COMS W4733: Computational Aspects of Robotics

Homework 5

Due: April 22, 2019

This assignment consists of two parts. Part 1 should be completed and submitted individually. Part 2 may be completed either by yourself or with one other student. If working with a partner, please be sure to use the "Group submission" option on Gradescope.

1 Reading on SLAM (70 points)

Carefully read "Simultaneous Localization and Mapping: Part I" by H. Durrant-Whyte and T. Bailey (IEEE Robotics & Automation Magazine, 2006). Please provide responses to the following.

1. Equations (4) and (5) form the theoretical underpinnings of the SLAM algorithm.
   (a) Write out a verbal explanation of what each equation means, including descriptions of variables and distributions. We are looking for sentences that look like "the posterior probability distribution of X conditioned on Y is equal to the sum of ..."
   (b) Show that each equation is correct from derivation using basic axioms of probability, including the chain rule, the law of total probability, and Bayes' theorem.
2. Equations (8)-(11) describe the core of the EKF approach to SLAM.
   (a) Assuming a planar robot with position and orientation along with n landmarks, what are the dimensionalities of the quantities $\hat{\mathbf{x}}_{k|k}$, $\hat{\mathbf{m}}_k$, $\mathbf{z}_k$, $\mathbf{P}_{k|k}$, $\mathbf{Q}_k$, $\mathbf{R}_k$, $\nabla\mathbf{f}$, $\nabla\mathbf{h}$, $\mathbf{S}_k$, and $\mathbf{W}_k$? Please fix the typo in Equation (10).
   (b) Equation (9) gives us $\mathbf{P}_{xx,k|k-1}$, while we need $\mathbf{P}_{k|k-1}$ in Equations (10) and (11). What is the relationship between these two, and how do we obtain $\mathbf{P}_{k|k-1}$ given $\mathbf{P}_{xx,k|k-1}$?
3. The FastSLAM algorithm uses particle filtering instead of a Kalman filter model.
   (a) Explain the intuition (in a few sentences, no math) behind why map landmarks are independent conditioned on the robot trajectory. How is this related to the observation that particle resampling causes exponential loss of historical state information?
   (b) FastSLAM 1.0 and 2.0 differ in their proposal distributions, which lead to different importance functions for weighting. Explain the intuition behind the two different approaches. What does it mean to sample according to Equation (16) vs Equation (18)?
   (c) Would FastSLAM work properly if we never resample? Explain.

2 Implementing PRM and RRT (80 points)

You will be implementing the PRM and RRT algorithms. There is no ROS requirement this time, since any paths you come up with can ideally be fed into the same motion controller that you implemented in the previous homework to get a robot to move.

2.1 Data Files

The environment is 600 by 600, and the start and goal positions are defined in `start_goal.txt`. The `world_obstacles.txt` file contains information about the obstacles in the same format as the previous homework. The first line of the file indicates the number of obstacles. Each subsequent line corresponds to an obstacle, with the first number specifying number of vertices, and remaining numbers specifying vertex locations.

The provided `visualize_map.py` file reads in the data files and visualizes the environment with obstacles and start and goal locations. You may start with this file and add to it to visualize your PRM and RRT.

2.2 Tasks

1. Build a probabilistic roadmap and visualize it on the environment, along with the shortest path. As an initial approach, sample configurations uniformly and use k-nearest-neighbors to find edges. As before, you are may use external libraries to perform any subtasks, such as collision checking, finding neighbors, and (in the last step) graph search. The number of samples and number of nearest neighbors are free parameters for you to tweak; it is recommended you leave them as inputs for easy testing.

2. Implement any variation or extension of the vanilla PRM algorithm, either one discussed in class or one you find in the literature. Examples include a more sophisticated sampling approach, a different method of finding and connecting to nearest neighbors, OBPRM, or a PRM enhancement phase. Note down any differences and improvements that you see.

3. In a different file, build a rapidly-exploring random tree and *incrementally* visualize it on the environment expanding step by step, along with the shortest path. As an initial approach, sample configurations uniformly and start a tree at the start configuration. As before, you may use external libraries to perform any subtasks. The step size and bias are free parameters for you to tweak; it is recommended that you leave them as inputs for easy testing.

4. Replicate the tree construction at the goal configuration to implement a bidirectional RRT. The main addition that you will have to implement is the connect algorithm to connect the two RRTs together. The choices of when and how often to attempt a connection are left up to you (you may use parameters to make the connection work for this map, but please make your *implementation* general enough for any other map).

2.3 Writeup and Submission

Place any Python files that you create in `/hw5`. You should also include a README file for the code, if necessary, as well as formal writeup with responses to the questions below. Finally, please include two videos (or links to videos) of a single and bidirectional RRT growing toward completion of a full path. Be sure to submit the entire directory to Gradescope.

1. Briefly discuss your findings for the parameter values that worked best for the vanilla PRM on the given environment, and provide a screenshot of the result. How does performance of the algorithm improve or worsen as you change the free parameters? Look at factors such as path quality, sampling time (number of discarded samples), graph structure (disconnected components, unnecessary connections), and search efficiency.

2. Briefly describe your extension to PRM and the improvements you observed, if any. What choices of free parameters did you have and how did you choose them? Include a representative screenshot of a PRM generated with this extended method.

3. Briefly discuss your findings for the parameter values that worked best for the vanilla RRT on the given environment, and provide a screenshot of the result. How does performance of the algorithm improve or worsen as you change the free parameters? Look at factors such as path quality, sampling time (number of thrown away samples), space coverage (too much or too little overall or in specific regions), and search efficiency.

4. Briefly discuss your choices of when and how often you call the RRT-connect procedure for the bidirectional RRT. Did you notice any decrease in performance if called too rarely or too often? Provide a screenshot of the result of a bidirectional RRT (color the two trees with different colors).