CIS 32.5 Fall 2009, Project 2

1 Description

This project involves some exercises in dead-reckoning navigation.

2 Dead reckoning

1. We will start from the same place as before. With the setup in the world you see when you type:

   `player world.cfg`

   As before this should pop up a square window labelled **Player/Stage: ./world.world** which contains a grey dot and some strangely shaped lumps. This is the simulated world in which your robot will operate.

2. You have a slightly different starting point this time. Rather than `roomba-roam.cc` you have `roomba-roam2.cc`.

3. You can see the difference if you look at the code — this time you have access to the information about how far the robot thinks it has gone.

4. Now, what I want you to do is to edit the original version of

   `roomba-roam2.cc`

   so that the robot makes a circuit of the obstacle just to the left.

5. Start from the position in the unedited config file (which has the robot start moving down and to the right.)

6. Remember that you compile the controller in `roomba-roam.cc` using:

   `./build roomba-roam2`

7. When you are done, save your program as **proj2-part1.cc** and make sure you put your name in the comments.

8. You'll need to submit this to Prof. Parsons after you are done with the project.

3 Keeping track

1. If you didn't yet try it, use the information about how far the robot has gone, along with information about where the robot starts (which is in the file `world.world`) to have the robot estimate where it is at all times.

2. You can check how good a job you do since you can see from the simulator where the robot is.

3. Now try to get the robot to navigate round the obstacle based on this information.

4. Hard isn't it?

5. At the point where you give up on this, save your program as **proj2-part2.cc** and make sure you put your name in the comments.

6. You'll need to submit this to Prof. Parsons after you are done with the project.

7. Remember that the mark you get for doing this will be determined after running the code.

4 Magically we know where we are

1. Now try running:

   `player world2.cfg`

   and then:

   `local-roomba`

2. The robot will look as though it is doing exactly the same as before (it is), but there is one important difference.

   The values the robot is printing out give its location.

   The robot knows where it is!

3. Now get it to move around the obstacle.

4. This is basically a cheat. Stage is telling the robot where it is, the robot isn't figuring its location out on its own.

5. As we will see in later weeks, it is possible for the robot to figure out where it is, but it is non-trivial and not particularly accurate.

6. When you are done, save your program as **proj2-part3.cc** and make sure you put your name in the comments.