Shreya Gummadi

ENPM 661: Planning for Autonomous Robots

Project 3

**Project Statement:**

1. Generate a path from initial to goal position which avoids obstacles for a differential drive robot using Astar algorithm.
2. Simulate the final path generated on Turtlebot-2 with the RRL lab map.
3. Take initial position and goal position as user input.

**Files:**

There are two files for the project:

- Project_3.py
- testVrep.py

**Modules:**

The following modules are needed to run Project_3.py:

- Numpy ( to use the pi,sqrt,trig functions etc.)
- Matplotlib.pyplot ( to plot)
- Time ( to get the run time of the particular algorithm)
- Ipython ( to get the animation of the plots in an outside window)

To run the simulation:

- Vrep

Make sure to copy vrep.py, vrepConst.py and remoteApi.dll into your current directory from the V-rep programming folder.

**Run:**

- To run the programs just hit run and provide the necessary user inputs.
- Run testVrep.py along with V-rep. Make sure that simulation has been started in V-rep before running the python script to get the simulation.
- Set the initial position of the Turtlebot-2 manually before starting the simulation.

**User Inputs:**

- The initial and goal positions can be gives as: x_co-ordinate,y_co-ordinate
  Ex: initial position: 5,3  goal position: 10,50

**Animation:**

The program shows the nodes explored and the final path.

**Note:** In case, from IPython import get_ipython, get_ipython().run_line_magic('matplotlib','qt') doesn't plot the graph in a window instead of inline. Type %matplotlib qt in the ipython console.

**Implementation:**

The algorithm makes use of a class Node to store the position of the node, theta, left wheel velocity, right wheel velocity, its corresponding cost and its parent id.

The program has 5 helper functions:

- **Heuristic() :** It takes two nodes as its parameters and gives the Euclidean distance between the two.
- **Moves()** : This function takes two velocities and generates a 8 connected space with the corresponding cost for the movement. The two velocities are set to 50 and 100 rpm.
- **Exists()** :This function takes the node, as its parameters and determines if the node exists in the obstacle free space.
- **Diff_constraint()** : It takes left wheel velocity, right wheel velocity, current node and time-step as its parameters. The time-step is kept as 15 to make sure the code runs fast. The function computes the new x,y, theta and the velocity of the robot.
- **Generate_id()** : This function takes the node as its parameter and generates a unique id for it. The node is stored in the dictionary with this id as the key.
- **Obstacles()** : This function plots the map of the obstacle space.

The main functions is:

- **Astar()** : It takes co-ordinates of the initial and goal position and returns the co-ordinates of the nodes in the optimal path and the corresponding left and right wheel velocity . It first checks if the initial and goal node lie in the obstacle free space otherwise it tells the user that the following position is not possible. Next it generates the next node position based on the differential constraints. The Astar algorithm uses a heuristic cost to go. It takes the node with the minimum cost+heuristic_cost and expands that until we reach the goal. The heuristic cost in the program is the Euclidean distance between the current node and the goal node.

The left and right wheel velocities are stored as text files and are feed to the Turtlebot-2 left motor and right motor respectively to get the simulation in V-rep.