Zixuan(Kevin) Fan

Path Planning: Dijkstra's Shorest Path Algorithm

ROBOT LOCALIZATION

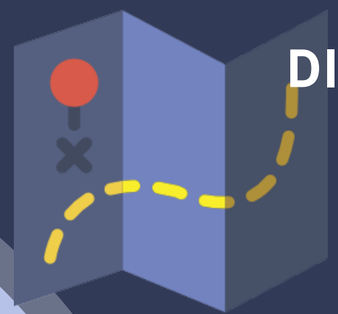
For the purpose of this project, we implemented an exact disretization grid system for the robot mapping. When we define the resolution to be large enough, we can calculate the position of the robot for every 0.5s.

During the time span, the robot can control its two PWM inputs of the continuous servo and make actions, such as turning and moving forward/backward.

```
Input : [0, 0]
Current Position [x, y] : [3, 10]
Estimated States [x, y, theta, x_hat, y_hat, theta_hat
0.0, 0.0]
Direction : +x
Map:
. . . . . .
. . . . . .
. . . . . .
. . . . . .
. . . A . . .
. . . # # # # # # . . . . .
. . . # # # # # . . . . . .
. . . . . # # # # # . . . .
. . . . . # # # # # . . . .
. . . . # # # # # . . . . .
. . . . . B . . . .
. . . . . .
. . . . . .
. . . . . .
```

PARTNERS

Zhuoran Duan
Likai Wei



DIJKSTRA'S SHORTEST PATH

The Python script calculates the shortest path to the destination using Dijkstra's Shortest Path Algorithm. The weight of each edge is all defined as 1 to its neighbor. Each time the robot got initialized, the shortest path will be planned for the robot and save the path to a Queue.

The robot then control its two PWMs on the two wheels to take actions according to the relative position of the next move to the current position.

EXTENDED KALMAN FILTER

The Extended Kalman Filter we implemented uses six state variables for the robot in 2-DOF space: its location x, y and angle θ relative to the reference frame and velocities x', y', and θ' representing the angular velocity of the robot. For this particular configuration of EKF, please view at https://github.com/kevinfan23/ee183da/tree/master/lab4

To view details, please visit https://github.com/kevinfan23/ee183da/tree/master/lab5