A

Major Project REPORT

On

"OBJECT DETECTION IN AUTONOMOUS VEHICLE BASED ON DEEP LEARNING"

BACHELOR'S OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

Submitted by

(Batch-13)

Mohammad Athar Shareef (207Y1A0507)

UNDER THE GUIDANCE OF

Mrs.G.Anitha

(Asst.Professor)

COMPUTER SCIENCE AND ENGINEERING

MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(UGC - AUTONOMOUS)

(Affiliated to JNTU-H, Approved by AICTE New Delhi and Accredited by NBA & NAAC With 'A' Grade)

MARCH 2024

CERTIFICATE

This is to certify that the project report titled **"Object Detection in Autonomous Vehicle Based on Deep Learning"** is being submitted by **MOHAMMAD ATHAR SHAREEF (207Y1A0507)** in IV B. Tech II Semester **Computer Science & Engineering** is a record bonafide work carried out by him. The results embodied in this report have not been submitted to any other University for the award of any degree.

Internal Guide HOD

Principal External Examiner

DECLARATION

I hereby declare that the Project Report entitled, "**Object Detection in Autonomous Vehicle Based on Deep Learning**" submitted for the B. Tech degree is entirely my work and all ideas and references have been duly acknowledged. It does not contain any work for the award of any other degree.

Date:

MOHAMMAD ATHAR SHAREEF
(207Y1A0507)

MARRI LAXMAN REDDY
INSTITUTE OF TECHNOLOGY AND MANAGEMENT
(AN AUTONOMOUS INSTITUTION)
(Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad)
Accredited by NBA and NAAC with 'A' Grade & Recognized Under Section2(f) & 12(B)of the UGC act,1956

ACKNOWLEDGEMENT

I am happy to express my deep sense of gratitude to the principal of the college Dr. K. Venkateshwara Reddy, Principal, Marri Laxman Reddy Institute of Technology & Management, for having provided me with adequate facilities to pursue my project.

I would like to thank Mr. Abdul Basith Khateeb, Assoc. Professor and Head, Department of Computer Science and Engineering, Marri Laxman Reddy Institute of Technology & Management, for having provided the freedom to use all the facilities available in the department, especially the laboratories and library.

I am grateful to my project guide Mrs.G.Anitha, Asst.Professor, Department of Computer Science and Engineering, Marri Laxman Reddy Institute of Technology & Management, for her extensive patience and guidance throughout my project work.

I sincerely thank my seniors and all the teaching and non-teaching staff of the Department of Computer Science and Engineering for their timely suggestions, healthy criticism, and motivation during the course of this work.

I would also like to thank my classmates for always being there whenever I needed help or moral support with great respect and obedience, I thank my parents and brother who were the backbone behind my deeds.

Finally, I express my immense gratitude with pleasure to the other individuals who have either directly or indirectly contributed to my need at right time for the development and success of this work.

ABSTRACT

This Project "Object detection in autonomous vehicles using deep learning" Autonomous vehicles are those without a driver that offer better security and comfort to passengers. The safety of their propulsion and their ability to avoid causing traffic accidents are the two most crucial factors with regard to autonomous cars. It involves the system and device functional safety of the vehicle. Object detection is a critical component in enabling autonomous vehicles to perceive and interact with their environment. In recent years, deep learning-based approaches have shown significant improvements in object detection accuracy and speed. We propose a method for object detection in autonomous vehicles using YOLO (You Only Look Once) and MobileNet SSD (Single Shot Multibox Detector). We first train the MobileNet SSD as a feature extractor, and then use YOLO to detect objects in real-time. Our approach achieves high accuracy and fast inference times, making it suitable for real-time applications in autonomous vehicles. We evaluate our method on the COCO dataset, and show that it outperforms state-of-the-art methods in terms of detection accuracy and speed. Our results demonstrate the effectiveness of using YOLO and MobileNet SSD for object detection in autonomous vehicles. One of the biggest issues for autonomous driving is that objects are wrongly classified. The data gathered by the vehicle's different sensors is collected and then interpreted by the vehicle's system. But with just a few pixels of difference in an image produced by a camera system, a vehicle might incorrectly perceive a stop sign as something more innocuous, like a speed limit sign. If the system similarly mistook a pedestrian for a lamp post, then it would not anticipate that it might move. To overcome it, although it might take huge resources, we train the model on huge data sets To improve accuracy of classification So that our system can identify the difference between different objects and signs much accurately. The detection and tracking of objects around an autonomous vehicle is essential to operate safely. This paper presents an algorithm to detect, classify, and track objects. All objects are

classified as moving or stationary as well as by type (e.g. vehicle, pedestrian, or other). The proposed approach uses state of the art deep-learning network YOLO (You Only Look Once) combined with data from a laser scanner to detect and classify the objects and estimate the position of objects around the car. The Oriented FAST and Rotated BRIEF (ORB) feature descriptor is used to match the same object from one image frame to another. This information fused with measurements from a coupled GPS/INS using an Extended Kalman Filter. The resultant solution aids in the localization of the car itself and the objects within its environment so that it can safely navigate the roads autonomously. The algorithm has been developed and tested using the dataset collected by Oxford Robotcar. The Robotcar is equipped with cameras, LiDAR, GPS and INS collected data traversing a route through the crowded urban environment of central Oxford. Object detection is a computer vision task that has become an integral part of many consumer applications today such as surveillance and security systems, mobile text recognition, and diagnosing diseases from MRI/CT scans. Object detection is also one of the critical components to support autonomous driving. Autonomous vehicles rely on the perception of their surroundings to ensure safe and robust driving performance. This perception system uses object detection algorithms to accurately determine objects such as pedestrians, vehicles, traffic signs, and barriers in the vehicle's vicinity. Deep learning-based object detectors play a vital role in finding and localizing these objects in real-time. This article discusses the state-of-the-art in object detectors and open challenges for their integration into autonomous vehicles. Applications for object detection and scene perception in autonomous vehicles. Unlike existing review papers, we examine the theory underlying self-driving vehicles from deep learning perspective and current implementations, followed by their critical evaluations. Deep learning is one potential solution for object detection and scene perception problems, which can enable algorithm-driven and data-driven cars. In this article, we aim to bridge the gap between deep learning and self-driving cars through a comprehensive survey. We begin with an introduction to self-driving cars, deep learning, and computer vision followed by an overview of artificial general intelligence. Then, we classify existing powerful deep learning libraries and their role and significance in the growth of deep learning. Finally, we discuss several techniques that address the image perception issues in real-time driving, and critically evaluate recent implementations and tests conducted on self-driving cars. The findings and practices at various stages are summarized to correlate prevalent and futuristic.

CONTENTS

LIST OF FIGURES

List of abbreviations in alphabetical order.

Acronym	Explanation	Acronym	Explanation
2D	Two Dimensional	IVS	Intelligent Vision Systems
3D	Three Dimensional	IVSS	Intelligent Visual Surveillance System
5G	Fifth Generation Mobile Networks	KITTI	Karlsruhe Institute of Technology Dataset
ADAS	Advanced Driver Assistance Systems	L1 & L2, L0-L5	NA (regularization techniques), levels of automation defined by SAE
AE	Auto Encoder	LiDAR	Light Detection and Ranging
AGI	Artificial General Intelligence	LSTM	Long Short-Term Memory
AI	Artificial Intelligence	MAP	Map Attention Decision
ANN	Artificial Neural Network	MCP	McCulloch & Pitts neural network
AR	Average Recall	MIT-AVT	Massachusetts Institute of Technology-Advanced Vehicle Technology
ATRI	American Transport Research Institute	ML	Machine Learning
AV	Autonomous Vehicles	MLP	Multilayer Perceptron
AWS	Amazon Web Services	NHTSA	National Highway Traffic Safety Administration
BP	Back Propagation	NN	Neural Network
CNN	Convolutional Neural Network	OBU	On-board Unit
CoreML	Core Machine Learning	openCV	Open Source Computer Vision
CPU	Central Processing Unit	PB	Petabytes
CUDA	Compute Unified Device Architecture	PD-DBM	Partially Directed DBM
cuDNN	CUDA Deep Neural Network Library	RADAR	Radio Detection And Ranging
CV	Computer Vision	RBM	Restricted Boltzmann Machines
DAE	Denoising Autoencoder	rCDN	Reverse Content Distribution Network
DBN	Deep Belief Network	R–CNN	Region-CNN
DBM	Deep Boltzmann Machine	ResNet	Residual Network
DIP	Digital Image Processing	RGB	Red, Green, & Blue
DIVS	Deep Intelligent Visual Surveillance	RNN	Recurrent Neural Network
DL	Deep Learning	RoI	Region of Interest
DLib	Deep Library	RPN	Region Proposal Network
DLR	Docklands Light Railway	S3C	Spike & Slab Sparse Coding

Acronym	Explanation	Acronym	Explanation
DMV	Department of Motor Vehicles	SAE	Society of Automotive Engineers, Stacked Autoencoder
DNN	Deep Neural Networks	SciPy	Scientific Python
DPM	Deformable Parts Model	SSD	Singleshot Multibox Detection
DRL	Deep Reinforcement Learning	STR	Smart Transportation Robots
DSRC	Dedicated Short-Range Communication	SVM	Support Vector Machines
DVS	Deep Vision Systems	SSVM	Structured Support Vector Machines
FCNN	Fully Connected Neural Network	TB	Terabytes
FPS	Frames Per Second	TL	Transfer Learning
GAN	Generative Adversarial Network	TLI	Traffic Light Information
GLAD	GoogLeNet for Autonomous Driving	TPU	Tensor Processing Unit
GM	General Motors	UAV	Unmanned Aerial Vehicle
GPS	Global Positioning System	V2V	Vehicle to Vehicle
GPU	Graphical Processing Unit	V2I	Vehicle to Infrastructure
HD	High Definition	V2X	Vehicle to Everything
HoG/HOG	Histograms of Oriented Gradients	VANET	Vehicular ad hoc Network
HRPN	Hyper Region Proposal Network	VaaS	Vehicle-as-a-Service
iOS	iPhone Operating System	VAE	Variational Autoencoder
ICT	Information and Communication Technology	VGG	Visual Geometry Group
IoT	Internet of Things	VOC	Visual Object Classes
IoU	Intersection over Union	VRS	Visual Recognition Systems
IoV	Internet of Vehicles	XOR	Exclusive-or
ITS	Intelligent Transportation Systems	YOLOv3	You Look Only Once version 3

CHAPTER 1

INTRODUCTION

1.1 Introduction

This Project "Object detection in autonomous vehicles using deep learning". Autonomous vehicles (AVs) are a cutting-edge innovation that several international auto manufacturers have been working to create. It empowers the driver to enable an autopilot that allows the car to navigate by itself without much or any human assistance. Not withstanding the potential of this technology, it is crucial to concentrate on the safety features of AVs to prevent any mishaps or object collisions when navigating. The people around it, the people within the car, and the car itself. Whether done automatically or manually, object detection is a crucial component of AV navigation. While navigating, the AV needs to identify nearby things so that it can forecast object movements to avoid collisions and accidents, especially with pedestrians and other cars. A real-time object detection algorithm is the You Only Look Once (YOLO) algorithm.YOLOv2 will then the Divide an image into areas, and forecast each region's bounding boxes and probability. In order to provide an informed prediction, the estimated probabilities then weigh these bounding boxes when testing the entire image. in the global context. YOLOv2 is more effective in object identification than the Single Shot Detection (SSD) technique. Simple things like pedestrians and automobiles can be detected by YOLOv2 and SSD with a precision of 100 percent.. In the The primary deep learning techniques for object detection are classified into one-stage detection algorithms and two-stages detection algorithms. One-stage detection techniques such as YOLO and SSD instantly transform the detection problem into a unified regression problem. The one-stage approaches are quicker than two-stage methods because of the peculiarities of the structure. Deep learning combined with computer vision has the potential to produce solid, reasonably priced solutions for the autonomous driving sector. The main aim of Applications for autonomous vehicles include the detection, tracking, and recognition of both static and moving objects, including pedestrians, motorbikes, cars, and other vehicles. One of the difficulties in the realm of computer vision is object recognition. In recent years, autonomous/self-driving cars have drawn much interest as a topic of research for both academia and industry. For a car to be a truly autonomous, it must make sense of the environment through which it is driving. The autonomous car must be able to both localize itself in an environment and identify and keep track of objects (moving and stationary). The car gets information about the environment using exteroceptive sensors such LiDAR, cameras, inertial sensors, and GPS. The information from these sensors can be used together and fused to localize the car and track objects in its environment, allowing it to travel successfully from one point to another. The process of path planning and autonomous vehicle guidance depends on three things: localization, mapping, and tracking objects. Localization is the process of identifying the position of the autonomous vehicle in the environment. Mapping includes being able to make the sense of the environment. Tracking of moving objects involves being able to identify the moving objects and track them during navigation. The processes of localization and mapping have been explored through the use of simultaneous localization and mapping (SLAM), which was initially proposed by Leonard and Durant-Whyte. [4] SLAM enables autonomous vehicles to simultaneously build the map of the unknown environment and localize itself in the unknown environment. The development of SLAM has been significant in the development of autonomous robots. Most of the research in SLAM assumes the environment to be static and considers the movement of objects as noise. The detection and tracking of moving objects (DATMO) is one of the most challenging issues and is important for safety and essential for avoiding collisions.

Evolution of self-driving cars

Brief history of self-driving vehicles

The concept of self-driving cars has been around for almost 80 years, first reported in 1939 World's Fair in New York by general motor's (GM) Futurama [51]. Contemporary developments in communication networks and wireless connectivity, arrival of accurate and robust sensors that continuously miniaturize in size and cost, coupled with AI have been the cornerstone for autonomous driving systems [52]. Embedded in these self-driving systems are human-machine interface applications, network enabled controls, multiple-sensor data fusion, 3D drive scene analysis, and software-defined signal processing to transport materials, payloads, goods, and people [53]. The AI based self-driving machines must be able to navigate successfully in all situations at all times [54]. The accuracy of autonomous navigation depends significantly on attaining precise localization, unobtrusive data collection, fused data-set generation, and uninterrupted high-level communication with other vehicles and surrounding smart infrastructure [55]. In the longer run, self-driving technology can also expected to be extended to tractor-trailers, cargo trucks, mining trucks, and buses [56]. In the last decade, Carnegie Mellon University and the Defense Advanced Research Projects Agency (DARPA) self-driving cars have contributed to autonomous vehicles advancement [57].

Tesla Motors implemented an autopilot technology to its electric vehicles where the cameras and sensors predicted collisions with up to 76% accuracy leading to collision prevention rate of over 90%. Google, Tesla Motors, General Motors, Waymo, Uber, autonomy and other automobile companies envision a future with autonomous vehicles in approximately 15–20 years' time [57].

Several infrastructure upgrades such as automated highway system, robotic vehicle cruising management systems, 6G cell-free mobile communication systems with real-time video processing and near-zero latency are parallel research areas that would contribute to realizing full-fledged autonomous vehicles kick-starting a greener future through autonomous electric vehicles [57].

Self-driving cars, also known as autonomous vehicles, driver-less cars, smart transportation robots (STR) or robocars are one of the most speculated scientific invention with a potential to change the world [54]. The recent and broader implications of self-driving cars incorporate integration with novel infrastructure, smart cities, urban planning with provisions for advanced cyber-security, privacy, and insurance [58]. It is worth mentioning that while the self-driving cars have gained intense attention in the last decade, driver-less transportation has been in existence for over a decade [59]. Trains are a prominent example of widespread use of self-driving technology [60]. Some of such train examples include the..SkyTrain in Vancouver, Canada [61].Docklands Light Railway (DLR) in London, United Kingdom [59].Yurikamome in Tokyo, Japan [60].,London Heathrow airport's ultra-pods [59].

These autonomous rail systems transport thousands of passengers on a daily basis. Authors in Ref. [59] note that the majority of passengers commuting through self-driving trains were not worried about using those trains. However, the aforementioned trains and autonomous pods operate on enclosed tracks, isolated from the public roads, and bypass the need to interact with other vehicles or pedestrians [61]. In contrast, self-driving cars are set to encounter various users, thereby resulting in complex interactions and the possibility of collision [12]. Whether people will be as accepting of self-driving cars as they appear to be of existing autonomous transport is an active area of research [61].

Advantages of self-driving cars

The advances in wireless networking, software-defined networking, and information and communication technology (ICT) have found applications in intelligent transportation systems (ITS) to reduce collisions, reduce pollution, ameliorate mobility issues, provide newer ways of public transportation, and share resources, materials and space [8]. According to studies, there are 1.3 million deaths every year due to drunk, drugged, distracted and drowsy driving, which can potentially be saved with the help of autonomous AI systems by eliminating some of the human follies [62]. The following advantages motivate the current research in self-driving cars:For users, the advantages may be reduced stress, faster commutes, reduced travel times, enhanced user productivity, optimum fuel consumption, reduced carbon emissions. These cars can be programmed to drive defensively, stay clear of blind spots, and follow speed limits [63].For

Governments, self-driving cars would assist in traffic enforcement, enhance roadway capacity, reduce road casualties and the number of on-road driving related accidents, and lead to better observance of speed limits [12].

Self-driving cars are envisioned to eliminate drunk driving issues, eliminate issues related to distracted driving, texting and other cell phone use, less braking and accelerating, and less gridlock on highways [64]. Reduced accidents are expected to be beneficial for children and the elderly, encouraging people to feel comfortable and amiable towards self-driving cars [65].•

Autonomous electric vehicles would introduce a greener mode of transport, leading to less greenhouse and noise pollution, along with increased mobility for the elderly and disabled people [65]. In current driving landscape, cars are parked for a long time. With self-driving cars, parking lots can be converted to parks and other green infrastructure [65].

Self-driving cars would be equipped to improve scheduling and routing, and provide best routes to improve travel times, while also lowering the travel cost [5].

Although self-driving cars would reduce or even eliminate car ownership, they would expand shared access, keep transportation personalized, efficient and reliable [65].

Probable disadvantages and drawbacks of self-driving cars

Cars are one of the most widespread and readily available modes of transportation and while technology has developed safer cars, driving is still a dangerous activity [66]. Self-driving cars formulate a scenario where a few lines of source codes, coupled with AI get to decide the life of a human beings [67]. Some disadvantages of self-driving cars are outlined as follows:

The foremost catastrophic consequence of self-driving cars would be elimination of jobs in the transportation industry.

Although the role of AI in our society is consistently evolving, an AI system making critical decisions need to respect societal values and conform to social norms to gain acceptance [68]. The acceptance of self-driving technology at philosophical, ethical and technological levels is a fundamental research problem in psychology and cognitive science. It is argued that in case autonomous vehicles and AI systems malfunction, a person would not die or suffer injuries if they themselves were in control of the system [66].

Driving at intersections without traffic lights, malfunctioning traffic lights, uncontrolled intersections, busy intersections, regions with humans in close proximity are a challenge for self-driving cars [69]. As self-driving cars use global positioning system (GPS) for localization, they are deemed unsuitable to drive in non-mapped areas [70].

The scope of car's connectivity, the car being online at all times, makes it susceptible to hacking. The safety and convenience offered by self-driving cars might compromise privacy of passengers as their moves will be tracked and logged [22].2.4.

Communication between different entities in self-driving cars Two well-known collisions mentioned below, involving vehicles operating with a certain degree of autonomous technology emphasize the benefits of vehicle to vehicle (V2V), vehicle to infrastructure (V2I), and vehicle to everything (V2X) communication in self-driving cars:A fatal accident involving a semi tractor-trailer that turned in front of a Tesla car operating on its autopilot program in Florida, caused due to sensors failing to detect a turning vehicle [32].

A fatal Uber crash in Arizona [32].Investigation and analysis of these accidents indicates that these accidents could have been avoided if the involved vehicles were communicating with each other [32]. The V2V and V2X broadcast a vehicle's current location to nearby traffic, alert traffic to upcoming maneuvers, traffic jams, accidents and road constructions.

A crash, three cars ahead, is too far to be detected by sensors but can easily be communicated over longer distances using V2V [71]. The V2I technology consists of sending traffic light information (TLI) to self-driving cars' acceleration and braking systems, which can assist in plannning routes based on the frequency of traffic light changes [72]. The V2V can provide 360-degree road-situation awareness to enhance safety.

Although the user of these techniques requires all vehicles to operate on a standard mode of communication such as dedicated short range communication (DSRC) to relay critical information, a formal policy to mandate DSRC in vehicles is still under development [73]. Scientists, researchers, and experts have historically viewed the lack of computational infrastructure as a major bottleneck that prevents achieving reliable V2V, V2I, and V2X communication.

Deploying roadside infrastructure partly mitigated the problem by providing uninterrupted wireless coverage while also improving handover and coverage [72]. Authors in Ref. [73] proposed vehicle-as-a-service (VaaS) by leveraging vehicular cloud, vehicular fog and internet of vehicles (IoV) to provide the necessary real-time computing platform that will define self-driving vehicular environment [74,75].

Levels of automation: semi automated, automated and self-driving cars

Autonomy in self-driving cars is based on progression from human centered autonomy to complete autonomy where all the driving tasks are governed and controlled by the vehicle's AI system, and human interaction is summoned only when necessary [11]. To investigate the capabilities of the present AI systems [76], Fig. 1 briefly outlines the six levels of vehicular automation defined by the society of automotive engineers (SAE). The Fig. 1 also highlights the differences between fully automated (level-5) and partially automated vehicles (levels-3 & 4). The functioning of involved AI systems at an accuracy less than 100% is dangerous to human life [10]. Even if the AI algorithms function exceptionally well on the engineering side, their performance remains disputable from an ethical point of view [77]. As an example, if the driver is intoxicated or incapable to drive, is it safe for AI system to demand human takeover? Offering technology to humans can make them trust the system all the time, rendering them lazy and not able to re-engage themselves [78]. A test on level-2 cars revealed that drivers fell asleep when the vehicle was set on auto-pilot, questioning the ways in which humans interact with AI [46]. The prevailing deep learning architectures for scene perception and object detection in autonomous vehicles are depicted in Fig. 1.2.

1.2 Purpose and scope

The purpose of the thesis is to develop the algorithm to detect, classify, and track the objects for autonomous vehicles. The proposed algorithm is developed using information from a camera, laser scanners, and GPS/INS. The images from the camera is used to detect and classify the object. The laser scans give the distance and direction of the detected objects. The GPS/INS give the information about the state of the autonomous vehicle. This information is all fused in an Extended Kalman Filter to track the objects and help the autonomous vehicle navigate safely.

1.3 Assumptions

In this thesis, any object detected, person, or any vehicles is represented by a point object.It is assumed that representing them as a point object would be sufficient to demonstrate the algorithm developed in this thesis. In this thesis, the detected objects are not modelled with the motion model but are assumed to have simple random walk model with expected variance.

1.4 Significance

The goal of this thesis is to identify the objects around the autonomous vehicle to aid in safe navigation of the autonomous vehicle. Information of the objects around the autonomous vehicle can be a great asset in avoiding collision. The detection, classification, and tracking of the objects will also help the autonomous vehicle to localize itself better in the environment

1.5 Limitations

The algorithm developed in the thesis is only tested using one dataset. The algorithm was not implemented in a real system. Due to resource constraints, the object detection and classification on the images were preprocessed separately to decrease computational runtime over multiple simulations when the navigation of the autonomous car was performed

DATASET

The algorithm developed on this paper and the results of the algorithm are all based on the datasets provided by Oxford University Robotcar. The Robotcar, an autonomous capable Nissan LEAF, traversed a route through central Oxford, UK twice a week on average over the period of May 2014 to December 2015 to collect data of different driving environments with camera, LiDAR, GPS and INS. The Robotcar was equipped with the following sensors and position of each sensor can be seen in figure 1

Cameras:

o 1 x Point Grey Bumblebee XB3 (BBX3-13S2C-38) trinocular stereo camera, 1280×960×3, 16Hz, 1/3" Sony

ICX445 CCD, global shutter, 3.8mm lens, 66° HFoV, 12/24cm baseline

o 3 x Point Grey Grasshopper2 (GS2-FW-14S5C-C) monocular camera, 1024×1024, 11.1Hz, 2/3" Sony

ICX285 CCD, global shutter, 2.67mm fisheye lens (Sunex DSL315B-650-F2.3), 180° HFoV

• LIDAR:

o 2 x SICK LMS-151 2D LIDAR, 270° FoV, 50Hz, 50m range, 0.5° resolution

o 1 x SICK LD-MRS 3D LIDAR, 85° HFoV, 3.2° VFoV, 4 planes, 12.5Hz, 50m range, 0.125° resolution

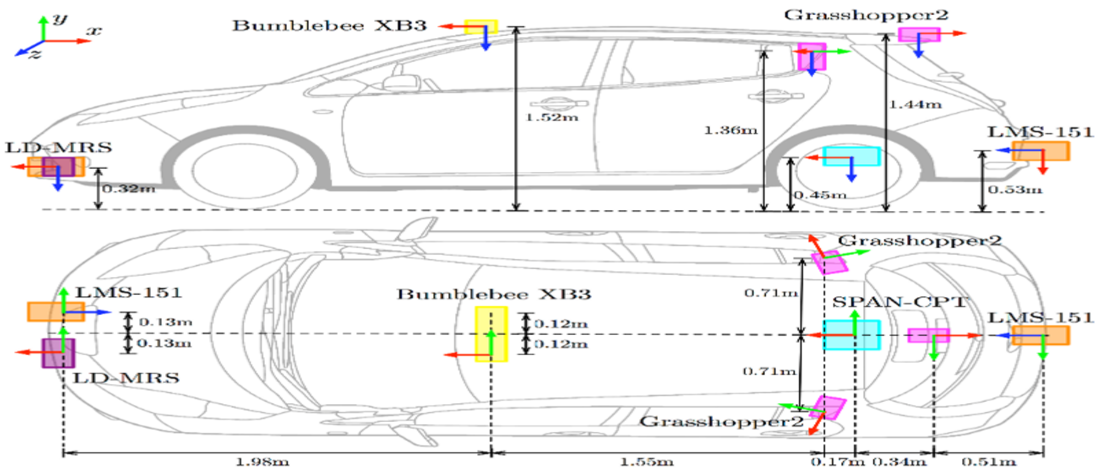

Fig 1.1 Robotcar with sensors location

• GPS/INS

o 1 x NovAtel SPAN-CPT ALIGN inertial and GPS navigation system, 6 axis, 50Hz, GPS/GLONASS, dual

antenna

As seen in Figure 1.1, sensors are in different places with the different coordinate frames. The information from each sensor must be transformed to the same coordinate system before processing and fusing in EKF. This work uses the Bumblebee XB3 sensor's location and orientation for the vehicle frame. Information in all other frames is transformed to this frame using homogenous transformation matrix

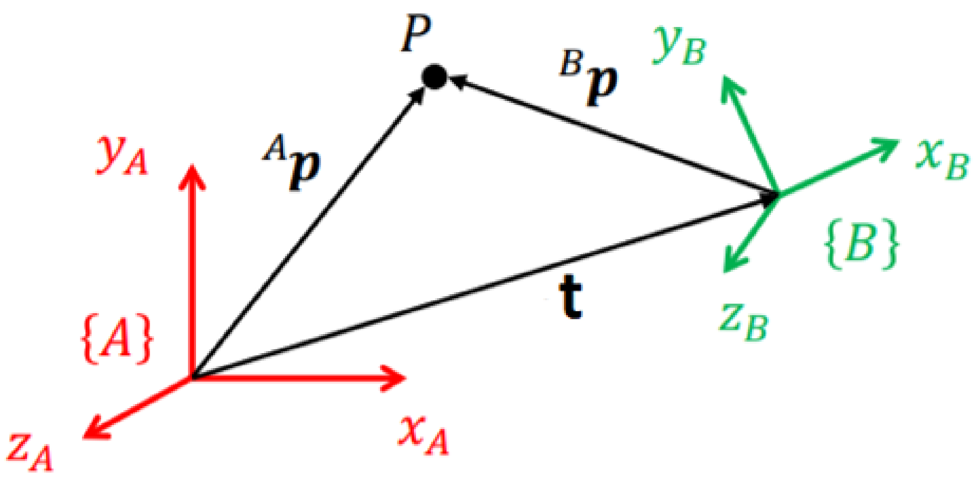

Fig 1.2 Transformation of point from one frame to another

The point P from frame B is to be transformed to the frame A as shown in the Figure 1.2. The origin offrame B is translated by vector t= [X, Y, Z] from frame A. Each x, y and z axis of the frame B is rotated by , , αβψ angles compared with respect to respective axis in the frame A. The point P in frame B is translated to frame A by equation 1

$$ {}^{A}\mathbf{p} = \mathbf{H}\,{}^{B}\mathbf{p} \tag{1} $$

H is the homogenous transformation matrix given by equation 2

$$ \mathbf{H} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix} \tag{2} $$

The rotation matrix, R is calculated using equation 3.

$$ \mathbf{R} = \begin{bmatrix} \cos\psi\cos\beta & \cos\psi\sin\beta\sin\alpha - \cos\alpha\cos\psi & \cos\psi\sin\beta\cos\alpha + \sin\alpha\sin\psi \\ \cos\beta\sin\psi & \cos\alpha\cos\psi + \sin\psi\sin\beta\sin\alpha & \cos\alpha\sin\psi\sin\beta - \cos\psi\sin\alpha \\ -\sin\beta & \cos\beta\sin\alpha & \cos\beta\cos\alpha \end{bmatrix} \tag{3} $$

The translation vector t is given by equation 4.

$$ \mathbf{t} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{4} $$

1.6SYSTEM OVERVIEW

In this paper, the information from the image from Bumblebee XB3 camera, SICK LD-MRS laser scanner and NovAtel SPAN CPT INS and GPS system is used. The details about these sensors is discussed in previous section. Figure 1.3 shows the block diagram of the system that is used to study the problem of detecting objects around the RobotCar and track them.

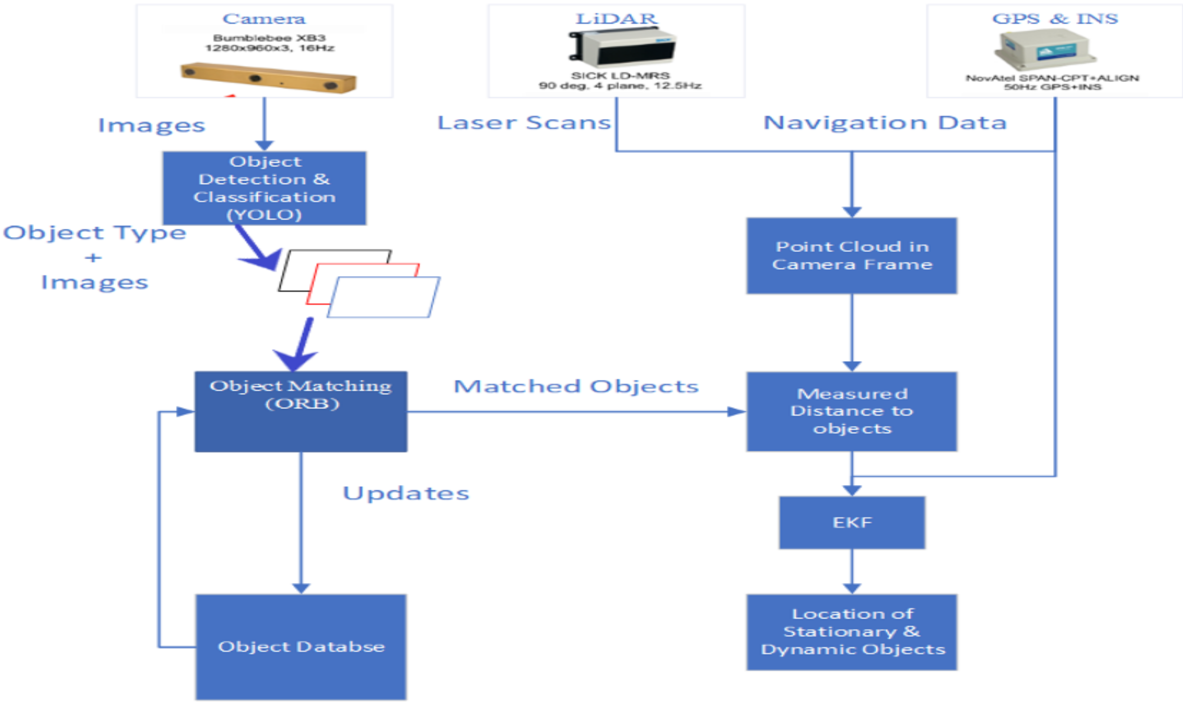

Fig 1.3 Block Diagram of Proposed System

The image obtained from the camera is used to detect the object and classify them. The objects in the image are detected using Convolutional Neural Networks (CNN). Once objects are detected they are stored in a database. Each detected object is matched with objects in the database to find association or added as a new object in the database. To manage the size of the database, objects that are no longer detected are deleted from the database.With object detection in the image complete, the data from the laser scanner is projected onto the image. This allows for the measurement of the distance and direction of detected objects from the RobotCar. This information along with the state of RobotCar is combined with an Extended Kalman Filter (EKF). Both the state of the objects and the RobotCar are updated using EKF allowing for a combined localization and tracking of objects in the environment.

OBJECT DETECTION

To perform object detection, this work uses datasets that provide information of the environment through the LiDAR and camera. Using the information from these sensors, objects are detected, classified, and the distance and direction of the object relative to the Robotcar is measured. Usually object detection is achieved using a combination of feature-based modelling and appearance-based modelling. The image has more information that can be used to identify objects as compared to laser scan and allows both features based and appearance-based modelling. In this paper, the image is primarily used to detect objects and classify them, and the LiDAR is used to measure the location of the object relative to the vehicle.

The laser scan combined with the pose of the vehicle is used to create the 3D point cloud of the environment. This is then projected onto the image. The transformation of 3D coordinates obtained from LiDAR scan to 2D image pixels is done using pinhole camera model. The LiDAR (x, y, z) coordinates are transformed into pixels (u, v) in the image using equation 5.

$$u = \frac{f}{z} x + u_0$$
$$v = \frac{f}{z} y + v_0 \tag{5}$$

where f is the focal length of the camera and $(u_0,v_0)$is the optical center of the camera.

The Figure 1.4 shows sample image and the laser scans projected onto the image.

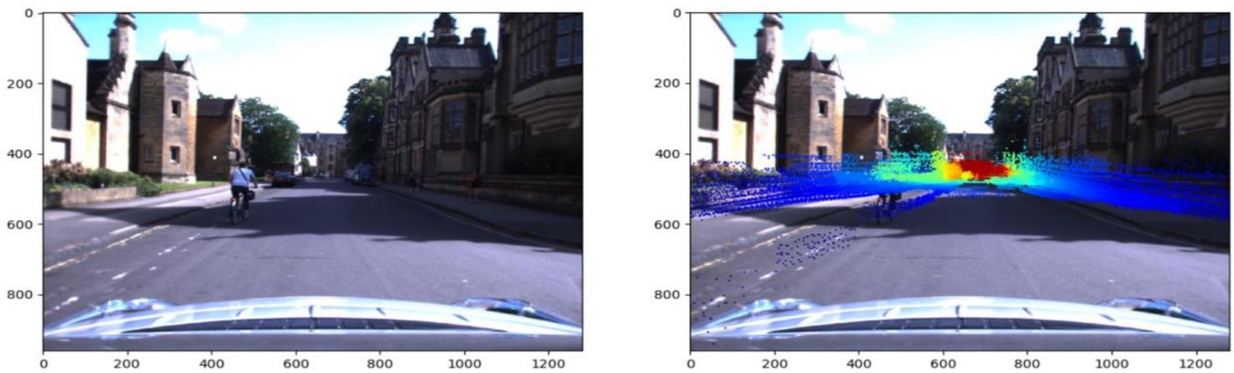

Fig 1.4 A Sample Image and Laser scan projected onto the image

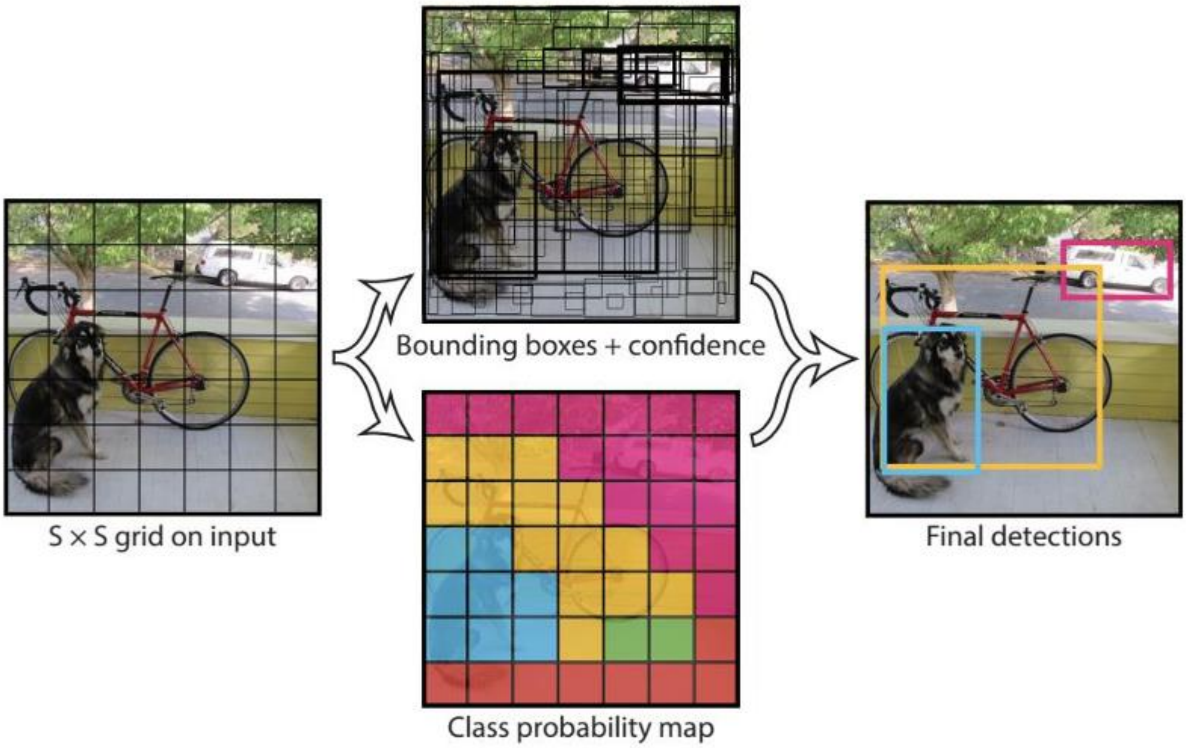

Fig 1.5 Illustration of YOLO algorithm

A single image will have multiple objects that need to be identified. Detecting multiple objects in an image is a challenging task; however, using current methods, it is achievable with accuracy in real time. Object detection is one of the most researched topics in the area of Computer vision. The object detection usually starts with extracting features from the input image using different algorithm such as Haar [8] or HOG [9].These features are fed to the classifier to identify the object. With advancement in deep learning, there are many convolutional neural network (CNN) architectures that have outperformed the object detection method using a feature extractor. These CNNs attempt to imitate the working of human neurons. The CNN learns the feature in object during training process and can generalize the feature for the object and detect the object.

The R-CNN [10], Faster R-CNN [11], YOLO algorithms are all based on CNN and perform very well at multiple object detection. This work uses YOLO algorithm for the detection and classification of objects. The advantage of using YOLO is that it is orders of magnitude faster (45 frames per seconds) than other algorithms while maintaining a high accuracy. Other algorithms detect objects using region-proposal techniques or sliding-windows method and iterate over the image for many times. YOLO sees the entire image and only once hence the name. This makes YOLO faster compared to iterative methods. The entire image is fed through the CNN and detects multiple objects simultaneously. YOLO works by taking an image and splitting it into an SxS grid.

For each grid cell, it predicts bounding boxes and confidence for those boxes and class probability. Objects are located within the image where the bounding boxes have a class probability above a threshold value. For every object detected YOLO classifies the object, gives the confidence and the bounding box for the object. Figure 5 illustrates the working principle of the YOLO algorithm [1]. Each object detected has a bounding box, which localizes the object in the image. This information is helpful in finding the distance of the objects from the RobotCar. Once the objects are detected and classified and located in the image, the projected laser scans to the image can be isolated so that only laser bouncing back from the objects detected remain.

Figure 1.6 shows the object detected by the YOLO algorithm and corresponding laser scan projected on the detected objects.

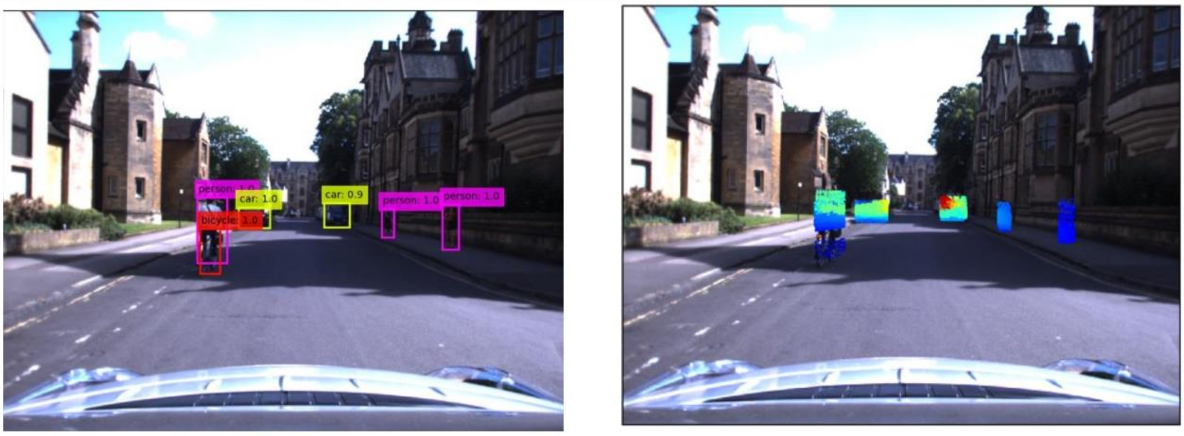

Fig 1.6 Object Detection by YOLO(left) and laser Projection on detected objects(right)

The detected objects are represented as the point object for the purpose of the tracking. The centroid of the laser scan projected onto the object is taken as distance of the object from the RobotCar. Figure 1.7 shows the representation of the detected objects in figure 1.7 as point object and in vehicle frame

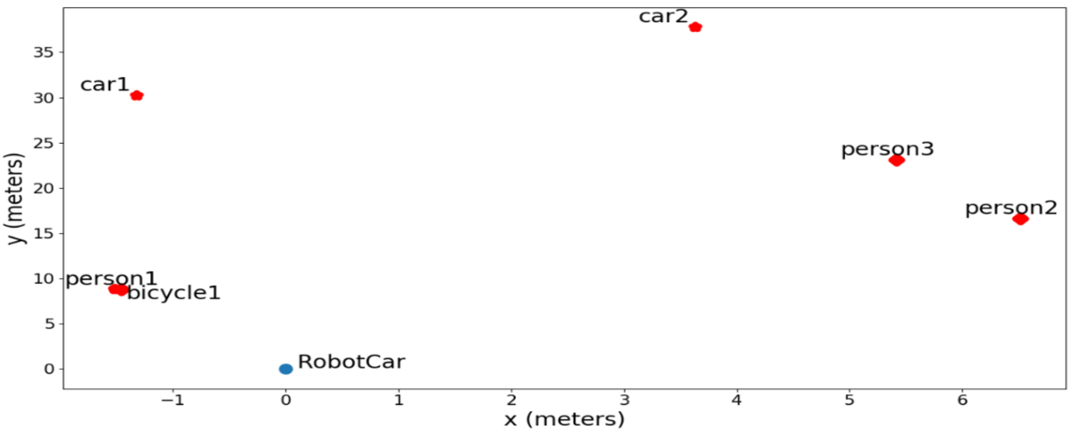

Fig 1.7 2D Map of Object detected in Vehicle Frame

Object Matching

Once the object is detected in one image, for successful tracking the object in one image must be associated with another image. This association is achieved by matching the features of the object in one image and matching that features with the object detected in the next image. In this work, Oriented FAST and Rotated BRIEF (ORB) is used to match features of objects and to tracking match objects from one image to another. ORB is a very fast binary descriptor computationally-efficient replacement to SIFT [12] with similar matching performance. Using ORB keypoint from the training image and the query image are extracted, and they are matched. The match for each key point is found by using the Brute-Force Matcher function provided by OpenCV library [13]. The BruteForce Matcher takes the descriptor of one feature in a training image and is matched with all features in query image using a distance calculation, and the one with the minimum distance is returned as matching. There is still a risk of a false match. The approach that best reduces this risk is to find second nearest neighbor and perform ratio of closest to second closest as described in [12]. All the matches with distance ratio between closest and second closest greater than 0.75 are discarded.

Fig 1.8 The Performance of the ORB on matching same object and different object

Figure 1.8 shows matching the key points from object detected in one image to objects detected in next frame. It can be observed that number of key points matched between same object in one image frame to another frame is very high compared to the match with a different object.If the object detected in one frame cannot be matched with an object detected in the next image, the object is removed from the database. If an object is detected again after deletion, it is treated as new object and tracking resumes.

EXTENDED KALMAN FILTER

The information from the different sensors is fused together using Extended Kalman Filter (EKF) for the tracking of the objects around the autonomous car. Based on the information from INS, LiDAR and camera the following states are chosen for the filter.

$$\mathbf{x_t} = (\mathbf{R}, \mathbf{O})^T = \left(\underbrace{X, Y, \psi, v_x, v_y}_{RobotCar's\ state}, \underbrace{o_{1,x}, o_{1,y}}_{object\,1}, \ldots\ldots\ldots, \underbrace{o_{n.x}, o_{n,y}}_{object\,n} \right)^T$$

At any given time, state vector in the EKF consists of the state of the Robotcar, R, and the position of the, n, number of objects detected. The state of the Robotcar include the position of the vehicle (X and Y), yaw angle of the vehicle (□), the velocity (vx and vy). The velocity of the RobotCar is provided in the local frame. The local frame and the body frame are shown in figure 9.

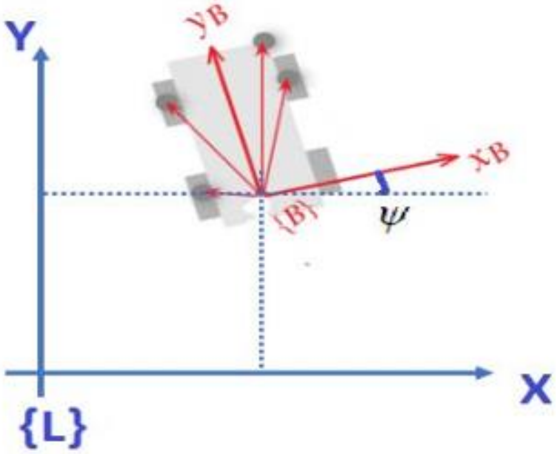

Fig 1.9 The Local Frame and the RobotCar body frame

The navigation of the car is done in UTM coordinates, Northing and Easting. The area navigated by the car is a small area around center of Oxford; hence, using local navigation coordinates gives accurate pose estimation of the RobotCar and the objects detected. The entire area falls under the 30U UTM zone, which is used for navigation.

The motion model for the Robotcar is modeled using equation 6.

$$\begin{aligned} X_t &= X_{t-1} + v_{x_{t-1}} \Delta t \\ Y_t &= Y_{t-1} + v_{y_{t-1}} \Delta t \end{aligned} \tag{6}$$

The velocity and the yaw are not defined by any motion model equation as the INS equipped in the RobotCar does not measures the acceleration or the yaw rate of the vehicle. These are updated using the motion noise. The rolling average for all the measurements provided by INS and GPS is calculated. After rolling average is calculated the variance and co-variance of these measurements is calculated as the motion noise, Q, given by:

$$\mathbf{Q} = \begin{bmatrix} \sigma^2_{X_{rm}} & \sigma^2_{X_{rm}Y_{rm}} & \sigma^2_{X_{rm}\psi_{rm}} & \sigma^2_{X_{rm}v_{xrm}} & \sigma^2_{X_{rm}v_{yrm}} \\ \sigma^2_{Y_{rm}X_{rm}} & \sigma^2_{Y_{rm}} & \sigma^2_{Y_{rm}\psi_{rm}} & \sigma^2_{Y_{rm}v_{xrm}} & \sigma^2_{Y_{rm}v_{yrm}} \\ \sigma^2_{\psi_{rm}X_{rm}} & \sigma^2_{\psi_{rm}Y_{rm}} & \sigma^2_{\psi_{rm}} & \sigma^2_{\psi_{rm}v_{xrm}} & \sigma^2_{\psi_{rm}v_{yrm}} \\ \sigma^2_{v_{xrm}X_{rm}} & \sigma^2_{v_{xrm}Y_{rm}} & \sigma^2_{v_{xrm}\psi_{rm}} & \sigma^2_{v_{xrm}} & \sigma^2_{v_{xrm}v_{yrm}} \\ \sigma^2_{v_x y X_{rm}} & \sigma^2_{v_{yrm}Y_{rm}} & \sigma^2_{v_{yrm}\psi_{rm}} & \sigma^2_{v_{yrm}v_{xrm}} & \sigma^2_{v_{yrm}} \end{bmatrix} \tag{7}$$

The covariance matrix is setup in following ways:

$$\mathbf{P} = \begin{bmatrix} \mathbf{P_{RR}} & \mathbf{P_{RO}} \\ \mathbf{P_{OR}} & \mathbf{P_{OO}} \end{bmatrix} = \begin{bmatrix} \mathbf{P_{RR}} & \mathbf{P_{RO_1}} & \cdots & \mathbf{P_{RO_n}} \\ \mathbf{P_{O_1R}} & \mathbf{P_{O_1O_1}} & \cdots & \mathbf{P_{O_1O_n}} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{P_{O_nR}} & \mathbf{P_{O_nO_1}} & \cdots & \mathbf{P_{O_nO_n}} \end{bmatrix} \tag{8}$$

where RR

P is the covariance matrix for the RobotCar, RO

P is the covariance matrix for the RobotCar and the landmarks, OR

P is the covariance matrix between the objects and the Robotcar, and OO

P is the covariance matrix between the objects being

tracked.

The motion modelling for the state of RobotCar didn't have any non-linearity so computing Jacobian is not required for predict step of EKF. The predict step in EKF is done by equation 9.

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & 0 & \ddots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{aligned} \hat{\mathbf{x}}_t &= \mathbf{F}\mathbf{x}_{\mathbf{t-1}} \\ \hat{\mathbf{P}}_{\mathbf{t}} &= \mathbf{F}\mathbf{P}_{\mathbf{t-1}}\mathbf{F}^{\mathrm{T}} + \mathbf{Q} \end{aligned} \tag{9}$$

After the predict step, the EKF is updated based on the measurements obtained. First the state, t x ,and state covariance matrix, Pt, in EKF is updated based upon the measurements obtained from the GPS/INS and then based upon the measurements of the detected objects obtained from the LiDAR. The measurement for the RobotCar state, t z , at time t is directly obtained from the GPS and INS

$$\mathbf{z_t} = \begin{bmatrix} X_t & Y_{\mathrm{t}} & v_{x_t} & v_{y_t} & \psi_t \end{bmatrix}^{\mathrm{T}}$$

Once the object has been detected it is fused in the EKF. The measurements for the object position are given in rectangular coordinates in vehicle frame. The second stage of EKF update is done with the measurements for landmarks. For all objects detected, the position of jth object in vehicle frame from laser scanner at time t is given by,

$$\mathbf{z^j_t} = \begin{bmatrix} o_{x,t} \\ o_{y,t} \end{bmatrix}$$

If the jth detected object is new object, then it is added to the state vector, xt, in navigation frame using equation 10.

$$\begin{bmatrix} O_{j,x} \\ O_{j,y} \end{bmatrix} = \begin{bmatrix} X_t \\ Y_t \end{bmatrix} + \begin{bmatrix} \cos(\psi_t) & -\sin(\psi_t) \\ \sin(\psi_t) & \cos(\psi_t) \end{bmatrix} \mathbf{z}^{\mathbf{j}}_{\mathbf{t}} \tag{10}$$

When an object is first detected, it is considered to be dynamic and is modelled with high process noise Qo. The motion noise Q is updated for each new object by,

$$\mathbf{Q}_{k+1} = \begin{bmatrix} \mathbf{Q}_k & \mathbf{0} \\ \mathbf{0} & \mathbf{Q}_o \end{bmatrix}$$

If the jth detected object is already in the state vector, then the estimated distance of the object from the Robotcar in navigation frame is calculated by equation 11

$$\mathbf{\Delta} = \begin{bmatrix} \Delta_x \\ \Delta_y \end{bmatrix} = \begin{bmatrix} \hat{O}_{j,x} - \hat{X} \\ \hat{O}_{j,y} - \hat{Y} \end{bmatrix} \tag{11}$$

$$\hat{\mathbf{z}}_t^j = \begin{bmatrix} \cos(\hat{\psi}) & \sin(\hat{\psi}) \\ -\sin(\hat{\psi}) & \cos(\hat{\psi}) \end{bmatrix} \mathbf{\Delta} = \mathbf{h}(\hat{\mathbf{x}}_t) \tag{12}$$

The difference in the measurements from the sensor and estimated measurements, t j y is given by equation 13.

$$\mathbf{y}_j^t = \mathbf{z}_t^j - \hat{\mathbf{z}}_t^j \tag{13}$$

The Jacobin for the jth object is given by equation 14.

$$\mathbf{H}_t^j = \frac{\partial h(\hat{\mathbf{x}}_t)}{\partial \hat{\mathbf{x}}_t} = \begin{bmatrix} -\cos(\hat{\psi}) & \sin(\hat{\psi}) & -\sin(\hat{\psi})\Delta_x - \cos(\hat{\psi})\Delta_y & \cos(\hat{\psi}) & \sin(\hat{\psi}) \\ -\sin(\hat{\psi}) & -\cos(\hat{\psi}) & \cos(\hat{\psi})\Delta_x - \sin(\hat{\psi})\Delta_y & \sin(\hat{\psi}) & \cos(\hat{\psi}) \end{bmatrix} \tag{14}$$

This Jacobian is only for one object with respect to estimated position and yaw angle of the RobotCar. The state vector for the EKF consists of the states for all objects and the vehicle, so this Jacobian has to be mapped to higher dimension which is done by matrix M, as in equation 15.

$$\mathbf{M}_j = \begin{bmatrix} 1 & 0 & 0 & 0 & 0\,0 & \cdots & 0\,0 & 0\,0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & 0\,0 & \cdots & 0\,0 & 0\,0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & 0\,0 & \cdots & 0\,0 & 0\,0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0\,0 & \cdots & 0\,1 & 0\,0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & 0\,\underbrace{0 \;\; \cdots \;\; 0}_{2j-2} & & 0\;\;1 & \underbrace{0 \;\; \cdots \;\; 0}_{2N-2j} & & \end{bmatrix} \tag{15}$$

$$\mathbf{H}_j^t = \mathbf{H}_j^t \mathbf{M}_j$$

The Kalman gain due to jth object t j K is given by equation 16.

$$\mathbf{K}_j^t = \widehat{\mathbf{P}}_t (\mathbf{H}_t^j)^T (\mathbf{H}_t^j \widehat{\mathbf{P}}_t (\mathbf{H}_t^j)^T + \mathbf{R}_t)^{-1} \tag{16}$$

where Rt is the measurement noise. The update in state xt state covariance Pt due to jth object is given by equation 17.

$$\begin{aligned} \widehat{\mathbf{x}}_t &= \widehat{\mathbf{x}}_t + \mathbf{K}_t^j \mathbf{y}_t^j \\ \widehat{\mathbf{P}}_t &= \widehat{\mathbf{P}}_t - \mathbf{K}_t^j \mathbf{H}_t^j \widehat{\mathbf{P}}_t \end{aligned} \tag{17}$$

This process is repeated for all detected objects. After each iteration, the estimated position of the object is compared with the average past position of the objects in navigation frame. If the difference in current estimated position and the average past position of the objects is found to be crossing a threshold based on the object type, then the process noise associated with the object, Qo, is decreased to model the stationary object

CHAPTER 2
LITERATURE SURVEY

2.1 Introduction

Deep Learning for Object Identification in LiDAR for Autonomous Vehicles[1].According to, a distant sensor called Light Detection and Ranging (LiDAR) utilises laser light in the form of pulses to compute distances to the Earth and has the ability to aid in mapping and monitoring procedures. Compared to monocular cameras, LiDAR offers various benefits for AV safety, including mapping the area. In the rain or at night, monocular cameras have trouble mapping the environment. By scanning with pulsed laser light and measuring the amount of time the pulsed light takes to return to the sensor, LiDAR is a sensor that may be used to determine the distance between two objects. An environment map of the area around the AV must be made in order to employ the LiDAR technology. Combs et al. state that the map depicts the measurement of density created by computing the point cloud's 3D and 2D histogram for the appropriate 3D and 2D models. It may use the data acquired to track and predict an object's movements. The bounding box methodology or various colours can be used by the environment map to categorise each object. Utilizing Patterns of Motion and Appearance to Detect Pedestrians [2]. LiDAR is a sensor that can scan with pulsed laser light and estimate the distance between two objects. These issues are addressed by the novel trainable similarity measure proposed in this study. The similarity is calculated using unique matches in a set of surrounding picture areas. The training procedure improves the collection of areas that are pertinent for a specific similarity evaluation. Highperformance data representations and classifiers are produced by the trainable similarity. As shown by a number of tests, including the classification of road signs, LiDAR is a sensor that can be used to estimate the distance between two objects by scanning with pulsed laser light Using a Trainable Similarity Metric to Create Road-Sign Classifiers[3]. Together with multi-class classification accuracy, the capacity to reject non-signals and the execution's computing requirements are also included. It appears that some of the difficulties faced by the trainable approaches currently being used in the categorization of traffic signs are mitigated by similarity representation. Deep neural network-based item identification on the road.[4] Methods for detecting objects on the road have been investigated for a while. Computer vision research has been particularly active in recognising moving objects, such as people and cars. During the past several years, an improved object detecting mechanism has emerged by employing Convolutional Neural Network in a more practical manner. To locate an item, object identification systems used region proposal techniques based on sliding windows. Such a technique can be helpful for finding an object in any part of the image. Unfortunately, it is required to search the unneeded zones where items never exist, and this adds to the computing load by producing false positive detections. Deep Learning in Self-Driving Vehicles for Traffic Light Detection and Recognition[5].

In order to keep up with current research and track current moves in TL detection, a literature study looking at creative and unique approaches to TL detection was conducted. It is common to classify methods for TL recognition as being based on image processing, machine learning, or map-based methods. In the image-processingbased approach, the picture is subjected to one or more actions or procedures in order to yield a certain result. With the use of RGB to HSV conversion, filtering, histogram of oriented gradients (HOG) characteristics, and other techniques, Guo Mu was able to identify and recognise TL and support

In the The primary deep learning techniques for object detection are classified into one-stage detection algorithms and two-stages detection algorithms. One-stage detection techniques such as YOLO and SSD instantly transform the detection problem into a unified regression problem. The one-stage approaches are quicker than two-stage methods because of the peculiarities of the structure. Deep learning combined with computer vision has the potential to produce solid, reasonably priced solutions for the autonomous driving sector. The main aim of Applications for autonomous vehicles include the detection, tracking, and recognition of both static and moving objects, including pedestrians, motorbikes, cars, and other vehicles. One of the difficulties in the realm of computer vision is object recognition.

It may use the data acquired to track and predict an object's movements. The bounding box methodology or various colours can be used by the environment map to categorise each object. Utilizing Patterns of Motion and Appearance to Detect Pedestrians [2]. LiDAR is a sensor that can scan with pulsed laser light and estimate the distance between two objects. These issues are addressed by the novel trainable similarity measure proposed in this study. The similarity is calculated using unique matches in a set of surrounding picture areas. The training procedure improves the collection of areas that are pertinent for a specific similarity evaluation. Highperformance data representations and classifiers are produced by the trainable similarity. As shown by a number of tests, including the classification of road signs, LiDAR is a sensor that can be used to estimate the distance between two objects by scanning with pulsed laser light

Using a Trainable Similarity Metric to Create Road-Sign Classifiers[3]. Together with multi-class classification accuracy, the capacity to reject non-signals and the execution's computing requirements are also included. It appears that some of the difficulties faced by the trainable approaches currently being used in the categorization of traffic signs are mitigated by similarity representation.

Aim of the project

The aim of implementing object detection in autonomous vehicles using deep learning is to enable these vehicles to perceive and understand their surroundings accurately and in real-time. This technology plays a crucial role in achieving the ultimate goal of autonomous driving: ensuring the safety of passengers, pedestrians, and other road users. Here are specific aims associated with implementing object detection in autonomous vehicles using deep learning

Project Domain

The project falls under the domain of “Object Detection in autonomous vehicles" and utilizes deep learning techniques to perceive and understand their surroundings accurately and in real-time.

Scope of the Project

The scope of a project involving object detection in autonomous vehicles using deep learning is extensive and multidisciplinary. It encompasses various aspects, technologies, and challenges related to implementing robust and reliable object detection systems. Here's an overview of the scope of such a project: Data Collection and Annotation: Gather diverse and extensive datasets containing annotated images, videos, and sensor data to train and validate deep learning models. Develop tools and methods for efficient data annotation, ensuring high-quality labeled data for training purposes. Algorithm Development: Research and develop advanced deep learning algorithms and architectures tailored for object detection in autonomous vehicles. Explore techniques for multi-sensor fusion to leverage information from cameras, LiDAR, radar, and other sensors.

CHAPTER 3
PROJECT DESCRIPTION

3.1 Existing System

Several existing systems and technologies have been developed and deployed for object detection in autonomous vehicles using deep learning. Here are a few notable examples: 1.Waymo (formerly Google Self-Driving Car Project):Waymo, owned by Alphabet Inc. (Google's parent company), has developed a robust object detection system for its autonomous vehicles. They use a combination of LiDAR, radar, and cameras for object detection and tracking. Waymo's deep learning algorithms process this sensor data to identify and predict the behavior of pedestrians, vehicles, cyclists, and other objects on the road.

2.Tesla Autopilot: Tesla's Autopilot feature utilizes deep learning algorithms for objectdetection and recognition. Tesla's vehicles are equipped with cameras and radar sensors, which feed data to their neural networks. Over-the-air updates enable continuous improvement of the system's object detection capabilities..

3. NVIDIA DRIVE Platform:NVIDIA offers the DRIVE platform, a scalable AI-based platform for autonomous vehicles. It includes hardware (such as GPUs) and software components for perception tasks, including object detection. The platform leverages deep learning techniques to process data from various sensors and detect objects in real-time.

3.2 Proposed System

The proposed system for object detection in autonomous vehicles using deep learning integrates cameras, LiDAR, and radar sensors. It employs state-of-the-art deep learning models, including EfficientDet and PointNet, for real-time and accurate object detection. The system incorporates multi-modal sensor fusion, semantic segmentation, object tracking, and prediction. It ensures safety through fail-safe mechanisms and redundant sensor systems, while continuous learning and improvement are facilitated through online learning. The proposed system also addresses regulatory compliance, ethical considerations, and includes comprehensive testing, validation, and documentation procedures.

Deep Learning Models: Implement state-of-the-art deep learning architectures, such asEfficientDet, CenterNet, or PointPillars, tailored for object detection in autonomous vehicles. Employ transfer learning techniques using pre-trained models on large datasets to leverage knowledge from related tasks and optimize training efficiency.

Real-Time Processing: Utilize high-performance GPUs or specialized accelerators like NVIDIA's GPUs designed for AI tasks, ensuring real-time processing capabilities.

Implement model quantization and optimization techniques to reduce computational complexity while maintaining accuracy. Object Tracking and Prediction:

Combine object detection with tracking algorithms like Kalman filters or deep SORT for robust object tracking over time, enabling the vehicle to anticipate object movements.

Implement trajectory prediction algorithms to anticipate future positions of moving objects, enhancing decision-making capabilities.

Edge Computing:

Explore edge computing solutions, allowing some processing tasks to be performed locally in the vehicle, reducing latency and enabling faster responses to dynamic scenarios.

Feasibility Study

Economic Feasibility

Estimate the costs associated with acquiring sensors, computing hardware, software licenses, and skilled personnel for system development and implementation.

Return on Investment (ROI): Analyze the potential benefits of the system, such as improved road safety, reduced accidents, and enhanced transportation efficiency, against the projected costs. Calculate the ROI to determine the project's financial viability.

Technical Feasibility

Availability of Technology: Assess the availability and maturity of deep learning frameworks, algorithms, and hardware accelerators necessary for real-time object detection Data Requirements:Evaluate the availability and quality of annotated datasets required for training the deep learning models. Consider the diversity and volume of data needed to ensure the system's accuracy and robustness. Sensor Integration: Investigate the compatibility and integration complexities of various sensors (cameras, LiDAR, radar) to ensure seamless data fusion and processing.

Market Feasibility

Market Demand: Analyze the market demand for autonomous vehicles and related technologies. Identify potential stakeholders, partners, or clients interested in adopting the proposed object detection system. Competitive Analysis: Evaluate existing solutions and competitors in the market. Identify unique selling points and differentiators for the proposed system.

System Specification

Hardware Specification

CPU-Intel.

RAM-4GB or 8GB.

Storage-512GB or more.

GPU-Intel.

Programming languages:

Python

C++

CUDA for developing deep learning algorithms

Standards and Policies

Standards and policies play a crucial role in the development, deployment, and operation of autonomous vehicles, especially concerning object detection systems. Adhering to established standards and policies ensures safety, reliability, and ethical considerations in the design and operation of autonomous vehicles. Here are some key standards and policies relevant to object detection in autonomous vehicles using deep learning:. **Standard Used: ISO/IEC 26262:**

1.ISO 26262 - Functional Safety for Road Vehicles: ISO 26262 is an international standard for functional safety of electrical and electronic systems in vehicles. It sets the standard for safety-related systems, including object detection systems in autonomous vehicles. Compliance ensures that the system is designed to be safe and reliable.

CHAPTER 4
ER DIAGRAMS

4.1 General Architecture

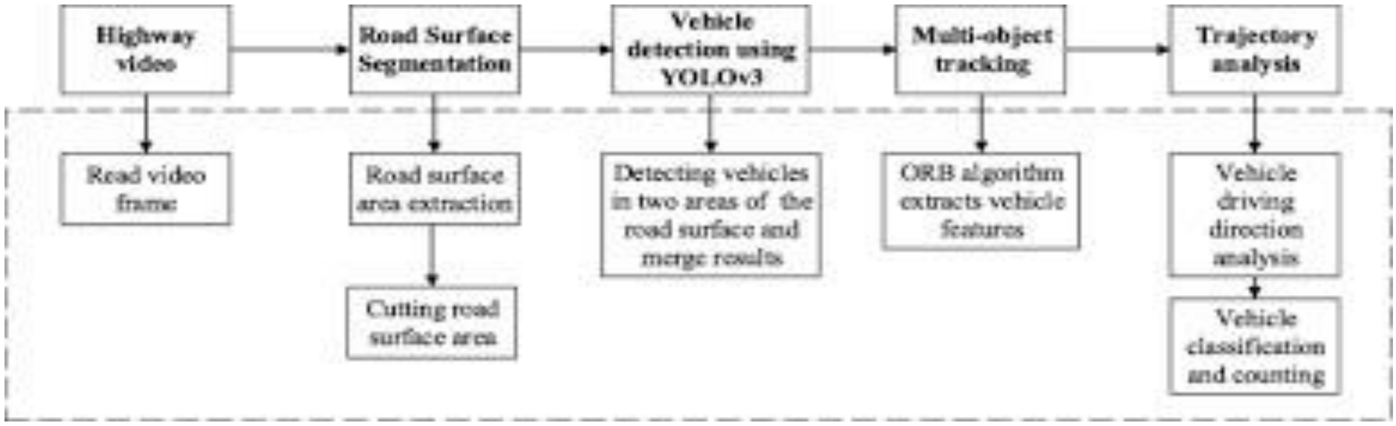

Fig 4.1 Architecture Diagram

From the above Fig 4.1 diagram an architecture diagram serves as a visual blueprint that provides a comprehensive view of the structure and components within a system, application, or technological infrastructure. Typically, it includes a range of key elements. Components are illustrated, representing the fundamental building blocks of the system, which may encompass servers, databases, applications, services, and external entities. Connections and lines with arrows establish the relationships and data flow between these components, clarifying how information moves within the system.

4.2 Use case Diagram

From the use case diagram doesn't go into a lot of details for example, don't expect it to model the order in which steps are performed. Instead, a proper use case diagram depicts a high-level overview of the relationship between use cases, actors, and systems. Experts recommend that use case diagrams be used to supplement a more descriptive textual use case. Use cases are represented with a labeled oval shape. Stick figures represent actors in the process, and the actor's participation in the system is modeled with a line between the actor.

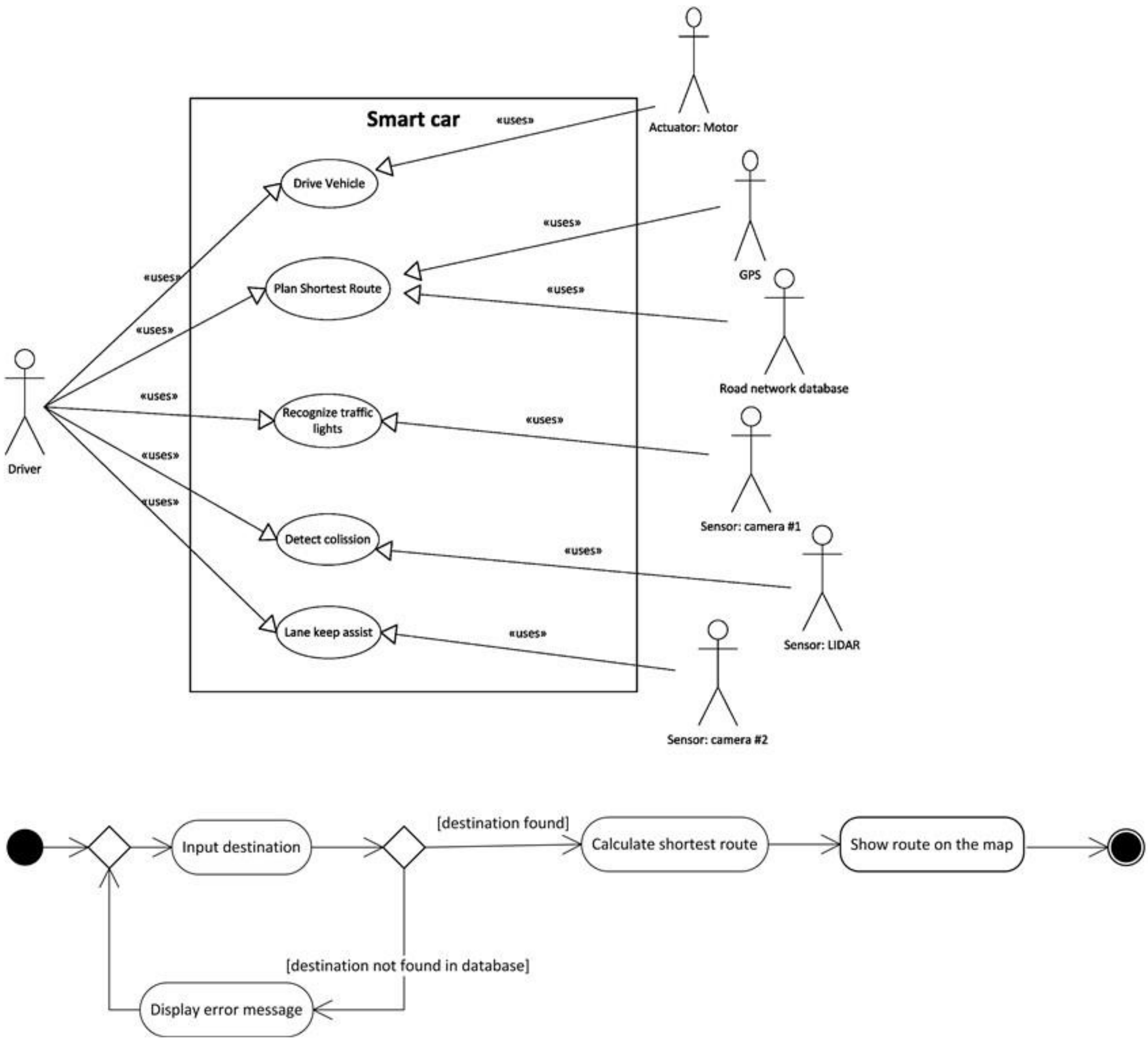

Fig 4.2 Use Case Diagram

4.3 Class Diagram

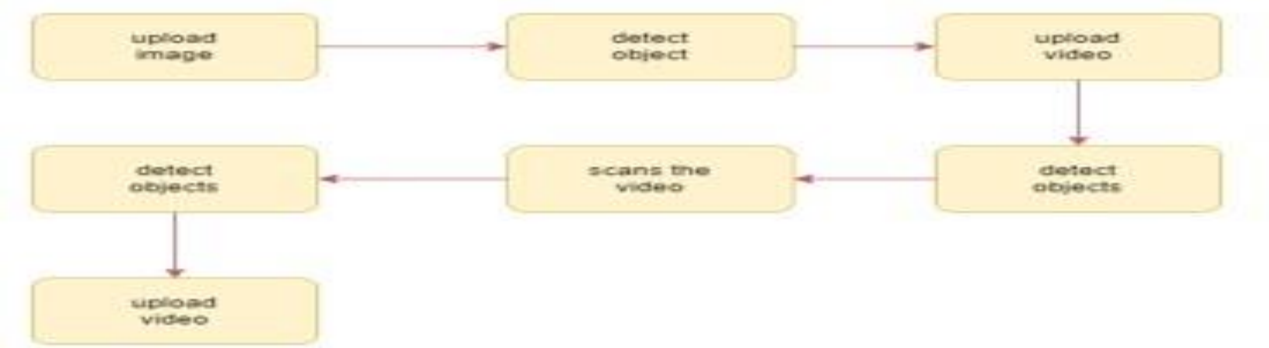

Figure 4.3: Class Diagram

From the above Fig 4.3 class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing and documenting different aspects of a system but also for constructing executable code of the software application. Class diagram describes the attributes and operations of a class and also the constraint's imposed on the system. The class diagrams are widely used in the modeling of object-oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

4.4 Sequence Diagram

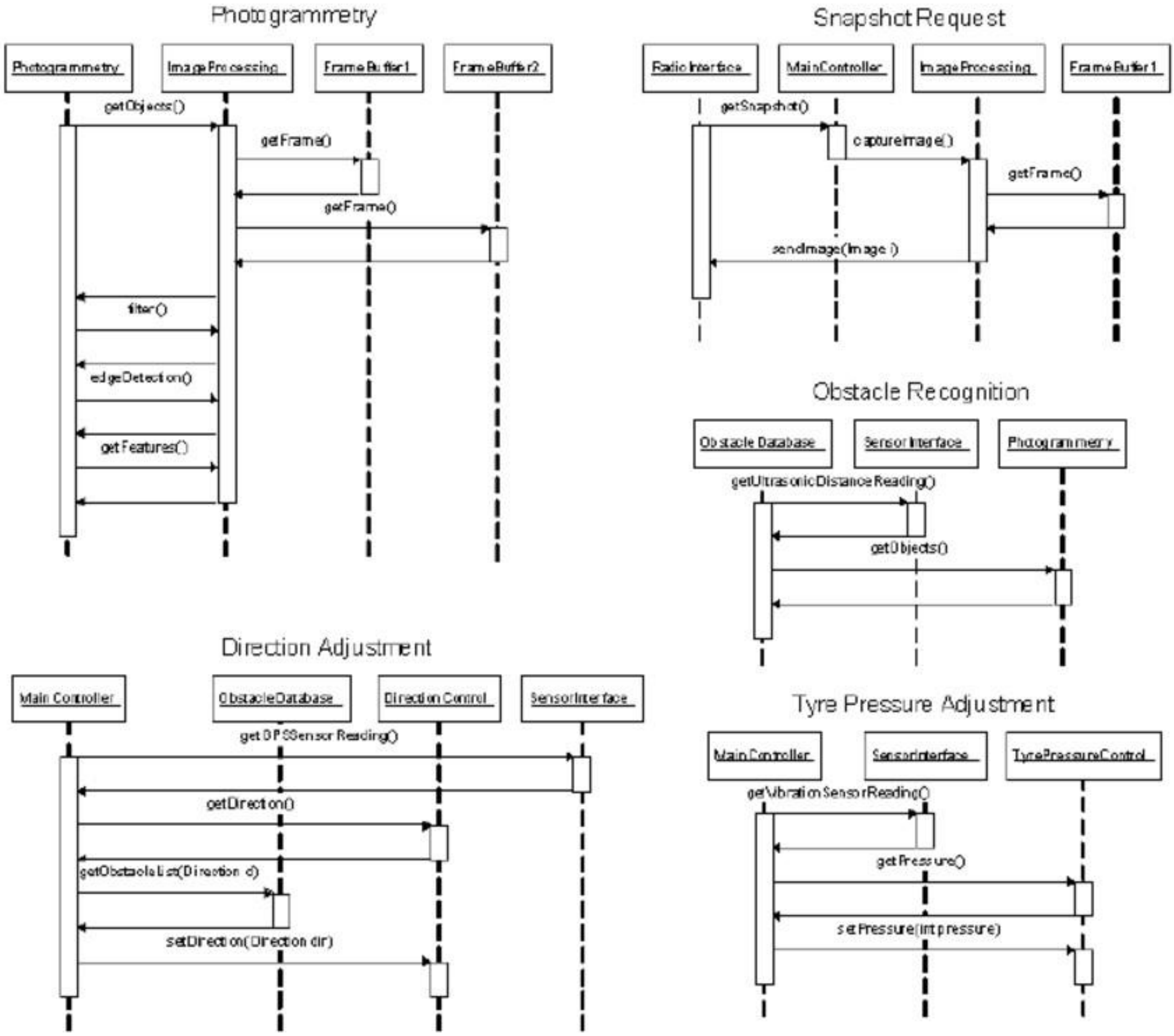

Figure 4.4: Sequence Diagram

From the above Fig 4.4 sequence diagram or system sequence diagram (SSD) shows object interactions arranged in time sequence in the field of software engineering. It depicts the object involved in the scenario and sequence of message exchanged between he objects needed to carry out the functionality of scenario. Sequence diagram are typically associated with the use case realization in the logical view of the system under development. Sequence diagram are sometimes called event diagrams or event scenarios.

4.5 Collaboration Diagram:

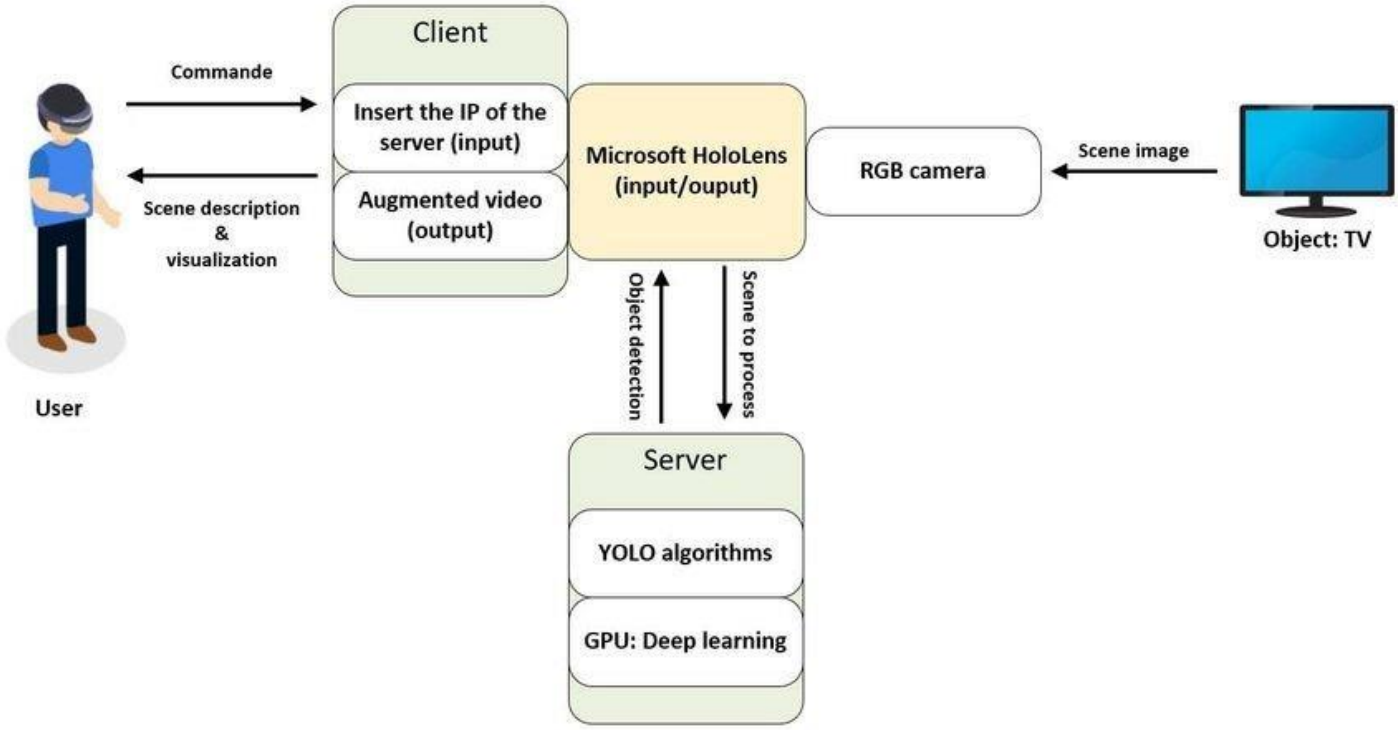

Fig 4.5: Collaboration Diagram

From the above Fig 4.5 collaboration diagram, also known as a communication diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML). These diagrams can be used to portray the dynamic behavior of a particular use case and define the role of each object. Collaboration diagrams are created by first identifying the structural elements required to carry out the functionality of an interaction. A model is then built using the relationships between those elements. Several vendors offer software for creating and editing collaboration diagrams.

4.6 Activity Diagram:

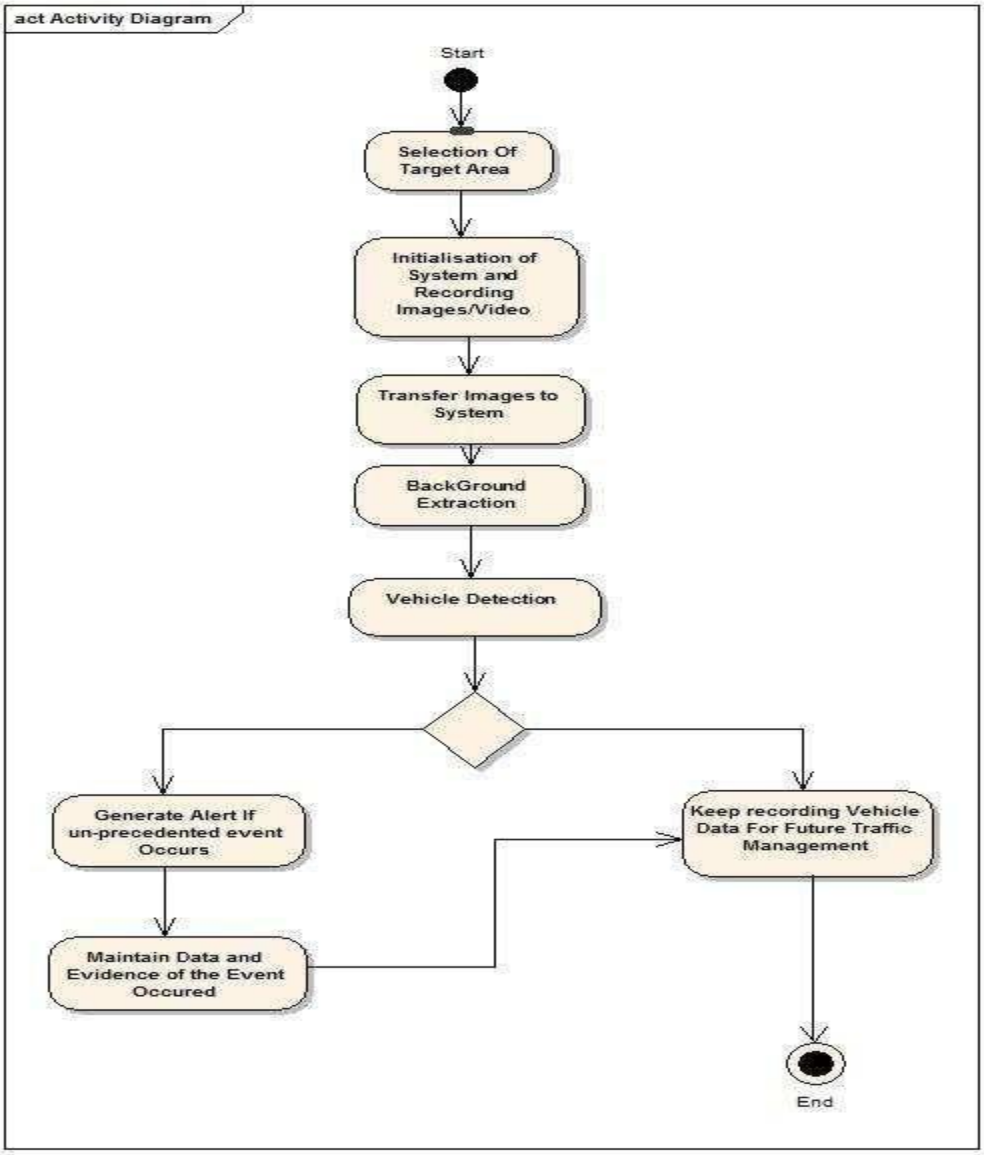

Figure 4.6: Activity Diagram

From the above Fig4.6 activity diagram visually presents a series of actions or flow of control in a system similar to a flowchart or a data flow diagram. Activity diagrams are often used in prediction process modeling. They can also describe the steps in a use case diagram.

Activities modeled can be sequential and concurrent.

CHAPTER 5
ANALYSIS

5.1 Introduction

In recent years autonomous/self-driving cars have drawn much interest as a topic of research for both academia and industry. For a car to be a truly autonomous, it must make sense of the environment through which it is driving. The autonomous car must be able to both localize itself in an environment and identify and keep track of objects (moving and stationary). The car gets information about the environment using exteroceptive sensors such LiDAR, cameras, inertial sensors, and GPS. The information from these sensors can be used together and fused to localize the car and track objects in its environment, allowing it to travel successfully from one point to another. The process of path planning and autonomous vehicle guidance depends on three things: localization, mapping, and tracking objects. Localization is the process of identifying the position of the autonomous vehicle in the environment. Mapping includes being able to make the sense of the environment. Tracking of moving object involves being able to identify the moving objects and track them during navigation. The process of localization and mapping have been explored through the use of Simultaneous localization and mapping (SLAM) which was initially proposed by Leonard and Durant-Whyte [4]. SLAM enables autonomous vehicles to simultaneously build the map of the unknown environment and localize itself in the unknown environment. The development of SLAM has been significant in the development of autonomous robots. Most of the research in SLAM suppose the environment to be static and consider the moving object as noise. The detection and tracking of moving objects (DATMO) is one of the most challenging issues. The detection of moving objects is important for safety and essential for avoiding collisions. SLAM combined with DATMO helps solve this problem.

The combination of SLAM and DATMO has been approached in different ways. Most of the literature [5], [6], [7] in the area of DATMO have solved the problem using a laser scanner as the main perception sensor of the vehicle. In these works, the laser data is used to detect the moving object. The problem with using only a laser-based sensor is that the observed shape of the object detected can change from one scan to another scan, making it hard to track the object. The object classified using the laser scanner usually depends upon the shape of the object. This can also lead to misclassification as the shape of one object might look like another, the result could be that a tree may be classified as a person. Consequently, using a laser scanner as the main or only perception sensor might not be right solution for tracking objects With advancements in the area of deep learning and incremental improvements in computing power, object detection using images outperforms other methods for the detection and classification of objects. An image is rich with information about the environment. Object detected using a camera fused with distance information from a laser scanner improves the performance of DATMO. In this paper, the problem of DATMO is explored with the use of a deep learning architecture for the detection and classification of the objects. To describe this work, this paper begins with a description of the dataset used in the Dataset section. In the System Overview section, the proposed algorithm is presented. This is followed by a description of object detection and by object matching. The results of this work are then presented and followed by a discussion.

The process of path planning and autonomous vehicle guidance depends on three things: localization, mapping, and tracking objects. Localization is the process of identifying the position of the autonomous vehicle in the environment. Mapping includes being able to make the sense of the environment. Tracking of moving object involves being able to identify the moving objects and track them during navigation. The process of localization and mapping have been explored through the use of Simultaneous localization and mapping (SLAM) which was initially proposed by Leonard and Durant-Whyte [4]. SLAM enables autonomous vehicles to simultaneously build the map of the unknown environment and localize itself in the unknown environment. The development of SLAM has been significant in the development of autonomous robots. Most of the research in SLAM suppose the environment to be static and consider the moving object as noise. The detection and tracking of moving objects (DATMO) is one of the most challenging issues. The detection of moving objects is important for safety and essential for avoiding collisions. SLAM combined with DATMO helps solve this problem.

CHAPTER 6
DESIGN

6.1 Design

You Only Look Once (YOLO) You only look once is an algorithm based on CNN used to detect, classify, and localize objects in an image. Figure 19 shows working of YOLO. When objects are to be detected in an image, first it is resized into the size of the input layer of the YOLO architecture, then the image is run through the CNN and the output is the bounding box for each detected object with classification and probability score.

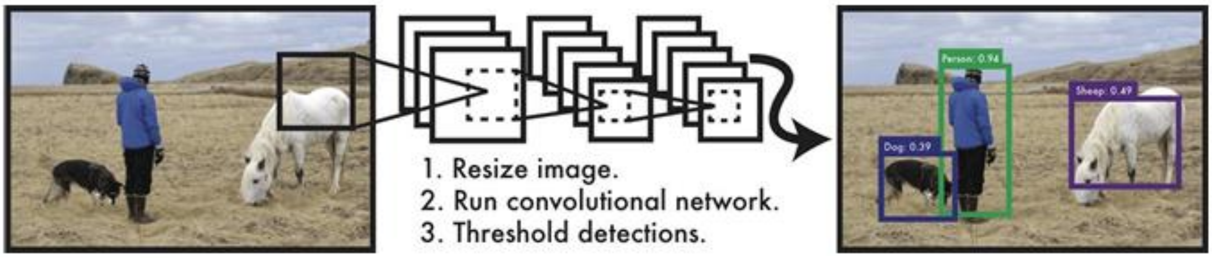

Fig 6.1 YOLO detecting,classifying and localizing obects in an image

YOLO can see an entire image at once and simultaneously predicts the multiple bounding boxes and class probabilities for each box, making it faster compared to other region-based method or sliding window methods such as DPM, R-CNN, Fast R-CNN, and Faster R-CNN. Figure 20 shows the performance of each of these method speed and accuracy on Nvidia Titan X GPU and the same test dataset Pascal 2007 [16]. The Faster R-CNN is accurate by 4 percent compared to YOLO, but YOLO's speed is far more superior. This makes YOLO suitable for using in real-time systems such as autonomous vehicles.

	Pascal 2007	Speed	
DPM v5	33.7	.07 FPS	14 s/img
R-CNN	66.0	.05 FPS	20 s/img
Fast R-CNN	70.0	.5 FPS	2 s/img
Faster R-CNN	73.2	7 FPS	140 ms/img
YOLO	69.0	45 FPS	22 ms/img

Fig 6.2 Performance of different object detection Algorithm

The working of YOLO is illustrated in the Figure 21. The figure is split into 13x13 grid. For each grid square, YOLO predicts 5 bounding boxes, with object probability, and class probability. The probability score for the bounding box and the class probability is combined into one final score that tells probability that this bounding box contains a specific type of object. For a 13x13 grid, there will be 845 bounding boxes predicted by the YOLO, but only the bounding boxes crossing a predefined threshold is kept. In this thesis, the bounding boxes with a probability greater than 50% are kept for the object.

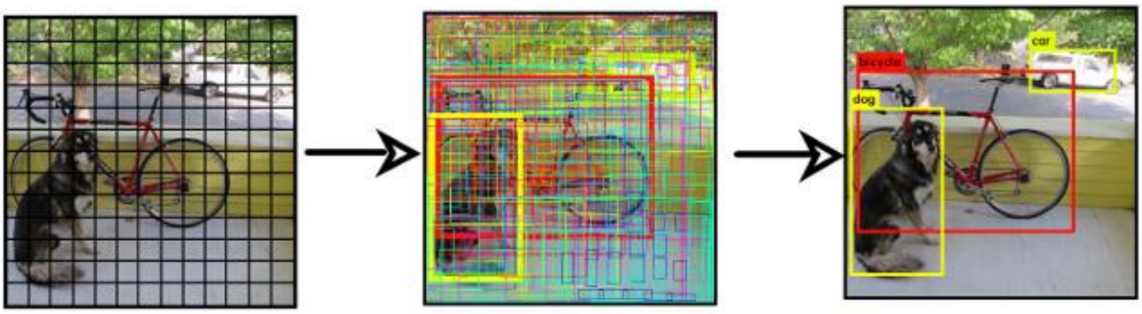

Fig 6.3 Working of YOLO

In this thesis, YOLO is used to detect six classes of objects. The classes include person, bicycle, motor bike, car, truck, and bus. The pretrained YOLO network obtained from [17] was used to detect and classify objects in the thesis

ORB (Oriented FAST and Rotated BRIEF)

ORB is a very fast algorithm that creates a feature vector from the detected keypoints. ORB is invariant to rotations and changes in illumination and noise [2]. The first step in ORB is used to find the keypoints in an image, which is done by using the FAST algorithm. FAST stands for Features from Accelerated Segments Test, which finds keypoints by looking at changes in intensity around a pixel in an image. These keypoints give us information about the edges in an image. After the keypoints are calculated the top N points are selected using Harris corner measure. ORB uses BRIEF to create binary descriptors from the keypoints. BRIEF starts by smoothing a given image with a Gaussian Kernel in order to prevent the descriptor from being too sensitive to high frequency noise.

Extended Kalman Filter

Bayesian filtering is the process of estimating states over time, using incoming measurements and a mathematical process model. A system is represented as a probabilistic state-space model where the noise is additive and Gaussian. The Kalman filter is a special case of Bayesian filter where the model is linear and gaussian. The Extended Kalman filter (EKF) is a non-linear version of the Kalman filter. The EKF is obtained using a linear approximation of a non-linear system.

The dynamics of the nonlinear system is given by equation 3.14 and 3.15,

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{u}_k) + \mathbf{w}_k \qquad (3.14)$$

$$\mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{v}_k \qquad (3.15)$$

where xk is the state at time k, and zk is the measurement for the system at time k, and uk is the input to the system. The dynamics of the system is modeled by ( ) f • and ( ) h• is measurement model for the system. wk is the process noise with zero mean and covariance k Q

$$\mathbf{w}_k \sim \mathcal{N}(0, \mathbf{Q}_k) \qquad (3.16)$$

vk is motion noise with zero mean and covariance k

$$\mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R}_k) \qquad (3.17)$$

In EKF, the state of the system at any time k is assumed as a Gaussian approximation with, mean k x and the covariance k P.

$$\mathbf{x}_k \sim \mathcal{N}(\bar{\mathbf{x}}_k, \mathbf{P}_k) \qquad (3.18)$$

The non-linearity in EKF approximated as linear system near a point using Taylor series and is modeled using Jacobian matrix. The state estimation of the system is done in two stage. First the predict stage and the update stage. The prediction state of EKF is given in following equations. k x is predicted estimate of the state based on prior information, k P is the predicted estimate of the covariance matrix.

$$\hat{\mathbf{x}}_k = f(\mathbf{x}_{k-1}, \mathbf{u}_k) \qquad (3.19)$$

$$\hat{\mathbf{P}}_k = \mathbf{F}_k \mathbf{P}_{k-1} \mathbf{F}_k^T + \mathbf{Q}_{k-1} \qquad (3.20)$$

where Fk is the Jacobian matrix given by equation (3.20)

$$\mathbf{F}_k = \frac{\delta f}{\delta x}\Big|_{\hat{x}_{k-1}, u_k} \qquad (3.21)$$

The update state of EKF is done by following equations,

$$\mathbf{S}_k = \mathbf{H}_k \hat{\mathbf{P}}_k \mathbf{H}_k^T + \mathbf{R}_k \tag{3.22}$$

$$\mathbf{K}_k = \hat{\mathbf{P}}_k \mathbf{H}_k^T \mathbf{S}_k^{-1} \tag{3.23}$$

37

$$\mathbf{x}_k = \hat{\mathbf{x}}_k + \mathbf{K}_k (\mathbf{z}_k - h(\hat{\mathbf{x}}_k)) \tag{3.24}$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \hat{\mathbf{P}}_k \tag{3.25}$$

where Hk is the Jacobian matrix given by equation (3.26)

$$\mathbf{H}_k = \frac{\delta \mathbf{h}}{\delta x} \Big|_{\hat{x}_{k|k-1}} \tag{3.26}$$

where Kk is the Kalman gain, Sk is the innovation covariance matrix. The Kalman gain tells us trust in the measurements. If the sensor gives bad measurements, the Kalman gain will be low and high if the sensors give reliable measurements. The innovation covariance matrix Sk is the covariance matrix of the measurements and tells about the errors in measurements.

CHAPTER 7
IMPLEMENTATION

7.1 OBJECT DETECTION

To perform object detection, this work uses datasets that provide information of the environment through the LiDAR and camera. Using the information from these sensors, objects are detected, classified, and the distance and direction of the object relative to the Robotcar is measured. Usually object detection is achieved using a combination of feature-based modelling and appearance-based modelling. The image has more information that can be used to identify objects as compared to laser scan and allows both features based and appearance-based modelling. In this paper, the image is primarily used to detect objects and classify them, and the LiDAR is used to measure the location of the object relative to the vehicle.

The laser scan combined with the pose of the vehicle is used to create the 3D point cloud of the environment. This is then projected onto the image. The transformation of 3D coordinates obtained from LiDAR scan to 2D image pixels is done using pinhole camera model. The LiDAR (x, y, z) coordinates are transformed into pixels (u, v) in the image using equation 5.

$$u = \frac{f}{z} x + u_0 \qquad (5)$$
$$v = \frac{f}{z} y + v_0$$

where f is the focal length of the camera and $(u_0,v_0)$is the optical center of the camera.

The Figure 4 shows sample image and the laser scans projected onto the image.

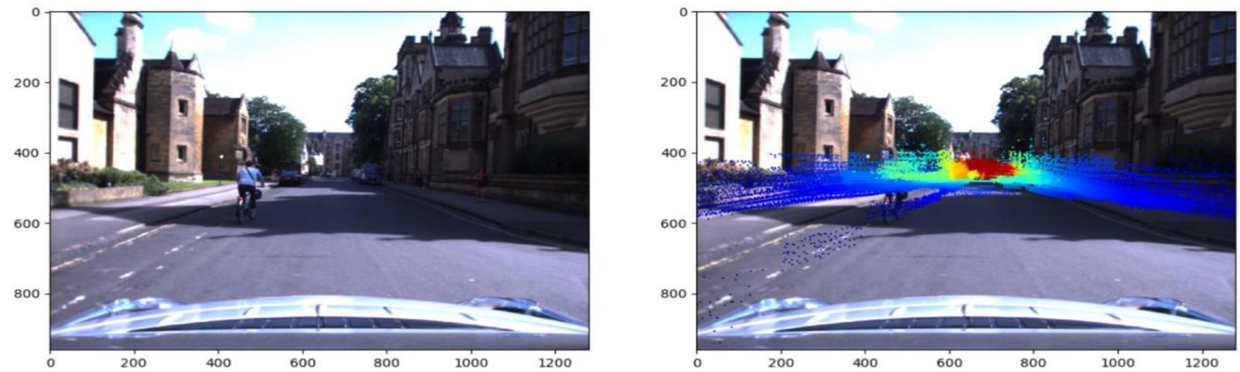

Fig 7.1 A Sample Image and Laser can projected onto the image

7.2 EXECUTION CODE

```
import cv2
import numpy as np
from time import sleep

largura_min=80 #Largura minima do retangulo
altura_min=80 #Altura minima do retangulo

offset=6 #Erro permitido entre pixel

pos_linha=550 #Posição da linha de contagem

delay= 60 #FPS do vídeo

detec = []
carros= 0


def pega_centro(x, y, w, h):
    x1 = int(w / 2)
    y1 = int(h / 2)
    cx = x + x1
    cy = y + y1
    return cx,cy

cap = cv2.VideoCapture('video.mp4')
subtracao = cv2.bgsegm.createBackgroundSubtractorMOG()

while True:
    ret , frame1 = cap.read()
    tempo = float(1/delay)
    sleep(tempo)
    grey = cv2.cvtColor(frame1,cv2.COLOR_BGR2GRAY)
    blur = cv2.GaussianBlur(grey,(3,3),5)
    img_sub = subtracao.apply(blur)
    dilat = cv2.dilate(img_sub,np.ones((5,5)))
    kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (5, 5))
    dilatada = cv2.morphologyEx (dilat, cv2. MORPH_CLOSE , kernel)
    dilatada = cv2.morphologyEx (dilatada, cv2. MORPH_CLOSE , kernel)
    contorno,h=cv2.findContours(dilatada,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)

    cv2.line(frame1, (25, pos_linha), (1200, pos_linha), (255,127,0), 3)
    for(i,c) in enumerate(contorno):
```

```
            (x,y,w,h) = cv2.boundingRect(c)        validar_contorno = (w >= largura_min) and (h >= altura_min)
            if not validar_contorno:
                continue

            cv2.rectangle(frame1,(x,y),(x+w,y+h),(0,255,0),2)
            centro = pega_centro(x, y, w, h)
            detec.append(centro)
            cv2.circle(frame1, centro, 4, (0, 0,255), -1)

            for (x,y) in detec:
                if y<(pos_linha+offset) and y>(pos_linha-offset):
                    carros+=1
                    cv2.line(frame1, (25, pos_linha), (1200, pos_linha), (0,127,255), 3)
                    detec.remove((x,y))
                    print("car is detected : "+str(carros))

        cv2.imshow("Video Original" , frame1)
        cv2.imshow("Detectar",dilatada)

        if cv2.waitKey(1) == 27:
            break

    cv2.destroyAllWindows()
    cap.release()
```

OUTPUT

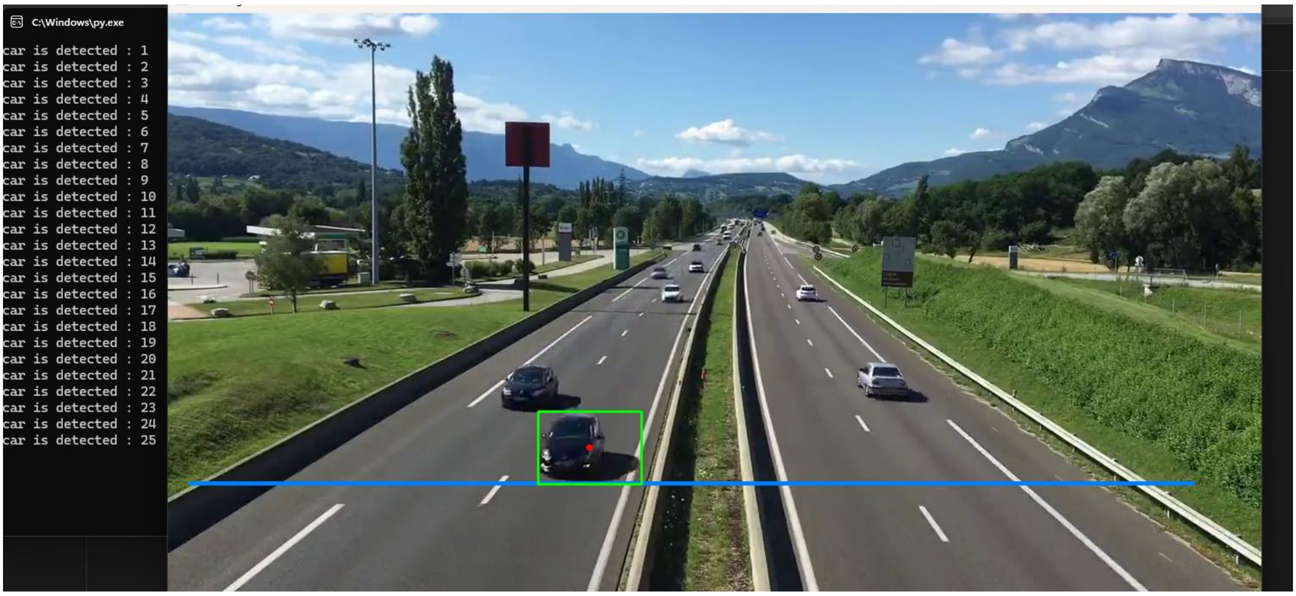

Fig 7.2 Output of the execution code

CHAPTER 8
RESULTS

In this section the algorithm developed is tested using the Oxford RobotCar dataset to see the performance on tracking the objects and position estimation of the RobotCar. The exact ground truth data for the position of the vehicles are not provided so the exact accuracy cannot be calculated.

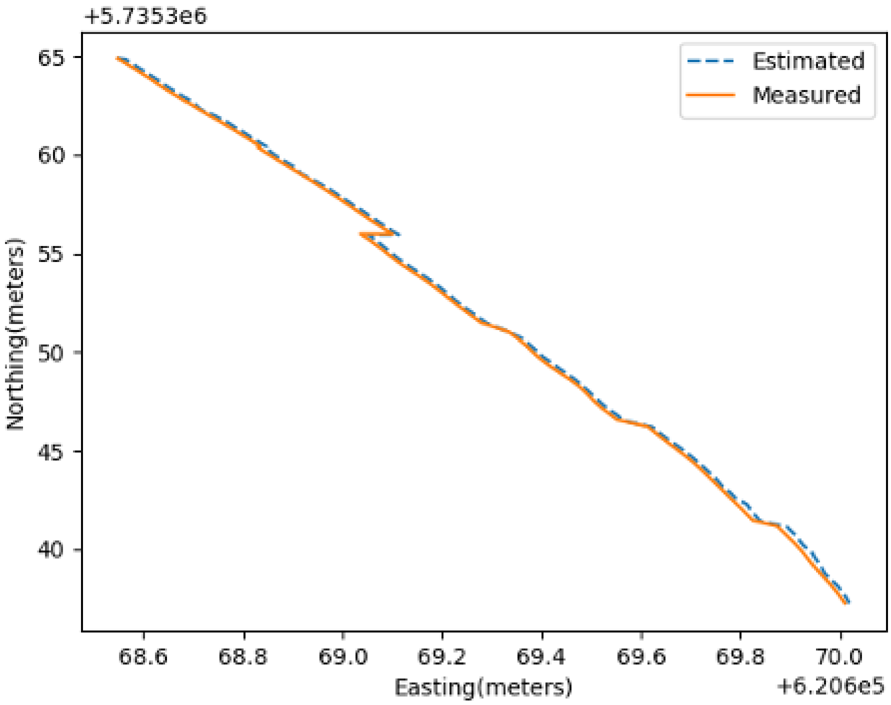

Fig 8.1 Estimation of position of the RobotCar while tracking objects

The aim of the algorithm developed is to aid in localization of the RobotCar while tracking different objects. One Extended Kalman Filter is used to both estimate the position of the RobotCar and track the objects. The RobotCar must successfully be able to estimate its position and localize in the environment. Figure 8.1 shows the position of the RobotCar as estimated versus the position of the RobotCar measured by GPS. As ground truth for the position of the RobotCar is not present, the performance is compared with the measured values. The estimated position closely follows the measured position, with a slight smoothing relative to measured position. The difference between the measured and estimated positions are displayed in Figure 8.2 and is broken down into northing and easting. It is calculated by taking the absolute difference between the estimated and measured position. From Figure 8.2, it can

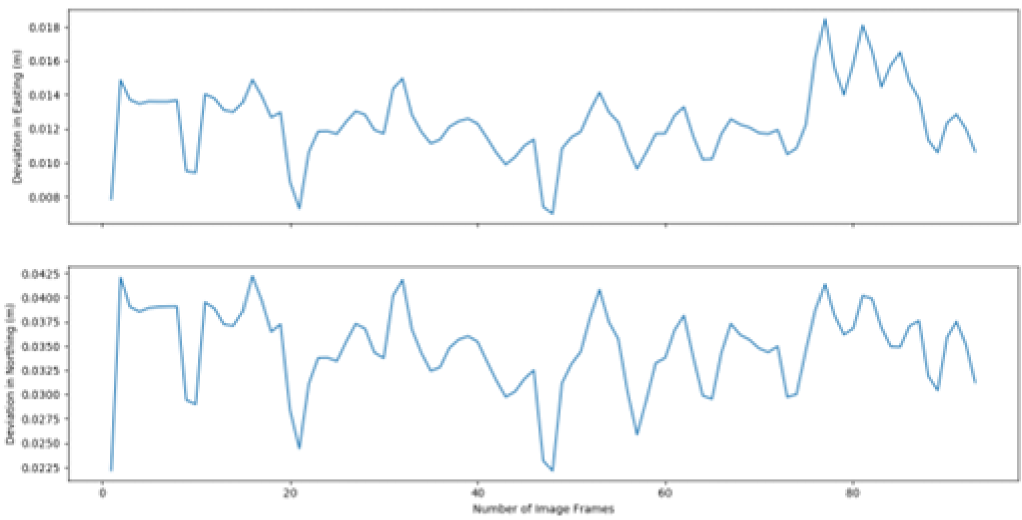

Fig 8.2 Deviation on the estimate for the position of the Robot Car

be observed that there is a translational deviation with peak of 0.018m in Easting and 0.0425m deviation in Northing Position. This shows that the RobotCar GPS measurements are not vastly different than the position estimation from the EKF, which is a good indication that the system is performing well, while tracking the objects around itself. The performance of algorithm on tracking of the moving objects is shown in Figures 8.3 and 8.4. Figure 8.3 shows the image of a person riding a bicycle on the right and the tracking of both objects. The graph shows that position of the both objects (bike and the person on it in navigation frame. The separate plots are the position estimates of the person and bicycle based on the information from lidar scans. As seen in the image, it can be observed that the person is riding a bicycle, so tracking algorithm must estimate them to be together. The graph of the position estimates shows the person and the bicycle to be very similar and

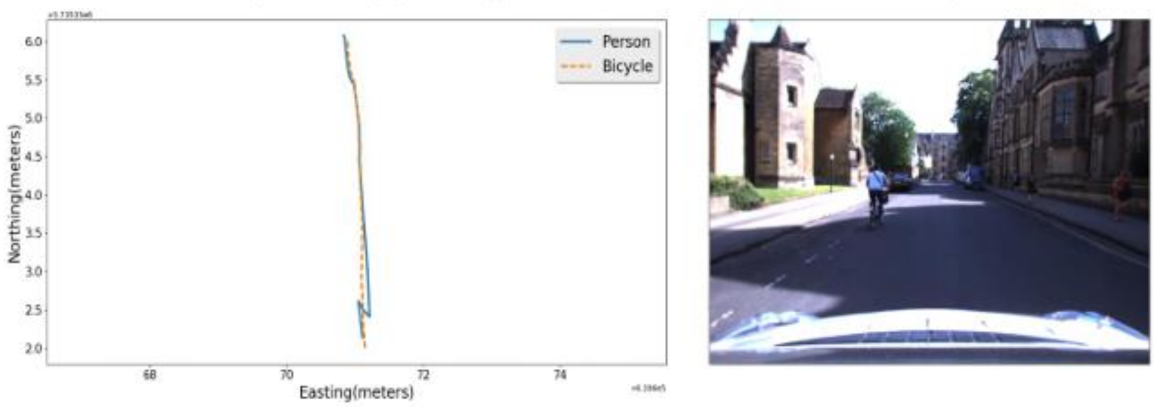

Fig 8.3 Tracking of Person Riding a bicycle

deviating by less than 20cm in the navigation frame. The minor differences in the estimates are due object not being modeled to their actual velocity and laser scanner might not pick both object in same scan. This shows the algorithm is able to track the different objects even if they are moving together. Figure 8.4 shows the performance of the algorithm in tracking another moving car. Figure 8.4 depicts the estimated position of the car by the EKF and measured position of the car based on lidar scans in the navigation frame. The graph shows the estimated position compared with the measured position from the lidar scans. The estimate position follows the measured position. Additional performance could be obtained by adding velocity states for the target vehicle, but the work presented assumed a simple random walk model with an expected variance based on the average velocity of moving vehicles.

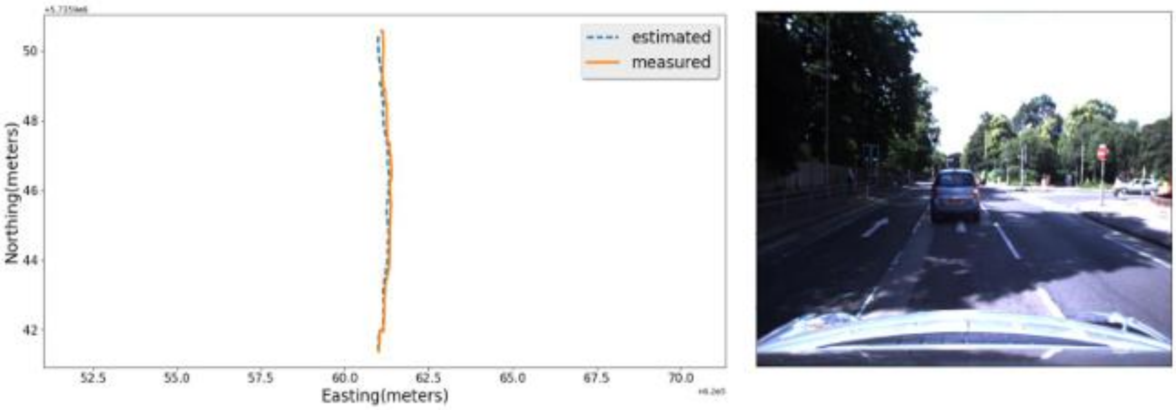

Fig 8.4 Tracking of moving car in Navigation frame

The algorithm developed in the paper tracks stationary object as well. Figure 8.5 shows Car_1 and Car_2 which are stationary being tracked simultaneously. The plot shows the estimated position of Car_1 and Car_2 when the RobotCar in navigating in navigation frame. Based on image frames the car are stationary and the tracking algorithm supports that as there is very minimal change in the position of both the cars. Car_1 was tracked for 25 image frames and Car_2 was tracked for 30 image frames.

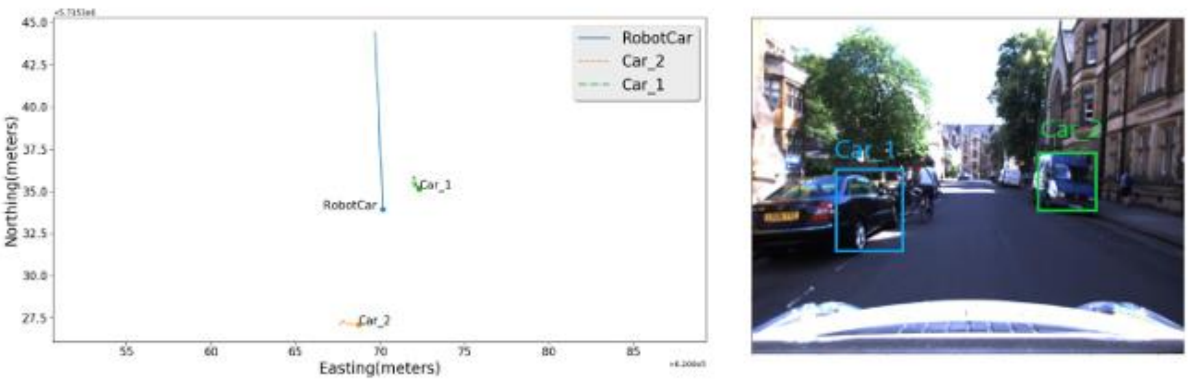

Fig 8.5 Tracking Stationary Cars

Figure 8.5 showed that tracking the stationary car there was some change in the position of the car. Figure 8.6 shows the estimated and measured position of the stationary Car1 in northing and easting with respect to number of image frame tracked. It can be observed that there is change in position of 0.5m in Eastings and 0.7m in Northings. The change in position is because of the measurement from the lidar scans. As the paper models object as a point object, during tracking the lidar scan return is not from the same point. When Car_1 is first detected the lidar scans returns from the bumper and that distance is associated with the object, as the Robotcar moves the lidar scan return is from the side of the car and that distance is associated with the object. As a result of this there is some change in object location.

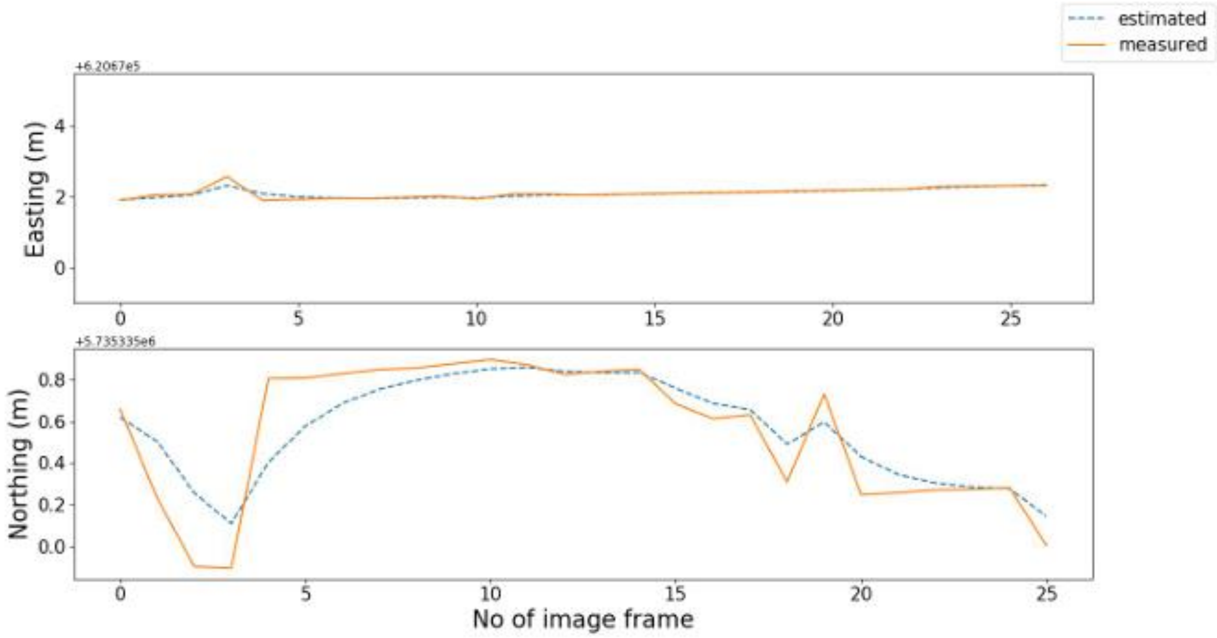

Fig 8.6 Tracking of car_1 in Northing and Easting with respect to Number of Image frames

Results with Snowy Road Condition

The performance of the camera greatly depends on the lighting condition and contrast. The lighting condition during the snowy weather is more diffuse than during normal conditions. Since the

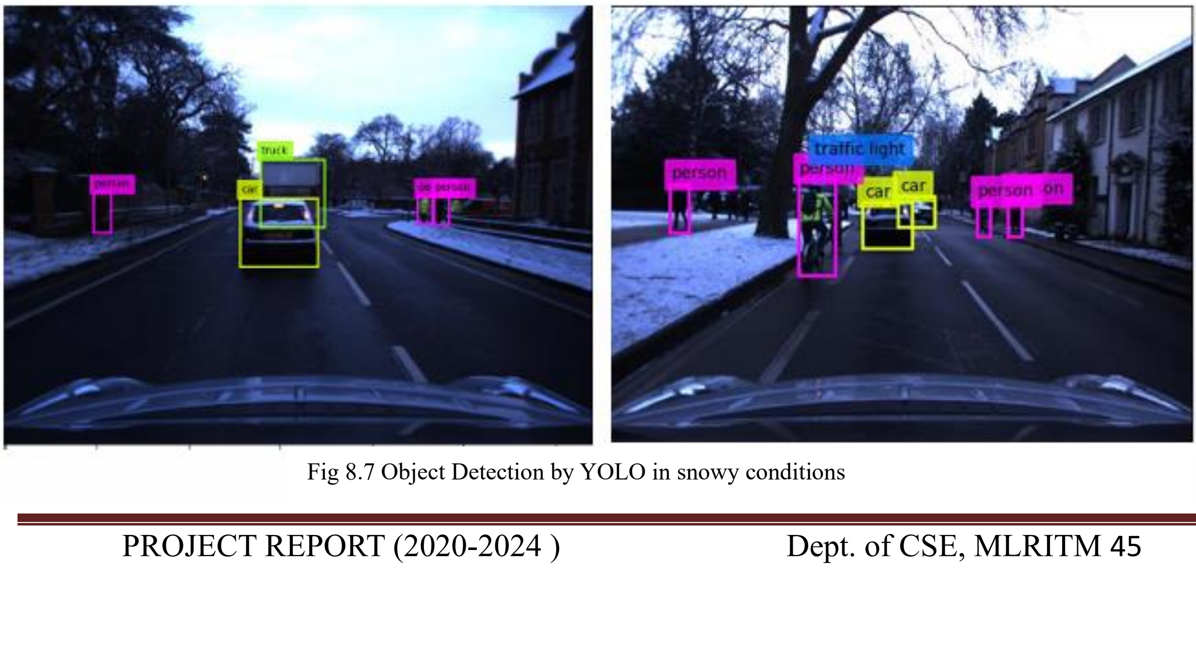

Fig 8.7 Object Detection by YOLO in snowy conditions

lighting conditions are different, the object on the images might not be clear or result in a match. Some of the images captured by the RobotCar during the snow along with the result of the object detection by the YOLO algorithm is shown in Figure 8.7

Figure 8.7 shows that even though the car is covered by snow and the lighting condition is different from normal sunny day, the YOLO detects most of the objects around the RobotCar. It misses a few pedestrians that are far from the RobotCar, but it does not miss any that are nearby. From this it is possible to use YOLO to detect object even in snowy conditions.

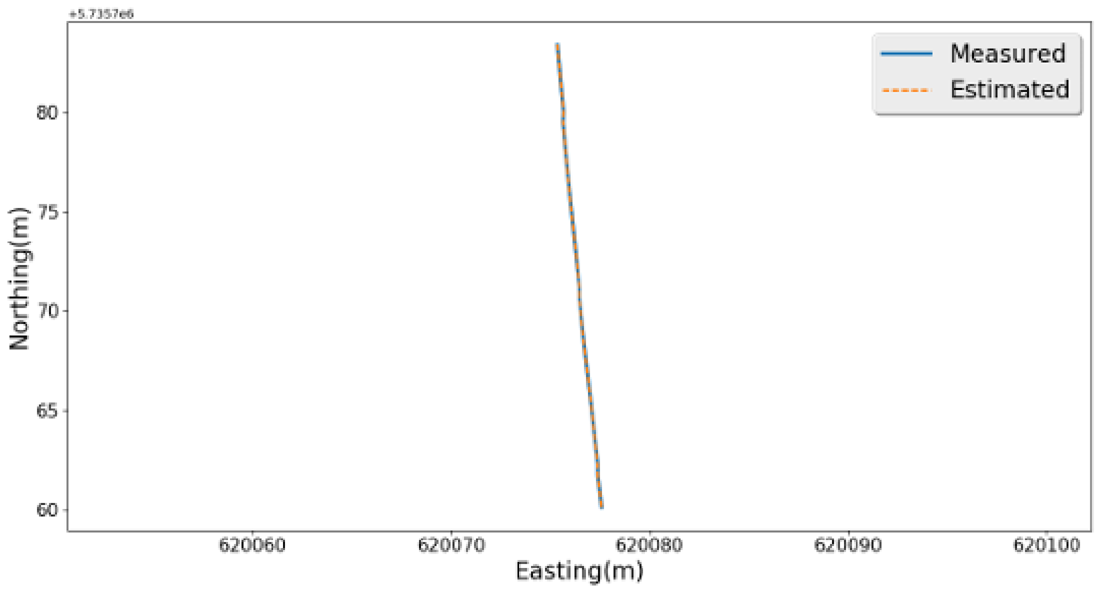

Fig 8.8 Navigation of the Robot car in the snow

The navigation of the RobotCar is presented in the Figure 8.8. We can see that the position estimated by the EKF closely follows the position of the RobotCar obtained from the GPS. The absolute deviation in the Northing and Easting is presented in the Figure 8.9. It is observed that the deviation in Northing is about 0.025m and in Easting it is 0.005m.

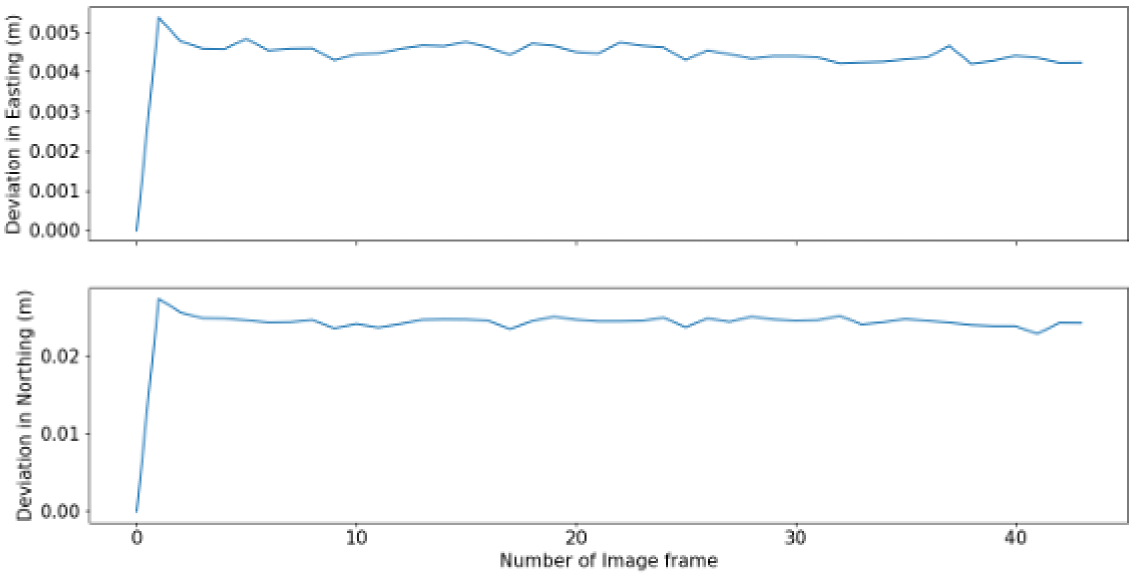

Fig 8.9 Deviation in Easting and Northing

The tracking of the moving objects during snowy is presented in Figure 8.10. The image in Figure 20 consists of three objects, two persons and a bus travelling in a opposite direction. Persons detected in the image were tracked for 10 frames and the bus was tracked for 37 frames by the algorithm developed in this thesis. This demonstrates that even in low light and snow, the feature matching algorithm developed in the thesis can match objects for multiple frames.

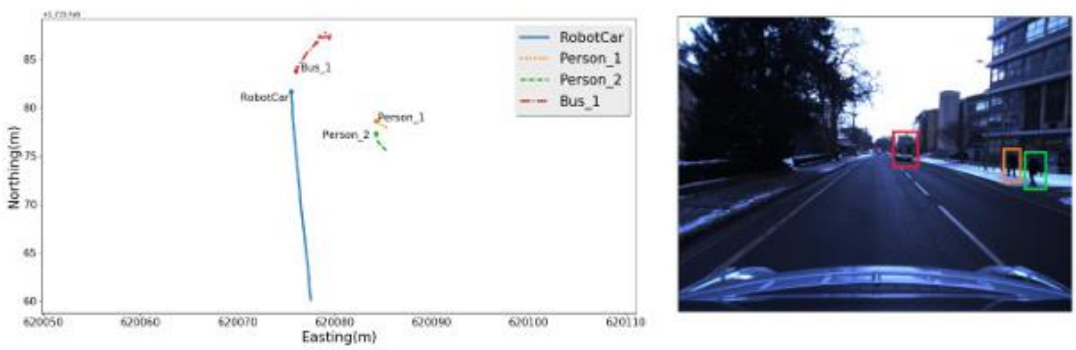

Fig 8.10 Tracking of multiple Objects in Snowy condition

Figure 8.11 shows the tracking of a person when the car is stopped at a red light. The car is stationary while the person is walking across the street as shown in the figure. This demonstrates the algorithm's ability to track the object even when at a stop.

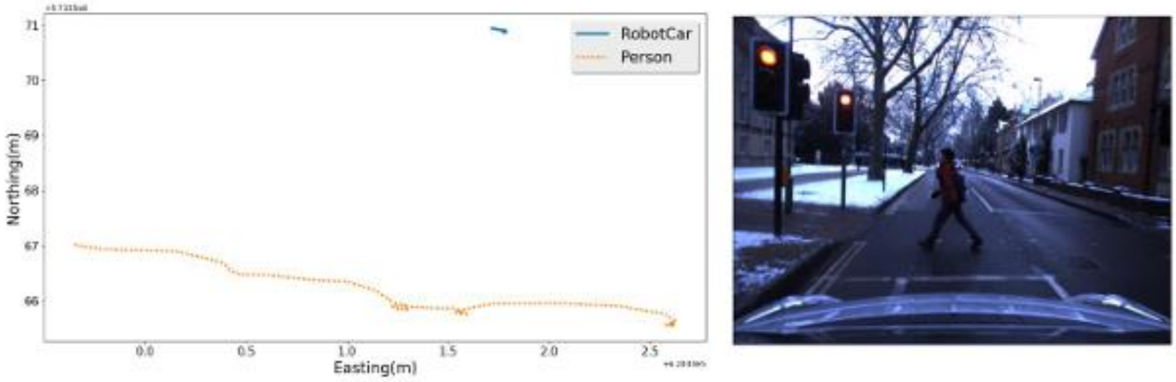

Fig 8.11 Tracking of person when Robot car is stopped in Navigation Frame

Results During the Nighttime

Cameras are not considered to be good sensors for capturing the images at nighttime because of low light. The performance of the YOLO at nighttime for object detection is presented in Figure 8.12 This figure shows the object detected by YOLO in nighttime. The image in left shows how YOLO can detect two buses and a person. However, it missed the bicycle that the person is riding. In the image on

Fig 8.12 Object detection at Night Time by YOLO

the right, it can detect the bus and person in the same image but misses the bus that is quite far from the RobotCar that was previously detected in the left image. The result is still promising given that the bus detected in the right image is blurry. YOLO is trained to detect objects in such conditions. The performance of YOLO for object detection and classification can be considered good even in low light conditions, as it is able to detect the objects that are nearby the car that needs to be tracked. Now the tracking of object during the night time is presented. In this case, a bus riding side by side the RobotCar is tracked. The graph of the tracking in navigation frame is presented below in the Figure 8.13, which demonstrates that, even during the night time, the tracking of object is possible using the algorithm developed in the thesis. Figure 8.13 shows person riding the bike, which is detected by YOLO and is tracked. Unfortunately, in every image frame, it is tracked as a new object.

This is the result of the object’s small size relative to the resolution of the image. In addition, the fact that it is dark contributes to the lack of keypoints to match it from one image to another.

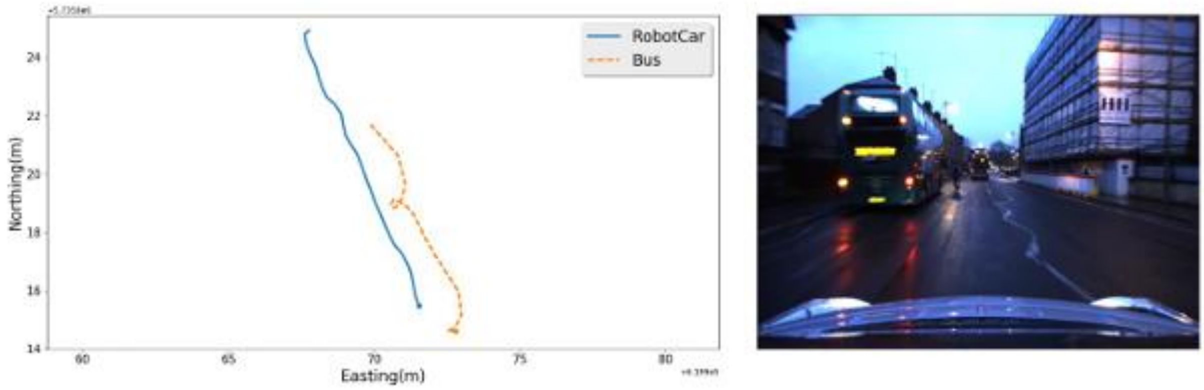

Fig 8.13 Tracking of Bus moving in same direction in Navigation Frame

CHAPTER 9
CONCLUSION

In this paper, an algorithm for object detection, classification, and tracking is presented. Tracking of moving objects for autonomous vehicle is important and is of great importance for the safety and collision avoidance. The data from cameras, LiDAR and INS is fused to achieve this goal. The paper presents how images can be of great source for the detection and classification of the objects around the vehicle. The paper uses the YOLO algorithm to detect and classify the object. This information is then fused with the information from the LiDAR and INS to successfully navigate the RobotCar with tracking and detection of objects. There are number of ways the algorithm developed in this paper can be improved. The paper assumes the objects as the point object and tracks them. This has resulted in a less than optimal performance in tracking as the laser scan for the same object might be from different parts of the same object as objects are moving. Being able to model the motion of the object detected will be great extension to the work carried out in this paper, such as adding velocity states to moving vehicles. LiDAR in this paper is used to get the distance of the object detected. This can be also be achieved using the stereo camera. If such a sensor were added, it could provide additional information, improving the result.

In the thesis, an algorithm for detection, classification and tracking of objects was developed. The algorithm developed was tested in three different road conditions: normal sunny day, night time, and during snowy weather. The objects were detected and classified using the YOLO. The performance of YOLO in different weather condition is presented in the result section. YOLO performs well under all three test conditions, including at night time and during snowy condition. In the results section, it was observed that YOLO missed a few objects during the two more adverse conditions but did not miss objects that were nearby the autonomous car and of importance for navigating safely. Images contain different amounts of information about the environment based on the lighting conditions and this affects the performance of YOLO on object detection and classification. Once the objects are detected and classified by YOLO, they are tracked using EKF given that the same object from one frame to another is matched using ORB. Using EKF, the state of the Robotcar was estimated, aiding in the localization of the RobotCar as well as tracking the position of objects detected in the navigation frame. The algorithm developed in the paper was able to track the multiple moving objects simultaneously and objects that are stationary. This was seen in the example where the algorithm tracked a pedestrian crossing the road when the RobotCar was stopped at a traffic light. There are a number of ways the work developed in the thesis can be extended. The performance of the camera varies in different lighting conditions, and this affects the algorithm developed in this thesis. Using a camera as the main perception sensor for object detection may not be ideal for all weather condition.

The performance of LiDAR remains undegraded at night time, so information from the laser scans can be combined with the images for the detection of the objects. Also matching objects from one from to another frame is performed using images only, this can be improved by adding motion modelling to the objects. The algorithm developed in the thesis is also limited by only tracking moving and stationary objects and does not focus on the mapping the environment around the Robotcar. Being able to map the 43 environment using the information from camera and LiDAR would help the RobotCar to localize in the environment and navigate safely even in places with poor GPS reception The objects were detected and classified using the YOLO. The performance of YOLO in different weather condition is presented in the result section. YOLO performs well under all three test conditions, including at night time and during snowy condition. In the results section, it was observed that YOLO missed a few objects during the two more adverse conditions but did not miss objects that were nearby the autonomous car and of importance for navigating safely

CHAPTER 10

REFERENCES

[1] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," 2015. [Online]. Available: http://arxiv.org/abs/1506.02640.

[2] E. Rublee, V. Rabaud, K. Konolige and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in International Conference on Computer Vision, Barcelona, 2011.

[3] W. Maddern, G. Pascoe, C. Linegar and P. Newman, "1 Year, 1000km: The Oxford RobotCar Dataset," 2017. [Online]. Available: http://robotcar dataset.robots.ox.ac.uk/images/robotcar_ijrr.pdf.

[4] J. &. D.-W. H. Leonard, "Simultaneous Map Building and Localization for an Autonomous Mobile Robot," in IROS, 1991.

[5] W. Chieh-Chih, C. Thorpe and Thrun.S, "Online Simultaneous Localization And Mapping with Detection And Tracking of Moving Objects: Theory and Results from a Ground Vehicale in Crowded Urban Areas," in 2003 IEEE International Conference on Robotics and Automation, Taipei, 2003.

[6] C. Mertz, L. E. Navarro-Serment, R. Maclachlan, P. Rybski, A. Steinfeld, A. Suppé, C. Urmson, N. Vandapel and M. &. T. C. Hebert, "Moving object detection with laser scanners," 2013.

[7] T.-D. Vu., " Vehicle Perception: Localization, Mapping with Detection, Classifcation and Tracking of Moving Objects.," Institut National Polytechnique de Grenoble - INPG, 2009.

[8] O. M. a. P. T. Papageorgiou C. P, "A general framework for object detection," pp. 555-562, 1998.

[9] D. N. a. T. B., "Histograms of oriented gradients for," in Computer Vision and Pattern Recognition, 2005.

[10] D. J. T. M. J. Girshick R., "Rich feature hierarchies for accurate object detection and semantic segmentation," in Computer Vision and Pattern Recognition, 2014.

[11] G. R, "Fast R-CNN," 2015.

[12] L. D.G, "Object recognition from local scale-invariant," in Computer vision, 1999. The proceedings of the seventh IEEE Conference, 1999.

[13] "Feature Matching," [Online]. Available: https://docs.opencv.org/3.0 beta/doc/py_tutorials/py_feature2d/py_matcher/py_matcher.html.

[14] H. L. &. J. C. B. Hofmann-Wellenhof, GPS Theory and Practice, Springer, 2001.

[15] "Convolutional Neural Networks for Visual Recognition," [Online]. Available: http://cs231n.github.io/convolutional-networks/.

[16] "The PASCAL Visual Object Classes Challenge 2007," [Online]. Available: http://host.robots.ox.ac.uk/pascal/VOC/voc2007/.

[17] J. Redmond, "YOLO: Real-Time Object Detection," [Online]. Available: https://pjreddie.com/darknet/yolo/.

[18] "Convolutional Neural Networks for visual Recognition," [Online]. Available: http://cs231n.github.io/convolutional-networks/.

[19] P. Corke, Robotics, vision and control fundamental algorithms in MATLAB, Springer, 2011. [20] I. R. N. &. D. S. R. Siegwart, Introduction to Autonomous Mobile Robots, 2011.

[20] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," 2015. [Online]. Available: http://arxiv.org/abs/1506.02640.

[21] J. &. D.-W. H. Leonard, "Simultaneous Map Building and Localization for an Autonomous Mobile Robot," in IROS, 1991.

[22] W. Chieh-Chih, C. Thorpe and Thrun.S, "Online Simultaneous Localization And Mapping with Detection And Tracking of Moving Objects: Theory and Results from a Ground Vehicale in Crowded Urban Areas," in 2003 IEEE International Conference on Robotics and Automation, Taipei, 2003.

[23] C. Mertz, L. E. Navarro-Serment, R. Maclachlan, P. Rybski, A. Steinfeld, A. Suppé, C. Urmson, N. Vandapel and M. &. T. C. Hebert, "Moving object detection with laser scanners," 2013.