COMP 581: Introduction to Robotics

Fall 2018
Instructor: Prof. Ron Alterovitz
Learning Assistant: Armaan Sethi <armaan@cs.unc.edu>

Lab 3: Moving Around an Obstacle

Deadline

October 31, 2018 at the beginning of class. For the late policy, see the course syllabus. **Before the due time, each team member should individually submit your team's code in a zip file on Sakai**; see specific instructions in the Code section below**.**

The Objective

You will create a robot that determines the location of an unknown obstacle, traces the boundary of the obstacle, and returns to where it started.

Robot Design Requirements

Your robot must be built only using pieces from your Lego Mindstorms kit. Your should design your robot such that the dark gray center button may be pressed without moving the robot and such that pushing the button does not disrupt any of the sensors. Your robot should remain in one piece at all times, its diameter with respect to the ground should not exceed 40 cm, and its diameter shall not change substantially during the task. All measurements will be made with respect to the *measuring point*: a distinct point of your choosing that is on the **frontmost surface** of your robot. More specifically, your robot will be placed on the ground behind a starting line drawn on the ground, and your chosen measuring point must be a point on your robot directly above the starting line, and no part of your robot is permitted to be in front of the starting line. Your robot should be designed to accomplish the task autonomously without any human intervention (except for pushing the dark gray center button as required).

The Task

You will place your robot behind a starting line such that its measuring point will be on a specific *starting point* on the starting line. Directly in front of your robot will be an obstacle. The front wall of the obstacle (i.e. the wall of the obstacle facing your robot when it is at the starting point) will be located between 30 cm and 1 m away from the starting point. The point directly in front of your robot and 20 cm away from the obstacle's front wall will be known as the *hit point*.

To begin, you will push the dark gray center button on your robot. Your robot should then move straight forward until it locates the obstacle. To indicate that the robot has found the obstacle, it should beep when it is less than 30 cm of (or in contact with) the obstacle. The robot is now near the hit point. There is no need for your robot to stop, although it may. Your robot should then turn right and travel completely around the obstacle with its measuring point staying within 30 cm from the obstacle at all times. After the robot returns to the neighborhood of the hit point (e.g., anywhere within 30 cm from the hit point), the robot should turn away from the obstacle and return to the start point.

In this lab you can assume that the obstacle is a closed shape comprised of straight and curved segments. It is not necessarily a convex shape. It will be comprised of bumpable, ultrasonic reflective walls that are at least 17 cm tall. The perimeter of the obstacle will be between 1 m and 6 m.

This lab is to be completed in a single run, without pressing any buttons after it has begun or picking up your robot (except as defined in the Points section below). All robot motion must be completed within the feasible workspace, which we define as all space within 2 meters of the obstacle. The task must be completed in under 2 minutes.

Points

Points Awarded:

- Attempt: 40 points. You will receive these points simply by having LeJOS successfully installed, showing up to the Robotics Lab on the lab due time, submitting your code on Sakai **on time**, and having your robot attempt the course.
- Find the hit point: 15 points. Points will be awarded for finding the obstacle and beginning to follow its wall. You will receive these points if your robot beeps when it is within 30 cm of (or in contact with) the obstacle.
- Percentage traced: 30 points. Points will be awarded based on the percentage of the obstacle's perimeter that the robot successfully traced while staying inside the required distance from the obstacle wall. The percentage will be computed approximately as (the amount of obstacle boundary traced until the robot either violates the distance constraint to the boundary or returns to the hit point)/(the total perimeter of the obstacle).
- Return to start point: 15 points. Points will be awarded for returning to and stopping at the start point. Points will be awarded based on accuracy (i.e., how close the robot's measuring point is to the start point when the robot stops). These points can only be earned if your robot circum-navigated the obstacle without being picked up.

Penalties:

- There will be a 10 point penalty if, after crossing the start line, your robot exits the feasible workspace.
- You may, at your discretion and only once, pick up your robot, press a single button, and put it down either at the start or where you picked it up. If you choose to do this, you will receive a 10 point penalty. Note that the 2 minute timer continues unchanged.

- If your robot does not stop by 2 minutes, then your run will be terminated and no further points after 2 minutes will be earned.

Code

To complete this lab you will use a subset of the leJOS API:
http://www.lejos.org/ev3/docs/
Important: You may only use classes from the following packages in Java and the leJOS API: any class in the java.io, java.lang, java.util, lejos.utility, lejos.hardware, and lejos.hardware.* packages. Note that the classes must be implemented in one of these packages; a class that simply inherits from one of these packages is not permissible. For example, you cannot use a class in lejos.robotics.

A robot relying on a Java class from a package not listed above will be disqualified. Also, any robot using Wi-Fi or Bluetooth communications in any way will be disqualified.

You must submit on Sakai a zip file of your Java source code files. The zip file should include the Java source code files you wrote (which should have a .java file extension). Do not submit Java class bytecode files (which have a .class file extension) or JAR files (which have a .jar file extension). In the comments at the top of the Java source file containing your main method, please include the names and PID's of all team members.

You should feel free to discuss concepts in the course with other students in natural language (e.g. English). You should not share any programming code with others and must write all the robot's programming code yourself. If you access any sources other than the textbook or documents on Sakai, you must cite them in comments at the top of your code and send an e-mail to the instructor with the citation. You must fully understand your code and be able to reproduce the algorithmic approach without references if asked. The Honor Code is in effect for these policies.

Good luck!