Project #2

Problem

Description

For this project, you will be writing an agent to successfully land the "Lunar Lander" that is implemented in OpenAI gym. You are free to use and extend any type of RL agent discussed in this class.

Lunar Lander Environment

The problem consists of a 8-dimensional continuous state space and a discrete action space. There are four discrete actions available: do nothing, fire the left orientation engine, fire the main engine, fire the right orientation engine. The landing pad is always at coordinates (0,0). Coordinates consist of the first two numbers in the state vector. The total reward for moving from the top of the screen to landing pad ranges from 100 - 140 points varying on lander placement on the pad. If lander moves away from landing pad it is penalized the amount of reward that would be gained by moving towards the pad. An episode finishes if the lander crashes or comes to rest, receiving additional -100 or +100 points respectively. Each leg ground contact is worth +10 points. Firing main engine incurs a -0.3 point penalty for each occurrence. Landing outside of the landing pad is possible. Fuel is infinite, so, an agent could learn to fly and then land on its first attempt. The problem is considered solved when achieving a score of 200 points or higher on average over 100 consecutive runs.

Agent Representation

As noted earlier, there are four actions available to your agent:

do nothing, fire the left orientation engine, fire the main engine, fire the right orientation engine

Additionally, at each time step, the state is provided to the agent as a 8-tuple:

(x, y, vx, vy, θ, vθ, left-leg, right-leg)

x and y are the x and y-coordinates of the lunar lander's position. vx and vy are the lunar lander's velocity components on the x and y axes. θ is the angle of the lunar lander. vθ is the angular velocity of the lander. Finally, left-leg and right-leg are binary values to indicate whether the left leg or right leg of the lunar lander is touching the ground. It should be noted, again, that you are working in a six dimensional continuous state space with the addition of two more discrete variables.

Procedure

This problem is more sophisticated than anything you have seen so far in this class. Make sure you reserve enough time to consider what an appropriate solution might involve and, of course, enough time to build it.

- Create an agent capable of solving the Lunar Lander problem found in OpenAI gym
  - Upload/maintain your code in your private repo at https://github.gatech.edu/gt-omscs-rldm
  - Use any RL agent discussed in the class as your inspiration and basis for your program
- Create graphs demonstrating
  - The reward for each training episode while training your agent
  - The reward per trial for 100 trials using your trained agent
  - The effect of hyper-parameters (alpha, lambda, epsilon) on your agent
    - You pick the ranges
    - Be prepared to explain why you chose them
  - Anything else you may think appropriate
- We've created a **private** Georgia Tech GitHub repository for your code. Push your code to the personal repository found here: https://github.gatech.edu/gt-omscs-rldm
  - The quality of the code is not graded. You don't have to spend countless hours adding comments, etc. But, it will be examined by the TAs.
  - Make sure to include a README.md file for your repository
    - Include thorough and detailed instructions on how to run your source code in the README.md
  - You will be penalized by **20 points** if you:

- - - ■ Do not have any code or do not submit your *full* code to the GitHub repository
      - ■ Do not include the git hash for your last commit in your paper
  - ● Write a paper describing your agent and the experiments you ran
    - ○ Include the hash for your last commit to the GitHub repository in the paper's header.
    - ○ The rubric includes a few points for formatting. Make sure your graphs are legible and you cite sources properly. While it is not required, we recommend you use a conference paper format. Just pick any one.
    - ○ 5 pages maximum -- really, you will lose points for longer papers.
    - ○ Explain your experiments
    - ○ Graph: Reward at each training episode while training your agent and discussion of results
    - ○ Graph: Reward per trial for 100 trials using you trained agent and discussion of the results
    - ○ Graph: Effect of hyperparameters and discussion of the results
    - ○ Explanation of pitfalls and problems you encountered
    - ○ Explanation of algorithms used, what worked best, what didn't work
    - ○ What didn't, what would you try if you had more time?
    - ○ Discuss your results
    - ○ Save this paper in PDF format
    - ○ Submit!

Resources

The concepts explored in this project are covered by:

- ● Lectures
  - ○ Generalization
- ● Readings
  - ○ Sutton Ch. 9 On-Policy Prediction with Approximation
  - ○ http://incompleteideas.net/book/the-book-2nd.html
- ● Source Code

  - ○ https://github.com/openai/gym

  - ○ https://github.com/openai/gym/blob/master/gym/envs/box2d/lunar_lander.py

- ● Documentation

- https://gym.openai.com/docs

- Examples (use for inspiration, you are still required to write **your own** code)

  - https://gym.openai.com/envs/LunarLander-v2

Tips

- If you worked on HW4, you most likely checked out an older version of OpenAI gym. Please remember to update to the latest version for this assignment.

- If you get a Box2D error when running gym.make('LunarLander-v2'), you will have to compile Box2D from source. Please follow these steps:

```
pip uninstall box2d-py
git clone https://github.com/pybox2d/pybox2d
cd pybox2d/
python setup.py clean
python setup.py build
sudo python setup.py install
```

Try running Lunar Lander again. Source: https://github.com/openai/gym/issues/100

- Even if you don't get perfect results or even build an agent that solves the problem you can still write a solid paper.

Submission Details

The due date is indicated on the Canvas page for this assignment.

Due Date: **Indicated as "Due" on Canvas**
Late Due Date [**20 point penalty**]: **Indicated as "Until" on Canvas**
Make sure you have set your timezone in Canvas to ensure the deadline is accurate.

The submission consists of:

- Your written report in PDF format (Make sure to include the git hash of your last commit)
- Your source code in your personal repository on Georgia Tech's private GitHub

To complete the assignment, submit your written report to Project 2 under your Assignments on Canvas: **https://gatech.instructure.com**

You may submit the assignment as many times as you wish up to the due date, but, we will only consider your last submission for grading purposes.

For a 20 point late penalty, you have 1 additional day to submit your assignment. That is, we will take late submissions for 1 full day after the first due date, as marked in Canvas. After the late submissions due date, we will not take any more submissions and you will receive a zero for that assignment.

Note: if you resubmit and your last submission is late, you will incur the 20 point late penalty. Late is late. It does not matter if you are 1 second, 1 minute, or 1 hour late. If Canvas marks your assignment as late, you will be penalized.

Grading and Regrading

When your assignments, projects, and exams are graded, you will receive feedback explaining your errors (and your successes!) in some level of detail. This feedback is for your benefit, both on this assignment and for future assignments. It is considered a part of your learning goals to internalize this feedback. This is one of many learning goals for this course, such as: understanding game theory, random variables, and noise.

If you are convinced that your grade is in error in light of the feedback, you may request a regrade within a week of the grade and feedback being returned to you. A regrade request is only valid if it includes an explanation of where the grader made an error. Send a private Piazza post to **only** Miguel Morales and Timothy Bail. In the Summary add "[Request] Regrade Project 2". In the Details add sufficient explanation as to why you think the grader made a mistake. Be concrete and specific. We will not consider requests that do not follow these directions.

It is important to note that because we consider your ability to internalize feedback a learning goal, we also assess it. This ability is considered 10% of each assignment. We default to assigning you full credit. If you request a regrade and do not receive at least 5 points as a result of the request, you will lose those 10 points.