THESIS

SENSING AND DATA FUSION TO CHARACTERIZE VEHICLE BEHAVIOR SURROUNDING AUTONOMOUS VEHICLES

Submitted by

Chon Chia Ang

Department of Systems Engineering

In partial fulfillment of the requirements

For the Degree of Master of Science

Colorado State University

Fort Collins, Colorado

Summer 2022

Master's Committee:

Advisor:Thomas Bradley

Samuel Bechara
Kamran Eftekhari Shahroudi

ABSTRACT

SENSING AND DATA FUSION TO CHARACTERIZE VEHICLE BEHAVIOR SURROUNDING AUTONOMOUS VEHICLES

Under the request and funding of Colorado Department of Transportation (CDOT), the Colorado State University (CSU) Autonomous Vehicle Research Labs aims to develop a portable and flexible tracking array as a means to collect data pertaining to travel trajectories of vehicles approaching an autonomous vehicle with and without autonomous status indication. This paper will outline the logic, structure and process of designing various features of an autonomous tracking array along with any modifications made as result of feedback from multiple users.

In addition, this paper will cover how the tracking array was used to collect left-lane travel trajectories of Human Driven Vehicle (HDV) in residential lanes with and without the presence of an obstructing vehicle as well as highlighting any significant difference in Fuel Economy (FE) distribution in both scenarios. The purpose of such experiment is to address whether the presence of an obstruction results in a significant decrease in FE of HDV. Findings of the experiment indicates the presence of obstruction in residential lanes results in results in a lower value FE distribution for HDV.

Finally, this paper will cover challenges faced in designing the tracking array, collecting data in residential lanes along with any additional work done by the author during his time at CSU.

ACKNOWLEDGEMENTS

I would like to express my gratitude to Dr.Thomas Bradley for giving me the opportunity to work on these interesting projects. It was thanks to him that I've been given the environment to better develop my skills and experience as a systems engineer along with the chance to undertake a project that I can call my own.

I wish to thank my family for their encouragement and care. I'm also thankful to Aaron Rabinowitz, Brandon Moore, Peter Lobato, David Trinko and Sam White from Dr.Bradley's research lab for providing me support throughout this project whether that be extra hands or additional knowledge. I also appreciate the expertise provided by the members of Western Michigan University's Energy Efficient and Autonomous Vehicles (EEAV) Lab namely, Dr. Zach Asher and Yara Hazem Mahmoud.

Additionally, I express my gratitude towards the Colorado Department of Transportation (CDOT), National Renewable Energy Laboratory (NREL) and United State Department of Energy (DOE) for the funding, equipment and staff necessary to enable the success of this project.

Finally, I would like to acknowledge Dr.Samuel Bechara, Dr.Eftekhari Shahroudi Kamran and Dr. Erika Miller for providing me feedback and suggestion on the writing of this document along with the defense for my thesis.

This material is based upon work supported by the U.S.Department of Energy's Office of Energy Efficiency and Renewable Energy (EERE). The specific organization overseeing this report is the Vehicle Technologies Office under award number DE-EE0008468.

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

In 1939, the idea of self-driving vehicles was introduced by General Motors as part of its Futurama exhibition in New York City [31, 27]. This idea proposed that no driver would need to touch the steering wheel of the vehicle to travel from point A to point B. Thanks to the advancement of autonomous vehicle technology, the idea of self-driving vehicles is no longer a mere concept, but instead a viable next step for the evolution of vehicles[38].

Despite the extraordinary advancement made in automation, little development has been made in terms of creating safety standards for autonomous vehicles operating in mixed traffic. To resolve this issue, Colorado State University (CSU) and Colorado Department of Transportation (CDOT) has proposed research to set up the foundation for autonomous vehicle safety [10]. This research will focus on how different will human driven vehicles react upon noticing an autonomous vehicle on road when compared to no indication of autonomy being provided.

According to the latest autonomous vehicle guidelines set by the Society of Autonomous Engineers (SAE), there are 6 levels of driving automation with level 0 being classified as having no driving automation and level 5 being classified as Full Driving Automation[4]. For this project, a level 3 Autonomous vehicle was provided by CDOT to use throughout this project. A level 3 Autonomous vehicle is classified as an autonomous vehicle with the ability to navigate without driver input, but still requires the driver's input in the event of automation system failure. For the remainder of this section, the Level 3 Autonomous Vehicle (L3AV) shall be referred to with its given acronym.

Figure 1.1: An image of the L3AV in CDOT's possession. This vehicle was integrated with a tracking array designed by CSU.

To capture the behaviour of human-driven vehicles, a tracking system with the ability to detect vehicles surrounding the L3AV (henceforth known as Data Acquisition (DAQ) system) is required. This system will seek to measure the trajectory of vehicles as they approach and pass the L3AV. To accomplish this, a tracking system consisting of radars and camera was proposed. Before integrating said tracking system to the L3AV, the tracking system shall be tested and verified through the use of a human driven vehicle. Once the tracking system was found to provide a sufficient level of performance, CSU coordinated with CDOT to integrate the tracking system with the L3AV and had the DAQ be operated on the L3AV for a period of 2 weeks.

Although the tracking system would be integrated with CDOT's L3AV, access to the autonomous truck was limited by a combination of scheduling, traveling and maintenance constraints. This meant the tracking system cannot be consistently tested and validated with the CDOT autonomous truck. To test and validate the functionality of the tracking system, CSU constructed its own Testing Vehicle Platform (TVP) by modifying a 2007 Chevy Malibu to easily mount the tracking system. Additionally, the sensors of the tracking system were mounted to the TVP in a manner similar to the mounting position and orientation on the CDOT L3AV.

Figure 1.2: An image of the 2007 Chevy Malibu TVP with radars attached.

Once the data was collected, the trajectories of the travelling vehicles was analyzed to provide a clear representation on how much following distance traveling vehicles gave when approaching a L3AV, and how the trajectories of drivers differs when approaching a non-autonomous vehicles. To clarify, the term following distance means the distance between the front of a vehicle and the rear of the vehicle in front of it.

The following sections provide an in depth look at the concepts used to develop the tracking system, verification of the tracking system and interpretation of the data collected.

1.1 Visualization of Questions and Approach

This section provides an in-depth look at the questions posed by this research and the proposed task and activities that were performed to address said questions.

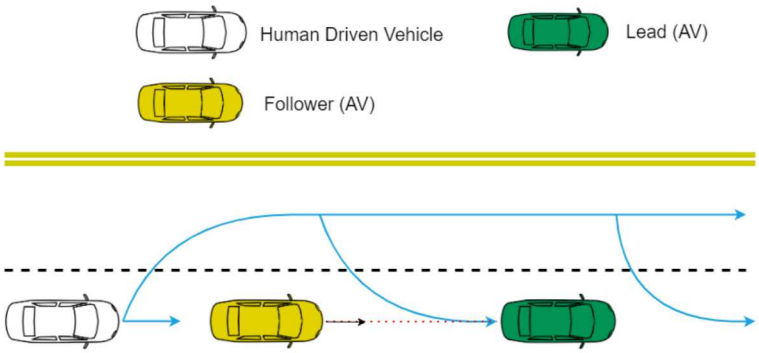

Figure 1.3: How will the human driven vehicle behave if it knows the vehicle in front of it (L3AV) is autonomous?

From the figure above, CDOT wishes to understand how the Human Driven Vehicle (HDV) will perform a left-lane cut depending on 2 different scenarios:

- HDV is notified the L3AV is autonomous.
- HDV is not notified the L3AV is autonomous.

It is CDOT's expectation that by studying how the HDV performs the left lane cut, better safety standards can be developed for autonomous vehicles. To accomplish this, CSU developed an autonomous tracking system that accomplished the following:

- Capture the travel trajectories of approaching HDV gives when performing the left lane cut.
- Capture how different types of HDV perform the left lane cut. (e.g: Do trucks react differently from cars when the driver is notified of the L3AV's autonomous status?)

By explaining the answers to these questions, this research will serve to advance our understanding of the interaction between L3AV and HDV, and will seek to enable improvement of on road safety for a mixed AV/HDV environment.

CHAPTER 2: OVERVIEW OF DATA ACCQUISITION

To determine the trajectory of traveling vehicles when approaching a L3AV, an external object detection and tracking system was developed to measure the distance between the external vehicles from the ego vehicle, which is the primary vehicle of interest controlled by the tester/data collector in an operational/testing scenario. In reviewing research article pertaining to work done on autonomous vehicles [30, 8, 35], it was decided that the best approach to collect the required data is to design a DAQ system consisting of the following components:

- ROS: The primary architecture used to enable communication between various subsystems. (Section 2.1 and 2.2)
- Radar: Detects the number of objects surrounding the ego vehicle in their respective longitudinal (x-axis) and latitudinal (y-axis) position axes through the use of radio waves. (Section 3)
- Camera: Capture images containing external vehicles for analysis/classification. (Section 4)

2.1 Coding Architecture

The Data Acquisition System (DAQ) was designed using a Robot Operating System (ROS)-based architecture written in Python3. This section details what ROS is and why such architecture was chosen for this project.

ROS is a flexible open-sourced framework for writing robot software with proven real-life application in autonomous technology industry[1]. Equipped with a massive repository, and maintained by numerous developers, ROS was chosen a starting point and backbone to develop the array. ROS's numerous packages helps reduces the number of lines of new code that needs to be written. Although not reflective of ROS' full capabilities, the best way to understand the framework is to visualize it as the following items:

- Master Node: A master 'terminal' that oversees the operation of all other nodes.
- Nodes: A 'terminal' where messages are processed, transmitted or received.
- Topics: Certain items that a node either publishes or is subscribed to.
- Messages: The contents of a topic (variable values or strings).

The primary reason for using ROS in designing robotic systems is the framework's modularity[45]. Since ROS-based architecture consists of a series of nodes and messages with different functionality, one can choose which function to access for each specific node and message. For instance, for a specific node one can choose a combination of the following functions to utilize:

- Publishing and subscribing: The ability to transmit and receive messages topic from other nodes.
- Recording and playback: Writing and replaying messages in ROS.
- Respond/request remote procedure calls: Processing a message when it is on its way to another node, potentially resulting in a different output. This is also called a service.
- Distributed parameter system: The ability to configure a global variable for all nodes.

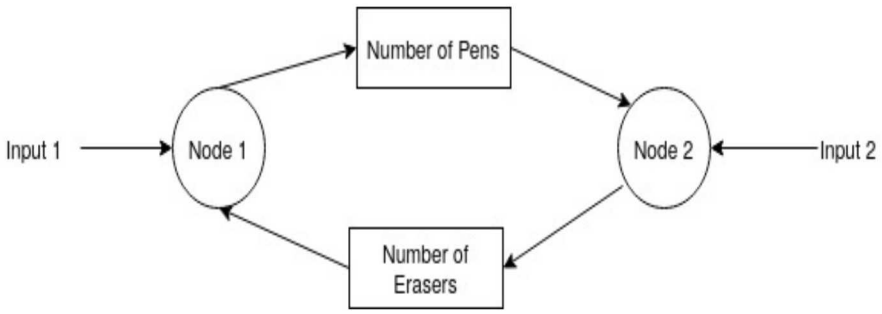

Figure 2.1: A Simple ROS communication

ROS' ability to publish and subscribe messages is a vital tool for this project. Figure 2.1 provides a simple representation of the ROS tool. In Figure 2.1, 2 nodes (Node 1 and Node 2) are communicating with each other, both sending a different topic to other. Node 1 sends messages relating to the topic of 'Number of Pens' to Node 2. In ROS terms, Node 1 is publishing topics of 'Number of Pens' while Node 2

is subscribed to said topic. Conversely, Node 2 is publishing topics of 'Number of Erasers' while Node 1 is subscribed to said topic.

Even though ROS does provide an advantageous foundation for designing the DAQ system, it is not without its weaknesses. ROS has a Master node that oversees the operation of all other nodes in its architecture. Should the Master node crash, the ROS architecture will fail and shut down. Though ROS 2 promises to address such vulnerability, a stable update was not available at the time of this research.

2.1.1 Zookeeper Node

A recommended way to mitigate the master node crashing problem is to implement a Zookeeper-like mechanism in the architecture. The purpose of the Zookeeper-like mechanism is to accomplish two purposes: to ensure there is a master node active at all time and to monitor/reactivate any 'dead' nodes.

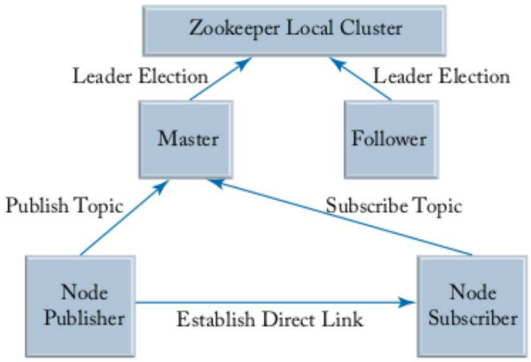

Figure 2.2: Zookeeper Master node supplement mechanism (Taken from pg 171 of 'Creating Autonomous Vehicle System')[48]

Figure 2.3 outlines how one can ensure there is always a master node active in ROS. Should the master node crashes, a follower node will be elected as a new master node, preventing ROS from crashing.

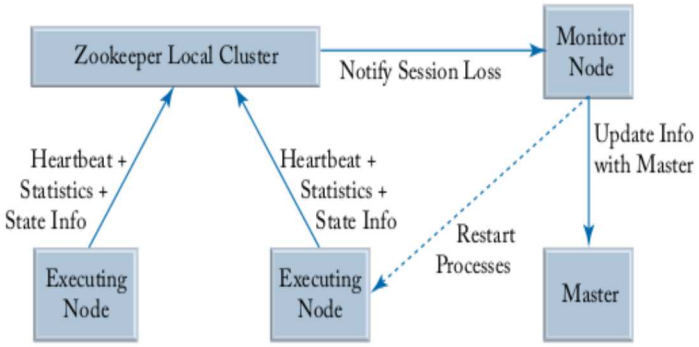

Figure 2.3: Zookeeper lesser node supplement mechanism (Taken from pg 172 of 'Creating Autonomous Vehicle System')[48]

Using a signal sent at fixed interval (once every 1 second), each node can periodically send updates on its status to the monitor node. Should a node be found unresponsive for certain period of time, the monitor will initiate a restart process to revive the deactivated node.

2.2 ROS Implementation of the DAQ System

The usage of ROS in the DAQ system enables a series of node to node communications. This type of communication not only modulates the codes that make up the array, but also helps make troubleshooting the code easier.

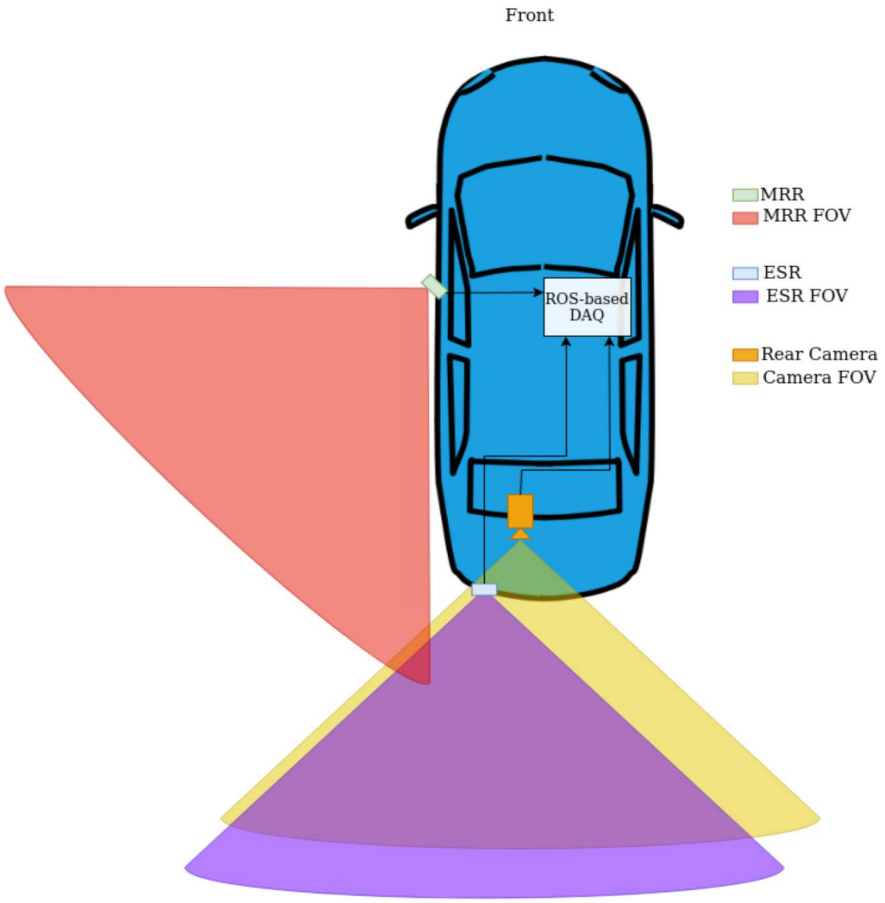

Figure 2.4: Overview of DAQ System in this project.

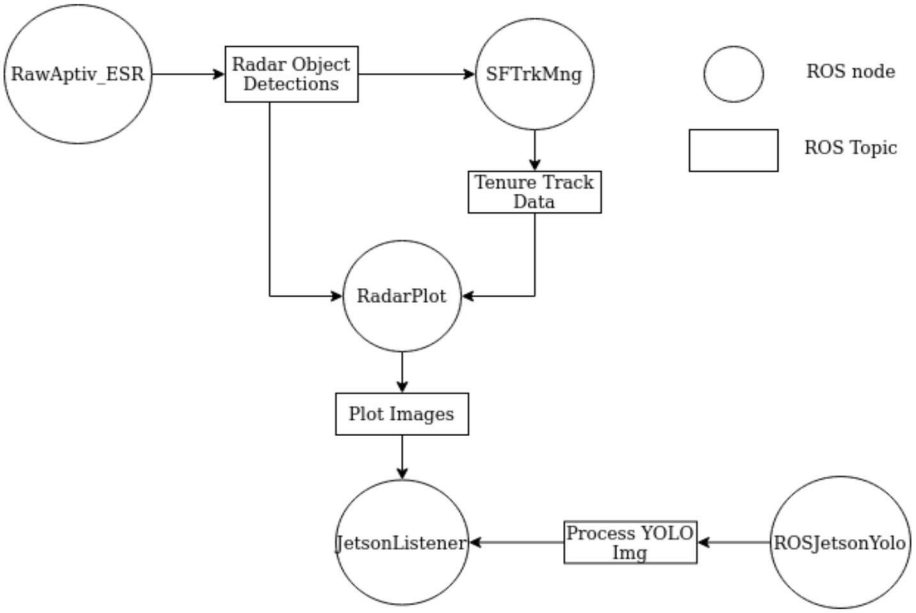

Figure 2.5: How ROS is implemented in the DAQ.

Figure 2.5 provides an overview of how ROS is implemented in the DAQ as a series of nodes communicating with each other on specific topics. The function of each node is listed below:

- RawAptiv_ESR: Grabs, rotates and translates detection inputs from radars that make up the DAQ to match the master axis of the ego vehicle before transmitting them to other nodes.
- SFTrkMng: Assembles detection measurements (position and velocity) into tracks and performs detection to track match.
- RadarPlot: Generates real time plot for raw detections from RawAptiv_ESR and tenured tracks from SFTrkMng.
- ROSJetsonYolo: Grabs raw camera images and process them through an object classification algorithm. Upon classification, processed images are sent to JetsonListener.
- JetsonListener: Grabs both real time plot and process images from RadarPlot and ROSJetsonYolo respectively. Both image and plot are then combined to provide a side-by-side view of real-time tracking.

Additionally, the topic of communication between each are described as:

- Radar Object Detections: Contains position and velocity measurements of objects detected by radar.
- Tenure Track Data: Contains history of latitude and longitudinal positions of tenure tracks.
- Plot Images: Contains plot outlining tenure tracks and raw detections from radars.
- Process YOLO Img: Contains captured camera images with vehicle identification.

CHAPTER 3: OBJECT DETECTION ARRAY-RADAR

In order to detect the presence of external vehicles approaching the ego, a detection array consisting of two radars was mounted onto the Autonomous Vehicle (AV). There are two types of radar making up the array, an Electronically Scanned Radar (ESR) and a Mid Range Radar (MRR) (See Figure 3.1 and Figure 3.2 respectively).

3.1 Electronic Scan Radar (ESR)

Figure 3.1: An image of the ESR used in the project

Table 3.1: ESR Specifications

Specs	Parameters
Model Number	Aptiv ESR 2.5V
Horizontal Field of View (Degrees)	-45 to 45
Mid Detection Range (m)	1 to 60
Mid Range Distance Accuracy (m)	0.25m
Long Horizontal Field of View (Degrees)	-10 to 10
Long Detection Range (m)	1 to 170
Long Range Distance Accuracy (m)	0.5m
Velocity Range (m/s)	-100 to 25
Velocity Accuracy (m/s)	0.12
Maximum Number of Object Detected	64
Update Rate (ms)	50

Contrary to conventional radars, the ESR performs a detection scan through the use of only electrical components. The lack of mechanical components in the device means the ESR is not susceptible to mechanical interference such as vibrations resulting from road quality variations and radar beam inflexibility, hence removing a potential source of failure.

The detection view of the ESR used in this projection consists of Long Range and Mid Range antennas. The Long Range antenna has a detection range of 170 m with a Field of View (FOV) of ± 10 deg while the Mid Range antenna has a detection range of 60 m with a FOV of ± 45 deg. These detection antennae make the ESR an ideal sensor for detecting external vehicles approaching from the rear as the Long Range antenna enables the capture of incoming vehicle while the Mid Range antenna enables the capture of travel trajectories (See Table 3.1).

Although not used in this project, the ESR possesses the capability to capture detected objects in 3D along with measuring the acceleration of the ego vehicle. Should later iterations of the DAQ require such

data, the ESR can be configured to provide such measurements.

3.2 Mid Range Radar (MRR)

Figure 3.2: An image of the MRR used in the project

Table 3.2: MRR Specifications

Specs	Parameters
Model Number	Aptiv MRR
Horizontal Field of View (Degrees)	-45 to 45
Mid Detection Range (m)	1 to 40
Mid Range Distance Accuracy (m)	0.5m
Long Horizontal Field of View (Degrees)	-45 to 45
Long Detection Range (m)	3 to 160
Long Range Distance Accuracy (m)	0.5m
Velocity Range (m/s)	-100 to 20
Velocity Accuracy (m/s)	0.3
Maximum Number of Object Detected	64
Update Rate (ms)	30

The MRR is used to detect vehicles cutting through the left side of the ego vehicle. Although another ESR could have been used to perform such a task, the price point of the ESR meant only one such radar could

be bought. Given the horizontal distance between two vehicles cutting each other is about 2 to 5 meters, a MRR's detection range of 60m and ±45 degrees FOV should be sufficient to detect vehicles surrounding the sides of the ego (See Table 3.2).

3.3 Radar Mounting

Because of limited access to CDOT's Autonomous Truck, this project will use a 2007 Chevy Malibu as the primary Testing Vehicle Platform (TVP) for operating and testing the tracking array. Due to the difference in mounting methods between the Testing Vehicle Platform (TVP) and the CDOT Autonomous Truck, this section shall be divided into two to provide a dedicated explanation for the mounting of array to each vehicle.

3.3.1 TVP Mounting

In order to properly mount the radars, two mounting brackets designed for the ESR and MRR were purchased and modified with magnets to enable the attachment to the TVP (See Figure 3.3). This design ensures the radar can be securely mounted and removed during every data collection session.

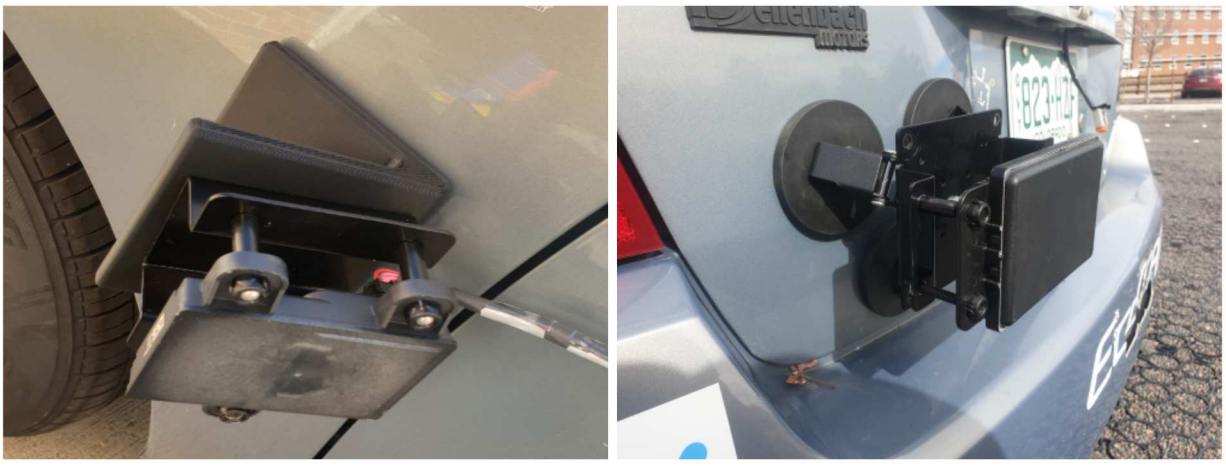

Figure 3.3: MRR radar mounted at 45 degree angle (left) and ESR mounted at the rear (right) of the 2007 Chevy Malibu.

Due to the need to perform side detection with the MRR, the MRR is mounted at a 45 degree angle. The reason for this is explained in the 'Doppler Effect' section.

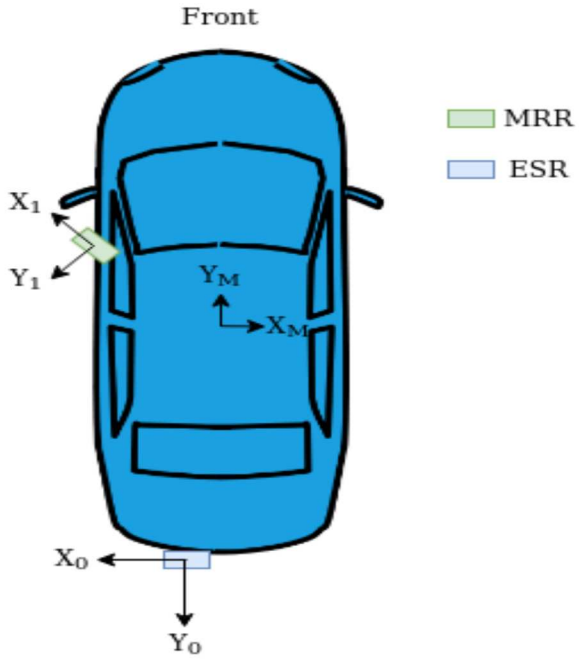

Figure 3.4: Layout of Radar Mounted on TVP (Top View)

Figure 3.4 shows how the ESR and MRR were attached. Due to the radars' positioning and orientation, the measurements captured by the radars will be rotated and translated to match the master axis $Y_M X_M$ of the TVP[18].

Table 3.3: Rotation and Translation Matrix of individual radars.

Radar No	Rotation Matrix, R_i	Translation Matrix, T_i
Radar0: ESR	$\begin{pmatrix} cos(180) & -sin(180) \\ sin(180) & cos(180) \end{pmatrix}$	$\begin{pmatrix} dx_{0M} \\ dy_{0M} \end{pmatrix}$
Radar1: MRR	$\begin{pmatrix} cos(225) & -sin(225) \\ sin(225) & cos(225) \end{pmatrix}$	$\begin{pmatrix} dx_{1M} \\ dy_{1M} \end{pmatrix}$

Table 3.3 contains the rotation and translation matrices required to convert the data captured by the

radars to match the master axis $Y_M X_M$. The conversion is carried out using Equation 3.1.

$$D = R_i * z_i + T_i \tag{3.1}$$

where D=radar distance measurements in terms of $Y_M X_M$ axis, R_i=rotation matrix for individual radars from Table 1, z_i=measurements captured by individual radars in terms of individual radar axis $Y_i X_i$, T_i=translation matrix for individual radars from Table 3.3 in terms of $Y_M X_M$ axis. i= Radar No registered in Table 3.3.

3.3.2 Doppler Effect

The Doppler Effect is defined as the change in frequency of wave emitted by a moving object in relation to the observer[46][39]. Given that the project involves tracking the movement of moving vehicles, this effect must be taken into account when mounting the radar on the TVP.

$$f_o = f_s * (v - v_o)/(v - v_s) \tag{3.2}$$

Equation 3.2 outlines the principles of the Doppler Effect in relation to a source moving towards and observer and an observer moving away from the source. f_o represents the frequency observed by the the observer, f_s is the frequency emitted by the source, v is the speed of sound, v_o is the velocity of the observer and v_s is the velocity of the source[39].

A radar determines the distance between itself and an object through the emission and rebound of microwaves. Depending on the change in wavelength between the emitted microwave and rebound microwave, a radar can determine if the object is approaching or moving away from it. For instance, if a source is moving towards a moving observer at a higher speed, it is means $v_s > v_o$. This causes the frequency observed, f_o to be much larger compared to the frequency of the wave emitted by the source, f_s. Inversely, this means the

wavelength captured by the observer is much shorter than the wavelength emitted by the source.

For the ESR radar mounted at the rear of the ego vehicle/TVP, the Doppler Effect of the approaching vehicle is negligible as the compression of the rebounded radio wave matches up with the direction movement of the approaching vehicle. (See Figure 3.5)

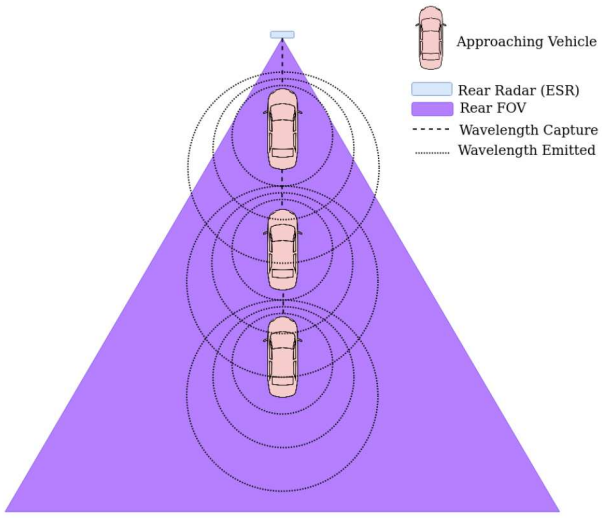

Figure 3.5: Wavelength change of the Approaching Vehicle is consistent in the direction of the rear radar, enabling accurate distance readings to be registered. (Top View)

The same cannot be said for the radar mounted on the side of ego vehicle/TVP. Given that the side radar is used to capture vehicles cutting laterally across it, the Doppler Effect alters the wavelength in a manner that causes the side radar to register inaccurate reading should it be mounted flat on the side of the TVP. (See Figure 3.6)

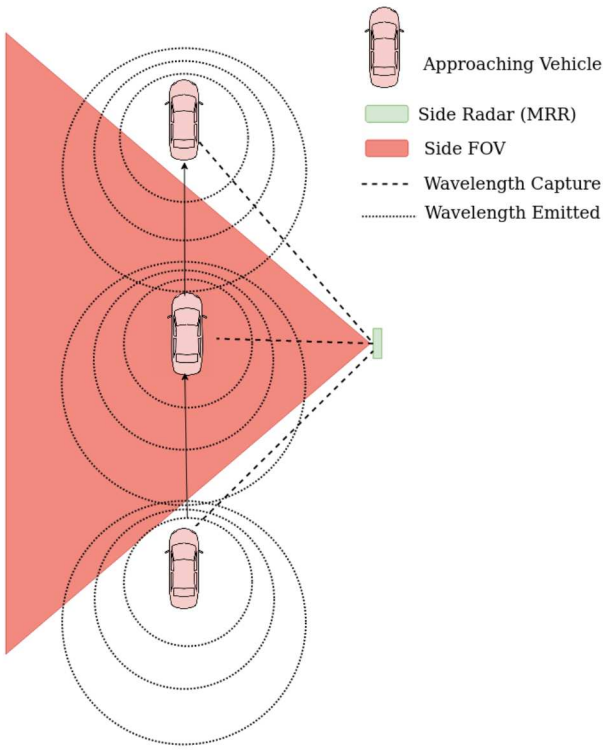

Figure 3.6: With side mounting, captured wavelength change of the approaching vehicle is inconsistent with approaching vehicle travel direction, causing the radar to register inaccurate readings. (Top View)

To mitigate this effect, the side radar was mounted at a 45 degree angle using a 3D printed mounting plate.(See Figure 3.7 and Figure 3.8) The are two benefits of mounting the side radar in such a manner:

- The Doppler Effect of moving vehicle will be minimized, allowing the MRR to capture a more accurate reading of the moving vehicle's distance.
- The MRR's ±45 degree FOV will enable coverage of the entire left side of the ego vehicle.

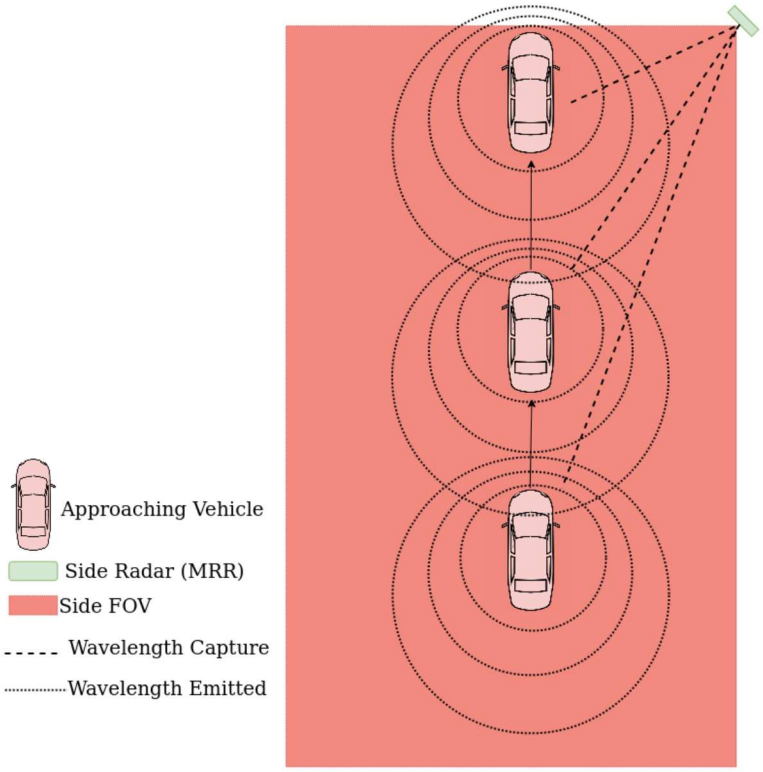

Figure 3.7: With 45 degree mounting, the wavelength change capture is more consistent and FOV covers the entire left side of the ego vehicle. (Top View)

Figure 3.8: The 3D printed plate used to mount the MRR at a 45 degree angle.

3.3.3 *CDOT Autonomous Truck*

The mounting of the radars on the CDOT Autonomous Truck will follow the same orientation as the TVP. However, with the structure of the Autonomous Truck differing from the TVP, modifications needed to be made to enable the attachment of the mounts.

Figure 3.9: MRR mount with buffer

As illustrated in Figure 3.9, the mount for the MRR radar is clamped to a side bar of the truck using another 3D printed plate. To prevent vibrations, a 0.25 inch aluminum buffer is used to separate the plates and lock them in place.

Figure 3.10: ESR mount at rear of truck

Similar to the MRR mount, the ESR is clamped to the back cage of the truck using another 3D printed plate. This set up securely holds the ESR in place, as illustrated in Figure 3.10.

CHAPTER 4: CAMERA SYSTEM SETUP

4.1 Object Detection Algorithm

One of the features the array must possess is the ability to identify the types of vehicles performing a left lane cut. Although the radars used comes with the ability to identify what kind of object it is detecting, this feature is not fully reliable as the radar determines what kind of object its is detecting through the use of an object's Radar Cross Section (RCS), which is a measurement of how detectable an object is to a radar, and testing of this feature has found that the radars mis-register a truck as a car and vice versa. Hence, a system where object detection via real time image capture was proposed.

There are two types of object detection algorithms for machine vision: multi-stage detection and single stage detection[20][34] algorithms. Multi stage detection can be defined as a method where a detection algorithm analyses an image and isolate potential objects into proposal regions. These regions are then classified into objects using a set of predefined weights [41][5][56], as illustrated in Figure 4.1.

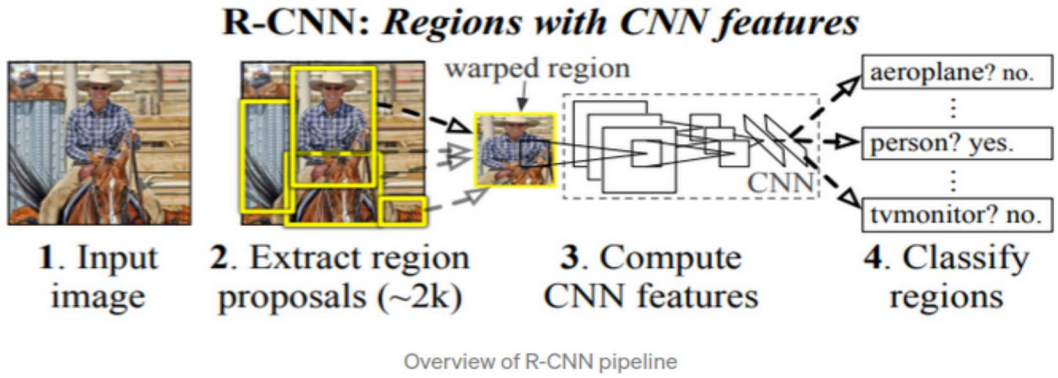

Figure 4.1: An example of a 3-stage detection algorithm. Reinforced Convoluted Neural Network (R-CNN) performs object detection by generating proposed regions, compute Convoluted Neural Network (CNN) features and classifying regions.[41]

Although multi-stage detectors provide detections with higher accuracy, the computational cost per image is expensive due to the need to generate proposal regions, resulting in a longer inference time per image.

Since this project requires real-time object classification, multi-stage detection algorithms were considered too computationally costly, and therefore infeasible[52][51].

In contrast, single stage detectors have less computational cost as images don't require proposal region generation, but instead performs object localization and classification in a single step. At the cost of a lower detection accuracy, single stage detectors can process images fast enough for real time applications[7][56]. Out of all single stage detectors, the one this project will be using is You Only Look Once (YOLO) due to the extensive amount of research has been done using the algorithm along with its large number of available libraries.

YOLO algorithm is an open sourced object detection algorithm that uses a predefined neural network to perform object classification with proven application in unmanned vehicles[11]. Developed by Joseph Redmon, the premise of the algorithm is to train a neural network (weights) using a collection of images[43][44]. Once the images are used to train a neural network, the neural network is then validated to determine the accuracy of the the weight's object classification. Because the localization and classification of objects are done in the same step, the computation time for YOLO is shorter with the accuracy being dependent on the quality of the training images/pre-trained weights. (See Figure 4.2)

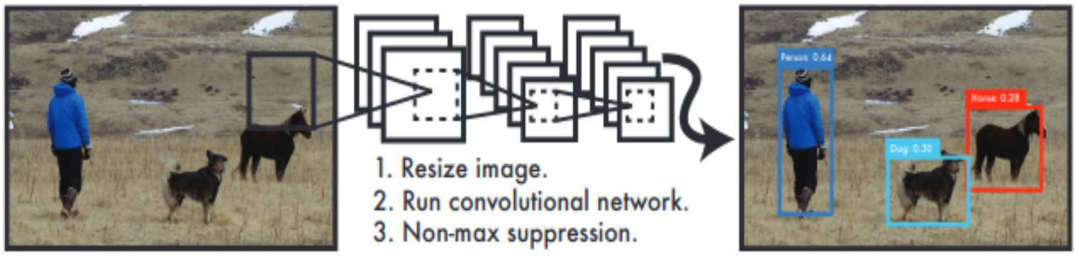

Figure 4.2: How YOLO works.[43].

Given this project's need for detecting only vehicles approaching the rear of the L3AV in real time, a YOLO-based object detection algorithm was found to provide a sufficient level of performance as there is only a small set of objects of interest[52, 51, 14].

For YOLO to be effective, a weight/network would need to be created with a sufficiently large number of sample image[50]. Normally, a large amount of time would be spent collecting and classifying images, but YOLO's open-sourced architecture meant such weights were already created by other developers. As of this paper's writing, the latest version of YOLO is YOLOv5 by Ultralytics. The weights along with the YOLO algorithms were implemented from Ultralytics' github page with citations to its author[25].

4.1.1 What is YOLO ?

This section of the paper provides an in depth look to how YOLO classifies object through the use of its neural network and highlight the working principles behind YOLO [43]. There 3 things that YOLO requires to see where and what objects are located within an image[28]:

- Residual Blocks: How an image is 'chopped up' into smaller square grids. The finer the grids, the higher the accuracy of the detection.
- Bounding Over Regression: How a box is drawn on an object in an image. One object may have more than one bounding box during the detection process. The generation of these bounding boxes is dependent on the neural network trained. (ie: If the predefined weight was not trained to detect an object like a computer, computers in an image will be ignored.)
- Intersection Over Union (IOU): How much confidence an object classification is. The higher the IOU, the higher the level of confidence.

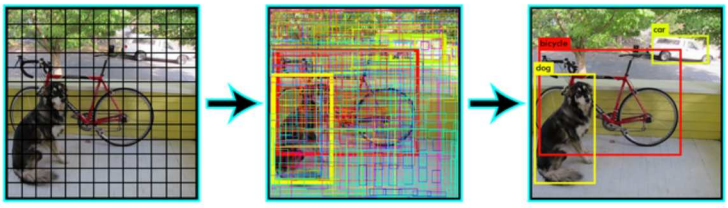

Figure 4.3: A breakdown of the three things YOLO needs to work. Image from [2]

Figure 4.3 outlines step-by-step how YOLO analyzes an image. Starting from the image on the left,

the image is sliced up into SxS grids called the residual blocks. The number of residual blocks generated is dependent on the level of resolution configured in YOLO with larger number of residual blocks equating to higher resolution for image analysis. Once the images have been sliced into SxS grids, bounding boxes are drawn using the residual blocks with a class object assigned to each box using the predefined weights. Once the class of each bounding box have been assigned, the Intersection Over Union of each bounding boxes are calculated and a detection class confidence is assigned to the class of each box.

4.2 Physical Camera Set Up

In order to perform object classification and detection, images of sufficient quality and resolution are required. After several product reviews, a Logitech BRIO 4K Webcam was found to provide images of sufficiently high resolution. Due to the camera being connected via USB, the camera can be interfaced through the use of Open CV2 via Python3 allowing ease of access and code modification, as illustrated in Figure 4.4.

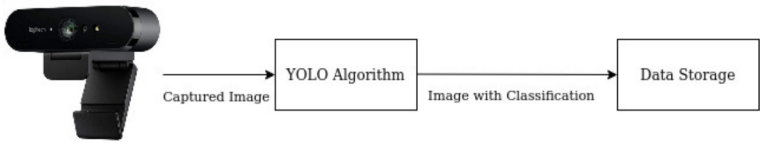

Figure 4.4: Image capture and classification set up.

Because the tracking array is expected to operate in an environment with vehicles travelling at high speed, small debris such as rocks or externalities could damage the camera's lens. Should such events occur, the camera's ability to perform image capture will be worsened. To prevent such events, a camera box with a front plastic layer was designed to serve as a housing unit for the camera. The camera box was designed to be 3D printable so more copies could be made, if required.(See Figure 4.5)

Figure 4.5: Housing Unit for Camera

4.2.1 Additional Modification-Polarized Lens

A notable issue that arise during testing was the capture of 'blank' images under sunlit environments. Upon investigation, the cause for the 'blank' images was found to be sunlight passing through the lens without being polarized, hence 'blinding' the camera. To resolve this issue, the camera box was modified to allow the attachment of polarized sun glass lenses. (See Figure 4.6)

Figure 4.6: Version 2 of Housing Unit for Camera with an internal sun-glass lens and plastic cover.

4.2.2 Jetson Xavier NX

In order to create a visual detection system with plug and play capabilities, along with relieving the primary control device of computation power for YOLO, the camera system was deployed using a Jetson Xavier NX Graphical Processing Unit (GPU) and interfaced through ROS. The Jetson Xavier NX was developed by Nvidia for the purposes of Artificial Intelligence development and has found success as a tool to accelerate image processing algorithm (See Figure 4.7). For this project, the Jetson Xavier NX was chosen for its ability to process images with a performance of 607 images per seconds using Tiny-YoloV3 at 416x416 image resolution[40]. This performance is expected to drop with the object detection system implemented as the images collected are of higher resolution (1280x960) and Yolov5 using more neural network layers compared to Tiny-YoloV3.

Figure 4.7: Jetson Xavier NX

To activate the YOLOv5 detection algorithm stored in the Jetson, the following requirements must be met:

- The camera is attached to one of the Jetson's USB ports.
- The Jetson is connected to another ROS capable device through USB Ethernet.
- The Jetson is powered by a 19V power source.

By modifying '.bashrc' script in the Jetson, the Jetson is configured to call the YOLOv5 object detection algorithm whenever a terminal is opened. This script will process images captured by the Logitech Camera via YOLOv5 and send out the bounding boxes and captured images as outputs via ROS.

In order to send ROS information to another device, the environment variable 'ROS MASTER' in the Jetson is to set to have the same 'ROS MASTER URI' variable in another device (ie: If a Laptop's desktop name is 'ETS00396', then set 'export ROS MASTER URI=http://ETS00396:11311' in the Jetson's working environment via the '.bashrc' script). By changing the 'ROS MASTER URI' environment variable in the Jetson, the ROS system in the Jetson will search for the ROS master node in the other device and receive/send messages to it.

CHAPTER 5: Casing and Protection of Array

To ensure the array is protected from external factors such as rain, snow or high-speed rocks, components considered 'delicate' were housed inside a protective case. This section will outline modifications made to provide additional layers of protection for the tracking array.

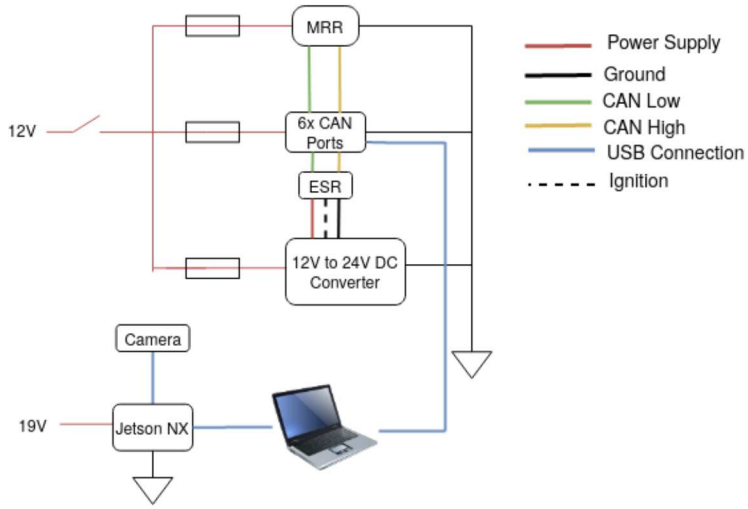

Figure 5.1: Radar Array Wiring Diagram

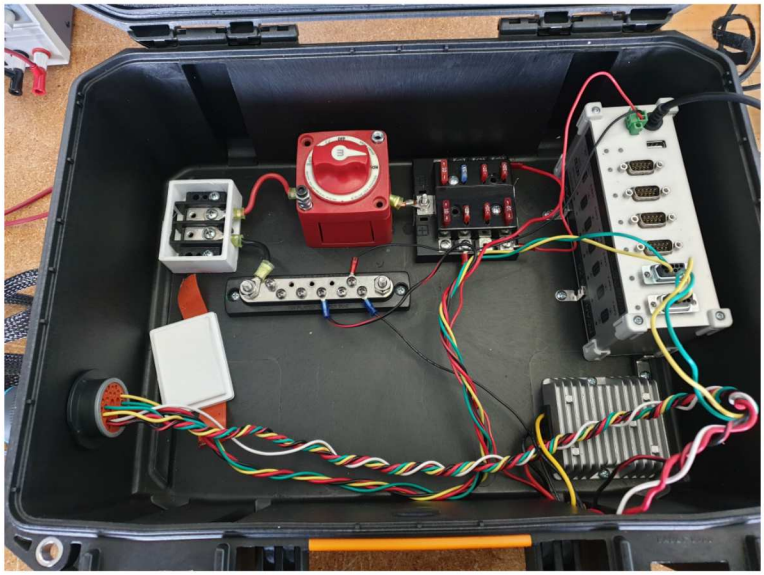

Figure 5.2: Wiring Inside Protection Case

All components susceptible to water and cutting damage were stored and fixed inside a protective case

as shown in Figure 5.2. This not only allows the components to remain operational in harsh environments, but also makes the array portable as all the necessary components can now be carried as a single unit.

Using a 29-pin Bulkhead Connector, all wires pertaining to powering and data transfer from the radars were consolidated into a single connection, eliminating the need to disconnect the radars' wires individually after every use (See Figure 5.3). Table 5.1 outlines what each pin in the 29-pin bulkhead connector were allocated for. Pins without purpose were left as reserves for future use.

Figure 5.3: 29-pin Bulkhead used to establish single connection between radar and array box.

Table 5.1: Wiring Number for 29 Pin Mass Wire Connector

Pin No.	Purpose
1	Rail Power 12V
2	Rail Ground
3	-
4	-
5	MRR CAN High (Yellow)
6	MRR CAN Low (Green)
7	MRR Power 12V
8	MRR Ground
9	ESR Power 12 V (Red)
10	ESR Power 12 V (White)
11	ESR Ground
12	ESR CAN High (Yellow)
13	ESR CAN Low (Green)
14-29	-

CHAPTER 6: EXTERNAL OBJECT TRACK MANAGEMENT

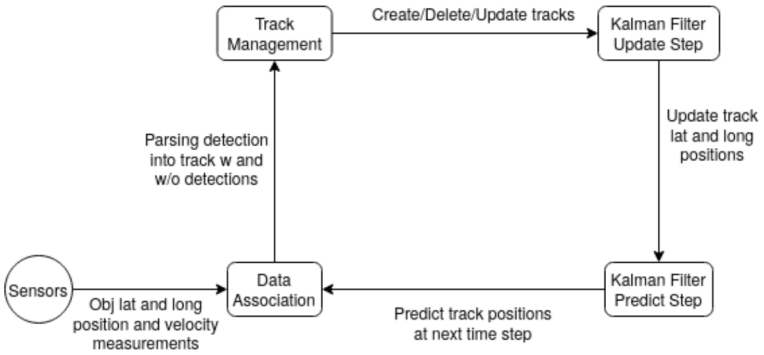

Figure 6.1: Detection Logic of the DAQ System

Figure 6.1 outlines the DAQ's processes to manage data collected by the radars. Starting with the position and velocity measurements, the data is fed into a Data Association algorithm to determine whether or not a detection has a match with an existing track. If a detection is found to have a match with an existing track, the detection is assigned to said track. If not, instructions are sent to the track management portion of the algorithm to create a new track using said detection.

The purpose of the Kalman Filter Update Step is to update the existing tracks by weighting the uncertainty between a track's predicted position and matched detection position. Once the tracks have been updated, the Kalman Filter Prediction Step will predict what position the tracks will be at the next time step using the current velocity of individual tracks. This prediction will then be used for data association with the next set of object position measurement.

6.1 Kalman Filter

Although the radars are capable of collecting position measurements, these measurements are susceptible to interference in the form of noise. To reduce the effect such interference, a Kalman Filter is used to remove noises in measurements registered by the radars [58][59][37][12].

In general, a Kalman Filter consists of two steps, a Prediction step and an Update step. In the Prediction step, the Kalman Filter predicts what the next state of a measurement will be using the its current state. Once a prediction has been made and compared to the next measurement received by a sensor, a new state measurement is obtained and the state of the Kalman Filter is updated in the Update step. The update of the Kalman Filter's state depends on the level of trust placed on the either the measured state or the predicted state[3][37][60].

The drawback of the Kalman Filter lies in its inability to provide accurate measurements in non-linear systems [59]. While such problems can be solved using an Extended Kalman Filter, a traveling vehicle is a relatively linear system, meaning non-linearity is not a concern for this project[49]. Hence, a normal 2D Kalman Filter is the algorithm chosen here for filtering position and velocity measurements from the radars.

Assuming external vehicles are approaching the ego vehicle with a constant velocity, the following equations are used to filter the distance and velocity measurements registered in the radar's latitudinal (y-axis) and longitudinal (x-axis) axis.

Predict Step

$$\hat{X} = X + FX \tag{6.1}$$

$$\hat{P} = FPF^T + Q \tag{6.2}$$

In the prediction step, the estimated state $\hat{X}$ at t+1 is calculated using the current state X and a state transition model F. In the instance of the DAQ, the current state, X is the filtered the position and velocity measurement collected by the radars at the time step t.

Similarly, the estimated posterior $\hat{P}$ is calculated using the state transition model, F, co-variance matrix, P and the process noise model, Q. The co-variance matrix, P represents the level of uncertainty for the measurement collected by the radar (This is equivalent to the accuracy of the radars). The process noise model, Q is an estimate of the noise that interferes with the measurement of the radars.

To initialize the Kalman Filter, the estimate for state, $\hat{X}$ and posterior $\hat{P}$ at t=0 are set to a 2x2 zero

matrix and a 2x2 identity matrix multiplied by the accuracy of the individual radars.

$$\hat{X}_{t=0} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \hat{P}_{t=0} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} * \alpha \tag{6.3}$$

where α represents the accuracy of the individual radars, which is 0.25m (Mid Range Accuracy) and 0.5m (Long Range Accuracy) for ESR and 0.5m for the MRR (Mid Range Accuracy).

Update Step

$$X = \hat{X} + KY \tag{6.4}$$

$$P = \hat{P} - KSK^T \tag{6.5}$$

where $\hat{X}$ and $\hat{P}$ represent the estimated state and posterior from the Predict Step of the Kalman Filter while the remaining variables are listed below:

$$Y = H * (Z - \hat{X}) \tag{6.6}$$

$$S = H\hat{P}H^T + R \tag{6.7}$$

$$K = \hat{P}H^T S^{-}1 \tag{6.8}$$

The variable Y, represents the residual/error between the measurement collected by the radar, Z and the estimated state, $\hat{X}$. The measurement model H is a matrix used to isolate the difference in position x and y from $Z - \hat{X}$. Note: This is slightly different from the normal $Y = Z - H * \hat{X}$, but it still serve the same purpose of getting the difference in position the estimated state and measurement readings.

The sum of uncertainty, S is computed using the predicted co-variance $\hat{P}$, measurement model H and the measurement noise model R.

To determine whether or not the current state, X should be skewed towards the measured state, Z or the estimated state, $\hat{X}$, the Kalman Gain, K is computed using the ratio between predicted uncertainty,

$\hat{P}$ and the sum of uncertainty, S. Due to the K functioning as a ratio between the uncertainties, its value is between 0 and 1. Should the value of K fall to 0, it would mean the radar's measurement, Z could not be trusted and the current state, X is skewed towards the estimated state, $\hat{X}$. Should the value of K approach 1, the current state, X is to be skewed towards the measurement, Z.

As previously mentioned, the vehicles of interest are assumed to be traveling at a constant velocity. Hence, the state transition model, F shall be represented using a constant velocity model. In order to use a constant velocity Kalman Filter, the data pertaining to the external vehicle's position and velocity is required.

$$X_{cv} = \begin{bmatrix} x & \dot{x} & y & \dot{y} \end{bmatrix}^T \tag{6.9}$$

X_{cv} represents the state of the Kalman Filter, which consists of 4 terms, which are:

- x, the position of the external object along the longitudinal axis of the vehicle
- $\dot{x}$, the velocity of the external object along the longitudinal axis of the vehicle
- y, the position of the external object along the latitudinal axis of the vehicle
- $\dot{y}$, the velocity of the external object along the latitudinal axis of the vehicle

$$F_{cv} = \begin{bmatrix} 1 & dt & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & dt \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.10}$$

The state transition model F_{cv} is used to determine the x and y position of the external object at the next time step by assuming said object to be traveling at a constant x and y velocity.

$$Q_{cv} = \sigma \begin{bmatrix} 0.25dt^4 & 0.33dt^2 & 0 & 0 \\ 0.33dt^2 & dt^2 & 0 & 0 \\ 0 & 0 & 0.25dt^4 & 0.33dt^2 \\ 0 & 0 & 0.33dt^2 & dt^2 \end{bmatrix} \tag{6.11}$$

The process co-variance model Q_{cv} is modeled as a Gaussian white noise with a modifiable tuning parameter σ. This is done to account for noise from unknown sources such as vibrations and incline. Given that the direction and magnitude of the noises are unknown, they are treated as random.

$$H_{cv} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{6.12}$$

$$Z = \begin{bmatrix} x & \dot{x} & y & \dot{y} \end{bmatrix}^T \tag{6.13}$$

The measurement model H_{cv} is configured to extract only the x and y position values from the $Z - \hat{X}$ so it can be used to compute for the residual Y.

$$R = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix} \tag{6.14}$$

R represents the measurement noise co-variance model where a represents the position uncertainty for the sensor in the x-direction and b represents the position uncertainty in the y-direction. According to the both radar's data sheet, the uncertainty for position measurements are 0.25 meters (Long Range Accuracy) and 0.5 meters (Mid Range Accuracy) for ESR in both latitudinal and longitudinal axes and 0.5 meters for MRR in both latitudinal and longitudinal axes.

6.2 Data Interpretation and Association

When the radars collect data pertaining to the position and velocity of a vehicle, these data can be represented as discrete detections at different points of time. To a human viewer, the task of matching a detection to an object (tracks) is a relatively simple task that can be done by observing how an object travels in real-time. To a computer however, an algorithm is required to match a detection to an object.

To enable a computer to determine which detection belongs to an object, a method known as Track Association was implemented into the DAQ. Depending on the method, Track Association works by matching a detection at t+1 to a track from t, and assigns a criteria value. If the criteria exceeds a certain threshold, the two are deemed to be belonging to the same object and the detection is joined to the track as the newest point [16]. This process repeats until the object travels outside of the radar array's FOV.

There are two criteria that can be used to match an object to a track: likelihood and age. The probability and likelihood of an object matching a track was calculated using Equation 6.15 and Equation 6.16 respectively.:

$$P(X) = f(Y,S) = \frac{1}{\sqrt{(2\pi)^n det(S)}} exp\left[-0.5 * Y^T * inv(S) * Y\right] \tag{6.15}$$

where Y, the residual and S, the sum of uncertainty from the Kalman Filter are used to calculate the probability of a match between two points at t and t+1. The next section (6.2.1) will explain how to compute the probability of a match between a detection and a track using a Mahalanobis Distance based cost method. The likelihood is modeled using Equation 6.16.

$$L = \frac{P(Y,S)}{P(\hat{0},S)} \tag{6.16}$$

where L is the likelihood of an object matching a track. The residual of the probability at the

denominator is set to 0 to reflect a scenario where the object is assumed to be stationary. Using the stationary scenario $P(\hat{0}, S)$ as a distribution and the measurement scenario $P(Y, S)$ as an event, the DAQ can determine the likelihood of a match between an object and an existing track. Should L exceed the threshold Likelihood, L_{thres}, the detection is deemed to have a match with an existing track and joined via a Modified Hungarian Algorithm.

To ensure a track formed is consistent, the track must transition from tentative to tenure status. In order to do so, a track must be matched to a detection for more than C number of time steps. This method is also known as the age-based Track Association. Once a track has a match for more than C time steps, it is deemed consistent and promoted to tenure, allowing the track's data to be recorded. Should a track travels outside the radar array's FOV or was found coasting (not having a point matched for more than J number of time steps), it is deleted.

Using both likelihood and age-based method, a Track Association method was created for the DAQ.

6.2.1 Mahalanobis Distance based Cost

As previously mentioned, during each scan the radars detects an unknown number of objects, n_d and the detections are then used to create new tracks or compared with the existing tracks created by the Track Management system. In order for the Track Management system to determine which action to take, a Mahalanobis Distance-based matching system is used determine which object detected is a likely match with an existing track and which object is new, therefore needs a new track.

Mahalanobis Distance (MD) is defined as the distance between two points in a multivariate space [9][42]. There are two reasons to use a MD-based Cost to determine a track to detection match namely, MD can measure the distance for correlated points with multiple variables (In this case, two points with longitudinal and latitudinal distance) and exclude any outliers for a dataset based on probability [23][36]. The general equation for MD is represented as the following:

$$d_{Mahalanobis} = \sqrt{(x_a - x_b)^T * C^{-1} * (x_a - x_b)} \tag{6.17}$$

where x_a and x_b represent a set of multivariate variables of size 1 by n for two different objects and C represents a covariance matrix of size n by n. Using MD, the discrepancies between a detection's position and a track's position can be calculated with smaller MD values equating lower discrepancies.

The computation of the MD in the context of the project was done via the equation below:

$$f_c(z_i, x_j) = \sqrt{(z_i - x_j)^T * S_j^{-1} * (z_i - x_j)} \tag{6.18}$$

where z_i represents detection i; x_j represents the current position of track j and S_j represents the sum of uncertainty for track j. Given that S_j functions as a 'record' for how track j changes throughout its lifetime, it was used as the covariance matrix, C to match detection i to track j. The usage of S_j ties the latitudinal and longitudinal position of a track based off its own history.

Should the detection i fall within a reasonable range of track j's possible position at the next time step, a high probability match would be assigned and represented as a small MD value.

For n_t, number of tracks and n_d, number of detected objects, there will be n_t*n_d number of Mahalanobis Distance, for every detection is matched with the existing tracks. This match is represented as a n_d by n_t matrix with each element being a unique Mahalanobis Distance Cost.

$$COST(Z, X) = \begin{bmatrix} f_c(z_1, x_1) & f_c(z_1, x_2) & \dots & f_c(z_1, x_j) \\ f_c(z_2, x_1) & f_c(z_2, x_2) & \dots & \vdots \\ \vdots & & \ddots & \\ f_c(z_i, x_1) & \dots & & f_c(z_i, x_j) \end{bmatrix} \tag{6.19}$$

The $COST(Z, X)$ matrix contains the Mahalanobis Distance for each detected object and existing

tracks with i being the index for detect objects and j being the index for existing track. f_c represents the Mahalanobis Distance between each object detection, z_i and existing track, x_j.

Upon computing the $COST$ matrix, a modified Hungarian Algorithm is used to sort out the minimum MD in the matrix (explained in the section 6.2.2), hence determining which detection is assigned to an existing track and which is used to form new tracks.

6.2.2 Modified Hungarian Algorithm

The Hungarian Algorithm (HA) is an optimization algorithm that determines a set of unique combination actions that generates an optimal output [55][29]. To clarify, assume there n_w number of workers and n_a number of actions with n_w being equal to n_a and time consumption of each action taken by each worker can be represented as a n_w by n_a matrix. Using the Hungarian Algorithm, the following steps are taken to compute a set of unique actions that results in minimum time consumption of the workers:

- Subtract each row of the n_w by n_a matrix by their corresponding minimum
- Subtract each column of the n_w by n_a matrix by their corresponding minimum
- Cover all zeroes in the matrix using horizontal and vertical lines
- If number of lines covering zeroes is equal to n_w, an optimal solution has been reached. If number of lines covering the zeroes is less than n_w, proceed to next step.
- The smallest uncovered element is subtract to each uncovered row and added to each uncovered column. Return to step 3.

Normally, the Hungarian Algorithm would require the number of workers, n_w to be equal to the number of actions, n_a. In the context of this project, this would the number of detection, n_d must be equal to the number of existing tracks, n_t, a statement that is not always necessarily true, for not every detection

will be assigned to an existing track. To resolve this issue, the Hungarian Algorithm was modified to compute an optimal cost for 2D rectangular matrices (where $n_t \neq n_d$) [13].

$$\begin{aligned} g(\mathbf{u},\mathbf{v}) &= \min_{\mathbf{x},\mathbf{x}\geq 0}\left[\sum_{i=1}^{N_R}\sum_{j=1}^{N_C} c_{i,j}x_{i,j} + \sum_{i=1}^{N_R} u_i\left(1-\sum_{j=1}^{N_C} x_{i,j}\right)\right. \\ &\quad \left. + \sum_{j=1}^{N_C} v_j\left(1-\sum_{i=1}^{N_R} x_{i,j}\right)\right] \\ &= \min_{\mathbf{x},\mathbf{x}\geq 0}\left[\mathbf{c}'\mathbf{x} + \mathbf{u}'(1-\mathbf{A}\mathbf{x}) + \mathbf{v}'(1-\mathbf{B}\mathbf{x})\right] \\ &= \min_{\mathbf{x},\mathbf{x}\geq 0}\left[(\mathbf{c}-\mathbf{A}'\mathbf{u}-\mathbf{B}'\mathbf{v})'\mathbf{x}\right] + \mathbf{u}'1 + \mathbf{v}'1 \end{aligned} \tag{6.20}$$

Equation 6.20 is a representation of a cost function to minimize the cost for a 2D rectangular matrix of size N_R by N_C by Crouse [13]. A requirement of the cost function is that N_R must be greater than N_C. u and v denotes a set of Lagrange multipliers used in eliminating constraints in optimization problems. **A** and **B** are represented by Equation 6.21 and 6.22.

$$\mathbf{A} = [\mathbf{I}_{N_R\times N_R} \quad \mathbf{I}_{N_R\times N_R} \quad \dots \quad \mathbf{I}_{N_R\times N_R}] \tag{6.21}$$

$$\mathbf{B} = \begin{bmatrix} 1_{1\times N_R} & 0_{1\times N_R} & \dots & 0_{1\times N_R} \\ 0_{1\times N_R} & 1_{1\times N_R} & \dots & 0_{1\times N_R} \\ \vdots & \vdots & \ddots & 0_{1\times N_R} \\ 0_{1\times N_R} & 0_{1\times N_R} & \cdots & 1_{1\times N_R} \end{bmatrix} \tag{6.22}$$

c is a single row vector generate by stacking the columns of the COST matrix. By solving the cost function above, a unique set of actions for each worker that results in an optimal solution can be computed.

Crouse's algorithm was meant for situations where the number of detection, n_d is greater than or equal to the number of existing tracks, n_t (matching all tracks to detections). To address scenarios where n_t exceeds n_d, Crouse suggested the COST matrix be transposed, allowing the modified HA to match all

detections to existing tracks instead. The implementation of the modified Hungarian Algorithm was done using the python library scipy[6].

CHAPTER 7: SYSTEM VALIDATION

Upon assembly, the array was mounted onto the 2007 Chevy Malibu TVP and tested under two scenarios: Stationary and Moving. The stationary scenario is a test where the TVP is parked at the sidewalk of a lane and used to detect any vehicle cutting across it from the left side. The moving scenario has the TVP travel on a road while staying only on the rightmost lane so vehicles may cut across it from the left.

7.1 Stationary Test

The Stationary Test has found that the array sufficiently tracks objects cutting across the ego vehicle as shown in Figure 7.1 and 7.2. However, there are a number of minor flaws with the tracking system, namely:

- Splintering Tracks: Tracks are formed when detecting approaching objects, but some tracks splintered mid detection. This issue is non-detrimental as the data collection format view these splintered tracks as a single track by identifying them based on their shared IDs.
- Side Detection: The detection performed by the MRR side radar is not ideal. Although the MRR was able to detect vehicles cutting across the left side of the TVP, the detection is not consistent even when taking the Doppler Effect into account. This indicates radars are not suitable sensors for side detections and tracking.

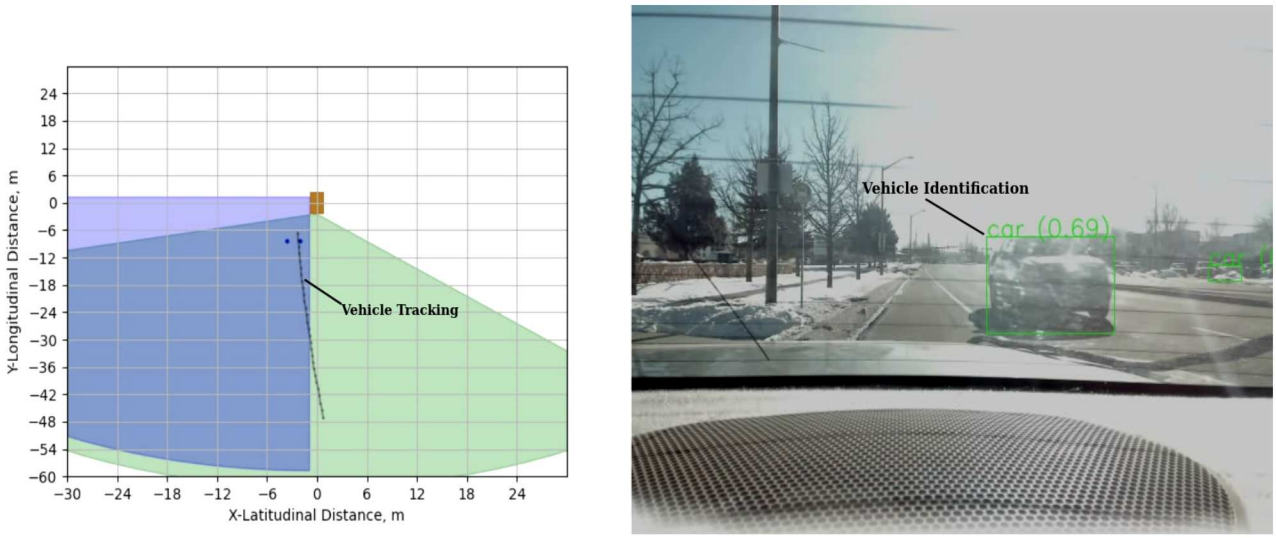

Figure 7.1: Successful vehicle tracking during stationary test, n_t=1.

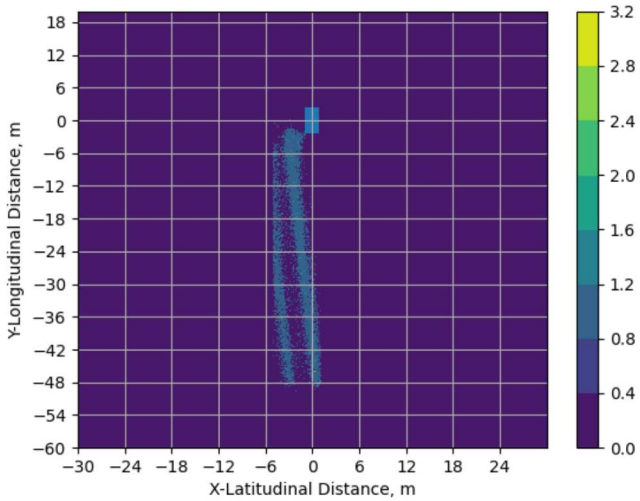

Figure 7.2: Heat map of tracks formed in Stationary Test, n_t=35.

7.2 Moving Test

The Moving Test has also found the array to sufficiently track objects cutting across the ego vehicle as shown in Figure 7.3 and 7.4. Due to the array's configuration, only objects traveling faster than the TVP will be recorded as surrounding vehicles need to travel faster than the ego vehicle to make a lane cut. The

issues that occur in the stationary test also occurred in this test and will be addressed in the Future Work section of this document.

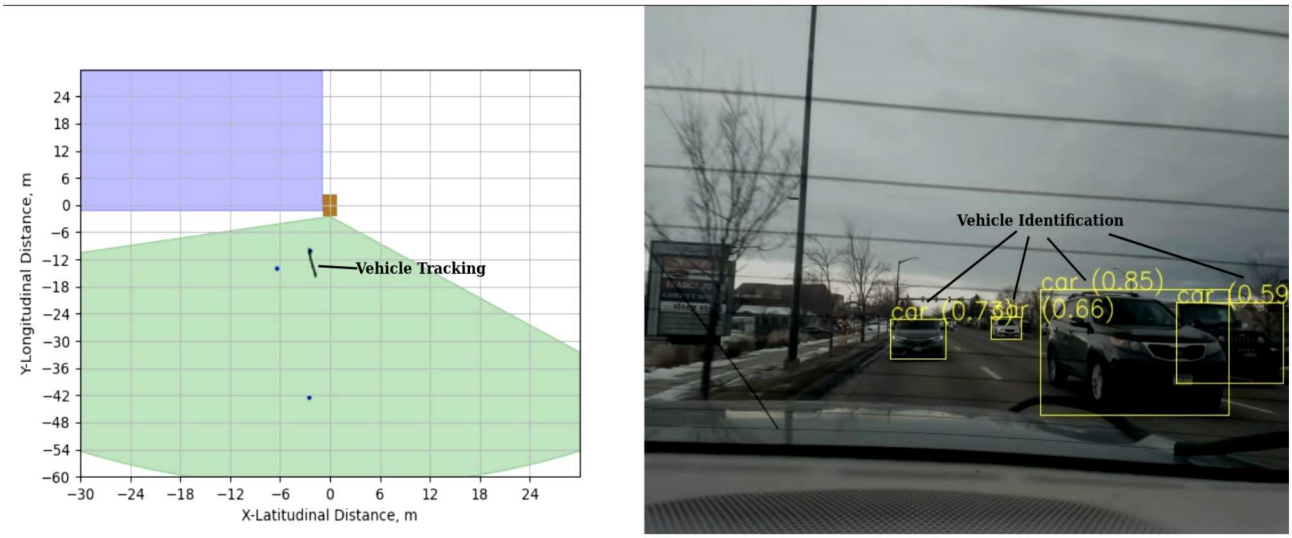

Figure 7.3: Successful vehicle tracking during moving test, n_t=3.

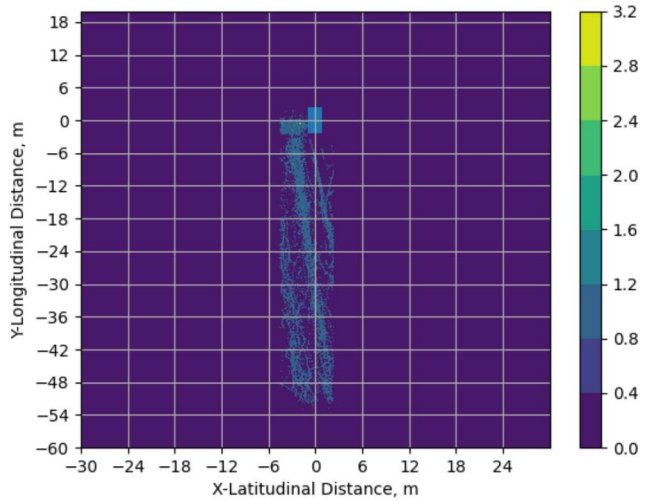

Figure 7.4: Heat map of tracks formed in moving test, n_t=17.

Although the heat map generated is less concise compared to the heat map from the Stationary Test, it should be noted that the TVP was traveling at the speed limit of each road (about 35 mph). This is not an exact replication of the scenario the array will be operating in as the CDOT L3AV is expected to be traveling at 8 mph or less than when it is operating.

CHAPTER 8: SYSTEM APPLICATION AND RESULTS

Due to issues pertaining to scheduling conflicts, the original goal of mounting the array to the CDOT truck and performing data collection was unrealized within the author's time at CSU. However, the array was used to perform another experiment and this section is dedicated to outline changes were made in terms of utilization of the array and data collection. The changes made changes were done in a manner that fulfills as many of the original goals as possible, but not every goal was accomplished in the re-purposed project. For instance, the autonomous status indication was not done in this project as it was deemed unnecessary to observe the travel trajectories of HDV in residential lanes.

8.1 Leeway Given in Dedicated Delivery Lane

Given that the array was proven capable of tracking surrounding vehicles, an idea to analyze travel trajectories of human drivers when encountering a double parked delivery vehicle was proposed. The event of interest is described as the following sequence:

1. A delivery vehicle double parks and fully occupies a lane.
2. A HDV drives up from the rear of the delivery vehicle.
3. HDV spots the delivery vehicle and executes a left-lane cut.
4. Repeat 2 and 3 for 20 minutes.

The item of interest here is step 3, namely the trajectory of the approaching HDV and the velocity profile of said HDV. Using the velocity profile of the HDV, the Fuel Economy (FE) of the HDV performing left lane cuts were computed using a FASTSIM model.

The author believes research into this topic will provide valuable results to either support or dissuades plans to construct a dedicated delivery lane for delivery vehicles. The hypothesis is the presence of delivery

vehicles obstructing residential lanes will lead to HDV driving with more caution. This will cause HDV to slow down to perform a lane cut before re-accelerating, resulting in a lower FE. If the delivery vehicle was parked in a dedicated delivery lane, the HDV will have no need to decelerate, allowing them to maintain a cruise velocity and therefore a higher FE, reducing the amount of fuel consumed per travel.

8.2 Method for Data Collection and Analysis

To determine how a HDV would react upon spotting a delivery vehicle, a Chevy van was disguise as a delivery vehicle and used to perform a double parking at specific residential lanes in Fort Collins. Using magnets, the array was mounted onto the van without introducing modifications to the van itself. (See Figure 8.1)

Figure 8.1: Chevy van disguised as delivery vehicle and magnetically mounted radars.

To ensure consistent generation of results, the data collections for left lane cuts were performed with the following criteria:

1. Data collection starts at 1:00 pm noon.
2. Data collection takes place at W Myrtle, E Myrtle, Meldrum and Strover street.
3. Each data collection session takes place for a period of 20 to 30 minutes.

W Myrtle, E Myrtle, Meldrum and Strover street were chosen as they represent the busiest residential

streets in Fort Collins with relatively low speed limit (25 to 30 mph). Additionally, each street had only one lane on both sides and such restriction meant all approaching HDV would need to perform a left lane cut by utilizing the opposite lane should they wish to continue their travel. (See Figure 8.2)

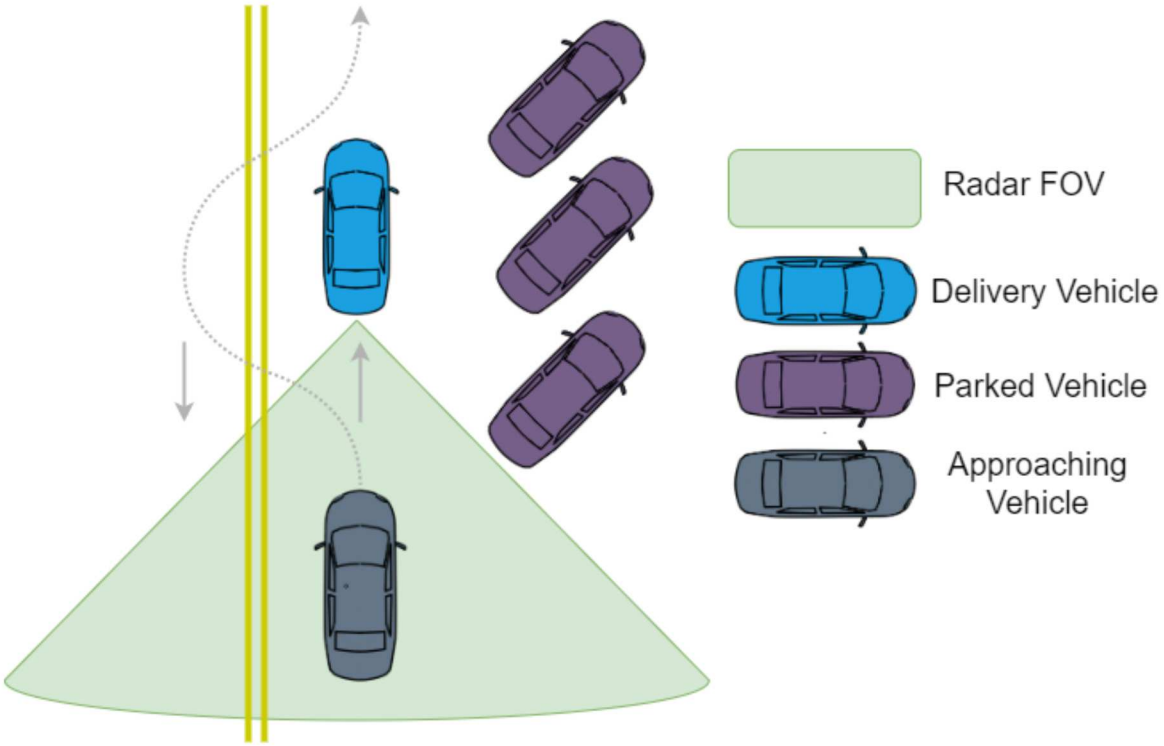

Figure 8.2: An example of how HDV will react upon spotting a double parked delivery vehicle.

To ensure a seamless data analysis process, each data collection session were logged in the form of a Comma-Seperated Values (CSV) file with each track being assigned a unique ID and logged with their corresponding latitude and longitudinal positions in meters. (See Figure 8.3)

Time (s)	ID0	X0	Y0	ID1	X1	Y1	ID2	X2
265.6	691	-2.33417395000112	-13.5482378435265					
265.7	691	-2.33616316036053	-12.8770835023801					
265.8	691	-2.4293193149402	-12.2119710450487					
265.9	691	-2.51999778322695	-11.3757917214931					
266	691	-2.57998852203224	-10.5639711793623					
266.1	691	-2.57758481616784	-9.78356398926171					
266.2	691	-2.57454693626582	-9.01626387132481					
266.3	691	-2.62584217355834	-8.27338536343074					
266.4	691	-2.60642093523126	-7.58157408429446					
266.5	691	-2.57893273192997	-6.88307179325451					
266.6	691	-2.49707686687898	-6.27871956123757					
266.7	691	-2.40920359990338	-5.75872220965539	696	3.43911849592314	-50.9366014683707		
266.8	691	-2.34271754776129	-5.16696298854485	696	3.33924083276208	-50.4317320974632		
266.9	691	-2.30749151502881	-4.60569555720935	696	3.1993060050781	-49.9660221383217		
267	691	-2.28974131657199	-4.01733549338776	696	3.1473130216652	-49.4743826448549		
267.2	691	-2.25567667873289	-3.41788075262418	696	3.10470878572967	-49.0046309984022		
267.3	696	3.00880519215189	-48.5589204447404					
267.4	696	2.9289783260697	-48.0924432724521	701	-7.85530375072321	-45.3223907384455		
267.5	696	2.89189916800993	-47.6434787089139	701	-7.80598419618659	-45.3089621590944		
267.6	696	2.82585304316631	-47.1144298813609	701	-7.73916740943019	-45.2699386373131		
267.7	696	2.74401615161012	-46.5900716457859	701	-7.65670020884193	-45.2140352281994		
267.8	696	2.6309124000394	-46.1010350639818	701	-7.53423706561663	-45.1493023806782		
267.9	696	2.53657297052266	-45.6076480231166					
268	696	2.47621448223092	-45.1098832170015					
268.1	696	2.40375712816425	-44.6117914591561					
268.2	696	2.3399238985662	-44.1123319626041					

Figure 8.3: How track data is stored. Every 3*n+1 column represents an ID while 3*n+2 and 3*n+3 column represents latitude and longitudinal positions between detected object and ego vehicle respectively.

Using python, the track data for each session were loaded and each individual track were called and compiled using their unique IDs. To generate velocity traces of each track, the position points were fitted into a 4th order polynomial using python's polyfit function from numpy library (see Figure 8.4). A 4th order polynomial is chosen for position trace as it enables the 2nd order derivative of the position trace to exhibit acceleration and deceleration behavior without introducing additional dynamics.

$$P_x(t) = c_{x0} * t^4 + c_{x1} * t^3 + c_{x2} * t^2 + c_{x3} * t + c_{x4} \tag{8.1}$$

where c_{xi} represents the polynomial coefficients for the 4th order latitudinal position trace.

$$P_y(t) = c_{y0} * t^4 + c_{y1} * t^3 + c_{y2} * t^2 + c_{y3} * t + c_{y4} \tag{8.2}$$

where c_{yi} represents the polynomial coefficients for the 4th order longitudinal position trace.

$$V_x = dP_x/dt \tag{8.3}$$

$$V_y = dP_y/dt \tag{8.4}$$

$$V = \sqrt{V_x^2 + V_y^2} \tag{8.5}$$

where V_x represents the latitudinal velocity, V_y represents the longitudinal velocity and V represents the magnitude of the velocity.

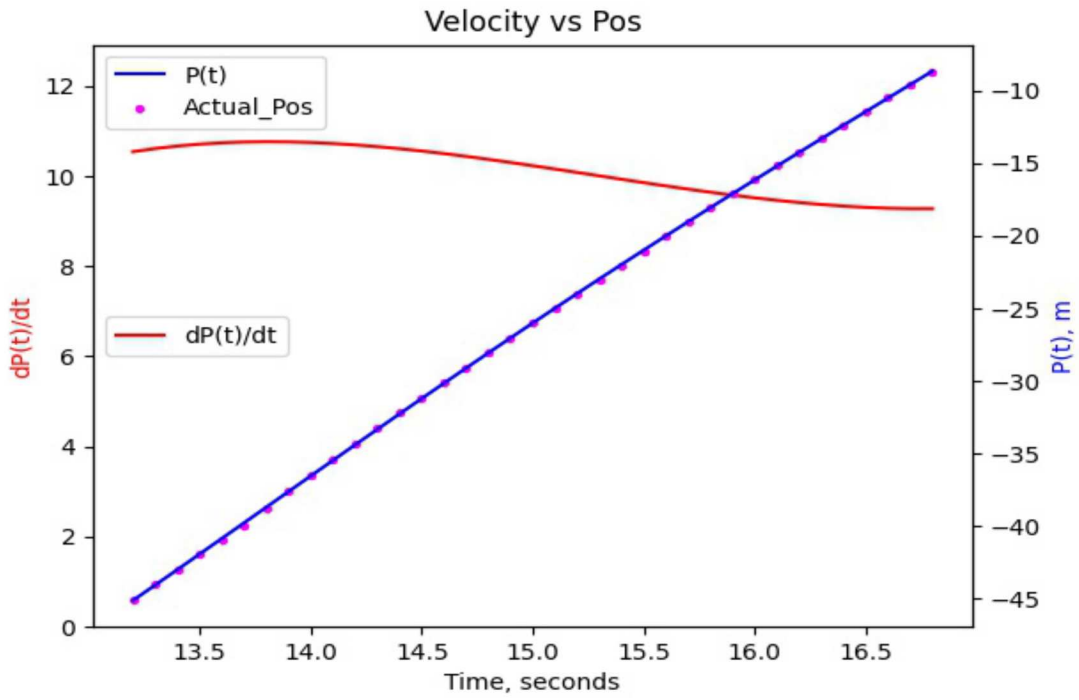

Figure 8.4: An example of generating a position polynomial (blue line) using collected position data point (pink dots) and deriving a velocity trace (red line) as a 1st order derivative.

Using the generated velocity profile in conjunction with FASTSIM's sim_drive method, the instantaneous fuel consumption of each track was generated using a 2016 Toyota Corolla as the vehicle model. The aforementioned vehicle model was chosen due to its characteristics as a gasoline-powered vehicle.

8.2.1 Trace Extension

A minor issue arise when using the generated velocity profile to calculate the instantaneous fuel consumption of each track, namely the tracks used were too short to generate viable fuel consumption results (Generally about 4 to 8 seconds worth of data) as FASTSIM assume the vehicle always starts up in the first 10 seconds of the simulation. Although a recommended way to resolve this issue is to collect longer traces, such data collection is not possible due to the limit in radar detection range.

Using the assumption that vehicles are traveling at a constant velocity prior to encountering an obstructing delivery vehicle, an idea was propose to extend the head of each trace. By setting each velocity trace to x number of seconds, the end of the velocity trace is made up of the original velocity trace while the front end is made up a constant velocity trace. This would extend the velocity trace long enough to allow the vehicle model FASTSIM to enter a steady state before execute the original velocity trace. (See Figure 8.5)

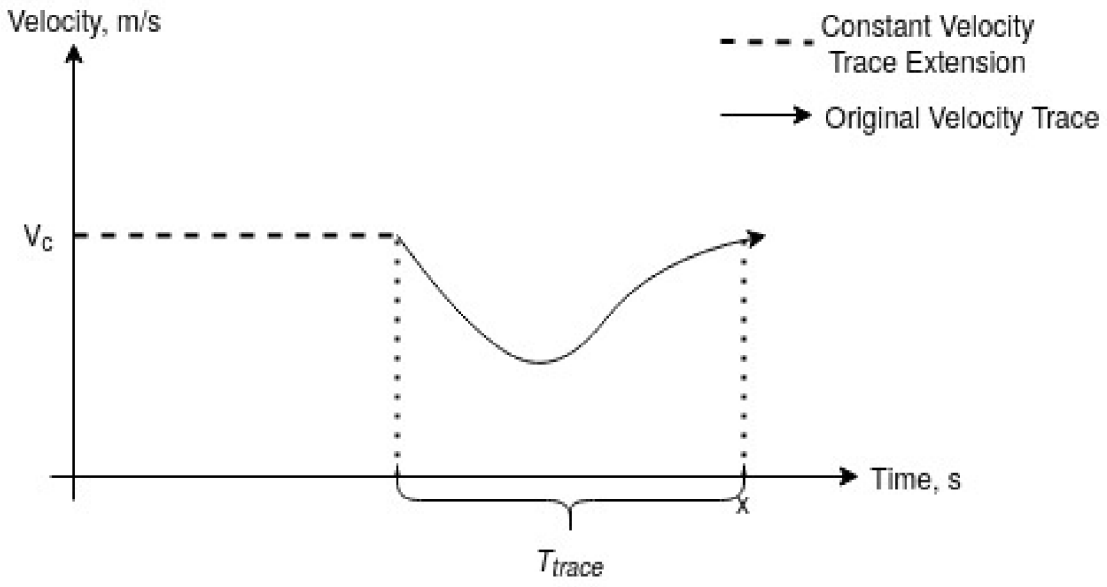

Figure 8.5: An example of extending a trace of time, T_{trace} to x number of seconds.

Using a 30 second extension time, the resulting instantaneous fuel consumption had the section corresponding to the original velocity trace extract and used to calculate the fuel economy of the original

velocity trace.(See Figure 8.6)

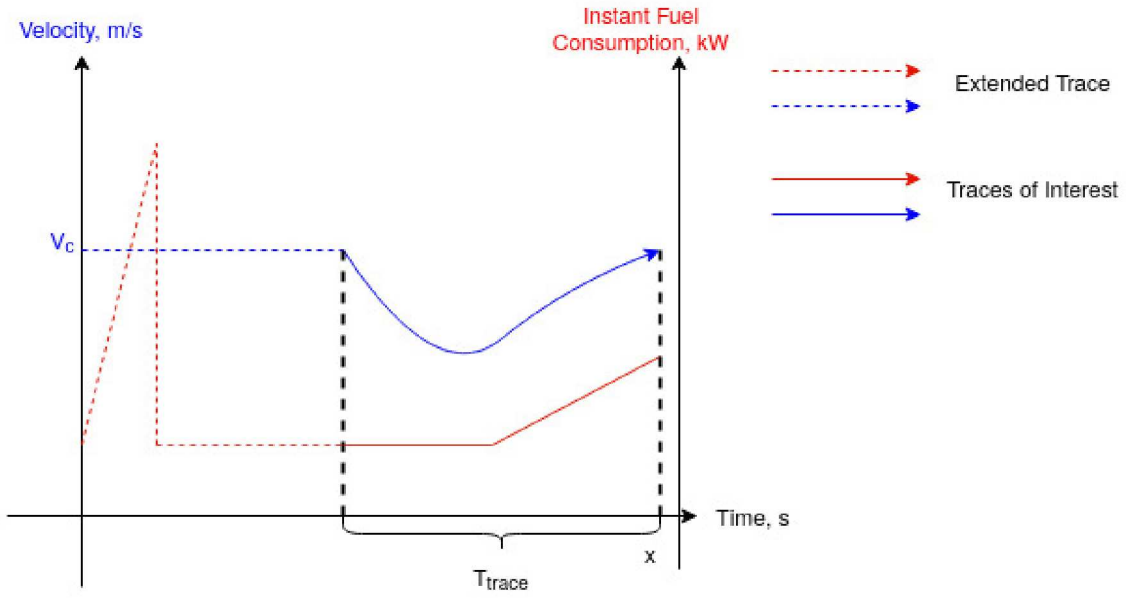

Figure 8.6: Only the T_{trace} section of instantaneous fuel consumption is extracted and used to calculated fuel economy.

The calculation of the Fuel Economy (FE) is performed using the equation 8.6:

$$FE = \int_{x-T_{trace}}^{x} \frac{FC(t)}{V(t)} dt \tag{8.6}$$

The individual FE of each trace are compiled into a histogram to represent the FE distribution for both blocking and non-blocking scenario. (See Figure 8.7)

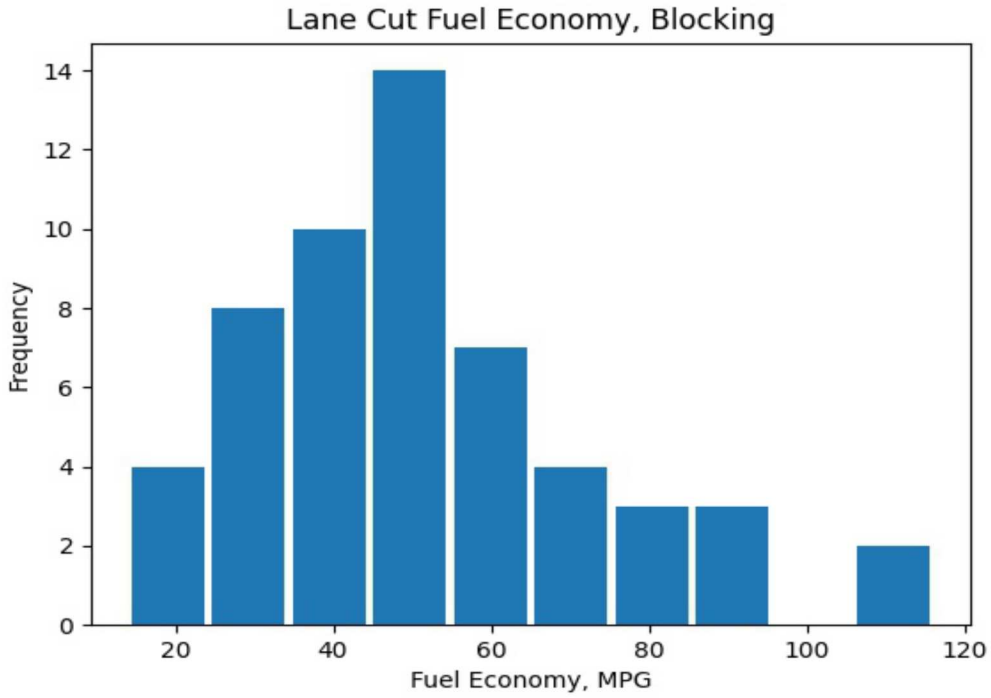

Figure 8.7: An example of a histogram for FE distribution in Blocking scenario.

Additionally, the average velocity of each trace was calculated using equation 8.7 and represented as a histogram distribution. (See Figure 8.8)

$$V_{avg} = \frac{\sqrt{(P_x(t_n) - P_x(t_0))^2 + (P_y(t_n) - P_y(t_0))^2}}{t_n - t_0} \tag{8.7}$$

where t_0 and t_n represent the start and end time of each trace respectively.

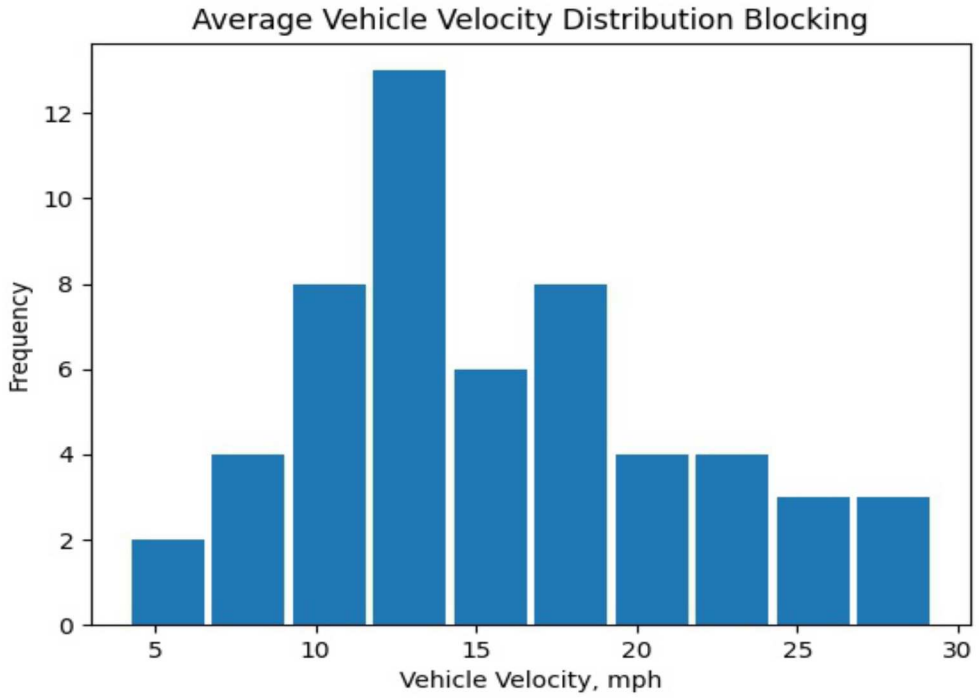

Figure 8.8: An example of a histogram for average velocity distribution in Blocking scenario.

8.2.2 Test for Statistical Difference

To identify whether the data collected for both scenario represented distribution under two different scenarios, the dataset was assumed to be skewed distributed and a Mann-Whitney test was used to determine if said dataset had a statistical difference. The data is assumed to be skewed distributed for the following reasons:

- In a blocking scenario, there would be more vehicle instance of vehicle traveling at lower velocity to perform a safe lane cut than there are vehicle performing lane cut at cruise velocity. This would cause the average velocity distribution in the blocking scenario to become more skewed towards lower velocity values.
- In a non-blocking scenario, there would be more vehicle instance of vehicle traveling at the speed limit of the lane with reduced instances of vehicle traveling at lower velocities. This would cause

the average velocity distribution in the non-blocking scenario to become more skewed towards higher velocity values.

- Given that Fuel Economy is a by product of velocity behavior, it can be assumed that if velocity distribution exhibits a skewed distribution, Fuel Economy would also exhibit a skewed distribution.

A Mann-Whitney is a non-parametric test conducted to determine if two samples come from the same population and share the same median[22]. Due to the skewed distribution assumption, a two independent sample T-test is not considered as the method requires two sample groups with normal distribution[21].

In the Mann Whitney tests, there is a null hypothesis, H_0 and an alternative hypothesis, H_1. The null hypothesis is an accepted fact that researchers work to nullify while alternative hypothesis is a validated hypothesis should the null hypothesis be negated. The nullification of the null hypothesis is dependent on the p-value of the study, which serves as a measurement to reject the null hypothesis with smaller p-values equating to stronger rejection of the null hypothesis. By default, the $p_{critical}$ value is set to 0.05.

For this research, the null, H_0 and alternative, H_1 hypothesis of the Mann-Whitney test are as follows:

- H_0: The FE/Vel population of the blocking and non-blocking scenario are equal (share the same median).
- H_1: The FE/Vel population of the blocking and non-blocking scenario are not equal (do not share the same median).

In a Mann Whitney T-test, the null hypothesis is rejected if the resulting minimum U-value, U_i is less than the $U_{critical}$ value. The $U_{critical}$ value is dependent on the sample size of the groups of interest. For large sample size comparison, the $U_{critical}$ value is estimated via equation 8.9:

$$U_{critical} = z * \sigma_U + m_U \tag{8.8}$$

where z represent the z-score determine by the critical p-value of the test, σ_U and m_U are determined

by equation 8.10 and 8.11 respectively.

$$m_U = n_1 * n_2/2 \tag{8.9}$$

$$\sigma_U = \sqrt{n_1 * n_2 * (n_1 + n_2 + 1)/12} \tag{8.10}$$

where n_1 and n_2 represent the sample size for two different groups. The calculation for U_i is done using equation 8.12:

$$U_i = n_i * n_j + \frac{n_i * (n_i + 1)}{2} - R_i \tag{8.11}$$

where n_i and n_j represents the sample size for the two groups of study in the test and R_i represents the sum of rank for sample group i [32]. The i in term U_i denotes that the U-value needs to be calculated for both sample group 1 and sample group 2 and the smaller of the two will be compared to $U_{critical}$ to nullify or support the null hypothesis.

8.3 Results

The Fuel Economy (FE) distribution for both blocking and non-blocking scenario are represented by Figures 8.9 and 8.10.

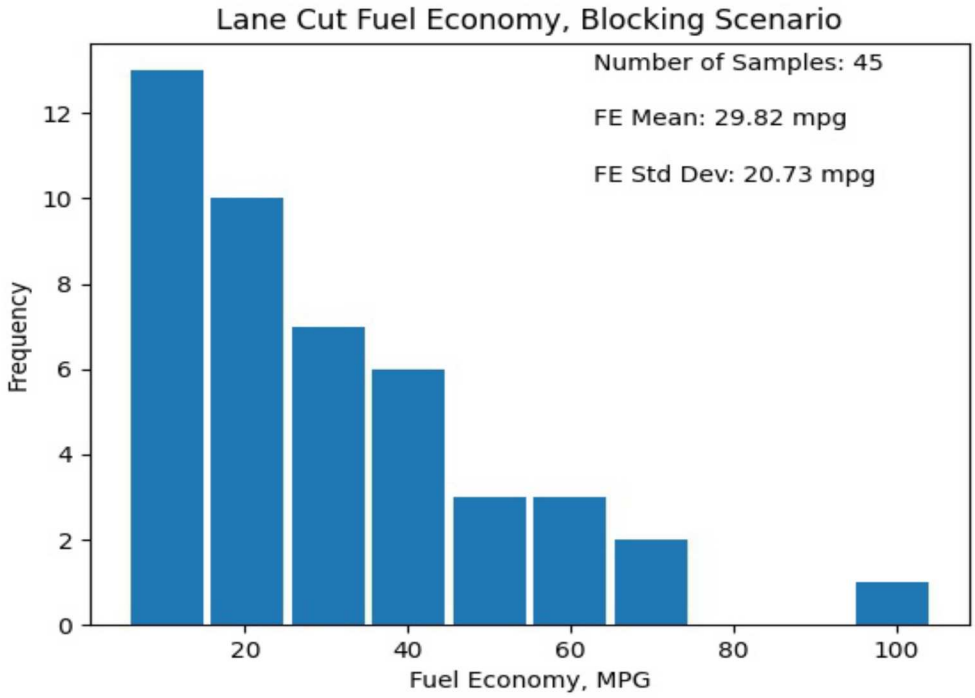

Figure 8.9: Histogram distribution of Fuel Economy (FE) in blocking scenario. Sample size, n_1=45

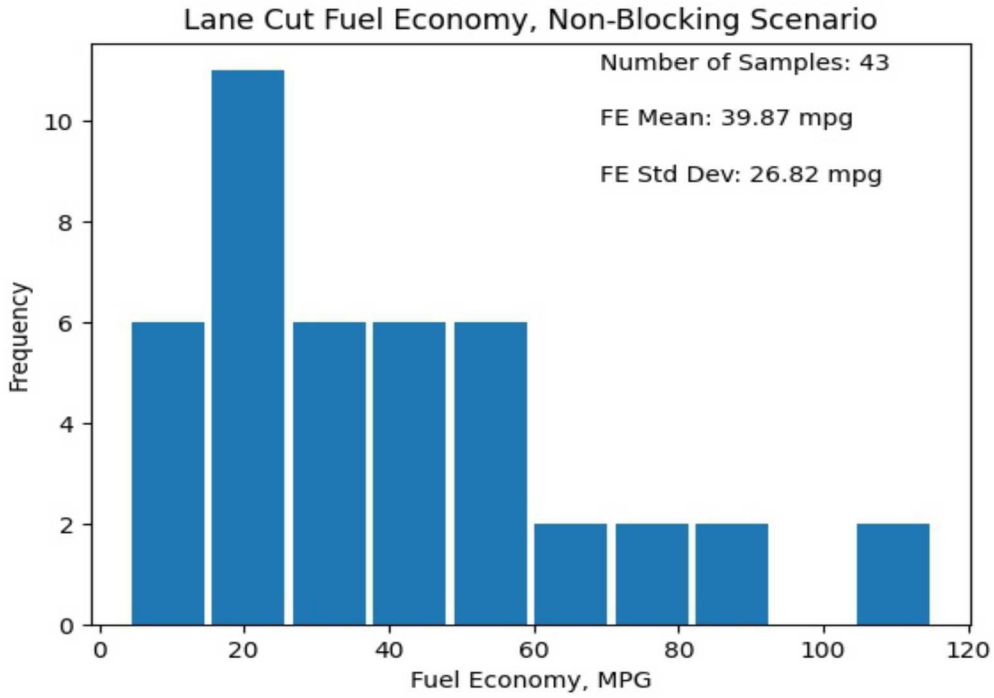

Figure 8.10: Histogram distribution of Fuel Economy (FE) in non-blocking scenario. Sample size, n_2=43

Table 8.1: Table for FE Mean and Standard Deviation with for Blocking and non-Blocking Scenario

Scenario	FE Mean, MPG	FE Standard Deviation, MPG
Blocking	29.82	20.73
Non-Blocking	39.87	26.82

Performing a Mann Whitney test using the blocking and non-blocking scenario FE distribution as the sample groups yielded the following results.

Table 8.2: Table for results of Mann Whitney Test for FE distribution

$U_{critical}$	U-value	p-value
1202.3	749	0.034

Given that U-value is less than $U_{critical}$ and the p-value is less than 0.05, there is enough evidence to

disprove the null hypothesis of the Mann Whitney test, meaning the population of the FE distribution for both groups are not equal (medians are not equal).

Velocity (FE) distribution for both blocking and non-blocking scenario are represented by Figures 8.11 and 8.12.

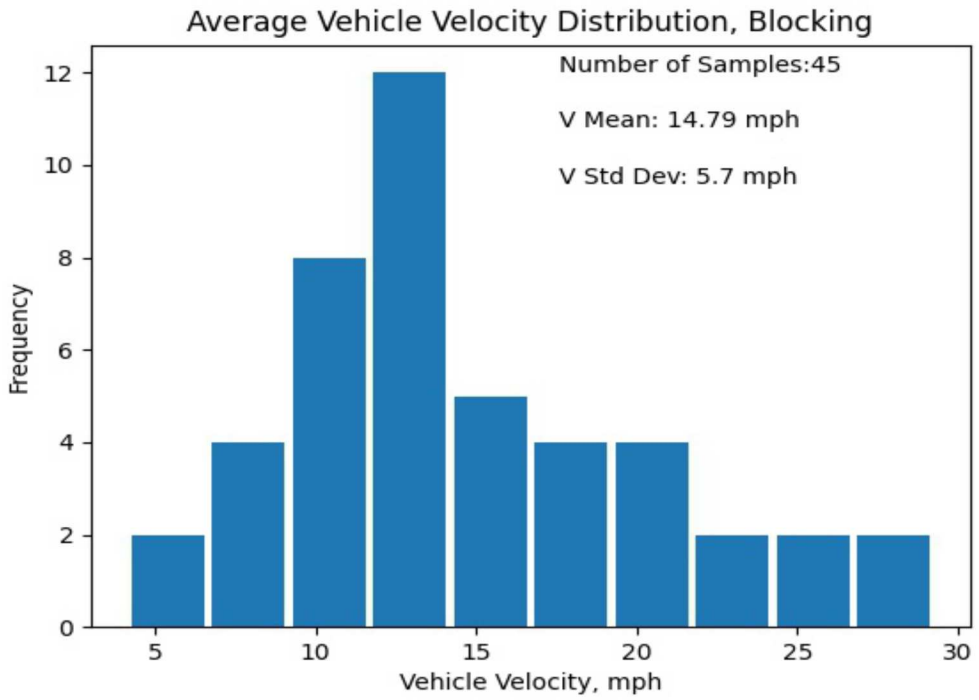

Figure 8.11: Histogram distribution of incoming vehicle velocity in blocking scenario. Sample size, n_1=45

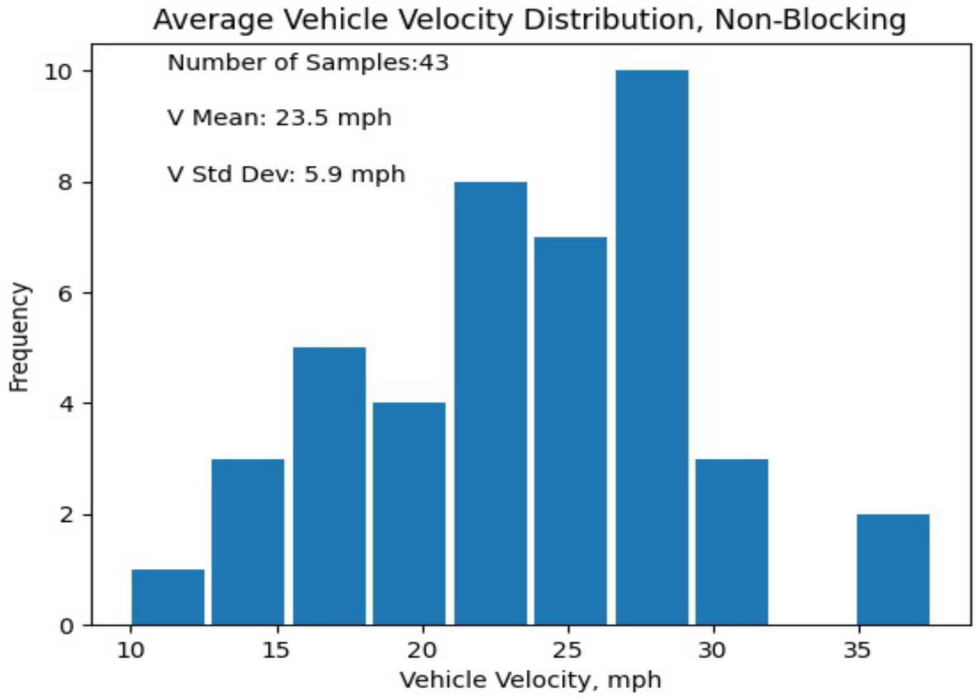

Figure 8.12: Histogram distribution of incoming vehicle velocity in non-blocking scenario. Sample size, n_2=43

Table 8.3: Table for Velocity Mean and Standard Deviation with for Blocking and non-Blocking Scenario

Scenario	Velocity Mean, MPH	Velocity Standard Deviation, MPH
Blocking	14.79	5.7
Non-Blocking	23.5	5.9

Performing Mann Whitney test using the blocking and non-blocking scenario velocity distribution as sample groups yielded the following results.

Table 8.4: Table for results of Mann Whitney Test for Velocity distribution

$U_{critical}$	U-value	p-value
1202.3	292	$8.78*10^{-9}$

The results of the Mann Whitney indicates that the null hypothesis is nullified as the resulting p-value

is less than 0.05 and U-value is less than $U_{critical}$ respectively. There is enough evidence to disprove the population of the velocity distribution (medians) of the two groups are equal.

8.3.1 High Fuel Economy Behavior

This section is dedicated to explain instance of high Fuel Economy (FE) from the results. The high fuel economy behavior is a result of two possible events, vehicles decelerating or maintaining a high travelling speed when approaching Testing Vehicle Platform (TVP).

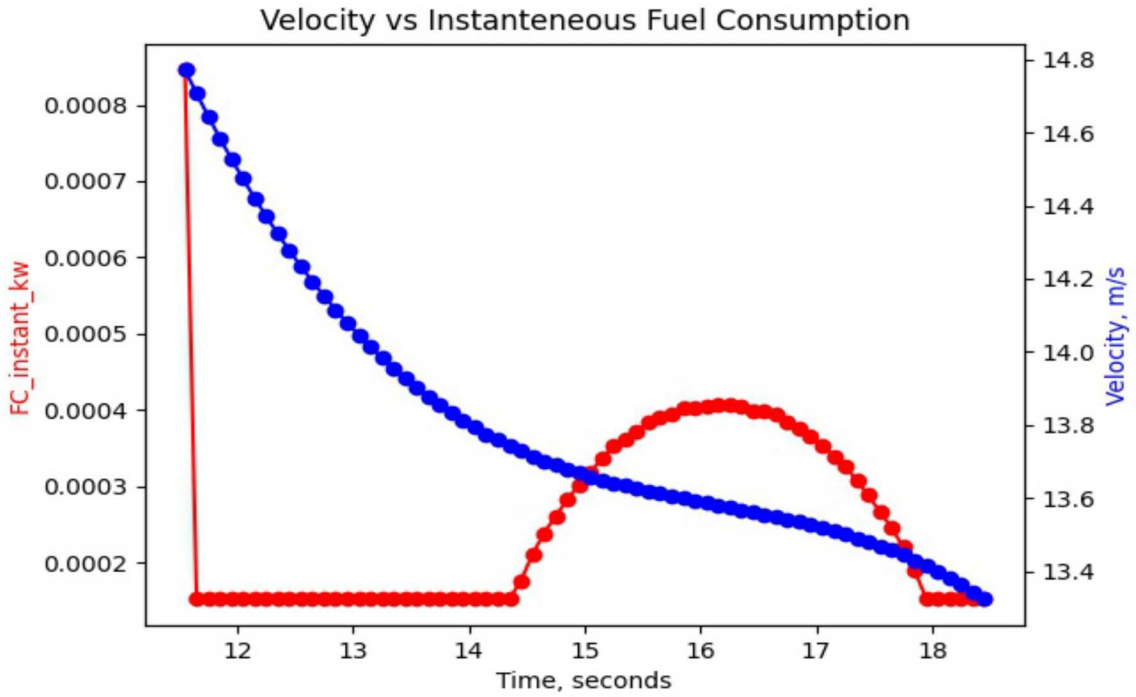

Figure 8.13: Instantaneous fuel consumption for decelerating vehicle. Fuel is consumed to reduce the rate of deceleration.

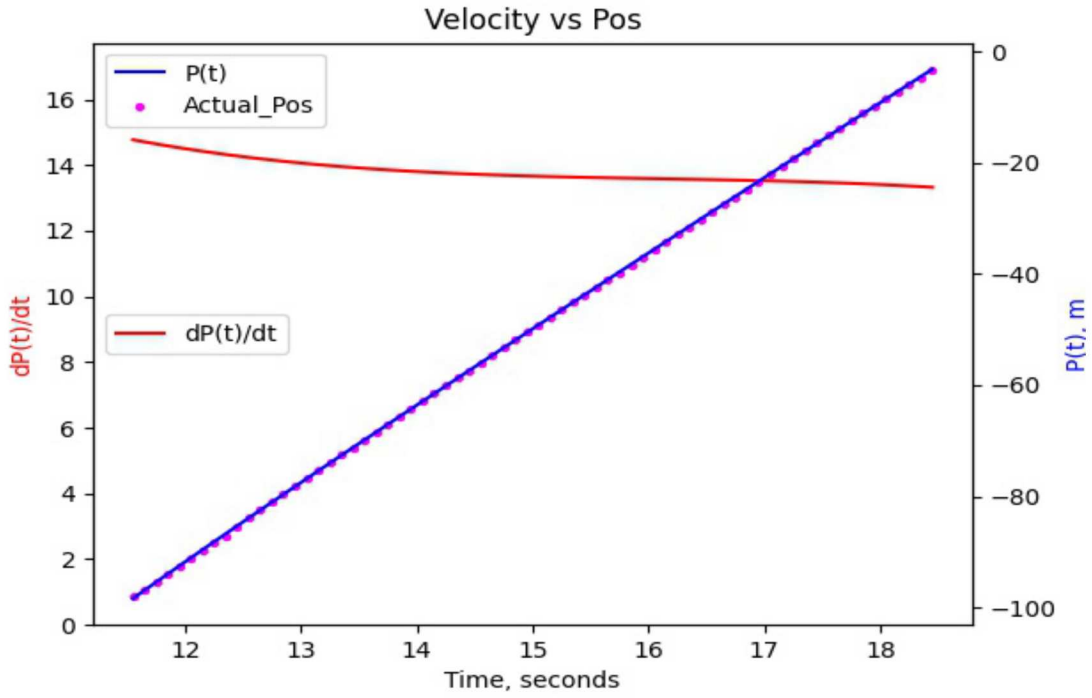

Figure 8.14: Position and velocity trace for decelerating scenario.

Figure 8.13 and 8.14 outline the velocity behavior for trace with 115 miles per gallon fuel economy. Due to the incoming vehicle travelling at high speed and decelerating, the resulting fuel economy is higher due to covering more distance with less fuel consumption.

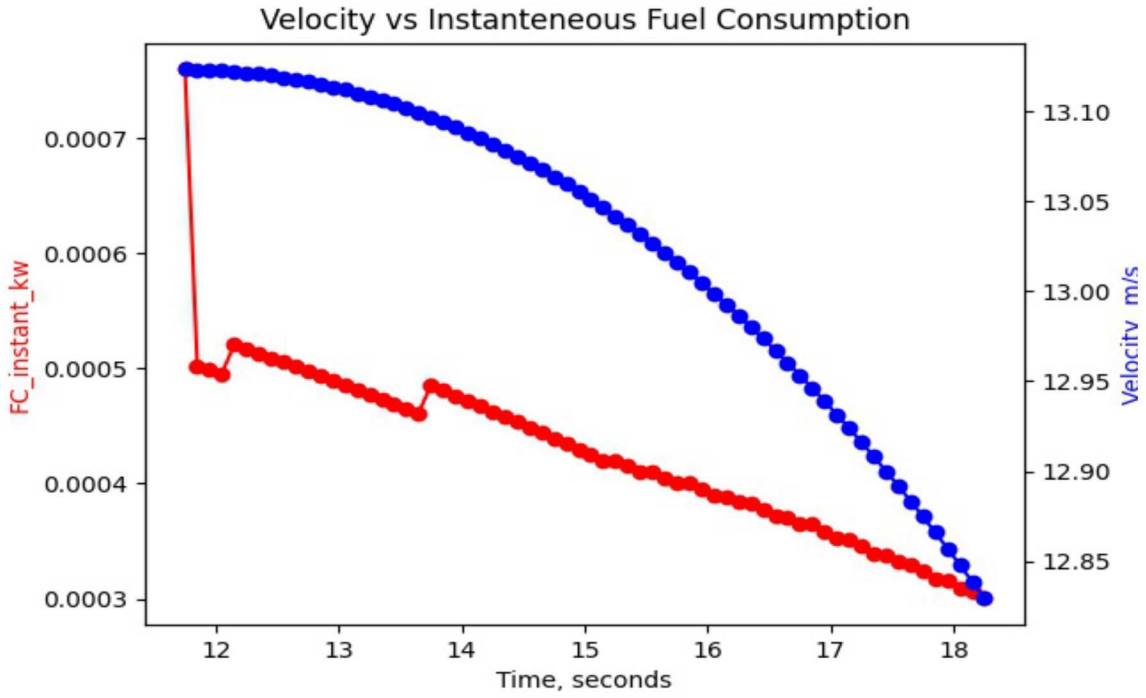

Figure 8.15: Instantaneous fuel consumption for vehicle maintaining relatively constant speed. Reduction in instantaneous fuel consumption is a result of incoming vehicle exhibiting less force to reduce deceleration as time passes.

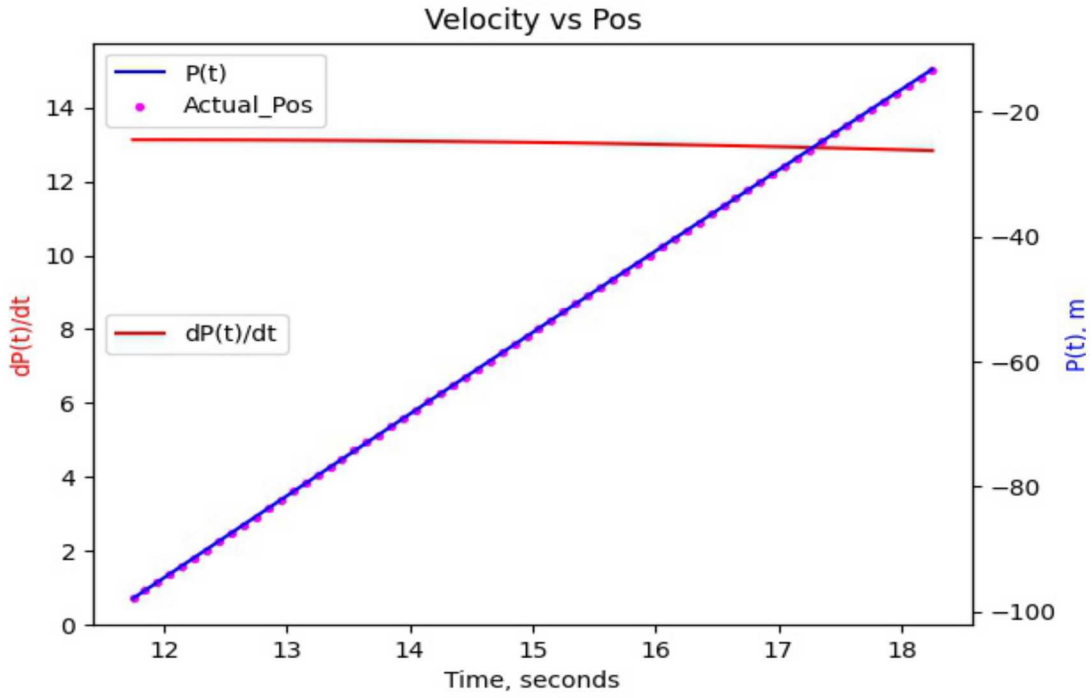

Figure 8.16: Position and velocity trace for relatively constant speed scenario.

Figure 8.15 and 8.16 outline the velocity behavior for trace with 64.6 miles per gallon fuel economy. Due to the incoming vehicle travelling exhibiting little change in speed, the resulting fuel economy is higher due to lower total fuel consumption.

8.4 Conclusion

Reviewing the results obtained from the study, the Mann Whitney tests indicates that there is a difference in velocity and fuel economy behavior of incoming vehicle between blocking and non-blocking scenario. This means that the presence of an obstruction results in lower fuel economy of incoming vehicles as a result of the need to decelerate to safely perform a lane cut.

It should be noted that this study only focuses on the residential area of Fort Collins, which are considered lightly packed compared to residential area of larger cities such as Denver. Potential future work

could focus on residential areas of the aforementioned city to observe if a similar velocity and Fuel Economy behavior is exhibited as the degree of traffic lane occupancy could have different impact on the velocity and Fuel Economy behavior of HDV.

Additionally, it should be noted that the tracking array designed in this project was yet to be use for CDOT's data collection. This data collection process will resume in the period of Summer 2022 once CDOT's L3AV is operational.

CHAPTER 9: DESIGN CHALLENGES

During the design of the TVP, several challenges were encountered and resolved to the best of the research lab's abilities. Although a majority of the challenges encountered were resolved, a small number of them were given 'band-aid' solutions. This section is dedicated to outline the problems encountered in these challenges along with recommendations on how to fix them.

9.1 Sensor Expectation and Recommendation

As previously mentioned, the MRR was mounted at a 45 degree angle to mitigate the Doppler Effects on the radar. Although this provided a better side detection compared to mounting the MRR in a flat position, the MRR still suffers from inconsistent tracking of side vehicles. After several experiments, the source of the inconsistencies was found to be:

- Distance between the side vehicle and MRR was too close during side detection (About 0.8 to 0.5 m). This is shorter than expected. Based on the MRR's datasheet, a minimum separation distance of 1 meter is required for the MRR to detect surrounding objects. This means radars are insufficient in performing side detection as there is not enough separation distance for the radar to consistently perform a detection.

A recommended solution would be to use a different sensor. However, it should be noted that in researching work done on autonomous vehicle, no recommendations were given to attach a sensor at the side of a vehicle, with the exception of proximity sensors. This indicates that attaching a sensor to perform side detection might be a fruitless venture. If such indication is true, the recommended solution would be to implement a 'constant speed zone' in the tracking array to assume the vehicle is traveling at constant speed while traveling beside the TVP.

9.2 Placement of Radar Array on CDOT Truck

In the initial stages of the project, it was desired for the laptop and the GPU be stored within the cabin of the CDOT truck. Due to issues pertaining to overhanging wires, this was not deemed possible and the laptop and GPU would need to be placed side-by-side with Radar Array Case. This means another case storing the laptop and GPU would need to designed to protect the laptop and GPU from the environment during operation.

9.3 Mounting of Radar on CDOT Truck

Due to the scarce number of viable mounting points on the CDOT Truck, positions that are normally considered 'too high' for the ESR and MRR were chosen as mounting points. Should the array encounter detection issues when tested with the CDOT truck, one of the following two idea is recommended:

- Points of lower height should be chosen. This is unlikely as the CDOT truck doesn't have any points that are considered low enough to mount unless physical modifications are allowed on the CDOT truck.
- The mounting brackets of the truck is modified to allow for adjustable. Although this is the more preferable option, care must be taken as sections of the bracket will be hanging as opposed to being directly mounted onto a surface, making the sensor vulnerable to vibration.

9.4 Computation Time and Vectorization of Algorithm

Although the code used in the tracking array is functional and sufficiently fast in terms of computation time, the code can be improved by vectorizing certain aspects of the algorithm. The current iteration of the algorithm performs a majority of its computation through the use of 'for' loops, meaning the computation time increases as the number of data points detected by the radar increases. By vectorizing the algorithm,

the increase in computation time of the algorithm won't be as steep as using 'for' loops when detecting an increasing number of data points. However, certain issues needs to be address for this to happen:

- Matrix in Matrix problem: Normally, should one wish to perform a calculation via vectorization, the matrix used must contain only 1-D elements. This poses a challenge as the matrices used in this algorithm requires the use of matrices containing 2-D elements (For instance: Measurement z_i, used in MD calculation, represents longitudinal and latitudinal position, hence 2-D). Unless this issue is resolved, a vectorization of the algorithm is not possible.
- Different data structure required: The current iteration stores information of each data point in the form of a list/dictionary. Although this enables easy access of specific information during run-time, it is not optimally configured for vectorization for values needs to be extracted individually via keys of the dictionary. Should vectorization be desired, another means of storing data point information is required.

9.5 Track Recording for non-Blocking Scenario

A minor issue that arose during data collection for non-Blocking scenario was the positioning of the delivery vehicle. Some of the streets chosen for the experiment had diagonal parking lanes, meaning the delivery vehicle needed to be parked 30 degrees diagonally relatively to the direction of travel. It was found that collecting data with the aforementioned parking position caused the Doppler Effect to distort the distance and velocity readings collected by the radar. (See Figure 9.1)

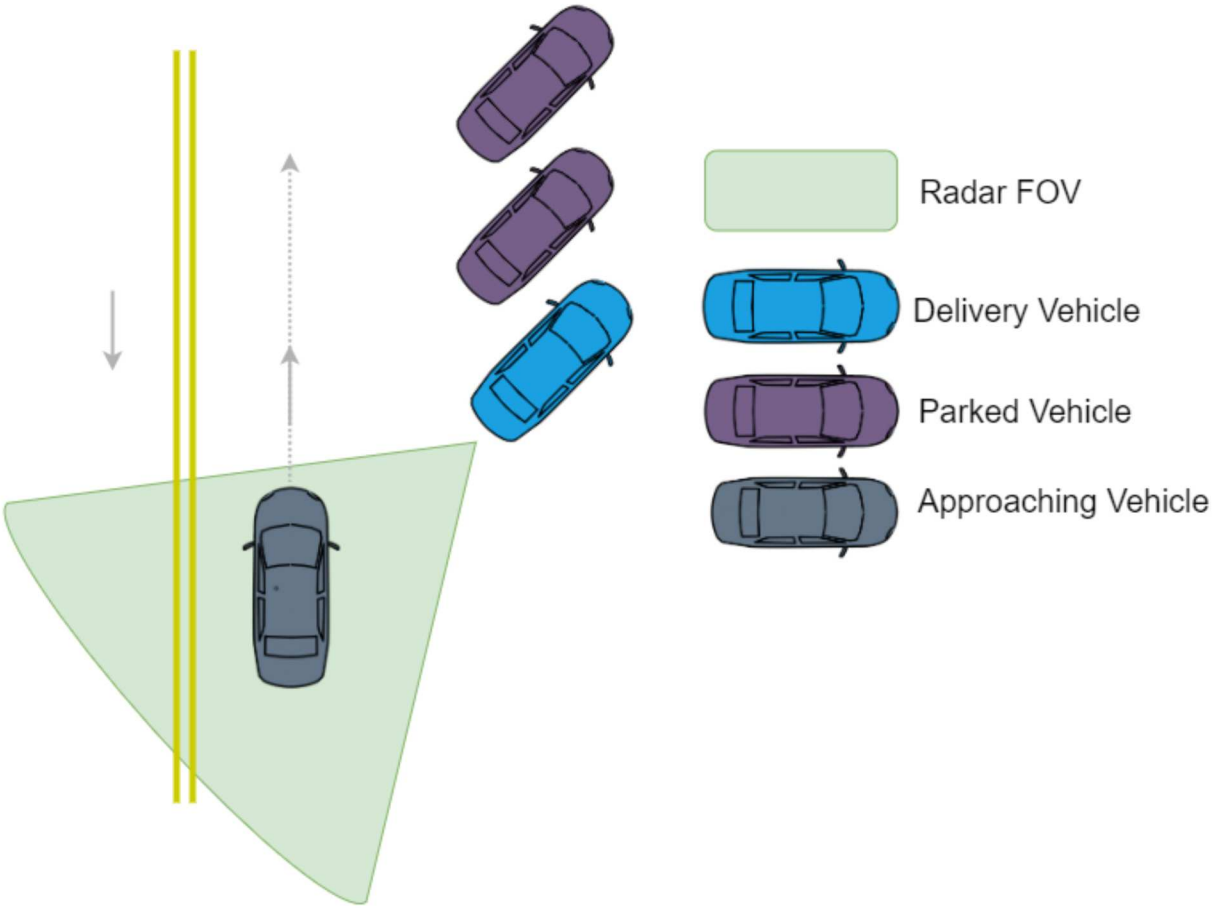

Figure 9.1: How data collection for non-blocking scenario was originally set up.

Normally, this issue cause be resolved by attaching the radar to the corner of the delivery vehicle, but the area of attachment was found to be non-magnetic meaning physical modifications must be performed should such attachment be desired. To avoid such need, it was decided that data recording for non-Blocking Scenario to be performed in the same manner as the blocking scenario, with the delivery vehicle parked in a manner that is parallel to the lane's direction of travel without obstructing traveling vehicles. (See Figure 9.2)

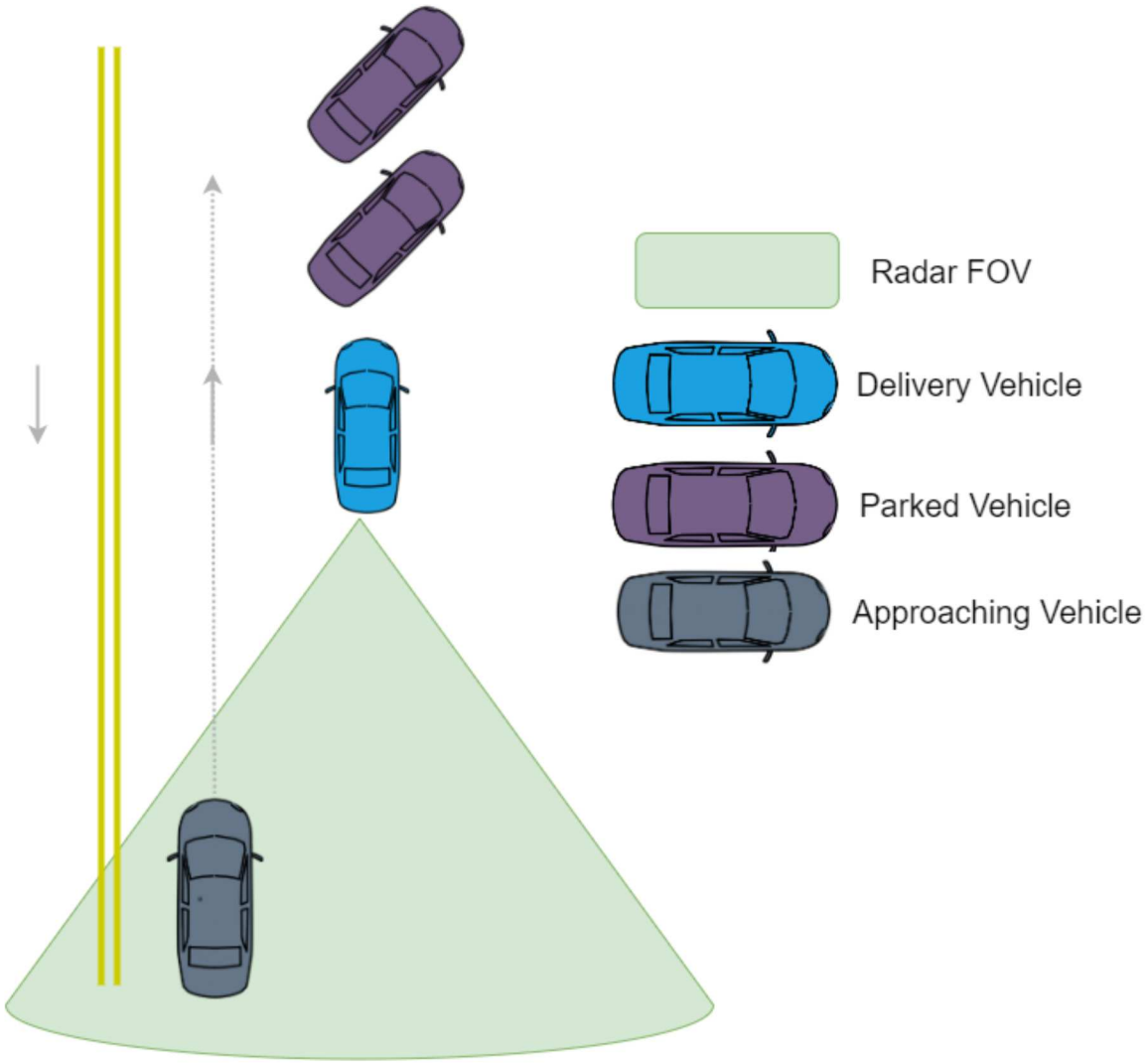

Figure 9.2: How data collection for non-blocking scenario is currently set up.

CHAPTER 10: FUTURE SYSTEM APPLICATION

With the successful validation of the tracking array, CSU Autonomous Vehicle Research Lab now possess the capability to capture behavior of vehicle surrounding the ego. As such, several proposal were discussed involving utilization of the tracking array developed by this project.

10.1 Autonomous FE vs Manual FE

Recalling the original goal of the project, the tracking array was meant to be mounted to the CDOT Autonomous Truck to determine how drivers would react upon spotting such vehicle on the road and have they would perform a left lane cut. After a discussion with other members of the project, it was found that such experiment can be performed without the need of the CDOT Autonomous Truck.

Using the same set up for the re-purposed project and adding a notice board/sticker to the rear of the delivery vehicle, a set up that notifies approaching drivers of the delivery vehicle's status as an autonomous vehicle can be accomplished. With this new setup, experiments can be conducted to capture the behavior of approaching vehicles attempting to perform a left lane cut with the following outline:

1. Starting from the CSU Powerhouse, the delivery vehicle is to drive down North College Avenue to East Harmony Street before returning to the CSU Powerhouse.
2. There are two acceleration behaviors the delivery vehicle should perform.
   (a) Manual vehicle acceleration: Vehicle accelerates to speed limit (when allowed) as quickly as possible at every start stop scenario.
   (b) Autonomous vehicle acceleration: Vehicle slowly accelerates to speed limit at every start stop scenario.
3. The delivery vehicle is to be mounted with an sign/sticker that would indicate its status as an autonomous vehicle.

Table 10.1: Table for Experiment Combinations

Test Scenario	Autonomous Sign Attached	Autonomous Sign Not Attached
Manual Acceleration		
Autonomous Acceleration		

Using the experiment combinations in Table 10.1, 4 experiments scenarios are to be conducted with the drive traces of all vehicles cutting across the ego collected and compiled into 4 separate cumulative heat map to determine the average cutting pattern for each scenario. Additionally, the FE distribution for each experiment are to be compiled to determine the difference in FE for every scenario.

This experiment would not only allow the data collection be collected as early as April, but it also negates the need to travel down to mount and unmounted the tracking array on the CDOT Truck as well as immediately addressing any unnoticed failures in the system.

CHAPTER 11: AUTONOMOUS DRIVE TRACE

In addition to the tracking array, another feature planned for implemented into the TVP is the ability to self-drive. The implementation of such feature will be done in the interest of testing autonomous vehicle algorithm in real-time and to determine discrepancies in fuel economy improvement between real-life and simulation. Although scheduled for implementation in Spring 2022, various factors had prevented construction/set-up of a physical infrastructure for autonomous driving.

Despite the lack of physical infrastructure, work has been done in the coding portion of this feature and simulations indicate the use of autonomous driving algorithm results in fuel economy improvements. This section shall dive into detail on the work done to simulate eco-driving using various autonomous driving algorithm.

Working in unison with Western Michigan University (WMU), CSU utilized a real-time traffic data to evaluate the effectiveness of each optimal drive trace generation method. The following section will highlight the procedures taken to convert the real-time traffic data into usable data along with generation of traffic constraints. Said constraints are then used to generated optimal drive traces through various methods. Each methods are then compared in terms of computation time and function call to whether such methods are implementable in real-time.

11.1 Code Conversion, Parsing and Constraint Generation

Due to the large size of the dataset collected by CSU, the dataset needed to be parse into a usable form. In order to determine which data is usable, the turn direction for each traffic light was identified beforehand.

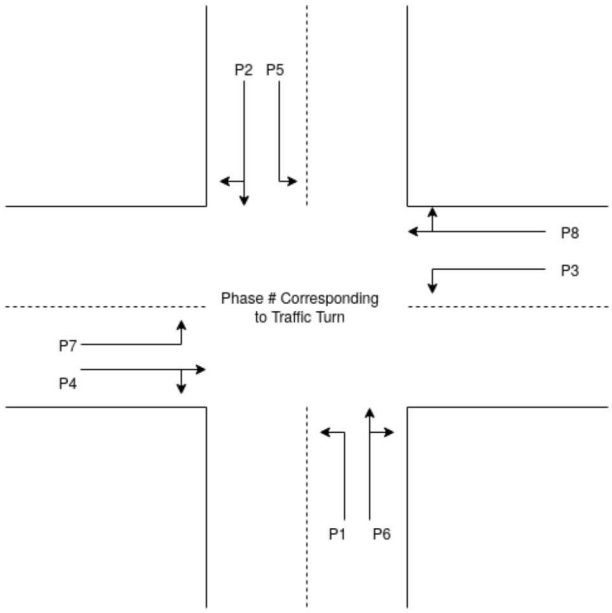

Figure 11.1: Turn direction represented as different numbers

Figure 11.1 outlines the unique numbers assigned by CDOT to each turn direction. These numbers help determine which data should be extracted for each traffic light.

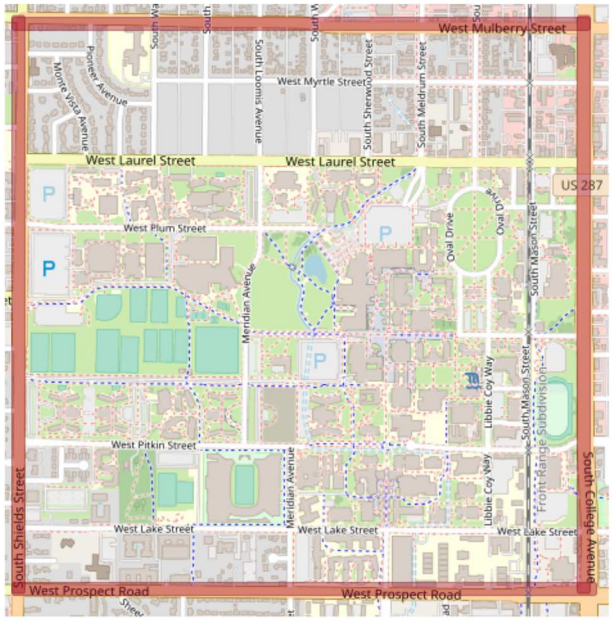

Figure 11.2: Travel route of ego vehicle in simulation

Figure 11.2 outlines the travel route data collected by CSU. The travel path starts from College and

Prospect and circle back through South College Avenue, West Laurel Street, South Shield Avenue and West Prospect Road. The entire travel path consists of 19 traffic lights. The table below specifies which turn direction data was extracted from each traffic light.

Table 11.1: Traffic Lights and their corresponding turn direction

Traffic Light Name	Turn Direction	Traffic Light Name	Turn Direction
010 College Prospect	P7	044 Shields Laurel	P2
011 College Pitkin	P6	045 Shields Plums	P2
012 College Elizabeth	P6	046 Shields Elizabeth	P2
013 College Laurel	P6	486 Shields Springfield	P2
014 College Mulberry	P1	116 Shields Lake	P2
029 Mason Mulberry	P8	047 Shields Prospect	P5
023 Howes Mulberry	P8	083 Whitcomb Prospect	P4
131 Meldrum Mulberry	P8	084 Centre Prospect	P4
078 Loomis Mulberry	P8	184 MAX Guideway Prospect	P4
043 Shields Mulberry	P3	-	-

Using Python, the dataset of interest are parsed and converted red light traffic signal. (See Figure 11.3)

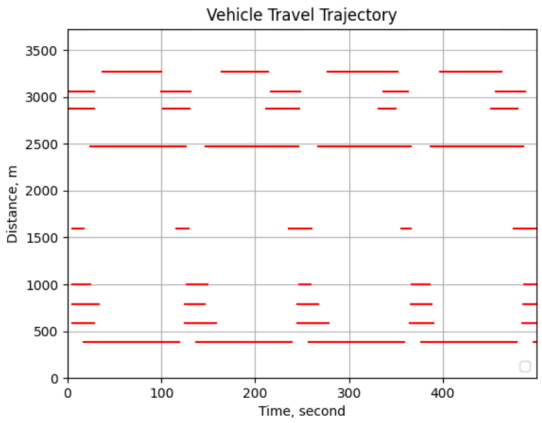

Figure 11.3: A segment of the red light duration for individual traffic lights using College Prospect as the starting point.

To generate the upper and lower bound of the data, an algorithm was written to represents the constraints as a step-series. Representing the constraints in this manner, provides a larger solution space for potential optimal drive traces. (See Figure 11.4)

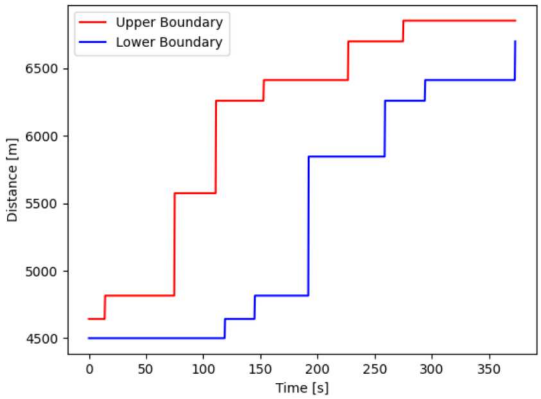

Figure 11.4: Upper and Lower Bound generated. Representing Upper and Lower Bound as step series provides more possible states for optimal solutions.

To converse the time needed to run simulations with the upper and lower bound constraints, the upper and lower bound data are stored in the form of an excel file and called using a simple Python function. This negates the need to regenerate the upper and lower bound at every run and minimizes the computation time for individual optimization simulations.

11.2 Summary of Optimization Algorithm Used

All optimization simulations used in this section was done with one objective: To minimize the amount of energy consumed to get from point A to point B within the allowed time without violating upper and lower bounds. This section will divide into how each optimization algorithm works and what modifications were made for potential real-time implementation.

11.2.1 Intelligent Driver Model (IDM)

Developed by Treiber, Hennecke and Helbing, Intelligent Driver Model (IDM) was created to emulate the behavior of human driven vehicle matching the velocity of the vehicle traveling in front of it with a safe following distance [54]. The IDM has seen success in real-time adaptive cruise control and as of 2022, vehicles with autonomous driver features have IDM or modified variant implemented. Due to its proven applicability in real-life, autonomous drive traces generated by IDM is used as a baseline solution for comparison with other eco-driving algorithms. IDM is represented by the equations below:

$$\dot{x}_i = \frac{dx_i}{dt} = v_i \tag{11.1}$$

$$\dot{v}_i = \frac{dv_i}{dt} = a\left(1 - \left(\frac{v_i}{v_0}\right)^{\delta} - \left(\frac{s^*(v_i, v_{i-1})}{s_i}\right)^2\right) \tag{11.2}$$

$$s_i = x_{i-1} - x_i - l_{i-1} \tag{11.3}$$

$$s^*(v_i, v_{i-1}) = s_0 + v_i T + \frac{v_i(v_{i-1} - v_i)}{2\sqrt{ab}} \tag{11.4}$$

Table 11.2 describes the parameters for IDM.

Table 11.2: Variables and Parameters for IDM

Parameter	Description	Representative Value
i	Ego vehicle (lead vehicle is vehicle i-1)	N/A
x	Distance	N/A
v	Velocity	N/A
s	Distance headway (space between lead and follow vehicle)	N/A
s*	Desired distance headway	N/A
s_0	Minimum distance headway	2.5 m
T	Desired time headway	2 s
δ	Velocity exponent	4
l	Vehicle length	2 m
a	Maximum forward acceleration	.5 m/s^2
b	Maximum deceleration	.5 m/s^2

11.2.2 Particle Swarm Optimization (PSO)

Developed by Russell C. Eberhart and James Kennedy to study how individual agents of a group can benefit from the experience of all other members [17], Particle Swarm Optimization (PSO) has found success as means of generating optimal drive trace in the field of autonomous driving[33][57][26]. PSO is generally represented as the following equations:

$$X_i(t+1) = X_i(t) + V_i(t) \tag{11.5}$$

$$V_i(t+1) = w * V_i(t) + c_1 * r_1 * (pbest_i - X_i(t)) + c_2 * r_2 * (gbest - X_i(t)) \tag{11.6}$$

The parameters for the PSO are described in Table 11.3.

Table 11.3: Variables and Parameters for PSO

Parameter	Description	Representative Value
i	Particle Designation	Up to number of swarm particles, n
X_i	Particle Position in State Space	N/A
V_i	Particle Velocity	N/A
w	Velocity update weight	0.8
c_1	Local position update weight	0.1
c_2	Global position update weight	0.1
r_1	Local random number	0 to 1
r_2	Global random number	0 to 1
pbest	Best solution found by individual particle	N/A
gbest	Best solution found by swarm	N/A

PSO is a heuristic search method that utilized a search and explore concept to find an optimal solution. By populating a solution space with a set number of particle, said particle will explore the allotted solution space until an optimal solution is found. A point of note is that PSO does not guarantee a globally optimal solution and the particles may converge into their respective locally optimal solution as a result of their starting points. To address this issue, [26] suggested adding a mutation step to PSO. Such step provides two benefits, namely:

1. Allows a subset of particles to explore other sections of the solution space that they otherwise wouldn't have, preventing particles from being stuck at locally optimal solutions.

2. Enable faster solution convergence as particles spread out to explore the solution space.

11.2.3 Dynamic Programming (DP)

Dynamic Programming (DP) is defined as a recursive method that solves a control problem until an optimal condition is satisfied. Developed by Richard Bellman [15], DP is generally represented as the following equation:

$$J_{opt} = \min_{u} b(x(t), u(t), t) + \int_{t_n}^{t_0} f(x(t), u(t), t)dt \tag{11.7}$$

The parameters for the DP are described in Table 11.4.

Table 11.4: Variables and Parameters for DP

Parameter	Description
x	States of Interest
u	Control Actions
t	Time
J	Cost Function
f	State Transition Function
b	Penalty function

The equation above is explained as finding a control action, u that results in the lowest possible cost, J_{opt}. The state transition function, $f(x(t), u(t), t)$ is primarily represented as a vehicle's position, velocity and acceleration model. The penalty function, $b(x(t), u(t), t)$ represents the combination of dynamic boundary constraint violation and end penalty violation.

11.2.4 Spline Non Linear Programming

Spline-Non-Linear Programming (NLP) is an optimization done through Direct Transcription (DT). Direct Transcription is a DTTO based optimization method that discretizes a set of control actions to compute an optimal solution. For eco-driving, the control action is set as acceleration and a set of acceleration from t_i to t_N with a dt time step.

1. Dynamics of DT:

$$V_k = V_{k-1} + U_k * dt \tag{11.8}$$

$$X_k = X_{k-1} + V_k * dt \tag{11.9}$$

where U, acceleration, is the control action of the DT method.

2. DT Optimal Problem:

$$\min_u : \sum_{k=1}^{N} COST(V_k, U_k) + ConstraintCOST(V_k) + BoundCOST(X_k) + FSCOST(X_N, V_N) \tag{11.10}$$

where the cost function is a combination of travel trace cost, COST(V_k, U_k), constraint violation cost, ConstraintCOST(V_k), boundary violation cost, BoundCOST(X_k) and final state cost, FSCOST(X_N, V_N).

3. Constraint Cost:

$$ConstraintCOST(V_k) = \begin{cases} V_k^2 * \alpha & V_k > V_{max} \text{ or } V_k < V_{min} \\ 0 & V_{min} \leq V_k \leq V_{max} \end{cases} \tag{11.11}$$

where α is a penalty constant for Constraint Cost.

4. Boundary Cost:

$$BoundCOST(V_k) = \begin{cases} \beta * (X_k - UB_k)^2 & X_k > UB_k \\ 0 & LB_k \leq X_k \leq UB_k \\ \beta * (X_k - LB_k)^2 & X_k < LB_k \end{cases} \tag{11.12}$$

where UB_k and LB_k are the dynamic upper and lower bound on vehicle position respectively with β functioning as a penalty constant.

5. Final State Cost:

$$FSCOST(V_k) = \gamma * (X_N - X_{target})^2 + (V_N - V_{target})^2)^{1/2} \tag{11.13}$$

where γ is a penalty constant.

11.2.5 Genetic Algorithm (GA)

Proposed by Prof John H. Holland, Genetic Algorithm (GA) was developed using the concept of 'survival of the fittest' to breed an ideal population[24]. Starting from an initial population set, GA utilizes crossover (exchange of genes between two subjects), mutation (editing of genes using probabilities) and ranking of chromosome (deciding which chromosome is 'ideal' in each generation) to breed consecutive generations with more ideal traits. Much research has been done using GA to generate optimal drive traces for eco-driving with promising results [47][19][53]. The order of procedures for GA are:

1. Populate a sample space with a set of viable solutions chromosomes.
2. Evaluate each solution and determine which solution is the best.
3. Ranking: Rank the solution in accordance with their performance.
4. Crossover: Splice and combine chromosomes from different solutions in order of rank. For example, best solution being spliced with second best, 3rd best solution is spliced with fourth best and so on.

5. Mutation: Randomly edit an element in solution chromosome. This step enables a wider exploration of the solution space and therefore convergence.

6. Repeat step 2 to 5 until solution convergences.

11.3 Findings

Table 11.5: Optimization Solution Feasibility and Run Time

Optimization Algorithm	Feasible Solution Percent	Computation Time (s)
Spline-GA	0.9	0.08s
Spline-NLP	0.99	0.73s
Spline-PSO	0.87	0.55s
DP	-	95s

Table 11.5 outlines the solution viability and computation time per iteration of each optimization algorithm. Due to Dynamic Programming's long run, no further attempts were made to analyze its feasibility as real-time optimization algorithm. DP's long run time was a result of the algorithm needing to consider a large number of states for optimizing both velocity and position simultaneously. The fastest optimization algorithm is Spline-GA with a run of time of 0.08s. Spline-GA's short run time is a result of the algorithm's ability to converge to a feasible optimal solution within a small number of generations (about 10). Spline-NLP has the highest percentage of feasible solutions and the longest run time per iteration. The results indicate that a Genetic Algorithm type autonomous eco-driving algorithm should be implemented for real-time autonomous driving.

References

[1] Tanya M. Anandan. Ros-industrial for real-world solutions. 2019. [https://www.automate.org/industry-insights/ros-industrial-for-real-world-solutions; Online; accessed 07-September-2021].

[2] Hmrishav Bandyopadhyay. Yolo: Real-time object detection explained. 2022. https://www.v7labs.com/blog/yolo-object-detection; Online; accessed 28-March-2022.

[3] Alex Becker. Introduction to kalman filter. 2021. doi:https://www.kalmanfilter.net/alphabeta.html.

[4] Sebastian Blanco. Sae updates, refines official names for 'autonomous driving' levels. 2021. https://www.caranddriver.com/news/a36364986/sae-updates-refines-autonomous-driving-levels-chart/; Online; accessed 15-June-2021.

[5] Gaudenz Boesch. Object detection in 2022: The definitive guide. 2021. https://viso.ai/deep-learning/object-detection/; Online; accessed 4-December-2021.

[6] Gael Varoquaux Brian M Clapper. scipy.optimize.linear sum assignment. 2008. https://github.com/scipy/scipy/blob/v1.7.1/scipy/optimize/_lsap.py#L16-L100; Online; accessed 4-January-2022.

[7] Jason Brownlee. A gentle introduction to object recognition with deep learning. 2021. https://machinelearningmastery.com/object-recognition-with-deep-learning/; Online; accessed 4-December-2021.

[8] Sean Campbell, Niall O'Mahony, Lenka Krpalcova, Daniel Riordan, Joseph Walsh, Aidan Murphy, and Conor Ryan. Sensor technology in autonomous vehicles : A review. In 2018 29th Irish Signals and Systems Conference (ISSC), pages 1–4, 2018. doi:10.1109/ISSC.2018.8585340.

[9] Sergen Cansiz. Mahalanobis distance and multivariate outlier detection in r. 2020. https://towardsdatascience.com/mahalonobis-distance-and-outlier-detection-in-r-cb9c37576d7d; Online; accessed 10-December-2021.

[10] CDOT. Connected and autonomous technology program. 2020. doi:https://www.codot.gov/programs/innovativemobility/mobility-technology/cav-technology-program.

[11] Haipeng Chen, Zhentao He, Bowen Shi, and Tie Zhong. Research on recognition method of electrical components based on yolo v3. IEEE Access, 7:157818–157829, 2019. doi:10.1109/ACCESS.2019.2950053.

[12] Urrea Claudio and Agramonte Rayko. Kalman filter: Historical overview and review of its use in robotics 60 years after its creation. Journal of Sensors, 2021, 2021. doi:https://doi.org/10.1155/2021/9674015.

[13] David F. Crouse. On implementing 2d rectangular assignment algorithms. IEEE Transactions on Aerospace and Electronic Systems, 52(4):1679–1696, 2016. doi:10.1109/TAES.2016.140952.

[14] Gabriel DiDomenico, Jamison Bair, Vipin Kumar Kukkala, Jordan Tunnell, Marco Peyfuss, Michael Kraus, Joshua Ax, Jeremy Lazarri, Matthew Munin, Corey Cooke, Eric Christensen, Logan Peltz, Nathan Peterson, Logan Wolfe, Zach Vinski, Daniel Norris, Corrie Kaiser, Jacob Collier, Nick Schott, and Thomas Bradley. Colorado state university ecocar 3 final technical report. 04 2019. doi:10.4271/2019-01-0360.

[15] S.E. Dreyfus. Richard bellman on the birth of dynamic programming. Operations Research, 50:48–51, 02 2002. doi:10.1287/opre.50.1.48.17791.

[16] Bharanidhar Duraisamy, Tilo Schwarz, and Christian Wöhler. On track-to-track data association for automotive sensor fusion. In 2015 18th International Conference on Information Fusion (Fusion), pages 1213–1222, 2015.

[17] R. Eberhart and J. Kennedy. A new optimizer using particle swarm theory. In MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science, pages 39–43, 1995. doi:10.1109/MHS.1995.494215.

[18] Philip Evans. Rotations and rotation matrices. Acta crystallographica. Section D, Biological crystallography, 57:1355–9, 11 2001. doi:10.1107/S0907444901012410.

[19] Mukesh Gautam, Narayan Bhusal, Mohammed Benidris, and Poria Fajri. A ga-based approach to eco-driving of electric vehicles considering regenerative braking, 2020. URL: https://arxiv.org/abs/2012.15195, doi:10.48550/ARXIV.2012.15195.

[20] Ross B. Girshick. Fast R-CNN. CoRR, abs/1504.08083, 2015. URL: http://arxiv.org/abs/1504.08083, arXiv:1504.08083.

[21] Stephanie Glen. Independent samples t test: Definition, excel and spss steps. https://www.statisticshowto.com/probability-and-statistics/t-distribution/independent-samples-t-test/; Online; accessed 20-March-2022.

[22] Stephanie Glen. Mann whitney u test: Definition, how to run in spss. https://www.statisticshowto.com/mann-whitney-u-test/; Online; accessed 20-March-2022.

[23] Stephanie Glen. Mahalanobis distance: Simple definition, examples. 2017. https://www.statisticshowto.com/mahalanobis-distance/; Online; accessed 4-January-2022.

[24] John H. Holland. Genetic algorithms. Scientific American, 267(1):66–73, 1992. URL: http://www.jstor.org/stable/24939139.

[25] Glenn Jocher. Yolov5, 2021.

[26] Sofiene Kachroudi, Mathieu Grossard, and Neil Abroug. Predictive driving guidance of full electric vehicles using particle swarm optimization. IEEE Transactions on Vehicular Technology, 61(9):3909–3919, 2012. doi:10.1109/TVT.2012.2212735.

[27] Kambria. The history and evolution of self-driving cars. 2019. URL: https://kambria.io/blog/the-history-and-evolution-of-self-driving-cars/.

[28] Grace Karimi. Introduction to yolo algorithm for object detection. 2021. https://www.section.io/engineering-education/introduction-to-yolo-algorithm-for-object-detection/; Online; accessed 07-September-2021.

[29] H. W. Kuhn. The hungarian method for the assignment problem. Naval Research Logistics Quarterly, 2:83–97, 03 195. doi:https://doi.org/10.1002/nav.3800020109.

[30] Vipin Kumar Kukkala, Jordan Tunnell, Sudeep Pasricha, and Thomas Bradley. Advanced driver-assistance systems: A path toward autonomous vehicles. IEEE Consumer Electronics Magazine, 7(5):18–25, 2018. doi:10.1109/MCE.2018.2828440.

[31] Adrienne LaFrance. Your grandmother's driverless car. 2016.

[32] Wayne W LaMorte. Mann whitney u test (wilcoxon rank sum test). 2017. https://sphweb.bumc.bu.edu/otlt/mph-modules/bs/bs704_nonparametric/bs704_nonparametric4.html; Online; accessed 20-March-2022.

[33] Ming Li, Xinkai Wu, Xiaozheng He, Guizhen Yu, and Yunpeng Wang. An eco-driving system for electric vehicles with signal control under v2x environment. Transportation Research Part C: Emerging Technologies, 93:335–350, 2018. doi:https://doi.org/10.1016/j.trc.2018.06.002.

[34] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: single shot multibox detector. CoRR, abs/1512.02325, 2015. URL: http://arxiv.org/abs/1512.02325, arXiv:1512.02325.

[35] Pablo Marín, Ahmed Hussein, David Martín Gómez, and Arturo de la Escalera. icab use case for ros-based architecture. Robotics and Autonomous Systems, 118, 05 2019. doi:10.1016/j.robot.2019.04.008.

[36] G. Mclachlan. Mahalanobis distance. Resonance, 4:20–26, 06 1999. doi:10.1007/BF02834632.

[37] Corey Montella. The kalman filter and related algorithms: A literature review. 05 2011. URL: file:///tmp/mozilla_thb0/TheKalmanFilterandRelatedAlgorithms.pdf.

[38] General Motors. 2018 self-driving safety report. General Motor Online, 12 2018. doi:https://www.gm.com/content/dam/company/docs/us/en/gmcom/gmsafetyreport.pdf.

[39] Cristian Neipp, A Hernández, Jose Rodes-Roca, Andrés Márquez, T Beléndez, and Augusto Beléndez. An analysis of the classical doppler effect. European Journal of Physics, 24:497, 07 2003. doi:10.1088/0143-0807/24/5/306.

[40] Nvidia. Jetson benchmarks. 2020. https://developer.nvidia.com/embedded/jetson-benchmarks; Online; accessed 09-September-2021.

[41] Sieun Park. A guide to two-stage object detection: R-cnn, fpn, mask r-cnn. 2021. https://medium.com/codex/a-guide-to-two-stage-object-detection-r-cnn-fpn-mask-r-cnn-and-more-54c2e168438c; Online; accessed 02-December-2021.

[42] Selva Prabhakaran. Mahalanobis distance – understanding the math with examples (python). 2019. https://www.machinelearningplus.com/statistics/mahalanobis-distance/; Online; accessed 10-December-2021.

[43] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2016.

[44] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. CoRR, abs/1804.02767, 2018. URL: http://arxiv.org/abs/1804.02767, arXiv:1804.02767.

[45] ROS. Ros: Why ros? 2021. https://www.ros.org/blog/why-ros/; Online; accessed 10-December-2021.

[46] Bill Moebs Samuel J. Ling, Jeff Sanny. University Physics Volume 1. OpenStax, 2016.

[47] Subramaniam Saravana Sankar, Yiqun Xia, Julaluk Carmai, and Saiprasit Koetniyom. Optimal eco-driving cycles for conventional vehicles using a genetic algorithm. Energies, 13(17), 2020. URL: https://www.mdpi.com/1996-1073/13/17/4362, doi:10.3390/en13174362.

[48] Liyun Li Shaoshan Liu and et al. Creating Autonomous Vehicle Systems. Synthesis Lectures on Computer Science. Morgan and Claypool, 2017.

[49] Minchul Shin and Dongsoo Kwon. Implementation of extended kalman filter with pi control and modeling effect reduction for precise motor speed estimation in disturbance. pages 72–76, 10 2015. doi:10.1109/URAI.2015.7358931.

[50] Jacob Solawetz. How to train a custom object detection model with yolo v5. 2020. https://towardsdatascience.com/how-to-train-a-custom-object-detection-model-with-yolo-v5; Online; accessed 4-December-2021.

[51] S Srivastava, AV Divekar, and C Anilkumar. Comparative analysis of deep learning image detection algorithms. Journal of Big Data, 8:–, 02 2021. doi:10.1186/s40537-021-00434-w.

[52] L Tan, T Huangfu, and L Wu. Comparison of retinanet, ssd, and yolo v3 for real-time pill identification. BMC Medical Informatics and Decision Making, 21:–, 06 2021. doi:10.1186/s12911-021-01691-8.

[53] Sina Torabi, Mauro Bellone, and Mattias Wahde. Energy minimization for an electric bus using a genetic algorithm. European Transport Research Review, 12:1–8, 2020.

[54] Martin Treiber, Ansgar Hennecke, and Dirk Helbing. Congested traffic states in empirical observations and microscopic simulations. Physical Review E, 62(2):1805–1824, Aug 2000. doi:10.1016/j.trc.2007.12.004.

[55] Yash Varyani. Hungarian algorithm for assignment problem. 2020. https://www.geeksforgeeks.org/hungarian-algorithm-assignment-problem-set-1-introduction/; Online; accessed 07-December-2021.

[56] Shenghui Wang, Leilei Niu, and Nan Li. Research on image recognition of insulators based on yolo algorithm. In 2018 International Conference on Power System Technology (POWERCON), pages 3871–3874, 2018. doi:10.1109/POWERCON.2018.8602149.

[57] Yan Wang, Rong Su, Wei Wang, and Bohui Wang. Distributed eco-driving algorithm of vehicle platoon using traffic light and road slope information, 2021. arXiv:2104.12499.

[58] Greg Welch and Gary Bishop. An introduction to the kalman filter. 1997. https://perso.crans.org/club-krobot/doc/kalman.pdf; Online; accessed 07-December-2021.

[59] Kim Youngjoo and Bang Hyochoong. Introduction to kalman filter and its applications. 2018. doi:10.5772/intechopen.80600.

[60] Chen Yukun, Si Xicai, and Li Zhigang. Research on kalman-filter based multisensor data fusion. Journal of Systems Engineering and Electronics, 18(3):497–502, 2007. doi:10.1016/S1004-4132(07)60119-4.

LIST OF ABBREVIATIONS

AV Autonomous Vehicle. 11

CDOT Colorado Department of Transportation. 1, 2, 4, 14, 44, 66

CNN Convoluted Neural Network. 21

CSU Colorado State University. 1, 2, 4, 70, 72

CSV Comma-Seperated Values. 47

DAQ Data Acquisition. 2, 5, 7, 31, 32, 36, 37

DP Dynamic Programming. 79

DT Direct Transcription. 80

EEAV Energy Efficient and Autonomous Vehicles. iii

ESR Electronically Scanned Radar. 11–15, 66

FE Fuel Economy. 45, 46, 71

FOV Field of View. 12, 36, 37

GA Genetic Algorithm. 81

GPU Graphical Processing Unit. 26, 66

HA Hungarian Algorithm. 39

HDV Human Driven Vehicle. 4, 45, 64

IDM Intelligent Driver Model. 76

IOU Intersection Over Union. 23

L3AV Level 3 Autonomous Vehicle. 1–5, 22, 44

MD Mahalanobis Distance. 37, 38

MRR Mid Range Radar. 11, 13–15, 42, 65, 66

NLP Non-Linear Programming. 80

PSO Particle Swarm Optimization. 77, 78

R-CNN Reinforced Convoluted Neural Network. 21

ROS Robot Operating System. 5–7

SAE Society of Autonomous Engineers. 1

TVP Testing Vehicle Platform. 14, 16, 17, 20, 42, 43, 65, 72

YOLO You Only Look Once. 22, 23