

A futuristic wireframe car, rendered in a glowing cyan color, is positioned on a dark blue grid floor. The car is shown from a side profile, facing left. A glowing cyan trail follows the car's path, curving around it. In the background, a large, semi-transparent sphere is visible. The overall scene has a high-tech, digital aesthetic.

**STEERING WHEEL
ANGLE
PREDICTION**

IN SELF DRIVING CARS

INTRODUCTION

- > Based on the latest statistics, the reason for major fatal road accident were due to over speeding, driver fatigue and other distractions etc.
- > These problems can be overcome with the help of autonomous cars.
- > As the cars become more automated, road fatalities can be controlled.
- > With the application of Deep Learning(CNN), cars can drive autonomously without the human intervention.



PROJECT OBJECTIVE

-> While building an autonomous cars, these are some of the factors to be considered,

- > Steering Wheel Angle
- > Acceleration system
- > Braking system

-> Our primary objective is about predicting the steering wheel angle of the car.



SOFTWARE AND HARDWARE REQUIREMENTS

SOFTWARE REQUIREMENTS

OS : GNU/Linux, Windows 10,8.1

Programming Language : Python3

Framework : Pytorch/ TensorFlow

Training Platform : Google Colabs/ Kaggle

HARDWARE REQUIREMENTS

GPU : 12GB NVIDIA Tesla K80 GPU

Processor : Intel Core i5-7200U (7th Gen)

Ram : 8GB

Memory : 1TB HDD



WORKFLOW

DATA GENERATION (BY
SIMULATOR)



TRAINING THE CNN



TESTING



PREDICTED ANGLE

STEP 1: DATA GENERATION USING A SIMULATOR



Dataset generated by manual driving.

Car driven without human intervention

Medium Level

Hard Level

TWO VARIANTS OF DATA GENERATION

MEDIUM



HARD



GENERATED DATASET

[illegible]

Path of Center camera image.

Path of left camera image.

Path of right camera image.

GENERATED DATASET CONTD.

	D	E	F	G	H	I	J
76	0	1	0	30.19031			
77	0	1	0	30.18985			
78	0	1	0	30.19006			
79	0.2	1	0	30.18193			
80	0.4	1	0	30.15741			
81	0.6000001	1	0	30.13366			
82	0	1	0	30.18476			
83	0	1	0	30.18926			
84	0	1	0	30.19019			
85	0	1	0	30.19036			
86	0	1	0	30.18876			

Reduced
Steering
angle

Throttle

Brake

Speed

PRE-PROCESSING: AUGMENTATION

i) CROPPING

sheared image



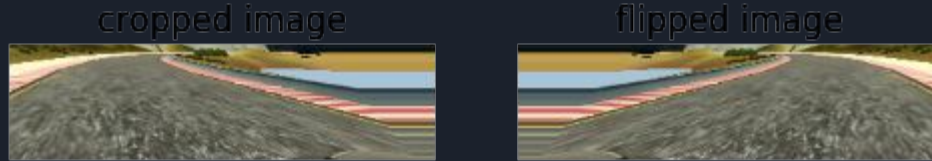
cropped image



- The images captured from the simulator include unnecessary information (like sky).
- So we are cropping out unwanted information.

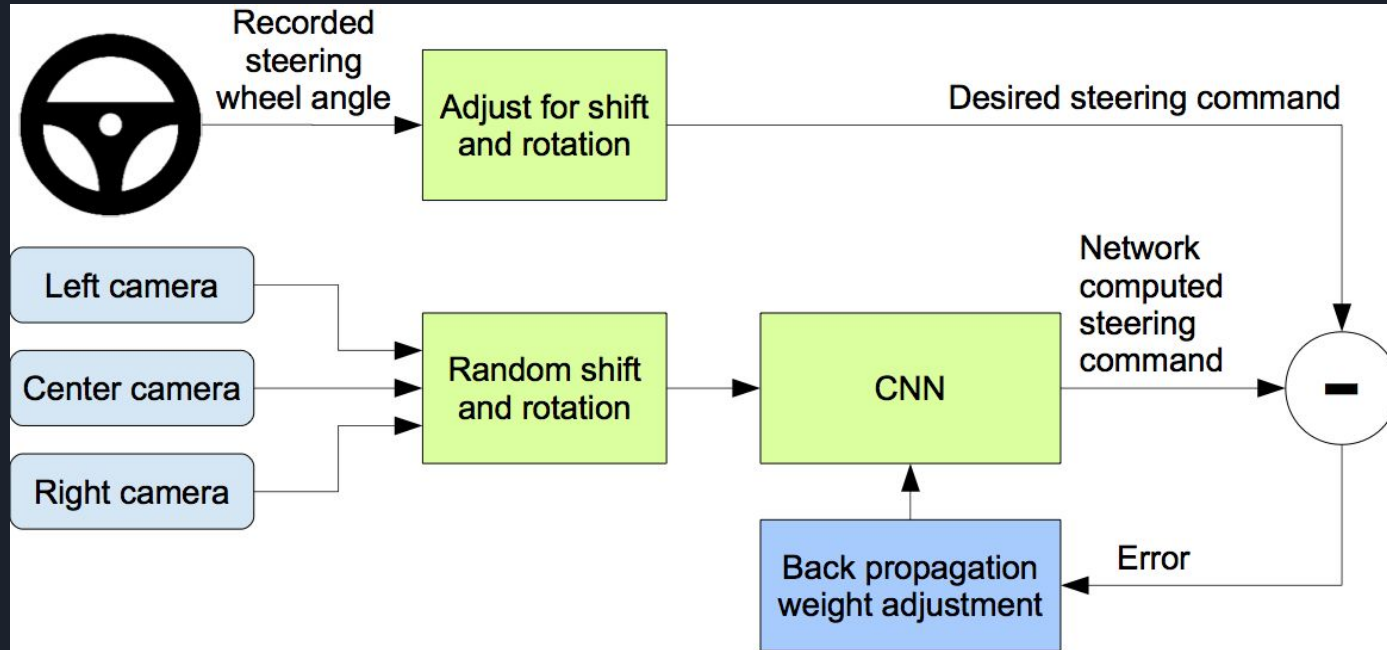
PRE-PROCESSING: AUGMENTATION Contd.

ii) FLIPPING



- Our dataset contains more images of right turns.
- To get better accuracy we are flipping the images, so that it appears like left turns.

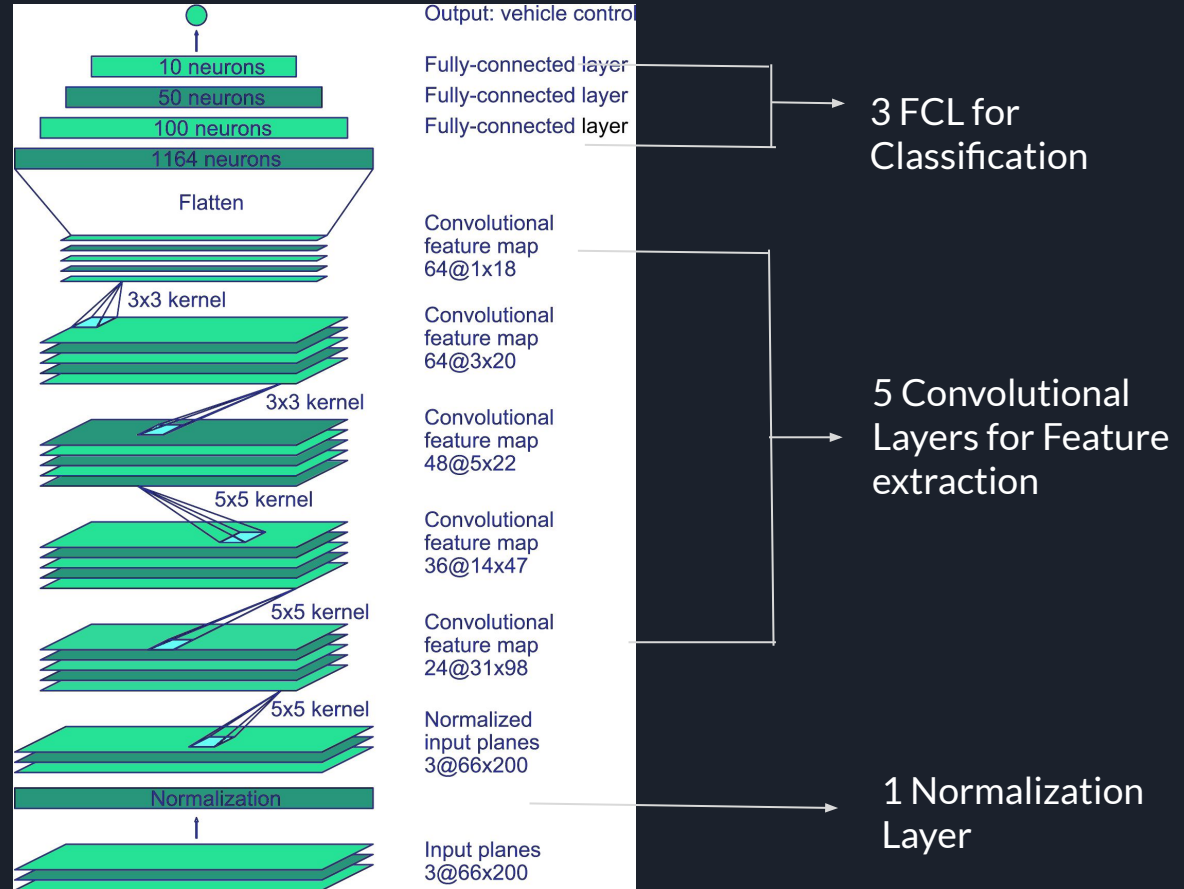
STEP 2 : TRAINING



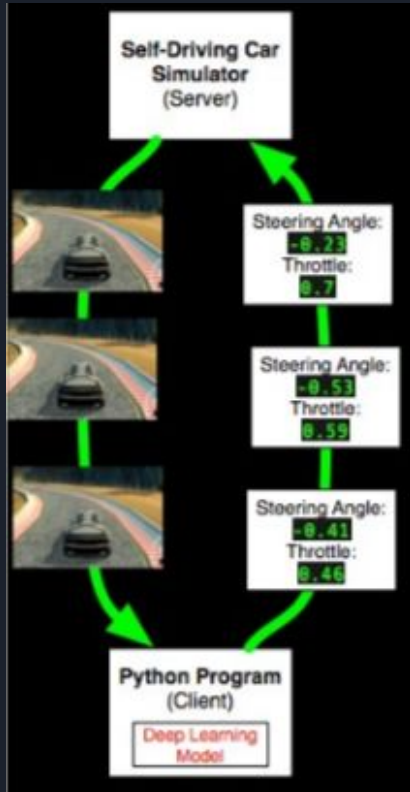
Training : High level overview of the project

CNN

9 layered Convolutional Neural Network



STEP 3 : TESTING



We can think as **Client - server** model:

Our *trained model acts as a client*. It pops out steering angles continuously.

The *simulator acts as server*. When it gets the steering angle it gives out corresponding image.



LOSS AND ACCURACY

```
torch.Size([32, 64, 2, 33])
torch.Size([32, 64, 2, 33])
torch.Size([32, 64, 2, 33])
torch.Size([32, 64, 2, 33])
torch.Size([32, 64, 2, 33])
torch.Size([32, 64, 2, 33])
torch.Size([32, 64, 2, 33])
torch.Size([32, 64, 2, 33])
torch.Size([32, 64, 2, 33])
torch.Size([32, 64, 2, 33])
torch.Size([32, 64, 2, 33])
torch.Size([32, 64, 2, 33])
torch.Size([32, 64, 2, 33])
torch.Size([32, 64, 2, 33])
torch.Size([32, 64, 2, 33])
torch.Size([32, 64, 2, 33])
torch.Size([32, 64, 2, 33])
torch.Size([32, 64, 2, 33])
torch.Size([32, 64, 2, 33])
torch.Size([32, 64, 2, 33])
torch.Size([28, 64, 2, 33])
torch.Size([28, 64, 2, 33])
torch.Size([28, 64, 2, 33])
Loss: 0.069
```

Our model was able to produce
the loss of 0.069

Not sure how to consider .

Accuracy _____



TEAM A11

**KIRAN U KAMATH
DHANUSH B RAJ
MEGHANA GS
PRAJWAL T**



THANK YOU