attached video.

Problem 2. Read the "prelab8.py" code script and summarize your understanding.

(1) What do "xc, yc, and tc" mean in the code?

xc, yc, and tc represents the robot's x, y, and angular positions, as reported by odometry callback.

(2) What do "rx and ry" mean in the code?

rx and ry are lists containing (x,y) coordinates for each target to be approached. Paired together, each set effectively creates a "setpoint"; however, only the lateral error "d" is used by the PID controller.

(3) Describe the meanings of "S and d" in the code.

S is the longitudinal error (distance from the robot to the setpoint coordinate, parallel to the robot's forward direction), and d is the lateral error (distance from the robot to the setpoint coordinate, perpendicular to the robot's forward direction). S is used to determine if a point has been passed (i.e. if it is less than 0.08 m), and d is used as the error for the PID-controller (to regulate heading, ideally keeping the lateral distance near 0 if approaching the target directly).

(4) Provide the variable name and command line in the code that defines the distance from the current robot position to the goal position.

```
82
83              for i in range(len(rx)):  # iterate through target coordinates
84                  # longitudinal error (forward distance)
85                  S = (rx[i] - xc) * np.cos(tc) + (ry[i] - yc) * np.sin(tc)
86
87                  # lateral error (sideways distance)
88                  d = (ry[i] - yc) * np.cos(tc) - (rx[i] - xc) * np.sin(tc)
89
90                  # move to next point if this point is near/behind the robot
91                  if S > 0.08:
92                      break
93
```

(5) Explain how this code can be integrated with the path planning code you implemented using the potential field method.

The generated paths can be input into this program by filling the (x,y) coordinates of each point into the rx and ry arrays, respectively. No further work should be needed, as this is where the example points are stored.