Homework 2:

Robotics Overview, Locomotion Studies, Braitenburg's "Exploration" Vehicle

---

Assigned: Thursday, Sept. 4, 2008
Due: Wednesday, Sept. 17 at 23:59:59 (i.e., before midnight)
(any homework turned in after 00:05:00 on Sept. 18 will be considered late, and will receive a grade of '0')

Part 1: Overview of Robotics. Read the paper "Silicon Babies", from *Scientific American*, December 1991 (handed out in class), which discusses the challenges of artificial intelligence (AI) and robotics. List the main topics of AI/robotics mentioned in this paper.

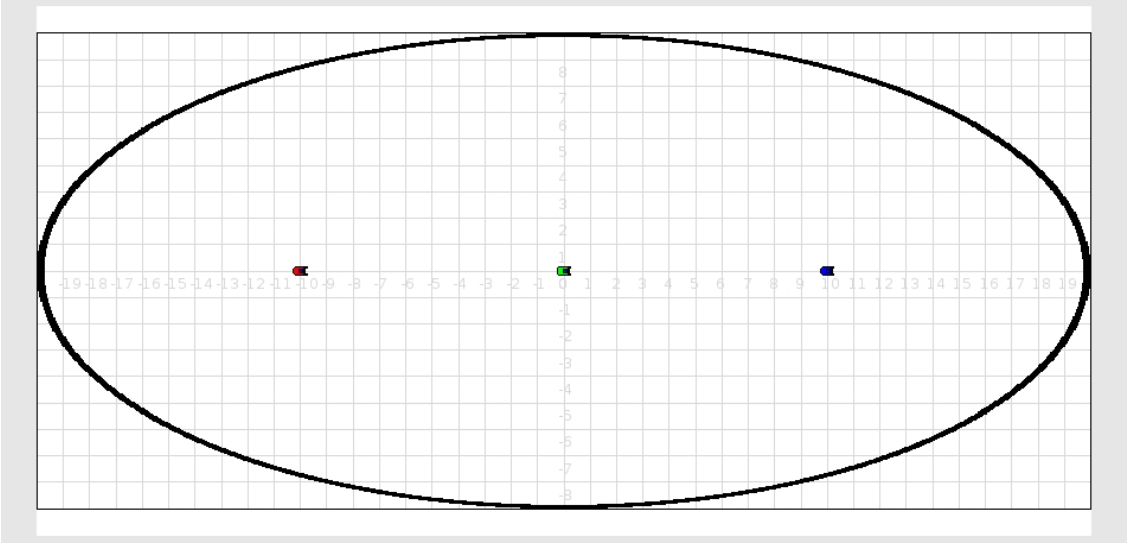
** Graduate Credit Only: For each topic, give an example of a robot or researcher that addresses the topic, by giving a current URL that describes that robot or that research activity.

Part 2. Locomotion. In this part of the homework, you will explore the impact of the wheel mechanism on the control of a robot. You will experiment with 3 robots side-by-side, but each with a different drive mechanism. For this exercise, we will use a "Pioneer2dx" type of robot, but we will define 3 variations of this robot – one with an "omnidirectional" drive, one with a "car" drive, and one with a "differential" drive. On the course website (under Homework Assignments) in the tar file HW2-files.tar.gz, you are provided with a configuration file (HW2-Part2.cfg), a world file (HW2-Part2.world), and a file that defines the three types of robots (robots.inc). You are also provided with the shell of a robot control program, called test-wheels.cc, that commands the robot simply to move forward at a specified turn rate. In this assignment, you'll use the exact same control code for each robot, which will be your modifications to the test-wheels.cc program.

Your job is to experiment with different velocities and turning rates, to determine how the hardware/wheel model of the robot affects its motion. (For this exercise, you do not have to include obstacle avoidance capabilities for the robots.)

Running your code for multiple robots:

When you run your experiments, each of the 3 robots will be running a separate process, which will run identical copies of the same program test-wheels.cc (with your own modifications) in the same window. To control multiple robots in the same simulation, do the following. Start up Player/Stage as always (i.e., "robot-player HW2-Part2.cfg"). Using the files I've given you, this will be what the starting environment looks like:



To run the robot control code, open up 3 separate windows, one for each robot. Connect each window to the directory where you have your compiled robot control codes. Then, enter the following commands, each in its own separate window:

- For controlling robot #1 (which is the omnidirectional robot):
  linux> ./test-wheels –p 6666

- For controlling robot #2 (which is the car robot):
  linux> ./test-wheels –p 6667

- For controlling robot #2 (which is the differential drive robot):
  linux> ./test-wheels –p 6668

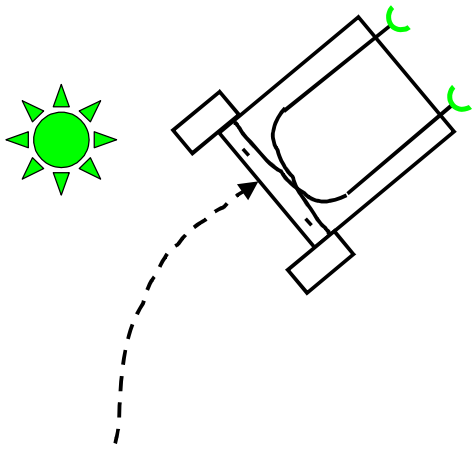
Note that the "-p" option specifies the port number being used by that robot. These port numbers are defined in the HW2-Part2.cfg file. These commands will then connect to each of the 3 separate robots, and your 3 robots will execute the control code.

What to submit for Part 2:

- Submit a screenshot that shows the 3 robots moving according to the same robot control code, but illustrating the different behaviors of the robots because of their hardware constraints. (Use "View -> Show Trails" to show the robot motions; turn off any sensor readings, such as ranger data (this will also help your simulation run faster).)
- State the speed and turn rate you used to create the screenshot. Discuss your findings, and why the robots behave as they do. Did you find differences in the motions for all 3 types? If so, explain why. If not, explain why not.

Part 3: Braitenburg's "Exploration" Vehicle (see lecture notes on Aug. 28th). In this programming exercise, you will implement the equivalent of Braitenburg's "Exploration"

vehicle, which is attracted to a light, but is always exploring . The abstract Braitenburg "Exploration" behavior is illustrated here:



In this exercise, the "light" will be a "fiducial" in Player/Stage. (Note: a *fiducial* is a special marker that can be easily detected with some appropriate sensor. For example, the fiducial could be a reflective barcode that can be detected easily by a laser. Or, it could be a distinctive visual feature that is easily detected by a camera. For this exercise, we don't really care exactly what the fiducial looks like, or what sensor is used to detect it. We'll just use the capabilities provided for this purpose in Player/Stage).

In order for the robot to detect the fiducial, it needs a "fiducialfinder", which is a sensor that detects fiducials (hence the name!). In fact, for this Braitenburg vehicle, the robot actually needs two fiducials, in order to differentially detect the strength of the light. Here, the fiducialfinder returns the range to the sensed fiducial. For this assignment, you should convert this range measure to an equivalent (arbitrary) light intensity. Note that the intensity of light falls off as the square of the distance from the target, whereas range is linear. So, to properly create the Braitenburg vehicle behavior, you'll need to map the range to a light value that falls off as the square of the distance.

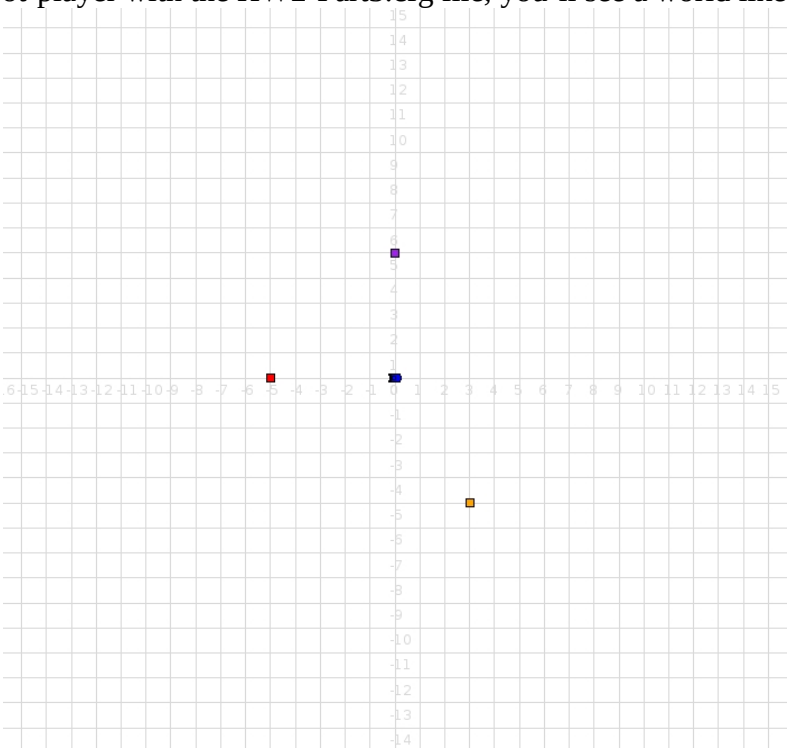
To help you out in setting everything up, I've provided a world model and a configuration file for your use in this part of the homework, called HW2-Part3.world and HW2-Part3.cfg, available in HW2-files.tar.gz. Check these files closely to be sure you understand how these new definitions for fiducials and fiducialfinders are added. You are free to make changes to the parameters or beacon locations, as needed to illustrate your robot's "Exploration" behavior.

(FYI: Here's the online documentation for the world file, which includes comments on parameters specific to the fiducials: http://playerstage.sourceforge.net/doc/Stage-2.0.0/group__model__fiducial.html, so that you can understand what the various parameters mean).

In this exercise, we'll use the same "rink" bitmap used in part 2 above. However, so that you don't have to deal with obstacle avoidance issues, we've expanded the size of the environment, so that walls are far away (i.e., you can't actually see the sides of the rink). For this exercise, you should only focus on the Braitenburg "exploration" behavior; you don't have to write an obstacle avoider. A starting point for your code is provided to you in HW2-files.tar.gz, in the

example code "fidRand-2.cc". Try running this code as provided (along with HW2-Part3.cfg), and turn on "View->fiducial config" and "View->fiducial data". Here, you'll see the robot detecting the fiducials with its two fiducialfinders.

If you start up robot-player with the HW2-Part3.cfg file, you'll see a world like this:



What to submit for Part 3:

- Add to the pdf file a screenshot that shows your robot performing the "Exploration" behavior, with a sufficient length of robot trace to make it obvious what behavior the robot is generating. (You'll also be submitting your files that create this behavior; see below.)

SUBMITTING YOUR HOMEWORK:

Place all your files in a single directory. These files should include:

- Your written answers and screenshots as requested above, in a single pdf file called "yourlastname-HW2.pdf")
- Your configuration file for Part 3, called "HW2-Part3.cfg"
- Your world file for Part 3, called "HW2-Part3.world"
- Your makefile, called "makefile" or "Makefile" for Part 3
- Your robot control code for Part 3, called "yourlastname-HW2-Part3.cc" .
- A README file giving the command line arguments to run your code (it may simply be "./yourlastname-HW2-Part3", but if you require command line arguments, please specify them here).

Remove all other unnecessary files. Use the submit script **494mr_submit** or **594mr_submit** to submit your files. (These will be emailed to Dr. Parker.)