

Project 1:

Reinforcement Learning for Robot Navigation

Assigned: Tuesday, Jan. 24
Due: Sunday, February 12, 2005, 23:59:59 (i.e., before midnight)

Overview

In this project, you will use the Sarsa(λ) online reinforcement learning algorithm to teach a robot to navigate to a goal position without running into obstacles. You will be using the Player/Stage simulator to simulate your robot. You'll be using a laser range scanner and odometry on the robot to perform the learning, where the robot moves using steering and velocity commands. You'll turn in your software that performs the learning, instructions for running your software, and a paper (3-6 pages) that describes your project and results.

Formulating the problem

As part of the project, you are responsible for formulating the details of the learning problem. For example, you'll need to determine how you will represent the input state and the motor output, and the best level of discretization of these values. You'll need to decide the reward function you'll use; presumably you'll have a positive reward for reaching the goal and a negative reward for running into an obstacle. You can decide yourself where the goal position and start position of the robot will be for each learning trial. You'll need to determine how many episodes are needed for your program to learn.

Some caveats:

- You are not allowed to use a grid-world representation of states. States should be a function of sensor values (and other measurements, as appropriate).
- For full credit, your approach should not have intermediate rewards for states that do not represent running into an obstacle or reaching a goal (this means you'll have to implement the multi-step temporal credit assignment approach of Sarsa(λ)). [However, although it is discouraged, if you can't get your program to work otherwise, you will still receive partial credit if you assign intermediate rewards.]

Player/Stage Simulator

On Thursday, 1/26, a tutorial on Player/Stage will be given. Full documentation on Player/Stage is available online, at <http://playerstage.sourceforge.net/doc/doc.html>. A shared version of Player/Stage is available for you on our UTK CS Linux machines (e.g., cetus0-9) in /research/playerstage, so you should not download any software to your directory. A sample program that shows you how to read the simulated sensor values and how to control the robot motion will be given. Three test environments will be set up for you in this directory, in /research/playerstage/project1, which represent small-, medium-, and large-sized environments for your robot learning. At a minimum, your learning project should work on one of these environments (the larger the environment, the better), which involves scattered small obstacles. Your program is not required to work in more complex environments, although you are welcome to create your own environments, as well, if they help you illustrate the strength of your learning approach.

Any questions you have on Player/Stage should be directed to the TA (Michael Bailey, mbailey@cs.utk.edu).

Paper Guidelines

As part of your project, you must prepare a paper (3-6 pages) describing your project. Your paper should be formatted using common word processing software (such as LaTeX or Word), and should include the following:

- An abstract of 200 to 300 words summarizing your findings.
- An introduction describing the robot learning task and your formulation of the problem, including the definition of the state, motor actions, and reward.
- A detailed description of your experiments, with enough information that would enable someone to recreate your experiments.

- An explanation of the results. Use figures, graphs, and tables where appropriate. Your results should make it clear that the robot has in fact learned. This means you'll need to give some pre-learning results with which to compare your learning approach. Ideally, results are quantitative, not qualitative. This means that you have some concrete measure by which to evaluate your results.
- A discussion of the significance of the results.

Undergraduate Grading

Your grade will be based primarily on the quality of your project implementation and your description of your findings in the paper writeup. You should have a “working” software implementation, meaning that the learning algorithm is implemented in software, it runs without crashing, performs learning iterations in Player/Stage, and is well-documented.

However, you may find that even if “correctly” implemented, your approach may still have difficulty in enabling the robot to learn. So, even if your learning program does not perform well, you will receive significant credit if you turn in a thorough paper that is readable and clearly outlines your experiments and their results, along with a discussion on why you think you obtained the results you did (or failed to obtain the results you were hoping for). That is, if your algorithm does not successfully train a robot to navigate, your work and results should clearly show that you spent considerable time trying to find the correct parameters, problem formulation, etc. (Note that you illustrate this scientifically in the form of research results – figures or graphs that show the results of using different parameters. You *do not* illustrate this by handwaving, making excuses, or saying you tried something for x hours. Instead, you show quantitative results of those experiments.)

Additionally, you must proofread your paper, ensuring no spelling or grammatical errors (such errors will reduce your grade). Figures and graphs should be clear and readable, with axes labeled and captions that describe what each figure/graph illustrates.

Graduate Grading

Graduate students will be graded more strictly on quality of the research and paper presentation. I expect a more thorough analysis of your results, and a working learning system (i.e., meaning the system actually learns). Your analysis should include a discussion (and perhaps results) on the following points (in addition to the points previously noted above):

- Rate of convergence of learning
- Effect of the size and layout of the learning environment, and relative starting positions of robot and goal position on the learning process
- Effect of different state and motor output representations (e.g., using more or less resolution) of the quality and speed of learning
- Effect of different reward assignments on the quality and speed of learning
- Future work that you believe would improve the learning
- Any other insightful observations you'd like to make

You should not address these points in a bullet-type fashion, but instead work the answers into your paper in a discussion-style format. The paper should have the “look and feel” of a technical conference paper, with logical flow, good grammar, sound arguments, illustrative figures, etc. Graduate students are expected to format their paper in standard IEEE conference format (see <http://www.ieee.org/portal/pages/pubs/transactions/stylesheets.html> for style files).

However, even with this additional information, your paper must not exceed 6 pages.

Turning in your project

You should email your project to the instructor AND the TA (parker@cs.utk.edu; m Bailey@cs.utk.edu) by the deadline.

Your submission should be in 2 parts:

1. Paper (in pdf format)
2. Tar or zip file of the programs and data files needed to run your learning algorithm, including a README file that gives instructions on how to run your code.