CISC 3415, Project-5 (**7-pt**)

1 Description

This project builds directly on the last one. Now that we can get the robot to localize, we can look at navigation. This project involves following a plan. Since a plan is a series of points, it should be simple to adapt the code you wrote for Project 4 to handle this task.

2 Start up

1. Login to your computer (user name is `student`, password is `student`), open terminal windows, get a copy of the project files (**project5.tgz**) from the instructor, put this on the Desktop and extract the files.
2. The control program is in the file `follow-plan.cc`, so first build that:

   ```
   ./build follow-plan
   ```

   then run player on the config file `world5.cfg`:

   ```
   player world5.cfg
   ```

   and start the robot controller in another window:

   ```
   ./follow-plan
   ```

3. The robot, as it has at the start of previous projects, will trundle off to the south-east.

   (Note that the robot is localized — for simplicity it uses the fake localization we used before in the first part of Project 4).

4. However, before the robot starts to move, the controller will display what we'll call a *plan*.

3 Follow a plan

1. Your task is to take a plan that gives a sequence of locations for the robot and make the robot move through this sequence of locations. Proportional control should be used to reach each of these waypoints.
2. The precise plan format is described in the next section.
3. The robot should come to a complete stop when it gets close to the final location.
4. While the plan that you are following should have been created in such a way that the robot can move in a straight line from one point to the next without hitting an obstacle.
5. However, depending on the technique you adopted in Project 4 for moving to the goal point, you may find that you need additional points in the plan. In that case, you can modify the plan in **plan.txt**.
6. After your code has worked, you should now include code for obstacle avoidance. (You never know when a faulty map or path-planning algorithm might have been used in creating the plan that your robot is following). You can use the laser or the bumpers, or both to do obstacle avoidance. To test this functionality, you need to use the following config file which uses a slightly different map that includes an obstacle near the destination:

   ```
   player world51.cfg
   ```

7. The plan in the file `plan.txt` should be suitable for testing the robot for most of you. If you need to make modifications, please do not add waypoints in an attempt to plan around the obstacle. That is, treat the obstacle as unexpected, and use obstacle avoidance to get around it.

8. When you are done, save your program as ***(your-names)*-proj5.cc** and make sure you put your name in the comments.

9. You'll need to submit the .cc file to me after you are done with the project, and make sure to include **plan.txt** if you modified it. But, please do not submit the whole folder.

4 Plan format

1. A plan is an integer $2n$ followed by $2n$ doubles.

2. The integer indicates how many coordinates there are in the plan (so it always has to be $2n$).

3. Each pair of doubles is a set of coordinates, and x value and a y value, in that order (so there are n x values and n y values).

4. The coordinates in my example follow the coordinate system used by Stage. This has the origin in the center, and has coordinates ranging from –8 to +8.