

University of Stuttgart
Institute of Industrial Automation and
Software Engineering

**This thesis is
confidential.**

Hardware-in-the-loop testing of commercial ADAS systems

Research Project No. FA-3286

Submitted at the University of Stuttgart by
Saiprasad Salkar

INFOTECH

Examiner: Prof.Dr.-Ing. Dr. h.c. Michael Weyrich

Supervisor: Alexander Schuster

20.04.2022



Table of Contents

TABLE OF CONTENTS.....	II
TABLE OF FIGURES	IV
TABLE OF TABLES	VII
TABLE OF ABBREVIATIONS.....	VIII
GLOSSARY	IX
ABSTRACT	X
1 INTRODUCTION.....	11
1.1 Literature Review	11
1.1.1 Necessity to test the ADAS system	11
1.1.2 How to test the ADAS system	15
1.2 Significance.....	15
1.3 Scope of the Thesis	15
2 TEST DESIGN TECHNIQUES.....	16
2.1 Systems Theoretical Analysis of Processes.....	17
2.2 Failure Mode and Effect Analysis.....	17
2.3 Fault Tree Analysis	18
2.4 Real Incidents	19
2.5 Responsibility-Sensitive Safety Model	19
2.5.1 Implementing RSS.....	19
3 TEST CASE CREATION	23
3.1 RoboTestWebApp.....	24
3.1.1 Description of Use Cases	25
4 TEST ENVIRONMENT	29
4.1 Validation Methods.....	29
4.1.1 Virtual Environment Simulation	29
4.1.2 X-in-the-loop Simulation	29
4.2 Hardware-in-the-loop Environment Setup	31
4.2.1 Camera Setup.....	31
4.2.2 CAN Communication Setup.....	32
4.3 Simulator Setup.....	34
4.4 Display	35
4.5 DC Power Supply.....	36

5	EVALUATION.....	37
5.1	Threats to Validity	37
5.1.1	Types of Threats to Validity:	37
5.2	Warning Detection.....	39
5.3	Results	41
5.3.1	Test case and result description	41
6	CONCLUSION	46
6.1	Future Work	46
6.1.1	More Test Cases	46
6.1.2	Automation.....	46
	APPENDIX.....	47
	Mobileye 6 Main Unit	47
	Mobileye 6 EyeWatch	47
	Mobileye 6 EyeCAN.....	47
	Mobileye CANSee.....	48
	Mobileye 6 Installation	48
	Camera Installation	49
	LGSVL Installation	53
	BIBLIOGRAPHY.....	56
	DECLARATION OF COMPLIANCE	60

Table of Figures

Figure 1: Pony.ai Autonomous Vehicle Crash.....	12
Figure 2: Tesla Crash Involving Overturned Truck.....	13
Figure 3: Tesla Crash with No Driver	13
Figure 4: Safe Longitudinal Distance.....	20
Figure 5: Response for Red Car when following Blue Car	20
Figure 6: One Way Traffic Scenarios	21
Figure 7: Cut-In and Drifting Scenarios	21
Figure 8: Cut-In and Drifting Scenario both Vehicles drifting same direction.....	21
Figure 9: Test Scenario Attributes	23
Figure 10: RoboTestWebApp Use-Case Diagram ^[4]	24
Figure 11: Sequence Diagram for “Register” ^[4]	25
Figure 12: Sequence Diagram “Login” ^[4]	25
Figure 13: Sequence Diagram “Manual Test Editor” ^[4]	26
Figure 14: Sequence Diagram “Run Test” ^[4]	27
Figure 15: RoboTestWebApp User Interface	27
Figure 16: SIL testing	29
Figure 17: Hardware-in-the-loop Process.....	31
Figure 18: HIL Setup	31
Figure 19: Vehicle Network Toolbox.....	32
Figure 20: Simulink Model to Send CAN signal.....	33
Figure 21: PCAN	33
Figure 22: DB9 Connector.....	34
Figure 23: LGSVL Simulator	34
Figure 24: LGSVL Simulator Environment Control Panel.....	35
Figure 25: Display Unit.....	36

Figure 26:DC Power Supply	36
Figure 27: HIL Environment Setup	36
Figure 28: CANSee User Interface.....	39
Figure 29: CAN Message 0x700	40
Figure 30: CAN Message-ID 0x700: Byte 2	40
Figure 31: CAN Message-ID 0x700: Byte 4	41
Figure 32: EyeWatch: Poor Visibility	42
Figure 33: CANSee: ID 0x700 (Byte 4: 0x80).....	42
Figure 34: Test Case Details on RoboTestWebApp: 100% Fog	43
Figure 35: LGSVL Simulator: 100% Fog	43
Figure 36: Ego Vehicle with Headlights Off	44
Figure 37: EyeWatch with No Warnings.....	44
Figure 38: Performance comparison of Mobileye 6 and MiniEye	45
Figure 39: Mobileye 6 Camera Sensor.....	47
Figure 40: Mobileye EyeWatch Display.....	47
Figure 41: Mobileye EyeCAN	48
Figure 42: Mobileye CANSee User Interface	48
Figure 43: Windshield.....	49
Figure 44: Mobileye Setup Wizard Login.....	50
Figure 45: Vehicle Selection.....	50
Figure 46: Vehicle Information and Measurements.....	51
Figure 47: Automatic Calibration Selection	51
Figure 48: Signal Source Selection	52
Figure 49: Automatic Calibration	52
Figure 50: LGSVL Simulator Link to Cloud.....	53
Figure 51: LGSVL Login.....	53
Figure 52: LGSVL New Cluster	54

Figure 53: LGSVL New Simulation.....	54
--------------------------------------	----

Table of Tables

Table 1: Threats to Internal Validity	37
Table 2: Threats to External Validity.....	38
Table 3: Test Case Result.....	41
Table 4: Mobileye 6 Features	55

Table of Abbreviations

ADAS	A dvanced D riving A ssistance S ystems
SAE	S ociety of A utomotive E ngineers
HIL	H ardware- I n-the- L oop
SIL	S oftware- I n-the- L oop
DIL	D river- I n-the- L oop
CIL	C amera- I n-the- L oop
LDW	L ane D eparture W arning
FCW	F ront C ollision W arning
CAN	C ontroller A rea N etwork
AV	A utonomous V ehicles
KPI	K ey P erformance I ndicator
UI	U ser I nterface
API	A pplication P rogramming I nterface
NHTSA	N ational H ighway T raffic S afety A dministration
RSS	R esponsibility- S ensitive S afety
AEB	A utomatic E mergency B raking

Glossary

LGSVL Simulator	SVL Simulator is an autonomous vehicle simulator designed to test and verify autonomous cars and robots throughout the development lifecycle.
AV software stack	Complete autonomous vehicle software embedded with Localization, Perception, Planning, Prediction, Routing and Control modules.
Django	An open-source Python-based web application development framework
Test Scenario	An aggregate of various concrete measurement values collectively constitutes a selected real-world or imaginary situation of surroundings wherein an autonomous vehicle needs to be tested.

Abstract

Autonomous Driving and Advanced Driver Assistance System is important technology in today's automotive world. It improves driving functionalities and reduce number of car accidents. Most of car manufacturers are targeting to develop semi-automated to fully automated vehicles. The complexity of system increases with introduction of multiple ADAS system, therefore the validation of automated and autonomous vehicles and their subsystems become necessary and urgent topic with a broad impact in current research.

ADAS system performance depends on different sensors like camera, RADAR, and LIDAR. So, validation of these sensors is important before their installation in vehicles and deployment to customer. Appropriate testing schemes, test design techniques, test creation methods, and test environment need to be used. It is very necessary to find blind spots, and limitation of this sensors for which proper corner test cases are needed to design. To test vision based ADAS system it is necessary to have virtual environment setup with good simulator capable of supporting different road scenarios, environment conditions, and traffic scenarios. Such vision based ADAS systems are needed to be tested in close loop, hardware-in-the-loop is one of the best closed loop testing methods and widely used by automotive industries. Deployment of HIL testing method helps to reduce errors, evaluate performance of system, and provide feasibility to add extreme test cases which are not possible to reproduce in real world. Also, helps tester to analysis and visualize system behavior virtually and interact with system.

HIL simulation enables systems to be evaluated efficiently while reducing testing time and costs and increasing quality of system. Taking advantage of the simulator and HIL testing method, we can generate many more test cases and test them beforehand in a simulation environment. HIL testing not only reduces time and cost for testing but also enhances the quality of the system.

This research will focus on finding different testing schemes and creating a test bench to evaluate the performance of forward-looking ADAS cameras, Mobileye 6. The test cases will be created by using the LGSVL simulator and RoboTestWebApp from MT-3136.

Key Words: Automated Driving, ADAS, Hardware-in-the-Loop, Testing and Validation, Test bench, Mobileye 6, LGSVL simulator, RoboTestWebApp

1 Introduction

Advanced driver assistance systems are intelligent systems located in the vehicle that supports the main driver in a variety of ways. These systems can be used to provide vital traffic information, road closures and blockages, congestion, suggested routes avoid congestion, etc. These systems can also be used to assess human driver fatigue and distraction to generate alerts or to rate driving performance and make suggestions about itself. These systems can take over humans by assessing any threat, performing simple tasks (like cruise control), or difficult manoeuvres (like overtaking and parking). The biggest advantage of using assistance systems is that they enable communication between different vehicles, vehicle infrastructure systems and traffic control centres. This enables the exchange of information for better vehicle visibility, tracking, planning, and decision-making.^[26]

ADAS functions can be divided into two groups: convenience functions and safety functions. The purpose of the convenience features is to warn the driver by activating a warning, such as a flashing light, a tone, a vibration or even a gentle suggestion of direction. The purpose of the safety functions is to act on the vehicle itself in cases where the driver does not react to a potentially dangerous situation. Actions include brake preload, seatbelt readiness, hood lift, automatic braking, avoidance steering, and so on. ADAS functions include Lane Departure Warning (LDW), Forward Collision Warning (FCW), Automatic High Beam assist (AHB), Traffic Sign Recognition (TSR), and Automatic Emergency Braking (AEB).^[27]

These ADAS functions are based on a front camera. Sometimes the information from the camera is supplemented by information from other sensors, such as Light Detection and Ranging (LIDAR) or Radio Detection and Ranging (RADAR). ADAS cameras are in the car on the windshield behind the center rear-view mirror. The field of view of the ADAS camera is in the wiper area to keep the glass in front of the camera as clean as possible. Sometimes RADAR detection, sight detection, and data fusion are combined into a single module.^[27]

Now we know in short what is ADAS system and how it works. But what makes it important to test this system. Below literature review gives different reasons why it is necessary to test the ADAS system before its deployment to customers.

1.1 Literature Review

1.1.1 Necessity to test the ADAS system

Autonomous automotive technology has developed very rapidly in recent years. Many automakers developing autonomous vehicles want to market them with SAE Level 4. The main goal in the development of autonomous cars is to reduce accidents caused by driver error.^[3] Automated driving will improve performance in most situations

compared to human drivers. However, it will not eliminate the risk of accidents or crashes.^[1]

The life-saving functions of autonomous vehicles are based on many sensors. One of the required sensors is the camera, which achieves its maximum functionality when connected to a process called computer vision. Computer vision is the ability of a computer to take a two-dimensional image or video and calculate relevant three-dimensional data from it. In automotive applications, computer vision-enabled cameras can generate information such as obstacle detection, object classification, and vehicle trajectory, all of which are critical to algorithms in automated vehicle control.^[5] So, this algorithm should be tested well beforehand.

1.1.1.1 Safety

According to the report on Road Safety, 1.2 million people die every year because of traffic accidents. Driving errors are the cause of 94% of accidents in the USA. Of these critical driver-related errors, 41% are recognition errors, 33% are decision errors, 11% are performance errors, and 7% are non-performance errors (sleep, etc.). Reducing the number of accidents is a major challenge and an ethical the mission for the automotive industry. This challenge can only be met with the development of Advanced Driver Assistance Systems and Autonomous Driving. These aspects must be considered when designing vehicle systems to improve safety. For this reason, many companies are investing in the development of ADAS and fully autonomous systems.

Safety is one of the major reasons for which the ADAS system has been developed. However, there are a lot of accidents caused by autonomous vehicles. Below are some accidents that were caused by autonomous vehicles.

1. Pony.ai loses driverless testing permit in California.

"On October 28, 2021, the Pony.ai Autonomous Vehicle ("Pony.ai AV") performed a left lane change maneuver in autonomous mode after turning right onto Fremont Blvd from Cushing Pkwy." During the lane change, the Pony.ai AV collided with a center divider on Fremont Blvd. and traffic signposted on the divider. The Pony.ai AV sustained moderate damage to the front end and undercarriage. Fortunately, no one was hurt, and no other vehicles were involved." ^[10]



Figure 1: Pony.ai Autonomous Vehicle Crash

2. Tesla Crash Involving Overturned Truck. “Steven Michael Hendrickson, 35, was a single father of two children. He adored his Tesla Model 3 and made at least two TikTok videos in which he praised FSD and how it made driving more enjoyable. On May 5, 2021, at 2:30 a.m., his Tesla collided with an overturned truck on California's State Highway 210. The crash injured the truck driver, as well as a bystander, and killed Hendrickson. The NHTSA decided to investigate the incident.” [11]



Figure 2: Tesla Crash Involving Overturned Truck

3. Tesla Crash with No Driver in Texas. “At 11 p.m. on April 16, William Varner and his best friend took a quick spin in a 2019 Tesla Model S. The car crashed into a tree and caught fire at 11:25 p.m. According to the police report, both men died, but neither was behind the wheel. This prompted the NHTSA to launch an investigation to determine what caused the crash – and whether Autopilot, FSD, or Smart Summon were involved.” [12]



Figure 3: Tesla Crash with No Driver

All the above incidents show how autonomous vehicles failed in different scenarios and human life was lost. Therefore, it become an urgent topic for validation of such a

system. Before these functions can be distributed to consumers, they must be validated. To validate this system, different test designs and testing methods need to be analysed and used.

1.1.1.2 Data collection for validation

ADAS system tests provide data for: validation, benchmarking, R&D information, and data for evaluations, making them valuable for this increasingly used technology. For example, testing during the R&D and validation phases can help reduce system transitions and even the number of formal qualification tests required. Benchmark tests can evaluate the performance of systems offered by many manufacturers to set performance requirements and goals. The test can be taken from basic (monitors speed, direction, location, and response) to intermediate (basic with additional audio/video recording) to advanced (adds capturing bus messages from the vehicle for a complete understanding of the vehicle range, vehicle behaviour, and the expected response).

This collected data from testing can be useful in a vision-based system for object classifications.

1.1.1.3 Immediate requirement of ADAS system in vehicles

To sustainably reduce the number of crashes and the number of road users involved in road accidents, the EU prescribe many other assistance systems in cars from 2022. Vehicle trailers as well as systems, components, and independent technical units for these vehicles about their general safety and the protection of Vehicle Occupants and Vulnerable Road Users” has been in force since the end of 2019 and applies to all homologated passenger cars in the EU from July 6, 2022. From July 7, 2024, all newly registered passenger cars must have these onboard systems in addition to the already prescribed assistants such as ABS, ESP, or tire pressure systems as standard.^[29]

Due to the short time to market and competition between OEMs, it becomes necessary to test ADAS systems in a virtual environment as it reduces time to test and cost. Also, to have such an ADAS system installed in vehicles, it must be reliable. For which it is necessary to test the ADAS components.

1.1.1.4 Blind spots

As per the report in auto motor and sports the ADAS assistance system especially Automatic Emergency Braking weakens in the dark as AEB relies on the information from the camera sensor, RADAR, or other sensors. When the sensor detects the object and the potential for collision with it, they activate AEB. The systems are less effective in the dark and have a strong dependence on headlights. Eight compact SUVs were tested which showed clear dependency on headlights. Better the light, the more efficient the AEB system works.^[30] When the object is out of visibility area camera cannot detect the object.

Camera-based ADAS systems have a certain blind spot in which it fails to detect objects. Such blind spots can be identified while testing by creating corner test cases and can be used by a blind-spot warning system to notify the driver during the actual drive.

1.1.2 How to test the ADAS system

Testing of ADAS system improves the performance of ADAS system and makes it more reliable. Data on system performance can be captured and blind spots can be identified prior before the deployment of the system. But the question is how to test this system and where to.

The detailed information on different test design techniques (testing schemes) is discussed in chapter 2. While how test cases can be created and where to test these test cases are discussed in detailed in chapters 3 and 4 respectively.

1.2 Significance

As mentioned, ADAS is already widespread and autonomous vehicles are expected to increase in popularity dramatically in the coming years. However, for these functions to be implemented, they must also be validated.

Different test design techniques and testing methods can be used to validate such a system. This thesis focuses on the study of different test designs and testing methods available for the validation and implementing appropriate test design techniques to evaluate the commercially available ADAS product Mobileye 6. The current mainstream safety assisted driving systems include a Lane Departure Warning system, and a Forward Collision Warning system.

1.3 Scope of the Thesis

Research into different testing schemes. How the performance of commercially available ADAS products like Mobileye 6 can be evaluated. To understand and evaluate the capabilities of the already existing RoboTestWebapp with the available AV-Stacks and simulators in combination with the CAN setup. Research on how threats to validity can be mitigated for the testing scheme. Creating a testing scheme and identifying blind spots for the Mobileye ADAS components based on the literature research and the capabilities of the RoboTestWebapp with the aim of holistic capability assessment and comparison of the ADAS Systems. Evaluating the test results with a focus on identified corner cases.

The introduction to the ADAS system and why it is necessary to test the ADAS system is given in chapter 1 with the scope of the thesis. The rest of the report is structured as follows. The test design technique (testing schemes) and test case creations are discussed in Chapters 2 and 3. The test environment and its setup are described in Chapter 4. Followed by an evaluation and results in Chapter 5. And conclusion and future work are described in Chapter 6.

2 Test Design Techniques

Most crashes in autonomous vehicles take place due to failure or limitation of sensors like LIDAR, RADAR, and Camera. So, it becomes necessary to test this ADAS component. To validate the ADAS system question is how the validation can be done.

In the automotive field, it is proposed for Automated Driving Systems that ADSs must be driven over eleven billion miles to prove that ADSs are safer than human drivers. Furthermore, even after eleven billion miles, such testing will “ensure, but not always guarantee, safety,” suggesting that vehicle-level testing or real-world testing before the start of production (SOP) would not be sufficient to determine the safety. While average customers are willing to pay up to \$3,500 to partially automate their car and \$4,900 for full automation, giving them truthful information about the system's limitations and capabilities is imperative to ensure safe and correct use and confidence to ensure the system.^[15]

One of the reasons for proposing the 11-billion-mile requirement for testing ADS safety is that it would potentially uncover all sorts of “black swans” (also known as “unknown unknowns”) and known unknown scenarios or unexpected potential accidents. Identifying such scenarios remains a problem for test engineers and risk analysts dealing with complex systems. Also, the lack of a standard set of validation metrics. Black swan scenarios, by their definition, may or may not be detected even at eleven billion miles. This has also led to growing concern from regulators about errors that have low probability and high impact. Therefore, rather than focusing on the number of miles, it is critical to focus the investigation on identifying “black swan” and “known unknown” scenarios.^[15]

While requirements-based testing efficiently captures “known knowns”, failure to ensure its integrity leads to “unknown knowns”, “known unknowns”, and “black swan” scenarios. The last three together could potentially be combined into “interesting” scenarios and defining such scenarios should be the goal of the testers.^[15]

The aim of testing should focus on coverage of “known known” scenarios. One of the difficulties with “black swans” is their unpredictability, when will they appear and their ad-hoc nature. Therefore, to ensure that the systems have a safe and robust functionality, it is important to be able to define test scenarios which are trigger real-world scenarios, represent user inputs, and define and identify “all” operating conditions.^[15]

Different test design techniques can be used to create test cases and validate the ADAS system.

2.1 Systems Theoretical Analysis of Processes

Systems Theoretical Analysis of Processes or STPA is a method for identifying hazardous events based on the accident causality model STAMP (Systems Theoretic Accident Model and Processes), which in turn is based on systems theory and control theory. It aims to analyze security in a socio-technical system with different interacting elements. Building on a systems-based approach, STPA identifies a broader spectrum of hazards that can arise for a variety of reasons, including component failures, component interactions, human error, human-automation interaction, software issues and requirements, incorrect, and even socio-technical and organizational factors. One of the main advantages of STPA is that it allows the person performing the analysis to identify the causal factors of the hazards and the corresponding requirements that, if implemented, would prevent the hazard from occurring. Therefore, it supports the identification of preventive actions for a hazard, not just its subsequent mitigation.^[15]

Other methods like FMEA, and FTA focuses on root cause analysis of failure, while STPA focuses on failure as well as non-failure. The quality of STPA results depends on the quality of data used for analysis.

STPA is a four-step process. The first is to define the purpose of the analysis. Includes 1) what type of failure the analysis is trying to prevent; 2) will STPA only apply to traditional security objectives such as preventing the loss of human life, or will it apply more generally to security, privacy, performance, and other system properties? The second step is to create a model of the system, called the control structure, which captures the functional relationships and interactions by modelling the system as a set of feedback control loops. The third step is to analyse the control measures in the control structure to examine how they could lead to the failures defined in the first step and to create functional requirements and constraints for the system. The fourth step identifies failure scenarios to explain why unsafe control and other unsafe behaviour can occur in the system.^[15]

2.2 Failure Mode and Effect Analysis

Another method like Failure Mode and Effect Analysis (FMEA) focuses on finding the root cause of the failure.

FMEA is a structured process for evaluating, quantifying, and reducing the risks associated with various elements of a design, i.e., It is a risk assessment process. The process assesses which characteristics or failure modes could impact customer-perceived product quality (regardless of where they are in the supply chain) and how to mitigate those risks. The important decisions are which design features are significant and what measures have been taken (or can be taken) to minimize the risks. The FMEA approach documents these decisions.^[16]

FMEA is a bottom-up analysis method, used to identify failure modes with root causes of all parts of the system to find negative effects. The analysis starts with the lowest

level components and continues up to the failure effect of the entire system. A lower-level failure effect becomes the next higher level component failure mode. The FMEA also measures severity, occurrence, and probability of detection, which are used to calculate risk priority numbers for the identified failure modes. The main objective of the FMEA is to identify potential problems in the initial design process of a system or product that may affect its safety and performance and to initiate countermeasures to mitigate or minimize the impact of the identified potential problems (failure types). In addition, the FMEA can complement the FTA and identify many other failure types and causes. Failure Mode and Effect Criticality Analysis (FMECA) is an extension of FMEA that ranks the identified failure types based on their severity, which is used to prioritize countermeasures.^[19]

Failure Mode and Effect Analysis is a quality engineering technique well used in the automotive industry and is suggested by ISO 26262 for various analyses.^[20]

There are different types of FMEA, Systems FMEA defines the failure mode as a problem, what is the effect of the problem and the direct and indirect cause of the problem. Design FMEA defines failure mode as the cause of the problem from SFMEA as the result of the problem occurring with detailed specification and root cause for failure. Process FMEA defines the cause of the problem from DFMEA as the result of a problem occurring and specific root causes for process failure modes.^[21]

2.3 Fault Tree Analysis

A system is a collection of components in a defined architecture with the sole purpose of fulfilling the function of that system. The probability of a functional failure of this function is determined by the integrity of the individual components as well as by the logic of the system architecture. The more complex the system, the greater the need for a thorough analysis technique to identify all combinations of errors that could result in loss of system integrity. Fault tree analysis (FTA) is one such technique. A fault tree¹ shows graphically, using a special notation, the logical relationship between a specific system fault and all its contributing causes.^[17]

FTA is a schematic analysis technique used for reliability, maintainability, and safety analysis. It is a top-down analysis, going through successively more detailed (i.e., lower) levels of design until the probability of the main event occurring in the context of its environment and operation can be predicted.^[17]

Any sufficiently complex system can fail due to the failure of one or more subsystems or components. The goal of FTA is to use deductive logic to understand all the causes of a failure in a system complex to the point where the probability of failure can be reduced through improved system design.^[17]

An FTA is performed to fulfil the understanding of the system characteristics by the schematic representation of the system architecture.^[17]

An FTA can be performed for both positive and negative events: The segments of the

logic tree that lead to a negative event, such as an accident, define all the things that could go wrong to cause the negative event. Negative event logic tree segments use more OR gates than AND gates, barring redundant safeguards. The segment of the logic tree that leads to a positive event defines all things that must work together for the machine to function or accomplish a successful mission. Positive event logic trees use more AND gates than OR gates, except for redundancy. Maintenance troubleshooting trees are a good example of positive event logic trees. Reversing the output of a positive event makes it a negative event.^[17]

2.4 Real Incidents

Analyzing real case incidents that are caused by autonomous vehicles replicating the same into a virtual test environment to evaluate the ADAS system. Real incidents can be analyzed for creating corner test cases.

Example tesla's white truck crash, where the tesla car goes under the white truck. In which tesla car's forward-looking system was not able to detect a white truck on a bright sunny day resulting in the crash. The occurrence of such a scenario is low but has high severity in terms of human life. The same scenario can be replicated in a simulation environment as a corner test case for the evaluation of the vision based system.

2.5 Responsibility-Sensitive Safety Model

The RSS model formalizes and contextualizes human judgment about all driving situations and dilemmas. AV can be programmed to meet these safety definitions, allowing AV to share the road with human drivers for the next couple of decades. This includes terms such as safe distance and safe spaces when merging and turning, right of way, how to define safe driving with limited detection and visibility.^[13]

This formalization is then translated into a set of rules in four main areas. First, RSS defines safety distances for all driving scenarios, from one-way traffic to intersections and scenarios with multiple geometries. Second, RSS accurately defines what is considered a dangerous situation as a derivation of all the semantics and rules defined in the 'human judgment' formalization. The third key definition in the model is the appropriate response that must be taken to avoid a dangerous situation. By formalizing these first three domains, RSS has created a set of parameters that are missing in the current development of safety systems for autonomous vehicles. The lack of these parameters is an issue that needs to be addressed as part of efforts to make AVs safer and more "human".^[13]

2.5.1 Implementing RSS

2.5.1.1 One-Way-Traffic Scenarios

Before going deep into scenarios what is safety distance as per RSS. Safe longitudinal distance in one way traffic is the longitudinal distance between a blue car and a red

car is considered to be safe even if the blue car abruptly applies full brake but still red car avoids the collision. Distance is considered to be safe if the accident is not possible, even if a blue car applies full brake ($a_{\max, \text{brake}}$) and during its response time (p) the red car accelerates at maximum acceleration ($a_{\max, \text{accel}}$) and then immediately brakes by at least the minimum reasonable braking force ($a_{\min, \text{brake}}$) that is how human driver reacts in such situations.^[13]

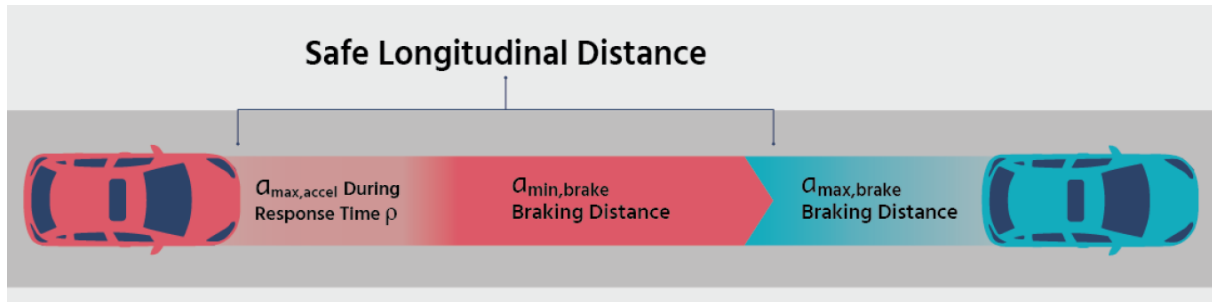


Figure 4: Safe Longitudinal Distance

When both the cars are not at a safe distance is called a dangerous situation. In case the lateral distance is not safe both cars should brake by at least ($a_{\min, \text{brake}}^{\text{lat}}$) until reaching a safe lateral distance.^[13]

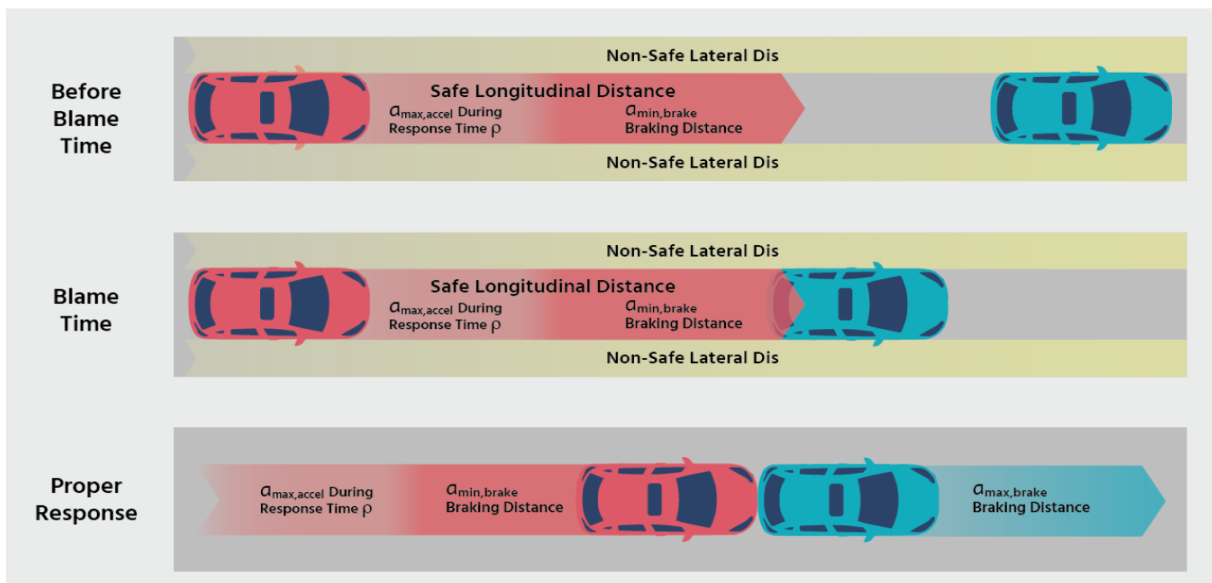


Figure 5: Response for Red Car when following Blue Car

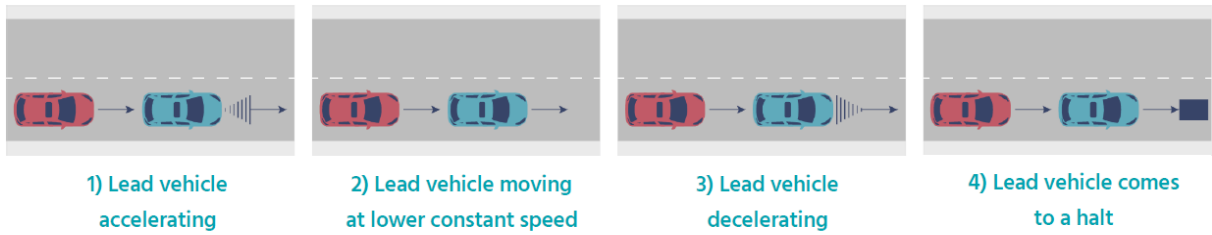


Figure 6: One Way Traffic Scenarios

The above scenarios are quite like each other when examining them through the RSS model and therefore, shall consider them as one.^[13]

In the scenarios above, dangerous longitudinal situations started when the safety distance was exceeded for the first time. The appropriate response to this set of scenarios is that the leading car is free to apply full brake power, while at the same time the rear car must use the designated RSS braking pattern³. Assuming an accident occurs because of any of these scenarios, the rear car is responsible for not having met the correct response to this situation.^[13]

2.5.1.2 Cut-In and Drifting Scenarios

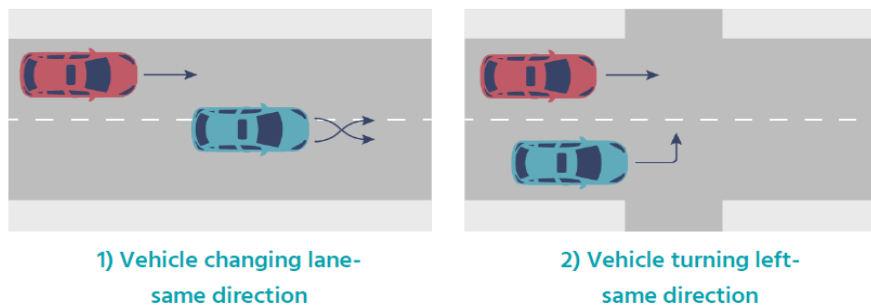


Figure 7: Cut-In and Drifting Scenarios

In scenarios 1 and 2, when a vehicle is changing lanes (i.e., cutting-in), we need to examine whether it was a safe cut-in or not.^[13] Assuming a non-safe cut-in, the dangerous situation is initiated when the red car violated both the longitudinal and lateral safe distance of the blue car. Therefore, it is to blame in case of a collision.^[13]

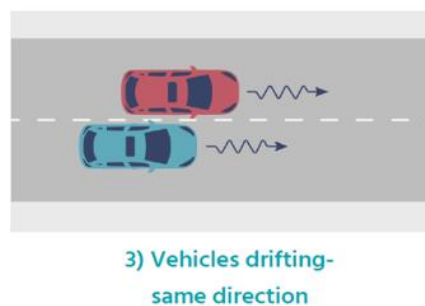


Figure 8: Cut-In and Drifting Scenario both Vehicles drifting same direction

In above scenario, regarding a car drifting from its lane, the longitudinal distance was non-safe, to start with. The case isn't considered dangerous, however, until one amongst the cars violates the safe lateral distance. Assuming that before the blame time there was a secure lateral distance, then the right response would be to brake laterally until reaching zero lateral velocity. The car which failed to manoeuvre laterally is already at zero lateral speed, hence only the opposite car should brake. If that car doesn't brake laterally then it didn't suit the right response and thus is responsible.

The above analysis illustrated how the RSS model covers two multi-agent scenarios presented within the NHTSA research (excluding only vehicle failure events, crashes with stationary objects etc.).^[13]

Different test design techniques were studied in this thesis, the RSS model is used among other design techniques because the RSS model focuses on how accidents can be caused in different driving scenarios if a safe distance is cut. This is important for the evaluation of the vision based ADAS system. The scope of this thesis is to test a vision-based system i.e., the ADAS forward-looking camera Mobileye 6.

MiniEye was not tested in this thesis due to its certain limitations and it was tested in MT-3166. MiniEye has failed in certain corner test scenarios and was not able to give a forward collision warning which is the result of MT-3166. So, this thesis focuses on testing Mobileye 6.

RSS model defined multiple scenarios that can lead to a crash. Using this scenario along with different test conditions the blind spots can be identified for testing of Mobileye 6. In this thesis, the RSS model scenario is used as a base for test case creation. While other techniques describe hazard events and failure causes which can be used to evaluate other ADAS components.

3 Test Case Creation

This chapter describes in detail what parameters need to consider and how to create test cases.

According to the ISO26262 standard, the risk associated with each specific scenario is determined by its severity, likelihood of occurrence, and controllability. Controllability will not be an issue because autonomous vehicles are entirely made up of self-learning algorithms. As a result, the risk associated with each scenario is calculated using two key factors: the severity associated with the selected concrete scenario and the frequency/probability of the selected concrete scenario occurring.^[6]

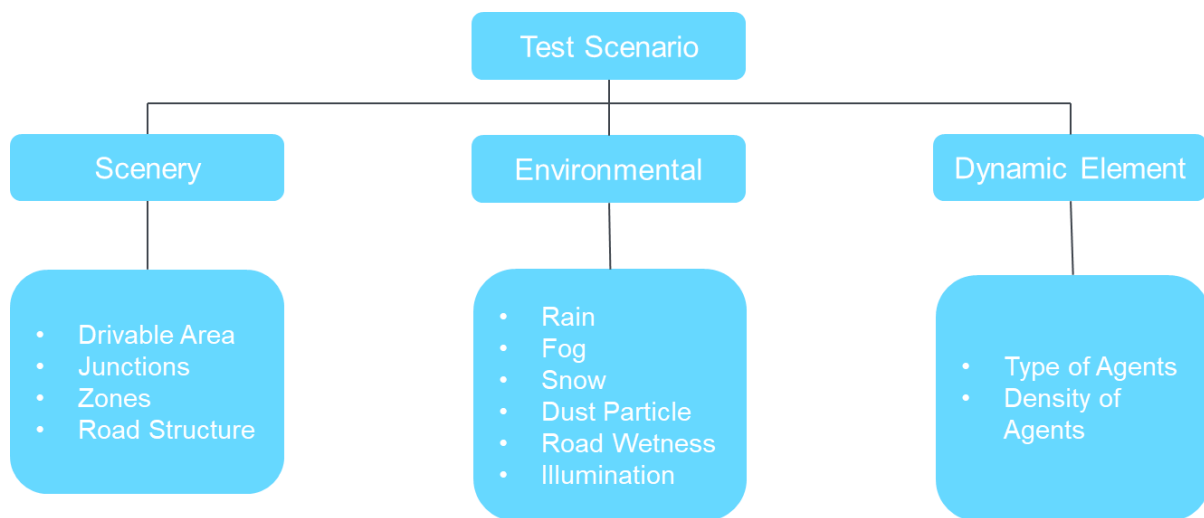


Figure 9: Test Scenario Attributes

For creating test scenarios different test attributes need to be considered. The scenery consists of road objects like road layout, road geometry etc. The environment includes attributes like rain, fog, snow etc. All moving objects like pedestrians, and NPC agents are dynamic elements. Once the parameters are decided, how to use this parameter along with the testing scheme RSS traffic scenario. Different methods can be used to create test cases. Below is one of the methods that are used for creating test cases in this thesis.

The Boundary Value Analysis test case design technique is used in this thesis.

Boundary value analysis is based on testing at the boundary values. It includes maximum, minimum, and typical values of parameters. For the creation of test cases in RoboTestWebApp, a one-way traffic scenario from the RSS model was used. Different boundary and typical values like 0%, 50%, and 100% for environmental parameters such as rain, fog, and road wetness were considered for test case creation.

Also, the different day time interval is considered. Table 3 shows how test cases were designed for the evaluation of Mobileye 6.

Once the testing scheme and test case creation method are fixed next step is to create test cases. For test case creation already existing product of MT-3136 in the IAS lab i.e., RoboTestWebApp is used.

3.1 RoboTestWebApp

The tool RoboTestWebApp is developed in MT-3136 using the Django web-based application framework. The tool is interfaced with the LGSVL driving simulator and the apollo AV stack. All the possible test dimensions of the driving simulator are analyzed, and a general test case model is developed along with a result model to store the meaningful data from the test execution.^[6]

Below is the use case diagram for RoboTestWebApp.

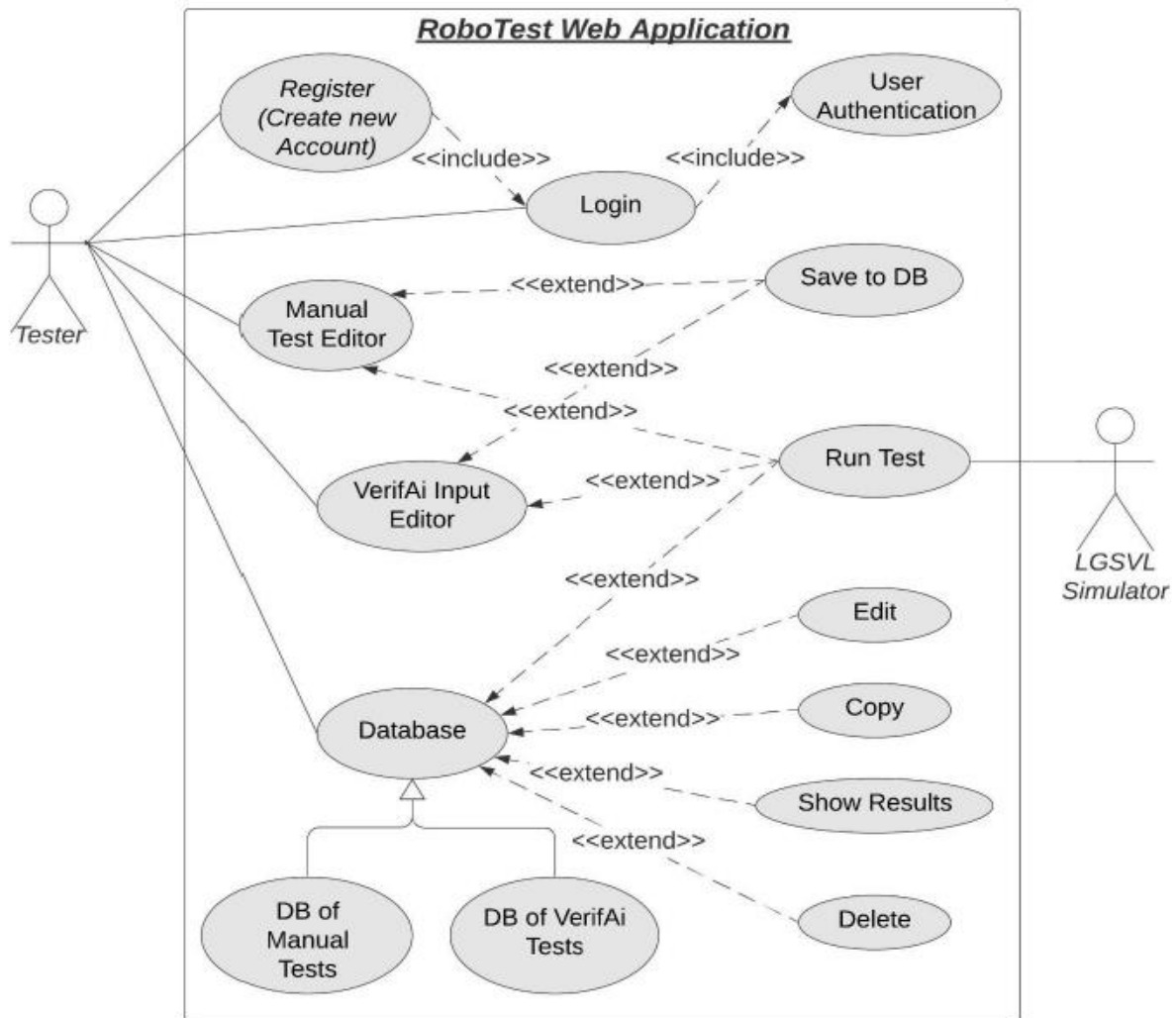


Figure 10: RoboTestWebApp Use-Case Diagram^[4]

3.1.1 Description of Use Cases

In this thesis, a manual test case editor is used. Below sequence diagrams show how to register with RoboTestWebApp and how test cases can be created.

3.1.1.1 “Register”

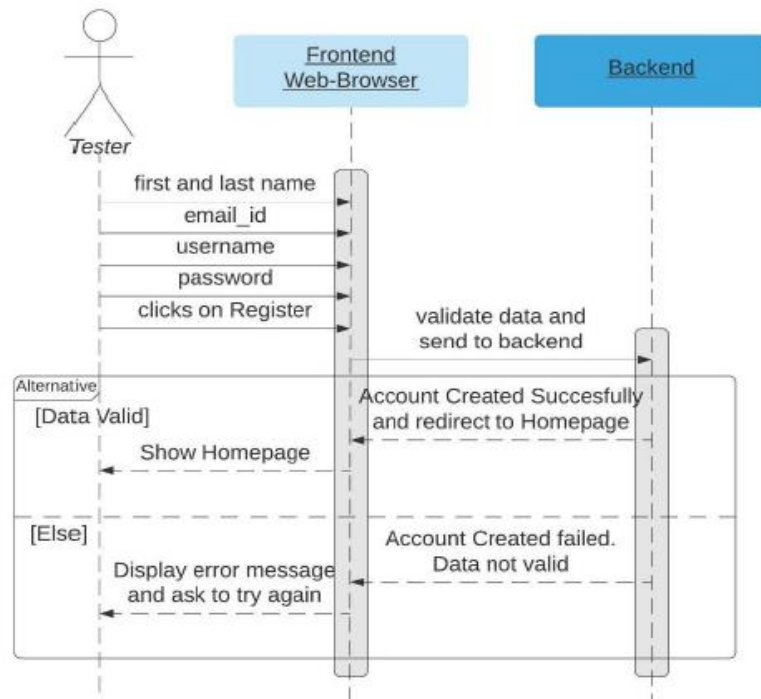


Figure 11: Sequence Diagram for “Register”^[4]

3.1.1.2 “Login”

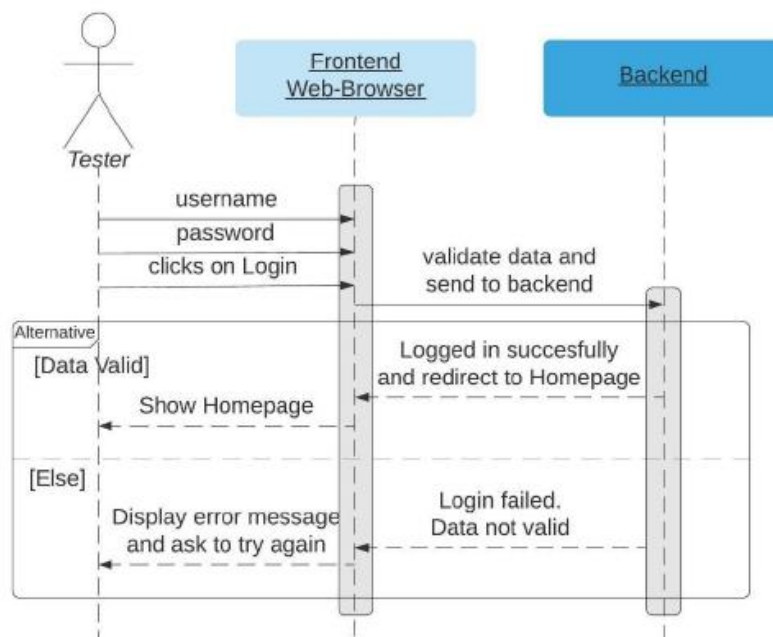


Figure 12: Sequence Diagram “Login”^[4]

3.1.1.3 “Manual Test Editor”

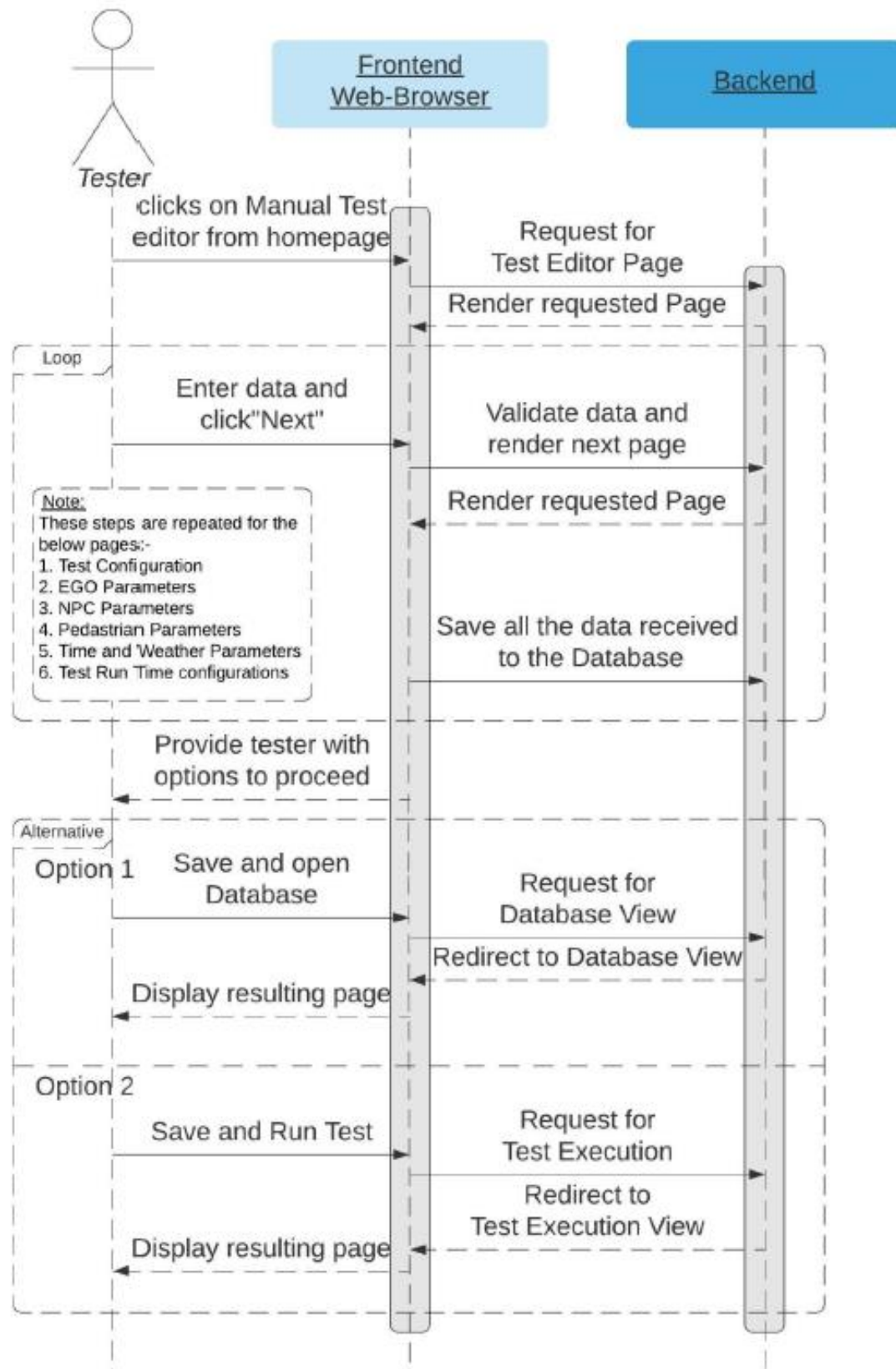


Figure 13: Sequence Diagram “Manual Test Editor”^[4]

3.1.1.4 “Run Test”

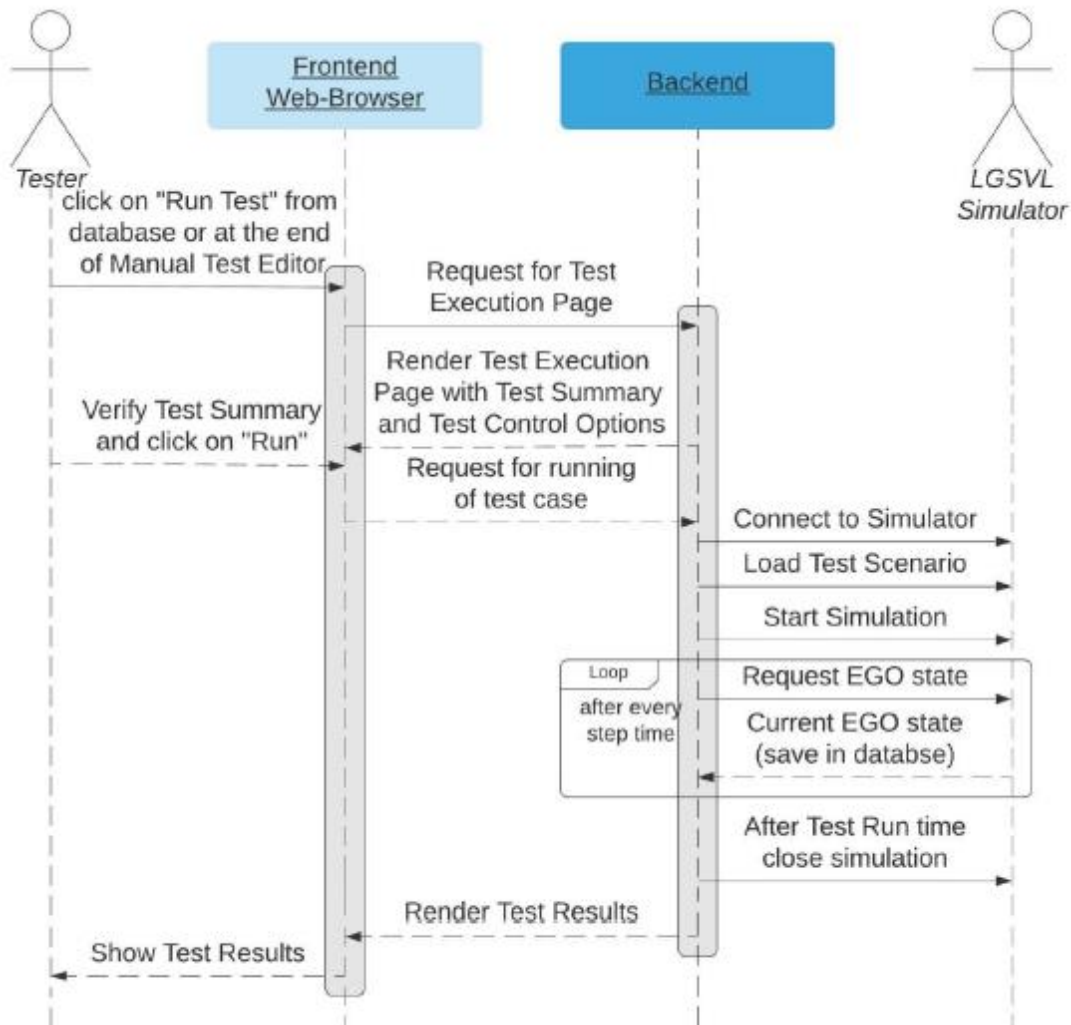


Figure 14: Sequence Diagram “Run Test”^[4]

Following the above steps test cases can be created using RoboTestWebApp. The user interface of RoboTestWebApp is shown below.

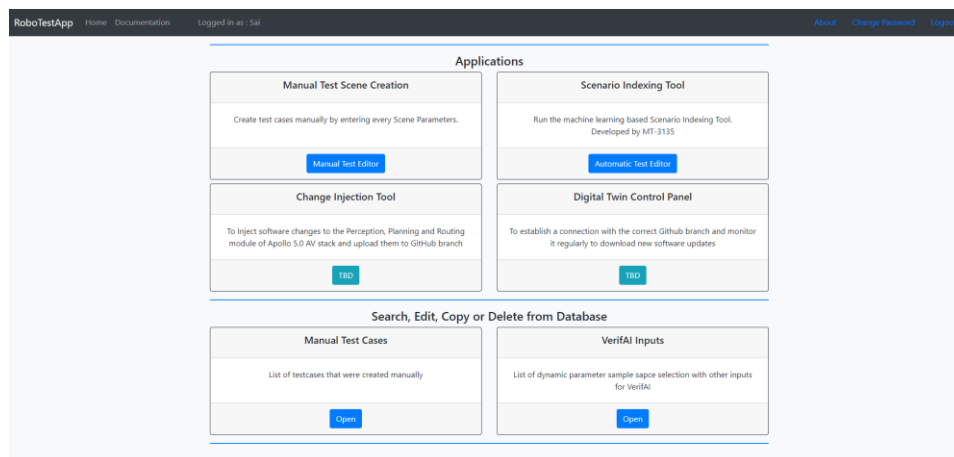


Figure 15: RoboTestWebApp User Interface

The above sequence diagram shows how test cases are created using RoboTestWebApp.

Certain limitations were observed while creating test scenarios using RoboTestWebApp. Firstly, creating multiple geometric scenarios using RoboTestWebApp was not feasible and secondly, the speed of the Ego vehicle was controlled manually, RoboTestWebApp does not work well with Apollo 5.0 in order to control Ego vehicle speed automatically. RoboTestWebApp should be improved in future for the above limitations.

Another limitation of RoboTestWebApp observed was the storage of test cases and user login. If the user is registered and test cases are created on one PC which is on a different network and the user needs to access that user account and those test cases on some other system which is on some different network. The account and test cases cannot be accessed on that system. This can be further improved where users can use the same user account on multiple systems and can create test cases accordingly without the need of creating test cases on the same system.

4 Test Environment

Once the testing scheme is finalized next step is where to perform testing. This chapter describes different test environments. To validate the ADAS system different validation methods are used. Like virtual environment simulation, and x-in-the-loop approaches. Testing the ADAS system prior reduces the risk and cost.

4.1 Validation Methods

4.1.1 Virtual Environment Simulation

Creating a virtual environment for ADAS system testing by implementing a complete scenario using the software. This includes driver, sensors, and test scenarios. Virtual environment testing is safe as compared with real drive testing. It helps to test different scenarios by reducing costs and efforts.

The virtual environment helps to prototype and develop new system features. Helps researchers create more reliable ADAS systems and integrate different advanced driver assistance systems to develop a better system.^[18]

4.1.2 X-in-the-loop Simulation

X-in-the-loop combines both real-world and simulated elements for ADAS system testing. Below are some x-in-the-loop approaches for testing autonomous vehicles.

4.1.2.1 Software-in-the-loop (SIL)

Software-in-the-loop (SIL) is a method to test ADAS software components. It links algorithms that correspond to certain vehicle hardware to the simulation. SIL provides feasibility to the developer to check code performance in a simulated environment without actual hardware parts.^[18] SIL testing is done in the early phase of the software development process. It is one of the important testing methods in the automotive industry as OEMs nowadays build software-defined vehicles that enable features and functions mostly through software.^[22]

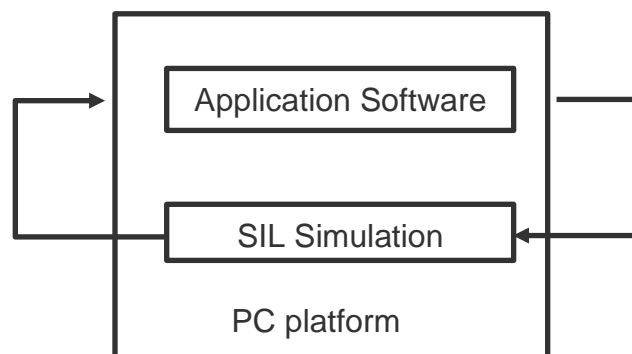


Figure 16: SIL testing

4.1.2.2 Driver-in-the-loop (DIL)

In Driver-in-the-loop (DIL) real person drives a simulated vehicle that has controls similar to the real vehicle and operates in a simulated environment.^[18]

4.1.2.3 Hardware-in-the-loop (HIL)

Hardware-in-the-loop (HIL) is one of the popular methods for ADAS system testing. It uses real-time simulation for checking vehicle hardware. In HIL real hardware is combined with simulated components. HIL includes the electrical emulation of sensors and actuators. These electric components act as interfaces between simulation and system under test. HIL testing is done in a simulated environment and thus can run critical test cases that can cause damage or harm to the system or people in the real world.

Some benefits of HIL testing are:

1) Increase Safety: Tests can be carried out without harming people or equipment. HIL simulation allows running test cases that can damage the vehicle in real testing. This also can lead to an increase in the cost of testing.^[31]

2) Enhance Quality: If HIL simulation is integrated into a model-based design process, it can already be used in the initial design phase. The HIL simulator grows with the actual plant design and can be used by control engineers to continuously test their control systems. These tests will uncover problems and errors that would otherwise have been discovered in the final stages of a design in which the plant and control system are integrated. Numerous research projects have shown that the early detection of problems and errors and the appropriate action leads to a significant increase in the quality of machines and systems.^[31]

3) Save Time: The cost of errors increases with the time it takes to discover them. Start-up errors are notorious for the time delays they can cause. In most companies, machines and controls are developed in parallel, which means that errors in the control can only be found during commissioning. With the HIL simulation, errors can be found and corrected quickly. Therefore, HIL simulation is an effective method to reduce commissioning time.^[31]

4) Save Money: Test in a real environment can damage the vehicle and can lead to an increase in the cost of testing. And it is not cheap to develop a prototype for testing. Also, it will increase the time for testing so more working hours.

5) Human Factor: Most testing methods do not consider human interaction. However, HIL simulation can be used to test human interaction with the system. In this way, control engineers can test whether their system can be operated easily and conveniently. If the HIL simulation is expanded to include screens, these can be used for operator training. These HIL simulators are called training simulators. It also allows the test engineers to switch weather conditions and other variables. And the system does not break down in case of human error while testing.

The typical process of hardware-in-the-loop testing is shown below. Traffic simulation is created using the simulator for example in this thesis LGSVL simulator is used. Environment conditions such as fog, rain and road wetness can be adjusted. And the performance of the system under test can be evaluated.

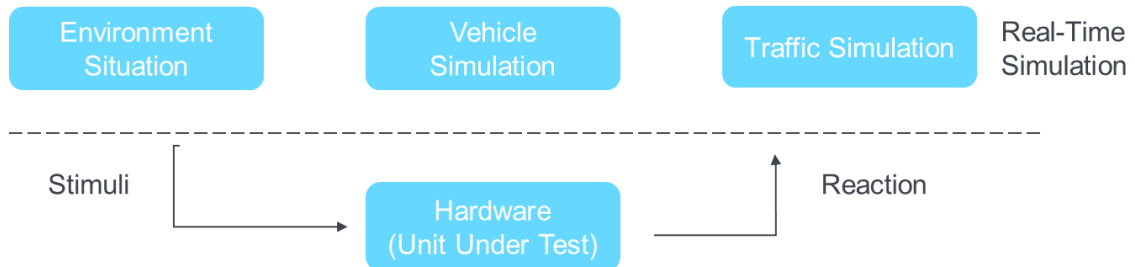


Figure 17: Hardware-in-the-loop Process

4.2 Hardware-in-the-loop Environment Setup

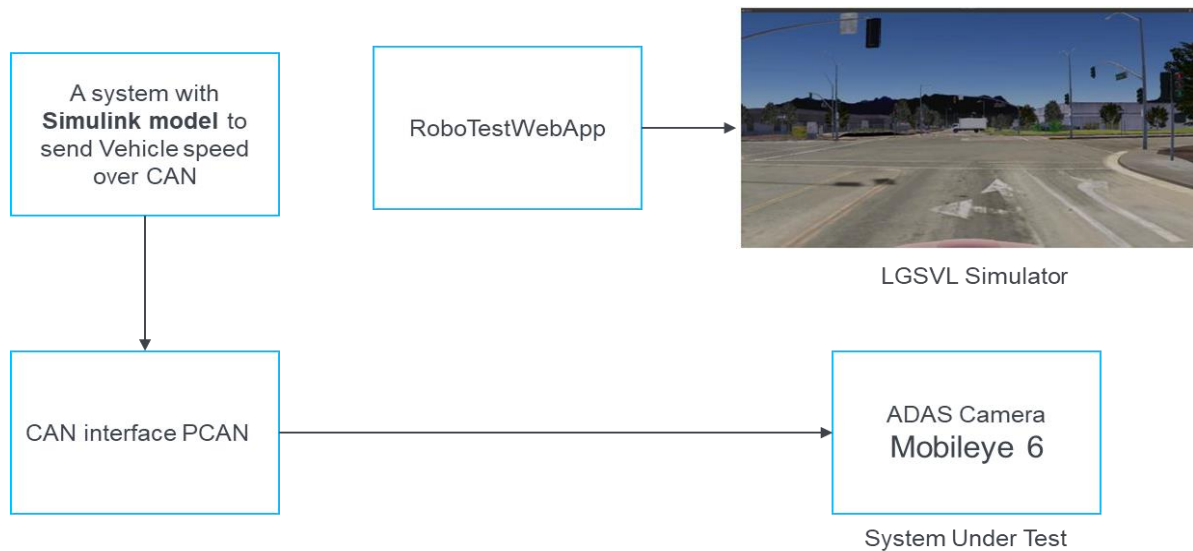


Figure 18: HIL Setup

The system under test is Mobileye 6 camera, which is validated using the HIL validation method.

4.2.1 Camera Setup

The complete installation procedure for camera setup is given in the appendix.

4.2.2 CAN Communication Setup

Once the camera is set up the next step is to set up a CAN communication link with Mobileye 6, to send a vehicle speed signal. This is done using the MATLAB Simulink model and PCAN as CAN interface.

4.2.2.1 Simulink Model

To send CAN signals to Mobileye 6 in a simulation environment MATLAB® Simulink® model is used.

Vehicle Network Toolbox™ offers MATLAB® functions and Simulink® blocks to send and receive CAN, CAN FD, J1939, and XCP messages. The toolbox helps to identify specific signals. It uses a standard database file for the transmission and reception of CAN signals.

Toolbox simplifies communication with in-vehicle networks and provides feasibility to monitor, filter, and analyze live CAN bus data or log and record messages for later analysis and replay. Message traffic can be simulated on a virtual CAN bus or connected to a live network or ECU. The Vehicle Network Toolbox supports CAN interface devices from Vector, Kvaser, PEAK-System, and National Instruments®.

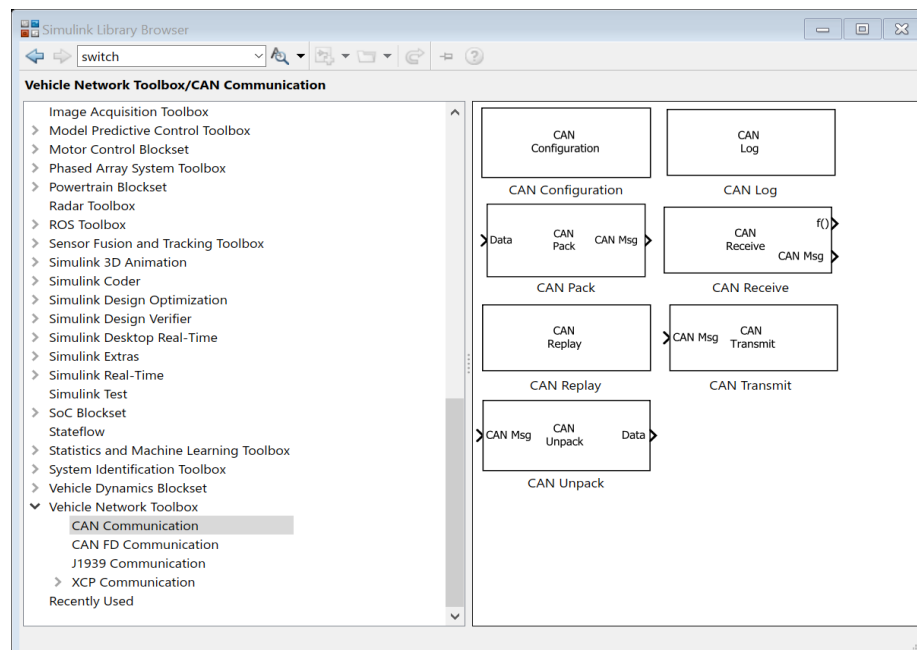


Figure 19: Vehicle Network Toolbox

A diagram of the model used to send messages to the CAN bus and Mobileye 6 can be seen below. Messages were sent with the CAN Pack and CAN Transmit blocks by referencing the .dbc file, received from an open-source project from Comma.ai on Github (GM Global Powertrain dbc, 2019). Those CAN message were uploaded to the PCAN from Simulink. In Simulink, the only vehicle information that must be sent to the CAN bus and Mobileye 6 is vehicle speed.

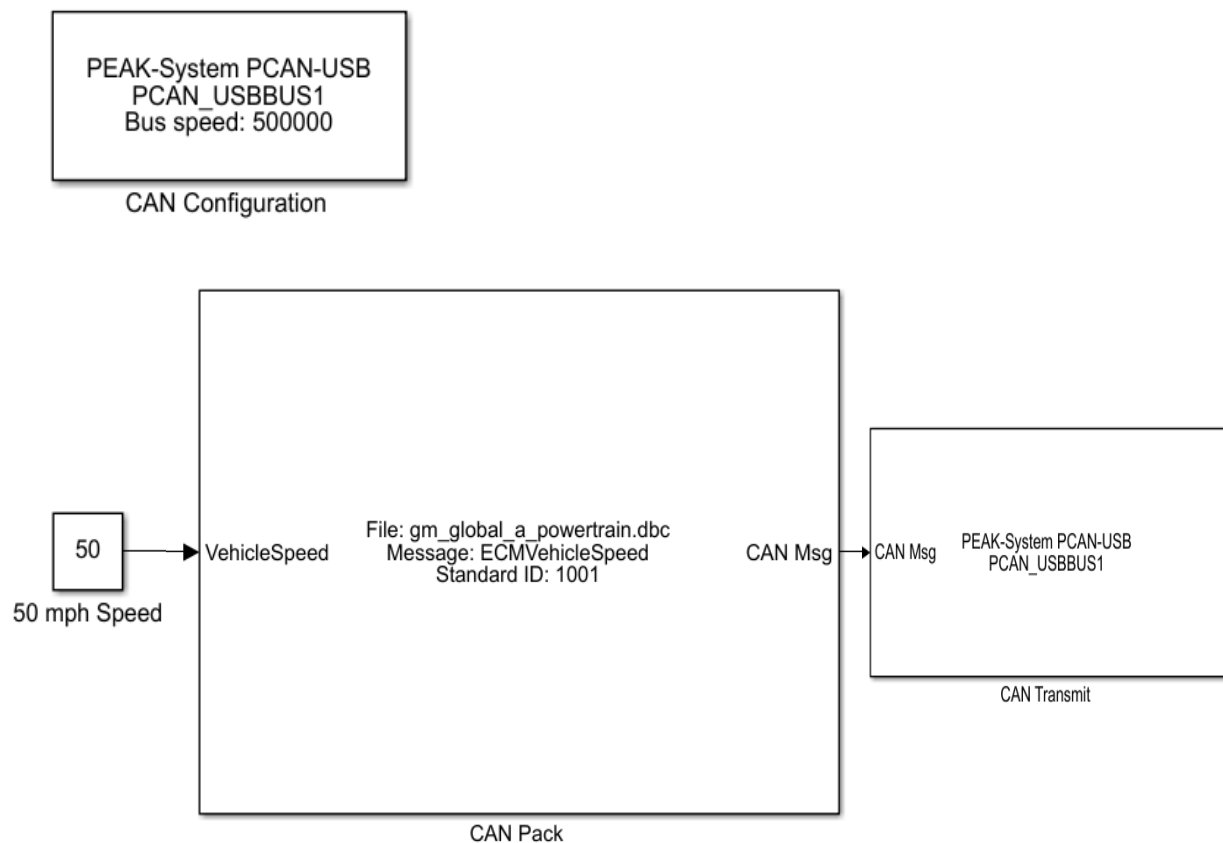


Figure 20: Simulink Model to Send CAN signal

4.2.2.2 PCAN Connection

As Mobileye is connected with vehicle CAN bus so to do the same, we need some CAN hardware that can communicate CAN signals to Mobileye 6.

So, in this thesis PCAN is used as CAN hardware. PCAN is quite easy to use and is supported by the Vehicle Network Toolbox from Simulink.



Figure 21: PCAN

4.2.2.3 DB9 Connector

PCAN DB9 male connector is connected with Mobileye 6 CAN high and low wire using DB9 female connector having 120-ohm impedance resistance. And another USB end with the system having MATLAB which is used for sending CAN signals through the Simulink model.

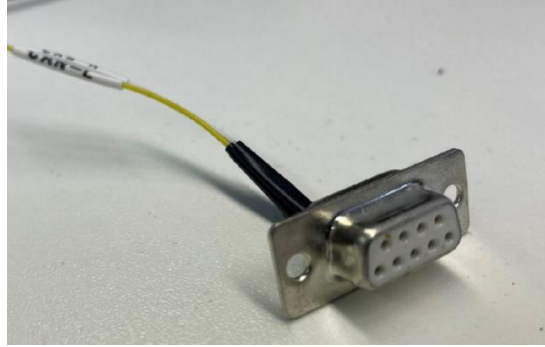


Figure 22: DB9 Connector

4.3 Simulator Setup

A virtual environment simulator needs to be implemented to reproduce the camera's view of a windshield-mounted camera for the camera-in-the-loop testing bench.

LGSVL is one of the open-source simulators developed by LG Electronics America R&D center providing a wide range of features. It is mainly used for testing AV functionalities. It can be used with AV stacks like Apollo and Autoware with the support of communication bridges like Robot Operating System (ROS, ROS2) and CyberRT.^[7] It provides a 3D view simulation. Different scenarios for testing can be created using the LGSVL simulator, like weather conditions (fog, rain), and time of day. Different maps can be created as per need.

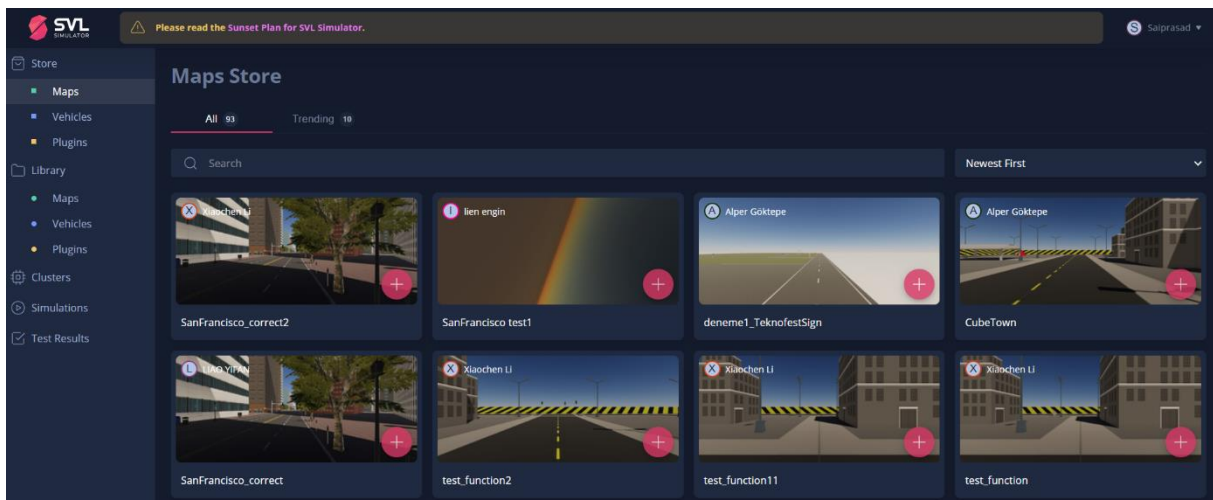


Figure 23: LGSVL Simulator

It provides web user interface options like Store, Library, Clusters, Simulations, Test Results and simulation user interface options like Simulator main menu, Simulation menu, Sensor visualizers, Bridge connection UI, Configuration file and command line parameters, Keyboard shortcuts.^[7]

Test cases can be created using Python API, Visual Scenario Editor or Random Traffic generator.



Figure 24: LGSVL Simulator Environment Control Panel

The detailed setup process for the LGSVL simulator is given in the appendix.

Here LGSVL simulator is used along with RoboTestWebApp for creating test scenarios. AV stack Apollo 5.0 is used to automatically drive ego vehicles. In the case of manual drive, keyboard controls are used.

4.4 Display

A Digital Display with a very high resolution of 3180 x 2160 is used to display the simulation. The resolution of display should be high enough to display simulation. As poor resolution is threat to validity. With poor resolution Mobileye 6 will not be able to detect objects or NPC agents properly which will result in poor evaluation (testing).



Figure 25: Display Unit

4.5 DC Power Supply

DC power supply is used to provide a 12v DC supply to Mobileye 6.



Figure 26:DC Power Supply

The final HIL environment setup created in this thesis is shown below figure.

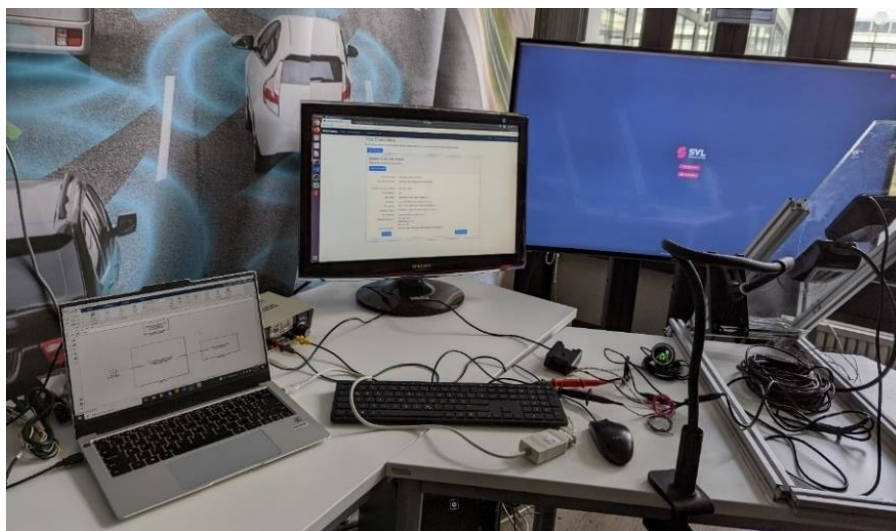


Figure 27: HIL Environment Setup

5 Evaluation

Once the test environment is setup, testing schemes, and test case creation techniques have been finalized next step is the evaluation of Mobileye 6.

Results are obtained on EyeWatch and CANSee. On CANSee ID 0x700 is used to monitor the response from Mobileye 6. Before we go into evaluation the threats to validity must be checked and overcome. What exactly are threats to validity and how to overcome them are explained in detail below.

5.1 Threats to Validity

Threats to validity define cause-and-effect relationships established in studies that cannot be explained by other factors. That means some outside factors can cause the result to be changed or not as expected.

5.1.1 Types of Threats to Validity:

1) Threats to Internal Validity: defines the cause-and-effect relationship that cannot be changed by any other factor.

Threats to internal validity are:

History	External or unanticipated events occur between the administration of evaluation surveys. ^[23]
Maturation	Aging or development of participants occurs. ^[23]
Instrumentation	Result change between pre and post test. ^[23]
Testing	The pre-test influences the outcome of the post-test. ^[25]

Table 1: Threats to Internal Validity

How to overcome these threats:

Having a large sample size can counter testing, and as result would be more sensitive. Random assignments of participants counter selection bias threat while blinding of participants counters the effect of social interaction.^[23]

2) Threats to External Validity: is the extent to which the result can be generalized to another context.^[25]

Threats to external validity are:

Situational/Contextual Factors	Specific conditions under which the research was conducted limit its generalizability. ^[24]
Pre-test/Post-tests Effects	Results that can only be found after pre-tests or post-tests. ^[24]
Hawthorne Effects	Participants' reactions to being studied alter their behaviour and therefore the study results. ^[24]
Experimenter Effects	Results are influenced by the actions of the researcher. ^[24]

Table 2: Threats to External Validity

How to overcome these threats:

Replication counteracts all threats by improving generalization to other environments, populations, and conditions. Field experiments counter the evidence and effects of the situation by using natural contexts. Probability sampling counteracts selection bias by ensuring that everyone in a population has an equal chance of being selected for a study sample. Recalibration or reprocessing also counteracts selection bias by using algorithms to correct factor weights within the study samples. ^[23]

The threats to validity that were observed in this thesis:

1. Lighting condition in the lab: As the camera is very sensitive to lighting conditions results may vary if testing is done at a different time of day with different light conditions.

Solution: To perform testing at the same time of day with all the lights off so Mobileye 6 can only perceive light from the digital display.

2. ADAS camera calibration: If the camera is not calibrated properly results may vary.

Solution: Mobileye 6 was calibrated multiple times to get better precision. Every time it was calibrated it was tested for the same test scenario to check if any deviations in the results.

3. Verification of Simulator: Poor simulation resolution may result in poor detection.

Solution: LGSVL simulator was used to get a high-resolution simulation so that Mobileye 6 can detect the NPC agents.

5.2 Warning Detection

CANSee and EyeWatch displays the warnings. The Mobileye CANSee application is installed automatically with the installation of the Mobileye Setup Wizard application.

EyeCAN is connected with the system having CANSee installed. CANSee is set up in normal mode to see all the CAN signals from Mobileye 6. The bus speed selected is 500Kbps. CAN messages having an active bit is colored in blue. CAN messages can be logged, to log the CAN messages log path must be mentioned and active box needs to be ticked.

CAN messages are transmitted in an 11bit CAN header standard ID format. The default baud rate is 500kbps. Messages are transmitted every 66-100 ms.

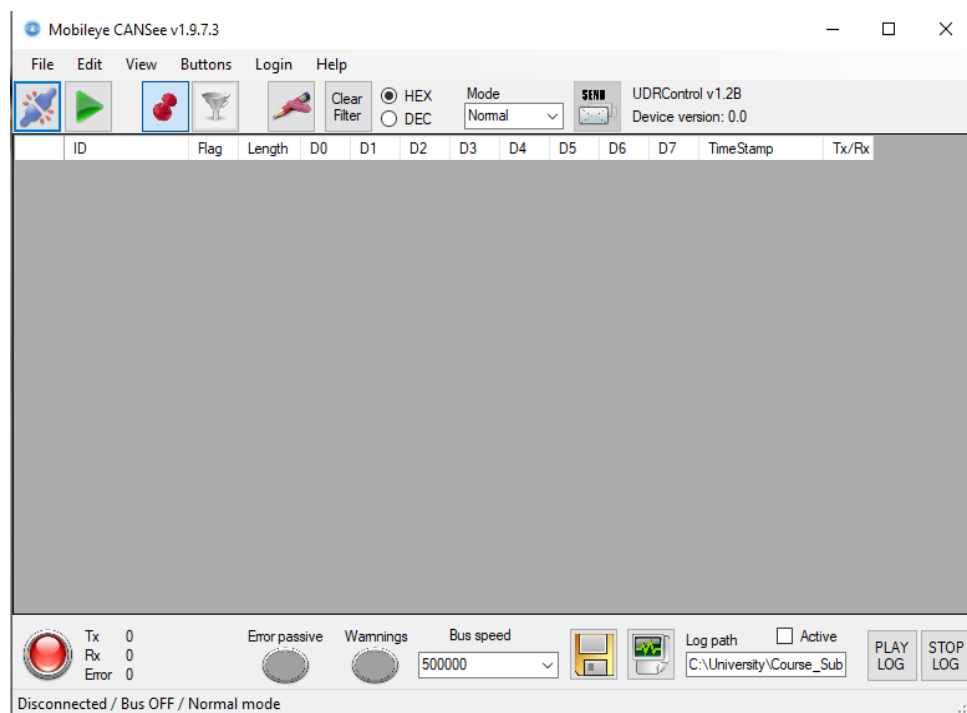


Figure 28: CANSee User Interface

CAN message ID 0x700 provides data about FCW and is monitored for evaluation purpose.

CAN message 0x700 provides data about:

- Display sound type
- Lane Departure Warning Left and Right
- Headway in seconds
- Forward Collision Warning
- Pedestrian detection and warning
- Hi/Low beam detection
- Failsafe events: Low visibility

Bit	7(msb)	6	5	4	3	2	1	0(lsb)
Byte 0	Undocumented			Time indicator		Sound type (0-7) *		
Byte 1	Reserved		Zero Speed	Reserved			0x0	
Byte 2	Headway measurement							Headway valid
Byte 3	Error code							0x0: error 0x1: no error
Byte 4	Failsafe	Maintenance (error)	Undocumented		FCW on	Right LDW ON	Left LDW ON	LDW OFF
Byte 5	TSR enabled	Reserved	Tamper Alert	Reserved		Peds in DZ	Peds FCW	0x0
Byte 6	Reserved					TSR Warning Level		
Byte 7	Reserved					HW repeatable enabled	Headway Warning Level	

Figure 29: CAN Message 0x700

Detailed bit structure if CAN message ID 0x700 is shown in above figure. Byte 2 and Byte 4 are important as they give information on time to collision and failsafe warnings.

Headway measurement byte 2 (bit 7 to 1) is used to measure time to collision in seconds. The value is presented in Hex -ex: 1.0 = 0A. It ranges from 0 to 9.9. Failsafe event i.e., poor visibility and LDW off is given by byte 4 (bit 7 and 0).

On CANSee CAN message ID 0x700 as shown in [figure 30] gives information on headway measurement i.e., time to collision (Byte 2: bit 7 to 1), visibility and lane departure warning off (Byte 4: bit 7 and 0).

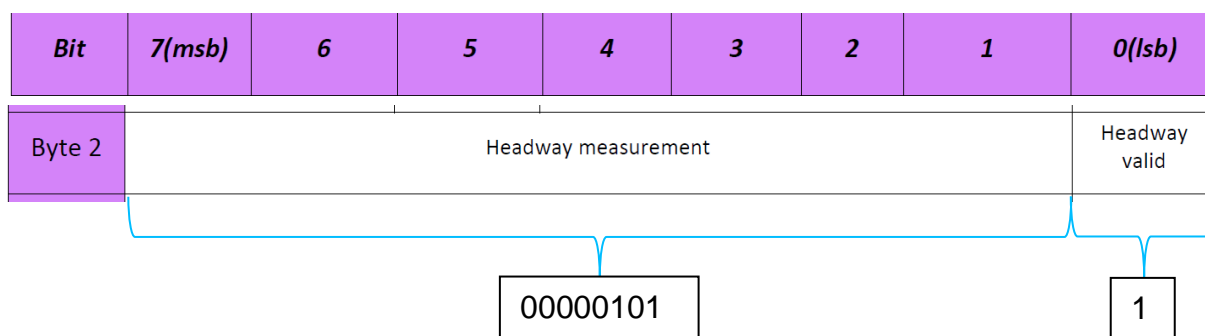


Figure 30: CAN Message-ID 0x700: Byte 2

The value of byte 2 for test cases is 0x0B i.e., (000001011). Giving value 0.5 seconds for headway measurement with headway valid bit set.

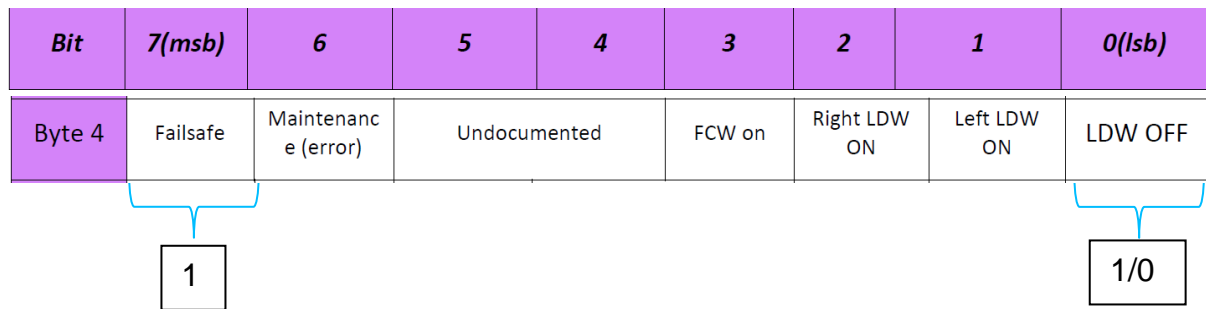


Figure 31: CAN Message-ID 0x700: Byte 4

When the value of byte 4 is 0x80 i.e., (10000000) failsafe bit is set due to poor visibility. If the value is 0x81 i.e., (10000001) failsafe bit along with LDW off bit is set due to poor visibility and poor lane markings.

5.3 Results

Different test cases were created based on one-way traffic scenarios and different climate conditions and daytime.

Test cases are shown in the below table.

Test Case No.	Scenario	Time of Day	Climate condition			CANsee Output ID 0x700		EyeWatch Output	Visibility	FCW		Result
			Fog rate in %	Rain rate in %	Road wetness in %	Byte 4 Poor Visibility	Byte 2 Headway			Expected	Actual	
1	One Way Traffic Scenario	12:00 PM	0	0	0	0x00	0x0B	0.5	Good	Yes	Yes	Pass
2		12:00 PM	0.5	0	0	0x00	0x0B	0.5	Good	Yes	Yes	Pass
3		12:00 PM	1	0	0	0x80	0x0B	0.5	Poor	Yes	Yes	Pass
4		12:00 PM	0	0.5	0	0x00	0x0B	0.5	Good	Yes	Yes	Pass
5		12:00 PM	0	1	0	0x00	0x0B	0.5	Good	Yes	Yes	Pass
6		12:00 PM	0	0	0.5	0x00	0x0B	0.5	Good	Yes	Yes	Pass
7		12:00 PM	0	0	1	0x00	0x0B	0.5	Good	Yes	Yes	Pass
8		12:00 PM	1	1	1	0x80	0x0B	0.5	Poor	Yes	Yes	Pass
9		6:00 PM	0	0	0	0x00	0x0B	0.5	Good	Yes	Yes	Pass
10		6:00 PM	0.5	0	0	0x00	0x0B	0.5	Good	Yes	Yes	Pass
11		6:00 PM	1	0	0	0x80	0x0B	0.5	Poor	Yes	Yes	Pass
12		6:00 PM	0	0.5	0	0x00	0x0B	0.5	Good	Yes	Yes	Pass
13		6:00 PM	0	1	0	0x00	0x0B	0.5	Good	Yes	Yes	Pass
14		6:00 PM	0	0	0.5	0x01	0x0B	0.5	Good	Yes	Yes	Pass
15		6:00 PM	0	0	1	0x01	0x0B	0.5	Good	Yes	Yes	Pass
16		6:00 PM	1	1	1	0x81	0x0B	0.5	Poor	Yes	Yes	Pass
17		11:00 PM	0	0	0	0x00	0x00	-	-	Yes	No	Fail

Table 3: Test Case Result

5.3.1 Test case and result description

The above table shows the different test cases for one way traffic scenarios and their respective results. The test cases have been created for different times of day i.e., 12:00 pm, 6:00 pm, and 11:00 pm with different climate condition combinations which include fog rate, rain rate and road wetness on RoboTestWebApp.

Result describes whether Mobileye 6 was able to detect NPC agents or not in different scenarios. It is expected that Mobileye 6 should detect NPC agents in all the conditions. We can see that for all the test cases Mobileye 6 was able to detect the NPC agent excluding test case no. 17 and provide a warning on EyeWatch with time to collision in seconds and a visibility symbol. The result is also captured on CANSee.

Observations: Using different test scenarios, we were able to evaluate different features of Mobileye 6 such as poor visibility warning in case of fog, and lane departure warning off in case of poor lane marking.

Observation 1:

1) From table 1. test cases no. 3, 8, 11, and 16 irrespective of the time of day, when the fog rate is 100%, EyeWatch displayed the poor visibility symbol as shown in the figure. On CANSee, CAN message ID 0x700, byte 4 value is 0x80 i.e., failsafe bit is set which means poor visibility.



Figure 32: EyeWatch: Poor Visibility

Mobileye CANSee v1.9.7.3

ID	Flag	Length	D0	D1	D2	D3	D4	D5	D6	D7	Time Stamp	Tx/Rx
0x400	Reg	8	48	05	5B	02	00	00	08	00	1554.198	Rx
0x401	Reg	8	45	05	05	72	0A	05	5A	06	1554.199	Rx
0x402	Reg	8	0F	00	0F	00	06	10	08	03	1554.199	Rx
0x403	Reg	8	6F	00	FF	03	00	00	00	00	1554.199	Rx
0x410	Reg	8	21	3C	04	00	00	00	00	00	1554.199	Rx
0x411	Reg	8	80	0F	F7	5A	00	00	00	00	1554.2	Rx
0x412	Reg	8	02	0A	02	06	05	03	00	00	1554.2	Rx
0x700	Reg	8	00	00	0B	01	80	00	00	02	1554.197	Rx
0x760	Reg	8	00	80	50	00	00	00	00	00	1554.198	Rx
0x727	Reg	8	FE	00	FE	00	FE	00	FE	00	1554.198	Rx
0x703	Reg	7	80	00	8A	00	78	00	0C	00	1554.198	Rx

Tx: 0, Rx: 219776, Error: 0
 Error passive: [X], Warnings: [X]
 Bus speed: 500000
 Log path: C:\University\Course_Sub
 PLAY LOG, STOP LOG
 Connected / Bus ON / Normal mode

Figure 33: CANSee: ID 0x700 (Byte 4: 0x80)

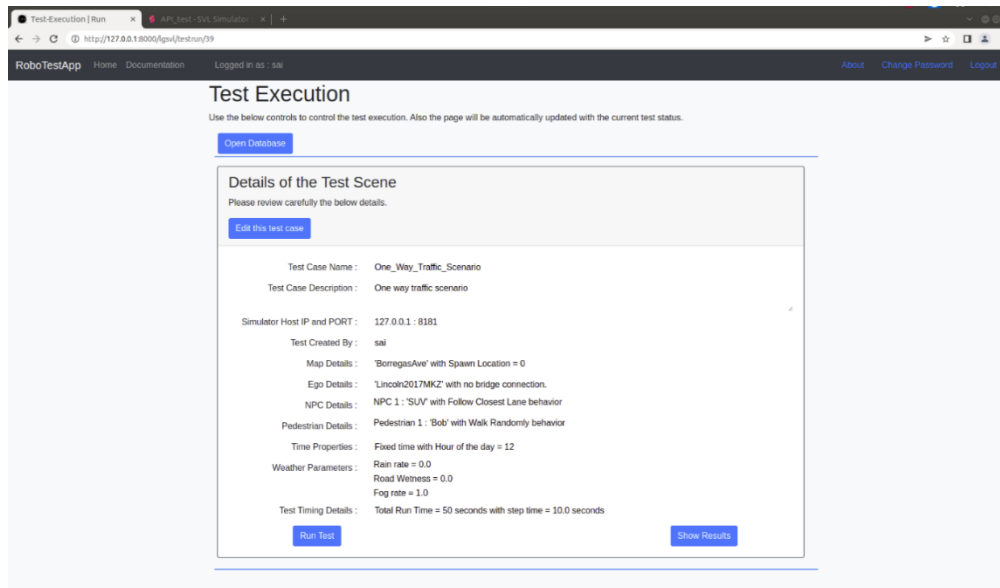


Figure 34: Test Case Details on RoboTestWebApp: 100% Fog



Figure 35: LGSVL Simulator: 100% Fog

Observation 2:

2) For test case no. 16 at 18:00 hr. of day, with fog rate, rain rate, and road wetness to 50% and 100%, EyeWatch displayed poor visibility and lane departure warning off. On CANSee, CAN message ID 0x700, byte 4 value is 0x81 i.e., failsafe bit and LDW off bit is set which means poor visibility and lane departure warning is off.

Observation 3:

Test case number 17 is based on a report from auto motor and sports as described in chapter 1 section 1.1.1.4 Blind spots. This gives assistance system Automatic Emergency Braking strongly (AEB) depends on headlights. As AEB relies on a camera

sensor and for a camera sensor to detect an object requires a good amount of light. Using the same case from the auto motor and sports magazine test case number 17 has been created. In this test scenario time of day is 11:00 pm and the headlights of the ego vehicle are off, and no streetlights are present. For which Mobileye 6 fails to detect NPC agent.

3) For test case no. 17 at 23:00 hr. of day, when the NPC agent is in the dark region with no streetlights, Mobileye 6 does not detect the NPC agent. No warning signed observed on EyeWatch as well on CANSee.

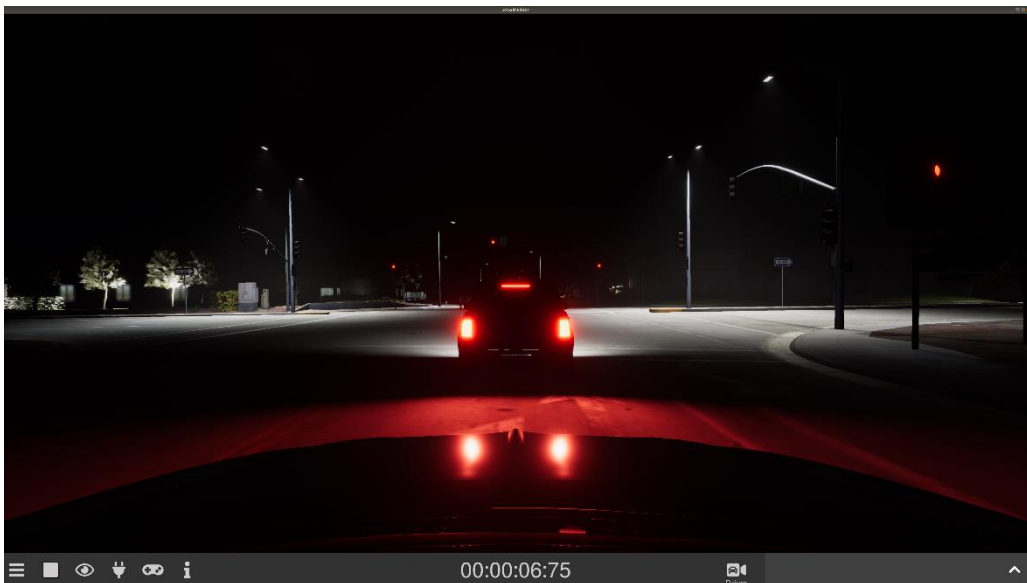


Figure 36: Ego Vehicle with Headlights Off



Figure 37: EyeWatch with No Warnings

From the above test result, we can say that Mobileye 6 was able to detect NPC agents in all the scenarios except scenario number 17. The different warning was shown on EyeWatch as well as on CANSee such as poor visibility lane departure warning, and forward-collision warning. Mobileye 6 was successfully tested, and the blind spot was found.

Performance comparison of Mobileye 6 with MiniEye.

The result for MiniEye is obtained from MT-3166

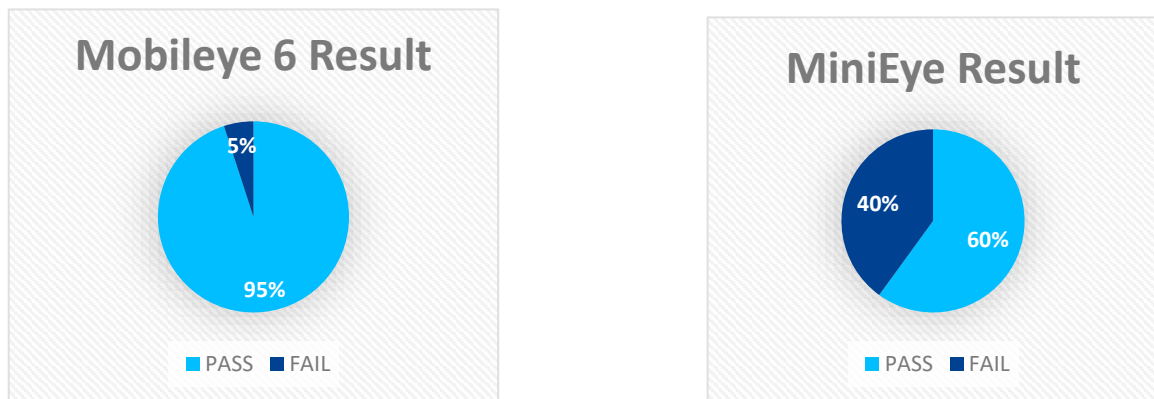


Figure 38: Performance comparison of Mobileye 6 and MiniEye

6 Conclusion

A thorough literature review was performed on why it is necessary to test the ADAS system. Of Which the important reasons are safety and identification of blind spots.

Different test design techniques (testing schemes) were analyzed and studied like STPA, FMEA, FTA and RSS. From which RSS model was selected as the test design because the RSS model gives different driving scenarios which can lead to accidents if a safe distance is cut. Which is one of the suitable methods for testing convenience ADAS systems like vision-based system (Mobileye 6) which is used for warning drivers of potentially dangerous situations.

For test case creation different parameters like Environment condition, road geometry and dynamic conditions like traffic density were studied from which one-way traffic scenarios with environmental conditions like rain, fog and road wetness were used in the creation of the test case. The boundary value test case creation method was used. RoboTestWebApp which is a product of MT-3136 was used for creating test scenarios along with the LGSVL simulator. A few limitations of RoboTestWebApp were observed while creating test cases such as multiple geometry scenarios were not supported. Also, Apollo 5.0 was not supported for auto speed control of Ego vehicles in the LSGVL simulator.

HIL environment was used for testing Mobileye 6. Response of Mobileye 6 for different test cases was captured were for one of the test cases during the night with headlights off and no streetlights the Mobileye 6 failed to detect the NPC agent. Therefore, the forward collision warning function of Mobileye 6 still needs to be Improved.

In comparison with MiniEye as shown in [figure 38], Mobileye 6 provide better performance. Mobileye 6 passed for all the test scenarios except one. While from the result of MT-3166 MiniEye failed almost in 40% of cases and passed in 60 % of test cases. Hence the performance of Mobileye 6 is far better than MiniEye.

6.1 Future Work

6.1.1 More Test Cases

More test cases can be added to evaluate the performance of Mobileye 6 like two-way traffic scenarios, multiple geometry scenarios, vulnerable road users' scenarios, and cut-in and drifting scenarios as given in the RSS model.^[13]

6.1.2 Automation

Instead of sending constant speed from the Simulink model, get real-time simulated ego vehicle speed from the simulator and send it to Mobileye 6.

Appendix

This chapter gives information on Mobileye 6 components and its installation and how to launch the LGSVL Simulator.

Mobileye 6 Main Unit

The Mobileye 6 Main Unit consists of the following components:

Camera unit, high-quality audio alert buzzer, Mobileye chip onboard system and connector cable.^[8]



Figure 39: Mobileye 6 Camera Sensor

Mobileye 6 EyeWatch

It displays warnings to drivers such as forward collision warning (FCW), lane departure warning, etc.^[8]



Figure 40: Mobileye EyeWatch Display

Mobileye 6 EyeCAN

It is used for Mobileye system calibration and updating firmware versions and to sniff CAN signal from Mobileye.^[8]



Figure 41: Mobileye EyeCAN

Mobileye CANSee

The Mobileye CANSee Application is a real-time viewer for CAN-bus messages developed by Mobileye for CAN-Bus analysis (CAN sniffing).^[8]

The CANSee application is functional with the Mobileye EyeCAN.^[8]

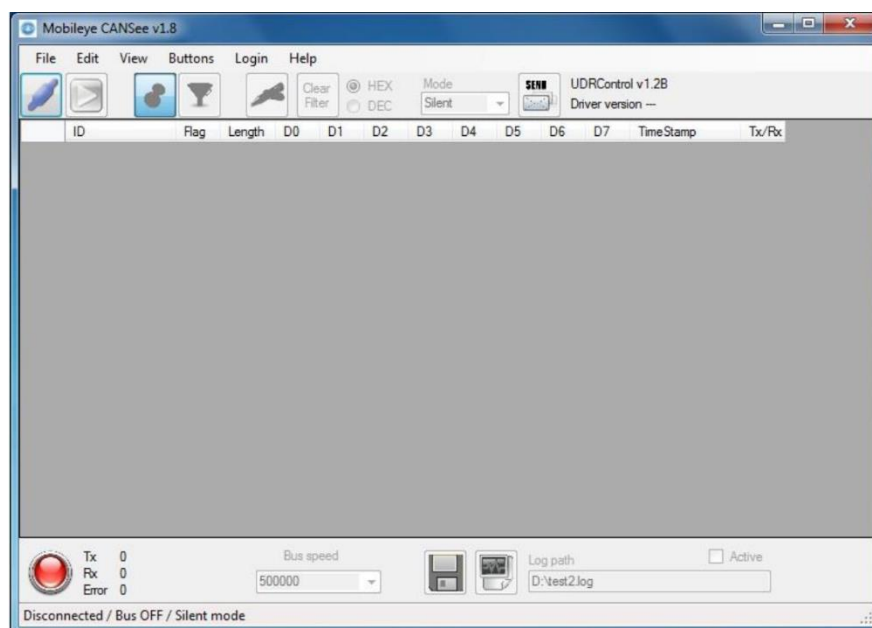


Figure 42: Mobileye CANSee User Interface

Mobileye 6 Installation

To have a car windshield like structure to mount a camera, we are using an already designed product that is installed in the IAS laboratory.



Figure 43: Windshield

Once the windshield is placed properly, the next step is to mount the Mobileye 6 main unit. The main unit should preferably be placed at the top of the windshield and the center of the vehicle width. Here we consider the center of the digital display.

Camera Installation

Once the camera is mounted the next step is installation. The installation procedure can be found in Mobileye 6 installation guide.^[8]

Camera Calibration

After installation, the next step is to calibrate Mobileye 6. Before starting with the calibration process, we need to install the Mobileye setup wizard. The details can be found in Mobileye Setup Wizard – Installation instruction v1.9.^[8]

Before starting calibration adjust the camera lens if required and take the camera measurements.

There are two types of calibration processes:

- Manual calibration using TAC board.
- Automatic calibration.

Using any of the calibration processes Mobileye 6 can be calibrated. The details regarding the calibration process are given in Mobileye 6 installation guide.^[8]

Automatic calibration was performed in this thesis.

Steps for automatic calibration:

1. Login to Mobileye Setup Wizard Application

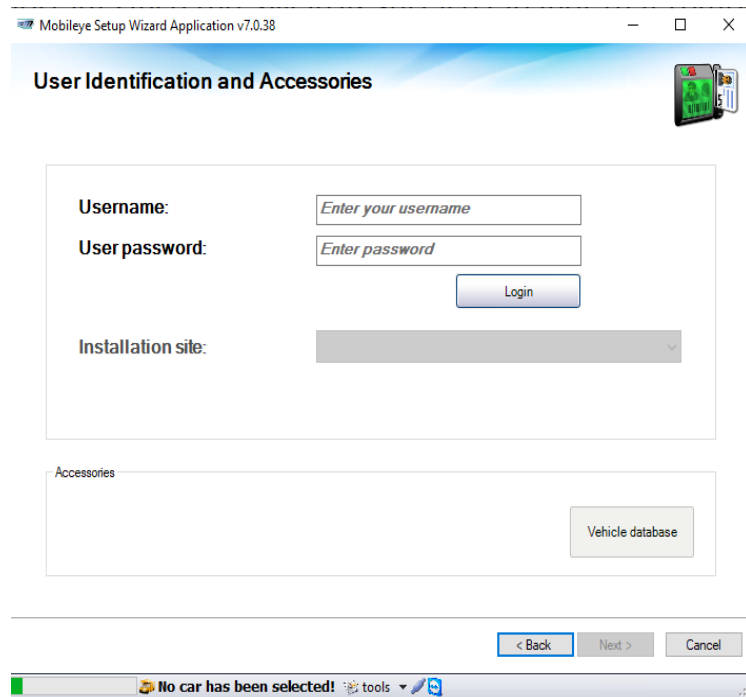


Figure 44: Mobileye Setup Wizard Login

2. Select a vehicle from the database

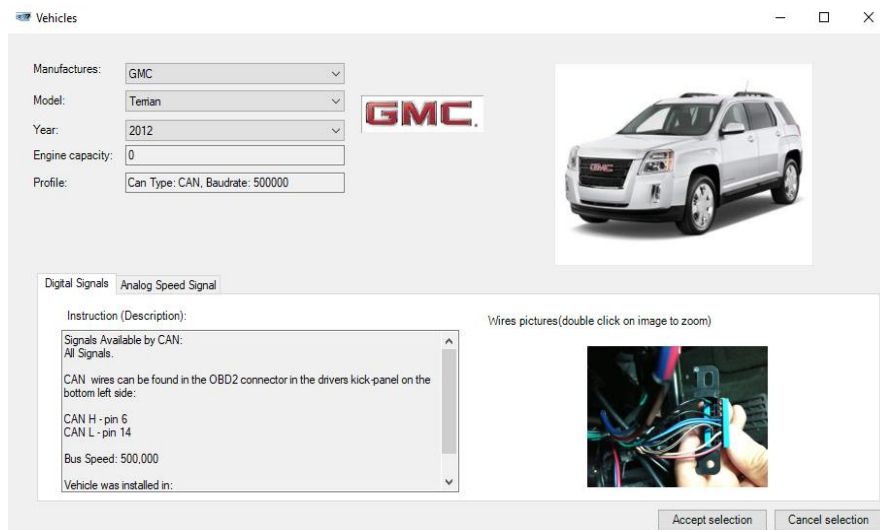


Figure 45: Vehicle Selection

3. Add vehicle information and measurements

Camera Installation

Camera Video

207

☒ Show grid

Vehicle

Vehicle chassis num
23-645-83

Manufacturer
Citroen

Model
C3...France,,OBD2, Y.2010, #.(2006-20

Production year
2010

Other...

Measurements

Camera height [Meter] 1.41

Distance to bumper [Meter] 1.42

Vehicle width [Meter] 1.72

Camera to windshield edge

Left [Meter] 0.6

Right [Meter] 0.59

< Back Next > Cancel

Citroen, C3,,,France,,OBD2, 2010

Figure 46: Vehicle Information and Measurements

4. From camera calibration select automatic calibration

Camera Calibration

Choose calibration method

Automatic Calibration

TAC

< Back Next > Cancel

No car has been selected!

Figure 47: Automatic Calibration Selection

5. Select signal source example speed and source as CAN

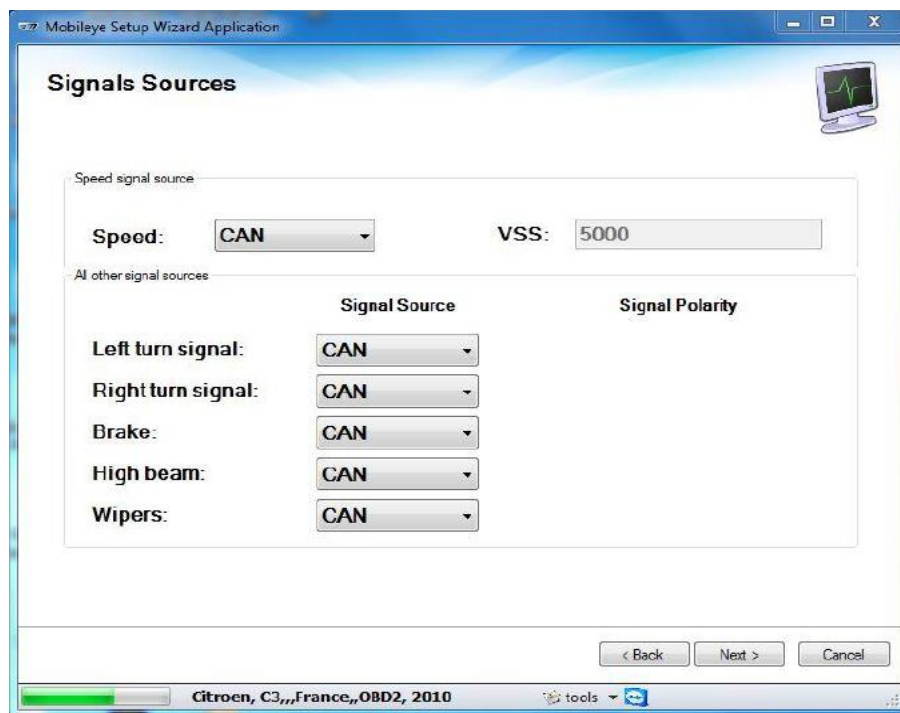


Figure 48: Signal Source Selection

Once everything is selected, Mobileye will enter automatic calibration mode. It will automatically calibrate during the first drive.

“CL” will be displayed on EyeWatch, and it will start counting from 0 to 99. It will take around 5 to 10 minutes for calibration.



Figure 49: Automatic Calibration

In case, if any warning errors are displayed on EyeWatch refer to the warning error manual.^[8]

LGSVL Installation

The LGSVL Simulator and Apollo 5.0 can be cloned from the university GitHub enterprise account and follow the instruction given in the respective readme files to install and setup the environment. After setting up the environment, you should be able to create a simulation in the LGSVL simulator and the connected Apollo should be able to control the EGO vehicle.

LGSVL launching steps:

1. Install the latest LGSVL simulator.^[7]
2. Link to cloud



Figure 50: LGSVL Simulator Link to Cloud

3. Login to the LGSVL account

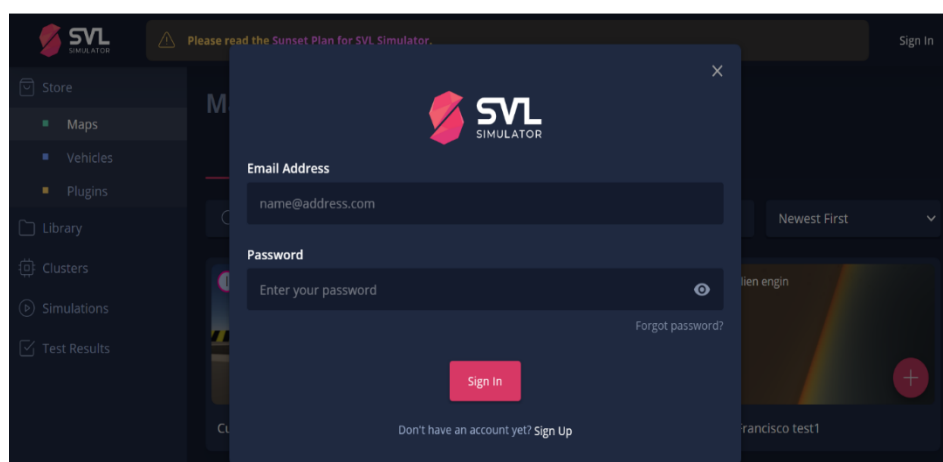


Figure 51: LGSVL Login

4. Create a new cluster

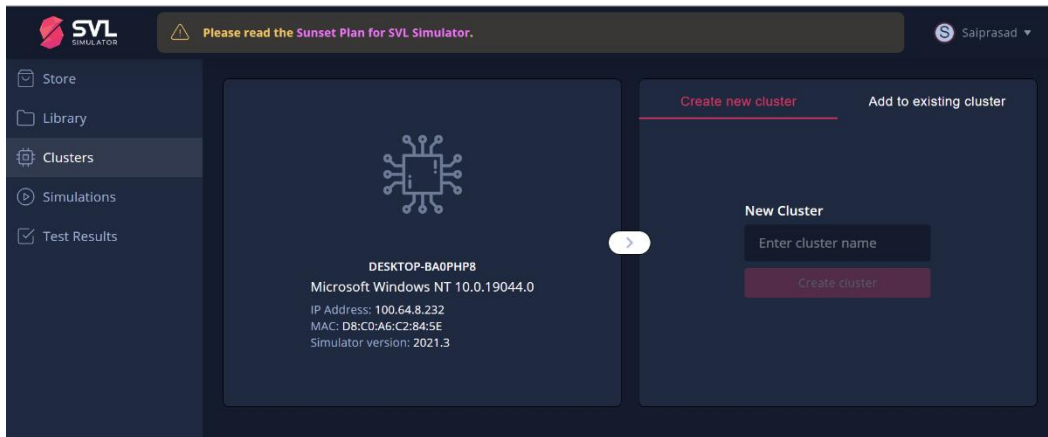


Figure 52: LGSVL New Cluster

Once the cluster is created, download maps, vehicles, and plugins from the library.

5. In Simulation, add a new simulation and select runtime template as API only.

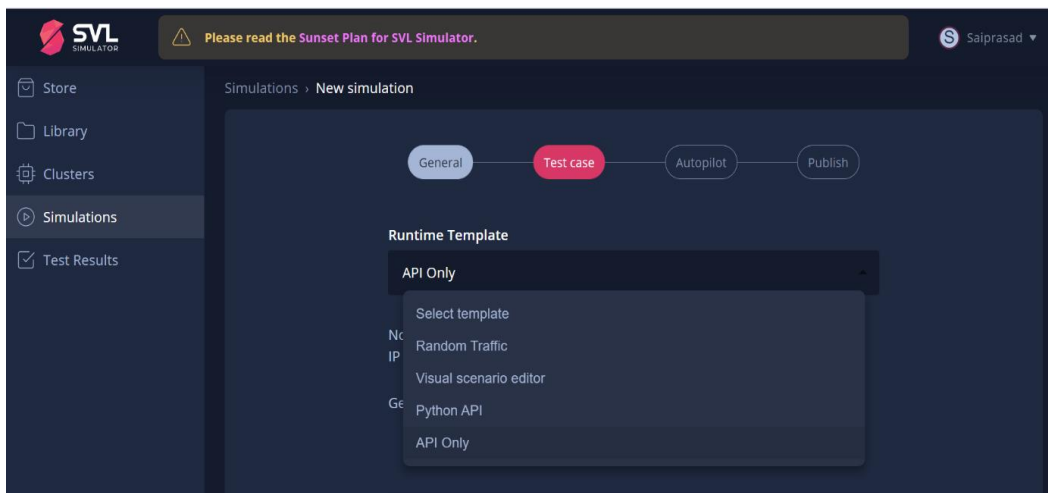


Figure 53: LGSVL New Simulation

Mobileye 6 Features:





Warnings	Descriptions	Considerations
	<p>Forward Collision Warning</p> <p>The FCW alerts you up to 2.7 seconds before an imminent collision with a vehicle or motorcycle ahead, both on highways and in urban areas.</p> <p>Under 30kph/19mph, Urban FCW alerts with a double beep, acting as a 4 meter virtual bumper as well.</p>	<p>The FCW alerts is always active, works from 30kph/19mph and cannot be disabled, muted or adjusted.</p>
	<p>Pedestrian & Cyclist Collision Warning</p> <p>The PCW alerts you up to 2 seconds before imminent collision with a pedestrian or cyclist ahead, allowing you enough time to react.</p>	<ol style="list-style-type: none"> 1. Operational during daylight hours only and at speeds under 50kph/31mph. 2. PCW cannot be disabled or muted. 3. Danger Zone -rectangle of 30m/99ft length (starting from the front bumper of the car) and 1.8 m/6ft from each side of the vehicle.
	<p>Lane Departure Warning</p> <p>The LDW alerts you with visual and audio warnings when there is an unintentional deviation from the driving lane without signaling.</p> <p>Yellow lane icon appears when LDW is not available; white lane icon appears when it is.</p>	<p>The LDW is active:</p> <p>At speeds greater than 55kph/34mph as default value.</p> <p>The LDW is inactive:</p> <ol style="list-style-type: none"> 1. When the driver uses the turn signals before changing lanes. 2. Below 55kph/34mph as default value. 3. If lanes are unmarked or poorly marked. 4. The system has been muted. 5. A sharp turn is made.
	<p>Headway Monitoring Warning</p> <p>HMW assists the driver in keeping a safe driving distance from the vehicle in front, by issuing an alert when the headway distance to the vehicle ahead becomes dangerous (less than 2 seconds).</p> <p>HMW displays the distance (measured in seconds) to the current position of the vehicle in front and issues an alert if the distance is lower than or equal to a pre-defined threshold.</p> <p>Green vehicle icon signifies safe headway; red icon unsafe.</p>	<p>HMW is available:</p> <p>When speed is greater than 30kph/19 mph.</p> <p>HMW alert is not active:</p> <ol style="list-style-type: none"> 1. When speed is below 30kph/19 mph. 2. The alert has been muted. 3. When a passing vehicle cuts in fronts of you in your lane and drives away quickly.

Table 4: Mobileye 6 Features

Bibliography

- [1] M. Wood, Dr. P. Robbel, Dr. M. Maass, Dr. R. D. Tebbens, M. Meijs, M. Harb, J. Reach, K. Robinson, D. Wittmann, T. Srivastava, M. E. Bouzouraa, S. Liu, Y. Wang, C. Knobel, D. Boymanns, M. Löhning, B. Dehlink, D. Kaule, R. Krüger, J. Frtunikj, F. Raisch, M. Gruber, J. Steck, J. Mejia-Hernandez, S. Syguda, P. Blüher, K. Klonecki, P. Schnarz, T. Wiltshko, S. Pukallus, K. Sedlaczek, N. Garbacik, D. Smerza, D. Li, A. Timmons, M. Bellotti, M. O 'Brien, M. Schöllhorn, U. Dannebaum, J. Weast, A. Tatourian, B. Dornieden, P. Schnetter, P. Themann, T. Weidner, P. Schlicht, "Safety First for Automated Driving," 2019, pp.72-97.

- [2] F. Reway, W. Huber, E. P. Ribeiro, "Test Methodology for Vision-Based ADAS Algorithms with an Automotive Camera-in-the-Loop," in 2018 IEEE International Conference on Vehicular Electronics and Safety (ICVES), September 2018.

- [3] B. Sari, "Fail-operational Safety Architecture for ADAS/AD Systems and a Model-driven Approach for Dependent Failure Analysis," "Wissenschaftliche Reihe Fahrzeugtechnik Universität Stuttgart, 2019, pp.31-74.

- [4] Wu Shuang, "ADAS Forward-looking camera hardware-in-loop testing development under corner test cases," IAS-MT3166, pp.1-62, March 2021.

- [5] T. Kirby, "Camera-In-The-Loop Test Bench Development for Advanced Driver Assistance Systems Applications," Engineering Undergraduate Research Theses and Honors Research Theses, May 2019.

- [6] J. John, "Development of a tool for the collection, prioritization and application of test cases," IAS-MT3136, pp. 1-80, November 2020.

- [7] L. Electronics, "LGSVL Simulator," 2020. [Online]. Available: <https://www.lgsvlsimulator.com/>.

- [8] Mobileye, "Mobileye 6 installation Guide," [Online]. Available: <https://www.mobileye.com/support/>.
- [9] Hungar, Hardi, Frank Köster, and Jens Mazzega. "Test specifications for highly automated driving functions: Highway pilot." 2017.
- [10] Single Vehicle Crash, [Online]. Available: <https://drivetestlacanada.ca/news/pony-ai-loses-driverless-testing-permit-in-california-following-single-vehicle-crash/>
- [11] Tesla Crash California, [Online]. Available: <https://insideevs.com/news/507038/nhtsa-investigate-tesla-crash-california/>
- [12] Tesla Crash Texas, [Online]. Available: <https://insideevs.com/news/502265/nhtsa-ntsb-tesla-crash-texas/>
- [13] RSS model, [Online]. Available: https://static.mobileye.com/website/corporate/rss/rss_on_nhtsa.pdf
- [14] J. Schindler, "Validation: Metrics & KPIs for Autonomous vehicles," AutoMate Workshop, Nov 2018, pp. 1-16.
- [15] S. Khastgir, S. Brewerton, J. Thomas, P. Jennings, "Systems Approach to Creating Test Scenarios for Automated Driving Systems." 2021, pp. 1-14.
- [16] R. Kent, "Design quality management." 2016.
- [17] D. Kritzing, "Fault tree analysis." 2017.
- [18] Three Ways of ADAS Testing in Autonomous Cars, [Online]. Available: <https://intellias.com/three-ways-of-testing-adas-in-autonomous-cars-beyond-a-test-drive>
- [19] S. Sulaman, A. Beer, M. Felderer, M. Host, "Comparison of the FMEA and STPA safety analysis methods—a case study," pp.3-4, 2017.

- [20] Hillenbrand, Martin & Heinz, Matthias & Adler, Nico & Matheis, Johannes & Müller-Glaser, Klaus. (2010). Failure mode and effect analysis based on electric and electronic architectures of vehicles to support the safety lifecycle ISO/DIS 26262. 1-7. 10.1109/RSP.2010.5656351.
- [21] L. Shoults, "Implementation of Design Failure Modes and Effects Analysis for Hybrid Vehicle Systems, "pp.1-2, May 2016.
- [22] Software-in-the-loop Testing, [Online]. Available: <https://www.aptiv.com/en/insights/article/what-is-software-in-the-loop-testing>
- [23] Threats to Validity, [Online]. Available: https://cyfar.org/ilm_3_threats
- [24] Threats to Validity, [Online]. Available: <https://www.scribbr.com/methodology/external-validity/>
- [25] Threats to Validity, [Online]. Available: <https://www.scribbr.com/methodology/internal-validity/>
- [26] B. Heinemann, "On-Road Intelligent Vehicles," pp.59-82, 2016.
- [27] C. De Locht, H. Van Den Broeck, "Complementary metal-oxide-semiconductor (CMOS) image sensors for automotive applications," in High-Performance Silicon Imaging, pp.235-249, 2014.
- [28] Testing Advanced Driver-Assistance Systems, [Online]. Available: <https://www.intertek.com/blog/2020-09-08-adas-testing>
- [29] Diese Assistenzsysteme sind ab 2022 vorgeschrieben, [Online]. Available: <https://www.auto-motor-und-sport.de/verkehr/diese-assistenzsysteme-sind-ab-2022-vorgeschrieben/>
- [30] The auto motor and sports, "Assistance system weaken in the dark," [Online]. Available: <https://www.auto-motor-und-sport.de/verkehr/city-notbremsassistent-pflicht-schwaeche-dunkelheit/>

- [31] HIL Testing, [Online]. Available: <https://www.hil-simulation.com/home/hil-testing.html>

Declaration of Compliance

I hereby declare to have written this work independently and to have respected in its preparation the relevant provisions, in particular those corresponding to the copyright protection of external materials. Whenever external materials (such as images, drawings, text passages) are used in this work, I declare that these materials are referenced accordingly (e.g. quote, source) and, whenever necessary, consent from the author to use such materials in my work has been obtained.

Signature: Saiprasad Salkar

A handwritten signature in blue ink, appearing to read 'Salkar', with a horizontal line extending to the right.

Stuttgart, on the 20.04.2022