RL and Autonomous Driving: Week 7

By Brandon Dominique, John Nguyen, Hoang Huynh, and Eric Av

Problem Statement

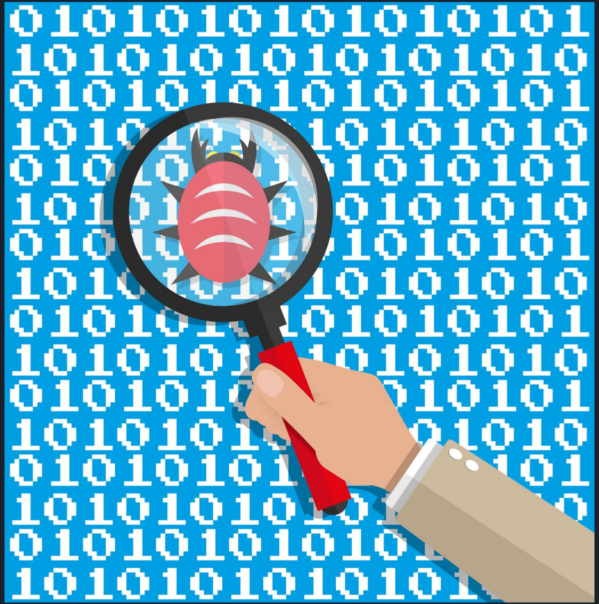
- Collision Prevention with CAT Vehicle in Gazebo
  - Reinforcement learning to train CAT Vehicle in Gazebo to detect and avoid potential collisions
  - Train the vehicle on a variety of situations such as collision detection and pedestrian avoidance
  - Use a meta-cognitive radio to relay information to nearby vehicles

Lit Review: Off-Road Obstacle Avoidance through End-to-End Learning

- End-to-End Learning: Type of deep reinforcement learning where model trains all parameters at once, from simulator to test.
  - Commonly used in autonomous driving.
- Trained a small model car to avoid obstacles by choosing turning angle.
- Uses 6 layer CNN to train on pictures of terrain/environment the car navigates.

Optimizing and Cleaning RL Framework

- Wrote a new reward function based on Euclidean distance to desired destination.
- Fixed errors in launch files which caused the car to start/stop spontaneously.
    - Car was given initial velocity 0, so after publishing to cmd_vel, the car would immediately go back to 0 velocity.
- Set up DistanceEstimator sensor.
    - We can control the range of DistanceEstimator.

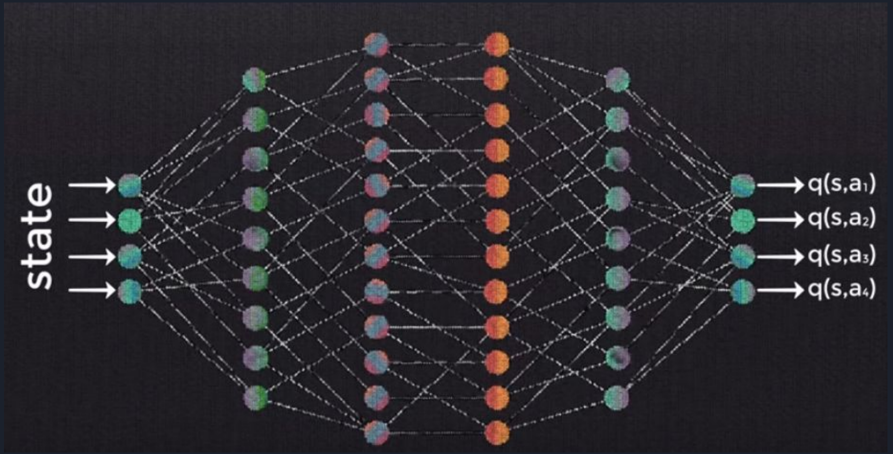


Resetting vs Restarting Gazebo

- Gazebo is poorly optimized for resetting.
    - When the car crashes or is done with an episode, we want the car to train again, with the new data it has gathered.
    - Due to Gazebo being poorly optimized, this leads to the simulation slowing down a lot, leading to errors.
    - The learning script runs in real time, but the real time factor of the simulation could be <0.9.
- Will need to write a script that will close Gazebo and restart it for the next episode.

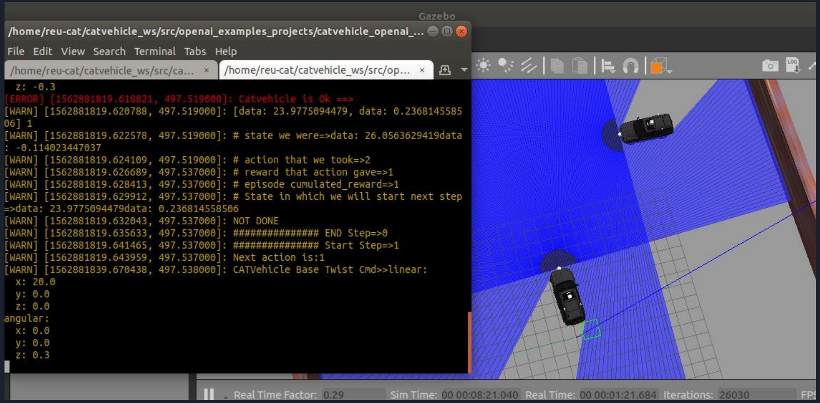
Algorithm Improvement

- From a basic Q learning to Deep Q learning
- Apply experience replay technique to the learning process



Gazebo World for Obstacle Collision

- Implemented a second CAT car to act as a moving obstacle
- Implemented this world with new learning algorithm
- Is much too slow to have any tangible progress


Gazebo
/home/reu-cat/catvehicle_ws/src/openai_examples_projects/catvehicle_openai_...
File
Edit
View
Search
Terminal
Tabs
Help
/home/reu-cat/catvehicle_ws/src/ca...
/home/reu-cat/catvehicle_ws/src/op...

```
  z: -0.3
[ERROR] [1562881819.618821, 497.519000]: Catvehicle is Ok ==>
[WARN] [1562881819.620788, 497.519000]: [data: 23.9775094479, data: 0.2368145585
06] 1
[WARN] [1562881819.622578, 497.519000]: # state we were=>data: 26.0563629419data
: -0.114023447037
[WARN] [1562881819.624109, 497.519000]: # action that we took=>2
[WARN] [1562881819.626689, 497.537000]: # reward that action gave=>1
[WARN] [1562881819.628413, 497.537000]: # episode cumulated_reward=>1
[WARN] [1562881819.629912, 497.537000]: # State in which we will start next step
=>data: 23.9775094479data: 0.236814558506
[WARN] [1562881819.632043, 497.537000]: NOT DONE
[WARN] [1562881819.635633, 497.537000]: ############### END Step=>0
[WARN] [1562881819.641465, 497.537000]: ############### Start Step=>1
[WARN] [1562881819.643959, 497.537000]: Next action is:1
[WARN] [1562881839.670438, 497.538000]: CATVehicle Base Twist Cmd>>linear:
  x: 20.0
  y: 0.0
  z: 0.0
angular:
  x: 0.0
  y: 0.0
  z: 0.3
```

Real Time Factor: 0.29 Sim Time: 00 00:08:21.040 Real Time: 00 00:01:21.684 Iterations: 26030 FPS

Lit Review: Simulation of a Cognitive Radio Environment

- Simulation of Cognitive Radio Networks using MATLAB
  - Simulation with multiple PUs and SUs competing for space on 6 different channels
  - Unclear Implementation and Simulation Setup
- Investigating the Challenges of Dynamic Spectrum Access in Cognitive Radio Enabled Vehicular Ad-Hoc Networks (CR-VANETS)
  - R for basic simulation of CR-Enabled Secondary Users
  - NS-2 with other software to obtain numerical results/performance of each individual Secondary User

Progress and Questions Going Forward

- 3 Modulations Schemes and 2 Channel Schemes done
- NS3 Provides code to run a CR Simulation
    - What parts of the code would I have to modify to make it compatible with the Python Code I’ve already Written?
- If NS3 isn’t a viable option, what simulation setup will be?
- How can my part of the project be easily implemented into a ROS Environment?

Future Plans

- Implement an actor world instead of a second CAT Vehicle for obstacle detection
- Set up script to restart entire Gazebo world, instead of resetting the world.
- Modulation and Channel Schemes are done; begin to build ns3 environment to measure numerical performance