CISC 7412 Fall 2011, Project II

Description

The project is to implement value iteration in the context of a robot finding its way around a grid world.

Start with a skeleton

The file:

```
value-iteration.nlogo
```

which you can download from the course website (projects page), implements a sinple grid world. When setup is run, the simulation creates a 20 by 20 world with a robot, a goal, and a variable number of obstacles.

When you hit "go", the robot moves randomly — at every step it picks between moving north, sourth, east and west with equal probability. When it reaches the goal, it stops.

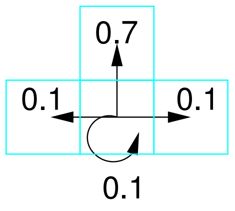
Implement value iteration

The idea of this project is for you to use value iteration to have the robot find the goal in a more intelligent manner than moving randomly.

You should do this by using the *value iteration* method that we covered in Lecture 8, and you should implement two versions of this.

For the first, assume that the robot moves deterministically, so that in the update on page 29 there is no need to do the weighted sum over the possible outcomes of a given action — you can just look up the utility of the state that the action takes you to.

For the second, assume that the robot moves non-deterministically. In particular, assume that the robot moves as follows:



That is when it attempts to move in any direction, it has a probability of 0.7 of moving that way, a probability of 0.1 of moving at right angles to the direction it selected, and a probability of 0.1 of remaining where it started.

This time you'll need to do the weighted sum in that update calculation.

Show the results

You should use the facilities that Netlogo provides to give some indication of the results of value iteration. For example, you might choose to color the patches so that they reflect the utility of each non-obstacle patch. Or you might use a shape from the shape library to show the action chosen by the optimal policy for each grid square.

Make the robot run

Given the values established by the value iteration, the robot can now figure out how to behave optimally — it chooses the action that maximises expected utility (page 26 of the notes). Add this ability to your program, both for the case where actions are deterministic and those where the actions are non-deterministic.

Provide controls

You should have implemented:

1. Value iteration using deterministic actions.
2. Value iteration using non-deterministic actions.
3. Robot movement using deterministic actions.
4. Robot movement using non-deterministic action.

Make sure that you have the right controls on the interface to let me choose to run all of these options, and make sure that the documentation (see below) explains how to do this.

Document it

All you are going to hand in is the Netlogo program so make sure you:

1. Write lots of comments in your code. If I don't understand what your code does, you won't get full credit.
2. Write a description of your program in the Information tab. This is where you should explain how to run the various options, and how you chose to display values and/or policies.

   Include in the description your thoughts on the effectiveness of value iteration and the behavior of the robot when executing the resulting policies.

Hand it in

Save your model as `<my-name>-value-iteration.nlogo`, where you replace `<my-name>` with your own name (so my program would be called `parsons-value-iteration.nlogo`) and email it to me at `parsons@sci.brooklyn.cuny.edu`.

The subject line of your email should say: 74010, Project 2.

If you don't get an acknowledgement within 24 hours, send me a follow-up email.

The due date for the project is December 17th.