CS221 Project Proposal
Fast and Furious: Reinforcement Learning for Race Cars

Mindy Huang (mindyh)

Problem Statement

The goal of this project is to have a race car learn the most efficient way to drive around a track autonomously. We will do this in simulation, which is discussed in further detail below.

Our car robot will take in data on its current position on the racetrack as well as the curvature of the track at that point - it has no other knowledge of the track outside of what it can see, just like a real race car. Its state will contain its current speed, gear, and steering wheel position. Its actions will be to accelerate, decelerate, and turn.

Our metric for success will be speed - the faster our bot completes a track the better. Specifically, we will time the bot for 3 laps and average the lap times as a metric for comparison.

Simulator

We will leverage The Open Racing Car Simulator (TORCS), an open source race car simulator used both for gaming and for AI research. TORCS has support for many different kinds of race tracks and differentiates the physics for driving on dirt and grass versus pavement. It also simulates physics for different car designs, spoilers, fuel consumption, damage, etc.

We use the python version of the Simulated Car Racing Championship (SCRC) client to TORCS - it provides a simple interface for control that only exposes sensor data to the bot at discrete intervals, just as an autonomous car would experience in reality. SCR also provides a platform that allows us to easily repeat races for learning purposes.

Baseline and Oracle

We choose our baseline to be the time of the very basic robot given to us by SCRC. All this robot does is keep to the middle of the racetrack at 100kph - slow enough that we are guaranteed to never have any slippage. Running it for 3 laps on track C-Speedway and

averaging the times gives us a baseline of 120.62 seconds per lap.

We choose our oracle to be the fastest robot given by the default TORCS package. Re-running the experiment, we get an oracle of 40.59 seconds per lap, 3 times faster than our baseline.

Techniques

We will make use of imitation and reinforcement learning. First, we will get the path of our oracle racing bot in the track. Then we will train our robot to follow this path - we will reward actions that takes us forward and keeps us en-route and penalize actions that take us backwards or deviates from it, with very high rewards for low lap times and extremely negative rewards for crashing or damage.

To mitigate the challenges with reinforcement learning and just produce any sort of result, we will start by constraining most variables and just focus on changing acceleration/deceleration and the steering wheel. We will let the gear change automatically and turn off simulations for noisy sensors, low fuel and cumulative damage to the car from driving off-road.

Related Reading

A quick Google search reveals that this project of teaching a robotic car how to drive via reinforcement learning has been researched a few times before.

1. This paper focuses on reinforcement learning of a car robot based only on the curvature of the road. After running it in simulation they try it out on a real car, and it works surprisingly well.

2. This paper uses reinforced neural networks. It was done with RARS simulation software, the precursor to TORCS.

3. This paper is a recent master's thesis, and provides a good summary of all the research on the topic thus far and explores in pain-staking detail the results of rewards based on lap time and/or damage.