Project – Self Driving Car

Abstract

The project Self Driving Car using machine learning and deep learning as the base technologies for designing and implementation . The idea behind this project is to satisfy the demands of the society i.e Automation , the car industry has already accomplished this goal of automations of cars and are continuously working on , to increase the use and efficiency of the model of automation . To implement the whole system physically ,the car is connected to number of sensors and processors, object detection model , uv ray sensors etc . The car with these and few more embedded essential components is capable of taking decision by itself with excellent accuracy .

Objective

The main aim behind the Self Driving car project is to reduce the human efforts , significantly fewer traffic accidents. More than 90% of all accidents are caused by some degree of human error, including distraction, impaired driving, or poor decision making. The project is the working model with were basis decision making capacity of the car i.e the real time data set of the car using the virtual environment called Unity , the real-time 3D rendering platform, is being used by engineering teams to efficiently create simulation environments for autonomous vehicle training. The Self Driving Car is helpful in reduction in car crash-related costs, reduced strain on the healthcare system, more efficient transportation, better fuel savings, and more can all contribute to the overall societal cost-savings.

Introduction

Self-Driving Car can analyse surrounding without any human interactions and can make decision of its own . The self Driving Car are generally , combination of sensors which are used to identify the pathway and road signal of the surroundings. It is has reduced costs due to less wastage of fuel, increased safety, increased mobility, increased customer satisfaction and that’s why it has more advantage than traditional cars. The biggest benefit of using a self-driving car is significantly fewer traffic accidents. More than 90% of all accidents are caused by some degree of human error, including distraction, impaired driving, or poor decision making. With self-driving cars making decisions and communicating with one another, the number of accidents should reduce.

This project is basically the protype of the Self Driving Car . i.e it is the virtual environment where the car is running itself real-time , performing all the actions accordingly , the data is being produced at the same time the car is about make the next move , So there no historical data ,its all real time .

The Udacity simulator , can manually drive a car to generate training data, or your machine learning model can autonomously drive for testing.

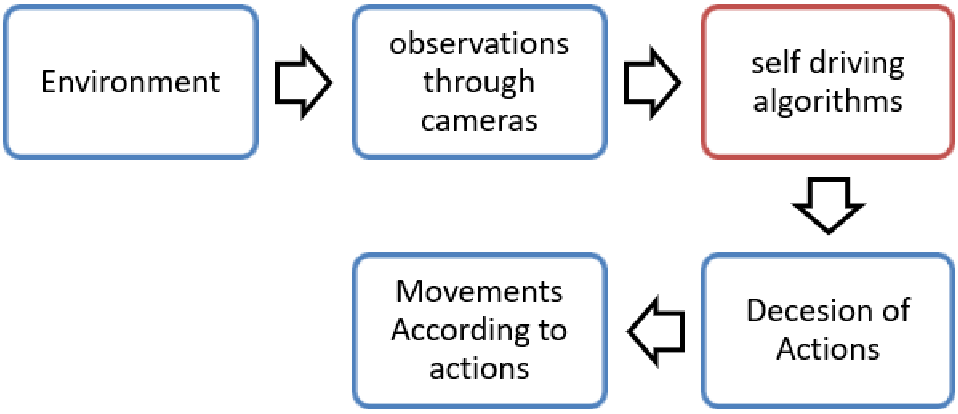


Figure 1 basic model for self-driving car

METHODOLOGY

System Architecture

Udacity Behavior cloning:

The Behavior Cloning Use the Udacity Simulator to collect data of good driving behavior. Build, a convolution neural network in Keras that predicts steering angles from images. Train and validate the model with a training and validation set. Test that the model successfully drives around track one without leaving the road.

- Use the Udacity Simulator to collect data of good driving behavior
- Build, a convolution neural network in Keras that predicts steering angles from images
- Train and validate the model with a training and validation set
- Test that the model successfully drives around track one without leaving the road

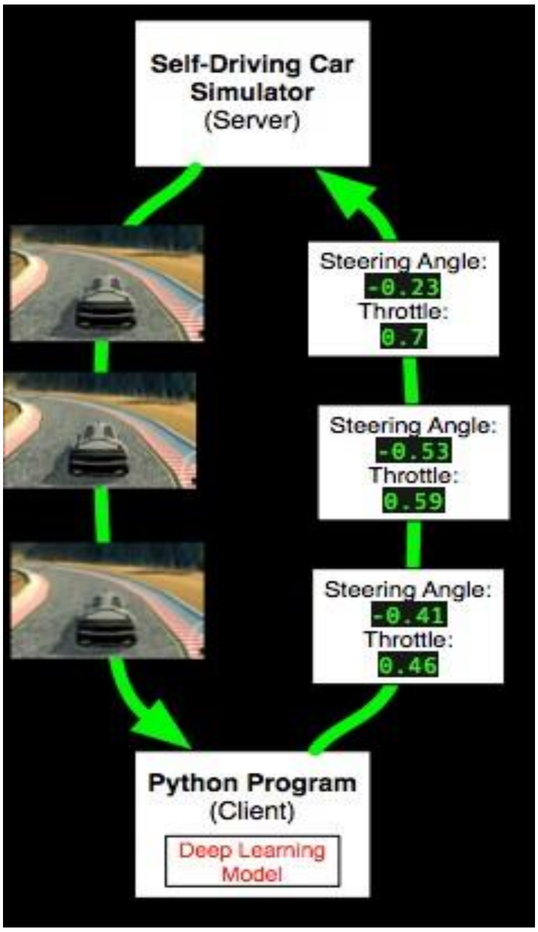
The file included in this cloning are :

- model.py The script used to create and train the model.
- drive.py The script to drive the car. You can feel free to resubmit the original drive.py or make modifications and submit your modified version.
- utils.py The script to provide useful functionalities (i.e. image preprocessing and augumentation)
- model.h5 The model weights.
- environments.yml conda environment (Use TensorFlow without GPU)
- environments-gpu.yml conda environment (Use TensorFlow with GPU)

- use a neural network to clone car driving behavior. It is a supervised regression problem between the car steering angles and the road images in front of a car.
- Those images were taken from three different camera angles (from the center, the left and the right of the car).
- The network is based on The NVIDIA model, which has been proven to work in this problem domain.
- As image processing is involved, the model is using convolutional layers for automated feature engineering.

the link of these file is : https://github.com/naokishibuya/car-behavioral-cloning

Working :



Model Architecture Design

The design of the network is based on the NVIDIA model, which has been used by NVIDIA for the end-to-end self driving test. As such, it is well suited for the project.

It is a deep convolution network which works well with supervised image classification / regression problems. As the NVIDIA model is well documented, I was able to focus how to adjust the training images to produce the best result with some adjustments to the model to avoid overfitting and adding non-linearity to improve the prediction.

Adding the following adjustments to the model.

- used Lambda layer to normalized input images to avoid saturation and make gradients work better.
- added an additional dropout layer to avoid overfitting after the convolution layers.
- also included ELU for activation function for every layer except for the output layer to introduce non-linearity.

In the end, the model looks like as follows:

- Image normalization
- Convolution: 5x5, filter: 24, strides: 2x2, activation: ELU
- Convolution: 5x5, filter: 36, strides: 2x2, activation: ELU
- Convolution: 5x5, filter: 48, strides: 2x2, activation: ELU
- Convolution: 3x3, filter: 64, strides: 1x1, activation: ELU
- Convolution: 3x3, filter: 64, strides: 1x1, activation: ELU
- Drop out (0.5)
- Fully connected: neurons: 100, activation: ELU
- Fully connected: neurons: 50, activation: ELU
- Fully connected: neurons: 10, activation: ELU
- Fully connected: neurons: 1 (output)

As per the NVIDIA model, the convolution layers are meant to handle feature engineering and the fully connected layer for predicting the steering angle. However, as stated in the NVIDIA document, it is not clear where to draw such a clear distinction. Overall, the model is very functional to clone the given steering behavior .

CODE

The practical working of the system using the simulator of Udacity called simulator Self Driving Car Nanodegree Program .

The environment interface is :

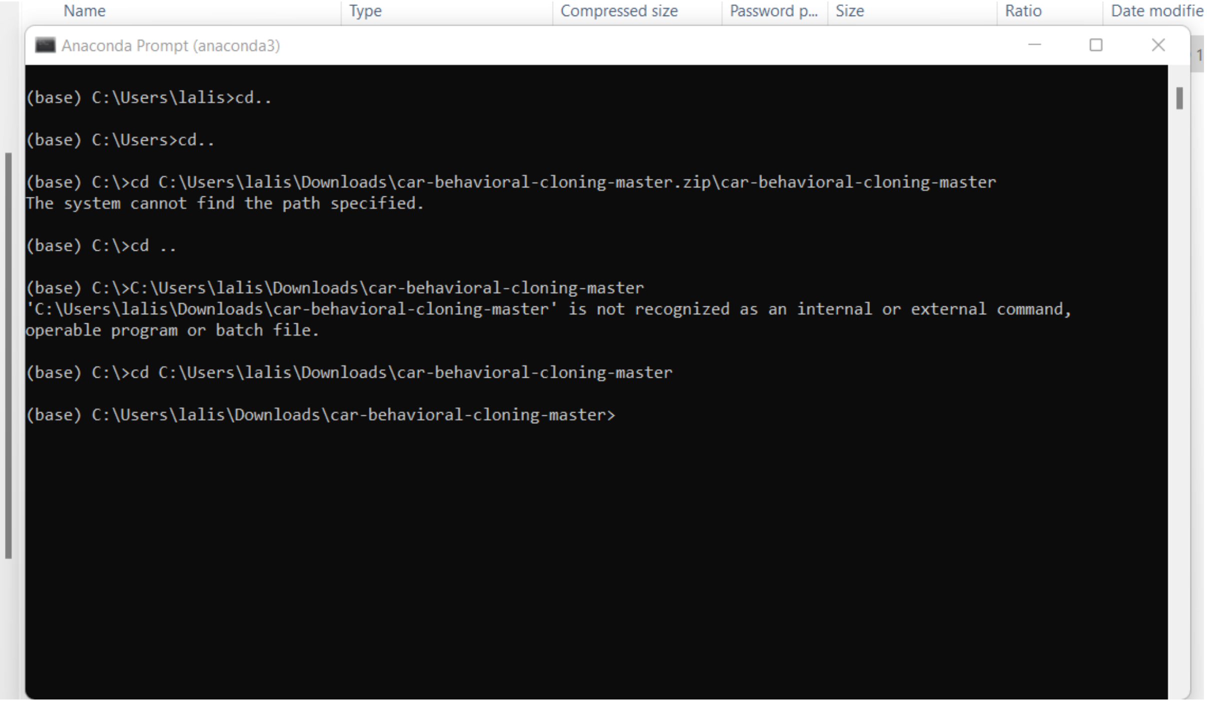
Download the file : https://github.com/naokishibuya/car-behavioral-cloning

Step 1 :

The anaconda prompt is used for the implementation .

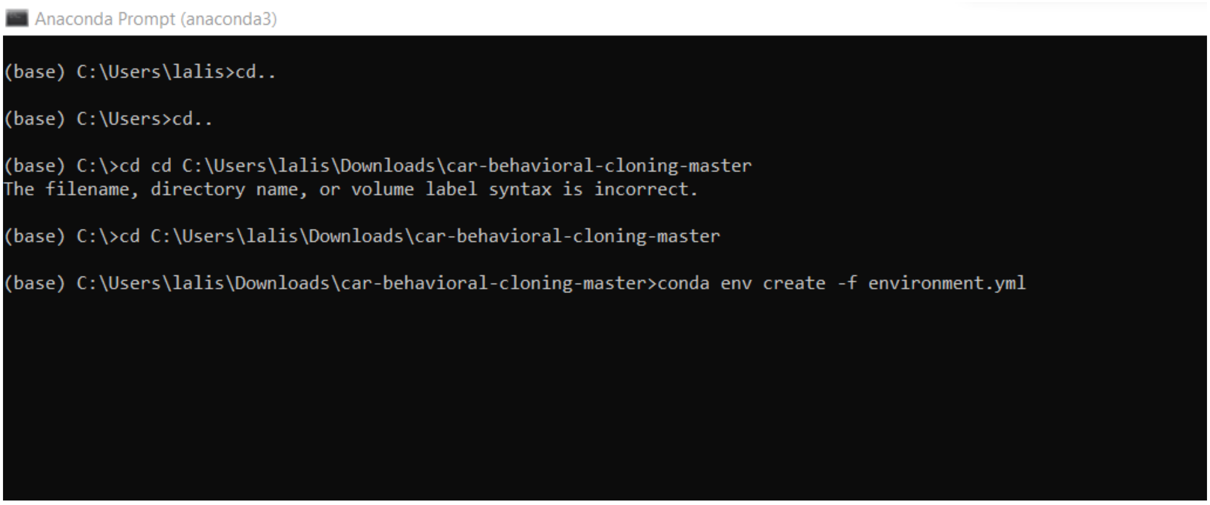
Loading the folder in form of zip file saved in the c drive

C:\Users\lalis\Downloads\car-behavioral-cloning-master.zip



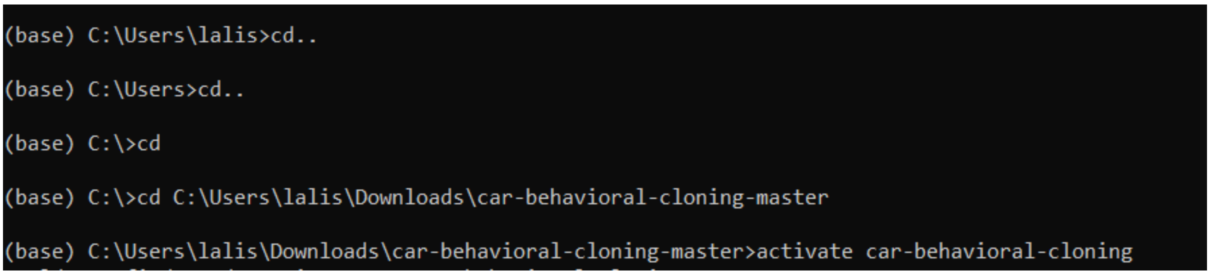
Step 2:

Path for the environment .yml

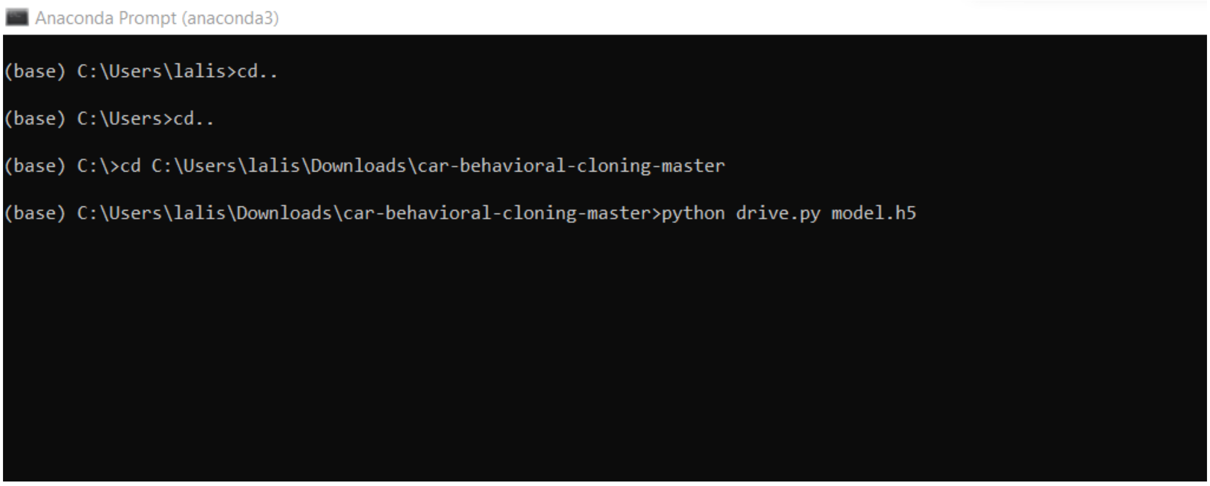


Step 3 :

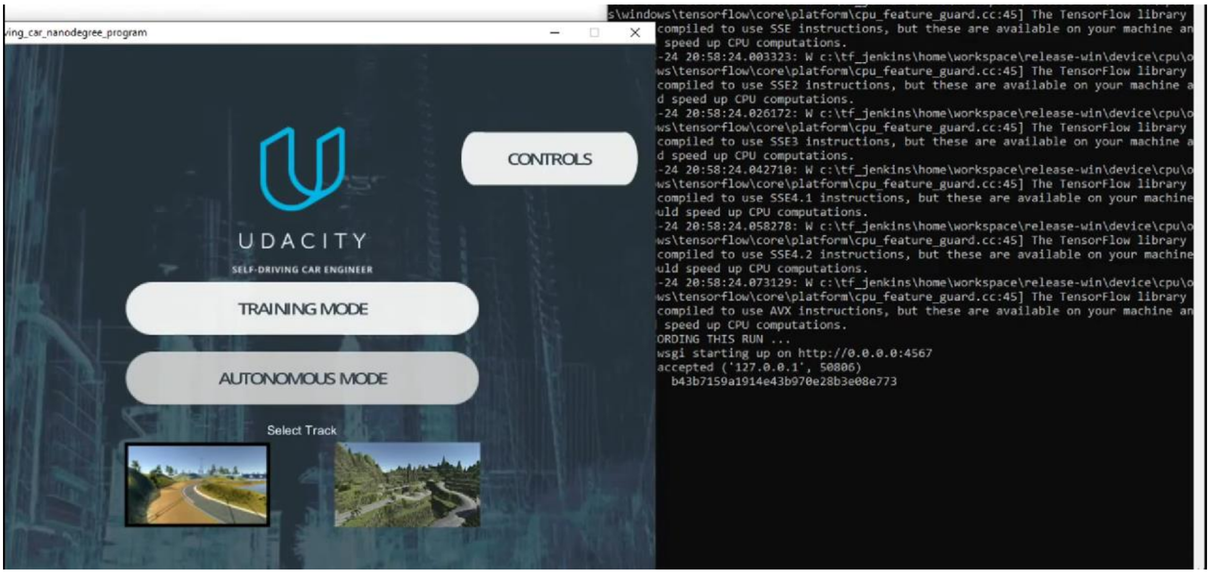
activate car-behavioral-cloning



python drive.py model.h5



Running :

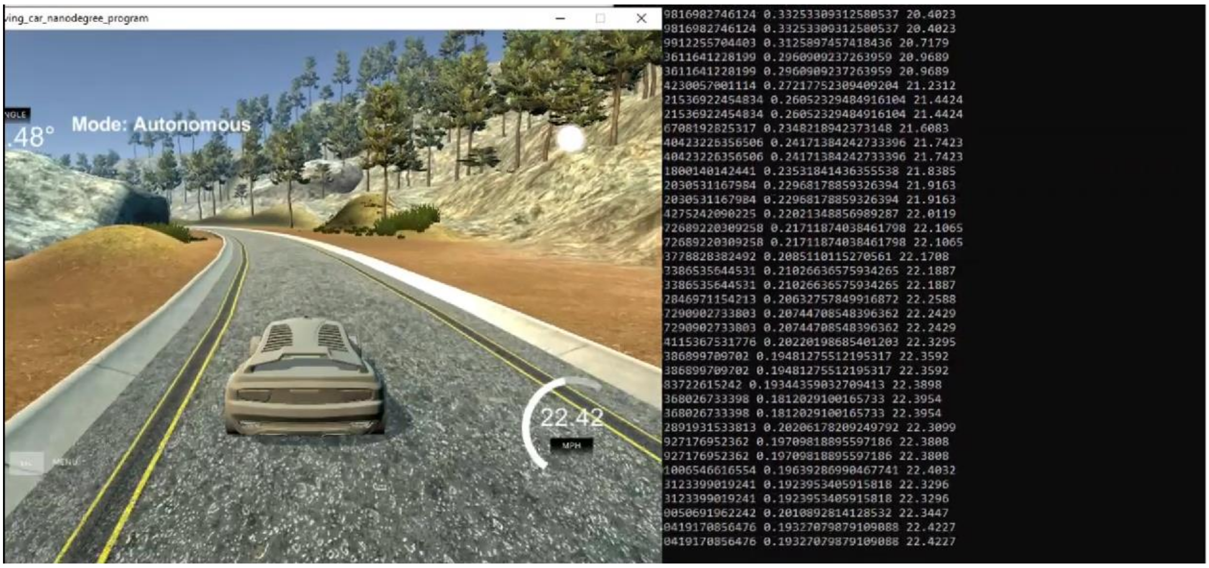


Running the car using the file

python drive.py model.h5 using tensorflow

selectin of environment in Udacity :

Run :



CONCLUSION

The project us just a prototype ,but in future this can be a very important and efficient innovation by humans for human , the combination of all modes of engineering from automobile to computer science ,from electronic and IOT ,sensors etc , every sectors are involved in development . AL and machine learning As technology expands throughout the world, self-driving cars will become the future mode of transportation universally. The legal, ethical, and social implications of self-driving cars surround the ideas of liability, responsibility, and efficiency.

Types of automated technologies, such as advanced driver assistance system technologies already in use on the roads and future automated driving systems at their mature state, have the potential to reduce crashes, prevent injuries, and save lives.

Also Udacity's self-driving car nanodegree is worth it for new machine learning and artificial intelligence engineers who want to specialize in the development of self-driving cars. Titled Self-Driving Car Engineer, this nanodegree was the first one I took after learning the core concepts of machine learning.