APRIL 25, 2024

DECISION MAKING IN AN ETHICALLY CHALLENGING SITUATION

ATHARVA AJAY WANI
ASUID: 1224422355
Project Supervisor: Dr. Abdel Raouf Mayyas

Index

Introduction:

In recent years, the landscape of transportation has undergone a transformative shift, marked by the exponential rise in the number of autonomous driving vehicles navigating our roads. Notably, industry leaders like Tesla have reported a staggering 1.8 million vehicles sold in 2023, underscoring the accelerating adoption of autonomous technology [2]. Furthermore, pioneers in autonomous driving, such as Waymo, have achieved a milestone by completing over 7 million fully autonomously driven miles [3]. While these advancements herald a promising future, an alarming trend has emerged in tandem – a steady increase in accidents involving autonomous vehicles.

The National Highway Traffic Safety Administration (NHTSA) reported thousands of crashes in 2023 involving autonomous vehicles, raising critical questions about their safety and response mechanisms [1]. While some of these incidents were triggered by abrupt movements from other drivers or unforeseen obstacles, a considerable number were the result of rare but high-impact scenarios, leading to multi-car accidents with the potential for fatal injuries. The escalating frequency of these incidents necessitates a thorough analysis of how autonomous systems respond to situations that, although rare, carry a heightened risk of causing severe accidents.

This project endeavors to address this critical concern by presenting a comprehensive analysis of one such rare and high-risk scenario. By delving into the intricacies of these infrequent yet impactful events, we aim to shed light on the vulnerabilities of current autonomous driving systems. Furthermore, this report proposes a method designed to minimize the severity of accidents in such situations, contributing to the ongoing discourse on enhancing the safety and reliability of autonomous vehicles.

Drawing on a synthesis of real-world data, simulation results, and relevant literature, this project aims to offer valuable insights into the intricacies of autonomous driving under challenging conditions. The findings are anticipated to inform the ongoing development of autonomous

systems, fostering a safer and more robust future for the integration of self-driving technology on our roads.

Literature Review:

In the dynamic landscape of autonomous vehicle development, ensuring safe and efficient navigation is paramount. Over the years, numerous strategies have been proposed to address the critical challenges associated with maintaining safe distances between vehicles and averting collisions. As autonomous driving technology advances, the intricate interplay of sensors, algorithms, and real-time decision-making becomes increasingly vital. This literature review delves into the wealth of research dedicated to enhancing the safety protocols of autonomous vehicles. By exploring the diverse array of strategies proposed, this review seeks to provide a comprehensive understanding of the current state of knowledge in the field. Unraveling the complexities inherent in autonomous vehicle safety not only illuminates the progress made but also underscores the persistent gaps and challenges that necessitate further investigation. Through a systematic exploration of existing literature, this review aims to contribute to the ongoing discourse on optimizing the safety mechanisms crucial for the widespread adoption of autonomous vehicles in real-world scenarios.

In their paper, Mohammad K et al. delve into the augmentation of Responsibility-Sensitive Safety (RSS) rules, which intricately define a set of safety principles designed to preclude vehicular accidents irrespective of other vehicles' behaviors. A pivotal addition to this framework is the introduction of "conflict zones," a concept leveraging the anticipated trajectories of surrounding vehicles relative to the ego vehicle. These zones identify segments of the ego vehicle's trajectory that may intersect with the path of a target vehicle, signifying a potential collision risk. The motion planner, crucial to this system, computes the maximum safe velocity for the ego vehicle by judiciously considering these conflict zones.

The paper elucidates three distinct scenarios wherein these rules find practical application: 1) Same Lane Following, 2) Intersection Navigation, and 3) Merging. Specifically tailored for

Connected Automated Vehicles (CAVs), the proposed algorithm relies on the periodic broadcast of critical information—vehicle ID, position, velocity, timestamp, and future path. The motion planning of the ego vehicle is intricately linked to these broadcasted factors. Furthermore, the algorithm exhibits a capability to predict deadlock situations based on the shared information and resolves such scenarios by adjusting the ego vehicle's speed. However, the paper notes that the resolution of deadlock situations may take some time, emphasizing the inherent limitations of instantaneous velocity changes for vehicles.

The experimental setup introduces a degree of randomness by initializing the positions, destinations, and velocities of CAVs on a map generated from OpenStreetMap. In comparison to autonomous driving vehicles, the algorithm empowers CAVs to navigate through the scenarios with heightened efficiency. Notably, the average velocity for vehicles during the experiment hovers around 11 m/s. However, it is imperative to note that the experiment is conducted within specific constraints, as it does not account for potential faults such as network delays, communication breakdowns among CAVs, or inaccuracies in the data streamed from CAVs. Additionally, high-speed emergency situations and scenarios involving high-speed highway driving are not considered in this experimental framework. [4]

In their work, Hong Wang et al. present a novel approach to crash mitigation within the domain of motion planning for autonomous vehicles. The proposed methodology incorporates the Potential Crash Severity Index (PCSI) and the utilization of Artificial Potential Field (APF) to characterize the nature of obstacles encountered during the motion planning process. The PCSI encapsulates the Relative Speed Factor, Relative Heading Angle Factor, and Mass Ratio Factor.

The Relative Speed Factor quantifies the disparity in velocities between the ego vehicle and obstacles, normalized by the distance between them [5]. Relative Heading Angle Factor categorizes potential collisions into three segments: Full Overlap, 2/3 Overlap, and 1/3 Overlap. Drawing from empirical crash data of Volvo vehicles traveling at speeds exceeding 20 mph, the study establishes that 1/3 Overlap Collisions exhibit 2 to 3 times higher severity than their full-frontal counterparts due to the presence of protective structures in the frontal region [6]. Furthermore, the severity hierarchy is elucidated based on vehicle types, highlighting that a collision between two passenger cars results in less severity compared to a collision involving a

passenger car and a truck. This observation aligns with data from the Insurance Institute of Highway Safety (IIHS), indicating that a full-frontal crash between a passenger car and a light truck is 3-4 times more severe, escalating to 27-48 times in the case of side-impact collisions [7]. The comprehensive PCSI is subsequently derived by summing these constituent factors.

The APF is generated by detecting and classifying obstacles into three distinct types: Crossable, Non-crossable, and Road Boundary. The aggregate APF is constructed by summing the individual potential fields generated from all encountered obstacles. This information is then leveraged to adopt an optimal strategy aimed at minimizing crash severity. For instance, in scenarios where avoiding an obstacle might lead to a partial overlap collision, the algorithm strategically opts for a full-frontal collision, maximizing crash protection.

The study explores diverse scenarios, encompassing a spectrum of speeds ranging from 10 m/s to 27 m/s. Demonstrating the efficacy of the crash mitigation strategies, the proposed framework successfully avoids collisions where feasible. In instances where collision is unavoidable, the system selects paths minimizing crash severity.

Michael Wolf and Joel Burdick delve deeper into the use of Artificial Potential Functions for Highway Driving with Collision Avoidance. They discuss the challenges of rule-based navigation algorithms due to the complexities of driving and propose the use of artificial potential fields for local trajectory generation as an elegant alternative. The method involves modelling vehicles as charged bodies in an electric field, with a potential function driving them towards their goal while avoiding obstacles. The unique characteristics of highway driving, such as lane preferences, limited direction of travel, and varying acceptable tolerances for obstacle proximity, are highlighted.

The paper presents a potential function for highway driving that prioritizes avoiding obstacles and road edges while favoring travel in lane centers at desired speeds. This approach can be applied to autonomous vehicle navigation or driver assistance systems. The desired behavior for highway vehicles is outlined, including maintaining a target speed and staying in the center of the lane, while avoiding collisions and leaving the road. The proposed approach assumes knowledge of vehicle states, roadway layout, and nearby vehicles' positions and velocities. It describes

potential functions for lane-keeping, preventing departure from the roadway, collision avoidance, and targeting desired speeds. The total potential function is defined as the sum of these components.

The lane potential creates a barrier to lane changing and guides the car to the center of the lane, but it is small enough that in case lane change is required, the vehicle can cross it. The road potential prevents the vehicle from going off the road by becoming infinite at the road edges. The car potential is used for keeping the vehicle a safe distance from the obstacle vehicle. The car potential is categorized into two: 1) Potential for forward and side of obstacle, 2) Potential for Obstacle. The velocity potential function generates a linear feedback law in the x-direction to guide the vehicle towards a desired speed. When the vehicle's velocity is below the desired speed, it produces a forward-propelling force, while exceeding the desired speed results in a retarding force.

The simulations conducted on a multi-lane highway using the gradient-descent law with the described potential function components provided insightful results. Initial testing without obstacles confirmed the vehicle's ability to center itself in its lane, maintain road adherence, and sustain the desired speed, with minor oscillations. The addition of obstacles, represented by other vehicles, effectively prevented lane changes in certain directions without significantly deviating the controlled vehicle from its lane center.

In Scenario 1, where the leading obstacle car travelled near the desired velocity, the controlled vehicle reduced speed and settled into a local minimum behind the obstacle. This showcased the potential field's adaptability to varying obstacle speeds, with the controlled vehicle reacting accordingly.

Scenario 2 demonstrated the system's ability to recognize slower-moving obstacles and execute lane changes to avoid them. When the leading obstacle's speed fell below a certain threshold, indicating a significant decrease in speed, the controlled vehicle changed lanes to maintain its desired speed.

Scenario 3 simulated heavier traffic conditions with multiple obstacles in adjacent lanes and a leading obstacle traveling well below the desired speed. In this scenario, the controlled vehicle stayed in its lane, slowing down to match the speed of the leading obstacle, highlighting the potential field's capability to handle complex traffic situations. [8].

Chuanyang Sun and Azim Eskandarian in their paper propose a collision mitigation system that incorporates the predicted trajectory of the obstacle vehicle and candidate trajectories of the ego vehicle to identify impact location and angle. A novel method evaluates accident severity and selects the best trajectory based on a cost function. The study focuses on frontal and oblique collision mitigation in a simplified two-vehicle scenario, representing a common and high-severity type of collision. This scenario involves a two-lane freeway where an obstacle vehicle suddenly enters the ego vehicle's lane with no space for evasive action, thus highlighting the importance of semi-frontal upcoming crash scenarios.

In the proposed approach, Gaussian Processes (GPs) are utilized to predict vehicle trajectories by modelling them as Gaussian probability distributions. Trajectories are represented as sequences of 2D points, with instantaneous velocity as the target variable. The GP model is characterized by mean and covariance functions, with hyperparameters obtained from training datasets collected from simulation using a double-track vehicle dynamic model.

After training, maneuver recognition is performed by comparing the likelihood of successive observations for each regression model. Maneuvers are categorized as either "stay-in-the-lane" or "rush-out-the-lane."

For long-term prediction, an unscented Kalman filter (UKF) is integrated with the GP model to transform the prediction problem into an estimation problem. This combination allows for efficient evaluation of iterated integrals online, thereby facilitating long-term trajectory prediction without significant computational burden.

In trajectory generation, a model-based local search planner is employed to generate candidate trajectories, incorporating the vehicle kinematic model and cubic spline for smooth path generation. The length of candidate trajectories is determined based on the current speed of the ego vehicle and the predicted obstacle trajectory. Collision-checking is performed using a hierarchical structure comprising several methods for efficiency and accuracy:

1. Intersection checking determines if there is a geometric intersection between generated ego trajectories and predicted obstacle trajectories.

2. Bounding box overlap checking constructs conservative bounding boxes along trajectories and checks for overlap to detect collisions.

3. Circle representation intersection checking involves redundant bounding circles for vehicles, considering time and searching for intersections step by step along trajectories.

4. Polygonal representation intersection checking utilizes accurate polygonal representations of vehicles, with a precise intersection check.

5. Impact location identification identifies impact locations based on polygonal representations, considering various relative positions between vehicles.

After trajectory generation, collision-checking is executed sequentially for each trajectory, involving steps such as initializing parameters, executing intersection and overlap checks, and identifying impact locations if necessary. Speed re-planning is conducted if overlaps are found, using maximum deceleration allowed by vehicle dynamics.

After collision-checking, trajectory selection proceeds by evaluating a cost function to rank candidate trajectories. The selection criteria focus on mitigating the severity of collisions and ensuring safety buffers:

1. Cost Function for Accident Severity: The severity of potential collisions is assessed using Δv, which accounts for factors like collision angle and speed. Impact location is also considered,

with weights assigned based on the proximity to vulnerable areas such as the left side of the vehicle. The cost function for accident severity combines Δv and impact location weights.

2. Cost Function for Safety Buffer: Safety buffers in both space and time dimensions are evaluated to ensure collision-free trajectories. The lateral offset between collision-free and closest collision trajectories measures space buffer, while the time difference between ego and obstacle vehicles passing through the collision point measures time buffer. Individual cost functions are formulated for space and time dimensions, respectively.

3. Total Cost Function for Trajectory Selection: The synthesis cost function selects the optimal trajectory considering both accident severity and safety buffer. If there is no collision-free trajectory, the trajectory with the least accident severity cost is chosen. If there is only one collision-free trajectory, it becomes the target trajectory. If multiple collision-free trajectories exist, the one with the best safety buffer cost is selected.

The proposed collision mitigation system was evaluated through numerical simulations conducted in MATLAB. Two scenarios were examined, both involving an obstacle vehicle rushing out from the opposite lane due to a front-left tire blow-out.

In the first case, where the obstacle vehicle's initial speed was set to 20 m/s, two collision-free candidate trajectories were identified. Safest trajectory was selected based on safety buffer evaluations to avoid collision with minimal lateral offset and time difference and collision was avoided.

In the second case, with the obstacle vehicle's initial speed set to 15 m/s, all candidate trajectories resulted in collisions. The speed re-planning module was then executed to mitigate collision risk. Re-planned trajectories showed emergency deceleration, resulting in shorter trajectories but still no collision-free options. The safest trajectory was chosen based on safety buffer evaluations to prevent collision, with minimal lateral offset and time difference and the collision was averted. [9]

In their paper titled "Model Predictive Path-Planning Controller with Potential Function for Emergency Collision Avoidance on Highway Driving", Pengfei Lin and Manabu Tsukada establish that Potential functions (PFs) have emerged as prominent planning algorithms for handling emergency scenarios, offering real-time path generation by modelling 3D risk fields. PFs are being integrated into self-driving vehicles for their fast computation and ability to establish repulsive and attractive potential fields for obstacles and goal points, respectively. However, existing studies have limitations, particularly in addressing emergency avoidance due to unexpected deceleration of obstacle vehicles. To address this, a novel model predictive path-planning controller (MPPC) with a built-in safe passage is proposed, aiming to handle situations where PFs fail to plan a safe and smooth path during sudden decelerations of obstacle vehicles. The proposed method, simulated and compared with the latest adaptive PF-based MPC (PF-MPC) approach, demonstrates superior performance in eliminating severe oscillations when passing through narrow passages in emergency scenarios.

The paper then defines the PFs which are similar to the definitions mentioned in the previously reviewed papers mentioned in the literature review. The paper further goes on to define the Model predictive controller. The process begins with the modelling of vehicle dynamics, incorporating lateral and yaw dynamics using a discrete-time linear model. Future predictions of model outputs are then derived, enabling the anticipation of the vehicle's state and control inputs over a specified time horizon. Constraints analysis is conducted to ensure the feasibility and safety of the control inputs, considering both motion states and control increments.

To overcome limitations in normal driving scenarios, such as sudden obstacles or emergencies, a safe passage (SP) is introduced. The SP is designed using a sigmoid-like function, which dynamically adjusts lateral position constraints based on the positions of the ego and obstacle vehicles. This ensures smoother path planning and obstacle avoidance, especially in emergency situations where potential field-based planning may fail due to sharp increases in obstacle potential fields.

The optimization design integrates vehicle dynamics and SP constraints into a quadratic programming problem. The objective function minimizes path-tracking error and control increment while adhering to motion state and SP constraints. A concise algorithm is proposed to trigger SP constraints in emergency scenarios, monitoring potential field gradients and obstacle overlaps to ensure safe vehicle operation.

The simulation results demonstrate the effectiveness of the proposed SP-based model predictive path-planning controller (PF-SPMPC). The simulations present two distinct scenarios: two-lane traffic and three-lane traffic resulting in a comparative study between PF-SPMC and PF-MPC.

In the two-lane traffic scenario, the PF-SPMPC successfully navigated around static and suddenly decelerating dynamic obstacles without exhibiting prominent oscillations in the vehicle trajectory. Conversely, the PF-MPC showed significant oscillations and exceeded empirical constraints in motion states such as sideslip angle and yaw angle. The SP constraint effectively mitigated these issues, ensuring smoother and more controlled vehicle movement.

In the three-lane traffic scenario, both controllers successfully navigated through closely spaced static obstacles. However, the PF-SPMPC outperformed the PF-MPC by avoiding a suddenly lane-changed obstacle and maintaining control within predefined constraints. Excessive oscillations observed in the PF-MPC were absent in the PF-SPMPC, further highlighting its superior performance.

Motivation:

Autonomous vehicles (AVs) have the potential to revolutionize transportation by reducing traffic accidents. However, a major challenge lies in their decision-making capabilities during unavoidable crash scenarios, especially at high speeds. Upon researching papers and studying the scenarios it became clear that all of them studied scenarios ranging 40 mph to 65 mph. While it is imperative that drivers and autonomous vehicles abide to the speed limits, it is common for

vehicles to be travelling at upwards of 80 mph. While this difference in speed may seem small, under emergency braking a vehicle in ideal conditions needs extra 60 feet to come to a stop. This is equivalent to almost the length of two school busses.

According to the National Highway Traffic Safety Administration (NHTSA) [10], in 2022, there were over 6.7 million traffic accidents reported in the United States alone. These accidents resulted in over 38,000 fatalities and millions of injuries.

While human error is a significant factor in most accidents, even the most skilled driver can struggle to react effectively at high speeds with limited escape routes. However, AVs have the potential to analyse situations faster and with greater precision than humans. By equipping them with robust decision-making algorithms for unavoidable crashes, we can potentially minimize the severity of accidents and save lives.

This project aims to explore the ethical and practical considerations involved in programming AVs for such high-speed, unavoidable crash scenarios. By analysing accident statistics and injury data, we can develop algorithms that prioritize minimizing overall casualties and injuries. This research can contribute to the advancement of AV safety protocols, ultimately leading to safer roads for all.

One of the primary ethical considerations surrounding AV decision-making in unavoidable crash scenarios is the issue of prioritization. AVs must be programmed to make split-second decisions that prioritize minimizing harm, which raises questions about how to weigh the value of different lives and the moral implications of such decisions. Balancing utilitarian principles, which aim to minimize overall harm, with individual rights and fairness is a critical ethical dilemma in AV programming.

Despite these challenges, AVs offer substantial benefits in reducing overall traffic accidents compared to human drivers. According to the NHTSA, human error is a significant factor in most traffic accidents, highlighting the potential for AVs, with their faster reaction times and advanced

sensing capabilities, to prevent accidents caused by human mistakes. Additionally, AVs have the potential to significantly reduce the number of fatalities and injuries on the road, as evidenced by the millions of accidents and thousands of fatalities reported annually.

Problem Scenario:

After reviewing papers attempting to address similar high-speed emergency situations, it was observed in most papers they primarily focus on scenarios where vehicles adhere to speed limits. However, given that many vehicles exceed these limits, and considering the additional stopping distance required at higher speeds (an extra 60 – 70 feet in ideal conditions), it becomes imperative to investigate how autonomous vehicle (AV) systems react and can be optimized to reduce collision damages.

The core objectives of this project are firstly, to design a responsive system capable of reacting to the sudden appearance of a non-crossable obstacle in the path of the ego vehicle, and secondly, to determine the necessity of adjusting mitigation strategies based on the surrounding vehicles' configurations.

The chosen scenario presents the ego vehicle navigating a straight three-lane highway with shoulders on both sides, where each lane measures 3.7 meters in width. The scenario involves a total of nine vehicles, including the ego vehicle. The ego vehicle is in the middle lane flanked by passenger cars in both adjacent lanes. Ahead of the ego vehicle, a row of three vehicles precedes, with a pickup truck carrying a heavy load in its bed directly in front. At a random point in time, this load falls out of the truck, serving as the obstacle the ego vehicle must avoid. To introduce realism, the load continues moving in the same direction for a few meters before coming to a halt. The distance it travels is determined by the formula: distance travelled = [initial speed / 3] meters.

In the row behind the ego vehicle, the two adjacent lanes are occupied by passenger cars, and a large semi-truck is positioned directly behind the ego vehicle. All vehicles in the scenario are traveling at a velocity of 36 meters per second (80 miles per hour). The gap between each vehicle is approximately 39 meters, as recommended by the highway code for vehicles traveling at 45

miles per hour [11]. Given the higher speeds of vehicles, this scenario represents one of the worst cases possible. To stop completely from 80 miles per hour, the vehicle takes about 94 meters. Since the vehicle operates autonomously, it reacts instantaneously upon detecting the obstacle. An assumption is made that the reaction time is negligible.

Proposed Approach:

To focus on the first objective of the problem statement, an accident mitigation system needs to be created which can 1) perceive its surroundings and detect obstacles, 2) drive autonomously at a set speed, and 3) mitigate collision.

Once the system is running, some basic scenarios will be tested to see if the system works as intended. During these tests, the problem scenario is also simulated to see how the system reacts when subjected to higher speeds, closer than recommended following distance, and sudden emergencies.

By the problem definition, the collision is supposed to be unavoidable. A strategy must be formulated to reduce the amount of harm done. Since the vehicle following directly behind the ego vehicle is a semi-truck, it will be carrying a lot of momentum, and this could cause fatal injuries if the semi-truck crashes into the ego vehicle after the ego vehicle crashes into the obstacle.

In the literature review section, it was established that when a collision is unavoidable, it is safer for the vehicle to choose a head on collision than a collision at one of the corners as there is more crash protection in the front of the vehicle. Although this may not be the best option if it means getting hit by a semi-truck in a secondary collision. The formulated new mitigation strategy would need to take this into account.

Once these steps are completed a comparison can be made between the outcomes of an accident featuring this scenario when the mitigation strategy is left unchanged versus the outcome with the mitigation strategy changed.

For this experiment MATLAB-Simulink is used as the program for developing the system and simulation. Mainly the standard Simulink components and components from the Automated Driving Toolbox and Model Predictive Control toolbox by MATLAB are used. Automated Driving Toolbox examples: Highway Lane Change Planner and Automated Emergency Braking are used to create the base system by modifying elements of each example and creating a new system which is an integration of both the examples to achieve all the required functionalities for this project. The following sections elaborate each step in designing the setup for the experiment followed by the Findings, Results, and Conclusions.

Setup:

Building the scenario:

Once the scenario has been defined and the problem statement defined, the scenario builder from the autonomous driving toolbox is used to generate the visualization for the scenario. It provides an easy interface for creating complex scenario visualization as well as defining the positions and properties of sensors used in such as camera, radar, ultrasonic sensor etc. It provides a seamless interface with Simulink to provide real-time visualization of the control inputs being made to the ego vehicle.

First the road network is defined with three main driving lanes of 3.7 meters each which is the standard width of highway lanes in the US. The highway has one shoulder on each side of 3.7 meters each. The total length of the road created is 500 meters as this should be sufficient space to simulate the problem.

The ego vehicle and its properties are defined next. The length of the vehicle is set to be 4.7 meters and a width of 1.8 meters. It is placed in the middle lane and the rest of the actors in the scenario are placed around it. Sensors for perception are added onto the ego vehicle as well. One camera is placed on the roof of the vehicle facing forward and one radar in the front bumper region along the longitudinal centerline of the vehicle.

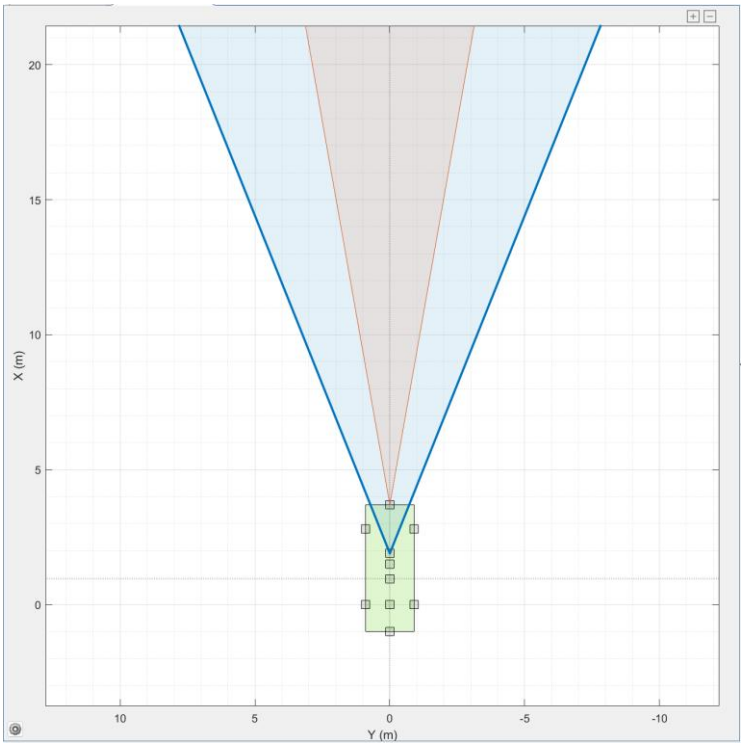

Fig 1. Sensor Placement on the ego vehicle. Red region: Radar coverage area, Blue region: Camera coverage area.

The obstacle is defined as a cuboid box that is set to fall out of the truck it is being carried once the truck reaches 250-meter point on the road. Once it falls out of the truck, it travels an additional distance before it comes to a complete stop to simulate the carried momentum. When an object falls it bounces, rolls, slides, breaks into many pieces etc. Although not a fully accurate representation of the highly complex dynamic system, this simplification provides enough insight into the situation. The distance it travels after falling out is proportional to the speed it is travelling initially and is given by the expression:

Extra distance travelled after falling = Initial speed / 3.

The rest of the vehicles are placed around the ego vehicle in a three-row formation of three vehicles in each row. Within a row the lateral gap between two vehicles is 3.7 meters and the gap between two rows is 39 meters. The initial speed of all actors is set to 36 m/s. The properties of all actors in the scenario are summarized in table 1.

Actor	Length	Width	Height	Initial X	Initial Y
Ego	4.7	1.8	1.4	75	0
Semi-Truck	22	2.6	3	14	0
Car	4.7	1.8	1.4	31.3	3.7
Car	4.7	1.8	1.4	31.3	-3.7
Car	4.7	1.8	1.4	75	3.7
Car	4.7	1.8	1.4	75	-3.7
Car	4.7	1.8	1.4	118.7	3.7
Pickup Truck	4.7	1.8	1.4	118.7	0
Car	4.7	1.8	1.4	118.7	-3.7
Obstacle	3	1.5	2	118.7	0

Table 1. Dimensions and Initial positions of actors.

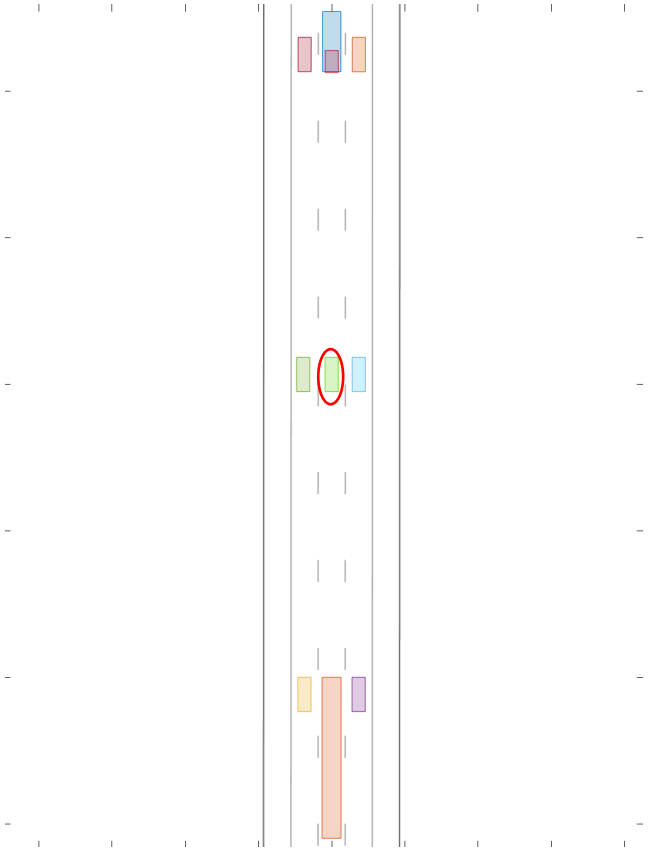

Fig 2. Initial Setup of the problem scenario. Ego vehicle is circled in red.

Scenario and Environment Subsystem:

The scenario is exported as a .m file from the scenario builder. This file contains all the information about the positions, speeds, sensors, and properties of all the elements present in the scenario. This can be read into Simulink using the scenario reader block from the automated driving toolbox. This block reads the scenario from the base MATLAB workspace and gives Actor co-ordinates, Sensor data, and Lane boundary data as the output.

The coordinates from this block are with reference to the ego vehicle. To use this data further for the rest of the subsystems, these coordinates must be transformed to world coordinates. This is done using the vehicle-to-world block from the automated driving toolbox which takes ego vehicle coordinates and actors coordinates as input and returns the actor coordinates transformed to world coordinates.

The sensor data from the scenario reader needs to be analyzed and classified into positive detections of obstacles. This is done using the Vision Detection Generator and the Driving Radar Data blocks. The Vision Detection Generator block takes the camera stream input and gives Object Detections as the output. This block can also be configured to output lane boundary detections if needed. The Driving Radar Data block gives Radar Detections in the form of point clusters which then need to be processed further.

The data from Vision Detection Generator and the Driving Radar Data blocks is then concatenated using the Detection Concatenation block to overlay the detections from both sensors on top of each other. This data is then processed through the Multi-Object Tracker block which helps in keeping track of multiple objects in the path of the ego vehicle.

This subsystem also outputs the struct mapInfo as a Constant which is used for coordinate transformations in subsequent blocks. MapInfo contains information such as number of lanes, Lane width, Lane centers, and other information pertaining to the geometry of the scenario map. It also outputs TimeStep information for synchronization.

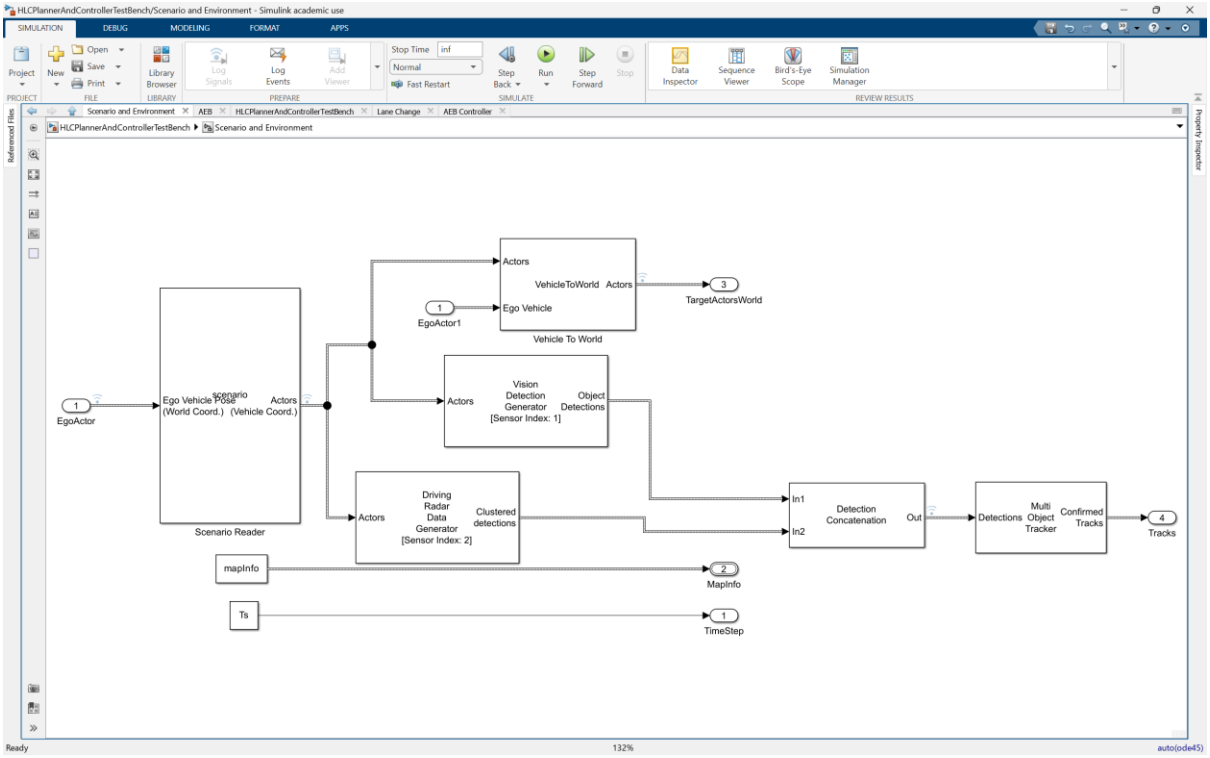

Fig 3. Scenario and Environment Subsystem layout.

Automatic Emergency Braking:

The first step in responding to any incident on the road is in most cases to slow the vehicle down to attain more control over the maneuvering of the vehicle and reduce the momentum of the crash. Hence the decision making for when to brake and how much to brake is very important. The Automatic Emergency Braking subsystem has two main components to it, findLeadCar and AEB Controller.

The findLeadCar MATLAB function is used to identify the lead car, defined as the vehicle in front of the ego car that meets specific criteria: it must be within the ego car's lane, positioned in front of the ego car, and closest to the ego car. The function takes confirmed tracks and a position selector as inputs. It iterates through each track, extracting its position relative to the ego car, and checks if the track satisfies the conditions for being the lead car. The lead car is determined based on its longitudinal position relative to the ego car and its lateral position within the lane boundaries. If a lead car is found, its longitudinal position and velocity are returned as outputs. If

no lead car is found, default values are returned. This function is crucial to maintain a safe following distance and adapt to the behavior of lead vehicle.

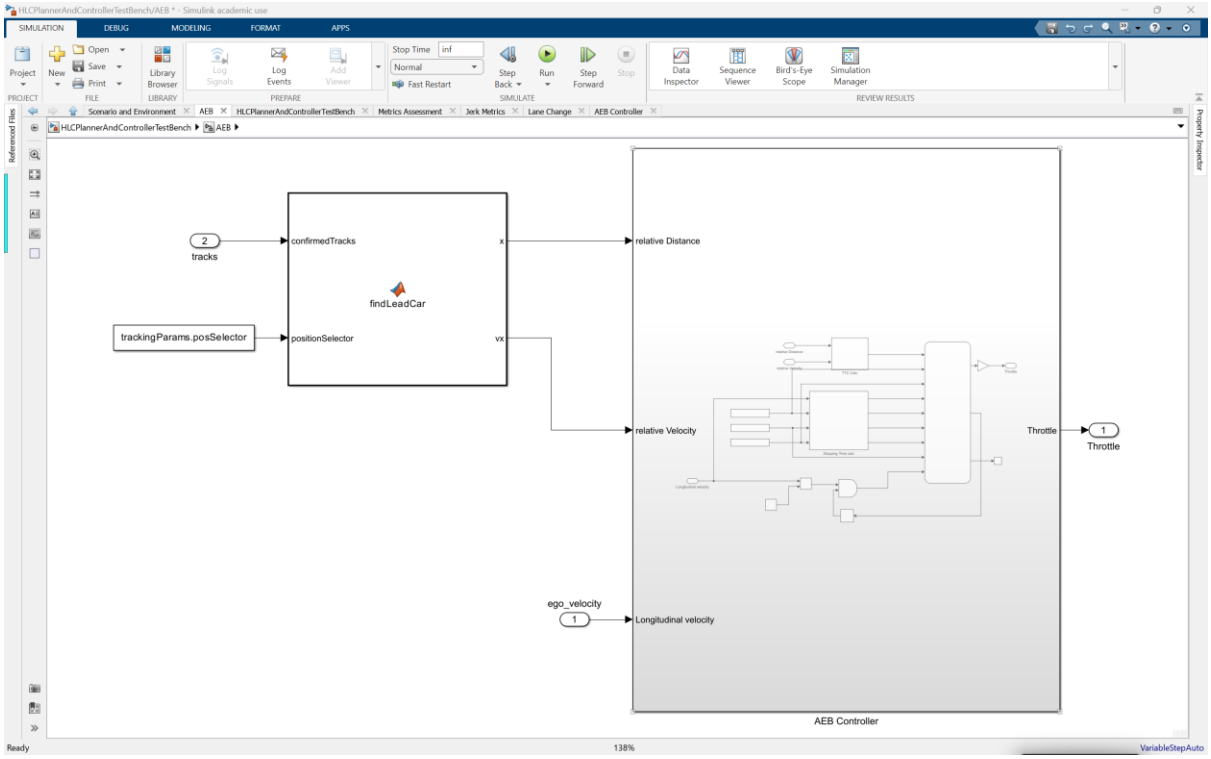

Fig 4. AEB Subsystem Layout.

AEB Controller:

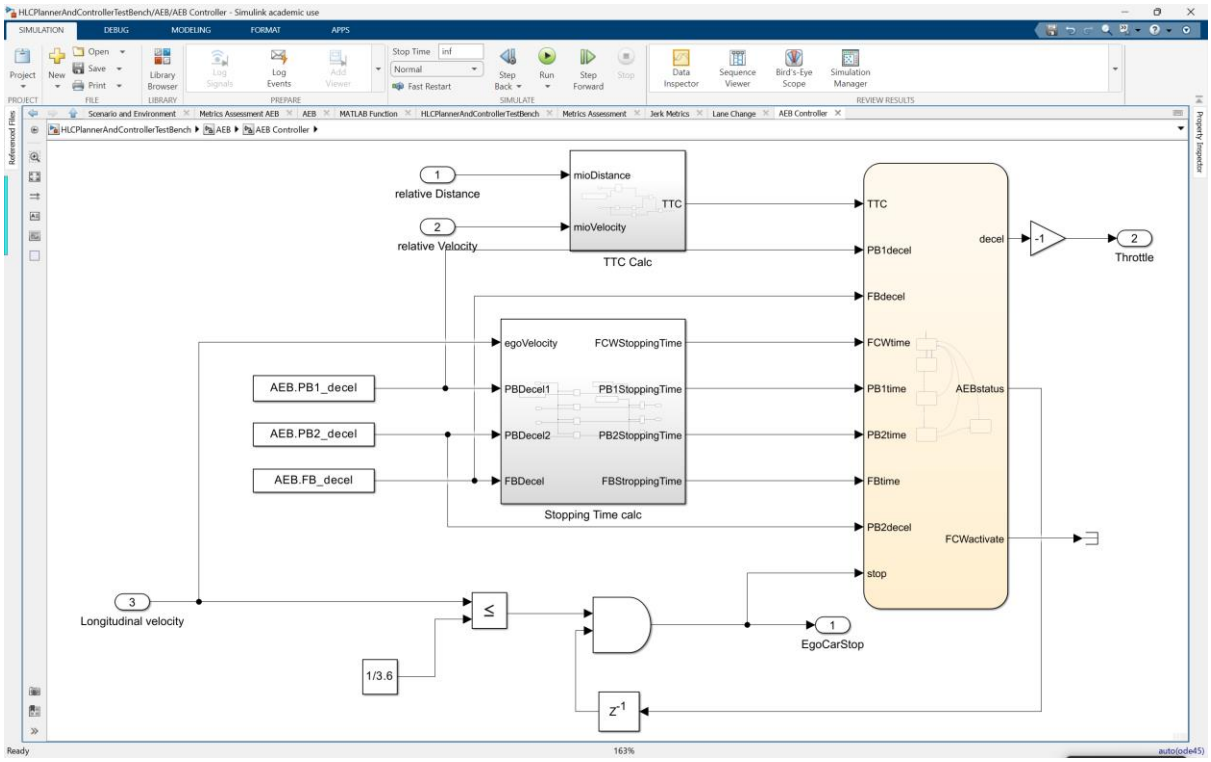

Fig 5. AEB Controller Subsystem Layout.

Time-To-Collision (TTC) is a computationally simple way to determine if an obstacle in the path of the ego vehicle poses a risk of colliding. The Emergency braking decision and amount of braking is determined by calculating this metric and comparing it to the stopping time at different braking levels. The TTC is calculated using the following formula.

$$TTC = \frac{\Delta d + headwayOffset}{\Delta v}$$

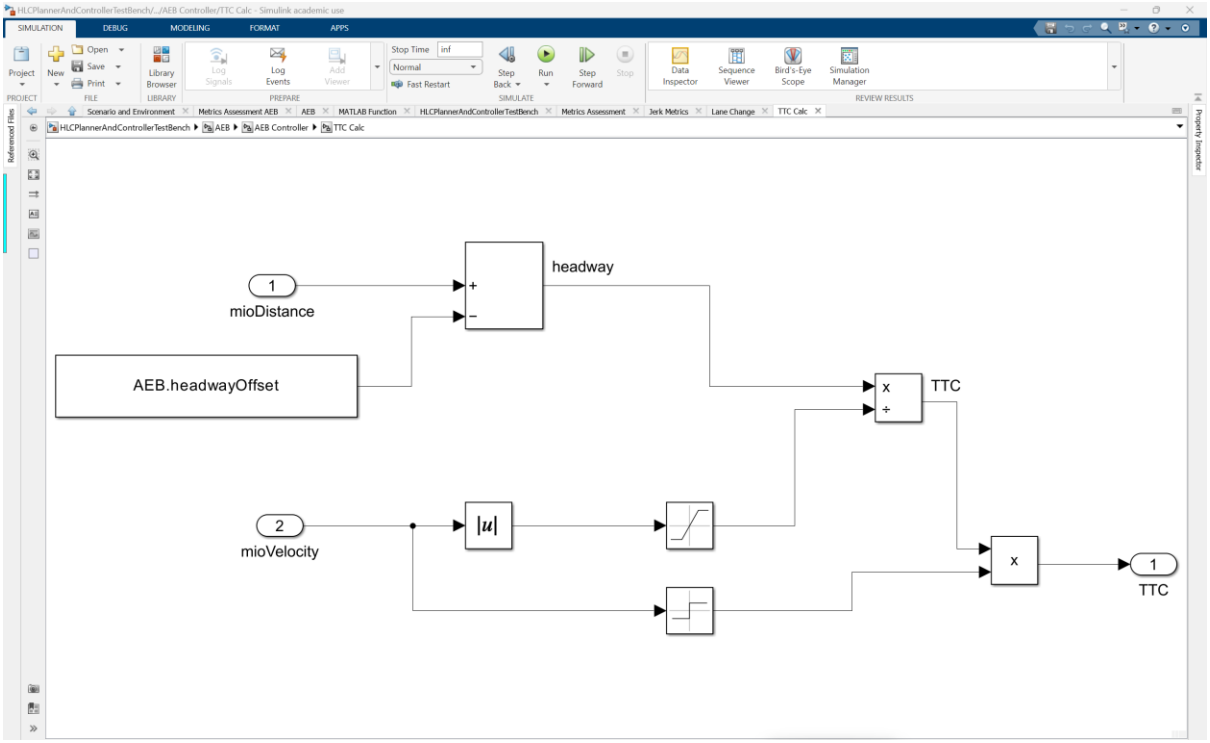

Fig 6. TTC Calculation.

headwayOffset is the factor used in the calculation to introduce the safety gap we expect. This is not the actual safety gap maintained by the vehicle but just a factor of safety for a conservative calculation of TTC. Further, a MATLAB Chart is used to program the logic for choosing the level of braking. The stopping distances at different braking levels are given by the following general formula.

$$d_{0_{braking\ i}} = \frac{v_{ego}}{a_{braking\ i}} + time\ margin$$

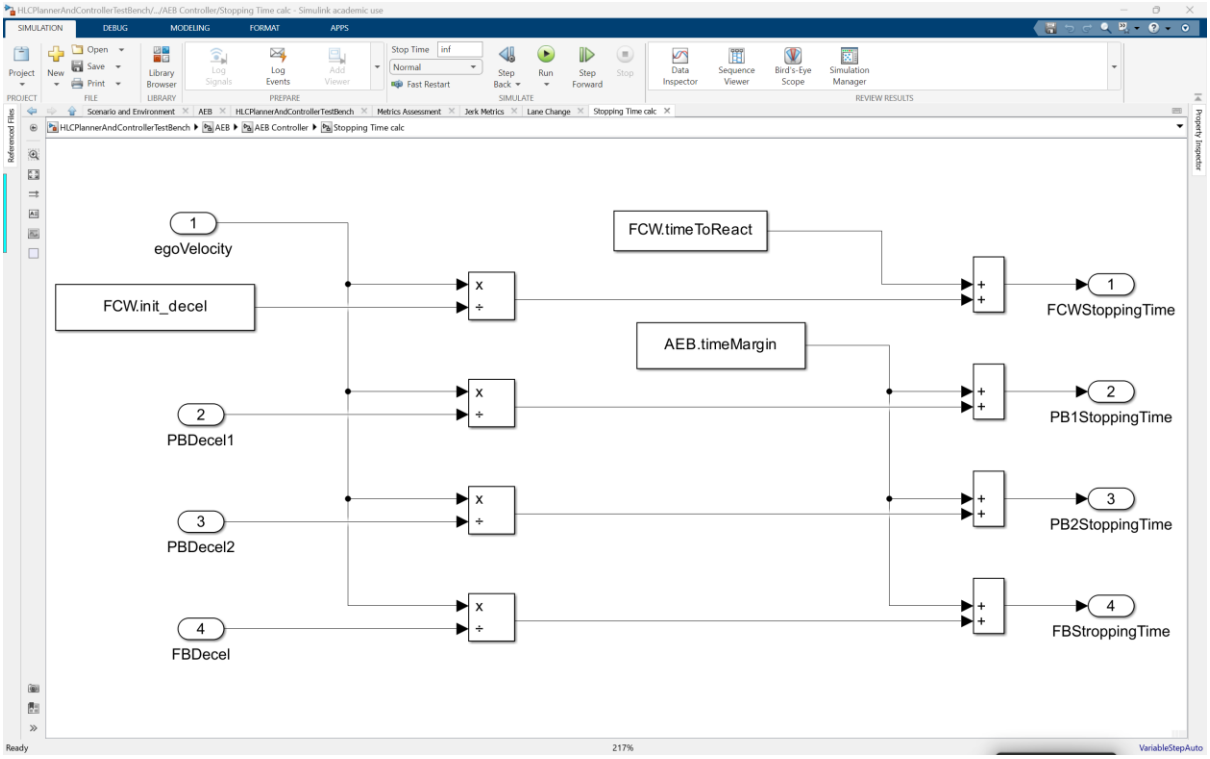

Fig 7. Stopping Distance calculations for different braking levels.

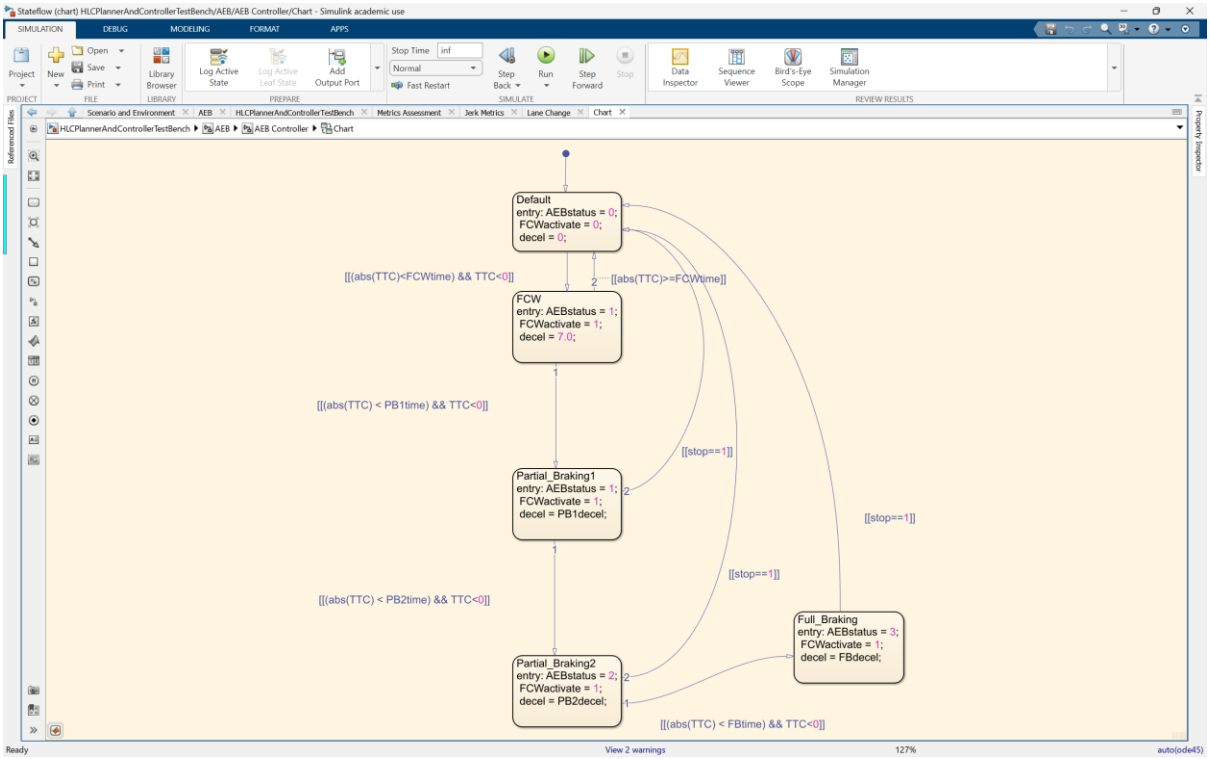

Fig 8. MATLAB Chart showing level of braking decision-making logic.

The final output of the Automatic Emergency Braking subsystem is a simple acceleration value. The amount of acceleration for each level of braking is as follows.

Braking Level	Acceleration (m/s^2)
Initial	-7.0
Partial Braking 1	-8.0
Partial Braking 2	-9.5
Full Braking	-10.5

Table 2. Braking levels and deceleration amount.

Collision Avoidance Maneuver:

The next step to reducing the severity of a collision is to try and avoid the obstacle completely if possible. This is handled by the Lane Change Subsystem. This Subsystem within itself contains two main subsystems, the first is the Lane Change Planner and the second is the lane change controller. The first subsystem handles the path planning, and the second subsystem receives the path information and outputs Steering and Acceleration commands.

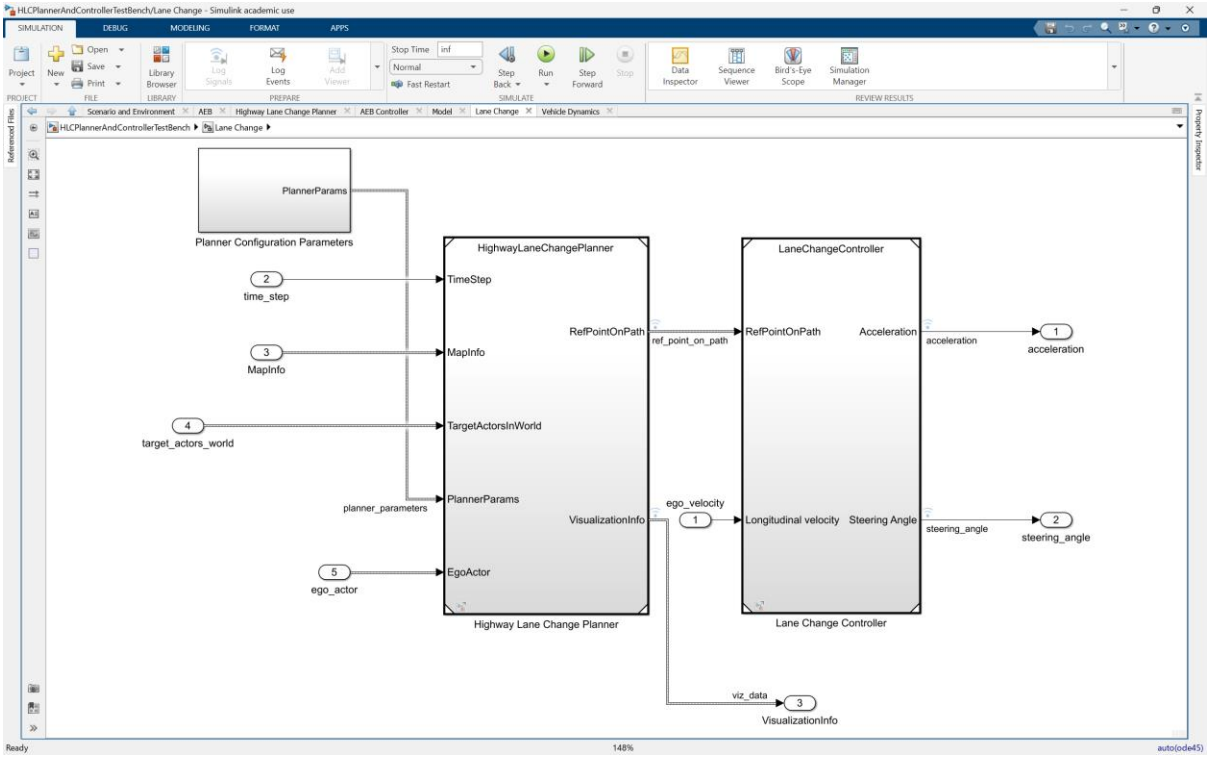

Fig 9. Lane Change Subsystem Layout.

Planner Configuration Parameters:

This Subsystem was created mainly to make the Subsystem less chaotic to look at. It contains many Constant blocks supplying various Parameters required for path planning calculations such as predictionHorizon, PreferredLane, SetSpeed, Safety Parameters, Cost Weights, Path Feasibility Parameters.

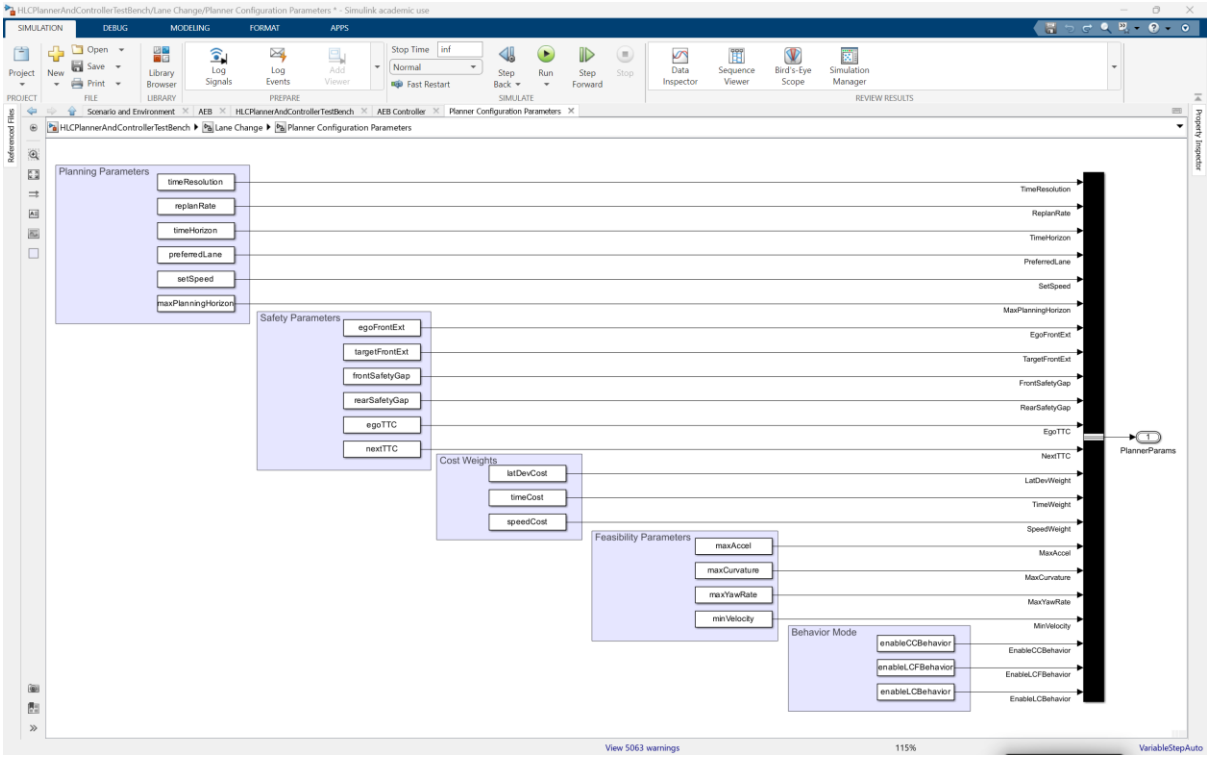

Fig 10. Planner Configuration Parameters Layout.

Highway Lane Change Planner:

The frenetStateConverter function converts the global states of the ego vehicle and target actors into Frenet states relative to a reference path. The Frenet coordinate system is commonly used in autonomous driving systems to describe vehicle positions along a reference path. In Frenet coordinates, the longitudinal position *S* represents the distance traveled along the reference path, and the lateral position *D* represents the distance perpendicular to the reference path. This coordinate system is particularly useful for trajectory planning and control algorithms. The function iterates over all the actors in the scenario and applies inbuilt function *global2frenet* and then combines the results into a Bus and presents that as an output.

The findMIOs function identifies the Most Important Objects (MIOs) in the environment. It does this by analyzing the Frenet states and checks if any of the object's path crosses the Ego vehicles path. If it does, it is flagged as a MIO based on the relative distance, relative speed, if the crossing path would cause collision or not. The function takes the Frenet states of the actors as an input and returns the Frenet states of the MIOs and Lead vehicle.

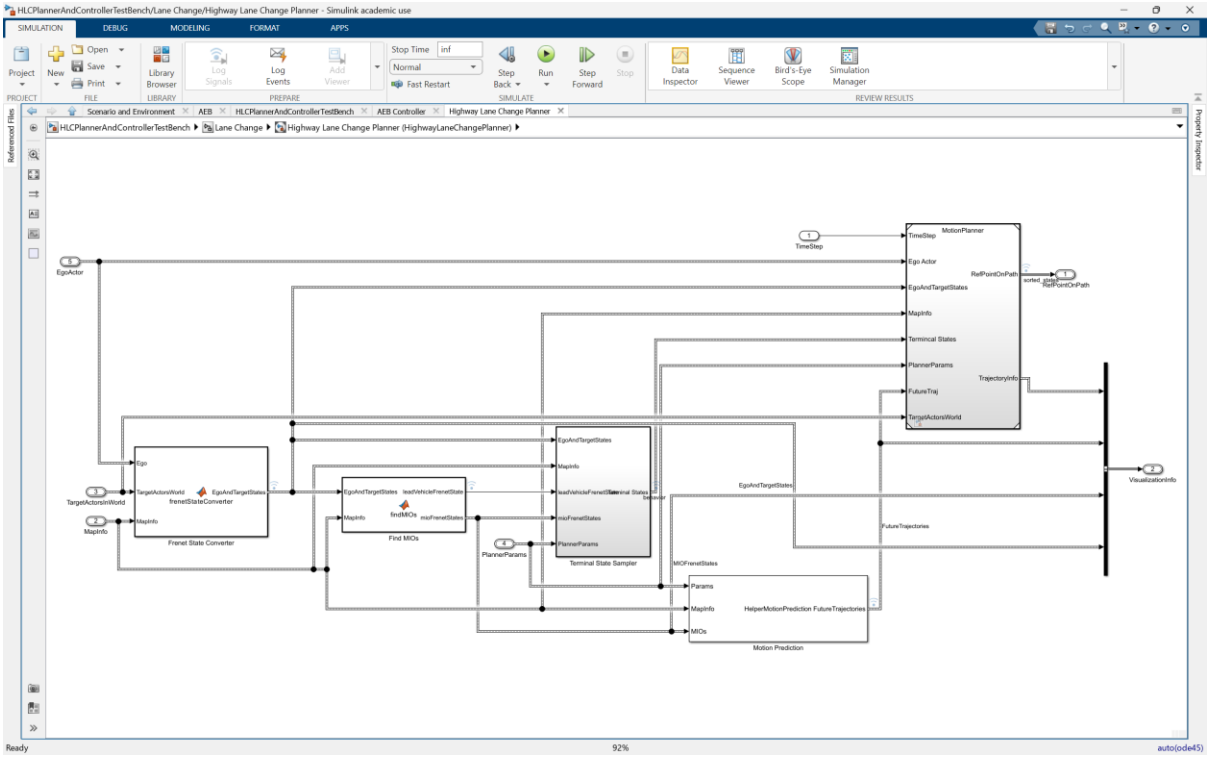

Fig 11. Highway Lane Change Planner Subsystem Layout.

Terminal State Sampler:

Terminal States are important in as they refer to the states that represent the end or goal conditions of a planned trajectory. These states define where the vehicle should ultimately end up or what conditions it should satisfy when the planning process concludes. Terminal states are essential in path planning algorithms as they help guide the vehicle towards specific objectives or destinations while considering various constraints and optimization criteria. The definition of terminal states depends on the specific task or goal of the autonomous driving system. In this case three conditions are checked. First is in Cruise control mode where vehicle is maintaining a set speed, in this case the obstacles do not play a role in the path planning process. Second is the Lead car following. In this case the ego vehicle may slow down to follow the lead car while maintaining a safe distance. In the third case, Lane Change, the end points at each time step are determined in case of a lane change. The terminal states are calculated for each behavior preemptively and concatenated so that the motion planner can modify the planned route to avoid collision.

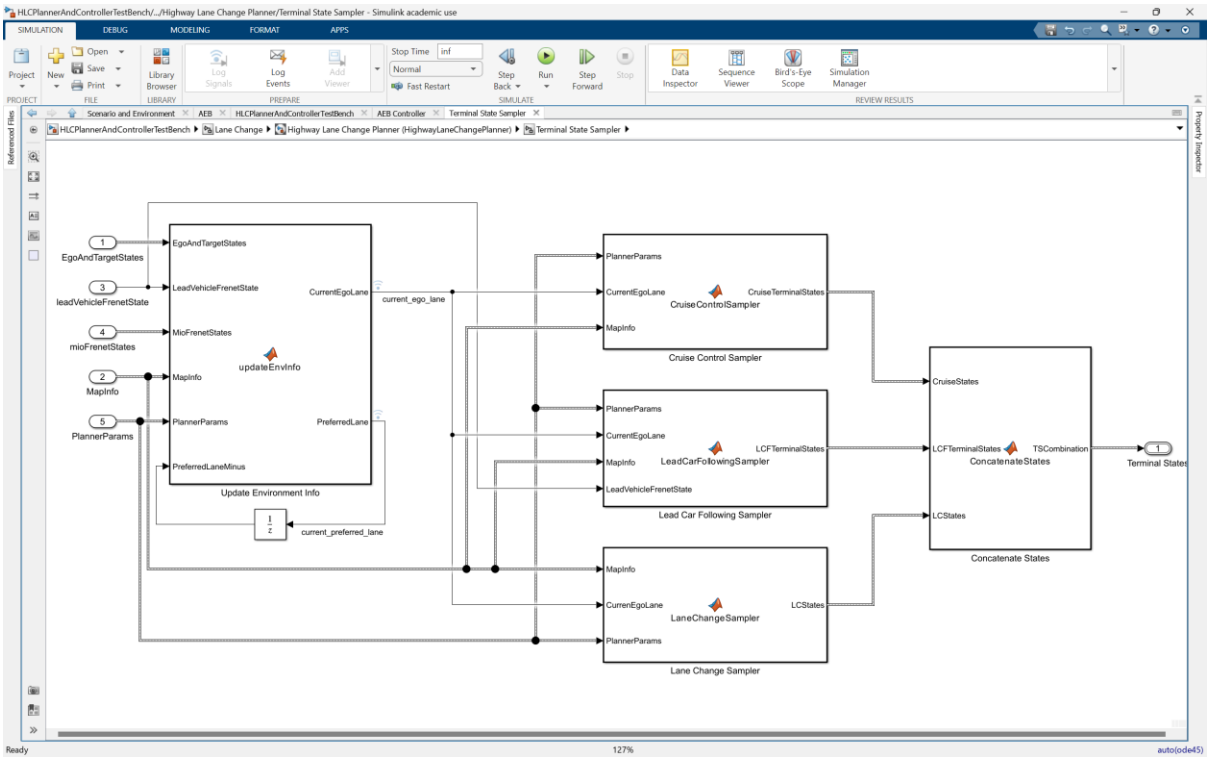

Fig 12. Terminal State Sampler Layout.

Motion Prediction:

The motion prediction block is similar to the terminal states blocks. Like the Terminal states block, using the same properties of the MIOs instead of the Ego vehicle, this block is used to predict the position of the MIOs using which future paths for the Ego vehicle can be generated.

Motion Planner:

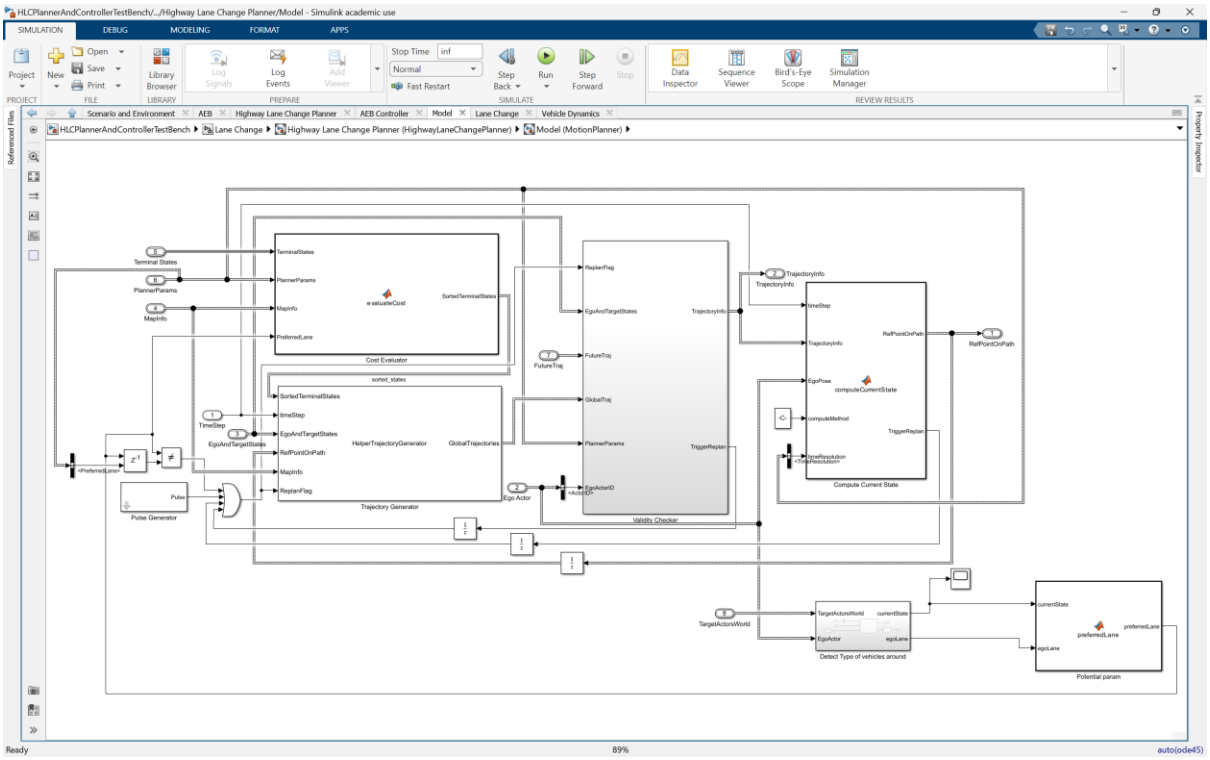

Fig 13. Motion Planner Layout.

In this subsystem, the evaluateCost function sorts the Terminal States based on the planner parameters for allowable safety gap, yaw rate, preferred lane, time to collision, and costs for lateral deviation, time, and speed. These sorted states are then used by the TrajectoryGenerator to generate possible trajectories. It also uses EgoAndTargetStates and Current state in the equation to generate trajectories around the obstacles. This list of trajectories is then used by the Validity checker that checks the kinematic feasibility of the proposed trajectories and checks that they are collision free. It then outputs the trajectory info the next block which calculates the current state of the ego vehicle and outputs a point on the path that the controller can follow. If it cannot find any trajectory that is valid, it raises a replan flag which makes the trajectory generator recalculate possible trajectories. Logical operators are used to check if the new trajectories are the same as old trajectories. The detect type of vehicles around block identifies the type of vehicles surrounding the ego vehicle. It classifies a vehicle to be either a car or a truck and

determines which lanes they are in. The preferred lane block chooses a lane to move into in case of emergency to avoid the lane in which there is a truck right behind the ego vehicle. If there are Trucks in other lanes as well, it is capable of choosing a lane that does not have a truck in it behind the Ego vehicle. The final output of the whole Lane Change Planner subsystem is a reference point on the map that the controller uses to follow a path. The other output of this Subsystem is the visualization data to create a visual representation of the scene.

Lane Change Controller:

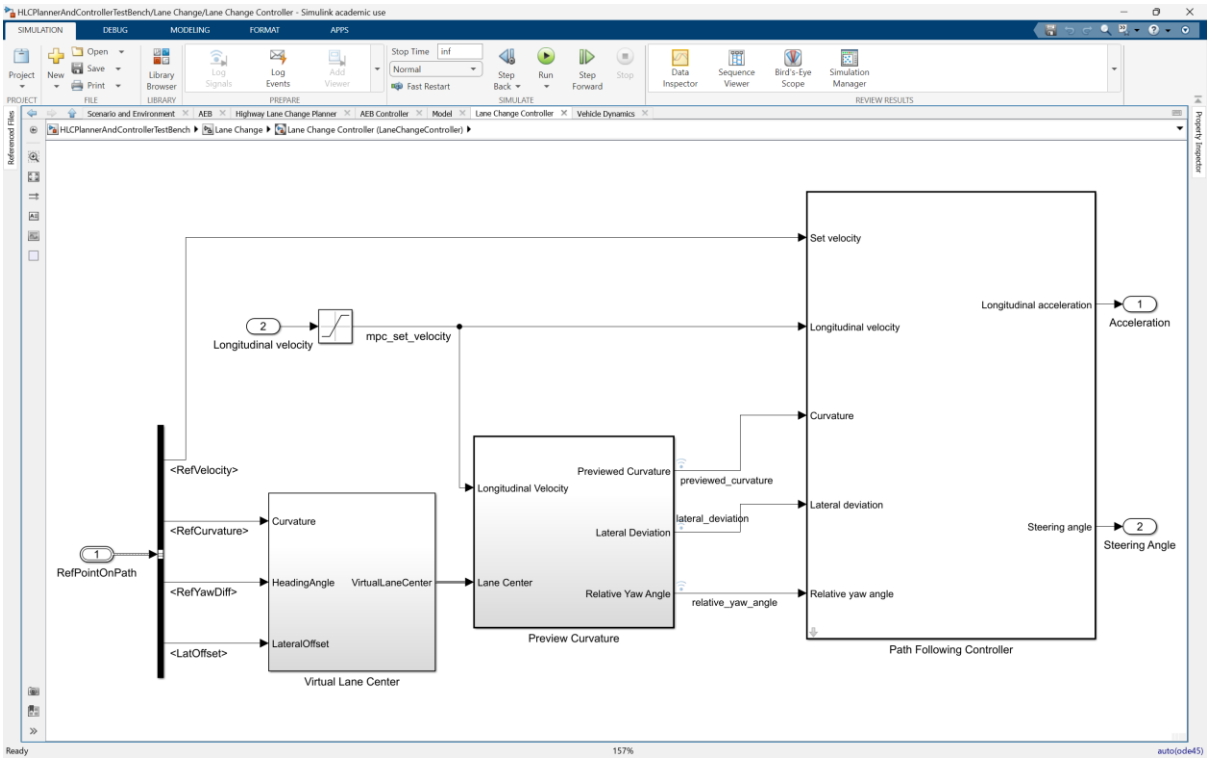

Fig 14. Lane Change Controller Layout.

The function of this subsystem is to generate Acceleration and Steering angle inputs required for the vehicle dynamics. The trajectory info from the lane change planner block is broken down into its individual components using a BusSelector. This information goes into the Virtual Lane Center block which converts the heading angle from degrees to radians as this is what the Path Following Controller needs. These signals are again packed up as a Bus and are given as an output.

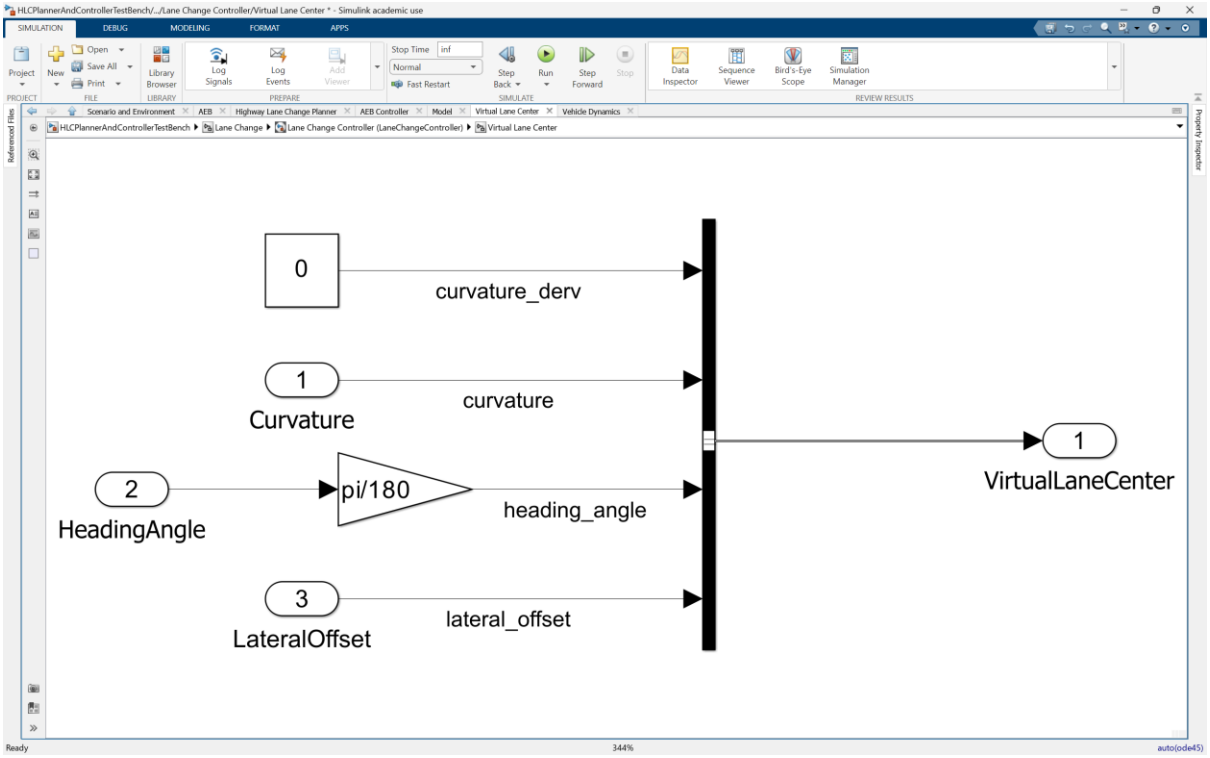

Fig 15. Virtual Lane Center Layout

Next the Preview Curvature block takes the Virtual Lane Center information and the Ego Longitudinal Velocity information and converts it from ISO 8855 standard which is used by all the systems in MATLAB to SAE J670E standard which is used by the Path Following controller and the Vehicle dynamics blocks. The preview curvature block within this subsystem then iterates over the maximum prediction time steps and generates a set of points for the Path Following Controller to follow. This is done by adding each subsequent point over the prediction horizon and adding it to the current curvature.

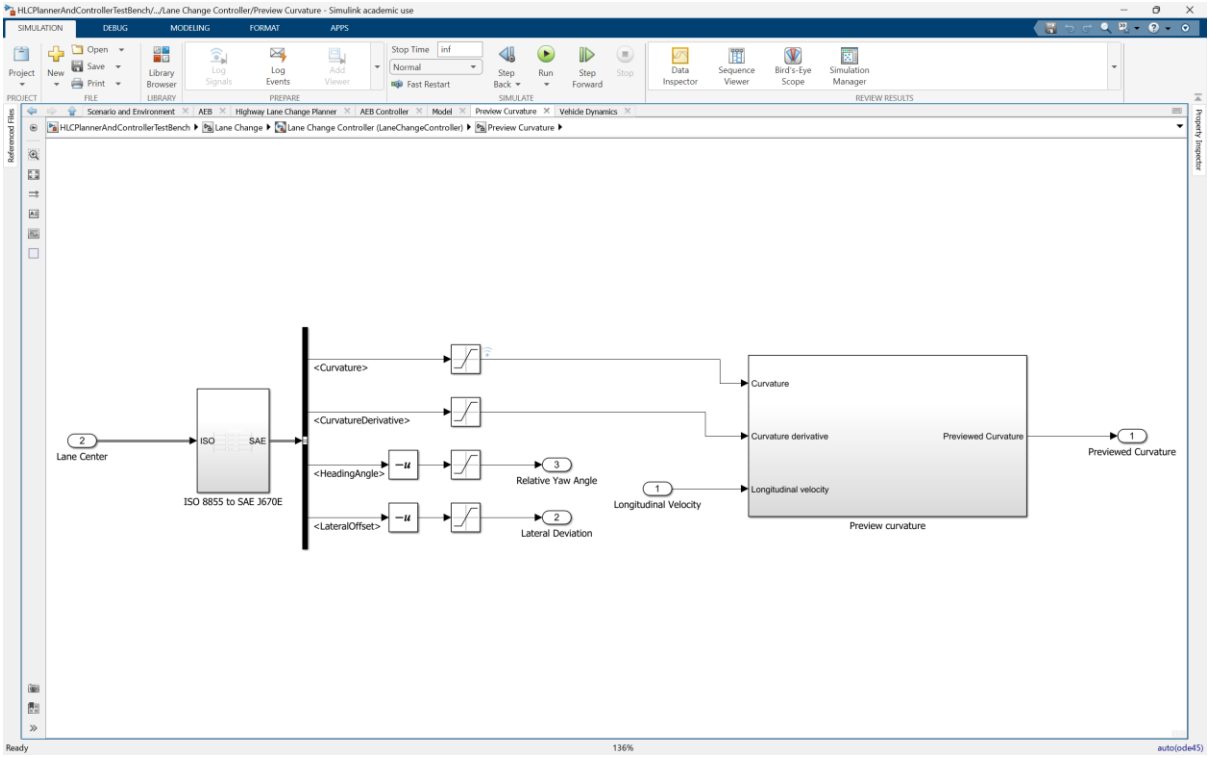

Fig 16. Preview Curvature Main Layout.

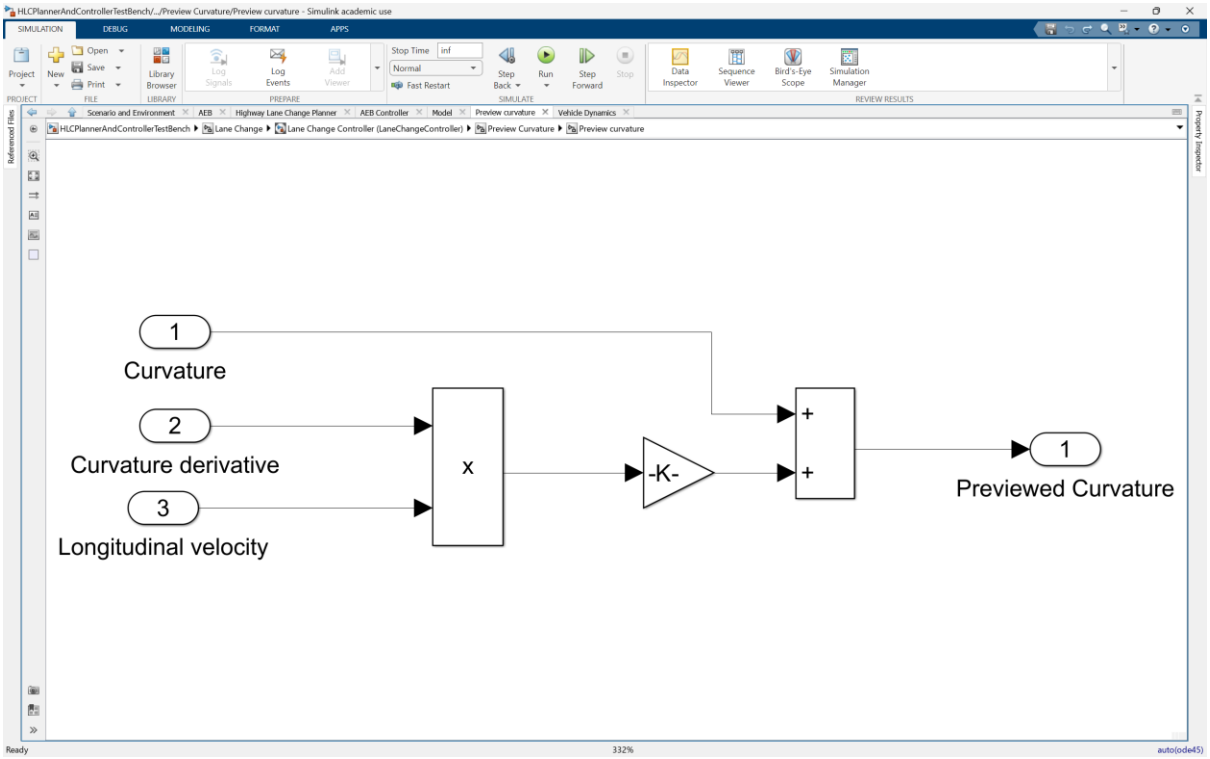

Fig 17. Preview Curvature Layout. (Gain block K is used to iterate over the prediction horizon with time step Ts)

The Path following Controller block from the Model Predictive Control toolbox is used for controlling the vehicle while following the path. This controller is based on the Model Predictive Control (MPC) controller and is a prebuilt controller for the application of path following in autonomous driving or mobile robots. It generates Acceleration information and Steering angle information for the ego vehicle to use which is utilized in the Vehicle Dynamics block.

Acceleration Logic:

As the two systems work alongside each other and have very specific tasks, a logic needs to be defined between them to switch between the two systems as the need arises. Both the subsystems, Automatic Emergency Braking and Lane Change output acceleration commands, so the Acceleration Logic subsystem is placed in between the two subsystems and before the vehicle dynamics block.

The Automatic Emergency Braking system only gives commands to stop the vehicle, whereas the Lane Change system can also increase the acceleration when it identifies a path it can follow without colliding. The Acceleration logic block gives braking command when an obstacle is detected until the lane change system can find a viable path. Once it finds a collision free trajectory, the acceleration control is given back to the Lane change system.

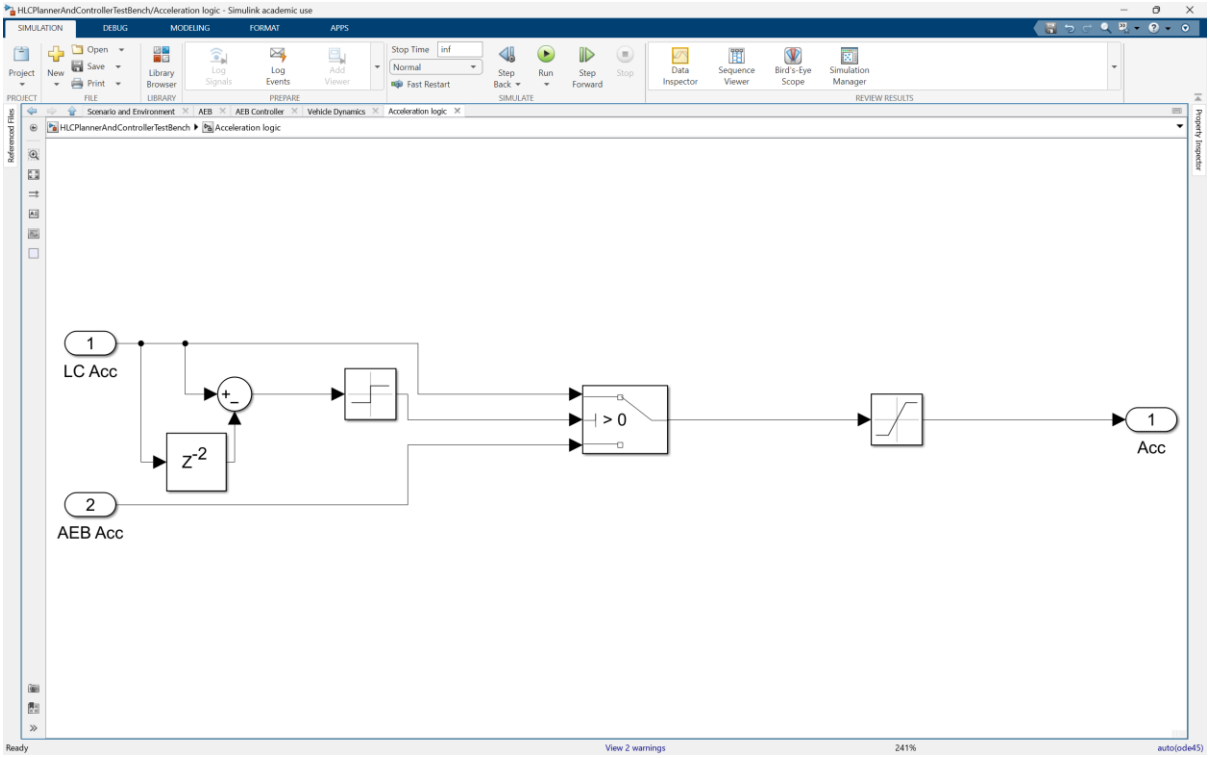

Fig 18. Acceleration Logic Layout.

Vehicle Dynamics:

This block performs the necessary Force and Motion calculations and then sends it to the scenario to simulate the motion visually. The vehicle's dynamic behavior is complex, necessitating a balance between model accuracy and computational efficiency. For this Experiment, a 2-DOF bicycle model and a 1-DOF longitudinal model are employed, with specific assumptions, to design the controller. These models adequately capture the longitudinal, lateral, and yaw dynamics, making them suitable for high-speed driving scenarios and small-angle cornering, such as lane changes. The vehicle's motion is described by equations (1) to (4) [12], governing longitudinal acceleration, lateral acceleration, and yaw rate. These equations consider parameters such as vehicle mass (m), moment of inertia (I_Z), distances from the center of gravity to the rear (l_r) and front (l_f) axles, and longitudinal tire forces (F_{xT}). ψ denotes vehicle heading angle, u, v represents vehicle longitudinal/lateral velocities, r represents vehicle yaw rate at CG (center of gravity), m is the vehicle mass. The model incorporates a front steering system and utilizes a linear tire model to calculate lateral forces (F_{yf} and F_{yr}) based on front and rear tire cornering

stiffness (C_{af} and C_{ar}) and steering angle (δ). Equations (5) and (6) [12] express the lateral forces in terms of tire slip angles (a_r and a_f) and velocities.

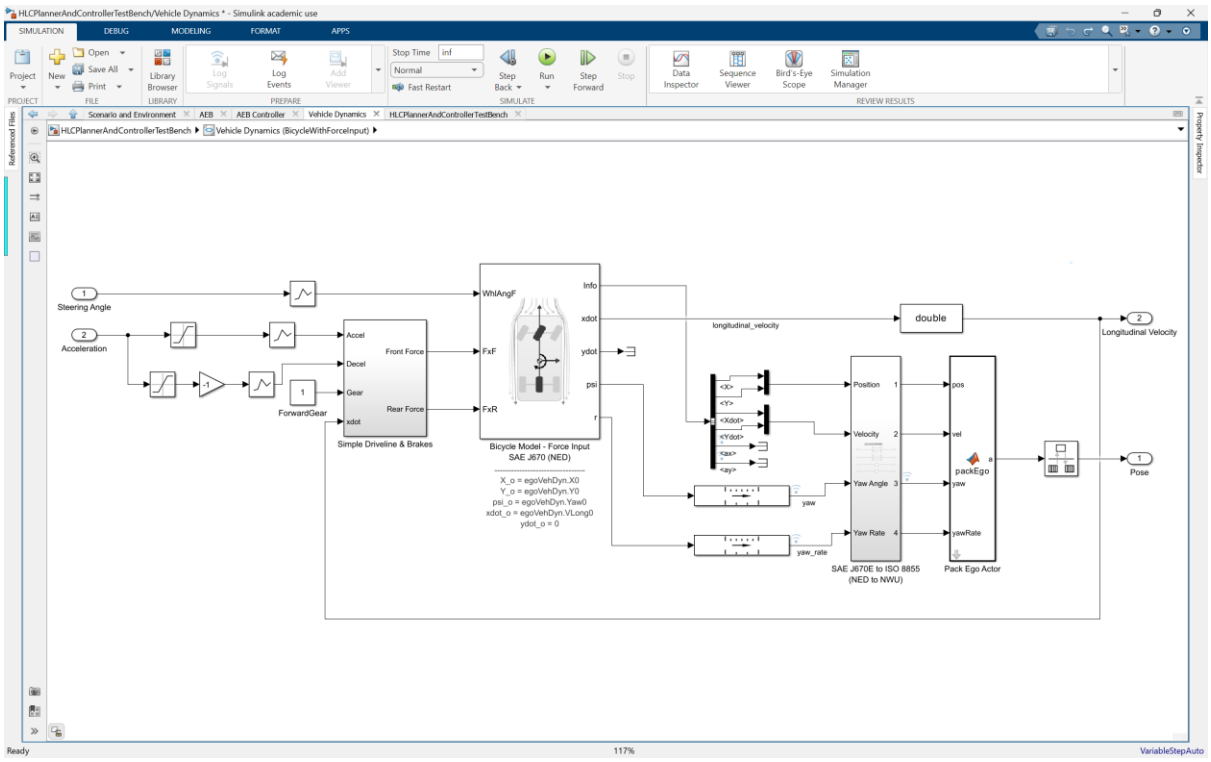

Fig 19. Vehicle Dynamics Layout.

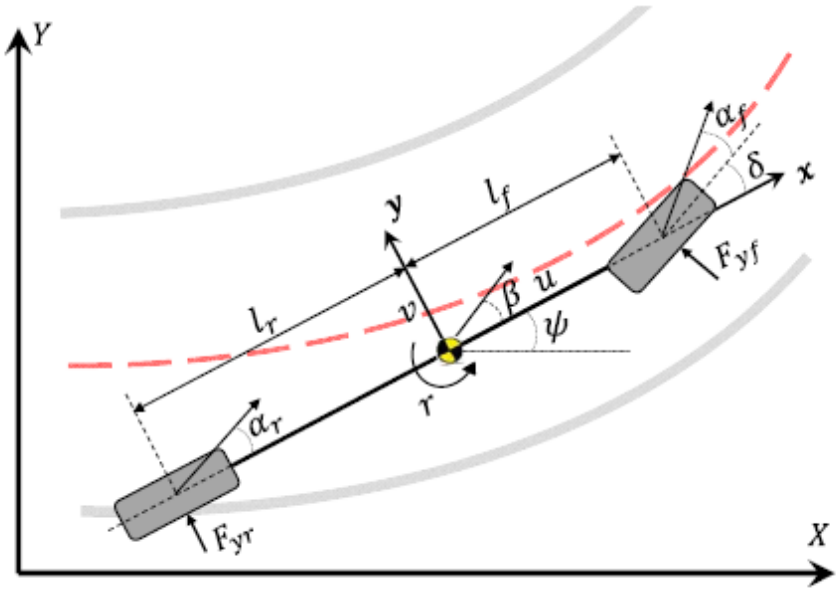

Fig 20. Bicycle Model Vehicle Dynamics. [12]

$$m(\dot{u} - vr) = F_{xT} \qquad (1)$$

$$m(\dot{v} + ur) = F_{yf} + F_{yr} \quad (2)$$

$$I_z\dot{r} = F_{yf}l_f - F_{yr}l_r \qquad (3)$$

$$\dot{X} = u\,cos\psi - v\,sin\,\psi, \qquad \dot{Y} = u\,sin\,\psi + v\,cos\psi \quad (4)$$

$$F_{yf} = -C_{\alpha f}\alpha_f = C_{\alpha f}\left(\delta - \frac{v + l_f r}{u}\right) \qquad (5)$$

$$F_{yr} = -C_{\alpha r}\alpha_r = C_{\alpha r}\left(-\frac{v - l_r r}{u}\right) \qquad (6)$$

These values are then converted back into ISO 8855 from SAE J670E so that it can be processed by MATLAB and scenario reader to update the scenario and create visual representation of the scene.

Experiment procedure:

Once the experiment setup is complete, it is run at various speeds to observe how the system reacts to the obstacle in the road that appears suddenly at high speeds. The experiment is repeated at different speeds to, and parameters are tuned to allow the ego vehicle to best avoid the collision while also keeping the parameters realistic. Observations are also made about how the ego vehicle reacts to having a large truck behind it. The parameters that were tuned include the safety gaps maintained by the ego vehicle, Jerk allowed, Acceleration and Deceleration, Costs for change of Speed, Lanes. Given the scenario, these parameters are set to be aggressive.

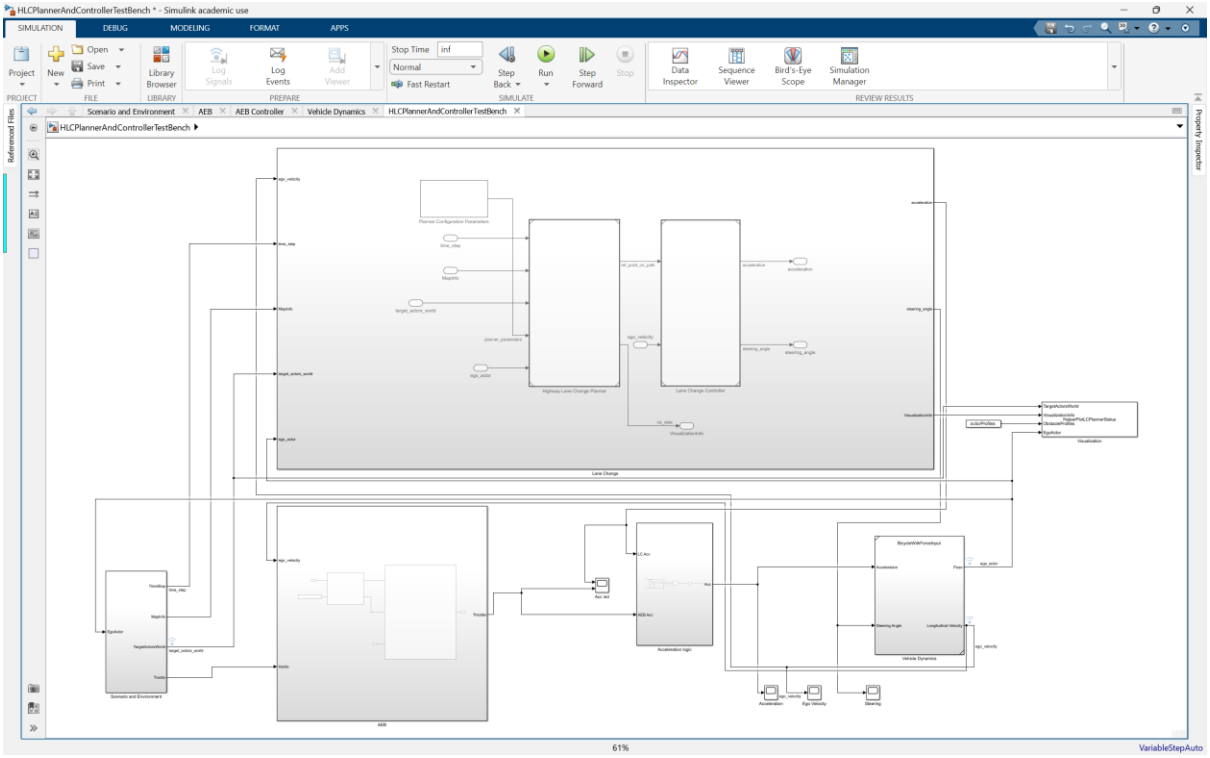

Fig 21. Complete System Layout.

Results:

The objectives of the experiment were firstly, to design a responsive system capable of reacting to the sudden appearance of a non-crossable obstacle in the path of the ego vehicle, and secondly, to determine the necessity of adjusting mitigation strategies based on the surrounding vehicles' configurations.

A system implementation of Lane Change Planner and Controller and Automatic Emergency Braking is created on MATLAB – Simulink. Using components from Simulink, Automated Driving Toolbox, and Model Predictive Toolbox and taking inspiration from the Highway Lane Change Planner and Controller [13] and Automated Emergency Braking examples from the Automated Driving Toolbox [14] a system integrating the functionalities of these two examples is created. A scenario pertaining to the specifications of the problem statement is created using the scenario builder application.

To test the functionality of the designed system, the same scenario is run while incrementing speed with each run up to the speed defined by the problem statement and observations are noted. The test is conducted at 20 m/s, 25 m/s, 30 m/s, 36 m/s.

20 m/s Test:

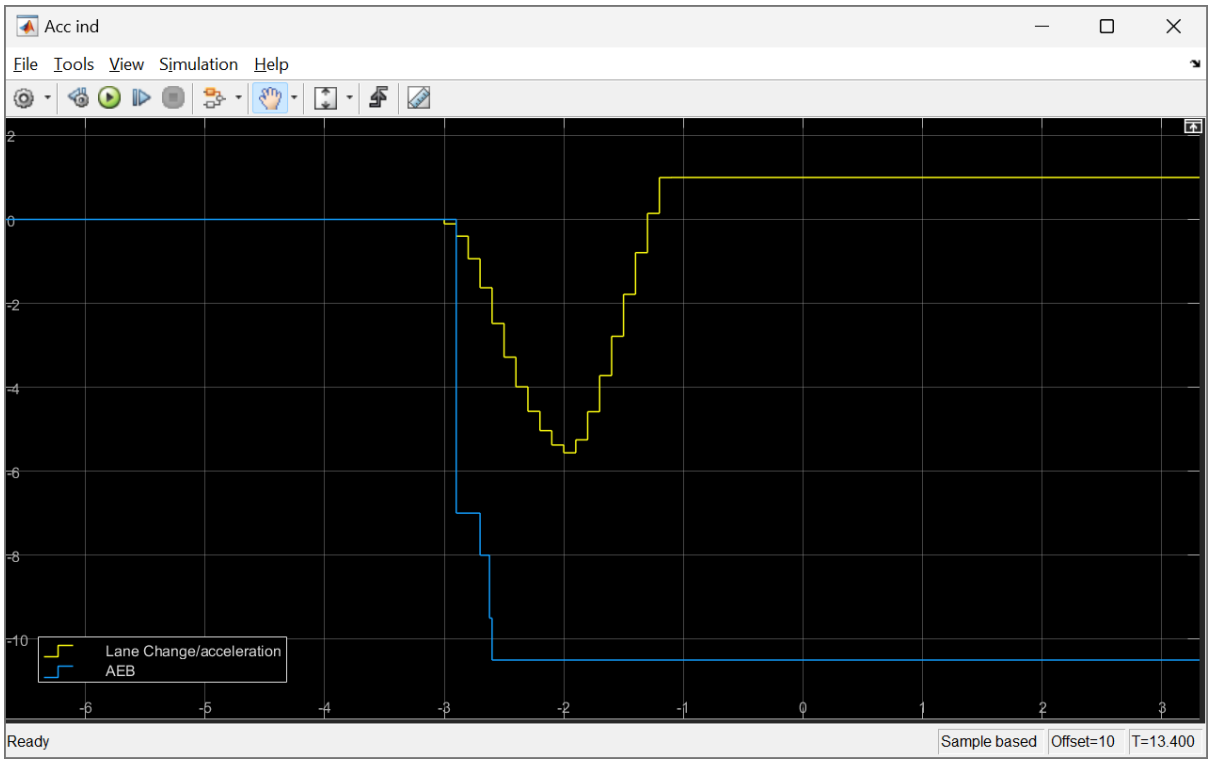

Fig 22. Individual accelerations from each subsystem. 20 m/s.

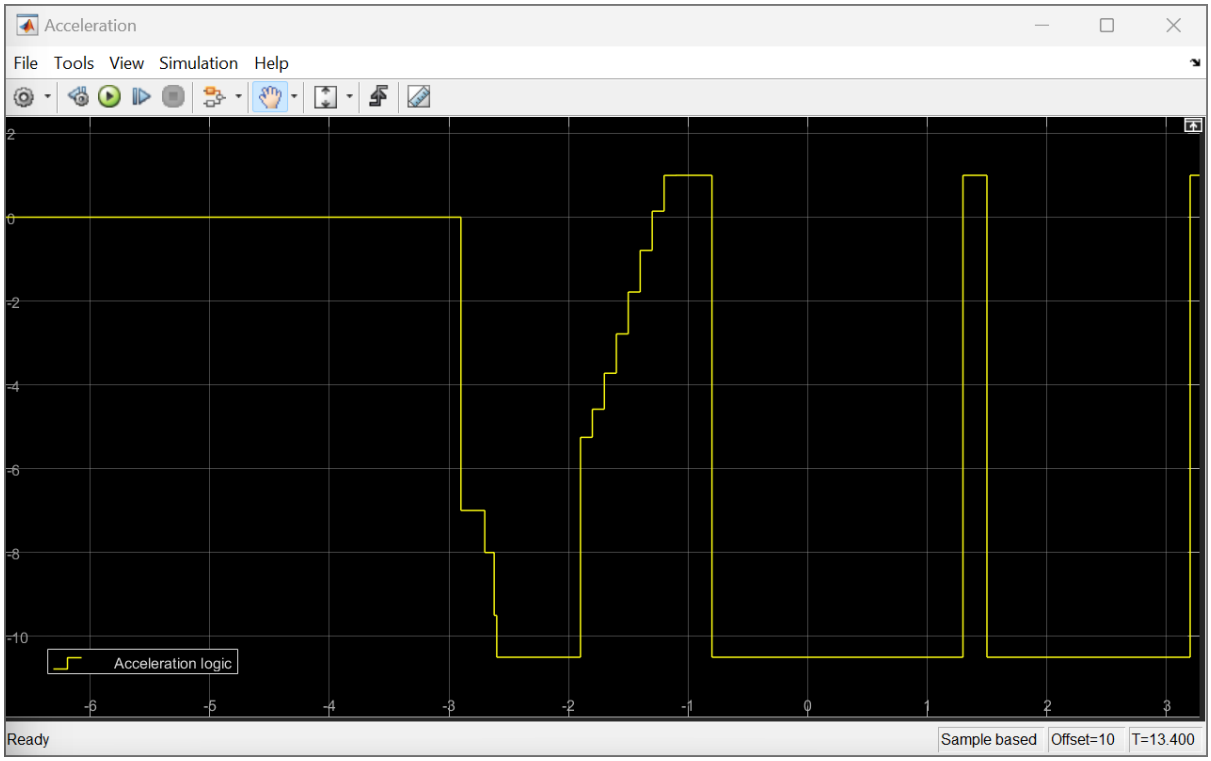

Fig 23. Actual acceleration after applying logic to switch between the two systems. 20 m/s.

Travelling at 20 m/s (~ 45 mph), when the ego vehicle encounters an obstacle that suddenly falls out of the truck in front of it, it applied emergency brakes following different levels of braking based on how close it gets to the obstacle. This brings the vehicle to a controllable speed where it can safely change lanes and maneuver around the obstacle.

The obstacle falls out of the truck at around 7th second in the simulation AEB deceleration is applied until the Lane Change system could find a trajectory to avoid the obstacle. Once it finds a trajectory, the Lane Change system accelerates, controls the steering input, and brings the car to a stop on the shoulder. Fig 1. shows the acceleration inputs from both the systems, Fig 2. shows the actual acceleration input given to the vehicle dynamics block.

Fig 4. tracks the Ego velocity and shows the ego vehicle slowing down and then accelerating again once away from the obstacle to match the speed of the traffic again before coming to a complete stop on the shoulder. Fig 5. tracks the steering inputs which shows deflection to the left at first and then subsequent inputs in the bid to keep the vehicle under control.

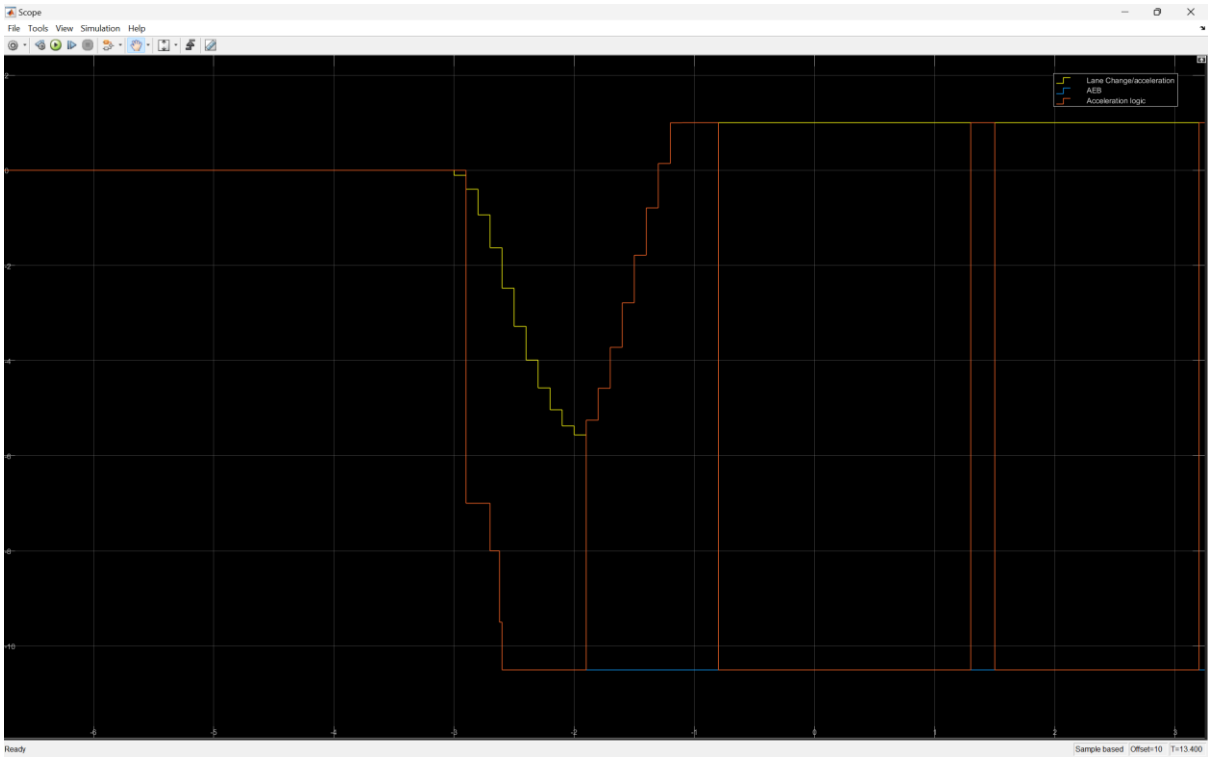

Fig 24. Fig 22, Fig 23 Overlap. 20 m/s.

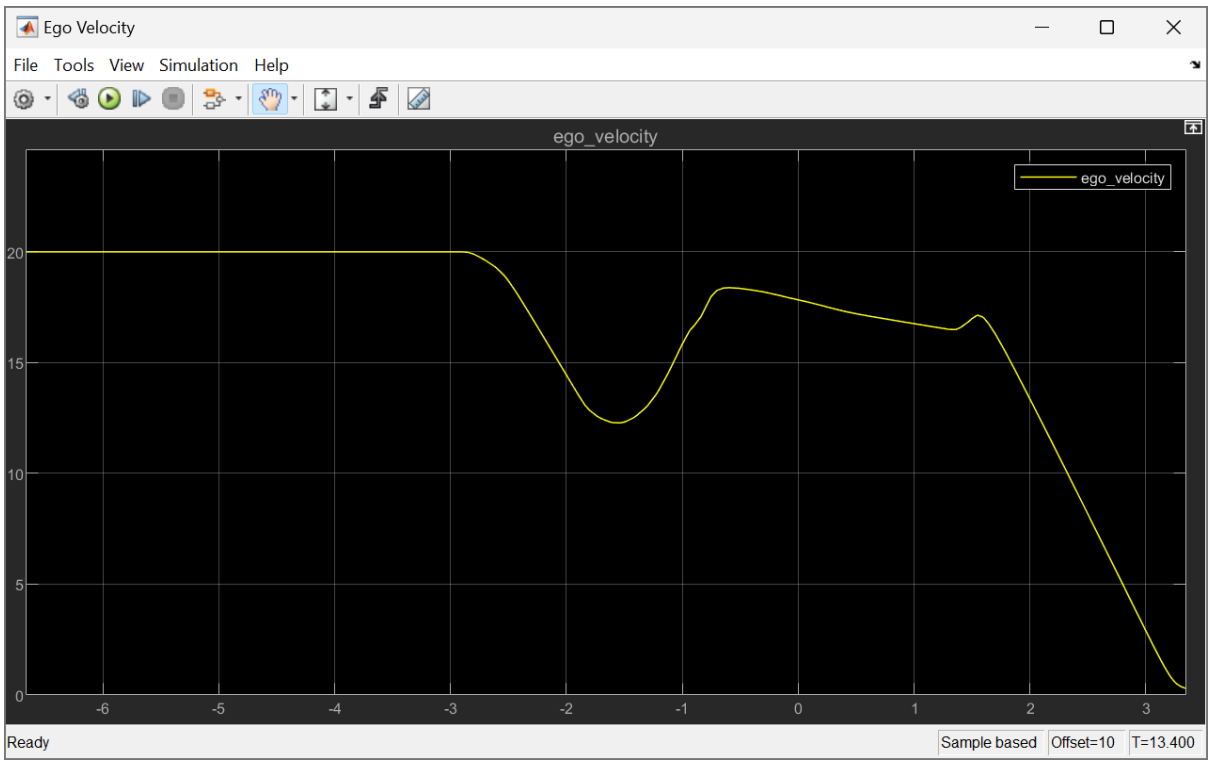

Fig 25. Ego Velocity. 20 m/s.

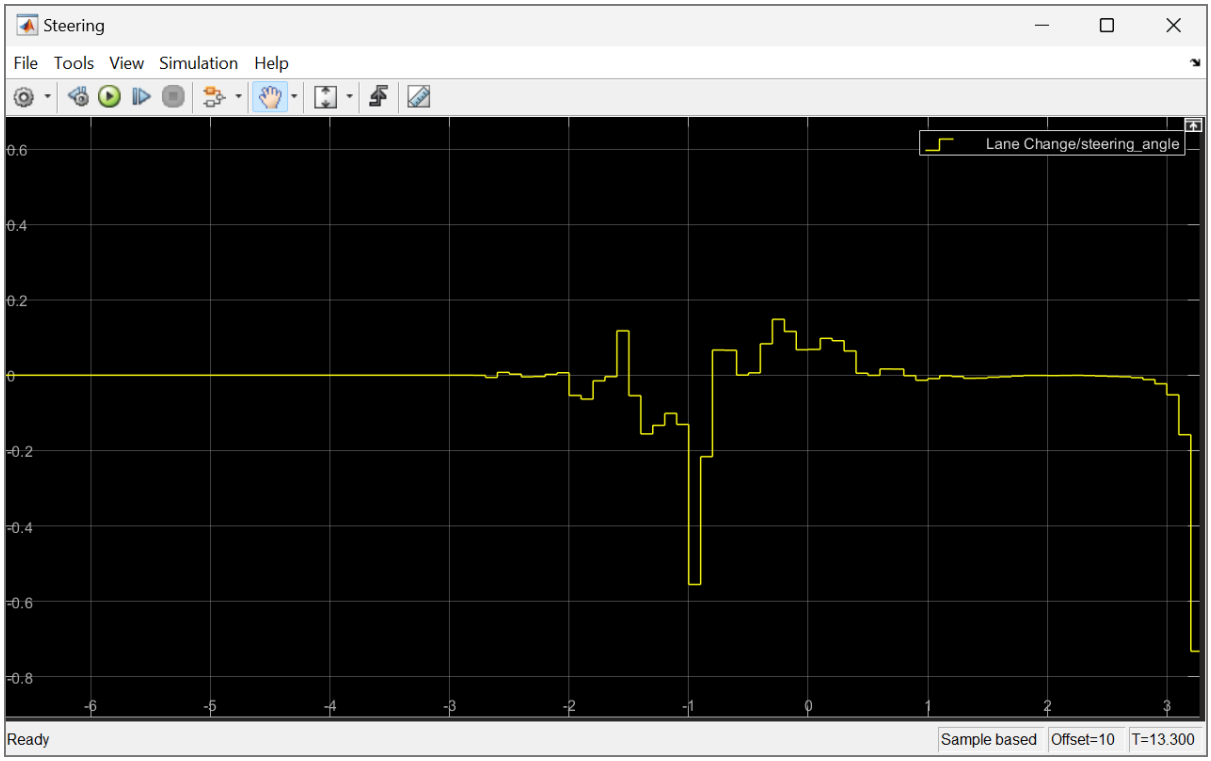

Fig 26. Steering Inputs. 20 m/s.

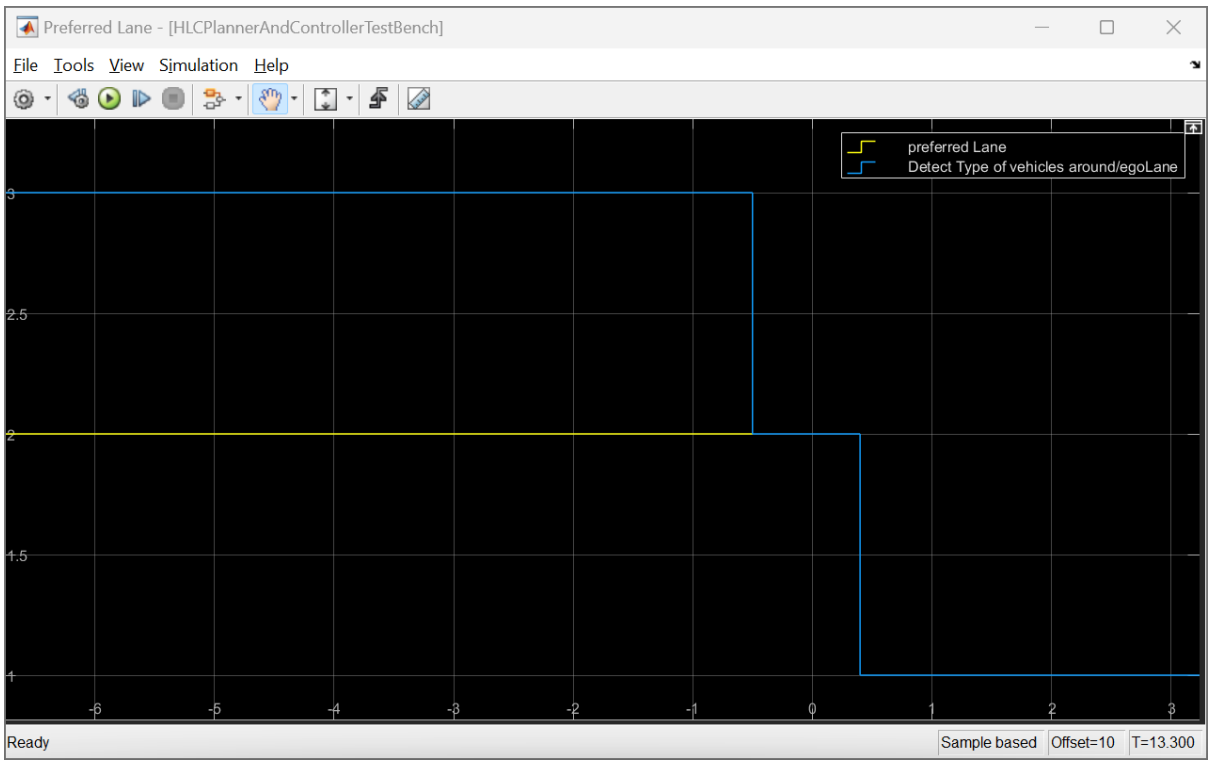

Fig 27. Preferred Lane. 20 m/s.

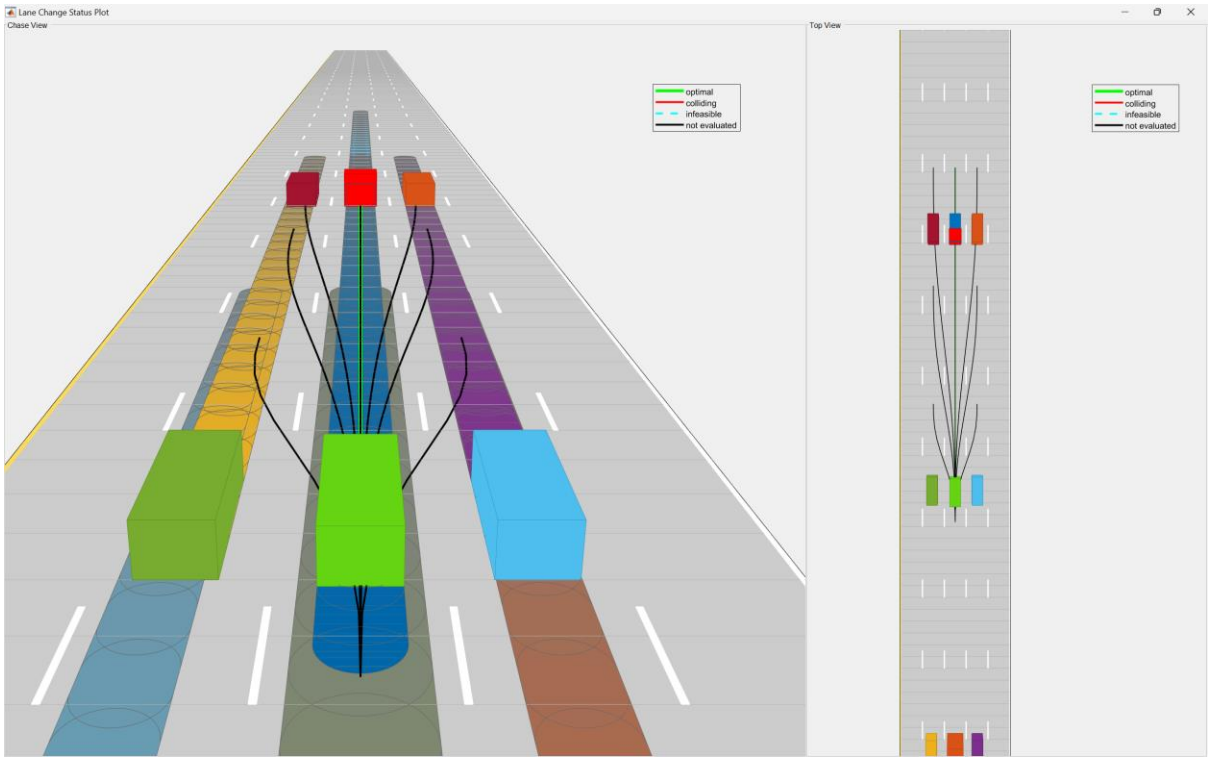

Fig 28. Ego vehicle and its projected paths in different states in normal operation. 20 m/s.

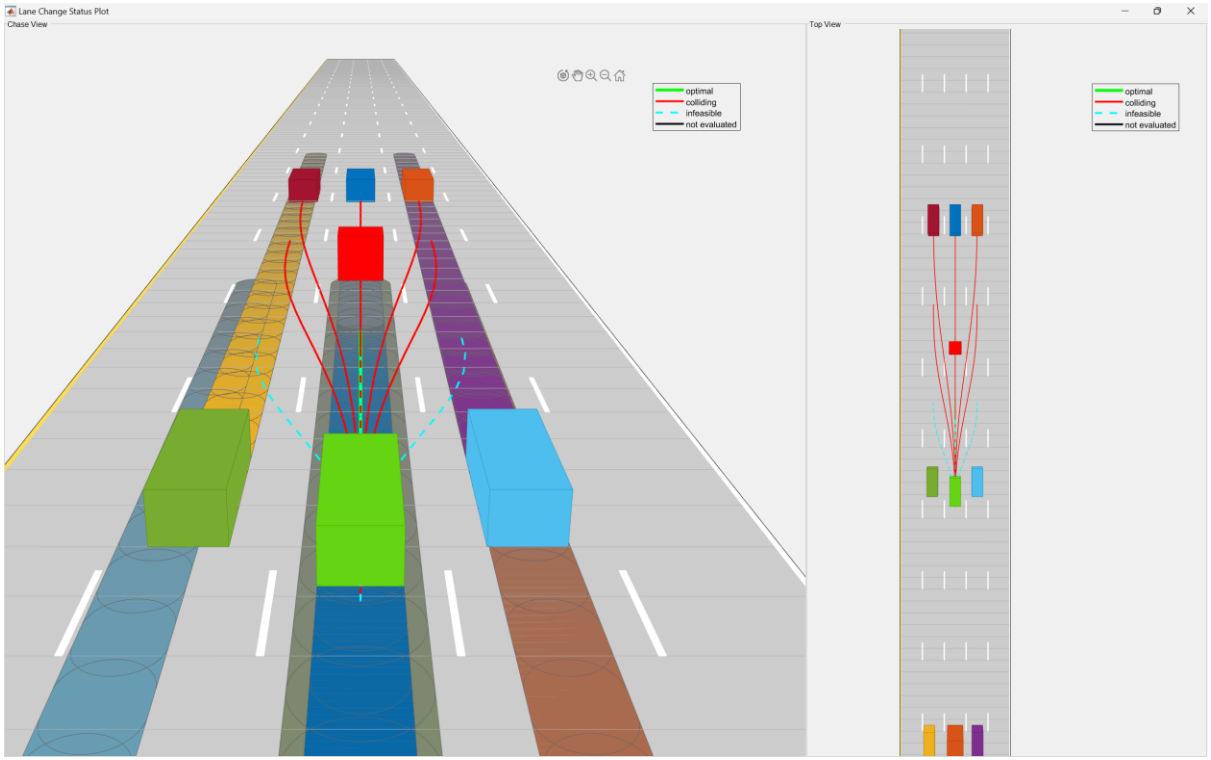

Fig 29. Ego vehicle and its projected paths when the obstacle is detected. 20 m/s.

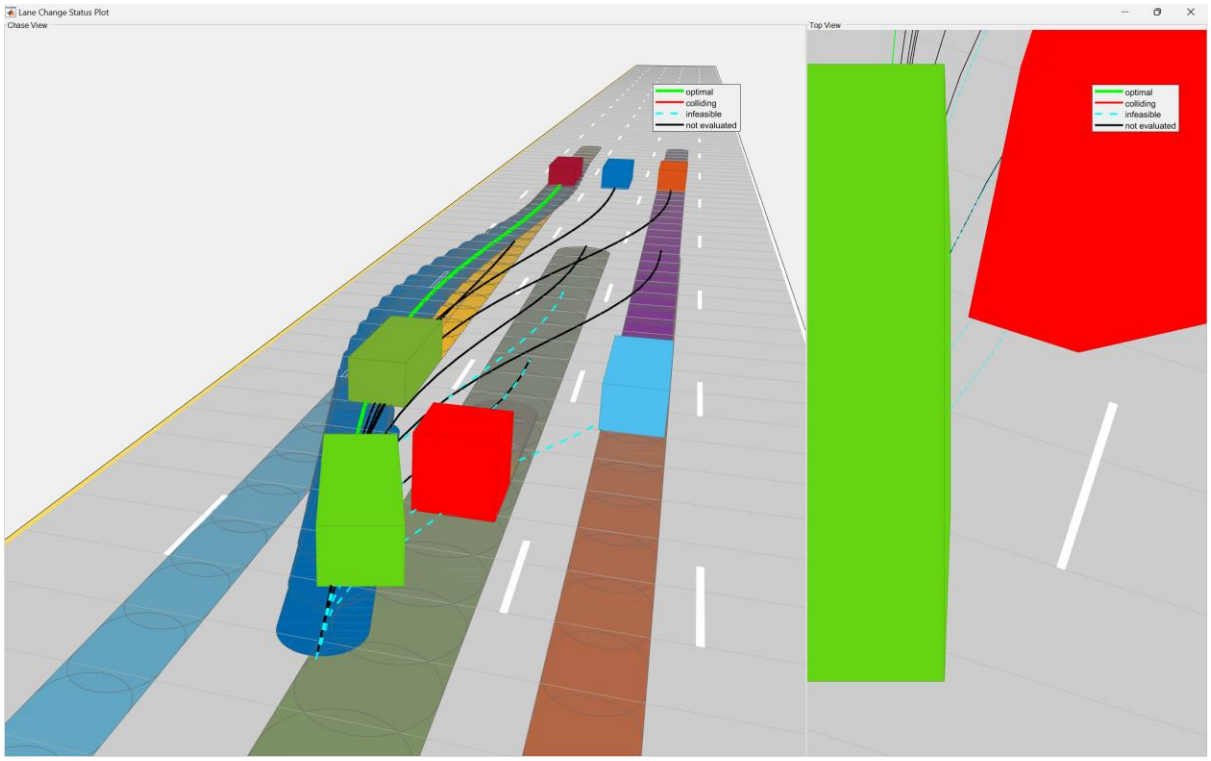

Fig 30. Collision avoided. 20 m/s.

When conducting tests at higher speeds, the scenario is such that there is absolutely no space for the ego vehicle to leave its own lane. Deviating from a straight path would lead to a partial collision which could send the vehicle in an uncontrollable spin or lead to secondary accidents. Due to unavailability of the space and any trajectory that is kinematically valid, the ego vehicle chooses straight path and collides with the obstacle. Due to the presence of the automated braking system, the speed at impact is significantly reduced, and the full collision protection from the crumple zones of the car is utilized.

Even while travelling at 36 m/s (~ 85 mph) the maximum braking allows the car to slow down to about 27 m/s (~ 60 mph). While is still a very high speed, it is within the crash protection design specifications of most modern cars.

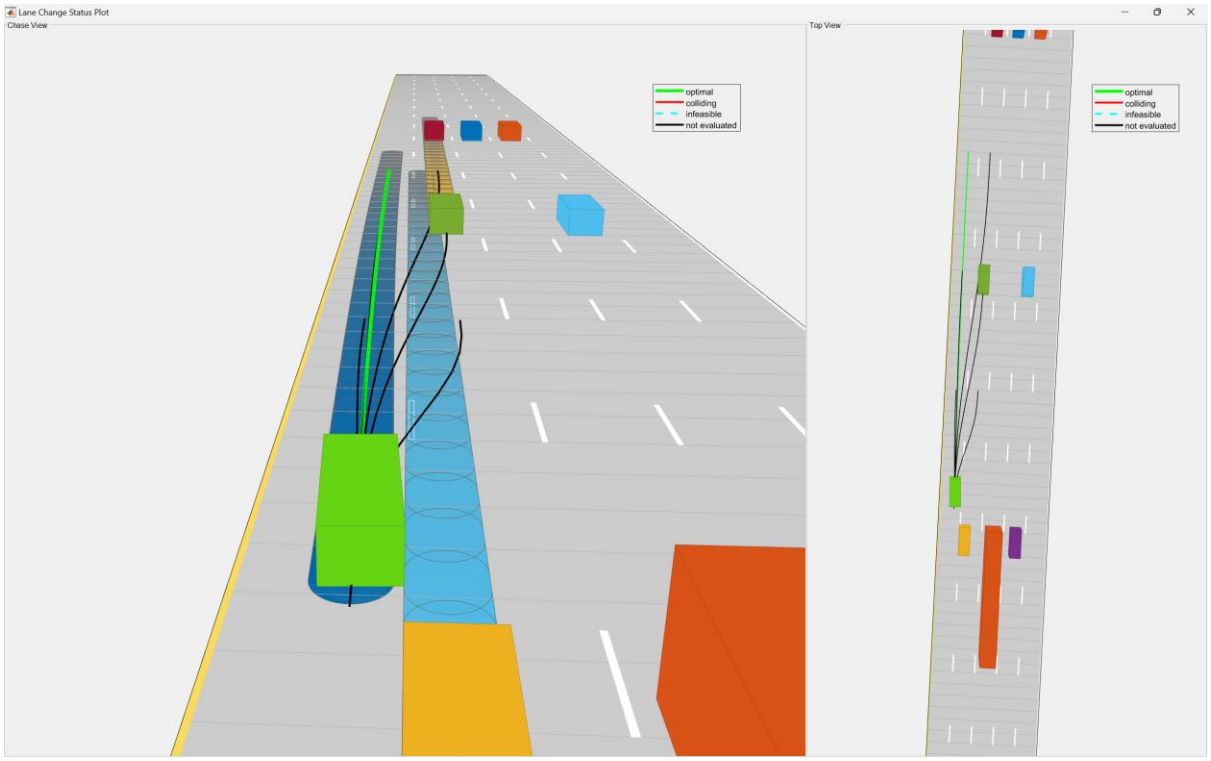

Fig 31. Ego vehicle comes to a stop safely on the shoulder.

This leads to address the Second objective of this experiment, to determine if an additional logic to move away from the lane where a truck is present is required and useful. As seen from the results of the experiment, the Ego vehicle never even gets a chance to change lanes. The logic to move away from the lane with a truck in it would be useful if the Ego vehicle had the space to change lanes or had the capacity to brake with higher deceleration. The deceleration figures chosen for this experiment are realistic to the amount of braking a car with excellent brakes and tires could provide. Since the amount of braking is further limited by the surface of the road, the vehicle cannot decelerate any faster than in the experiment.

While the logic for this has been implemented in the setup of the experiment and it is seen in Fig 11 that the preferred lane is proactively selected so that the decision is already made in case of an emergency. The preferred lane is the one adjacent to the one without the truck in it, but the vehicle dynamics are incapable of changing the lane or applying brakes any harder.

The tests at 25 m/s and above had similar results. In each case, the ego vehicle crashed head-on at different speeds as listed in the following table.

Test Speed (m/s)	Final speed at impact. (m/s)
25	14
30	19
36	27

Table 3. Final Impact speeds.

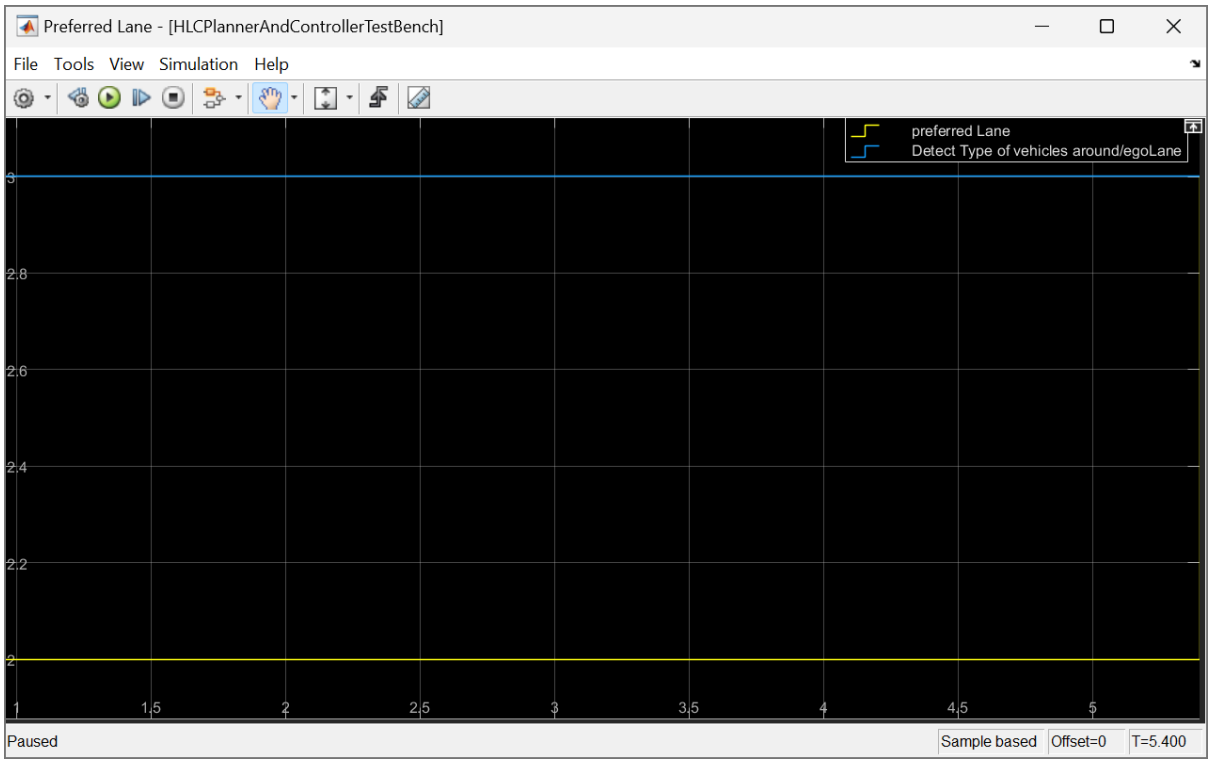

Fig 32. Preferred Lane. 36 m/s.

25 m/s Test:

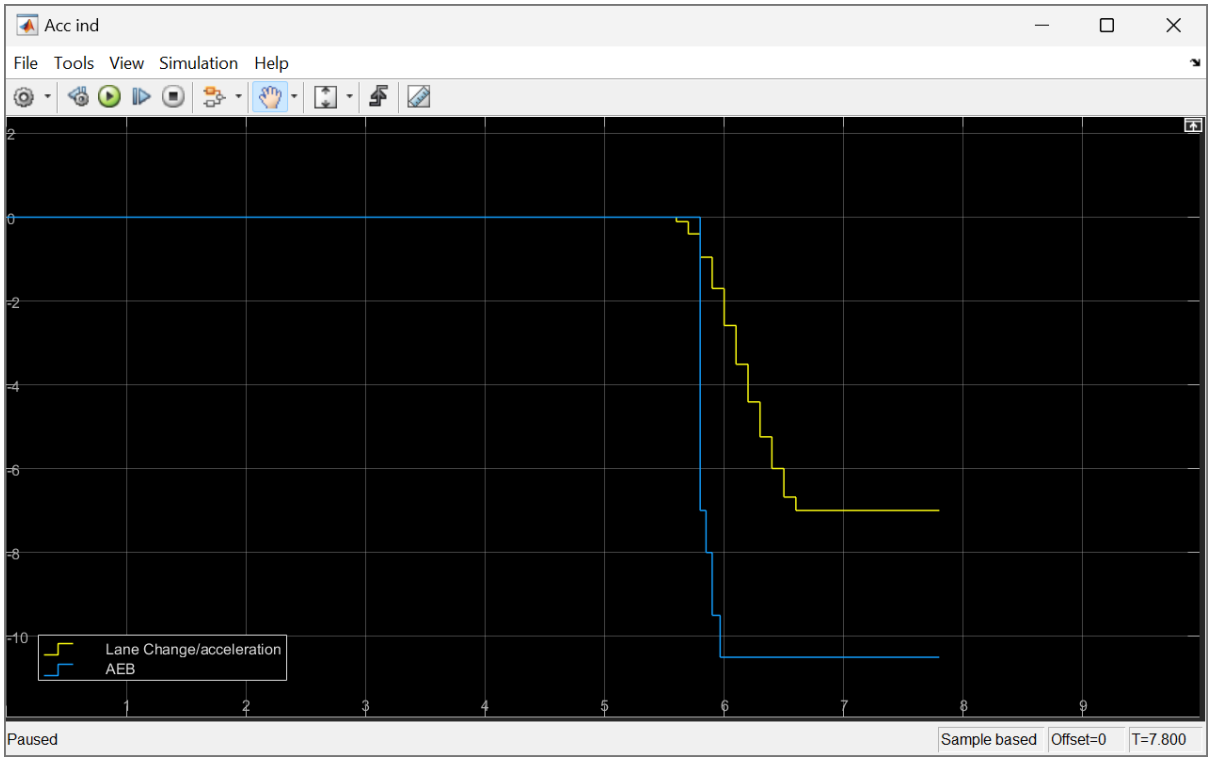

Fig 33. Individual accelerations from each subsystem. 25 m/s.

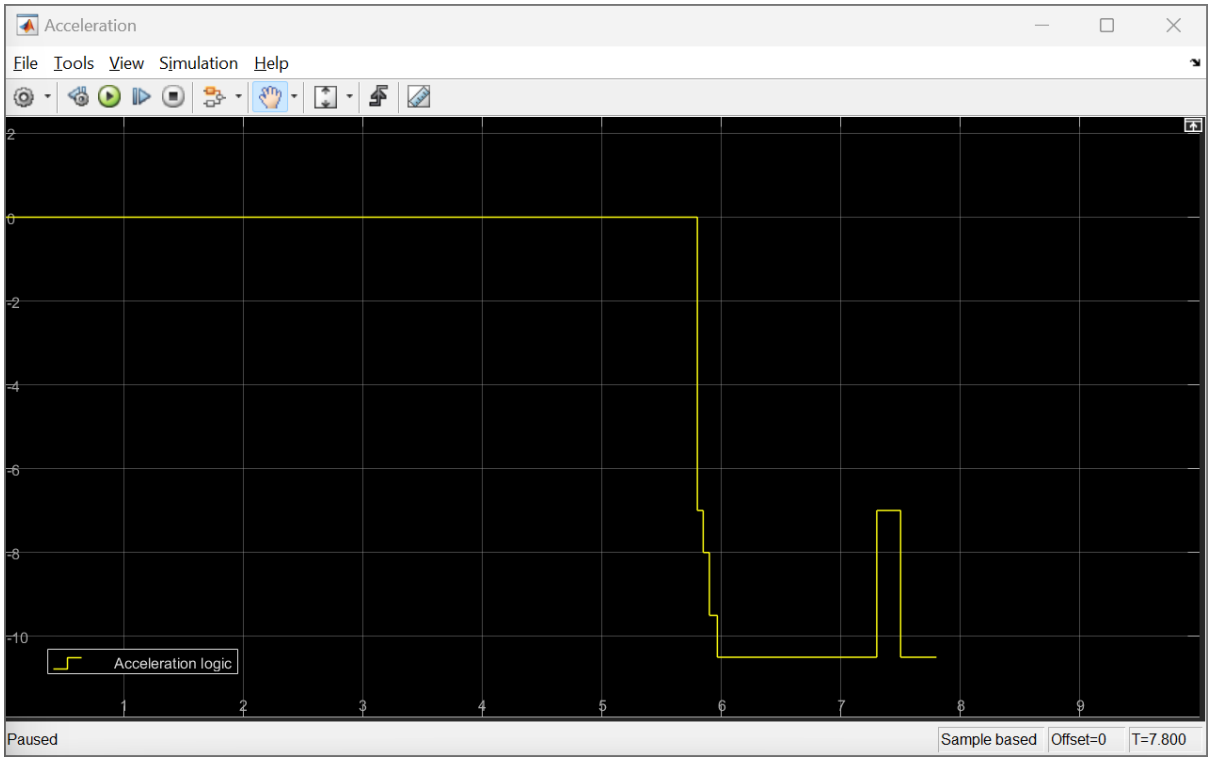

Fig 34. Actual acceleration after applying logic to switch between the two systems. 25 m/s.

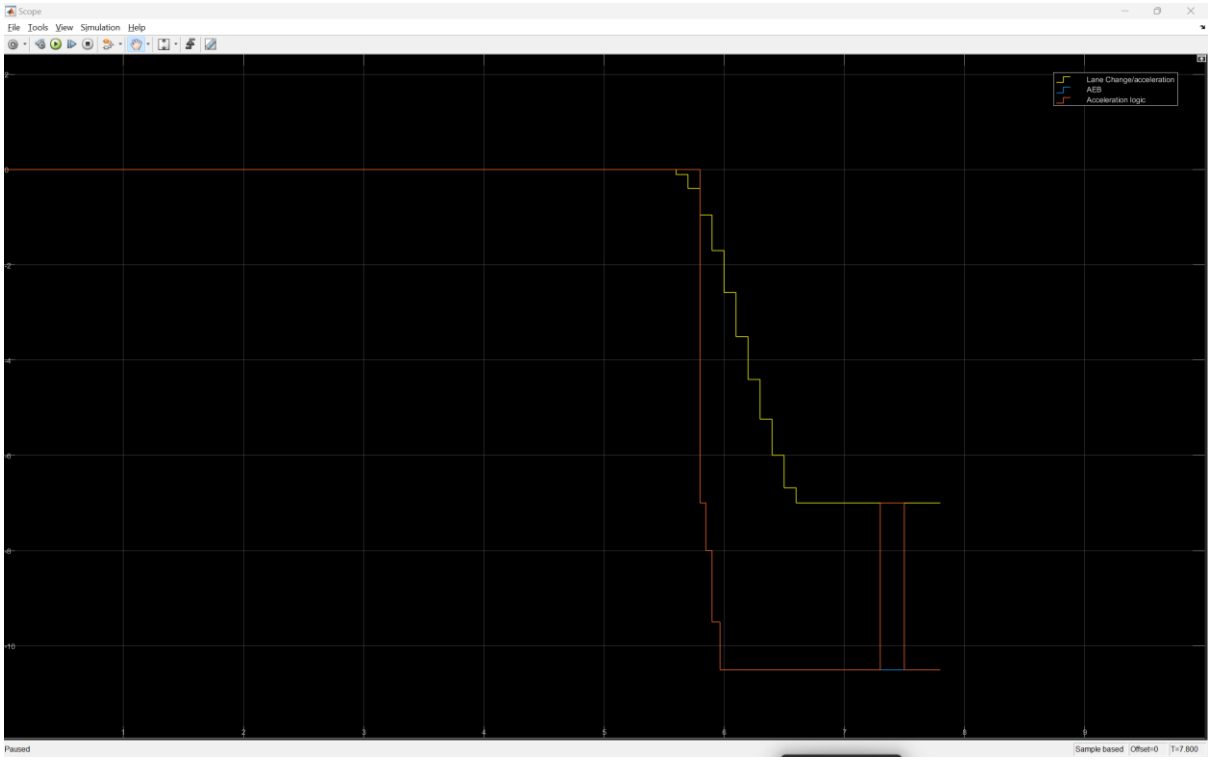

Fig 35. Fig 33, Fig 34 Overlap. 25 m/s.

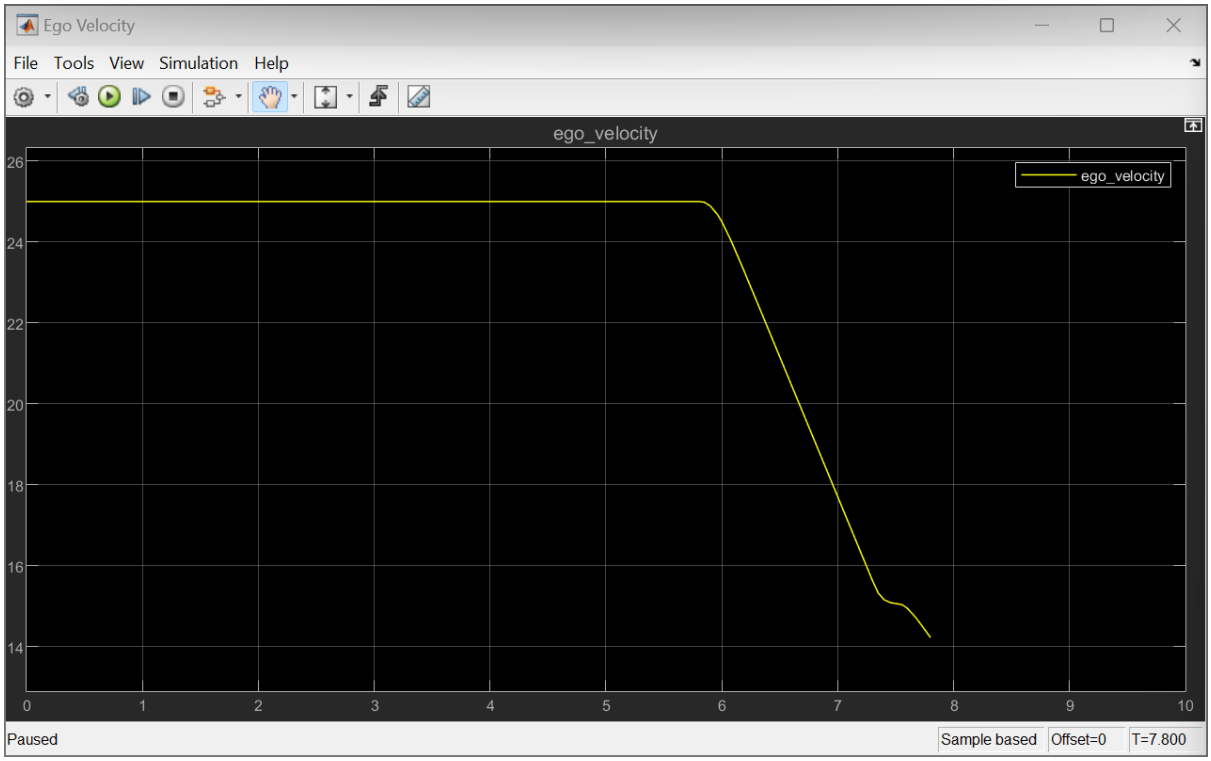

Fig 36. Ego Velocity. 25 m/s.

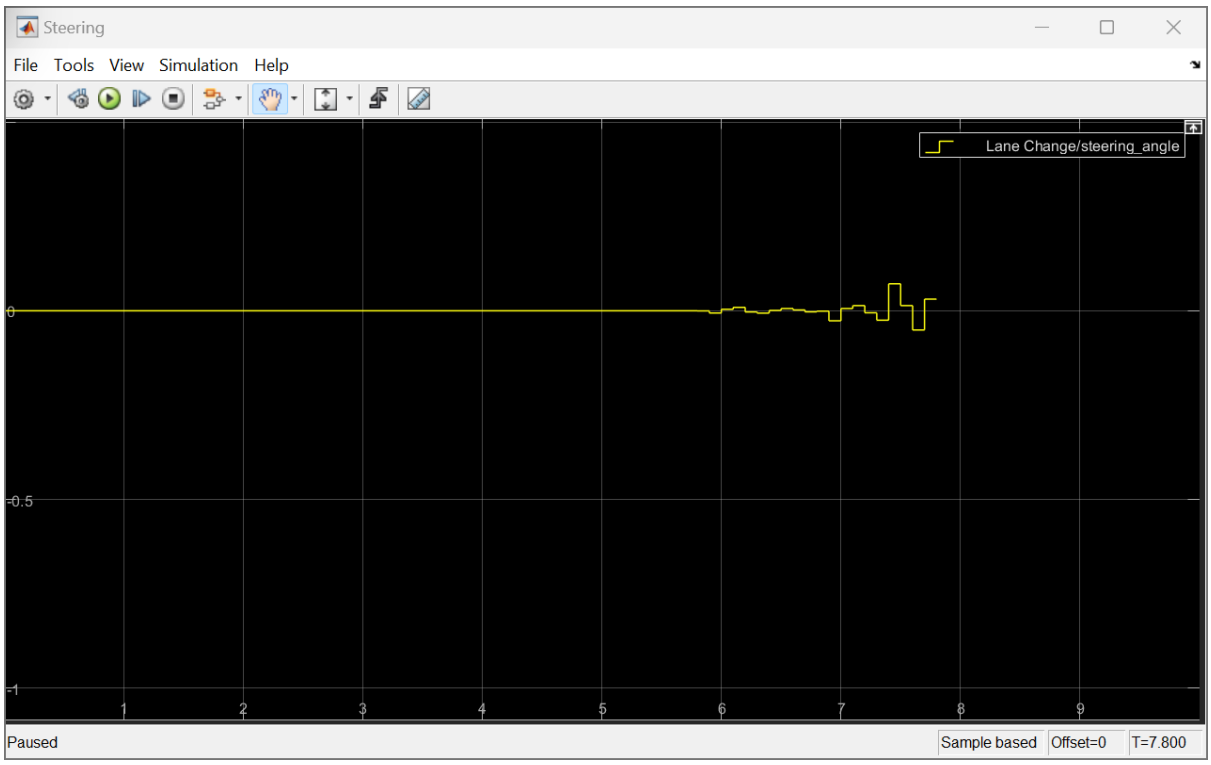

Fig 37. Steering Inputs. 25 m/s.

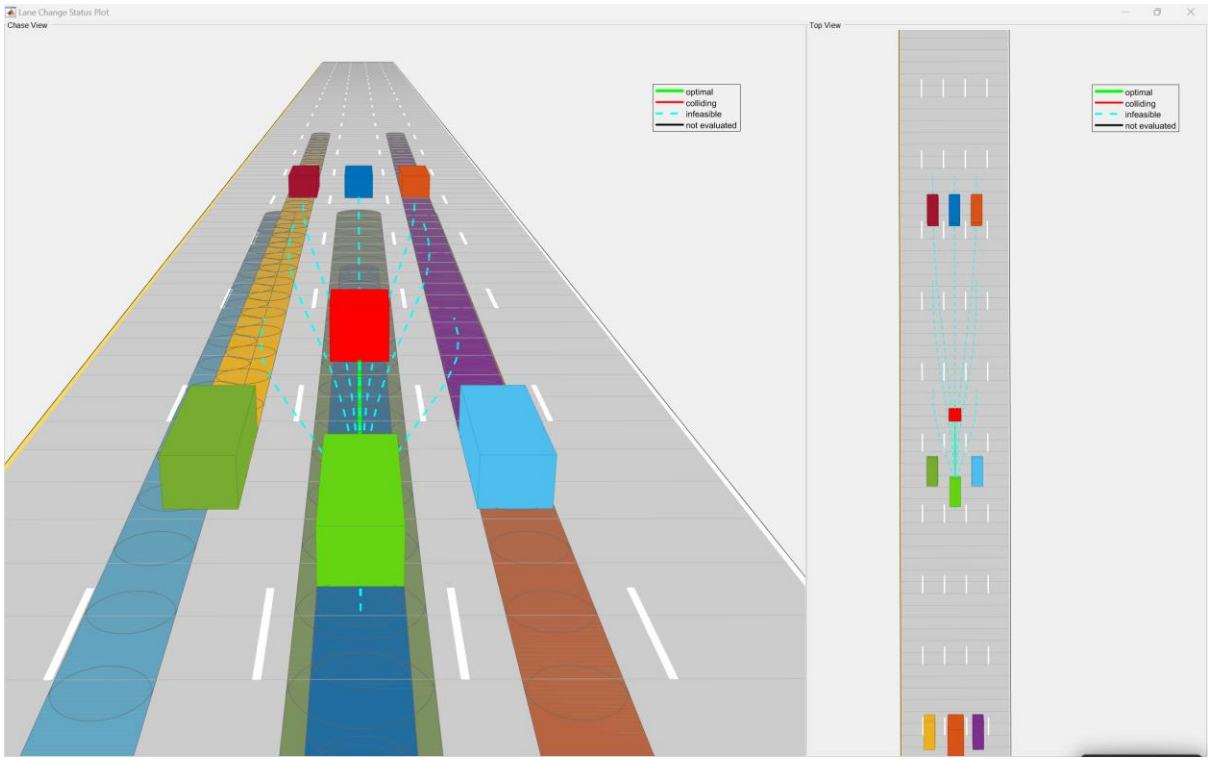

Fig 38. Ego Vehicle cannot find any collision free trajectories and head-on collision is safer. 25 m/s.

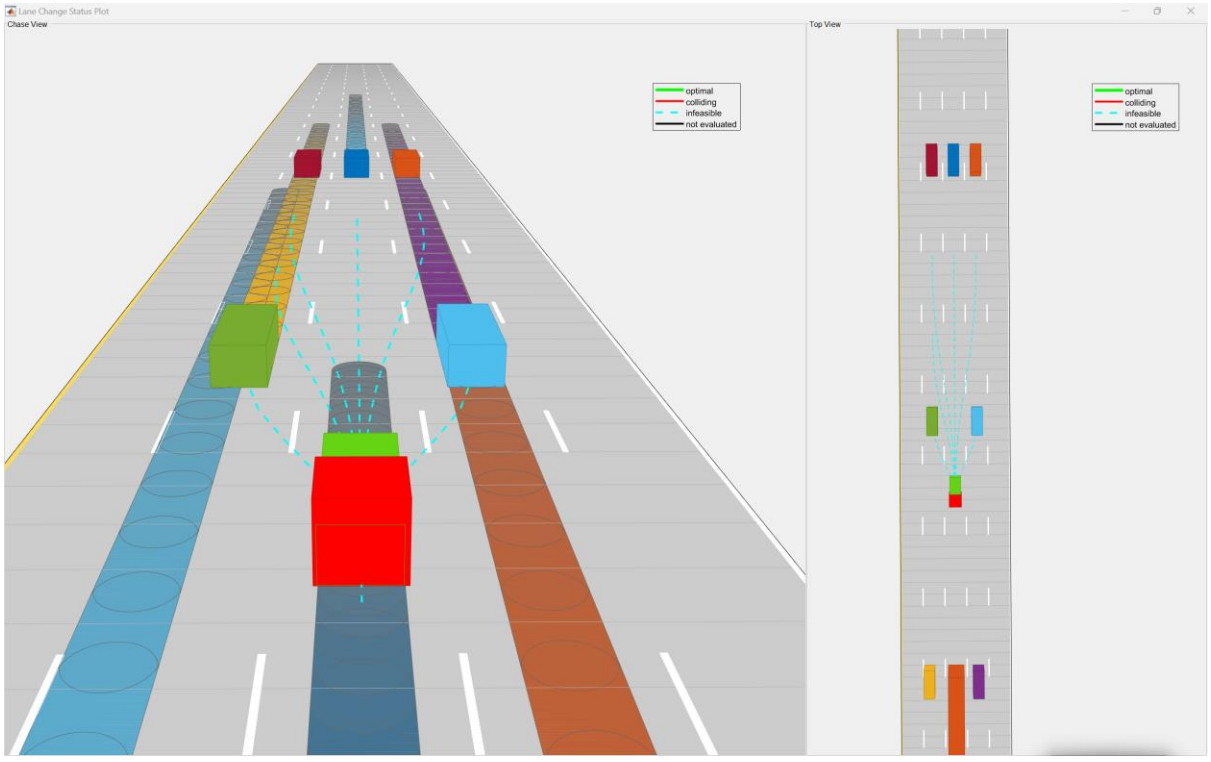

Fig 39. Ego Vehicle collides with the obstacle at 14 m/s.

30 m/s Test:

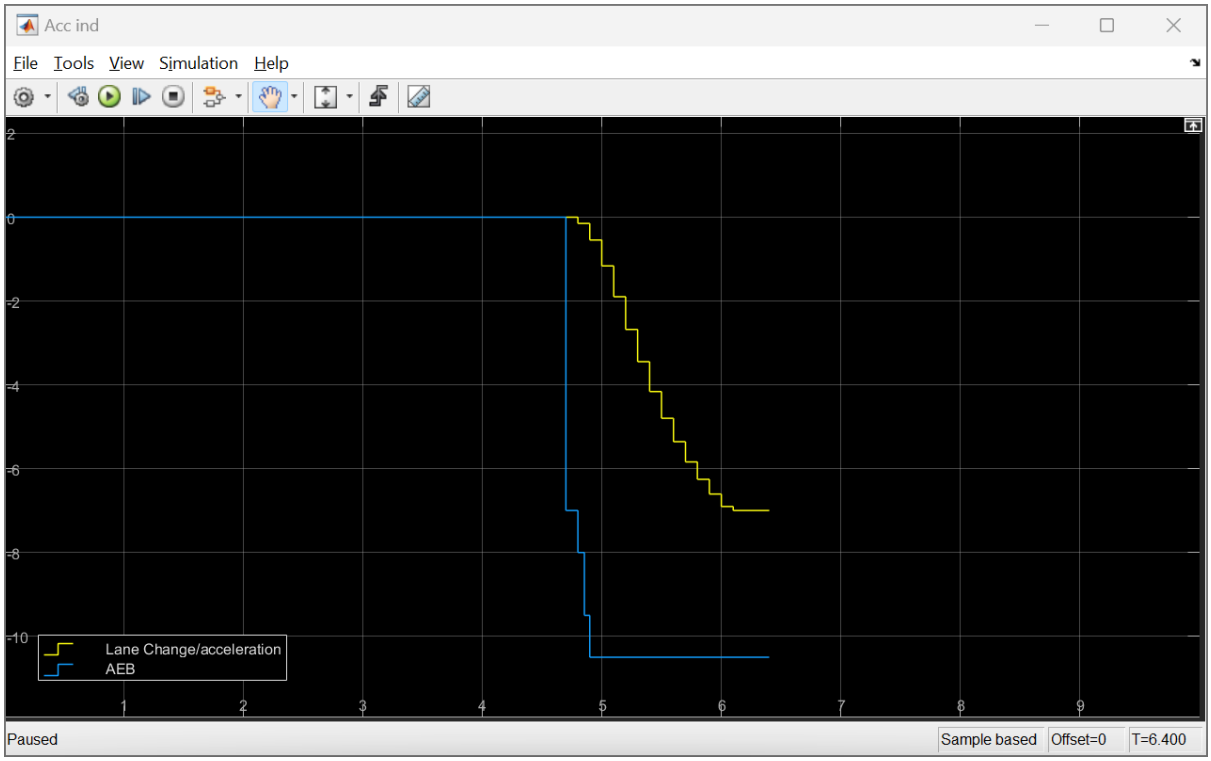

Fig 40. Individual accelerations from each subsystem. 30 m/s.

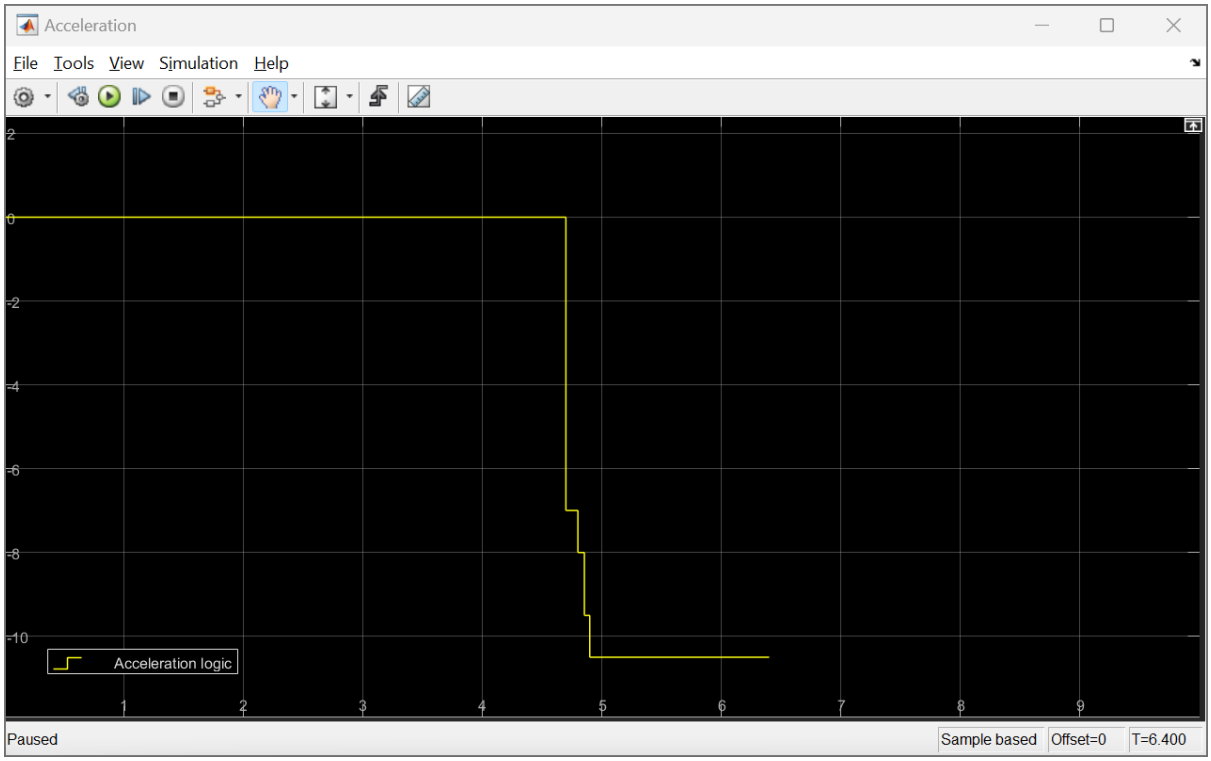

Fig 41. Actual acceleration after applying logic to switch between the two systems. 30 m/s.

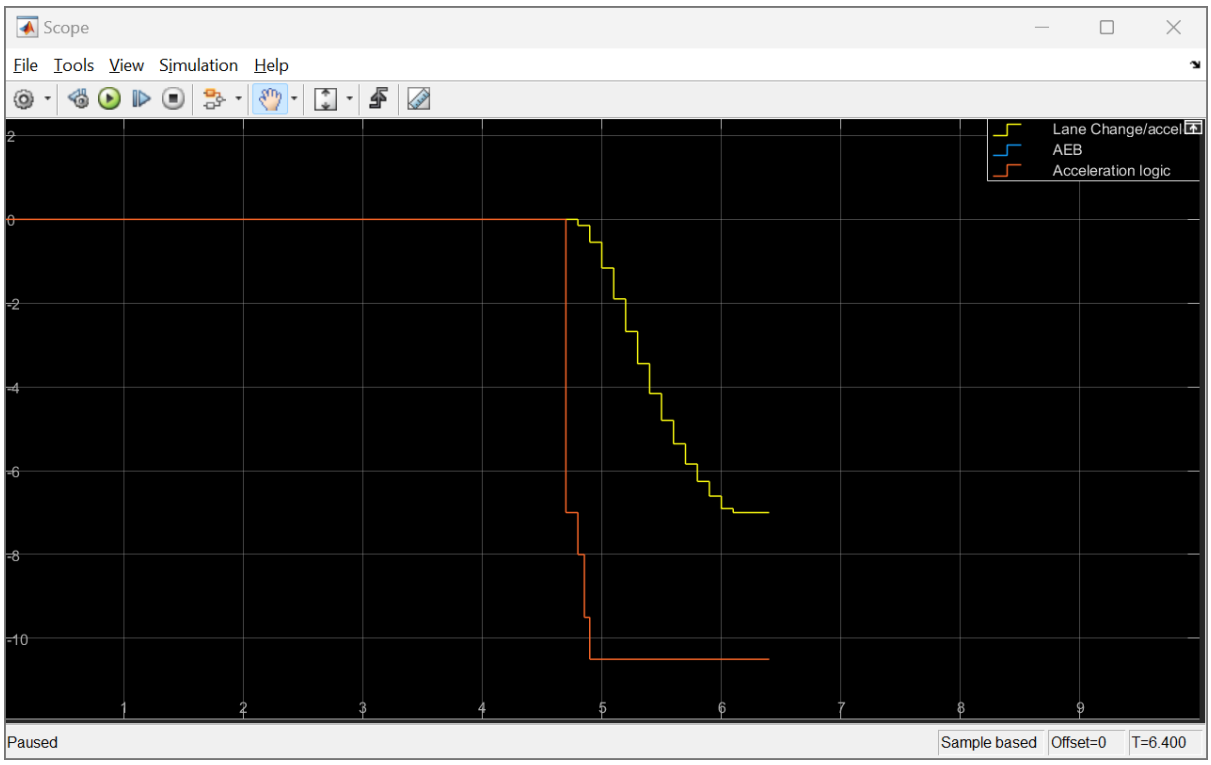

Fig 42. Fig 40, Fig 41 Overlap. 30 m/s.

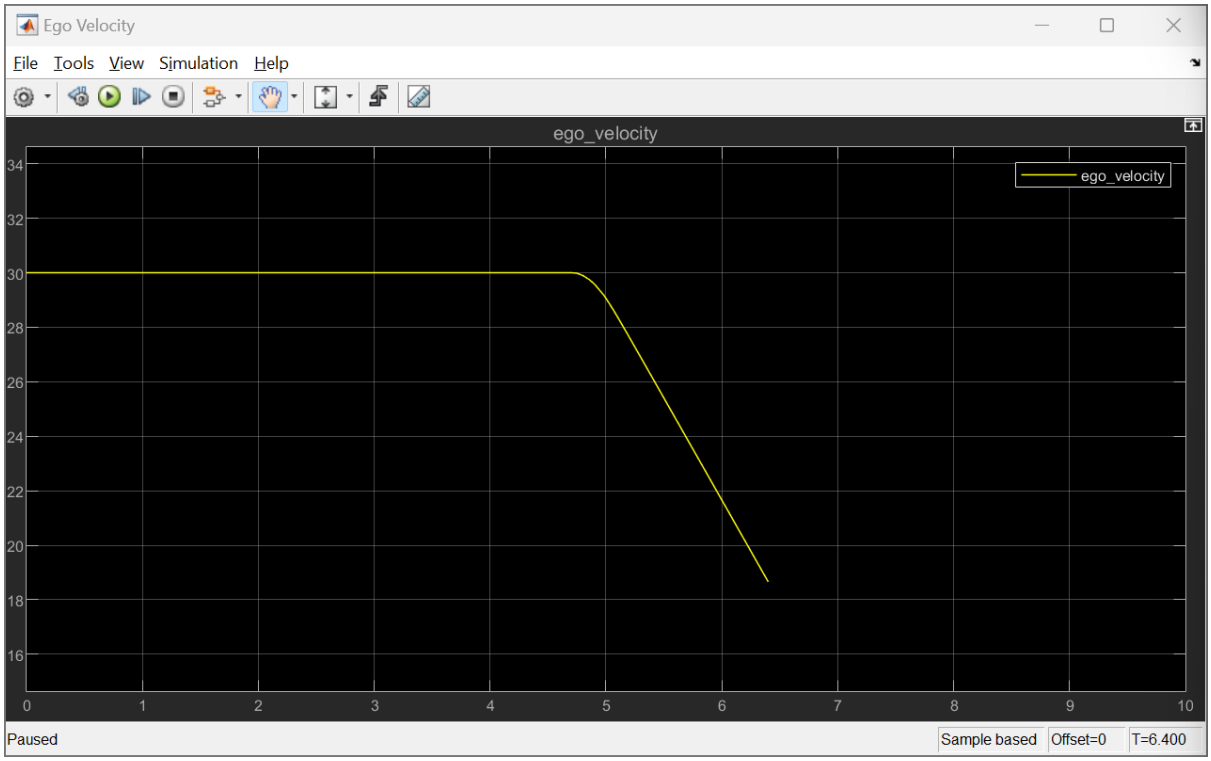

Fig 43. Ego Velocity. 30 m/s.

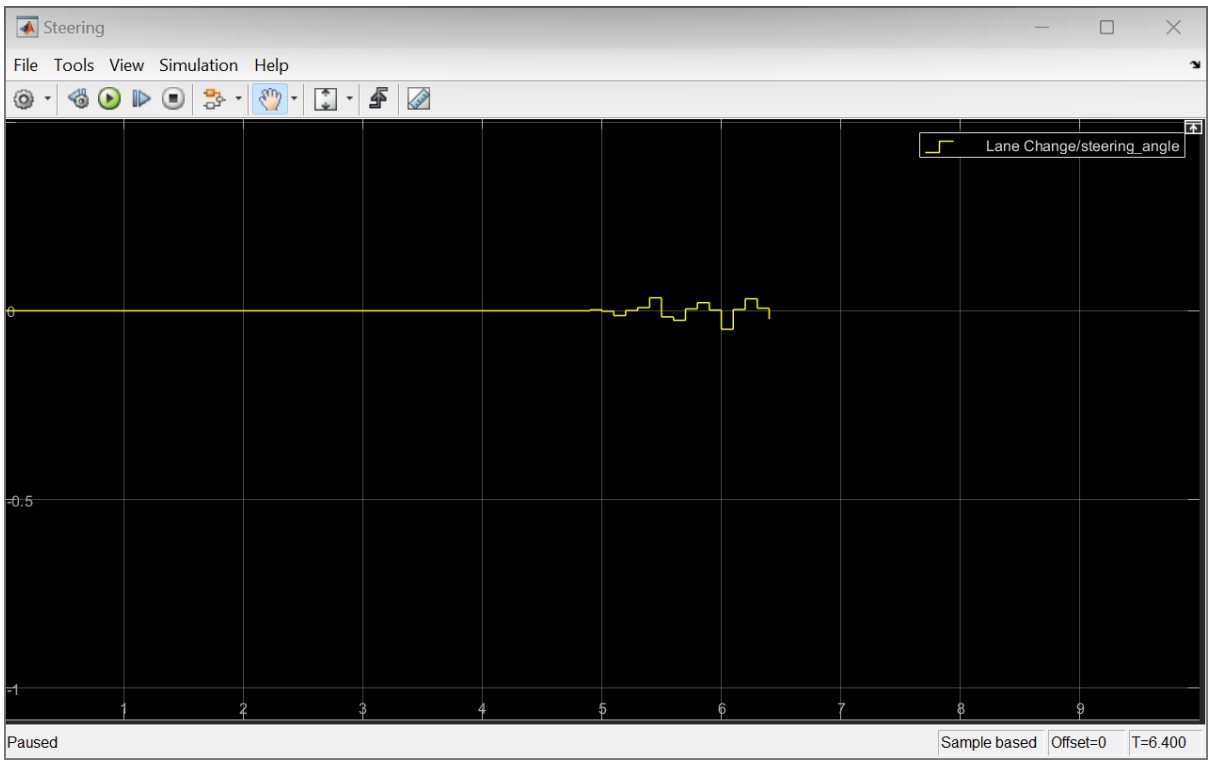

Fig 44. Steering Inputs. 30 m/s.

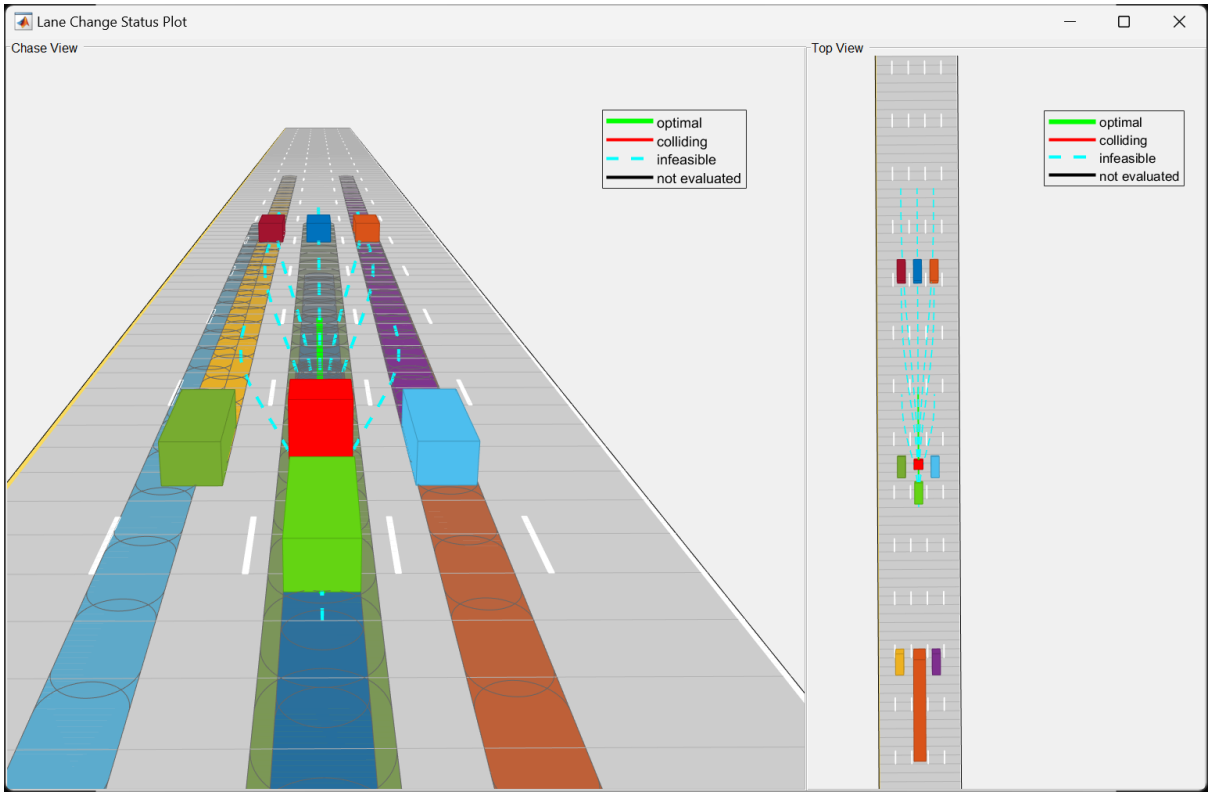

Fig 45. Ego Vehicle cannot find any collision free trajectories and head-on collision is safer.

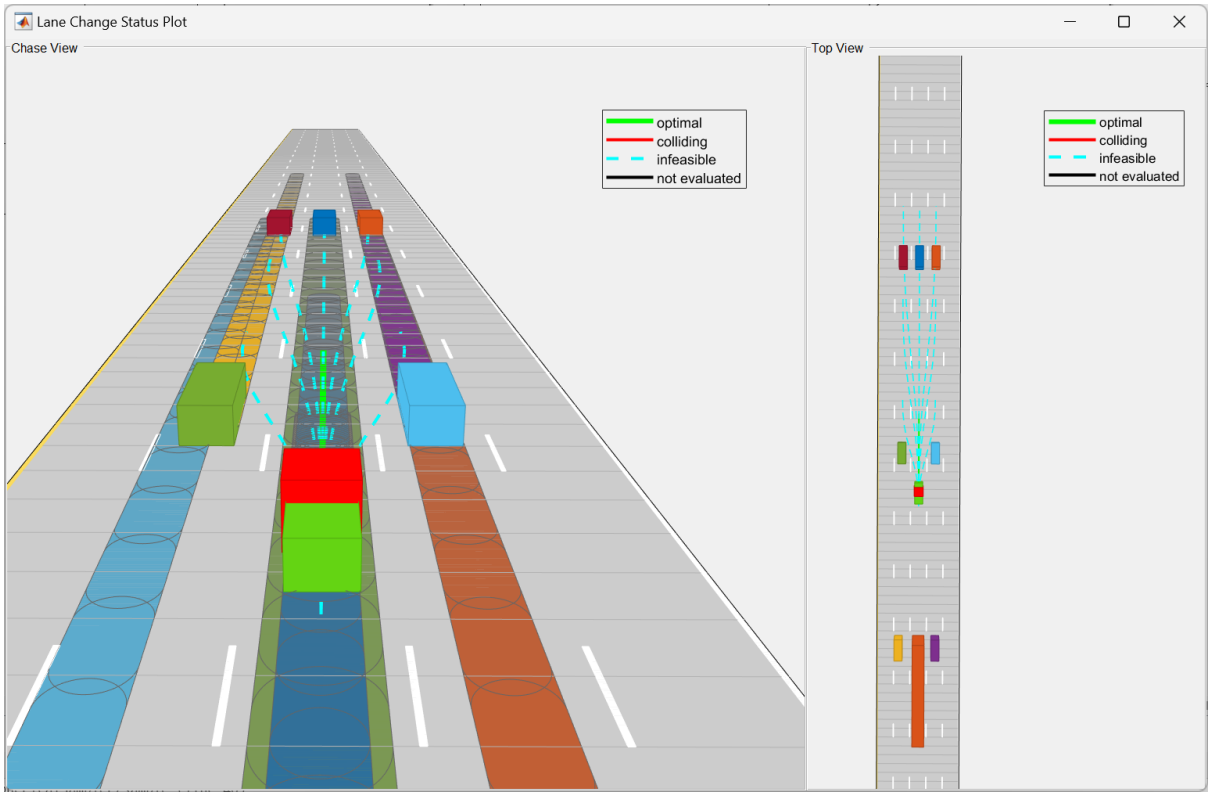

Fig 46. Ego Vehicle collides with the obstacle at 19 m/s.

36 m/s Test:

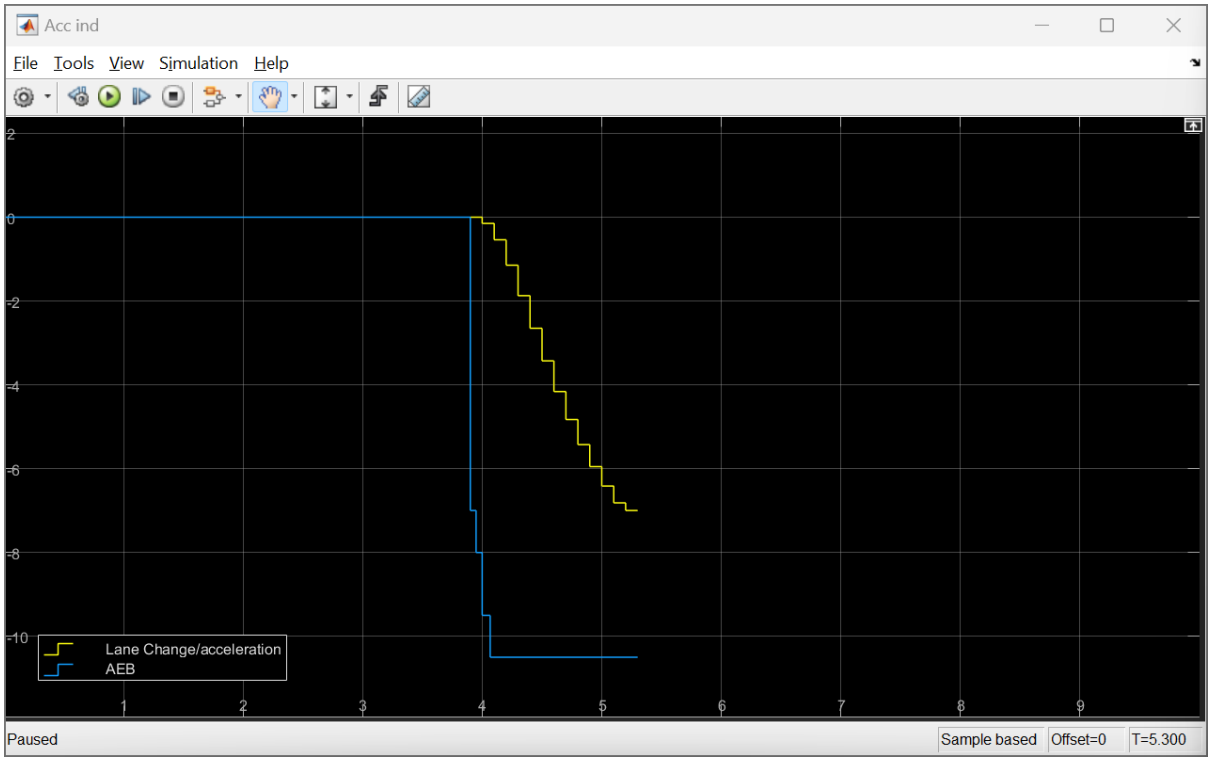

Fig 47. Individual accelerations from each subsystem. 36 m/s.

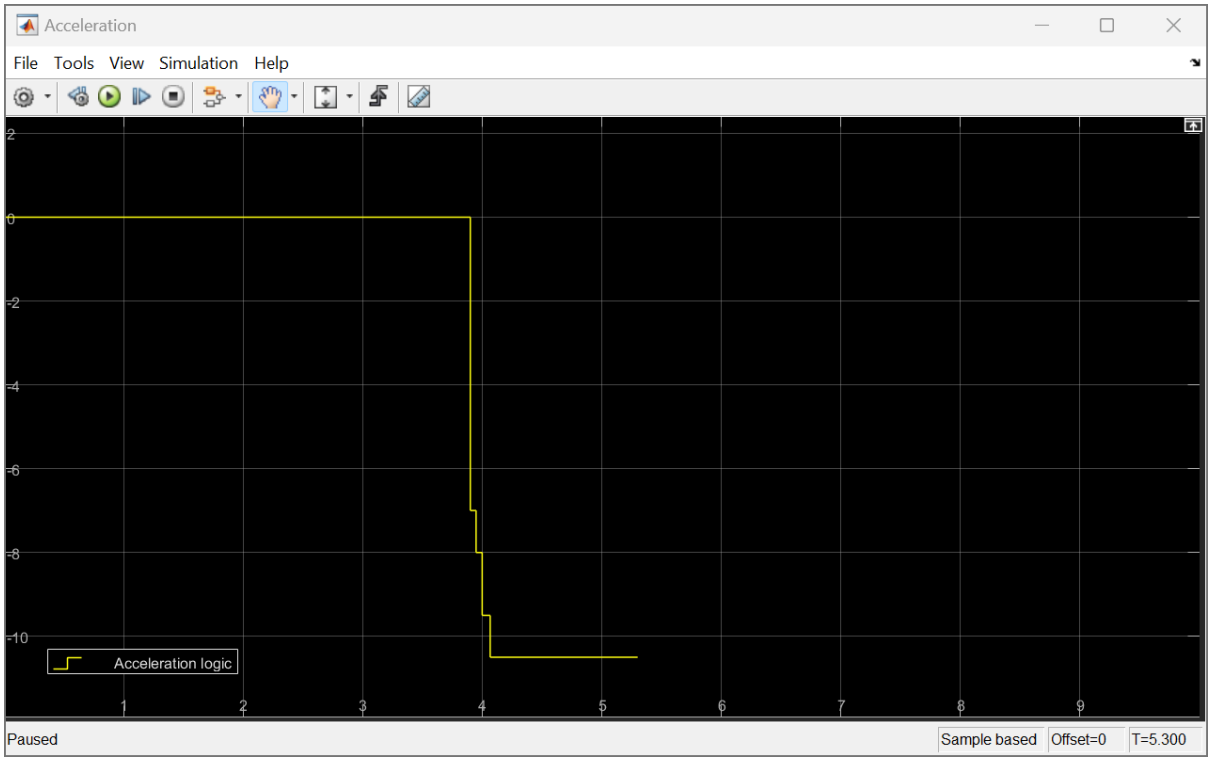

Fig 48. Actual acceleration after applying logic to switch between the two systems. 36 m/s.

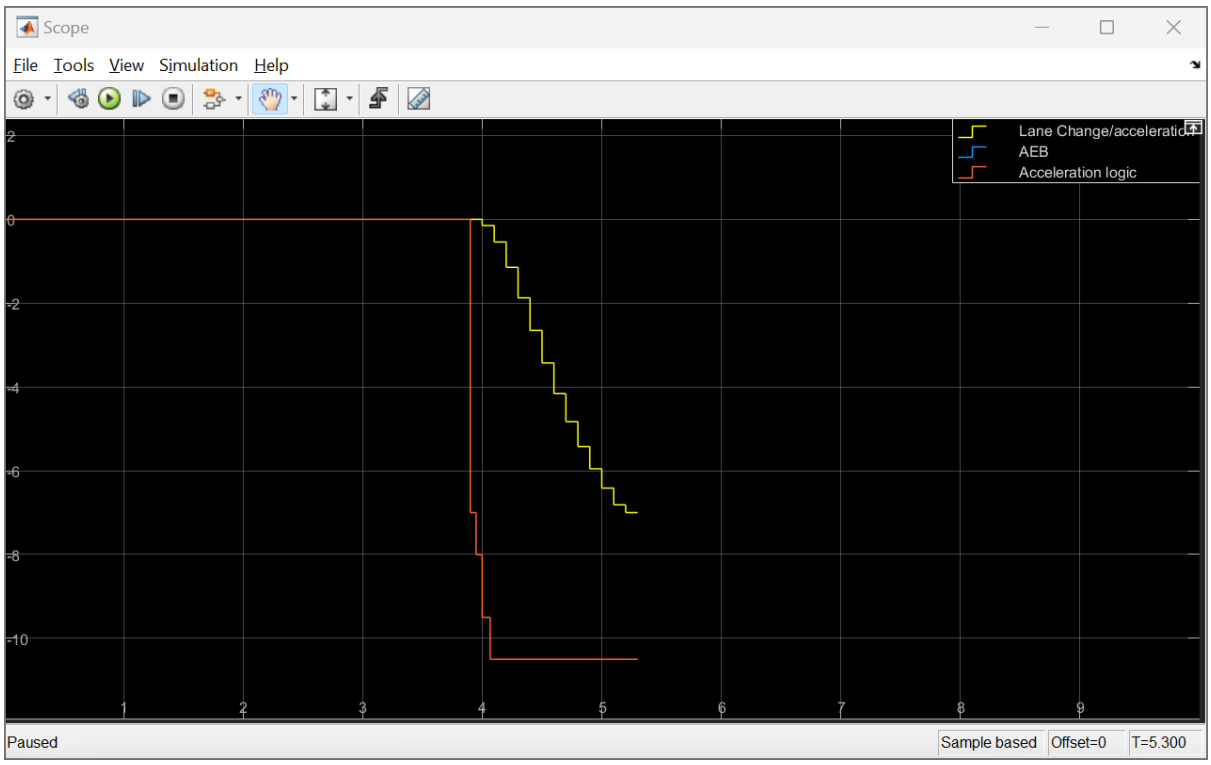

Fig 49. Fig 47, Fig 48 Overlap. 36 m/s.

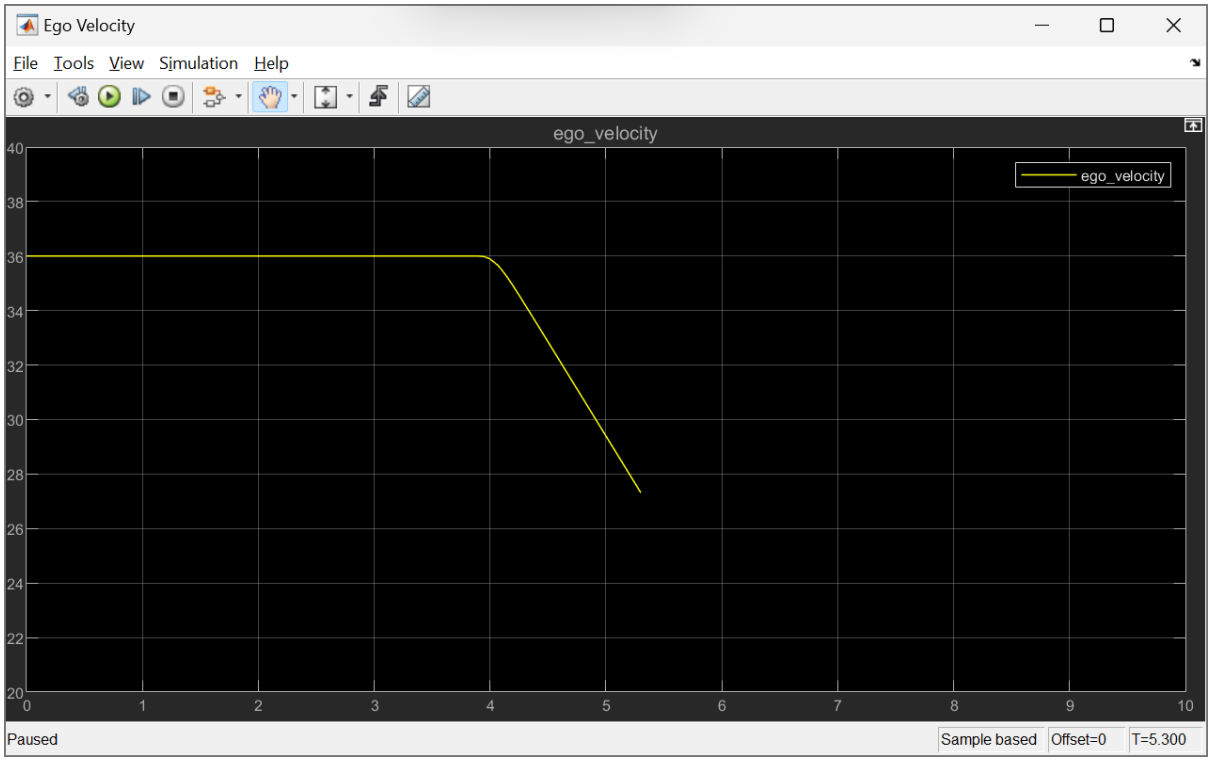

Fig 50. Ego Velocity. 36 m/s.

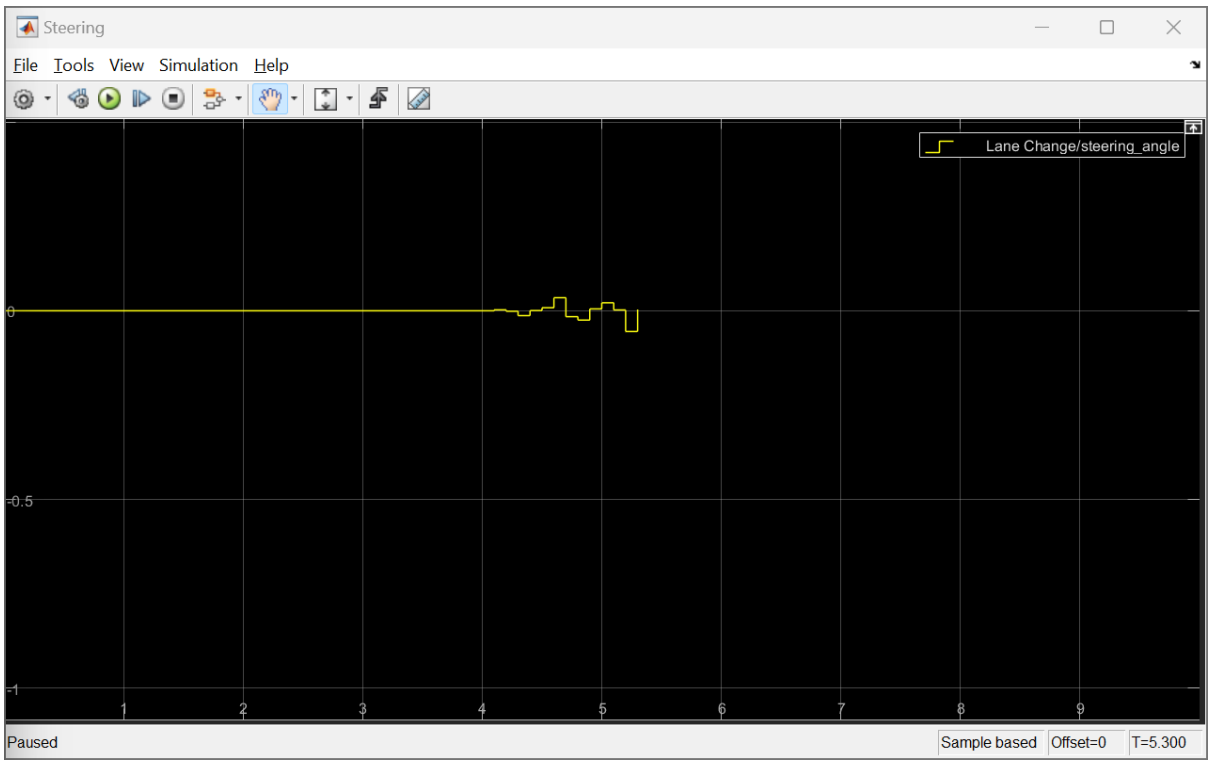

Fig 51. Steering Inputs. 36 m/s.

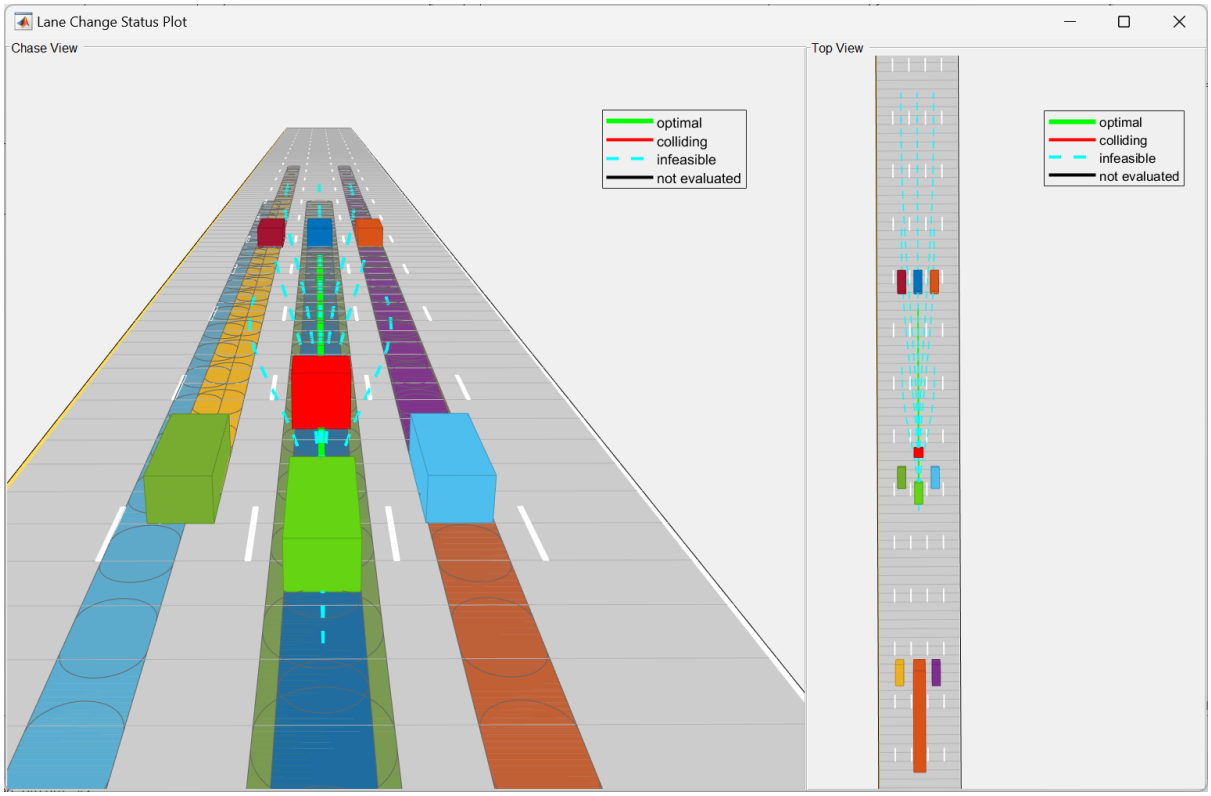

Fig 52. Ego Vehicle cannot find any collision free trajectories and head-on collision is safer.

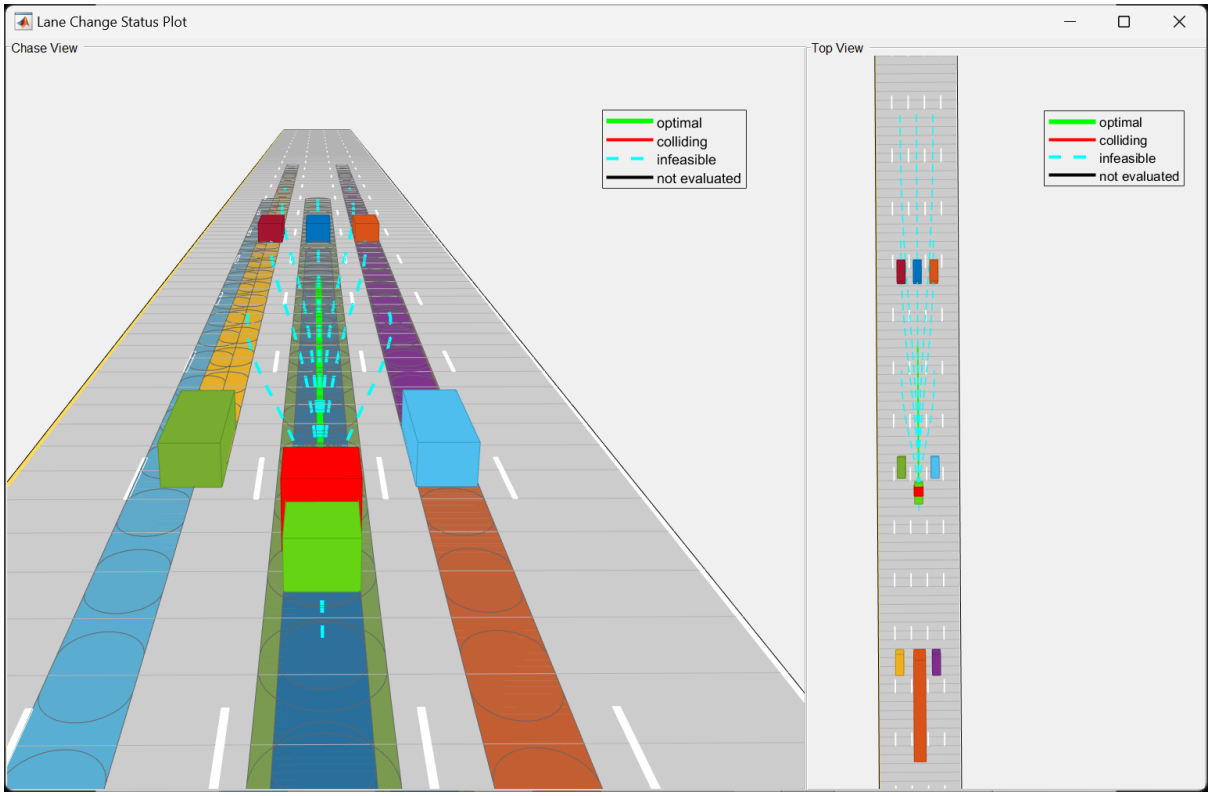

Fig 53. Ego Vehicle collides with the obstacle at 27 m/s.

36 m/s with an additional 25 m separation Test:

After conducting previous tests, a question arises, how much extra separation does the ego vehicle need to safely maneuver around the obstacle. For this slight modification to the scenario is made, the gap between ego vehicle and the row of vehicles in front is increased incrementally, but the distance between ego vehicle and vehicles behind is kept the same at 39 m.

Incrementally increasing the distance between each run, it was found that at about 25 m additional gap, making the total gap 64 m, the ego vehicle has enough room to apply emergency brakes and maneuver around the obstacle. The ego vehicle chooses the lane with no truck in it and moves to the shoulder.

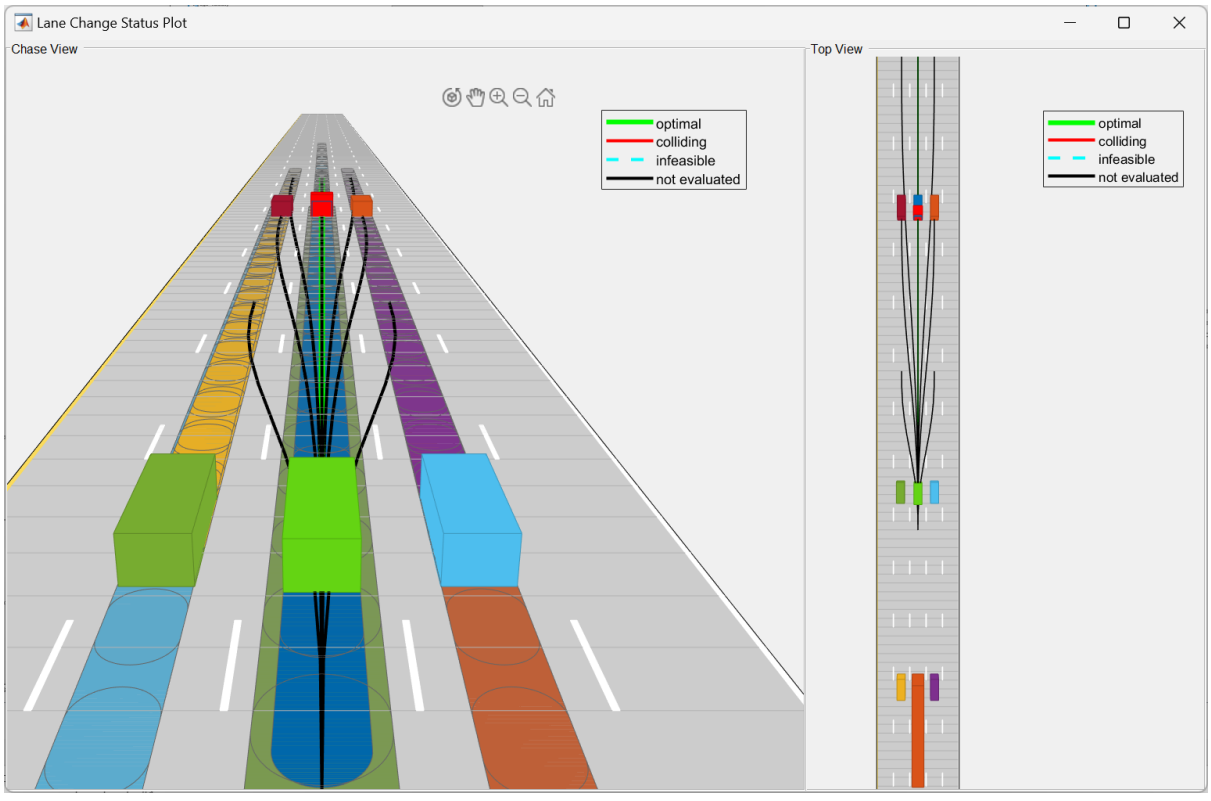

Fig 54. Modified Scenario with additional 25 m gap between ego vehicle and obstacle.

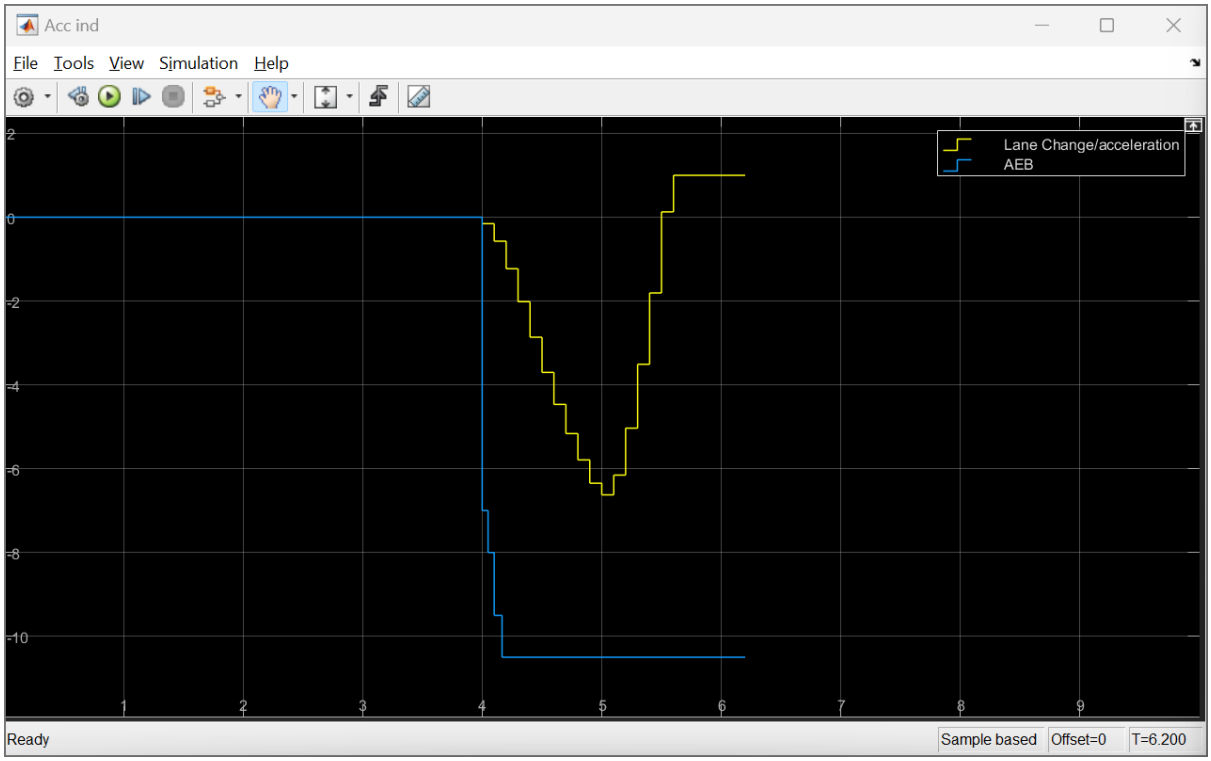

Fig 55. Individual accelerations from each subsystem. Modified Scenario.

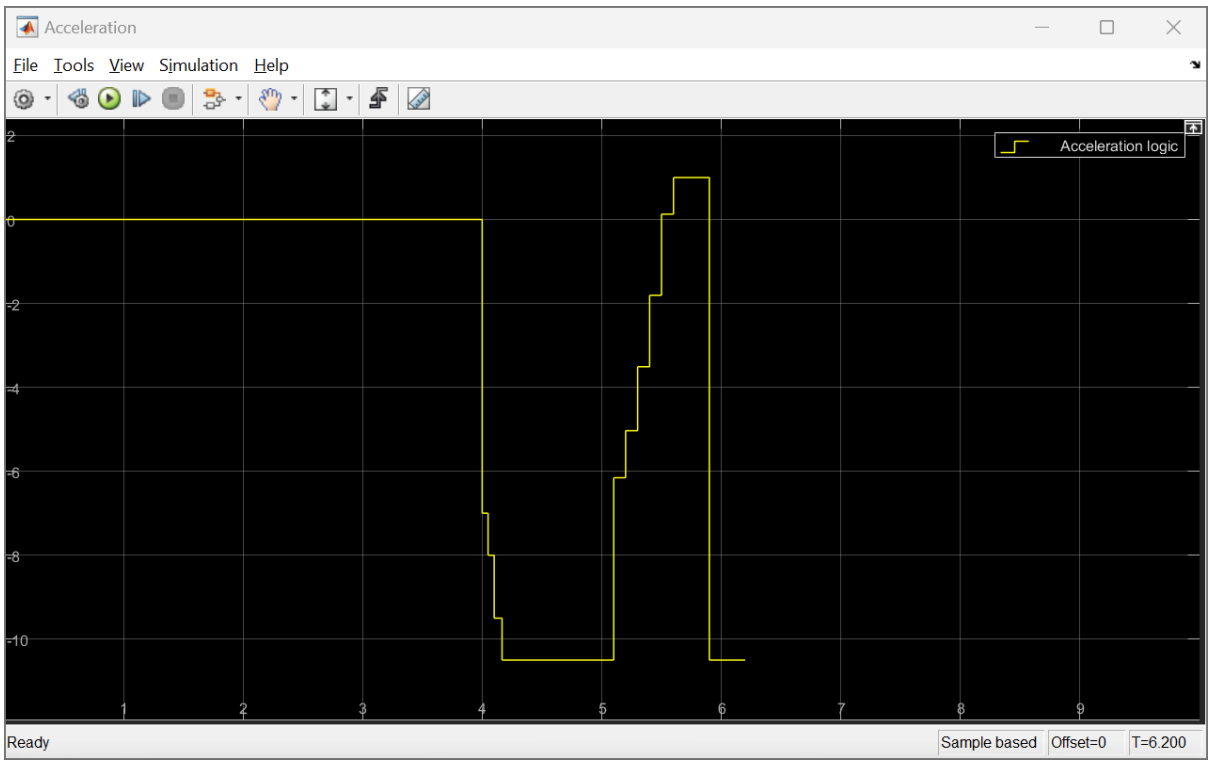

Fig 56. Actual acceleration after applying logic to switch between the two systems. Modified Scenario.

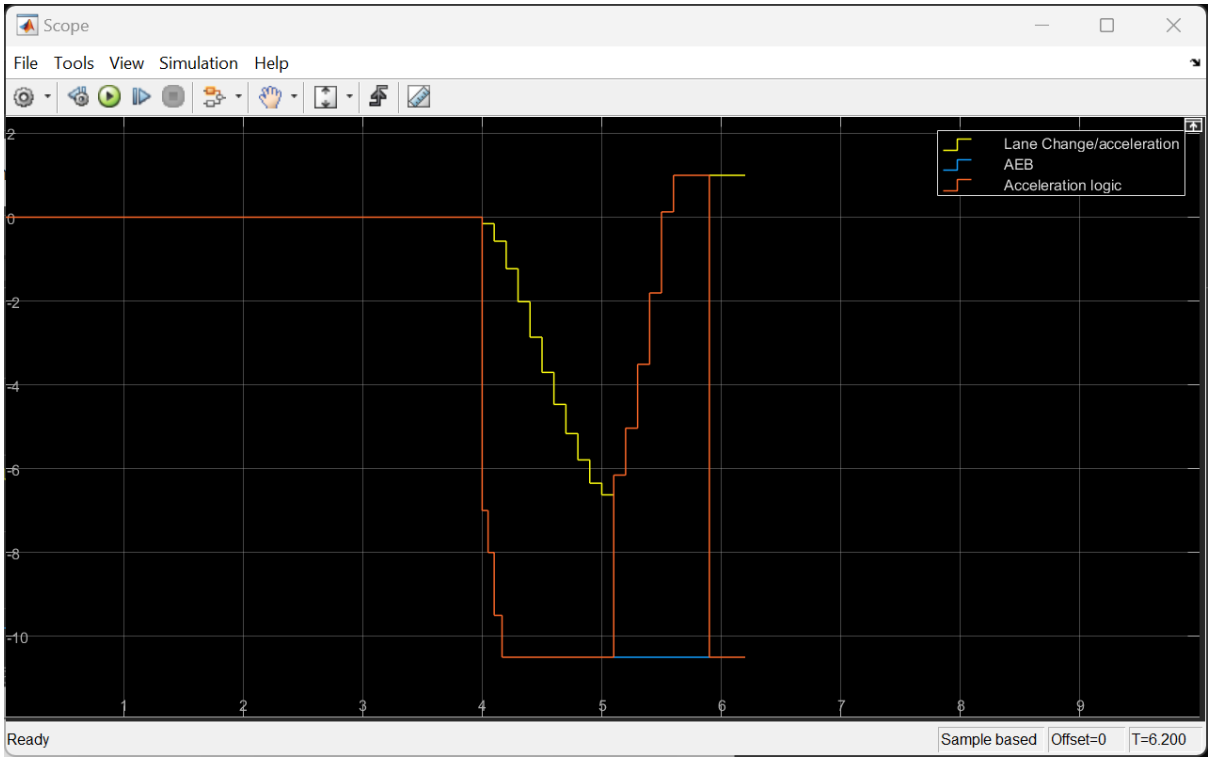

Fig 57. Fig 55, Fig 56 Overlap. Modified Scenario.

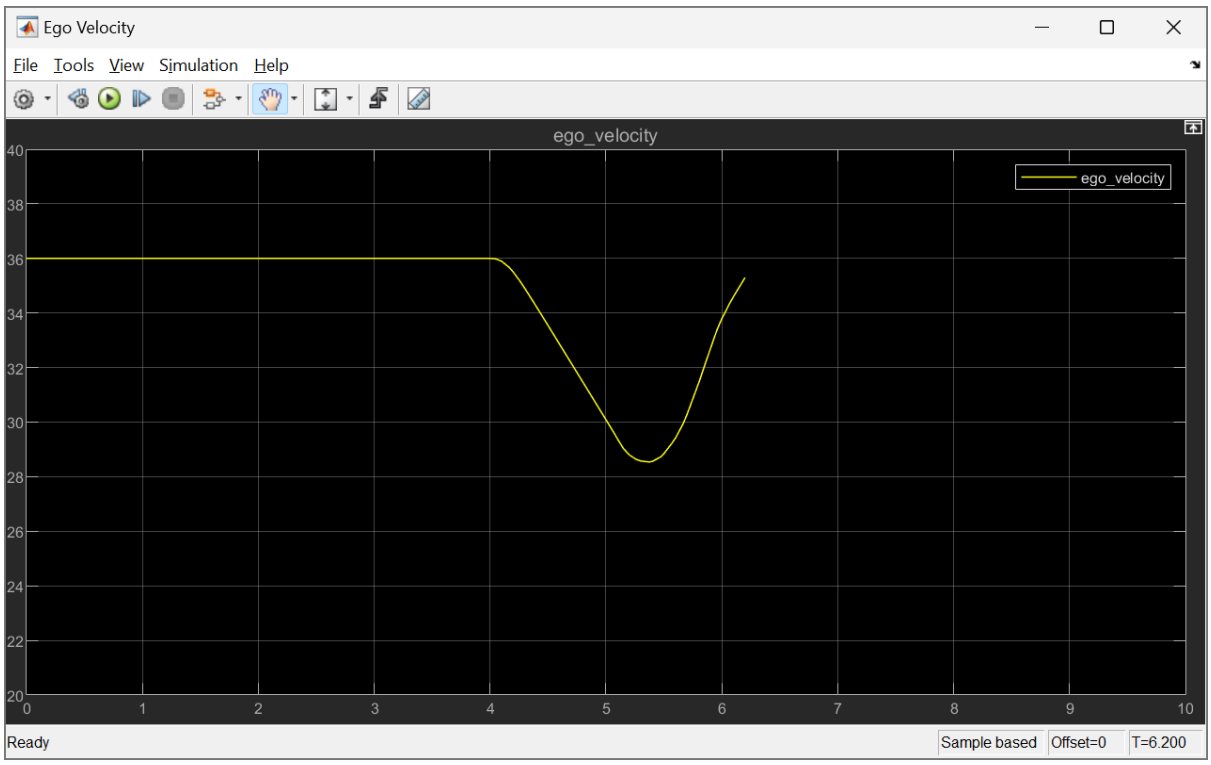

Fig 58. Ego Velocity. Modified Scenario.

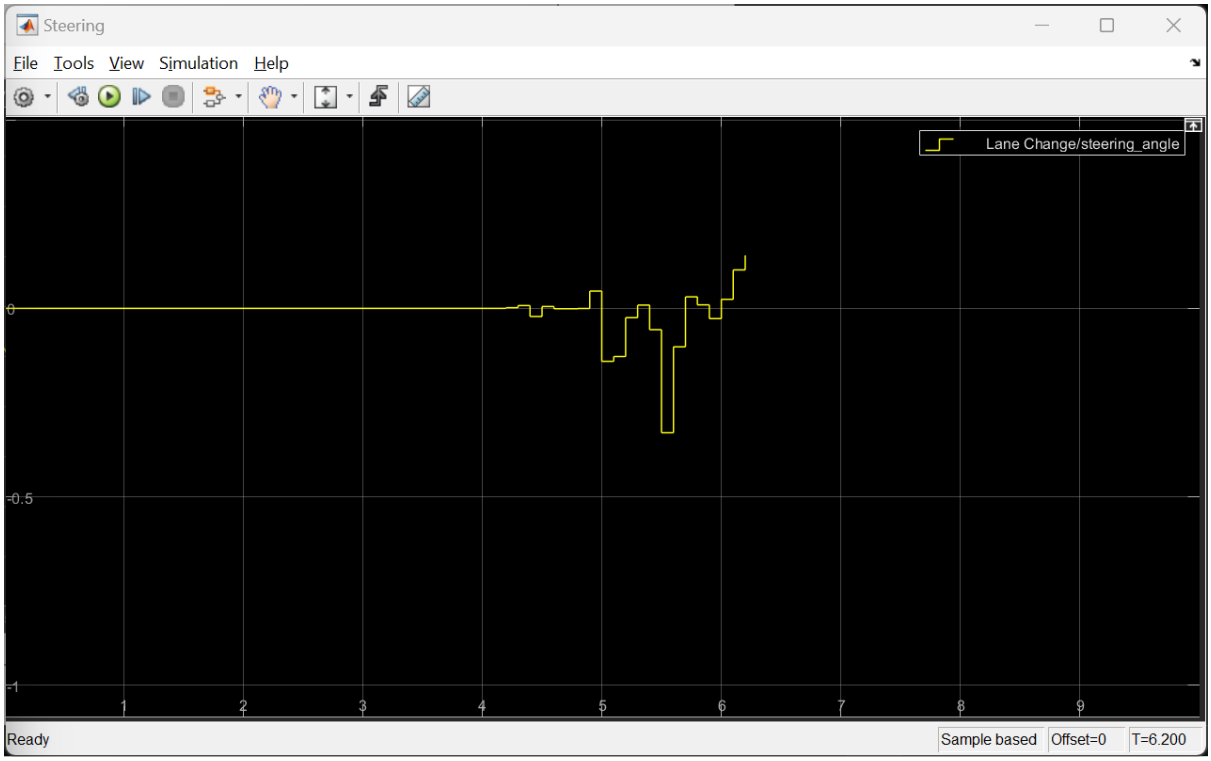

Fig 59. Steering Inputs. Modified Scenario.

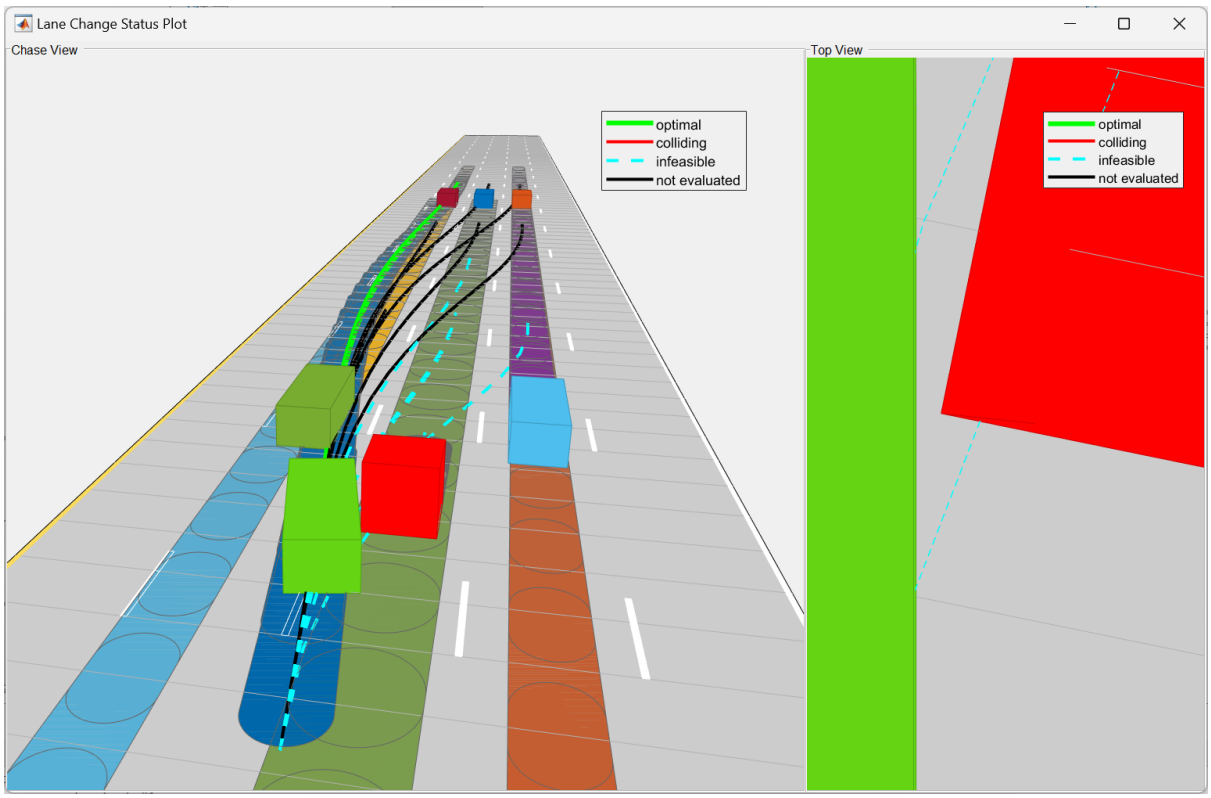

Fig 60. Ego vehicle avoids collision. Modified Scenario.

Even with the extra separation, the gap of 64 m is far less than the safe allowable gap [15] that should be maintained at 36 m/s (~ 85 mph). The Ego vehicle is still able to safely avoid all collisions.

Conclusion:

This project intended to expand the reach of the study in the field of autonomous driving. It specifically focused on understanding the response an autonomous vehicle equipped with all the latest accident mitigation strategies encounters the challenge of making a split-second decision to avoid an obstacle that suddenly appears in front of it. An additional challenge to this problem was the introduction of very high speed, very less gap between two vehicles, and a boxed in situation for the Ego vehicle.

Another novel question answered by this project was that, if it was necessary to implement a logic into the system that considers the size of the vehicle behind the Ego vehicle in decision making of whether to attempt to change lanes or hit the obstacle. Given the less-than-optimal gap between vehicles, it is necessary to understand if a large vehicle like a semi-truck, would have enough space when the Ego vehicle is making emergency maneuvers.

After designing the implementation for Emergency Braking and Lane Changing in Simulink, the test is run at different speeds ranging from 20 m/s to 36 m/s. In these tests, the longitudinal gap between each vehicle is 39 m. It was found that at speeds greater than 20 m/s the Ego vehicle does not have enough space to bring the vehicle down to a controllable speed and find a slot in the other lanes without colliding with anything else and causing more secondary accidents.

The logic implemented to identify the type of vehicle behind the Ego vehicle successfully selects the adjacent lane with no semi-truck in it as the preferred lane to be in, in case of an emergency. During the test at 20 m/s, the Ego vehicle slows down enough and avoids colliding with the

obstacle. In this test it was observed that the truck, which in the simulation is not programmed to decelerate, does not come close enough to the Ego vehicle as it safely moves out of the truck's way. But due to lack of space in adjacent lane in tests at 25 m/s and above, the Ego vehicle stays in its lane and crashes head-on with the obstacle.

To find what was the required gap to safely avoid the obstacle at 36 m/s, the scenario was modified slightly to have the gap between Ego vehicle and the obstacle to have an additional gap. Incrementally increasing the gap between the Ego vehicle and the obstacle, it was found that an additional gap of about 25 m gives the Ego vehicle just enough space to safely maneuver around the obstacle. In this test, the gap between the semi-truck and the Ego vehicle is kept at the original 39 m and the truck still has enough space that it does not collide with the Ego vehicle.

In all tests, without the logic for changing the lane when there is a truck behind the Ego vehicle, it tends to reduce the speed to a relatively safe speed before crashing into the obstacle head-on as that is safer than pulling off a maneuver that leave little room for an incorrect input. Therefore, it is a key addition to the system especially if there is enough space for the Ego vehicle. In cases where it is impossible to find space, it makes no difference to the result of the test.

In the future, the scope of this study can be expanded to include more scenarios such as a scenario where there is a truck in only one of the rear adjacent lanes, truck in two rear adjacent lanes, and trucks in all the rear lanes. The current system should be able to handle those situations to relative degree, but some parametric changes could be necessary to further accommodate those type of scenarios.

References

[1] NHTSA, “Standing General Order on Crash Reporting | NHTSA,” *www.nhtsa.gov*. https://www.nhtsa.gov/laws-regulations/standing-general-order-crash-reporting

[2] Tesla, “Tesla Vehicle Production & Deliveries and Date for Financial Results & Webcast for Fourth Quarter 2023 | Tesla Investor Relations,” *ir.tesla.com*, 2024. https://ir.tesla.com/press-release/tesla-vehicle-production-deliveries-and-date-financial-results-webcast-fourth-quarter-2023

[3] Waymo, “Waymo significantly outperforms comparable human benchmarks over 7+ million miles of rider-only driving,” *Waymo*, Dec. 20, 2023. https://waymo.com/blog/2023/12/waymo-significantly-outperforms-comparable-human-benchmarks-over-7-million/

[4] M. Khayatian *et al.*, “Cooperative Driving of Connected Autonomous Vehicles Using Responsibility-Sensitive Safety (RSS) Rules.” Accessed: Apr. 10, 2024. [Online]. Available: https://par.nsf.gov/servlets/purl/10321994#:~:text=RSS%20rules%20(two%20CAVs%20driving

[5] R. Gallen, N. Hautiere, A. Cord, and S. Glaser, “Supporting Drivers in Keeping Safe Speed in Adverse Weather Conditions by Mitigating the Risk Level,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 4, pp. 1558–1571, Dec. 2013, doi: https://doi.org/10.1109/tits.2013.2262523.

[6] C. Ragland and G. Dalrymple, “Overlap car-to-car Tests Compared to car-to-half Barrier and car-to-full Barrier Tests,” *Proceedings of the 13th International Technical Conference on Experimental Safety Vehicles (ESV)*, vol. 1, no. 2, Jan. 1991.

[7] S. Acierno, R. Kaufman, F. P. Rivara, D. C. Grossman, and C. Mock, “Vehicle mismatch: Injury Patterns and Severity,” *Accident Analysis & Prevention*, vol. 36, no. 5, pp. 761–772, Sep. 2004, doi: https://doi.org/10.1016/j.aap.2003.07.001.

[8] M. T. Wolf and J. W. Burdick, “Artificial potential functions for highway driving with collision avoidance,” *IEEE Xplore*, May 01, 2008. https://ieeexplore.ieee.org/document/4543783?arnumber=4543783 (accessed Dec. 14, 2020).

[9] C. Sun and A. Eskandarian, “A Predictive Frontal and Oblique Collision Mitigation System for Autonomous Vehicles,” *ASME Letters in Dynamic Systems and Control*, pp. 1–5, Mar. 2021, doi: https://doi.org/10.1115/1.4050502.

[10] “TRAFFIC SAFETY FACTS Crash • Stats Early Estimate of Motor Vehicle Traffic Fatalities in 2022,” Apr. 2023. Available: https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/813428

[11] “azdot.gov/mvd Arizona Driver License Manual and CUSTOMER SERVICE GUIDE,” Oct. 2023. Available: https://apps.azdot.gov/files/mvd/mvd-forms-lib/99-0117.pdf

[12] H. Wang, Y. Huang, A. Khajepour, Y. Zhang, Y. Rasekhipour, and D. Cao, “Crash Mitigation in Motion Planning for Autonomous Vehicles,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 9, pp. 3313–3323, Sep. 2019, doi: https://doi.org/10.1109/tits.2018.2873921.

[13] “Highway Lane Change Planner and Controller - MATLAB & Simulink,” *www.mathworks.com*. https://www.mathworks.com/help/driving/ug/highway-lane-change-planner-and-controller.html (accessed Apr. 10, 2024).

[14] “Autonomous Emergency Braking with Sensor Fusion - MATLAB & Simulink,” *www.mathworks.com*. https://www.mathworks.com/help/driving/ug/autonomous-emergency-braking-with-sensor-fusion.html

[15] “Vehicle Stopping Distance and Time.” Available: https://nacto.org/docs/usdg/vehicle_stopping_distance_and_time_upenn.pdf