CIS 390 Fall 2015
Robotics: Planning and Perception
Kostas Daniilidis
Project 2 Phase 1

1 Introduction

In this lab you will be making a particle filter for estimation of pose in environments with ambiguous measurements. Here the April Tag measurements will not all be distinguishable from each other, so the particle filter will be needed to keep track of the multiple possibilities. This phase will allow you to go into the next phase of the project where you will be combining this with the path planning algorithms to move through an obstacled environment.

2 Updated Codebase

You will be given simulation code in `ParticleFilterSim.py` to allow you to test your particle filter implementation. In the simulation you will be able to see "tags" - points with an orientation - relative to you within a certain angular radius in front of you. You will use these tags as the measurements to the particle filter. The interface is now:

- `get_measurements()` - This gives you the position, orientation, and ID number of the April Tags you can see in a list of lists of (x, y, θ, ID). Several of the points have the same ID number which is where the ambiguity comes from. The positions are given relative to you.
- `self.x_t` - This is a 3 by 1 numpy array (similar to the Kalman filter project) where you store your state (x, y, θ). This is what will be plotted in the simulation, and on the real robot what will be used as your estimated state.
- `self.plot` - member variable, boolean, to tell the simulation to plot or not. This gives you the ability to choose when to refresh your plot, since toward the beginning the particles will be scattered and the plots will not be clear. This is only relevant to the simulation.
- `self.particles` - This is a list of x, y, θ, and weights w that will be used to store and plot your particles in the simulation

For the simulation, in the `main` function you will need to pass in the map to the constructor. Several examples are given, and you are encouraged to try others. The format is a list of lists consisting of (x, y, θ, ID) in the global frame.

3 On the Robots

Before running the code on the robots, you will be making an environment and a map of it yourselves. The environment will consist of boxes with april tags attached on them. Since they exact positions will change from lab session to lab session, you will have to measure the relative positions of the April tags. The dimensions of the box and the positions of the april tags on the boxes are listed. As in the simulation, once you have made this map you will need to place it into the constructor of the controller code.

The code on the robots will have the same interface as the simulation.

4 Grading

This lab will be graded based on completion.