ÇANKAYA UNIVERSITY
FACULTY OF ENGINEERING
COMPUTER ENGINEERING DEPARTMENT

Project Report

Version 2

CENG 408

Innovative System Design and Development 2

202017

3D OBJECT DETECTION FOR SELF-DRIVING CARS

Burak Çolakoğlu
201611012
Furkan Han Keçeli
201611037

Advisor: *Faris Serdar Taşel*
Co-Advisor: *Roya Choupani*

Table of Contents

Abstract

Fully autonomous vehicles are vehicles that we believe will become very common in the future, and are intended to be produced to transport passengers to the target destination without any driver intervention. A lot of research and work has been done on autonomous vehicles so far. The reason for the acceleration of these efforts today is that we are now very close to the mass production of autonomous vehicles. Perhaps the most important tools of autonomous vehicles, which are still being studied, are cameras and sensors that observe their surroundings. However, even now, there are various cameras and sensors in the vehicles used. What matters is how the data from these cameras will be used by autonomous vehicles. Autonomous vehicles need to recognize the objects they see on the camera and make a decision accordingly. Frankly, if we look at car accidents in traffic, we cannot say that people are very successful in this regard. In traffic, especially when driving, response time is of vital importance. The same is true for autonomous vehicles. He needs to brake or change lanes as soon as he sees a pedestrian on the road in traffic. Our aim in this project is to gain time by improving the ability of these self-driving tools to perceive the objects around them in three dimensions.

Key words:

3D Object Detection, Self-Driving Cars, Deep Learning

Özet:

Tam otonom araçlar herhangi bir sürücü müdahalesi bulunmaksızın gidilmesi gereken hedefe yolcuları taşımak üzere üretilmek istenen, gelecekte çok yaygın hale geleceğine inandığımız araçlardır. Otonom araçlar hakkında şu ana kadar çok fazla araştırma ve çalışma yapılmıştır. Günümüzde bu çalışmaların hızlanmasının sebebi artık otonom araçların seri üretimine çok yakın olmamızdır. Şu anda üzerinde hala çalışmalar yürütülen otonom araçların belki de en önemli araçları çevrelerini gözlemleyen kameraları ve sensörleridir. Ancak şu anda bile kullanılan araçlarda çeşitli kameralar ve sensörler mevcut. Önemli olan bu kameralardan alınan verilerin otonom araçlar tarafından nasıl kullanılacağıdır. Otonom araçların kamerada gördüğü cisimleri tanıması ve buna göre bir karar vermesi gerekiyor. Açıkçası trafikteki araba kazalarına bakarsak insanların bu konuda çok başarılı olduğunu söyleyemeyiz. Trafikte özellikle araç kullanırken tepki süresi hayati bir öneme sahiptir. Otonom araçlarda da aynısı geçerlidir.

Trafikte giderken yola çıkan bir yayayı görür görmez fren yapması ya da şerit değiştirmesi gerekir. Bizim bu projedeki amacımız kendi kendini kullanabilen bu araçların çevrelerindeki nesneleri üç boyutlu olarak algılamasını geliştirerek zamandan kazanç sağlamaktır.

Anahtar Kelimeler:

3 Boyutlu nesne algılama, Otonom arabalar, Derin Öğrenme.

1. Introduction

1.1 Problem Statement

Since object detection is generally studied in 2D, the necessary importance and work has not been shown to 3-dimensional object detection. This reduces the success rate in recognizing objects.

1.2 Background or Related Work

We know that there are studies of large companies and computer scientists on object detection in autonomous vehicles. However, these studies are generally about 2-dimensional object detection. There are limited studies in 3D object detection. Kaggle organized a competition to increase these studies, and many computer scientists and engineers participated in this competition and came up with their own solutions.

1.3 Solution Statement

The algorithm needs to be developed in order to minimize the time it takes to create these correctly aligned bound boxes.

For this, it is necessary to understand the data first. To understand the data, it is necessary to visualize the data, load the training data frame and then group the data by object categories.

After understanding the data and processing the data with the Convolutional Neural Network, we need to predict bound boxes. If the objectness score is over than 0.5, we identify the class confidence then we apply the non-maximum suppression after that we bound boxes with object detection.

1.4 Contribution

If we build and optimize algorithms successfully. It will make a great contribution to the development of autonomous vehicles and we will contribute to other computer scientists working on this subject with this project.

2. Literature Search

An autonomous car is a vehicle capable of operating driving tasks and performing necessary functions itself without human involvement. It is most often used as an ego-vehicle term. Meaning the self-vehicle which refers to the vehicle being controlled autonomously. Ego vehicles driving tasks can be defined in three sub-tasks: Perception(or perceiving the surrounding environment), Motion Planning(planning to arrive from X to Y) and Vehicle Control(controlling the ego vehicle). These three sub-tasks are main driving functions and must be performed continuously while ego vehicles are on the way. The standards for the purpose of driving tasks are constantly evolving for driving automation, we will consider the suggestion of the Society of Automotive Engineers which has been examined in 2014.Lateral Control which refers to performing steering actions like turning left, right, going forward and so on. Longitudinal Control, controlling braking and accelerating actions of the ego-vehicle along the roadway. Object and Event Detection and Response(OEDR) is a vital ability to detect events and objects to affect driving tasks immediately and react to its current environment appropriately. OEDR consists of a large portion of Autonomous Driving. Planning is another important aspect of driving. Planning primarily involved by long term and short term plans to travel to a desired destination and perform manoeuvres. Finally, there are miscellaneous tasks that drivers do during driving. Actions include interacting with other car drivers, signaling, hand waving and so on.

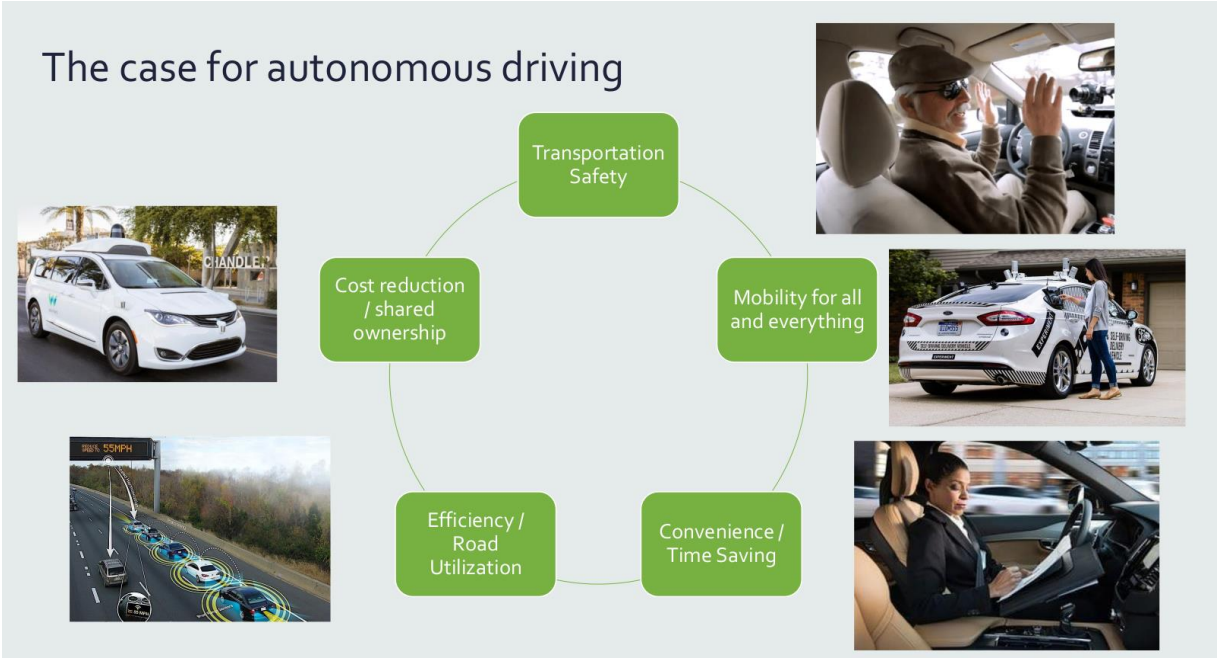

Figure 1. Representing the case of autonomous driving.

Autonomous Vehicles consisted of 5 different automation levels from zero to five.

Level 0- No Automation: There is no driving automation. Everything is done by the driver.

Level 1- Driving Assistance: We are in Level 1 autonomy, if the autonomous system assists the driver for performing either longitudinal or lateral tasks. Adaptive Cruise Control is a good example of the Level 1.

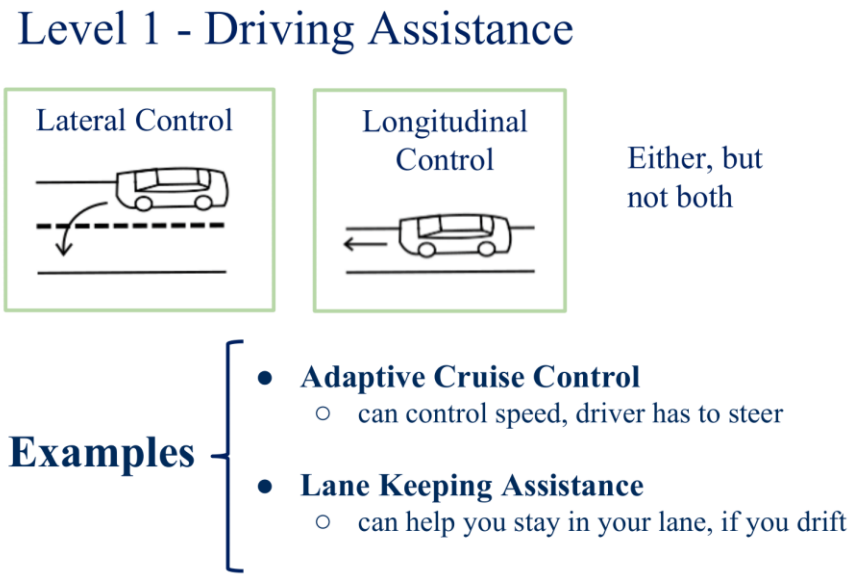

Figure 2. Showing Level 1 automation.

Level 2- Partial Automation: In this level, autonomous vehicles perform both control tasks, longitudinal and lateral tasks. General Motors Super Cruise and Nissan's Pro Pilot Assist features are good examples of automation level of 2. Still, car drivers monitoring to the system is necessary.

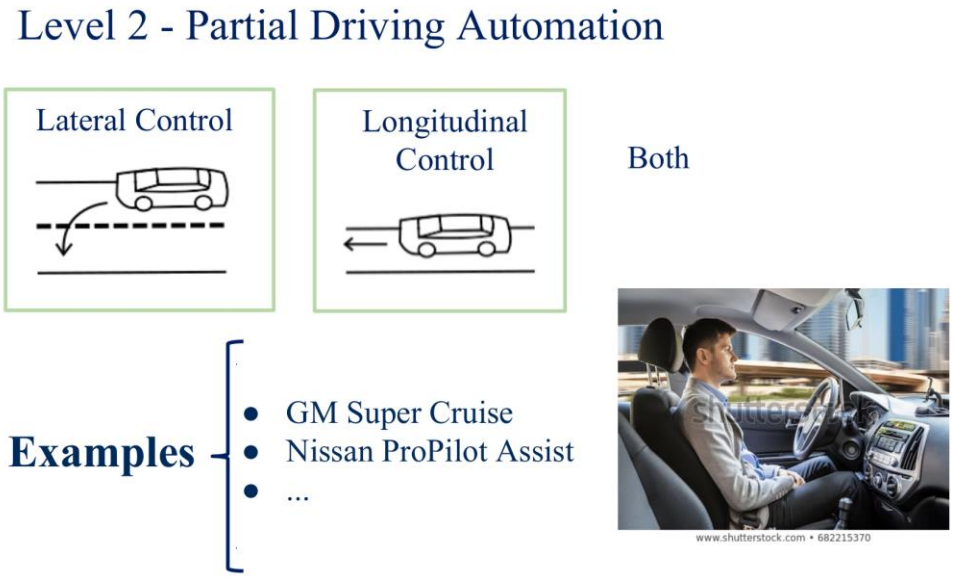

Figure 3. Showing Level 2 automation.

Level 3- Conditional Automation: In this level of 3 also known as an Level of Conditional Automation, ego system can perform OEDR in addition to control tasks. However, to take control in the case of any failure is vita for the ego system. Essential difference between partial and conditional automation, it is not obligatory to focus in some of the situations by driver. Ego vehicles can alert the driver at this level. Also, it is not always possible to know for the autonomy system when the failure will happen. Audi Luxury Sedan an example of level three, which can navigate unmonitored in slow traffic.

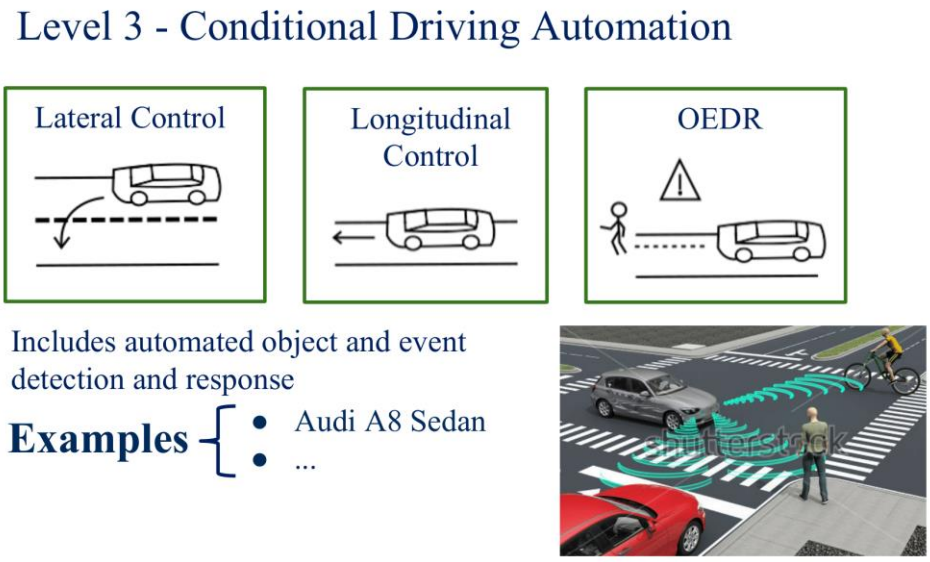

Figure 4. Showing Level 3 automation.

Level 4- High Automation: In this level of 4, the ego system is adequate to reach minimal risk conditions. The driver doesn't need to intervene in the case of an emergency. The Ego system can handle emergencies itself but may still consider giving control to the driver to avoid pulling over to the side of the road unnecessarily. Passengers can do some other stuff like checking phones or watching movies. The ego system is capable of managing emergencies and able to keep passengers safe. Waymo, the parent company of Google, has deployed for public transport in this level in fall of 2018.

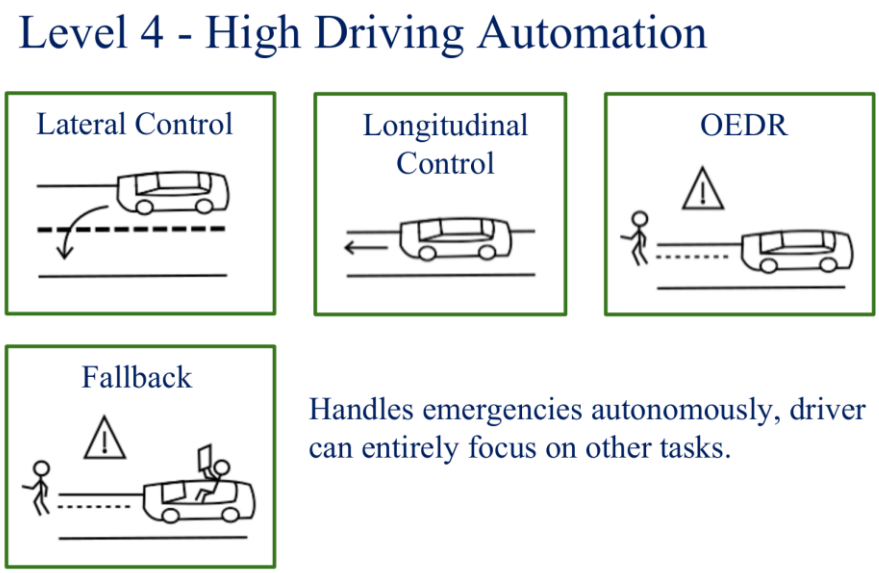

Figure 5. Showing Level 4 automation.

Level 5- Full Driving Automation: In this level of 5, the ego system is fully autonomous and ODD is unlimited. The Ego system can operate under all conditions. There is no steering and pedals. All driving tasks operating by the autonomous system. There is any examples for level five yet.

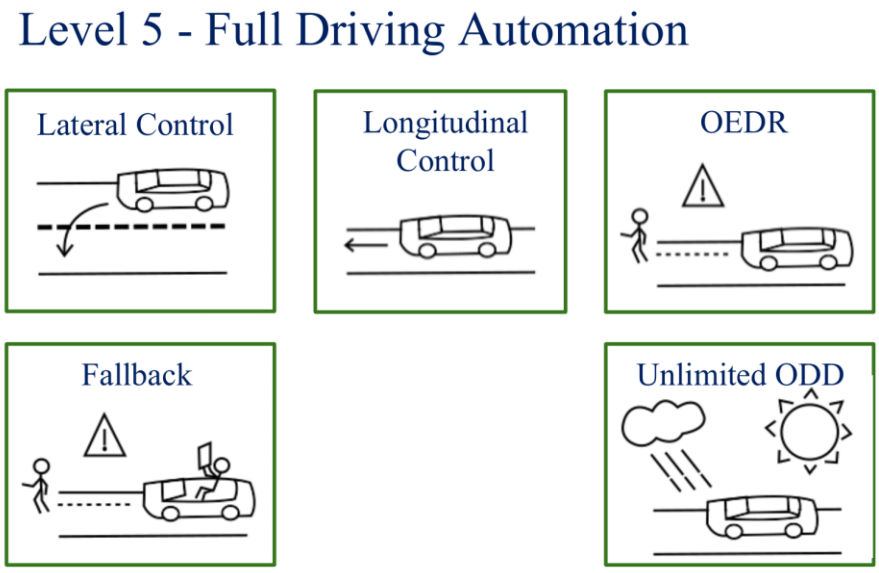

Figure 6. Showing Level 5 automation.

With the increase of the popularity Self-driving Cars and availability of big data and the cutting-edge technologies of computing hardwares, investments and launching startup companies are increasing.

Ford Otosan: Ford Otosan will be developing Turkey's first Level-4 Autonomous Truck that will be a game-changer in the transportation and logistics ecosystem.

TRI-AD: Toyota Research Institute - Advanced Development also known as TRI-AD is a partnership between Toyota, Aisin, Denso established in March 2018. The company's mission is to "Become a world-class software & technology company and build the safest car in the world". TRI-AD is a bridge to leverage the cutting-edge research in AI for automated driving created by TRI into real products that can have a substantial positive impact on mobility for the world [2].

Argo AI: Argo AI is a technology platform company working with leading automakers to deliver a fully integrated self-driving system that makes getting around cities safe, easy, and enjoyable for all [3].

Tesla Autopilot: Tesla is one of the famous companies in the field Autonomous Driving. They are planning to become the first company to realize Level 5 automation to the world.

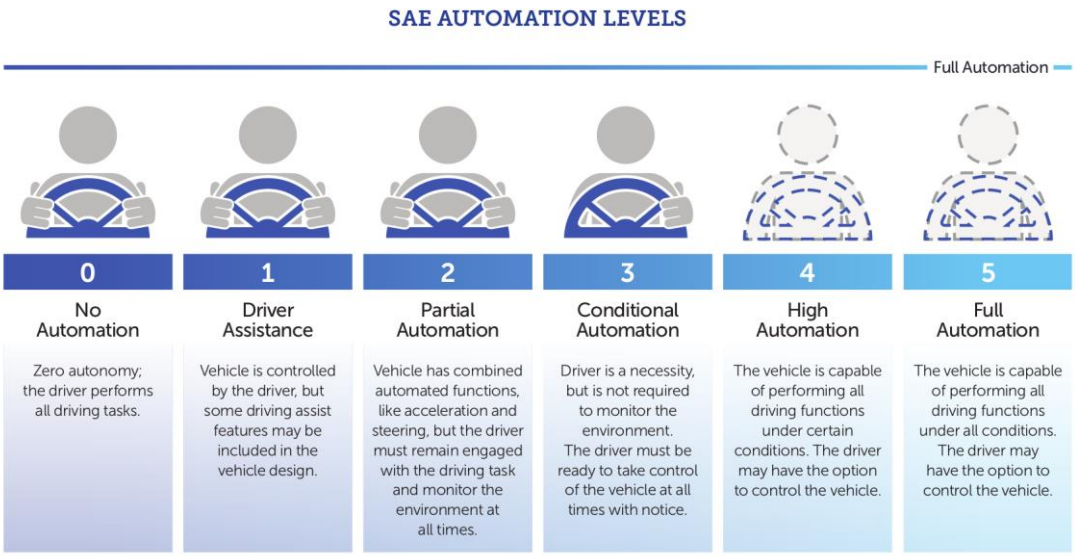

Figure 7. Representing Automation Levels.

2.1 Requirements for Perception

2.1.1 What is perception?

We have to be able to identify what objects around us and what it is; a bus, a car, a pedestrian etc. It is not easy for computer systems to detect patterns quickly as humans do.

2.1.2 Goals for perception

We have to be able to identify what objects around us and what it is; a bus, a car, a pedestrian etc. It is not easy for computer systems to detect patterns quickly as humans do.

Figure 8. Showing static objects in real life.

Second is dynamic elements. We need to identify other vehicles on the road such as for wheelers like trucks, cars, buses, and so on.

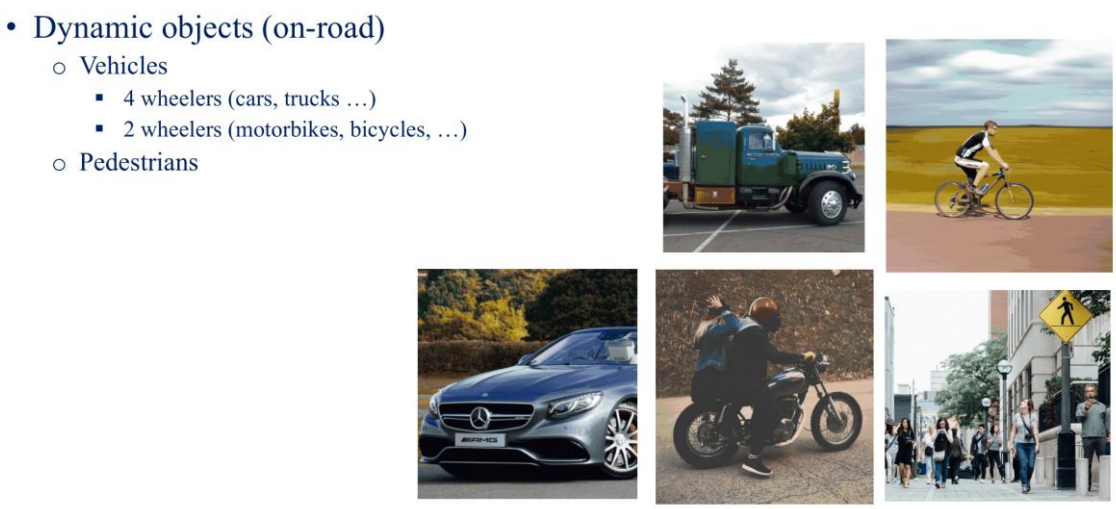

Figure 9. Showing dynamic objects in real life.

2.1.2 Challenges to perception

There are some challenges in perception tasks. First challenge is performing robust perception. Detection and segmentation can be solved with modern machine learning algorithms and methods, but there is still academic and industrial research to improve the performance and reliability to achieve human level.

2.1.2.1 Possible solution to perception challenges

Access to large datasets is critical to come over with robustness problems. Segmentation and detection models would be performed better and robustly with more training data.

2.1.3 Sensors for perception

Our dataset has been collected by 10 host cars. Each of the host cars has seven cameras and one LIDAR sensor. So, we will just discuss Camera and LIDAR sensors. Cameras are an essential exteroceptive sensor and widely used in autonomous driving. Some of the group say that cameras are the only required one for self-driving. But the state of the art performance is not enough alone.

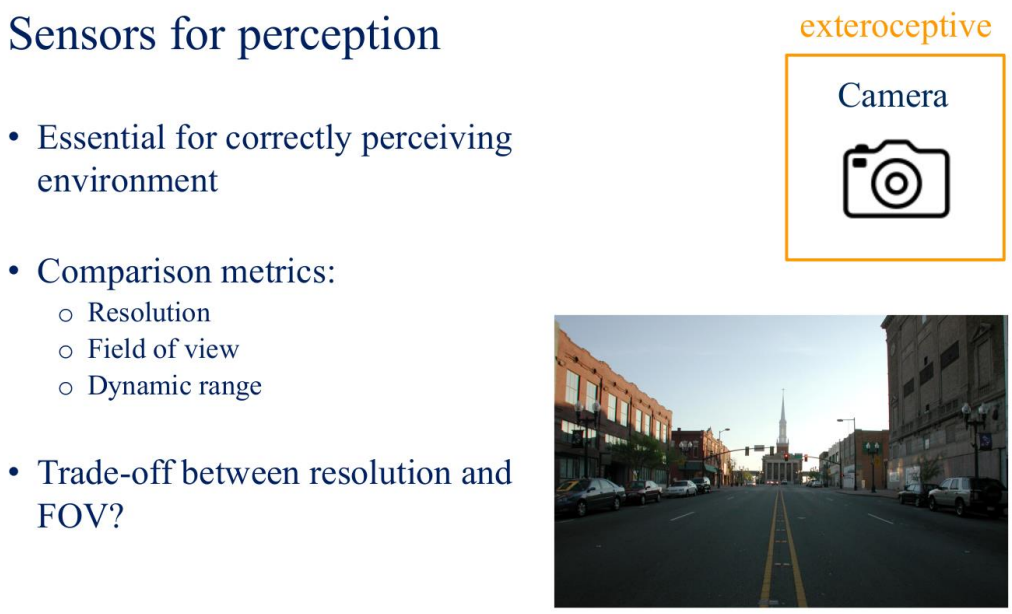

Figure 10. Measure metrics of the Camera.

LiDAR sensors use pulsed lasers at a surface and calculate the time it takes to return the source.It provides multilayered units, so that it is capable of capturing high resolution point cloud of the surrounding environment. It can be used for 3D object detection. LiDARs have a very good range of sensing. However, it can be affected by bad weather and lighting conditions. Furthermore, a LIDAR requires at least two returns to identify an object, which means the effective detection range decreases with object size. LIDARs produce large sizes of data, which increase cost of the processing [4].

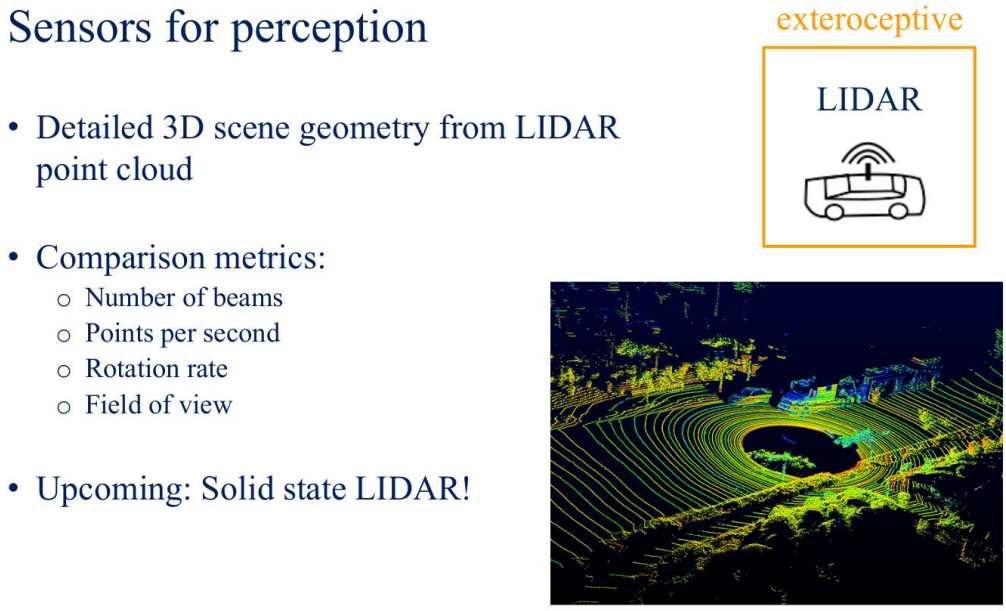

Figure 11. Measure metrics of the LIDAR.

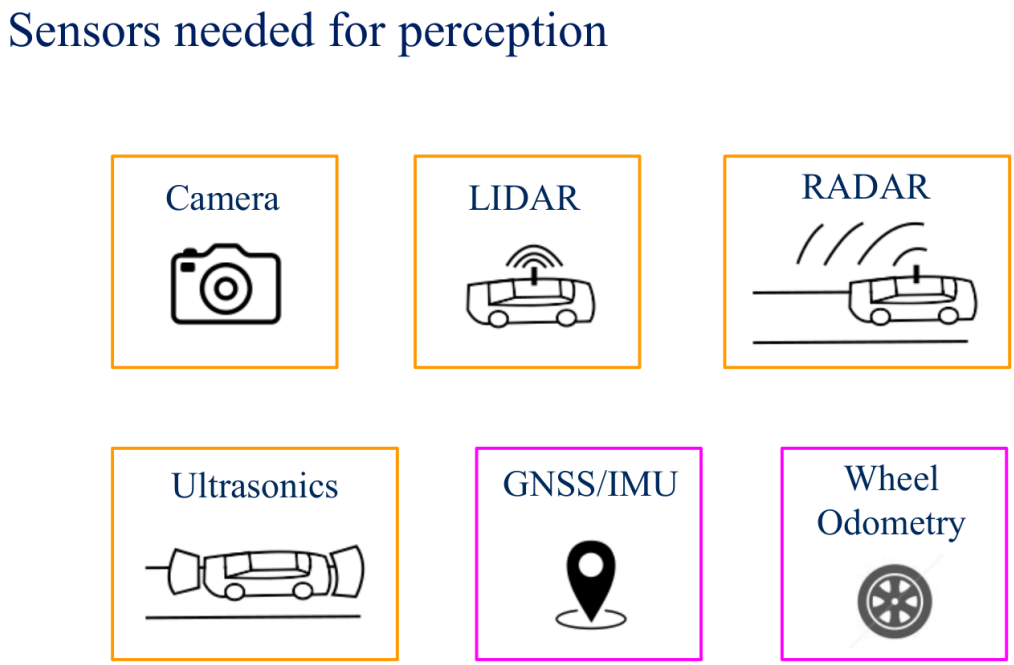

Figure 12. Orange boxes representing exteroceptive sensors. Pink's representing proprioceptive sensors.

2.2 Previous Works

2.2.1 SCNN-based Real-time Object Detection for AV Using LiDAR Data

In this article, researchers apply a new network called Spiking Neural Network(SNN) in order to high energy consumption of the CNN-based approaches. But as they said, SNN is an upcoming candidate because of its lower energy consumption than Convolution operations [5].

2.2.1.1 Conclusion and Dataset

Researchers use KITTI 3D point cloud dataset by considering the power consumption [6] of 3D object detection. Low energy consumption is an essential necessity for practical implementation. Autonomous Vehicles are one of them [5]. In Conclusion, high energy consumption is a biggest concern for practical applications. The mentioned paper considers the energy consumption over 3D point cloud data. They have worked on the development of a SCNN-based YOLOv2 architecture for 3D object detection. The proposed neural network reached the state-of-the-art accuracy in BEV and full 3D detection [5].

2.2.2 Multi-column Deep Neural Networks for Image Classification

In this paper, researchers applied Small (often minimal) receiver fields of convolutional, winner take neurons to large deep networks, and resulting to sparsely connected layers. Just the winner neural layers are trained [7].

2.2.2.1 Conclusion and Dataset

Researchers use MNIST, NIST SD 19, Chinese Characters, Traffic Signs, CIFAR 10 and NORB dataset. It's reported that this was the first human-competitive results put to use in computer vision benchmarks.

Dataset	Best result of others [%]	MCDNN [%]	Relative improv. [%]
MNIST	0.39	0.23	41
NIST SD 19	see Table 4	see Table 4	30-80
HWDB1.0 on.	7.61	5.61	26
HWDB1.0 off.	10.01	6.5	35
CIFAR10	18.50	11.21	39
traffic signs	1.69	0.54	72
NORB	5.00	2.70	46

Figure 13. Results and relative improvements on different datasets.

2.2.3 Sparse Voxel-Graph Attention Network for 3D Object Detection from Point Clouds

In this paper, researchers say SVGA-Net, a unique end-to-end trainable neural network which generally contains sparse-to-dense regression and voxel-graphs to complete 3D object detection missions using LIDAR point clouds [8].

2.2.3.1 Conclusion and Dataset

Researchers use KITTI 3D detection dataset. They reported that they have proposed a novel SVGA-Net for 3D Object Detection from LIDAR data.

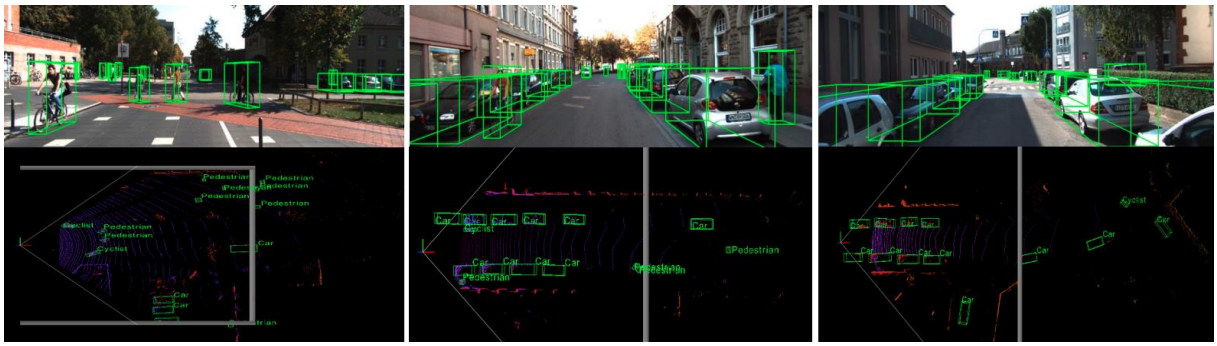

Figure 14. Qualitative 3D detection results of SVGA-Net on the KITTI test set. The detected objects are shown with green 3D bounding boxes and the relative labels. The upper is the 3D object detection result projected onto the RGB image and the bottom is the result in the corresponding point clouds.

2.3 3D Object Detection for Self-Driving Cars

2.3.1 Dataset

We choose a dataset from the Lyft company. It is a total of 117 GB. The data split between training and testing. Dataset includes camera images and 3D point cloud. Dataset has been collected by Lyft ego vehicles which is a number of 10 cars. Each ego vehicles equipped with seven cameras and one LIDAR sensor. There are pre-labelled 55,000 3D annotations on objects. LIDAR sensors shoot lasers 360 degrees in order to identify patterns and get 3D spatial geometric information. LIDAR produces PCD with 216,000 points at 10 Hz.

2.3.2 Tools and Libraries

We decided to use Python language for training and visualize the dataset clearly. We plan to use our school CUDA labs or Google Cloud Platform(GCD) for training and pre-processing of the dataset. For mislabelled data, we are planning to use CVAT and Scalabel which are browser based tools to annotate the data. Bash the command processor will be used to run python script over the remote server.

3. Summary

3.1 Technology Used

Camera, Monocular LIDAR and Stereo lidar were used for data collection. These cameras and sensors were installed in 10 host vehicles and were enabled to actively collect data in Palo Alto, California. The dataset lyft generated with this data. We used Kaggle API to download lyft dataset. To process data we used Jupyter Notebook with Deep Learning frameworks and processing libraries. Also we use Tensorflow and Keras which uses the backend of Tensorflow. Jupyter Notebook is an open source program that provides an interactive environment for various programming languages. We also use Jupyter Lab which is an interactive web based development environment for Jupyter Notebook. Jupyter Lab has a good user interface to support wide range of workflows in data science, machine learning and deep learning.

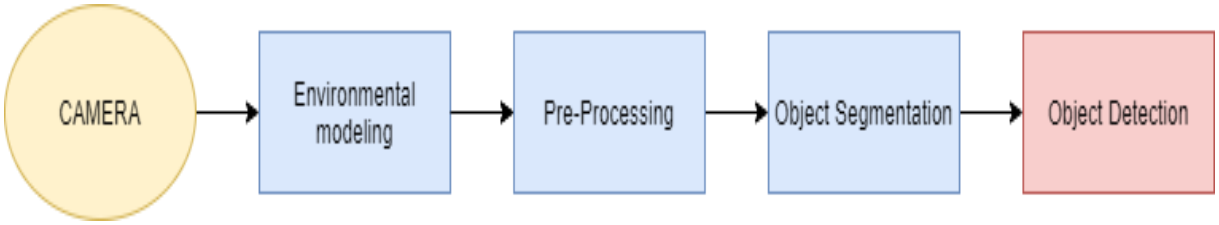

Figure 15. Block Diagram

4. Software Requirements Specification

4.1 Introduction

4.1.1 Purpose

The purpose of this document is to provide all software requirements for 3D Object Detection for Self-Driving cars research project. This documentation includes the project requirements and Software Requirements Specification for 3D Object Detection task.

4.1.2 Scope of Project

In this research project, we aim to focus on Perception and Prediction improvements which are primary concepts for object detection for autonomous vehicles.

We are expecting to identify the following objects during runtime:

- Moving vehicles(Cars, Trucks)
- Parked vehicles(Cars, Trucks)
- Pedestrians
- Cyclist

4.1.3 Glossary

Table 1 Glossary of SRS

Term	Definition
AV	Autonomous Vehicles
LiDAR	Light Detection and Ranging
DL	Deep Learning
CNN	Convolutional Neural Network
PCD	Point Cloud Data
AWS	Amazon Web Services
GCP	Google Cloud Platform
VPN	VIrtual Private Network
GPU	Graphical Processing Unit
EDA	Exploratory Data Analysis

4.1.4 References

- IEEE Std 830™-1998(R2009) Recommended Practice for Software Requirements Specifications.

4.1.5 Overview of Document

The documents are divided into 3 main chapters:

- First chapter explains LiDAR, Camera, point cloud data and perception in the introduction level.
- Second chapter focused on Product perspective and function, User characteristic and operating environment of the system.
- Third chapter explains technical behavior of the hardware interface, functional requirements and software requirements.

4.2 Overall Description

This chapter consists of the main aspect of Perception and Prediction of the Autonomous Vehicles.

4.2.1 Product Perspective

4.2.1.1 System Interfaces

This project will be modelling and preprocessed under Jupyter Notebook interface by using DL frameworks and processing libraries. Spark Computing Cluster lab or other cloud platforms such as AWS, GCP etc. will be used as a development environment.

4.2.1.2 User Interfaces

Since the project isn't planning to put up an interface to the end-user, There will not be specific interfaces for users.

4.2.1.3 Hardware Interfaces

For the training and preprocessed LiDAR and camera data in the server, cutting-edge GPU, high capacity RAM and VRAM would be better.

4.2.1.4 Software Interfaces

Python will be used as a main programming language in the development phase of the project.

4.2.1.5 Communication Interfaces

Since, we plan as a first option to work in Spark Computing Cluster lab, an active internet connection for both developers and server is needed. Communication will be provided via FortiClient VPN network application between our computers and cluster server. If we decide to work in online cloud platforms, we also need internet connection during the development.

4.2.1.6 Memory Interfaces

One of the biggest constraints are computational power and memory constraint for training large-scale LiDAR and camera data if all of the data will be using.

Constrain	Description
Performance	Performance is a very important requirement we need when processing data. Because we use a large dataset to increase accuracy.
Accuracy	Accuracy is another important requirement. When processing this data, the objects must be recognized correctly.

4.2.2 Product Functions

4.2.2.1 Dynamic Object Detection

The ego-vehicle system detects dynamic objects(on-road) while mobilizing on the highway and streets. Distinguishes type of the object such as 4 wheelers(cars, trucks) or 2 wheelers(motorbikes, bicycles) or pedestrians.

4.2.2.2 Static Object Detection

The ego-vehicle system detects static objects while mobilizing on the highway and streets. Distinguishes type of the object such as Road and Lane markings(on-road), Curbs(off-trafic), Traffic lights(off-road) and Road signs(off-road).

4.2.3 User Characteristics

If a normal user wants to test its own sequence image or video data which contain dynamic or static traffic objects, can use Jupyter Notebook interface with necessary libraries and imports. Users can test unique data with our trained data file which doesn’t require to train again whole large-scale data.

4.2.4 Constraints

The system is designed to detect objects reliably and faster which is a vital process for self-driving cars to make a decision for the next movement. If the system isn't able to identify objects in milliseconds, it should give a message about the system identification condition.

4.2.5 Assumptions and Dependencies

It is assumed that the objects which occur frequently in the dataset, should be clearly identified in the environment of the ego vehicle.

4.3 Requirements Specification

4.3.1 External Interface Requirements

4.3.1.1 User interfaces

There are no user interfaces since there will be no application or website to be developed.

4.3.1.2 Hardware interfaces

Cutting-edge GPU is required if it will be developed in a local lab. Preferably, 5800+ cuda core GPU would be fine. 8000+ better. Since the dataset is more than 100 GB, preferably, 64 GB or more amount of RAM is needed. 1 TB + of mass storage device is needed.

4.3.1.3 Software interfaces

Kaggle API will be used to download Lyft dataset.

4.3.1.4 Communications interfaces

There are no external communication interfaces needed.

4.3.2 Functional Requirements

The system will start immediately when the sequence of images or video data flows as input. System will be able to identify new unseen collected ego-vehicle data.

4.3.2.1 Get Semantic Map

LiDAR data will be used for transformations. Since LiDAR is capable of scanning in a 3D environment, it is provide to create geometric maps. For the ego vehicle, LiDAR data will collect such information like road surface, environment feature etc.

4.3.2.2 Exploratory Data Analysis

EDA will be provided to understand the main aspects and characteristics by visualizing the dataset. It is also provided to detect outliers. So that, we will be able to ignore outlier and noise data.

4.3.2.3 Build model

Build model will be a function which will contain an algorithmic and architecture approach to the LiDAR and Camera captions after observing semantic map and EDA results. Prediction and training phases will be developed here.

4.3.2.4 Evaluation

The training and prediction result will be evaluated in this section. We will evaluate the outcome of the different gradient based optimizer algorithms by visualizing in 2D graph.

4.3.3 Performance Requirements

Performance Requirements	Description
Response Time	The system will work with real-life data. Response time to identifying shouldn't exceed 300 msec.
Error Handling	When the system fails to predict in a given response time, it should give an alert message to the interface.
Workload	The system should be able to handle identifying given input while the ego-vehicle operates on the way.

4.3.4 Design constraints

The following specifications should be provided for development environment:

- A CPU preferably 8 core or more.
- Continuous power supply unit.
- GPU which has 8000+ cuda core.
- 64GB memory or more

The project will be developed under Anaconda distribution which provides the necessary development packages using Ubuntu server/Cloud services.

4.3.5 Software system attributes

Term	Definition
Functionality	System designed to identify the objects.
Reliability	System should operate 99,9% in runtime.
Robustness	System should successfully identify traffic objects which will be trained with training data.
Portability	System should work Ubuntu 16.04 or later OS version.
Efficiency	Source codes of the system should be clear and operate efficiently.

4.3.6 Other Requirements

No other requirements needed.

5. Software Design Description

5.1 Introduction

Purpose of this Software Design Document is to provide a brief overview of the function of the Object Detection for Self-Driving Cars and the purpose of the development, its scope, and references to the development context.

5.1.1 Purpose

Self-driving cars will be indispensable for people to transport. However, these cars need to get to know their surroundings before they go on a ride. Object detection plays a big role here. Our aim is to investigate how we can improve object detection and use it more effectively by using CNN and similar algorithms.

5.1.2 Project Scope

In this project, we aim to develop an object detection algorithm in various objects such as pedestrians, bicycles, parked vehicles and moving vehicles in the environment.

5.1.3 Glossary

Table 2 Glossary of SDD

Term	Definition
Self-Driving Car	A full autonomous car that doesn't need a driver assistance.
SDD	The report that provides an overview of system's design.
DL	Deep Learning
CNN	Convolutional Neural Network, that applied to analyze visual imagery with the way of Deep Neural Network.
Deep Neural Network	A neural network built to simulate the activity of the human brain. A Deep Neural Network is a neural network with more than two layers.

AWS	Amazon Web Services
LIDAR	Light Detection and Ranging
GCP	Google Cloud Platform
VPN	Virtual Private Network
GPU	Graphic Processing Unit
BEV	Bird Eye View

5.1.4 Overview

The documents are divided into 3 main chapters:

- First chapter is an introduction about the project. It contains purpose, scope of the project, Terminologies and their definitions and motivation.
- Second chapter is architecture design. It includes description of the problem, technologies used and Architecture design.
- Third chapter includes a list of the resources that we used in this project.

5.1.5 Motivation

Technology is improving and people's workforce is getting smaller and smaller. The wheel is one of the greatest inventions in human history. We used this wheel to get us from one point to another. However, we have started to produce self-driving cars in recent years. Self-driving vehicles, which are not more common in the world, need improvement. We want to have a hand in this technology movement by helping to improve object detection in self-driving vehicles, which are the transport vehicles of the future. This is the source of our motivation.

5.2 Architecture design

5.2.1 Problem Description

Object detection is very vital in self-driving cars. A small mistake in object detection can have big consequences. That is why self-driving cars need to detect the objects around them accurately. We use a large dataset for this, but processing this dataset is very problematic and slow in terms of hardware. To speed up this slow process, it is necessary to use an optimized algorithm.

5.2.2 Technologies Used

- Sensors such as Monocular LIDAR, Solid State LIDAR, Monocular Camera and Stereo Camera.
- Dataset as Lyft Dataset.
- Kaggle API to download Lyft Dataset.
- Jupyter Notebook by using DL frameworks and processing libraries.

5.2.3 Architecture Design

5.2.3.1 Activity Diagram

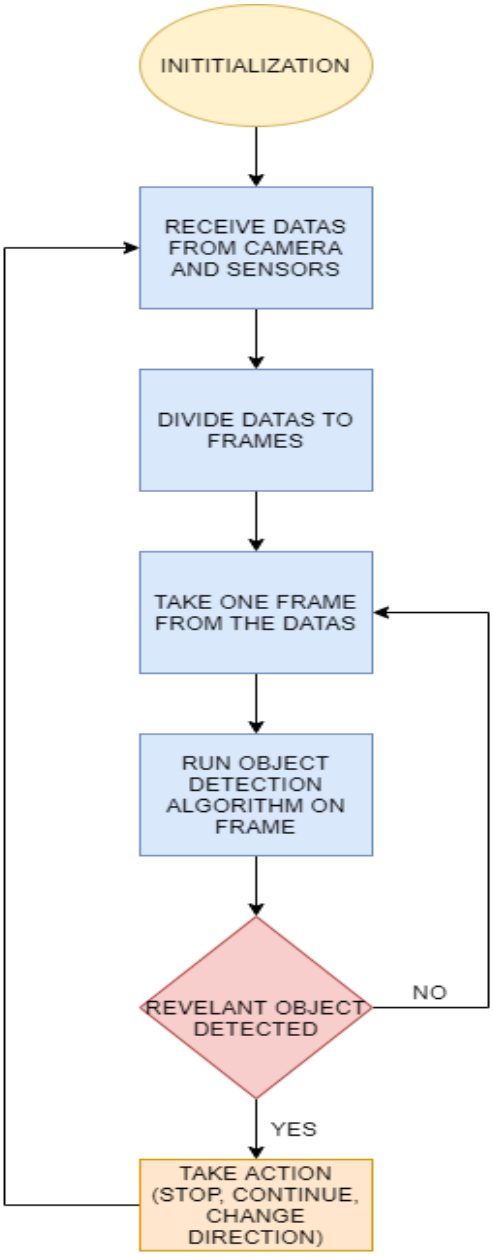

Figure 16. Activity Diagram

In the activity diagram above, the object recognition process in autonomous vehicles is shown in a simple way. The important step here is the function marked in red. Correct perception of the relevant object is one of the main objectives of this project. Examples of relevant objects are pedestrians, bicycles, parked vehicles and moving vehicles. After receiving the data from the Lyft dataset, we divide the data we receive from the cameras and sensors into frames, take one of these frames and perform an object detection algorithm on that frame. If our object detection algorithm detects any relevant object, we come to the part where the tool will choose what to do with that relevant object. If our object detection algorithm does not detect any relevant objects, we continue with the other frames.

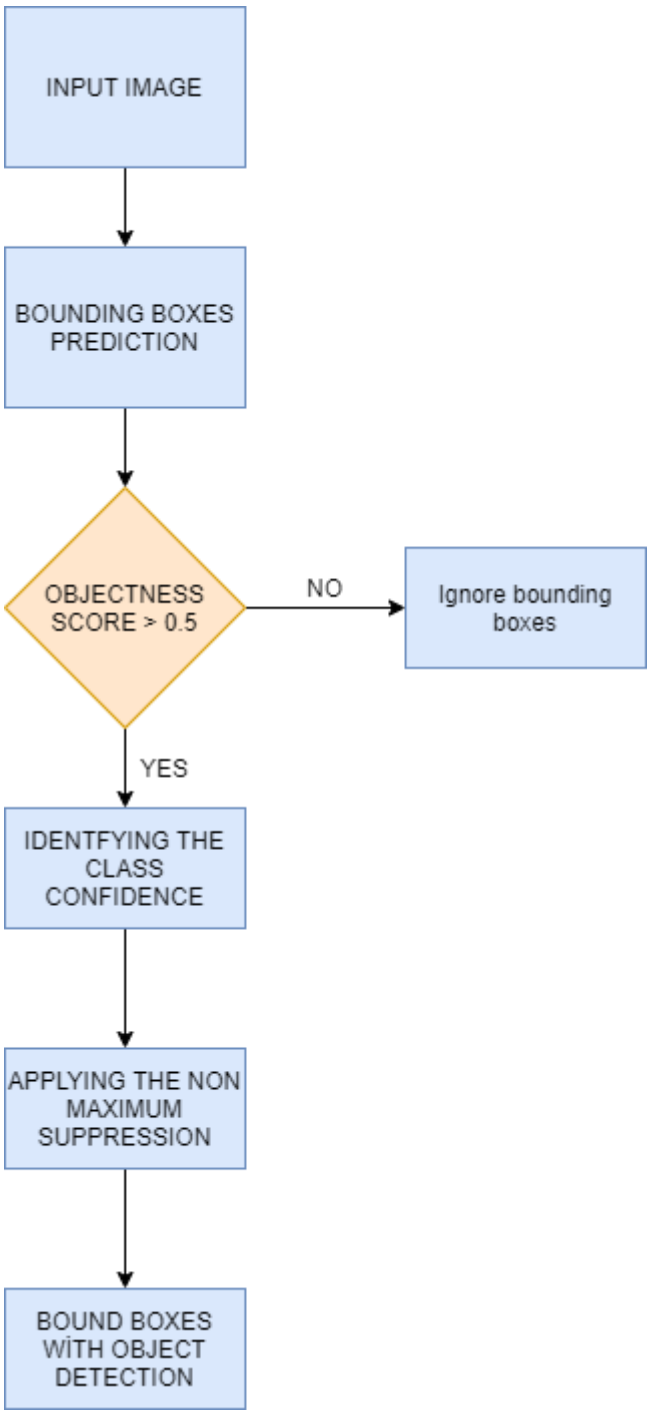

Figure 17. Activity Diagram

Inside the object detection part in Figure 1, we apply the flowchart steps in Figure 2 above. In this flowchart after we take the input image we make a bound box prediction on the picture if the Objectness score is less than 0.5 we ignore the bounding boxes. If objectness score is more than 0.5 then we identify the class confidence then we apply the non-maximum suppression and finally we bound the boxes with object detection. So our relevant objects like pedestrians, bicycles, parking vehicles and moving vehicles will be bound boxed.

5.2.3.2 Dataflow Diagram

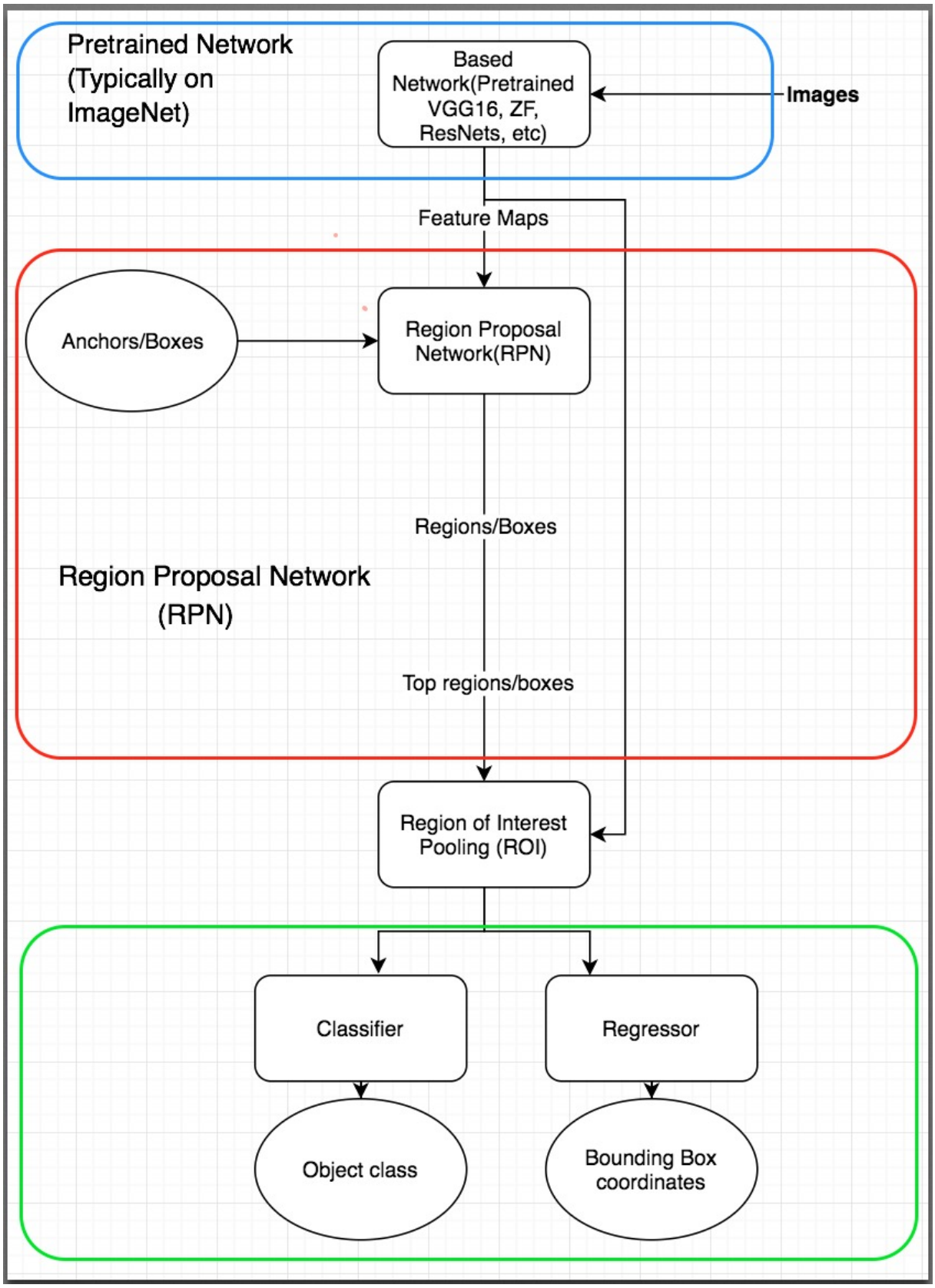

Figure 18. Dataflow diagram

The above class diagram shows the main architecture of the CNN with Faster R-CNN backend. Blue box represents CNN based networks like ResNets, VGG16 etc. Red box represent adjustment of the regions and boxes into objects. Green box represents decision making which generates class labels and bounding box coordinates. The blue part of the diagram shows how we compute the image Feature Map with Based Networks (Typically on ImageNet) like Pretrained VGG16, ResNets, ZF, etc. Then we move to the redbox which includes the Region Proposal Network which we can call RPN. RPN is a separate network that runs on Feature Maps which came from the Pretrained Network. Region Proposal Network ranks anchors (anchors are fixed-size rectangles or boxes) and proposes those region boxes that most likely contain objects. After that to create bound boxes we fed these Regions of Interests into a layer in ROI Pooling. ROI Pooling layer generates vectors with fixed size from Feature Maps. Then we start to create bound boxes on the green side of the diagram.

5.2.3.3 Architecture Design

5.2.3.3.1 Route Trajectory

Summary: Raw image data contain objects in its environment along the way.

Actor: None.

Precondition: Image data should be flow sequentially in order to properly work of the detection system.

Exception:

- Some of the captured images could be noisy due to bad weather conditions.
- The lack of light in the environment. The photos to be taken from the camera can only show objects illuminated by the light from the car headlight and cannot detect other objects.

Post Conditions: Camera and the LIDAR need to work well.

Priority: Medium

5.2.3.3.2 Identifying Objects

Summary: The system will try to recognize objects.

Actor: None.

Precondition: The trained image csv data must be located in the working directory.

Exception:

- Lack of any relevant objects on the road, such as pedestrians, bicycles, parked vehicles and moving vehicles.

- Being in the blind spot of relevant objects can hinder the detection of those objects.

Post Conditions: Camera and the LIDAR need to work well.

Priority: High

5.2.4 Dataset

The Lyft data set contains a total of 85GB of data. These data were split between test and training sets. This dataset gave us data from test vehicles taken with cameras and a 3-Dimensional point cloud. These data were taken from 10 vehicles. Each of these vehicles had seven fixed cameras and one LIDAR detector on the roof. There were also 2 more small detectors under each vehicle's headlights. These vehicles traveled around Palo Alto, California, taking photos, creating this 85GB data set. Within this dataset, there are 55,000 human-tagged 3D relevant objects. Examples of these relevant objects are pedestrians, bicycles, parked vehicles and moving vehicles.

5.2.4.1 File Formats

Picture formats are .jpeg files that are a commonly used method of lossy compression for digital images. LIDAR formats are .bin files that contain a series of spatial x spatial x time cubes for each pulse with some platform ephemeris data mixed in.

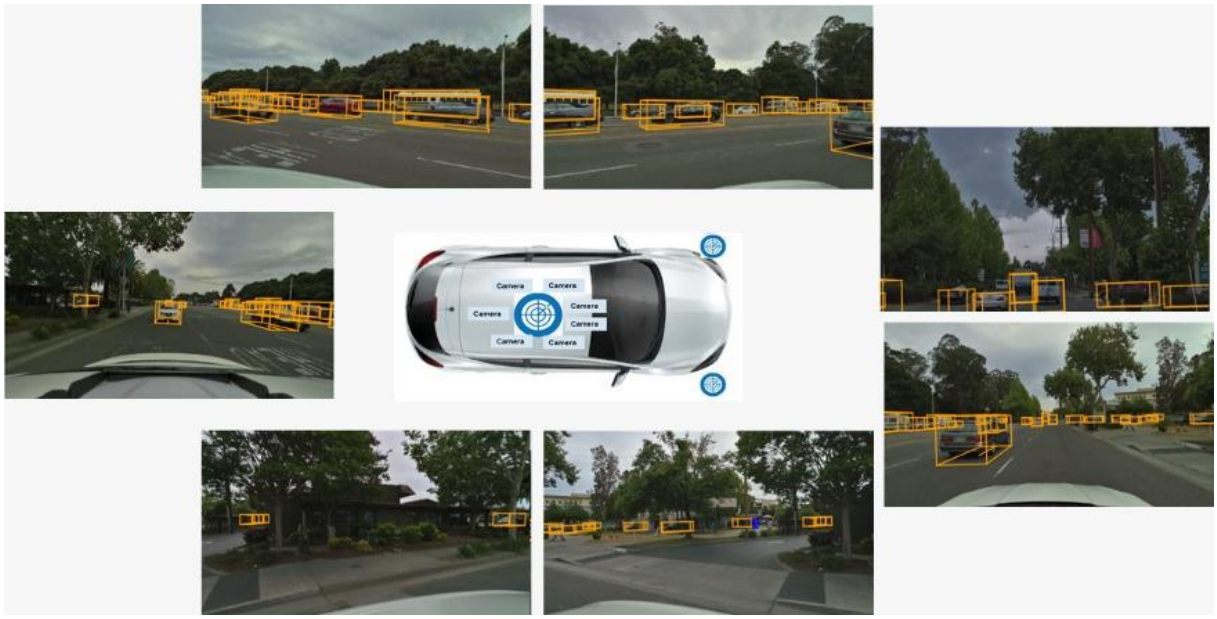

Figure 19. Rendered image of environment

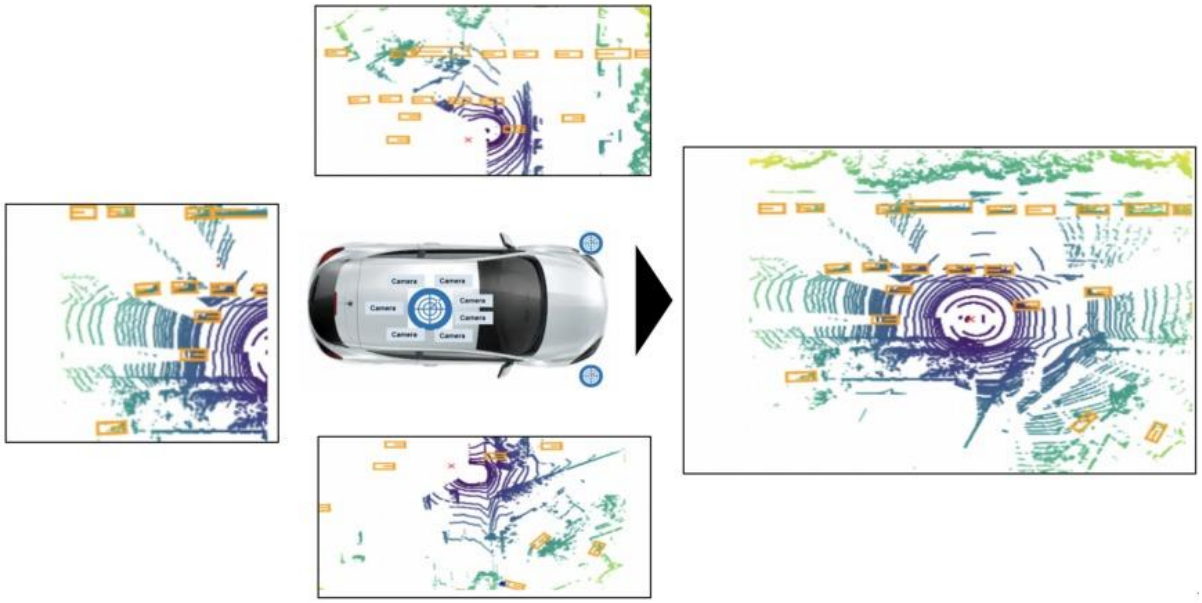

Figure 20. Lidar points that transformed into BEV.

6. Test Plan and Result

6.1 Introduction

6.1.1 Version

Version No	Description of Changes	Date
1.0	First Version	Apr 16, 2021

6.1.2 Overview

This document is the test plan for the 3D Object Detection for Self-Driving Cars project. In this document, we will test the functions and methods whether they are working properly or not.

Also, we will test the object detection score if it's giving a high precision result or not.

6.1.3 Scope

This document encapsulates the test plan of the 3D Object Detection for Self-Driving Cars.

6.1.4 Terminology

Acronym	Definition
EDA	Exploratory Data Analysis
DT	Data Transformation
CNN	Convolutional Neural Network
LIDAR	Light Detection and Ranging
U-NET	Convolutional Network Architecture for fast and precise segmentation of images
Adam	Adaptive Moment Estimation
RAdam	Rectified Adam
IoU	Intersection Over Union
mAP	Mean Average Precision
BEV	Bird Eye View

6.2 Features To Be Tested

6.2.1 Exploratory Data Analysis

We analyzed LiDAR cameras scanned objects x, y, and z coordinates via seaborn and matplotlib libraries and determined the scanning range of the LiDAR during the routing.

We observed that most of the objects are automobiles. Seaborn's distplot visualizing method also prove within a particular length, width, and height range.

6.2.2 Data Transformation

Affine transformation methods have been used in both 2D and 3D. In 2D data, we have used to bounding boxing target objects during the routing.

In 3D, affine transformations are one of the stages for defining certain positions of the objects. 3D point cloud provides accurate reflection and depth.

6.2.3 Model Building

U-Net and Voxelnet architectures with fully connected CNN used for semantic segmentation. The objects are predicted pixel by pixel and feeding to BEV. Morphological operations applied to fit boxes. Fast R-CNN mainly used for 2D modeling.

6.2.4 Evaluation

Thresholding and morphological operations are the steps for evaluating with IoU and mAP. Monitoring loss function in each epoch with different optimization algorithms like Adam, RAdam will be considered.

6.3 Features Not To Be Tested

Since it is a research project, we will not go over the improving loading/reversing time and time complexity of the learning architectures, because of the limited deadline.

6.4 Test Results

Feature	Description	Date of Run	Result	Explanation
2.1	Exploratory Data Analysis	27.05.2021	Pass	EDA has done on 2D/3D Data
2.2	Data Transformation	27.05.2021	Pass	Data Transformation has implemented on 2D/3D Data
2.3	Model Building	04.06.2021	Pass	Model has been built
2.4	Evaluation	04.06.2021	Pass	Evaluation results passed into model

6.5 Item Pass Fail Criteria

6.5.1 Exit Criteria

Criteria	Status
100% of the source code should be executed.	Met
%95 of the thresholding will be passed into semantic segmenting.	Met
%95 of the IoU will pass into objects.	Met

7. Conclusions

Before we started this project, we were both interested in data analysis, artificial intelligence, object detection machine learning and deep learning. After we started this project, we conducted studies and research on autonomous vehicles. We shared these studies with each other online and created the Literature Review file. Then we determined our requirements and prepared the relevant SRS file. However, our requirements generally include high capacity hardware. Because we had to pass 85 gigabytes of data through train and test stages. After we finished writing the SRS file, we started to write the SDD file. Since we did not use a certain user interface or create a certain product when starting to prepare the SDD file, we created an SDD file suitable for the research project.

In this project that we will be coding, we want to enable autonomous vehicles to perceive relevant objects such as parked cars, moving cars, pedestrians, and cyclists in 3D while they are cruising. Of course, we know that vehicles' object recognition must be very fast to react. Because in traffic, a pedestrian can suddenly get in front of the vehicle, and the autonomous vehicle must detect the pedestrian and quickly decide on actions such as slowing down, stopping and changing lanes. Therefore, while creating 3-dimensional object detection, we also need to use a good algorithm and increase the object detection speed without sacrificing accuracy. Since autonomous vehicles are newer, we think that projects like this have a vital value for autonomous vehicles.

To be successful in this project, we first need to acquire the hardware we need to have. In this way, we can test the effectiveness of the codes we write and measure their efficiency. Of course, we need to code with the most optimal algorithm to increase efficiency, accuracy and speed.

8. Acknowledgement

We are grateful to our advisors Roya Choupani and Faris Serdar Taşel for their advice and support in this project. We also thank Lyft for making this dataset public and available.

9. References

[1] A Survey on 3D Object Detection Methods for Autonomous Driving Applications: https://ieeexplore.ieee.org/document/8621614

[2] Toyota Research Institute - Advanced Development(About)

https://www.tri-ad.global/about/index

[3] Argo AI: We're building self-driving technology you can trust.

https://www.argo.ai/purpose/

[4] Sensing requirements for an automated vehicle for highway and rural environments

https://repository.tudelft.nl/islandora/object/uuid:2ae44ea2-e5e9-455c-8481-8284f8494e4e

[5] Deep SCNN-based Real-time Object Detection for Self-driving Vehicles Using LiDAR Temporal Data

https://arxiv.org/abs/1912.07906

[6] Geiger, A., Lenz, P., Urtasun, R. (2012) "Are we ready for autonomous driving? the KITTI vision benchmark suite," Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), 128.

https://journals.sagepub.com/doi/full/10.1177/0278364913491297

[7] Multi-column Deep Neural Networks for Image Classification

https://arxiv.org/abs/1202.2745

[8] SVGA-Net: Sparse Voxel-Graph Attention Network for 3D Object Detection from Point Clouds

https://arxiv.org/abs/2006.04043

[9] https://medium.com/@aidlp20180201/building-your-own-object-detector-model-using-fasterrcnn-in-dlp-3e1bc5ada64b

[10] http://wrap.warwick.ac.uk/114314/1/WRAP-survey-3D-object-detection-methods-autonomous-driving-applications-Arnold-2019.pdf

[11] Arnold, E., Al-Jarrah, O. Y., Dianati, M., Fallah, S., Oxtoby, D., & Mouzakitis, A. (2019). A survey on 3d object detection methods for autonomous driving applications. IEEE Transactions on Intelligent Transportation Systems, 20(10), 3782-3795.

[12] https://pythonawesome.com/stereo-r-cnn-based-3d-object-detection-for-autonomous-driving/

[13] https://www.kaggle.com/c/3d-object-detection-for-autonomous-vehicles/

[14] https://towardsdatascience.com/3d-object-detection-for-autonomous-vehicles-b5f480e40856