Automotive Collision Warning System Retrofit

Ethan Clark Najmy

Senior Project | Electrical Engineering Department | 2022-2023

California Polytechnic State University, San Luis Obispo

Advised by Dr. Payam Nayeri

Table of Contents

List of Figures

List of Tables

1 Abstract

In the early 2000s, few automakers began implementing forward collision warning systems in their cars. As technology advanced this system became available on more and more luxury cars. In recent years, this technology has spread to more affordable vehicles driven every day. However, as this technology has only recently advanced to less expensive, more economical cars, older vehicles of the same model may not have this advanced and important safety feature.

This project investigates and creates a preliminary design for an affordable, easy-to-install, forward collision warning system that can be retrofitted to vehicles without the system currently installed. Using a density-based spatial clustering of applications with noise (DBSCAN) clustering algorithm, an extended Kalman filter, and a time-to-collision algorithm, a forward collision warning system was developed and simulated using the Insurance Institute of Highway Safety (IIHS) test scenarios. Software testing and implementation was done in MATLAB and has provided a foundation for future hardware implementation using Texas Instruments mmWave automotive radar (AWR1843BOOST) .

2 Introduction

The primary objective of this project was to research, design, and develop an affordable Forward Collision Warning (FCW) system that could be retrofitted to existing vehicles. Recognizing that many vehicle owners lack a FCW system, this project aimed to create a solution that enhanced the safety of their driving experience. The proposed system was designed to issue acoustic and visual warnings when the vehicle's following or closing distance was dangerously short.

When a rapid closure between the main vehicle and another was detected, the system was designed to issue both an acoustic and visual warning to the driver. Using a recommended following distance and reaction time of 3 seconds as a warning threshold, a warning would be issued through a speaker and LED lights installed inside the vehicle.

Compared to existing camera-based FCW retrofit systems on the market, this project aimed to offer a more reliable and affordable alternative. Existing camera-based systems can be expensive and may not always provide accurate information, therefore development was made towards using radar – specifically Texas Instrument's mmWave Automotive radar (AWR1843BOOST).

However, due to shipping issues and time constraints, the development of a final working prototype was not realized. Nevertheless, significant progress was made in terms of research, simulation, and the development of algorithms for a working forward collision warning system. These developments included a time-to-collision distance algorithm, DBSCAN clustering methodology, and an Extended Kalman Filtering algorithm for object detection. This groundwork creates a promising foundation for the future realization of this ambitious project.

3 Background

3.1 ADAS Overview

Advanced driver assistance systems (ADAS) in passenger vehicles have developed greatly over the past few decades. ADAS were born into the automotive market with rudimentary systems, compared to the modern systems we have today, such as the Antilock Braking System (ABS), cruise control systems, or Electronic Stability Programs (ESP). These ADAS have focused on providing drivers with a better driving experience, reducing driver strain, and providing a safer driving experience. In recent years, sensor-based and software-driven ADAS have emerged in the automotive market – with radar adaptive cruise control, autonomous emergency braking, lane departure warning, and now to autonomous hands-free driving. These active, more sophisticated ADAS are becoming more mainstream in today's automotive market and have been proven to reduce collisions due to driver error, while also providing a safe, less stressful driving experience [1]. Shown below, these systems can use a variety of sensors to implement multiple systems including cameras for lane keep assist and pedestrian detection, radar for forward collision warning and adaptive cruise control, LiDAR for emergency braking, and ultrasonic sensors for parking distance warnings.

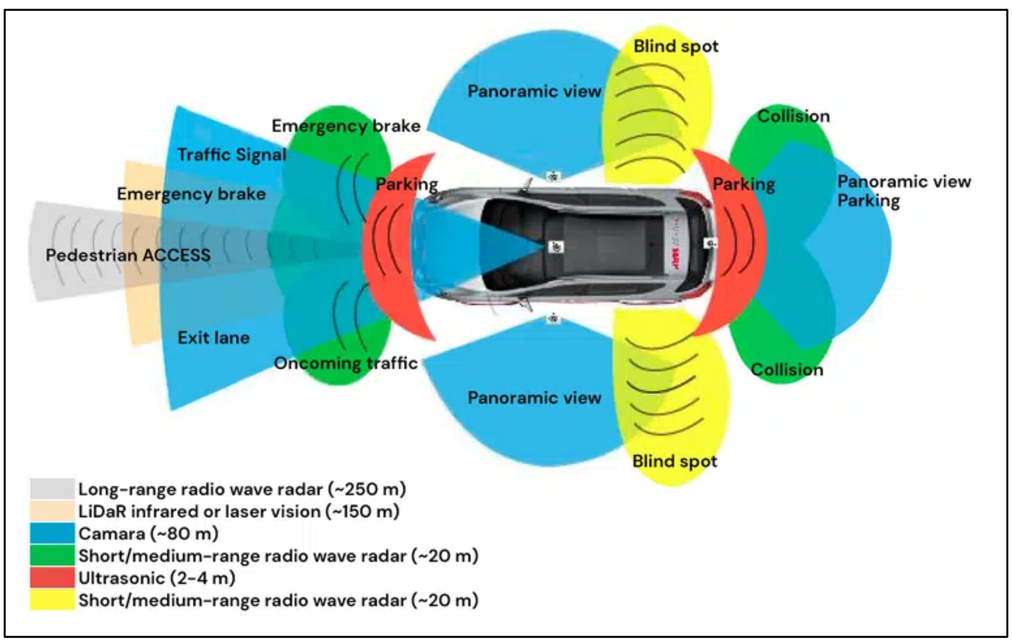

Figure 3.1 – ADAS Sensors and Usage [3]

Modern ADAS primarily consists of sensor arrays, control units, actuators, and display or feedback systems. The sensors act as the ‘eyes’ of the system, capable of recognizing, interpreting, and understanding the vehicle’s surroundings in multiple directions and with multiple measurements (distance, velocity, angle, etc.). These sensors generate a constant stream of data which is then processed by a control unit using various algorithms and data processing methods to understand the current driving situation and conditions, detecting potential threads and or, automating certain tasks.

Overall, the focus and purpose of ADAS are to enhance vehicle and road safety and improve driver comfort. By using modern sensor technology, ADAS serve as an extra pair of eyes on the road, alerting drivers of potential hazards and alerting the drivers to these hazards to help prevent or even autonomously avoid accidents. With many vehicles on the road today that were built before the introduction of these ADAS, it’s important to be mindful of this technology gap and find any solutions to fix this gap.

3.2 Forward Collision Warning

A very important and basic feature that has been proven to reduce rear-end collisions is forward collision warning (FCW). FCW systems have become one of the key components of ADAS, aiming to mitigate or prevent front-end collisions.

Figure 3.2 - Mercedes-Benz Distronic Plus [13]

This technology emerged in the early 2000s with Mercedes-Benz and Toyota. Mercedes-Benz is often credited as one of the pioneers of the FCW system, having introduced its Distronic Plus FCW System in the S-Class model around 2005. Other automakers followed suit, and by the 2010s, FCW systems had become increasingly common, particularly in high-end vehicles, however, it is becoming more and more mainstream in new production vehicles [2].

FCW systems typically work by utilizing one or more types of sensors – radar, LiDAR, or cameras – to detect objects in the path of the vehicle. These sensors constantly scan the road ahead and send this data to a central control unit. This unit processes the data with advanced algorithms to identify potential collision threats based on factors such as relative speed, distance, and the object's trajectory. The concept of FCW systems remains constant among automakers, and some have even combined these systems with Automatic Emergency Braking (AEB) systems to apply the brakes if the driver fails to react to the warnings in time.

Figure 3.3 - FCW Alert [14]

According to the Insurance Institute for Highway Safety, in a 2017-2019 study, forward collision warning reduced police-reportable accidents by 22% per vehicle miles traveled [1]. Along with this, recent studies by the Highway Loss Data Institute (HLDI) have shown that crash avoidance features in vehicles appear to provide more significant benefits for younger drivers (under 25). The studies observed a larger decrease in collision and property damage claims for drivers below 25 years old when using these systems [14].

FCW systems have demonstrated significant potential in reducing collisions and enhancing vehicular safety by reducing the likelihood of collisions. Despite this, its implementation into vehicles has been rather slow. Only 30% of vehicles on the market were equipped with the technology in 2018 [2]. This relatively slow adoption rate can be attributed to many factors, including cost, technological limitations, and lack of regulatory mandates. This delay in widespread FCW implementation means a substantial portion of the automotive population, particularly older model vehicles, may not be equipped with this important technology. This project aims to investigate and provide solutions to this technological gap.

4 Requirements & Specifications

The primary purpose of this project was to conduct research, create a design, and develop an affordable, safe, and retrofittable forward collision warning system for automotive vehicles. The feature requirements and corresponding engineering requirements are described below in *Table I* and are based on typical FCW systems on the market today. The ideal and originally proposed project specifications are outlined in *Table II* below.

Overall, the originally proposed design concept for the FCW system was centered around improving safety in vehicles lacking this feature. The system's key components include a radar sensor, a microcontroller for data processing, a speaker for acoustic warnings, and LEDs for visual warnings. The system should provide a warning if a collision will occur or if the distance between the vehicles contributes to an unsafe reaction time.

These requirements and specifications reflect the desired and final outcome for a fully realized, functional retrofit FCW system project.

TABLE I - SYSTEM SPECIFICATIONS

Spec. #	**Parameter**	**Target (units)**	**Tolerance**	**Risk** (H, M, L)	**Compliance** (A, T, S, I)
1	Power – DC Voltage	5V	Exact	H	T
2	Reaction Time (Warning Distance)	3 seconds	Min	H	T
3	Cost	$200	Max	M	A
4	Installation Time	3 hours	± 1 hour	L	T
5	Sensing Distance	40m, 131ft.	Max	H	T
6	Field of View	90˚	Max	M	T

TABLE II - CUSTOMER & SYSTEM REQUIREMENTS

Customer Requirements	Engineering Specifications	Justification
1-3	1. Should use a speaker and LED lights to provide a warning to the driver.	Acoustic and visual forms of FCW are the most popular and widely used forms of warning. They have been proven to provide adequate warning to the vehicle operator. [1, 5]
2	2. Will provide a warning if the test vehicle is rapidly approaching another vehicle with less than 3 seconds of reaction time.	A 3 second or greater reaction time is recommended for following distance between vehicles. [4]
3	3. Will provide a warning if the cruising distance between the vehicles is less than 3 seconds of reaction time.	A 3 second or greater reaction time is recommended for following distance between vehicles. [4]
4, 5	4. System turns on and off with the car's ignition and powers on and off appropriately.	Turning on and off with the ignition will prevent any battery consumption when the vehicle is off.
5, 6	5. A button will be installed to disable or enable system functionality.	Allowing for the user to disable the system if unwanted will provide for system control and normal operation of the vehicle.
7, 8	6. The system can be installed and operational in less than 3 hours.	Based on camera, parking sensor, and blind spot sensor installation times, 3 hours is a comparable estimate.

Customer Requirements

1. Has acoustic and visual warnings.
2. Provides warnings if the vehicle is rapidly approaching another vehicle.
3. Provides warnings if the distance between the vehicle and the vehicle in front is too close.
4. Does not affect standby car battery.
5. Does not affect normal operation of vehicle.
6. Can be disabled easily.
7. Easy to install to vehicle.
8. Requires minimal setup.

5 Design

The design of this project incorporated MATLAB's Automated Driving Toolbox & Driving Scenario Designer in conjunction with Texas Instrument's mmWave automotive radar and software development kit (SDK). The design approach involved breaking down the project into Level 0 and 1 functional components, illustrating the system's architecture and the complexities of its design.

5.1 Functional Decomposition

5.1.1 Level 0 Functional Decomposition

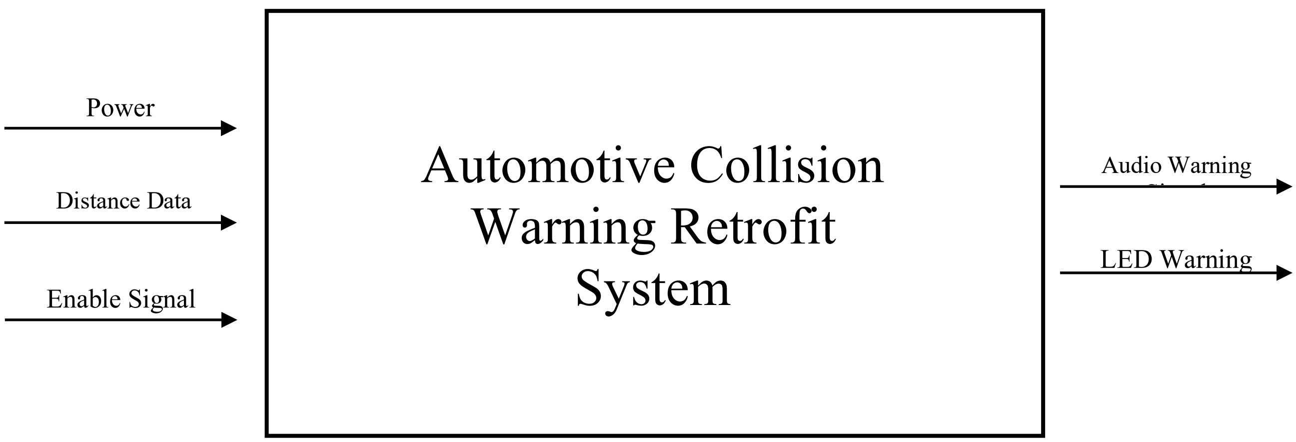

Figure 5.1 – Level Zero Functional Decomposition Diagram

Table III - MODULE 0.1 DESCRIPTION

Module	**0.1 – Automotive Collision Warning Retrofit System**
Inputs	• Nominal 12V DC voltage from car battery • Following distance data from external sensor • Enable signal controlled by user
Outputs	• Signal for audible warning • Signal for visual warning
Functionality	Retrofit collision warning system for automotive applications. Given vehicle distance, warn driver using audible and visual warnings if collision is eminent. Integrate button for user to enable or disable system.

5.1.2 Level 1 Functional Decomposition

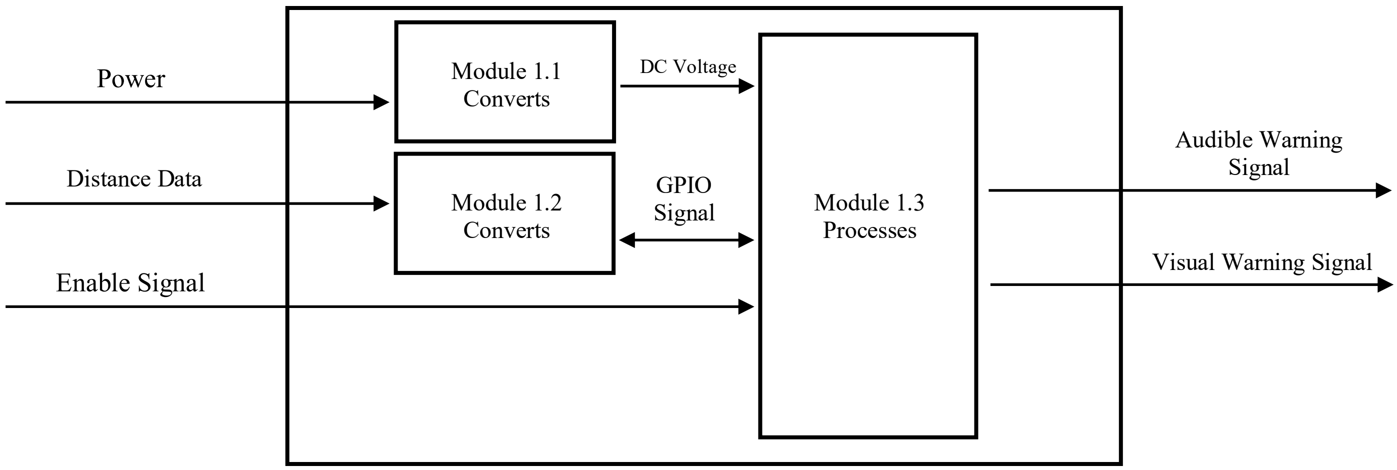

Figure 5.2 - Level One Functional Decomposition Diagram

Table IV - MODULE 1.1 DESCRIPTION

Module	**1.1**
Inputs	• Nominal 12V DC voltage from car battery
Outputs	• Usable DC voltage for module 1.3
Functionality	Converts 12V DC car battery voltage to usable DC voltage for module 1.3

Table V - MODULE 1.2 DESCRIPTION

Module	**1.2**
Inputs	• Following distance data from environment • GPIO signals from module 1.3
Outputs	• GPIO signals to module 1.3
Functionality	Converts following distance data to GPIO signal(s) usable by module 1.3 and reads any control signal(s) from module 1.3

Table VI - MODULE 1.3 DESCRIPTION

Module	**1.3**
Inputs	• 5V DC voltage • GPIO signal(s) from module 1.2 • Enable signal controlled by user
Outputs	• GPIO signal(s) to module 1.2 • Signal for audible warning • Signal for visual warning
Functionality	If enabled, processes following distance data and sends warning signals if needed. If disabled, no data will be processed, and no signals will be sent.

The design concept for this project was to capture object distance data from the vehicle surroundings, interpret this data and send it to a microcontroller to process and issue warnings if necessary.

As this project began, Texas Instruments (TI) mmWave radar was selected and development began. The finalized design (for this project) would not have needed a secondary microcontroller as TI's mmWave radar development boards have the necessary processing capabilities onboard. If this project was continued for mass production where maximum space and cost efficiency were considered, the design could change – rather than using the whole radar development board, the radar integrated circuit (IC) could be used and a proprietary printed circuit board (PCB) could be used.

5.2 Simulation – MATLAB's Automated Driving Toolbox

All simulations regarding this project were done in MATLAB. A very significant contributor to these simulations was MATLAB's Automated Driving Toolbox [15]. The Automated Driving Toolbox provides algorithms and tools for designing, simulating, and testing ADAS and different types of scenarios. The toolbox also provides an application, Driving Scenario Designer. This tool can be used to simulate numerous different driving scenarios and generate sensor data.

In Driving Scenario Designer, the user can create a road, – curved, straight, intersecting, freeways, numerous lanes, varying lane width, lane lines – create different actors, – vehicles (car or truck), pedestrians, cyclists, motorcycles – and generate different sensor data – LiDAR, radar, ultrasonic, camera – allowing them to create whatever driving scenario, they desire.

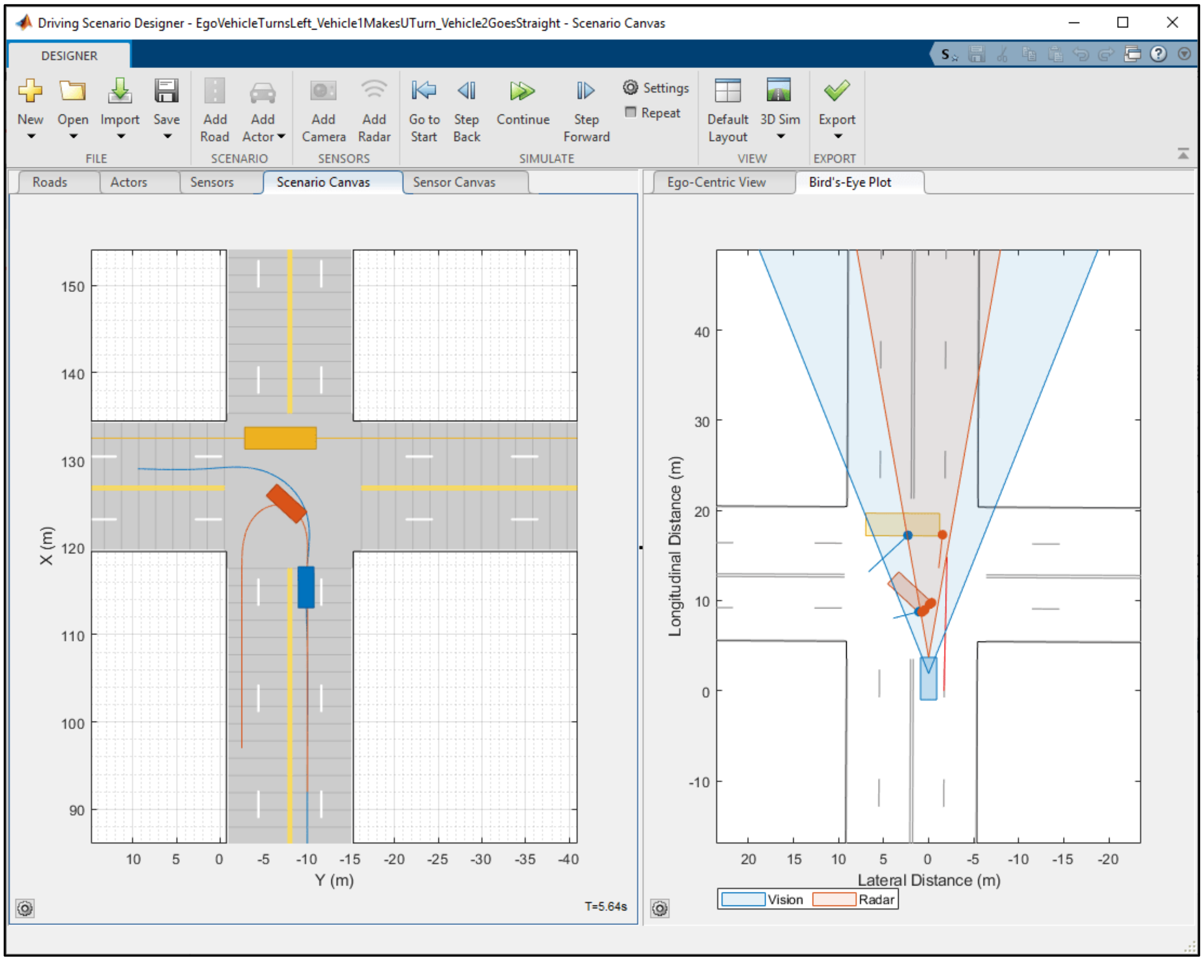

Figure 5.3 - Driving Scenario Designer Example Scenario [16]

This generates sensor data whose parameters can be customized to account for sensing distance, FOV, error rate, and numerous other parameters. In this project's scope, the radar sensor was used and configured approximately to match the respective TI radar that was used. Data was then captured from the radar and was used and processed as needed.

The Automotive Driving Toolbox provided by MATLAB also has numerous useful functions, test algorithms, and visualization tools available. For example, the toolbox provides a multi-object tracker that can be configured to work with different filtering options such as linear Kalman filters or extended Kalman filters. It also provides the ability to visualize these scenarios in bird's eye plots or three-dimensional views.

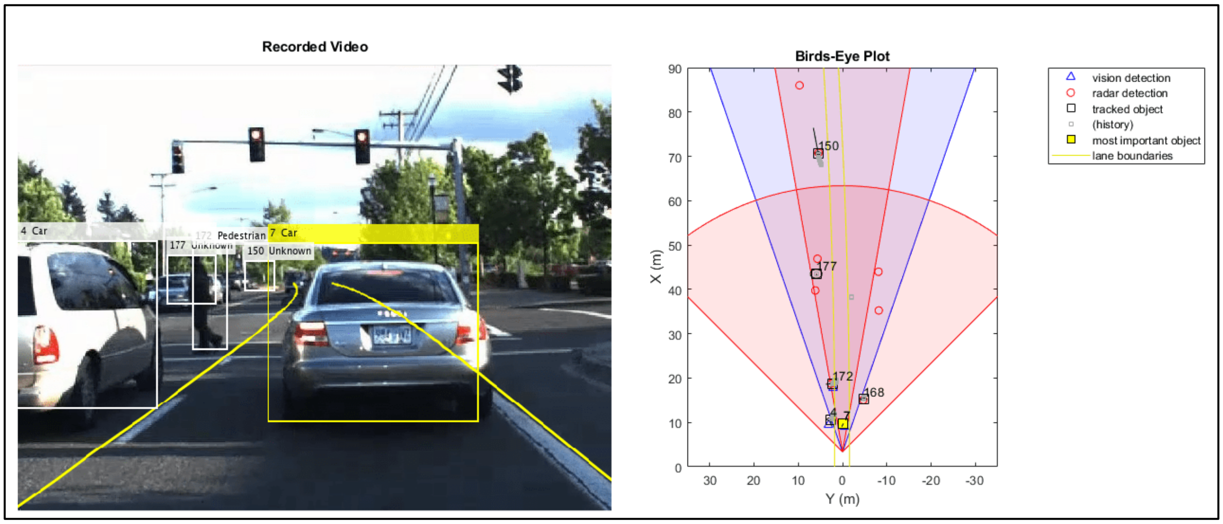

Figure 5.4 - Forward Collision Warning Example with Sensor Fusion using the Toolbox [16]

The Automotive Driving Toolbox is used in many real-world applications and used by many automakers to test their equipment and ensure the proper functionality of their ADAS code, algorithms, and designs. Projects that utilized MATLAB's Automotive Driving Toolbox consisted of lane following systems, automated emergency braking and forward collision avoidance, automated parking systems, traffic negotiation as intersections, and more. This toolbox provided great utility to this project and was the cornerstone of development.

5.3 Hardware – Texas Instruments' mmWave Radar

Progress towards a working prototype began using Texas Instruments' mmWave Radar, specifically the AWR1843BOOST development board. Texas Instruments' automotive mmWave radar sensors combined with their design and development ecosystem help simplify the design process and make it easier to get working devices up and running.

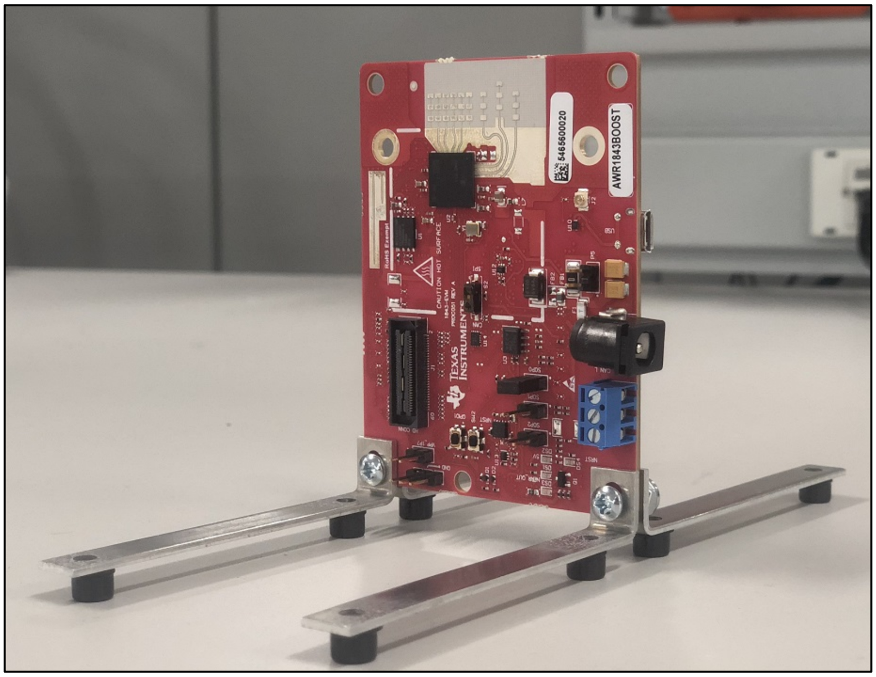

Figure 5.5 - AWR1843BOOST Board [17]

The AWR1843BOOST development board features an on-chip digital signal processing (DSP) core with an ARM-based controller, allowing for onboard programming and debugging. The board also has user GPIO that could be used for this project to trigger warnings or connect to another microcontroller.

Table VII - AWR1843BOOST Specifications [17]

Specification	**Value**
Frequency Range	76 – 81 GHz
FOV - Azimuth	Up to 120 degrees
FOV - Elevation	Up to 30 degrees
Range Resolution	Configurable, as low as 4cm
Maximum Range	Up to ~200 meters
Velocity Resolution	Configurable, as low as 0.5m/s
Maximum Velocity (relative)	± 90 km/h (56 mph)

Texas Instruments also provides a very thorough and useful software development kit that was used greatly in this project. Using an example project provided by Texas Instruments, users can start developing quickly and build off of existing projects. The user can download example projects from TI's radar toolbox development kit and import them into TI's Code Composer Studio application for code analysis and further development. A few of the example projects for their automotive mmWave radar are related to ADAS. Some examples of ADAS projects include automated parking, high-end corner radar, medium-range radar beam-steering (rotating/adjusting the radar beam), medium-range radar, and short-range radar.

Overall, the development board has been very useful in researching radar algorithms and data processing methods, as well as initial project development.

6 Development

Development for a working FCW system began with software simulations and transitioned to hardware testing. The software simulation will be used as a strong foundation for developing a working prototype.

6.1 Simulation

To begin development on the FCW system, MATLAB was used to create a working and functional algorithm for radar data interpretation and time-to-collision measurements. The goal of developing in MATLAB was to simulate various radar data manipulation methods, sensing algorithms, and obtain the most accurate time-to-collision algorithm[1].

6.1.1 Setting Up Design Toolbox & Scenario Creation

As mentioned, MATLAB's Automotive Driving Toolbox and Driving Scenario Designer played a very important role in the development of this project. Simulation development began with the driving scenario designer and creating various scenarios. Below is a preview of the Driving Scenario Designer application when first opened (*Figure 6.1*). The application provides the ability to add roads, lane markings, cars, trucks, pedestrians, cyclists, as well as various different sensors, shown in the top toolbar. Starting with this blank canvas, vehicles, roads, and actors (other vehicles) were added to the scenario and tested.

[1] All MATLAB code is uploaded to the GitHub repository which can be found in the appendix

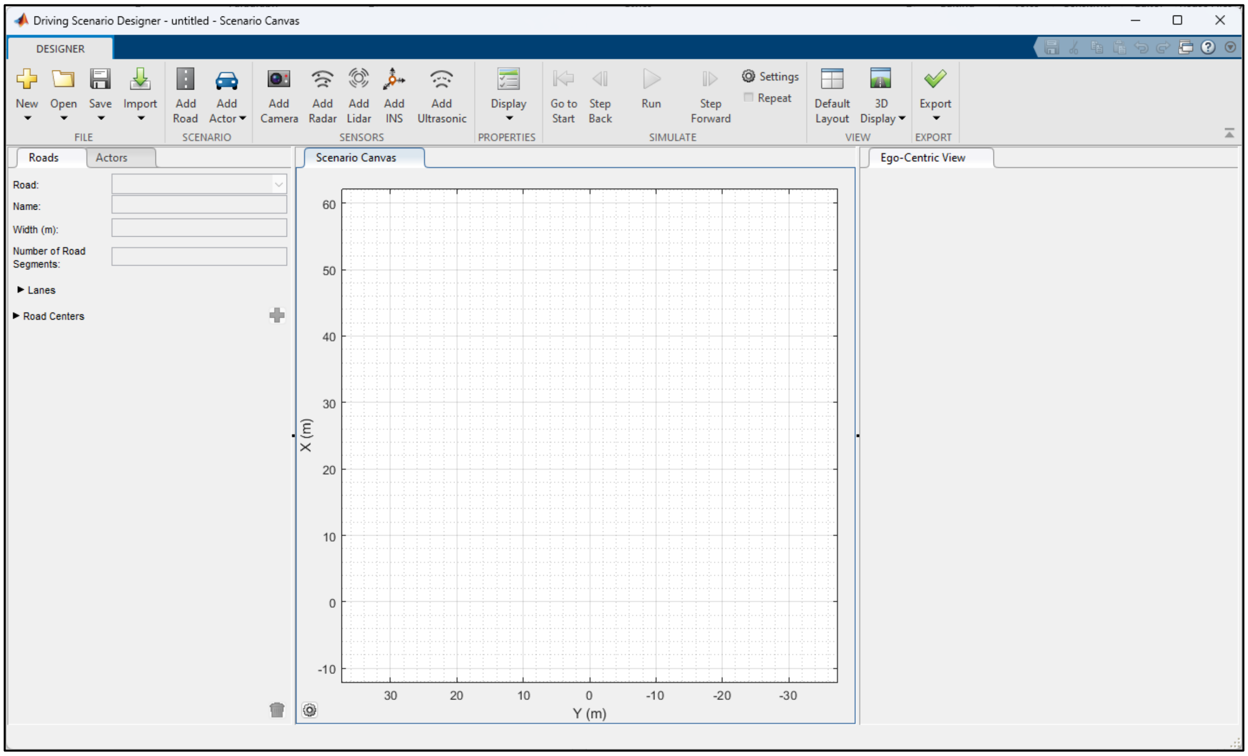

Figure 6.1 - Driving Scenario Designer Application, Startup Preview

Using this application, various scenarios were generated to simulate real world scenarios and situations that may be encountered. These scenarios include barriers, multiple vehicles traveling in the same direction, vehicle traveling in the opposite direction, road curvatures, and non-automobile objects. Below are a few different scenarios that were created; however, these are not an inclusive list.

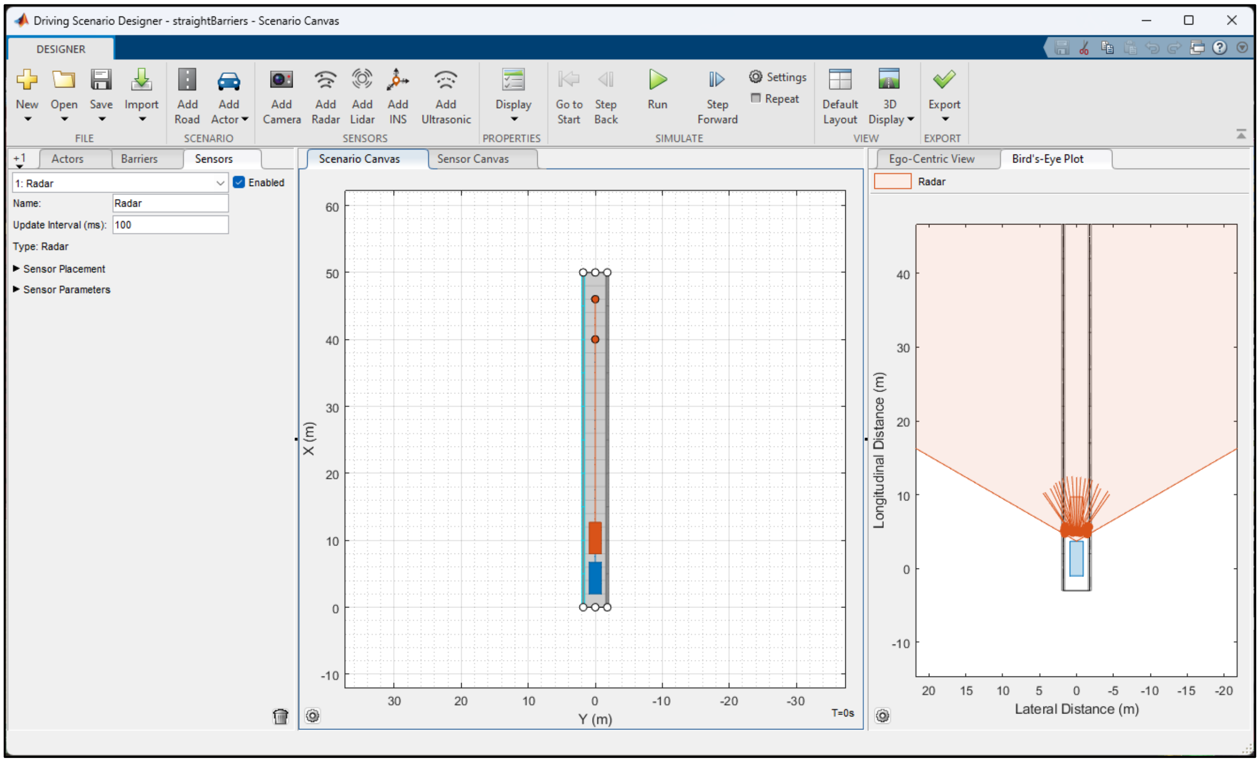

Figure 6.2 - Target Vehicle Suddenly Stopping, with Barriers

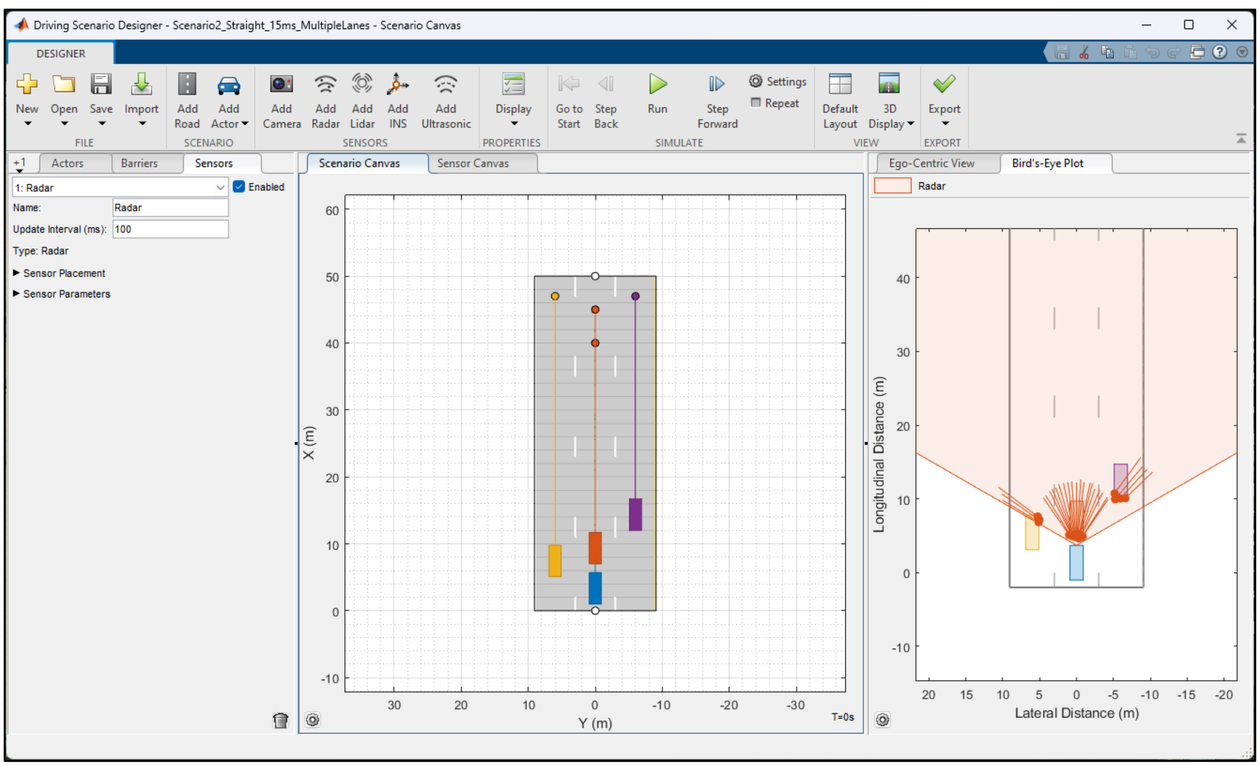

Figure 6.3 - Multiple Target Vehicles, Vehicle Suddenly Stops

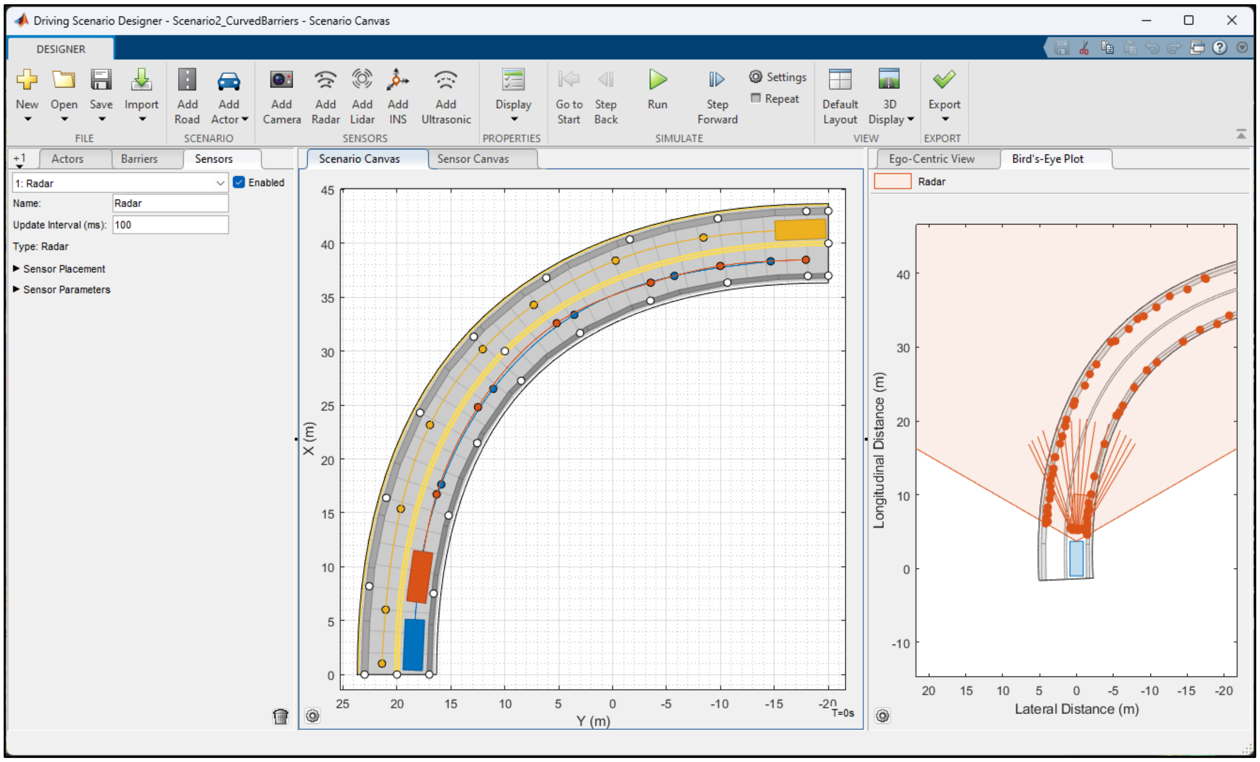

Figure 6.4 - Target Vehicle Driving with Oncoming Vehicle, Curved with Barriers

These scenarios were tested and are considered more advanced scenarios in regard to the IIHS FCW test scenarios. The Driving Scenario Designer was vital in developing tests for this project and generating radar data used for simulation. Once the test scenarios were generated, the simulations were run to generate radar data. The resulting data were object detections with a specific XYZ location and velocity. The next step in development was determining how to use this radar data and eliminate unwanted detections and use this for a FCW alert.

6.1.2 Clustering Radar Objects

A large part of radar data interpretation is determining what detections make up an object – rather than having numerous different single point detections, the goal is to have one large cluster of detections to represent an object.

To achieve this goal, a density-based algorithm for clustering data was researched and decided upon [19, 20, 21, 22] to cluster data points that are close to one another in a dimension and assign them to a single cluster. MATLAB's radar toolbox had an implementation of this available for use that is based off the concept that clusters are dense regions in data space separated by lower density regions, assuming that all dense regions have similar densities [23].

The algorithm measures density by counting the number of data points in a neighborhood, defined as a P-dimensional hyperellipse in the feature space. Distances between points are then calculated using the Euclidean distance metric. The radii of this these hyperellipses are determined by a vector ε, with the ε-neighborhood value being defined by the user.

dBScan clustering starts by identifying all core points, which are points with enough other points in their ε-neighborhood. Points in the ε-neighborhood that are considered core points can either be core points themselves or border points. All points, regardless of being a border point or core point are measured directly from the most directly reachable core point. Points that are neither core nor border points are classified as noise and are not assigned to any cluster. To better understand this, reference the figure below.

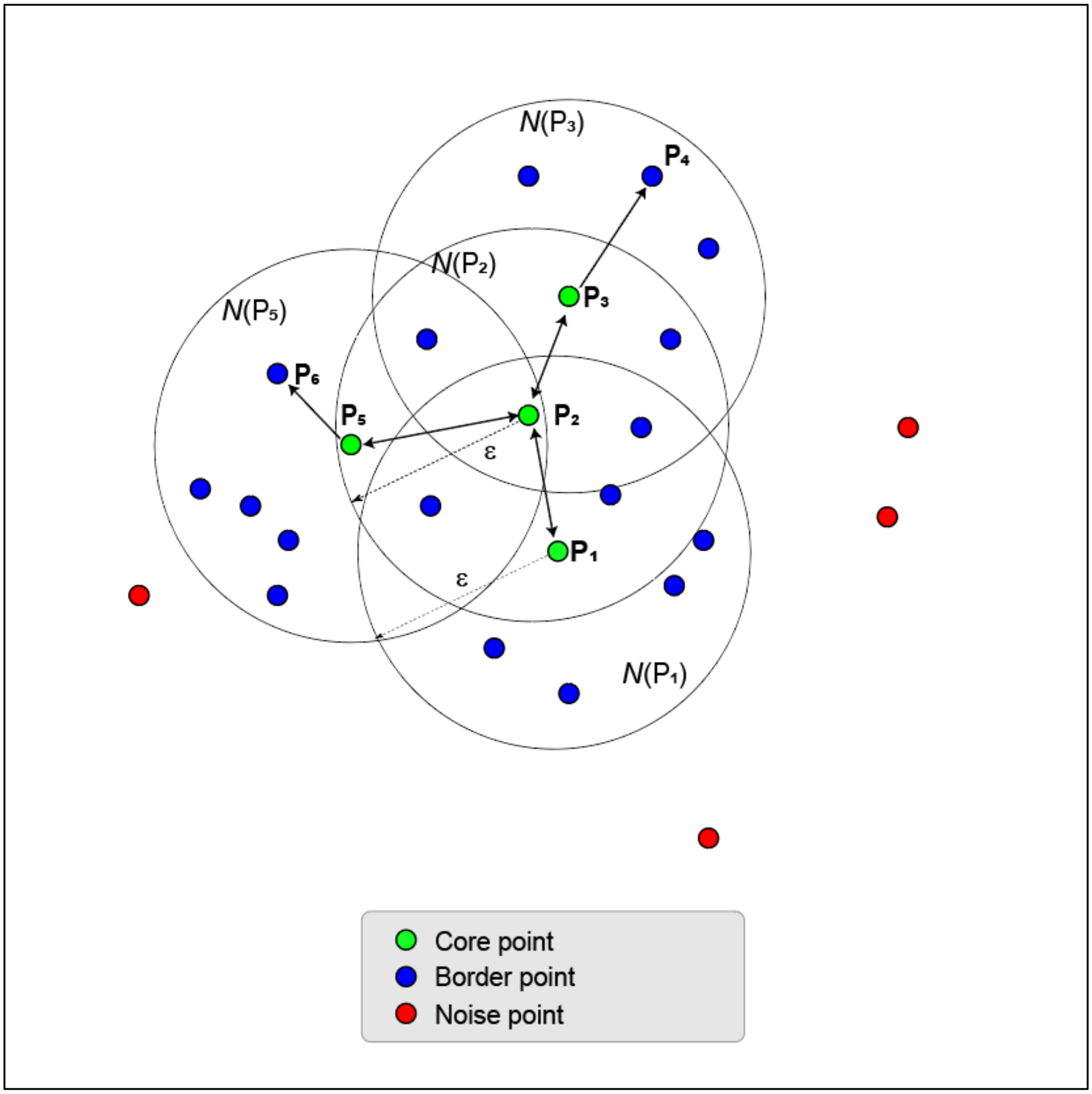

Figure 6.5 - dBScan Clustering

In this figure, the core points are P1, P2, P3, and P5, determined by their distance, ε, relative to all other points. The algorithm used the ε-neighborhood value, scanned the points, and determined the central most points, the core points. The other points shown in blue are border points, those points that are surrounding the core points with a distance ε. The points outside this ε value are shown in red and referred to as noise points.

Once clustering has been applied, the output can be visualized. The ε-neighborhood value can also be adjusted and visualized to see the effect it has on clustering.

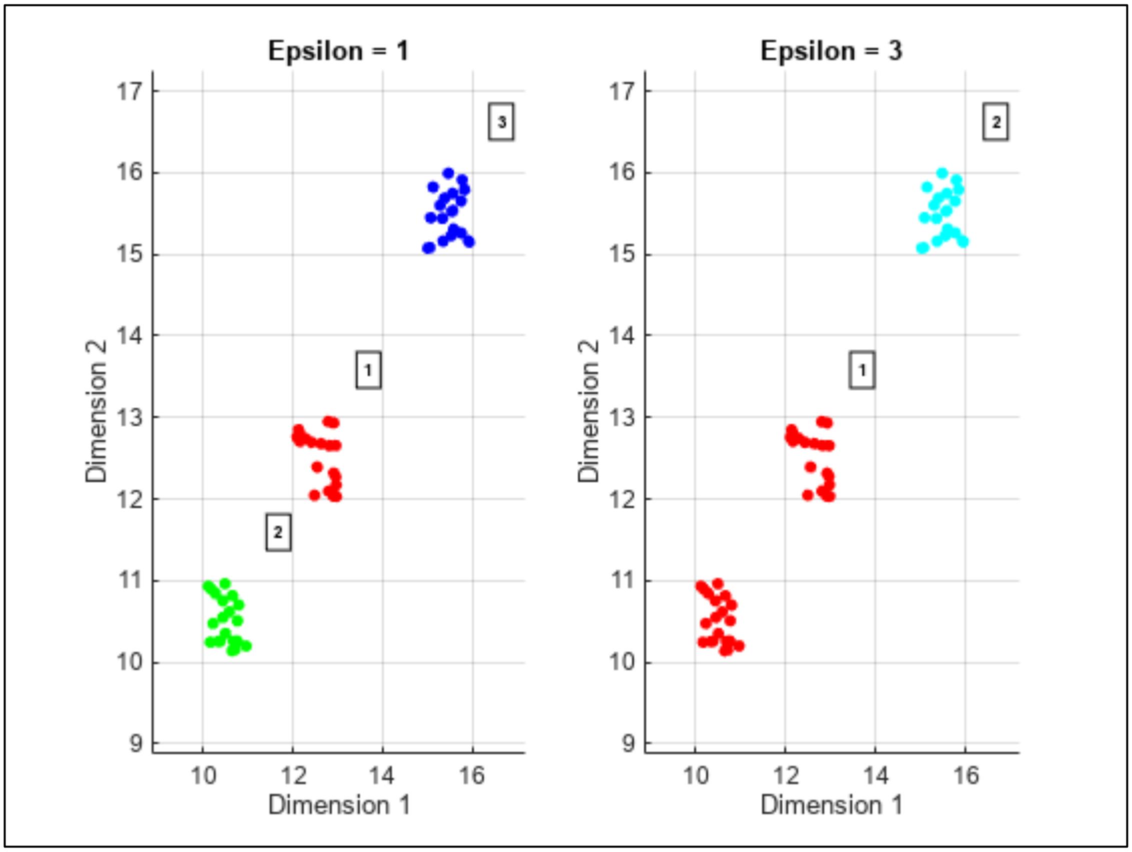

Figure 6.6 - Clustering Visualization using MATLAB

As shown in the figure above, the dBScan algorithm provided by MATLAB's radar toolbox can separate radar detections into individual clusters. From these clusters, the center or edge positions can be found, which will be used to determine which cluster is in front of the vehicle and how close it is, useful for the FCW algorithm.

6.1.3 Removing Radar Clutter & Stationary Objects

Radar data also needs to be filtered to remove noisy measurements and to track objects and their trajectories. While researching how to do this, extended Kalman filtering (EKF) was found and decided upon [23, 24].

The EKF works by linearizing non-linear systems, systems where the state of an object changes not with a linear equation, but rather with respect to some complex equation. For example, this can be a vehicle driving that has unpredictable acceleration and velocity in three dimensions. The EKF approximates the non-linearities of a system with a first-order Taylor series expansion, done to make the non-linear equations suitable for the Kalman filter. The filter has two primary steps, the prediction step, which estimates the current state variables and their uncertainties, and the update step, which refines these estimates based on new observations.

EKF has found a significant role in ADAS where radar sensors collect raw data of surrounding objects providing information regarding range, velocity, and angle of the detected objects. The EKF is then utilized to filter out the noise and account for any uncertainties in object detections. It also can allow for object tracking, predicting the object's state in the current frame based on previous state estimates, then correcting the prediction using the new radar estimates.

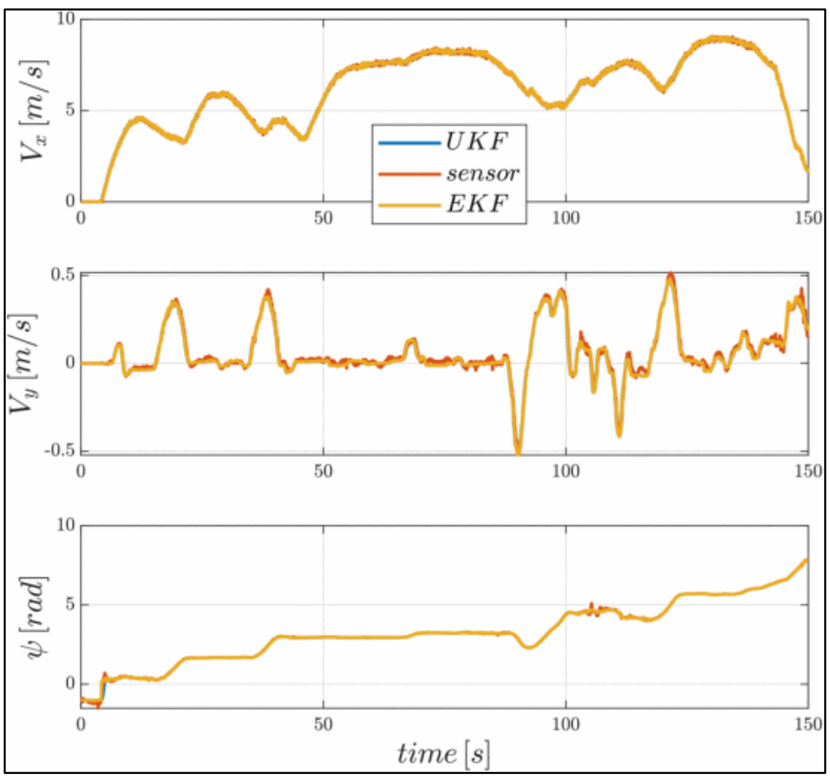

Figure 6.7 - EKF's Effect on Velocity [24]

In the figure above, we can see the comparison of an unscented Kalman filter (UKF), the basic sensor data, and an extended Kalman filter. We can exclude the UKF for our project purposes. In the figure, yellow represents the velocities in each direction of an object with an EKF applied.

The orange line is the same velocities without any filtering. As seen, the data with an EKF is much smoother and less noisy compared to the unfiltered data. The project will demonstrate the effectiveness of EKF and how it can be implemented in a FCW system.

6.1.4 Forward Collision Warning Algorithm

The project will use the Euro NCAP Automatic Emergency Braking (AEB) algorithm to determine when to issue forward collision warnings. This algorithm uses the relative velocity of the object in front of it and a deceleration constant to determine a distance. If the vehicle in front of the ego vehicle is closer than this distance, a warning will be issued. The formula for the algorithm is shown below:

$$d_{FCW} = 1.2 * v_{rel} + \frac{v_{rel}^2}{2a_{max}}$$

where:

- d_{FCW} is the forward collision warning distance
- v_{rel} is the relative velocity between the two vehicles
- a_{max} is the maximum deceleration, defined to be 40% of the gravity acceleration (9.8m/s x 0.4)

The system will use this algorithm to determine the forward collision warning distance using the position of the objects in front of the ego vehicle, along with the relative velocity in both the x (forward/backward) direction and the y (left/right) direction.

A warning will be issued if the following conditions are met:

- The object is within radar sensing distance (~150 meters)
- The object is within the ego vehicle's lane (a lane is considered 6 meters wide, ∴ inside ego vehicle lane is considered ± 3 meters left and right of the vehicle)
- The object's relative speed is < 0 m/s.
- The object's distance is < d_{FCW}.

If all these conditions are met, a warning will be issued to the driver.

6.2 Hardware

Development on the hardware aspect of the project had begun but was not fully realized due to time constraints. Using the AWR1843BOOST radar development board and example projects from TI, research and development occurred towards implementing a functional FCW system.

6.2.1 TI's Automotive mmWave Radar

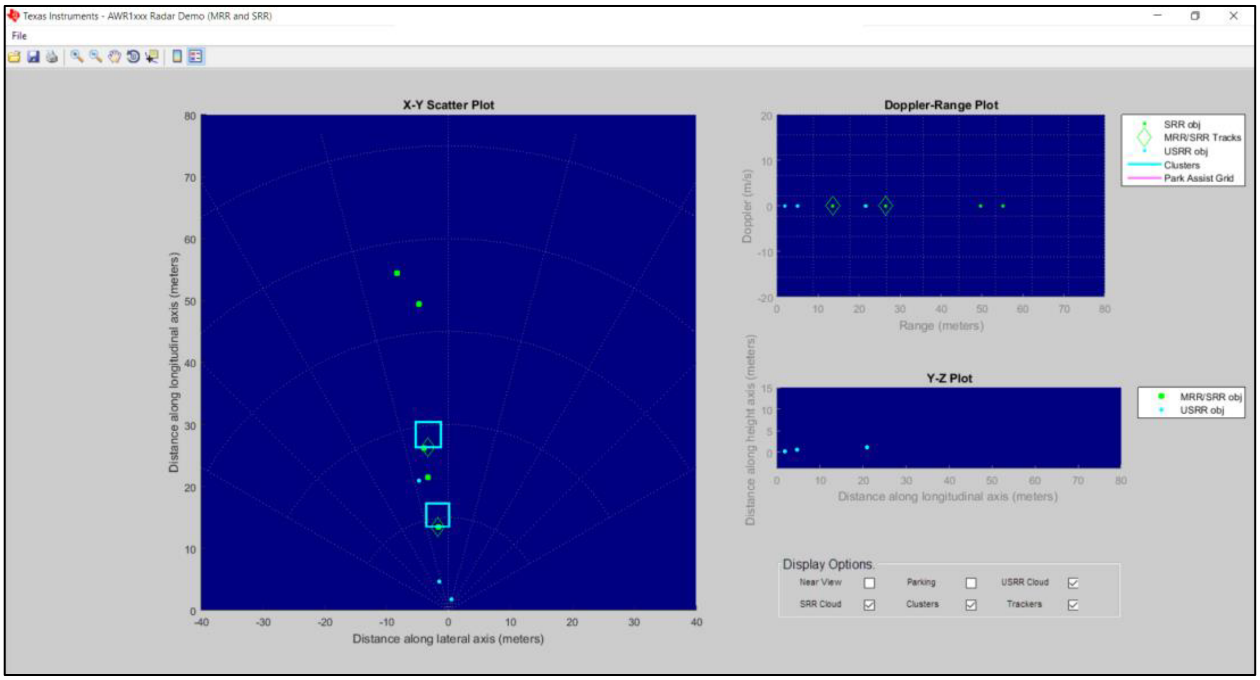

Figure 6.8 - MRR Example Project GUI [18]

This project used a Medium Range Radar (MRR) example project from TI's website [18] and planned to implement an FCW algorithm on top of the already functional code. The example project utilizes the DBSCAN clustering methodology to process radar data and displays the object clusters in blue squares. It also uses an extended Kalman filter to process moving radar objects and remove radar clutter.

Code analysis has begun on this example project, however implementation of the FCW algorithm has not begun due to previously mentioned shipping and time constraints. However, the example project was able to be used to better understand and interpret the DBSCAN clustering and extended Kalman filtering algorithms which has been important for project development and simulation.

6.2.2 Test Setup

In order to begin testing the radar to ensure proper functionality and real-world testing, the radar needed to be mounted to a vehicle. To do this, a GoPro video camera suction mount was used in combination with a custom-made bracket. This setup is only meant to be temporary and functional; this is not what would be implemented in a production environment.

Figure 6.9 - Radar Mounting Setup

This bracket and mounting setup were used with an already owned GoPro suction mount, suitable to be mounted on any flat surface – for example the hood of a car. The bracket connecting the GoPro suction mount to the AWR1843BOOST radar was fabricated by hand out of 1/16 in. thick by ½ inch wide aluminum. The bracket is connected to the radar using size #8 machine screws.

To perform testing, the radar was attached to the hood of the test vehicle. Power, supplied with a 5V DC barrel connector, and USB laptop connection were ran along the hood, through the passenger side door, and into the vehicle cabin. A 12V DC car outlet to AC outlet converter was used to power the radar. A laptop was also positioned inside the vehicle for data collection.

Figure 6.10 - Radar Mounted to Test Vehicle

6.2.3 Radar Software & Forward Collision Warning Implementation

As mentioned previously, an dBScan clustering algorithm and extended Kalman filter were used in the demo project[2]. If additional time for development were available, FCW would have been tested and implemented on the radar in real-world conditions. Implementation of the FCW system would have followed the same algorithm mentioned in the simulation section (6.1.4). The demo project provides the object clusters with an average XY position, as does the MATLAB simulation. Therefore, the output of the clustering function, which is a clusteringObject structure

---

[2] Project code pertaining to the EKF and dBScan, along with all other code and the full project can be found in the GitHub repository which is linked in the appendix.

that contains x-position and y-position info, as well as average velocity, can be used in another function to check the time-to-collision and issue a warning. A sample of how the FCW algorithm might be implemented is below, however, testing, analysis, and modifications would need to be done on the dbScan and EKF functions.

```
// Function to check for threat event
void checkForThreatEvent(clusteringDBscanReport_t *clusterData) {
    float32_t d;

    // Check if the cluster yCenter is inside the vehicle lane
    if (clusterData->yCenter >= -3 && clusterData->yCenter <= 3) {
        // Calculate time-to-collision distance using Euro NCAP AEB formula
        d = fabs(clusterData->avgVel) * 1.2 + (clusterData->avgVel *
clusterData->avgVel) / (2 * 0.4 * 9.8);

        // Check if yCenter < time-to-collision distance
        if (clusterData->yCenter < d) {
            GPIO->ODR |= GPIO_OUTPUT; // Issue FCW, toggle GPIO
        }
    }

    GPIO->ODR &= ~GPIO_OUTPUT; // Issue FCW, toggle GPIO // No threat event
```

Figure 6.11 - Sample FCW Code for Radar Implementation

7 Testing & Results

This project consists of both software and hardware testing. The software implementation includes the FCW system and various test scenarios. The hardware implementation simply demonstrates and explains the radar demo project.

7.1 Simulation

The software was tested using various scenarios, ranging from less advanced to more advanced, demonstrating the filtering, clustering, and time-to-collision algorithms as one unified system.

7.1.1 I.I.H.S. Test Scenarios

Vehicles in the United States are tested by the Insurance Institute of Highway Safety (IIHS) for crash safety, crash prevention safety, and much more. The test scenarios follow the IIHS tests that are performed for front crash prevention which consist of the test vehicle approaching a stationary vehicle a 12 MPH and 25 MPH.

The first scenario was set up as shown below in *Figure 7.1* – the ego vehicle (in blue) traveled ~50 meters towards a stopped vehicle at 12 MPH (5.36 m/s). The initial ego vehicle point is shown by a blue dot near X = 0 and the end ego vehicle point is shown by a blue dot near X = 55.

In *Figure 7.2*, the ego vehicle begins travel from around X = 0 and initially the doesn't detect enough points to represent a cluster. A cluster needs to be represented by a minimum of 3 radar points to ensure noise is not being detected. As shown, when the system does not detect any clustered objects, there is no danger of collision shown by a green warning symbol.

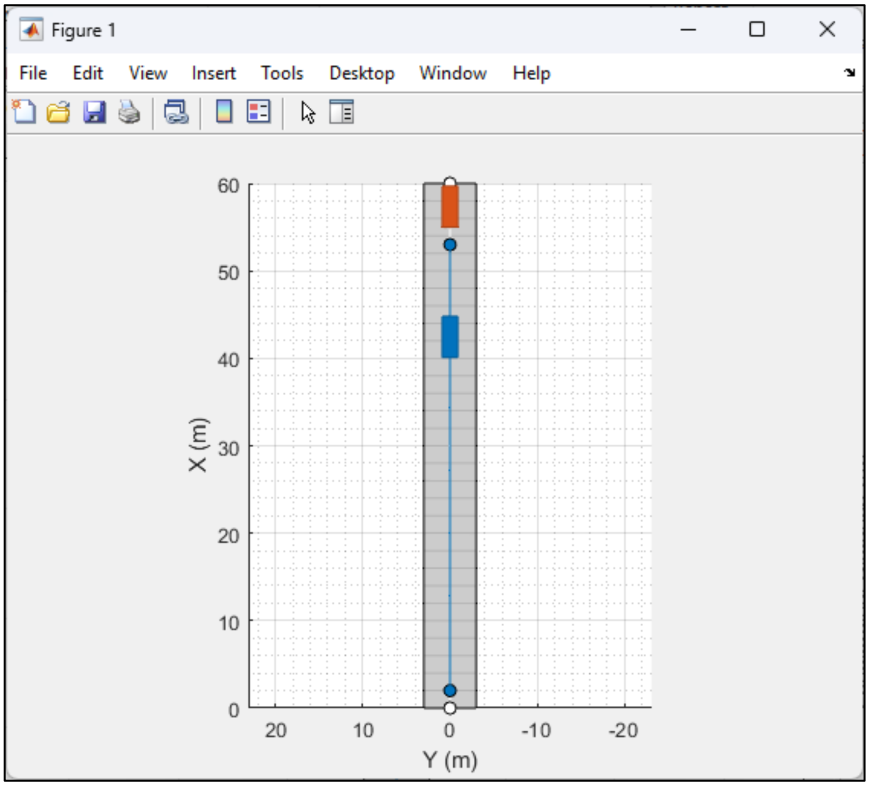

Figure 7.1 - IIHS Scenario View, Ego Vehicle towards stopped vehicle)

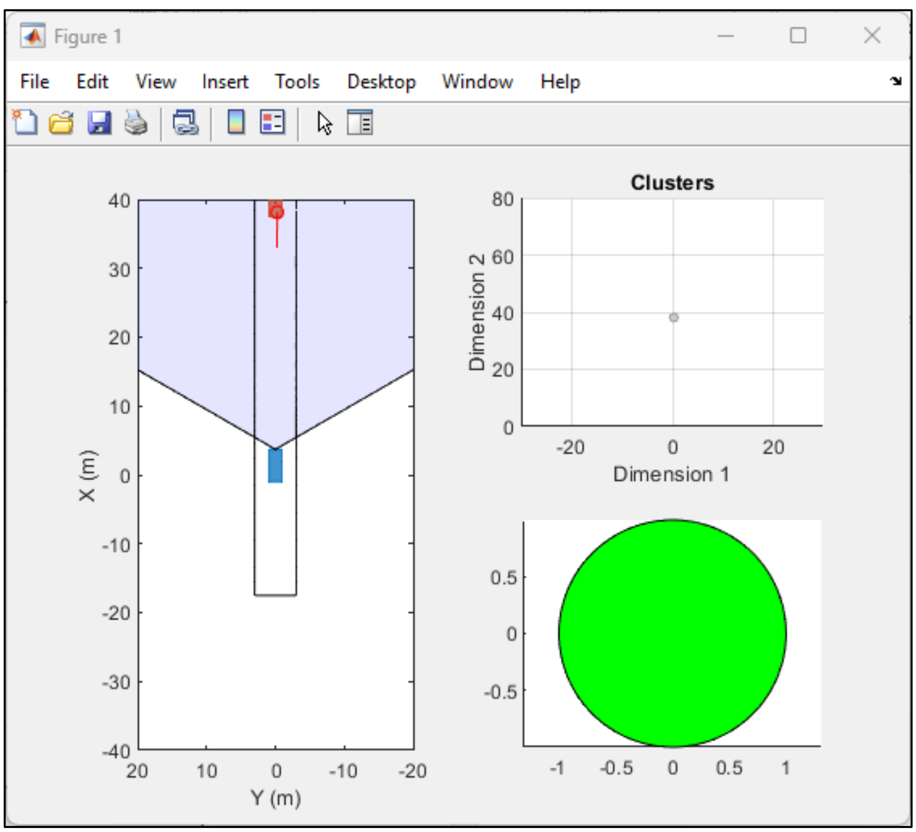

Figure 7.2 – 12MPH IIHS Scenario, No Warning

When enough data points are detected to generate a cluster, the system then determines if that cluster is getting closer to the ego vehicle with a negative relative velocity. If this occurs, the warning symbol turns yellow – simply an indication a vehicle is approaching, no warning is issued. This can be seen in *Figure 7.3*.

Eventually, when the ego vehicle's speed and distance relative to the vehicle ahead are less than safe – as determined by the FCW algorithm, the warning symbol turns red, indicating a potential for collision, and a warning is issued, shown in *Figure 7.4*. The system first detected a cluster when the stationary vehicle was ~82 ft (~25 m) away from the ego vehicle.

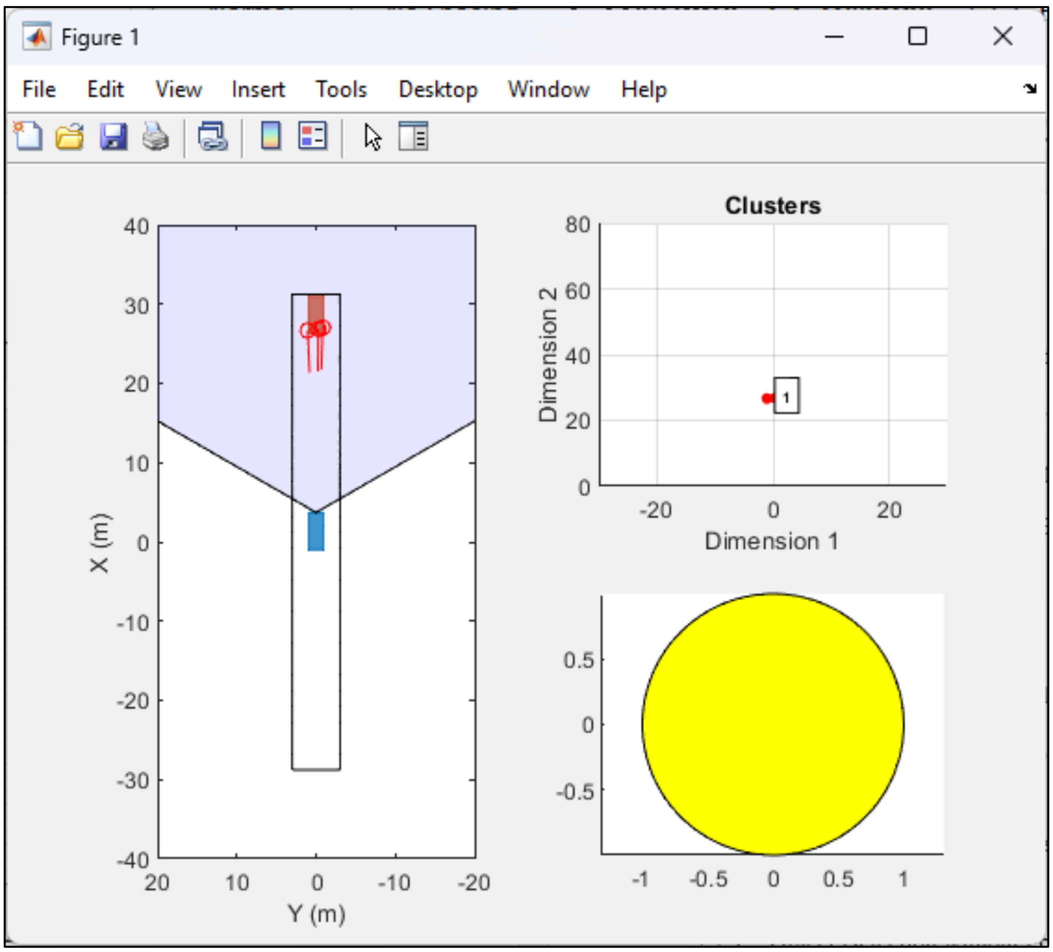

Figure 7.3 – 12MPH IIHS Scenario, Vehicle Ahead Detected, No Warning

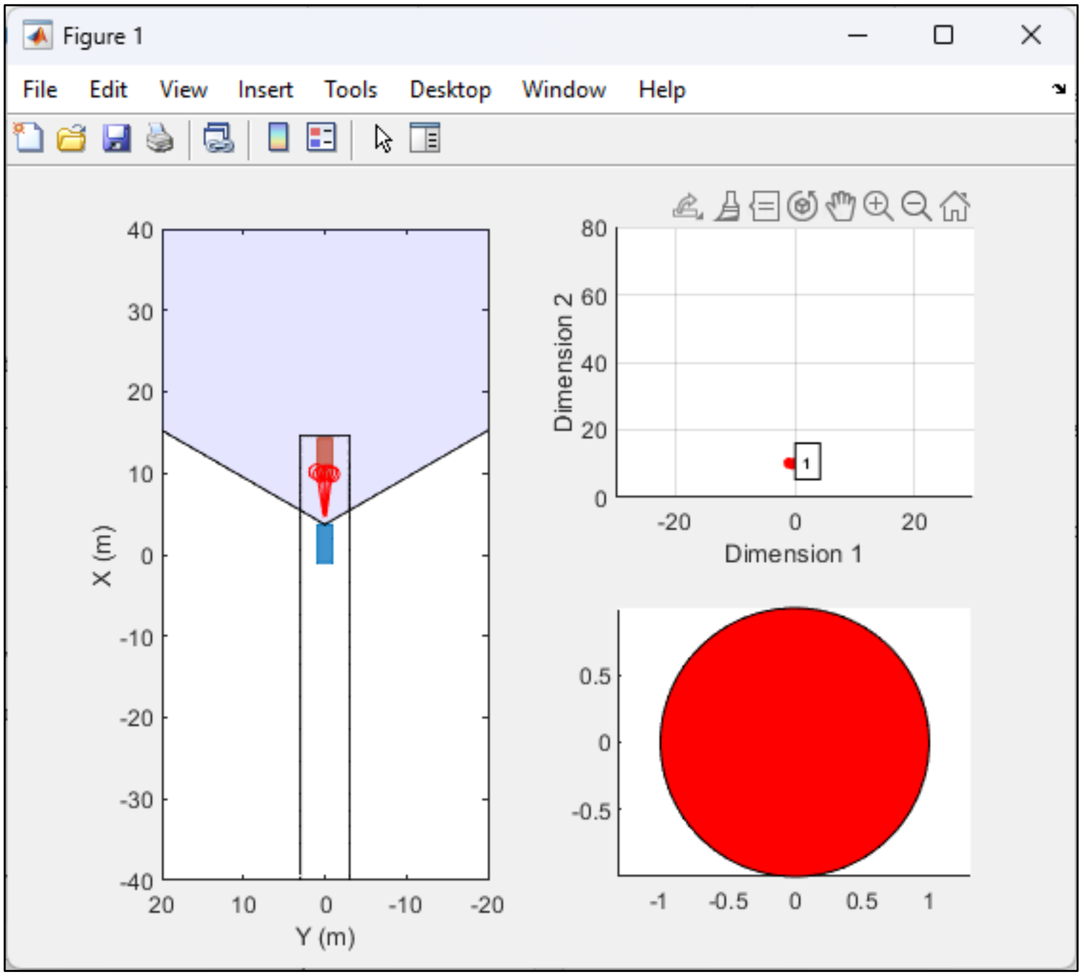

Figure 7.4 - 12MPH IIHS Scenario, FCW Alert Active

In Figure 7.4, pictured is the very first frame/time step the collision alert is active. At this point, the ego vehicle is ~32.2 ft (~9.81 m) away from the stationary vehicle. The safe distance determined by the FCW algorithm for this speed is ~32.6 ft (~9.96 m). Using the vehicle's speed and distance, the time-to-collision can be calculated.

$$12\ MPH * \frac{5280\,\frac{ft}{mile}}{3600\,\frac{sec}{hr}} = 17.6\ ft/sec$$

$$\frac{32.2ft}{17.6\,\frac{ft}{sec}} = 1.83\ seconds\ until\ collision$$

As we can see, the system provided a forward collision warning when the ego vehicle was approaching the stationary vehicle, and the algorithm determined the distance to relative speed was unsafe.

Moving on to the 25 MPH test, the scenario will remain the same as shown in *Figure 7.1*, except for the increase in speed. The vehicle radar obtains sufficient number of points for clustering at ~89.2 ft (~27.2 m), shown in *Figure 7.6*. The moment the cluster is detected, the system calculates the safe distance using the FCW algorithm and issues a warning, as shown by the red warning in *Figure 7.6*. In *Figure 7.5*, we see the moment before the cluster was detected – no alert was issued.

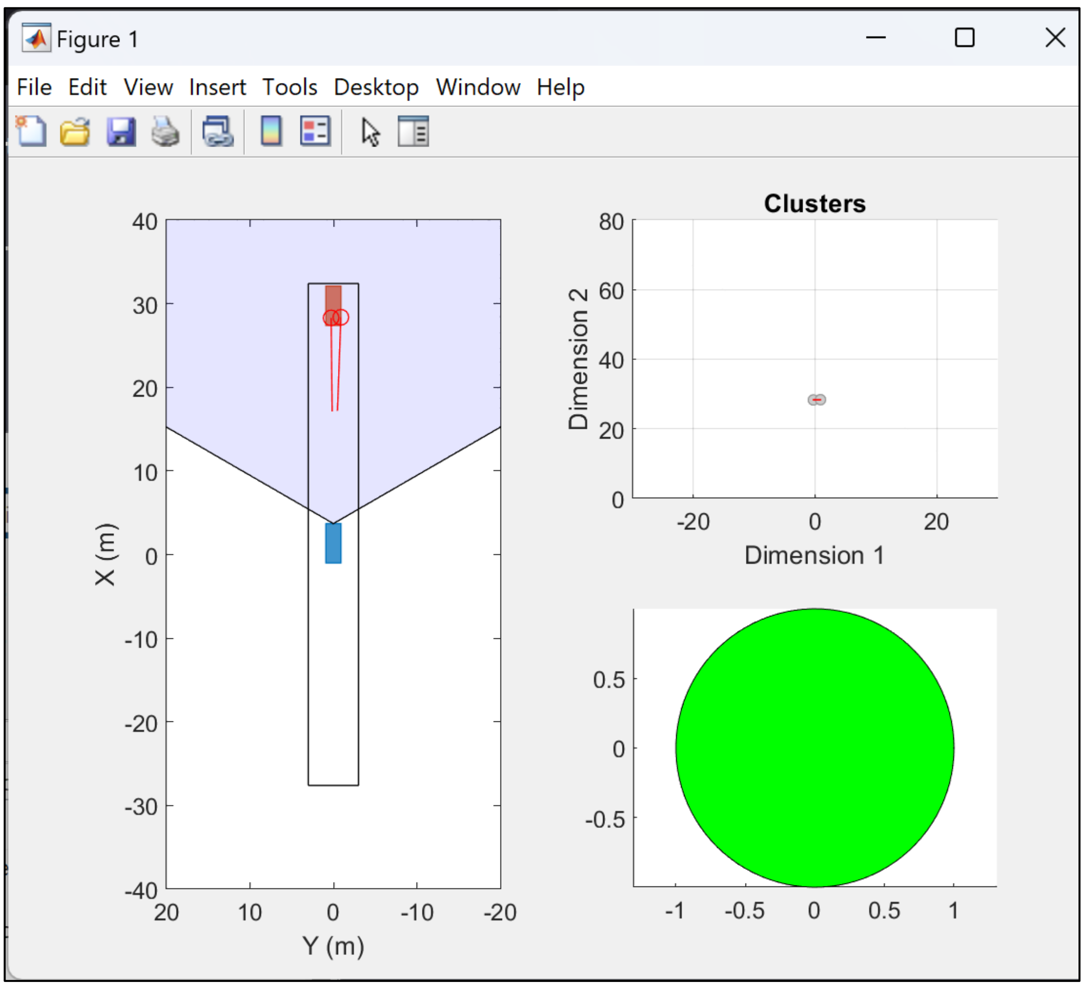

Figure 7.5 - 25MPH IIHS Scenario, No Warning

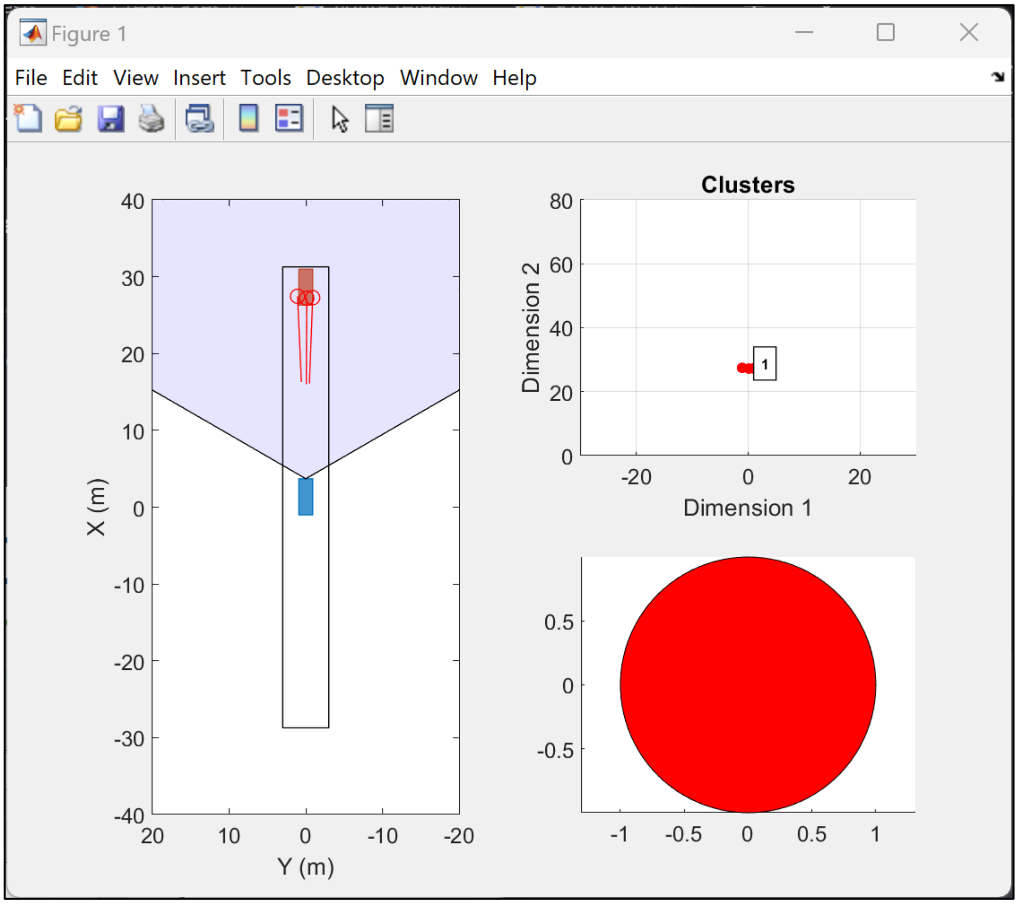

Figure 7.6 - 25MPH IIHS Scenario, FCW Alert Active

In *Figure 7.6* above, the first frame where the FCW alert is active is shown. The distance between the ego vehicle and the stationary vehicle is ~89.2 ft (~27.2 m). The distance-to-collision from the FCW algorithm is ~96.2 ft (~29.3m)[3]. Calculating the time-to-collision for this scenario is shown below.

$$25\ MPH * \frac{5280\,\frac{ft}{mile}}{3600\,\frac{sec}{hr}} = 36.6\ ft/sec$$

$$\frac{89.2ft}{36.6\,\frac{ft}{sec}} = 2.43\ seconds\ until\ collision$$

As we can see, these simulations function well and as expected when compared to the Insurance Institute of Highway Safety's front collision prevention tests.

[3] The 7ft difference between the FCW alert distance and the actual vehicle distance is due to the simulation update time. The simulation updates every 0.1ms whereas the radar updates every 0.01ms, ∴ the FCW alert could have triggered earlier, at a distance less than 7ft.

7.1.2 Advanced & Real-World Test Scenarios

Simulation was also performed in more advanced scenarios that may appear in the real world, rather than a controlled environment like the IIHS test scenarios. There are three scenarios that were simulated that represent many different real-world conditions and test the overall robustness of the system. These scenarios consist of – driving with multiple cars and a barrier on the left (*Figure 7.7*), driving around a curved road with multiple lanes and vehicles (*Figure 7.11*), and lastly, driving around a curve with barriers on both sides and an oncoming vehicle with a vehicle ahead stopping (*Figure 7.16*). These scenarios are meant to test the system's ability to filter out other objects besides the vehicle in front.

In the first scenario, the ego-vehicle was traveling along a road with a barrier on the left, a vehicle in front, and a vehicle in the lane to the right, shown in *Figure 7.7*. The ego vehicle is traveling at ~33 MPH (~15 m/s), the vehicle ahead is traveling at ~29 MPH (~13 m/s), and the vehicle in the lane to the right is traveling at ~20 MPH (9 m/s). The scenario begins with the system immediately detecting the vehicle ahead getting closer, as shown by the yellow FCW alert in *Figure 7.8*. As the scenario progresses, the system still does not detect the barrier on the left and is about to stop detecting the vehicle to the right, shown in *Figure 7.9*. Ending the scenario, the distance between the vehicles is about ~9.2 ft (~2.8 m) and the system sends a FCW alert, shown in *Figure 7.10*.

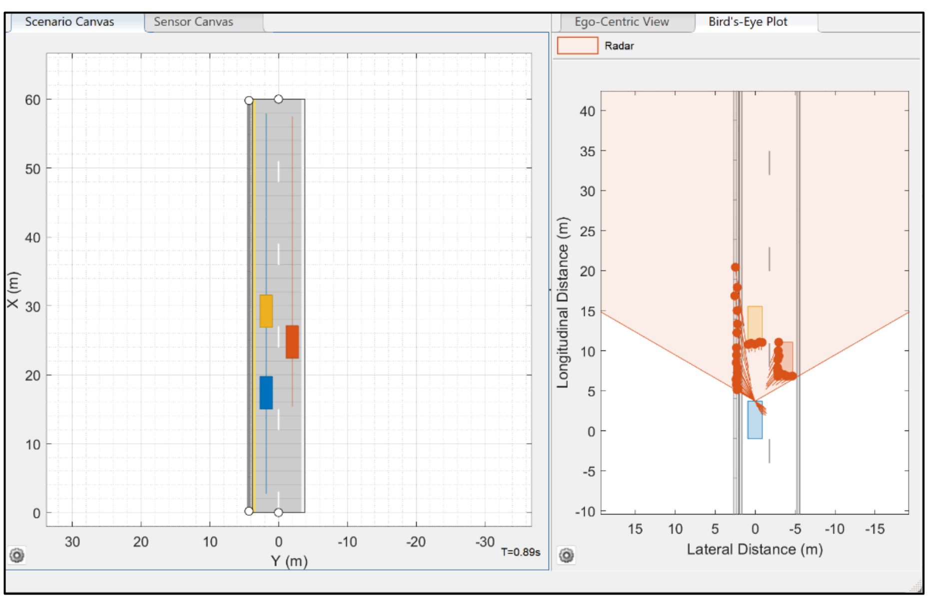

Figure 7.7 - Roadway with Barrier Scenario

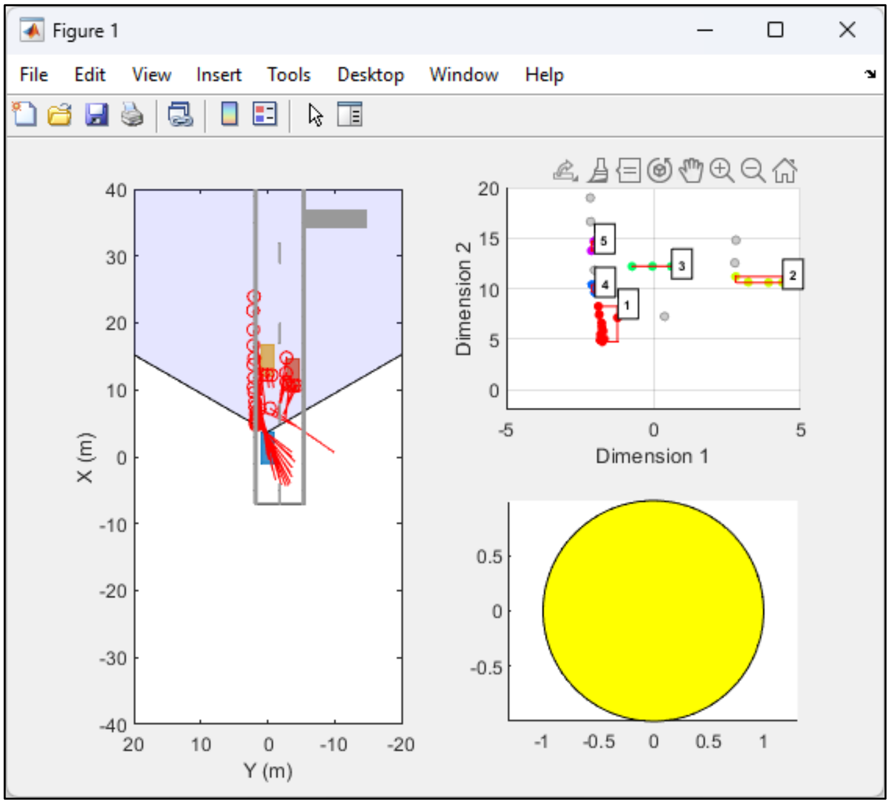

Figure 7.8 - Roadway w/Barrier, Vehicle Detected, No FCW Alert

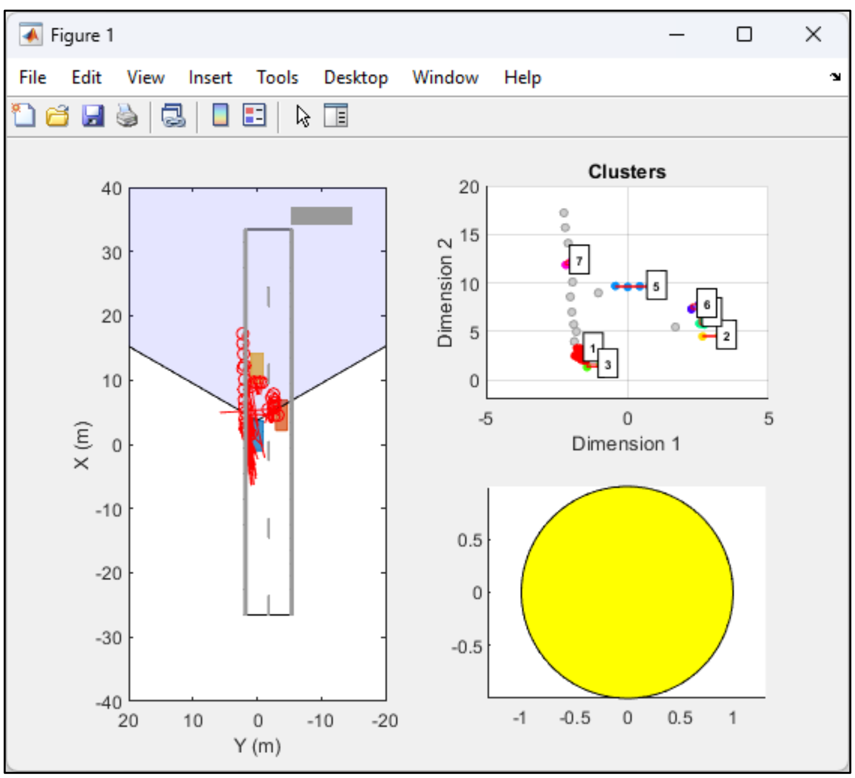

Figure 7.9 - Roadway w/Barrier Scenario 2, Vehicle Detected, No FCW Alert

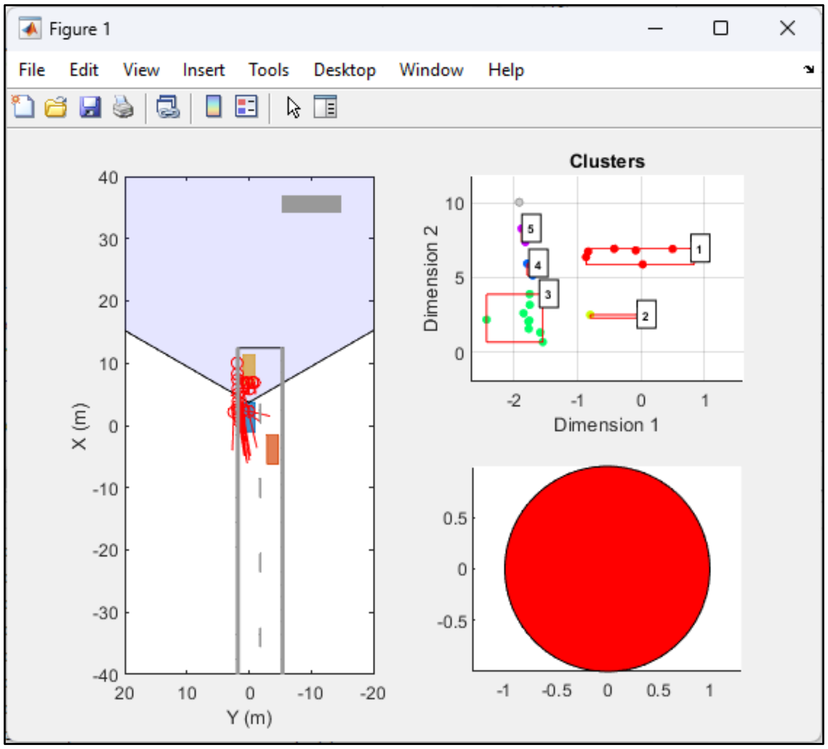

Figure 7.10 - Roadway w/Barrier, FCW Alert Active

This scenario performed very well, filtering out the barrier on the left and not issuing a FCW alert until the end. The FCW alert performed, but not adequately. The alert was issued with < 10ft of following distance with a speed difference of ~4 MPH, giving a time-to-collision of about 1.70 seconds at the current speed. Comparing this result to the FCW algorithm, this is acceptable, but in real world conditions, the driver might want to see a FCW alert earlier.

Next, the scenario with the curved barriers and oncoming vehicles was tested, as shown in *Figure 7.11*. The ego-vehicle, vehicle in front, and the vehicle traveling the opposite direction had a speed of ~33 MPH (~15 m/s). When beginning driving forward, the clustering algorithm struggles to differentiate both barriers and the vehicles, instead creating one large cluster, shown in *Figure 7.12*. Continuing, once the ego-vehicle enters the curve, the system detects the barrier as in front of the vehicle and issues a FCW alert shown in *Figure 7.13*. One back in the middle of the curve, the system does not detect any object in front and does not issue any warning as expected (shown in *Figure 7.14*). Finally, the vehicle ahead of the ego-vehicle stops and the ego-vehicle begins to approach it, triggering a FCW alert as expected, shown in *Figure 7.15*. The ego vehicle was ~51 ft (~15.6 m) behind the stopped vehicle with a relative speed difference of ~33 MPH (~15 m/s).

The FCW algorithm determines a warning should be issued at ~150 ft (~46 m). This large difference is due to the radar system not being able to detect the stopped vehicle until after the ego-vehicle has rounded the curve. This results in a 1.04 second time-to-collision which is considerably late at this speed.

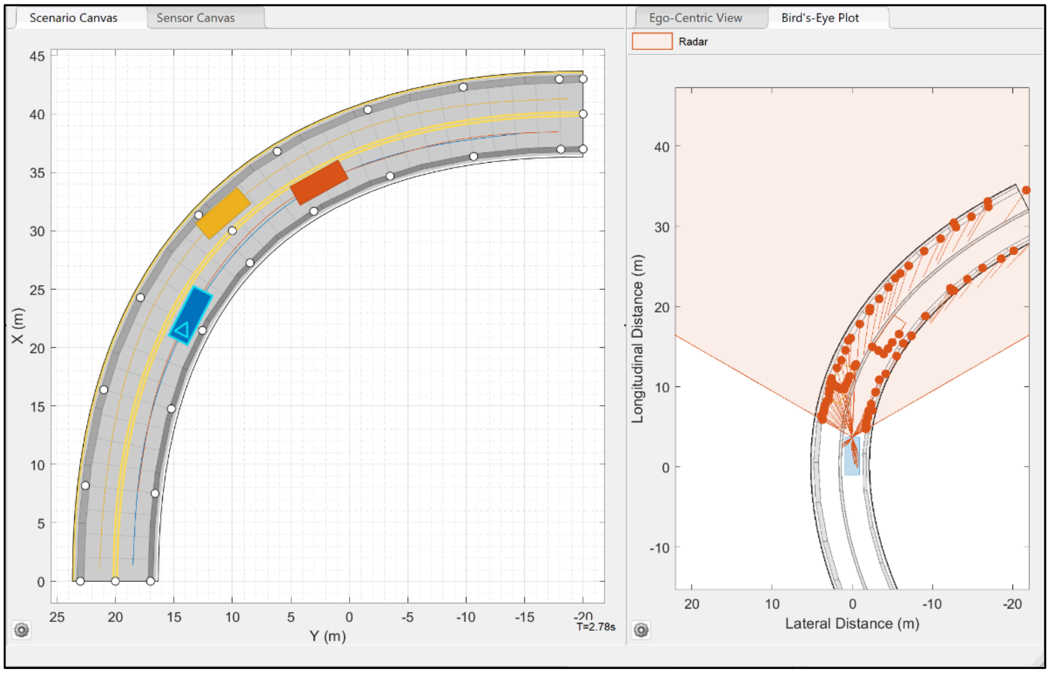

Figure 7.11 - Curved Barrier w/Oncoming Traffic Scenario

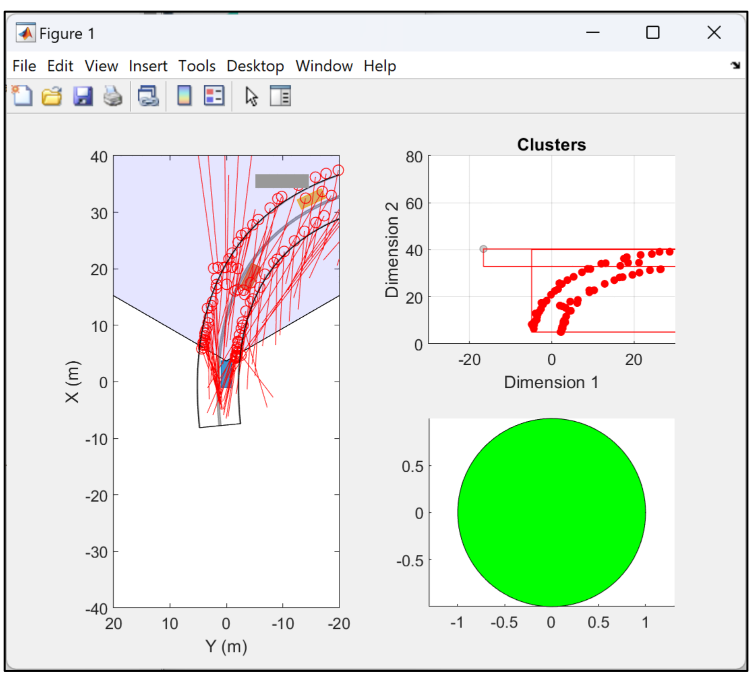

Figure 7.12 - Curved Barrier Scenario, Clustering Error

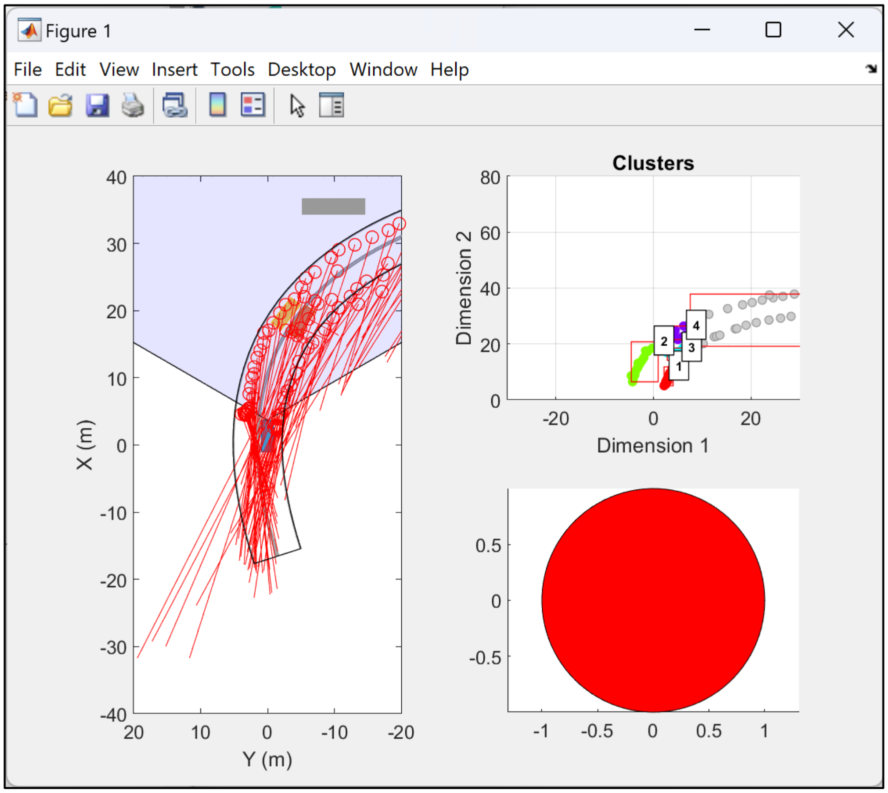

Figure 7.13 - Curved Barrier Scenario, FCW Alert Error

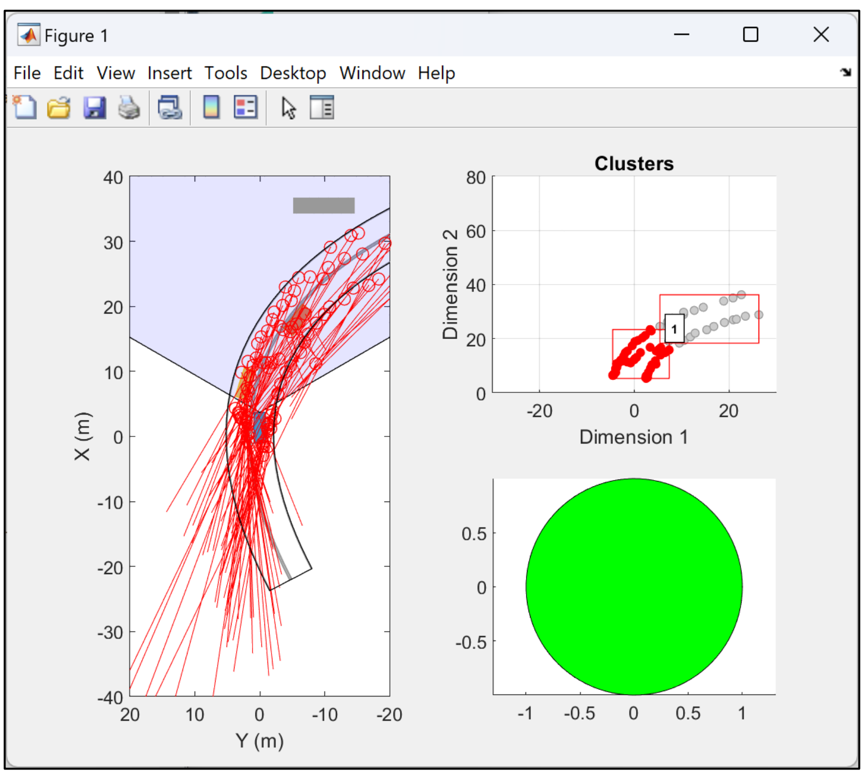

Figure 7.14 - Curved Barrier Scenario, No Alert in Curve

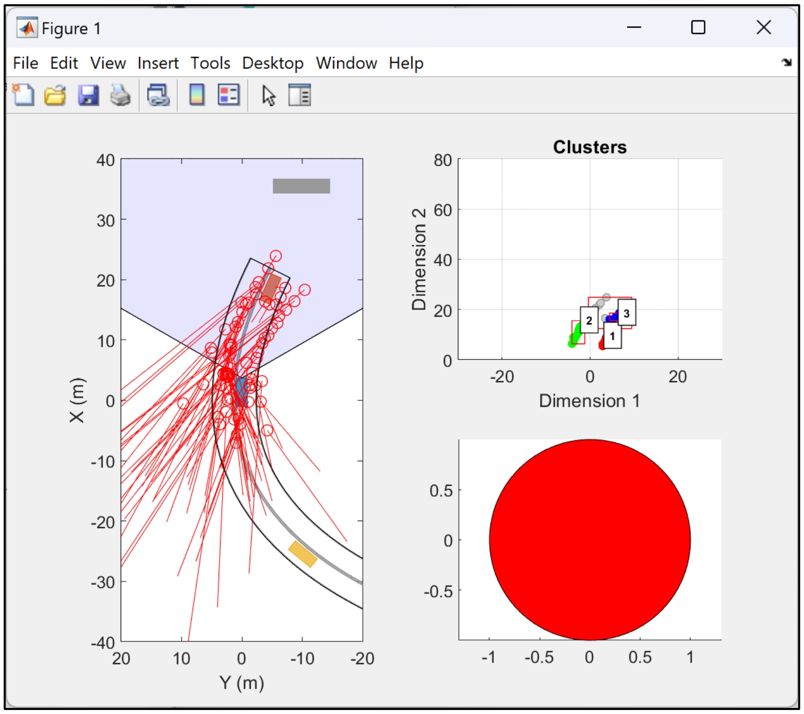

Figure 7.15 - Curved Barrier Scenario, FCW Alert Active

This scenario challenged the system and its ability to filter out objects that are not the vehicle in front of the ego-vehicle. The system's clustering algorithm and tracking algorithm was confused due to the barriers and caused false and inadequate alerts from the FCW system.

Lastly, simulation was performed on a scenario with multiple vehicles on a curved road. The ego-vehicle is traveling at a constant speed of ~33 MPH (~15 m/s), as well as the vehicle in the lane to the right. The vehicle in the lane to the left is traveling ~29MPH (~13 m/s). Both vehicles around the ego-vehicle begin ahead of the ego-vehicle and follow the trajectory shown in *Figure 7.16*. In this scenario, no FCW alert was issued, as expected; however, the system did detect each vehicle when going around each bend, as shown in *Figure 7.17* and *Figure 7.18*. In between the curves, the system did go back to green (no warning), as expected shown in *Figure 7.19*.

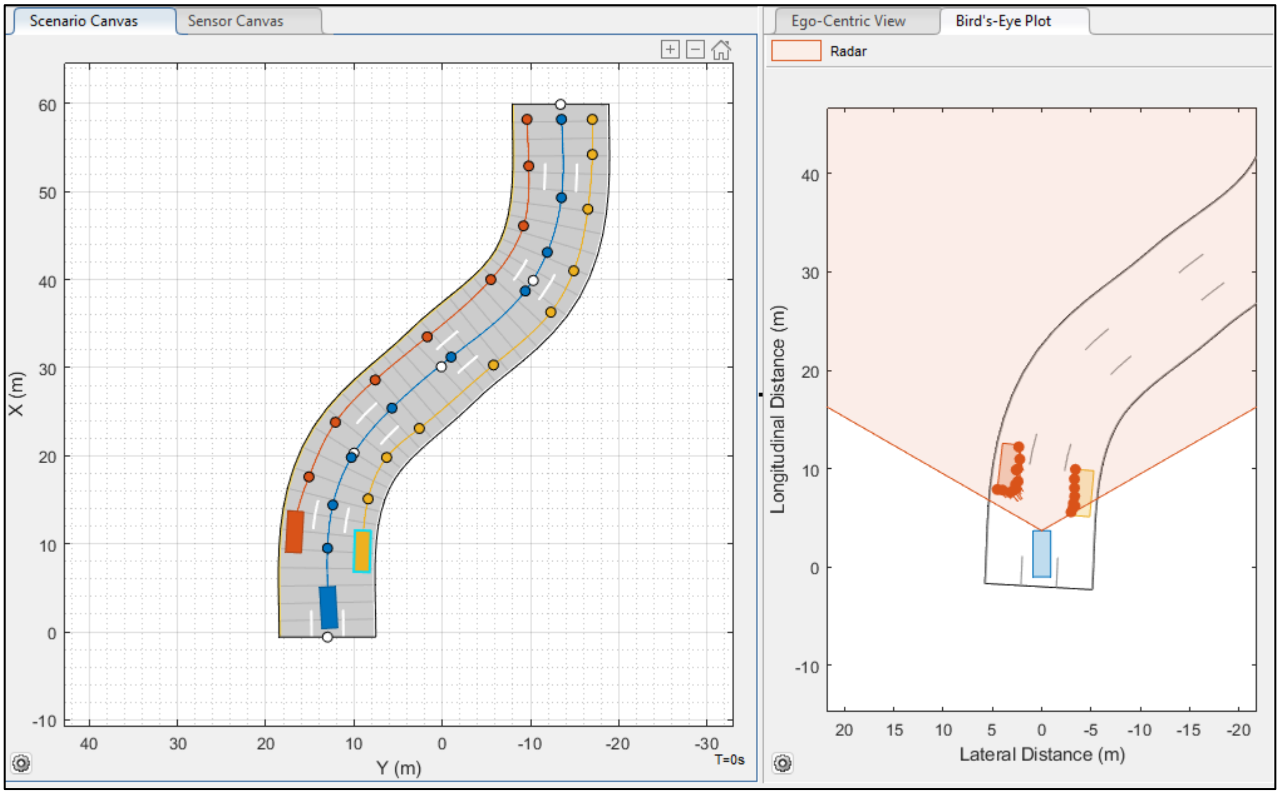

Figure 7.16 - Curved Multi-Lane Scenario

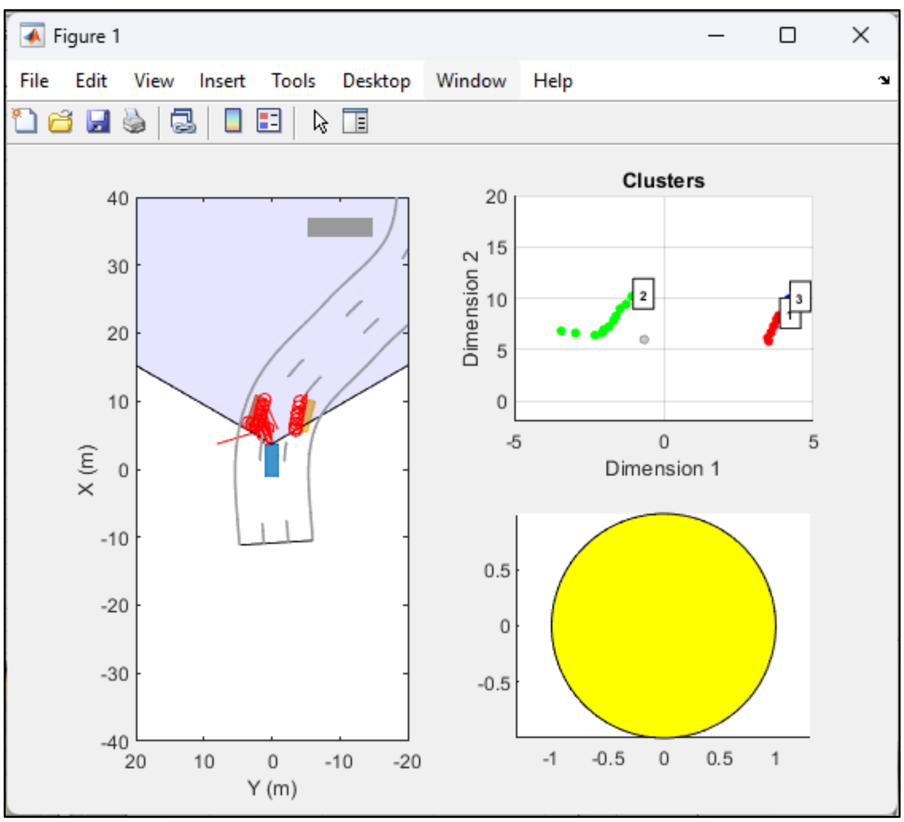

Figure 7.17 - Curved Multi-Lane Scenario, Incorrect Vehicle Detection, First Curve

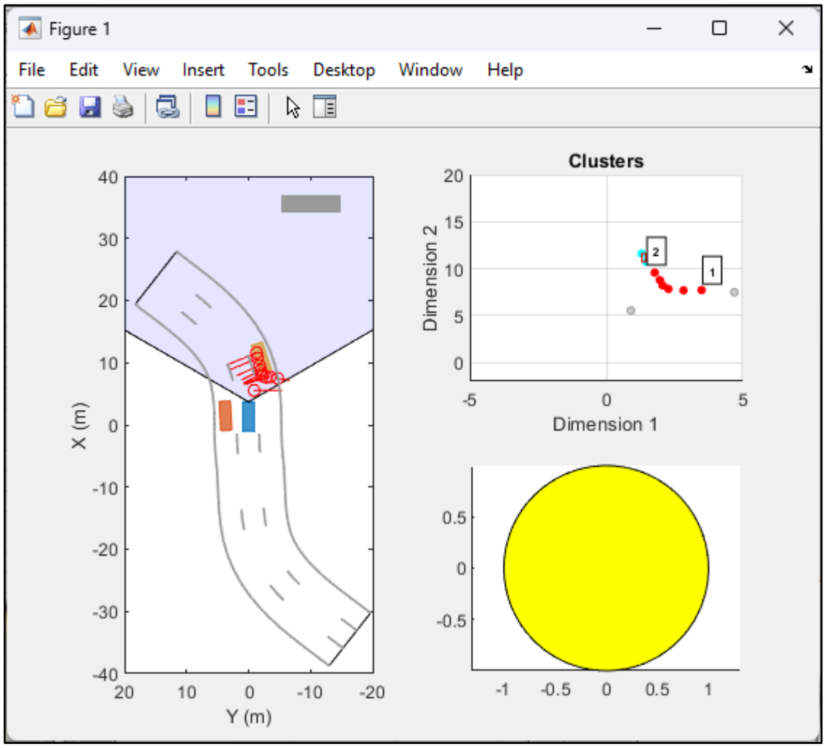

Figure 7.18 - Curved Multi-Lane Scenario, Incorrect Vehicle Detection, Second Curve

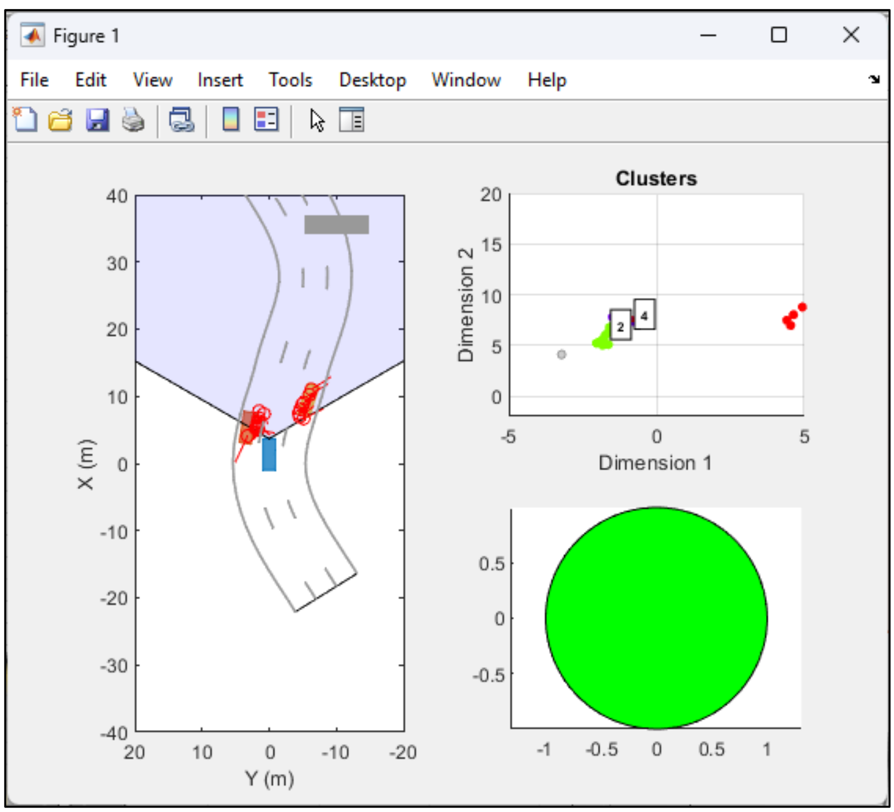

Figure 7.19 - Curved Multi-Lane Scenario, No Alert, Middle of Curve

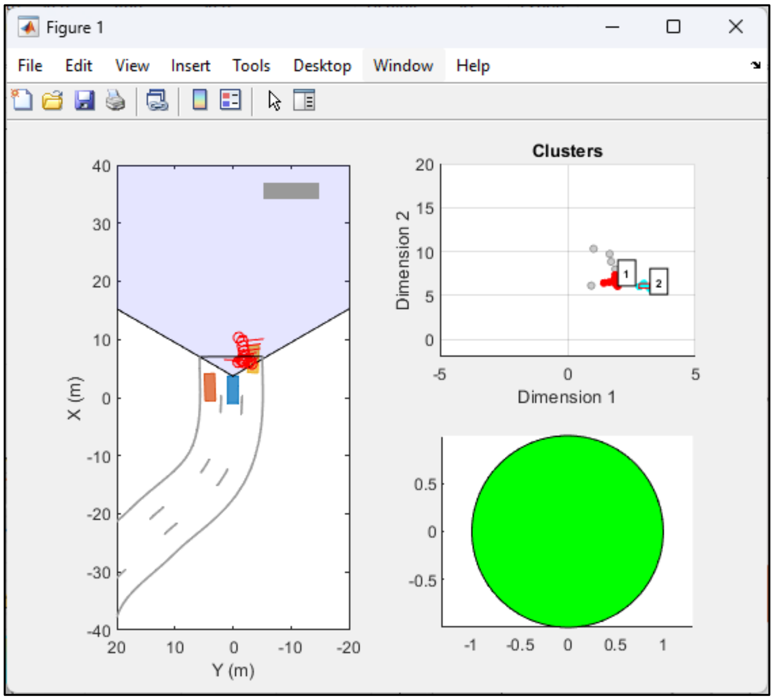

Figure 7.20 - Curved Multi-Lane Scenario, No Alert, End of Scenario

The simulation performed well in this scenario, not issuing a FCW alert despite detecting the vehicles ahead. It is understandable that the system detected the vehicles when going around the curve, as the radar detected it as ‘in front of the vehicle’. A solution to this issue would be to have a camera able to detect the lane lines and have another ‘filter’ to ensure only vehicles inside the ego-vehicle’s lane are detected.

In all of these tests, the system performed as expected when considering the implementation of the system as a whole. It functioned without error in situations where a vehicle is directly in front of the ego-vehicle, however struggled when curves in the road or other objects like barriers or oncoming vehicles were in the scenario. These objects can be filtered out with more sophisticated and advanced data processing methods or with the use of a camera mounted on the vehicle, as many modern vehicles with ADAS have today.

Overall, the results of these simulations are a successful and great beginning for this project. With more time for testing, debugging, and adjusting algorithms, this project can become very successful and perform even better.

7.2 Hardware

As mentioned, this project implemented an example project for object detection using TI's mmWave radar (AWR1843BOOST). Testing the radar simply consisted of gathering data and understanding how the radar functions. The results are discussed below.

7.2.1 Object Detection without Forward Collision Warning

Testing for this project was conducted with the setup shown in *Figure 6.10*. The radar was mounted to the vehicle and connected to a laptop inside the cabin to record data. The data was shown on a visualizer that displays a X-Y scatter plot and Doppler-range plot.

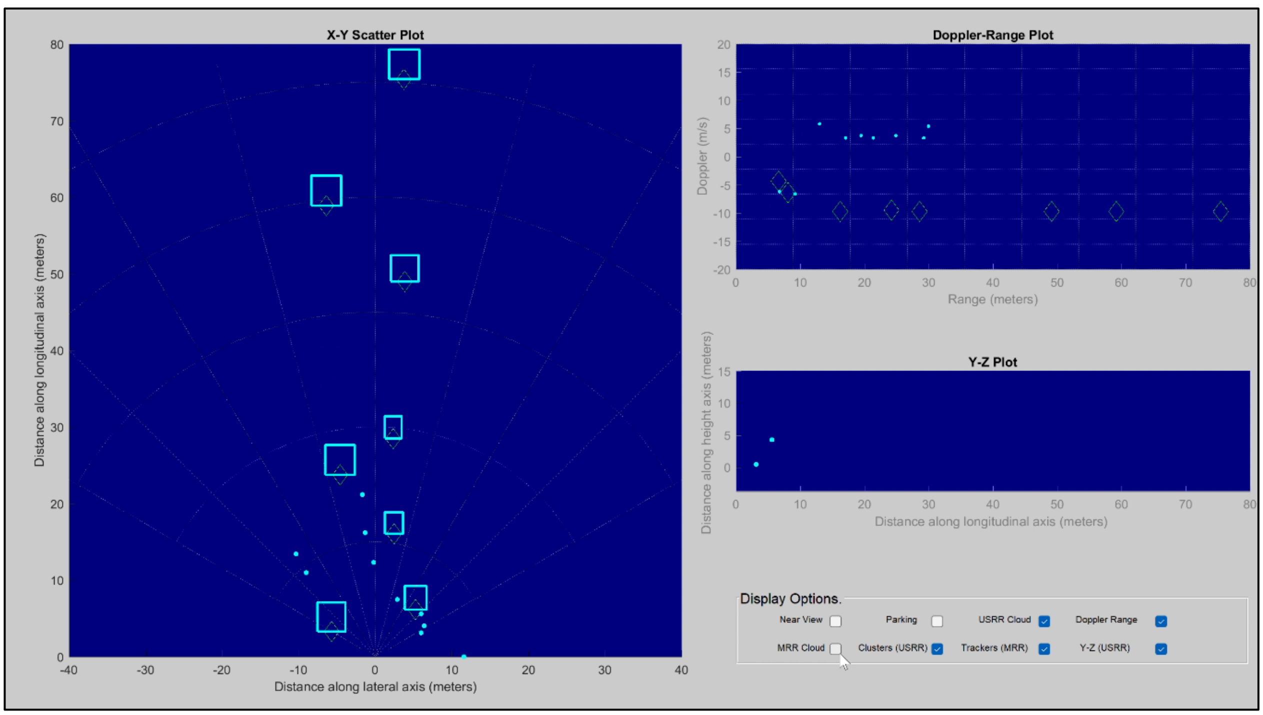

Figure 7.21 - Radar Visualization, No Lead Vehicle, Parked Vehicles on Left and Right

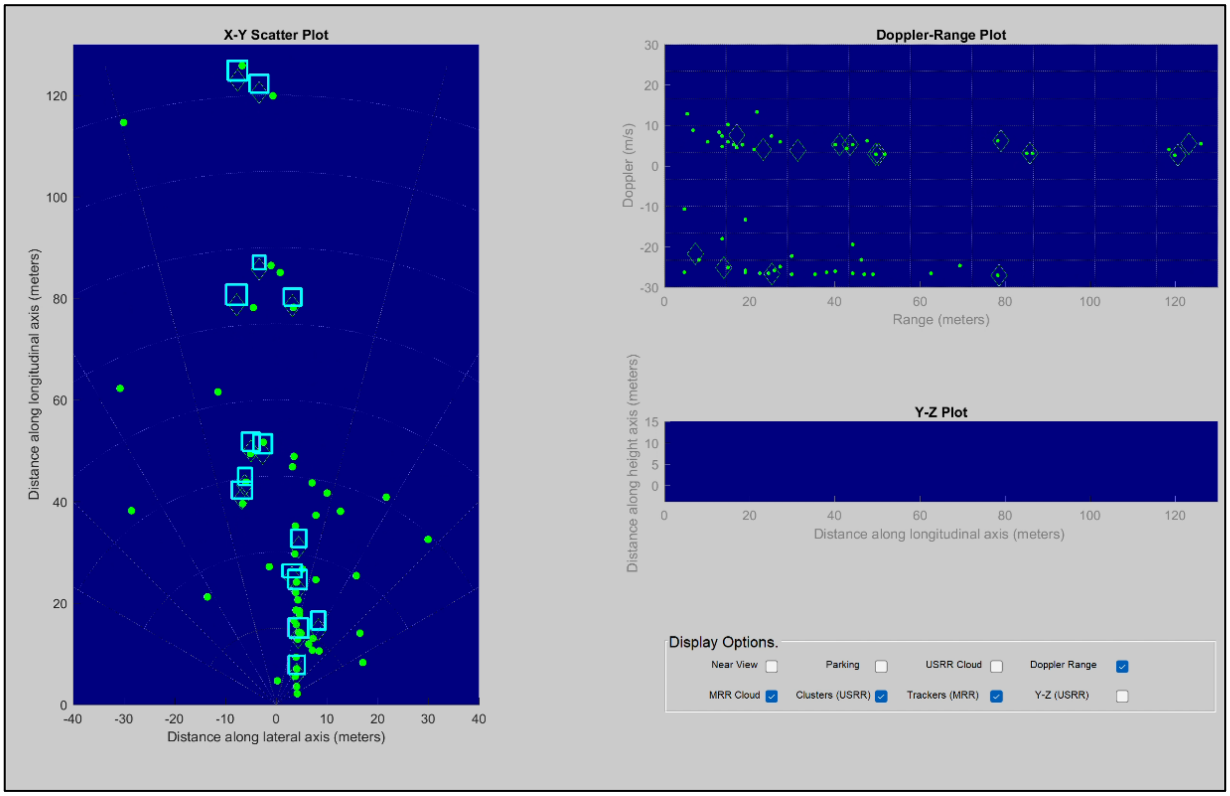

Figure 7.22 - Radar Visualization, Freeway Driving, Barrier on Right, Vehicles Ahead

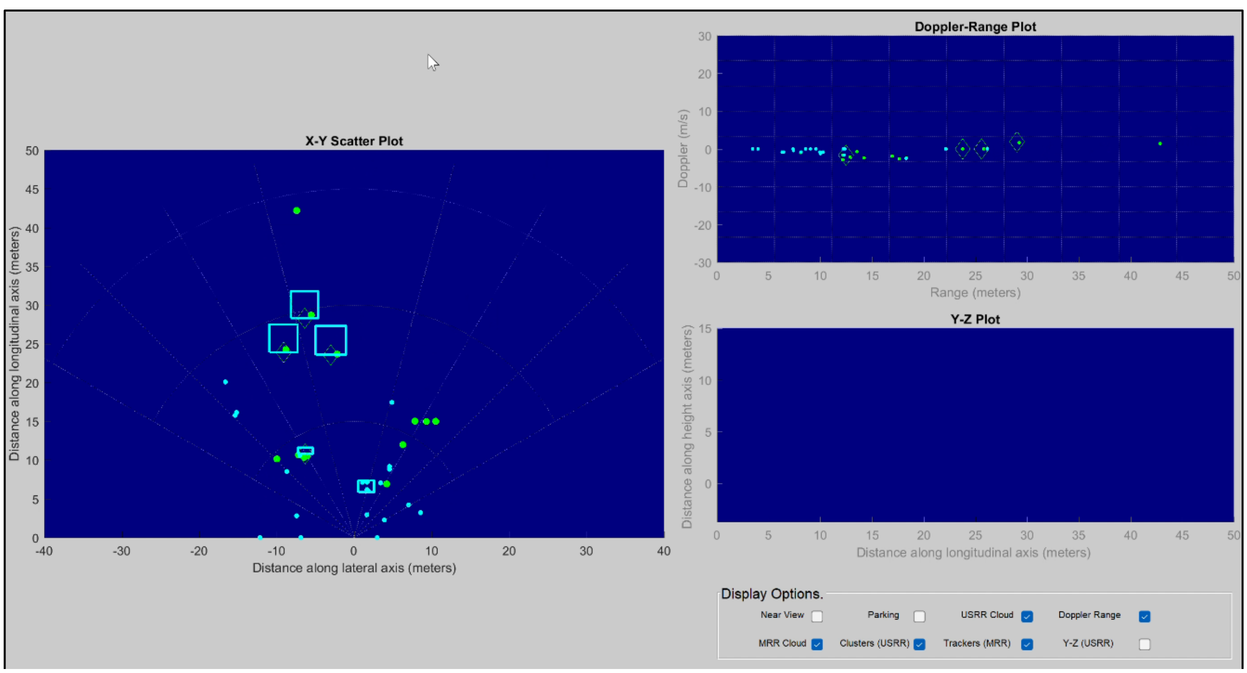

Figure 7.23 - Radar Visualization, Downtown Driving, Stopped Vehicles with Pedestrians Crossing

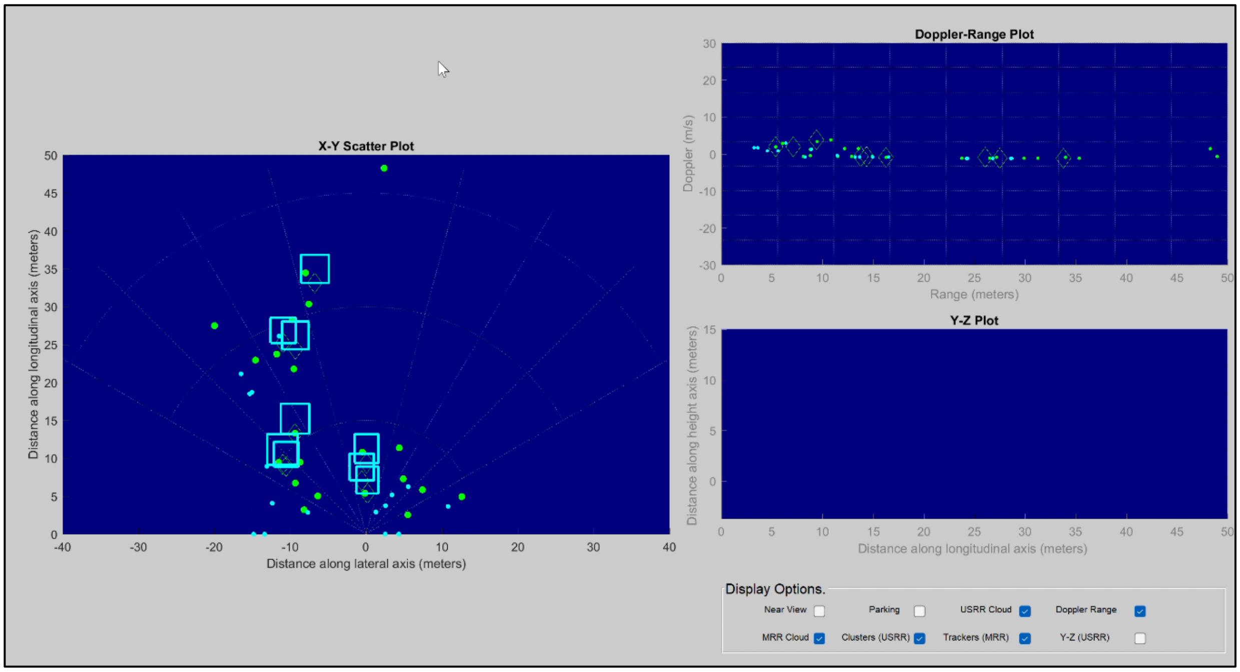

Figure 7.24 - Radar Visualization, Downtown Driving, Multiple Vehicle Tracking

Various situations were captured in the previous figures and provide a good understanding of the radar's sensing capabilities based on the example project. In *Figure 7.21*, the radar captures multiple parked vehicles on both sides of the road and clusters them appropriately. The system also calculates the velocities shown in the upper right Doppler plot very well, showing that they are traveling towards the test vehicle with a negative velocity equal to the test vehicle's speed.

In *Figure 7.22*, we see a great example similar to one of the software simulations. The test vehicle is traveling on the freeway with a barrier on the right side and other vehicles ahead. The barrier is detected, shown by all the green dots near the middle of the screen and separated into multiple clusters. The system displays the velocities as ~ -25 m/s or ~ -56 MPH, equal to the speed of the test vehicle. The system does a great job at capturing these points, however for FCW implementation, they will need to be filtered out and removed. This potentially could be done by setting a limit on the maximum relative velocity to not capture stationary objects when the test vehicle is traveling at freeway speeds.

In both Figure 7.23 and Figure 7.24, driving in downtown San Luis Obispo was conducted. In Figure 7.23, the system is able to detect two pedestrians crossing in front of the vehicle, shown by the very small cluster boxes in the bottom-center of the scatter plot. In Figure 7.24, the system detects multiple cars while stopped at a traffic light, shown by the 3 separate, but overlapping blue squares in the center bottom of the screen.

Testing the radar was again meant to provide a basic understanding of how the system operates, gain insight into what data the radar captures, and the capabilities of the example project. The information was quite useful, giving a great representation of how the system works with stationary objects along the road such as parked cars or barriers, and also demonstrating how it can detect pedestrians crossing in front of the vehicle. All of this information is very useful for testing and development of the project and with more time, the system can be improved regarding filtering and object detection, as well as FCW implementation.

8 Conclusion & Future Work

The Automotive Forward Collision Warning System Retrofit project accomplished a good portion of its goal considering the delays and time constraints faced. Originally, the project was meant to implement a functional and working prototype, however due to unforeseen shipping delays with the radar, time implementing a physical prototype was lost. Fortunately, simulation was able to be achieved using MATLAB's Automated Driving Toolbox and Driving Scenario Designer.

The simulation included a dBScan clustering algorithm for clustering similar datapoints into one object, an Extended Kalman filtering algorithm meant to reduce noise and track similar objects, and a forward collision warning algorithm, providing a distance used to determine a risk of collision. As seen, the system performed very well and achieved its goal of implementing a FCW system. There were more advanced scenarios where the system did not perform as desired; however, this could be solved with more advanced algorithms or through the implementation of a windshield mounted camera that can detect vehicles and fuse that data with the radar data, similar to many modern vehicles today. However, this project's purpose was to only use an affordable radar to implement a FCW system and great, usable progress has been made.

Some conflicts that arose during the software development and simulation testing were related to the filtering ability of the system and the ability to exclude non-moving or non-vehicle objects. This can be related to the lustering algorithm and extended Kalman filter. As seen in some of the simulations, the system clustered multiple objects together that should not have been – such as the barriers or other vehicles. The clustering algorithm works on top of the Kalman filter, therefore adjustments can be made to the Kalman filtering function and work can be done towards developing and implementing a more efficient and functional filtering method.

In regard to the radar and it's physical implementation, a great amount of knowledge and understanding was gained from the testing done. The example project provided real world examples of what can be detected and what will get filtered out, especially regarding freeway barriers and pedestrians. The example project is a great baseline project that can be adjusted and modified in the future to provide a functional prototype.

Overall, the experience provided by this project has exemplified many of the skills learned and developed through Cal Poly's Electrical Engineering program and "Learn By Doing" model. While more development and improvements can be made, the project was an overall success.

9 References

[1] “Effectiveness of front crash prevention systems in reducing large truck real-world crash rates,” *IIHS-HLDI crash testing and highway safety*. https://www.iihs.org/topics/bibliography/ref/2211

[2] R. R. M. updated, “A Brief History of The High-Tech Safety Features In Your Car,” *www.automoblog.net*, 2019. https://www.automoblog.net/brief-history-high-tech-safety-features/

[3] “What are ADAS systems?,” *Fundación MAPFRE*. https://www.fundacionmapfre.org/en/education-outreach/road-safety/adas-systems/what-are-adas-systems/ (accessed Oct. 22, 2022).

[4] T. R. Control, “3-Second Rule for Safe Following Distance,” *www.travelers.com*. https://www.travelers.com/resources/auto/travel/3-second-rule-for-safe-following-distance

[5] “Forward Collision Warning: MyCarDoesWhat.org,” *My Car Does What*. https://mycardoeswhat.org/safety-features/forward-collision-warning/

[6] “ADAS+ Advanced Driver Assistance System ADAS-1100,” *Brandmotion*. https://brandmotion.com/product/adas-advanced-driver-assistance-system-adas-1100/ (accessed Oct. 22, 2022).

[7] “MiniEye - MiniEye Forward collision warning” *www.facebook.com*. https://fb.watch/gi_U898zwU/ (accessed Oct. 22, 2022).

[8] “comma – introducing comma three,” *comma – introducing comma three*. https://comma.ai (accessed Oct. 22, 2022).

[9] F. A. Butt, J. N. Chattha, J. Ahmad, M. U. Zia, M. Rizwan and I. H. Naqvi, "On the Integration of Enabling Wireless Technologies and Sensor Fusion for Next-Generation Connected and Autonomous Vehicles," in IEEE Access, vol. 10, pp. 14643-14668, 2022, doi: 10.1109/ACCESS.2022.3145972.

[10] S. Campbell et al., "Sensor Technology in Autonomous Vehicles : A review," 2018 29th Irish Signals and Systems Conference (ISSC), 2018, pp. 1-4, doi: 10.1109/ISSC.2018.8585340.

[11] C. Brinsfield, “Reverse Engineering and Enhancement of Automobile Infotainment System,” *Electrical Engineering*, Jan. 2022, Accessed: Dec. 02, 2022. [Online]. Available: https://digitalcommons.calpoly.edu/eesp/549

[12] A. Samman et al., "Potential use of near, mid and far infrared laser diodes in automotive LIDAR applications," Vehicular Technology Conference Fall 2000. IEEE VTS Fall

VTC2000. 52nd Vehicular Technology Conference (Cat. No.00CH37152), 2000, pp. 2084-2089 vol.5, doi: 10.1109/VETECF.2000.883239.

[13] M. Hodzic, "Distronic Plus and brake assist plus reduce rear-end collisions by 20%," BenzInsider.com - A Mercedes-Benz Fan Blog, https://www.benzinsider.com/2008/06/distronic-plus-and-brake-assist-plus-reduce-rear-end-collisions-by-20/ (accessed May 28, 2023).

[14] "Insurance data show bigger benefits for young drivers from crash avoidance tech," IIHS, https://www.iihs.org/news/detail/insurance-data-show-bigger-benefits-for-young-drivers-from-crash-avoidance-tech (accessed May 28, 2023).

[15] "Automated driving toolbox," Automated Driving Toolbox Documentation, https://www.mathworks.com/help/driving/index.html?s_tid=CRUX_lftnav (accessed May 29, 2023).

[16] "Driving Scenario Designer," Design driving scenarios, configure sensors, and generate synthetic data - MATLAB, https://www.mathworks.com/help/driving/ref/drivingscenariodesigner-app.html (accessed May 29, 2023).

[17] "User Guide - xWR1843 Evaluation Module (xWR1843BOOST) Single-Chip mmWave Sensing Solution." Texas Instruments

[18] Medium Range Radar Users Guide, https://dev.ti.com/tirex/content/radar_toolbox_1_00_01_07/source/ti/examples/ADAS/medium_range_radar/docs/Medium_Range_Radar_users_guide.html (accessed May 29, 2023).

[19] S. Laohakiat, P. Ratanajaipan, L. Navaravong, R. Ungrangsi and K. Maleewong, "A Fuzzy Density-based Incremental Clustering Algorithm," *2018 15th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, Nakhonpathom, Thailand, 2018, pp. 1-5, doi: 10.1109/JCSSE.2018.8457385.

[20] O. Limwattanapibool and S. Arch-int, "Detecting cluster numbers based on density changes using density-index enhanced Scale-invariant density-based clustering initialization algorithm," *2017 9th International Conference on Information Technology and Electrical Engineering (ICITEE)*, Phuket, Thailand, 2017, pp. 1-5, doi: 10.1109/ICITEED.2017.8250481.

[21] J. Chen and P. S. Yu, "A Domain Adaptive Density Clustering Algorithm for Data With Varying Density Distribution," in *IEEE Transactions on Knowledge and Data Engineering*, vol. 33, no. 6, pp. 2310-2321, 1 June 2021, doi: 10.1109/TKDE.2019.2954133.

[22] Zheng Hua, Wang Zhenxing, Zhang Liancheng and Wang Qian, "Clustering algorithm based on characteristics of density distribution," *2010 2nd International Conference on*

Advanced Computer Control, Shenyang, 2010, pp. 431-435, doi: 10.1109/ICACC.2010.5486640.

[23] A. Hanumegowda, S. Dewangan, S. Bhupala, F. Gruson and D. Steinhauser, "Extended Object Tracking with IMM Filter for Automotive Pre-Crash Safety Applications," *2021 18th European Radar Conference (EuRAD)*, London, United Kingdom, 2022, pp. 177-180, doi: 10.23919/EuRAD50154.2022.9784586.

[24] M. Bersani, M. Vignati, S. Mentasti, S. Arrigoni and F. Cheli, "Vehicle state estimation based on Kalman filters," *2019 AEIT International Conference of Electrical and Electronic Technologies for Automotive (AEIT AUTOMOTIVE)*, Turin, Italy, 2019, pp. 1-6, doi: 10.23919/EETA.2019.8804527.

10 Appendices

10.1 Senior Project Analysis

10.1.1 Summary of Functional Requirements

This project uses MATLAB's Automated Driving Toolbox and Radar Toolbox to implement a clustering and filtering algorithm alongside a forward collision warning algorithm to provide an alert when the distance between two vehicles is unsafe relative to the speed between them. The project also has a hardware aspect using Texas Instruments' Automotive mmWave Radar (AWR1843BOOST) to track and display objects on a computer.

10.1.2 Primary Constraints

The largest constraint of this project was shipping delays which led to time constraints. The AWR1843BOOST radar could not be obtained quickly which led to a delayed start on physical implementation, making the implementation of a physical FCW system not possible in the time provided. Besides this constraint, the other main constraint was knowledge regarding radar technology. Going into this project, not much was known regarding radar technology, object detection, object filtering, and radar data processing. This caused a time constraint with learning about this methodology, as well as debugging and updating the software algorithms. Each iteration of software required an intensive and lengthy debugging process.

10.1.3 Economic

The direct cost of this project is estimated to be approximately $430, the cost of the mmWave radar ($410) and the cost of the mounting hardware ($20) – funded personally. No other direct costs were incurred with this project as everything else was available. In addition to these direct costs, there were indirect costs of labor (research and development), development equipment (laptop, software programs), and test equipment (vehicle). There is no expected return from this project, however if implemented commercially, cost benefits would arise due to potential reduction in vehicle accidents. Once installed, there is no maintenance required or cost to the user.

- Human Capital

    - Development of this project required time for development and implementation as well as the knowledge of programming and radar systems as well as the ability to research various topics related to these fields.
- Financial Capital
    - This project required the use of high cost items such as a vehicle (2012 Nissan Altima, $8k, used) and a computer (Dell Inspiron, $1.5k) to develop and test.
- Manufactured or Real Capital
    - There are labor costs associated with the software design and debugging process. There is also labor costs associated with research of various techniques or algorithms.
- Natural Capital
    - There are a few direct natural capital costs associated with this project. The testing performed was done using a gasoline powered vehicle which can cost the environment due to emissions. The radar was manufactured containing materials such as silicon, copper, nickel, and similar resources.

10.1.4 If Manufactured on a Commercial Basis

While there are currently no plans to manufacture this on a commercial basis, there could be a reduced cost if done. The radar used for this project (AWR1843BOOST) is a development board with various features that can be removed after development and testing. The board itself costs approximately $410, however, the radar sensor and processor can be purchased separately for approximately $30. This lower cost combined with custom PCB fabrication can significantly reduce the cost of this project if manufactured commercially. After the design and testing has been completed, this project could be manufactured greatly on a commercial scale and thousands of units could be manufactured yearly (considering adequate supply of radar chips and PCBs). The profit on this product could be substantial, considering similar products sell upwards of $200. The cost to the user after purchasing and installing is nothing.

10.1.5 Environmental

There were no significant environmental impacts from this project. The main impact was the fuel used during the testing process when operating the vehicle. Another environmental impact that can be considered is the electricity usage of the development and testing equipment (laptop, monitors). The radar itself typically uses about 2W with 5W being the maximum, with the source being the vehicle, which is powered by gasoline, therefore contributing to emissions.

10.1.6 Manufacturability

The main issue regarding manufacturing was supply chain issues when trying to obtain the radar. Electrical chip component shortages combined with low stock of the radar led to long lead and shipping times. The only physical manufacturing performed on the project was creating a bracket for the radar to connect to the GoPro suction mount which was relatively easy to implement using an aluminum bar and pliers to bend.

10.1.7 Sustainability

Once implemented to the vehicle, the user should be able to use the product without any upkeep or maintenance. As a retrofit project, this modernizes older vehicles, bringing a new aspect to them that previously did not exist, promoting sustainability. Rather than purchasing a new vehicle, this project allows users to retrofit their existing vehicle, rather than sending it to a junkyard. This project also can be implemented using minimal components during the manufacturing process, reducing e-waste and unnecessary manufacturing.

10.1.8 Ethical

Advanced Driver Assistance Systems are under scrutiny today. Misuse of these systems have occurred before and are not necessarily preventable. People using this product can over rely on it while driving. Drivers can get distracted and assume that the system will work which can cause legal issues if the product does not work when assumed it should.

10.1.9 Health and Safety

The main safety concern associated with this project is if the system does not work when it is assumed that it should. This could occur if the system does not detect a vehicle or object and does not issue a forward collision warning, leading to an accident. Regarding the installation process, an inexperienced user may face safety issues when working with tools or installing to their vehicle. Regarding the manufacturing process, there are no major concerns.

10.1.10 Social and Political

This product is meant for road usage and would have to get the required certifications or approvals to be used. This can cause political impact in regard to whether or not the product should be allowed to operate on public roads. This can impact lawmakers as well as the general public due to the safety or legal concern regarding this product.

10.1.11 Development

Various different tools were learned and used during the development of this project. Most of them relate to programming in MATLAB – prior to development, there was only limited experience with MATLAB. During development, the ability to use MATLAB was expanded greatly. Along with MATLAB's basic functionality, the Automated Driving Toolbox and Radar Toolbox were learned to develop scenarios, radar data, clustering and filtering algorithms, and more. Texas Instruments' Code Composer Studio IDE was also learned and used during this project for implementation on the radar.

10.2 GitHub Repository

The below GitHub repository contains all code related to software development in MATLAB and the example project provided by Texas Instruments.
https://github.com/enajmy/FCW-System-Retrofit-Senior-Project