# CS 30 Lab 14 — Robots in Pyro

Copy the files for today's lab using the following Linux commands:

```
cp -r /common/cs/cs30/lab14/ .
cd lab14
ls
```

Start up Pyro by typing `pyrobot` at the Linux prompt. Click Server and choose the Pyrobot Simulator with the Room world. Next, click Robot and connect to the PyrobotRobot60000 robot. Finally, click Brain and navigate to your **lab14** directory, then choose **NNPredict.py**. This brain uses a neural network to control a robot after it has been trained by a teacher. Open the brain in an editor window by clicking on the file name next to the Brain button. If you get a "gedit" window instead of IDLE, call me over so I can fix your settings.

The teacher is represented by the `teacher()` method, which returns a motor action (speed and rotation) in response to the robot's current sensory state and its previous motor action. This action is used as the training signal for the neural network. After training, the neural network itself is used as the source of motor actions. The network has 18 input units: 16 for the sonar sensors plus two for the robot's previous motor action (speed and rotation). It has two outputs that specify the motor response of the robot to its current input state. It also has a hidden layer consisting of 5 units.

Let's train the robot to move forward and slow down as it approaches a wall. We'll then test the trained robot in a situation which it was not trained on, to see how well it generalizes what it has learned.

1. Turn learning off by typing `self.net.setLearning(0)` at the Pyro command line in the interaction window. Position the robot near the south wall facing north and click Run. When learning is off, the neural network is in control of the robot, but since it hasn't been trained yet, the robot won't do much at this point. Try placing the robot at various locations in the environment, including near walls and in corners.

2. Now click Stop and turn learning on by typing `self.net.setLearning(1)`.

3. Position the robot near the south wall of the world facing north and click Run. This will begin training. The teacher will drive the robot forward while updating the weights of the neural network with the corresponding motor actions (provided by the `teacher` method). When the robot stops near the north wall, click Stop.

4. Repeat the previous step several times until the error produced by the network (shown in the Pyro interaction window) approaches zero for the duration of the robot's travel. It may actually take quite a few repetitions of this process for the error to reliably go down.

5. Now turn learning back off by typing `self.net.setLearning(0)` at the Pyro command line in the interaction window.

6. Position the robot near the south wall again and click Run. This time the robot should respond by moving forward and then slowing down as it reaches the wall. Try this from various other places in the world (you don't need to click Stop each time while running with learning off, since the weights of the network aren't being updated). What happens when the robot is placed very close to the middle of the north wall, facing the wall? During training, it never experienced this particular situation, but what does it do now in response? Does its behavior seem reasonable to you? What about in corners?

*(continued on back)*

7. Since a large number of training episodes may be necessary in step 4 for the robot to learn the task well, I've included a set of pre-trained weights that you can try out. Load in the weights from the file **trained-room.wts** by typing the following command in the interaction window:

   ```
   self.net.loadWeightsFromFile('trained-room.wts')
   ```

   Then make sure learning is off and try step 6 again.

8. Exit Pyro by choosing Exit from the File menu, then restart it with the **BigRoom** world. (using the Pyrobot Simulator as before). Create a new brain that traces out a circle in the room. Next, change the brain so that it causes the robot to trace a figure-eight. Finally, see if you can get the robot to trace out a square. Hint: you can use the `time.sleep(n)` function from the `time` module to pause execution for *n* seconds. To turn a corner, set the rotation speed of the robot to some value and then pause for some number of seconds (you'll need to experiment with different values to see what works best).

9. Create a brain to follow the hallway in the **AndrewHallway** world. The robot should move south through the hallway from the starting location, staying as near to the center of the hallway as possible. Once your robot can do this, see if you can get it to turn around when it reaches the end of the hallway and return to the starting location.

10. Start up the Pyrobot Simulator with the **Braitenberg** world. This world contains a single light source, represented by the yellow circle. The size of the circle represents light intensity. Try out the various **BraitenbergVehicle** brains available in Pyro with this world.

11. The **Braitenberg2Lights1Robot** world contains two light sources instead of just one. How do the various Braitenberg vehicles behave in this world?