**NICELY FORMATTED**
**https://drive.google.com/file/d/1b7hU241SRka0obds-xCyMEnvdPBMYwD1/view?usp=sharing**

# Exploring an Elementary Autonomous Car System

Raveesh Avashia, Nicolas Berio LeBeau, Mirai Sahara

## Abstract

The problem addressed is a simple one. Car transmissions have a significant impact on the performance and efficiency of the car. While humans have been able to reach great levels of proficiency in driving, a computer has the potential to reach near perfect levels. Various sensors and technologies are able to collect far more data than our simple senses, allowing a computer to gather and respond to a vast amount of data. The usage of artificial intelligence algorithms in a car that receives data about the track in front of it was explored by our group. In our case the car receives data about the road ahead such as elevation and curvature, along with data about the car's performance; this means engine RPM, car speed, acceleration, etc.

## 1. Motivation

This is an interesting problem because autonomous cars are the future of urban transportation. It's an increasingly large field that shows no signs of stopping with giants like Google entering the field in the past decade. Autonomous cars are a difficult problem because the mechanics and intelligence behind a autonomous car are immense and complex. In fact, it is so complex that we are only focusing on a very small portion of the total problem. And most importantly, for us cars are just a really interesting topic.

## 2. Overview

We decided to use reinforcement learning to teach an agent to run a track over and over, refining its method and decisions at 10 meter intervals every time.

## 3. Related Work

It is difficult to find work that is directly relevant as we are exploring what is essentially a small subproblem of a much richer, larger problem that is far beyond our scope and abilities. There are many significant and interesting projects out there but there were a few that caught our eye. One is Deepdrive (deepdrive.io), an Unreal Engine simulation of a autonomous car in a real life environment. A car is simulated along with a multitude of sensors, all in a full environment. This is basically a framework to test a multitude of different cars and agents for the car without having to test in the real world. In terms of the real world, there are many companies tackling autonomous cars with several running real tests in the streets. One example of this is drive.ai, a startup out of Silicon valley that has autonomous cars running on the streets of San Francisco already. Their approach is very different from ours; their cars use purely deep learning methods to learn how to drive. With how complex it is to process all the information coming in, this is probably the best option in terms of power and ease of learning.

## 4. Approach

Our approach was reinforcement learning by having the car "drive" through a track over and over. The first step was modeling a car, which was done by copying the various attributes of one specific car such as weight or engine torque. The car is able to gear up or down at every interval and receives a reward for each decision. This is what builds up the reinforcement learning model and creates a policy that the car can follow for a given track.

We broke up the development into steps, first focussing on developing a realistic driving environment. The environment takes 2 lists as input, the layout of the track, and a list of instructions. A car object is created and based on the instructions given it simulates a lap, and reports back with the performance. The report includes the time, a step by step record, and notifies if the car spun out or stalled during the run. There are a lot of physics at play, which can be found in the class 'WRX' in utils.py.

Next we developed the reward function, which is meant to reward speed, horsepower, and optimal torque, to motivate the agent to keep the car at it best performance. It penalizes spins or stalls heavily.

Our agent is designed to take the previous lap record, and find a place where it can improve on it. This design isn't memory heavy, and incorporates our 'sight' concept. It determines the value of each action based on the visible track in front of it, aiming for the maximum speed it can maintain without slipping.

## 5. Evaluation

Evaluation was approached by comparing the results produced by our agent to the the actual test results of the car on various tracks. The agent is rewarded on the basis of speed, gear shifts, horsepower and torque. We used six different tracks for our evaluation where the car had already been tested by actual drivers. The results produced were as follows:

i) Track 1: This is a 1000m test track used by the manufacturers to test the car, our agent clocked 25.79s compared to 25.1s, clocked by a professional driver.

ii) Track 2: This is a S-shaped track (test for car against curves), our agent clocked 51.06s compared to 47.9s, clocked by Nico.

iii) Track 3: This is the Daytona circuit (An official NASCAR circuit) where our agent clocked 68.6s compared to 64.2s, clocked by a professional driver.

iv) Track 4: This track tests the agent on a uphill drive, our agent clocked 27.38s compared to 24.2s, clocked by a Nico.

v) Track 5: This track was used by Nico to test the car, where our clocked clocked 33.46s compared Nico's 45.57s.

vi) Track 6: This is a quarter mile test track used by the manufacturers to test the car, our simulation produces a time of 14.65s compared to 13.3s, produced by a professional driver.

There is approximately 92% accuracy in the results produced by our agent against professional drivers, because there might be small differences in the Physics used by us to setup the environment and the actual car Physics.

## 6. Acknowledgement

Huge thanks to Nicolas who's car knowledge single handedly put the simulation together and carried the team.

http://www.asawicki.info/Mirror/Car%20Physics%20for%20Games/Car%20Physics%20for%20Games.html

http://rototest-research.eu/popup/performancegraphs.php?ChartsID=838