CS 7638 Artificial Intelligence for Robotics
Warehouse Project
Fall 2019

Project Description

The goal of this project is to give you practice implementing the planning and control algorithms introduced in the "Search" lesson and the path smoothing in the 2nd half of the "PID Control" lesson by Professor Thrun. In each of the parts of the project, you will write a planner to guide a robot through a warehouse to retrieve and deliver packages.

In the first part, we will represent the warehouse as a grid in the manner of the planning problems from the Search lesson, and the robot and target packages will have rigid, blocky representations in this "gridworld." In the second part, the warehouse will have a continuous instead of a discrete representation, and the robot will have to navigate carefully around corners to avoid crashing into them.

Submitting your Assignment

Each of the parts is in a single file, warehouse.py. You should complete each class within the file to satisfy the specifications described therein (by implementing a planning function already present), and then you should submit this single file to the Warehouse Project Assignment on Canvas. Do not separate the parts and ensure it remains named warehouse.py . Submissions that do not follow this format will receive a penalty of -20 points.

The submission must consist of a single Python 2.7 file named warehouse.py . You may use external modules such as numpy, scipy, etc that are present in the Udacity Runaway Robot autograder. [Try to ensure that your code is backwards compatible with numpy version '1.13.3' and scipy version '0.19.1']

Your python file must execute NO code when they are imported. We encourage you to keep any testing code in a separate file that you do not submit. It should also **NOT** display a GUI or Visualization when we import or call your function under test. If we have to manually edit your code to comment out your own testing harness or visualization you will receive a -20 point penalty.

Suggested Schedule
All parts of this assignment are due Monday, November 11th, Midnight (Anywhere On Earth). We strongly suggest that you complete part A of the assignment before October 28th, giving yourself at least two weeks to complete part B.

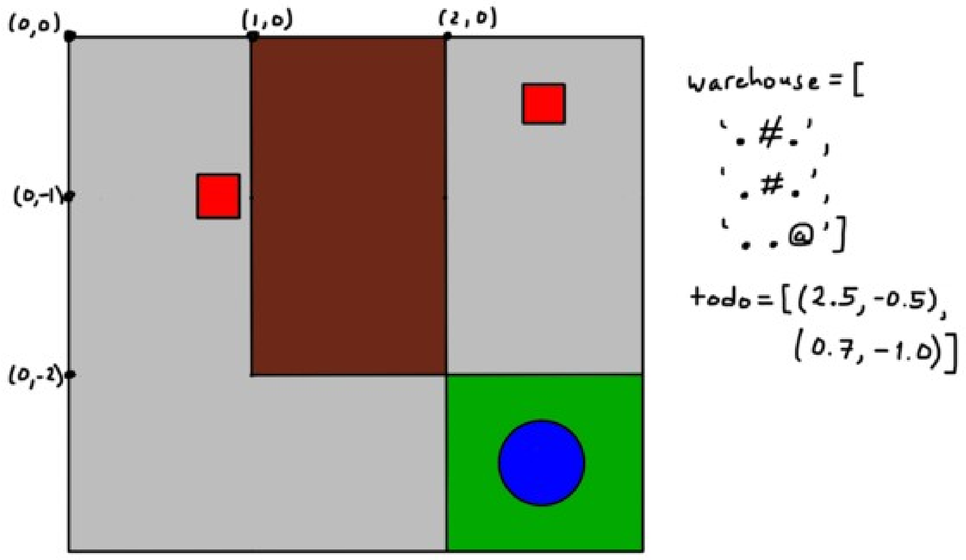
Testing Your Code
We have provided testing suites similar to the one we'll be using for grading the project, which you can use to ensure your code is working correctly. These testing suites are NOT complete, and you will need to develop other, more complicated, test cases to fully validate your code. We encourage you to share your test cases (only) with other students on Piazza.

You should ensure that your code consistently succeeds on each of the given test cases as well as on a wide range of other test cases of your own design, as we will only run your code once per graded test case. For each test case, your code must complete execution within the proscribed time limit (10 seconds) or it will receive no credit. Note that the grading machine is relatively low powered, so you may want to set your local time limit to 5 seconds to ensure that you don't go past the CPU limit.

Bonnie Auto-Grader
We are using the bonnie autograder system which allows you to upload and grade your assignment with a remote / online autograder. See the submit.py file posted as part of the assignment on Canvas for details. You are not required to use this online/autograder feature, but if you do, it will give you more assurance that your code will work correctly with our autograder when we grade the files you submit on Canvas. We may also choose to use the grade you receive via the remote autograder as your final grade at our discretion. (See the "Online Grading" section of the Syllabus.)

Grading
Part A and Part B have equal weight. Each part will be tested on a set of test cases. Each test case will have equal weight in the final score your project receives. As described in the warehouse.py file, each test case will have a "minimum cost" associated with it. If your planner fails to produce a valid route, you will receive zero credit. (A valid route delivers all boxes.) If your planner produces a plan that achieves exactly this minimum cost, you will receive full credit for that test case. If your planner produces a valid route of higher cost, you will receive partial credit as described in the python files. If your planner produces a route that results in a LOWER cost, you will receive extra credit for that test case. [NOTE: Your overall grade on the assignment will be capped at 100%, so this will not affect your grade on other assignments in the course.]

Example warehouse coordinate system for part B:



Academic Integrity
You must write the code for this project alone. While you may make limited usage of outside resources, keep in mind that you must cite any such resources you use in your work (for example, you should use comments to denote a snippet of code obtained from StackOverflow, lecture videos, etc).

You must not use anybody else's code for this project in your work. We will use code-similarity detection software to identify suspicious code, and we will refer any potential incidents to the Office of Student Integrity for investigation. Moreover, you must not post your work on a publicly accessible repository; this could also result in an Honor Code violation [if another student turns in your code]. (Be sure to set any repositories you use to Private.)