CS148 – Project 3
Planning

1 Dates

- Project demonstration in class: November 16, 2006
- Project writeups due: November 18, 2006, electronic handin by 10pm

2 Introduction

The purpose of this project is to implement a deliberative control policy for the Roomba Pac-Man task. This deliberative control policy should allow your Pac-Man client to explicity navigate to food pellets and power-ups at specific locations in a given environment. You will have the option of implementing this project in simulation or in the real world.

3 Specification

For this project, your client will deliberatively plan and navigate paths in order for the robot to maximize the number of pellets eaten and power-ups visited. This client will use robot pose estimates given by your localization system from Project 2 for path planning. Your control policy will take in a location estimation (x, y, θ) and use it in conjunction with knowledge of the world (i.e., the map, sensor readings) to plan a path. Your robot should then be able to navigate this path, and then plan a new path once it has reached its destination.

For planning algorithms you have a number of options:

- Dijkstra's Algorithm[1],
- A*[2]
- Potential fields (Choset Ch. 4), Wavefront planning[3]
- Probablistic roadmaps [1] (Kavraki et al. 1996)
- RRT-Connect [2] (Kuffner et al. 2000)

Once you have implemented a planner, you will need to incorporate it into a controller for the Roomba Pac-Man task. For the Roomba Pac-Man task, your client should use the output of the planner and localization to determine when it has reached its current destination and situations where replanning the path is warranted. More generally, your client controller can use the output of the planner and any other information to functionally play Roomba Pac-Man. The decision making used in the controller need not be purely deliberative. You may want to consider using your planner in conjunction with your reactive modules from Project 1. In combining deliberation and reaction, you can consider behavior-based [3] and hybrid architectures.

As stated previously you have two options for implementation of this project:

[1] http://en.wikipedia.org/wiki/Dijkstra%27s_Algorithm
[2] http://en.wikipedia.org/wiki/A%2A_serach
[3] http://playerstage.sourceforge.net/doc/Player-2.0.0/player/classPlayerCc_1_1PlannerProxy.html

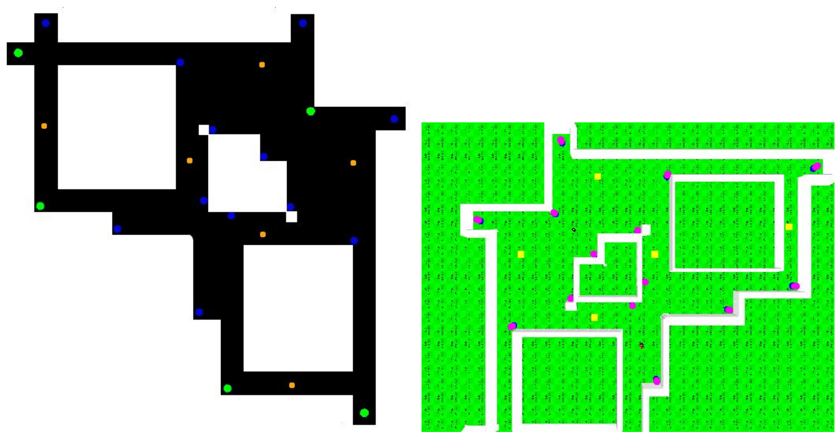

Figure 1: (left) The map for simulation implementation. The green dots are the power up fiducials, while the blue dots are fiducials to help with vision-based localization. The orange dots are the locations of food, they have corisponding fiducials in the real world. (right) A birdseye view of the simulation world in Gazebo.

3.1 Implementation in Simulation

We have provided a gazebo world file that you may implement your project in. This file has power-up fiducials and food fiducials throughout it. There is one other robot with a fiducial on it, representing a ghost, that will also be in the environment. We expect that your client will be able to navigate to the food fiducials and the power-up fiducials, while avoiding the ghost. In this implementation you will use your localization (laser based or vision based) from project 2 to plan your path. The recommended progression for this project is as follows:

- Implement a planning algorithm using the Truth Proxy. The truth proxy returns you the correct $(x.y, \theta)$ of your robot. You can use this to prove the functionality of your planner.
- Implement path navigation of your planned path without the ghost running.
- Replace the truth proxy with your location estimate from your project 2 localization. It **IS NOT** sufficient to implement this project using the Truth Proxy, we expect you to use your estimate from localization. If you feel you do not have functional localization from project 2, please contact the course staff **ASAP**.
- Incorporate the planning algorithm into viable Roomba Pac-Man control strategy, with a running ghost. To run the ghost type:

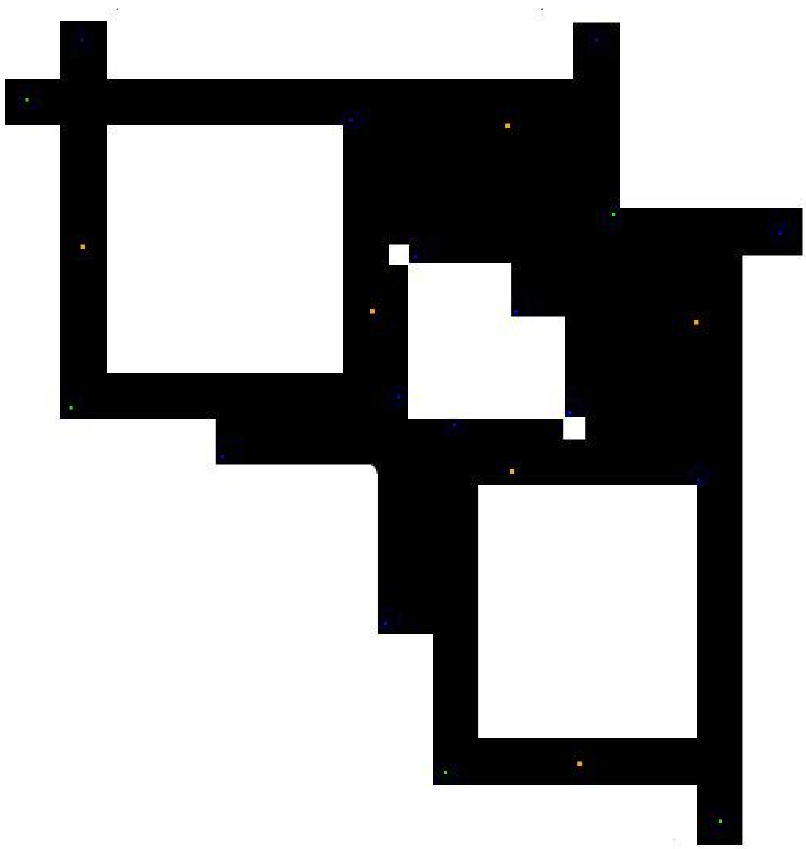

Figure 2: The 5th floor map for real world RPM implementation. The dots are drawn to scale of the size of the fiducials in the real world.

```
./ghost
```

. The ghost will subscribe to the second robot in the world, as well as the second robot's sensors.

3.2 Implementation on 5th floor

We have provided a map of 5th floor that you can use to plan your path. It contains the locations of the fiducials and food pellets. You will have to rely on vision based localization to plan your path. The recommended progression is as follows:

- Implement a planning algorithm in Gazebo using the Truth Proxy. The truth proxy returns you the correct $(x.y, \theta)$ of your robot. You can use this to prove the functionality of your planner.
- Implement path naviagation of your planned path in Gazebo without the ghost running.
- Replace the truth proxy with your location estimate from vision based localization in Gazebo, and turn the ghost on. To run the ghost type:

  ```
  ./ghost
  ```

  . The ghost will subscribe to the second robot in the world, as well as the second robot's sensors.
- Once you have functional vision based localization and planning running in Gazebo transfer this code to the 5th floor. We recommend using the Roomba Discoveries (the grey ones), as we found they had the most accurate odometry. When running at night, we have found that accurate distances to fiducials can be estimated to within 10 cm at a distance of 10 meters.
- Incorporate the planning algorithm into viable Roomba Pac-Man control strategy.

4 Support

4.1 Planning Algorithms

There is pseudocode available for Dijkstra's and A* search on Wikipedia. There are many verisons of Dijkstra's and A* available on the internet, you are free to use these just be sure to cite anything that is not your own.

4.2 Simulation

The environment from which you will localize your robot is a mockup of the CIT 5th floor. Illustrated in Figure 1, we have provided a Gazebo world file (proj3.world), Player configuration (proj3.cfg), and image raster map (proj3-Sim.ppm) of the 5th floor for you to use. proj3.ppm was used to generate the gzb file in proj3.world. The power-up fiducials are the green dots in the .ppm image, while the blue dots are other fiducials for localization. The orange dots represent food pellets, and have corresponding fiducials in the Gazebo world. The corresponding fiducials are on the ground and thin enough to be driven over. In the .world file we have added a second robot, a ghost (with a fiducial atop it), and provided a binary to command that ghost.

4.3 Real World

For real world implementations you should use the .cfg file from Project 1. Illustrated in Figure 2, we have also provided a map that represents the 5th floor of the CIT. Here are some important things to take note of:

- The conversion from pixels to meters in the real world is .0384375 meters/pixel.
- The fiducials are 56 cm tall.
- The field of view of the webcams are:
  - Horizontal = 35.82 Degrees
  - Vertical = 27.3 Degrees.
- Finally a Roomba Discovery is 34 cm wide.

5 Project 3 Deliverables and Grading

Project 3 involves two main deliverables, a demonstration of your planning client (in class on 11/16) and electronic submission of your work (project writeup, source code, and other materials). In particular, your project writeup should address design choices with regards to your planner as well as how you intergrate your planner into Pac-Man player. Of particular interest are how and when you decide to re-plan paths and how you know you've reached your destination as you navigate your path. Please refer to the course missive for details about electronic submission and project writeup format. Addendums to electronic submission:

- the writeup should be named "<login>_planning.pdf" (example, cjenkins_planning.pdf).
- other materials, such as a video of your client in action, should be put in a subdirectory named "materials/"

Both your handin and your actual robot will determine your grade. Your implementation and writeup will be scored in the following manner:

Implementation		Writeup	
Planner and Navigation	30%	Thesis/motivation	5%
Roomba Pac-Man Architecture	15%	Approach	15%
Performance	5%	Evaluation	10%
Real World Implementation	(up to) +20%	Discussion	15%
		Conclusions	5%

6 References

[1]L.E. Kavraki, P. Svestka, J.C. Latombe, and M. Overmars. Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces. IEEE Transactions on Robotics and Automation, 12(4):566-580, 1996
http://robotics.stanford.edu/ latombe/cs326/2004/class6/paper1.pdf

[2]J.J. Kuffner and S.M. LaValle. RRT-Connect: An efficient approach to single-query path planning. In Proc. IEEE Int'l Conf. on Robotics and Automation (ICRA'2000), pages 995-1001, San Francisco, CA, April 2000.
http://www.kuffner.org/james/papers/kuffner_icra2000.pdf

[3] Ronald C. Arkin Behavior-Based Robotics (MIT Press, 1998) ISBN 0-262-01165-4