1 Dates

Assigned: September 23, 2005 Due: October 3, 2005

2 Introduction

In the previous project, you implemented a robot client to perform Monte Carlo Localization. You will use your localization routine to develop a path planning mechanism to navigate a robot to specific locations in a known environment.

3 Assignment

Once again, for this assignment you will be using a Pioneer robot outfitted with a laser to accomplish your task. The goal of this assignment is to use Dijkstra's algorithm to compute a good path from your current location to your goal. The goal will be specified by setting a 2x2 pixel area of the image fed to gzbuilder to red. Again, for this assignment you may not use truth data to help your localization and path planning, but you may use it to determine when you have reached your goal. We have given you a world file to work with, maze.world corresponding maze.jpg and maze.gzb files, which are located in /asgn/atrack/path. The command used to convert maze.jpg into a terrain file was:

```
>>gzbuilder -i maze.jpg -o maze.gzb -n -h .05 -v 1.2 -e 0.1
```

feel free to recompile if you wish. Also, you can make your own, comparable world if you would like.

4 Implementation

Building on the localization that you implemented for the previous assignment, you will be implementing:

- Dijkstra's algorithm to help you determine a good path from where you think you are to your goal
  - Here is a good reference for Dijkstra's, including links at the bottom to a Java applet that shows it graphically: http://en.wikipedia.org/wiki/Dijkstra's_algorithm
  - You will be distributing Dijkstra nodes around your environment and establishing connectivity between nodes. Connectivity, in this sense, means a "valid" (or collision free) path between nodes.
  - From your localization, you will determine which node you are closest to and then plan a valid path to the goal

- A driving method to help your robot reach it's goal
- If you want, another interesting environment in which to test your program

5 Deliverables

Once completing this implementation, you will turnin the following:

1. the **commented** source code and compiled executable for your client program
   - usage for the client must be obtainable from the command line and the source file header
   - build instructions should be available from the source file header
   - one source file is preferred, but not required
2. directions, scripts, or source for displaying coverage after client execution
3. world files, images, and other necessary files for your Stage and Gazebo worlds
4. a project writeup (refer to course missive)
5. refer to the course missive for the format of the electronic submission
   - one suggestion: a plot showing area covered by the robot on its path to the goal

5.1 Electronic Submission

You will submit your project deliverables via the cs148 handin script. The identifier for this project will be "advanced_path". The handis script can be invoked by the following:

```
>> cs148_handin advanced_path <directory to handin>
```

Refer to the missive for proper structuring of your project submission.

6 References

1 PSG User Documentation, http://playerstage.sourceforge.net/doc/doc.html

2 Dijkstra's, http://en.wikipedia.org/wiki/Dijkstra's_algorithm

3 Another applet showing Dijkstra's, http://carbon.cudenver.edu/ hgreenbe/sessions/dijkstra/DijkstraApp