A Micro Project Report on

INTELLIGENT VEHICLES USING ARTIFICIAL INTELLIGENCE

Submitted to the CMR Institute of Technology in partial fulfillment of the requirement for the award of the Laboratory of

Artificial Intelligence Lab (20-CS-PC-317)

of

III-B.Tech. I-Semester

in

Computer Science and Engineering Department

Submitted by

ALLAGADDA DILEEP	(20R01A0561)
B.SWATHI	(20R01A0563)
BALE VIGNESH	(20R01A0565)
BANSWADA KARTHIK REDDY	(20R01A0567)
BOUTH SWARNALATHA	(20R01A0569)
BIJJALA SUPRAJA	(21R05A0507)

Under the Guidance Of

Mrs. A. Mahalakshmi

Associate Professor, Department of Computer Science and Engineering

CMR INSTITUTE OF TECHNOLOGY

(UGC AUTONOMOUS)

(Approved by AICTE, Affiliated to JNTU, Kukatpally, Hyderabad)

Kandlakoya, Medchal Road, Hyderabad

2022-2023

CMR INSTITUTE OF TECHNOLOGY

(UGC AUTONOMOUS)

(Approved by AICTE, Affiliated to JNTU, Kukatpally, Hyderabad)

Kandlakoya, Medchal Road, Hyderabad.

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that a Micro Project entitled with: " **INTELLIGENT VEHICLES USING ARTIFICIAL INTELLIGENCE** " is being Submitted By

ALLAGADDA DILEEP	(20R01A0561)
B.SWATHI	(20R01A0563)
BALE VIGNESH	(20R01A0565)
BANSWADA KARTHIK REDDY	(20R01A0567)
BOUTH SWARNALATHA	(20R01A0569)
BIJJALA SUPRAJA	(21R05A0507)

In partial fulfillment of the requirement for award of the Artificial Intelligence Lab (20-CS-PC-317) of III-B.Tech I- Semester in Department of Computer Science and Engineering (CSE) towards a record of a bonafide work carried out under our Guidance and Supervision.

Signature of Faculty
Mrs.A.Mahalakshmi
(Assistant Professor)

Signature of HOD
Mr.A.Prakash

Course Coordinator

Dr.A.Nrimal Kumar
(Associate Professor)

ACKNOWLEDGEMENT

We are extremely grateful to **Dr.M.JangaReddy**, **Director**, **Dr. B. Satyanarayana**, **Principal** and **Mr. A. Prakash**, **Head of Department**, Department of Computer Science and Engineering, CMR Institute of Technology for their inspiration and valuable guidance during entire duration.

We are extremely thankful to our Artificial Intelligence Lab faculty in-charge, **Mrs.A.Mahalakshmi**, Assistant Professor, Department of Computer Science and Engineering, CMR Institute of Technology for her constant guidance, encouragement and moral support throughout the project.

We express our thanks to all staff members and friends for all the help and coordinationextended in bringing out this Project successfully in time.

Finally, we are very much thankful to our faculty , parents and our team members who guided directly orindirectly for successful completion of the project.

ALLAGADDA DILEEP	(20R01A0561)
B.SWATHI	(20R01A0563)
BALE VIGNESH	(20R01A0565)
BANSWADA KARTHIK REDDY	(20R01A0567)
BOUTH SWARNALATHA	(20R01A0569)
BIJJALA SUPRAJA	(21R05A0507)

CONTENTS

Abstract

Abstract: In the modern era, the vehicles are focused to be automated to give human driver relaxed driving. In the field of automobile various aspects have been considered which makes a vehicle automated. Google, the biggest network has started working on the self-driving cars since 2010 and still developing new changes to give a whole new level to the automated vehicles. In this paper we have focused on two applications of an automated car, one in which two vehicles have same destination and one knows the route, where other don't. The following vehicle will follow the target (i.e., Front) vehicle automatically. The other application is automated driving during the heavy traffic jam, hence relaxing driver from continuously pushing brake, accelerator, or clutch. The idea described in this paper has been taken from the self-driving car, defining the one aspect here under consideration is making the destination dynamic. This can be done by a vehicle automatically following the destination of another vehicle. Since taking intelligent decisions in the traffic is also an issue for the automated vehicle so this aspect has been also, under consideration in this paper.

We all know self-driving cars is one of the hottest areas of research and business for the tech giants. What seemed like a science-fiction, a few years ago, now seems more like something which is soon to become a part and parcel of life. The reason, I am saying "soon to be" is because of the fact that even though companies like Tesla, Nissan, Cadillac do have self-drivingcar assistance software, but, they still require a human to keep an eye on the road and take control when needed. However, it is fascinating to see how far we have come in terms of innovation and how fast technology is advancing. So much so, that now, with the help of basic deep learning, neural network magic, we can build our own pipeline for autonomous driving! .

Most people usually do not consider the car sitting in their drive way to be on the leading edge of new technology. However, for most people, the personal automobile has now become their initial exposure to new intelligent computational technologies such as fuzzy logic, neural networks, adaptive computing, voice recognition and others. In this chapter we will discuss the various intelligent vehicle systems that are now being deployed into motor vehicles.These intelligent system applications impact every facet of the driver experience and improve both vehicle safety and performance. We will also describe recent developments in autonomousvehicle design and demonstrate that this type of technology is not that far away from deployment. Other applications of intelligent system design apply to adapting the vehicle to thedriver's preferences and helping the driver stay aware. The automobile industry is very competitive and there are many other new advances in vehicle technology that cannot be discussed yet. However, this chapter provides an introduction into those technologies that have already been announced or deployed and shows how the automobile has evolved from a basic transportation device into an advanced vehicle with a host of on-board computational technologies.

Keywords :
Computational Intelligence, Vehicle System, Self-Driving Car, AI, Machine Learning Techniques.

1.INTRODUCTION

1.1 ARTIFICIAL INTELLIGENCE (AI)

AI technologies power self-driving car systems. Developers of self-driving cars use vast amounts of data from image recognition systems, along with machine learning and neural networks, to build systems that can drive autonomously. The neural networks identify patterns in the data, which is fed to the machine learning algorithms.
That data includes images from cameras on self-driving cars from which the neural network learns to identify traffic lights, trees, curbs, pedestrians, street signs and other parts of any given driving environment. Road accidents are one of the major causes of passing, as concurring to report by Deshpande et al that about 3000 individuals passed on day by day because of road accidents, among which half of them are not within the car, other than that it has also been reported that in case a few safety measures are not taken this will develop up to 2.4 million a year making the 5th largest cause of passing within the World.
This number can be enormously diminished by putting independent cars into activity which are far more dependable and respond swiftly than people. It'll moreover cause reduction in the traffic congestion, as the effectiveness of driverless car makes it reliable in a way of keeping exceptionally little crevices between vehicles, and its exceptional administration of speed and time. Taking after the navigation track without considering any other diversion makes it friendlier than the ordinary cars worked by drivers.
AI is much more about the process and the capability for superpowered thinking and data analysis than it is about any particular format or function. Although AI brings up images of high-functioning, human-like robots taking over the world, AI isn't intended to replace humans. It's intended to significantly enhance human capabilities and contributions. That makes it a very valuable business asset.
Nowadays, we consider that the characteristics that make it possible to say whether a technical solution involves AI or not are:

• perception
• learning
• reasoning ```````
• problem-solving
• using natural language

1.2 WHY IS ARTIFICIAL INTELLIGENCE IMPORTANT?

AI automates repetitive learning and discovery through data. Instead of automating manual tasks, AI performs frequent, high-volume, computerized tasks. And it does so reliably and without fatigue. Of course, humans are still essential to set up the system and ask the right questions.

AI adds intelligence to existing products. Many products you already use will be improved with AI capabilities, much like Siri was added as a feature to a new generation of Apple products. Automation, conversational platforms, bots and smart machines can be combined with large amounts of data to improve many technologies. Upgrades at home and in the workplace, range from security intelligence and smart cams to investment analysis.

AI adapts through progressive learning algorithms to let the data do the programming. AI finds structure and regularities in data so that algorithms can acquire skills. Just as an algorithm can teach itself to play chess, it can teach itself what product to recommend next online. And the models adapt when given new data.

AI analyses more and deeper data using neural networks that have many hidden layers. Building a fraud detection system with five hidden layers used to be impossible. All that has changed with incredible computer power and big data. You need lots of data to train deep learning models because they learn directly from the data.

AI achieves incredible accuracy through deep neural networks. For example, your interactions with Alexa and Google are all based on deep learning. And these products keep getting more accurate the more you use them. In the medical field, AI techniques from deep learning and object recognition can now be used to pinpoint cancer on medical images with improved accuracy.

AI gets the most out of data. When algorithms are self-learning, the data itself is an asset. The answers are in the data. You just have to apply AI to find them. Since the role of the data is now more important than ever, it can create a competitive advantage. If you have the best data in a competitive industry, even if everyone is applying similar techniques, the best data will win.

1.3 HOW AI TECHNOLOGY CAN HELP ORGANIZATIONS

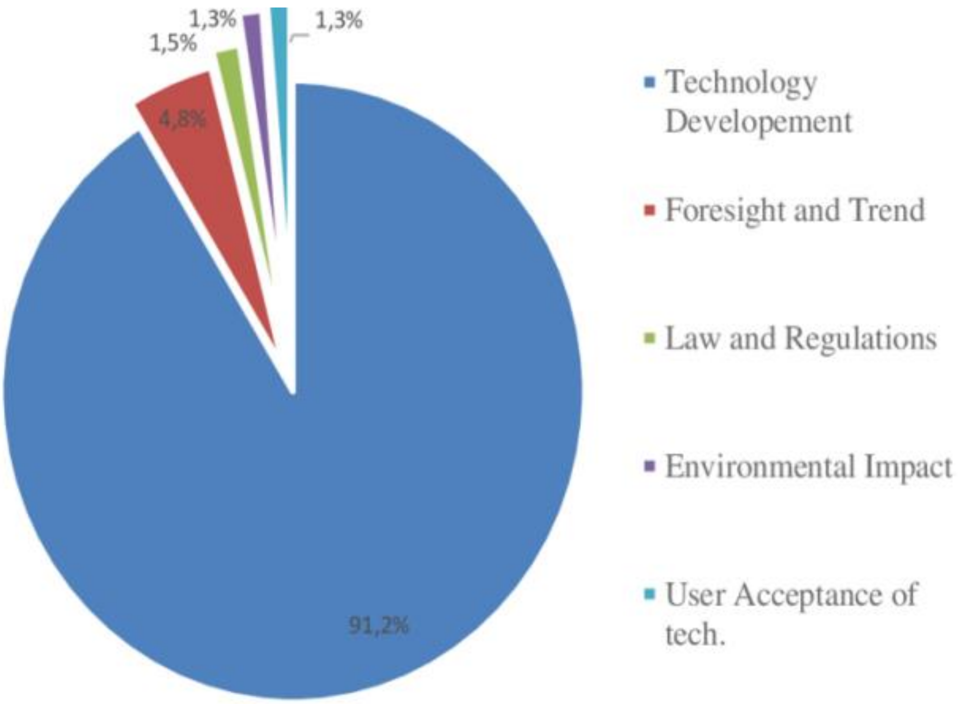
The central tenet of AI is to replicate—and then exceed—the way humans perceive and react to the world. It’s fast becoming the cornerstone of innovation. Powered by various forms of machine learning that recognize patterns in data to enable predictions, AI can add value to your business by:

• Providing a more comprehensive understanding of the abundance of data available

• Relying on predictions to automate excessively complex or mundane tasks

1.4 SIGNIFICANCE AND MOTIVATION

Self-driving technology is perhaps the most debated technology in the automotive industry right now and many companies are developing autonomous driving features to be added on to their production cars. The motivation behind this technology showdown is to improve car safety and efficiency. Sebastian Thrun, the leading engineer of Google's self-driving car project, wrote in his blog that the goal of developing self-driving car is to "help prevent traffic accidents, free up people's time and reduce carbon emissions." According to the World Health Organization, more than 1 million people lose their lives on the road due to car accidents, and C2ES (Center for Climate and Energy Solutions) states that about 60% of total energy consumed by transportation is from automobiles. These numbers show that cars cause serious casualties and are major source of greenhouse gas emission. There have been attempts to solve these problems, but none of the solutions have been particularly effective. However, self-driving technology has potential to change all these problems. If self-driving cars can be driven safer and much more efficiently, it could save valuable lives and preserve the environment.



2.LITERATURE SURVEY

Most of the available systems currently provide the information in terms of number of bicycles parked in each docking stations by means of services available via web. However, When the departure station is empty, a BSS user could be happy to know how the situation will evolve in the next few minutes and if a bike is going to arrive. Vice versa, when an arrival station is full, a prevision on the expected pick-ups at that time of the day would be very useful for the users.

To fulfill these expectations, we envisage services able to make predictions inferring the expected movements towards and from a given docking station. Indeed, several possible smart predictive services can be identified for improving the satisfaction of a user of a BBS using the available information and data.

The basic premise of the bike-sharing concept is sustainable transportation. They differ from traditional, mostly leisure-oriented bicycle rental services, in the following ways:

- They can be “rented” at one location and either returned there or at another location
- They provide fast and easy access
- They have diverse business models
- They make use of applied technology(smart cards and/or mobile phones)
- They are often designed as part of the public transport system.

Rental charges are time-based pay-per ride fees, and in most systems, the first half hour is free of charge. Bike “pick up” and “return” stations operate 24 hours per day, 7 days a week. They are strategically placed at regular intervals throughout the city, making them easily accessible from public transport stations as well as office and shopping areas. The latest systems operate with smart technologies and provide users with real-time bike availability information on the internet. These “smart” bike-sharing systems provide the missing link between existing points of public transportation and desired destinations, offering a new form of mobility that complements the existing public transport systems.

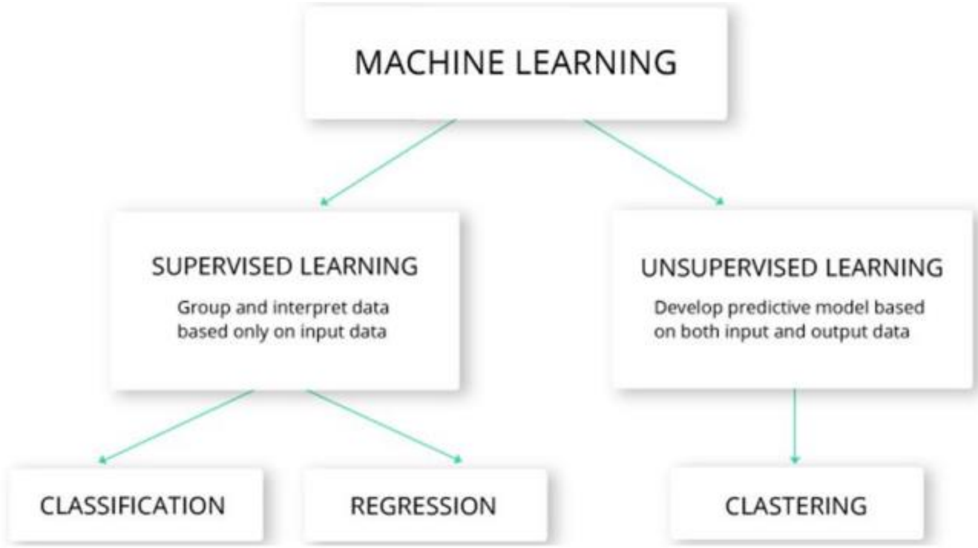
Bike-sharing systems are currently operating in 78 cities in 16 countries using around 70,000 bikes. With the exception of systems in Australia, Canada, China and the recently introduced system in Washington D.C. (USA), all systems are in Europe and most are in France.

Bike-Sharing system has been around since 1965 when a group called Provo introduced it in the bicycle-loving Amsterdam. But the idea of bike-sharing created a buzz only after the entry of smart city concept. It began to be recognized as an effective tool to reduce air pollution, traffic congestion, travel costs, and fossil fuel-dependency while improving public health. Besides, making cities look much more vibrant, cool and cosmopolitan.

However, launching a bike-sharing system in cities and over time making it an appealing mode of transportation is no easy feat. It requires a significant private and public investment and alterations to the built environment while understanding the needs of citizens. Many bike-sharing programs have been initiated amidst much hype yet their popularity has soon declined. They have ended up being used mainly on weekends and for recreation purpose. There are a number of factors that lead to the failure of a bike-sharing system.

3. ALGORITHMS

AI-powered vehicles see the physical world, but how are they able to identify things like street signs, other cars, road markings and many other things encountered on the road? This is where data annotation plays a crucial role. This is when all of the raw training data is prepared through various annotation methods that allow the AI-system to understand what it needs to learn. For the automotive sector, the most common data annotation methods include 3D Point Cloud annotation, video labeling, full scene segmentation and many others.



Machine learning has two learning models: supervised and unsupervised. With unsupervised learning, a machine learning algorithm receives unlabeled data and no instructions on how to process it, so it has to figure out what to do on its own.

With the supervised model, an algorithm is fed instructions on how to interpret the input data. This is the preferred approach to learning for self-driving cars. It allows the algorithm to evaluate training data based on a fully labeled dataset, making supervised learning more useful where classification is concerned.

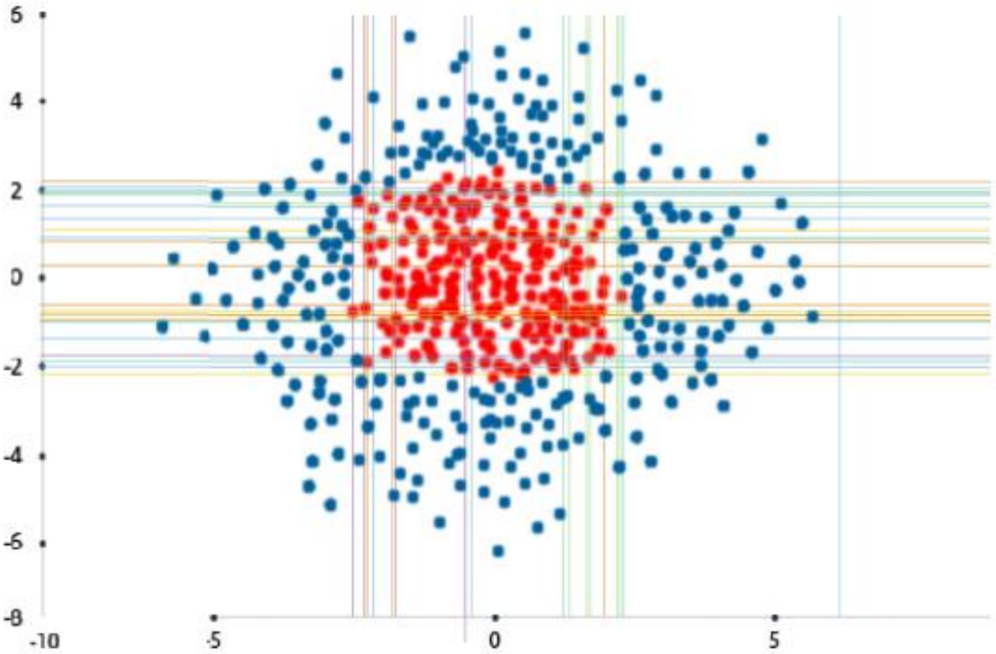
3.1 Machine Learning Algorithms used by Self-Driving Cars

SIFT (Scale-Invariant Feature Transform) for Feature Extraction

SIFT algorithms detect objects and interpret images. For example, for a triangular entered as features. A car can then easily identify the sign using those points.

3.2 AdaBoost for Data Classification

This algorithm collects data and classifies it to boost the learning process and performance of different low-performing classifiers to get a single high-performing classifier for better decision-making.



3.3 TextonBoost for Object Recognition

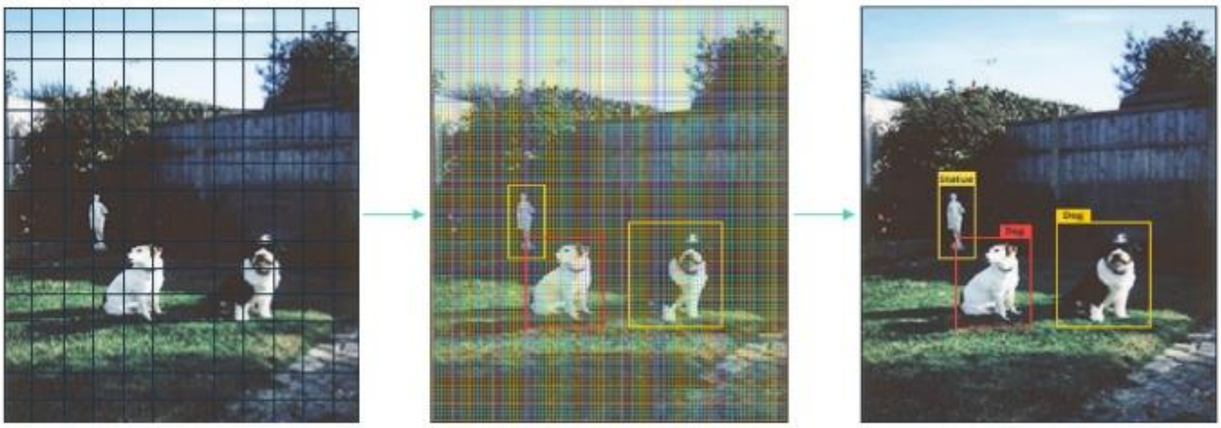
The TextonBoost algorithm does a similar job to AdaBoost, only it receives data from shape, context, and appearance to increase learning with textons (micro-structures in images). It aggregates visual data with common features.

3.4 Histogram of Oriented Gradients (HOG)

HOG facilitates the analysis of an object's location, called a cell, to find out how the object changes or moves.

3.5 YOLO (You Only Look Once)

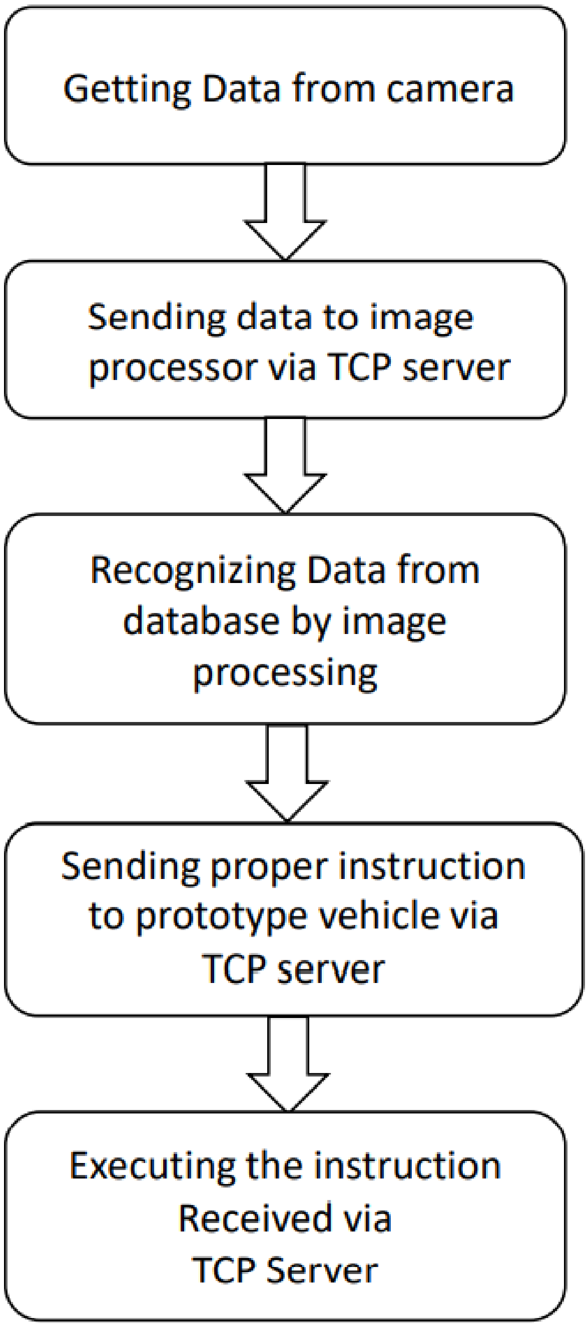
This algorithm detects and groups objects like humans, trees, and vehicles. It assigns specific features to e object that it groups to help the car easily identify them. YOLO is best for identifying and grouping objects.



3.6 Wrap-up

Machine learning algorithms make it possible for self-driving cars to exist. They allow a car to collect data on its surroundings from cameras and other sensors, interpret it, and decide what actions to take. Machine learning even allows cars to learn how to perform these tasks as good as (or even better than) humans.

4.Flowchart of Implementation



5. SYSTEM REQUIREMENTS

1. Udacity's self-driving car simulator

2. Of course, Python and the Pytorch Framework

3. If your machine does not support GPU, then I would recommend using Google Colab to train your network. It provides GPU and TPU hours for free!

4. If you are facing problems gathering your training data, you can use the one provided by Udacity to train your network.

5. The complete code from GitHub - ManajitPal/DeepLearningForSelfDrivingCars: This is an implementation in Pytorch of nvidia's model to build a deep learning neural network for self-driving cars.
and the Colab notebook is available from
(https://colab.research.google.com/drive/1W5I8NYsavde4iy-1uiztNKwwYw7_Pfs_)

6. The Nvidia research paper mentioned in this article can be found (1604.07316v1.pdf (arxiv.org)

6.SOURCE CODE

```
import torch.nn as nn
import torch
import torch.optim as optim
from torch.utils import data
from torch.utils.data import DataLoader
import torchvision.transforms as transforms

import cv2

import numpy as np

import csv

# Step1: Read from the log file
samples = []
with open('data/driving_log.csv') as csvfile:
    reader = csv.reader(csvfile)
    next(reader, None)
    for line in reader:
        samples.append(line)

# Step2: Divide the data into training set and validation set
train_len = int(0.8*len(samples))
valid_len = len(samples) - train_len
train_samples, validation_samples = data.random_split(samples, lengths=[train_len,
valid_len])


# Step3a: Define the augmentation, transformation processes, parameters and dataset
for dataloader
def augment(imgName, angle):
  name = 'data/IMG/' + imgName.split('/')[-1]
  current_image = cv2.imread(name)
  current_image = current_image[65:-25, :, :]
  if np.random.rand() < 0.5:
    current_image = cv2.flip(current_image, 1)
    angle = angle * -1.0
  return current_image, angle

class Dataset(data.Dataset):

    def __init__(self, samples, transform=None):

        self.samples = samples
        self.transform = transform
```

```
    def __getitem__(self, index):

        batch_samples = self.samples[index]

        steering_angle = float(batch_samples[3])

        center_img, steering_angle_center = augment(batch_samples[0], steering_angle)
        left_img, steering_angle_left = augment(batch_samples[1], steering_angle + 0.4)
        right_img, steering_angle_right = augment(batch_samples[2], steering_angle - 0.4)

        center_img = self.transform(center_img)
        left_img = self.transform(left_img)
        right_img = self.transform(right_img)

        return (center_img, steering_angle_center), (left_img, steering_angle_left), (right_img, steering_angle_right)

    def __len__(self):
        return len(self.samples)

# Step3b: Creating generator using the dataloader to parallasize the process
transformations = transforms.Compose([transforms.Lambda(lambda x: (x / 255.0) - 0.5)])

params = {'batch_size': 32,
          'shuffle': True,
          'num_workers': 4}

training_set = Dataset(train_samples, transformations)
training_generator = data.DataLoader(training_set, **params)


validation_set = Dataset(validation_samples, transformations)
validation_generator = data.DataLoader(validation_set, **params)

# Step4: Define the network
class NetworkDense(nn.Module):

    def __init__(self):
        super(NetworkDense, self).__init__()
        self.conv_layers = nn.Sequential(
            nn.Conv2d(3, 24, 5, stride=2),
            nn.ELU(),
            nn.Conv2d(24, 36, 5, stride=2),
            nn.ELU(),
            nn.Conv2d(36, 48, 5, stride=2),
            nn.ELU(),
            nn.Conv2d(48, 64, 3),
            nn.ELU(),
```

```
            nn.Conv2d(64, 64, 3),
```

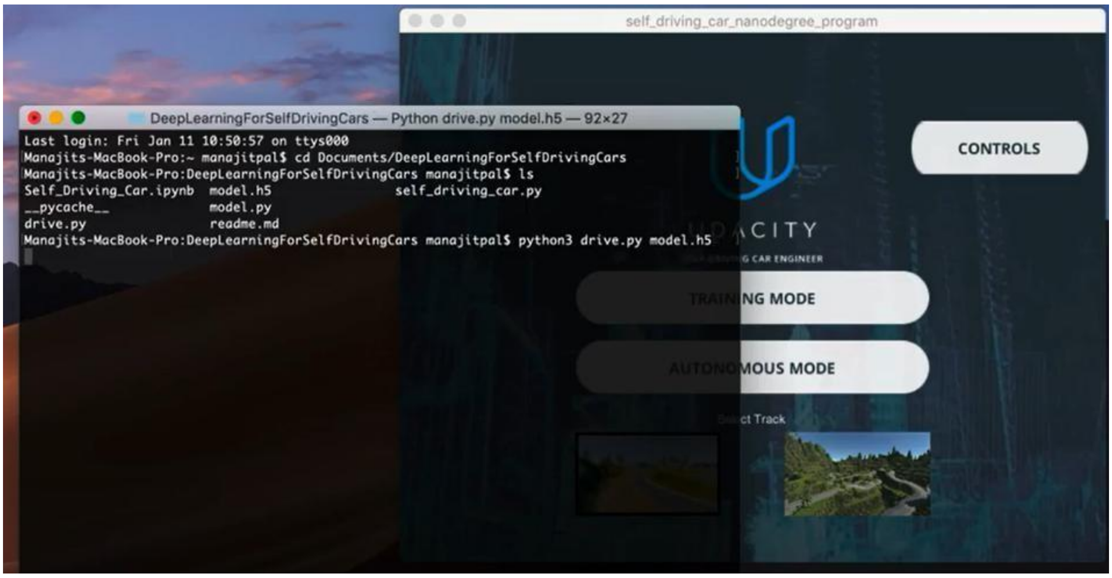


```
            nn.Dropout(0.25)
        )
        self.linear_layers = nn.Sequential(
            nn.Linear(in_features=64 * 2 * 33, out_features=100),
            nn.ELU(),
            nn.Linear(in_features=100, out_features=50),
            nn.ELU(),
            nn.Linear(in_features=50, out_features=10),
            nn.Linear(in_features=10, out_features=1)
        )

    def forward(self, input):
        input = input.view(input.size(0), 3, 70, 320)
        output = self.conv_layers(input)
        output = output.view(output.size(0), -1)
        output = self.linear_layers(output)
        return output


class NetworkLight(nn.Module):

    def __init__(self):
        super(NetworkLight, self).__init__()
        self.conv_layers = nn.Sequential(
            nn.Conv2d(3, 24, 3, stride=2),
            nn.ELU(),
            nn.Conv2d(24, 48, 3, stride=2),
            nn.MaxPool2d(4, stride=4),
            nn.Dropout(p=0.25)
        )



 self.linear_layers = nn.Sequential(
            nn.Linear(in_features=48*4*19, out_features=50),
            nn.ELU(),

nn.Linear(in_features=50, out_features=10),
            nn.Linear(in_features=10, out_features=1)
        )


    def forward(self, input):
        input = input.view(input.size(0), 3, 70, 320)
        output = self.conv_layers(input)
        output = output.view(output.size(0), -1)
        output = self.linear_layers(output)
        return output
```

```
# Step5: Define optimizer
model = NetworkLight()
optimizer = optim.Adam(model.parameters(), lr=0.0001)

criterion = nn.MSELoss()

# Step6: Check the device and define function to move tensors to that device
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
print('device is: ', device)

def toDevice(datas, device):

  imgs, angles = datas
  return imgs.float().to(device), angles.float().to(device)

# Step7: Train and validate network based on maximum epochs defined
max_epochs = 22

for epoch in range(max_epochs):

    model.to(device)

    # Training
    train_loss = 0
    model.train()
    for local_batch, (centers, lefts, rights) in enumerate(training_generator):
        # Transfer to GPU
        centers, lefts, rights = toDevice(centers, device), toDevice(lefts, device),
toDevice(rights, device)


# Model computations
        optimizer.zero_grad()
        datas = [centers, lefts, rights]
        for data in datas:
            imgs, angles = data
#             print("training image: ", imgs.shape)
            outputs = model(imgs)
            loss = criterion(outputs, angles.unsqueeze(1))
            loss.backward()
            optimizer.step()

            train_loss += loss.data[0].item()


  if local_batch % 100 == 0:
            print('Loss: %.3f '
                 % (train_loss/(local_batch+1)))
```

```
    # Validation
    model.eval()
    valid_loss = 0
    with torch.set_grad_enabled(False):
        for local_batch, (centers, lefts, rights) in enumerate(validation_generator):
            # Transfer to GPU
            centers, lefts, rights = toDevice(centers, device), toDevice(lefts, device),
toDevice(rights, device)

            # Model computations
            optimizer.zero_grad()
            datas = [centers, lefts, rights]
            for data in datas:
                imgs, angles = data
#                   print("Validation image: ", imgs.shape)
                outputs = model(imgs)
                loss = criterion(outputs, angles.unsqueeze(1))

                valid_loss += loss.data[0].item()

            if local_batch % 100 == 0:
                print('Valid Loss: %.3f '
                     % (valid_loss/(local_batch+1)))

# Step8: Define state and save the model wrt to state
state = {
        'model': model.module if device == 'cuda' else model,

}

torch.save(state, 'model.h5')
```

7.OUTPUT


self_driving_car_nanodegree_program
DeepLearningForSelfDrivingCars — Python drive.py model.h5 — 92×27
Last login: Fri Jan 11 10:50:57 on ttys000
Manajits-MacBook-Pro:~ manajitpal$ cd Documents/DeepLearningForSelfDrivingCars
Manajits-MacBook-Pro:DeepLearningForSelfDrivingCars manajitpal$ ls
Self_Driving_Car.ipynb model.h5 self_driving_car.py
__pycache__ model.py
drive.py readme.md
Manajits-MacBook-Pro:DeepLearningForSelfDrivingCars manajitpal$ python3 drive.py model.h5
CONTROLS
TRAINING MODE
AUTONOMOUS MODE


self_driving_car_nanodegree_program
DeepLearningForSelfDrivingCars
torch.Size([1, 48, 4, 19])
0.06576285 0.03587919999999972
torch.Size([70, 320, 3])
torch.Size([1, 3, 70, 320])
torch.Size([1, 48, 4, 19])
-0.0150141865 0.04178759999999976
torch.Size([70, 320, 3])
torch.Size([1, 3, 70, 320])
torch.Size([1, 48, 4, 19])
-0.0150141865 0.04031599999999977
torch.Size([70, 320, 3])
torch.Size([1, 3, 70, 320])
torch.Size([1, 48, 4, 19])
-0.0150141865 0.03884439999999978
torch.Size([70, 320, 3])
torch.Size([1, 3, 70, 320])
torch.Size([1, 48, 4, 19])
-0.020333782 0.04326839999999964
torch.Size([70, 320, 3])
torch.Size([1, 3, 70, 320])
torch.Size([1, 48, 4, 19])
-0.020333782 0.04191239999999964
torch.Size([70, 320, 3])
torch.Size([1, 3, 70, 320])
torch.Size([1, 48, 4, 19])
-0.020333782 0.04055639999999646
ANGLE
-0.51°
Mode: Autonomous
9.58
MPH


self_driving_car_nanodegree_program
DeepLearningForSelfDrivingCars
torch.Size([1, 48, 4, 19])
torch.Size([70, 320, 3])
torch.Size([1, 3, 70, 320])
torch.Size([1, 48, 4, 19])
torch.Size([70, 320, 3])
torch.Size([1, 3, 70, 320])
torch.Size([1, 48, 4, 19])
torch.Size([70, 320, 3])
torch.Size([1, 3, 70, 320])
torch.Size([1, 48, 4, 19])
torch.Size([70, 320, 3])
torch.Size([1, 3, 70, 320])
torch.Size([1, 48, 4, 19])
torch.Size([70, 320, 3])
torch.Size([1, 3, 70, 320])
torch.Size([1, 48, 4, 19])
torch.Size([70, 320, 3])
torch.Size([1, 3, 70, 320])
torch.Size([1, 48, 4, 19])
ANGLE
Mode: Autonomous
MPH

8. ADVANTAGES AND DISADVANTAGES

8.1 ADVANTAGES

1. Decreased the Number of Accidents: Autonomous cars prevent human errors from happening as the system controls the vehicle. It leaves no opportunity for distraction, not just like humans who are prone to interruptions. It also uses complicated algorithms that determine the correct stopping distance from one vehicle to another. Thereby, lessening the chances of accidents dramatically.

2. Lessens Traffic Jams: Driverless cars in a group participate in platooning. This allows the vehicles to brake or accelerates simultaneously. Platoon system allows automated highway system which may significantly reduce congestion and improve traffic by increasing up the lane capacity. Autonomous cars communicate well with one another. They help in identifying traffic problems early on. It detects road fixing and detours instantly. It also picks up hand signals from the motorists and reacts to it accordingly.

3. Stress-Free Parking: Autonomous cars drop you off at your destination and directly heads to a detected vacant parking spot. This eliminates the wasting of time and gas looking for a vacant one.

4. Time-Saving Vehicle: As the system takes over the control, the driver has a spare time to continue work or spend this time catching up with their loved ones without the having the fear about road safety.

5. Accessibility to Transportation: Senior citizens and disabled personnel are having difficulty driving. Autonomous vehicles assist them towards safe and accessible transportation.

8.2 DISADVANTAGES

1. Expensive: High-technology vehicles and equipment are expensive. They prepare a large amount of money for research and development as well as in choosing the finest and most functional materials needed such as the software, modified vehicle parts, and sensors. Thus, the cost of having Autonomous cars is initially higher. However, this may lower down after 10 years giving way for the average earner people to have one.

2. Safety and Security Concerns: Though it has been successfully programmed, there will still be the possible unexpected glitch that may happen. Technologies are continuously updating and almost all of this equipment may have a faulty code when the update was not properly and successfully done.

3. Prone to Hacking: Autonomous vehicles could be the next major target of the hackers as this vehicle continuously tracks and monitors details of the owner. This may lead to the possible collection of personal data.

4. Fewer Job Opportunities for Others: As the artificial intelligence continues to overcome the roles and responsibilities of humans, taxi, trucks, or even co-pilots may be laid off as their services will no longer be needed. This may significantly impact the employment rate and economic growth of a certain country.

5. Non-Functional Sensors: Sensors failures often happened during drastic weather conditions. This may not work during a blizzard or a heavy snowfall. Our technology still continues to develop and to be tested. Autonomous cars may provide a significant comfort we needed. However, we need to bear in mind that there are still disadvantages affiliated with it.

9. AREAS OF APPLICATIONS OF AI

- Social Media Monitoring
- Marketing Chartbots
- Manufacturing Robots
- Healthcare Management
- Self- driving Cars
- Smart Assistants
- Automated Financial Investing
- Virtual travel booking agent

10. CONCLUSION

The scientific community now accepts autonomous vehicles and autonomous driving as feasible solution due to advancement in AI. With artificial intelligence, autonomous vehicles and driving systems may make a choice that propels the industry into a new era of rapid development. Despite this, artificial intelligence has significant limitations, limiting the growth of autonomous driving. This work has conducted a comprehensive survey over artificial intelligence in autonomous vehicles, systems, and driving experiences. In observations, it is found that there is a lack of safety standards for autonomous systems, and AI is an important concept while designing the safety standards for futuristic autonomous systems. Furthermore, a comparative analysis of various studies on autonomous systems shows that integrating two or more advanced technologies (blockchain, IoT, cloud computing, fog computing, edge computing, and artificial intelligence) is required to make autonomous systems a reality. Here, the focus is drawn on how artificial intelligence
monitors the vehicle’s activities and movements. Intelligent tools are necessary for autonomous vehicle design and development. In this work, various latest release of tools and frameworks are analyzed in context of design techniques and programming languages used. The operational testing is essential for effective functionality of AVs. Thus, various testing techniques employed by organizations and researchers are highlighted in this work. The limitations of existing testing techniques are also discussed.

11. References

https://towardsdatascience.com/deep-learning-for-self-driving-cars-7f198ef4cfa2

Udacity’s self-driving car simulator
Of course, Python and the Pytorch Framework
If your machine does not support GPU, then I would recommend using Google Colab totrain your network. It provides GPU and TPU hours for free.