Lab 5 – Path Planning with A*
Posted: April 5, 2017
Due: April 17, 2017
100 Points Possible

Introduction

In this lab we will use A* to plan a path for our robot to follow on a map.

Setup Procedure

1. Download the .zip file from Blackboard for Lab 5.
2. First, we will copy everything over into our existing lab folder.
    - Unzip files from blackboard into the `ai_labs` folder
    - Then copy the python scripts into the scripts folder by running
        - `mv *.py scripts/`
    - We need to make the scripts executable, so now type (and don't forget the *)
        - `chmod +x scripts/*`
    - And we need to copy over the necessary files into the world folder.
        - `mv *.world world/`
        - `mv *.png world/`
3. Now we will test that this is all working. Run the following command from the terminal. You should see the simulator window pop up and a robot (circle) sitting there.
    - `roslaunch ai_labs lab5a.launch`
    - (If it doesn't work, you can try to run `catkin_make` from the `catkin_ws` folder or to run `source catkin_ws/devel/setup.bash`)
4. Now you can modify the code in `lab5_astar.py` and `lab5_waypoint_manager.py` to complete the rest of the lab.

Lab Overview

For this lab you will write code that will run A* to plan paths in four different worlds. This code will plan and save out a path that will be used by the other parts of the code to navigate the map. The other code consists of a `lab5_waypoint_manager.py.` Your code from lab 2 will also be used to move towards the waypoints and avoid obstacles. Make sure that it is working, and that the files are still in the correct `ai_labs` folders.

Part A [25 points]

1. Your first task will be to write the code in `lab5_astar.py` to generate all the neighbors for a given grid cell on our map. Look for "Lab 5 Part A Code START" to

see where to put your code. The code comments will help you know what to do, but you should, given a grid cell, return all the neighboring grid cells in a list.

Grid cells are represented as (x,y) tuples. To be a valid neighbor a grid cell must be touching the input grid cell and also be on the map and also not be in an obstacle.

Part B [30 points]

1. Now you will write the code for expanding a node in the A* code. Your task will be to write the code in `lab5_astar.py` in the place marked "Lab 5 Part A Code START". The code comments give you all the steps of this process. For each neighbor of the current cell, you need to create a new SearchNode, compute the relevant values that are needed for this cell, like its cost, heuristic value, and priority value. Then you will insert this node into the FRINGE (which is a priority queue).
2. Once you have completed this code you can run A* to generate paths across the maps. To do this, type (from the `ai_labs` folder)

```
python scripts/lab5_astar.py world/world.world
```

but replacing `world.world` with the path to one of the world files for this lab. They are

- `house1.world`
- `maze2.world`
- `maze3.world`
- `maze4.world`

When you run this code, your code will compute a path to the goal and an image will be saved in the world folder that show the path that was computed for that map. The start location is shown by a red circle, the goal by a blue circle, and the path by a green line. Once your paths look good for each map, then you can move onto the next part of the lab. The path will also be saved to a file which will be loaded by the other code in the lab.

Part C [19 points]

1. You will now implement the code that will load the path for a map and decide which waypoint to set for the current goal of the robot as it moves through the map. The file you will modify is called `lab5_waypoint_manager.py` and your code goes where it says "LAB 5 PART C CODE START". Once you have implemented this, you should be able to run the lab launch files and watch the robot move across the map. You can also modify the code in `lab5_waypoint_follower.py` to modify the behavior of the potential field code.
2. Your waypoint manager code can be tested independently of the path-planning code from parts A and B by running

```
roslaunch ai_labs pe6.launch
```

This will run your waypoint manager code on a map with a provided path.

3. Launch the code on each map, using the four lab5 launch files. The robot should make it to the green box indicating the location of the goal

    - `roslaunch ai_labs lab5a.launch`
    - `roslaunch ai_labs lab5b.launch`
    - `roslaunch ai_labs lab5c.launch`
    - `roslaunch ai_labs lab5d.launch`

Lab Question

1. Modify the priority calculation to have the search run Greedy Best-first search and uniform cost search for at least one of the maps. For each type of search, report the solution quality, the time it took for the search to run, and the number of nodes expanded in the search. What are the advantages and disadvantages of each search on these maps?

Deliverables

You should turn in (on blackboard) a .zip file with your `lab5_star.py` and `lab5_waypoint_manager.py`, and your lab write-up. The write-up should include (if possible) the images of the planned paths and screen captures showing the robot's trail/footprint showing the path it took to get to the goal. You may do this in multiple screen shots for a single map if the robot doesn't move fast enough to get the whole path at the same time (the footprints disappear after a little while).

Extra Credit

1. [10 points] Run Greedy Best-First Search and Uniform-Cost Search on all of the maps and report the results. Compare and contrast the different searches. Include the images showing the path for each search and map.
2. [10 points] Find or create a map for which Greedy-Best-First search computes a noticeably less-than-optimal path to the goal. Run your waypoint manager code on this map and path. Discuss the performance of your code on this less-than-optimal path. If the robot's performance is noticeably different than with an optimal path, discuss anything you could change in the waypoint manager code to overcome this. If the performance of the robot doesn't seem as bad as the path, comment on why this is the case. Include images of planned paths and actual followed paths to support your discussion.