CSCI/ARTI 4530/6530 Introduction to Robotics - Fall 2018
Assignment 2: Robot Sensing and Perception

Instructor: Ramviyas N. Parasuraman

General Information

Deadline: 12:30 pm, Oct 30 (Tuesday)
Worth: 80 pts (undergraduate); 100 pts (graduate)

This assignment is to make yourself familiar with the robotics perception, especially computer vision and range based perception systems, and to have a practical understanding of Robot Operating System (ROS) based software development of robotic perception solutions. Please be as specific as possible while writing the answers because your grade will be based not only on the correctness of the solutions but also the approaches you take.

Important note: This is a two-part assignment. This assignment is NOT a group assignment and everybody should work on it INDIVIDUALLY. I encourage you to discuss the problems in this assignment but you are expected to solve and write the solutions individually.

In order to complete this assignment successfully, you must first carefully read your own notes and associated slides on an robot perception.

Problems

PART I (40/60 pts) - Written/printed solutions

1. (10 pts) From a research paper of your choice that is highly cited ($\geq$100) in the field of Robotic Perception, explain the robot perception system outlined in the paper. Cite the paper and write your answers with respect to the specific questions below [2 pts each]. Restrictions: the chosen paper should be unique per student; papers chosen in Assignment 1 shall not be chosen again for this assignment as well.

   - What sensors are used and **why**?
   - What types of features are extracted from the sensor data (e.g., point features from images, line features from laser scan data, etc.) and **why**?
   - How those features are used? (e.g., what algorithms are applied on those features.)
   - What algorithm/method is used for localization or mapping or SLAM? (if the paper does not discuss one, then choose a different paper for Q1.)
   - What are the limitations of their proposed method/algorithm/sensors?

2. (10 pts) A sensor can provide you with a measurement of distance and bearing to a landmark as $z = (z_d, z_\theta)$. Both the measurements are subject to independent zero-mean Gaussian noise with variances σ_d^2 and σ_θ^2. For this sensor, what is the sensor model that provides the probability of a measurement z of a landmark observed by a robot from its pose (x, y, θ)?

3. (10 pts) Consider a drone with a camera that uses probabilistic color detection and mapping with the following specifications:

Sensed color	Actual color	p(Sensed | Actual)
red	red	0.8
red	green	0.1
red	blue	0.1
green	red	0.1
green	green	0.6
green	blue	0.2
blue	red	0.1
blue	green	0.3
blue	blue	0.7

Let the drone fly into a room containing 5 boxes: 2 red, 2 green and a blue one. The robot moves toward a box and it reads green. How likely is it that the box is actually green?

4. (10 pts) Explain Kalman Filter based localization in laymen terms without equations.

5. (20 pts **Graduate students only**) A robot represented by a point moves in an infinite uniform 2D grid. The coordinates of a grid node are (x,y), where x and y are two integers in Z. See the figure below. The *belief state* of the robot at a given time is the set of all locations (x,y) where the robot thinks it may be with non-zero probability. We assume that the robot is always in one of the positions listed in its belief state. So, for example, a belief state can be (-1,0), (0,0), (0,+1), meaning that the robot thinks it may be at (-1,0), or at (0,0), or at (0,+1). It's actual location is then any of these three locations.

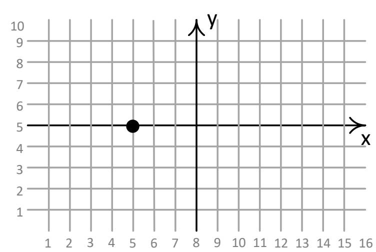

Figure 1: The bold dot at (-3,0) is the robot's initial position.

For this problem, we assume that the robot can execute only motion command, **move_right**. The intended effect of this command is to make the robot move right by one increment. But robot control is not perfect as we know, so that there is uncertainty in the outcome of this action.

The uncertainty model (given to the robot) of the action **move_right** is the following: by executing **move_right** from its current location (x,y), the robot will actually move to ($x + 1$,y), (x+1,y+1), or (x+1,y-1), or stay at (x,y). Each outcome has non-zero probability, but this probability is unknown. The robot can perform the action **move_right** as many times in a row as it wants. The robot has a position sensor, but this sensor is also imperfect. The model of sensing uncertainty (again, given to the robot) is the following: if the robot is at (x,y), then the sensor returns anyone of the 5 positions (x,y), (x-1,y), (x+1,y), (x,y-1), and (x,y+1), each with a non-zero (but unknown) probability.

Let the initial belief state of the robot be (-3,0). So, the robot is actually at (-3,0). Let the robot execute **move_right** once and then use its sensor to measure its location. The sensor returns (-2,-1). What is the robot's new belief state?

PART II (40 pts) Programming – Using laser scan data and RVIZ

Preliminaries

1. First, install the driver for Hokuyo laser scanner (urg_node ros package: follow the tutorial here: `http://wiki.ros.org/urg_node`)
   sudo apt-get install ros-kinetic-urg-node
2. Record a rosbag file of the LIDAR data for 2 minutes in a static environment.
3. Use the recorded rosbag file and visualize it in rviz (set the "Fixed Frame" to "laser" in the global options). Add a new topic (search by Topic in the add function and select /scan topic).
4. Follow the example of extracting lines python script from Prof. Andrew Vardy's course: `http://www.cs.mun.ca/~av/courses/4766-current/?page_id=84`.

Question

Extend the single line extraction script by applying split-and-merge algorithm we discussed in class (The segmentation part of the above link). For this, create a new script called "extract_all_lines.py" (as detailed here: `http://www.cs.mun.ca/~av/courses/4766-current/?page_id=378`).

The laser scanner publishes 'nan' value if the value is out of range of the scanner. Replace 'nan' with large distance (e.g., 10m) in your script. Note, currently this extract_singe_line.py script subscribes to \base_scan topic and not to \scan, which is published by the laser scanner driver. So, instead of changing the script, use a launch file to remap the topic names as we discussed in our class on teleoperating multiple robots in Gazebo.

Evaluate: Experiment with the parameters and aspects of the split-and-merge algorithm to try and improve performance on real data. Note your observations in the README file of your tutorial submission.

What and how to hand it in

You'll hand in the *typed or handwritten* answers to Part I in class to the instructor by the deadline. Please be sure to indicate the question numbers alongside the answers. The document should include your name, student id, and all the answers. You'll submit Part II of your assignment in a *single zipped file* by the deadline via eLC. The zip file should contain your *ros_package* called *yourlastname_a2*. (Please use all lower case letters in the package name). In a separate README file, describe how to compile and run your programs, and mention the output that you obtained in your evaluations.

Assignments that are **late** but within a day of the deadline will be penalized 25% of the total number of points. Two days of delay will be penalized 50%. Assignments submitted later than two days will not be accepted.