Student Declaration of Authorship

HERIOT WATT UNIVERSITY
UK | DUBAI | MALAYSIA

Course code and name:	F21MP Masters Project and Dissertation
Type of assessment:	**Individual**
Coursework Title:	*Automated Drivers Assistance System with Object Detection and Tracking for Low-Visibility Conditions*
Student Name:	**Aman Haris**
Student ID Number:	**H00409752**

Declaration of authorship. By signing this form:

- **I declare** that the work I have submitted for individual assessment OR the work I have contributed to a group assessment, is entirely my own. I have NOT taken the ideas, writings or inventions of another person and used these as if they were my own. My submission or my contribution to a group submission is expressed in my own words. Any uses made within this work of the ideas, writings or inventions of others, or of any existing sources of information (books, journals, websites, etc.) are properly acknowledged and listed in the references and/or acknowledgements section.

- I confirm that I have read, understood and followed the University's Regulations on plagiarism as published on the University's website, and that I am aware of the penalties that I will face should I not adhere to the University Regulations.

- I confirm that I have read, understood and avoided the different types of plagiarism explained in the University guidance on Academic Integrity and Plagiarism

Student Signature *(type your name):* *Aman Haris*

Date: *18/08/2023*

HERIOT-WATT UNIVERISTY

MASTERS THESIS

Automated Drivers Assistance Systems with Object Detection and Tracking For Low-Visibility Conditions

---

Author:
Aman Haris

Supervisor:
David Corne

A thesis submitted in fulfillment of the requirements
for the degree of MSc.Artificial Intelligence
in the
School of Mathematical and Computer Sciences

August 2023

Declaration of Authorship

I, *Aman Haris*, declare that this thesis titled, '*Automated Drivers Assistance System with Object Detection and Tracking for Low-Visibility Conditions*' and the work presented in it is my own. I confirm that this work submitted for assessment is my own and is expressed in my own words. Any uses made within it of the works of other authors in any form (e.g., ideas, equations, figures, text, tables, programs) are properly acknowledged at any point of their use. A list of the references employed is included.

Signed: [illegible]

Date: 18th August 2023

“He who can listen to the music in the midst of noise can achieve great things.”

Vikram Sarabhai

Abstract

Every year the lives of approximately 1.3 million people die as a result of a road traffic crash. Road traffic injuries cause considerable economic losses to individuals, their families, and to nations as a whole. The World Health Organization (WHO) has been concerned with this issue for over four decades [1].

In low-income and middle-income countries, the phenomenon of pedestrians and vehicles not being properly visible is frequently a serious problem [1]. Traffic collisions in India are a major source of deaths, injuries and property damage every year. During the calendar year 2021, road crashes in India claimed about 1.5 lakh lives and caused injuries to more than 3.8 lakh people. Around 27.43% of these accidents occur due to low visibility [2].

Automated Drivers Assistance System (ADAS) are being actively used in autonomous vehicles for assisting the drivers, then why not use them to improve the visibility of a driver while driving? To avoid the accidents caused by the above said reasons and to help save the lives of both driver and passenger, an ADAS is proposed in this report. The proposed ADAS should be able to detect objects on the road irrespective of the weather and lighting conditions and displays an almost accurate information of the identity and location of the detected object on the road. We shall look into the basic structure of an ADAS and then explore how we can repurpose the existing system to aid our objectives. We shall evaluate the system to see if the system meets our objectives in this report.

Acknowledgements

I would like to thank my supervisor, David Corne for his immense guidance and support while writing this report.

Finally I would like to thank my parents and my sister for their love and support during all times.

Contents

List of Figures

Abbreviations

WHO World Health Organization

ADAS Automated Drivers Assistance System

HUD Heads Up Display

RGB Red Green Blue

IR Infrared Radiation

ML Machine Learning

YOLO You Only Learn One Representation

YOLOv4 You Only Learn One Representation Version 4

FPS Frames Per Second

GPU Graphics Processing Unit

SORT Simple Online and Realtime Tracking

CNN Convulated Neural Network

MS COCO Microsoft Common Objects in Context

OpenCV Open Source Computer Vision

IDE Intergrated Development Environment

GPU Graphical Processing Unit

Chapter 1

Introduction

1.1 Overview

In developing countries such as India, roadways are the major transportation network. Given that India is a tropical country where the weather keeps changing constantly, it poses a major threat to drivers. Weather condition affects road surface condition and the visibility of the motorist, thereby increasing the chances of mishaps. Adverse weather conditions such as heavy rain, thick fog and hail storms make driving riskier as visibility reduces and road surface gets slippery. Even with advancements in technology, there is no specific advanced system to help tackle issues such as bad weather conditions and lighting conditions while driving. Being a rider myself who constantly use motorbike as my main mode of transport to travel around, I have faced this issue many times, where because of lack of good vision, I have to stop riding and wait for a while for the weather to get clear. This in turn wastes a lot of time waiting for an ideal driving weather.

An ADAS which is repurposed to help identify the objects on road during low visibility can help prevent a lot of accidents. This will also improve the drivers' cognitive sense and help them drive safely. My proposed idea should be able to help every driver using different types of vehicles, to drive safely during bad weather and lighting conditions.

1.2 Motivation

Every year, automobile manufactures spend millions in funding development of cutting edge technologies to keep drivers safe and accident free while operating their vehicles. These technologies are known in the industry as Advanced Driver Assistance Systems and each one is controlled by complex real time embedded systems [4]. The necessity of ADAS is [that] driver error will be reduced or even eliminated, and efficiency in the traffic and transport is enhanced. The benefits of an ADAS implementations are potentially considerable because of the significant decrease in human suffering or stress, economical costs and pollution [3]. ADAS aim to support drivers by either providing warning to reduce risk exposures, or automating some of the control tasks to relieve a driver from manual control of a vehicle. ADAS functions can be achieved through an autonomous approach with all instrumentation and intelligence on board the vehicle, or through a cooperative approach, where assistance is provided from roadways and/or from other vehicles [5].

From these we can understand that the future of automobile will consist of an ADAS in one form or another. Every autonomous vehicle does need an ADAS to operate and function correctly. Therefore the system proposed in this report can be integrated to the already existing ADAS system or can be built into a new ADAS unit for any type of vehicles.

1.3 Aim and Objective

Currently the existing ADAS main job is to do driving task assistance. Some examples of the driving task assistance provided by the ADAS are:

- Adaptive Cruise Control (ACC): ACC helps maintain a safe following distance from the vehicle ahead by automatically adjusting the vehicle's speed to match the traffic flow. It reduces the need for constant braking and accelerating in heavy traffic conditions.
- Automatic Emergency Braking (AEB): AEB detects potential collisions with vehicles, pedestrians, or obstacles and automatically applies the brakes to prevent or mitigate a collision.
- Parking Assistance Systems: ADAS can aid in parking by providing visual and/or auditory cues to help the driver maneuver into parking spots more easily.
- Driver Drowsiness Detection: This feature monitors the driver's behavior and provides alerts if signs of drowsiness or inattention are detected, encouraging the driver to take a break and avoid accidents caused by fatigue.
- Adaptive Headlights: These headlights adjust their beam patterns based on steering input and road conditions, providing better visibility during nighttime driving.

Other uses of ADAS are mainly in cars such as Tesla where automatic driving mode is available. There also exists automotive night vision, which are only found in the premium high end vehicles to aid the driver driving at night.

The aim of this project is to build an ADAS whose primary objective is to **improve the way drivers drive during bad weather conditions such as fog and rain and bad lighting conditions such as night time**. We also aim to make this system easy to use and to be able to install on any type of vehicles. In order to do this, an ADAS is developed using object detection and object tracking algorithms. The software takes thermal imagery as input and the IR camera can capture the feed of the road irrespective of weather and lighting conditions, unlike the Red Green Blue(RGB) cameras whose video capture gets affected by bad weather and lighting conditions. The software detects and tracks the various objects that are on the road and displays it on the Heads Up Display(HUD) of the vehicle, helping the driver understand what's on the road, which improves the cognitive sense of the driver and helps to prevent avoidable mistakes. The system is built entirely using open source software and public APIs. The system has the capability to be upgraded and improved easily.

1.3 Report Outline

The first chapter is the introduction and provides an outline of the problem being addressed and the main goal of the thesis. The second chapter will illustrate previous and related works that have been done on the research area and the associated works that will help for the accomplishing of the thesis. It will provide an understanding of ADAS and the technologies used in ADAS. The third chapter will show the various issues regarding the development and implementation of this thesis. The fourth chapter refers to the requirements of the project and the requirements of the application will be built. The fifth chapter will show us the architecture and flowchart of the system to be developed. In sixth chapter we will demonstrate the method of implementation of the system using the guidance from the previous chapters. In seventh chapter the evaluation methodology is discussed. The chapter eight, which is the final, includes the conclusion of this thesis.

1.4 Expected Results and Beneficiaries

This project would result in the development of an ADAS with the following properties:

1. Easy to develop and maintain
2. Easy to use and learn
3. Able to give an highly accurate prediction and information using latest ML algorithms

This system is mainly beneficial to the safety of the drivers and their co-passengers. This system can be installed by the automobile companies as an safety feature for their vehicles.

Chapter 2

Literature Review

2.1 Overview

An extensive literature analysis pertaining to the aim and dissertation is presented in this section. We shall look into the main components that make up an ADAS and get a brief idea on what components we will be needing to build the system. After that, we shall look into the choice of components that we chose, their working and the reason why they were chose to be the part of the system by comparing their performance with other similar products. And finally we shall conclude by looking into how these components come together to act as an ADAS.

2.2 Effect of Low Visibility while Driving

Adverse weather conditions have significant impacts on pavement conditions, vehicle performance, visibility distance, drivers' behavior, travel demand, traffic flow characteristics and traffic safety. Visibility in particular is critical to the task of driving and reduction in visibility due to Fog/Smoke (FS) or Heavy Rain (HR) is a major traffic operation and safety concern. Although, the percentage of FS related crashes is small compared to crashes that occurred at clear visibility conditions, these crashes tend to be more severe and involve multiple vehicles. Thus, there is a need to detect any reduction in visibility and develop ways to convey warnings to drivers in a timely and effective way [22].

Drivers need to be able to see a sufficient length of the roadway ahead in order to (i) avoid hitting a stationary object along the path, (ii) overtake slower vehicles where permitted, and (iii) make appropriate driving decisions at complex locations such as intersections or when coming across traffic diversion signs [23]. A foggy area on a road will reduce a driver's visibility. The visible distance is an important indicator for the driver to judge the needed operation. The visibility reduction caused by heavy fog is the main factor affecting the efficiency of traffic operations and traffic crashes [24].

Thus from the studies stated above we can see that how important it is solve the issue of low visibility while driving.

2.3 Components of the ADAS

In this section we shall briefly look into the main components of an ADAS system. In the upcoming sections we shall dive in detail and understand the type of the component used and why they are used.

2.3.1 Vision Sensors

The different types of sensors used in ADAS is shown in Figure 2.2.1. From those sensors we shall be focusing on camera sensors. Cameras are the most commonly used vision sensors in vehicles. Vision-based ADAS uses one or more cameras to capture images and an embedded system to detect, analyze, and track different objects in them. Cameras capture information such as color, contrast, and texture, which gives them a unique advantage over other sensors. Two types of cameras are often used in vision-based ADAS: 1) monocular and 2) stereo [6] (pg2 p6).

Camera sensors usually capture either RGB or Infrared wavelengths. For this project we are focusing on using IR cameras to capture the feed of the road.

There are two main types of IR cameras. Active IR cameras use a near-IR light source (with wavelengths from 750 to 1,400 nm) built in the vehicle to illuminate the scene (which cannot be seen by the human eye) and a standard digital camera sensor to capture the reflected light. Passive IR cameras use an IR sensor, where every pixel on the IR sensor can be considered as a temperature sensor that can capture the thermal radiation emitted by any material. Unlike active IR cameras, passive IR cameras do not require any special illumination of the scene. Still, popular night-vision solutions mainly use active IR cameras to assist the driver by displaying video data on a screen during low light conditions [6] (pg3 p3).

The reason why we chose to use IR sensors over RGB sensors is that, the video capturing ability of the IR sensors are independent of outside factors such as illumination, weather and visibility. As these feature coincides with the objectives of the project we will be using IR sensors.

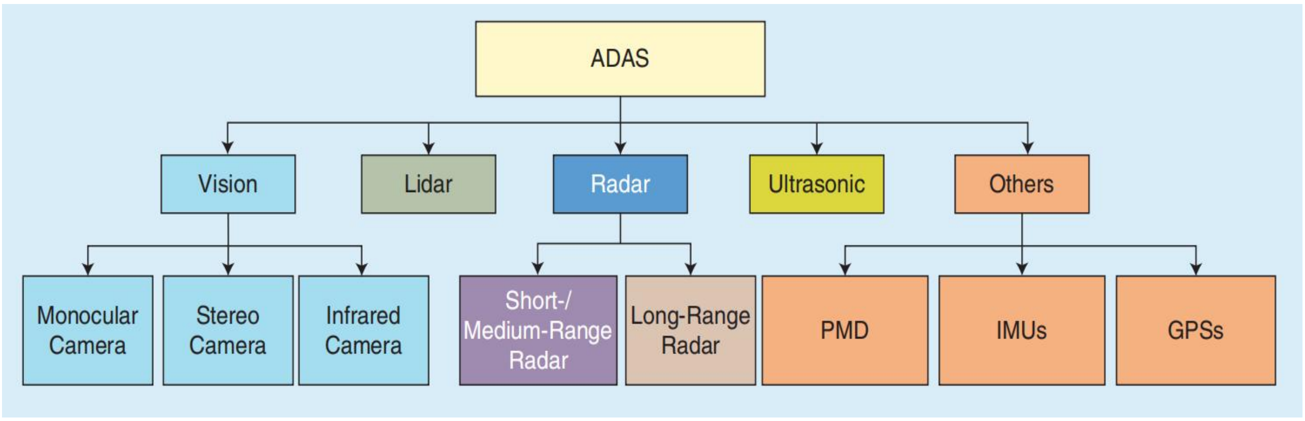

Figure 2.3.1: The taxonomy of an ADAS [6].

2.3.2 Computer Vision Data Flow for ADASs

In this section we are going to look into computer vision data flow. Figure 2.2.2 shows a typical look of the flowchart in an ADAS. For this project we'll be excluding depth estimation and system control as they are not part of the objectives of the project.

Image Acquisition refers to the process of capturing a frame from a video. The frame is often represented as a matrix of pixel data where each frame contains three channels of information, e.g., red, green, and blue (RGB) sets of pixels. Typical frame rates in ADASs range from five frames per second (fps) to 60 fps depending on the application [6] (pg4 p7).

Object Detection and Tracking is the process of classifying an object in an image (e.g., determining if an object ahead is a vehicle, sign, or pedestrian) and predicting its movement. It is often accomplished with various machine-learning (ML) algorithms. ML algorithms are provided large training data sets (thousands of images) to learn and differentiate between vehicles and common objects found around them [6] (pg5 p2).

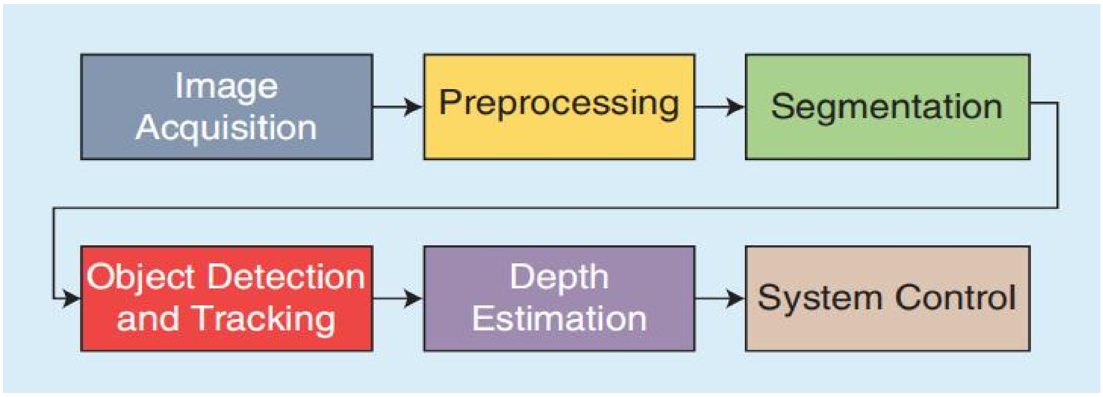

Figure 2.3.2: The vision data flow for the ADAS used [6].

2.4 Algorithms

In this section we shall look into the algorithms chosen to act as object detector and object tracker for our project, along with comparisons with other similar algorithms to understand why we choose this particular algorithm.

2.4.1 Object Detector

While choosing an object detector for this project, we had two main criteria: Should be able to detect objects in real-time and should be easy and simple to build, train and upgrade the algorithm.

Wang et al proposed a unified network to integrate implicit knowledge and explicit knowledge, and enable the learned model to contain a general representation, and this general representation enable sub-representations suitable for various tasks. Figure 2.3.1(a) illustrates the proposed unified network architecture. The proposed network effectively improves the performance of the model with a very small amount of additional cost (less than one ten thousand of the amount of parameters and calculations) [17]. The above research established the basis of a new and fast object detecting algorithm called as "You Only Learn One Representation" (YOLO).

This formed the basis of a series of high speed and highly accurate object detectors called as the YOLO. Various versions of YOLO has been released of which YOLOv4 has shown to give the best performance while running in real time.

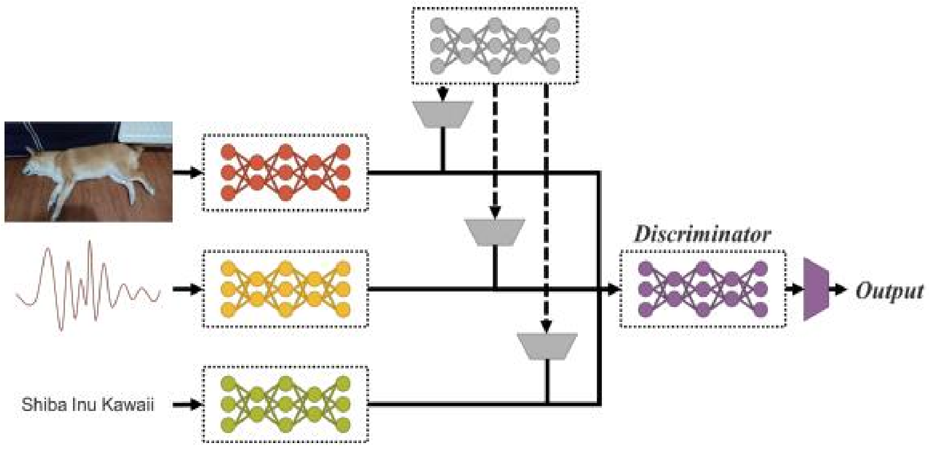

Figure 2.4.1(a): Multimodal Unified Network [17].

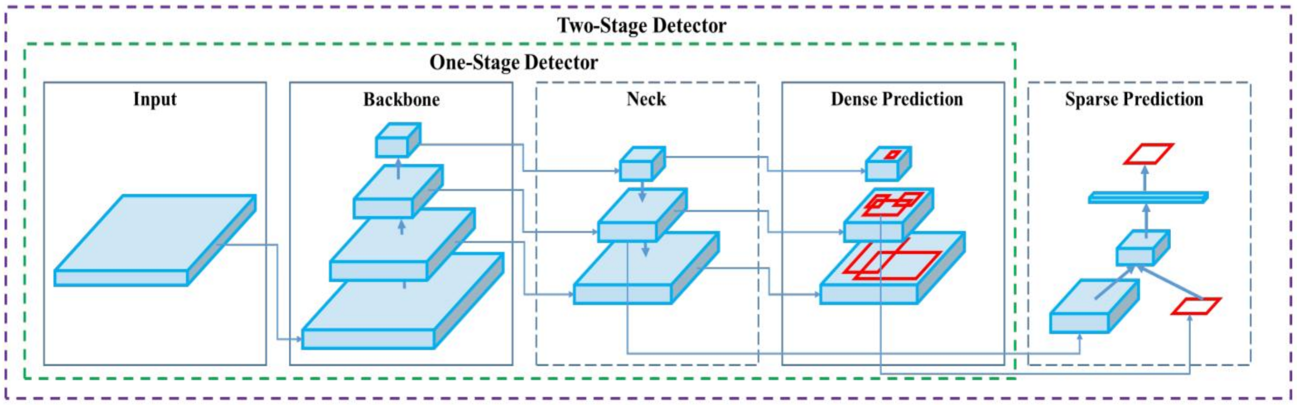

Figure 2.4.1(b): Architecture of YoloV4 Object Detector [18].

Bochkovskiy et al's YOLOv4 are located on the Pareto optimality curve and are superior to the fastest and most accurate detectors in terms of both speed and accuracy. They offer a state-of-the-art detector which is faster (FPS) and more accurate (MS COCO AP50...95 and AP50) than all available alternative detectors. The detector described can be trained and used on a conventional GPU with 8-16 GB-VRAM this makes its broad use possible. The original concept of one-stage anchor-based detectors has proven its viability. [18] Figure 2.3.1(b) shows the architecture of the YoloV4.

We can see the comparison graph of YoloV4 to other existing object detectors in Figure 2.3.1(c). YoloV4 runs much faster with a better range of accuracy than compared to other detectors. Even though we lose accuracy as the frame rates increase, for a real time system such as ADAS we need faster processing detector over highly accurate slower running detectors.

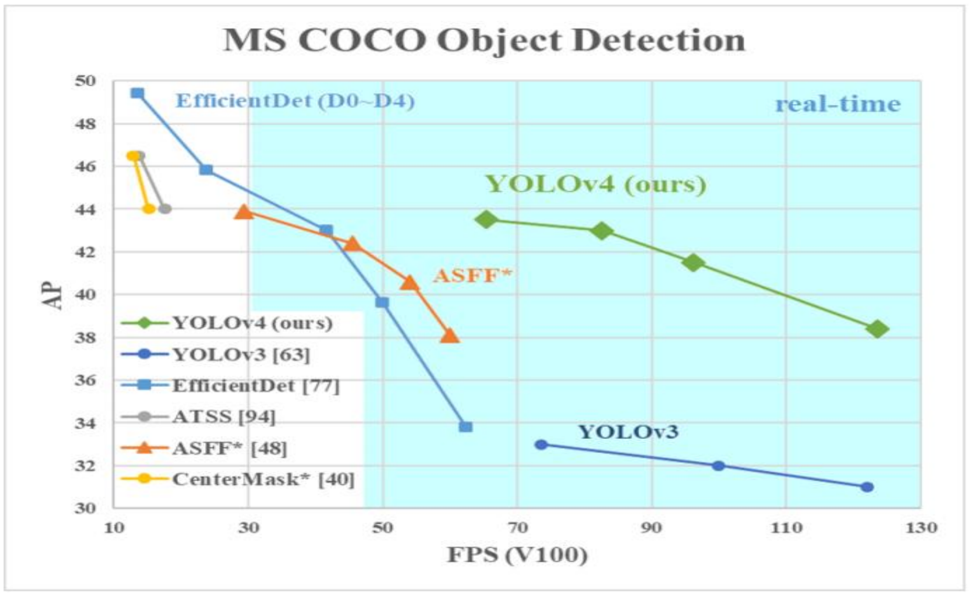

Figure 2.4.1(c): Comparison of the proposed YOLOv4 and other state-of-the-art object detectors.

2.4.2 Object Tracker:

While choosing an object tracker for this project, we had two main criteria: it should track the object accurately even when an occlusion happens and it should be easy to setup and integrate with our object detector.

Simple Online and Realtime Tracking (SORT) is a pragmatic approach to multiple object tracking with a focus on simple, effective algorithms. In this paper, Wojke et al integrate appearance information to improve the performance of SORT. Due to this extension they are able to track objects through longer periods of occlusions, effectively reducing the number of identity switches. SORT is a much simpler framework that performs Kalman filtering in image space and frame-by-frame data association using the Hungarian method with an association metric that measures bounding box overlap. This simple approach achieves favorable performance at high frame rates. SORT has a deficiency in tracking through occlusions as they typically appear in frontal-view camera scenes. They overcome this issue by replacing the association metric with a more informed metric that combines motion and appearance information. They adopt a conventional single hypothesis tracking methodology with recursive Kalman filtering and frame-by-frame data association. DeepSort algorithm is usually paired up with an object detector such as YOLO to get the bounding boxes. Figure 2.3.2 shows a typical architecture of a DeepSort algorithm [7].

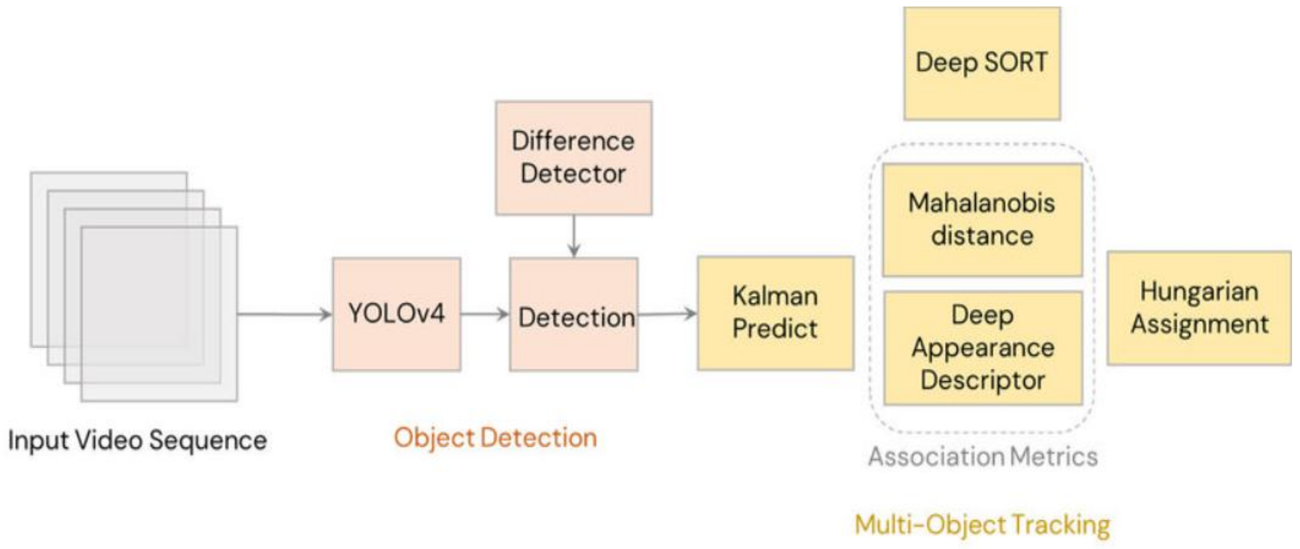

Figure 2.4.2: Architecture of DeepSort in conjunction with YoloV4 [19].

Wojke et al have presented an extension to SORT that incorporates appearance information through a pre-trained association metric. Due to this extension, they are able to track through longer periods of occlusion, making

SORT a strong competitor to state-of-the-art online tracking algorithms. Yet, the algorithm remains simple to implement and runs in real time [7].

As we can see from the above discussion DeepSort algorithm seemingly fits our criteria. It can handle occlusions exceptionally and can give highly accurate result in real time.

2.5 Datasets

To train our models, we needed a very big dataset which contains various classes in it. We also wanted to dataset to be accurate and segmented properly, so that we can do the training fastly and easily. We ideally wanted a thermal imagery dataset, but since none of the existing datasets matched our criteria, we looked into RGB dataset to train our models.

Lin et al present a new dataset with the goal of advancing the state-of-the-art in object recognition by placing the question of object recognition in the context of the broader question of scene understanding. This is achieved by gathering images of complex everyday scenes containing common objects in their natural context. Objects are labeled using per-instance segmentations to aid in precise object localization. Our dataset contains photos of 91 objects types that would be easily recognizable by a 4 year old. With a total of 2.5 million labeled instances in 328k images, the creation of our dataset drew upon extensive crowd worker involvement via novel user interfaces for category detection, instance spotting and instance segmentation.

Lin et al harvested a large set of images containing contextual relationships and non iconic object views. Next, each image was labeled as containing particular object categories using a hierarchical labeling approach. For each category found, the individual instances were labeled, verified, and finally segmented. The Microsoft Common Objects in COntext (MS COCO) dataset contains 91 common object categories with 82 of them having more than 5,000 labeled instances. In total the dataset has 2,500,000 labeled instances in 328,000 images. A critical distinction between our dataset and others is the number of labeled instances per image which may aid in learning contextual information. MS COCO contains considerably more object instances per image as compared to ImageNet and PASCAL.

Lin et al may roughly group images into three types : iconic-object images, iconic-scene images and non-iconic images, as shown in Figure 2.3.3. Their goal was to collect a dataset such that a majority of images are non-iconic. It has been shown that datasets containing more non-iconic images are better at generalizing. [8]

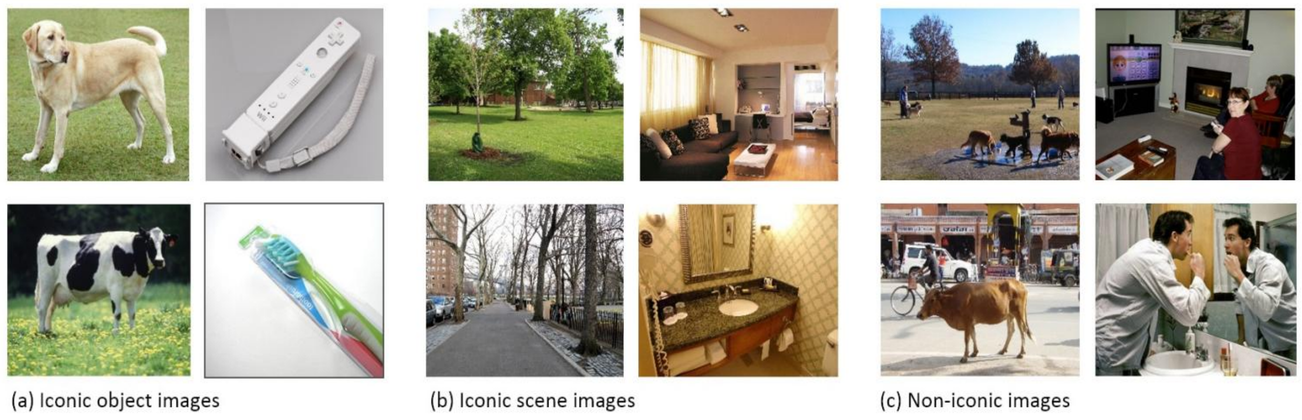

Figure 2.5: Example of (a) iconic object images, (b) iconic scene images, and (c) non-iconic images.

2.6 Hardware

2.6.1 IR Camera

An infrared camera is a device that creates an image using infrared (IR) radiation, similar to a normal camera that forms an image using visible light. Instead of the 400–700 nanometre (nm) range of the visible light camera, infrared cameras are sensitive to wavelengths from about 1,000 nm (1 micrometre or μm) to about 14,000 nm (14 μm).

A IR camera outputs pixel values which represent heat (temperature), and the output is gray-scale images. Since the thermal cameras do not depend on whether there is the light or not unlike other visible range cameras, object detection using the thermal camera is reliable without ambient surrounding.[9]

The reason why chose IR cameras is because, we need to capture the feed of the road irrespective of the weather and lighting condition. A typical RGB camera video capturing quality gets affected by the above said conditions, whereas irrespective of the nature, IR cameras can capture the objects on the road because it captures infrared radiation of an object, which is the goal of this project.

2.6.2 HUD

Fixed displays will not grab the driver’s attention if he or she is looking away. Auditory (and haptic) displays may not be able to convey the same amount of information quickly and succinctly. Drivers may already know about impending danger, and they may be annoyed by systems that provide redundant warnings. Sidescreen displays might take the driver’s attention off the road. Several cars these days also use heads-up displays (HUDs) to convey information to the driver. As a subset of visual interfaces, HUDs are designed to present information to the driver, which is closer to his or her field of view. Figure 2.4.2 shows how HUD work in low visibility condition and put up useful information for the driver. Thus, the driver does not have to look down to see the information but can spend more time looking at the road. By presenting information that is closer to the normal field of view, HUDs require less effort on the part of the driver than other kinds of visual displays; therefore, it may present an effective choice of conveying visual information [10].

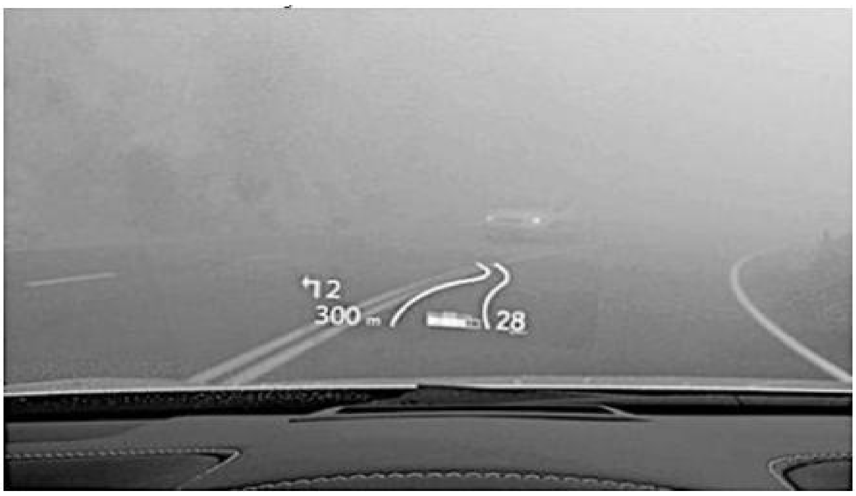

Figure 2.6.2: HUD in low visibility environment [20]

It has been shown that a HUD significantly helps a driver to recognize both the objects in the foreground and displayed information concerning vehicle operating conditions [11].

HUD can reduce the frequency and duration of the driver's eyes-off-the-road by projecting the required information in front of the driver. This enables the driver to steer easily and to respond quickly to information provided by the road conditions and communication systems. Given the advantages that it can help reduce the driver's need to shift attention from the road in front to the related visual display interface and the time needed for re-accommodation, HUD seems to be a feasible and superior alternative or auxiliary visual display interface.

To sum up briefly, use of HUD can enable drivers to respond faster to unanticipated road events (i.e. speed-limit detection and response tasks designed for this study) under both low and high driving loads. Furthermore, under low driving load, drivers have improved driving behaviours as evidenced by smaller variances in lateral acceleration and steering wheel angle. These valid indicators of required attention for driving show that drivers need to pay less attention when using the HUD.[12]

2.7 Important Python Modules

2.7.1 OpenCV

OpenCV is widely used in any projects involving computer vision. OpenCV supports a lot of algorithms related to Computer Vision and Machine Learning and it is expanding day-by-day. Also, interfaces based on CUDA and OpenCL are also under active development for high-speed GPU operations. OpenCV-Python is the Python API of OpenCV. So OpenCV-Python is an appropriate tool for fast prototyping of computer vision problems [13].

OpenCV contains many modules including module for image processing, for identification of object, and ML. By the use of it, we achieve, constrict, build up, replace, retrieve information [14].

As our project involves computer vision processes such as displaying symbols and text, OpenCV is an excellent module to achieve these needs.

2.7.2 Tensorflow

TensorFlow is a machine learning system that operates at large scale and in heterogeneous environments. TensorFlow supports a variety of applications, but it particularly targets training and inference with deep neural networks. It serves as a platform for research and for deploying machine learning systems across many areas, such as speech recognition, computer vision, robotics, information retrieval, and natural language processing [15]. Tensorflow allows using custom models by training custom data-sets [16].

We have used Tensorflow as the foundation to build and deploy our algorithms. The training of these algorithms are also done using Tensorflow packages. The advantages of using Tensorflow are that it is platform independent and it is easy to scale. It is also easier to creates pipelines between models.

2.8 Conclusion

In conclusion, we saw what are the different technologies that make up an ADAS and it reveals the significant potential for these technologies to improve road safety and enhance the driving experience. The literature on the components of ADAS has highlighted the critical role that each of these components plays in ensuring the safe and effective functioning of these systems. However, the implementation of ADAS is not without challenges. The reliability and accuracy of these systems can be impacted by various factors. Furthermore, the integration of ADAS into existing vehicles and infrastructure can be costly, and the regulatory framework for these systems is still evolving. To fully realize the potential of ADAS, ongoing research and development are necessary to improve their performance, reliability, and accessibility. Overall, the literature suggests that ADAS technologies hold great promise for improving road safety and enhancing the driving experience, and that continued investment and collaboration will be necessary to fully realize these benefits.

Chapter 3

Professional, Legal, Ethical and Social Issues

3.1 Professional and Legal Issue

All papers, code, or libraries will be referenced and used under the terms of the publisher licenses. The software produced will be tested, and documented according to the professional software engineering practices. The produced software will be published under the GNU GENERAL PUBLIC LICENSE [42]. The GNU license allows any user to modify, share, and distribute the code given that they make the source code available under the same license. They also must give a contribution to the original author.

This project will be done in the Anaconda Environment using Spyder IDE. The project will follow and respect the British Computer Society (BSC) Code of Conduct rules. The dataset will be used under Data Protection Law Policy. Any third party libraries, software or any other products will be used only if permitted by their licence. Any citations or external information will be referenced as appropriate.

The base code of this project has been taken from the Github repository of theAIGuysCode which has been licensed under GNU General Public License v3.0 [21].

3.2 Ethical Issue

One of the main ethical issue faced by an ADAS is determining the responsibility and liability in cases of accidents. Even though our ADAS does not directly involve in driving, it does give additional information to the driver while driving and if in case the information is wrong, it may lead to accidents. Another issue to be considered is the human-machine interaction while driving and whether it is advantageous and safe to have ADAS assisting while driving.

Another major issue is the potential for over reliance on technology, as drivers might become complacent and less attentive, assuming the ADAS can handle the challenging driving situations. This false sense of security may lead to risky behavior or delayed reactions when the system encounters limitations or unexpected conditions. Additionally, as the ADAS system rely on sensors and machine learning algorithms could introduce biases and discrimination, prioritizing certain objects or pedestrians over others. Data privacy and security are also crucial, as ADAS might collect sensitive information, necessitating measures to protect user data from misuse or unauthorized access. Ensuring equal access to advanced ADAS technologies and addressing unintended consequences, such as altering driver behavior during low visibility conditions, are essential aspects of ethical ADAS design. To navigate these ethical complexities, transparent design, responsible guidelines, ongoing monitoring, public education, and effective regulations are necessary to ensure the safety and privacy of drivers and road users alike.

3.3 Social Issue

The widespread implementation of ADAS to address low visibility can potentially exacerbate existing social issues in our transportation landscape. One major concern is the digital divide, as access to these advanced technologies might not be equally available to all members of society. Low-income individuals and marginalized communities may face challenges in affording or accessing vehicles equipped with ADAS, further deepening existing transportation inequalities. Additionally, the reliance on ADAS could potentially lead to a decline in driving skills among some individuals, as they may become overly dependent on the technology, resulting in reduced competence in handling challenging situations when ADAS is unavailable or experiences malfunctions. Addressing these social issues requires proactive efforts to ensure equitable access to ADAS technologies and public awareness campaigns to educate users about responsible usage and the importance of maintaining essential driving skills. By taking these considerations into account, society can strive to maximize the benefits of ADAS while mitigating potential adverse social impacts.

Chapter 4

Requirement Analysis

4.1 Project Objectives

The project's objective is to develop an ADAS which would help the driver to drive during bad weather and lighting conditions, by helping to identify whats on the road while driving. We also have added extra feature such as showing present weather conditions, which gives additional information to the driver. The following points explains the objectives in detail.

- The system captures the live feed through the camera and displays the processed video on a HUD of the vehicle.
- The thermal camera captures a live infrared feed of the road and passes it to the YOLOv4 algorithm for object detection.
- YOLOv4 identifies and labels the detected object and passes it to the DeepSort algorithm for object tracking.
- DeepSort gives the object an ID and tracks it during its entire time in the frame. The neural networks are built using Tensorflow.This output is then displayed on a HUD using OpenCV library.
- Corresponding symbols are used to indicate the identity and location of the detected object.
- The system also displays current weather conditions, temperature, and time.

4.2 Requirements

In this section we shall look into the essential requirements to build the system which will help us to build a system according to the project objectives.

4.2.1 Hardware

- Processor: Intel i5 8th Gen (min)
- Graphics Card: NVIDIA GeForce GTX 1050Ti (min)
- RAM: 8 GB (min)
- Hard Disk: 50 GB (min)

4.2.2 Software

- Operating System: Windows 10 (min)
- Programming Language: Python 3.9.12
- IDE: Anaconda Distribution - Spyder IDE

4.2.3 Functional

- The system should be able to identify various objects found on the road.
- The system should track the identified object.
- The system should display identified and tracked object on the screen.
- The system should start on hearing the voice command.
- The system should display time and weather details on the screen.

4.2.4 Non-Functional

- **Speed**

The system should work in real-time without any lags or delays.

- **Accuracy**

The system should try to achieve maximum accuracy while identifying and tracking the objects.

- **Usability**

The user must find the system easy to use and understand. The output must also be easy to understand.

- **System Stability and Performance**

The system must be able to handle and process real-time data without crashing or lowering its performance rate.

Chapter 5

Architecture and Flowchart

5.1 Overview

In this section we shall look into how the system and software architectures are designed using the modules introduced in Chapter 2. We shall also look into the flowchart which shows various processes expected to be in the system. It is important to have an initial design to act as an guide before developing the system.

5.2 System Architecture

The architectural configuration procedure is concerned with building up a fundamental basic system for a framework. It includes recognizing the real parts of the framework and interchanges between these segments. The beginning configuration procedure of recognizing these subsystems and building up a structure for subsystem control and correspondence is called construction modeling outline and the yield of this outlined procedure is a portrayal of the product structural planning. The proposed architecture for this system is shown in Figure 4.2.

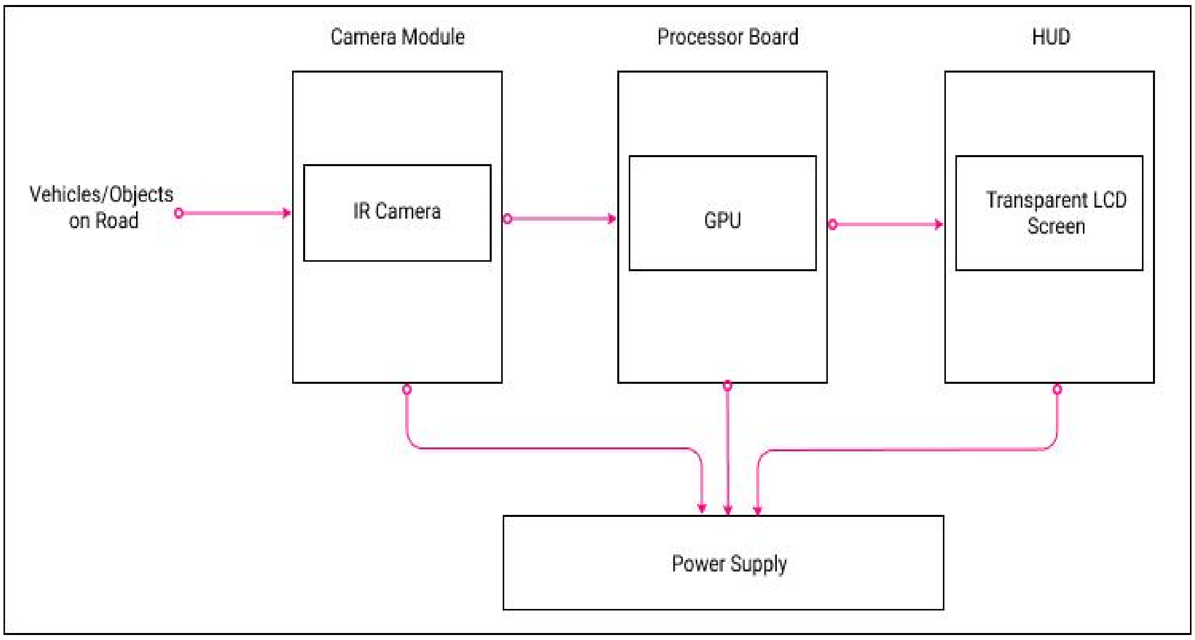

Figure 5.2: Diagram of the proposed system architecture

The first block is called as the camera module, this contains the IR camera. This block captures the video of the road and then passes it onto the next block. The next block is called the processor board. It contains the GPU which stores the software and processes on the video feed received. It the passes on the processed information to the HUD which shows the information on a transparent LCD screen.

5.3 Software Architecture

Two algorithms are used to build the backbone software of this system.

YOLOv4 is a two-stage detector with several components to it. The first stage detector consists of input, backbone, neck, and dense prediction. This stage is connected to a sparse prediction block. Input takes images, patches, or pyramids. We can see that YOLOv4 can be implemented in any combination of input, backbone, neck, and head.

DeepSort is a machine learning model for tracking people, and assigning IDs to each person. Using the bounding boxes detected by YOLOv4, we can assign an ID and track a person by mapping bounding boxes of similar size and similar motion in the previous and following frames. In DeepSort, the process is as follows:

- Compute bounding boxes using YOLO v4 (detections)
- Use Sort (Kalman filter)and ReID (identification model) to link bounding boxes and tracks
- If no link can be made, a new ID is assigned and it is newly added to tracks.

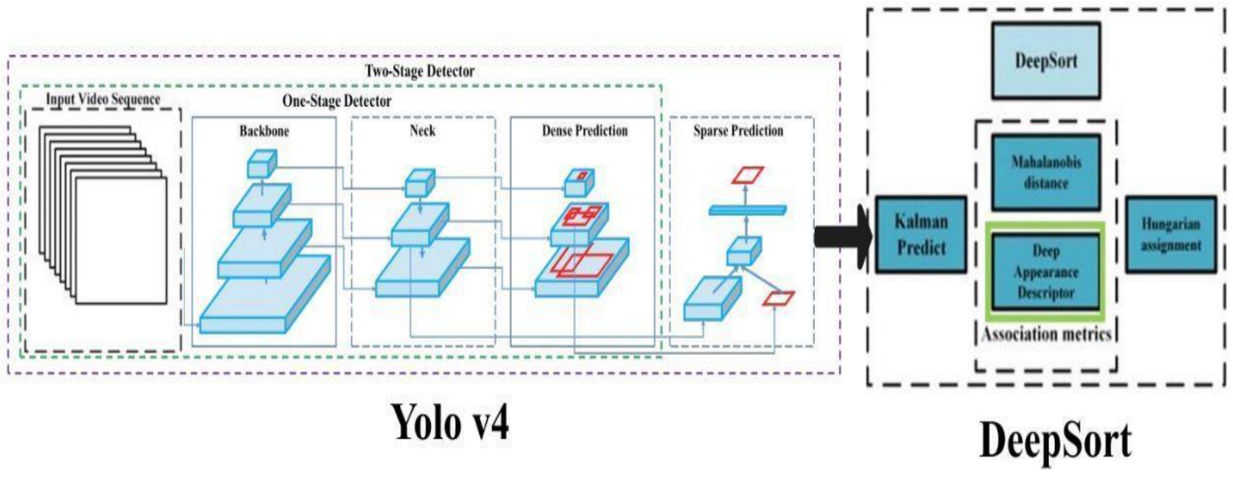

Figure 5.3: Software Architecture

Figure 4.3 shows the software architecture of the system. As we can see, input video sequence is first passed on to YOLOv4, which detects and identifies the object. The bounding box received from this detection is then passed on to the DeepSort algorithm for object tracking.

5.4 Flowchart

A flowchart is a diagram that depicts a process, system or computer algorithm. In this section we shall look into the flowchart of the processes of our system. Figure 4.5 shows the flowchart of this system.

The application will start with a start command, this application is integrated with voice command. Once the product or device is turned on, with voice input like, “START” the application will get started. Once the application is started, the camera starts to record the video in live to get the feed from road. This feed is given to detection and classification of object algorithm for detection and classification. Then the objects are tracked using a algorithm names DeepSort algorithm, once the objects get detected, classified and tracked, we will replace the bounding box with a symbol in order to make the interface hassle free for the drivers. In addition, there is a well designed interface, that shows the weather condition and time at the top corners of the display. Where we are extracting this information from the public API using python programming. Once all the data is gathered, the output is displayed in the HUD display. With the “STOP” voice command the application will get terminated, else it can be done manually.

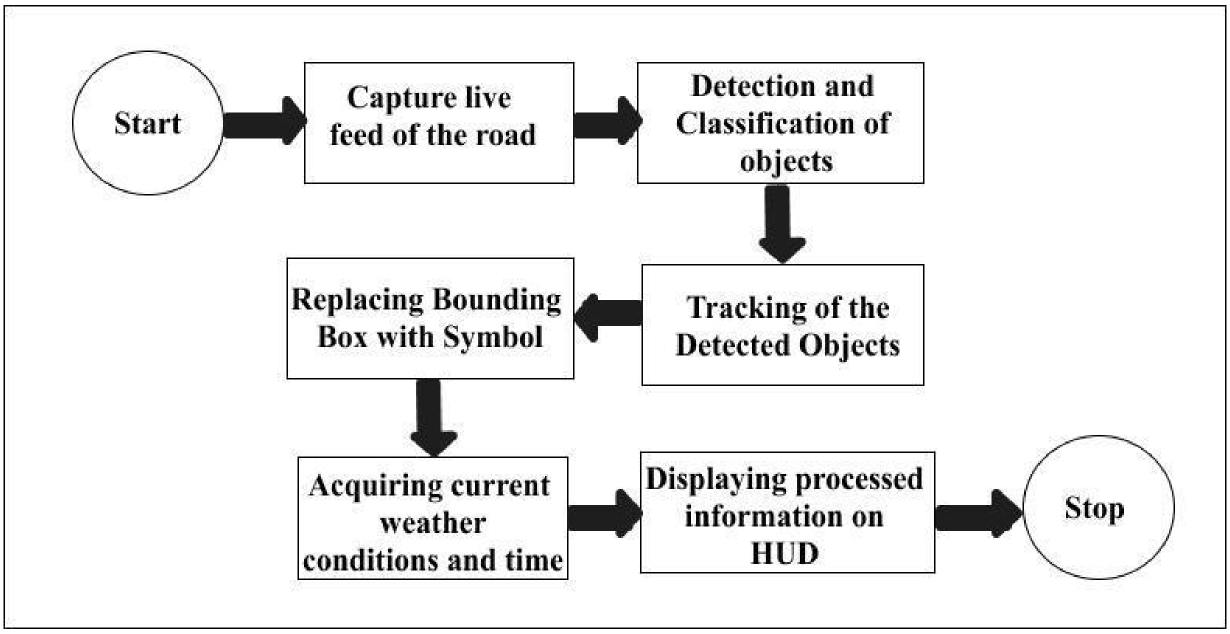
Start
Capture live feed of the road
Detection and Classification of objects
Tracking of the Detected Objects
Replacing Bounding Box with Symbol
Acquiring current weather conditions and time
Displaying processed information on HUD
Stop

Figure 5.4: System Flowchart

Chapter 6

Implementation

6.1 Overview

In this chapter we shall look into how the system is developed. We shall be using Python to code the softwares of the system. We will be using Spyder IDE which is available on the Anaconda platform to build the system. We shall now look into the detailed implementation and development of the system in the following sections.

6.2 Installing Dependencies

To get started we need to first setup an environment in Anaconda and install required dependencies. The main dependencies required to run the system are given below.

- TensorFlow: TensorFlow is an open-source machine learning library developed by Google, widely used for building and training artificial neural networks. It provides a flexible framework for numerical computation using data flow graphs, where nodes represent mathematical operations and edges represent data arrays called tensors.

- OpenCV: OpenCV (Open Source Computer Vision Library) is an open-source computer vision and machine learning software library. It provides a rich set of tools and functions that enable developers to perform various image and video processing tasks, including object detection, face recognition, and image filtering.

- Lxml: lxml is a Python library that provides a fast and easy-to-use interface for parsing and manipulating XML and HTML documents. It is widely used for web scraping, data extraction, and XML processing tasks.

- Tqdm: tqdm is a Python library that provides a fast and extensible progress bar to visualize the progress of iterations and tasks in loops or iterable

- processes. It offers a simple way to add informative and interactive progress tracking to Python scripts and programs.

- Absl-py: absl-py is a Python library developed by Google that provides a set of utility functions and annotations to simplify and enhance code readability, maintainability, and testability. It includes features like flags, logging, and testing utilities, commonly used in Google's Python projects.

- Matplotlib: Matplotlib is a Python library for creating static, interactive, and high-quality visualizations, including charts, plots, and graphs, to display data in a clear and visually appealing manner. It is widely used for data exploration, analysis, and presentation in various scientific and data-driven applications.

- Easydict: EasyDict is a Python library that allows users to create nested dictionaries using both dot notation and dictionary-style syntax, making it easier to work with and access nested data structures in a more readable and intuitive manner. It simplifies dictionary manipulation and navigation, particularly in scenarios involving complex configurations or settings.

- Pillow: Pillow is a Python imaging library that provides extensive support for opening, manipulating, and saving various image file formats. It allows developers to perform a wide range of image processing tasks, including resizing, cropping, filtering, and color adjustments.

- Os: os (Operating System Interface) is a Python built-in module that provides a platform-independent way to interact with the operating system, allowing users to access files, directories, and system information, and execute various system-related tasks. It serves as a bridge between Python programs and the underlying operating system, enabling system-level interactions and operations.

The other packages that are essential to add some additional functionality for the system are given below.

- Datetime: datetime is a Python module that provides classes and functions for working with dates, times, and time intervals. It allows users to manipulate and format dates, perform date arithmetic, and work with time zones, providing comprehensive support for various time-related operations.

- Pyowm: PyOWM is a Python wrapper library for the OpenWeatherMap API, allowing users to easily access weather data, forecasts, and historical weather information for various locations worldwide. It simplifies weather data retrieval and integration into Python applications, making it useful for weather-related applications and projects.

- Pyttsx3: pyttsx3 is a Python library that enables text-to-speech conversion, allowing users to synthesize and play natural-sounding speech using various voice engines in Python applications. It provides a simple interface for adding speech capabilities to projects, making it useful for accessibility, voice-based interfaces, and other speech-related tasks.

- Speech_recognition: pyttsx3 is a Python library that enables text-to-speech conversion, allowing users to synthesize and play natural-sounding speech using various voice engines in Python applications. It provides a simple interface for adding speech capabilities to projects, making it useful for accessibility, voice-based interfaces, and other speech-related tasks.

6.3 Setting up GPU

We will be using GPU to run our softwares for the system. Using a GPU instead of a CPU to run our software offers significant advantages due to the parallel processing capabilities of GPUs such as faster processing, improved performance, large model support, training speed and cost-effectiveness. But the main advantage of using GPU is that it supports in developing real-time applications, which is an essential feature of our system.

We shall install the latest CUDA Toolkit to ensure the smoothing running of our software using the GPU. The CUDA Toolkit is a software development kit provided by NVIDIA for programming GPUs using the CUDA (Compute Unified Device Architecture) platform. It includes a set of libraries, compilers, and tools that enable developers to harness the parallel processing capabilities of NVIDIA GPUs for high-performance computing tasks, such as machine learning, scientific simulations, and data processing. The CUDA Toolkit allows developers to write and optimize GPU-accelerated applications using the CUDA programming model, which significantly improves performance compared to traditional CPU-based processing.

6.4 Training Models

The object tracker in our system uses YOLOv4 to make the object detections, which then DeepSort uses to track. There exists an official pre-trained YOLOv4 object detector model that is able to detect 80 classes. The model is trained against MS COCO dataset. We can also use yolov4-tiny.weights, a smaller model that is faster at running detections but less accurate. The models are in darknet style and therefore we need to convert it into tensorflow model so that we can run them in our system. The code which is given below, is executed in the Anaconda Prompt.

```
python save_model.py --model yolov4
```

6.5 System Software

In this section we shall look into some important parts of the system software. We shall look how YoloV4 and DeepSort algorithms are implemented and pipe-lined and how essential informations are extracted.

The code given below shows how important modules required to run YoloV4 are called into the main program:

```
import core.utils as utils
from core.yolov4 import filter_boxes
from tensorflow.python.saved_model import tag_constants
from core.config import cfg
```

The code given below shows how important modules required to run DeepSort are called into the main program:

```
from deep_sort import preprocessing, nn_matching
from deep_sort.detection import Detection
from deep_sort.tracker import Tracker
from tools import generate_detections as gdet
```

The code given below shows how the system captures video input from the sensors using OpenCV:

```
try:
    vid = cv2.VideoCapture(int(video_path))
  except:
    vid = cv2.VideoCapture(video_path)
```

The code given below shows how the program is set to filter out detect only the required classes for this project from the set of 80 classes in the trained model introduced in the above section. We have decided to include only a vehicles, humans and certain animals which can be found on the road to be detected by the system.

```
allowed_classes = ['person', 'bicycle', 'car', 'motorcycle', 'bus', 'train', 'truck',
'boat', 'traffic light', 'stop sign', 'parking meter', 'bench', 'bird', 'cat', 'dog',
'horse', 'sheep', 'cow', 'elephant']
```

The code given below shows how we feed the detected objects from the YoloV4 to DeepSort for the tracking of the detected object.

```
features = encoder(frame, bboxes)
detections = [Detection(bbox, score, class_name, feature) for bbox, score,
class_name, feature in zip(bboxes, scores, names, features)]
```

The code given below shows how we changed the output of our system to show symbols for the detected object instead of the standard bounding box which is found in object detector.

```
color = colors[int(track.track_id) % len(colors)]
color = [i * 255 for i in color]
cv2.putText(frame, class_name + "-" + str(track.track_id),(int(bbox[0]),
int(bbox[1]-10)),0, 0.75, (255,255,255),2)

#add symbol
path = './symbol/'+class_name+'.png'
symbol = cv2.imread(path, cv2.IMREAD_UNCHANGED)
w = 75
h = 75
size = (w,h)
symbol = cv2.resize(symbol,size)
x = int(bbox[0])
y = int(bbox[1])
```

```
frame = addsymbol.overlay_transparent(frame,symbol,x,y)
```

The code given below shows how we get the date, time and weather information of the user by accessing the accurate location of the user.

```
 cd,ct= dateweather.dttm()
 temp = str(dateweather.temp())
 status = str(dateweather.status())
 edge = int(frame.shape[1])
 cv2.putText(frame, cd, (0,50), 0, 0.75, (255,0,0), 2)
 cv2.putText(frame, ct, (0,100), 0, 0.75, (255,0,0), 2 )
 cv2.putText(frame, temp, (edge-200,50), 0, 0.75, (255,0,0), 2)
 cv2.putText(frame, status, (edge-200,100), 0, 0.75, (255,0,0), 2)
```

The main() class of the program is given below:

```
if __name__ == '__main__':
    try:
        app.run(dot)
    except SystemExit:
        pass
```

Chapter 7

Evaluation

7.1 Overview

Evaluation in a project is a critical component of project management that involves assessing the progress, performance, and outcomes of a project. It includes measuring the achievement of project goals and objectives, evaluating the efficiency of resource utilization, assessing the quality of deliverables, and analyzing the overall project performance against established criteria or benchmarks. Project evaluation is essential for enhancing project effectiveness, ensuring accountability, and maximizing project success. Evaluation for this project will be done on a formative and summative basis. Formative evaluation is an ongoing, iterative process that involves gathering feedback and data throughout the project's implementation to inform and improve its design and effectiveness. Summative evaluation is a comprehensive assessment conducted at the end of a project or program to determine its overall effectiveness, outcomes, and impact against predetermined criteria or benchmarks.

7.2 Evaluation Criteria for this Project

- Accuracy percentage of detecting the objects.
- Accuracy percentage of tracking the objects.
- Execution speed of the model in real-time.
- Is the user interface and the information displayed eye pleasing and easy to use?
- Is correct information being displayed on the HUD?
- Is IR camera capturing the correct data ?
- Is the system improving the cognitive sense of the driver while driving ?
- Can we add more features or should we reduce the existing features to get optimal results?

7.3 Formative Evaluation

Formative evaluation for our system involves continuous assessment and improvement throughout the development and testing phases. Here is how we plan to conduct formative evaluation for our ADAS:

1. Prototype Testing: We will create early-stage prototypes or simulations of the ADAS. And then conduct controlled experiments and simulations to assess the accuracy and effectiveness of the ADAS algorithms and features.

2. User Testing: We plan to involve end-users in the formative evaluation process. We shall conduct usability tests to assess the user interface, information presentation, and overall user experience and gather feedback from users on the ease of use, effectiveness, and relevance of the ADAS system.

3. Performance Evaluation: We will evaluate the accuracy and performance of object detection, tracking, and other critical ADAS functions. Collect data on false positives, false negatives, and overall accuracy to inform improvements in the system's capabilities.

4. Safety Assessment: We will conduct safety assessments to identify potential safety risks and hazards associated with our ADAS and implement fail-safe mechanisms and safety protocols to mitigate risks.

5. Real-World Testing: We plan to conduct field trials and real-world testing to assess the ADAS system's performance in actual driving conditions. And to collect data and feedback from real-world scenarios to identify and address issues that may not arise in controlled environments.

6. Ethical Considerations: We shall address ethical concerns, such as algorithmic bias and the impact on human behavior, during the formative evaluation. Ensure that the ADAS system operates transparently, respects user privacy, and adheres to ethical guidelines.

9. Validation and Verification: We will validate the ADAS system against predefined requirements and performance criteria and verify that the system meets the intended objectives and delivers the expected outcomes.

Formative evaluation for ADAS is an ongoing process that involves continuous monitoring, testing, and improvement to ensure that the system is accurate, reliable, user-friendly, and aligned with safety and ethical standards. By integrating user feedback and making iterative adjustments, formative evaluation helps optimize the ADAS system's performance and usability for enhanced road safety.

7.4 Summative Evaluation

Summative evaluation for our system involves conducting a comprehensive assessment of the system's overall effectiveness, performance, and impact after its implementation. Here is how we plan to conduct summative evaluation for our ADAS:

1. Define Evaluation Criteria: We will clearly define the evaluation criteria and performance metrics to assess the ADAS system. This may include measures of accuracy, reliability, safety, user satisfaction, and overall impact on road safety.

2. Data Collection: We will gather data from various sources, including real-world usage, field trials, and user feedback.

3. Comparative Analysis: We will compare our ADAS system's performance with other existing ADAS solutions or benchmark it against industry standards to identify areas for improvement.

4. Cost-Benefit Analysis: We will conduct a cost-benefit analysis to assess the overall value and return on investment of the ADAS system in terms of improved safety and efficiency.

5. Report and Recommendations: We will prepare a comprehensive evaluation report summarizing the findings and insights from the evaluation process. Provide actionable recommendations for further improvements and optimizations.

6. Decision-making and Improvements: We shall use the evaluation results and recommendations to inform decision-making processes for system updates, enhancements, and future iterations of the ADAS system.

Summative evaluation for ADAS helps assess the system's overall effectiveness and impact on road safety and driver assistance. It provides valuable insights for optimizing the system's performance, ensuring compliance with safety standards, and identifying potential areas for enhancement and innovation.

Chapter 8

Conclusion and Future Work

8.1 Conclusion

In conclusion, my master's dissertation has presented the development and evaluation of an ADAS aimed at addressing the challenges posed by low visibility conditions on the road. Through a comprehensive analysis of various research paper, we have proposed an ADAS that has the ability to enhance driver safety and situational awareness. The research involved the implementation of sophisticated object detection algorithms and sensor fusion techniques, allowing the ADAS to accurately detect and track obstacles, vehicles, and pedestrians even in adverse weather conditions. The evaluation phase conducted rigorous testing to assess the system's performance and validate its effectiveness.

In summary, this dissertation's contributions underscore the importance of ADAS technology in addressing low visibility issues and its potential to significantly enhance road safety. The findings and insights presented in this study serve as a valuable foundation for further advancements in ADAS development and pave the way for more comprehensive and intelligent driver assistance solutions in the future.

8.2 Future Work

Despite the significant achievements, this study acknowledges the need for continuous research and refinement to optimize the ADAS's performance and adaptability to various real-world conditions. Future work could focus on expanding the system's capabilities to handle extreme weather scenarios and exploring potential collaborations with vehicle manufacturers for practical implementation. We also plan build a, robust version of our older project and additional features such as navigation and Bluetooth connectivity so that it can be used in every vehicle, from low range to high range vehicles.

Appendix A

Important Codes

Given below is the code of the program which calls the pipelined YoloV4 and DeepSort program which was discussed in Chapter X.

```
import dot
import sys
import speech_recognition as sr
import pyttsx3
import dateweather
r = sr.Recognizer()
engine = pyttsx3.init()
voice_id =
"HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Speech\Voices\Tokens\TTS_MS_EN-
US_ZIRA_11.0"
engine.setProperty('voice', voice_id)
rate = engine.getProperty('rate')
engine.setProperty('rate', 190)
dot
while(1):
   try:
       with sr.Microphone(device_index=(1)) as source2:
          r.adjust_for_ambient_noise(source2, duration=1)
          print('Waiting for command')
          audio2 = r.listen(source2)
          command = r.recognize_google(audio2)
          command = command.lower()
          print(command)
          if (command == 'start'):
             detstat = dateweather.detstatus()
             engine.say("welcome user")
             engine.say("Today's weather forecast is" + detstat)
             engine.runAndWait()
             dot.dot(0)
             #if (command == 'exit'):
                #sys.exit("Program has ended")
   except sr.RequestError as e:
      print("Could not request results; {0}".format(e))
   except sr.UnknownValueError:
      print("unknown error occured")
```

Appendix B

Snapshots

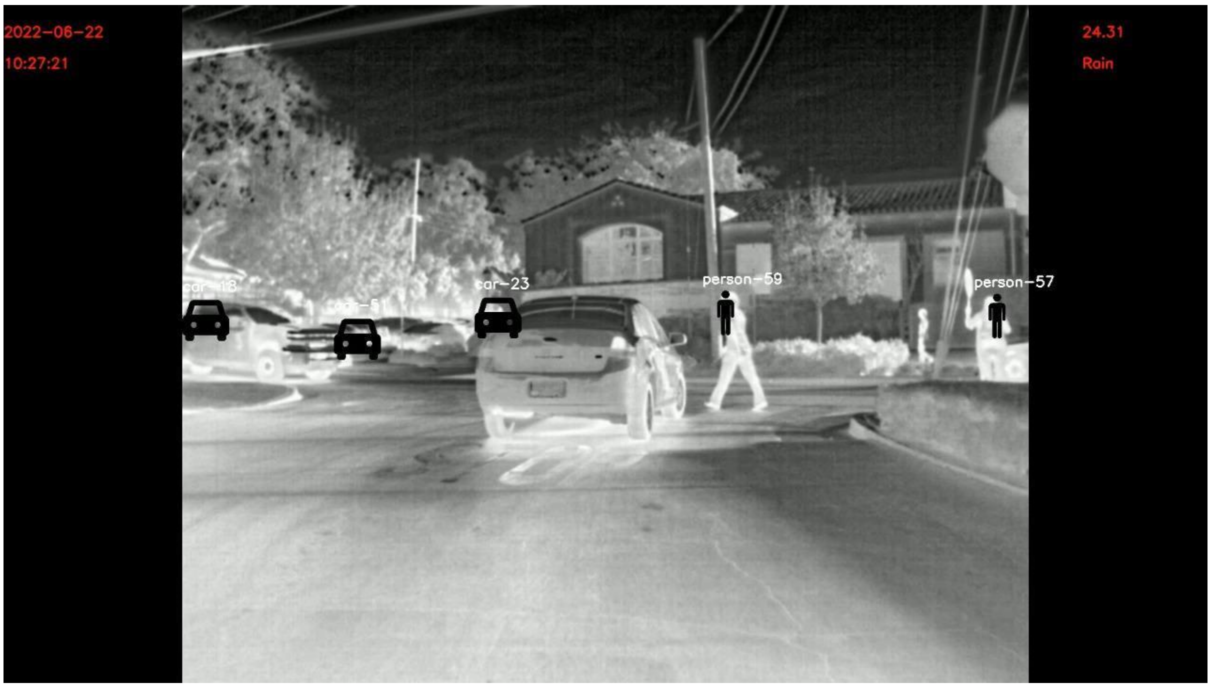

Figure A.1: Detecting Humans in thermal camera

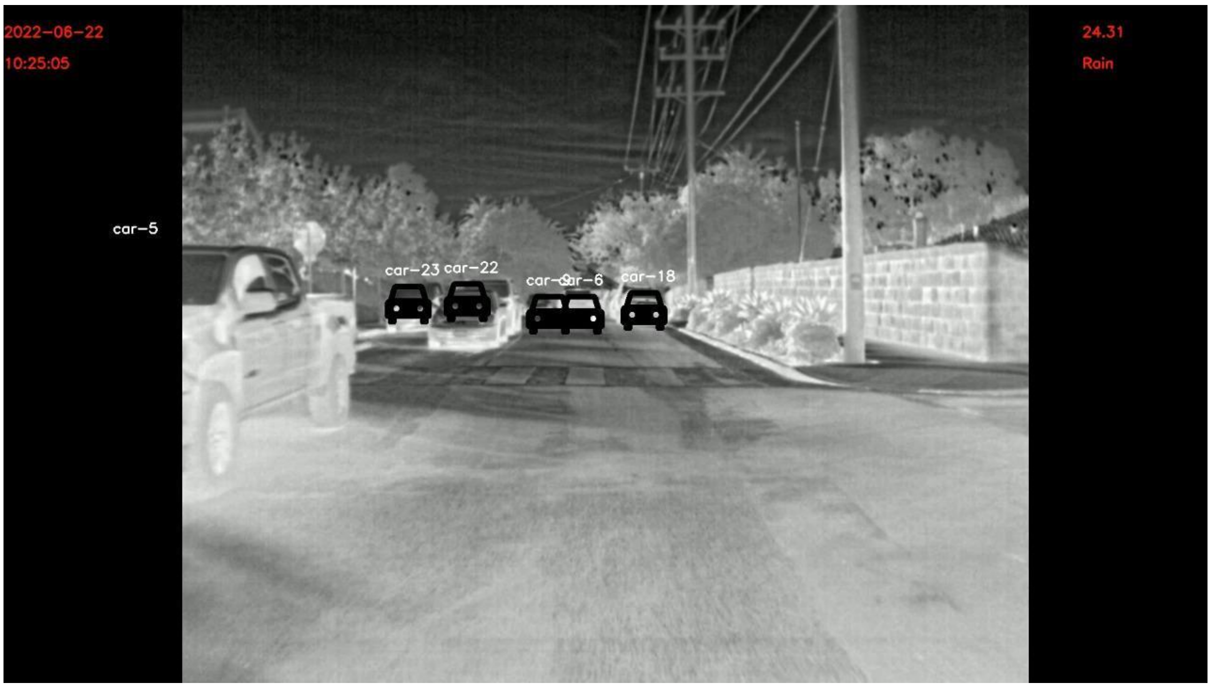

Figure A.2: Detecting Vehicles in thermal camera

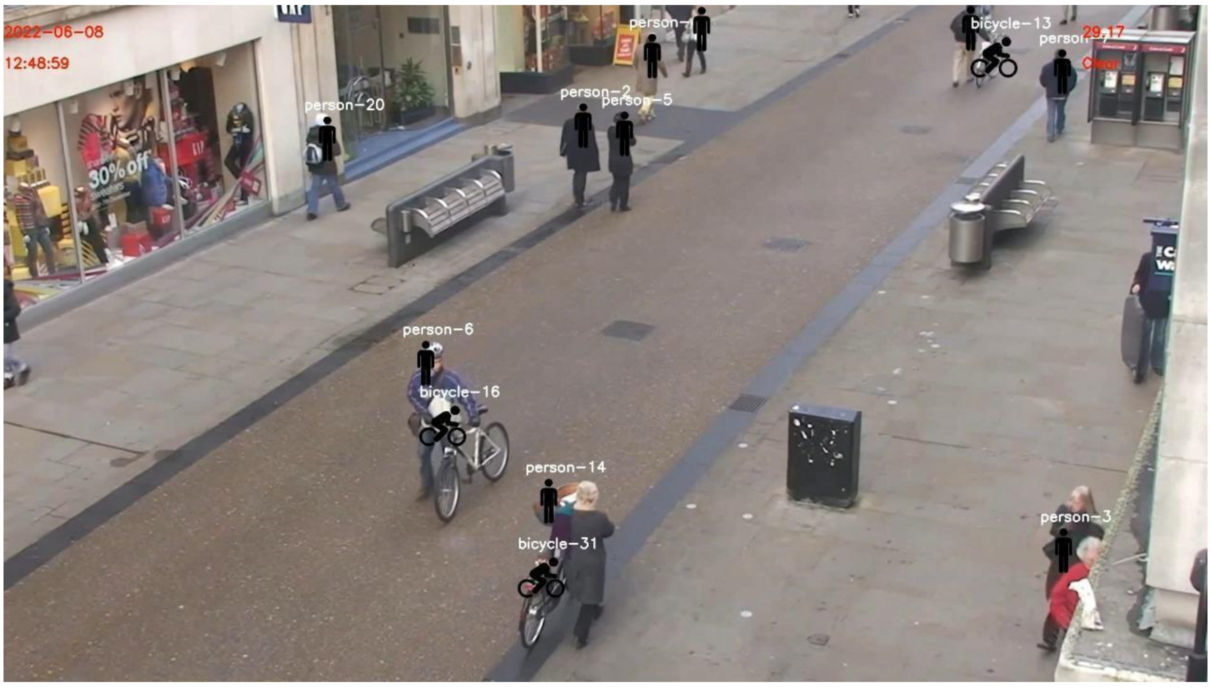

Figure A.3: Detecting Vehicles and Humans in RGB camera

References

[1] WHO. Global Status Report On Road Safety 2018. Available at: https://www.who.int/publications/i/item/9789241565684, pages 1-9, 2018.

[2] Government of India. Road Accidents in India 2021. Available at: https://morth.nic.in/sites/default/files/RA_2021_Compressed.pdf, pages 4,54, 2022.

[3] Prof. Arun Tigadi, Prof. Rudrappa Gujanatti, Anil Gonchi, "ADVANCED DRIVER ASSISTANCE SYSTEMS". International Journal of Engineering Research and General Science Volume 4, Issue 3, May-June, 2016. Available at: http://pnrsolution.org/Datacenter/Vol4/Issue3/22.pdf

[4] A. Shaout, D. Colella and S. Awad, "Advanced Driver Assistance Systems - Past, present and future," *2011 Seventh International Computer Engineering Conference (ICENCO'2011)*, Cairo, Egypt, 2011, pp. 72-82, doi: 10.1109/ICENCO.2011.6153935. Available at: https://ieeexplore.ieee.org/abstract/document/6153935

[5] J. Piao & M. McDonald (2008) Advanced Driver Assistance Systems from Autonomous to Cooperative Approach, Transport Reviews, 28:5, 659-684, DOI: 10.1080/01441640801987825. Available at: https://www.tandfonline.com/doi/abs/10.1080/01441640801987825

[6] V. K. Kukkala, J. Tunnell, S. Pasricha and T. Bradley, "Advanced Driver-Assistance Systems: A Path Toward Autonomous Vehicles," in IEEE Consumer Electronics Magazine, vol. 7, no. 5, pp. 18-25, Sept. 2018, doi: 10.1109/MCE.2018.2828440. Available at: https://ieeexplore.ieee.org/abstract/document/8429957

[7] N. Wojke, A. Bewley and D. Paulus, "Simple online and realtime tracking with a deep association metric," 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, 2017, pp. 3645-3649, doi: 10.1109/ICIP.2017.8296962. Available at: https://ieeexplore.ieee.org/abstract/document/8296962

[8] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, Piotr Dollar, “Microsoft COCO: Common Objects in Context”. Available at: https://arxiv.org/abs/1405.0312

[9] M. Shimoda, Y. Sada, R. Kuramochi and H. Nakahara, "An FPGA Implementation of Real-Time Object Detection with a Thermal Camera," 2019 29th International Conference on Field Programmable Logic and Applications (FPL), Barcelona, Spain, 2019, pp. 413-414, doi: 10.1109/FPL.2019.00072. Available at: https://ieeexplore.ieee.org/abstract/document/8892223

[10] A. Doshi, S. Y. Cheng and M. M. Trivedi, "A Novel Active Heads-Up Display for Driver Assistance," in IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 39, no. 1, pp. 85-93, Feb. 2009, doi: 10.1109/TSMCB.2008.923527. Available at: https://ieeexplore.ieee.org/abstract/document/4581392

[11] Okabayashi, Shigeru, Masao Sakata, Jun’ichi Fukano, Shigetoshi Daidoji, Chikara Hashimoto, and Tomonari Ishikawa. “Development of Practical Heads-Up Display for Production Vehicle Application.” SAE Transactions 98 (1989): 638–47. Available at: http://www.jstor.org/stable/44472311.

[12] Yung-Ching Liu,"Effects of using head-up display in automobile context on attention demand and driving performance",Displays Volume 24, Issues 4–5, 2003, Pages 157-165, ISSN 0141-9382, https://doi.org/10.1016/j.displa.2004.01.001. Available at: https://www.sciencedirect.com/science/article/pii/S0141938204000022

[13] Alexander Mordvintsev & Abid K, "OpenCV-Python Tutorials Documentation Release 1",2017.

[14] A. Sharma, J. Pathak, M. Prakash and J. N. Singh, "Object Detection using OpenCV and Python," 2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N), Greater Noida, India, 2021, pp. 501-505, doi: 10.1109/ICAC3N53548.2021.9725638. Available at: https://ieeexplore.ieee.org/abstract/document/9725638

[15] Abadi, M. (2016). TensorFlow: Learning functions at scale. In *Proceedings of the 21st ACM SIGPLAN international conference on functional programming*. Available at: https://dl.acm.org/doi/abs/10.1145/2951913.2976746

[16] M. Noman, V. Stankovic and A. Tawfik, "Object Detection Techniques: Overview and Performance Comparison," 2019 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), Ajman, United Arab Emirates, 2019, pp. 1-5, doi: 10.1109/ISSPIT47144.2019.9001879. Available at: https://ieeexplore.ieee.org/abstract/document/9001879

[17] Chien-Yao Wang, I-Hau Yeh and Hong-Yuan Mark Liao, "You Only Learn One Representation: Unified Network for Multiple Tasks", 2021. Available at: https://arxiv.org/abs/2105.04206

[18] Alexey Bochkovskiy, Chien-Yao Wang and Hong-Yuan Mark Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection", 2020. Available at: https://arxiv.org/abs/2004.10934

[19] Figure taken from: https://learnopencv.com/understanding-multiple-object-tracking-using-deepsort/

[20] S. M. Mahajan, Sudesh B. Khedkar, Sayali M. Kasav,“Head up Display Techniques in Cars”,International Journal of Engineering Science and Innovative Technology (IJESIT) Volume 4, Issue 2, March 2015. Available at: https://www.researchgate.net/profile/Sudesh-Khedkar/publication/283534782_Head_up_Display_Techniques_in_Cars/links/563d82f308ae34e98c4ae99a/Head-up-Display-Techniques-in-Cars.pdf

[21] https://github.com/theAIGuysCode/yolov4-deepsort/blob/master/LICENSE

[22] Hany M. Hassan, Mohamed A. Abdel-Aty, “Analysis of drivers’ behavior under reduced visibility conditions using a Structural Equation Modeling approach”,2011. Available at: https://doi.org/10.1016/j.trf.2011.07.002

[23] Bassani M. , L. Catani , A. Salussolia , C.Y.D. Yang, “A driving simulation study to examine the impact of available sight distance on driver behavior along rural highways”,2019. Available at: https://doi.org/10.1016/j.aap.2019.07.003

[24] Xin Chang , Haijian Li , Lingqiao Qin , Jian Rong , Yao Lu , Xiaoxuan Chen, “Evaluation of cooperative systems on driver behavior in heavy fog condition based on a driving simulator”,2019. Available at: https://doi.org/10.1016/j.aap.2019.04.019