CMPE 565 AUTONOMOUS ROBOTS
Assignment 4: PID Controller

Deadline: 8 March 2018

In this assignment, you are expected to navigate a robot in a narrow corridor. This is an easier control problem compared to the common ones we encounter in our projects, e.g. lane following of an autonomous car. To make this problem just a little bit harder, we want you to control the robot while it is moving with a **constant forward speed**. The controller is expected to keep the robot on the track just by giving turn commands.

For any control problem, you are trying to keep the error as small as possible. Therefore, the general idea behind any controller is to generate an opposite action/movement against the error whose magnitude is proportional to the error. However, this simple controller is not enough to control complex systems. PID is one of the standard control mechanism and we will develop a **PID controller**.

Since any controller needs a measure of error, we will provide the error as follows:

1. Measure the distances of left N laser scans. (removing erroneous scans).
2. Calculate the mean of them.
    - This value corresponds to the robot's distance from the left wall.
3. Measure the distances of right N laser scans. (removing erroneous scans).
4. Calculate the mean of them.
    - This value corresponds to the robot's distance from the right wall.
5. The **error** is defined as the difference between the greater distance and the average of two means. This is a good estimate of the deviation from the midpoint.

Check the source code we provide for the implementation of the above algorithm.

1 Tips

- Note that derivative of the error is generally calculated as the difference between two consecutive errors: `derivative_error = error - lastError`

- Note that integral of the error is generally calculated as the sum of errors through whole scenario: `integral_error = integral_error + error`
- After you calculate error, derivate of error and integral of error, you will calculate a control signal (turn command) as follows: `turn_command = error * P + derivative_error * D + integral_error * I`
- You are expected to find the best values for PID constants (P, D, and I). You may use **Ziegler-Nichols method**.

You can also check following resources:

- `http://pages.mtu.edu/~tbco/cm416/zn.html`
- `https://en.wikipedia.org/wiki/PID_controller`

2 Deliverables

To complete this assignment, you should provide the following in your report:

- The link of the video in which you show the performance of your controller in vrep.
- 3 plots which shows the error curve.
- Another plot which shows the trend of the absolute total error (i.e. integral error). This trend shows how successful your controller is.
- Also another plot the last error value, when robot reaches to the end of the corridor.

3 Implementation

- Run **roscore**.
- Open the simulator and load **assignment4.ttt** scene.
- Edit the source code in *move.cpp* of moving_in_corridor to change the P, I, D parameters.
  - You will notice that the code we give you uses a predefined range of laser readings to calculate the mean distances from the walls. This is not the ideal way; since, as the robot turns to one side, the readings of this static and predefined range misleads us to calculate its distance lower than it should be to the side it turns into and greater than it should be to the other side.
  - Your controller would perform better if you use a dynamic range changing with respect to your current orientation. The robot's orientation is constatly published on the **robotPose** topic.
- Compile your code with catkin_make.

- Run the executable that you generate in the **moving_in_corridor** package.
- You can visualize the error by using **rqt_plot** package. It should be already installed with your ros full installation. Run it by tpying rqt_plot. Write /error as topic name box which is located in left top side. You can zoom in/out the axes. For that, first click the button shown below. Then while pressing on the right button of the mouse, drag the mouse to left for zoom out, to right for zoom in.