ELEE 4200/5200: Autonomous Mobility Robotics
Term I, 2018
Homework 7: Wall Following

Guidelines:

- Due date: Thursday, November 12, 2018 by 12 Noon.
- Each group of no more than two students must work on its own in completing this assignment! Feel free to consult with others (and with me and the TAs) in developing solution ideas, but the final implemented code must be your work product alone. Refer to the Syllabus, where the policy on academic integrity is clearly outlined, our classroom discussion on this topic, and consult with me if you have any questions!
- State the full names and T# of the students in the group on the cover page of every document that you submit.
- Submit the report by responding to this assignment posting in Blackboard.
- The submission should at least include the following documents, bundled together into a single zip file with the name *YourNameHW7* (use one of the group member names).
    - The main report (following the template provided).
    - The main report in 'pdf' form.
    - The MATLAB program code.
- A hard copy (printout) of the 'pdf' report with MATLAB code; staple all pages together and follow the TA's instructions on how to submit.

Goals:

a) To investigate the use of "wall following", a very useful type of behavior in robotics, as navigational guidance for a robot. This is what we do when we walk down a hallway or walk down a pathway (in the latter case, the edges of the pathway act like walls!).
b) To do so with the use of motion feedback (output) rather than calculations based on drive commands (input); that is, you are required to construct a simple control algorithm to drive the robot using wall following. In this case, the motion feedback should be based on sideways spacing of the robot to the wall.
c) To create a map of the outer perimeter of a structure, with the help of basic mapping concepts, while driving autonomously using wall following.

Specific Tasks:

a) Design a control algorithm based on the wall-following algorithm to drive the Turtlebot around the outside of the given "island" structure (as shown in Figure 1 – you will be provided with a separate Gazebo environment for this assignment). The robot needs to maintain an approximate spacing of 1.5 meters from the wall and keep the wall to the right.

The robot has to keep moving – that is, stopping and turning is not allowed, but you can drive at any forward speed, ‘v’, that you want (within the Turtlebot’s limitations).

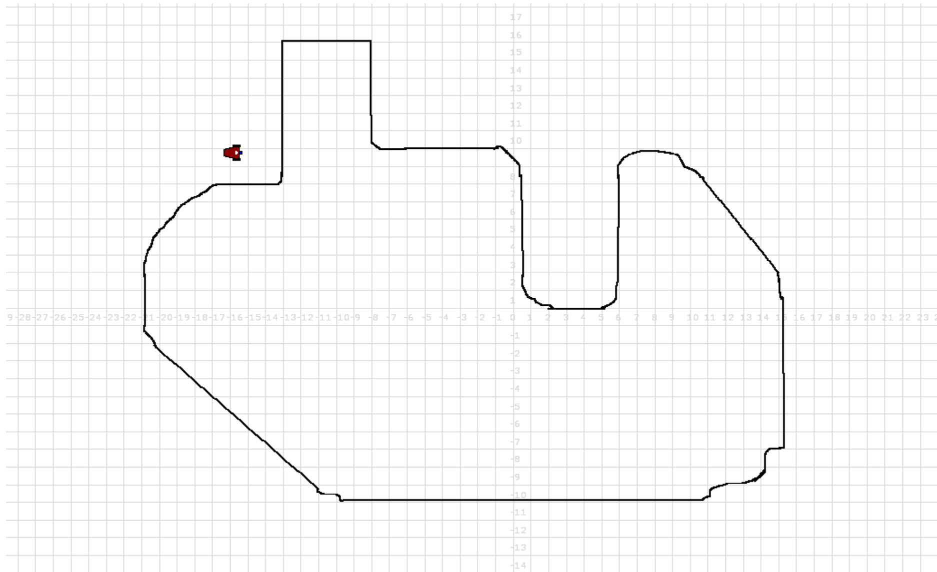

Figure 1: "Island"

b) Record and report the distance traveled by the robot over one full circuit around the island using both model state and odometry. Also, plot the robot path around the island using both model state and odometry.

c) Repeat parts (a) & (b) keeping the wall on the left.

d) Repeat parts (a), (b), (c) by starting the robot on the inside of the wall (both ways – left wall following and right wall following). The purpose of this part of the exercise is to see whether you can follow the wall, while simultaneously avoiding hitting the wall if it curves sharply in front of the robot.

e) Develop a map of the “coastline” of the island and also estimate and report the length of the coastline. While the robot is also required to be autonomous on this mission, you are free to select run parameters. It is not necessary that you include wall-following to accomplish the mapping task! The main focus here is to understand the effects of poor localization on the consistency of the created map; you don’t have to fix it.

f) Include meaningful videos of robot runs that you feel are appropriate and support your descriptions in the report.

Other Related Information:

- ~~Use the given “set model state” commands to bring the robot back to the same starting position for a new run (for left and right wall following).~~

- In mapping the coastline, remember that the robot can only see what its LIDAR sensor reveals at that time. Specifically, the view as seen in Figure 1 is not available! You don't have a map – that is why you are looking for one!
- You need to think about how many and which of the LIDAR rays you will use in your algorithm. A single ray might be subject to wide variations (large sensitivity), particularly when you consider the presence of small openings in the wall. Averaging several neighboring rays might be better, but how many should you average? Averaging too many rays might make the result somewhat insensitive to distance from the wall. Also, you might need to consider how this consideration changes for the wall-following versus the mapping task? Additionally, what is the best standoff distance for the mapping task (for the wall-following task it is specified!)?
- Looking at rays ahead of the robot might help anticipate what is coming.
- Is a wall-following operation necessary for the mapping task? OR, is it enough to get views from several perspectives to construct the map?
- Explain your algorithm clearly in your report! Just a reminder that it is not just the final results that are of interest here. You are also responsible for explaining how you arrived at your program and the attendant robot behavior. As I have said before, *the process is more important than seemingly correct results.* This is embodied in the following sequence – an algorithmic idea translated correctly to code, examination of the attendant behavior of the robot in multiple experiments, modification of the algorithm to change behavior desirably, etc.

Notes for myself: Some questions worth considering (if for no reason than the spirit of enquiry)

a) Is a continuous wall-following maneuver necessary for mapping or is it enough to look at the island from several different perspectives? What if we want the map generated as fast as possible?

b) Is it better to be closer or further from the shoreline while attempting to “grab” the map?

c) How do we consolidate the various “hits” from the different rays of the LADAR if we decide that use of multiple rays is a good move?

d) What is the math necessary to carry out this task?

e) What extra steps would be needed to convert the map that you obtained into an occupancy grid-based map?