CSE 468/568 Lab 1: Robot Control and Using `tf` in ROS

The objective of this assignment is to get started with robot simulation in ROS, and use `tf` to relate the coordinate system of one robot with the other.

Before you get started, please make sure you have done the following:

- Complete ROS tutorials 1 through 12 (in Section 1.1 Beginner) and be able to write a simple publisher and subscriber in ROS

Simulation Setup

In this section, you will learn about launch files. Create a package named `lab1` that depends on `std_msgs rospy roscpp`. Download the world and launch files for this assignment from here. It contains three files - `playground.world` and `playground.pgm` which together form the initial stage world, and `lab1.launch` that is the initial launch file.

Please read the roslaunch tutorial to understand the `roslaunch` file. Drop the world files and the launch files in the appropriate sub-directories in the new package. Test to see if you can launch stage using the launch file by running the command

```
$ roslaunch lab1 lab1.launch
```

Note:

- You might need to re-run the path commands after creating the package for `roslaunch` to identify the new package `lab1`.
- You don't need to run `roscore` when you use `roslaunch`. However, you do need to have `roscore` running when you use `rosrun`

Evader Controller

In this section, you will write your own controller. Familiarize yourself with the given robot by checking the information it publishes and subscribes. The robot also has a laser range finder attached to it. Move the robot by dragging it with your mouse pointer to face a wall. Monitor the output of the laser by using `rostopic echo` in the command line. You should be able to understand the output of the sensor from the stage world file and its output you observed.

Write a controller node that drives the robot straight at a constant speed of 2m/s. When the robot is close to an obstacle, the robot should stop, turn in a random new direction, and drive at the same speed. Create a new launch file `evader.launch` that adds the execution of this node to what was in the earlier launch file.

Pursuer-Evader

Read through the `tf` tutorial. Publish the coordinate frame of the robot wrt the global frame. Create a new world file with a second robot called `pursuer`, and drop it into the world close to the first robot. Write a controller node for the pursuer that subscribes to the `tf` messages from the evader, and follows the evader by going to the spot it was at from one second before. Create a third launch file `pursuer-evader.launch` to run the new world with the two robots, the evader controller, and the pursuer controller as separate nodes.

Submission Instructions

You will submit `lab1.tar.gz`, a compressed archive file containing the lab1 folder. The folder should contain the two world files and the pgm file in an appropriate sub-folder, two launch files `evader.launch` and `pursuer-evader.launch` in an appropriate sub-folder, and two controllers - one for the evader and another for the pursuer in an appropriate sub-folder. The folder should compile if I drop it into my catkin workspace and call `catkin_ make`. Please take care to follow the instructions carefully so we can script our tests, and not have to dig into each submission. Problems in running will result in loss of points.

Please use the `submit` script for submission using the syntax

```
$ submit_cse468 lab1.tar.gz
```

or

```
$ submit_cse568 lab1.tar.gz
```

depending on whether you are taking cse468 or cse568 respectively.

Details on the usage of the submit script can be found here.

The assignment is due Tuesday, Sep 20 before midnight. It should really not take you more than a 4-6 of hours to finish this assignment once you have done the tutorials.