Introduction to Robotics - CMSC 479/679
Homework #1
Due Monday, February 20^{th} at the start of class

Every roboticist knows that, in addition to being lots of fun, working with robots can at times be frustrating. Because designing and building robots requires coordination of mechanical, electrical, and software components, there are many possible failure points. To paraphrase the philosopher Karl Popper, it's a good idea to "let robot simulations die instead of real robots". Therefore, many simulation environments exist. In some, the goal is to simulate specific robots (such as the Pioneer or Khepera); in others, the goal is to accurately simulate the physical world. In this homework assignment you will get experience working with the latter type of simulation environment by implementing a simulated SumoBot. You'll also build intuitions for what kinds of "tricks" might make one SumoBot better than another, which will be helpful when you start building a physical SumoBot.

The simulation environment is called breve (pronounced "brev-ay"). As the documentation for breve says: "breve is a free simulation environment designed for multi-agent simulation. breve allows users to define the behaviors of autonomous agents in a continuous 3D world, then observe how they interact. breve includes support for a rich visualization engine, realistic physical simulation and an easy-to-use scripting language. breve can be used as a tool to explore any type of simulated world. breve has been used for a wide variety of simulation applications: simulated virtual creatures, artificial ecosystems, simulations of molecular biology, visualization and much more. breve facilitates the construction of complex agent-based simulations by automatically handling agent communication, representation in 3D space, graphical rendering, physical simulation and a number of other features which are useful to agent-based simulations."

Here are your tasks:

- Download breve from the following location:

  http://www.spiderland.org/breve/download.php

  Command line breve version 2.7.2 (released 2/25/08) exists for Mac OS X, Linux, and Windows. Download whichever distribution is appropriate for your machine and install it. Installation instructions are in the file README.txt in the top level breve directory. In most cases, you need do nothing more than set the `BREVE_CLASS_PATH` environment variable appropriately. Browse through the documentation for breve which is available via `index.html` in the `docs` directory.

  There are a number of demos in the `demos` directory. Files ending in `.tz` are breve programs. For example, there is a program that builds random creatures trying to find one that can "walk". Run it using this command line:

  ```
  ./bin/breve ./demos/Physics-Examples/Creatures.tz
  ```

  The simulation can be started and stopped by pressing the space bar. Look through the demos and feel free to take and use code from them.

- Part of breve is an object oriented language, called steve, used to write simulations. I've written some steve code to get you started. It resides in a file named SumoBot.tz and can be found on the course web page (as can a soft copy of this assignment) in the syllabus entry for February 6^{th}. Get this code and make sure that you can run it. You'll modify this code in the next task.

- As with the physical SumoBot, your goal is to get the simulated SumoBot to move using only 3 servos. Modify SumoBot.tz to achieve this goal. There are two sub-tasks here:

  - First, there is a class in SumoBot.tz called Servo. It inherits from RevoluteJoint, which is a built in class that can be used to easily model a freely spinning wheel. Read and understand this class.
  - Second, there is a partially complete class named SumoBot that creates part of a SumoBot body. You must flesh out this class so that the final body can locomote using only 3 servos. This will

require you to add more steve code to build the body, and the iterate method to control the servos to construct an appropriate gait.

- When you first get a SumoBot that can locomote, make two copies of that class and call them, for example, SumoBot1 and SumoBot2. Modify the SumoBotArena code so that rather than creating two SumoBot instances it creates one SumoBot1 and one SumoBot2. The result will be two identical SumoBots fighting against each other, which is often rather boring because they will probably meet in the middle of the arena and become gridlocked. Now start "tweaking" SumoBot1 to see if you can improve it. For example, experiment with the masses of the links, make the legs shorter or longer, make the servos rotate faster or slower, add a wheel or two (that are not powered). The goal is to build intuitions for ways in which you can make the SumoBot better by modifying the physical configuration and properties of the simulated body.

What to submit for grading:

- Your modified SumoBot.tz. I know it isn't good programming practice, but please keep all of your code in one file. Not much code is required for this assignment.
- Using the breve Movie class, or any other tool you like, make a movie of a match between your SumoBot1 and SumoBot2. Submit the movie.
- Write up the results of your experimentation with ways of improving your SumoBot. What did you try? What worked and what didn't work? Why? Did you learn anything in this exercise that will translate to the physical SumoBot? Why or why not?

The movie should be submitted using BlackBoard. I prefer hard copy of the source code and write up. The name of the assignment in Blackboard is "Homework 1". Instructions on how to upload files for assignments can be found here:

`https://wiki.umbc.edu/pages/viewpage.action?pageId=5244774`