Path Planning (in 3D!)

MEAM 620 Project 1, Phase 2

Due: Tuesday, Feb 14, at 11:59 PM

For this assignment, you will write a planner which can generate paths from a start position to an end position in a 3D environment. You will need Matlab and must submit your code via turnin. See the course website for function stubs.

1 Mad Maps (10 pts)

Complete the implementation of `load_map.m` and `collide.m`. `load_map()` takes the filepath to an environment file and resolutions (size of each voxel grid) at which to discretize the environment and returns a representation of the environment for your planner. This representation can take any form that you like, but how you store the data about the environment can have a big impact on the performance of your code so think carefully!

The environment file format is straightforward. Here's an example:

```
# An example environment
# boundary xmin ymin zmin xmax ymax zmax
# block xmin ymin zmin xmax ymax zmax r g b
boundary 0 0 0 45 35 6
block  1.0  1.0  1.0  3.0  3.0  3.0   0 255   0
block 20.0 10.0  0.0 21.0 20.0  6.0   0   0 255
```

The format uses `#` for comments. The rectangular boundary extents for this environment are $(x_0, y_0, z_0) = (0, 0, 0)$ (lower left) and $(x_1, y_1, z_1) = (45, 35, 6)$ (upper right). A `block` line also specifies the rectangular boundary, and the blocks' color as a RGB triple with values from 0-255. All values are whitespace delimited and all numerical values are represented as a float. All distances are in meters. Blocks can overlap.

NOTE: Because this assignment is in Matlab – somewhat limiting the speed of your code – we will always coarsely discretize the z-dimension.

2 That Dastardly Dijkstra (50 pts)

Implement Dijkstra's algorithm as described in class. See `dijkstra.m` and `plot_path.m`. Things to note:

- In addition to returning the path, `dijkstra()` should return the total number of states that were visited.
- `plot_path()` should create a figure which displays the environment as well as the planned path.
- One of the main considerations of any planner is how fast it is. Part of your grade will be determined by how quickly your planner runs. The Matlab profiler will help you considerably. Also see this Mathworks page. For those of you know how to use C and MEX, please do **not** use those for this assignment. You are also **not** allowed to use any external libraries or implementations on this assignment.

3 A-Star is Born (40 pts)

Modify `dijkstra()` so that when the astar parameter is `true`, it uses distance to the goal as a heuristic to guide the search. Remember that any heuristic must be admissible and valid; if not, the algorithm is no longer guaranteed to return the shortest path.

4 Grading

For this assignment your grade will be determined by automated tests. You can see an example of the types of tests we will run in `dijkstra_test.m` and `collide_test.m`. These tests should run and pass if you type `runtests` from the Matlab prompt when you are in the directory that contains your code. We assure you that your code will be run against more tests than the ones we have given you. Feel free to develop your own tests as you go along using the information on this Mathworks page.

5 Submission

When you are finished, upload your code to Eniac and submit via `turnin`. The project name for this assignment is titled "`proj1phase2`" so the command to submit should be
"`turnin -c meam620 -p proj1phase2 -v *`". Your turnin submission should contain:

1. A `README` file detailing anything we should be aware of.
2. Files `dijkstra.m`, `load_map.m`, `plot_path.m`, `collide.m` as well as any other Matlab files you need to run your code.

Shortly after submitting you should receive an e-mail from `meam620@seas.upenn.edu` stating that your submission has been received. You can check on the status of your submission at `https://alliance.seas.upenn.edu/~meam620/monitor/`. Once the automated tests finish running, you should receive another e-mail containing your test results. This report will only tell you whether you passed or failed tests; it will not tell you what the test inputs were or why your code failed. Your code will be given at most 10 minutes to complete all the tests. There is no limit on the number of times you can submit your code.

Please do not attempt to determine what the automated tests are or otherwise try to game the automated test system. Any attempt to do so will be considered cheating, resulting in a 0 on this assignment and possible disciplinary action.