CSCI/ARTI 4530/6530 Introduction to Robotics - Fall 2018
Assignment 1: Introduction to Robotics and ROS

Instructor: Ramviyas N. Parasuraman

General Information

Deadline: 12:20 pm, Sept 10 (Monday)
Worth: 80 pts (undergraduate); 100 pts (graduate)

Assignment overview

This assignment is to make yourself familiar with the basics of robotics, kinematics, probability theory, and to have a practical understanding of Robot Operating System (ROS) based software development of robotic solutions. Please be as specific as possible while writing the answers because your grade will be based on the correctness of the solutions and 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 overview of robotics, kinematics, and basics of probability.

Problems

PART I (40/60 pts) - Paper and Pencil

1. (10 pts) Assume that an Influenza virus based on blood test is 95% accurate. Suppose that 1 in 100 of the subjects who gets tested is infected. What is the probability that a person is actually infected if his blood test comes out positive?

2. (10 pts) Explain the way how probabilistic localization works with an illustration of an arbitrary 2D environment (described by an $n \times n$ Boolean array) at any 3 instants of time. Provide supporting short explanations and the plots of the corresponding probability distributions. The probability of the robot being at the correct location is expected to increase over time/measurements.

3. (10 pts) From any research paper (conference or a journal publication only) of your choice published between 2013-2018, select a robot that interests you and for which sufficient details are available. Mention the publication and for the robot appearing in that paper, please briefly describe the following:

   (a) Its purpose, which may include a specific task or an application.

   (b) All sensors available to the robot. Be sure to indicate which sensors are active and which are passive.

(c) All effectors available to the robot. Also, please indicate the degrees of freedom of each effector, and whether you think that the robot is holonomic or not.

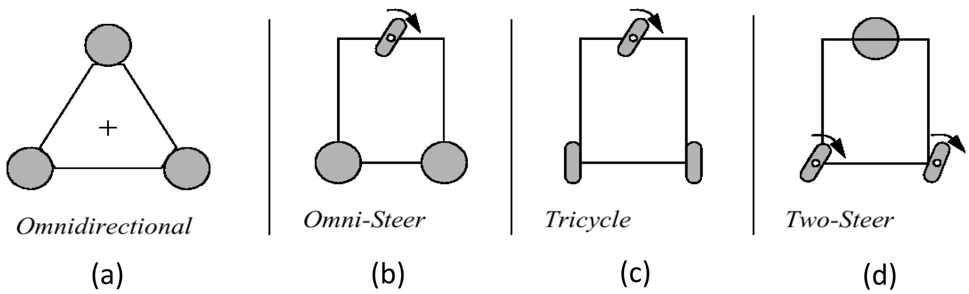
(d) Novelty that the paper brings compared to similar types of robots (commercial or research).

(e) What is the localization and/or mapping algorithm the paper uses/proposes/talks about.

4. (10 points) A robot uses a range sensor that can measure ranges from 0m to 3m. For simplicity, assume that the ranges are distributed uniformly in this interval. Unfortunately, the sensor can be faulty. If faulty, the sensor constantly outputs a range below 1m, regardless of the actual range. Let the prior probability for a sensor to be faulty be 0.01.

   Suppose the robot queries its sensor N times, and every single time the measured value is below 1m. What is the posterior probability of a sensor fault for $N = 1,2,3,\ldots 10$. Formulate the corresponding probabilistic model.

5. (20 points; **Additional question for graduate students**) Derive the forward and inverse kinematics of vehicle that is assigned to you. Explain your derivation step by step, following the constraints equation of the wheels using the same format and notations of the orientations and functions as discussed in the class.



PART II (40 pts) Programming – Introduction to ROS and stage_ros

Preliminaries

1. Please use a computer/Virtual Box (as used in the class) with Ubuntu 14.04 LTS as the OS with ROS Indigo (or higher). Go to `http://wiki.ros.org/ROS/Tutorials` and navigate to the tutorials for ROS **Indigo**.

   (a) Read *Beginner Level - Topic 1-6, 10, 12-13* tutorial for installing ROS, creating a workspace, building a workspace, creating a package, understanding ROS Topics, and creating a simple publisher/subscriber node in Python

   (b) Read and understand the tutorials. As you read, be sure to practice handy tools like rostopic, rosrun, rospack and catkin_make. For example, command roscd package_name takes you to the location of a package.

   (c) Use the following commands to install the required packages on your system:

   sudo apt-get install ros-indigo-turtlesim ros-indigo-stage-ros ros-indigo-rviz ros-indigo-navigation ros-indigo-navigation-tutorials ros-indigo-turtlebot-simulator ros-indigo-teleop-twist-keyboard ros-indigo-rqt ros-indigo-rqt-common-plugins

(d) RVIZ is a 3D visualization tool for ROS, included in the *ros-indigo-rviz* package installed above. Understand its features using this reference: `http://wiki.ros.org/rviz/UserGuide`.

2. STAGE is a general-purpose 2D robot simulator included with ROS. It provides a virtual world that can be populated by mobile robots and sensors, along with various objects for the robots to sense and manipulate.

   (a) Understand the operation of STAGE from reference for STAGE 4.0 at `http://rtv.github.io/Stage/modules.html`

   (b) The *stage_ros* package (included in *ros-indigo-stage-ros*) provides ROS specific hooks (wrappers) for STAGE. Read carefully the details of this package at `http://wiki.ros.org/stage_ros`. Go through each of the following steps to learn more about using *stage_ros*:

      i. Check using rospack that you have *stage_ros*, *rviz* and *turtlebot_stage* packages. Install again if the package is missing.

      ii. Open the first STAGE in ROS tutorial here: `http://wiki.ros.org/turtlebot_stage/Tutorials`

      iii. Bring up the robot in STAGE using the command given in the tutorial. You should see two windows: RVIZ and STAGE.

      iv. For each of the two *stage_ros* tutorials above, the STAGE simulator publishes several topics. Inspect the nodes and their topics using: rosnode list, rostopic list, rostopic echo [TOPIC], rqt_graph. Note: STAGE simulation can be paused by pressing P or using View - pause.

      v. In RVIZ, modify the size of laser-scan rangefinder (named Kinect) to make scans clearly visible.

Questions

(a) (20 pts) For this question, you will need to first understand the semantics of *geometry_msgs::Twist* using these references: `http://docs.ros.org/api/geometry_msgs/html/msg/Twist.html` and `http://docs.ros.org/api/geometry_msgs/html/msg/Vector3.html`. Run turtlesim (in STAGE) and perform the following:

   i. Create a new ROS package named "my_turtle_yourlastname" inside the *src* folder of the workspace you have created in class/previous tutorials.

      A. Write a publisher node (Python script) that makes the turtlebot move with a linear speed of 0.2 units until it hits one of the walls. Call it **Move_Turtle.py** and save it in the *scripts* folder of your my_turtle_yourlastname package. [Hint: Use the topic *cmd_vel* to send a *geometry_msgs::Twist* message to the turtle.]

      B. Make your script executable by doing sudo chmod +x Move_Turtle.py .

   ii. Run this node using the command line rosrun my_turtle_yourlastname Move_Turtle.py.

   iii. Now, add a subscriber node (Python script) called **Listen_Turtle.py**, which listens to the turtlebot's velocity commands on the topic */turtle1/cmd_vel*, and converts the linear and angular velocity to the wheel velocities of an arbitrary differential drive robot using the inverse kinematics example discussed in the class. Publish the calculated wheel velocities (commands for a differential drive robot) on a topic called /DIFFERENTIAL_INVERSE_KINEMATICS_OUTPUT. Assume the parameters of the differential drive robot such as wheel radius (5 cm), separation (1m).

Evaluate:

i. 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 (you can submit it in the class on 09/10). 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 *my_turtle_yourlastname*. 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.