2022

Non-Line of Sight Test Scenario Generation for Connected Autonomous Vehicle

PhD Thesis

Tanvir Allidina

This thesis is submitted in partial fulfilment of the requirements for the Doctor of Philosophy

Faculty of Computing, Engineering and Media

De Montfort University

January 2022

Declaration of Authorship

I, Tanvir Allidina declare that this thesis entitled Non-Line of Sight Test Scenario Generation for Connected Autonomous Vehicle and the work presented in it are my own and original. It is submitted for the fulfilment of the requirements for the degree of Doctor of Philosophy at De Montfort University. The work was undertaken between October 2016 and January 2022.

ABSTRACT

Connected autonomous vehicles (CAV) level 4-5 use sensors to perceive their environment. These sensors are able to detect only up to a certain range and this range can be further constrained by the presence of obstacles in its path or as a result of the geometry of the road, for example, at a junction. This is termed as a non-line of sight (NLOS) scenario where the ego vehicle (system under test) is unable to detect an oncoming dynamic object due to obstacles or the geometry of the road.

A large body of work now exist which proposes methods for extending the perception horizon of CAV's using vehicular communication and incorporating this into CAV algorithms ranging from obstacle detection to path planning and beyond. Such proposed new algorithms and entire systems needs testing and validating, which can be conducted through primarily two ways, on road testing and simulation. On-road testing can be extremely expensive and time-consuming and may not cover all possible test scenarios. Testing through simulation is inexpensive and has a better scenario space coverage. However, there is currently a dearth in simulated testing techniques that provides the environment to test technologies and algorithms developed for NLOS scenarios.

This thesis puts forward a novel end-to-end framework for testing the abilities of a CAV through simulated generation of NLOS scenarios. This has been achieved through following the development process of Functional, Logical and Concrete scenarios along the V-model-based development process in ISO 26262. The process begins with the representation of the NLOS environment (including the digital environment) knowledge as a scalable ontology where Functional and Logical scenarios stand for different abstraction levels. The proposed new ontology comprises of six layers: 'Environment', 'Road User', 'Object Type', 'Communication Network', 'Scene' and 'Scenario'. The ontology is modelled and validated in protégé software and exported to OWL API where the logical scenarios are generated and validated. An innumerable number of "concrete" scenarios are generated as a result of the possible combinations of the values from the domains of each concept's attributes. This research puts forward a novel genetic- algorithm (GA) approach to search through the scenario space and filter out safety critical test scenarios. A critical NLOS scenario is one where a collision is highly likely because the ego vehicle was unable to detect an obstacle in time due to obstructions present in the line-of-sight of the sensors or created due to the road geometry. The metric proposed to identify critical scenarios which also acts as the GA's fitness function uses the time-to-collision (TTC) and total stopping time (TST) metric. These generated critical scenarios and proposed fitness function have been validated through MATLAB simulation. Furthermore, this research incorporates the relevant knowledge of vehicle-to-vehicle (V2V) communication technologies in the proposed ontology and uses the communication layer instances in the MATLAB simulation to support the testing of the increasing number of approaches that uses communications for alerting oncoming vehicles about imminent danger, or in other word, mitigating an otherwise critical scenario.

Acknowledgement

First and foremost, I would like to begin by thanking God the Almighty, without who's support this PhD will never have been possible, followed by my heartfelt gratitude to my funders: Muhammad Hussein Jaffer, Sajjad M Rashid, Khoja Shia Ithna Asheri Jamaat – Mombasa and Africa Federation of Khoja Shia Ithna Asheri for providing me the loans and grants for pursuing my PhD

I would like to express my sincerest appreciation to my first supervisor (and mentor) Dr. Lipika Deka for not only proposing the research topic but for also being continuously and reliably available throughout. She has been a pillar of strength and support throughout my PhD journey, and I will remain forever indebted to her for her friendship, guidance, advice, and support.

I would like to extend my thanks to my second supervisor Dr. Daniel Paluszczyszyn who has also been very generous with his time and advice. Whose guidance and advice have helped me grow academically as well as a developer. In addition, I would like to thank my supervisory team along with Professor David Elizondo and Dr. Jethro Shell for providing me with the necessary feedback as my research developed. I cannot thank them enough for allowing me to draw from their collective knowledge and experience in their respective fields, an invaluable source of information for this thesis.

I would like to thank Dr. George Okeya, for your expertise in ontology modelling and for making me practice my Swahili

I would like to thank Abdul Hakeem Ibrahim, for supporting me with his expertise in Ontology and OWL API. He has been a constant source of valuable technical input, suggestions, and support during my research journey.

I would like to thank my peers for their company and simulating conversation: Nick, Saul, Javier and Siham.

I would like to thank my parents Hasnain and Arzina Allidina, and my siblings; Mehdi, Ayman and Mohd Abbas for their love, motivation, and support in my journey. I would also like to thank my grandparents (Banu Ma and GulamAbbas Allidina and Nargis and Mustafa GulamAbbas Abdulrasul Soda) for their love and prayers.

I am extremely grateful to my dearest friend Juvereya and her wonderful family (Najma Mulla, Moaaz, Zainab, Ahmed, Muhammed, Aisha and Zakariya) for accommodating me and welcoming me as part of the family. I thank them for providing me with a family surrounding that I will always call my "second home" and showering me with moral support, love, and patience whenever I missed my home or family.

I would like to thank my cousins Asif & Fatema Allidina, Shahid Ali & Siddika, I cannot express my gratitude enough to them all for opening both heart and home to me when I first arrived in the country and for the innumerable and unique ways, they have helped make my life easier as I have continued my stay and studies.

I would like to thank my best friend Syed Umair Iftikhar who has been a great rock through this journey. For your constant love and support. It would been impossible to get this far without you.

I would like to dedicate this research in the memory of my beloved grandmother (Nargis Abdul Rasul), who was kind, immensely generous of spirit and had a witty sense of humour. It is impossible to express how much of an influence she has had on me and what a driving force she has been in my life. This dissertation stands as a testimony to her love and encouragement, and I hope and pray this would have made her proud.

Contents

List of Abbreviation

Abbreviation	Definition
AV	Autonomous vehicle
ADAS	Advance Driver Assistance System
SAE	Society of Automotive Engineers
NLOS	Non-line of sight
V2V	Vehicle to Vehicle
V2X	Vehicle to Everything
CAV	Connected Autonomous Vehicle
TTC	Time to Collision
TST	Total Stopping Time
CP	Collision Point
TD	Time-to-Detect
KPI	Key Performance Indicators
GA	Genetic Algorithm
ADS	Autonomous Driving System
CT	Combinatorial Testing
SOTIF	Safety of Intended Functionality
ISO	International Organisation for Standardisation

Chapter 1: Introduction

1.1. Background

Autonomous vehicle (AV) technology, in the form of advanced driver assistance systems (ADAS) to fully autonomous vehicles, is becoming more prevalent in our society. The Society of Automotive Engineers (SAE International) (SAE International, 2018) defines six levels of autonomy which are as follows:

- Level 0: Manual control. The human performs all driving tasks, though often assisted through technology such as a camera for collision detection and warning
- Level 1: Driver Assistance. The human driver is largely in control apart from specific tasks, such as automated systems like cruise control on highways, where the driver is able to give up control for a substantial period
- Level 2: Partial Automation - ADAS. The vehicle can perform more complex tasks combining different actions such as steering and acceleration. However, the human still is in charge of all driving task.
- Level 3: Conditional Automation. The vehicle is capable of environmental detection capabilities. The vehicle can perform most driving tasks in certain allowed conditions, but the human must take over when the system alerts or when the human sees the need to do so. It is considered as the first entry point to a fully autonomous system whereby the humans can do other tasks while the car is driving itself.
- Level 4: High Automation. In this level, under certain conditions, the vehicle performs all driving tasks on routine driving routes. If the vehicle encounters conditions, such as heavy snow, which are at the boundary of its capabilities, it signals the driver to take over but can still secure itself if the driver ignores. Human override is still an option.
- Level 5: Full Automation. In this level, under all conditions, the vehicle completes all driving tasks. There is no need for human interaction or attention.

The focus of this thesis is on AV and CAV in SAE Level 4-5. For an AV autonomy may involve a single function autonomy (Level 1-2) to all function autonomy (Level 4-5) which means that to successfully operate it requires the ability to navigate from a source to destination through a network of roads, avoiding obstacles and following traffic rules. To navigate safely and efficiently, information required includes accurate position and heading details of itself and the surrounding objects; also referred to as the localisation and perception phase of the autonomous driving algorithmic cycle followed by the planning, control, and system management phase. To accomplish the task, localisation information is required in reference to an accurate, robust, consistent and an up-to-date map. Autonomous vehicles gather information on obstacles and understand the world around

them as they travel using on-board sensing technologies. An autonomous vehicle sensors include: Exteroceptive sensors, which are used to perceive the environment and calculate distances between objects, whereas proprioceptive sensors are used to measure values from within the system, such as motor speed, wheel position, joint angles, and so on (Suganuma, 2012), (Campbell *et al.*, 2018), (Ilci and Toth, 2020).

I. Exteroceptive Sensors:
    a. Light Detection and Ranging (LiDAR): LiDAR is a form of remote sensing technology used to calculate distances. It operates on the time of flight (TOF) principle, which entails firing a pulsed laser and measuring how long it takes for the pulse to be reflected back. These measurements can then be used to create three-dimensional representations of the surrounding environment. LiDAR sensors can measure distances at rates of more than 150 kilohertz (150,000 pulses per second) (Ilas, 2013)(Royo and Ballesta-Garcia, 2019) and are classified as long-range sensors with a range of more than 250 metres.
    b. RADAR: Radar is a technique that uses radio waves to measure the distance, angle, and velocity of objects. It works on the principle of electromagnetic radiation, which can be used in a wide range of frequency bands. (For example, 24 GHz, 77 GHz, and 79 GHz). The higher the frequency, the higher the resolution, allowing the radar sensor system to differentiate between various objects in real time. The most common radar sensors are short to medium range (50-100m); however, some radar sensors can detect an object at a distance of more than 150m. (Lee *et al.*, 2010)(Roos *et al.*, 2019).
    c. Camera: Using the principle of passive light sensors, a camera generates a digital image of a covered area. Cameras can detect both moving and stationary objects in their environment. Cameras are distinguished from other types of sensors by their ability to see colours and textures. This is a significant benefit for improving the perception system of an autonomous vehicle because the technology allows the vehicle to recognise road signs, traffic lights, and other objects. Cameras can calculate the distance to a specific object as well, but this requires the use of complex processing algorithms.(Liu *et al.*, 2010)(Vriesman *et al.*, 2021).
    d. Ultrasonic: An ultrasonic sensor is a device that uses sound waves to measure the distance between two objects. A sound wave at a specific frequency is emitted towards an object, and the time it takes for the wave to return is used to calculate the distance.(Park *et al.*, 2008)(Xu *et al.*, 2018).

II. Proprioceptive Sensors:
    a. Satellite-based navigation system: Satellite-based navigation systems, such as the United States owned Global Positioning System (GPS) using four or more satellite, GPS can provide geolocation and time information to a compatible radio receiver anywhere on the planet. A satellite-based radio receiver can calculate its position by timing signals sent from orbiting

satellites using the 'trilateration' method. Trilateration is the process of determining absolute or relative point locations by measuring distances using the geometry of circles, triangles, or spheres. (Rahiman and Zainal, 2013)(Perea-Strom *et al.*, 2020).

b. Inertial Measurement Unit (IMU): An IMU is a type of electronic sensor that measures a body's force, angular rate, and magnetic field. In most IMU devices, three accelerometers, gyroscopes, and magnetometers are used, one for each of the orthogonal X, Y, and Z axes. IMUs are a common control and guidance device for autonomous vehicles. They're commonly used in Inertial Navigation Systems (INS), which calculate linear velocity, attitude, and angular positions in relation to a global reference frame using raw IMU data. (Borenstein, Everett and Feng, 1996)(Ilci and Toth, 2020) .

c. Encoders: Encoders are electromechanical devices that convert a shaft's linear or angular position into an analogue or digital signal, producing a linear or angular transducer. It can provide information on position, direction, and velocity within a control system (Campbell *et al.*, 2018).

1.2. Motivation

A combination of sensor data are brought together using sensor fusion algorithms and integrated onto the maps available in the vehicle or in the cloud to support navigation. An AV determines its real-time moving strategy based on its dynamic surroundings perceiving through its sensors. Once all perceptive sensors are incorporated, the longest perception range of all exteroceptive sensors, is 250 m (Vargas *et al.*, 2021) which enables an AV to see 4.5 – 7.5 s ahead at motorway speed of 120 km/h (Van Brummelen *et al.*, 2018)(Rasshofer and Gresser, 2005) (Mohammed *et al.*, 2020). A challenge for AV is hence the limited perception range, coupled with sensor occlusion, sensor noise, or scenarios with blind-spots, cornering and intersections (Leibe, Seemann and Schiele, 2005)(Bogdoll *et al.*, 2021). Some limitation for sensor systems (de Ponte Müller, 2017)(Vargas *et al.*, 2021) and in turn perception systems are as follows:

- Sensors are limited by line of sight and are unable to see through a large obstacle such as a truck ahead or around corners either due to the road geometry, being an intersection (non-signalised intersections in particular) or because of the presence of obstacles such as a building. Such scenarios are referred to as NLOS scenarios in this thesis.
- In bad weather such as heavy snow or heavy rain, some sensors can perform poorly.
- Sensors may mistakenly identify some innocuous objects (such as plastic bags) as obstacles.
- Sensors are unable to discern human signs.

NLOS scenarios causing blind spot in the vehicle's perception (Elliott, Keen and Miao, 2019) as explained above is the primary focus of the current study, with easy transferability to other challenging scenarios touched

on above . The need to increase perception beyond the current visibility of driver or vehicle and address the safety issues at blind spots has always been a matter of concern (Elliott, Keen and Miao, 2019) (Leibe, Seemann and Schiele, 2005)(Saito *et al.*, 2021).

To provide some examples, consider the scenario as shown in Figure1.1 of two vehicles (Ego and Car as indicated in the Figure) approaching an intersection. Where the ego vehicle is a connected autonomous vehicle under test and a car is any other vehicle on the road (may either be a connected autonomous vehicle or a non-autonomous vehicle with communication capabilities or a simple non-autonomous vehicle). Both vehicles are unable to perceive the other due to a bend and carry on with their current set trajectory as indicated as P_1, P_2 and P_3. A circumstance arises when the ego vehicle is at P_3 and the car is at P_2 at the same instance in time. The vehicles perceive the oncoming dynamic object and attempts to perform a manoeuvre change (breaking, speed change, trajectory change). However, due to the time at which the object was detected and the time required for the system to react, this would have fatal consequences (Greenemeier, 2016) (Allidina, Deka and Paluszczyszyn, 2018)(Gelbal *et al.*, 2019). If the vehicle was able to detect the oncoming dynamic object at either P_1 or P_2 the time to react would be sufficient.

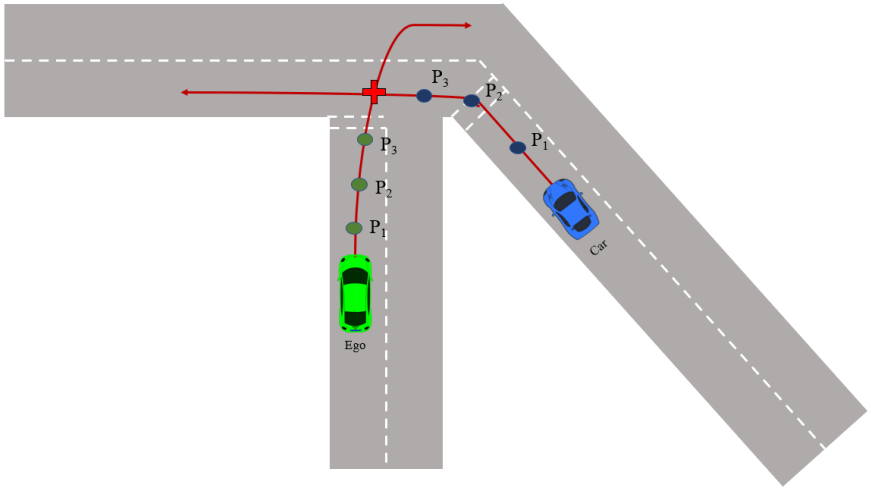


Figure 1.1: NLOS scenario

Autonomous vehicles are designed to improve safety and require operating safely in all environments and regardless of the condition, however in certain scenarios as just depicted, on-board sensors are not enough to avoid NLOS collisions.

There are several steps taken by industry and academia to develop methods/algorithms to improve blind spot detection for example the work of Hussein et al that looks into vehicle-to-everything (V2X) communication(Hussein, Erol-Kantarci and Sorour, 2020) . Garanderie et al. aimed to eliminate blind spot by using 360° panoramic imagery (Payen de La Garanderie, Atapour Abarghouei and Breckon, 2018) . To address

blind spots Kwon et al. developed a camera-based blind spot assistance system (Kwon, Park, *et al.*, 2018)(Kwon *et al.*, 2018). Several patents, such as (Urmson, 2013) (Chu, 2017) explore improving sensor placement or rate which sensors can detect dynamic objects.

Yoshihira et al. (Yoshihira *et al.*, 2016) talk about the challenges a driver faces in making a right turn due to poor visibility. They provide results showing that the poor visibility is due to the presence of a right turn and most right-turn accidents occur at signalized intersection. The leading cause of such accidents has been reported to be blind spots. The research (Naser *et al.*, 2018) addresses the issues of accidents occurring at intersections due to blind spots. Their solution includes equipping an autonomous vehicle with a drone such that as the vehicle approaches an area of blind spot, the drone takes off and surveys the surrounding. There are several other research looking at NLOS scenarios (Wen, Zhang and Hsu, 2018)(Noor-A-Rahim *et al.*, 2019)(Brambilla, Pardo and Nicoli, 2020)(Naser *et al.*, 2019)(Fokin, 2019)(Mohan, Ali and Joo Chong, 2020) . Furthermore, there is research looking into ultra-wideband systems for localization (Naser *et al.*, 2013) and ranges up to using Wi-Fi signals for NLOS perception. The research (Jain and Banvait, 2016) (Henion and Sinelli, 2010) explores into other ways of improving blind spot , by improving features of the vehicle, such as the location of the side view mirror.

Through the literature it is evident that a lot of work is currently attempting to address safety on roads and improving on the ability of autonomous vehicles to "see" beyond the line of sight of its sensors. For a vehicle to be able to see beyond its own perception horizon, it needs information received from sources other than its own sensors, such as those received from surrounding vehicles, pedestrian and infrastructure via the V2X communication (Minn, Zeng and Bhargava, 2015)(Anaya *et al.*, 2015)(Hobert *et al.*, 2016)(Hussein *et al.*, 2018)(Naranjo *et al.*, 2016)(Xin, Dan and Shuo, 2020). Information transmitted include, states and intentions of nearby vehicles, road conditions, weather information, traffic status, etc. (Lu *et al.*, 2014). The extended perception possible via V2X communication allows vehicles to predict the behaviour of dynamic objects and to anticipate hazardous events. This is an emerging area of research and some of the projects proposed have been funded through competitions such as, 'Meridian 2: Connected Vehicles Data Exchange' issued by the UK Government in 2018 , inviting UK business or research organization to apply for a share of up to £5 million, to create the world's best-connected vehicles data exchange or trading platform(UK GOV, 2018). This funding was awarded to Transport Systems Catapult and Cranfield University to develop their proposed project Connected On-Road Autonomous Mobility (CORAM). CORAM is researching on how to look ahead in NLOS situations, using low-latency, over-the-air messaging to autonomous vehicles. Another project in this area is the Autoplex project by Jaguar Land Rover (Jaguar Land Rover, 2018), whose aim is to combine connected, automated and live mapping technology to provide autonomous driving cars with information such as those alluded to in the preceding paragraphs, much earlier. Similar research and development is now being conducted by many research groups around the world (Anaya *et al.*, 2015; Naranjo *et al.*, 2016; Hussein *et al.*, 2018; Chen *et al.*, 2020; Xin, Dan and Shuo, 2020) , thus bringing many solutions onboard.

1.3. Problem Definition

The safety and validation of autonomous vehicle between level 3-5 would require a more comprehensive test that would account for complex and unpredictable environment. Autonomous vehicles with higher levels of autonomy use a larger number of algorithms which include but not limited to environment recognition using camera image (traffic lane/road recognition, semantic image segmentation and identification of objects, image-based environment mapping), environment recognition using other sensors such as radar, lidar etc (sensor data-based environment mapping), detection and tracking of moving objects, planning and decision making etc (Bugała, 2018). Through the verification and validation process the capabilities and limitations of these algorithms must be tested.

The different methods of testing primarily include on-road and simulation-based testing. Kalra et. al. (Kalra and Paddock, 2016) investigated the required miles that an autonomous vehicle must operate for the validation and verification process. Kalra and Paddock considered the fatality rate in the U.S.A and argued that 275 million miles are required to fully validate the functions of an autonomous vehicle, which is difficult to achieve even when using a fleet of autonomous vehicles. If a 100 autonomous vehicles driving for 24 hours a day, 7 days a week with a speed of 25 miles per hour, such a fleet would require driving for about 400 years to meet the verification and validation requirement. Even if this driving distance were achievable, a distance base validation does not guarantee all possible scenarios can be covered, as some scenarios may never appear during validation but would come up on actual driving. Hence, there is a strong need for more sophisticated verification and validation methods. Several researchers and developers have resorted to scenario based testing (Minnerup, 2017)(Xia *et al.*, 2017)(Distribu, 2013)(Maurer, Gerdes and Winner, 2016)(Menzel, Bagschik and Maurer, 2018b). Scenarios are a great tool to utilize and can represent a large number of critical situations which would otherwise be impractical to replicate on roads due to the danger, expense, and time-consuming nature. This robust testing methods becomes particularly important as connected autonomous vehicle detection and navigation is highly safety critical in nature, involving the safety of the vehicle occupants as well as other road users.

Despite there being works in scenario generation, to the best of our knowledge there are no scenario generation techniques that specifically generates NLOS scenarios with incorporated V2V communications capable for providing a common benchmark for testing the newly developed methods as cited above. Furthermore, as every NLOS location such as that depicted in Figure 1.1, can pose hundreds of possible scenarios (with different vehicle position, speed heading etc), effective testing needs to identify scenarios that are safety critical, while incorporating such scenarios into a practical and usable implementation platform. Hence, this research proposes to develop a framework for generating of NLOS scenarios for connected autonomous vehicle. For the purposes of this thesis, the focus is on CAV (Connected Autonomous Vehicle) who have the capability of communicating.

1.4. Research Aims and Objectives

The aim of the current research work is to develop a framework to generate NLOS critical test scenarios for CAV's . These test scenarios can be utilized by researchers and developers to test their algorithms, such as obstacle detection and avoidance, path and trajectory planning etc. Furthermore, the test scenario can also assist developers working on communication protocols to check performance parameters against. The objectives in order to achieve these aims has been as follows:

- To conduct and systematically evaluate the current state of the artwork for testing and validating connected autonomous vehicle systems.
- Design an ontology for developing Functional NLOS Scenarios by,
    - Develop and implementing an ontology suitable for NLOS scenario generation incorporating a communication layer in an ontology modelling software Protégé
    - Validating the ontology using hermit reasoner to evaluate logical consistency and test the correct generation of scene to validate the relations.
- To generate concrete scenarios from the developed ontology.
- Identify safety critical scenarios from generated test scenarios space through,
    - Determining a suitable metric to evaluate the safety criticality measure of a given scenario.
    - Utilize genetic algorithm with the above identified metric as the fitness function to search the scenario space and identify the critical scenarios.
- The proposed test generation approach is validated and verified through the development of a simulation environment and conducting a series of suitable experiments.

1.6. Research Contribution

The contributions include:

Contribution 1: The overarching contribution of this research is the development of a novel framework (Illustrated diagrammatically in Figure 7.18) for generating NLOS test scenarios which incorporates the communication network.

- To develop this framework current state-of-the-art methodologies and techniques have been investigated as discussed in Chapter 2, to in many ways inspire the proposed ontology. Additionally, this framework looks into developing the test scenarios by complying to current safety standards in the automotive domain.
- Menzel et. al. (Menzel, Bagschik and Maurer, 2018b) proposed three abstraction level (Functional, Logical and Concrete) for the development of scenarios along a V-model-based development process

in ISO 26262. The research presented in this thesis follows this methodology to develop the novel NLOS test scenarios for connected autonomous vehicle.

Contribution 2: The development and conceptualization of an ontology for functional NLOS scenarios. This ontology incorporates vehicular communication and generates test scenarios.

Contribution 3: The development of a new and effective methodology to search for critical NLOS scenarios from the generated scenario space. This has been achieved through these secondary contribution

- Defining what constitutes a critical scenario as used in this thesis and the development of a methodology to evaluate critical scenarios for NLOS cases which require communication in order to avoid collision. This has been done by combining several indicators, for example TTC and TST. This has been presented in detail in chapter 5
- Searching for critical scenarios within the scenario search space through the unique use of genetic algorithms.

Contribution 4: The development of a practical testbed to implement, execute and evaluate critical scenarios as identified by the evolutionary algorithms and utilising communication networks to intervene and revert the criticality of the situation by sharing vital information of obstacles between vehicles.

1.7. Thesis Layout

The thesis will be organized as follows:

- **Chapter 1: Introduction**
  Correspond to the current chapter
- **Chapter 2: Literature Review**
  This chapter investigates the methods for testing autonomous vehicle systems. A review is conducted of initial methods of testing and the current state-of-the-art methodologies. This chapter also looks at how ontologies have been utilized for generation of functional scenarios. Furthermore, the different practices of selecting critical scenarios have been looked at.
- **Chapter 3: Methodology**
  This chapter gives an overview of the development process to generate test scenarios. A discussion on how the development will adhere to current safety standards for automotive vehicle is presented. An overview is provided of functional, logical, and concrete scenario. Furthermore, this chapter will investigate prerequisites for the development of scenarios.
- **Chapter 4: Ontology**
  This chapter discusses the six-layer ontology to develop NLOS scenarios. The layers are as follows: Environment, Road User, Object Types, Communication Network, Scene and Scenario. A detailed

description for each layer and its Concepts *(C)*, Attributes *(A)* and Relations *(R)* are given within this chapter.

- **Chapter 5: Critical Scenarios**

  This chapter defines safety critical scenarios within a NLOS scenario. An example of junction scenario is used to calculate critical scenarios. A safety metrics are used to determine safety critical scenario in a NLOS environment which would require communication in order to avoid a collision.

- **Chapter 6: Genetic Algorithm**

  This chapter discusses the evolutionary algorithm which is utilized to select safety critical scenario. This chapter discusses the process at which the GA is developed, what parameters are selected as well as an evaluation of the fitness discussion

- **Chapter 7: Experiment and Results**

  This chapter discusses the implementation of NLOS test scenario generation framework.

- **Chapter 8: Conclusion**

  This chapter discusses how the research aims and objectives are met. Highlights the contribution of the thesis. Discusses some of the limitation of the work as well as future work.

1.8. Publications

Allidina, T.; Deka, L.; Paluszczyszyn, D.; Elizondo, D. Selecting Non-Line of Sight Critical Scenarios for Connected Autonomous Vehicle Testing. *Software* **2022**, *1*, 244-264.

Allidina, T., Deka, L. and Paluszczyszyn, D. (2018) 'Developing Functional Test Scenarios for Around the Corner Navigation by Autonomous Vehicles', *ACM womEncourage* **2018**.

Chapter 2: Literature Review

This chapter reviews the literature on automotive software testing and concepts supporting it. The sections within this chapter are as follows:

Section 1: This section gives an overview of automotive software testing. This includes:

- The importance of software testing, in particular for autonomous vehicle.
- The standards for developing automotive vehicle
- Methods of testing complying to these standards
- A further discussion on scenario-based approach to test

Section 2: This section discusses ontologies for scenario generation. This includes:

- Definition of Ontologies
- A review of literature on scenario generation using ontologies
- A review on literature utilizing ontologies in vehicle communication

Section 3: Discusses the scenario selection methodology used within this research

2.1. Automotive Software Testing

As AV become more widely available, road safety, travel times, highway and intersection capacity, fuel efficiency, emissions per kilometre, travel options, mobility, accessibility, and sharing opportunities are all expected to improve (Milakis, Van Arem and Van Wee, 2017). The most significant anticipated benefit of increasing vehicle automation is improved safety, which will reduce the number of people killed or injured every day in traffic accidents. However, because of the technology's infancy, this introduces new safety concerns. AVs are increasingly entering more complex environments as robotics advances. The concept of safety is particularly challenged in these new environments. The autonomous and secure operation of vehicles is essential for increasingly complex applications in environments with human presence. Like any other system that has the potential to generate potentially dangerous events, the autonomous vehicle must be designed to ensure the safety of its occupants and other road users. A method of ensuring safety of an autonomous vehicle operation is through testing. Testing is an integral part of software development; it ensures a system is behaving safely, doing what it is intended to do and nothing that it was not developed to do. Testing is performed with help of test cases or test input generation that are derived from system requirement specification to spot errors.

These cases are generated with an aim to cover all possible scenarios the software may encounter (Orso and Rothermel, 2014). Software testing consumes a significant amount of development time, making it an extremely costly process as software systems grow and become more complex, software testing takes a large amount of development time making it an extremely expensive process. Similar to most software systems, it holds true for the automotive domain as well (Dadwal *et al.*, 2018), which is the primary area of this study.

Before going into the details, a few terminologies as used in the text are defined here. The definitions (Jorgensen, 2013) adhere to the standards set by The International Software Testing Qualification Board (ISTQB) and The Institute of Electronics and Electrical (IEEE) Computing Society.

Fault, is the effect resulting from a bug. An example could be the effect of mission or omission of a requirement specification such as the doors of a vehicle not automatically locking when the gears step up to gear status 1.

Failure, transpires when the code with the fault is executed. Hence in the example above, *failure* occurs when the doors fail to lock automatically as the car moves.

Incident, is the actual occurrence of a failure that is noted by the end user, whether it is the tester or customer.

Test, or testing a system is the procedure followed to check a system against requirement specification and hence test cases.

Test cases, articulates a particular behaviour of the system and is associated with certain inputs and corresponding outputs.

Autonomous Vehicles have become a very popular area of research with new software and algorithms constantly being developed. Vehicles currently have over 1 million lines of codes across more than 100 ECU's (Ayres, Deka and Paluszczyszyn, 2021) and any undetected errors in the code may result in critical consequences. Hence, one of the most pressing queries in autonomous vehicle research is: "*How to prove an autonomous vehicle is capable to drive in live traffic?"*. The accidents made by the non-comprehensively assessed Honda cars (Zubinskiy, 2013) have demonstrated the disastrous consequences of improper testing.

To determine on what method will be used to test the autonomous driving system (ADS), this section will review the following:

- Standards developed by industry and government for the development of automotive vehicle.
- A review on testing methods that comply with current standards as well as the state-of-the-art.

2.1.1. Standards for Test Autonomous Vehicle

Autonomous vehicle SAE level 3 and higher need to have the capability to make decision and must be validated for a much broader operational domain. These highly automated systems employ a large number of Artificial Intelligence (AI)-based algorithms. For the roll out of these vehicles they must comply with current regulations. The laws governing automated transportation differ from country to country, but the majority of them can be grouped together under the guidelines of two agreements (Takács *et al.*, 2018).

- **Vienna Convention:** The Vienna convention on Road Traffic, is an international treaty that aims to improve road safety by facilitating international road traffic and establishing common traffic rules. The convention was ratified by 74 countries (excluding the United States) in 1968. The original articles of the convention stated that every moving vehicle should have a driver (though the term "driver" became more flexible later), and that every driver should be able to control the vehicle at all times. Because these restrictions were clearly not applicable to L3+automated vehicles, multiple amendments were required. The most notable one took effect on December 13, 2016, in Germany (Takács *et al.*, 2018), allowing the transfer of driving tasks to the vehicle if the technology used is UN-compliant or can be manually overridden or turned off.
- **NHTSA:** The guideline “A Vision for Safety” released in 2017 (*NHTSA | National Highway Traffic Safety Administration*) encourages entities to define and document Operational Design Domains (ODD) for each automated driving system on vehicles. The ODD should include documentation of the processes and procedures for assessing, testing, and validating the given functionalities, as well as the specific conditions under which they are intended to function. The type of roadway, geographic area, speed range, and environmental conditions are some of information required for the definition of ODD for a given functionality. This is meant to increase development flexibility by allowing manufacturers to limit the complexity of challenges in a pre-defined ODD at the outset.

2.1.1.1. European New Car Assessment Programme

The European New Car Assessment Program (Euro NCAP) was established in 1997 with the cooperation of European Union countries, and it is the most widely used performance assessment system (van Ratingen, 2017). The primary goal of this voluntary vehicle safety rating system is to publish safety reports on new serial-production cars based on their performance in four crash scenarios. Despite providing a de facto certification, the Euro NCAP is not legally recognised. These certifications could be used in future legislative processes in general. The bodies responsible for introducing such legislation are the United Nations Economic

Commission for Europe (UNECE) in ECE/TRANS/WP.29/2016/2 and the National Highway Traffic Safety Administration (NHTSA) in ECE/TRANS/WP.29/2016/2.

In 2010, Euro NCAP introduced a new reward system, Euro NCAP Advanced, to supplement the original rating scheme with ADAS features. BSM (Blind Spot Monitoring), ISA (Lane Support and Speed Alert Systems), and AEB (Autonomous Emergency Braking) are some of the new safety technologies targeted in the new evaluation (AEB) (Schram, Williams and Ratingen, 2013) .

In the most recent NCAP roadmaps for 2020 and 2025, special emphasis is placed on the true contribution of passive (crash tests) and active (ADAS) safety measures to overall safety performance, as well as the associated human-machine interface (HMI) (Van Ratingen *et al.*, 2016). This includes actuation performance tests as well as modular functional assessment (in the case of AEB, perception of the environment and the intended functionality delivery are assessed separately). Because Euro NCAP remains a non-de jure certification of new cars, functional requirements are still defined by Original Equipment Manufacturers (OEMs), taking into account the performance of corresponding components from automotive suppliers.

2.1.1.2. ISO 26262 and SOTIF standards

The ISO 26262's "Road vehicles—Functional safety" international standard defines the functional safety of electrical and/or electronic (E/E) systems in production automobiles. Its main goal is to address potential hazards caused by E/E system failures by providing an automotive safety lifecycle that covers the entire development process and determining Automotive Safety Integrity Levels (ASIL) as risk classes.

Traditionally, automotive designers have built their overall safety strategy around the idea that safety is ultimately determined by the human driver. This has resulted in, among other things, a focus on functional safety in ISO 26262 (ISO 26262, 2018). Functional safety ensures that the system can adequately mitigate failure risk for identified hazards. The amount of mitigation required is determined by the severity of a potential loss event, operational exposure to hazards, and the system's human driver controllability when it fails. These elements are combined to create an Automotive Safety Integrity Level (ASIL) based on a risk table. The ASIL assigned to a function determines which technical and process mitigations must be implemented, as well as which design and analysis tasks must be completed.

The automotive industry has recently developed a safety standard for driver assistance functions that may fail to function properly even if there is no fault with the equipment. These concerns are addressed in the ISO/PAS 21448 "Safety of the Intended Functionality" (SOTIF) standard (SOTIF, 2019b).It focuses on mitigating risks from unexpected operating conditions (the intended function may not always work in these due to sensor and algorithm limitations) and requirements gaps (lack of complete description about what the intended function actually is). Therefore, ISO 21448 expands ISO 26262's scope to include ADAS functionality. Both explicitly

allow for further scope expansion. SOTIF is an important standard because it is used in conjunction with ISO 26262 to cover any safety measures that may have been overlooked. Traditional standards, such as ISO 26262, would need to be used as a baseline to guide the development process of an autonomous vehicle. This methodology within this research will adhere to the guidelines within these standards

2.1.2. Methods of Autonomous Vehicle Testing

There are several methods for assessing the SOTIF capabilities of AVs. Figure 2.1 depicts various testing methods. Each of the approaches can be used to evaluate AVs in general. This research will be using a scenario-based method, this has further been discussed in Section 1.3 and highlighted in red in Figure 2.1. Furthermore, there are two approaches for selecting concrete scenarios in a scenario-based approach and this research will focus on falsification-based method, further discussed in Chapter 3

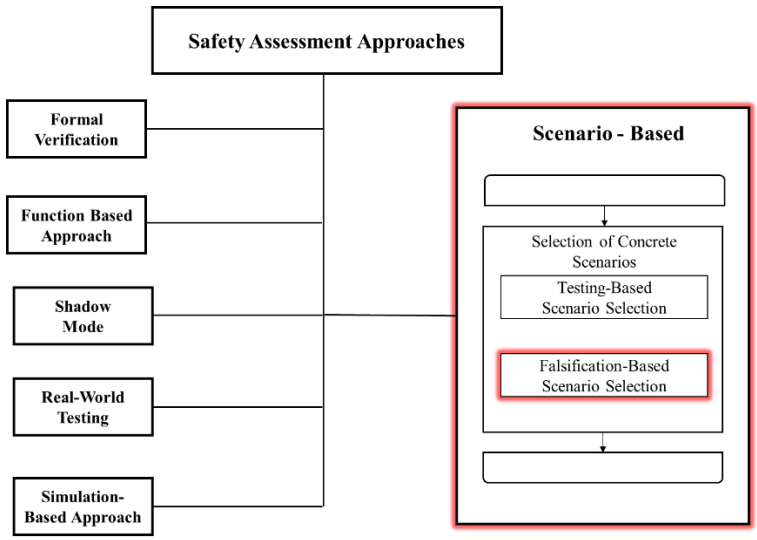


Figure 2.1: Overview of safety-assessment approaches, with a focus on scenario-based approach

The next section outlines the different safety assessment approaches that can be used to help test and validate ADS, with a focus on scenario-based methods. The different approaches are as follows:

A. Formal Verification

Formal verification, in the context of hardware and software systems, is the act of using formal methods of mathematics to prove or disprove the exactness of intended algorithms underlying a system with a certain formal specification or property (Riedmaier *et al.*, 2020). Formal verification ensures "a formal proof on an abstract mathematical model of the system, the correspondence between the mathematical model and the nature of the system being otherwise known by construction" (Wongpiromsarn and Murray, 2008). Formal verification is

used to validate systems such as combinatorial circuits, digital circuits with internal memory, and software expressed as source code. This approach has the potential to be exhaustive and reach full coverage with a validated mathematical model of the entire system, ensuring safety. Murray et al. (Ireland *et al.*, 2016) created a framework for formal verification of automated vehicles that relies on Monte Carlo simulation to validate models during the early stages of development. Althoff et al. (Althoff *et al.*, 2013) proposed a formal verification method that involves predicting all possible AV occupancy of space as well as other traffic participants. Shalev-Shwartz et al. (Shalev-Shwartz, Shammah and Shashua, 2017) created a formal model for determining who is at fault in an accident occurrence using AVs. The authors made the following assumptions in order to create a mathematical model:

- The sensor readings are accurate. Offline testing of the sensor framework's correctness is possible.
- A standard definition for determining who is to blame for an accident. If these assumptions are correct, demonstrating safety is reduced to a series of induction steps.

Formal verification methods, on the other hand, are currently not scalable due to the lack of a validated AV model. Other tools, such as required chips, operating systems, compilers, sensors, and so on, must also be verified in order to use formal verification.

B. Function-Based Approach

System functions are defined based on requirements in function-based testing, and then tested in simulation on the test tracker. This is a commonly used ADAS procedure. Current ISO standards (for example, ISO 15622 for Adaptive Cruise Control) and UN ECE regulations (for example, UN ECER131 for Advanced Emergency Braking Systems) follow a function-based approach and define a few fixed tests for individual systems, which confirm the latter's basic functionality. The system's functionalities must be defined before the function-based testing method can be used. This works for ADAS but is difficult for AVs because the required functionality of AVs cannot be defined in every possible situation (Riedmaier *et al.*, 2020).

C. Shadow Mode

Shadow mode or passive AV is another variation of real-world testing methods, in which the AV functionality is tested in an open-loop fashion (Koenig *et al.*, 2018). An AV receives all of the real-world sensor data in this type of testing, but its control output is not used to operate the vehicle. The vehicle is instead driven by a human. After retrieving the recording, a simulation environment is used to perform closed-loop replay (Wachenfeld and Winner, 2015). A variation on this is to run the simulation in the vehicle while only retrieving the critical data (Wang and Winner, 2019). A behaviour comparison between a human-driven vehicle and an

AV in the same scenario serves as a key feature for validation and a rubric for finding interesting scenarios in this online variation. This method has the advantage of posing no risk to other road users because a human-driver is actively driving while the AV functionality remains passive

D. Real-World Testing

Real-world testing is the most realistic of the validation methods, but it necessitates more effort and comes at a higher cost due to the high level of realism required. Furthermore, testing beta prototypes on the road can endanger other road users as well as the prototype vehicle itself. Nonetheless, test drives with prototype vehicles are always the final step in the validation process, allowing the system's performance to be evaluated in real-world scenarios. Every self-driving car company is putting their prototype vehicles through extensive real-world testing, and Waymo is one of them (Waymo, 2020b). Aside from testing, another reason for deploying the prototype vehicle in the real world is to gather real-world driving data for later use in simulation. Real-world testing with statistics based on miles travelled: The main idea behind the miles-driven statistical approach is to compare the number of miles driven between accidents caused by human drivers and accidents caused by autonomous vehicles. We can have some measure of AV safety if they can drive more miles between two fatal accidents than a regular human driver. Barnard et al. (Barnard *et al.*, 2016) proposed a methodology for conducting real-world testing. They proposed a framework to improve the previous method of conducting real-world testing to meet the requirements of AVs. Previous ADAS real-world testing was primarily driver-focused, and the evaluation was based on comparing performance between a normal vehicle and an ADAS-enabled vehicle. This comparison is less useful for the ADS because it takes drivers out of the loop of the driving task. Additionally, several large-scale real-world testing reports were also in (Waymo,2020)(Ziegler *et al.*, 2014). However, the likelihood of a fatal accident for a human driver is relatively low, necessitating a massive amount of testing miles to prove that AVs are at least as safe as humans. Kalra and Paddock (Kalra and Paddock, 2016) stated that driving 11 billion miles is required to prove statistically with 95% confidence that an autonomous vehicle is 20% better than a human driver (based on the failure rate). They came to the conclusion that in the real world, the traditional "miles driven" approach is not economically viable. Another issue with this statistical approach is that it fails to account for how eventful the miles driven are. For example, an AV can complete the miles requirement by driving on a relatively simple road. To address this, companies in California are now required to publish "disengagement reports" as per California State law. Disengagements are situations in which human intervention is required during an automated operation due to an unsafe decision made by AV due to the criticality of the situation or a technological failure (Jawad, 2021).

E. Simulation-Based Approach

AVs can be tested through simulations with varying levels of fidelity instead of being tested in real-world settings. Dedicated software with a simple or complex mathematical representation of the subsystems can be used depending on the purpose. A simplified vehicle point mass model, for example, can be used to test a high-level trajectory planning algorithm that abstracts out the actual vehicle physics (Althoff and Dolan, 2014). In April 2020, Waymo announced that it had driven 20 million miles on public roads and 15 billion miles in its simulator (Waymo, 2020c). Tesla's Dojo supercomputer replays recorded data in order to retrain its full self-driving software stack (Tesla, 2020).The type of simulation varies. Scalability and cost-efficiency are two advantages of simulation testing. Simulation models, on the other hand, must be tested in the real world. Nonetheless, AV testing can be accelerated by combining simulation and real-world testing.

Real-world data re-simulation is one simulation technique takes data logged during real-world test drives and applies it directly to testing. The recorded data is replayed to provide the necessary sensor information to AVs to test their response in this data-driven approach. Because the human driver's actual response in the real world is available, AVs can be compared against those decisions. Autonomous driving can be trained using a similar approach. Bach et al.(Bach *et al.*, 2017) described the Reactive-replay method for automated system assessments based on the reuse of recorded driving data in closed-loop simulation and its use in the early development of cruise control systems. Dojo, Tesla's supercomputer, is used to train the neural network (Tesla, 2020). Waymo re-stimulates real-world data with their simulator CarCraft (Waymo, 2020c). Waymo uses reactive agents, such as other cars, pedestrians, and cyclists, to respond with the new vehicle behaviour when the AV's position and behaviour differ from the logged scene thanks to software improvements. Modifying recorded data is a variation of data re-simulation. Validation of ADAS and computer vision-based ADS modules is done using augmented reality (Nilsson *et al.*, 2011)(Nilsson, 2014).

DeepTest generates test cases automatically based on real-world environmental changes such as rain, fog, lighting conditions, and so on (Tian, Jana and Ray, 2018). It generates test inputs that maximise the number of activated neurons to systematically explore different parts of the DNN logic. To test the AVs' trajectory planner, DeepRoad (Zhang, 2018) used a GAN to generate augmented data. Another method of simulation testing is to combine recorded spatial data with a traffic simulation model. Waymo augments collected data with fuzzy techniques and uses it in their simulator CarCraft (Waymo, 2020c).

F. Scenario-Based Approach

Scenario-based testing focuses on specific critical scenario cases to reduce testing time and effort by generating proof of correct functionality using a scenario coverage matrix rather than relying on the statistical

distribution used in real-world testing. In the PEGASUS project, scenario-based testing was used to test ADAS systems. The ability to use only the data that is required is a significant benefit of scenario-based testing. Several proposed methods, including clustering, rule-based, and machine learning-based methods, are used to identify representative scenarios (Kruber, Wurst and Botsch, 2020)(Bolte *et al.*, 2019)(Lenard, Badea-romero and Danton, 2014). Other methods deal with categorising scenarios based on their rarity.

Testing scenarios were developed by Lesemann et al. based on accident statistics to represent the majority of accidents in which active safety functions could potentially mitigate the outcome (Lesemann *et al.*, 2020) . Tuncali et al. also concentrate on collisions. Their main goal is to determine the conditions at the intersections of safe and collision scenarios (Tuncali, Pavlic and Fainekos, 2016). Annpureddy et al. (Annpureddy, LiuGeorgios and Sankaranarayanan, 2011) developed a test case generation for systematic testing of hybrid system with a focus on collision based on the S-TaLiRo tool. This is a MATLAB toolbox.

Tuncali et al. (Tuncali *et al.*, 2018) present a framework for Simulation-based Adversarial Testing of Autonomous Vehicles (Sim-ATAV) to check closed-loop properties of autonomous driving systems with machine learning components. VishnuKumar et al. (Vishnukumar *et al.*, 2017) propose a methodology for testing and validating ADAS and autonomous vehicles that uses machine learning and deep neural networks.

Although there are several research looking at test case generation using scenario-based methods, none of these focus on incorporating scenarios that include blind spots, i.e., presence of obstacles that are not in direct sight of the onboard sensors. This research uses a scenario-based approach to generate NLOS safety-critical scenarios for connected autonomous vehicle. Scenario based approach is further discussed in section 1.3 and the methodology is further discussed in Chapter 3.

2.1.2. Scenario-Based Testing

Scenario-based testing, in which individual traffic scenarios are tested using virtual simulation, is a promising method (PEGASUS, 2016)(NITSCHE, 2018). It has a number of advantages. It can be significantly faster than other approaches because it does not require real-time execution and testing can be done in parallel. Another significant advantage of this method is the assurance of safety during testing. Critical accident scenarios can be simulated in a virtual environment without endangering real-world traffic participants. However, because the simulated world deviates from the real world at times, model-world mismatches can occur. The primary goal of reproducing failure scenarios after they have occurred is to reduce as many failures as possible ahead of time.

2.1.2.1. Definition

A. Scenario

According to (SOTIF, 2019a), based on the work of (Geyer *et al.*, 2014) and (Ulbrich *et al.*, 2015), a scenario is "the temporal development between several scenes in a sequence of scenes." As a result, scenarios describe the chronological sequence of still images represented by scenes, which can be supplemented with actions and events (e.g., driving manoeuvres)

B. Scene

A scene is a snapshot of the vehicle's environment, including both static and mobile elements, as well as their relationships. All geospatially fixed elements, such as highway infrastructure and weather conditions, are referred to as static elements. Elements that move or have the ability to move are known as dynamic elements. They include autonomous vehicles and other modes of transportation.

C. Test Case

A test case is a description of a specific driving environment in which the autonomous vehicle will operate. It consists of a scenario that describes a specific situation and assigns values to the properties of each scenario element. The values chosen are determined by the test case's goal.

Figure 2.2, illustrates the test case structure as adapted from Chen et al. (Chen, 2020)

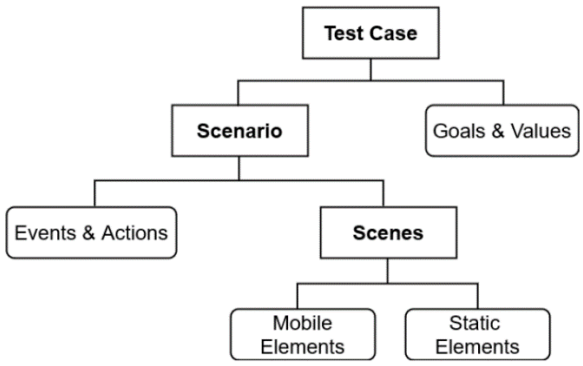


Figure 2.2: Structure for Test-Case(Chen, 2020)

2.1.2.2. Scenario-based Testing Methods

Scenarios can be derived from a number of different sources. Naturalistic driving data recorded in the real world is the most common type. Another source of information is abstract knowledge derived from guidelines, standards, and expert insight. As a result, we can divide the scenario sources as follows:

A. Knowledge-Driven Sources:

Guidelines for road construction, traffic infrastructure regulations, scenario catalogues, accident scenarios from police reports, and expert knowledge are sources of knowledge-driven scenarios. These resources provide useful information for describing scenarios and how to parametrize them. Ontology is a common method of representing knowledge (Bernus *et al.*, 2009). Ontology is a technique for organising and describing entities, their behaviour, relationships, and constraints in a readable formal way for both humans and machines. The primary goal of knowledge-driven sources is to use ontology to represent knowledge and derive scenarios from their relationships.

B. Data Driven Sources

Real-world scenarios can be collected by recording real-world measurement data during the test drive. For this purpose, a variety of sensors are used in tandem. One challenge of collecting data from the driver's perspective is that data is limited by sensor range. As a result, long-distance interaction between traffic participants is difficult to detect. Furthermore, the presence of the recording vehicle has an impact on the surrounding traffic. For this research a knowledge-driven method is used in the development of the test scenario. This is further discussed in Chapter 3

2.1.2.3. Scenario Development

Several researchers and practitioners have used scenario and test case generation. Some of these approaches include: (Lesemann *et al.*, 2020; Gregoriades, 2007; Hummel, Thiemann and Lulcheva, 2008; Hülsen, Zöllner and Weiss, 2011; Pollard, Morignot and Nashashibi, 2013; Armand *et al.*, 2014; Tuncali, Pavlic and Fainekos, 2016; Schuldt, 2017). Menzel et al. (Menzel, Bagschik and Maurer, 2018a) has categorized the scenarios into functional, logical, and concrete scenarios. This research will be using this categorization to develop scenarios and generate test cases. This has been further discussed in Chapter 3. Functional scenarios are the most abstract layer scenario development, this can be described in terms of language. A common approach to develop functional scenarios is the use of ontologies. This has been expanded further in Chapter 3.

A review on the current state of the art for ontologies has been outlined in Section 2 within this Chapter. A logical scenario is a state-space representation of a scenario space with parameter ranges. Each parameter can be linked to an influencing factor. For example, speed parameter is linked to the influencing factor speed limit. Probability distributions can be used to specify parameter ranges. This has been further discussed in Chapter 3. A parameterized representation of a specific scenario is called a concrete scenario. Each concrete scenario is an instance of a logical scenario, complete with a concrete value for each parameter. A review on methods of selecting a concrete scenario is discussed in section 3.

2.2. Ontologies

A conceptualization is a simplified and abstract representation of the world that we want to re-present (Gruber, 1993). It is the process of constructing and clarifying concepts through the use of words and examples in order to arrive at precise verbal definitions. A formally represented knowledge base contains objects, concepts, and other entities that are thought to exist in the area of interest, as well as the relationships that hold them together (R. Genesereth and Nilsson. J, 2012).

An ontology is a formal description of a concept. It is a structural framework for expressing knowledge about the world or a portion of it. It is made up primarily of concepts and their relationships, and it denotes a shared understanding of a topic. This understanding can help to reduce or eliminate conceptual and terminological confusion, which can aid in the resolution of problems that impede communication between people, organisations, and/or software systems. As a result, an ontology can serve as a unifying framework for various points of view and as a foundation for communication between people with diverse needs. It also enables interoperability between systems created by translating different modelling methods, language paradigms, and software tools, as well as the benefits for the engineering system. The engineering system benefits from reusability, reliability, and specification (Uschold, 1996).

The term ontology appears for the first time in a computer science-related discipline in a work on the foundations of data modelling (Mealy, 1967). After that, ontologies have been used in a wide range of IT applications. Ontology theories are used to solve database integration problems and provide a solid foundation for the selection of modelling concepts in the field of data and information modelling (Milton and Kazmierczak, 1AD; Opdahl, Henderson-Sellers and Barbier, 2001; Fettke and Loos, 2003; Shanks, Tansley and Weber, 2003). Ontology approaches are used in engineering to reduce disproportionate costs in software maintenance and to improve software reuse (Falbo, Guizzardi and Duarte, 2002)(Falbo *et al.*, 2002). To develop functional scenarios this research will use ontology as just described.

2.2.1. Ontology Applications

Ontologies have been widely applied in all kind of research areas, e.g., computing (Chen *et al.*, 2004), vehicle platooning (Maiti, Winter and Kulik, 2017), system performance (Benvenuti *et al.*, 2017), transportation (Fox and Engineering, 2019), and vehicle planning (Provine *et al.*, 2004)(Schlenoff *et al.*, 2003). An ontology has the following advantages:

- Sharing Knowledge: Ontologies is very useful for sharing knowledge between people and software agents (Van Dam, 2009)(Universiteit, 2001). For instance, ontology can be used for communication between different software agents for example different simulation tools (Ma *et al.*, 2019)
- Easy to Render: Ontologies can be directly translated into a class structure for an object-oriented software implementation (Van Dam, 2009).
- The ontology than postulates the relationships between the different classes and provides information on the attributes of a class and the plausible values.
- Transparency: Domain assumptions about terms and definitions are made explicit (McGuinness, 2000). Exemplary, terms have been defined in Chapter 4, this helps with preventing ambiguity when interacting with the terms used and helps understand the implementation.
- Conceptual Schema: An ontology can be used as a conceptual schema in database systems (Gruber, 1993). A conceptual schema allows "databases to interoperate without having to share data structures" (Gruber, 1993). A conceptual schema can also be used to directly design the structure of a database. The implementation of an ontology can be done in a variety of languages. The Web Ontology Language (OWL) and the Unified Modelling Language (UML) are two major technologies. We refer the interested reader to for a detailed comparison of OWL and UML (Atkinson and Kiko, 2005).

2.2.2. Ontology for Scenario Generation

Ontologies have been developed and proposed for various tasks such as ontology reengineering (Bringuente, Falbo and Guizzardi, 2011), ontology learning (Cimiano *et al.*, 2009), ontology evaluation (Gómez-Pérez, 2004), ontology evolution (Zablith *et al.*, 2015), ontology merging (Hitzler *et al.*, 2005)etc. In the field of AV ontologies have been used for navigation (Schlenoff *et al.*, 2003) , motion planning (Morignot and Nashashibi, 2013) (Provine *et al.*, 2004)and scenario generation (Wotawa and Li, 2018a) (Menzel, Bagschik and Maurer, 2018b). This research will focus using ontologies for test scenario generation and in particular for situations where oncoming obstacles are not in the line-of-sight of the autonomous vehicle. This section will look what the current works are and the state of the art of ontologies as used within the domain of

autonomous vehicles. Ontologies have been used by some researchers in the autonomous vehicle domain to conceptualise and characterise the driving environment.

Hülsen et al. (Hülsen, Zöllner and Weiss, 2011) uses a description logic to describe the scenes. The first work uses the terms Car, Crossing, Road Connection, and Sign at Crossing to describe road intersections in general. They use description logic to reason about car relationships and to describe and define the semantics of a traffic intersection situation in the ontology. The results are shown for a five-road intersection with eleven lanes and six cars approaching the intersection. This model is restricted to intersections, and both the infrastructure and the number of vehicles is fixed. To test such a system generated test cases are insufficient.

An ontology of recognition for driving assistance systems is presented in (Armand *et al.*, 2014). The authors define ontology as a collection of concepts and their manifestations. Contextual concepts and context parameters are included in this ontology. On global road contexts, it can process human-like reasoning. Pollard et al. (Pollard, Morignot and Nashashibi, 2013) propose another ontology for situation assessment for automated ground vehicles. It includes the status of sensors and actuators, as well as environmental and driver conditions. However, because both ontologies' concepts have not been sufficiently subdivided, they are insufficient to describe test cases for simulating and validating ADSs.

To better understand road infrastructure at intersections, Hummel et al. (Hummel, Thiemann and Lulcheva, 2008) propose an ontology. At several levels, the approach focuses on geometric details related to topological information. All of the ontology's concepts are introduced and organised in a hierarchical structure known as taxonomy. This method proposes scene comprehension frameworks based on description logic, which can detect erroneous sensor data by checking consistency. However, building testcases for testing the functions of ADSs requires more than just road infrastructure at intersections.

Zhao et al. (Zhao *et al.*, 2015) propose three ontologies to build a knowledge base for smart vehicles and implement various types of driving assistance systems: map ontology, control ontology, and car ontology. They concentrate on algorithms for autonomous vehicle decision-making. They offer a knowledge base and decision-making system based on ontologies that can make safe decisions about uncontrolled intersections and narrow roads.

In order to assist drivers, Morignot et al.(Morignot and Nashashibi, 2013) propose an ontology to relax traffic regulations in unusual but practical situations. The following are some examples of unusual but practical situations to consider: “a truck stopping and unloading before you and your car’s lane is delimited by a continuous line and a side-walk. After having waited for some amount of time, you might decide to cross the continuous line". The vehicles, infrastructure, and traffic regulation for the general road are represented by their ontology. It is based on the experience of lab members who have a driver's licence, rather than a corpus of texts.

Bagschik et al.(Bagschik and Maurer, 2018) propose that ontology be used to create scenarios for the development of automated driving functions. They propose a method for creating an ontology-based scene and

a knowledge representation model with five layers: road, traffic infrastructure, temporary road and traffic infrastructure manipulation, objects, and environment. From the first to the fifth layer, a scene is created. From the first to the fifth layer, a scene is created. With 284 classes, 762 logical axioms, and 75 semantic web rules, this ontology has modelled German highways. A number of scenes in natural language could be generated automatically. Natural language, on the other hand, is not machine-understandable knowledge, and converting natural language-based scenes to simulation data formats with such a large ontology is a huge undertaking.

Wotawa et al. (Wotawa and Li, 2018b) used ontologies to input in a combinatorial testing algorithm in order to generate scenarios. The Monte Carlo sampling technique is used in (Gregoriades, 2007) to sample the most likely events that could occur from the ontology of accident scenarios. Because autonomous driving and traditional driving cars are so different, the types of accidents they cause can be quite different. We must not only investigate existing collisions, but also prevent collisions that have never happened before or were unexpected.

Functional scenarios are described in a linguistic manner as discussed in section 1.3. Ontologies can be used to develop functional scenario as discussed in (Menzel, Bagschik and Maurer, 2018a). Furthermore, through the literature it can be seen that ontologies can also be used for scenario generation (Klueck *et al.*, 2018)(Bagschik and Maurer, 2019). Therefore, this research will be using ontologies to develop the functional scenarios.

2.2.3. Vehicle Communication and Ontologies

Ontologies have been used in the context of information exchange between vehicles or V2V communication to formalise concepts and properties. Lee et al. (Lee and Meier, 2007) proposed an ontology-based spatial context model for heterogeneous intelligent transport system (ITS). Their research combined methods for modelling contextual information used by ubiquitous transportation services. The model enables interoperability between autonomous Intelligent Transportation Systems, with the primary context pervasive transportation services can reason about shared context information and react accordingly, thanks to ontology. The distributed, independently defined data is correlated based on its primary context: location, time, identity, and service quality. This is tested in a car park system in order to provide a smart parking space service.

Eigner et al. (Eigner and Lutz, 2008) demonstrated the need for ontological context models for VNs safety environments in the domain of Intelligent Transportation System (ITS) Safety, as well as how the components would be able to understand one another through these models. They thought the vehicles should have a variety of sensors to collect data from both themselves and their surroundings. Additionally, using a vehicle network, the data collected by these sensors could be shared with other vehicles. They demonstrated how ontological model characteristics such as partial validation, information richness and quality, and a certain level of formality could benefit vehicular applications. They demonstrated that the model's application is fast enough to meet real-

time requirements imposed by the vehicle's active safety system through calculations. They did not, however, create a specific ontology model.

Kannan et al. (Kannan, Thangavelu and Kalivaradhan, 2010) proposed an ontology modelling approach for assisting vehicle drivers through warning messages during time critical situations. They concentrated on sending out alert messages based on context-aware parameters like driving situations, vehicle dynamics, driver activity, and the surrounding environment.

A Car Accident Ontology for VANETs (CAOVA) was proposed in (Barrachina *et al.*, 2012a), and it was further discussed in (Barrachina *et al.*, 2012b) as a Vehicular Accident Ontology designed to improve road safety (VEACON), which focused on road safety applications. This ontology unifies the information available in the General Estimates System (GES) accident database, as well as the data gathered during an accident. CAOVA (Barrachina *et al.*, 2012a) and VEACON (Barrachina *et al.*, 2012b) are ontology-based solutions for improving traffic safety and enabling interoperability among vehicles, roadside units (RSUs), road management specialists, and crisis support vehicles.

Distribu et al. (Distribu, 2013) created an ontology to represent traffic within highways. They wanted to build a reliable traffic information system that would provide data on roads, traffic, and scenarios involving vehicles on the road. It aids the traffic information system in determining the severity of a given situation. Here's an example: A toll plaza's congestion status may be important to an ambulance. If the ambulance is moving in the aftermath of an accident, obtaining this information is critical. This information request, on the other hand, is not so critical if an ordinary car is driving down the road with no time constraints.

For adding reasoning capabilities to autonomous vehicles, Morignot et al.(Morignot and Nashashibi, 2013) and Pollard et al. (Pollard, Morignot and Nashashibi, 2013) propose an ontology-based approach. The primary application is for self-evaluation of the perception system in order to monitor co-driving. The assessment module formalises knowledge such as environmental conditions, moving obstacles, driver state, and navigable space, all of which are important concepts in the VANET.

In order to model security in terms of identifying intrusion based on semantics, ontology was used by Erritali et al. (Erritali *et al.*, 2013). They demonstrated various ontology-based intrusion detection techniques and provided an overview of ontology web languages. This research used ontology to solve the problem of intrusion detection in vehicular ad hoc networks. Other aspects of ontology VANET and their environment were not considered in this study.

A method to improve situation awareness while transporting patients in an emergency was proposed by Groza et al. (Groza , A, Marginean, 2014). The method combines semantic reasoning with V2X technology (vehicle to everything). The developed system matches data from inter-vehicular communication with structured knowledge from vehicular ontologies and OpenStreetMap on a continuous basis.

FuzzOnto is a novel cognitive network framework for heterogeneous wireless networks presented by Al-Saadi et al. (Al-Saadi, Setchi and Hicks, 2017). The framework makes use of a semantic knowledge base, which includes ontologies and a semantic rule base for defining Quality of Service (QoS) parameters and network characteristics.

Bibi et al. (Bibi, Rehman and Ahmed, 2018) described a method for disseminating messages in VANETs. This was created to deal with the problem of broadcast storm in VANETs. According to their method, the ontology encodes and transfers information about the vehicle and its surroundings.

In situations where messages are forwarded through V2V, Santos et al. (Santos J and Moreira, 2019) developed an ontological representation of a reputation system in VANET to calculate the reputation score of a vehicle. This would allow a receiving vehicle to accept or reject a message based on the transmitting vehicle's reputation. The main goal is to improve the security of messages sent over the network for V2X applications. They assessed the performance in a city setting. A reputation model for VANTET was presented by (Santos J and Moreira, 2019). Two of the six functions in the model have been computed using ontology.

Aina et. al. (2020) developed a protocol for information exchange in a VANET system (Aina and Oyelade, 2020). Their knowledge-based system was implemented using a set of VANET-specific rules. Their proposed work allows for information sharing among nodes via retrieval.

Despite there being work to implement vehicle communication within ontologies, there is no work to the best of our knowledge that looks into generating NLOS scenarios that includes vehicular communication which is the focus of this research.

2.3. Scenario Selection

Once scenarios have been defined and parameterised by the functional and logical scenarios there are large number of scenarios that will be available. However, not all scenarios will be relevant to test the ADS and testing against all scenarios would be too time consuming. Therefore, particular scenarios adhering to certain criteria will need to be selected. In the process of scenario development, scenario selection comes under the third phase which is concrete scenarios. There are two methods of scenario selection, namely: ‘Testing-Based Methods’ and ‘Falsification-Based Methods’. Testing-Based Methods, samples a subset of concrete scenarios to test against. Falsification-Based Methods finds counter examples that violate the safety of the vehicle under test, these have been further discussed in chapter 3. This research will be using Falsification-based Method to select safety critical scenarios (Chapter 5). In this case only critical scenarios or scenarios that lead to or are highly likely to lead to accidents, will be selected for the autonomous vehicle to be tested against. The meaning of critical scenarios and related concepts are explained in more detail later. There are different ways of selecting such scenarios, and evolutionary algorithms is one such method (Kluck *et al.*, 2019).

2.3.1. Critical Scenario Definition

In the research literature, the concept of a critical scenario is defined in a variety of (different) ways. Related terms, such as edge case and corner case, are also used in some papers (and are sometimes used as direct synonyms). This section defines the term critical scenario as referenced within this research.

A. Corner case and Edge Case

Similar to critical scenarios, corner cases and edge cases are those scenarios which would lead to a collision (Chou *et al.*, 2018). Corner cases and edge cases are often used interchangeably because they are related terms. The most significant distinction between them is the probability of occurrence. Combinations of normal operational parameters and a rare or unusual condition are known as corner cases. According to Koopman et al. (Koopman, Kane, *et al.*, 2019), not all edge cases are also corner cases, and vice versa. Only edge cases are considered edge cases when they have a unique combination of conditions that are both uncommon and rare. Rare, according to Koopman et al. (Koopman, Kane, *et al.*, 2019) refers to situations or conditions that happen frequently enough in a fully deployed fleet to be a problem but is not documented during the system design or requirements process.

B. Critical scenario

Critical Scenarios are those scenarios which are relevant to the functional requirement and have a potential for harm i.e., those scenarios which will help validate the functional requirement as defined in chapter 1. The risk of harm is defined in ISO 26262 (*ISO 26262,2018*) based on the likelihood of the scenario and the magnitude of the harm that would result .We define critical scenarios in this study as any scenario that could cause harm.

2.3.2. Scenario Space Exploration

A parameter space exploration method is used to generate a set of concrete scenarios from a logical scenario. Critical scenarios are identified among these generated concrete scenarios using a pre-defined criticality assessment method. The criticality assessment mapping function is a function that maps a point in the scenario space to a point in the scoring space. The scoring space is a tool for determining the criticality of specific scenarios. Because criticality is difficult to assess directly, the evaluation is done using surrogate measures for example, TTC is commonly used metric (Schwarz, 2014).If criticality is defined on multiple dimensions, each dimension in the scoring space refers to a criticality measure.. Therefore, a logical scenario

can be instantly transformed into a number of concrete scenarios. Assumed parameters, such as the length of road segment and the number of lanes, have fixed values for all instances (i.e., concrete scenarios). The scenario space to be explored is built around the parameters of interest. These parameters include those that are constant over time for example, the weather or the position of a stationary obstacle on the road. As well as those that are variable over time for example, the speed of a surrounding vehicle or position of surrounding dynamic objects. A parameter trajectory can be used to represent a parameter that changes over time. The parameters' values can be categorical for example, weather, colour, or numerical. Numerical values can be either continuous for example, speed, heading etc. or discrete for example, the number of other vehicles, the number of road segment and speed limit. The following factors must be considered when exploring the scenario space and determining the critical scenarios:

- The method used to explore the scenario space
- Method in which the criticality of a scenario is assessed
- Mechanisms used to improve the coverage of the exploration.

2.3.3. Exploration Methods

Exploration methods are methods used to search the scenario space. These can be divided into two types:

A. Naïve Search: Randomly searching the scenario space is a naive approach to scenario exploration. The samples are independent of one another. As a result, these methods can be used in tandem to speed up the exploration process. However, if critical scenarios are uncommon, these methods may be ineffective because sampling a critical scenario is rare. These include sampling and combinatorial testing
B. Guided Search: A guided search method is more efficient as each iteration is adjusted based on the previous iteration's search results, allowing the exploration to be converge to critical regions. These include learning-based technique and optimization

A. Sampling

In this method a concrete scenario is generated assigning a value to each parameter in a logical scenario space at random. The sampling size is determined by the required coverage and simulation computation time, and a predetermined number of samples are taken statistically based on parameter probability distributions. When the sampling size is small, near-random sampling, such as Latin Hypercube sampling (Batsch *et al.*, 2019), can improve coverage. It divides the multi-dimensional parameter space into even grids and selects a specific number of samples from each grid. Before sampling, studies (Bithar and Karumanchi, 2019)(Masuda, Nakamura and Kajitani, 2018) looked into the parameter distributions. Each dimension in the sampled scenarios

is viewed as independent and uniformly distributed in these studies. The covariance between different parameters isn't taken into account, so their values can be determined independently. However, some other studies (Akagi *et al.*, 2019) (Zhou and Del Re, 2018) have looked into correlations between variables. More sampling restrictions are imposed when considering the relationship between variables. Instead of Monte Carlo sampling, the Markov Chain Monte Carlo (MCMC) method (Akagi *et al.*, 2019) can be used effectively when parameter distribution and covariance are known. The proposed method in (Akagi *et al.*, 2019) uses a risk index derived from naturalistic driving data to efficiently select risky traffic conditions.

An exhaustive search can be used to ensure full coverage if the scenario space is fully discrete and reasonably small. If you have access to a scenario database, you can conduct an exhaustive search by checking every test case derived from the data (Wang, Peng and Zhao, 2019). Otherwise, a full-scale grid search can be used to examine every existing test case in the scenario space (Zhou and Del Re, 2018) (Reiterer *et al.*, 2019)(Stumper and Dietmayer, 2018).

B. Combinatorial Testing

Combinatorial Testing (CT) is a widely used software testing method that focuses on identifying failures caused solely by specific combinations of inputs(Kuhn *et al.*, 2015) . The CT's main goal is to produce a small number of test cases (a covering array) that satisfy N-wise coverage (where N is the number of variables influencing a test case). CT can be used to find unknown combinations of influential factors that could cause the ADS or a specific AD function to fail. For example, when a cyclist crosses a road in front of the ego vehicle in low visibility (evening or foggy). The criticality is influenced significantly by a combination of scenario parameters, such as the initial speed of the vehicle, visibility, and the distance between the cyclist and the vehicle. The test data is specified by the covering array, which is a matrix that stores the testing configurations. Each array row can be thought of as a set of parameter values for a single test (Nie and Leung, 2011). For example, if three-dimensional coverage is required, the generated covering array should cover all possible combinations of values from any three parameters, where three is the number of parameters in combination. A good covering array can significantly reduce computational costs and increase test efficiency. On the other hand, finding the smallest covering array is an NP-hard problem.(Kuhn *et al.*, 2015). A test database contains all of the test scenarios in the preceding array, and it can cover all possible parameter values to a predetermined degree. Only discrete parameter space can be used to define N-wise coverage. In a real-life scenario, however, both continuous and discrete variables are present in the logical scenario's parameter space. Discretisation, which is a process of quantizing continuous attributes by converting continuous data into a finite set of intervals and assigning some specific data values to each interval (Liu *et al.*, 2002), is a common method for dealing with continuous parameters. Tuncali et al. (Tuncali *et al.*, 2018) used a two-step approach to get around the problem. In the first step, CT is applied to all of the discrete parameters. In the second step, they optimised each

combination of discrete parameters by simulating annealing on all of the continuous parameters. To identify critical scenarios, the optimization process is used as a falsification process. Covering array generation methods can be customised instead of focusing on the smallest number of test cases in the covering array. For example, in (F. Gao *et al.*, 2019)(Xia *et al.*, 2017)(Xia *et al.*, 2018)(Duan, Gao and He, 2020), the generated covering array not only meets the N-wise coverage requirements, but also maximises the overall complexity of all scenarios.

C. Learning-based Methods

Methods for learning-based testing include combining a model checking algorithm with an efficient model inference algorithm and integrating the two with the system of interest (SoI) in an iterative loop. (Meinke, Niu and Sindhu, 2011). During optimization, it trains a surrogate model to learn the properties of the SoI. In either the scenario space or the scoring space, the diversity can be maximised by maximising the distances between samples. To improve sampling coverage, Mullins et al. (Mullins *et al.*, 2018) used a surrogate model for training. It takes a set of samples as input and outputs the estimated diversity (i.e., the mean distance) on the input samples' scoring space. Learning-based methods were used by Nabhan et al. (Nabhan *et al.*, 2020) to try to maximise the distance between scenarios. During scenario generation, non-safety-critical qualities (such as deceleration and jerk effects indicating passenger comfort) are also taken into account.

D. Optimization

An optimization-based approach uses the design variables, the constraints, the objective functions, and the optimizer (i.e., the solver) to search the scenario space for critical scenario (Kochenderfer and Wheeler, 2019). The objective functions in the vast majority of studies are quantified criticality measures, this has been further discussed in detail in Chapter 5. The problem formulation, particularly the transparency and complexity of the underlying models, has a significant impact on the optimizer chosen. When estimating criticality measures, a simulator is frequently used. Because of the high model complexity and lack of availability of plant interior structure models, simulation, and system analysis of the system of interest (SoI) can be computationally expensive in many cases. As a result, the SoI's behaviour is treated as a black box. As a result, the corresponding search procedure can be considered a black-box optimization problem.

The relationship between the input (i.e., scenario parameters) and output (i.e. simulation results) for a black-box optimization problem can only be analysed by external observation through simulations (Li *et al.*, 2015). Various heuristic methods, such as genetic algorithm (Kluck *et al.*, 2019)(Abdessalem *et al.*, 2018)(Cutrone *et al.*, 2019), Bayesian optimization (Gladisch *et al.*, 2019)(Gangopadhyay *et al.*, 2019) and simulated annealing (Klück *et al.*, 2019), can be used to approximate the fitting function and find the global minimum iteratively to

solve this problem. Furthermore, domain-specific heuristic techniques such as rule-based searching (Masuda, Nakamura and Kajitani, 2018), heuristic simulation-based gradient descent (Huang, Lam and Zhao, 2017), and zoom-in sampler (Beglerovic, Stolz and Horn, 2018) help to identify multiple local minimums of feasible solutions. (Feng *et al.*, 2020)(Feng *et al.*, 2021)(Feng *et al.*, 2020) formulate scenario searching as a two-step optimization problem. The first step in the optimization process is to find as many local optimal solutions as possible. The neighbourhood of the local optimal solutions is searched in the second step to find all the scenarios whose criticalities are within a certain threshold.

It is a well-known approach to use genetic or evolutionary algorithms for test generation and optimization. Stahmer (Sthamer, 1995) demonstrated in his Ph.D. thesis in 1995 that using genetic algorithms to generate test data is an order of magnitude more efficient than using random selection. Abdessalem et al. (Abdessalem *et al.*, 2018) were also inspired by the idea of using evolutionary algorithms. They demonstrated in their paper that evolutionary algorithms are well-suited for testing vision-based control systems and characterising critical input space regions. In addition, (Bühler and Wegener, 2004) proposed a genetic algorithm-based approach for testing an autonomous parking system. (Buehler and Wegener, 2005) used a similar technique to perform evolutionary functional testing of a vehicle brake assistant system, concluding that system errors discovered by evolutionary function testing are fairly unlikely to be discovered using traditional testing techniques.

Acknowledging the power of evolutionary algorithms to search through solutions spaces to identify optimum solutions based on certain criteria and associated constraints, in this research an optimization based (using GA) approach is used to select NLOS safety critical scenarios which require communication in order to avoid a collision. Unlike previous works, this is the first piece of work that uses evolutionary algorithms to select critical scenarios for testing the functionalities for connected autonomous vehicle. The use of Genetic Algorithm for scenario selection has been further discussed in chapter 7.

2.4. Summary

This chapter review the various methodologies for testing autonomous vehicle. The current state of the art is scenario-based testing. The scenario-based methodology goes from functional, logical to concrete scenarios. There has been much research into using this methodology to test automated driving functionalities. However, there is no research that looks at generating NLOS scenarios for connected autonomous vehicle. Furthermore, the functional scenarios will be developed using ontologies. Through the review it can be seen there is lots of works that looking into using ontologies for scenario generation, however, none of these works look into incorporating the communication network. Once scenarios are generated not all generated scenarios would be relevant to test against. To search the scenario, this research proposes innovative way of using the GA to select critical scenarios for testing the functionalities for connected autonomous vehicle.

Chapter 3: Methodology

The goal of this study is to generate NLOS safety critical scenarios to aid in the testing and validation of autonomous vehicle driving capabilities. To develop this, a systematic derivation of scenarios must be documented throughout the development process to ensure traceability (Menzel, Bagschik and Maurer, 2018a). To accomplish this, the chapter will discuss previous study methodologies as well as current safety standards, along with the methodology that will be used in this research. This chapter cover the following:

Section 1: This section covers the current safety standard for automotive vehicle and how scenarios can be developed adhering to these standards.

Section 2: The scenario requirements have been outlined in this section

Section 3: The development of the most abstract level of scenario representation is covered in this section, which is the first step in scenario generation. This section will go over the process of creating scenarios.

Section 4: This section discusses logical scenarios which depict a detailed representation of functional scenarios with the state space variable.

Section 5: This section will be discussing how scenario will be selected using distinct parameters in the state space.

3.1. Safety Standards for Automotive Vehicle

A briefly mentioned in chapter 2, the ISO 26262 (ISO 26262, 2018) and Safety of the Intended Functionality (SOTIF)(SOTIF, 2019a) are two frameworks for automotive safety development. The ISO 26262(ISO 26262, 2018) is about functional safety, addressing the reduction of safety risks from known component failures. SOTIF (SOTIF, 2019a) is a safety standard for driver assurance functions that may fail to function as intended even when there are no equipment failures. Figure 3.1, shows some terms within the ISO26262 and the SOTIF, this has been adapted from (Chan, 2021).The primary goal of the standard is to reduce risks associated with unexpected operating conditions, such as when an intended function fails to operate due to an algorithm or sensor limitation. It also aims to fill requirements gaps, such as a poor description of the intended functionality (Koopman, Ferrell, *et al.*, 2019). SOTIF is an important standard because it is used in conjunction with ISO 26262 to cover any safety measures that may have been overlooked. As the industry innovates and moves vehicles from L0 to L5 in the scale of autonomous driving capabilities, there are numerous safety challenges that must be understood and solved before AVs are cleared for a broader rollout (by regulatory bodies and society). Traditional standards, such as ISO 26262, would need to be used as a baseline to guide the

development process of an autonomous vehicle. The V-cycle development process is a reference model for the product development cycle within the ISO 26262 (shown in Figure 3.2).

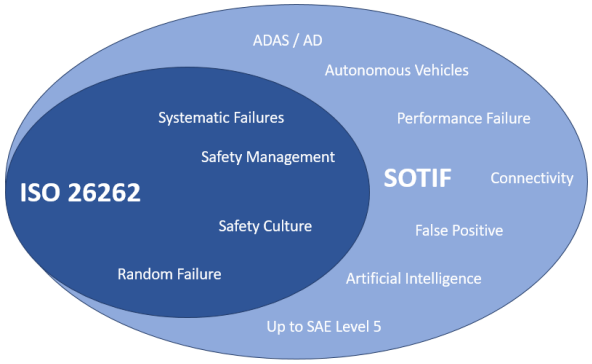


Figure 3.1: Depicts terms for the ISO 26262 and SOTIF (adapted from (Chan, 2021)*)*

T Menzel et al. (Menzel, Bagschik and Maurer, 2018b), introduced description for the development process of scenarios along the V-cycle , which goes from highly abstract in the concept phase (functional scenarios), to depiction of the scenario space with possible parameter ranges (logical scenarios) and scenarios with concrete parameter values in the test execution phase (concrete scenarios). Additionally, to ensure the safety of an autonomous vehicle, the vehicle's intended operational design domain (ODD) must be taken into consideration. This necessitates analysts and engineers thinking about the different operating environments (OEs) that can occur in the ODD, as well as the various scenarios that can occur within each OE (Wang *et al.*, 2021).

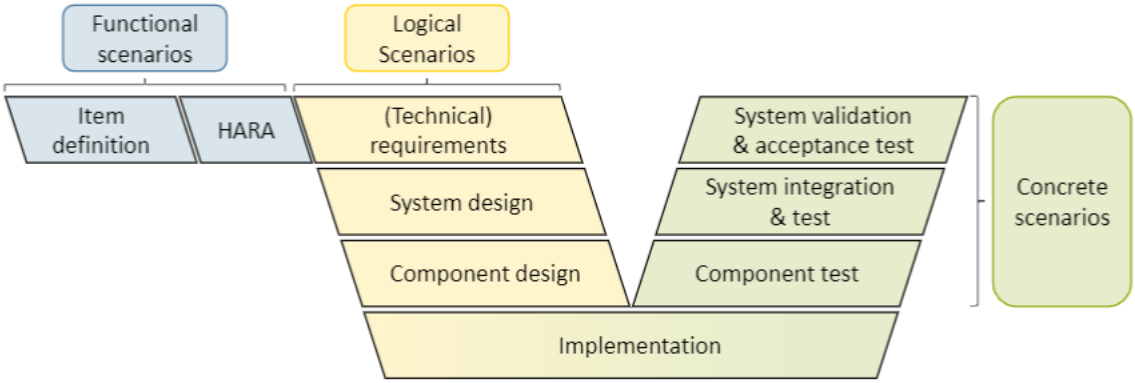


Figure 3.2: The development of scenarios along the V- model.(Menzel *et al.*, 2019)

Over the years, several organizations have proposed recommendations, voluntary guidance, and best practices as well as scenarios for the development and deployment of autonomous vehicles. This is summarised in Table 3.1.

Table 3.1: Summary of organizations

Name	Description	Reference
BSI	The British Standards Institution (BSI) is the national standards body of the United Kingdom.	(*Standards, Training, Testing, Assessment and Certification | BSI*)
NHTSA	The National Highway Traffic Safety Assessment (NHTSA). This document aims to aid industry as it moves forward with testing and deploying ADS and each level of autonomy to draft and establish plans for ADS according to the ISO26262 and its regulations.	(*NHTSA | National Highway Traffic Safety Administration*)
SAE International	The Society of Automotive Engineers (SAE International), is a United States-based, globally active professional association and standards developing organization for engineering professionals in various industries.	(SAE, 2016)
PEGASUS	The Project for the Establishment of Generally Accepted quality criteria, tools and methods as Scenarios and Situations for the release of highly automated driving functions (PEGASUS). Their goal is to create a procedure for testing automated driving functions in order to speed up the adoption of automated driving in the real world.	(*PEGASUS - pegasus-EN*)
Euro NCAP	The European New Car Assessment (Euro NCAP) have assessments for different type of scenario some e.g., include Blind Spot Monitoring (BSM), Lane Support and Speed Alert system (ISA) and Autonomous Emergency Breaking (AEB). The provide performance certification on their assessments.	(*Euro NCAP | The European New Car Assessment Programme*)
ASAM Open X standard	The International Standardization Organization for Automotive and Measurement System (ASAM OpenX standard) consist of OpenDrive, which defines a file format for the description of road	(Engel, 2020)

	networks (i.e., maps) and OpenScenario, which defines a file format for the description of the dynamic content in simulation (i.e., driving manoeuvres).	

Additionally there is a good quantity of research that look into testing autonomous driving system using scenario-based and simulation based method these include for free driving (Chen, Yuan and Tomizuka, 2019), following (Amersbach and Winner, 2019), lane change (K. Gao *et al.*, 2019)(Chen *et al.*, 2021)(S. Zhang *et al.*, 2018)(Zeng *et al.*, 2019), overtaking (Hegedus, Németh and Gáspár, 2019)(Huang, Zhang and Li, 2019)(Ortega, Lengyel and Szalay, 2020), cut-in (Zhao *et al.*, 2020)(Erdogan *et al.*, 2019)(Chen, Hu and Wang, 2019), leave lane (F. Gao *et al.*, 2019), cut-through (Erdogan *et al.*, 2019), slow-traffic (Best *et al.*, 2017), stop and go (Stern *et al.*, 2018), traffic-jam (Figueiredo *et al.*, 2009), incident traffic (Park *et al.*, 2021), obstacle detection (Geiger *et al.*, 2013)

Despite there being a lot of work in this area, to the best of our knowledge there currently exist no scenario generation method for testing NLOS which require some form of V2V communication (for vehicles to be informed of potential obstacles or oncoming traffic) in order to avoid collision. Such scenarios will help validate algorithms that aim at detection and avoidance of obstacles present or approaching, through knowledge received or acquired at an instant in time before the on-board sensors would detect, at when it would be too late to avoid a safety critical scene. Additionally, such scenarios could also help developers working to develop algorithms within the vehicular communication network test their work, say for example to test latency of communication protocols.

3.2. Scenario Specification

This section outlines the development of the functional, logical, and concrete scenarios. Discuses what type of scenarios will be developed as well as the software's used in the development process. Figure 3.3 shows an overview of the scenario development process as well as the software and tools used.

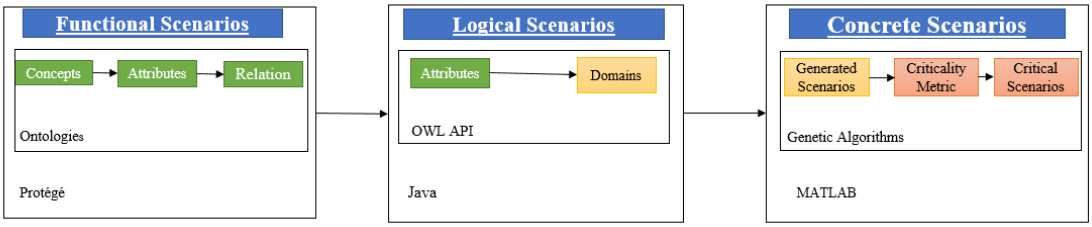


Figure 3.3: Scenario Development Process

3.2.1. Scenario Formation

To develop a scenario T Menzel et al. (Menzel, Bagschik and Maurer, 2018b) have discussed definition for the development process of scenario to test and validate autonomous driving functions. These definitions look at scenarios that are functional, logical, and concrete. Beginning with a verbal description of functional scenarios, the level of detail and machine readability increases, progressing to logical scenarios defined by parameter ranges and distribution, and finally to concrete scenarios defined by exact parameter values. Since the PEGASUS project developed and used these definitions (*PEGASUS - pegasus-EN*) and since been used in several studies (Colwell *et al.*, 2018; Klueck *et al.*, 2018; Queiroz, Berger and Czarnecki, 2019; Weber *et al.*, 2019) in the development process of scenarios . Furthermore, Automotive and Measurement System (ASAM) provide standards along the development process in order to describe scenarios uniformly and efficiently integrate them into different simulation environments (ASAM Standard). Therefore, since these definitions are known to a large part of the community, this research will adopt these definitions along the development process for the NLOS test scenarios. A further description of the methodology used to develop Functional Scenario (Section 3), Logical Scenario (Section 4) and Concrete Scenario (Sections 5) has been discussed in this chapter in the respective sections. Figure 3.4 presents an overview of the scenario development process used to develop the NLOS test scenarios.

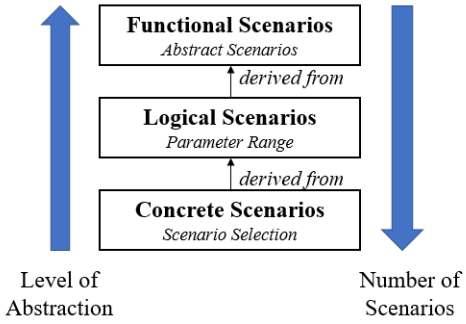


Figure 3.4: Scenario Development Process

3.2.2. Scenario Overview:

This section outlines the bird's eye view of the scenario and its content. Fahrenkrog et al. (Fahrenkrog *et al.*, 2014) have described two types of scenarios : Traffic and driving scenario.

- Traffic Scenario: A traffic scenario operates in a broad context to include a variety of vehicle within its scenario. This focuses the operation of autonomous vehicle in traffic condition
- Driving Scenario: The scenario simulation focuses on time and space-constrained driving scenarios that are important for safety. These driving scenarios frequently include a potentially dangerous

driving situation that can be handled by a highly automated driving function. By simulating driving scenarios in a repeatable manner, the safety performance will be determined and compared. The first step is to identify and specify relevant driving scenarios so that they can be sufficiently modelled.

The NLOS scenarios developed within this research will be driving scenarios. These scenarios will be modelled in a manner which require communication information to be sent to the vehicle under test and this data needs to be processed in a timely and efficient manner so that collision may be avoided. Thus, developing scenarios which are time and space-constrained and potentially dangerous.

3.2.3. Scenario Flexibility

A scenario would include the vehicle under test (Ego Vehicle) and other traffic participants for example other vehicles, pedestrians, cyclist etc. The actions and reactions of each of the traffic participants depend on the type of scenario that is developed. Rocklage et al. (Rocklage *et al.*, 2018) divided scenarios into static, dynamic and hybrid scenarios. The definitions for these are as follows:

- Static Scenario: Dynamic objects, such as other nearby vehicles and pedestrians, follow a predefined trajectory in static scenarios. The trajectories are defined in the scenario description, and traffic participants must adhere to them at all times, even if they collide with another entity. The simulation is unaffected by an AV's behaviour change as a result of a system update.
- Dynamic Scenario: This is the inverse of static scenarios. Dynamic objects consider the movements of other participants and can exhibit cooperative or non-cooperative behaviour. Their motion with respect to the AV is calculated online during simulation. Dynamic objects can adapt their trajectory due to the behavioural change of AVs with the system update
- Hybrid Scenarios: It combines static and dynamic scenarios. Dynamic objects follow a predefined path until the AVs influence their trajectory. The scenario is dynamic from that point forward.

The NLOS test scenario will be a static scenario, where the traffic participants all have a fixed trajectory and will be unaffected by the ego vehicle behavioural change. A static scenario where no other traffic participant reacts allow to determine the effectiveness of the algorithm under test as a conclusion can be drawn of the performance of the algorithm based on whether there is collision or not.

3.2.4. Assessment Granularity

This section will be discussing at what level the scenario will be assessing the system under test. (Junietz, 2019) have defined two types of assessments for AV. These are as follows:

- Macroscopic Assessment: These types look into the overall impact of the AV.
- Microscopic Assessment: Looks into the evaluation of a single scenario.

This thesis aims to select a critical scenario to assess the correctness of a particular sub-system of a CAV and thus will be using microscopic assessment to develop the scenarios.

3.2.5. Scenario Goal

The goal is to develop NLOS driving static critical scenarios for microscopic assessment for connected autonomous vehicle. Furthermore, the test scenarios should be developed from functional, logical to concrete scenarios. The focus is on SAE Level 4+, AV which are capable of communication.

3.2.6. Software and Tools

The software's used within this research include Protégé (Protégé ,Stanford University), to develop the ontologies for functional and logical scenarios. Protégé is a free open-source ontology software which comes with a built-in reasoner. A semantic reasoner is used to find logical inconsistencies, provide information to improve and validate the concept model. This research will be using the HermiT reasoner. HermiT is a highly efficient reasoner (HermiT Reasoner) and can validate the ontology (depending on the complexity) in seconds. To generate scenarios this research uses an OWL API. An OWL API is a java interface and implementation for OWL. It provides data structures representing OWL ontologies like their concepts, relations and attributes, this helps create, manipulate, parse, render and reason about those structures. The java file is converted into a JSON format and imported to MATLAB environment. MATLAB offers an Automated Driving Toolbox (Automated Driving ToolBox,MathWorks)to help design, simulate and test ADAS and automated driving systems. Other advantages of using MATLAB are its interoperability with other simulation environment for example Open Drive (MATLAB, Mathworks) .Figure 3.5 shows the process of the software and tools used.

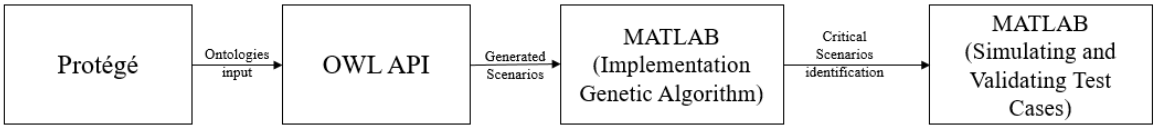


Figure 3.5: Scenario Implementation process

3.3. Functional Scenario

The most abstract level of scenario portrayal is termed as functional scenarios. Natural language is used to describe them so that human experts can easily understand, discuss, analyse and create new scenarios.

3.3.1. Definition:

Functional scenarios is an abstract method of describing knowledge of the concepts and relations within the scenario pertaining to the aims and objectives of the tests to be undertaken.

Menzel et. al. (Menzel, Bagschik and Maurer, 2018a) define a functional scenario as follows:

"Functional scenarios include operating scenarios on a semantic level. The entities of the domain and the relations of those entities are described via a linguistic scenario notation. The scenarios are consistent. The vocabulary used for the description of functional scenarios is specific for the use case and the domain and can feature different levels of detail."

On a semantic level, the representation of functional scenarios includes a linguistic and consistent description of entities and their relations/interactions. A consistent vocabulary must be defined for the linguistic description. This vocabulary includes terms for various concepts (vehicle A, vehicle B) as well as phrases describing their relationships (vehicle A communicates with vehicle B). The level of detail required for functional scenarios is determined by the current development phase and the item being developed. Hence, as the use case in the current research is NLOS road scenarios for connected autonomous vehicles, the concept described will be in the context of such scenarios. For example, the vocabulary used to describe this scenario would include different types of road junctions that lend itself to providing blind corners for approaching vehicles. Furthermore, vocabulary is used to also describe roadside infrastructure that obstructs direct line of sight views as obtained by on-board sensors, etc. When a comprehensive vocabulary is used to describe entities and their relationships, a large number of scenarios can be derived from the vocabulary. All terms in the vocabulary must be distinct in order to generate consistent functional scenarios. Actual standards and guidelines, such as road traffic regulations or the United Kingdom standard (*The Highway Code - Guidance - GOV.UK*) for constructing motorways, are examples of sources for terms that define the entities of a domain.

3.3.2. Scenario Development

There are two methods in which scenarios can be derived, a 'Data-Driven approach' and a 'Knowledge-Driven approach. In a data-driven approach scenario can be gathered from real world by taking measurements during test drives. This is accomplished through the use of combination of sensors (Park *et al.*, 2021)(Erdogan *et al.*, 2019). In a knowledge driven approach, the source of information is abstract knowledge derived from guidelines, standards, and expert insight which is a knowledge-driven approach. The use of ontology is a standard method for describing knowledge (Riedmaier *et al.*, 2020). As discussed in the scenario specification (section 2), this research will develop the functional scenarios using the "Knowledge Driven" approach and develop this using ontology. Ontologies are defined as "a formal, explicit specification of a shared conceptualisation" (Gruber, 1995). This is a method for formally organizing and describing entities, their behaviour, relationships, and constraints in a domain that is understandable by both humans and machines.

Ontologies are typically expressed in the Web Ontology Language (OWL), which is based on the Resource Description Framework (RDF) model, which conveys information in the form of subject-predicate-object, or triples. This is usually done with software like Protégé, which already has the Semantic Web Rule Language (SWRL) for expressing rules and reasoners like Pellet, Hermit, and others (Abburu, 2012) for inferring logical reasoning. Unified Modelling Language (UML) is a visual modelling language often used to describe and build software-based systems, business processes, and other things, is another way utilised by researchers for knowledge representation (Robles-Ramirez, Escamilla-Ambrosio and Tryfonas, 2017) (Mayr *et al.*, 2018)(Fredj *et al.*, 2021)(Li, Tao and Wotawa, 2020)(ASAM Standard, 2018).

Bagschik et al. (Bagschik and Maurer, 2018) work demonstrates the use of OWL. He represents knowledge and derives functional scenarios using OWL in a 5-layer architecture. Each layer in the layer model represents a distinct set of scenario pieces. The first layer specifies the road's layout, markings, and topology, while the second layer specifies traffic infrastructure implemented at the road level. Within the third layer, temporary manipulations of the prior two layers, such as building sites, are incorporated. The fourth layer contains fixed and dynamic items such as pedestrians and cars that are not part of traffic infrastructures, while the last layer is concerned with the environment, such as weather. This type of knowledge representation serves as the foundation for the automated generation of scenes and functional situations. The next step is parametrisation and development of logical situations after specifying functional scenarios.

Despite the fact that there has been research into developing functional scenarios, no work has included the creation of test scenarios specifically to NLOS autonomous vehicle driving context that include the communication network for testing scenarios. The importance of this has been highlighted in (Mohan, Ali and Joo Chong, 2020)(Brambilla, Pardo and Nicoli, 2020)(Wen, Zhang and Hsu, 2018)(Liu *et al.*, 2018) and is included in the NCAP road map for 2024 (Euro NCAP, 2021). However, to the best of our knowledge no work has been done to integrate the communication network layer into an ontology for the purpose of creating test scenarios. Ontologies will be used to create NLOS test scenarios for connected autonomous vehicles in this research

3.2.3. Developing Functional Scenarios

To develop the functional scenarios ontologies will be conceptualized using the UML standard (*Unified Modeling Language (UML)*). UML has a very large community and it allow understanding of information by experts and non-experts. As well as provides a graphical representation to model the ontologies. Additionally, UML has been used by previous research (Li, Tao and Wotawa, 2020) as well as ASAM standard (ASAM Standard) (*Design driving scenarios, MATLAB - MathWorks United Kingdom*,2021)to conceptualise their scenario. Once the ontologies have been conceptualised within UML, as a proof of concept and to validate the ontologies a simple ontology from the conceptualized ontology will be modelled in OWL using Protégé.

The ontology can then be validated by passing it through the in-built Hermit Reasoner to check for any logical inconsistencies as well as show the inference of knowledge based on the relationship among concepts. After that an OWL API will be used to generate scenarios. (This has been further discussed in Chapter 7).To ensure consistency in describing the terminology for the ontology, this research will be using the terminology by Wotawa et al. (Wotawa and Li, 2018b) . This piece of work defines terminology used within an ontology, for example concepts, attributes and relation. As well as rules to describe an ontology using this terminology, for example a concept cannot have an inheritance relation as well as composition relation with another concept. Using this method as a basis to develop the ontology allows for consistency as well as a methodology to check for correctness of the developed ontology. This terminology has been used by Wotawa et al. (Wotawa and Li, 2018a) to develop their ontology and use their ontology as inputs for combinatorial testing (CT) algorithm. This become advantageous as it allows the flexibility of the conceptualised ontology to be integrated and/or applied in various formats such as XML, Protégé, CT etc

3.2.4. Ontology Elements

In this section the rules and syntax to define ontology is outlined. The terminology is used is as follows:

- **Concepts:** A concept is defined as an entity either from the real world, example, road *(Road)* or from a nonmaterial description example, Communication Network *(Communication Network)*. Concepts form the building blocks of the ontology. They can be referred to as "Class" in OWL (OWL Web Ontology Language Reference). A concept is denoted with *'C'*.
- **Attribute:** A concept can have one or more attributes. Attributes are properties that characterize the concepts, for example a road *(Road)* has the attribute length. An attribute can be of a certain date type like a string or integer. They can be referred to as "Data Properties" in OWL (OWL Web Ontology Language Reference). An attribute is denoted with '*A'*.
- **Domain:** Each attribute is associated with a domain, which is set of valid values for that attribute. A domain is an attribute for a given concept: $C \times A \mapsto 2^D$ .For example, the concept road segment *(RoadSegment)* has attribute length and a domain for this attribute is the range of acceptable values. A domain is denoted with a *'D'*. ω is a function mapping concepts to a set of tuples specifying the attribute and its domain elements, which is a subset of D.
- **Relationship:** A concept can have an inheritance *(i)* inheritance relation or a composition *(c)* relation. An inheritance relation is described when one concept is derived from another, for example the concept road user *(RoadUser)* has an inheritance *(i)* relation with the concept car *(Car)*. A composition relation is described when one concept is part of another, for example the concept road segment *(RoadSegment)* has a composition *(c)* relation with concept lane segment *(LaneSegment)*. An inheritance *(i)* relation in OWL is shown as sub-class. A composition *(c)* relation is referred to as

'Object Property' in OWL (OWL Web Ontology Language Reference). A relation is denoted as *'R'*. A relation can be of inheritance or composition, and they are denoted as i and c respectively. Therefore, R is a finite set of tuples from $C \times C$ stating the concept pairs that are related. The function $\tau : R \times R \mapsto \{c, i\}$ assigns a type to each relation using i for inheritance and c for composition. Furthermore, $\psi : R \times R \mapsto \mathbb{N}_0 \times \mathbb{N}_0$ is a function mapping relationships solely of type c to its minimum and maximum arity. Futhermore, the arity is for specifying how many concepts a particular concept may comprise of with a range from 0 to any arbitrary natural number. For example, stating the concept road segment *(RoadSegment)* has a composition *(c)* relation of [1,..,4] with the concept lane segment. This means the concept road segment *(RoadSegment)* must have a minimum of 1 relation with concept lane segment *(LaneSegment)* and a maximum of 4. When describing the minimum and maximum arity of concepts * denotes the arity is any value. Otherwise, we have 0...* indicating a minimum arity of 0.

The definitions for ontology as outline by Wotawa et al. (Wotawa and Li, 2018a) and adopted for this research is as follows:

Definition 1 (Ontology):

"An ontology is a tuple (C, A, D, ω, R, τ, ψ) where C is a finite set of concepts, A is a finite set of attributes, D is a finite set of domain elements, ω:C⟶2A×2D is a function mapping concepts to a set of tuples specifying the attribute and its domain elements, and R is a finite set of tuples from C×C stating that two concepts are related. The function τ: R×R⟶{c,i} assigns a type to each relation using i for inheritance and c for composition. Furthermore, ψ: R×R⟶IN0×IN0is a function mapping relationship solely of type c to its minimum and maximum arity. The arity is for specifying how many concepts a particular concept may comprise and ranges from 0 to any arbitrary natural number."

This definition of ontologies ensures that only one relationship can exist between two concepts. Therefore, if that one concept is a sub-concept of another than a compositional relationship between them is not possible.

Definition 2 (Root concept; leaf concept):

"Given an ontology (C, A, D, ω, R, ψ), a concept c ∈C is a root concept if and only if there exists no relation (c', c, x) ∈R for c' ∈C, x∈{i,c}. A concept c ∈C is a leaf concept if and only if there is no relation (c, c',x)∈R for c'∈C, x ∈{i ,c}."

Definition 3 (Cyclic ontology):

"An ontology (C, A, D, ω, R, ψ) is called a cyclic ontology if and only if (i) there is a relation (c, c, x) ∈R for a concept c∈C and x∈{i,c}, or (ii) there are relations(c_0,c_1,x_0),...,(c_i,c_0,x_i) in R for i>0, concepts c_0,...,c_i in C and x_0,...,x_i∈{i,c}. In this case the sequence of relations is called a cycle. If an ontology is not cyclic, it is called a cyclic ontology."

Definition 4 (Well-formed ontology):

"An ontology (C, A, D, ω, R, ψ) is a well-formed ontology if and only if (i) it comprises exactly one root concept, (ii)it is acyclic, and (iii) where all its leaf concepts have attributes."

3.2.5. Ontology Layers

A layered model will be used to organize the information within the ontology, based on the work of Schuldt (Schuldt, 2017). Bagschik et al. (Bagschik and Maurer, 2018) used a similar approach and in their work they have used five layer to model high way scenes. Sauerbieret al. (Sauerbier *et al.*, 2018) supplement this layer in the form of a sixth layer (digital information), however, there is only a mention of the layer and has not been developed further. This research will use six layers to model the NLOS ontology. As shown in Figure 3.6, the layers include 'Environment', 'Road User', 'Object Type', 'Communication Network', 'Scene' and 'Scenario'.

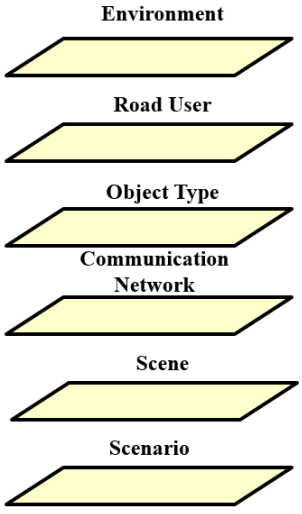


Figure 3.6: Ontology Layers Outline

The information and data within the layers has been derived from government guidelines for the creation of traffic infrastructure (*The Highway Code - Guidance - GOV.UK*). These guidelines describe the

creation of infrastructure, marking, junctions etc. There are also various scenario libraries from past projects or system descriptions, which contain knowledge about scenarios (Engel, 2020)(ASAM Standard)(PEGASUS, 2016)(NHSTA, 2021)(Ouyang *et al.*, 2020).

The concepts within a layer are related to one another, for example concepts with the 'Environment Layer' are related to each other to build up the topology of the road network (discussed further in Chapter 4). Concept across layers is related to one another, which allows interaction between layers. For example, the concept car *(Car)* in the 'Road User Layer' has a relation with the concept lane segment *(LaneSegment)* in the 'Environment Layer'.

To ensure the development of valid scenarios restriction are placed within the ontology, to ensure the arity value is not exceeded. For example, the concept road segment *(RoadSegment)* has one to [1,..,4] relation with the concept lane segment *(LaneSegment)* . Therefore, there cannot be 20 lane segment within a road segment depending on ODD. Furthermore, the first layer will be modelled according to the U.K. traffic rules. Details into modelling each traffic is not covered in this research as this goes beyond the scope and has been discussed extensively in previous research (Li *et al.*, 2016)(Buechel *et al.*, 2017)(Bagschik and Maurer, 2018).

The number of scenarios, which can be generated by a combination of all concepts is very large. However, only a limited number of scenarios are relevant for investigation, depending on the functional range. As a result, the knowledge base is influenced by the functional description of the system. However, in order to ensure the traceability of assumptions made during the analysis, it is critical to explicitly represent decisions on which entities or relations from the knowledge are irrelevant for the system. Expert knowledge obtained from the literature is another important aspect, in addition to traffic domain guidelines and functional descriptions. On the one hand, each guideline defines only a portion of the traffic infrastructure, and we must represent how these components can interact and what dependencies they have.

Once functional scenarios have been modelled, value ranges need to be assigned to it to model the logical scenarios. This research will model the parameter relation within the ontology. Each concept (represented by a word) in ontology represents multiple relation attributes (represented by parameters) in the physical state space. For example, roads in the traffic network may include attributes like length, width, road friction coefficient etc. Figure 3.7 illustrates a simple abstract functional scenario, which includes concept ego vehicle *(EgoVehicle)* from the 'Road User' layer and concept lane segment *(LaneSegment)*, road segment *(RoadSegment)* and building *(Building)* from the 'Environment' layer.

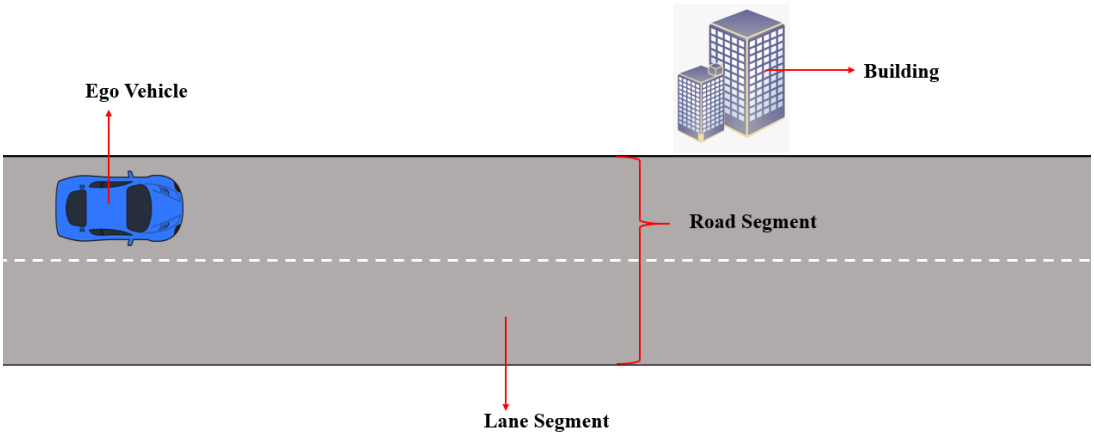


Figure 3.7: Abstract Functional Scenario

3.4. Logical Scenarios

Logical scenarios depict a detailed representation of functional scenarios. This is done by defining a value range of each attribute within a concept. For example, the concept road user *(RoadUser)* has the attribute speed, a logical scenario would be the domain of possible values this attribute can take.

3.4.1. Definition

Menzel et. al. (Menzel, Bagschik and Maurer, 2018a) define a logical scenario as follows:

"Logical scenarios include operating scenarios on a state space level. Logical scenarios represent the entities and the relations of those entities with the help of parameter ranges in the state space. The parameter ranges can optionally be specified with probability distributions. Additionally, the relations of the parameter ranges can optionally be specified with the help of correlations or numeric conditions. A logical scenario includes a formal notation of the scenario."

Once the functional scenarios have been developed, logical scenarios represent the concepts and their relations with parameter value. For example, Figure 3.8 depicts functional scenarios that are transformed into logical scenarios by converting the linguistic representation into state space and specifying the scenario describing parameters. As a result, each term in the vocabulary must be assigned to parameters that define the term. In this example, the road segment is described via length, width and curve etc.

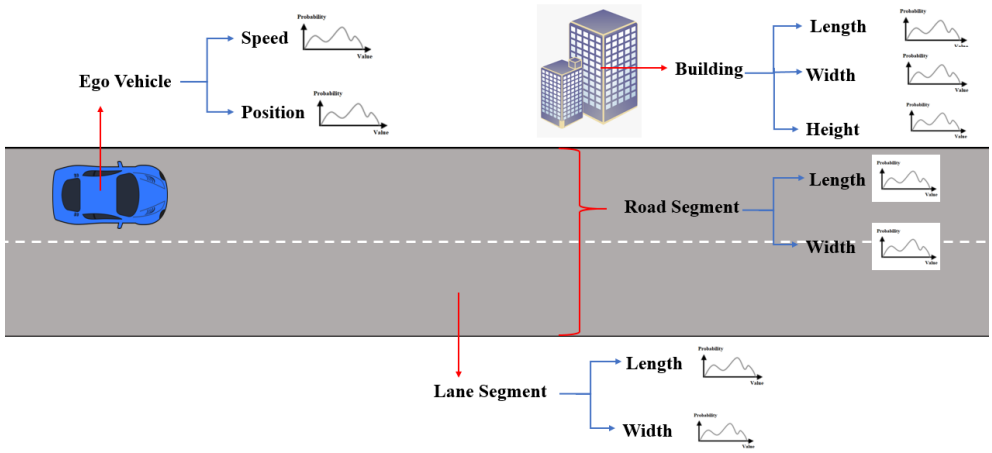


Figure 3.8: An Abstract Logical Scenario

Probability distributions e.g., Gaussian distribution, Uniform distribution, can optionally be specified for each parameter range to provide a more detailed description of those parameter ranges. Numeric conditions can also be used to specify relationships between parameter ranges e.g., the maximum speed of vehicle is less than or equal to speed. Alternatively, a correlation function can be used e.g., road width correlates with curve radius. A reduced set of parameters have been used to outline logical scenarios in this chapter. In reality, describing a single term from the vocabulary will necessitate many more parameters. For example, in Figure 3.6, additional parameter can be specified for example dimension, weight, engine power etc. for the ego vehicle.

The parameter's value range as well as the probability distribution with which it occurs in reality are specified. This information has been acquired from sources like U.K. government data publicly available (speed, road width etc.) (*Speed limits - GOV.UK*) (*The Highway Code - Guidance - GOV.UK*)(UK GOV, 2020a)(UK GOV, 2020a)(UK GOV, 2020a)(UK GOV, 2020a)(UK GOV, 2020a)(UK GOV, 2020a)(UK GOV, 2020a)(UK GOV, 2020a)(UK GOV, 2020a)(UK GOV, 2020a) ASAM standard (Engel, 2020)(ASAM Standard) and Scenario libraries (*PEGASUS - pegasus-EN* (NHSTA)(Ouyang *et al.*, 2020).Once the value ranges have been defined next concrete scenarios need to be selected.

3.5. Concrete Scenarios

Concrete scenarios use different parameters in the state space to describe concepts and their relationships. By selecting a concrete value from a parameter range of each attribute of a given concept, any logical scenario can be converted to a concrete scenario. During the testing phase, concrete scenarios can be used to generate test cases.

3.5.1. Definition

Menzel et. al. (Menzel, Bagschik and Maurer, 2018a) defines concrete scenarios as follows:

"Concrete scenarios distinctly depict operating scenarios on a state space level. Concrete scenarios represent entities and the relations of those entities with the help of concrete values for each parameter in the state space."

Any number of concrete scenarios can be derived from each logical scenario with continuous value range. For example, an infinite number of concrete scenarios can be generated by selecting an infinitesimal sampling step width for each parameter. By identifying and combining representative discrete values for each parameter, concrete scenarios can be efficiently selected. These scenarios can be turned into test cases in order to put autonomous driving to the test. Figure 3.9 depicts a concrete scenario derived from the logical scenario depicted in Figure 3.6. For each parameter, a concrete value within the defined value range as indicated with a red line on the graph has been chosen, while the parameters' specified conditions have been met.

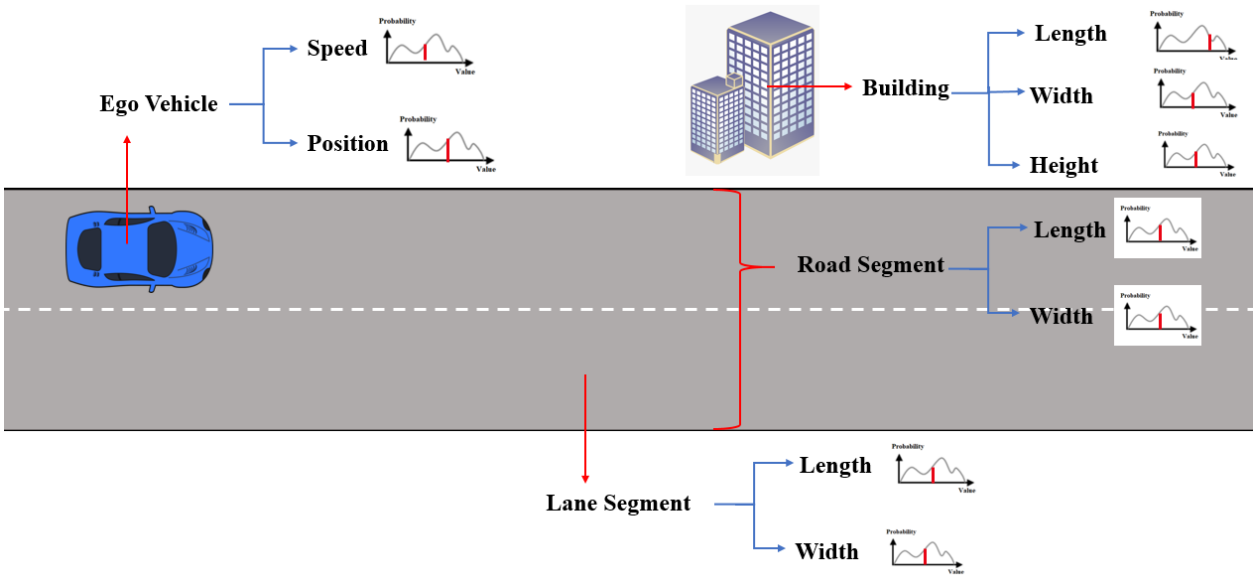


Figure 3.9: An Abstract Concrete Scenario

3.5.2. Scenario Selection

There are different methods in which concrete scenarios can be selected, these approaches include 'Testing – Based Selection' and 'Falsification-Based Selection' (Riedmaier *et al.*, 2020). These have been outlined below:

I. Testing-Based Selection

Within each logical scenario, the testing-based approach chooses a subset of concrete scenarios for microscopic safety assessments, with the option of combining the individual or microscopic results into a macroscopic assessment. These scenarios can be tested in a number of different ways. (Riedmaier *et al.*, 2020). These methods take samples from parameter distributions or ranges defined by minimum and maximum values. The former contains the probability of occurrence (exposure) of the scenarios and thus allows for a weighting of the results in order to make a true statistical macroscopic statement about accident probabilities. The latter aims to cover the entire parameter range while ignoring the importance of scenarios in the real world, allowing only a broad statement to be made based on the coverage

II. Falsification- based Selection

A falsification-based approach finds counter examples during microscopic assessment that violate the safety requirements or in other words are safety critical in nature. This can be accomplished by using either existing concrete scenarios from the database or logical scenarios with parameter ranges. Using an accident database as a scenario generator is one option. Another option is to take a realistic concrete scenario and make it more critical. The third option is to start with a logical scenario and look for critical scenarios within its defined parameter ranges (Riedmaier *et al.*, 2020). Instead of directly increasing criticality, increasing scenario complexity can increase the likelihood of finding a counter example.

This research will look into selecting concrete scenarios using a falsification-based method. Using this method scenarios are selected that violate the safety of the functionality under test. The different method within the literature for falsification methods are as follows:

A. Scenario from Accident database

Accident data is used to develop test scenarios as part of the safety evaluation of driver assistance systems. Various publications focus on different aspects of the of test scenarios from accident data. The GIDAS accident database is used by Stark et al.(Stark, Düring, *et al.*, 2019) (Stark, Obst, *et al.*, 2019) to investigate which requirements automated systems must meet in order to avoid as many accidents as possible, focus on urban areas. A motorway chauffeur (Feig, Gschwendtner and Borrack, 2019) goes through a similar procedure. In order to make a more general statement about the system's accident-avoidance potential, Ponn et al. (Ponn *et al.*, 2019) use accident scenarios as a starting point for a simulation-based parameter variation.. (So *et al.*, 2019) uses Big Data techniques to derive representative logical scenarios from a large accident database. The sole application of accident data do not correspond to a safety assessment of an AV (Level 3 and higher), as the

accident data only explores which accidents the system can avoid, not which accidents/risks will occur in the future as a result of the system, such as navigational algorithms of autonomous vehicles. Furthermore, the accident database does not cover all scenarios as there are certain scenarios that have not yet occurred or has not been captured.

B. Criticality- Based Selection

In Pierson et al., (Pierson *et al.*, 2019), a presented a methodology for determining the risk of real-world traffic situations in order to determine critical scenarios for AV testing. Their attention is drawn to the actions of other road users. The risk at a given location is determined by the road users' position and speed. A critical scenario exists if the AV's environment is deemed to pose a high level of risk. Real data from High D (dataset of naturalistic vehicle trajectories recorded on German highways) and NGSIM dataset of naturalistic vehicle trajectories recorded on U.S roads) data sets are used to evaluate the procedure. Althoff et al (Althoff and Lutz, 2018), and Klischat et al (Klischat and Althoff, 2019) focus on the generation of critical scenarios from complex urban scenarios. The criticality of a scenario is determined by the amount of space that the AV can safely use.

C. Complexity Based selection

Research conducted by Gao et al. and Xia et al. (F. Gao *et al.*, 2019) (Xia *et al.*, 2017)(Xia *et al.*, 2018) create a methodology that utilises combinatorial testing and complex test scenario definition to generate concrete scenarios. The Analytic Hierarchy Process is used to create a complexity index, that assigns a weighting to each parameter of a scenario. A lane-departure warning system used to evaluate and validate the process, and 16 different parameters are assigned in their work. The authors demonstrate more complex scenarios reveal more system errors. A scenario's complexity is divided into two categories in (Wang *et al.*, 2018) . The first is a Road Semantic Complexity score, which is calculated using Support Vector Regression and describes the static environment's complexity. The second is Traffic Element Complexity, which defines the complexity of the behaviour of up to eight traffic participants in the immediate vicinity. Zhang et al. (C. Zhang *et al.*, 2018) create a framework for selecting scenarios that takes complexity into account in order to efficiently test an AV's cognitive capabilities. They describe the cognitive task complexity based on the type of road, semantic content of the road segment, and difficult conditions such as fog. In a comparison of two autonomous driving platforms, they can demonstrate a negative correlation between performance and scenario complexity. Qi et al. (Qi *et al.*, 2019) make use a Scenario Character Parameter (SCP) that is based on failure trajectories. Scenario groups can be created and reduced to one relevant or challenging scenario by analysing the SCP. Ponn et al. (Ponn, Diermeyer and Gnandt, 2019) presents an optimization-based approach (without concrete implementation) for defining challenging scenarios. Parts of this approach are discussed in greater

depth in (Ponn *et al.*, 2019).(Guo, Kurup and Shah, 2020) and (Koopman and Fratrik, 2019) provide an overview of complexity-influencing factors.

D. Simulation-Based Falsification

In simulation-based falsification, an optimizer is used to take the microscopic assessment results of a scenario simulations from the visualised feedback loop. In a genetic algorithm, for example, depending on the population size, the optimizer processes one or more scenarios in parallel per iteration. To initialize, the optimizer must be loaded with concrete scenarios and the corresponding assessment results. The optimizer chooses the next concrete scenarios based on a cost function that includes a vehicle safety expression and sends them to the simulation tool for execution and evaluation. The optimizer determines the next concrete scenarios based on the new assessment results, and the next iteration begins. The optimizer can determine more and more critical scenarios for the vehicle under test by minimising the cost function with each iteration. This method usually necessitates a lot of iterations in a simulation environment. Other methods to determine critical scenario include Reinforcement Learning as used by the research group of Prof. Kochenderfer (Koren *et al.*, 2018; Lee *et al.*, 2018; Corso *et al.*, 2019) in a method they refer to Adaptive Stress Testing. Monte Carlo Tree Search and Deep Reinforcement Learning are used by Koren et al. (Koren *et al.*, 2018) . As a proof of concept, they chose a scenario in which a vehicle approaches pedestrian on a crosswalk. To account for both actions and sensor failures, the reinforcement learner produces pedestrian trajectories as well as sensor noise. The optimizer controls the coordinates of an object that the AV is approaching in this example, and the cost function is based on the TTC. They place an error in the sensor's field of view to see if the optimizer can find scenarios that take advantage of this flaw. To generate test scenarios, Mullins (Mullins, 2018) creates the Range Adversarial Planning Tool (RAPT) framework. He iteratively generates new concrete scenarios based on previous results using adaptive search algorithms. Tuncali et al. (Tuncali, Pavlic and Fainekos, 2016) use formal system requirements to falsify them rather than to verify them. They optimise using their automatic test generation tool S-TaLiRo and simulated annealing. They define a robustness metric as a cost function that quantifies the gap between the formal system requirements and their falsification. The robustness metric is attempted to be minimised by the optimization engine and stochastic sampler. Their metric for robustness is based on the TTC in this example, the collision-avoidance configuration, the optimizer controls the vehicle's target speed. It does not directly generate the time signal, but rather controls a set of control points that are then interpolated. Tuncaliet al. (Tuncali *et al.*, 2017)extend the approach by combining high-fidelity and low-fidelity models in a hierarchical framework. They employ a functional gradient descent optimization technique that outperforms simulated annealing. In (Tuncali *et al.*, 2018, 2020; Tuncali, 2019), covering arrays for combinatorial testing and simulated annealing for falsification are both applied to the overall system, including sensors, in a simulation-based framework. The idea is to improve the optimizer's initialization by using the results of another

scenario-selection technique. Starting with more promising conditions, the optimizer finds a better local optimum or converges faster to the global optimum. For an Autonomous Emergency Braking (AEB) System, Felbingeret al. (Felbinger *et al.*, 2019)compare falsification and combinatorial testing. The authors' findings show that both methods have been shown to find critical scenarios, but they do not evaluate their efficacy.

3.6. Summary

This research will use a falsification-based approach to select concrete scenarios. The concrete scenario selected will be critical in nature. This will be done by using key performance indicators (KPI) to determine the criticality metric of a scenario. A genetic algorithm will be used to select a critical scenario (concrete scenario) from the range of logical scenarios. This is further discussed in Chapter 5 and chapter 6. Figure 3.10 shows the workflow process of this thesis. Chapter 4 will conceptualise the functional scenarios, Chapter 5 will discuss the critically metric used to select critical scenarios. Chapter 6 will discuss the implantation of the choice of tool used to search the scenario space. Chapter 7, will test and validate the scenarios selected from the GA.

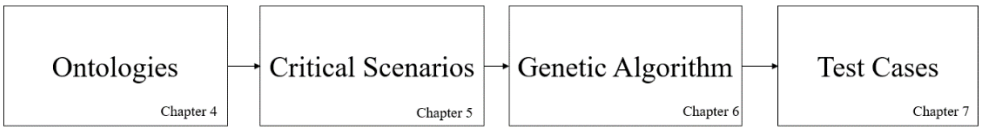


Figure 3.10: Thesis workflow process

Chapter 4: Development of Functional Scenarios

4.1. Introduction

The development of functional scenario within this research has been via ontologies. This chapter discusses the structure of the NLOS ontology. Furthermore, logical scenarios will be included within this chapter as domains *(D)* to the attributes. As discussed in Chapter 3, the knowledge within the ontology will be organized in six layers namely, 'Environment', 'Road User', 'Object Type', 'Communication', 'Scene' and 'Scenario'. The aim of developing this ontology is to generate NLOS scenarios through the relations and constraints defined within the ontology. To show the capability of scenario generation within the conceptualised ontology, this research will create a simple ontology from the conceptualised ontology and utilize OWL API to generate scenarios, this has been further discussed in Chapter 7. This means since this is a proof of concept, not all concepts and their attribute will be modelled on protégé, only a few concepts will be modelled to demonstrate scenario generation as well as communication within the scenario.

The ontology aims to develop NLOS scenario with the primary focus on junction for this thesis. However, the ontology can be easily expanded to include other NLOS scenarios by either adding a new concept or sub concepts. The road structure and road rules within the ontology will be following the United Kingdom (UK) road rules. The data for the domains of the attribute has been acquired as discussed in Chapter 3 as well U.K government website for example, speed limit. The ontology will be defined as a tuple in the form $(C, A, D, \omega, R, \tau, \psi)$ where:

- $C \rightarrow$ a set of concepts
- $A \rightarrow$ a set of attributes
- $D \rightarrow$ a set of domain elements
- $\omega \rightarrow$ This is a function mapping concepts to a set of tuples specifying the attribute and its domain elements. This is in the form $\omega: C \rightarrow 2A \times 2D$
- $R \rightarrow$ This states that two concepts are related
- $\tau \rightarrow$ Assigns a type of relation using inheritance (*i)* and composition *(c)*. An inheritance relation is described when one concept is derived from another, for example the concept road user *(RoadUser)* has an inheritance *(i)* relation with the concept car *(Car)*. A composition relation is described when one concept is part of another, for example the concept road segment *(RoadSegment)* has a composition *(c)* relation with concept lane segment *(LaneSegment)*.

- $\psi \rightarrow$ is a function that maps a composition relation to its minimum and maximum arity. For example, the concept road segment *(RoadSegment)* can have a minimum of 1 and maximum of 4 relation with concept lane segment *(LaneSegment)*.

This chapter is laid out as follows:

- **Section 2:** An overview of the ontology layers and their interaction
- **Section 3:** The coordinate system used to define the position within each concept.
- **Section 4:** The 'Environment' layer and its concepts, attributes and relation
- **Section 5:** The 'Road User' layer and its concepts, attributes and relation
- **Section 6:** The 'Object Type' layer and its concepts, attributes and relation
- **Section 7:** The 'Communication' layer and its concepts, attributes and relation
- **Section 8:** The 'Scene' layer and its concepts, attributes and relation
- **Section 9:** The 'Scenario' layer and its concepts, attributes and relation

4.2. NLOS Ontology Introduction

In this section we give an outline of the proposed ontology layers. Figure 4.1 shows and overview of the NLOS ontology layer.

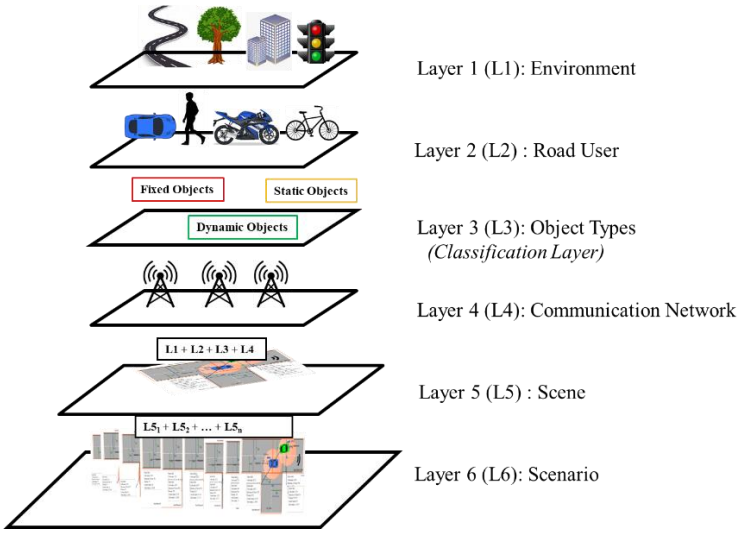


Figure 4.1: NLOS ontology Layers

4.2.1. Layer 1

The first layer within the ontology is 'Environment'. The environment layer has the following concepts: road segment *(RoadSegment)*, lane segment *(LaneSegment)*, junction *(Junction)*, lane marking *(LaneMarking)*, road network *(RoadNetwork)*, road infrastructure *(RoadInfrustructure)*, pavement *(Pavement)*, road hazard *(RoadHazard)*,traffic infrastructure *(TrafficInfrastructure)*, weather *(Weather)*. This layer focuses on building a road network with concepts, attributes and their domain. For example, Figure 4.2 shows a simple road network *(RoadNetwork)* which includes a road segment *(RoadSegment)*, lane segment *(LaneSegment)* and lane marking *(LaneMarking)*. Each of these concepts *(C)* have attributes *(A)* as shown in the figure (within boxes). Section 4 describes all concepts, attributes their domain and relation of layer 1 in details.

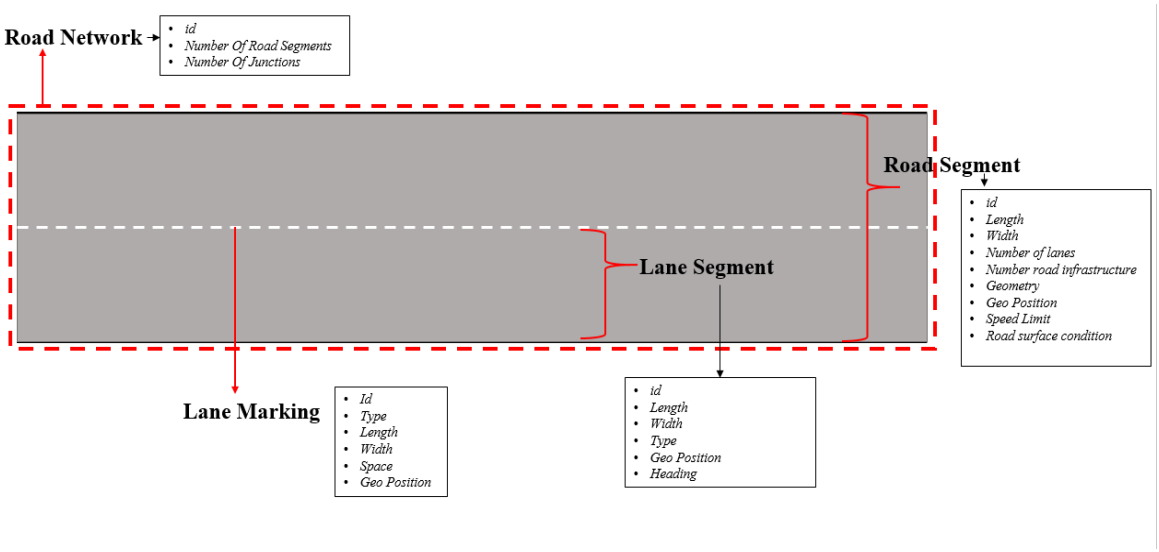


Figure 4.2: Layer 1, example, Concepts and Attributes

4.2.2. Layer 2

Layer 2 within the NLOS ontology is the 'Road User' layer. The concepts within this layer include road user *(RoadUser)* connected autonomous vehicle *(ConnectedAV)*, ego vehicle *(EgoVehicle)*, car *(Car)*, pedestrian *(Pedestrian)*, cyclist *(Cyclist)*. Road user parts *(RoadUserParts)*: sensors *(Sensors)*, OBU *(OBU)*. This layer defines all participants within the road environment and their attribute. The concepts within this layer interact with concepts in layer 1 through a composition relation (shown in Figure 4.3). The concept road user *(RoadUser)* has a many to one relation with the concept lane segment. Therefore, multiple road user *(RoadUser)* can occupy a position within a lane segment *(LaneSegment)*. Figure 4.3 shows an instance of a connected AV *(ConnectedAV)* and its attributes on a lane segment *(LaneSegment).* Section 5 describes all concepts, attributes and their domain as well as the relation of layer 2 in details.

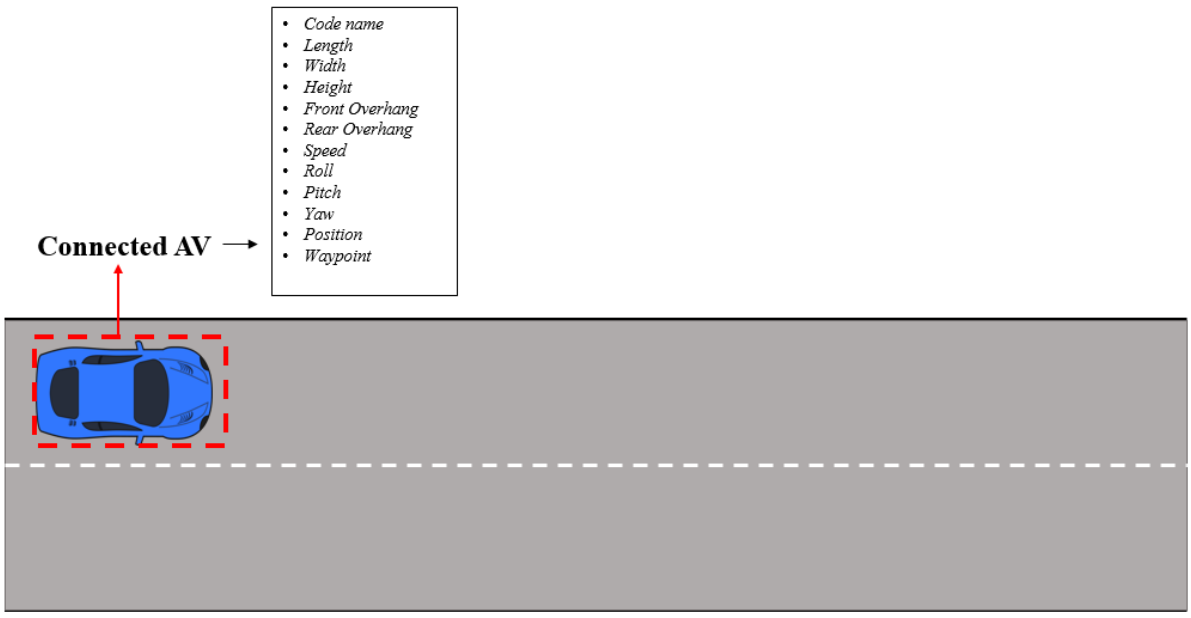


Figure 4.3: Layer 2, example, Concepts and Attributes

4.2.3. Layer 3

The next layer (Layer 3) is 'Object Types'. The concepts within this layer includes fixed object *(FixedObjects)*, stationary objects *(StationaryObjects)* and dynamic objects *(DynamicObjects)*. This layer categorizes the concepts of Layer 1 and 2. The importance of this layer is to avoid a cyclic ontology by having an autonomous vehicle detect itself. Therefore, autonomous vehicle *(AutonomousVehicle)* has sensors *(Sensors)*, and these detect object types *(ObjectTypes)*. This has been depicted in Figure 4.4. Section 6 describes all concepts, attributes, their domain and relation to layer 3 in details.

4.2.4. Layer 4

The next layer (Layer 4) is the 'Communication Network'. The concepts within this layer include application *(Application)*, radio communication *(RadioCommunication)*, network communication *(NetworkCommunication)*, system performance *(SystemPerformance)*, where radio communication *(RadioCommunication)* represents the physical aspects of communication, while network communication *(NetworkCommunication)* represent aspects of the protocols. A connected AV *(ConnectedAV)* has a relation with the concept communication network *(CommunicationNetwork)*. Section 7 describes all concepts, attributes their domain and relation of layer 4 in details. Figure 4.4 presents depicts the interaction of these layers

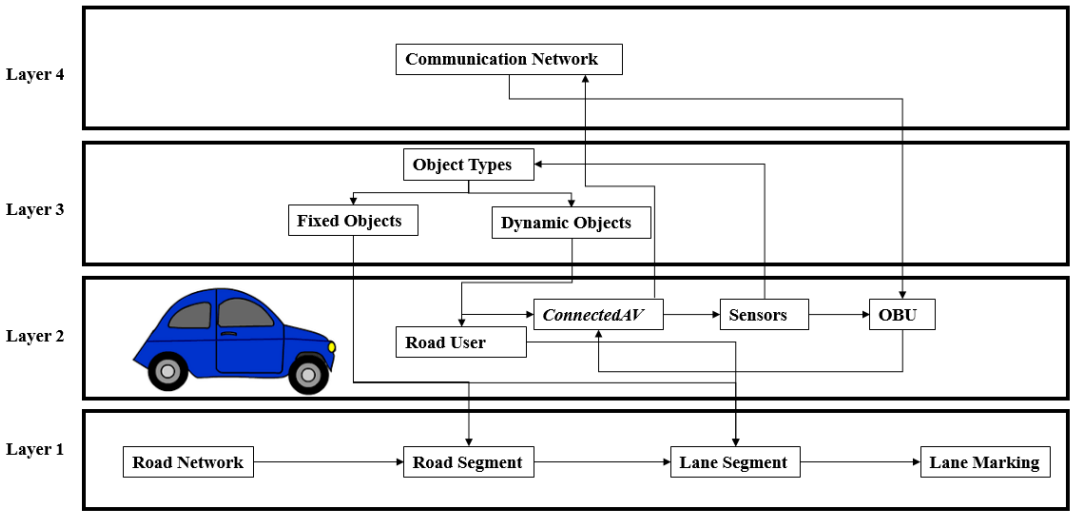


Figure 4.4: Layer 1-4 interactions

4.2.5. Layer 5

Figure 4.5 shows a combination of the domains from the first four layers through the red dotted box which makes up a 'Scene' (Layer 5). The concept scene *(Scene)* is a layout in which of a driving scenario. Section 8 describes all concepts, attributes their domain and relation of layer 5 in details.

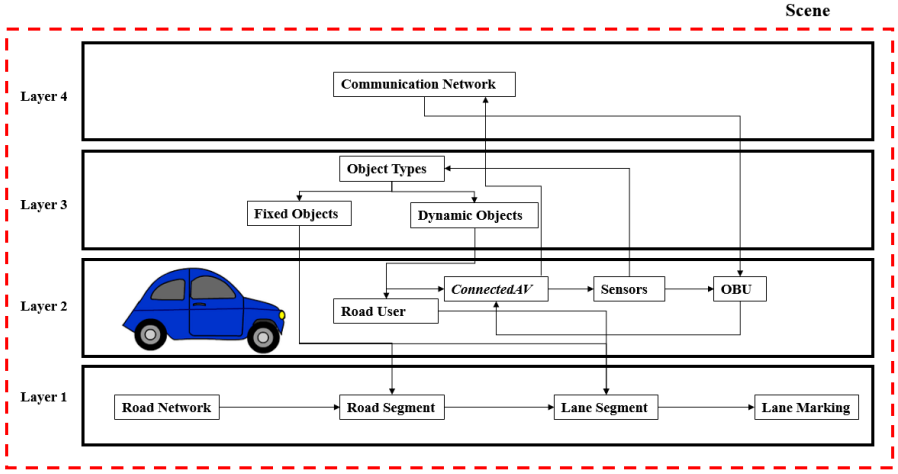


Figure 4.5: Scene Layer

4.2.6. Layer 6

Layer 6 is the 'Scenario' layer and has one concept *(Scenario)*. A Scenario is a collection of chronologically consecutive scenes, that take place one after the other. The collation of the domain of scene *(Scene)* needs to be in a successive manner starting from the first scene, as shown in Figure 4.6, the last scene is determined by the last position in the waypoint. Section 9 describes all concepts, attributes their domain and relation of layer 6 in details.

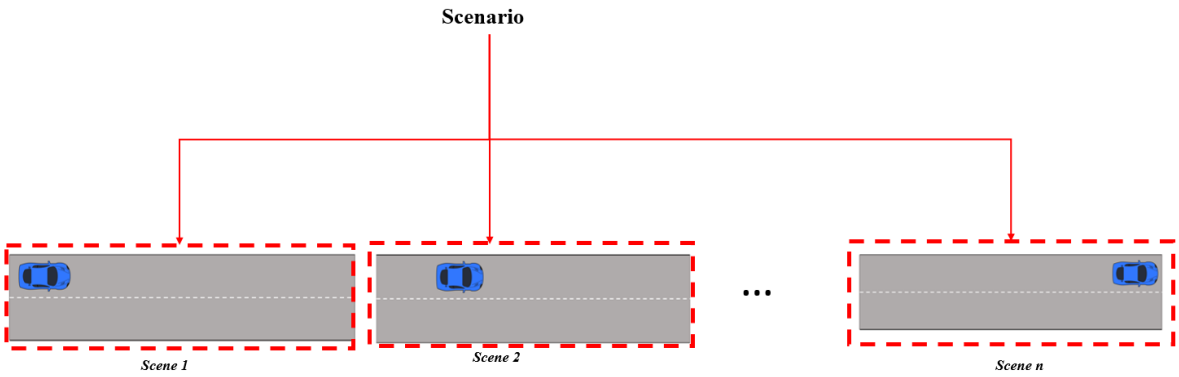


Figure 4.6: Scenario Layer

4.3. Coordinate System

The coordinate system used to define the position of concepts is the right-handed Cartesian world coordinate system defined in ISO 8855, where the Z-axis points up from the ground. Units are in meters. According to ISO 8855, the inertial system is a right-handed coordinate system with axes pointing in the direction depicted in Table 4.1. This is following the simulation environment (MATLAB) that the ontologies have been imported to.

Table 4.1: X,Y,Z direction and geographic reference

	Direction	**Geographic Reference**
X	right	East
Y	up	North
Z	Coming out of drawing plane	Up

4.4. Environment Layer

The layer ‘Environment’ represents knowledge of the physical surrounding of the ego vehicle. This layer aims to develop the foundation layer for NLOS scene. An example of a NLOS scene is illustrated in Figure 4.7 where the Ego Vehicle’s on-board sensors does not “see” AVCar_X456 due to the presence of the truck or potentially a platoon of trucks in between, and hence is not in the line of sight of the Ego Vehicle. Another example are junctions. The road segment and lane segments of this scene is part of the ‘Environment’ layer.

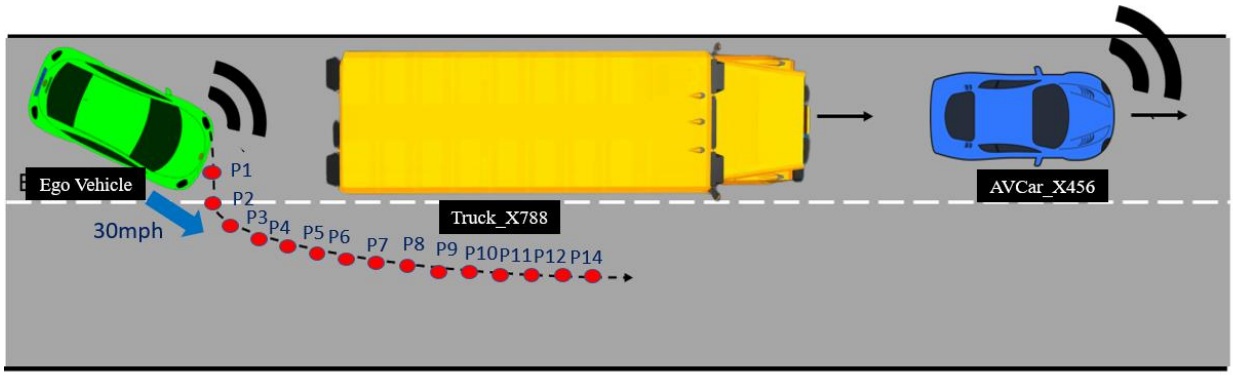


Figure 4.7: NLOS scene

The development of the ‘Environment’ layer has taken inspiration from previous research ontologies as well ASAM Open Drive. For the simulation environment, this research will use MATLAB (*MATLAB - MathWorks - MATLAB & Simulink*). Terms such as road segment *(RoadSegment)* and lane segment *(LaneSegment)* along with some of their relations has been extended from (Buechel *et al.*, 2017). Properties to describe road surface condition follow from the work of Kluech et al. (Klueck *et al.*, 2018). Attributes to describe the geometry of the road like gradient, curvature and width have been acquired from Li et al. (Li, 2020).

The structure of the ‘Environment’ layer is shown in Figure 4.8 and described as follows:

$$C_{Environment} = \{RoadSegment,\ LaneSegment, Junction, LaneMarking, RoadNetwork,\ RoadInfrustructure,\ Pavement,\ RoadHazard,\ TrafficInfrastructure,\ Weather\}$$

4.4.1. Concepts and Attributes

The terminology and attributes of each concept within the 'Environment' layer is discussed in Appendix A , Section 1.

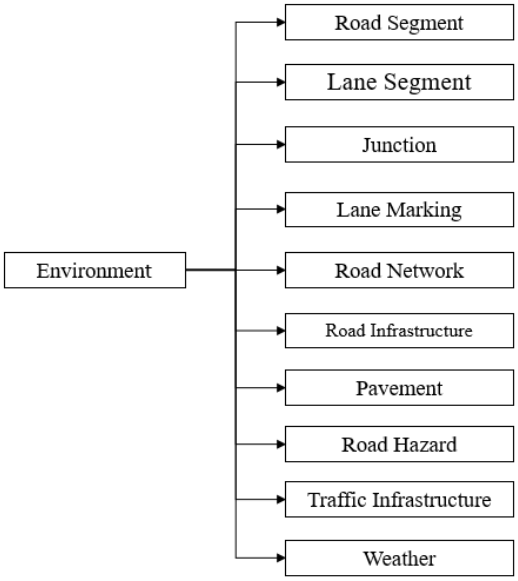


Figure 4.8: Environment Layer

4.4.2. Relations

This section will be discussing the relationship between the concepts *(C)* within the 'Environment' Layer. Relations could either be inheritance *(i)* or composition *(c)*.

4.4.2.1. Environment

The concept *(C)* environment *(Environment)* has an inheritance *(i)* relation with the following concepts: road segment *(RoadSegment)*, lane segment*(LaneSegment)*, lane marking *(LaneMarking)*, junction *(Junction)*, road network *(RoadNetwork)* , road infrastructure *(RoadInfrastructure)*, pavement *(Pavement)*, road hazard *(RoadHazard)* , traffic infrastructure *(TrafficInfrastructure)* and weather *(Weather)*. This relation is shown in Appendix B.

4.4.2.2. Road Segment, Lane Segment and Lane Marking

The concept road segment *(RoadSegment)* has a composition *(c)* relation [1,…,4] with the concept lane segment *(LaneSegment)*, this represents that the road segment *(RoadSegment)* has to have a minimum of 1 lane segment *(LaneSegment)* and a maximum of 4. The concept lane segment *(LaneSegment)* has a one-to-

one composition *(c)* relation with the concept lane marking *(LaneMarking)*, this represents that the lane segment *(LaneSegment)* must include a lane marking *(LaneMarking)*. For the lane segment *(LaneSegment)* with no lane marking the attribute type for lane marking *(LaneMarking)* would include 'unmarked'. The relation between road segment *(RoadSegment)*, lane segment *(LaneSegment)* and lane marking *(LaneMarking)* is represented

4.4.2.3. Road Segment, Road Hazard, Pavement

The concept road segment *(RoadSegment)* has a composition *(c)* relation [0,…,*] with the concept road hazard *(RoadHazard)*, this represents that the road segment *(RoadSegment)* can any number of road hazard *(RoadHazard)*in an instance or a maximum of 2. The concept road segment *(RoadSegment)* has a composition *(c)* relation [0,…,2] with the concept pavement *(Pavement)*, this represents that the road segment *(RoadSegment)* can have 0 pavement *(Pavement)* in an instance or a maximum of 2. The relation between road segment *(RoadSegment)*, road hazard *(RoadHazard)* and pavement *(Pavement)* is represented in Figure 4.17

4.4.2.4. Road Segment, Road Infrastructure and Traffic infrastructure

The concept road segment *(RoadSegment)* has a composition *(c)* relation [1,…,*] with the concept road infrastructure *(RoadInfrastructure)*, this represents that the road segment *(RoadSegment)* has to have a minimum of 1 road infrastructure *(RoadInfrastructure)*and no limit of the maximum. The concept road segment *(RoadSegment)* has a composition *(c)* relation [0,…,*] with the concept traffic infrastructure *(TrafficInfrastructure)*, this represents that the road segment *(RoadSegment)* has to have a minimum of 1 traffic infrastructure *(TrafficInfrastructure)* and no limit of the maximum.

4.4.2.5. Road Network, Junction, Road Segment and Weather

The concept junction *(Junction)* has a composition *(c)* relation [2,…,*] with the concept road segment *(RoadSegment)*, this represents that the junction *(Junction)* has to have a minimum of 2 road segment *(RoadSegment)* and no limit of the maximum. The concept road network *(RoadNetwork)* has a composition *(c)* relation [1,…,*] with the concept road segment *(RoadSegment)*, and a composition *(c)* junction *(Junction)* [0,…,*] with the concept road segment. This represents that the road network *(RoadNetwork)* can have a minimum of 1 road segment *(RoadSegment)*and no limit of the maximum and a road network *(RoadNetwork)* can have 0 junction *(Junction)* in an instance or a maximum of 2. The concept (*Weather)* has a composition *(c)* relation of one to one with the concept road segment *(RoadSegment)* . All the relation of the above concepts has been depicted in Figure 4.17. The UML diagram is in Appendix F

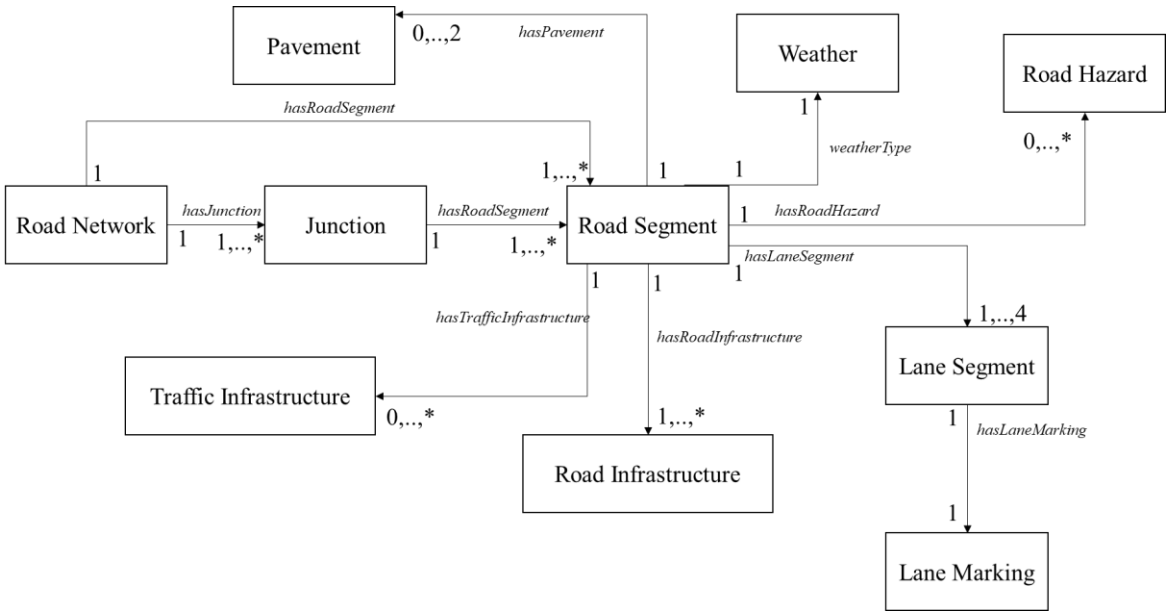


Figure 4.17: Environment Layer Relation

Each of the concepts are related to one another and the relation between the concepts are shown below:

R= {(Environment, RoadSegment), (Environment, LaneSegment), (Environment, Junction), (Environment, LaneMarking), (Environment, RoadHazard), (Environment, RoadInfrastructure), (Environment, RoadSideObjects), (Environment, Pavement), (Environment, RoadNetwork), (Environment, Weather), (RoadSegment, LaneSegment), (RoadNetwork, Junction), (Junction, RoadSegment), (RoadNetwork, RoadSegment),(RoadSegment, Pavement), (RoadSegment, RoadInfrastructure), (RoadSegment, RoadSideObjects), (LaneSegment, LaneMarking), (LaneSegment, RoadHazard).

For the relations we further specify their type they are either composition (c) or inheritance (i):

τ = {(Environment, RoadSegment) = i, (Environment, LaneSegment) =i, (Environment, Junction)=i, (Environment, LaneMarking)=i, (Environment, RoadHazard)=i, (Environment, RoadInfrastructure)=i, (Environment, RoadSideObjects)=i, (Environment, Pavement)=i, (Environment, RoadNetwork)=i, (Environment, Weather)=i, (RoadSegment, LaneSegment)=c, (RoadNetwork, Junction) =c, (Junction, RoadSegment)= c, (RoadNetwork, RoadSegment) =c, (RoadSegment, Pavement)=c, (RoadSegment, RoadInfrastructure)=c, (RoadSegment, RoadInfrastructure)=c, (LaneSegment, LaneMarking)=c, (RoadSegment, RoadHazard)=c, (RoadNetwork, Weather)=c

For ψ we give the minimum and maximum arity solely for the relation type ‘c’ and the values given to each is from 1…*, where * can be any positive integer and is shown for each relation as follows:

*ψ(RoadSegment, LaneSegment)= (1,..,4), ψ (RoadNetwork, Junction), = 1,…, * ψ (Junction, RoadSegment)= 1,…,*, ψ (RoadNetwork, RoadSegment)= 1,…,*, ψ (RoadSegment, Pavement)=(0,..,2), ψ (RoadSegment, RoadInfrastructure)=(1,..,*), ψ (RoadSegment, TrafficInfrastructure)=(0,…,*), ψ (LaneSegment, LaneMarking)=(1), ψ (RoadSegment, RoadHazard)=(0,.., *),), ψ (RoadSegment, Weather)=(1)*

4.5. Road User Layer

The concept road users *(RoadUsers)* define the different participants and actors that can be found within a road environment. Actor or participant include but not limited to vehicles, pedestrian and cyclist. Some of the concepts and sub concepts for the class road users *(RoadUsers)* has been extended from existing ontologies (Czarnecki, 2018).The structure of the ‘Road User’ layer is shown in Figure 4.18 and described as follows:

C = {RoadUsers, RoadUserParts}

$C_{RoadUser}$ ***= {ConnectedAV, Car, Pedestrian, Cyclist}***

$C_{ConnectedAV}$ ***= {EgoVehicle}***

$C_{RoadUserPart}$ ***= {OBU,Sensors }***

C_{OBU}***= {CommunicationBit, SensorData }***

$C_{Sensors}$ ***= {SensorInfo, SensorPlacement , Camera, Lidar, INS, GPS}***

C_{camera} ***= {CameraParameters, CameraDetection, CameraLimits }***

C_{Radar} ***= {RadarParameters, RadarDetection, RadarLimits }***

C_{Lidar} ***= {LidarParameters }***

C_{INS} ***= {INSParameters }***

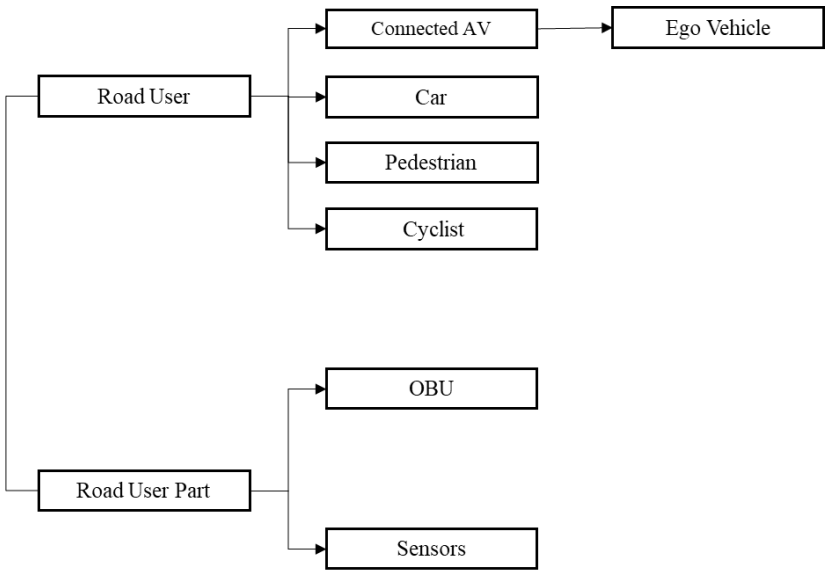


Figure 4.18: Road User Layer Concepts

4.5.1. Concepts and Attributes

The terminology and attributes of each concept within the 'Road User' layer is discussed in Appendix A, Section 2

4.5.2. Relations

This section describes the relations within the 'Road User' layer. The relation is discussed in terms of inheritance *(i)* and composition *(c)*. The concept *(C)* road user *(RoadUser)* has inheritance *(i)* relation with the following concepts: connected AV *(ConnectedAV)* , car *(Car)*, pedestrian *(Pedestrian)* and cyclist *(Cyclist)*. The concept *(C)* connected AV *(ConnectedAV)* has inheritance *(i)* relation with the concept ego *(Ego)*.The concept *(C)* road user parts *(RoadUserParts)* have an inheritance *(i)* relation with the following concepts: sensors *(Sensors)* and OBU *(OBU)*.The concept *(C)* sensors *(Sensors)* have an inheritance *(i)* relation with the following concepts: sensor info *(SensorInfo)*, sensor placement *(SensorPlacement)*, camera *(Camera)*, radar *(Radar)*, lidar *(Lidar)*, INS *(INS)*, GPS *(GPS)*.The concept *(C)* camera *(Camera)* has an inheritance *(i)* relation with the following concepts: camera parameters *(CameraParameters)*, camera detection *(CameraDetection)*, camera limits *(CameraLimits)*.The concept *(C)* radar *(Radar)* has an inheritance *(i)* relation with the following concepts: radar parameters *(RadarParameters)*, radar detection *(RadarDetection)* and radar limit *(RadarLimit)*.The concept *(C)* lidar *(Lidar)* has an inheritance *(i)* relation with the concept lidar parameters *(LidarParameters)*.The concept *(C)* INS *(INS)* has an inheritance *(i)* relation with the concept INS Parameters *(INSParameters)*. These relations can be visualised in Figure 4.18. This has been represented in Appendix C.

4.5.2.1. Road User and Environment

This section discusses concepts within the ‘Road User’ layer having a relation with concepts from the ‘Environment’ layer. The concept *I* road user has a composition *I* relation of many to one with the concept lane segment *(LaneSegment),* this represents that multiple road user *(RoadUser)* can have a relation with one lane segment *(LaneSegment)*.The concept pedestrian *(Pedestrian)* additionally has a composition *I* relation of many to one with the concept pavement *(Pavement).* These relations can be visualised in Figure 4.21

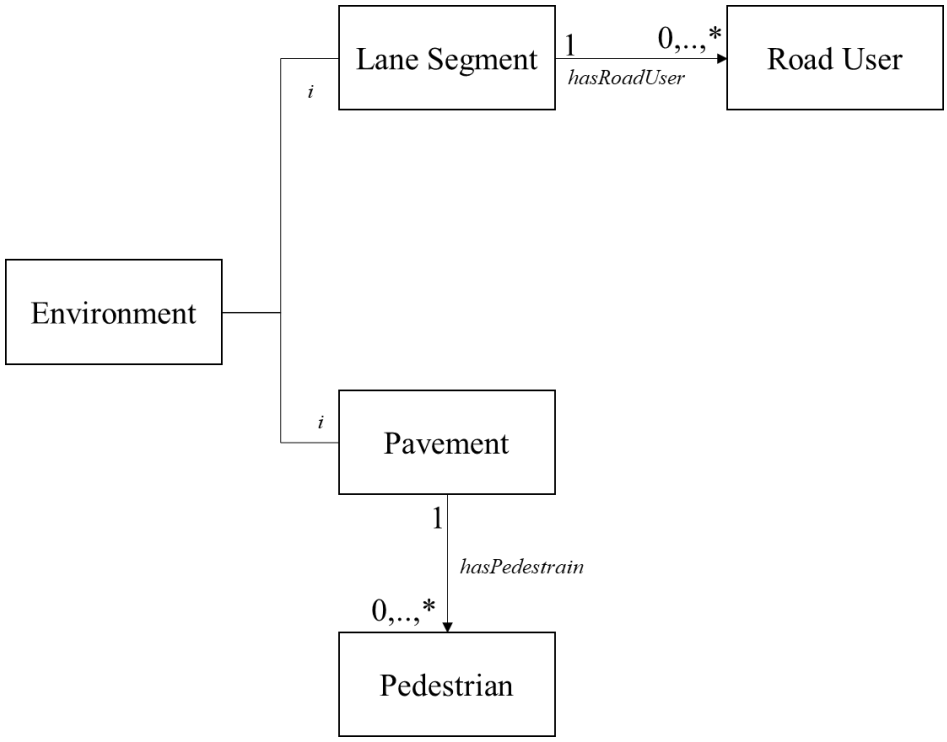


Figure 4.21: Environment Layer and Road User Layer relation

4.5.2.2. Connected AV, OBU and Sensors

The concept *I* sensors *(Sensors)* has a composition *I* relation of many to one with the concept OBU *(OBU),* this represents that multiple sensors has a relation with one OBU *(OBU)*.The concept *I* connected AV *(ConnectedAV)* has a composition *I* relation of one to one with the concept OBU *(OBU)*, this represents that one connected AV *(ConnectedAV)* has a relation with one OBU *(OBU)*. These relations can be visualised in Figure 4.22, where ‘i’ defines an inheritance relation.

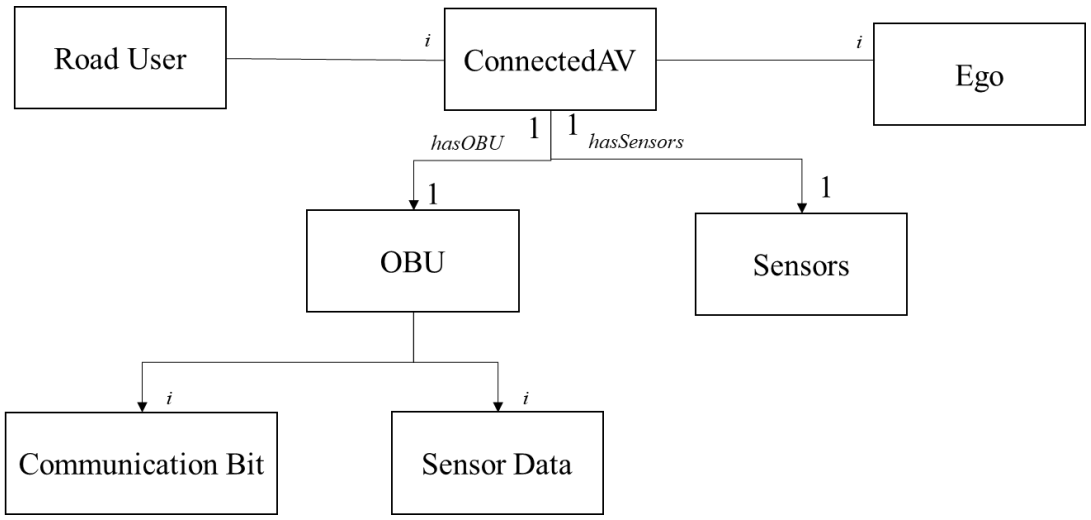


Figure 4.22: Connected Av, OBU and Sensor relation

4.5.2.3. Sensors

The concepts *I* camera *(Camera)*, radar *(Radar)*, lidar *(Lidar)*, INS *(INS)* and GPS *(GPS)* has a composition relation of one to one with the concept *I* sensor info *(SensorInfo)* and sensor placement *(SensorPlacement)*. Furthermore, the concept *I* sensors *(Sensors)* have a composition *I* relation of many to many with the concept object types *(ObjectTypes)*. This relation allows sensors to detect object types. These relations can be visualised in Figure 4.23, where 'i' defines an inheritance relation.

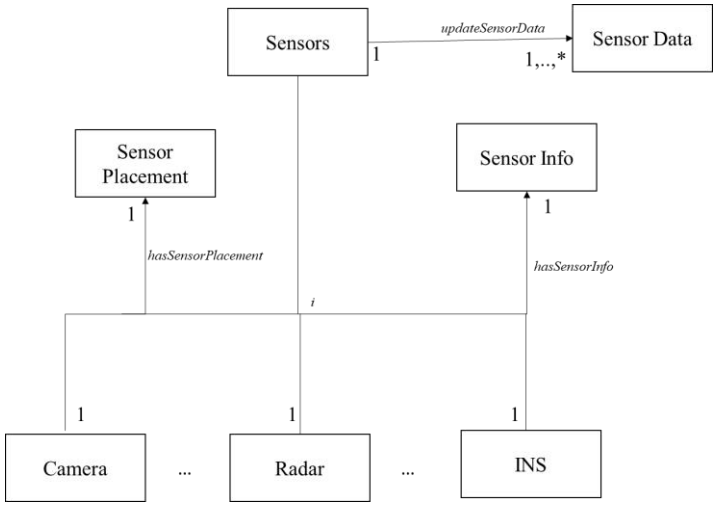


Figure 4.23: Sensor Relation

Each of the concepts are related to one another and the relation between the concept are shown below:

R= (RoadUser, Pedestrian), (RoadUser, Car), (RoadUser, ConnectedAV), (RoadUser, Cyclist), (ConnectedAV, EgoVehicle), (Sensors,SensorPlacement), (Sensors,SensorInfo), (Sensors,Camera), (Sensors,Radar), (Sensors,Lidar), (Sensors,INS), (Sensors,GPS), (OBU, CommunicationBit), (Sensors,SensorData),(RoadUser, LaneSegment), (Pedestrian, Pavement), (ConnectedAV,OBU) , (ConnectedAV, Sensors),(Sensor, SensorData),(Camera,SensorPlacement), (Camera,SensorInfo), (Radar,SensorPlacement), (Radar,SensorInfo), (Lidar,SensorPlacement), (Lidar,SensorInfo), (INS, SensorPlacement), (INS, SensorInfo), (GPS,SensorPlacement), (GPS,SensorInfo).

For the relations we further specify their type they are either composition I or inheritance (i):

τ = {(RoadUser, Pedestrian)=I, (RoadUser, Car) =I, (RoadUser, ConnectedAV) =I, (RoadUser, Cyclist) =I, (ConnectedAV, EgoVehicle) =I, (Sensors,SensorPlacement) =I, (Sensors,SensorInfo) =I, (Sensors,Camera) =I, (Sensors,Radar) =I, (Sensors,Lidar) =I, (Sensors,INS) =I, (Sensors,GPS) =I, (OBU, CommunicationBit) =I, (OBU,SensorData) =I ,(RoadUser, LaneSegment) =c, (Pedestrain, Pavement) =c, (ConnectedAV,OBU) =c, (ConnectedAV, Sensors) =c,(Sensor, SensorData) =c,(Camera,SensorPlacement) =c, (Camera,SensorInfo) =c, (Radar,SensorPlacement) =c, (Radar,SensorInfo) =c, (Lidar,SensorPlacement) =c, (Lidar,SensorInfo) =c, (INS, SensorPlacement) =c, (INS, SensorInfo) =c, (GPS,SensorPlacement) =c, (GPS,SensorInfo) =c }

For ψ we give the minimum and maximum arity solely for the relation type 'c' and the values given to each is from 1…*, where * can be any positive integer and is shown for each relation as follows:
ψ (RoadUser, LaneSegment) =[0,…,], ψ (Pedestrian, Pavement) =[0,…,*], ψ (ConnectedAV,OBU) =1, ψ (ConnectedAV, Sensors) =1, ψ (Sensor, SensorData) =[1,…,*], ψ (Camera,SensorPlacement) =1, ψ (Camera,SensorInfo) =1, ψ (Radar,SensorPlacement) =1, ψ (Radar,SensorInfo) =1, ψ (Lidar,SensorPlacement) =1, ψ (Lidar,SensorInfo) =1, ψ (INS, SensorPlacement) =1, ψ (INS, SensorInfo) =1, ψ (GPS,SensorPlacement) =1, ψ (GPS,SensorInfo) =1*

4.6. Object Types Layer

The 'Object Type' layer the concept I object type represents the various type of objects within the operational world that may be fixed, stationary or dynamic. This layer is important in bringing together the concepts of the ontology. It enables to prevent in developing a cyclic ontology, which would no longer fit the description of a well-formed ontology. This layer prevents forming a cyclic ontology in the following manner: An autonomous vehicle *(AutonomousVehicle)* cannot have a composition *I* relation with the concept road user *(RoadUser)* as well as an inheritance *(i)*. Furthermore, it classifies the different entities within the layers. The 'Object Type' Layer can be further expanded to include more concepts, for example, temporary object which

could represent the construction sites. To note, all environmental concepts as well as road user concepts fall under one of these categories.

The structure of the 'Object Type' layer is shown in Figure 4.24 and described as follows:

$$C_{Object\ Type} = [Fixed\ Objects, Stationary\ Object, Dynamic\ Object]$$

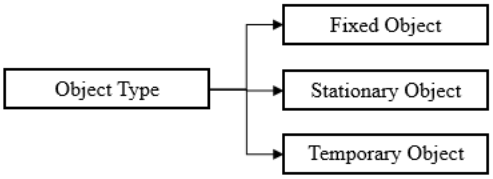


Figure 4.24: Object Type Concept

4.6.1. Concepts and Attributes

The terminology and attributes of each concept within the 'Object Type' layer is discussed in Appendix A, Section 3.

4.6.2. Relation

This section describes the relations within the 'Object Type' layer. The relation are discussed in terms of inheritance *(i)* and composition *I*.

The concept *I* object type *(ObjectType)* has inheritance *(i)* relation with the following concepts: fixed object *(FixedOBject)*, stationary object *(StationaryObject)* and dynamic object *(DynamicObject)*. The UML diagram is represented in Appendix D

4.6.2.1. Object Type and Environment

The concept road segment *(RoadSegment)* has a composition *(c)* relation of one to one with the concept fixed objects *(FixedObjects)*, this represents that each instance of the concept road segment *(RoadSegment)* has to have a relation with one fixed object *(FixedObjects)*. For example, if an instance of the concept road segment *(RoadSegment)* exist it must have the concept fixed object *(FixedObject)* existing to define the properties of the object in order to allow this data to be collated by the sensors and sent through the communication network. Similar logic holds for the relation that follows.

The concept lane segment *(LaneSegment)* has a composition *(c)* relation of one to one with the concept fixed objects *(FixedObjects)*, this represents that each instance of the concept lane segment *(LaneSegment)* has to have a relation with one fixed objects *(FixedObjects)*.The concept lane marking *(LaneMarking)* has a composition *(c)* relation of one to one with the concept fixed objects *(FixedObjects)*, this represents that each instance of the concept lane marking *(LaneMarking)* has to have a relation with one fixed objects *(FixedObjects)*. This relation exists for example to enable a sensor to detect a lane marking. The concept road infrastructure *(RoadInfrastructure)* has a composition *(c)* relation of one to one with the concept fixed objects *(FixedObjects)*, this represents that each instance of the concept road infrastructure *(RoadInfrastructure)* has to have a relation with one fixed objects *(FixedObjects)*.The concept pavement *(Pavement)* has a composition *(c)* relation of one to one with the concept fixed objects *(FixedObjects)*, this represents that each instance of the concept pavement *(Pavement)* has to have a relation with one fixed objects *(FixedObjects)*.The concept road hazard *(RoadHazard)* has a composition *(c)* relation of one to one with the concept fixed objects *(FixedObjects)*, this represents that each instance of the concept road hazard *(RoadHazard)* has to have a relation with one fixed objects *(FixedObjects)*. The concept traffic infrastructure *(TrafficInfrastructure)* has a composition *(c)* relation of one to one with the concept fixed objects *(FixedObjects)*, this represents that each instance of the concept traffic infrastructure *(TrafficInfrastructure)* has to have a relation with one fixed objects *(FixedObjects)*. These relations can be visualised in Figure 4.25. The UML diagram is represented in Appendix H.

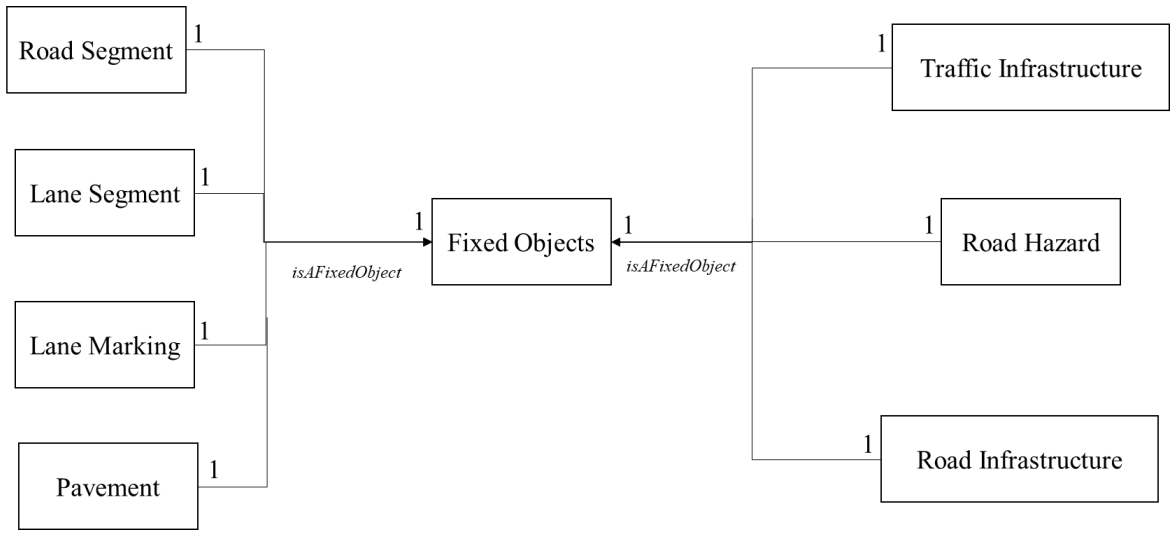


Figure 4.25: Fixed Object Composition Relation

The concept road user (*RoadUser)* has a composition *(c)* relation of one to one with the concept stationary objects *(StationaryObjects)*, this represents that each instance of the concept road user *(RoadUser)* has to have a relation with one stationary objects *(StationaryObjects)*.The concept road user (*RoadUser)* has a composition *(c)* relation of one to one with the concept dynamic objects *(DynamicObjects)*, this represents that each instance

of the concept road user (*RoadUser)* has to have a relation with one dynamic objects *(DynamicObjects).* These relations can be visualised in Figure 4.26, where 'i' defines an inheritance relation. The concept road user *(RoadUser)* must have a composite relation with either stationary objects *(StationaryObjects)* or dynamic objects *(DynamicObjects)*. The UML diagram is represented in Appendix I.

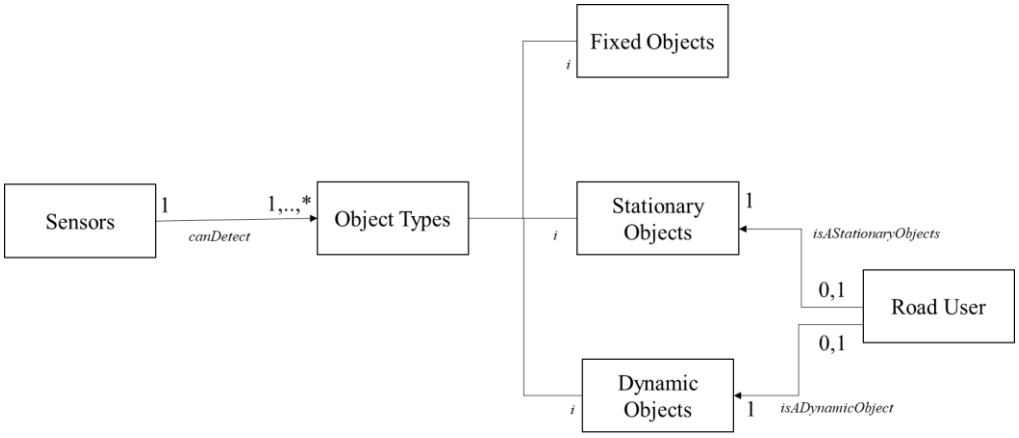


Figure 4.2: Sensor, Object Type and Sensor relation

Each of the concepts are related to one another and the relation between the concept are shown below:

R= (ObjectTypes, FixedObjects), (ObjectTypes, StationaryObjects), (ObjectTypes, DynamicObjects), (RoadSegment, FixedObjects), (LaneSegment, FixedObjects), (LaneMarking, FixedObjects), (Pavement, FixedObjects), (TrafficInfrastructure, FixedObjects), (RoadHazard, FixedObjects), (RoadInfrastructure, FixedObjects), (RoadUsers, StationaryObjects), (RoadUsers, DynamicObjects), (Sensors,ObjectTypes)
For the relations we further specify their type they are either composition (c) or inheritance (i):

τ = {(ObjectTyes, FixedObjects) =i, (ObjectTypes, StationaryObjects) =i, (ObjectTypes, DynamicObjects) =i, (RoadSegment, FixedObjects) =c, (LaneSegment, FixedObjects) =c, (LaneMarking, FixedObjects) =c, (Pavement, FixedObjects) =c, (TrafficInfrastructure, FixedObjects) =c, (RoadHazard, FixedObjects) =c, (RoadInfrastructure, FixedObjects) =c, (RoadUsers, StationaryObjects) =c, (RoadUsers, DynamicObjects) =c, (Sensors,ObjectTypes) =c }

For ψ we give the minimum and maximum arity solely for the relation type 'c' and the values given to each is from 1…*, where * can be any positive integer and is shown for each relation as follows:

ψ(RoadSegment, FixedObjects) =1 ψ(LaneSegment, FixedObjects) =1, ψ(LaneMarking, FixedObjects) =1, ψ(Pavement, FixedObjects) =1, ψ(TrafficInfrastructure, FixedObjects) =1, ψ(RoadHazard, FixedObjects)

=1, ψ (RoadInfrastructure, FixedObjects) =1, ψ (RoadUsers, StationaryObjects) =[0,1], ψ (RoadUsers, DynamicObjects) =[0,1], ψ (Sensors,ObjectTypes) =[1,..,]*

4.7. Communication Network

This section is looking into modelling the communication network layer requirements (Mittag *et al.*, 2009) (Papadimitratos, Gligor and Hubaux, 2006) for the ontology. To model the communication some assumptions will be made on some requirements.

4.7.1. Assumptions

This section will be outlining the assumption that will be made when modelling the 'Communication' Layer. The primary assumptions are as follows:

- Security aspects have been adhered to and the communication is secured.
- The structure of environment and road user support vehicle communication

4.7.2. Communication Network Layer

The concept communication network *(CommunicationNetwork)* has the following sub-concept: application *(Application)*, radio communication *(RadioCommunication)*, network communication *(NetworkCommunication)*, system performance *(SystemPerformance)*, security *(Security) as has been explained earlier* .

The structure of the 'Communication Network' layer is shown in Figure 4.27 and described as follows:

C = {Application, RadioCommunication, NetworkCommunication, SystemPerformance, Security}

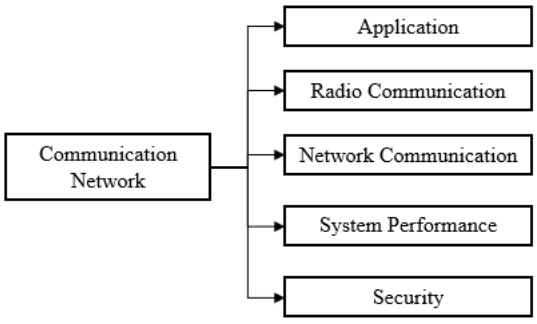


Figure 4.22: Communication Network Concept

4.7.2.1. Concepts and Attributes

The terminology and attributes of each concept within the 'Communication Network' Layer is discussed in Appendix A, Section 4 .

4.7.2.2. Relations

4.7.2.2.1. Communication Network

The concept communication network *(CommunicationNetwork)* has an inheritance *(i)* relation with the following concept: application *(Application)*, radio communication *(RadioCommunication)*, network communication *(NetworkCommunication)*, system performance *(SystemPerformance)* and security *(Security)*. The UML diagram is represented in Appendix E

4.7.2.2.2. Commination Network and connected AV

The concept *(C)* connected AV *(ConnectedAV)* has a composition *(c)* relation of one to many with the concept communication network *(CommunicationNetwork)*. This relation allows a connected AV to send information via a communication network. The concept *(C)* communication network *(CommunicationNetwork)* has a composition relation *(c)* of many to many with the concept communication bit *(CommunicationBit)*. This relationship allows many data from communication network to be sent to one vehicle. These relations can be visualised in Figure 4.28

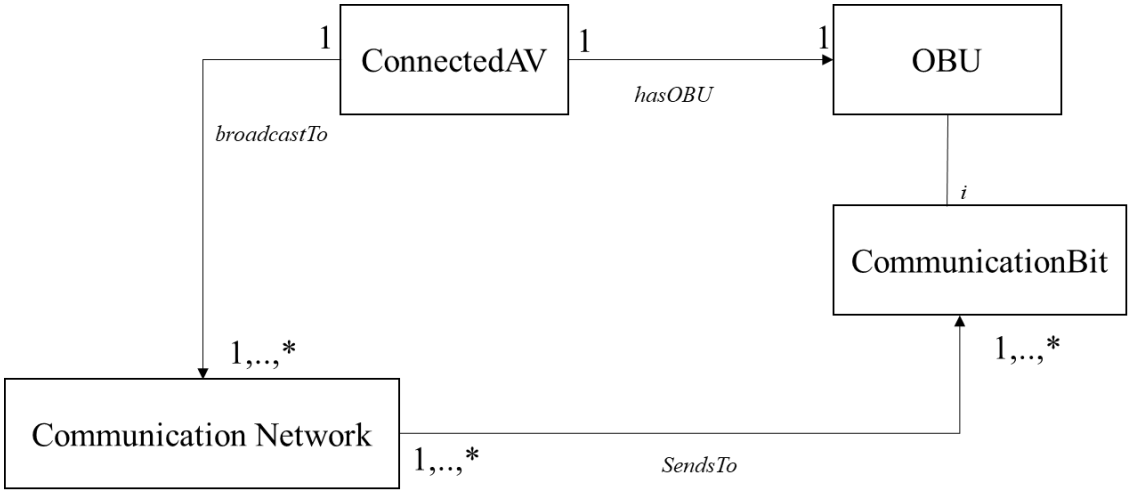


Figure 4.28: Communication Network and Connected AV relation

Each of the concepts are related to one another and the relation between the concept are shown below:

R= (CommunicationNetwork, Application), (CommunicationNetwork, RadioCommunication), (CommunicationNetwork, NetworkCommunication), (CommunicationNetwork, SystemPerformance), (CommunicationNetwork, Security), (RoadUser, CommunicationNetwork), (CommunicationNetwork, CommunicationBit)

For the relations we further specify their type they are either composition (c) or inheritance (i):

τ *= {(CommunicationNetwork, Application)=i, (CommunicationNetwork, RadioCommunication) =i, (CommunicationNetwork, NetworkCommunication) =i, (CommunicationNetwork, SystemPerformance) =i, (CommunicationNetwork, Security) =i, (RoadUser, CommunicationNetwork) =c, (CommunicationNetwork, CommunicationBit) =c }*

For ψ we give the minimum and maximum arity solely for the relation type 'c' and the values given to each is from 1…*, where * can be any positive integer and is shown for each relation as follows:

ψ*(RoadUser, CommunicationNetwork) =[1,..,*],* ψ*((CommunicationNetwork, CommunicationBit) =[1,..,*]*

4.8. Scene

4.8.1. Concept and Attributes

The terminology and attributes of each concept within the 'Scene' layer is discussed in Appendix A, Section 5

4.8.2. Relations

The concept *(C)* scene *(Scene)* has a composition *(c)* relation of one to many with the following concepts: road network *(RoadNetwork)* and object types *(ObjectTypes).* These relations have been depicted at Figure 4.30.Each of the concepts are related to one another and the relation between the concept are shown below:

R= (Scene, RoadNetwork), (Scene, ObjectType)

For the relations we further specify their type they are either composition (c) or inheritance (i):

τ *= {(Scene, RoadNetwork)=c, (Scene, ObjectType)=c}*

For ψ we give the minimum and maximum arity solely for the relation type 'c' and the values given to each is from 1…*, where * can be any positive integer and is shown for each relation as follows:

ψ *(Scene, RoadNetwork)=1,* ψ *(Scene, ObjectType)= [1,..,*]*

4.9. Scenario

4.9.1. Concept and Attributes

The terminology and attributes of each concept within the *'Scenario'* Layer is discussed in Appendix A, Section *6* .

4.9.2. Relation

The concept *(C)* scenario *(Scenario)* has a composite *(c)* relation of one to many with the concept scene *(Scene)*. This represents that one scenario could have a range of one-to-many scene.

The concept *(C)* object type *(ObjectType)* has a composite *(c)* relation of many to one with the concept scenario *(Scenario)*. This means that many object types *(ObjectType)* can have a relation with one scenario. These relations can be visualised in Figure 4.30. This can be visualised in Appendix E.

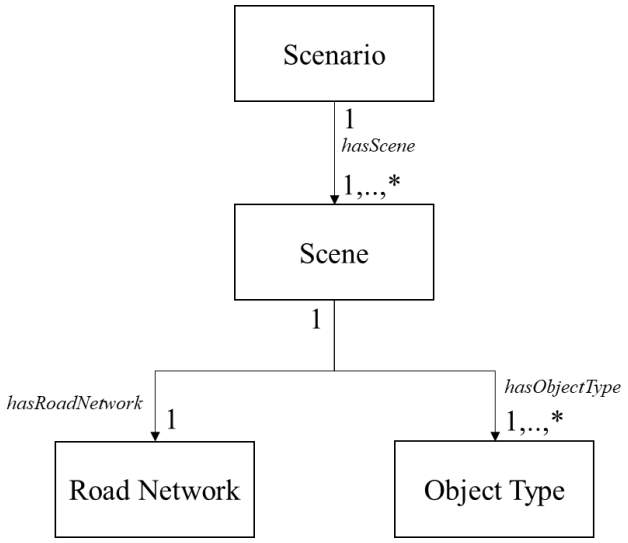


Figure 4.30: Scenario, Scene, Road Network and Object Type relation

Each of the concepts are related to one another and the relation between the concept are shown below:

$$R= (Scenario, Scene)$$

For the relations we further specify their type they are either composition (c) or inheritance (i):

$$\tau = \{(Scenario, Scene)) =c\}$$

For ψ we give the minimum and maximum arity solely for the relation type ‘c’ and the values given to each is from 1…*, where * can be any positive integer and is shown for each relation as follows:

$$\psi((Scenario, Scene) = [1,..,*]$$

4.10. Summary

This chapter defines the functional scenarios, through knowledge modelling using ontologies. The knowledge within the ontology is organised in six layers namely, ‘Environment’, ‘Road User’, ‘Object Type’, ‘Communication Network’, ‘Scene’ and ‘Scenario’. Each of these layers have concepts and their attributes which make up the layer. These layers have inter-layer as well as intra-layer relation which facilitates the generation of NLOS scenarios. The scenarios have been generated using OWL API, the process by which this is achieved is discussed in Chapter 7. The scenarios generated within functional scenarios are large in number. The next chapter will be discussing the selection of critical scenarios from a set of generated scenarios.

Chapter 5: Identification of Critical Scenarios

To test autonomous vehicle algorithms against test scenarios, requires the selection of a set of scenarios complying with testing criteria's from the entire range of scenarios which would help with the evaluation of the algorithm (Riedmaier *et al.*, 2020). It must be noted that selecting scenarios is an essential step as the number of possible test scenarios a vehicle can encounter is very large and it will be very time consuming to test against each scenario. In addition, many scenarios are trivial, in that, it will not lead to a safety critical scene. Further, many scenarios are essentially similar and hence testing against one is enough for the cluster of similar scenarios. While clustering is part of the future work, this thesis will select scenarios that have the potential to lead to a collision as a result of not being aware of the obstacle until it is too late to avoid collision. These scenarios are referenced as ***Critical Scenarios*** within this thesis. Chapter 4 discussed the development of functional scenarios with attributes having a range of values (logical scenarios). The next step is to select a concrete scenario from the range of possible scenarios. The functional scenarios output a range of abstract scenarios such as, a X-junction, a T-junction, a straight or curve road segment. After which logical scenarios provides a range of possible value to the attribute of the concepts defined in the functional scenario, example of some attributes include position, speed etc. These would output a large number of scenarios. However, not all scenarios can be tested as it would be too time consuming and costly. This chapter will be looking at the proposed methodology for selecting critical scenarios that satisfies using a defined safety critical measure condition Figure 5.1 show demonstrates the concept of generation of scenarios and only a couple are considered critical scenarios. The scenario id 4 and 6 represented with red dots represents critical scenarios.

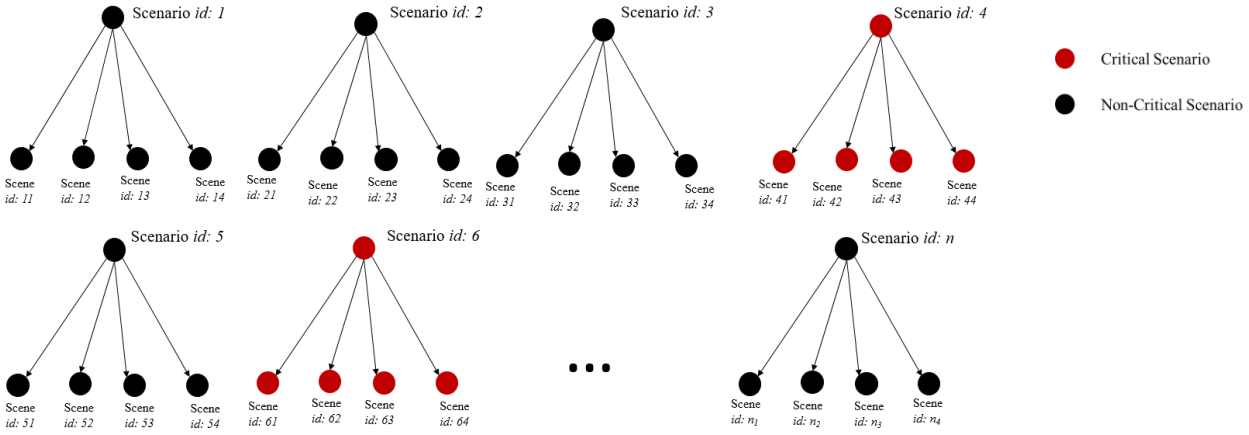


Figure 5.1: Critical Scenario Selection

5.1. Previous Works

In order to evaluate the safety within a specific concrete scenario, KPIs are required. Since accidents are rare, criticality metrics as KPIs are advantageous (Mahmud *et al.*, 2017). The most famous measure is Time to Collision(TTC) (Hayward, 1972). The goal is to detect scenarios which jeopardise vehicle safety by using criticality measurements. This section will be looking at what critically metric was used by previous works. Table 5.1 shows Key Performance indicators that have been used by previous research.

Table 2.1: Indicator Review

Indicator	Description	Collision Type	Reference
TTC or Time-Measured-To-Collision (TMTC)	The time it would take for the vehicles to collide if they continued on their current paths at their current speeds.	Rear-end collision, weaving/turning, colliding with objects/parked vehicles, crossing, and colliding with a pedestrian.	(Junietz *et al.*, 2018) (SchreierMatthias, WillertVolker and AdamyJurgen, 2016)
Time Exposed Time-to-Collision (TET)	Sum of all times (during the time period under consideration) when a driver approaches a front vehicle with a TTC-value less than TTC. A	As a result of a rear-end collision, turning/weaving, hitting objects/parked vehicles, crossing, and hitting a pedestrian, a vehicle-vehicle collision occurred.	(Minderhoud and Bovy, 2001)
Time Integrated Time-to-Collision (TIT	The integral when TTC is below the threshold	As a result of a rear-end collision, turning/weaving, hitting objects/parked vehicles, crossing, and hitting a pedestrian, a vehicle-vehicle collision occurred.	(Minderhoud and Bovy, 2001)
Modified Time-to-Collision (MTTC)	Models that were modified to account for all possible longitudinal conflict scenarios	As a result of a rear-end collision, turning/weaving, hitting objects/parked vehicles, crossing, and hitting a pedestrian,	(Yang, Ozbay and Bartin, 2010) (Yang, 2012)

	caused by acceleration or deceleration discrepancies.	a vehicle-vehicle collision occurred.	(Saunier and Sayed, 2008)
Crash Index (CI)	The effect of collision speed on the kinetic energy involved.	As a result of a rear-end collision, turning/weaving, hitting objects/parked vehicles, crossing, and hitting a pedestrian, a vehicle-vehicle collision occurred.	(Ozbay *et al.*, 2008)
Headway (H)	The time it takes for the front of the leading vehicle to pass a point on the road and for the front of the following vehicle to pass the same point.	As a result of a rear-end collision, turning/weaving, hitting objects/parked vehicles, crossing, and hitting a pedestrian, a vehicle-vehicle collision occurred.	(Taieb-Maimon and Shinar, 2016) (Vogel, 2003)
Time-to-Accident (TA)	The Time-to-Accident (TA) is the amount of time it takes for an accident to occur from the time one of the road users begins an evasive action and continues at the same speed and direction.	As a result of a rear-end collision, turning/weaving, hitting objects/parked vehicles, crossing, and hitting a pedestrian, a vehicle-vehicle collision occurred.	(Shbeeb, 2000) (Archer and Kungl, 2005)
Post-Encroachment Time (PET)	The time between the moment that a road user (vehicle) leaves the area of potential collision and the other road user arrives collision area	Mostly for right-angle or crossing crashes that result in a pedestrian being hit. Head-on collision when merging/diverging (to a certain extent).	(Songchitruksa and Tarko, 2006) (Cunto, 2008)

As mentioned earlier TTC is a well know representative of temporal metric and it is presented in (Saffarzadeh *et al.*, 2013). The TTC measure, while simple to calculate and interpret, is unable to account for the participants' acceleration and deceleration behaviour because it is a stationary metric. As a result, combining different metrics from different basic methods for a comprehensive assessment is what this chapter will discuss.

5.2. Critical Scenario Overview

For a scenario to be deemed a critical scenario, it must be possible to represent and evaluate a scenario's safety criticality measure. Each scenario has a unique combination of concepts and their attribute variable values as extracted from the ontology that can be evaluated against safety criticality measures to deem it to be a critical scenario. The most used metrics to measure the safety criticality of an on-road scene is TTC as used in (Minderhoud and Bovy, 2001). Time-to-collision can be defined as the time remaining from the current time t_c to a collision if the colliding actors (for example, the ego vehicle and an obstacle such as a pedestrian) continue in their current planned trajectory (speed and direction). The lesser the TTC, the more safety critical a scenario is. It must be noted that the presented research is focused on those critical scenarios where current time t_c denotes a time where the to-be-colliding objects are not in direct line-of-sight of each other. For example, obstacles present around corners in an intersection or in the blind corners of the ego vehicle. If the ego vehicle receives the location and trajectory information of the obstacle at time t_c it will have enough time to act (by changing speed and/or direction) and avoid the collision. The "time to act" is formally termed as total-stopping-time as explained below.

The total time taken for an automated vehicle to "see" i.e., detected by its sensor suite and react can be termed as the TST. Where TST is the time taken for the vehicle to travel the total-stopping-distance (TSD) which is made up of the braking distance and reaction distance. Reaction distance is itself the product of the perception-reaction time and the speed of the vehicle (Liang, Guler and Gayah, 2019).

Hence, a **critical scenario** as used within this research can be defined as an on-road traffic scenario involving at least two dynamic objects, the ego vehicle and an obstacle (a CAV, pedestrian or a cyclist), such that the ego vehicle and the obstacle are not in the line-of-sight of each other and their trajectories are such that it will definitely lead to a collision between the ego vehicle and the obstacle, unless the ego vehicle receives the information regarding the obstacle in time (TTC<TST) for it to react and avoid the collision.

The next section will be looking at how the critical scenario will be calculated. As mentioned above for a scenario to be critical certain conditions must be met, which goes as follows:

1. The position of the *'Ego Vehicle'* and *'Obstacle'* are such that they are not in line-of-sight
2. There must be a point of intersection between the trajectories of the vehicles also known as collision point (CP)
3. The time for the *'Ego Vehicle'* and *'Obstacle'* to reach the CP must be within a safety parameter also known as TTC

4. The time taken for the *'Ego Vehicle'* to perceive the *'Obstacle',* plus the time taken to react, plus time taken for the vehicle to apply its brakes and its velocity reach zero also known as the TST
5. A scenario will be considered critical if TTC < TST

5.3. Critical Scenario Calculation Framework

This section will be looking at how critical scenarios can be calculated. To demonstrate this an example of a T-Junction will be used with two road users and road infrastructure (building) as shown in Figure 5.2.

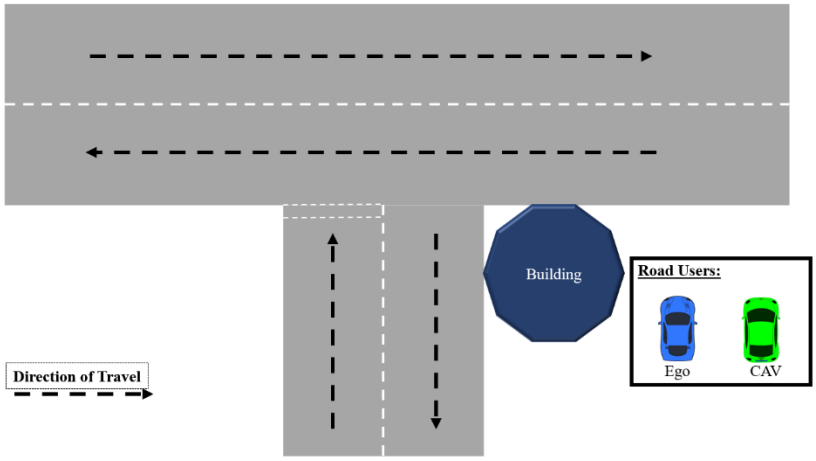


Figure 5.2: T-Junction

From Figure 5.2, it can be observed that that the road segment A is two-way and road segment B is one-way.

5.3.1. Vehicle Position and Placement

This section will be looking at the placement of the vehicle starting position. The position and placement of the road user will have within a NLOS format as discussed earlier. The model of the road segments and positions are shown in Figure 5.3. Chapter 7 gives a full description of the concepts and attributes used.

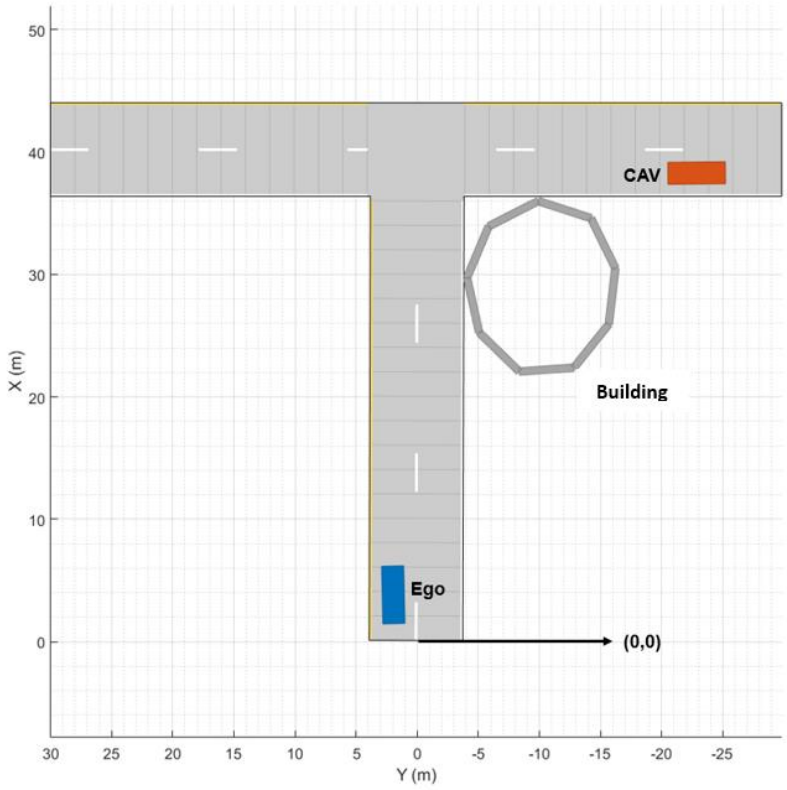


Figure 5.3: The position of the road segment and road user

Within our example scenario, there are two possible range of position (Shown in Figure 5.4) in which road user can be at in order to comply with the first condition i.e., 'The position of the 'Ego Vehicle' and 'CAV' are not within line-of-sight'.

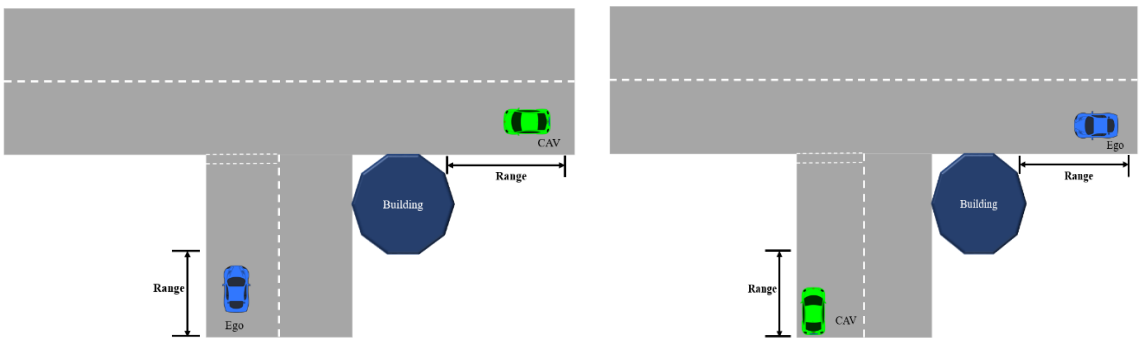


Figure 5.4: NLOS Vehicle Position in T-Junction Scenario

Figure 5.5 shows, two possible positions range the road user (Ego and CAV) can be at in order to have a NLOS. These position ranges have been determined by running simulations on MATLAB Driving Scenario (Automated Driving ToolBox, MathWorks) by using sensors to determine up to what point would the vehicles be unable to detect one another as shown in Figure 5.6

For validation of our proposed methodology, the current work will l be focusing on the vehicle position combination A from Figure 5.4. The position can have trajectory which comply with the road rules as shown in Figure 5.5. Many other trajectories are possible depending on the presented traffic scene at the moment and the path planning module of the ego vehicle and the CAV

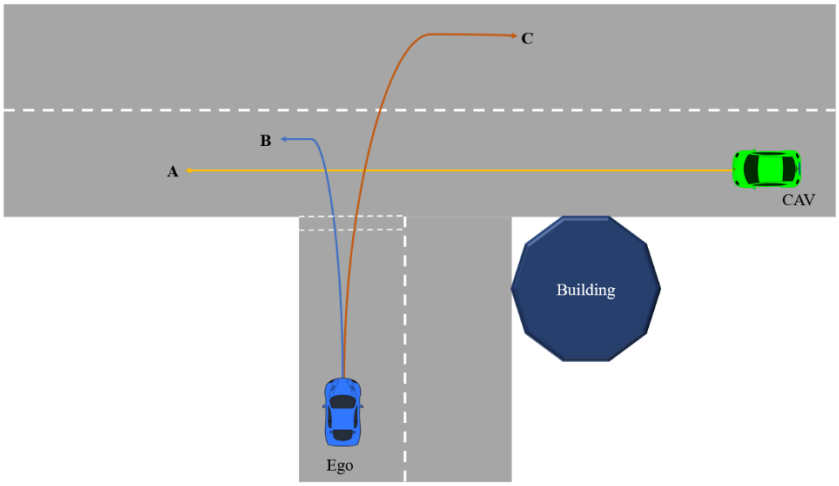
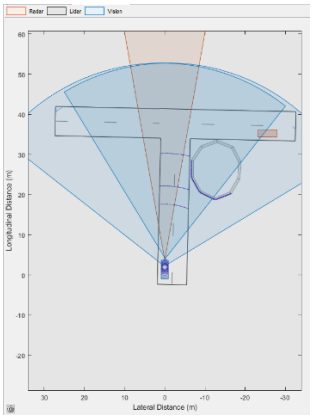


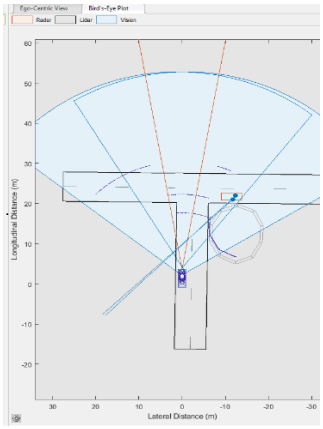
Figure 5.5: Vehicle Trajectory for NLOS scenario

Figure 5.5 shows the possible trajectory for the position combination selected. For this research we will focus on trajectory C for the ego vehicle and trajectory A for the CAV. However, it is important to note that trajectory B is an equally viable choice as it can produce scenario which will lead to a collision within NLOS scenario between the Ego and CAV. These positions have a range in which they will be considered as NLOS scenario.



A. Birds-Eye plot of initial position

B. 3D plot of initial position



C. Birds-Eye plot of point of detection

Figure 5.6: Position and Point of Detection

Figure 5.6 shows the simulation in MATLAB – Driving Scenario to determine the point of detection. This simulation was run a few times and sensor data was analysed to determine an upper bound and lower bound of the position ranges for the ego and CAV. Figure 5.7 illustrates both (Ego and CAV) have a range NLOS positions they can occupy, as represented by the black bounding box for each position.

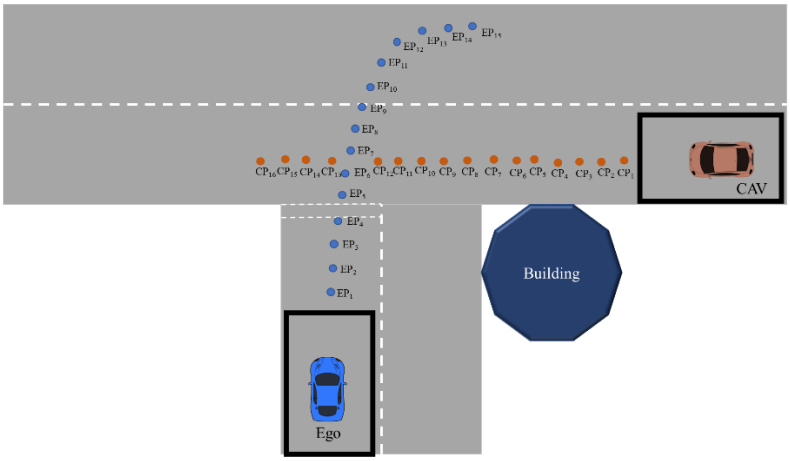


Figure 5.7: Shows range of possible position an Ego and CAV

5.3.2. Time-to-Collision and Collision Point

TTC is a proximal safety indicator (Archer and Kungl, 2005)(SVENSSON, 1998) defined by (Hayward, 1972): as Time to collision. This is the time required for two vehicles to collide if they continue at their present speed and trajectory where trajectory is the sequence of waypoints in consideration with the speed. To calculate TTC, the input variable used are as follows:

- Position, **p** (x_1, y_1, z_1).
- Speed, ***v*** (m/s)
- Sequence of waypoint [x_1, y_1, z_1, …, x_n,y_n, z_n]
- Orientation (Pitch, Roll and Yaw rate, ω (°/s))

Figure 5.8 shows a scenario of two vehicle (Ego and CAV) at a T-Junction. The scenario further shows the trajectories of the two vehicles and potential collision points if these vehicles carry on the same trajectory.

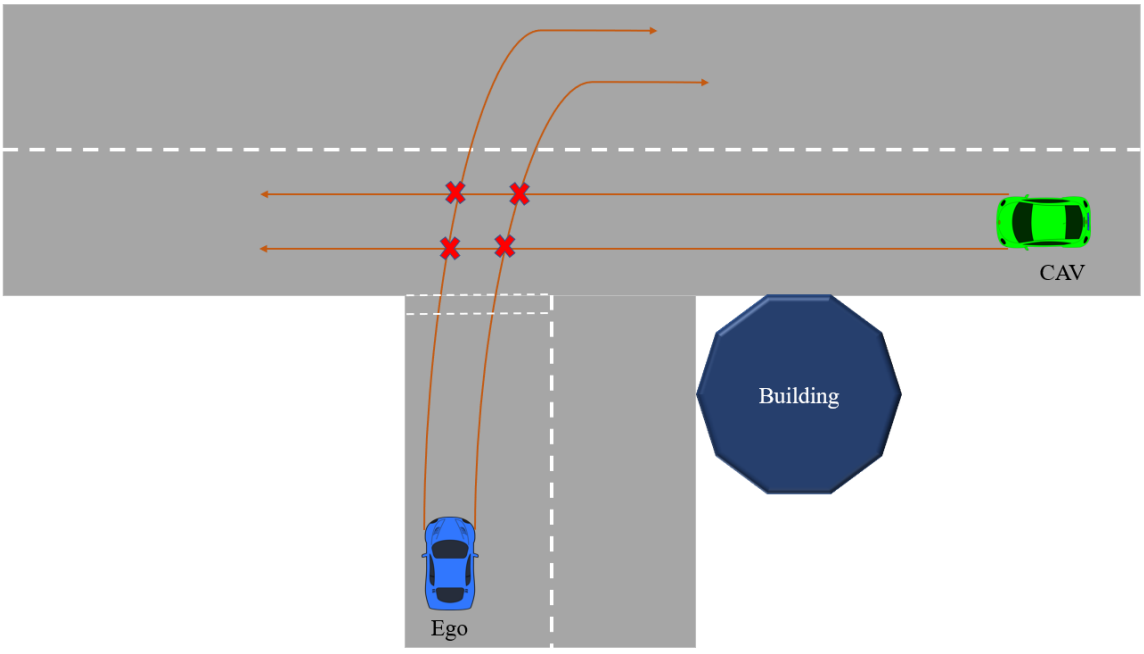
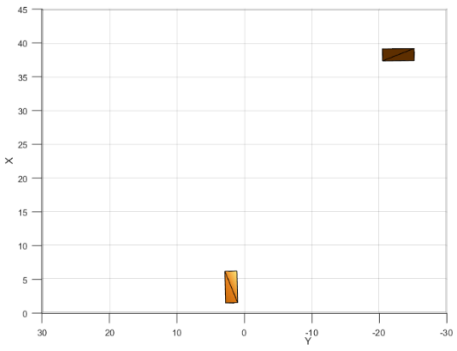


Figure 5.8: Collision Points between two vehicles

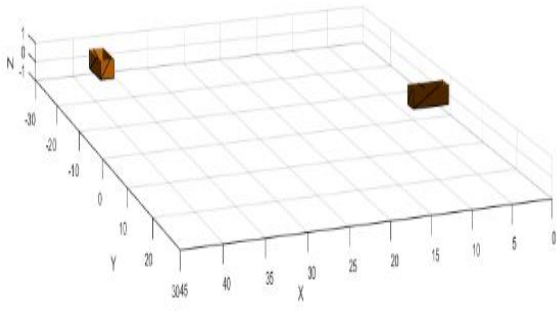
Much research has utilized TTC to determine the criticality of their scenarios (Kluck *et al.*, 2019). The two methods in which TTC can be calculated, are as follows, a calculation-based approach (Minderhoud and Bovy, 2001) and a simulation-based approach (Schwarz, 2014).A calculation-based approach first determines the point of collision and then calculates the time taken to reach that point . However, not all NLOS scenarios

are within a straight road and some of these scenarios may occur at junctions. Some calculation approaches have calculated TTC in scenarios where the vehicle does not have constant Yaw, however, not all inputs for this method of calculation will be available to the simulator at the start of the scenario. Therefore, the TTC determination needs to take into account the constant change in Yaw, especially within a junction scenario. Hence, a simulation-based approach would need to be utilized to determine TTC. Some simulation-based approaches have used a Kalman filter (Chen, Das and Bajpai, 2009) to determine the TTC. This is done by predicting the next possible trajectory.

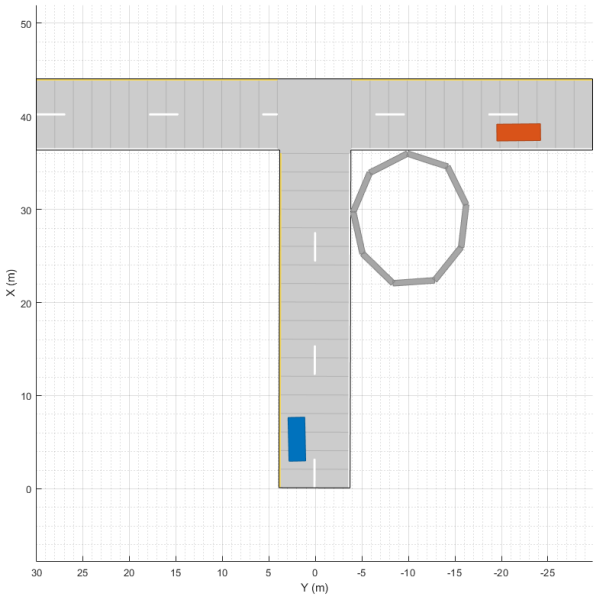
For this research the TTC is determine via the simulation-based approach. A simulation-based approach is used as this can be easily implemented for any scenario with any underlying road network with no or little change to the function to determine the TTC. This is more advantageous any mathematical calculation as it takes into consideration all varying parameters for example yaw and can be easily implemented to corner scenarios. To calculate TTC, the collision box function (Create box collision geometry, MATLAB, MathWorks) was used in MATLAB to create a bounding box around the vehicle. Next, checkCollison function within MATLAB to determine the position and time at which the collision box would intersect. The advantage of using this simulation approach is that is does not take too much computation power enabling to be utilized in an optimisation algorithm as shown in the next chapter (Chapter 6). Figure 5.9 shows the placement of the collision box on the ego and CAV respectively. A and B show the collision box view and position and C show the scenario view



A. Collision Box 2D



B. Collision Box 3D



C. Vehicle in Scenario

Figure 5.9: Vehicle and Corresponding Collision Box

To create the collision box, a function is developed in the following manner:

- The position and orientation (pitch, roll and yaw) of the vehicle
- The centre of the vehicle is determined considering the position and orientation of the vehicle
- With the centre resolved the length, width and height of the vehicle is calculated, which then provides the information to generate the collision box

When the simulation is running, the collision box is constantly updated to taking into account the change in position and/or orientation

5.3.3. Total-Stopping-Time

The TST for the vehicle which will include the breaking distance / time, time to detect and time to react , as shown in Equation 5.1.

$$TST = Time\ to\ Stop + Time\ to\ Detect + Time\ to\ React \quad (5.1)$$

I. Time-to-Stop (TTS):

The TTS can be acquired by first determining the braking distance. The braking distance is shown in Equation 5.2, 5.3 and 5.4 (Kavitha *et al.*, 2017)(Greibe, 2008).

$$d = \frac{1}{2K_a}\ln\left(1 + \frac{K_a}{K_t}v^2\right) \quad (5.2)$$

$$K_a = \frac{\rho}{2m}(C_D) \quad (5.3)$$

$$K_t = (0.01 + \mu)g \quad (5.4)$$

Where:

v → Velocity

ρ → Atmospheric density

m → Mass

g → Acceleration due to gravity

μ → Road friction coefficient

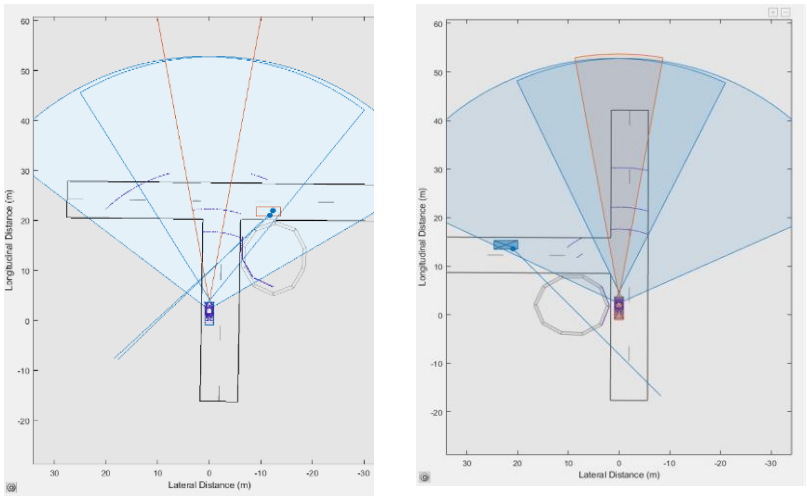
C_D → Drag factor

ation 5.5

$$'TS = {}^{v}/_{d} \quad (5.5)$$

II. Time-to-Detect (TTD):

To calculate TTD, the following inputs is required: Starting Position selected randomly from the range as determined by MATLAB simulation mentioned earlier, Position of detection and velocity. Figure 8 demonstrates the potential starting position and the position of detection.



A. Ego points of detection B. CAV point of detection

Figure 3: Point of Detection

The TTD, is shown in Equation 5.6

$$TTD = [Position\ of\ Detection - Starting\ Position]/velocity \tag{5.6}$$

III. Time-To-React:

This is the time taken to process the information received, which is assumed as negligible as system reaction time can be as low as 10 milliseconds (Hammerschmidt, 2019) thus enabling to select critical scenario for best case reaction time.

Therefore, TST can be acquired using Equation 5.7:

$$TST = TTS + TTD + TR \tag{5.7}$$

The next chapter will be discussing on utilizing this function to determine the criticality of a scenario

5.4. Summary

This chapter defines a critical scenario as an on-road traffic scenario involving at least two dynamic objects whose trajectories are such that it would lead to a collision and will be critical in nature when the TTC <TST or within a certain safety parameter. To search for critical scenarios in a scenario space, this research will utilise a genetic algorithm. The genetic algorithm uses a fitness function to guide it through the search space. The fitness function used in the genetic algorithm is the criticality metric which uses the KPI TTC and TST as discussed within this chapter. The limiting factor for selecting NLOS scenarios using this framework accommodates to two vehicles.

Chapter 6: Searching the Scenario Space using Genetic Algorithm

To search the scenario space a GA is utilised. A GA searches a solution space for one of the optimal solutions to a problem. This search is done in a fashion that mimics evolution where an initial population of possible solutions is formed, and new solution are formed by breeding the best solutions from the members of the population to form a new generation. The population evolves for a number of generations; when the algorithm is completed the best solution is returned. Chapter 4 discusses utilizing ontology for functional scenarios. The attributes that form the functional scenarios are each assigned a parameter range, resulting in a large number of possible concrete scenarios within each functional scenario. However, every scenario generated, though valid is not a safety critical scenario as discussed in Chapter 5. The challenge now is to search within the set of scenarios (or scenario space), those scenarios which are safety critical

In this chapter, we discuss on utilizing GA on discovering and identifying those safety critical scenarios or test cases for autonomous vehicle testing within a certain boundary. These boundaries are known as KPI and have been discussed in the previous chapter (Chapter 5). The KPI would help in selecting critical scenarios by determining different parameter value combination that give different level of criticality. By utilizing this approach, the total number of test scenarios are reduced to only having scenarios which are of interest within the study context. Hence, the algorithm that searches for critical scenarios must look for scenarios within the global scenario space, where the trajectories of objects intersect in the future and then determine the time to a collision to quantify its criticality. A measure used is TTC and smaller the TTC, the chances for a collision is higher. The value for the TTC is compared to the TST and if the values TTC < TST, the scenario is considered more critical. This is the fitness function in the context of GA. Searching algorithms have been used in the recent times to identify critical scenarios that satisfy a certain criterion from a database of scenarios (Feng *et al.*, 2019)(Klück *et al.*, 2019)(Kluck *et al.*, 2019). For example, this (Feng *et al.*, 2019) paper proposes a critical scenario searching method that selects local critical scenario through optimization methods (greedy sampling policy and ϵ-greedy sampling policy) and then searches the neighbourhood scenarios. On the other hand, (Klück *et al.*, 2019) and (Kluck *et al.*, 2019) uses genetic algorithm techniques while comparing it with random sampling and simulated annealing respectively.

The genetic algorithm within this research uses TTC as one of the indicators for the fitness function which we shall seek to minimize. Once an optimal critical scenario is identified, the GA is re-initiated to search for another optimal solution as it is well acknowledged that a solutions space may have several local alongside the global optimal solution. A solution in this context is a safety critical scenario.

This chapter discuss the final stage of scenario development (discussed in Chapter 3) – concrete scenarios. As discussed in the precious chapter the concrete scenario that are selected need to be a critical scenario. To search the critical scenario within the scenario space GA are utilized.

Problem Description: The problem the GA is trying to optimize is to identify a set of safety critical NLOS scenarios from the set of all NLOS scenarios in a particular road layout. Given a NLOS scenario what is one of the optimal solutions of the position and speed of both the vehicle that would lead to a critical scenario. As prove of concept for the proposed method, the NLOS scenario that the GA will seek to optimize is shown in Figure 6.1.

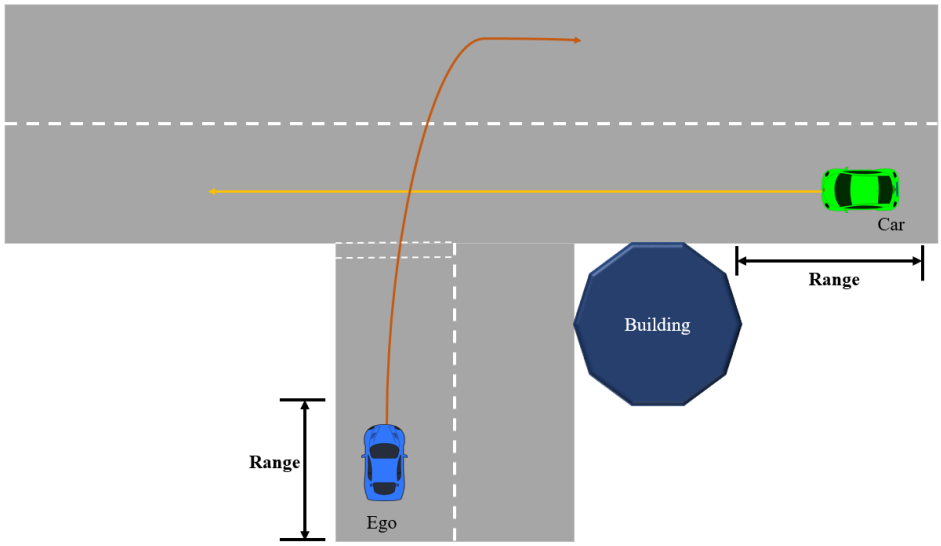


Figure 6.1: NLOS scenario

To generate a set of critical scenarios to test an autonomous vehicle, an underlying road network (which is a type of junction in this case) , with speed limits of the roads included as well as the surrounding infrastructure is selected from the ontology. Given this surrounding, the road user number and position range are selected as detailed below. Now with the given road layout, number of road user and position range of each road user, the number of possible scenarios is enormous and while some scenarios can lead to critical scenarios, others are benevolent. To select those that are malicious is the task of the GA. For the purpose of this work, a T-junction is chosen with speed limits determined from the ontology which itself was adopted from the UK speed limit data. Figure 6.1 shows the T-junction scenario with two vehicles (Ego and CAV). The input parameters for this scenario is the position of the ego, position of the CAV, speed of the ego and speed of the CAV. Some constraints to this are as follows:

→ Position of the Ego: The ego vehicle position can only be within a certain boundary condition to ensure it is still a NLOS scenario as shown in Figure 6.1 and explained in Chapter 5.

→ Position of the CAV: The CAV position can only be within a certain boundary condition to ensure it is still a NLOS scenario as shown in Figure 6.1 and explained in Chapter 5.

→ Speed: The ego vehicle and the CAV have speeds which comply with speed limit of the road and is determined using the actual vehicle speed distribution data from U.K government data (*Speed limits - GOV.UK*), which also considers over speeding .

Given the values of the attributes position and speed, the goal of the GA is to identify scenarios that are critical through utilization of the fitness function.

6.1 Terminology

The terminology that will be used to discuss the GA is as follows:

- *Individual:* Any possible solution in the scenario space
- *Population:* Group of all individuals in the scenario space
- *Search Space:* All possible solutions to the problem
- *Chromosome:* Blueprint for an individual
- *Gene:* A single attribute for an individual within a chromosome

6.2 Overview Genetic Algorithm

The GA is an optimization technique inspired by evolution theory that was first proposed by J. Holland (Whitley, Starkweather and Bogart, 1990). GA does not guarantee that it will find the best solution to the problem, but there is empirical evidence (Whitley, Starkweather and Bogart, 1990) that it will find acceptable solutions in a comparable amount of time to the other combinatorial optimization algorithms (Whitley, Starkweather and Bogart, 1990)

The GA is given a set of potential solutions to the problem that have been encoded in some way, as well as a metric called a fitness function that allows each candidate to be evaluated quantitatively. These options could be tried-and-true solutions that the GA is trying to enhance, but they're more likely created at random. The GA then assesses each candidate based on their fitness function activity; candidates who have a higher or lower (depending on the problem at hand) fitness have a better probability of being chosen than those who do not. These individuals with high/low fitness can be termed as promising candidates. These promising candidates are retained and given the opportunity to reproduce. Multiple copies are made from them, but the copies aren't flawless since random modifications are incorporated during the copying process. These digital progenies are then passed on to the next generation, generating a fresh pool of candidate solutions that are then put through a second round of fitness testing. Changes to their code resulted in a worsening or no improvement of candidate

solutions, which are again deleted; however, purely by chance, some individuals may have been improved, becoming better, more complete, or more efficient solutions to the problem at hand. The process is repeated after these winning individuals are chosen and copied over into the following generation with random changes. The expectation is that the population's average fitness will improve with each round, and that by repeating this process for a certain number of times (either a set number or till the fitness value does not improve for a certain number of threshold generations), very good solutions to the problem will emerge.

In most cases, each member of the population exists for a "generation" at the end of which the reproductive cycle begins, with all members of the population being considered for reproduction and selection being stochastically linked to fitness. Frequently, the term "elitism" is used, with a fixed percentage of the population's fittest members being passed down to the next generation.

Algorithm 1: ***Genetic algorithm for NLOS critical scenario selection***
Input: *Population Size, n* *Maximum number of iterations, MAX* ***Output:*** *Critical Scenario*
Begin: *Generate initial population (randomly) of n chromosomes Y_i (i = 1,2, …,n)* *Set iteration counter t= 0* *Compute the fitness values of each chromosome* ***while*** *(t < MAX)* *Step 1: The individuals are ranked according to the fitness value* *Step 2: The elitism rate is checked and depending on the value, the appropriate number of individuals are taken over to the next generation* *Step 3: Select a pair of chromosomes from current generation based on fitness* *Step 4: Apply crossover operation on selected pair with crossover probability* *Step 5: Select a chromosome from the current generation based on fitness* *Step 6: Apply mutation on the selected individual with mutation probability* *Step 7: Replace old population with newly generated population* *Step 8: Increment the current iteration t by 1* ***end while*** *return Critical Scenario* ***end***

6.3. Parameters of GA

There are five basic parameters of GA, including the following:

I. Population size: The population size determines the size of the initial and every population. If the population is small, there is little search space, so the local optimum is reached rather than the global optimum. However, if the population is too large, it expands the search area and increases the mathematical load, slowing down the process (Roeva, Fidanova, & Paprzycki, September 2013). As a result, the population size must be reasonable. The population size used within this research is 50, which is determined empirically through experimentation.

II. Number of generations: This is the number of cycles before the programme is terminated. Hundreds of loops are sufficient in some cases, but not in others, and this is dependent on the problem's complexity. All of these GA parameters are crucial because they determine the solution's quality (Yang, 2002) . The maximum number of generations used in this research is 100, which is determined empirically through experimentation.

III. Probability of Crossover (Pc): This value is the probability crossover occurs in the chromosomes in one generation is determined by this factor, which ranges from 0 – 1. The crossover rate is 0.85, which is determined empirically through experimentation.

IV. Probability of mutation (Pm): This determines the probability a mutate in a single generation; it ranges from 0-1. The probability of mutation used within this research is 0.1, which is determined empirically through experimentation.

V. Elitism rate (Er): The percentage of elitism. The percentage of elitism used within this research is 0.2, which is determined empirically through experimentation.

6.4. Genetic Representation

There are many methods in representing the chromosomes within a GA, depending on the problem. Some of the method of represent a GA are as follows:

- Binary Encoding: In this type of encoding, a chromosome is represented by a binary string; the Knapsack problem is an example of its application (Kumar, 2013).
- Permutation Encoding: This type of chromosome represents a position in a sequence, and it is used in problems involving ordering, such as the Traveling Salesman problem (Moz and Vaz Pato, 2007).
- Value encoding: A sequence of values is used to represent each chromosome. These values could be a character, a real number etc.

- Tree Encoding: Each chromosome is a collection of objects arranged in a tree (Kumar A., 2013). In genetic programming, this encoding is used.

This research will be representing the chromosomes using value encoding. Therefore, each chromosome will consist of the following genes:

[Position of the Ego, Position of the CAV, Speed of the Ego, Speed of the CAV]

Position of Ego and CAV: The Ego and the CAV have a possible range of position that they can be placed within a scenario in order to ensure the scenario remain NLOS. The waypoint and trajectories of both vehicles will remain consistent the only changes will be their initial position. This can be seen in Figure 6.1.

Figure 6.2 shows the position range for the ego and CAV. The upper and lower bound for the positions as used in the current context and derived from MATLAB simulation are shown in Table 6.1. The referencing coordinates system is explained in Chapter 5.

Table 6.1: Upper bound and Lower bound positions

	Ego	**CAV**
X (upper bound)	17.336	38.8
X (lower bound)	1.6323	37.2
Y (upper bound)	2.2	-17.6
Y (lower bound)	0.9	-27.9
Z (upper bound)	0	0
Z (lower bound)	0	0

Speed of Ego and CAV: The speed of both vehicles will be in accordance with the speed limit of the respected roads and the probability of the exact speed value selected will be determined from data acquired from government speed distribution statistics. The speed will be selected by first randomly selecting a speed limit with equal probability, the speed limit range is 20, 30, 50,60, 70 mph. Figure 6.2 shows the speed distribution for each speed limit.

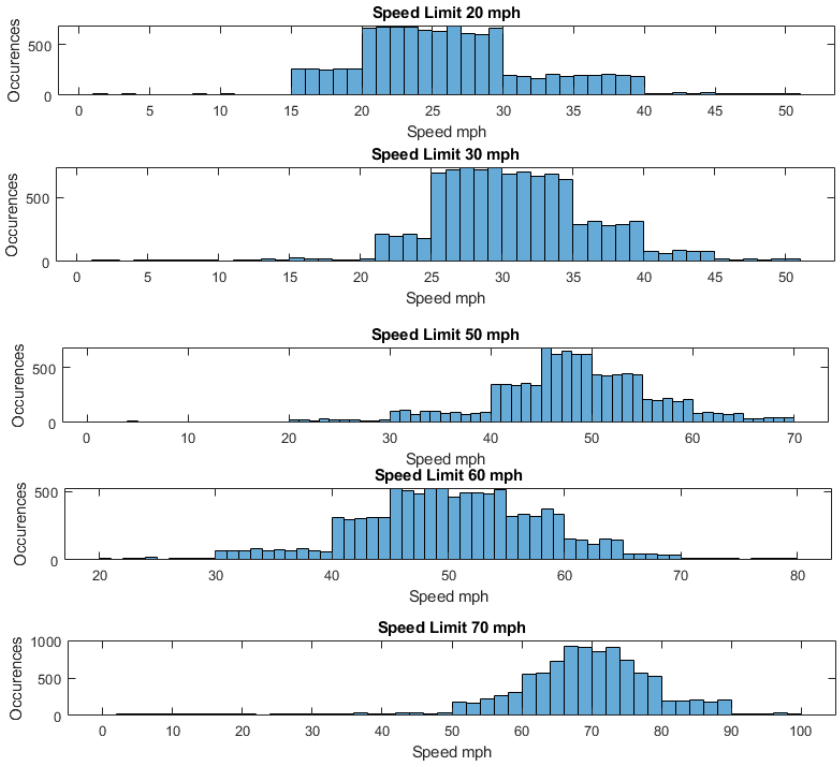


Figure 6.2: Speed Distribution

6.5. Fitness Function

The fitness function evaluates the fitness value f(Y) of each individual, Y_i, in the population. The previous chapter discussed on how critical scenarios can be selected by using KPI. The fitness function will be utilizing this calculation to determine the fitness of each individual. The KPI utilises the following to determine the criticality of a scenario:

1. Position: The position of the participants needs to be satisfying NLOS conditions. This has been covered in the population initialization and only position that are NLOS will be selected.
2. TTC: To calculate TTC a simulation-based approach will be used. A bounding box will be placed on each vehicle and TTC will acquired when these intersect.
3. TST: The TST is acquired using the following equation:

The TST is acquired by the Equation 6.1:

$$TST = Time\ to\ detect + Time\ to\ React + Time\ to\ stop \quad (6.1)$$

Next, the fitness values for each individual will be computed using the Equation 6.2:

$$Fitness\ Value = TTC - TST \tag{6.2}$$

Due, to negative values for fitness value for scenarios where TST > TTC, the number line needs to be shifted to the right, this is done using Equation 6.3.

$$Fitness\ Value = (TTC - TST) + 50 \tag{6.3}$$

This is shown in Table 6.2.

Table 6.2: TTC, TST and Fitness Value

Simulation Time / TTC	Collision	TST	Fitness Value
1240	0	412.0553	877.9447
780	1	564.8717	265.1283
770	1	645.1152	174.8848
2470	0	452.5063	2067.494
1700	0	548.5799	1201.42
710	1	528.0068	231.9932
730	1	525.4158	254.5842
1380	0	416.5212	1013.479
730	1	552.1673	227.8327
1960	0	368.9897	1641.01
830	1	582.4018	297.5982
730	1	614.5106	165.4894
770	1	577.9638	242.0362
1200	0	470.7775	779.2225
640	1	456.416	233.584

Figure 6.3 shows the result of a Monte Carlo simulation calculating the fitness value for 1000 scenarios. From the graph it can be seen for scenarios that lead to a collision (yellow dots) the TTC is much lower. Furthermore, for those scenarios that have relatively smaller TTC compared to TST have a lower fitness value. This helps test the effectiveness of the metric used to calculate the fitness function.

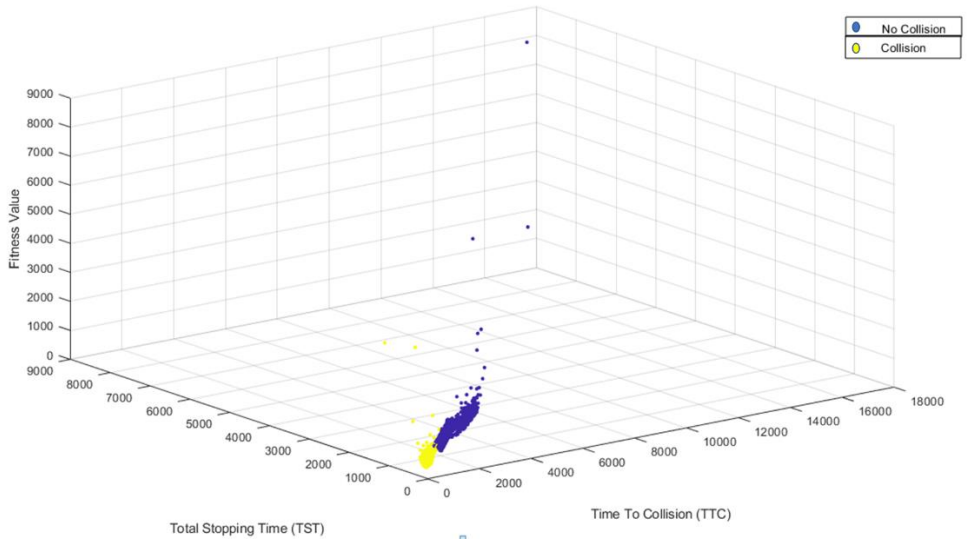


Figure 6.3: Monte Carlo Results for Fitness Function

Depending on the fitness values which again is based on the starting position of the ego vehicle and CAV as well as their speed and trajectory, individuals (in GA terms) or scenarios can be categorised into the different categories as shown in Figure 6.4. The transition of the category is shown as follow

- Category 1:

Category 1 comprises of individuals whose TTC < TST. Individuals in this category require information before hand in order to avoid a collision

- Category 2:

Category 2 comprises of individuals whose TTC is not less than TST but within a certain safety parameter. Individuals in this category could result in a collision and can move to category 1 if any parameter is changed negatively, for example, road surface conditions become icy. The individuals in this category require communication despite the possibility being able to avoid a collision narrowly.

- Category 3:

The individuals in this category TST < TTC, therefore, would not require communication in order to avoid a collision. However, if any parameter is changed negatively, the individuals in this category can move to category 2

- Category 4:

Category 4 comprises of those individuals in which no collision occurs. Since there would have been no collision that has occurred there would be no value for TTC. Therefore, the individuals in this category TTC = scenario simulation time. Thus, having a larger value for TTC. This can be seen in Table 6.2; the green highlighted individuals in Figure 6.4 are part of this category.

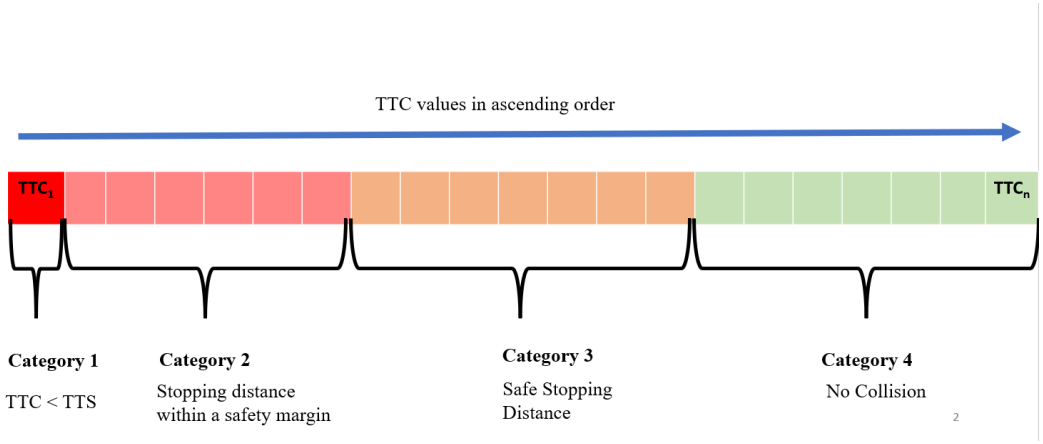


Figure 6.4: TTC category

6.6. Selection

The selection operator chooses parents from a population based on their fitness; depending on the context solutions with lower fitness values (in the current context it is individuals with lower fitness) have a higher chance of becoming parents of new solutions during reproduction. All selection functions are stochastically designed to allow for the selection of a small proportion of less fit solutions. This helps to maintain population diversity, preventing premature convergence on ineffective solutions. The most popular type of selection methods are as follows:

- Roulette Wheel Selection: Roulette Wheel Selection is a traditional selection technique that uses a fitness function to assign a fitness to each individual (chromosome) in the population. The best solution is determined by fitness level, which is used to link each chromosome's probability of selection (Pedersen, 1998).
- Elitism Selection: Initially, the best individuals from the current generation are copied, and then the evolution process continues (crossover, mutation). As a result, elitism aids in the rapid improvement of the GA's performance by preventing the loss of the best solution that was reached (Sharma & Wadhwa, 2014).

- Rank Selection: Individuals in this type are chosen based on their rank, but first the chromosomes must be sorted according to their fitness value. The best chromosome will be assigned a rank of (N), while the worst chromosome will be assigned a rank of 1.
- Tournament Selection: This type involves randomly selecting two or more individuals from a population; the fitter (individual with the best fitness) is chosen as a parent and inserted into the mating pool. Because of several characteristics, such as less time complexity, this method proved to be effective in much of the research (Oladele & Sadiku, 2013).

For this research we will be utilizing Roulette Wheel selection and Elitism. The Roulette Wheel selection method is utilized because this is a well-known selection method that has been tried and tested (Sharma, Wadhwa and Komal, 2014)and suitable for the current problem. Elitism is also used in order to maintain the best individuals in each generation.

6.6.1 Roulette Wheel Selection Methodology:

Roulette wheel selection is a type of fitness-proportionate selection that is used in Genetic Algorithms to select potentially useful recombination solutions (crossover). Candidate solutions with a better fitness value are more likely to be chosen. There is, however, a chance that they will not be. On the other hand, with fitness proportionate selection, there is a chance that some weaker solutions will survive the selection process. This is a benefit: even if an individual is weak among the entire population in one generation, it may contain some components that may prove useful in subsequent recombination processes; and through this selection rule, those potentially useful individual components may be passed down to subsequent generations. Imagine a roulette wheel with each candidate solution representing a pocket on the wheel; the size of the pockets is proportionate to the fitness as well as the likelihood of selection. Because each candidate is drawn independently, selecting N chromosomes from the population has the same probability as playing N roulette games.

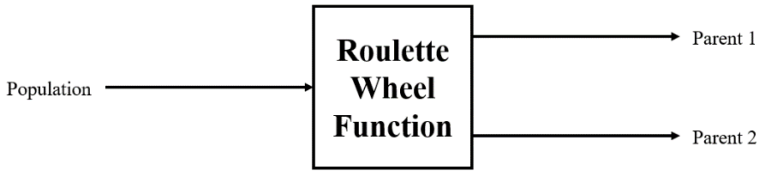


Figure 6.5: Roulette Wheel Function

Figure 6.5 shows how the how the roulette wheel function operates, it takes a population as input and selects two individuals. The next section will be discussing the Roulette wheel selection algorithm. A sample of ten individuals with their corresponding fitness values will be used to discuss this. This can be seen in Table 6.3.

Table 6.3: Individuals and Fitness Values

Individual Number	Individuals	Fitness Values
1	[60.20,44.60,10.22,2.14,0,38.74, -26.27,0]	509.7111966
2	[84.90,69.10,14.07,2.14,0,38.24, -27.53,0]	629.8069083
3	[61.60,53.70,4.32,1.81,0,37.25, -25.04,0]	489.6610621
4	[19,36.30,8.52,1.39,0,38.42, -19.70,0]	1129.030963
5	[28.20,23.70,12.30,1.75,0,37.46, -26.67,0]	969.4090159
6	[43.20,27.20,9.57,1.80,0,38.62, -18.029,0]	569.5880109
7	[41.80,41.80,5.45,2.10,0,37.75, -25.87,0]	679.5277551
8	[32.50,31.80,16.04,1.27,0,38.41, -20.14,0]	559.4938482
9	[33.90,21.50,3.67,1.64,0,37.95, -27.78,0]	1889.375883
10	[25.60,31.50,5.76,1.75,0,38.30, -20.19,0]	1319.231372

The roulette wheel selection algorithm is as follows:

- Step 1: Normalize Fitness Values:

The first step is to normalize the fitness values, this is done by using Equation 6.4.

$$Normalized\ Fitness\ Values = \frac{fitness\ values}{sum(fitness\ values)} \tag{6.4}$$

- Step 2: Sort the Normalized Fitness Values: This sort normalized fitness values in ascending order
- Step 3: Create a temporary population: Next a temporary population is created with the sorted fitness values and normalized values as shown in Table 6.4.

Table 6.4: Temporary Population

Genes	Fitness Values	Normalized Values
[61.60,53.70,4.32,1.81,0,37.25, -25.04,0]	489.6610621	0.05599431
[60.20,44.60,10.22,2.14,0,38.74, -26.27,0]	509.7111966	0.058287107
[32.50,31.80,16.04,1.27,0,38.41, -20.14,0]	559.4938482	0.063979913
[43.20,27.20,9.57,1.80,0,38.62, -18.029,0]	569.5880109	0.065134213
[84.90,69.10,14.07,2.14,0,38.24, -27.53,0]	629.8069083	0.072020437
[41.80,41.80,5.45,2.10,0,37.75, -25.87,0]	679.5277551	0.077706175
[28.20,23.70,12.30,1.75,0,37.46, -26.67,0]	969.4090159	0.110855025
[19,36.30,8.52,1.39,0,38.42, -19.70,0]	1129.030963	0.129108306
[25.60,31.50,5.76,1.75,0,38.30, -20.19,0]	1319.231372	0.150858332
[33.90,21.50,3.67,1.64,0,37.95, -27.78,0]	1889.375883	0.216056182

- Step 4: Cumulative sum: Next, the cumulative sum is for each is calculated, this is done by calculating the previous value to the current value. The process begins from column 10 to column 1. If done correctly column 1 should result in. This has been demonstrated in Figure 6.6.

```
cumsum: 1x10 double =

  Columns 1 through 9

    1.0000    0.9440    0.8857    0.8217    0.7566    0.6846    0.6069    0.4960    0.3669

  Column 10

    0.2161
```

Figure 6.6: Cumulative Sum

- Step 5: Selecting Parents_Next, a *rand* function is used to select an individual randomly.

6.6.2. Elitism

The elitism strategy reduces genetic drift by allowing the most fitting individuals among candidates for selection, the elites, to pass on their traits to the next generation (Chakraborty and Chaudhuri, 2003). In evolution, such a strategy can increase selective pressure while also speeding up convergence (Ahn and Ramakrishna, 2003). The total number of elite individuals is determined by the elitism rate (Er). Depending on the Er, the top best individuals will be passed on to the next generation. This will ensure that the best individuals are not lost after each generation.

6.7. Crossover Operator

To improve the current value of the fitness function, a new set of parameters must be chosen, as with any optimization algorithm. The crossover operator in a GA was inspired by the combination of genes in reproduction. To generate the new offspring, the strings of parameters representing the chromosomes for the two parents are cut and mixed in this operator. There are different techniques for cross over and they are as follows:

- Simulated Binary Cross Over: This was designed with respect to one point cross over property in a binary GA.
- Blend Crossover: Given two parents X_1 and X_2 the blend crossover randomly selects a child in the range: $[X_1 - \alpha(X_2 - X_1), X_2 + \alpha(X_2 - X_1)]$, with $\alpha = 0.5$ (Wen, Xia and Zhao, 2006)
- Arithmetic Crossover: Arithmetical crossover entails using arithmetic procedures to create new offspring that are sandwiched between the parents.
- Laplace Crossover: A self-adapting parent-centric crossover operator is the Laplace operator. (Ma, 2009) explain how it uses the Laplace distribution to generate offspring near parent solutions. The Laplace distribution (also known as the double exponential distribution) can be thought of as two exponential distributions joined together back-to-back (with an additional location parameter). it generates two symmetrically placed exponentially distributed offspring solutions with respect to the parents.

The current work will be using the arithmetic crossover as the search space between the two parents is explored. This is shown in Figure 6.7 (which is in section 6.8).

The equation used for the arithmetic cross over is as follows:

$$Child1_i = \beta_i \times Parent1_i + (1 - \beta_i) \times Parent2_i \quad (6.5)$$

$$Child2_i = (1 - \beta_i) \times Parent1_i + \beta_i \times Parent2_i \quad (6.6)$$

$Child1_i = \beta_i \times Parent1_i + (1 - \beta_i) \times Parent2_i Child2_i = (1 - \beta_i) \times Parent1_i + \beta_i \times Parent2_i$

Where:

β_i *:* is a random number in the interval [0,1] generated for the i-th gene

Child1$_i$: shows the i-th gene in child1

Child2$_i$: shows the i-th gene in child2

Parent1$_i$: indicates the i-th gene in Parent1

Parent2$_i$: indicates the i-th gene in Parent1

Arithmetic crossover applied to each gene:

The arithmetic crossover has been applied to each gene of the chromosome. The gene within each chromosome is shown as follows:

[Position of the Ego(x), Position of the Ego(y), Position of the Ego(z), Position of the CAV (x), Position of the CAV(y), Position of the CAV (z), Speed of the Ego, Speed of the CAV]

Due to the boundary that each chromosome can exist between, the crossover is represented as follows:

- → Ego Position: The Ego vehicle (x) position of chromosome 1 has been crossed over with the Ego vehicle (x) position of chromosome 2. This has been done for y- position and z- position.
- → CAV Position: The CAV (x) position of chromosome 1 has been crossed over with the CAV (x) position of chromosome 2. This has been done for y- position and z- position.
- → Speed: The Speed of the Ego chromosome 1 has been crossed over with the Speed of the Ego of chromosome 2, the same is done for the Speed of the CAV

Once the crossover has been performed the next is taking the individuals to the next population:

6.7.1. Recombination:

The recombination step takes the probability of crossover (Pc) into consideration. The steps for recombination are as follows:

1) Generate a random number (R1) in the range [0-1]
   a) If R1 ≤ Pc add child 1 to new population
   b) If R1 ≥ Pc add Parent 1 to new population
2) Generate a random number (R2) in the range [0-1]
   a) If R2 ≤ Pc add child 2 to new population
   b) If R2 ≥ Pc add Parent 2 to new population

This would result in the following outcomes:

- **Case 1:** R1 ≤ Pc and R2 ≤ Pc both children are transferred
- **Case 2:** R1 > Pc and R2 > Pc both parents are transferred
- **Case 3:** R1 > Pc and R2 ≤ Pc Parent 1 and child 2 are transferred
- **Case 4:** R1 ≤ Pc and R2 > Pc Child1 and parent2 are transferred

6.8. Mutation

Mutation is a genetic operator that is used to maintain genetic diversity in a population of genetic algorithm chromosomes from one generation to the next. It's comparable to biological mutation. A mutation changes the value of one or more genes in a chromosome from its original state. Some of the mutation operators are as follows:

- Uniform Mutation:
  In this operator a random gene is selected from the chromosome and uniform random value is assigned to it. For example, let x_i be the random gene selected from the chromosome within the range $[a_i, b_i]$, we then assign U (a_i, b_i) to x_i. Where U (a_i, b_i) denotes a uniform random number for the range $[a_i, b_i]$.
- Non- Uniform Mutation:
  In non-uniform mutation a random gene is selected from the chromosome and assigned a non-uniform random value.
  For example, let x_i be the random gene selected within the rang $[a_i, b_i]$, we then assign a non-uniform random value using a function.
- Boundary Mutation:
  In this operator a random gene is selected from the chromosome and assigned the upper bound or the lower bound to it.
  For example, let x_i be the random gene selected within the rang $[a_i, b_i]$, we then assign a_i or b_i to x_i. Additionally, a random variable is selected r=U (0,1) (r ranges from 0 to 1). When $r \geq 0.5$, b_i is assigned to x_i else a_i is assigned to x_i.
- Gaussian Mutation
  Gaussian Mutation utilizes the Gauss error function. The chosen gene is given a unit Gaussian distributed random value by this operator. The new gene value is clipped if it falls outside of the user-specified lower or upper bounds for that gene.

For this research will be utilizing the uniform mutation. The uniform mutation of the genes are as follows:

[Position of the Ego(x), Position of the Ego(y), Position of the Ego(z), Position of the CAV (x), Position of the CAV (y), Position of the CAV (z), Speed of the Ego, Speed of the CAV]

A random gene is selected and depending on the selected gene a random value is chosen from the upper and lower bound values of that gene. The upper and lower bounds of each gene are shown in Table 6.5.

Table 6.5: Upper and Lower Bound of Genes

	Position Ego(x)	Position Ego(y)	Position Ego(z)	Position CAV(x)	Position CAV(y)	Position CAV(z)	Speed Ego	Speed C CAV
Upper Bound	17.336	2.2	0	38.8	-17.6	0	31.29	31.29
Lower Bound	1.6323	0.9	0	37.2	-27.9	0	1	1

Note: The units for the speed are in m/s. Additionally, upper bound of the speed is dependent on the speed limit.

The mutation will only occur if the probability of mutation is within a certain range, this is known as probability of mutation (Pm)

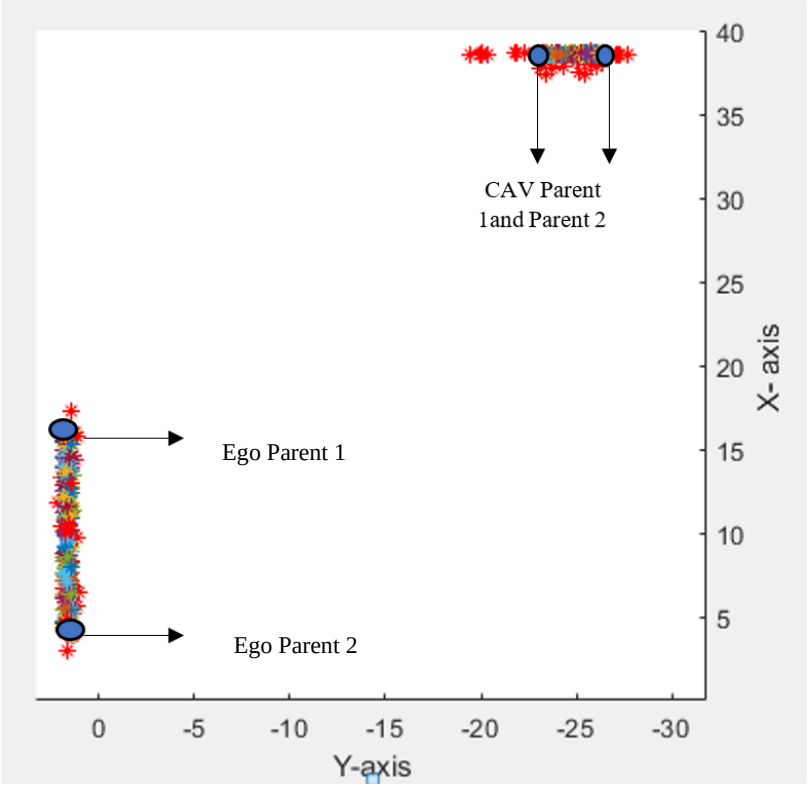


Figure 6.7: Mutation and Crossover

Figure 6.7 shows a montecarlo simulation performed for crossover and mutation with two individuals. This was done in order to determine how the local and global space has been searched. As it can be seen from Figure 11

the local space between the two individuals has been thoroughly searched with the cross-over operator. The red (*) show the mutation and from Figure 6.7 it can been mutation search the global space. Note: The Ego and CAV can exist within a certain boundary (thus confirming to NLOS requirement) and Figure 6.7 shows the mutation and cross-over remain within this boundary.

6.9. Summary

This chapter discusses step by step the genetic algorithm process to search the scenario space. The GA takes in a number of individuals which is determined by the population size. Each individual has a set of genes, these genes include the position of the ego, the position of CAV, the speed of the ego and speed of the CAV. The GA is guided to find individuals with a lower fitness value using the fitness function, which employs the TTC and TST. From the graph (A and B) in Figure 6.8, it can be seen that the genetic algorithm was searching the scenario space effectively by having a population with a lower fitness after each generation. This means the GA is effectively able to search the scenario space with scenarios with a lower TTC. The next chapter will be looking at the results output from the GA and validating these results.

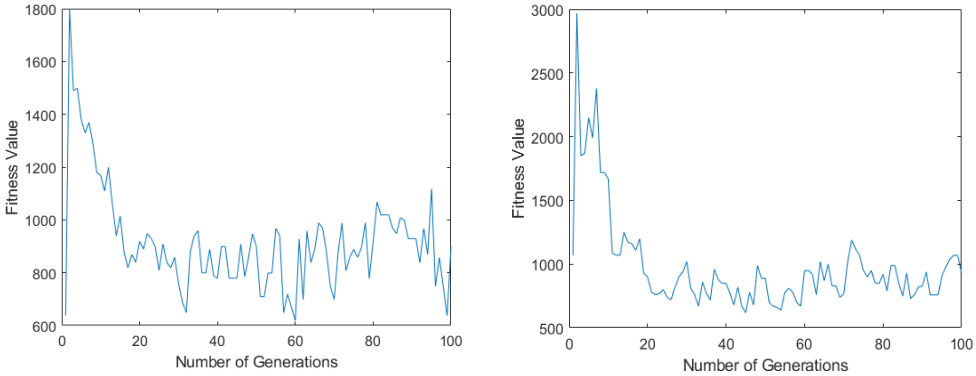


Figure 6.8: Fitness value convergence curve

Chapter 7: Simulations and Results

This chapter discusses the actual implementation for scenario generation from the functional scenarios to concrete scenarios. Chapter 3 discusses the methodology used for the development of NLOS safety critical scenarios. This has been done through the development of functional, logical and concrete scenarios. The functional scenario has been described with the use of ontologies and the logical scenarios have been defined as the domains to the attribute of each concept. For a full description please refer to Chapter 4. The functional scenario creates a large number of abstract scenarios, for example for a X-junction, D-junction, T-junction etc. Selecting one of these scenarios for example a T-Junction, logical scenarios add parameter range for each attribute, for example, number of road users, their position, speeds, number of road infrastructure etc. All possible combination of values from the parameter range, lead to a large number of concrete scenarios being generated and testing against all these scenarios would be costly and time consuming. While testing against all types of scenarios is required, the current research focuses on safety critical scenarios and hence techniques to identifying them is required. To accomplish this, GA have been used as detailed in Chapter 6. Figure 7.1 shows the scenario development from functional to concrete.

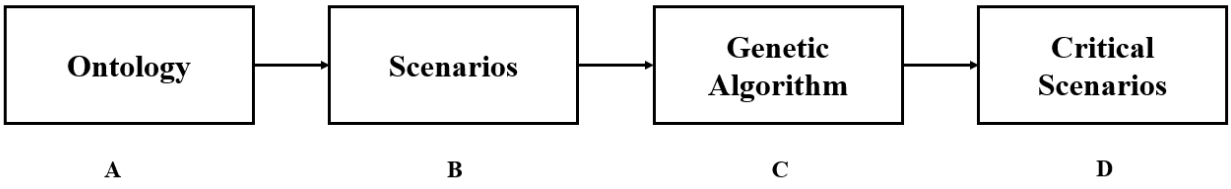


Figure 7.1: NLOS Scenario generation

This chapter goes as follows:

- The implementation and execution of ontologies to generate scenario (part A and B in Figure 7.1)
- The implementation and execution of genetic algorithm to select critical scenarios (part C and D in Figure 7.1)
- Testing and validation critical scenarios generated by the GA

7.1. Scenario Generation

This section discusses how the conceptualized ontology in chapter 4, can be used to generate scenarios practically. The conceptualized ontology in chapter 4 have been modelled in Protégé. Since this research is showing a proof of concept, a simple ontology has been developed from the conceptualized ontology in Chapter

4, this has also been developed in protégé software which uses the OWL syntax. The simple ontology includes a relevant subset of the concept (classes as known in OWL) with some of their attributes.

7.1.1. Selected underlying road layout and parameter ranges

The first step to generating scenarios is the selection of the underlying road network and roadside infrastructure together with their parameter ranges. From the list of concepts modelled within Protégé detailed in Section 7.1.2.,the concept Junction is assumed to be selected as T-Junction with two road segment *(RoadSegment)*, each road segment *(RoadSegment)* has two lane segment *(LaneSegment)*, and each lane segment has a lane marking *(LaneMarking)*. This selected underlying road layout will remain fixed in one (the only one within this thesis) set of scenario generation. The selected attributes and the value ranges for these concepts are shown in Table 7.1, 7.2 and 7.3, additionally the junction is depicted in Figure 7.2 and Figure 7.3

Table 7.1: Road Segment

Road Segment		
Id	*Road1*	*Road2*
Length (m)	*38*	*60*
Width (m)	*7.65*	*7.65*
Number of Lanes	*[1,2]*	*[2,2]*
Number of Road Infrastructure	*1*	*1*
Geometry	*Line*	*Line*
Topography	*Flat*	*Flat*
Geo Position	*[0,0,0] [38,0,0]*	*[40.2.30.0] [40.2, -29.8,0]*
Speed Limit (mph)	*[20,30,50,60,70]*	*[20,30,50,60,70]*
Road surface condition	*0.5*	*[0.5]*

Table 7.3: Lane Segment

Lane Segment				
Id	*Lane1*	*Lane2*	*Lane3*	*Lane4*
Length(m)	*7.65*	*7.65*	*7.65*	*7.65*
Width(m)	*3.75*	*3.75*	*3.75*	*3.75*
Type	*Driving*	*Driving*	*Driving*	*Driving*

Geo Position	[0,2.2,0][38,2.1,0]	*[0,-1.9,0][38,-1.7,0]*	*[38.3,30,0][38.3,-29.8,0]*	*[42.1,30,0] [42.1, -29.8,0]*
Heading(Deg)	*-4.1850e-17*	*4.1850e-17*	*-90*	*-90*

Table 7.4: Lane Marking

Lane Marking		
Id	Marking 1	Marking 1
Type	Solid	Solid
Width (m)	0.15	0.15
Space	-	-
Geo Position	*[0,0,0][38,0,0]*	*[40.2.30.0] [40.2,-29.8,0]*

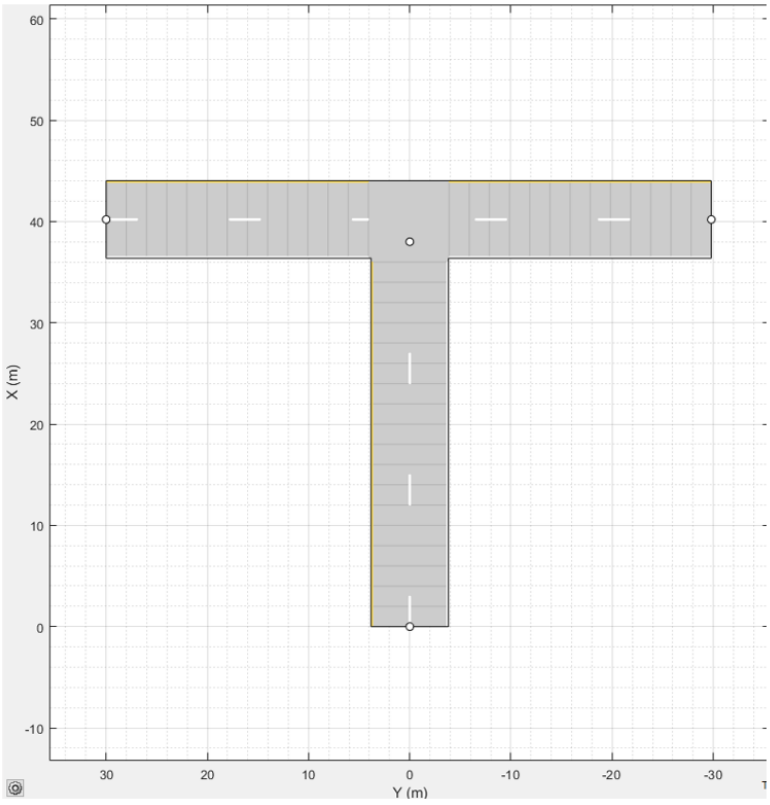


Figure 7.2: T Junction Plot

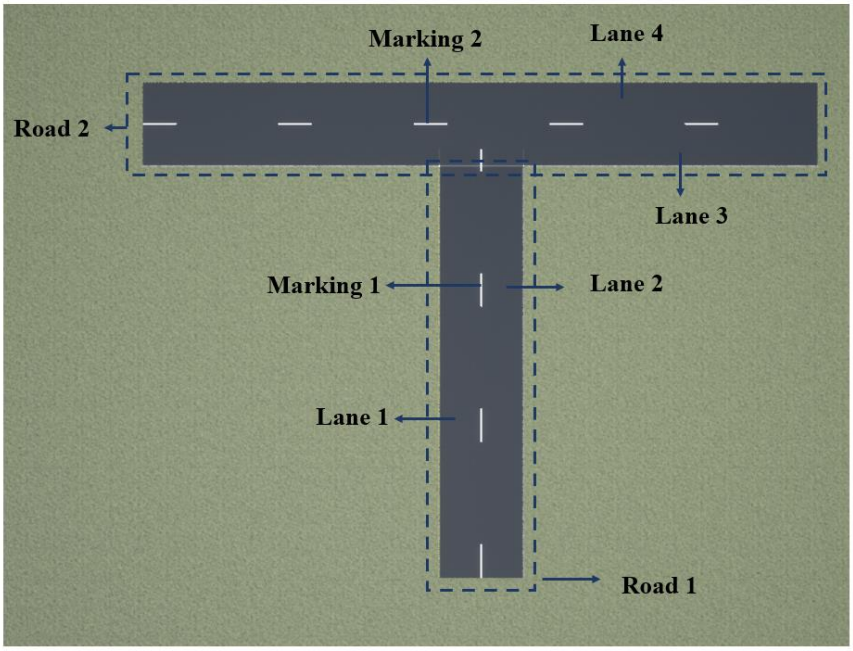


Figure 7.3: T junction in simulation environment

7.1.2. Ontologies

The concepts modelled in Protégé are as follows: scenario *(Scenario)*, scene *(Scene)*, junction *(Junction)*, road infrastructure *(RoadInfrastructure)* road user *(RoadUser)*, sensors *(Sensors)*, OBU *(OBU)*, communication bit *(communicationBit)*, object type *(ObjectType)*, communication network *(CommunicationNetwork)*, vehicle broadcasting group *(VehicleBroadcastingGroup)*. Due to the conversion of concept (classes) from UML to OWL additional concept (classes) need to be included within to comply with the OWL logic. These include position *(Position)*, waypoints *(Waypoints)*, location *(Location)* and time *(Time)*. These additional concepts allow relation which will help with scenario generation. For example, Figure 7.4 shows the concept (class) and their relations, so the concept (class), junction *(Junction)* has a relation with location *(Location)*, position *(Position)* and road user *(RoadUser)*. And the class concept location *(Location)* has relation with the concept has road user *(RoadUser)*, position *(Position)* and waypoints *(Waypoints)*. Through this relation the following inferences are made: a junction has location, and each location has a position and waypoint (note waypoints have a position), therefore a road user can have a location with a position and a set waypoints within a junction.

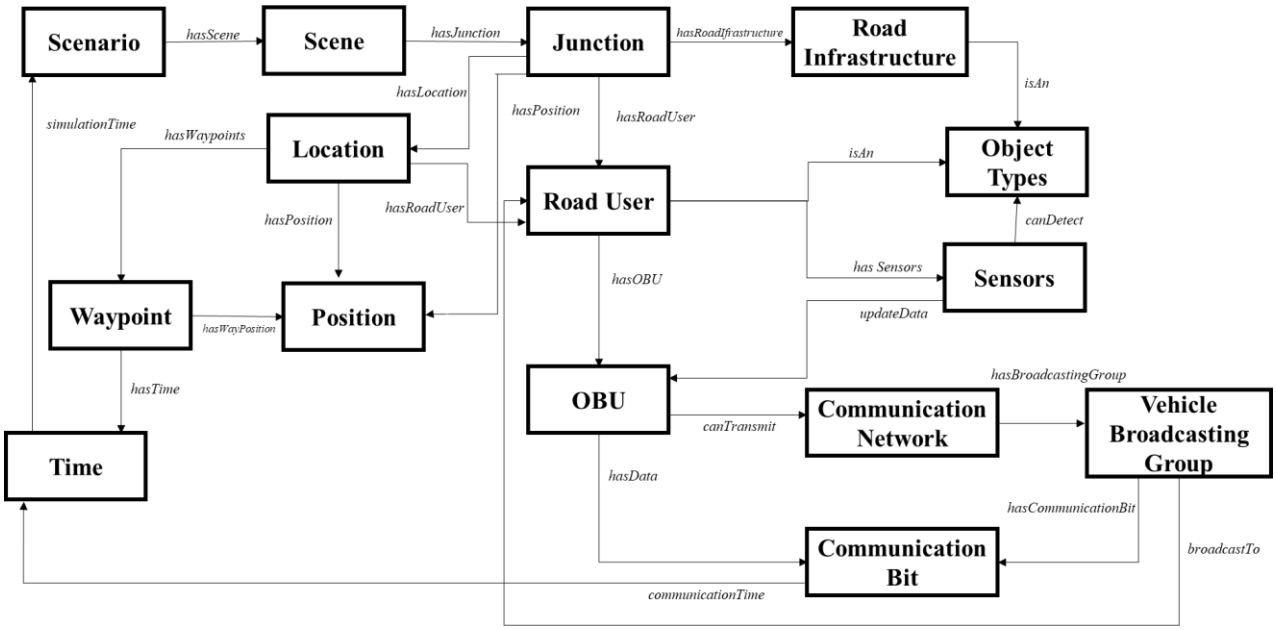


Figure 7.4: NLOS Ontology Structure

The conceptualised ontology as shown in Figure 7.4 was modelled in protégé. This is illustrated in Figure 7.5

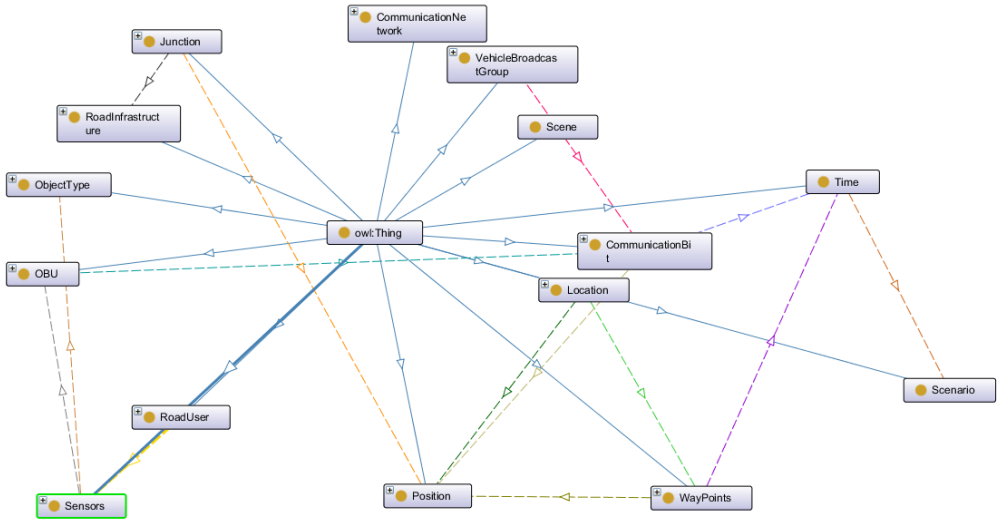


Figure 7.5: Protégé model

7.1.3. OWL API

The (partly modelled for the purpose of this research) ontology saved in the format RDF/XML and imported to OWL API. OWL API is a java library to interact with OWL. The OWL API is a Java API and reference implementation for creating, manipulating, and serializing OWL ontologies. This research will be using OWL API to generate scenarios. The first step is to load the ontology into OWL API. Then a reasoner is used to validate the ontology. A semantic reasoner is used to find logical inconsistencies, provide information to improve and validate the concept model. The reasoner used in this research is the HermiT reasoner (*The role of ontologies and reasoners*, 2021). This reasoner is written using the web ontology language (OWL). Given an OWL file, HermiT can identify subsumption relationships between classes and any logical inconsistency. Once the ontology has successfully passed the reasoner and the instances for each concept are defined. Instances within protégé are the domain to each attribute, an instance is single value within the domain. OWL API is used along with java to access the modelled ontology and use the relation and instances to create scenarios. A loop is used to perform scenario generation this process links the relation and generates different type of scenarios. Figure 7.6 shows a snippet of the output.


```
{
"Scenario" : {
        "Junction" : {
                "hasName" : "London_Road_TJunction" ,
                "hasCentralCordinates" : "P2_1" ,
                "hasVehicleLocation" : {
                        "LocationD": {
                                "hasVehicle": "EgoVehicle",
                                "hasPosition": "P4_1",
                                "hasWayPoints": {
                                        "positions": [ (38.7, 21, 0 ), (38.52, 28.58, 0 ), (38.7, 14.7, 0 ), (38.8, 8.9, 0 ), (38.97, 1.93, 0 ), (37.9, -1.2, 0 ), (35.71, -1.87, 0 ), (23.6, -2.3, 0
                                },
                        },
                        "LocationD": {
                                "hasVehicle": "ConnectedAV",
                                "hasPosition": "P4_1",
                                "hasWayPoints": {
                                        "positions": [ (38.7, 21, 0 ), (38.52, 28.58, 0 ), (38.7, 14.7, 0 ), (38.8, 8.9, 0 ), (38.97, 1.93, 0 ), (37.9, -1.2, 0 ), (35.71, -1.87, 0 ), (23.6, -2.3, 0
                                },
                        },
                },
                "Scenes" : {
                        "Scene0" : {
                                "EgoVehicle" : {
                                        "positions":  [(42.04, -27.71, 0 ), (42, -23.2, 0 ), (42.3, -11.6, 0 ), (42.1, -6.3, 0 ), (40.1, -3.1, 0 ), (35.8, -1.3, 0 ), (29.4, -1, 0 ), (19.4, -1.3, 0 )
                                        "hasSpeed": "77.3",
                                        "hasVehicleBroadcastGroup" : "VBG1",
                                        "broadCastTime" : "t0",
                                },
                                "ConnectedAV" : {
                                        "positions":  [(42.4, -26.7, 0 ), (42.3, -16.1, 0 ), (42.4, -10.8, 0 ), (42.1, -3.1, 0 ), (40.7, -0.4, 0 ), (38.4, 1.3, 0 ), (34.5, 2.1, 0 ), (24.87, 2.29, 0
                                        "hasSpeed": "93.5",
                                        "hasVehicleBroadcastGroup" : "VBG2",
                                        "broadCastTime" : "t0",
                                },
                        },
                        "Scene1" : {
                                "EgoVehicle" : {
                                        "positions":  [(42, -23.2, 0 ), (42.3, -11.6, 0 ), (42.1, -6.3, 0 ), (40.1, -3.1, 0 ), (35.8, -1.3, 0 ), (29.4, -1, 0 ), (19.4, -1.3, 0 ), (12.3, -1.5, 0 ), (
                                        "hasSpeed": "6.6",
                                        "hasVehicleBroadcastGroup" : "VBG1",
                                        "broadCastTime" : "t1",
                                },
```


Figure 7.6: Snippet of Scenario generation

Once the scenarios are generated, they are published in JSON file which is imported to the simulation environment MATLAB. To import into the GA, the initial position of both road user will be used, and the position will be varied of the two-road user within the NLOS limit as well as their speed to determine the critical scenario. A Monte-Carlo simulation was run to generate about 5000 scenarios as shown in Figure 7.7 to provide an indicative representative of the possible scenarios that can be generated starting from the NLOS ontology. It

must be noted that the actual scenario space is much larger than that acquired from the Mote-Carlo simulation. The blue lines represent scenarios that did not lead to a collision and the red line represent scenarios that did lead to a collision. From this sample scenario space, the GA selects scenarios that are critical as the initial population.

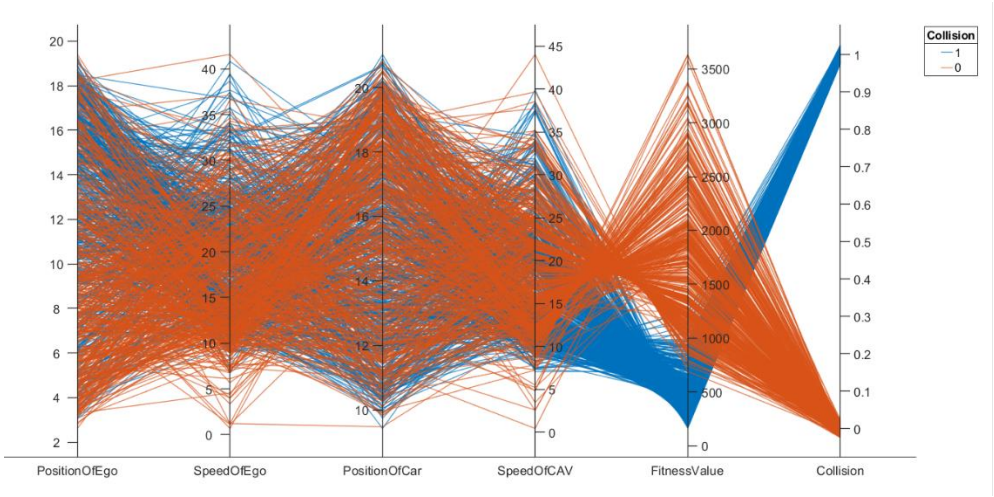


Figure 7.7: Sample Scenario Space from Monte-Carlo Simulation

7.2. Critical Scenarios

7.2.1. Importing to MATLAB

The scenarios generated in the JSON file is imported to the MATLAB environment. For the purpose of this work, the design and simulation environment (Figure 7.10) and vehicle dynamics (Figure 7.11) have been implement using the built-in libraries, models, and algorithms available within MATLAB (Automated Driving Toolbox) and Simulink. Most concepts and their attribute can be easily translated as their structures comply with the automated driving toolbox environment. These include road segment, lane segment, lane marking and road user. However, some concepts for example building weren't as simple, since the MATLAB toolbox had two road infrastructure which included a jersey barrier and rail. A jersey barrier was used and positioned within the T-junction as depicted in Figure 7.8.

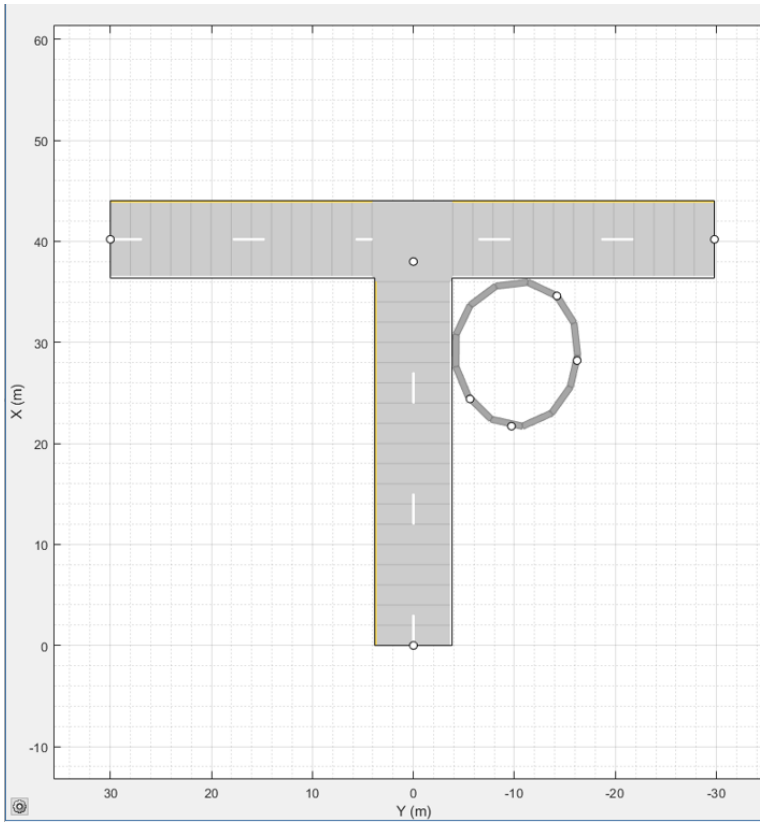
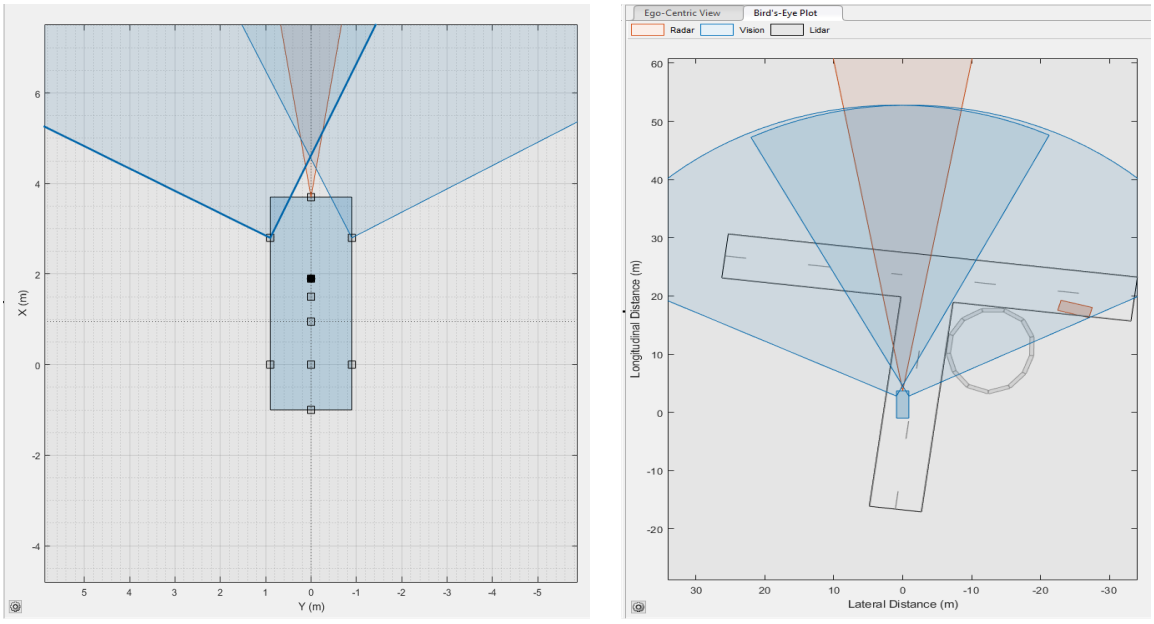


Figure 7.8: Road Structure with jersey barrier

For the concept object type, the MATLAB toolbox environment has a default method of classifying them which has not been changed. The sensors have been placed on the ego vehicle to represent a system under test as shown in Figure 7.9 (A). Figure 7.9 (B) shows the field of view of the sensor. We assume there are no error within the sensors and if the sensor is unable to detect is due to the geometry or obstacle (building) in its path.



A B

Figure 7.9: Ego model with sensors

A Simulink model for the simulation environment and vehicle dynamics are shown in Figure 10 and 11 respectively. The vehicle model used is a bicycle model as shown in Figure 7.11. Such a model has been widely adopted in many studies such as academia and industry (Kong *et al.*, 2015)(Paden *et al.*, 2016)

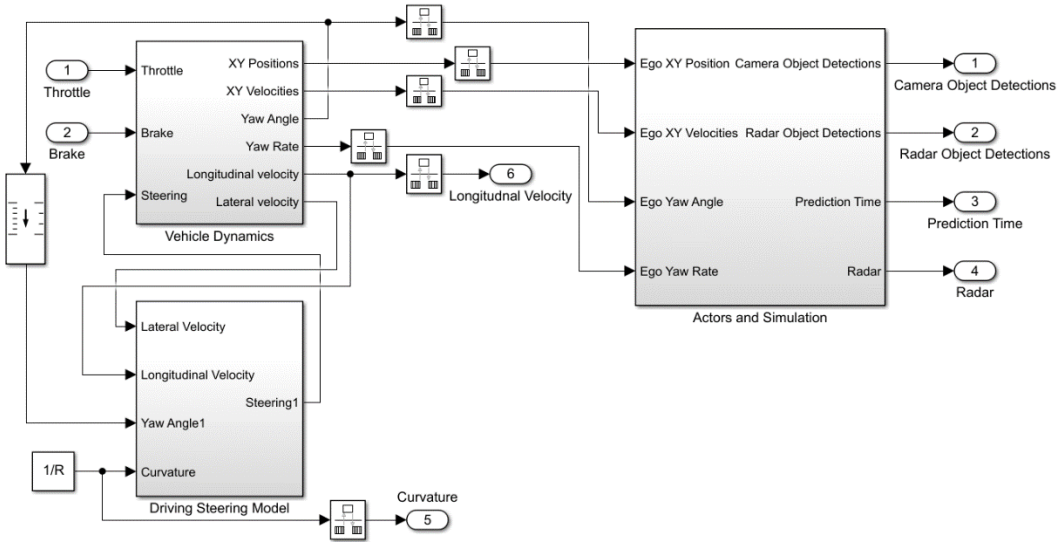


Figure 7.10: MATLAB/Simulink model of a vehicle and environment

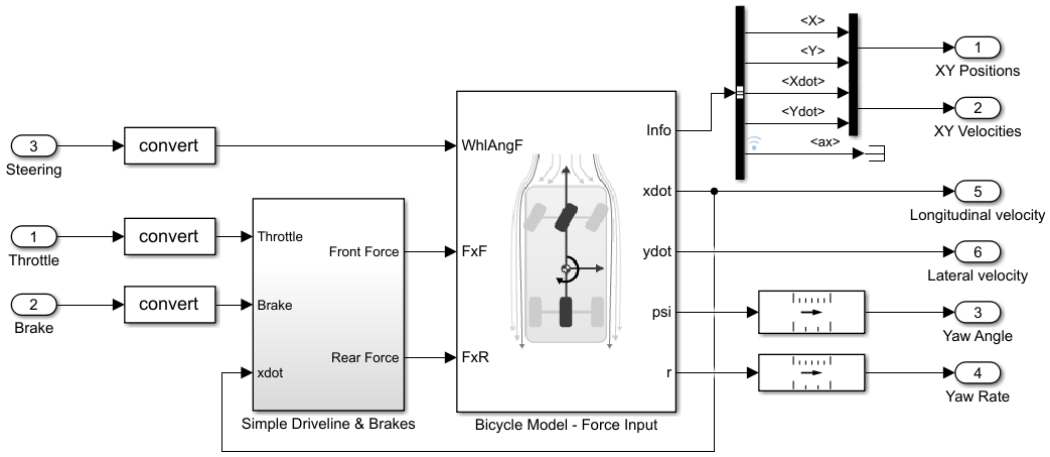
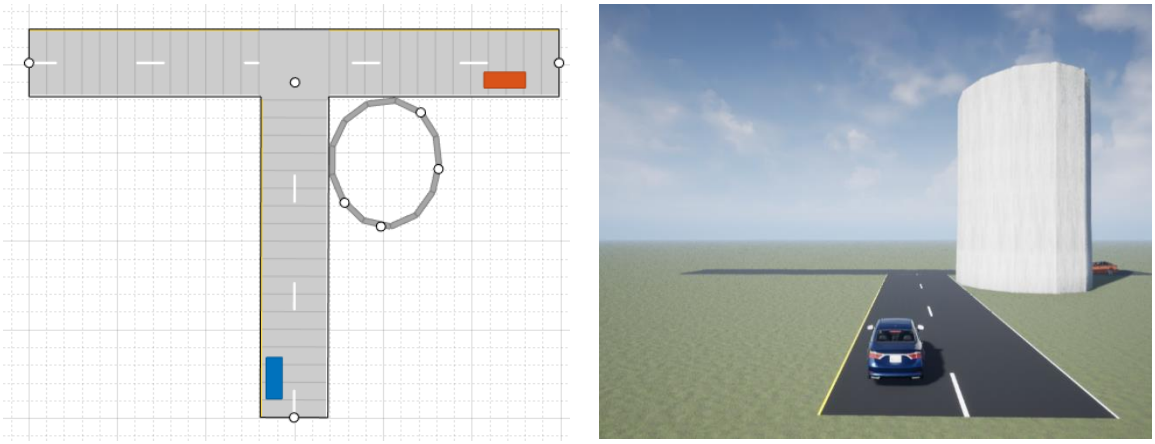


Figure 7.11: MATLAB/Simulink vehicle dynamics model

A simulation-based approach was used to compute the TTC this was done using the Automated Driving Toolbox and MALAB-based function. This is advantageous as this can be easily implemented for any scenario with any underlying road network with no or little change to the function to determine the TTC. Given that it takes into account the dynamics of the surrounding environment and the participants in a scenario, this is preferable to mathematical calculation.

For the communication network this research will be using dedicated short-range communication (DSRC) to communicate between the vehicles. The information sent between the vehicles include the unique identification, current time, position, heading and speed (Ali, 2019) . The communication network does not currently exist within the automated driving toolbox, therefore, to depict this, a simple function has been written to communicate messages at regular intervals when the vehicles are within the communication range. DSRC has a communication range of 300m (Knowles Flanagan, He and Peng, 2019), however, for the purpose of this work we assume the maximum range is limited to 30 m due to environmental conditions (for e.g. noise, interference and geo graphical environment between transmitting and receiving terminals) (Bae *et al.*, 2021). It must be noted here that aspects of the communication such as latency, maximum range of communication and other protocols can be tested using the generated scenarios to validate for example, how suitable a particular communication technology is to mitigate arising critical scenarios. The concept scene and scenario have been used to model the trajectory of the vehicle as well as the simulation time. The scenario layout is depicted in Figure 7.12



A B

Figure 7.12: Scenario Layout

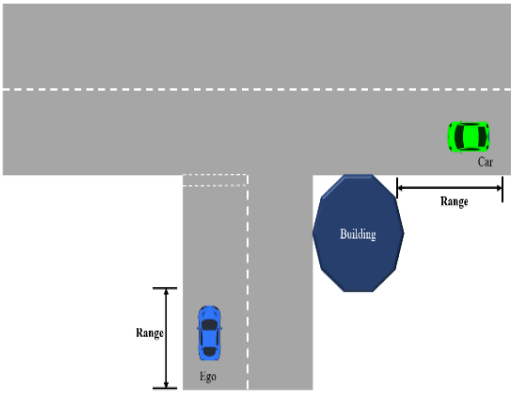
7.2.2. Genetic Algorithms

Once the scenario layout has been imported and modelled. a genetic algorithm (GA) is used to search the scenario space. To guide the GA, a fitness value is assigned to each scenario using the metrics, TTC and TST. The fitness function is outlined in chapter in Chapter 6, section 6.5. The lower a fitness value, the more critical a scenario and the more likely a scenario is taken to the next generation. The entire implementation of the GA was done within the MATLAB environment utilising the automated driving toolbox. The automated driving toolbox environment was used within the GA to determine the TTC of each scenario. The GA will take a set of chromosomes as an input which is known as the population size. Each chromosome have a set of eight genes. This is represented as follows:

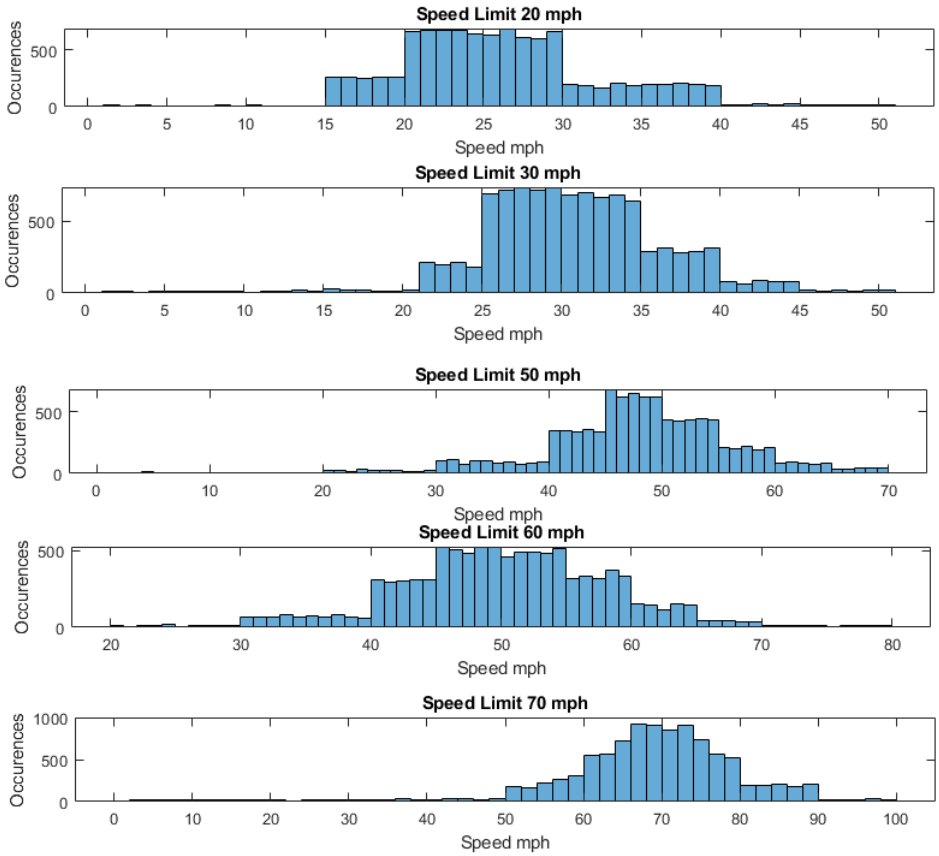
[Speed of the Ego, Speed of the CAV, Position of Ego, Position of CAV]

The position of the ego and CAV each contain three genes to represent the [x,y,z] position. The position of the ego and CAV are ranged within the NLOS ranges as indicated in Figure 7.13(A). The range values for this position have been outlined in Table 6.5. The speed values have been acquired from a speed distribution as shown in Figure 7.13(B). The parameters used to run the GA are as follows:

- **Population Size:** 50
- **Number of Generation:** 100
- **Crossover Rate:** 0.85
- **Mutation Rate:** 0.1
- **Elitism Rate:** 0.2



A



B

Figure 7.13: Input Scenario

To conduct, Experiment 1, the GA was executed multiple times (greater than 100) for the selected T-junction Road structure with road segments having a different combination of speed limits. For each speed limit

combination, the GA outputs a slightly different optimum scenario i.e., a T-junction with the chosen speed limit of the road segment, with the GA selecting a certain value of road user positions and speeds (corresponding to the speed limit) that leads to a critical scenario. As part of Experiment 1, for each speed-limit combination (within the U.K context which are in mph and are as follows: [20,30,50,60,70]), the optimum chromosome is outputted after maximum generation has been reached. The GA is executed several times with different initial population sets at each run to acquire a set of optimum scenarios. A sample of the results are shown in Table 7.4.

Table 7.4: Genetic Algorithm Simulation Results (Experiment 1)

U.K Speed Limit (mph)	Scenario No.	Best Chromosome				Fitness Value
		Ego Speed (m/s)	CAV Speed (m/s)	Ego Initial Position [x,y,z]	CAV Initial Position [x,y,z]	
20	1	17.1216	16.6746	[13.9123 ,0.9566 ,0]	[37.2579,-24.120 ,0]	264.4975
	2	20.3850	17.1663	[10.8570 ,1.4168 ,0]	[38.3650,-22.8102,0]	261.9352
	3	15.8699	17.5240	[13.9682 ,1.6926 ,0]	[38.3652,-26.8626,0]	248.927
	4	11.3101	14.3053	[15.2373 1.2462 0]	[38.1183 -26.7917 0]	247.7065
	5	16.9875	13.8135	[14.5902 ,1.3325 ,0]	[38.3544,-21.8706,0]	282.012
30	6	17.6581	16.1381	[13.9372 ,1.9566 ,0]	[37.6517,-25.8287,0]	226.7168
	7	13.4365	3.6227	[13.9145 1.8202 0]	[38.5678 -26.9920 0]	227.4729
	8	19.3568	18.4180	[15.1747 ,2.1254 ,0]	[38.3473,-22.8737,0]	291.2602
	9	15.8401	13.8679	15.4586 1.6820 0	[38.2426 -24.9960 0]	250.674
	10	13.0830	15.0444	[16.7418 1.7627 0]	[37.8074 -23.8606 0]	194.2535
50	11	28.3870	24.0955	[15.1085 ,1.3212 ,0]	[37.3036,-24.7745,0]	137.2221
	12	25.8525	24.2067	[14.6735 1.5998 0]	[38.5406 -25.5059 0]	151.7116
	13	24.1807	28.6790	[15.2795 ,1.5112 ,0]	[38.4594,-25.8000,0]	158.0999
	14	27.9594	24.1532	[15.4182 1.6639 0]	[38.0090 -22.3574 0]	147.2162
	15	25.0342	22.3073	[11.9768 1.9588 0]	[38.6151 -27.0784 0]	142.7278
60	16	39.2580	39.1673	[14.0173 1.4701 0]	[37.8618 -22.9792 0]	111.7926
	17	24.7473	27.1098	[16.4642 ,2.0535 ,0]	[37.5853,-27.4239,0]	130.46
	18	25.6746	24.0203	[16.6797 ,1.5251 ,0]	[38.0375,-26.4105,0]	138.1223
	19	35.5844	32.8574	[16.8729 1.1449 0]	[37.3043 -23.1958 0]	107.8794
	20	28.3003	23.8816	[14.2782 ,2.0787 ,0]	[37.6071,-25.5958,0]	137.0851
70	21	37.3725	34.6903	[12.1313 ,2.1444 ,0]	[38.6101,-27.5995,0]	107.4706
	22	20.1189	22.1138	[15.1336 1.5602 0]	[38.1173 -27.6611 0]	178.8526
	23	35.1820	34.1092	[15.8508 ,1.6245 ,0]	[38.6951,-27.3866,0]	173.3251
	24	19.5267	22.9965	[17.1321 1.8705 0]	[38.7331 -26.5862 0]	175.1897
	25	21.7708	20.9215	[15.3727 1.1598 0]	[38.5685 -26.9392 0]	165.6935

Table 7.4, hold the record of a sample of 25 experiments (the best 25 of all the GA runs), each resulting to a best chromosome selected with its corresponding fitness value. Next, a lower bound test is conducted to ensure all scenarios selected by the GA do lead to a collision. To conduct this lower bound test, the chromosome is inputted to the simulation environment, where through simulation it is determined whether a collision will occur or not. All scenarios within Table 7.4 leads to a collision. The scenario crash validates that none of outputted scenario lay within scenario category 4 (as discussed) in chapter 6. These simulations within this section

assumed that the ego vehicle did not react, even when the sensors detected the other vehicle (of course detection range is still satisfying the criticality scenario) and hence the term lower bound tests has been used to depict tests in the most pessimistic layout.

However, the Ego vehicle will react depending upon its software algorithms (for example, collision avoidance algorithm, trajectory planning etc) and control systems. A straightforward braking has been applied as the next step or Experiment 2 within this research (assuming negligible algorithm processing and reaction time) at the point when the ego vehicle "sees" the other road user. Based on where the ego vehicle comes to a stop and the distance from the collision point (as identified in Experiment 1 as the location where the two trajectories intersect) is then used to categorise the criticality of the scenario in the next section. This no-reaction simulated results (all leading to crashes) and instant hard braking simulated results (leading to ego vehicle stopping at a small distance from the crash point) provides the lower and upper (most pessimistic and most optimistic) benchmarks of the scenarios generated respectively. The hard-braking simulation also considers road and environment conditions to be optimum, meaning that the road surface and weather condition allows for the immediate stopping condition (which in reality may be much larger). Figure 7.14 shows the fitness value spread and the spread restricted between 100 and 300 may indicate that it was only local optimum that was reached and not global optimum. Future work will explore tweaking the GA algorithm to achieve a wider spread i.e., even lower fitness function.

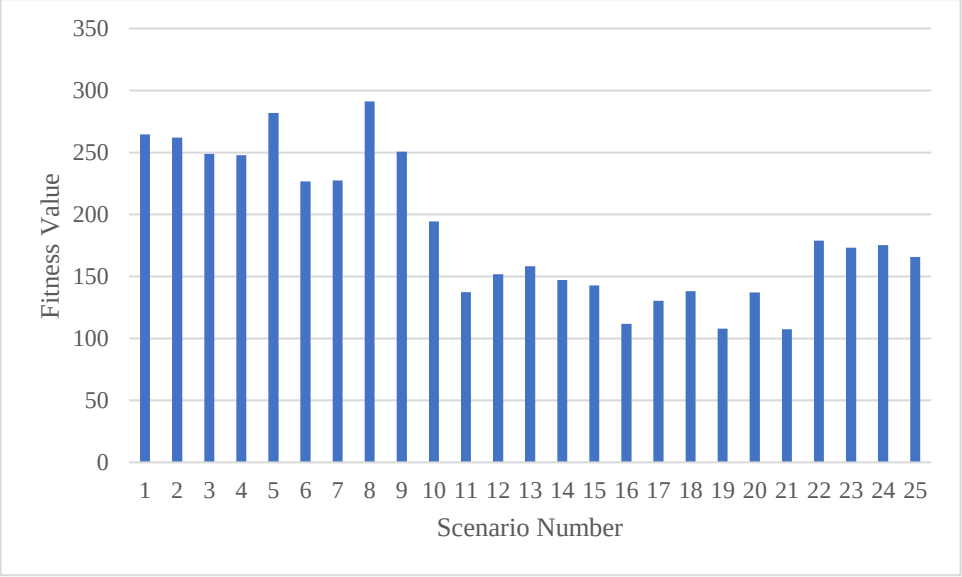


Figure 7.14: Critical scenarios against fitness value

7.3. Scenario Categorisation

As discussed in Chapter 6, a scenario can be categorised within category 1, category 2, category 3, and category 4. These are outlined as follows:

- Category 1: A scenario where the ego vehicle is unable to detect the oncoming vehicle in enough time and leads to a collision, where TTC < TTS
- Category 2: In this scenario the ego vehicle manages to avoid a collision, however, only within a certain safety margin (which has been assumed to be 5m), however if any parameter like road surface condition or system reaction was to change the scenario could lead into a collision.
- Category 3: A scenario where the ego vehicle is able to detect the oncoming dynamic object in enough time to brake at a safe distance.
- Category 4: The speed and position of the vehicles do not lead to a collision.

In the last section (Section 7.2) as part of Experiment 1, it was validated that all scenarios identified by the GA were in the first three categories and none of the output scenarios are within category 4. Next, we need to determine in which of the other category (i.e., 1, 2 or 3) the outputted scenarios reside. To determine this the simulation environment of Experiment 2 was set up. Experiment 2 differs from Experiment 1 by including braking as soon as the inbuilt sensors detect the oncoming obstacle. Thus, the simulation environment takes in an input from the outputted GA results which comprise of the initial position of the ego, initial speed of the ego, initial position of the CAV and the initial speed of the CAV. The ego vehicle is equipped with sensors and when it detects the CAV, braking is initiated. Furthermore, another simulation environment is set up for Experiment 3, which incorporates vehicular communication, this sends information from the CAV to the ego vehicle. The information includes its current time, position, speed and heading (Ali, 2019). This simulation environment is set-up to determine how early detection has an effect on distance from collision point. The sample results from Experiment 1(Table 7.4), when inputted into both simulation environment (a scenario with no communication i.e., Experiment 2 and a scenario with communication i.e., Experiment 3). Table 7.5 shows the stopping distance from the collision point without communication and with communication.

Table 7.5: Simulation Results (Experiment 2 and 3)

Scenario Number	Distance to collision in meters without communication	Distance to collision in meters with communication
1	2.5238	17.8781
2	2.4177	17.6315
3	2.955	16.641
4	1.3852	14.3623
5	2.5198	18.1309
6	3.6387	17.5777
7	5.2369	18.8516

8	4.7573	17.3119
9	5.2107	18.9167
10	3.2336	17.2777
11	9.2491	23.2523
12	3.0005	16.318
13	2.7581	14.6636
14	3.9501	17.915
15	4.0123	18.1029
16	5.1561	20.3304
17	7.5509	19.4814
18	6.5971	17.6146
19	5.4206	18.1526
20	6.0151	18.3675
21	8.7142	20.8467
22	7.385	20.74281
23	2.5965	14.4329
24	4.1371	17.0601
25	5.6058	21.4896

From Table 7.5, it can be observed that the distance from the point where the trajectories of the ego vehicle and CAV intersect (otherwise known as the collision point) is much more when communication is introduced or when the ego vehicle is aware of the obstacle much earlier. Looking into the results of Table 7.5 under column 1 more minutely, in scenario number 4 the ego vehicle was only 1.3852 m away from the collision point when it came to a stop on braking initiation. Similarly, several other scenarios have less than 3 m distance from the collision point. This distance is the distance where the ego vehicle comes to a complete halt upon initiating braking from the point the sensors detect the oncoming CAV. This is in the most optimistic road and weather condition. The results from Table 7.5, have been illustrated in Figure 7.15.

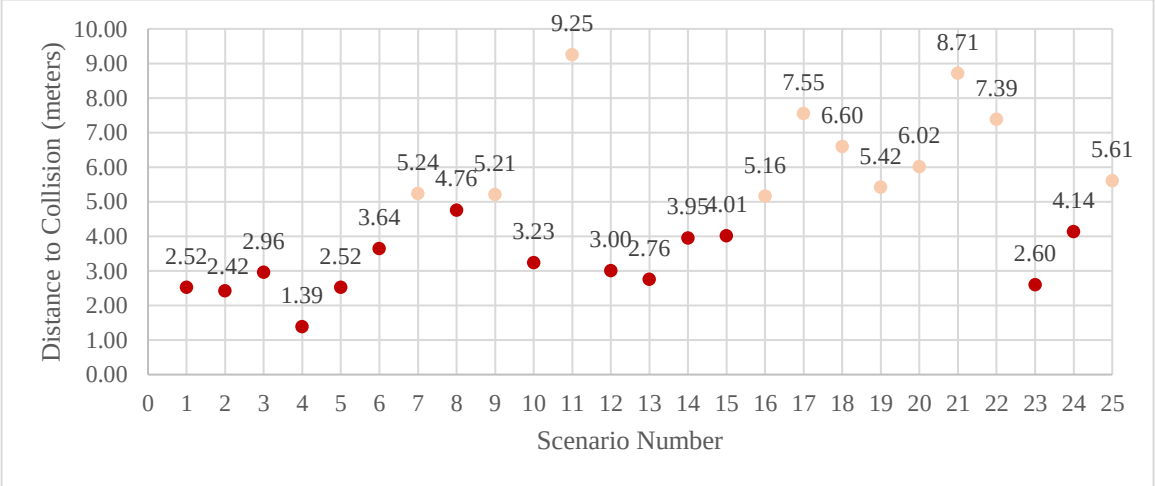


Figure 7.15: Scenario Categories (Orange dots indicate category 3 scenarios and Red dots category)

From Figure 7.16, it can be observed that a large number of scenarios have a distance of less than 5 m (category 2) with many coming under 3 m. These scenarios (illustrated with red dots in Figure 7.15) are scenario which are within category 2. From the results it can be seen that the scenarios outputted from the GA, fit within the definition of critical scenarios as discussed in Chapter 5. This definition for critical scenarios as discussed in Chapter 5 is as follows: A scenario is considered critical if TTC < TST (category 1) or within a safety value (Category 2). The safety value is when the stopping distance is within a small distance from the collision point, which has been assumed to be 5m. If there was a change in weather condition, which would lead to a much a larger braking distance resulting to a collision. Furthermore, if the system performance or reaction time was to change as suggested by Hammerschmidt et. al. (Hammerschmidt, 2019), an autonomous vehicle could have a reaction time of up to 0.5s, with these parameters all the scenarios from category 2, will be within category 1 and all scenarios within category 3 will be either within category 1 or within the upper or lower limits of category 2, but very close to being category 1. Figure 7.16, compares these results with communication to determine the braking distance if communication was received before hand

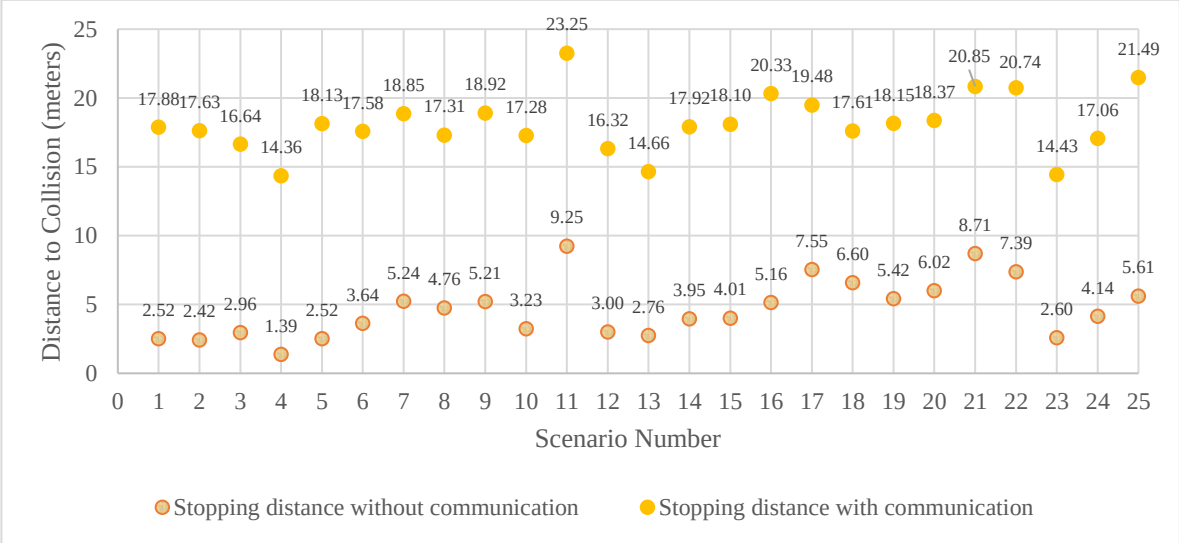


Figure 7.16: Comparison of distance from collision point

Figure 7.16 shows a comparison of the results of stopping within distance within the NLOS scenario which compares the stopping distance with communication (yellow) and stopping distance without communication (orange). From the figure it can be observed most of the critical scenarios acquired from the GA have a stopping distance less than 5m, with a few a little distance over 5 meters and an outlier at just under 10. These scenarios are critical scenarios as discussed earlier with any change in parameter leading to a collision. Comparing this, with results acquired with a communication network, it can be observed that the stopping

distance from the collision point for most for the scenario has been over 15 m. This evaluates the need for communication beforehand in order to avoid a collision. From this it is clearly seen that the scenarios are also capable of being used as test scenarios for any new development within the vehicular communication area apart from in vehicular algorithms.

7.4. Summary

This chapter discusses the implementation of the proposed new end-to-end framework as illustrated in Figure 7.17 and summarised through the consecutive steps:

- Step1: Knowledge is collected from knowledge sources including Government data sets, road traffic regulations, road layout norms, existing knowledge sources, expert knowledge, and scientific literature.
- Step 2: This knowledge is represented using ontology and organised in layers. This ontology has been modelled in Protégé. This has been discussed in chapter 7.1.2.
- Step 3: The ontology is imported to OWL API, which in java where logical scenario is defined, and NLOS scenarios generation occurs. This scenario is outputted to a JSON file format. This has been discussed in section 7.1.3.
- Step 4: The JSON file is imported to MATLAB and the NLOS environment is created utilising the automated driving toolbox and this has been discussed in section 7.2.1.
- Step 5: This NLOS scenario has a large scenario space with many possible concrete scenarios. This has been depicted in Figure 7.7.
- Step 6: A genetic algorithm is used to search the scenario space and output one of the possible critical scenarios. Many runs of the GA is conducted to provide us with several critical scenarios. Discussed further in section 7.2.2.
- Step 7: Experiment 1, was conducted to ensure the selected scenarios are all critical, meaning all lead to possible collisions. Experiment 1 is considered to be in the most pessimistic condition as the ego vehicle does not react at all upon detecting the approaching obstacle. This has been further discussed in Section 7.2.2.
- Step 8: Experiment 2, was conducted which repeats Experiment 1 but this time with the ego vehicle capable of reacting instantaneously upon obstacle detection. This experiment is also used to categorise the scenario. This has been further discussed in Section 7.22
- Step 9: Experiment 3, was conducted which validates the need for communication in a NLOS scenario. This has been further discussed in section 7.3

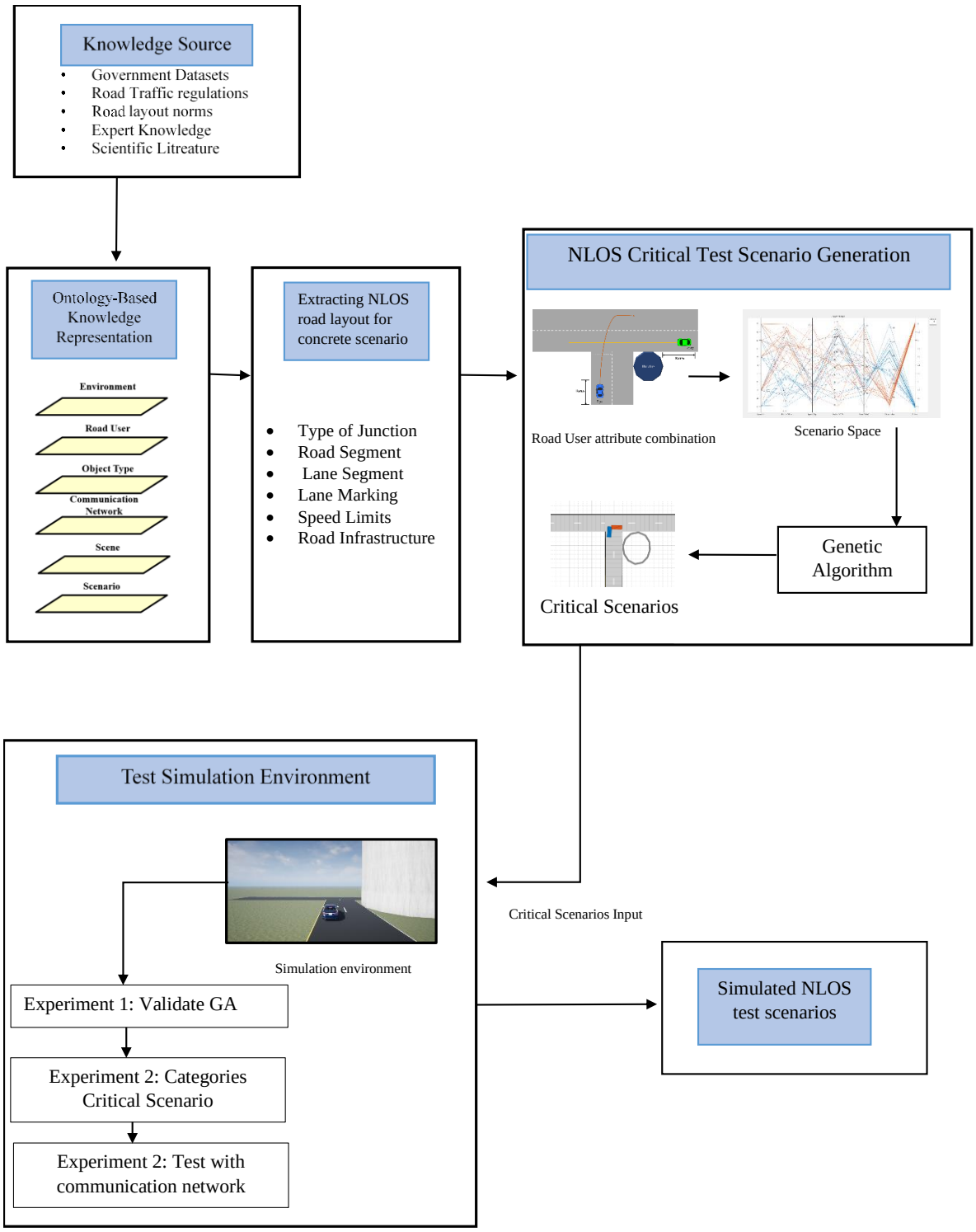


Figure 7.17: NLOS scenario generation framework

Chapter 8: Conclusion

8.1. Overview

The presented thesis has put forward a novel end-to-end, NLOS, safety critical test scenario generation framework, for testing the functionalities of CAV when presented with such potentially risky situations.

The developed framework primarily addresses a critical gap in the current literature, as, to the best of the authors knowledge there is no existing solution that facilitates simulation of NLOS test scenarios alongside incorporating the digital layer for communication. In addition, the thesis also addresses the need of an effective and inexpensive method to identify test scenarios with certain characteristics (critical scenarios in the current context) through the development and validation of a Genetic Algorithm approach to scenario space searching. The identified scenarios have finally been represented on a simulated platform on MATLAB to both validate the proposed approach as well as to provide a starting point or foundation for a complete simulated testing environment to be developed in the future. The scenarios developed as proof-of-concept are basic, but this is not a limitation as every component has been developed in a way to allow for easy scaling and expansion. The proposed framework alongside its components, and hence the unique contributions of this piece of work has effectively been evaluated throughout this thesis. This has been achieved as follows:

Chapter 1, identifies the crucial gap in the current literature and puts forward a plan for addressing the gap, alongside highlighting the new and unique contributions made within this thesis for knowledge advancement. Chapter 2 provides a thorough literature review of practice and theory as well as standards, in order to lay the foundation from where knowledge contributions begins in the context of this research. Chapter 3 provides the content that fulfils the overarching aim of the thesis and addresses contribution 1 through the discussions of the entire framework and workflow methodology of how these scenarios are generated from functional, logical to concrete scenarios. Chapter 4, the first component of the framework, delves in detail the knowledge representation of scenarios through ontologies in the context of the United Kingdom regulations and acknowledged vehicular communication standards. The ontologies have been validated through Hermit reasoner and scenario generation. Chapter 4 includes contribution 2. Chapter 5 details the contributions made in the context of defining and stating the identification criteria of critical scenarios. Following which Chapter 6, puts forward the GA approach to critical scenario identification through its fitness function defining the criticality of a scenario (or in other words an individual within the GA structure). Chapter 5 and 6 address and include contribution 3. Chapter 7 discusses the implementation of the proposed workflow as a proof of concept whereby the identified critical scenarios and fitness functions are validated, and experiments are run to show how the scenarios may be used going forward. This chapter discusses the simulation environment which each scenario from the GA is tested against to determine its criticality. Chapter 7 addresses contribution 4. The

critical scenarios selected within the genetic algorithm can be used to validate how robustly object detection and avoidance, path planning, trajectory planning etc is performed when receiving time sensitive data to update and process. Furthermore, these scenarios can be used to check latency issues, by determining the maximum latency safety message can have before it is considered irrelevant. The proposed framework has followed the development process of Functional, Logical and Concrete scenarios along the V-model-based development process in ISO 26262 as described by (Menzel, Bagschik and Maurer, 2018a). The following sections summarises each element in this process

8.2. Functional Scenarios

Functional scenarios are the development of scenarios in an abstract manner and concepts are defined in a linguistic form. To develop functional scenario, this research used ontologies. Despite there being a lot of work available in the literature for using ontologies for functional scenario generation, there does not exist work that allows for NLOS scenario development and which incorporates vehicular communication to test NLOS scenarios. This research used layers to organize the knowledge within the ontology. The ontology uses six layers, these include: 'Environment', 'Road User', 'Object Type', 'Communication Network', 'Scene' and 'Scenario', Chapter 4 discusses in detail each layer. The ontology is modelled in form of tuples by defining concepts, attributes, and domain. The ontology was modelled using protégé and a Hermit reasoner was used to check for any logical inconsistencies. Once the functional scenarios are modelled, next logical scenarios are defined. A logical scenario defines parameter range to the functional scenarios. For proof of concept for the conceptualised ontology in Chapter 4, a simple ontology was developed (shown in Chapter 7) in protégé and imported to OWL API where logical scenarios are defined by defining instance to the concept to demonstrate scenario generation incorporating vehicle-to-vehicle communication. The scenario generation validates the ontologies as well as the relations within the ontology.

8.3. Logical Scenarios

Every instance of a logical scenario would create a concrete scenario, however not all scenarios generated would be valid scenario and would be able to test the required functionality of the system. Therefore, the scenario space needs to be searched in order to select critical scenarios. The metric proposed to identify critical scenarios uses the TTC and TST metric. Using this metric, a function was written to determine the fitness value of a scenario. This function compares the TTC and the TST values such that: if the TTC is less than TST, a lower fitness value will be assigned compared to if the TTC > TST. This was evaluated by running several Monte Carlo simulations within the scenario space to determine the effectiveness of the fitness value. This has been demonstrated in Figure 8.1 which demonstrates the result of a Monte Carlo simulation calculating the fitness value for 1000 scenarios. From the Figure 6.3, it can be seen for scenarios that lead to a collision

(yellow dots) the TTC is much lower than those that do not lead to a collision (blue dots). Furthermore, for those scenarios that have relatively smaller TTC compared to TST, have a lower fitness value. Lower the fitness value, more critical the scenario.

8.4. Concrete scenarios

Using the fitness function, a genetic algorithm is used to search the scenario space and filter out identified scenarios. From Figure 8.2. it can be seen the results of how effectively the genetic algorithm searched the scenario space. Several simulations of the genetic algorithm were run (setting parameters as discussed in Chapter 6 and 7), and results can be visual in Figure 6.8. It can be observed that the genetic algorithm was searching the scenario space effectively by having a population with a lower fitness after each generation. Based on the empirical data collated when testing the scenario, we can assume that there is high probability to get a collision scenario. From this data it can be noted that the GA is able to effectively locate valid scenarios. From the category defined in chapter 6, most scenarios seem to fall between category 2 and 3. An improvement to the GA would be to locate category 1 scenarios. A possibility because the current test scenarios are not category 1 is they are not sufficiently complex. A more complex scenario would include more traffic participants various driving manoeuvres and over more underlying parameters.

8.5. Limitations and Future work

Some of the limitation of the work include:

- A limitation within this research is the context of scenario generation with communication for all possible scenarios. The ontology in its current state does not provide a relation between a pedestrian and communication network and traffic infrastructure and communication network. This limits the type of scenarios that can currently be generated, for example, it would be difficult to replicate a NLOS scenario with an ego and a pedestrian as their will be no communication, unless a CAV is present which detects the pedestrian and passes that message. This limitation limits the type of NLOS scenarios that can be generated.
- The GA currently varies only four parameters, namely the speed and position of each road users. It does not include other parameters that could influence the criticality of the scenario for example, field of view (FOV) of the sensors or weather condition which will affect the road friction coefficient and in turn the total stopping time. This limits the search dimension
- Test environment to validate the scenario does not take consideration of certain parameters which affect the criticality of the scenario. For example, currently processing and reaction time is negligible which is not the case in the real world. Furthermore, the communication network currently does not include latency which does not reflect real world. Despite having these limitations this does not

invalidate the test environment as the test environment produces an upper and lower limit of scenario which take into account these limitations.

Future work will address the limitations identified above and in particular:

- An extension of the traffic infrastructure ontology to include roadside unit communication capabilities. As well as an extension of scenarios where pedestrians are able to communicate with other road users. This will help generate a variety of critical test scenarios
- Future test scenarios would include multiple vehicles sending information to the ego vehicle and determining how the algorithm perform especially if it is safety critical message.
- Future work could also look into how algorithms perform when the data received from the other vehicles are not accurate
- The proposed genetic algorithm's performance will have to be analysed well, in order to ensure it is able to break away from local maxima and reach globally optimised solution. This may lead to the development of enhanced genetic algorithm approaches in the future.
- A fully functional simulated platform will have to be developed that provides developers multiple adaptable interfaces to plug in developed CAV functionalities for testing
- Work will also be undertaken in the future to allow the proposed framework and simulated platform to be integrated with other platforms serving testing requirement not covered with the current research.

References

Abburu, S. (2012) ‘A Survey on Ontology Reasoners and Comparison’, *International Journal of Computer Applications*, 57(17), pp. 33–39.

Abdessalem, R. Ben *et al.* (2018) ‘Testing vision-based control systems using learnable evolutionary algorithms’, *Proceedings - International Conference on Software Engineering*, pp. 1016–1026. doi: 10.1145/3180155.3180160.

Ahn, C. W. and Ramakrishna, R. S. (2003) ‘Elitism-based compact genetic algorithms’, *IEEE Transactions on Evolutionary Computation*, 7(4), pp. 367–385. doi: 10.1109/TEVC.2003.814633.

Aina, S. K. and Oyelade, O. N. (2020) ‘An Intelligence Aided VANET System: A Development of its Ontology Knowledgebase and Rule Set’, *International Journal of Scientific & Engineering Research*, 10(September).

Akagi, Y. *et al.* (2019) ‘A Risk-index based Sampling Method to Generate Scenarios for the Evaluation of Automated Driving Vehicle Safety’, *2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019*. Institute of Electrical and Electronics Engineers Inc., pp. 667–672. doi: 10.1109/ITSC.2019.8917311.

Al-Saadi, A., Setchi, R. and Hicks, Y. (2017) ‘Semantic Reasoning in Cognitive Networks for Heterogeneous Wireless Mesh Systems’, *IEEE Transactions on Cognitive Communications and Networking*, 3(3), pp. 374–389. doi: 10.1109/TCCN.2017.2712136.

Ali, S. (2019) ‘Vehicle to Vehicle Communication’, *Studies in Computational Intelligence*, 978(June), pp. 125–139. doi: 10.1007/978-981-16-2217-5_9.

Allidina, T., Deka, L. and Paluszczyszyn, D. (2018) ‘Developing Functional Test Scenarios for Around the Corner Navigation by Autonomous Vehicles’, *womEncourage*, p. 2018.

Althoff, M. *et al.* (2013) ‘Formal verification of phase-locked loops using reachability analysis and continuization’, *Communications of the ACM*, 56(10), pp. 97–104. doi: 10.1145/2507771.2507783.

Althoff, M. and Dolan, J. M. (2014) ‘Online verification of automated road vehicles using reachability analysis’, *IEEE Transactions on Robotics*. IEEE, 30(4), pp. 903–918. doi: 10.1109/TRO.2014.2312453.

Althoff, M. and Lutz, S. (2018) ‘Automatic Generation of Safety-Critical Test Scenarios for Collision Avoidance of Road Vehicles’, *IEEE Intelligent Vehicles Symposium, Proceedings*. IEEE, 2018-June(Iv), pp. 1326–1333. doi: 10.1109/IVS.2018.8500374.

Amersbach, C. and Winner, H. (2019) ‘Functional decomposition—A contribution to overcome the parameter space explosion during validation of highly automated driving’, *Traffic Injury Prevention*. Taylor & Francis, 20(sup1), pp. S52–S57. doi: 10.1080/15389588.2019.1624732.

Anaya, J. J. *et al.* (2015) ‘Vulnerable Road Users Detection Using V2X Communications’, *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 2015-Octob, pp. 107–112. doi: 10.1109/ITSC.2015.26.

Annpureddy, Y., LiuGeorgios, C. and Sankaranarayanan, F. (2011) ‘Tools and Algorithms for the Construction and Analysis of Systems’, *Springer*. Edited by P. A. Abdulla and K. R. M. Leino. Berlin,

Heidelberg: Springer Berlin Heidelberg (Lecture Notes in Computer Science), 6605. doi: 10.1007/978-3-642-19835-9.

Archer, J. and Kungl, T. högskolan. (2005) 'Indicators for traffic safety assessment and prediction and their application in micro-simulation modelling : a study of urban and suburban intersections'. Royal Institute of Technology.

Armand, A. *et al.* (2014) 'Ontology-Based Context Awareness for Driving Assistance Systems To cite this version : HAL Id : hal-01012078 Ontology-Based Context Awareness for Driving Assistance Systems'.

ASAM Standard (2016) *Standards*. Available at: https://www.asam.net/standards/ (Accessed: 22 January 2021).

Atkinson, C. and Kiko, K. (2005) 'A Detailed Comparison of UML and OWL', *University of Mannheim*.

Ayres, N., Deka, L. and Paluszczyszyn, D. (2021) 'Continuous automotive software updates through container image layers', *Electronics (Switzerland)*, 10(6), pp. 1–17. doi: 10.3390/electronics10060739.

Bach, J. *et al.* (2017) 'Reactive-Replay Approach for Verification and Validation of Closed-Loop Control Systems in Early Development', *SAE Technical Papers*. SAE International, 2017-March(March). doi: 10.4271/2017-01-1671.

Bae, J. K. *et al.* (2021) 'Implementation and Performance Evaluation for DSRC-Based Vehicular Communication System', *IEEE Access*, 9, pp. 6878–6887. doi: 10.1109/ACCESS.2020.3044358.

Bagschik, G. and Maurer, M. (2018) 'Ontology based Scene Creation for the Development of Automated Vehicles', *IEEE Intelligent Systems*. Available at: https://arxiv.org/pdf/1704.01006.pdf (Accessed: 3 May 2019).

Barnard, Y. *et al.* (2016) 'Methodology for Field Operational Tests of Automated Vehicles', *Transportation Research Procedia*. Elsevier B.V., 14, pp. 2188–2196. doi: 10.1016/j.trpro.2016.05.234.

Barrachina, J. *et al.* (2012a) 'CAOVA: A car accident ontology for VANETs', *IEEE Wireless Communications and Networking Conference, WCNC*, (May 2014), pp. 1864–1869. doi: 10.1109/WCNC.2012.6214089.

Barrachina, J. *et al.* (2012b) 'VEACON: A Vehicular Accident Ontology designed to improve safety on the roads', *Journal of Network and Computer Applications*. Elsevier, 35(6), pp. 1891–1900. doi: 10.1016/j.jnca.2012.07.013.

Batsch, F. *et al.* (2019) 'Performance Boundary Identification for the Evaluation of Automated Vehicles using Gaussian Process Classification', *2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019*, pp. 419–424. doi: 10.1109/ITSC.2019.8917119.

Beglerovic, H., Stolz, M. and Horn, M. (2018) 'Testing of autonomous vehicles using surrogate models and stochastic optimization', *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. Institute of Electrical and Electronics Engineers Inc., 2018-March, pp. 1–6. doi: 10.1109/ITSC.2017.8317768.

Benvenuti, F. *et al.* (2017) 'An ontology-based framework to support performance monitoring in

public transport systems', *Transportation Research Part C*. Elsevier Ltd, 81, pp. 188–208. doi: 10.1016/j.trc.2017.06.001.

Bernus, P. *et al.* (2009) *Handbook on Ontologies, Springer*. Springer Berlin Heidelberg. doi: 10.1007/978-3-540-92673-3.

Best, A. *et al.* (2017) 'AutonoVi: Autonomous vehicle planning with dynamic maneuvers and traffic constraints', *IEEE International Conference on Intelligent Robots and Systems*, 2017-Septe, pp. 2629–2636. doi: 10.1109/IROS.2017.8206087.

Bibi, A., Rehman, O. and Ahmed, S. (2018) 'An Ontology based Approach for Messages Dissemination in Vehicular Ad Hoc Networks', *EAI Endorsed Transactions on Scalable Information Systems*, 4(16). doi: 10.4108/10.4108/eai.13-4-2018.154468.

Bithar, V. and Karumanchi, A. (2019) 'Application of Collision Probability Estimation to Calibration of Advanced Driver Assistance Systems', *SAE Technical Papers*. SAE International, 2019-April(April). doi: 10.4271/2019-01-1133.

Bogdoll, D. *et al.* (2021) 'Description of Corner Cases in Automated Driving: Goals and Challenges', *IEEE Intelligent Systems*, pp. 1023–1028.

Bolte, J. A. *et al.* (2019) 'Towards corner case detection for autonomous driving', *IEEE Intelligent Vehicles Symposium, Proceedings*, 2019-June(November), pp. 438–445. doi: 10.1109/IVS.2019.8813817.

Borenstein, J., Everett, H. R. and Feng, L. (1996) *Navigating Mobile Robots: Sensors and Techniques*. University of Michigan.

Brambilla, M., Pardo, D. and Nicoli, M. (2020) 'Location-assisted Subspace-based Beam Alignment in LOS/NLOS mm-wave V2X Communications', *IEEE International Conference on Communications*, 2020-June. doi: 10.1109/ICC40277.2020.9148587.

Bringuente, A. C. de O., Falbo, R. de A. and Guizzardi, G. (2011) 'Using a Foundational Ontology for Reengineering a Software Process Ontology', *Journal of Information and Data Management*, 2(3), pp. 511–511. Available at: https://periodicos.ufmg.br/index.php/jidm/article/view/140 (Accessed: 5 December 2021).

Van Brummelen, J. *et al.* (2018) 'Autonomous vehicle perception: The technology of today and tomorrow', *Transportation Research Part C*, pp. 384–406. doi: 10.1016/j.trc.2018.02.012.

Buechel, M. *et al.* (2017) 'Ontology-based traffic scene modeling, traffic regulations dependent situational awareness and decision-making for automated vehicles', *IEEE Intelligent Vehicles Symposium, Proceedings*. IEEE, 7(Iv), pp. 1471–1476. doi: 10.1109/IVS.2017.7995917.

Buehler, O. and Wegener, J. (2005) 'Evolutionary Functional Testing of a Vehicle Brake Assistant System Introduction to Evolutionary Testing', *6th Metaheuristics International Conference*, pp. 157–162.

Bugała, M. (2018) 'Algorithm Applied in Autonomous Vehicle System', *Szybkobiezne Pojazdy Gasienicowe*, 50(December 2018), pp. 119–138.

Bühler, O. and Wegener, J. (2004) 'Automatic testing of an autonomous parking system using evolutionary computation', *SAE Technical Papers*. SAE International. doi: 10.4271/2004-01-0459.

Campbell, S. *et al.* (2018) 'Sensor Technology in Autonomous Vehicles : A review', *29th Irish Signals and Systems Conference, ISSC 2018*. IEEE. doi: 10.1109/ISSC.2018.8585340.

Chakraborty, B. and Chaudhuri, P. (2003) 'On the use of genetic algorithm with elitism in robust and nonparametric multivariate analysis', *Austrian Journal of Statistics*, 32(1), pp. 13–27. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.90.5252&rep=rep1&type=pdf.

Chan, V. W. S. (2021) 'Autonomous Vehicle Safety', *IEEE Communications Magazine*, 59(9), p. 4. doi: 10.1109/MCOM.2021.9566512.

Chen, B. *et al.* (2021) 'Adversarial Evaluation of Autonomous Vehicles in Lane-Change Scenarios', *IEEE Transactions on Intelligent Transportation Systems*. IEEE, pp. 1–10. doi: 10.1109/tits.2021.3091477.

Chen, H. *et al.* (2004) 'SOUPA : Standard Ontology for Ubiquitous and Pervasive Applications ∗', *The First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services*. IEEE, (1), pp. 258–267.

Chen, J., Yuan, B. and Tomizuka, M. (2019) 'Model-free Deep Reinforcement Learning for Urban Autonomous Driving', *2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019*. IEEE, pp. 2765–2771. doi: 10.1109/ITSC.2019.8917306.

Chen, S. *et al.* (2020) 'A Vision of C-V2X: Technologies, Field Testing, and Challenges with Chinese Development', *IEEE Internet of Things Journal*. IEEE, 7(5), pp. 3872–3881. doi: 10.1109/JIOT.2020.2974823.

Chen, W. (2020) *Formal Modeling and Automatic Generation of Test Cases for the Autonomous Vehicle*. Université Paris-Saclay.

Chen, Y., Das, M. and Bajpai, D. (2009) 'Improving time-to-collision estimation by IMM based Kalman filter', *SAE Technical Papers*. doi: 10.4271/2009-01-0162.

Chen, Y., Hu, C. and Wang, J. (2019) 'Human-Centered Trajectory Tracking Control for Autonomous Vehicles with Driver Cut-In Behavior Prediction', *IEEE Transactions on Vehicular Technology*. IEEE, 68(9), pp. 8461–8471. doi: 10.1109/TVT.2019.2927242.

Chou, G. *et al.* (2018) 'Using control synthesis to generate corner cases: A case study on autonomous driving', *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(11), pp. 2906–2917. doi: 10.1109/TCAD.2018.2858464.

Chu, K. (2017) *US20190011913A1 - Methods and systems for blind spot detection in an autonomous vehicle - Google Patents*. Available at: https://patents.google.com/patent/US20190011913A1/en (Accessed: 17 January 2020).

Cimiano, P. *et al.* (2009) 'Ontology Learning', *Handbook on Ontologies*. Springer, Berlin, Heidelberg, pp. 245–267. doi: 10.1007/978-3-540-92673-3_11.

Colwell, I. *et al.* (2018) 'An Automated Vehicle Safety Concept Based on Runtime Restriction of the Operational Design Domain', *IEEE Intelligent Vehicles Symposium, Proceedings*. IEEE, 2018-June(Iv), pp. 1910–1917. doi: 10.1109/IVS.2018.8500530.

Corso, A. *et al.* (2019) 'Adaptive Stress Testing with Reward Augmentation for Autonomous Vehicle Validatio', *2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019*, pp. 163–168. doi: 10.1109/ITSC.2019.8917242.

Cunto, F. (2008) ‘Assessing Safety Performance of Transportation Systems using Microscopic Simulation’. University of Waterloo. Available at: https://uwspace.uwaterloo.ca/handle/10012/4111 (Accessed: 1 September 2021).

Cutrone, S. *et al.* (2019) ‘A Framework for Identifying and Simulating Worst-Case Animal-Vehicle Interactions’, *Proceedings - 2018 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2018*. Institute of Electrical and Electronics Engineers Inc., pp. 1995–2000. doi: 10.1109/SMC.2018.00344.

Czarnecki, K. (2018) ‘Operational World Model Ontology for Automated Driving Systems - Part 2 : Road Users , Animals , Other Obstacles , and Environmental Conditions Operational World Model Ontology for Automated Driving Systems Part 2 : Road Users , Animals , Other Obstacles’, *Technical Report ·*, (July), pp. 1–58. doi: 10.13140/RG.2.2.11327.00165.

Dadwal, A. *et al.* (2018) ‘Prioritization in automotive software testing: Systematic literature review’, *CEUR Workshop Proceedings*, 2273(QuASoQ), pp. 52–58.

Van Dam, K. H. (2009) ‘Capturing socio-technical systems with agent-based modelling’. Next Generation Infrastructures Foundation. Available at: https://repository.tudelft.nl/islandora/object/uuid%3A1b36bc02-c7fe-4773-9ab2-3daa3b891754 (Accessed: 5 December 2021).

Distribu, S. (2013) ‘An Ontology for a fault tolerant vehicular information system’, *22nd International Congress of Mechanical Engineering (COBEM 2013)*, (Cobem), pp. 10187–10195.

Duan, J., Gao, F. and He, Y. (2020) ‘Test Scenario Generation and Optimization Technology for Intelligent Driving Systems’, *IEEE Intelligent Transportation Systems Magazine*. Institute of Electrical and Electronics Engineers. doi: 10.1109/MITS.2019.2926269.

Eigner, R. and Lutz, G. (2008) ‘Collision avoidance in VANETs an application for ontological context models’, *6th Annual IEEE International Conference on Pervasive Computing and Communications, PerCom 2008*, pp. 412–416. doi: 10.1109/PERCOM.2008.42.

Elliott, D., Keen, W. and Miao, L. (2019) ‘Recent advances in connected and automated vehicles’, *Journal of Traffic and Transportation Engineering (English Edition)*. Elsevier Ltd, 6(2), pp. 109–131. doi: 10.1016/j.jtte.2018.09.005.

Engel, B. (2020) ‘ASAM OpenX’, *ASAM Technical Seminar*, (October).

Erdogan, A. *et al.* (2019) ‘Real-world maneuver extraction for autonomous vehicle validation: A comparative study’, *IEEE Intelligent Vehicles Symposium, Proceedings*. IEEE, 2019-June(Iv), pp. 267–272. doi: 10.1109/IVS.2019.8814254.

Erritali, M. *et al.* (2013) ‘An ontology-based intrusion detection for vehicular ad hoc networks’, *Journal of Theoretical and Applied Information Technology*, 53(3), pp. 410–414.

Euro NCAP (2021) *Euro NCAP | The European New Car Assessment Programme*. Available at: https://www.euroncap.com/en (Accessed: 16 September 2021).

Fahrenkrog, F. *et al.* (2014) ‘Deliverable D7.1 Test and Evaluation Plan Dissemination level RE Version 1.1’. Available at: www.adaptive-ip.eu.

Falbo, R. A. *et al.* (2002) ‘Developing Software for and with Reuse : An Ontological Approach’, *In Conference on Computer Science, Software Engineering, Information Technology, e-business and*

Applications (CSITeA-02), (June), pp. 477–488.

Falbo, R. D. A., Guizzardi, G. and Duarte, K. C. (2002) 'An ontological approach to domain engineering', *ACM International Conference Proceeding Series*, 27, pp. 351–358. doi: 10.1145/568760.568822.

Feig, P., Gschwendtner, K. and Borrack, M. (2019) 'Assessment of Technical Requirements for Level 3 and Beyond Automated Driving Systems Based on Naturalistic Driving and Accident Data Analysis', *26th International Technical Conference on the Enhanced Safety of Vehicles*, pp. 1–10.

Felbinger, H. *et al.* (2019) 'Comparing two systematic approaches for testing automated driving functions', *2019 8th IEEE International Conference on Connected Vehicles and Expo, ICCVE 2019 - Proceedings*, pp. 19–24. doi: 10.1109/ICCVE45908.2019.8965209.

Feng, S. *et al.* (2019) 'Testing scenario library generation for connected and automated vehicles, part II: Case studies', *IEEE Transaction on Intelligent Transportation Systems*, pp. 1–10.

Feng, S. *et al.* (2020) 'Safety assessment of highly automated driving systems in test tracks: A new framework', *Accident Analysis & Prevention*. Pergamon, 144, p. 105664. doi: 10.1016/J.AAP.2020.105664.

Feng, S. *et al.* (2021) 'Testing Scenario Library Generation for Connected and Automated Vehicles, Part II: Case Studies', *IEEE Transactions on Intelligent Transportation Systems*, 22(9), pp. 5635–5647. doi: 10.1109/TITS.2020.2988309.

Fettke, P. and Loos, P. (2003) 'Ontological Evaluation of Reference Models Using the Bunge-Wand-Weber', *AMCIS 2003 Proceedings*, (January), pp. 2944–2966.

Figueiredo, M. C. *et al.* (2009) 'An approach to simulate autonomous vehicles in urban traffic scenarios', *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. IEEE, pp. 322–327. doi: 10.1109/ITSC.2009.5309524.

Fokin, G. (2019) 'AOA measurement processing for positioning using unmanned aerial vehicles', *2019 IEEE International Black Sea Conference on Communications and Networking, BlackSeaCom 2019*. IEEE, pp. 2019–2021. doi: 10.1109/BlackSeaCom.2019.8812834.

Fox, M. and Engineering, I. (2019) 'An Ontology-Based Standard for Transportation Planning'.

Fredj, N. *et al.* (2021) 'Component ensemble-based UML/MARTE extensions for the design of dynamic cyber-physical systems', *Proceedings of the 16th International Conference on Software Technologies, ICSOFT 2021*, (Icsoft), pp. 158–166. doi: 10.5220/0010546401580166.

Gangopadhyay, B. *et al.* (2019) 'Identification of Test Cases for Automated Driving Systems Using Bayesian Optimization', *2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019*. Institute of Electrical and Electronics Engineers Inc., pp. 1961–1967. doi: 10.1109/ITSC.2019.8917103.

Gao, F. *et al.* (2019) 'A test scenario automatic generation strategy for intelligent driving systems', *Mathematical Problems in Engineering*, 2019. doi: 10.1155/2019/3737486.

Gao, K. *et al.* (2019) 'Conditional artificial potential field-based autonomous vehicle safety control with interference of lane changing in mixed traffic scenario', *Sensors (Switzerland)*, 19(19). doi: 10.3390/s19194199.

Geiger, A. *et al.* (2013) 'Vision meets robotics: The KITTI dataset', *International Journal of Robotics Research*, 32(11), pp. 1231–1237. doi: 10.1177/0278364913491297.

Gelbal, S. Y. *et al.* (2019) 'Cooperative Collision Avoidance in a Connected Vehicle Environment', *SAE Technical Papers*. SAE International, 2019-April(April). doi: 10.4271/2019-01-0488.

Geyer, S. *et al.* (2014) 'Concept and development of a unified ontology for generating test and use-case catalogues for assisted and automated vehicle guidance', *IET Intelligent Transport Systems*, 8(3), pp. 183–189. doi: 10.1049/iet-its.2012.0188.

Gladisch, C. *et al.* (2019) 'Experience Paper: Search-Based Testing in Automated Driving Control Applications', *2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE Computer Society, pp. 26–37. doi: 10.1109/ASE.2019.00013.

Gómez-Pérez, A. (2004) 'Ontology Evaluation', *Handbook on Ontologies*. Springer, Berlin, Heidelberg, pp. 251–273. doi: 10.1007/978-3-540-24750-0_13.

Greenemeier, L. (2016) 'Driverless cars will face moral dilemmas', *Scientific American*.

Gregoriades, A. (2007) 'Towards a user-centred road safety management method based on road traffic simulation', *Proceedings - Winter Simulation Conference*, pp. 1905–1914. doi: 10.1109/WSC.2007.4419818.

Greibe, P. (2008) 'Determination of Braking Distance and Driver Behaviour Based on Braking Trials', *Transportation Research Board*, (January).

Groza , A, Marginean, A. and I. (2014) 'Towards Improving Situation Awareness during Emergency Transportation through Ambulance-2-X Communication and Semantic Stream Reasoning', *IFMBE Proceedings*, 44, pp. I–IV. doi: 10.1007/978-3-319-07653-9.

Gruber, T. R. (1993) 'A translation approach to portable ontology specifications', *Knowledge Acquisition*. Academic Press, 5(2), pp. 199–220. doi: 10.1006/KNAC.1993.1008.

Gruber, T. R. (1995) 'Toward principles for the design of ontologies used for knowledge sharing', *International Journal of Human - Computer Studies*, 43(5–6), pp. 907–928. doi: 10.1006/ijhc.1995.1081.

Guo, J., Kurup, U. and Shah, M. (2020) 'Is it Safe to Drive? An Overview of Factors, Metrics, and Datasets for Driveability Assessment in Autonomous Driving', *IEEE Transactions on Intelligent Transportation Systems*, 21(8), pp. 3135–3151. doi: 10.1109/TITS.2019.2926042.

Hammerschmidt, C. (2019) 'Radar sensor module slashes response time for autonomous cars'. Available at: https://www.eenewsautomotive.com/news/smart-signal-data-processing-slashes-response-time-cars (Accessed: 15 January 2022).

Hayward, J. C. (1972) 'Near-Miss Determination Through', *Highway Research Board*, pp. 24–35. Available at: http://onlinepubs.trb.org/Onlinepubs/hrr/1972/384/384-004.pdf.

Hegedus, T., Németh, B. and Gáspár, P. (2019) 'Graph-based Multi-Vehicle Overtaking Strategy for Autonomous Vehicles', *IFAC-PapersOnLine*, 52(5), pp. 372–377. doi: 10.1016/j.ifacol.2019.09.060.

Henion, P. and Sinelli, G. J. (2010) 'US9308867B2 - Side rear view mirror assembly indicator of blind spot occupancy - Google Patents'. Available at: https://patents.google.com/patent/US9308867B2/en (Accessed: 17 January 2020).

HermiT Reasoner (2016) *HermiT Reasoner*. Available at: http://www.hermit-reasoner.com/ (Accessed: 3 January 2021).

Hitzler, P. *et al.* (2005) 'What Is Ontology Merging?', *Web*, pp. 1–4. Available at: papers2://publication/uuid/28D629A7-A9B9-4EA8-8943-D1FEBDDB2CA5.

Hobert, L. *et al.* (2016) 'Enhancements of V2X communication in support of cooperative autonomous driving', *Infocommunications Journal*. IEEE, 8(3), pp. 27–33.

Huang, X., Zhang, W. and Li, P. (2019) 'A Path Planning Method for Vehicle Overtaking Maneuver Using Sigmoid Functions', *IFAC-PapersOnLine*, 52(8), pp. 73–80. doi: 10.1016/j.ifacol.2019.08.098.

Huang, Z., Lam, H. and Zhao, D. (2017) 'Sequential experimentation to efficiently test automated vehicles ', *IEEE*, (Winter Simulation Conference (WSC)), pp. 3078–3078. Available at: https://ppms.cit.cmu.edu/docs/detail/2421 (Accessed: 19 December 2021).

Hülsen, M., Zöllner, J. M. and Weiss, C. (2011) 'Traffic intersection situation description ontology for advanced driver assistance', *IEEE Intelligent Vehicles Symposium, Proceedings*, (Iv), pp. 993–999. doi: 10.1109/IVS.2011.5940415.

Hummel, B., Thiemann, W. and Lulcheva, I. (2008) 'Scene Understanding of Urban Road Intersections with Description Logic', *Dagstuhl Seminar Proceedings*. Available at: http://drops.dagstuhl.de/opus/volltexte/2008/1616/.

Hussein, A. *et al.* (2018) 'Autonomous cooperative driving using V2X communications in off-road environment', *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 2018-March, pp. 1–6. doi: 10.1109/ITSC.2017.8317790.

Hussein, M., Erol-Kantarci, M. and Sorour, S. (2020) *Connected and Autonomous Vehicles in Smart Cities*, *CRC Press*. CRC Press. doi: 10.1201/9780429329401/CONNECTED-AUTONOMOUS-VEHICLES-SMART-CITIES-HUSSEIN-MOUFTAH-MELIKE-EROL-KANTARCI-SAMEH-SOROUR.

Ilas, C. (2013) 'Perception in autonomous ground vehicles: A review', *2013 International Conference on Electronics, Computers and Artificial Intelligence, ECAI 2013*, (June 2013). doi: 10.1109/ECAI.2013.6636180.

Ilci, V. and Toth, C. (2020) 'High Definition 3D Map Creation Using GNSS/IMU/LiDAR Sensor Integration to Support Autonomous Vehicle Navigation', *Sensors 2020, Vol. 20, Page 899*. Multidisciplinary Digital Publishing Institute, 20(3), p. 899. doi: 10.3390/S20030899.

Ireland, M. L. *et al.* (2016) 'A Continuous-Time Model of an Autonomous Aerial Vehicle to Inform and Validate Formal Verification Methods', *Robotics*. Available at: http://arxiv.org/abs/1609.00177.

ISO 26262 (2018) *ISO - ISO 26262-1:2018 - Road vehicles — Functional safety — Part 1: Vocabulary*. Available at: https://www.iso.org/standard/68383.html (Accessed: 14 January 2021).

Jaguar Land Rover (2018) *JAGUAR LAND ROVER'S VIRTUAL EYES LOOK AT TRUST IN SELF-DRIVING CARS | JLR Corporate Website*. Available at: https://www.jaguarlandrover.com/news/2018/08/jaguar-land-rovers-virtual-eyes-look-trust-self-driving-cars (Accessed: 15 September 2021).

Jain, J. and Banvait, H. (2016) *US10115025B2 - Detecting visibility of a vehicle to driver of other*

vehicles - Google Patents. Available at: https://patents.google.com/patent/US10115025B2/en (Accessed: 17 January 2020).

Jawad, A. (2021) 'Repurposing Microscopic Driver Modeling for Scenario Generation', pp. 1–20.

Jorgensen, P. C. (2013) *Software testing: A craftsman's approach, third edition*, *Software Testing: A Craftsman's Approach, Third Edition*. doi: 10.1201/9781439889503.

Junietz, P. *et al.* (2018) 'Criticality Metric for the Safety Validation of Automated Driving using Model Predictive Trajectory Optimization', *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. Institute of Electrical and Electronics Engineers Inc., 2018-November, pp. 60–65. doi: 10.1109/ITSC.2018.8569326.

Junietz, P. M. (2019) 'Microscopic and macroscopic risk metrics for the safety validation of automated driving', *TU Darmstadt, Darmstadt*. Available at: http://tuprints.ulb.tu-darmstadt.de/9282/.

Kalra, N. and Paddock, S. M. (2016) 'Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability?', *Transportation Research Part A: Policy and Practice*. Elsevier Ltd, 94, pp. 182–193. doi: 10.1016/j.tra.2016.09.010.

Kannan, S., Thangavelu, A. and Kalivaradhan, R. (2010) 'An Intelligent Driver Assistance System (I-DAS) for Vehicle Safety Modelling using Ontology Approach', *International Journal of UbiComp*, 1(3), pp. 15–29. doi: 10.5121/iju.2010.1302.

Kavitha, C. *et al.* (2017) 'Braking distance algorithm for autonomous cars using road surface recognition', *IOP Conference Series: Materials Science and Engineering*, 263(6). doi: 10.1088/1757-899X/263/6/062034.

Klischat, M. and Althoff, M. (2019) 'Generating critical test scenarios for automated vehicles with evolutionary algorithms', *IEEE Intelligent Vehicles Symposium, Proceedings*, 2019-June, pp. 2352–2358. doi: 10.1109/IVS.2019.8814230.

Kluck, F. *et al.* (2019) 'Genetic Algorithm-Based Test Parameter Optimization for ADAS System Testing', *Proceedings - 19th IEEE International Conference on Software Quality, Reliability and Security, QRS 2019*, pp. 418–425. doi: 10.1109/QRS.2019.00058.

Klück, F. *et al.* (2019) 'Performance Comparison of Two Search-Based Testing Strategies for ADAS System Validation', in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Springer, pp. 140–156. doi: 10.1007/978-3-030-31280-0_9.

Klueck, F. *et al.* (2018) 'Using Ontologies for Test Suites Generation for Automated and Autonomous Driving Functions', *Proceedings - 29th IEEE International Symposium on Software Reliability Engineering Workshops, ISSREW 2018*, pp. 118–123. doi: 10.1109/ISSREW.2018.00-20.

Knowles Flanagan, S., He, J. and Peng, X. H. (2019) 'Improving Emergency Collision Avoidance with Vehicle to Vehicle Communications', *Proceedings - 20th International Conference on High Performance Computing and Communications, 16th International Conference on Smart City and 4th International Conference on Data Science and Systems, HPCC/SmartCity/DSS 2018*. Institute of Electrical and Electronics Engineers Inc., pp. 1322–1329. doi: 10.1109/HPCC/SMARTCITY/DSS.2018.00220.

Kochenderfer, M. J. and Wheeler, T. A. (2019) 'Expression Optimization', *Algorithms for Optimization*, pp. 361–385. Available at: https://mitpress.mit.edu/books/algorithms-optimization (Accessed: 19 December 2021).

Koenig, A. *et al.* (2018) 'Overview of HAD validation and passive HAD as a concept for validating highly automated cars', *At-Automatisierungstechnik*. Walter de Gruyter GmbH, 66(2), pp. 132–145. doi: 10.1515/AUTO-2017-0113.

Kong, J. *et al.* (2015) 'Kinematic and dynamic vehicle models for autonomous driving control design', *IEEE Intelligent Vehicles Symposium, Proceedings*. IEEE, 2015-Augus(Iv), pp. 1094–1099. doi: 10.1109/IVS.2015.7225830.

Koopman, P., Ferrell, U., *et al.* (2019) 'A Safety Standard Approach for Fully Autonomous Vehicles', *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11699 LNCS(August 2019), pp. 326–332. doi: 10.1007/978-3-030-26250-1_26.

Koopman, P., Kane, A., *et al.* (2019) 'Credible Autonomy Safety Argumentation', *Proceedings of 27th Safety-Critical Systems Symposium*, 27, pp. 1–27. Available at: https://users.ece.cmu.edu/~koopman/pubs/Koopman19_SSS_CredibleSafetyArgumentation.pdf.

Koopman, P. and Fratrik, F. (2019) 'How many operational design domains, objects, and events?', *CEUR Workshop Proceedings*, 2301, pp. 1–4.

Koren, M. *et al.* (2018) 'Adaptive Stress Testing for Autonomous Vehicles', *IEEE Intelligent Vehicles Symposium, Proceedings*. IEEE, 2018-June(Iv), pp. 1898–1904. doi: 10.1109/IVS.2018.8500400.

Kruber, F., Wurst, J. and Botsch, M. (2020) 'An Unsupervised Random Forest Clustering Technique for Automatic Traffic Scenario Categorization', *Proc. 21st International Conference Intelligent Transport System (ITSC)*, pp. 2811–2818.

Kuhn, D. R. *et al.* (2015) 'Combinatorial Testing. Theory and Practice', *Advances in Computers*. Academic Press Inc., 99, pp. 1–66. doi: 10.1016/BS.ADCOM.2015.05.003.

Kumar, A. (2013) 'Encoding Schemes in Genetic Algorithm', *International Journal of Advanced Research in IT and Engineering*, 2(3), pp. 1–7. Available at: www.garph.co.uk.

Kwon, D., Park, S., *et al.* (2018) 'A study on development of the blind spot detection system for the IoT-based smart connected car', *2018 IEEE International Conference on Consumer Electronics, ICCE 2018*. IEEE, 2018-Janua, pp. 1–4. doi: 10.1109/ICCE.2018.8326077.

Kwon, D., Malaiya, R., *et al.* (2018) 'A Study on Development of the Camera-Based Blind Spot Detection System Using the Deep Learning Methodology'. doi: 10.3390/app9142941.

Lee, D. and Meier, R. (2007) 'Primary-context model and ontology: A combined approach for pervasive transportation services', *Proceedings - Fifth Annual IEEE International Conference on Pervasive Computing and Communications Workshops, PerCom Workshops 2007*, (May), pp. 419–424. doi: 10.1109/PERCOMW.2007.95.

Lee, J. *et al.* (2010) 'A fully-integrated 77-GHz FMCW radar transceiver in 65-nm CMOS technology', *IEEE Journal of Solid-State Circuits*, 45(12), pp. 2746–2756. doi: 10.1109/JSSC.2010.2075250.

Lee, R. *et al.* (2018) 'Differential adaptive stress testing of airborne collision avoidance systems', *AIAA Modeling and Simulation Technologies Conference, 2018*. IEEE, (209959), pp. 1–13. doi: 10.2514/6.2018-1923.

Leibe, B., Seemann, E. and Schiele, B. (2005) 'Pedestrian detection in crowded scenes', *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*, I, pp. 878–885. doi: 10.1109/CVPR.2005.272.

Lenard, J., Badea-romero, A. and Danton, R. (2014) 'Typical pedestrian accident scenarios for the development of autonomous emergency braking test protocols', *Accident Analysis and Prevention*. Elsevier Ltd, 73, pp. 73–80. doi: 10.1016/j.aap.2014.08.012.

Lesemann, M. *et al.* (2011) 'A TEST PROGRAMME FOR ACTIVE VEHICLE SAFETY', *Tecnalia Transport Spain*, pp. 1–10.

Li, L. *et al.* (2016) 'Intelligence testing for autonomous vehicles: A new approach', *IEEE Transactions on Intelligent Vehicles*, 1(2), pp. 158–164. doi: 10.1109/TIV.2016.2608003.

Li, W. *et al.* (2015) 'Optimization on Black Box Function Optimization Problem', *Mathematical Problems in Engineering*. Hindawi Publishing Corporation, 2015. doi: 10.1155/2015/647234.

Li, X. (2020) 'A Scenario-Based Development Framework for Autonomous Driving'. Available at: http://arxiv.org/abs/2011.01439.

Li, Y., Tao, J. and Wotawa, F. (2020) 'Ontology-based test generation for automated and autonomous driving functions', *Information and Software Technology*. Elsevier B.V., 117(February 2019). doi: 10.1016/j.infsof.2019.106200.

Liang, X. (Joyce), Guler, S. I. and Gayah, V. V. (2019) 'Joint Optimization of Signal Phasing and Timing and Vehicle Speed Guidance in a Connected and Autonomous Vehicle Environment', *Transportation Research Record*, 2673(4), pp. 70–83. doi: 10.1177/0361198119841285.

Liu, C. *et al.* (2018) 'Performance analysis for practical unmanned aerial vehicle networks with LoS/NLoS Transmissions', *2018 IEEE International Conference on Communications Workshops, ICC Workshops 2018 - Proceedings*. IEEE, pp. 1–6. doi: 10.1109/ICCW.2018.8403635.

Liu, H. *et al.* (2002) 'Discretization: An Enabling Technique', *Data Mining and Knowledge Discovery 2002 6:4*. Springer, 6(4), pp. 393–423. doi: 10.1023/A:1016304305535.

Liu, X. *et al.* (2010) 'Real-time distance measurement using a modified camera', *2010 IEEE Sensors Applications Symposium, SAS 2010 - Proceedings*, (June 2016), pp. 54–58. doi: 10.1109/SAS.2010.5439423.

Lu, N. *et al.* (2014) 'Connected vehicles: Solutions and challenges', *IEEE Internet of Things Journal*. IEEE, 1(4), pp. 289–299. doi: 10.1109/JIOT.2014.2327587.

Ma, X. (2009) 'A new hybrid evolution genetic algorithm with Laplace crossover and power mutation', *CIS 2009 - 2009 International Conference on Computational Intelligence and Security*, 2, pp. 88–91. doi: 10.1109/CIS.2009.64.

Ma, Z. *et al.* (2019) 'The application of ontologies in multi-agent systems in the energy sector: A scoping review', *Energies*, 12(16). doi: 10.3390/en12163200.

Mahmud, S. M. S. *et al.* (2017) 'Application of proximal surrogate indicators for safety evaluation:

A review of recent developments and research needs', *IATSS Research*. International Association of Traffic and Safety Sciences, 41(4), pp. 153–163. doi: 10.1016/j.iatssr.2017.02.001.

Maiti, S., Winter, S. and Kulik, L. (2017) 'A conceptualization of vehicle platoons and platoon operations q', *Transportation Research Part C*. Elsevier Ltd, 80, pp. 1–19. doi: 10.1016/j.trc.2017.04.005.

Masuda, S., Nakamura, H. and Kajitani, K. (2018) 'Rule-based searching for collision test cases of autonomous vehicles simulation', *IET Intelligent Transport Systems*. Institution of Engineering and Technology, 12(9), pp. 1088–1095. doi: 10.1049/IET-ITS.2018.5335.

MathWorks (2021a) *Automated Driving Toolbox - MATLAB*. Available at: https://uk.mathworks.com/products/automated-driving.html (Accessed: 3 January 2022).

MathWorks (2021b) *Create box collision geometry - MATLAB - MathWorks United Kingdom*. Available at: https://uk.mathworks.com/help/robotics/ref/collisionbox.html (Accessed: 4 January 2022).

MATLAB - MathWorks - MATLAB & Simulink (2021). Available at: https://uk.mathworks.com/products/matlab.html (Accessed: 30 December 2021).

MATLAB, M. (2021a) *Design driving scenarios, configure sensors, and generate synthetic data - MATLAB - MathWorks United Kingdom*. Available at: https://uk.mathworks.com/help/driving/ref/drivingscenariodesigner-app.html (Accessed: 15 January 2022).

MATLAB, M. (2021b) *Import ASAM OpenDRIVE Roads into Driving Scenario - MATLAB & Simulink - MathWorks United Kingdom*. Available at: https://uk.mathworks.com/help/driving/ug/import-opendrive-roads-into-driving-scenario.html (Accessed: 5 January 2022).

Maurer, M., Gerdes, J. C. and Winner, B. L. H. (2016) *Autonomous driving use cases and their specs, Springer Nature*. doi: 10.1007/978-3-662-48847-8_19.

Mayr, A. *et al.* (2018) 'Concept for an integrated product and process development of electric drives using a knowledge-based system', *2017 7th International Electric Drives Production Conference, EDPC 2017 - Proceedings*, 2017-Decem, pp. 1–7. doi: 10.1109/EDPC.2017.8328157.

McGuinness, N. F. N. and D. L. (2000) 'Ontology Development 101 : A Guide to Creating Your', *Stanford Knowledge Systems Laboratory*, pp. 1–25. Available at: http://protege.stanford.edu/publications/ontology_development/ontology101.pdf.

Mealy, G. H. (1967) 'Another look at data', *Proceedings of the November 14-16, 1967, fall joint computer conference*, pp. 525–534.

Meinke, K., Niu, F. and Sindhu, M. (2011) 'Learning-Based Software Testing: A Tutorial', *Communications in Computer and Information Science*. Springer, Berlin, Heidelberg, 336 CCIS, pp. 200–219. doi: 10.1007/978-3-642-34781-8_16.

Menzel, T. *et al.* (2018) *From Functional to Logical Scenarios: Detailing a Keyword-Based Scenario Description for Execution in a Simulation Environment*. Available at: https://vires.com/vtd-vires-virtual-test-drive/, (Accessed: 29 May 2019).

Menzel, T., Bagschik, G. and Maurer, M. (2018a) 'Scenarios for development, test and validation of

automated vehicles', *arXiv*, (January).

Menzel, T., Bagschik, G. and Maurer, M. (2018b) 'Scenarios for Development , Test and Validation of Automated Vehicles', *IEEE Intelligent Vehicles Symposium, Proceedings*.

Milakis, D., Van Arem, B. and Van Wee, B. (2017) 'Policy and society related implications of automated driving: A review of literature and directions for future research', *Journal of Intelligent Transportation Systems: Technology, Planning, and Operations*. Taylor & Francis, 21(4), pp. 324–348. doi: 10.1080/15472450.2017.1291351.

Milton, S. K. and Kazmierczak, E. (1AD) 'An Ontology of Data Modelling Languages: A Study Using a Common-Sense Realistic Ontology', *Journal of Database Management*. IGI Global, 15(2), pp. 19–38. doi: 10.4018/JDM.2004040102.

Minderhoud, M. M. and Bovy, P. H. L. (2001) 'Extended time-to-collision measures for road traffic safety assessment', *Accident Analysis and Prevention*, 33(1), pp. 89–97. doi: 10.1016/S0001-4575(00)00019-1.

Minn, H., Zeng, M. and Bhargava, V. (2015) 'Towards a definition of the {Internet of Things (IoT)}', *IEEE Internet Initiative*, pp. 1–86.

Minnerup, P. M. (2017) *An Efficient Method for Testing Autonomous Driving Software against Nondeterministic Influences*. Technischen Universit¨at M¨unchen. Available at: https://mediatum.ub.tum.de/doc/1353122/675343.pdf.

Mittag, J. *et al.* (2009) 'A comparison of single- and multi-hop beaconing in VANETs', *VANET'09 - Proceedings of the 6th ACM International Workshop on VehiculAr Inter-NETworking*, (formerly VII), pp. 79–88. doi: 10.1145/1614269.1614282.

Mohammed, A. S. *et al.* (2020) 'The perception system of intelligent ground vehicles in all weather conditions: A systematic literature review', *Sensors (Switzerland)*, 20(22), pp. 1–34. doi: 10.3390/s20226532.

Mohan, D., Ali, G. G. M. N. and Joo Chong, P. H. (2020) 'Machine Learning Algorithm for NLOS Millimeter Wave in 5G V2X Communication', *Wireless Personal Communs*, 106, pp. 41–70. doi: 10.5121/csit.2020.101706.

Morignot, P. and Nashashibi, F. (2013) 'An ontology-based approach to relax traffic regulation for autonomous vehicle assistance', *IASTED Multiconferences - Proceedings of the IASTED International Conference on Artificial Intelligence and Applications, AIA 2013*, pp. 122–129. doi: 10.2316/P.2013.793-024.

Moz, M. and Vaz Pato, M. (2007) 'A genetic algorithm approach to a nurse rerostering problem', *Computers & Operations Research*. Pergamon, 34(3), pp. 667–691. doi: 10.1016/J.COR.2005.03.019.

Mullins, G. (2018) *Adaptive Sampling Methods for Testing Autonomous Systems, Ph.D. dissertation*. University Maryland, College Park, MD, USA. Available at: https://www.proquest.com/openview/a89e26ad2b492970222edd4fe5852c47/1?pq-origsite=gscholar&cbl=18750 (Accessed: 30 November 2021).

Mullins, G. E. *et al.* (2018) 'Adaptive generation of challenging scenarios for testing and evaluation of autonomous vehicles', *Journal of Systems and Software*. Elsevier, 137, pp. 197–215. doi:

10.1016/J.JSS.2017.10.031.

Nabhan, M. *et al.* (2020) 'Optimizing coverage of simulated driving scenarios for the autonomous vehicle To cite this version : HAL Id : hal-02433530 Optimizing coverage of simulated driving scenarios for the autonomous vehicle'.

Naranjo, J. E. *et al.* (2016) 'Application of vehicle to another entity (V2X) communications for motorcycle crash avoidance', *https://doi.org/10.1080/15472450.2016.1247703*. Taylor & Francis, 21(4), pp. 285–295. doi: 10.1080/15472450.2016.1247703.

Naser, F. *et al.* (2013) 'Non-Line-of-Sight localization in multipath environments', *IEEE Transactions on Intelligent Transportation Systems*. IEEE, 16(5), pp. 75–86. doi: 10.1145/2534169.2486039.

Naser, F. *et al.* (2018) 'ShadowCam: Real-Time Detection of Moving Obstacles behind A Corner for Autonomous Vehicles', *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 2018-Novem, pp. 560–567. doi: 10.1109/ITSC.2018.8569569.

Naser, F. *et al.* (2019) 'Infrastructure-free NLoS Obstacle Detection for Autonomous Cars', *IEEE International Conference on Intelligent Robots and Systems*, pp. 250–257. doi: 10.1109/IROS40897.2019.8967554.

NHSTA (2021) *NHTSA | National Highway Traffic Safety Administration*. Available at: https://www.nhtsa.gov/ (Accessed: 16 January 2021).

Nie, C. and Leung, H. (2011) 'A survey of combinatorial testing', *ACM Computing Surveys (CSUR)*. ACM PUB27 New York, NY, USA , 43(2). doi: 10.1145/1883612.1883618.

Nilsson, J. *et al.* (2011) 'Using Augmentation Techniques for Performance Evaluation in Automotive Safety', *Handbook of Augmented Reality*. Springer, New York, NY, pp. 631–649. doi: 10.1007/978-1-4614-0064-6_29.

Nilsson, J. (2014) 'Computational verification methods for automotive safety systems', p. 203.

NITSCHE, P. (2018) 'Safety-critical scenarios and virtual testing procedures for automated cars at road intersections', (August), p. 1 file. Available at: https://dspace.lboro.ac.uk/2134/34433%0Ahttps://trid.trb.org/view/1602620.

Noor-A-Rahim, M. *et al.* (2019) 'Broadcast Performance Analysis and Improvements of the LTE-V2V Autonomous Mode at Road Intersection', *IEEE Transactions on Vehicular Technology*, 68(10), pp. 9359–9369. doi: 10.1109/TVT.2019.2936799.

Opdahl, A. L., Henderson-Sellers, B. and Barbier, F. (2001) 'Ontological analysis of whole–part relationships in OO-models', *Information and Software Technology*. Elsevier, 43(6), pp. 387–399. doi: 10.1016/S0950-5849(00)00175-0.

Orso, A. and Rothermel, G. (2014) 'Software testing: A research travelogue (2000-2014)', *Future of Software Engineering, FOSE 2014 - Proceedings*, pp. 117–132. doi: 10.1145/2593882.2593885.

Ortega, J., Lengyel, H. and Szalay, Z. (2020) 'Overtaking maneuver scenario building for autonomous vehicles with PreScan software', *Transportation Engineering*. Elsevier Ltd, 2(September), p. 100029. doi: 10.1016/j.treng.2020.100029.

Ouyang, J. *et al.* (2020) ‘Baidu Kunlun An AI processor for diversified workloads’, *2020 IEEE Hot Chips 32 Symposium (HCS)*. IEEE Computer Society, pp. 1–18. doi: 10.1109/HCS49909.2020.9220641.

OWL Web Ontology Language Reference (2016) *OWL Web Ontology Language Reference*. Available at: https://www.w3.org/TR/owl-ref/ (Accessed: 3 January 2021).

Ozbay, K. *et al.* (2008) ‘Derivation and Validation of New Simulation-Based Surrogate Safety Measure’:, *https://doi.org/10.3141/2083-12*. SAGE PublicationsSage CA: Los Angeles, CA, (2083), pp. 105–113. doi: 10.3141/2083-12.

Paden, B. *et al.* (2016) ‘A survey of motion planning and control techniques for self-driving urban vehicles’, *IEEE Transactions on Intelligent Vehicles*, 1(1), pp. 33–55. doi: 10.1109/TIV.2016.2578706.

Papadimitratos, P., Gligor, V. and Hubaux, J. P. (2006) ‘Securing vehicular communications-assumptions, requirements, and principles’, *Workshop on Embedded Security in Cars (ESCAR)*, 2006, pp. 24pp21-29. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.140.4967&rep=rep1&type=pdf.

Park, Sangmin *et al.* (2021) ‘Scenario-mining for level 4 automated vehicle safety assessment from real accident situations in urban areas using a natural language process’, *Sensors*, 21(20). doi: 10.3390/s21206929.

Park, W. J. *et al.* (2008) ‘Parking space detection using ultrasonic sensor in parking assistance system’, *IEEE Intelligent Vehicles Symposium, Proceedings*, pp. 1039–1044. doi: 10.1109/IVS.2008.4621296.

Payen de La Garanderie, G., Atapour Abarghouei, A. and Breckon, T. P. (2018) ‘Eliminating the blind spot: Adapting 3D object detection and monocular depth estimation to 360 ° Panoramic Imagery’, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11217 LNCS, pp. 812–830. doi: 10.1007/978-3-030-01261-8_48.

PEGASUS (2016) *PEGASUS - pegasus-EN*. Available at: https://www.pegasusprojekt.de/en/about-PEGASUS (Accessed: 16 September 2021).

Perea-Strom, D. *et al.* (2020) ‘GNSS Integration in the Localization System of an Autonomous Vehicle Based on Particle Weighting’, *IEEE Sensors Journal*. Institute of Electrical and Electronics Engineers Inc., 20(6), pp. 3314–3323. doi: 10.1109/JSEN.2019.2955210.

Pierson, A. *et al.* (2019) ‘Learning risk level set parameters from data sets for safer driving’, *IEEE Intelligent Vehicles Symposium, Proceedings*. IEEE, 2019-June(Iv), pp. 273–280. doi: 10.1109/IVS.2019.8813842.

Pollard, E., Morignot, P. and Nashashibi, F. (2013) ‘An ontology-based model to determine the automation level of an automated vehicle for co-driving’, *Proceedings of the 16th International Conference on Information Fusion, FUSION 2013*. ISIF (Intl Society of Information Fusi, (December 1997), pp. 596–603.

Ponn, T. *et al.* (2019) ‘A method for the selection of challenging driving scenarios for automated vehicles based on an objective characterization of the driving behavior’, *9. Tagung …*. Available at:

https://mediatum.ub.tum.de/1535144.

Ponn, T., Diermeyer, F. and Gnandt, C. (2019) ‘an Optimization-Based Method To Identify Relevant Scenarios for Type Approval of Automated Vehicles’, *Esv*, (July 2020), pp. 1–19. Available at: http://indexsmart.mirasmart.com/26esv/PDFfiles/26ESV-000024.pdf.

de Ponte Müller, F. (2017) ‘Survey on ranging sensors and cooperative techniques for relative positioning of vehicles’, *Sensors (Switzerland)*, 17(2), pp. 1–27. doi: 10.3390/s17020271.

Provine, R. *et al.* (2004) ‘Ontology-based methods for enhancing autonomous vehicle path planning’, *Robotics and Autonomous Systems*, 49(1-2 SPEC. ISS.), pp. 123–133. doi: 10.1016/j.robot.2004.07.020.

Qi, Y. *et al.* (2019) ‘A Trajectory-Based Method for Scenario Analysis and Test Effort Reduction for Highly Automated Vehicle’, *SAE Technical Papers*. SAE International, 2019-April(April). doi: 10.4271/2019-01-0139.

Queiroz, R., Berger, T. and Czarnecki, K. (2019) ‘GeoScenario: An open DSL for autonomous driving scenario representation’, *IEEE Intelligent Vehicles Symposium, Proceedings*. IEEE, 2019-June(Iv), pp. 287–294. doi: 10.1109/IVS.2019.8814107.

R. Genesereth, M. and Nilsson. J, N. (2012) *Logical Foundations of Artificial Intelligence - Michael R. Genesereth, Nils J. Nilsson - Google Books*.

Rahiman, W. and Zainal, Z. (2013) ‘An overview of development GPS navigation for autonomous car’, *Proceedings of the 2013 IEEE 8th Conference on Industrial Electronics and Applications, ICIEA 2013*. IEEE, pp. 1112–1118. doi: 10.1109/ICIEA.2013.6566533.

Rasshofer, R. H. and Gresser, K. (2005) ‘Automotive Radar and Lidar Systems for Next Generation Driver Assistance Functions’, *Advances in Radio Science*, 3, pp. 205–209. doi: 10.5194/ars-3-205-2005.

van Ratingen, M. R. (2017) ‘The Euro NCAP Safety Rating’. Springer Vieweg, Wiesbaden, pp. 11–20. doi: 10.1007/978-3-658-18107-9_2.

Van Ratingen, M. *et al.* (2016) ‘The European New Car Assessment Programme: A historical review’, *Chinese Journal of Traumatology - English Edition*. Elsevier Ltd, 19(2), pp. 63–69. doi: 10.1016/j.cjtee.2015.11.016.

Reiterer, F. *et al.* (2019) ‘Beyond-design-basis evaluation of advanced driver assistance systems’, *IEEE Intelligent Vehicles Symposium, Proceedings*. Institute of Electrical and Electronics Engineers Inc., 2019-June, pp. 2119–2124. doi: 10.1109/IVS.2019.8813893.

Riedmaier, S. *et al.* (2020) ‘Survey on Scenario-Based Safety Assessment of Automated Vehicles’, *IEEE Access*, 8, pp. 87456–87477. doi: 10.1109/ACCESS.2020.2993730.

Robles-Ramirez, D. A., Escamilla-Ambrosio, P. J. and Tryfonas, T. (2017) ‘IoTsec: UML extension for internet of things systems security modelling’, *Proceedings - 2017 International Conference on Mechatronics, Electronics, and Automotive Engineering, ICMEAE 2017*, 2017-Janua, pp. 151–156. doi: 10.1109/ICMEAE.2017.20.

Rocklage, E. *et al.* (2018) ‘Automated scenario generation for regression testing of autonomous vehicles’, *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 2018-March, pp. 476–483. doi: 10.1109/ITSC.2017.8317919.

Roos, F. *et al.* (2019) 'Radar sensors for autonomous driving', *IEEE Microwave Magazine*. Institute of Electrical and Electronics Engineers Inc., 20(9), pp. 58–72. doi: 10.1109/MMM.2019.2922120.

Royo, S. and Ballesta-Garcia, M. (2019) 'An Overview of Lidar Imaging Systems for Autonomous Vehicles', *Applied Sciences 2019, Vol. 9, Page 4093*. Multidisciplinary Digital Publishing Institute, 9(19), p. 4093. doi: 10.3390/APP9194093.

SAE (2016) 'Surface Vehicle', *SAE International*, p. 30.

SAE International (2018) 'Surface Vehicle', *SAE International*, 4970(724), pp. 1–5.

Saffarzadeh, M. *et al.* (2013) 'A general formulation for time-to-collision safety indicator', *Proceedings of the Institution of Civil Engineers: Transport*, 166(5), pp. 294–304. doi: 10.1680/tran.11.00031.

Saito, Y. *et al.* (2021) 'A context-aware driver model for determining recommended speed in blind intersection situations', *Accident Analysis and Prevention*. Elsevier Ltd, 163(October), p. 106447. doi: 10.1016/j.aap.2021.106447.

Santos J, L. M. and Moreira, E. (2019) 'An evaluation of reputation with regard to the opportunistic forwarding of messages in VANETs', *Eurasip Journal on Wireless Communications and Networking*, 2019(1). doi: 10.1186/s13638-019-1518-x.

Sauerbier, J. *et al.* (2018) 'Definition of Scenarios for Safety Validation of Automated Driving Functions', *ATZ worldwide 2018 121:1*. Springer, 121(1), pp. 42–45. doi: 10.1007/S38311-018-0197-2.

Saunier, N. and Sayed, T. (2008) 'Probabilistic Framework for Automated Analysis of Exposure to Road Collisions':, *https://doi.org/10.3141/2083-11*. SAGE PublicationsSage CA: Los Angeles, CA, (2083), pp. 96–104. doi: 10.3141/2083-11.

Schlenoff, C. *et al.* (2003) 'Using ontologies to aid navigation planning in autonomous vehicles', *The Knowledge Engineering Review*. Cambridge University Press, 18(3), pp. 243–255. doi: 10.1017/S0269888904000050.

Schram, R., Williams, A. and Ratingen, M. van (2013) 'Implementation of autonomous emergency braking (AEB), the next step in euro NCAP' s safety assessment', *The 23rd International Technical Conference on the Enhanced Safety of Vehicles (ESV)*, pp. 1–6.

SchreierMatthias, WillertVolker and AdamyJurgen (2016) 'An Integrated Approach to Maneuver-Based Trajectory Prediction and Criticality Assessment in Arbitrary Road Environments', *IEEE Transactions on Intelligent Transportation Systems*. IEEE Press PUB767 Piscataway, NJ, USA , 17(10), pp. 2751–2766. doi: 10.1109/TITS.2016.2522507.

Schuldt, F. (2017) 'Ein Beitrag für den methodischen Test von automatisierten Fahrfunktionen mit Hilfe von virtuellen Umgebungen'.

Schwarz, C. (2014) 'On computing time-to-collision for automation scenarios', *Transportation Research Part F: Traffic Psychology and Behaviour*. Elsevier Ltd, 27(PB), pp. 283–294. doi: 10.1016/j.trf.2014.06.015.

Shalev-Shwartz, S., Shammah, S. and Shashua, A. (2017) 'On a Formal Model of Safe and Scalable Self-driving Cars', pp. 1–37. Available at: http://arxiv.org/abs/1708.06374.

Shanks, G., Tansley, E. and Weber, R. (2003) ‘Using ontology to validate conceptual models’, *Communications of the ACM*. ACM PUB27 New York, NY, USA , 46(10), pp. 85–89. doi: 10.1145/944217.944244.

Sharma, P., Wadhwa, A. and Komal, K. (2014) ‘Analysis of Selection Schemes for Solving an Optimization Problem in Genetic Algorithm’, *International Journal of Computer Applications*, 93(11), pp. 1–3. doi: 10.5120/16256-5714.

Shbeeb, L. (2000) *Development of traffic conflicts technique for different environments: A comparative study of pedestrian conflicts in Sweden and Jordan*. Available at: https://lup.lub.lu.se/search/publication/19830 (Accessed: 1 September 2021).

So, J. (Jason) *et al.* (2019) ‘Generating Traffic Safety Test Scenarios for Automated Vehicles using a Big Data Technique’, *KSCE Journal of Civil Engineering 2019 23:6*. Springer, 23(6), pp. 2702–2712. doi: 10.1007/S12205-019-1287-4.

Songchitruksa, P. and Tarko, A. P. (2006) ‘Practical Method for Estimating Frequency of Right-Angle Collisions at Traffic Signals’:, *https://doi.org/10.1177/0361198106195300111*. SAGE PublicationsSage CA: Los Angeles, CA, 1953(1), pp. 89–97. doi: 10.1177/0361198106195300111.

SOTIF (2019a) *ISO/PAS 21448:2019(en), Road vehicles — Safety of the intended functionality*. Available at: https://www.iso.org/obp/ui/#iso:std:iso:pas:21448:ed-1:v1:en (Accessed: 16 January 2021).

SOTIF (2019b) *ISO - ISO/PAS 21448:2019 - Road vehicles — Safety of the intended functionality*. Available at: https://www.iso.org/standard/70939.html (Accessed: 4 January 2021).

Standards, Training, Testing, Assessment and Certification | BSI (2018). Available at: https://www.bsigroup.com/en-GB/ (Accessed: 16 September 2021).

Stanford University (2016) *protégé*. Available at: https://protege.stanford.edu/ (Accessed: 9 January 2021).

Stark, L., Düring, M., *et al.* (2019) ‘Quantifying Vision Zero: Crash avoidance in rural and motorway accident scenarios by combination of ACC, AEB, and LKS projected to German accident occurrence’, *Traffic Injury Prevention*. Taylor & Francis, 20(sup1), pp. S126–S132. doi: 10.1080/15389588.2019.1605167.

Stark, L., Obst, S., *et al.* (2019) ‘Towards vision zero: Addressing white spots by accident data based ADAS design and evaluation’, *2019 IEEE International Conference on Vehicular Electronics and Safety, ICVES 2019*. IEEE. doi: 10.1109/ICVES.2019.8906409.

Stern, R. E. *et al.* (2018) ‘Dissipation of stop-and-go waves via control of autonomous vehicles: Field experiments’, *Transportation Research Part C: Emerging Technologies*, 89(May 2017), pp. 205–221. doi: 10.1016/j.trc.2018.02.005.

Sthamer, H.-H. (1995) *The Automatic Generation of Software Test Data Using Genetic Algorithms*. ProQuest Dissertations Publishing.

Stumper, D. and Dietmayer, K. (2018) ‘Towards Criticality Characterization of Situational Space’, *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. Institute of Electrical and Electronics Engineers Inc., 2018-November, pp. 3378–3382. doi: 10.1109/ITSC.2018.8569505.

Suganuma, N. & U. (2012) ‘Development of an autonomous vehicle — System overview of test ride

vehicle in the Tokyo motor show 2011', *SICE Annual Conference (SICE)*, pp. 215–218. Available at: https://www.researchgate.net/publication/261157295_Development_of_an_autonomous_vehicle_-_System_overview_of_test_ride_vehicle_in_the_Tokyo_motor_show_2011 (Accessed: 15 September 2021).

SVENSSON, A. A. (1998) *A METHOD FOR ANALYSING THE TRAFFIC PROCESS IN A SAFETY PERSPECTIVE*. Available at: https://trid.trb.org/view/715427 (Accessed: 1 September 2021).

Taieb-Maimon, M. and Shinar, D. (2016) 'Minimum and Comfortable Driving Headways: Reality versus Perception':, *http://dx.doi.org/10.1518/001872001775992543*. SAGE PublicationsSage CA: Los Angeles, CA, 43(1), pp. 159–172. doi: 10.1518/001872001775992543.

Takács, Á. *et al.* (2018) 'Assessment and Standardization of Autonomous Vehicles', *INES 2018 - IEEE 22nd International Conference on Intelligent Engineering Systems, Proceedings*, (Level 0), pp. 000185–000192. doi: 10.1109/INES.2018.8523899.

Tesla (2020) *Tesla's 'Dojo' deployment in 2021 will make Autopilot and FSD a 'distant first' against rivals*. Available at: https://www.teslarati.com/tesla-dojo-autopilot-fsd-improvements-release-date/ (Accessed: 4 December 2021).

The role of ontologies and reasoners (2016). Available at: https://docs.nomagic.com/display/CCMP190/The+role+of+ontologies+and+reasoners (Accessed: 14 January 2022).

Tian, Y., Jana, S. and Ray, B. (2018) 'DeepTest : Automated Testing of Deep-Neural-Network-driven Autonomous Cars'.

Tuncali, C. E. *et al.* (2017) 'Functional gradient descent optimization for automatic test case generation for vehicle controllers', *IEEE International Conference on Automation Science and Engineering*, 2017-Augus, pp. 1059–1064. doi: 10.1109/COASE.2017.8256245.

Tuncali, C. E. *et al.* (2018) 'Simulation-based Adversarial Test Generation for Autonomous Vehicles with Machine Learning Components', *IEEE Intelligent Vehicles Symposium, Proceedings*, 2018-June(Iv), pp. 1555–1562. doi: 10.1109/IVS.2018.8500421.

Tuncali, C. E. (2019) 'Search-Based Test Generation for Automated Driving Systems: From Perception to Control Logic', *ProQuest Dissertations and Theses*, (May), p. 212.

Tuncali, C. E. *et al.* (2020) 'Requirements-Driven Test Generation for Autonomous Vehicles with Machine Learning Components', *IEEE Transactions on Intelligent Vehicles*. Institute of Electrical and Electronics Engineers Inc., 5(2), pp. 265–280. doi: 10.1109/TIV.2019.2955903.

Tuncali, C. E., Pavlic, T. P. and Fainekos, G. (2016) 'Utilizing S-TaLiRo as an automatic test generation framework for autonomous vehicles', *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. Institute of Electrical and Electronics Engineers Inc., pp. 1470–1475. doi: 10.1109/ITSC.2016.7795751.

UK GOV (2018) *Competition overview - Meridian 2: connected vehicles data exchange - Innovation Funding Service*. Available at: https://apply-for-innovation-funding.service.gov.uk/competition/173/overview (Accessed: 15 September 2021).

UK GOV (2019) *The Highway Code - Guidance - GOV.UK*. Available at: https://www.gov.uk/guidance/the-highway-code/motorways-253-to-273 (Accessed: 9 August 2021).

UK GOV (2020a) *Nationally significant transport infrastructure projects - GOV.UK*. Available at: https://www.gov.uk/government/publications/nationally-significant-transport-infrastructure-projects (Accessed: 28 November 2021).

UK GOV (2020b) *Speed limits - GOV.UK*. Available at: https://www.gov.uk/speed-limits (Accessed: 9 August 2021).

UK GOV (2020c) *The Highway Code - Guidance - GOV.UK*. Available at: https://www.gov.uk/guidance/the-highway-code/general-rules-techniques-and-advice-for-all-drivers-and-riders-103-to-158 (Accessed: 25 August 2021).

Ulbrich, S. *et al.* (2015) 'Defining and Substantiating the Terms Scene, Situation, and Scenario for Automated Driving', *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. IEEE, 2015-Octob, pp. 982–988. doi: 10.1109/ITSC.2015.164.

Unified Modeling Language (UML) (2018). Available at: https://www.omg.org/spec/UML/2.5/About-UML/ (Accessed: 28 November 2021).

Universiteit, V. (2001) 'Ontologies and Electronic Commerce'.

Urmson, D. A. D. P. (2013) 'Modifying behavior of autonomous vehicles based on sensor blind spots and limitations'. Google Patents. Available at: https://patents.google.com/patent/US9367065B2/en (Accessed: 17 January 2020).

Uschold, M. (1996) 'Ontologies : Principles , methods and applications Ontologies : Principles , Methods and Applications Mike Uschold & Michael Gruninger AIAI-TR-191 February 1996 To appear in Knowledge Engineering Review Volume 11 Number 2 , June 1996 Mike Uschold Tel : Mi', (January 1996).

Vargas, J. *et al.* (2021) 'An overview of autonomous vehicles sensors and their vulnerability to weather conditions', *Sensors*, 21(16), pp. 1–22. doi: 10.3390/s21165397.

Vishnukumar, H. J. *et al.* (2017) 'Machine learning and deep neural network — Artificial intelligence core for lab and real-world test and validation for ADAS and autonomous vehicles: AI for efficient and quality test and validation', *undefined*. Institute of Electrical and Electronics Engineers Inc., 2018-January, pp. 714–721. doi: 10.1109/INTELLISYS.2017.8324372.

Vogel, K. (2003) 'A comparison of headway and time to collision as safety indicators', *Accident Analysis & Prevention*. Pergamon, 35(3), pp. 427–433. doi: 10.1016/S0001-4575(02)00022-2.

Vriesman, D. *et al.* (2021) 'A Fusion Approach for Pre-Crash Scenarios based on Lidar and Camera Sensors', *IEEE Vehicular Technology Conference*. Institute of Electrical and Electronics Engineers Inc., 2021-April. doi: 10.1109/VTC2021-SPRING51267.2021.9449039.

Wachenfeld, W. and Winner, H. (2015) 'Virtual Assessment of Automation in Field Operation. A New Runtime Validation Method', *Robotics and Autonomous Systems*, 10. Available at: https://www.researchgate.net/publication/283326636_Virtual_Assessment_of_Automation_in_Field_Operation_A_New_Runtime_Validation_Method (Accessed: 4 December 2021).

Wang, C. and Winner, H. (2019) 'Overcoming Challenges of Validation Automated Driving and Identification of Critical Scenarios', *2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019*. IEEE, pp. 2639–2644. doi: 10.1109/ITSC.2019.8917045.

Wang, J. *et al.* (2018) 'Traffic Sensory Data Classification by Quantifying Scenario Complexity',

IEEE Intelligent Vehicles Symposium, Proceedings. IEEE, 2018-June(Iv), pp. 1543–1548. doi: 10.1109/IVS.2018.8500669.

Wang, X. *et al.* (2021) 'Operational design domain of autonomous vehicles at skewed intersection', *Accident Analysis and Prevention*. Elsevier Ltd, 159(January), p. 106241. doi: 10.1016/j.aap.2021.106241.

Wang, X., Peng, H. and Zhao, D. (2019) 'Combining reachability analysis and importance sampling for accelerated evaluation of highly automated vehicles at pedestrian crossing', *ASME 2019 Dynamic Systems and Control Conference, DSCC 2019*. American Society of Mechanical Engineers (ASME), 3. doi: 10.1115/DSCC2019-9179.

Waymo (2020a) *Safety Report and Whitepapers – Waymo*. Available at: https://waymo.com/safety/ (Accessed: 22 November 2021).

Waymo (2020b) *Waymo Opens Robo-Taxi Service to the Public in US City of Phoenix | Technology News*. Available at: https://gadgets.ndtv.com/transportation/news/google-waymo-self-driving-cars-phoenix-launch-2307438 (Accessed: 4 December 2021).

Waymo (2020c) *Waypoint - The official Waymo blog: Off road, but not offline: How simulation helps advance our Waymo Driver*. Available at: https://blog.waymo.com/2020/04/off-road-but-not-offline--simulation27.html (Accessed: 4 December 2021).

Weber, H. *et al.* (2019) 'Traffic Injury Prevention A framework for definition of logical scenarios for safety assurance of automated driving A framework for definition of logical scenarios for safety assurance of automated driving'. doi: 10.1080/15389588.2019.1630827.

Wen, W., Zhang, G. and Hsu, L. T. (2018) 'Exclusion of GNSS NLOS receptions caused by dynamic objects in heavy traffic urban scenarios using real-time 3D point cloud: An approach without 3D maps', *2018 IEEE/ION Position, Location and Navigation Symposium, PLANS 2018 - Proceedings*. IEEE, pp. 158–165. doi: 10.1109/PLANS.2018.8373377.

Wen, X., Xia, Q. and Zhao, Y. (2006) 'An effective genetic algorithm for circularity error unified evaluation', *International Journal of Machine Tools and Manufacture*. Pergamon, 46(14), pp. 1770–1777. doi: 10.1016/J.IJMACHTOOLS.2005.11.015.

Whitley, D., Starkweather, T. and Bogart, C. (1990) 'Genetic algorithms and neural networks: optimizing connections and connectivity', *Parallel Computing*. North-Holland, 14(3), pp. 347–361. doi: 10.1016/0167-8191(90)90086-O.

Wongpiromsarn, T. and Murray, R. M. (2008) 'Formal Verification of an Autonomous Vehicle System', *Conference on Decision and Control*. Available at: http://www.cds.caltech.edu/~murray/papers/wm08-cdc.html.

Wotawa, F. and Li, Y. (2018a) *From ontologies to input models for combinatorial testing, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Springer International Publishing. doi: 10.1007/978-3-319-99927-2_14.

Wotawa, F. and Li, Y. (2018b) 'From ontologies to input models for combinatorial testing', in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Springer Verlag, pp. 155–170. doi: 10.1007/978-3-319-99927-2_14.

Xia, Q. *et al.* (2017) 'Automatic Generation Method of Test Scenario for ADAS Based on

Complexity', *SAE Technical Papers*, Part F1298(September). doi: 10.4271/2017-01-1992.

Xia, Q. *et al.* (2018) 'Test Scenario Design for Intelligent Driving System Ensuring Coverage and Effectiveness', *International Journal of Automotive Technology 2018 19:4*. Springer, 19(4), pp. 751–758. doi: 10.1007/S12239-018-0072-6.

Xin, C., Dan, L. and Shuo, H. (2020) 'Research on deceleration early warning model based on V2X', *Proceedings - 2020 12th International Conference on Measuring Technology and Mechatronics Automation, ICMTMA 2020*, pp. 318–321. doi: 10.1109/ICMTMA50254.2020.00077.

Xu, W. *et al.* (2018) 'Analyzing and Enhancing the Security of Ultrasonic Sensors for Autonomous Vehicles', *IEEE Internet of Things Journal*. Institute of Electrical and Electronics Engineers Inc., 5(6), pp. 5015–5029. doi: 10.1109/JIOT.2018.2867917.

Yang, H. (2012) *Simulation-based evaluation of traffic safety performance using surrogate safety measures*. The State University of New Jersey.

Yang, H., Ozbay, K. and Bartin, B. (2010) 'Application of Simulation-Based Traffic Conflict Analysis for Highway Safety Evaluation', *Selected Proceedings of World Conference on Transport Research*, 12, p. 14pages. Available at: http://intranet.imet.gr/Portals/0/UsefulDocuments/documents/02303.pdf.

Yoshihira, M. *et al.* (2016) 'Developing an Autonomous Vehicle Control System for Intersections Using Obstacle/Blind Spot Detection Frames', in *SAE Technical Papers*. SAE International. doi: 10.4271/2016-01-0143.

Zablith, F. *et al.* (2015) 'Ontology evolution: a process-centric survey', *The Knowledge Engineering Review*. Cambridge University Press, 30(1), pp. 45–75. doi: 10.1017/S0269888913000349.

Zeng, D. *et al.* (2019) 'A novel robust lane change trajectory planning method for autonomous vehicle', *IEEE Intelligent Vehicles Symposium, Proceedings*. IEEE, 2019-June(Iv), pp. 486–493. doi: 10.1109/IVS.2019.8814151.

Zhang, C. *et al.* (2018) 'A Graded Offline Evaluation Framework for Intelligent Vehicle's Cognitive Ability', *IEEE Intelligent Vehicles Symposium, Proceedings*. IEEE, 2018-June(Iv), pp. 320–325. doi: 10.1109/IVS.2018.8500622.

Zhang, M. (2018) 'DeepRoad : GAN-Based Metamorphic Testing and Input Validation Framework for Autonomous Driving Systems', pp. 132–142.

Zhang, S. *et al.* (2018) 'Accelerated Evaluation of Autonomous Vehicles in the Lane Change Scenario Based on Subset Simulation Technique', *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. IEEE, 2018-Novem, pp. 3935–3940. doi: 10.1109/ITSC.2018.8569800.

Zhao, C. *et al.* (2020) 'Influence of Cut-In Maneuvers for an Autonomous Car on Surrounding Drivers: Experiment and Analysis', *IEEE Transactions on Intelligent Transportation Systems*. IEEE, 21(6), pp. 2266–2276. doi: 10.1109/TITS.2019.2914795.

Zhao, L. *et al.* (2015) 'Ontology-based decision making on uncontrolled intersections and narrow roads', *IEEE Intelligent Vehicles Symposium, Proceedings*. IEEE, 2015-Augus(Iv), pp. 83–88. doi: 10.1109/IVS.2015.7225667.

Zhao, Y. Q. *et al.* (2017) 'Estimation of Road Friction Coefficient in Different Road Conditions Based on Vehicle Braking Dynamics', *Chinese Journal of Mechanical Engineering (English Edition)*.

Chinese Mechanical Engineering Society, 30(4), pp. 982–990. doi: 10.1007/s10033-017-0143-z.

Zhou, J. and Del Re, L. (2018) 'Safety Verification of ADAS by Collision-free Boundary Searching of A Parameterized Catalog', *Proceedings of the American Control Conference*. Institute of Electrical and Electronics Engineers Inc., 2018-June, pp. 4790–4795. doi: 10.23919/ACC.2018.8431291.

Ziegler, J. *et al.* (2014) 'Making Bertha Drive — An Autonomous Journey on a Historic Route', pp. 8–20.

Zubinskiy, A. (2013) *The quality of embedded software, or the mess has happened*. Available at: https://pvs-studio.com/en/a/0083/ (Accessed: 7 April 2021).

Appendix

Appendix A

Section 1: Environment Layer

1. Concepts and Attributes

This section defines the terminology and attributes of each concept within the 'Environment' layer. The concepts within this layer are illustrated in Figure 4.8.

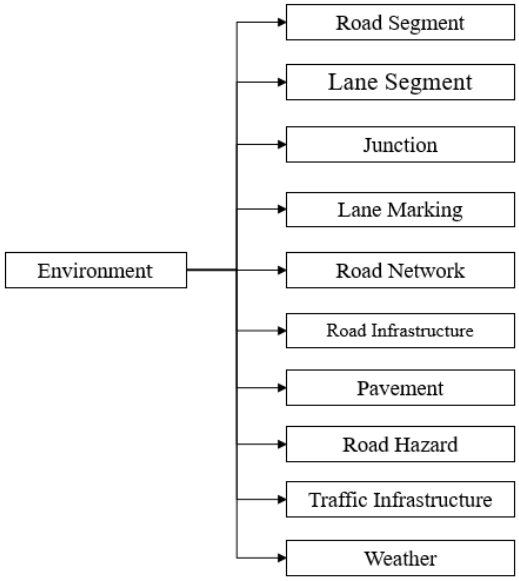


Figure 4.8: Environment Concept

1.1. Environment

Concept: environment *(Environment)*

Definition: The concept *(C)* environment *(Environment)* represents the operational surrounding. It has the following sub concepts: road segment *(RoadSegment)*, lane segment *(LaneSegment)*, junction *(Junction)*, lane marking *(LaneMarking)*, road network *(RoadNetwork)*, road infrastructure *(RoadInfrastructure)*, pavement *(Pavement)*, road hazard *(RoadHazard)*, traffic infrastructure *(TrafficInfrastructure)* and weather *(Weather)*

1.2. Road Segment

Concept: road segment *(RoadSegment)*

Definition: The concept road segment *(RoadSegment)* represents a section of the road. An abstract of this is depicted in Figure 4.9.

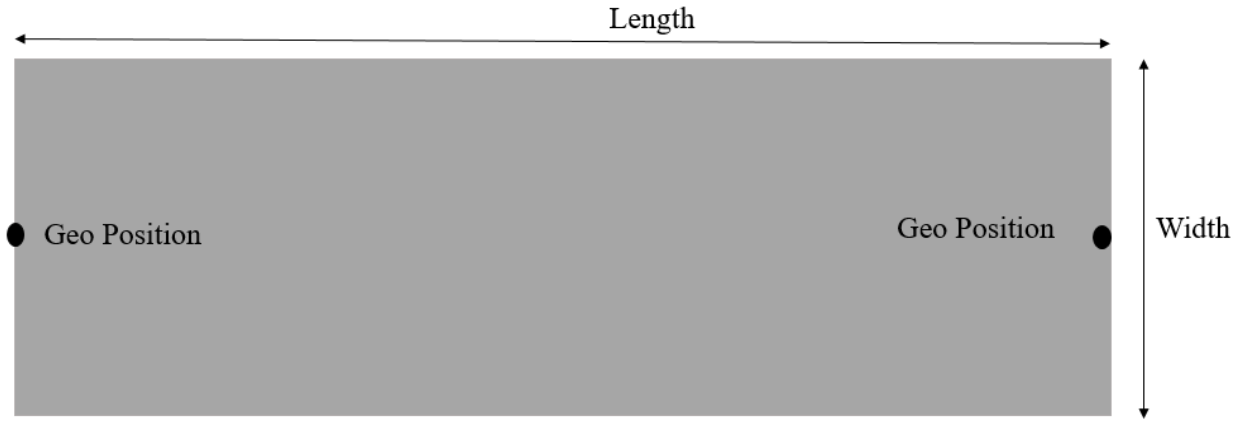


Figure 4.9: Road Segment

The concept road segment *(RoadSegment)* has the following attributes:

- *Id (id):* This attribute uniquely identifies each concept of the road segment *(RoadSegment)*.
- *Length (length):* The length is a positive scalar value defining the length of the road segment *(RoadSegment)* and the domain of value will define all possible length the concept road segment *(RoadSegment)* can have. The length is the distance between the start and the end segment. The units are in meters. The length of the road segment must be greater than or equal to the lane segment(s) it contains.
- *Width (width):* The width is a positive scalar value defining the width of the road segment and a domain of values will define all possible width for the concept road segment *(RoadSegement).* The units are in meters. The width of the road segment must be greater than or equal to the lane segment it contains.
- Number of lanes *(numberOfLanes)*: This attribute defines the number of lanes within road segment. The domain for this attribute includes one of these values:
  - An Integer, M, in the range [1,4] which creates an M-lane road
  - A two-element vector, [M, N] where M and N are positive integers whose sum must be in the range [2,4] where M defines a one way or two-way traffic and N defines the number of lane segments.

- Number of road infrastructure *(numberOfRoadInfrastructure)*: This attribute defines the number of road infrastructure available within the edges of a road segment *(RoadSegment)*. This is in the range [P, Q] where P is a positive integer for number of road infrastructure and Q is the road infrastructure *(RoadInfrastructure)* id.
- Geometry *(geometry)*: The attribute geometry *(geometry)* defines the shape of the road. The domain within this attribute includes the following:
  - Line: These include straight roads
  - Spiral: These include helical roads
  - Arc: These include curved roads
- Topography *(Topography)*: This attribute defines the topographic slope which is the tangent between two position in the road. The domains for this attribute include:
  - Slope: These include transverse slope
  - Ramp: These include longitudinal slope
  - Flat: This defines a flat road
- *Geo Position (geoPosition):* This attribute represents the centre position of the ends of the road segment with [x,y,z] coordinates as depicted in figure 4.9.
- Number of pavements *(NumberOfPavements)*: This is a positive scalar value in the range of [0,2] indicating the number of pavements within a road.
- *Speed Limit (speedLimit):* This attribute defines the maximum speed allowed on the road segment *(RoadSegment)* in the United Kingdom. The speed limit generally depends type of road and location.
- *Road Surface Condition (roadSurfaceCondition):* This represents the surface condition of the concept road segment *(RoadSegment)* and is provided by the road-friction-coefficient (μ) value. The μ is a value that ranges from [0-1]. The consequence of this value is affecting the breaking distance of a vehicle. Hence, as μ tends to 0, the length of distance travelled is longer after brake is applied. This value is influenced by the weather, Table 4.2 shows the relation of different weather conditions with is corresponding μ.

Table 4. 5: Road Friction Coefficient Values (Zhao *et al.*, 2017)

Weather Conditions	Road Friction Coefficient Values
Ice	<0.25
Snow	0.50< 0.25
Wet/moist	>0.5

Dry	>0.5

The road-friction-coefficient (μ) could have a relation with weather such that if the domain of weather is sunny than the μ value will be > 0.5 accordingly.

Within the road segment *(RoadSegment)* concept, the attributes length, width, speed limits etc. can take values from its respective domain. The combination of these values has constraints for example, the length and width of the concept road segment will take values from U.K government website.. The speed limits for all road segment will be according to the U.K road rules (20,30,50,60,70) (*Speed limits - GOV.UK*).

The concept road segment *(RoadSegment)* can hence be represented as follows:

$$C=\{RoadSegment\} \rightarrow A=\{id, length, width, numberOfLanes, geometry, topography, numberOfPavement, numberOfRoadInfrastructure, speedLimit, surfaceCondition, geoPosition\} \rightarrow D=\{ D_{id}, D_{length}, D_{width}, D_{SpeedLimit}, D_{surfaceCondition}, D_{geoPosition} D_{numberOfLanes}, D_{geometry}, D_{topography}, D_{numberOfPavement}, D_{numberOfRoadInfrastructure} \}$$

The domain *(D)* for each attribute represents all possible values within the attribute.

Hence ω can be defined as follows:

$$\omega(RoadSegment)= (id, length, width, numberOfLanes, geometry, topography, numberOfPavement, numberOfRoadInfrastructure, speedLimit, surfaceCondition, geoPosition \{ D_{id}, D_{length}, D_{width}, D_{SpeedLimit}, D_{surfaceCondition}, D_{geoPosition} D_{numberOfLanes}, D_{geometry}, D_{topography}, D_{numberOfPavement}, D_{numberOfRoadInfrastructure} \})$$

1.3. Lane Segment

Concept: lane segment *(LaneSegment)*

Definition: The concept lane segment *(LaneSegment)* represents the segmentation within a road i.e., the drive-able areas. Figure 4.10, depicts an example of this

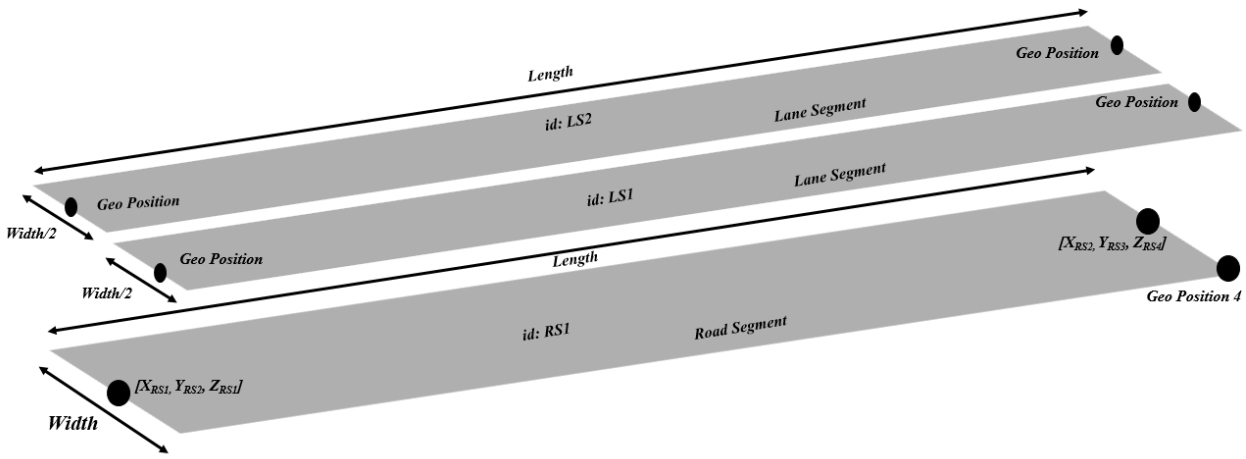


Figure 4.10: Lane Segment LS1, LS2

The attributes for the concept lane segment *(LaneSegment)* are as follows:

- *Id (id):* This attribute uniquely identifies each concept lane segment *(LaneSegment).*
- *Length (length):* The length is a positive scalar value defining the length of the lane segment *(LaneSegment)* and a domain of value will define all possible length the concept lane segment *(LaneSegment)*. The length is the distance between the start and the end of the segment. The units are in meters. The length of the lane segment must be less than or equal to the road segment it is within
- *Width (width):* The width is a positive scalar value defining the width of the lane segment and a domain of values will define all possible width for the concept lane segment *(LaneSegement).* The units are in meters. The width of the lane segment must be greater than the width of the lane marking. The width of the lane segment must be less than or equal to the width of the road segment it is within
- Type *(type)*: The domain for this attribute include:
    - Driving – lanes for driving
    - Shoulder – Lanes reserved for high occupancy vehicles
    - Parking – These are alongside driving lanes and are intended for parking
- *Geo Position (geoPosition):* This attribute represents the centre position of the end of the lane with [x,y,z] coordinates as depicted in figure 4.10.
- Heading *(heading):* The heading angle of the lane is about its x-axis at the lane centre, is in degrees and specified as a decimal scalar. This determines the direction of travel.

The concept lane segment *(LaneSegment)* can be represented as follows:

$$C=\{LaneSegment\} \rightarrow A=\{id, length, width, type, geoPosition, heading\} \rightarrow D=\{D_{id,}\ D_{length},\ D_{width},\ D_{type}\ ,\ D_{geoPosition,}\ D_{heading}\}$$

Hence ω can be defined as follows:

$$\omega(LaneSegment)=(id, length, width, type, geoPosition, heading\ \{\ D_{id,}\ D_{length},\ D_{width},\ D_{type}\ ,\ D_{geoPosition,}\ D_{heading}\ \})$$

1.4. Lane Marking

Concept: lane marking *(LaneMarking)*

Definition: The concept *(C)* lane marking *(LaneMarking)* represents the marking within a road to separate one lane from another. Figure 4.11, depicts an example of this:

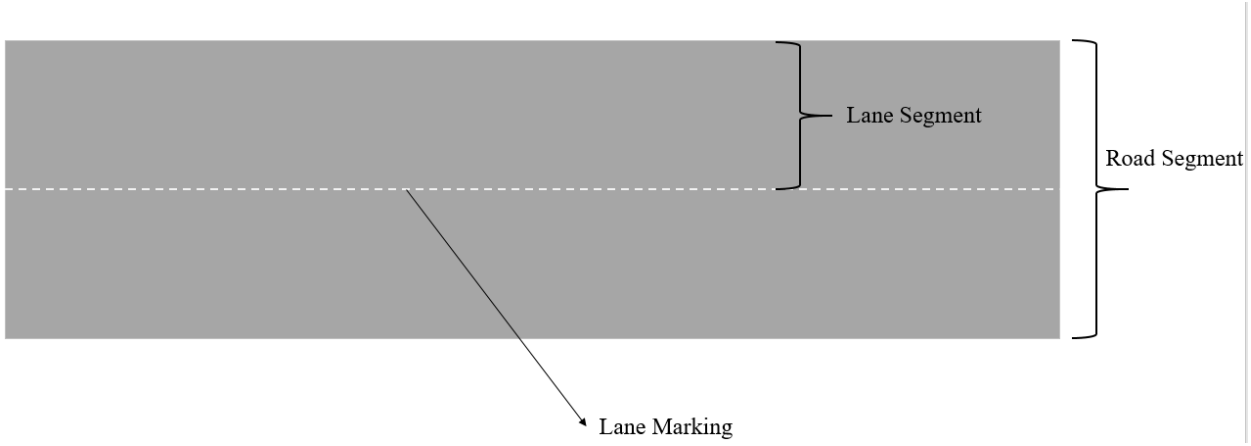


Figure 4.11: Road Segment, Lane Segment, Lane Marking

This concept lane marking *(LaneMarking)* has the following attributes:

- id *(id)*: which identifies the lane marking *(LaneMarking)*.
- Type *(type)*: This attribute defines the type of lane marking. The domains for these include:
  - Unmarked: These include no lane marking
  - Solid: These include solid line
  - Dashed: These include dashed line
  - Double Solid: These include two solid lines
  - Solid Dashed: These include two dashed lines

- Width *(width)*: This attribute defines the width of the lane marking in meters and its decimal scalar. The lane marking's width must be less than the enclosing lane segment's width. The lane directly to the left of the lane marking is the enclosing lane segment. When a lane marker specifies a double line, both lines are the same width. A default value of 0.15 is used to correspond with simulation environment, however the domain for this can be expanded.
- Length *(length)*: This attribute defines the length of the lane marking in meters and it is decimal scalar. When a lane marker specifies a double line, both lines are the same length.
- Space *(space)*: The length defines in metres of the spaces between dashes in dashed lane markings and is specified as a decimal scalar. The same space is used for both lines when a lane marker specifies a double line.
- Geo Position *(geoPosition)*: This attribute represents the centre position of the end of the lane marking with [x,y,z] coordinates.

  This can be represented as follows:

$$C=\{LaneMarking\} \rightarrow A=\{id, type, width, length, space, geoPosition\} \rightarrow D=\{D_{id}, D_{type}, D_{width}, D_{length}, D_{space}, D_{geoPosition}\}$$

Hence ω can be defined as follows:

$$\omega(LaneMarking) = (id, type, width, length, space, geoPosition, \{D_{id}, D_{type}, D_{width}, D_{length}, D_{space}, D_{geoPosition}\})$$

1.5. Junction

Concept: junction *(Junction)*

Definition: The concept junction *(Junction)* represents a road layout where two or more road segment *(RoadSegement)* meets as defined by the various junction types below. However, not all combination of road segment *(RoadSegment)* form a junction, for example, two or more road segment *(RoadSegment)* can meet where the end of one road segment *(RoadSegment)* aligns exactly with the beginning of another road segment *(RoadSegment)* and to form a continuous road layout. The junctions included within this work will be those defined in the UK highway code (*General rules, techniques and advice for all drivers and riders (103 to 158) - The Highway Code - Guidance - GOV.UK*). This concept *(C)* will focus on non-signalised junction. However, this can be expanded to include other types of junctions. Figure 4.12 depicts an example of this.

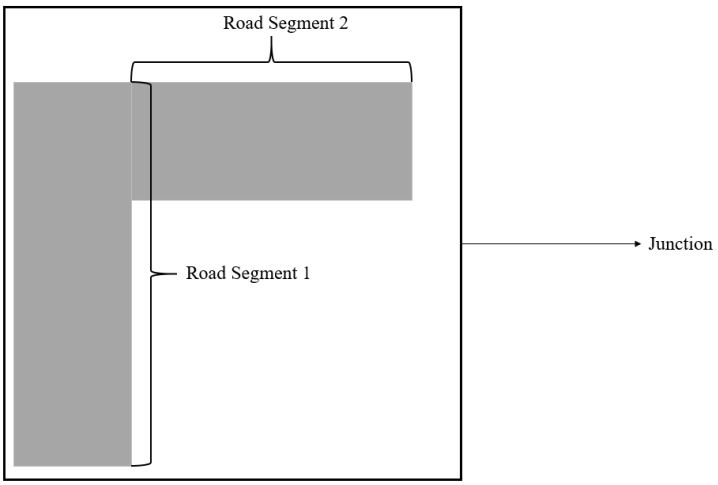


Figure 4.12: Junction Concept

This concept junction *(Junction)* has the following attributes:

- Id *(id)*: which identifies the junction *(Junction)*.
- Number of road segment *(numberOfRoadSegemnt)*: This is a positive scalar value which greater than two [≥2] indicating the number of road segments within a junction.
- Type *(type)*: This is attribute defines the type of junction. The domain of this include:
    - T-junction *(TJunction)*: These are intersections where the route you're travelling comes to an end and you must turn left or right. Their structure resembles the letter T. Figure 4.13 shows an example of a T-junction. A domain of a T- junction *(TJunction)* must include a minimum of two road segement *(RoadSegment)*

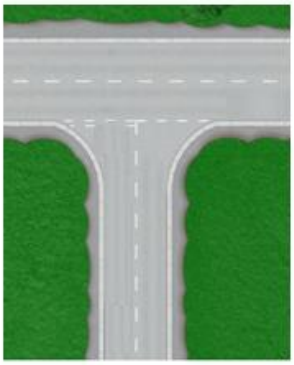


Figure 4.13: T-Junction

- X-junction *(XJunction)*: Three roads meet at the same road junction at a crossroads. Figure 4.14 shows an example of a X-Junction A domain of a X- junction *(XJunction)* must include a minimum of three road segement *(RoadSegment)*

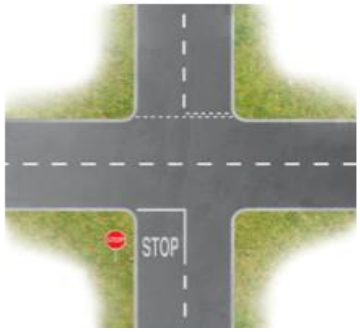


Figure 4.14: X-Junction

- Y-Junction (*YJunction)*: A Y-junction, like a T-junction, takes its name from the fact that it is formed like the letter Y. A Y-junction is where a minor road intersects with a major road at an acute angle. Y-junctions exist in a variety of shapes and sizes, some of which are misleading and surprising. As with any other form of junction, Y-junctions can be either "open" (easy to observe traffic on the opposite road) or "closed" (a blind junction concealed by greenery, fences, etc.). Figure 4.15 show example of Y-junctions. A domain of a Y-junction *(YJunction)* must include a minimum of two road segement *(RoadSegment)*

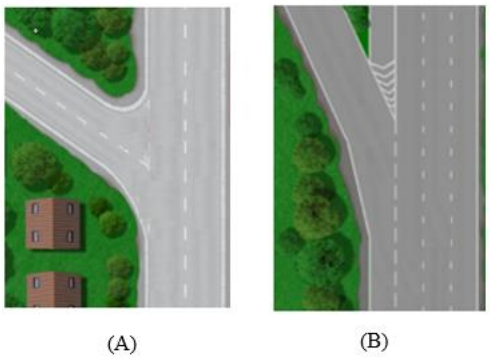


Figure 4.15: Y-Junction

- D-Junction: D junctions, acquire their name from the fact that they resemble the letter D. Within these, they are different types of junctions which include unmarked, marked, controlled and box. Figure 4.16 shows an example of a D junction. A domain of a D- junction *(DJunction)* must include a minimum of four road segement *(RoadSegment)*

Figure 4.16: D Junction

This can be represented as follows:

$$C=\{Junction\} \rightarrow A=\{id, numberOfRoadSegment, Type\} \rightarrow D=\{D_{id}, D_{numberOfRoadSegment}, D_{numberOfRoadInfrastructure}, D_{Type}\}$$

Hence ω can be defined as follows:

$$\omega(Junction) = (id, numberOfRoadSegment, Type \{ D_{id}, D_{numberOfRoadSegment}, D_{numberOfRoadInfrastructure}, D_{Type} \})$$

1.6. Road Network

Concept: road network *(RoadNetwork)*

Definition: The concept *(C)* road network *(RoadNetwork)* represents different structure of roads e.g., junction or highway connected together to form a network. This concept *(C)* has the following attributes

- Id *(id)*: This uniquely identifies a road network
- Number of road segment *(numberOfRoadSegment)*: This is a positive scalar value in the range of equal to or greater than two [<2] indicating the number of road segments within a road network.
- Number of junction *(numberOfJunction)*: This is a positive scalar value indicating the number of junctions within a road network.

$$C=\{RoadNetwork\} \rightarrow A=\{id, numberOfRoadSegment, numberOfJunction\} \rightarrow D=\{D_{id}, D_{numberOfRoadSegment}, D_{numberOfJunction}\}$$

Hence ω can be defined as follows:

$$\omega(RoadNetwork) = (id, numberOfRoadSegment, numberOfJunction \{D_{id}, D_{numberOfRoadSegment}, D_{numberOfJunction}\})$$

1.7. Road Infrastructure

Concept: road infrastructure *(RoadInfrastructure)*

Definition: The concept *(C)* road infrastructure *(RoadInfrastructure)* represents fixed assets and structures outside the area of the road used for travelling. This concept *(C)* currently has two sub concepts *(C)* this includes: tree *(Tree)* and building *(Building)*. This can be extended to include other types of road infrastructure *(RoadInfrastructure)* for example, bus stands, bench etc. Attributes of road infrastructure *(RoadInfrastructure)*:

- Id *(id):* This attribute uniquely identifies each domain of the concept road infrastructure *(RoadInfrastructure)*.
- Name *(name)*: This attribute references the name of the road infrastructure *(RoadInfrastructure)*.
- Length *(length)*: The length is a decimal scalar defined in meters. value defining the length of the road infrastructure *(RoadInfrastructure)*
- Width *(width):* The width of the road infrastructure *(RoadInfrastructure)* is specified in meters as a decimal scalar.
- Height *(height)*: The height is specified in meters as a decimal scalar value for defining the concept road infrastructure *(RoadInfrastructure)*
- Segment gap *(segmentGap)*: This attribute defines the gap between consecutive road infrastructure *(RoadInfrastructure)*, is a decimal scalar defined in meter in the range [0, Segment Length]
- *Geo Position (geoPosition):* This attribute defines the [x,y,z] position for the centre of the road infrastructure *(RoadInfrastructure)*.
- Type *(type)*: This attribute indicates the type of road infrastructure *(RoadInfrustructure)*. The domain of this concept include:
    - Building
    - Tree

  The domain for this attribute can be increased for future work
- Road segment edge offset *(roadSegmentEdgeOffset)*: This attribute defines the distance at which the road infrastructure *(RoadInfrastructure)*is offset from the road edge in the lateral direction. This is defined in meters and is specified as a decimal scalar.

For this research we will consider two types to correspond with the simulation environment, however the domain can be expanded.

This can be represented as follows:

$$C=\{RoadInfrastructure\} \rightarrow A= \{id, name, segmentLength, width, height, segmentGap, geoPosition, type, roadSegmentOffSet\} \rightarrow D= \{ D_{id}, D_{name}, D_{segmentLength}, D_{width}, D_{height}, D_{segmentGap}, D_{geoPosition}, D_{type}, D_{roadSegmentOffSet}\}$$

Hence ω can be defined as follows:

$$\omega(RoadInfrastructure) = (id, name, segmentLength, width, height, segmentGap, geoPosition, type, roadSegmentOffSet \{ D_{id}, D_{name}, D_{segmentLength}, D_{width}, D_{height}, D_{segmentGap}, D_{geoPosition}, D_{type}, D_{roadSegmentOffSet} \})$$

1.8. Pavement

Concept: pavement *(Pavement)*

Definition: The concept *(C)* pavement *(Pavement)* represents a raised paved path on the side of the road which can be used as a walkway. This concept *(C)* has the following attributes (A):

- *Id (id):* This attribute uniquely identifies each concept pavement *(Pavement)*.
- *Length (length):* The length is a positive scalar value defining the length of the concept pavement *(Pavement)*, The length is the distance between the start and the end segment. The units are in meters.
- *Width (width):* The width is a positive scalar value defining the width of the concept pavement *(Pavement).* The units are in meters.
- *Height (height)*: The height is a positive scalar value defining the width of the concept pavement *(Pavement).* The units are in meters.
- *Geo Position (geoPosition):* This attribute represents the centre position of the end of the pavement with [x,y,z] coordinates .

This can be represented as follows:

$$C= \{Pavement\} \rightarrow A= \{Id, Length, Width, Height, GeoPosition\} \rightarrow D= \{D_{Id}, D_{Length}, D_{Width}, D_{Height}, D_{GeoPosition}\}$$

Hence ω can be defined as follows:

$$\omega(Pavement) = (GeoPosition, Length, Width, Height \{ D_{Id}, D_{Length}, D_{Width}, D_{Height}, D_{GeoPosition} \})$$

1.9. Road Hazard

Concept: road hazard *(RoadHazard)*

Definition: The concept *(C)* road hazard *(RoadHazard)* represents different types of hazards that may be present on the drivable road. These hazards can be temporary, for example oil spill or can be long lasting which are normally a result of the road surface becoming damaged due to wear and distress, which can be caused by adverse weather, heavy truck traffic etc. The attributes for this concept include:

- *Id (id):* This attribute uniquely identifies each instance of road hazard *(RoadHazard)*.
- Length *(length):* The length is a positive scalar value defining the length of the concept road hazard *(RoadHazard)* along the length of the road segment. The units are in meters.
- Width *(width)*: The width is a positive scalar value defining the width of the concept road hazard *(RoadHazard*) along the width of the road segment. The units are in meters.
- Depth *(depth)*: The width is a positive scalar value defining the width of the concept road hazard *(RoadHazard)*. The units are in meters.
- Area *(area)*: The width is a positive scalar value defining the width of the concept road hazard *(RoadHazard)*. The units are in meters.
- *Geo Position (geoPosition):* This attribute represents the centre position of the road hazard [x,y,z] coordinates.
- Type *(type)*: This attribute defines the type of road hazard *(RoadHazard)*. The domains for this attribute can increase with future work. However, they currently include:
    - Pothole: This domain defines a hazard which has been caused due to heavy rain over time.

This can be represented as follows:

$$C=\{RoadHazard\} \rightarrow D=\{ Id, length, width, depth\ area, geoPosition, type \} \rightarrow A=\{ D_{id}, D_{length}, D_{width}, D_{depth}, D_{area}, D_{geoPosition}, D_{type} \}$$

Hence ω can be defined as follows:

$$\omega(RoadHazard) = (\{ Id, length, width, depth\ area, geoPosition, type \{ D_{id}, D_{length}, D_{width}, D_{depth}, D_{area}, D_{geoPosition}, D_{type} \})$$

1.10. Traffic Infrastructure

Concept: traffic infrastructure *(TrafficInfrastructure)*

Definition: The concept *(C)* traffic infrastructure *(TrafficInfrastructure)* are structures which are used to inform or guide traffic. This concept *(C)* has one domain traffic light *(TrafficLight)* ; however, this can be expanded to include other sub concepts *(C)* like traffic signs. This concept *(C)* has the following attributes:

- *Id (id):* This attribute uniquely identifies each concept traffic infrastructure *(TrafficInfrastructure)*.
- *Length (length):* The length is a positive scalar value defining the length of the concept traffic infrastructure *(TrafficInfrastructure)*. The units are in meters.
- *Width (width):* The width is a positive scalar value defining the width of the concept traffic infrastructure *(TrafficInfrastructure)*. The units are in meters.
- *Height (height)*: The height is a positive scalar value defining the height of the concept traffic infrastructure *(TrafficInfrastructure)*. The units are in meters.
- *Geo Position (geoPosition):* This attribute defines the [x,y,z] position for the centre of the traffic infrastructure *(TrafficInfrastructure)*.
- Type *(type)*: This attribute defines the type of traffic infrastructure *(TrafficInfrastructure)*. The current domains for this concept include:
    - Traffic Lights

This can be represented as follows:

$$C=\{TrafficInfrastructure\} \rightarrow A=\{ id, length, width, height, geoPosition, type \} \rightarrow \quad D=\{D_{id}, D_{length}, D_{width}, D_{height}, D_{geoPosition}, D_{type} \}$$

Hence ω can be defined as follows:

$$\omega(TrafficInfrastructure) = (id, length, width, height, geoPosition, type \{ D_{id}, D_{length}, D_{width}, D_{height}, D_{geoPosition}, D_{type} \})$$

1.11. Weather

Concept: Weather *(Weather)*

Definition: The concept weather *(Weather)* represents the current weather within a scenario. This concept includes on attribute, however, can be expanded in future works. The attributes within this concept include:

- Type *(type)*: This attribute defines the type of weather. The current domains include:

- **Sun:** This domain would result in the domain for *Road Surface Condition (roadSurfaceCondition)* to be > 0.5
- **Rain:** This domain would result in the domain for *Road Surface Condition (roadSurfaceCondition)* to be > 0.5
- **Snow:** This domain would result in the domain for *Road Surface Condition (roadSurfaceCondition)* to be 0.50< 0.25

This can be represented as follows:

$$C=\{Weather\} \rightarrow A=\{type\} \rightarrow \quad D=\{Sun, Rain, Snow\}$$

Hence ω can be defined as follows:

$$\omega(Weather) = (type\ \{Sun, Rain, Snow\})$$

Section 2: Road User Layer

2. Concepts and Attributes

This section defines the terminology and attributes of each concept within the 'Road User' layer

2.1. Road User

Concept: road user *(RoadUser)*

Definition: The concept *(C)* road user *(RoadUser)* represents the different actors and participants, their attributes, and relations within a scenario. This concept has the following sub-concepts Connected AV *(ConnectedAV)*, car *(Car)*, pedestrian *(Pedestrian)*, cyclist *(Cyclist)*

2.2. Connected AV

Concept: Connected AV *(ConnectedAV)*

Definition: The concept connected AV *(ConnectedAV)* represents a connected autonomous vehicle that is capable of sensing its environment and communicating with other road users or traffic infrastructure and moving safely with no human input. This concept can be further extended to different vehicle of different make, model etc. However, for the purpose of this thesis will have the same dynamic model. The attributes for this concept are as follows:

- Code name *(codeName):* Uniquely identifies a connectedAV *(ConnectedAV)*
- Length *(length)*: This attribute defines the length of connectedAV *(ConnectedAV)* is a decimal scalar defined in meter in the range [0,60]. It must be greater than (Front Overhang + Rear Overhang)
- Width *(width)*: This attribute defines the width of connectedAV is a decimal scalar defined in meter in the range [0,20]
- Height *(height)*: This attribute defines the height of connectedAV is specified as a decimal scalar, in meters, in the range [0,20]
- Front Overhang *(frontOverhang)*: This attribute defines the distance between the front axle and front bumper, is a decimal scalar defined in meters. The front overhang must be less than (Length – Rear Overhang).

- Rear Overhang *(rearOverhang)*: This attribute defines the distance between the rear axle and rear bumper, and is a decimal scalar defined in meters. The rear overhang must be less than (Length – Front Overhang)
- Speed *(speed)*: Defines a value of the rate at which a ConnectedAV *(ConnectedAV)* is travelling across a road network *(RoadNetwork)*. The units for speed are in meters per second (m/s)
- Roll *(roll)*: This attribute defines the orientation angle about its x-axis, in degrees and is specified as a decimal scalar.
- Pitch *(pitch)*: This attribute defines the orientation angle about its y-axis, in degrees and is specified as a decimal scalar.
- Yaw *(yaw)*: This attribute defines the orientation angle about its z-axis, in degrees and is specified as a decimal scalar.
- Position *(position)*: This attribute defines the centre position of the connectedAV *(ConnectedAV)* in the form [x,y,z]
- Waypoints *(wayPoints)*: This attribute defines a matrix of all position the connectedAV *(ConnectedAV))* will navigate through. This is in the form of $\begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_n & y_n & z_n \end{bmatrix}$ where $[x_1, y_1, z_1]$ defines the first position and $[x_n, y_n, z_n]$ defines the last position within a test scenario.

This can be represented as follows:

$$C = \{ConnectedAV\} \rightarrow A = \{codeName, length, width, height, frontOverhang, rearOverhang, speed, roll, pitch, yaw, position, wayPoints\} \rightarrow D = \{ D_{codeName}, D_{length}, D_{width}, D_{height}, D_{frontOverhang}, D_{rearOverhang}, D_{speed}, D_{roll}, D_{pitch}, D_{yaw}, D_{position}, D_{wayPoints}\}$$

$$\omega(ConnectedAV) = (codeName, length, width, height, frontOverhang, rearOverhang, speed, roll, pitch, yaw, position, wayPoints, \{ D_{codeName}, D_{length}, D_{width}, D_{height}, D_{frontOverhang}, D_{rearOverhang}, D_{speed}, D_{roll}, D_{pitch}, D_{yaw}, D_{position}, D_{wayPoints} \})$$

2.3. Ego Vehicle

Concept: ego vehicle *(EgoVehicle)*

Definition:

The concepts *(C)* ego vehicle *(EgoVehicle)* is a sub-concept of autonomous vehicle *(AutonomousVehicle)* represents the subject vehicle and is the vehicle under test. This concept can be further extended to include testing more than one vehicle within a scene. The attributes for this concept are as follows:

- Code name *(codeName):* Uniquely identifies an ego vehicle *(EgoVehicle)* this attribute has one domain [EgoVehicle]
- Length *(length)*: This attribute defines the length of ego vehicle *(EgoVehicle)* is a decimal scalar defined in meter in the range [0,60]. It must be greater than (Front Overhang + Rear Overhang)
- Width *(width)*: This attribute defines the width of ego vehicle *(EgoVehicle)* is a decimal scalar defined in meter in the range [0,20]
- Height *(height)*: This attribute defines the height of ego vehicle *(EgoVehicle)* is specified as a decimal scalar, in meters, in the range [0,20]
- Front Overhang *(frontOverhang)*: This attribute defines the distance between the front axle and front bumper, is a decimal scalar defined in meters. The front overhang must be less than (Length – Rear Overhang).
- Rear Overhang *(rearOverhang)*: This attribute defines the distance between the rear axle and rear bumper, and is a decimal scalar defined in meters. The rear overhang must be less than (Length – Front Overhang)
- Speed *(speed)*: Defines a value of the rate at which an ego vehicle *(EgoVehicle)* is travelling across a road network *(RoadNetwork)*. The units for speed are in meters per second (m/s)
- Roll *(roll)*: This attribute defines the orientation angle about its x-axis, in degrees and is specified as a decimal scalar.
- Pitch *(pitch)*: This attribute defines the orientation angle about its y-axis, in degrees and is specified as a decimal scalar.
- Yaw *(yaw)*: This attribute defines the orientation angle about its z-axis, in degrees and is specified as a decimal scalar.
- Position *(position)*: This attribute defines the centre position of the ego vehicle *(EgoVehicle)* in the form [x,y,z]
- Waypoints *(wayPoints)*: This attribute defines a matrix of all position the ego vehicle *(EgoVehicle)* will navigate through. This is in the form of $\begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_n & y_n & z_n \end{bmatrix}$ where $[x_1, y_1, z_1]$ defines the first position and $[x_n, y_n, z_n]$ defines the last position

This can be represented as follows:

$$C=\{ EgoVehicle \} \rightarrow A= \{codeName, length, width, height, frontOverhang, rearOverhang, speed, roll, pitch, yaw, position, wayPoints\} \rightarrow D= \{ D_{codeName}, D_{length}, D_{width}, D_{height}, D_{frontOverhang}, D_{rearOverhang}, D_{speed}, D_{roll}, D_{pitch}, D_{yaw}, D_{position}, D_{wayPoints}\}$$

$$\omega(EgoVehicle) = (codeName, length, width, height, frontOverhang, rearOverhang, speed, roll, pitch, yaw, position, wayPoints, \{ D_{codeName}, D_{length}, D_{width}, D_{height}, D_{frontOverhang}, D_{rearOverhang}, D_{speed}, D_{roll}, D_{pitch}, D_{yaw}, D_{position}, D_{wayPoints} \})$$

2.4. Car

Concept: car *(Car)*

Definition: The concept *(C)* car *(Car)* represents a four-wheeled road vehicle that is powered by an engine just like the concept *(C)* autonomous vehicle *(AutonomousVehicle)*. However, a car would require human input to mobilize. This concept can be further be extended to different vehicle of different make, model, engine size etc. However, for the purpose of this thesis, there will one dynamic model. The attributes for this concept are as follows:

- Code name *(codeName):* Uniquely identifies a car *(Car)*
- Length *(length)*: This attribute defines the length of a car *(Car)* is a decimal scalar defined in meter in the range [0,60]. It must be greater than (Front Overhang + Rear Overhang)
- Width *(width)*: This attribute defines the width of a car *(Car)* is a decimal scalar defined in meter in the range [0,20]
- Height *(height)*: This attribute defines the height of a car *(Car)* is specified as a decimal scalar, in meters, in the range [0,20]
- Front Overhang *(frontOverhang)*: This attribute defines the distance between the front axle and front bumper, is a decimal scalar defined in meters. The front overhang must be less than (Length – Rear Overhang).
- Rear Overhang *(rearOverhang)*: This attribute defines the distance between the rear axle and rear bumper, and is a decimal scalar defined in meters. The rear overhang must be less than (Length – Front Overhang)
- Speed *(speed)*: Defines a value of the rate at which a car *(Car)* is travelling across a road network *(RoadNetwork)*. The units for speed are in meters per second (m/s)
- Roll *(roll)*: This attribute defines the orientation angle about its x-axis, in degrees and is specified as a decimal scalar.
- Pitch *(pitch)*: This attribute defines the orientation angle about its y-axis, in degrees and is specified as a decimal scalar.
- Yaw *(yaw)*: This attribute defines the orientation angle about its z-axis, in degrees and is specified as a decimal scalar.

- Position *(position)*: This attribute defines the centre position of the car *(Car)* in the form [x,y,z]
- Waypoints *(wayPoints)*: This attribute defines a matrix of all position the car *(Car)* will navigate through. This is in the form of $\begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_n & y_n & z_n \end{bmatrix}$ where $[x_1, y_1, z_1]$ defines the first position and $[x_n, y_n, z_n]$ defines the last position

This can be represented as follows:

$$C=\{Car\} \rightarrow A=\{codeName, length, width, height, frontOverhang, rearOverhang, speed, roll, pitch, yaw, position, wayPoints\} \rightarrow D=\{ D_{codeName}, D_{length}, D_{width}, D_{height}, D_{frontOverhang}, D_{rearOverhang}, D_{speed}, D_{roll}, D_{pitch}, D_{yaw}, D_{position}, D_{wayPoints}\}$$

$$\omega(Car) = (codeName, length, width, height, frontOverhang, rearOverhang, speed, roll, pitch, yaw, position, wayPoints,\{ D_{codeName}, D_{length}, D_{width}, D_{height}, D_{frontOverhang}, D_{rearOverhang}, D_{speed}, D_{roll}, D_{pitch}, D_{yaw}, D_{position}, D_{wayPoints} \})$$

2.5. Pedestrian

Concept: pedestrian *(Pedestrian)*

Definition: The concept *(C)* pedestrian *(Pedestrian)* represents a person walking rather than traveling in a vehicle. . The attributes for this concept are as follows:

- Code name *(codeName):* Uniquely identifies a pedestrian *(Pedestrian)*
- Length *(length)*: This attribute defines the length of pedestrian *(Pedestrian)* in meters and is specified as a decimal scalar.
- Width *(width)*: This attribute defines the width of an actor in meters and is specified as a decimal scalar.
- Height *(height)*: This attribute defines the height of an actor in meters and is specified as a decimal scalar.
- Speed *(speed)*: Defines a value of the rate at which a pedestrian *(Pedestrian)* is travelling across a road network *(RoadNetwork)*. The units for speed are in meters per second (m/s)
- Roll *(roll)*: This attribute defines the orientation angle about its x-axis, in degrees and is specified as a decimal scalar.
- Pitch *(pitch)*: This attribute defines the orientation angle about its y-axis, in degrees and is specified as a decimal scalar.

- Yaw *(yaw)*: This attribute defines the orientation angle about its z-axis, in degrees and is specified as a decimal scalar.
- Position *(position)*: This attribute defines the centre position of the pedestrian *(Pedestrian)* in the form [x,y,z]
- Waypoints *(wayPoints)*: This attribute defines a matrix of all position the pedestrian *(Pedestrian)* will navigate through. This is in the form of $\begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_n & y_n & z_n \end{bmatrix}$ where $[x_1, y_1, z_1]$ defines the first position and $[x_n,y_n,z_n]$ defines the last position

This can be represented as follows:

$$C=\{Pedestrian\} \rightarrow A=\{codeName, length, width, height, speed, roll, pitch, yaw, position, wayPoints\} \rightarrow D=\{D_{codeName}, D_{length}, D_{width}, D_{height}, D_{speed}, D_{roll}, D_{pitch}, D_{yaw}, D_{position}, D_{wayPoints}\}$$

$$\omega(Pedestrian) = (codeName, length, width, height, speed, roll, pitch, yaw, position, wayPoints, \{D_{codeName}, D_{length}, D_{width}, D_{height}, D_{speed}, D_{roll}, D_{pitch}, D_{yaw}, D_{position}, D_{wayPoints}\})$$

2.6. Cyclist

Concept: cyclist *(Cyclist)*

Definition: The concept*(C)* cyclist *(Cyclist)* represents a person riding a bicycle. The attributes within this concept are as follows:

- Code name *(codeName):* Uniquely identifies a cyclist *(Cyclist)*
- Length *(length)*: This attribute defines the length of cyclist *(Cyclist)* in meters and is specified as a decimal scalar.
- Width *(width)*: This attribute defines the width of a cyclist *(Cyclist)* in meters and is specified as a decimal scalar.
- Height *(height)*: This attribute defines the height of a cyclist *(Cyclist)* in meters and is specified as a decimal scalar.
- Speed *(speed)*: Defines a value of the rate at which a cyclist *(Cyclist)*is travelling across a road network *(RoadNetwork)*. The units for speed are in meters per second (m/s)
- Roll *(roll)*: This attribute defines the orientation angle about its x-axis, in degrees and is specified as a decimal scalar.
- Pitch *(pitch)*: This attribute defines the orientation angle about its y-axis, in degrees and is specified as a decimal scalar.

- Yaw *(yaw)*: This attribute defines the orientation angle about its z-axis, in degrees and is specified as a decimal scalar.
- Position *(position)*: This attribute defines the centre position of the cyclist *(Cyclist))* in the form [x,y,z]
- Waypoints *(wayPoints)*: This attribute defines a matrix of all position the pedestrian cyclist *(Cyclist)* will navigate through. This is in the form of $\begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_n & y_n & z_n \end{bmatrix}$ where $[x_1, y_1, z_1]$ defines the first position and $[x_n,y_n,z_n]$ defines the last position

This can be represented as follows:

$$C=\{ Cyclist \} \rightarrow A= \{codeName, length, width, height, speed, roll, pitch, yaw, position, wayPoints\} \rightarrow D= \{ D_{codeName}, D_{length}, D_{width}, D_{height}, D_{speed}, D_{roll}, D_{pitch}, D_{yaw}, D_{position}, D_{wayPoints}\}$$

$$\omega (Cyclist) = (codeName, length, width, height, speed, roll, pitch, yaw, position, wayPoints,\{ D_{codeName}, D_{length}, D_{width}, D_{height}, D_{speed}, D_{roll}, D_{pitch}, D_{yaw}, D_{position}, D_{wayPoints} \})$$

2.7. Road User Parts

Concept: Road User Parts *(RoadUserParts)*

Definition: The road user parts *(RoadUserParts)* concept represents and extension to the concepts within road user *(RoadUser)*. The sub concepts within this concept include: OBU *(OBU)* and sensor *(Sensor)*

$$C_{RoadUserPart}=[OBU, Sensors]$$

2.7.1. OBU

Concept: OBU *(OBU)*

Definition: This concept represents the system (hardware and software) within which information from the sensors and communication network are combined together to take a cohesive picture of the surrounding. This concept has the following sub concepts:

$$C_{OBU} = [CommunicationBit , SensorData]$$

2.7.1.1. Communication Bit

Concept: Communication Bit *(CommunicationBit)*

Definition: This concept represents the communication data the vehicle will be transmitting. The attributes of this concept are as follows:

- Position *(position)*: This attribute represents the current position of the vehicle transmitting the data.
- Heading *(heading)*: This attribute defines the direction of in which the vehicle is at
- Time *(time)*: This attribute the time at which this information is sent
- Trans receive *(transRecieve)*: This attribute defines the type of data, whether it is data which needs to be transmitted or received. The domains for this attribute include:
    - Transmission
    - Receive

2.7.1.2. Sensor Data

Concept: sensor data *(SensorData)*

Definition: This concept represents the data acquired by the sensors. The attributes for this concept include:

- Type *(type)*: This attribute defines they type of data detected for example a building, a pedestrian etc.
- Position *(position)*: This defines the position of the detected object.
- Object type *(objectType)*: This attribute defines the type of object detected for example, fixed object or dynamic object (from layer 3)
- Speed*(speed)*: The speed of the detected object, if the object is not adynamic object the domain for this attribute will be zero:
- Heading *(heading)*: The heading of the object. If the object is not dynamic this attribute will be null.
- Time *(time)*: This attribute references the time of detection

2.7.2. Sensors

Concept: sensors *(Sensors)*

Definition: The concepts*(C)* sensor *(Sensors)* represents the various sensors present in an autonomous vehicle for example, radar, lidar, GPS etc. This concept can be further extended to include each of the sensor as a sub concept. Currently the concepts sensors have the following sub-concepts:

$C_{sensors}$=[*SensorInfo, SensorPlacement, Camera, Radar, Lidar, INS, GPS]*

2.7.2.1. Sensor Info

Concept: sensor info *(SensorInfo)*

Definition: This concept defines the information of a sensor.

The attributes within this concept are as follows:

- Id *(id)*: This attribute uniquely identifies each sensor
- Sensor name *(sensorName)*: This attribute defines the name of the sensor
- Update intervals *(updateIntervals)*: This attribute defines the frequency at which the sensor is updated and is specified as an integer.
- Sensor type *(sensorType)*: This defines the type of sensor and the domains for this concept include Radar, Vision, Lidar, INS, GPS

This can be represented as follows:

$$C=\{SensorInfo\} \rightarrow A= \{sensorName, updateInterval, sensorType\} \rightarrow D= \{ D_{sensorName}, D_{updateInterval}, D_{sensorType}\}$$

$$\omega(SensorInfo) = (sensorName, updateInterval, sensorType ,\{ D_{sensorName}, D_{updateInterval}, D_{sensorType} \})$$

2.7.2.2. Sensor Placement

Concept: sensor placement *(SensorPlacement)*

Definition:

This concept defines the position and orientation of a sensor within a vehicle. This does not include sensor which have a constant change in orientation. The attribute of the sensors are as follows:

- X *(x)*: This attribute defines the position of the sensor in x-axis in the vehicle coordinate system. This has been defined in meters and is specified as a decimal scalar. The location of origin is the centre of the vehicle axis and points forward for the vehicle at the X axis.
- Y *(y)*: This attribute defines the position of the sensor in y-axis in the vehicle coordinate system. This has been defined in meters and is specified as a decimal scalar. The location of origin is the centre of the vehicle axis and points left for the vehicle at the Y axis.
- Height *(height)*: This attribute defines the height of the sensor above the ground, in meters, and is specified as a positive decimal scalar.

- Roll *(roll)*: This attribute defines the orientation angle of the sensor in degrees about its x-axis and is specified as a decimal scall. The roll is clockwise- positive when looking in the forward direction of the x-axis which points forward from the sensor.
- Pitch *(pitch)*: This attribute defines the orientation angle of the sensor in degrees about its y-axis and is specified as a decimal scall. The pitch is clockwise- positive when looking in the forward direction of the y-axis which points to the left of the sensor.
- Yaw *(yaw)*: This attribute defines the orientation angle of the sensor in degrees about its z-axis and is specified as a decimal scall. The point up from the ground is a clockwise- positive roll when observing the z axis in the forward orientation.

This can be represented as follows:

$$C=\{SensorPlacement\} \rightarrow A=\{x, y, height, roll, pitch, yaw\} \rightarrow D=\{ D_x, D_y, D_{height}, D_{roll}, D_{pitch}, D_{yaw} \}$$

$$\omega(SensorPlacement) = (x, y, height, roll, pitch, yaw,\{ D_x, D_y, D_{height}, D_{roll}, D_{pitch}, D_{yaw} \})$$

2.7.2.3. Camera

Concept: camera *(Camera)*

Definition: This concept represents the camera sensor within a connected AV. This concept has the following sub concepts. This is shown in Figure 4.19

$$[Camera] = [CameraParameters, CameraDetection, CameraLimit]$$

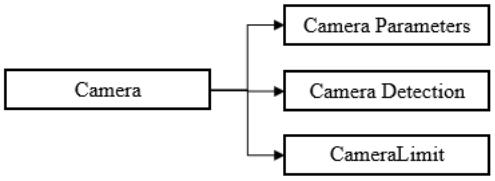


Figure 4.19: Camera Concept

2.7.2.3.1. Camera Parameters

Concept: camera parameters *(CameraParameters)*

Definition: This concept camera parameters *(CameraParameters)* define the parameters of the camera sensor. This concept has the following attributes:

- Focal length X (*focalLengthX)*: Depending on the location of sensor on the ego vehicle, the default focal length varies. Pixels is listed to attribute the definition for the focus of the camera at a horizontal point and is identified as a positive decimal scalar.
- Focal length Y *(focalLengthY)*: Depending on the location of sensor on the ego vehicle, the default focal length varies. Pixels is listed to attribute the definition for the focus of the camera at a vertical point and is identified as a positive decimal scalar

- Image width *(imageWidth)*: Horizontal camera resolution, in pixels, specified as a positive integer.
- Image height *(imageHeight)*: Vertical camera resolution, in pixels, specified as a positive integer.
- Principal point X *(principalPointX)*: Horizontal image center, in pixels, specified as a positive decimal scalar.
- Principal point Y *(principalPointY)*: Vertical image center, in pixels, specified as a positive decimal scalar.

This can be represented as follows:

$C = \{CameraParameter\} \rightarrow A = \{ focalLengthX, focalLengthY, imageWidth, imageHeight, principalPointX, principalPointY \} \rightarrow D = \{ D_{focalLengthX}, D_{focalLengthY}, D_{imageWidth}, D_{imageHeight}, D_{principalPointX}, D_{principalPointY} \}$

$\omega(CameraParameter) = (focalLengthX, focalLengthY, imageWidth, imageHeight, principalPointX, principalPointY, \{ D_{focalLengthX}, D_{focalLengthY}, D_{imageWidth}, D_{imageHeight}, D_{principalPointX}, D_{principalPointY} \})$

2.7.2.3.2. Camera Detection

Concept: camera detection *(CameraDetection)*

Definition:

This concept has the following attributes:

- Detection type *(detectionType)*: This attribute defines the type of detection reported by the camera and the domain is as follows:
  - Road Segment
  - Lane Segment
  - Lane Marking
  - Road Infrastructure
  - Traffic Infrastructure
  - Road User
- Detection probability *(detectionProbability)*: In the range of (0, 1) which is listed as a decimal scalar for the object's probability which is detected by the camera
- False Positive Per Image *(falsePositivesPerImage)*: This attribute defines the number of false reported at each update interval, this is specified as a non-negative decimal scalar. This value must be less than or equal to the maximum number of detections specified in the Limit no. of Detections attribute.

This can be represented as follows:

$$C= \{CameraDetection\} \rightarrow A= \{detectionType, detectionProbability, falsePositivesPerImage\} \rightarrow D= \{ D_{detectionType}, D_{detectionProbability}, D_{falsePositivesPerImage},\}$$

$$\omega(CameraDetection) = (\{ detectionType, detectionProbability, falsePositivesPerImage,\{ D_{detectionType}, D_{detectionProbability}, D_{falsePositivesPerImage}\})$$

2.7.2.3.3. Camera Limits

Concept: camera Limits *(CameraLimits)*

Definition:

This concept has the following attributes:

- Max speed *(maxSpeed)*: This attribute defines the fastest speed the camera can detect an object. This is specified as a nonnegative decimal scalar.
- Max range *(maxRange)*: This attribute defines the farthest distance at which a camera can detect objects. This is a positive decimal scalar and specified in meters,
- Max allowed occlusion *(maxAllowedOcclusion)*: This attribute defines the maximum percentage an object can still blocked while being detected, specified as a decimal scalar in the range [0, 1].
- Min object image width *(minObjectImageWidth)*: This attribute defines the minimum horizontal size a camera can detect of objects in pixels and is specified as positive decimal scalar.
- Min object image height *(minObjectImageHeight)*: This attribute defines the minimum vertical size that a camera can detect of objects, in pixels, and is specified as positive decimal scalar.

This can be represented as follows:

$$C= \{ CameraLimits \} \rightarrow A= \{ maxSpeed, maxRange, maxAllowedOcclusion, minObjectImageWidth, minObjectImageHeight \} \rightarrow D= \{ D_{maxSpeed}, D_{maxRange}, D_{maxAllowedOcclusion}, D_{minObjectImageWidth}, D_{minObjectImageHeight} \}$$

$$\omega(CameraLimits) = (\{ maxSpeed, maxRange, maxAllowedOcclusion, minObjectImageWidth, minObjectImageHeight,\{ D_{maxSpeed}, D_{maxRange}, D_{maxAllowedOcclusion}, D_{minObjectImageWidth}, D_{minObjectImageHeight} \})$$

2.7.2.4. Radar

Concept: radar *(Radar)*

Definition:

This concept has the following sub concepts. This is shown in figure 4.20

[Radar]= [RadarParameters, RadarDetection, RadarLimit]

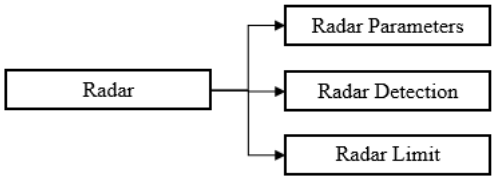


Figure 4.20: Radar Concept

2.7.2.4.1. Radar Parameter

Concept: radar parameter *(RadarParameter)*

Definition: This concept defines the parameters of the radar sensor. This concept has the following attributes:

- Detection probability *(detectionProbability)*: This attribute defines the probability at which a radar detects an object and is specified as a decimal scalar in the range [0, 1]
- False alarm rate *(falseAlarmRate)*: This attribute defines the probability of false detection per resolution rate and is specified as a decimal scalar in the range [1e-07, 1e-03].
- Field of view azimuth *(fieldOfViewAzimuth)*: This attribute defines the horizontal field of view of a radar. a positive decimal scalar and is specified in degrees.
- Field of view elevation *(fieldOfViewElevation)*: This attribute defines the vertical field of view of radar. This is a positive decimal scalar and is specified in degrees
- Max range *(maxRange)*: This attribute defines the farthest distance at which a radar can detect an objects. This is a positive decimal scalar and is specified in meters

This can be represented as follows:

$$C= \{ RadarParameter \} \rightarrow A= \{ detectionProbability, falseAlarmRate, fieldOfViewAzimuth, fieldOfViewElevation, maxRange \} \rightarrow D= \{ D_{detectionProbability}, D_{falseAlarmRate}, D_{fieldOfViewAzimuth}, D_{fieldOfViewElevation}, D_{maxRange} \}$$

$$\omega(RadarLimits) = (\{detectionProbability, falseAlarmRate, fieldOfViewAzimuth, fieldOfViewElevation, maxRange, \{ D_{detectionProbability}, D_{falseAlarmRate}, D_{fieldOfViewAzimuth}, D_{fieldOfViewElevation}, D_{maxRange} \})$$

2.7.2.4.2. Radar Detection

Concept: radar detection *(RadarDetection)*

Definition:

This concept has the following attributes:

- Reference range *(referenceRange)*: This is the range at which the radar detects a target of the size specified by Reference RCS, given the probability of detection specified by Detection Probability. Reference range for a given probability of detection is a positive decimal scalar and is specified in meters.
- Reference RCS *(referenceRCS)*: For a given probability of detection the reference RCS is in decibels per square meter and is specified as a nonnegative decimal scalar. The reference RCS is the target size at which the radar detects a target, given the reference range specified by Reference Range and the probability of detection specified by Detection Probability

This can be represented as follows:

$$C = \{RadarDetection\} \rightarrow A = \{ referenceRange, ReferenceRCS \} \rightarrow D = \{ D_{referenceRange}, D_{referenceRCS} \}$$

$$\omega(RadarDetection) = (\{ referenceRange, ReferenceRCS, \{ D_{referenceRange}, D_{referenceRCS} \})$$

2.7.2.4.3. Radar Limit

Concept: radar limit *(RadarLimit)*

Definition:

This concept has the following attributes:

- Azimuth resolution *(azimuthResolution)*: The ability of the radar to differentiate between multiple targets in degrees due to the minimum separation in azimuth angle and is listed as a positive decimal scalar.
- Azimuth bias fraction *(azimuthBiasFraction)*: The maximum azimuth precision of the radar and is a nonnegative decimal scalar.
- Range resolution *(rangeResolution)*: Minimum range separation at which the radar can distinguish between two targets, and it is a positive decimal scalar and is specified as in meters.
- Range bias fraction *(rangeBiasFraction)*: This attribute defines the accuracy of a maximum range accuracy of the radar, specified as a nonnegative decimal scalar. The range bias is expressed as a fraction of the range resolution specified in the Range Resolution parameter. Units are dimensionless
- Has Noise *(hasNoise)*: This attribute enables adding noise to sensor measurements.
- Has false alarm *(hasFalseAlarm)*: This attribute enables false alarm to sensor measurements.

This can be represented as follows:

$$C = \{Radar\ Limit\} \rightarrow A = \{ azimuthResolution, azimuthBiasFraction, rangeResolution, rangeBiasFraction, hasNoise, hasFalseAlarm \} \rightarrow D = \{ D_{azimuthResolution}, D_{azimuthBiasFraction}, D_{rangeResolution}, D_{rangeBiasFraction}, D_{hasNoise}, D_{hasFalseAlarm} \}$$

$$\omega(RadarLimit) = (\{referenceRange, ReferenceRCS, \{ D_{referenceRange}, D_{referenceRCS} \})$$

2.7.2.5. Lidar

Concept: Lidar *(Lidar)*

Definition:

This concept has the following sub concepts.

$$[Lidar] = [LidarParameters]$$

2.7.2.5.1. Lidar Parameters

Concept: Lidar parameter *(LidarParameter)*

Definition:

This concept has the following attributes:

- Max Range *(maxRange)*: This attribute defines the farthest distance a lidar can detect objects. It is a positive decimal scalar and is specified in meters.
- Range accuracy *(rangeAccuracy)*: This attribute defines the accuracy of range measurements. It is a positive decimal scalar and is specified in meters.
- Azimuth *(azimuth)*: This attribute defines the azimuthal resolution which is the minimum separation in azimuth angle at which the Lidar can distinguish two targets. It is a positive decimal scalar and is specified in degrees.
- Elevation *(Elevation)*: This attribute defines the elevation resolution which is the minimum separation in elevation angle at which the Lidar can distinguish two targets. It is a positive decimal scalar and is specified in meters.
- Azimuthal Limits (*Azimuthal Limits)*: The lidar sensor azimuthal limits and is a (min, max) form of the decimal scalar for a two-element vector. It is specified in degrees.
- Elevation Limits *(ElevationLimits)*: This attribute is the elevation limits of a lidar sensor specified as a two-element vector of decimal scalars of the form [min, max]. It is in degrees
- Has noise *(hasNoise)*: This attribute enables the summation of noise to the measurements of sensors.

This can be represented as follows:

$$C = \{ LidarParameter \} \rightarrow A = \{ maxRange, rangeAccuracy, azimuth, Elevation, Azimuthal\ Limits, ElevationLimits, hasNoise \} \rightarrow D = \{ D_{maxRange}, D_{rangeAccuracy}, D_{azimuth}, D_{Elevation}, D_{azimuthalLimits}, D_{ElevationLimits}, D_{hasNoise} \}$$

$$\omega(LidarParameter) = (\{ maxRange, rangeAccuracy, azimuth, Elevation, Azimuthal\ Limits, ElevationLimits, hasNoise, \{ D_{maxRange}, D_{rangeAccuracy}, D_{azimuth}, D_{Elevation}, D_{azimuthalLimits}, D_{ElevationLimits}, D_{hasNoise} \})$$

2.7.2.6. INS

Concept: INS*(INS)*

Definition: This represents a navigation device which uses motion sensors (accelerometers) and rotation sensors (gyroscope) to continuously calculate the position, the orientation and the velocity of the object it is placed within.

This concept has the following sub concept

$$[INS] = [INSParameter]$$

2.7.2.6.1. INS Parameter

Concept: INS parameter *(INSParameter)*

Definition: This concept represents the parameters of the INS sensors. This concept has the following attribute:

- Roll Accuracy *(rollAccuracy)*: This is the accuracy of roll, it is a non-negative and is specified in degrees, specified as anon negative decimal scalar. The standard deviation of the roll measurement noise is set by this value.
- Pitch Accuracy *(pitchAccuracy)*: This is the accuracy of pitch, it is a non-negative and is specified in degrees, specified as anon negative decimal scalar. The standard deviation of the pitch measurement noise is set by this value.
- Yaw Accuracy *(yawAccuracy)*: This is the accuracy of yaw, it is a non-negative and is specified in degrees, specified as anon negative decimal scalar. The standard deviation of the yaw measurement noise is set by this value.
- Position accuracy *(positionAccuracy)*: This is the accuracy for the x-, y-, and z-position measurements, it is a decimal scalar or three-element decimal scalar and is specified in meters. The standard deviation of the position measurement noise is set by this value. Specify a scalar to set the accuracy of all three positions to this value

- Velocity Accuracy *(velocityAccuracy)*: This is the accuracy of velocity measurements; it is a as a decimal scalar and is specified in meter per second. The standard deviation of the velocity measurement noise is set by this value.
- Acceleration Accuracy *(accelerationAccuracy)*: This is the accuracy of acceleration measurements; it is a as a decimal scalar and is specified in meter per second squared. The standard deviation of the acceleration measurement noise is set by this value.
- Angular Velocity Accuracy *(angularVelocityAccuracy)*: This is the accuracy of angular velocity measurements; it is a as a decimal scalar and is specified in degrees per second. The standard deviation of the angular velocity measurement noise is set by this value.

This can be represented as follows:

$$C = \{INSParameter\} \rightarrow A = \{ rollAccuracy, pitchAccuracy, velocityAccuracy, accelerationAccuracy, angularVelocityAccuracy \} \rightarrow D = \{ D_{rollAccuracy}, D_{pitchAccuracy}, D_{velocityAccuracy}, D_{accelerationAccuracy}, D_{angularVelocityAccuracy} \}$$

$$\omega(INSParameter) = (\{ rollAccuracy, pitchAccuracy, velocityAccuracy, accelerationAccuracy, angularVelocityAccuracy, \{ D_{rollAccuracy}, D_{pitchAccuracy}, D_{velocityAccuracy}, D_{accelerationAccuracy}, D_{angularVelocityAccuracy} \})$$

2.7.2.7. GPS

Concept: GPS *(GPS)*

Definition: This concept represents the global position system (GPS) sensor. The attribute for this concept includes:

- Position *(position)*: This attribute represents the current position acquired by the sensor
- Accuracy *(accuracy)*: This attribute defines the accuracy of the sensor

Section 3: Object Type Layer

3.1.Concepts and Attributes

This section defines the terminology and attributes of each concept within the 'Object Type' layer

3.1.1. Fixed Objects

Concept: fixed objects *(FixedObjects)*

Definition: The concept fixed objects *(FixedObjects)* represent objects which are immovable, static or motionless and therefore cannot change their position over time. Examples include, trees, buildings etc. These objects will remain static within a driving scenario. This concept has the following attributes:

- Object id *(objectId)*: This attribute defines the unique identity of the fixed object.
- Object position *(objectPosition)*: This attribute represents the centre position of the object with [x,y,z] coordinates
- Object length *(objectLength)*: The length is a positive scalar value defining the length of the concept fixed object *(FixedObject))*. The units are in meters.
- Object width *(objectWidth)*: The width is a positive scalar value defining the width of the concept fixed object *(FixedObject))*. The units are in meters.
- Object height *(objectHeight):* The height is a positive scalar value defining the height of the concept fixed object *(FixedObject))*. The units are in meters.
- Object type *(objectType)*: This attribute defines the type of object. The domains for this include:
  - Building
  - Road Segment
  - Lane Segment

  The domains for this attribute can be extended in future work

$C=\{FixedObject\} \rightarrow A=\{objectId, objectPosition, objectLength, objectWidth, objectHeight, objectType\} \rightarrow D=\{ D_{objectId}, D_{objectPosition}, D_{objectLength}, D_{objectWidth}, D_{objectHeight}, D_{objectSpeed}, D_{objectType} \}$

Hence ω can be defined as follows:

$$\omega(FixedObject) = (objectId, objectPosition, objectLength, objectWidth, objectHeight, objectType \{ D_{objectId}, D_{objectPosition}, D_{objectLength}, D_{objectWidth}, D_{objectHeight}, D_{objectSpeed}, D_{objectType} \})$$

3.1.2. Stationary Objects

Concept: stationary objects *(StationaryObjects)*

Definition: The class stationary objects *(StationaryObjects)* are those objects whose current speed with respect to the ground is zero but have the potential to change position over time. Example of these include standing pedestrian and parked vehicles. This concept has the following attributes:

- Object id *(objectId)*: This attribute defines the unique identity of the fixed object.
- Object position *(objectPosition)*: This attribute represents the centre position of the object with [x,y,z] coordinates
- Object length *(objectLength)*: The length is a positive scalar value defining the length of the concept stationary objects *(StationaryObjects)*. The units are in meters.
- Object width *(objectWidth)*: The width is a positive scalar value defining the width of the concept stationary objects *(StationaryObjects)*. The units are in meters.
- Object height *(objectHeight):* The height is a positive scalar value defining the height of the concept stationary objects *(StationaryObjects)*. The units are in meters.
- Object speed *(objectSpeed)*: This attribute defines the speed of the object and the domain of this attribute must be zero
- Object type *(objectType)*: This attribute defines the type of object. The domains for this include:
    - connectedAV: This represents the connected autonomous vehicle
    - Car: This represents a non-autonomous vehicle

  The domains for this attribute can be extended in future work

$$C=\{ StationaryObjects \} \rightarrow A= \{objectId, objectPosition, objectLength, objectWidth, objectHeight, objectSpeed\ objectType \} \rightarrow D= \{ D_{objectId}, D_{objectPosition}, D_{objectLength}, D_{objectWidth}, D_{objectHeight}, D_{objectSpeed}, D_{objectType} \}$$

Hence ω can be defined as follows:

$$\omega(StationaryObjects) = (objectId, objectPosition, objectLength, objectWidth, objectHeight, objectSpeed\ ,objectType \{ D_{objectId}, D_{objectPosition}, D_{objectLength}, D_{objectWidth}, D_{objectHeight}, D_{objectSpeed}, D_{objectType} \})$$

3.1.3. Dynamic Object

Concept: dynamic object *(DynamicObject)*

Definition: The class dynamic objects *(DynamicObjects)* whose current speed with respect to the ground is non-zero. Examples of these include moving vehicles, walking pedestrians. This concept has the following attributes:

- Object id *(objectId)*: This attribute defines the unique identity of the fixed object.
- Object position *(objectPosition)*: This attribute represents the centre position of the object with [x,y,z] coordinates
- Object length *(objectLength)*: The length is a positive scalar value defining the length of the concept stationary objects *(StationaryObjects)*. The units are in meters.
- Object width *(objectWidth)*: The width is a positive scalar value defining the width of the concept stationary objects *(StationaryObjects)*. The units are in meters.
- Object height *(objectHeight):* The height is a positive scalar value defining the height of the concept stationary objects *(StationaryObjects)*. The units are in meters.
- Object speed *(objectSpeed)* : This attribute defines the speed of the object in m/s. The value of this attribute must be >1
- Object heading *(objectHeading)*: This attribute defines the heading angle of the object
- Object type *(objectType)*: This attribute defines the type of object. The domains for this include:
    - ConnectedAV: This represents the connected autonomous vehicle
    - Car: This represents a non-autonomous vehicle
    - Pedestrian: This represents a walking road user
    - Cyclist: This represents a road user on a bike

  The domains for this attribute can be extended in future work

$$C=\{\ DynamicObjects\ \} \rightarrow A=\{objectId, objectPosition, objectLength, objectWidth, objectHeight, objectSpeed, objectHeading, objectType\ \} \rightarrow D=\{\ D_{objectId}, D_{objectPosition}, D_{objectLength}, D_{objectWidth}, D_{objectHeight}, D_{objectSpeed}, D_{objectHeading}, D_{objectType}\ \}$$

Hence ω can be defined as follows:

$$\omega(StationaryObjects) = (objectId, objectPosition, objectLength, objectWidth, objectHeight, objectSpeed, objectHeading, objectType \{ D_{objectId}, D_{objectPosition}, D_{objectLength}, D_{objectWidth}, D_{objectHeight}, D_{objectSpeed}, D_{objectHeading}, D_{objectType} \})$$

Section 4: Communication Network Layer

4.1. Application

Concept: Application *(Application)*

Definition: This concept represents the utilization of a vehicular networking application in a particular situation with a specific purpose. The attribute within this concept include:

- Road safety *(roadSafety)*: This attribute defines the road safety application which primarily provides information and assistance to drivers to avoid collisions with other vehicles. The domain within this attribute include:
  - **Intersection Collision Warning:** This represents a risk of a lateral collisions for a vehicles approaching road intersections. This is detected by the vehicles or roadside units.
  - **Lane Change Assistance:** This represents a risk of a lateral collisions for a vehicles attempting a lane change manoeuvre to reduce blind spot for trucks.
  - **Overtaking vehicle warning:** This prevent collision between vehicles in an overtake situation.
  - **Head on collision warning:** This represents a warning of a head on collision.
  - **Rear end collision warning:** This represents a warning of a risk of a rear end collision in front.
  - **Co-operative forward collision warning:** This represents a warning of forward collision through the cooperation between vehicles.
  - **Emergency vehicle warning:** This is a warning from an active emergency vehicle (for example, ambulance or police car) informing another vehicle to free the emergency corridor.
  - **Pre-crash sensing/warning:** This is a warning when a crash is considered unavoidable and will take place.
  - **Stationary vehicle warning:** This is a warning of a disabled due to an accident, breakdown or any other reason, informing other vehicles and roadside units about this situation.
  - **Traffic Condition warning:** This informs other vehicles and roadside units of rapid traffic evolution, detected by a vehicle

- Signal violation warning: one or more roadside units detect a traffic signal violation. This violation information is broadcasted by the roadside unit(s) to all vehicles in the neighbourhood
- **Collision risk warning:** Vehicle incapable of communicating with each other which are at a threat of colliding can be identified by the roadside units. The data of the event is then broadcasted to all vehicles in the district
- **Hazardous Location Notification:** In case of a barrier on the road, construction or hazardous road conditions, the vehicle or roadside units signal other vehicles about the locations of the hazard.
- **Control Loss Warning:** In a secondary use case is defined where if a vehicles malfunctions, the driver is able to send a distress broadcast control-loss signal to the surrounding vehicles. The relevancy of the scenario is determined by the vehicles and issue a warning to the drivers if needed

- Traffic efficiency *(trafficEfficiency)*: This attribute focus on improving the vehicle traffic flow, traffic coordination and traffic assistance and provide updated local information, maps and in general, messages of relevance bounded in space and/or time. The domain within this attribute include:
  - **Speed Management:** Speed management applications intend to allow a smoother driving experience to maintain speeds for the vehicles and prevent any unwanted stops. Such an example of this are green light optimal speed advisory and Regulatory/contextual speed limit notification.
  - **Cooperative navigation:** Communication amongst vehicles and between vehicles & roadside units by supervision of navigation will lead to a growth in traffic efficiency with these types of applications. Such an example of this are co-operative adaptive cruise control & platooning and traffic information & recommended itinerary provision.
- Infotainment application *(infotainmentApplication)*: This attribute focuses one entertainment services. The domain for this concept can be increased with future work

This can be represented as follows:

$$C=\{Application\} \rightarrow A=\{roadSafety, trafficEfficiency, infotainmentApplication\} \rightarrow D=\{ D_{roadSafety}, D_{trafficEfficiency}, D_{infotainmentApplication} \}$$

$$\omega(Application) = (roadSafety, trafficEfficiency, infotainmentApplication, \{ \{ D_{roadSafety}, D_{trafficEfficiency}, D_{infotainmentApplication} \})$$

4.2. Radio Communication

Concept: radio communication *(RadioCommunication)*

Definition: This concept represents the radio communication technologies used. The attributes for this concept are as follows:

- RAT *(RAT)*: This attribute defines the type of radio access technology (RAT) used, the domains for this attribute include:
    - **ITS-G**: This domain represents the DSRC communication utilizing the IEEE 802.11p
    - **CV2-X:** This domain represents the 5G communication network. This RAT will not be further explored in this work. However, the ontology can be expanded in the future to include this.
- Communication type *(communicationType)*: This attribute defines the type of communication being sent, the domains for this concept include:
    - **Send:** This transmits a message to a node within a radius of the transmitting node
    - **Broadcast**: This domain transmits a message to all nodes within the radius of the transmitting node
    - **Receive**: This domain receives a message transmitted by a node within a certain radius
- Range *(range)*: This attribute defines the communication range (in meters) for a single hop.
- Radius *(radius)*: This attribute defines a radius in which a message can transmitted or received
- Channel *(channel)*: This attribute defines the channel that has been accessed, channels differ with in the frequency they use and the application they support. The domain for this concept includes:
    - **Channel 1:** This represents the 20 MHz spectrum between 5.885 GHz – 5.905 GHz. This channel is allocated for inter vehicle *(IVC)* and roadside to vehicle *(R2V)* communication providing critical road safety application including the control channel.
    - **Channel 2:** This represents the 30 MHz spectrum between 5.875 GHZ – 5.885 and between 5.905 – 5.925 GHZ). This channel is for IVC and R2V providing road safety and traffic efficiency application.
    - **Channel 3:** This represents the 20MHz below 5.875 GHz. This channel is for IVC and R2V providing non-safety related application
- Bandwidth *(bandWidth)*: This attribute defines the available bandwidth for a communication network
- Bit rate *(bitRate)*: This attribute defines the transmission of number of bits per second.
- Vehicle Broadcast Group *(vehicleBroadcastGroup)*: This attribute contains the communication bit data that is to be sent to other vehicles.

This can be represented as follows:

$$C=\{RadioCommunication\} \rightarrow A=\{ type, range, channel, bandWidth, bitRate \} \rightarrow D=\{ D_{type}, D_{range}, D_{channel}, D_{bandWidth}, D_{bitRate} \}$$

$$\omega(RadioCommunication) = (type, range, channel, bandWidth, bitRate, \{ D_{type}, D_{range}, D_{channel}, D_{bandWidth}, D_{bitRate} \})$$

4.3. Network Communication

Concept: network communication *(NetworkCommunication)*

Definition:

The concept defines the network capabilities. The attribute of this concept are as follows:

- Mode of dissemination *(modeOfDissemination)*: This attribute defines the method at which a message is conveyed from the source to destination. The domains for this concept include
  - **Unicast:** This type of routing constructs a source to destination path.
  - **Multicast:** This is used to deliver data from one source to many interested recipients.
  - **Geocast:** This is used to deliver a data to a predefined geographic region
  - **Broadcast:** This is used to deliver data to all nodes in the network
- Data aggregation *(dataAggregation)*: This attribute defines the process by which data is collected and aggregated.
- Congestion control *(congestionControl)*: This attribute controls the entry of data packets into the network, enabling a better use of shared network infrastructure and avoiding congestive collapse. The domain for this attribute includes:
  - **CAA**: This represents the congestive avoidance algorithm which is implemented at the TCP layer as a mechanism to avoid congestive collapse in a network.

  The domain for this can be expanded for future work
- Message Priority *(messagePriority)*: This attribute schedules a message according to its priority. To determine the priority of a message the following is usually considered: Importance, Delivery probability
- Support *(support)*: support of IPv6 or IPv4 addressing
- Mobility *(mobility)*: This attribute defines the rate of mobility of the vehicle.

This can be represented as follows:

$$C=\{ NetworkCommunication \} \rightarrow A= \{modeOfDissemination, dataAggregation, congestionControl, messagePriority, support, mobility \} \rightarrow D= \{ D_{modeOfDissemination}, D_{dataAggregation}, D_{congestionControl}, D_{messagePriority}, D_{support}, D_{mobility} \}$$

$$\omega (NetworkCommunication) = (modeOfDissemination, dataAggregation, congestionControl, messagePriority, support, mobility,\{ D_{modeOfDissemination}, D_{dataAggregation}, D_{congestionControl}, D_{messagePriority}, D_{support}, D_{mobility} \})$$

4.4. System performance

Concept: system performance *(SystemPerformance)*

Definition:

This concept System performance requirements. These requirements are related to the system performance, which are vehicle communication performance. The attribute within this concept are as follows:

- Latency *(latency)*: This attribute is a measure of delay within the communication network. This is the time taken for some data to get to its destination.
- End-to-end delay *(endToEndDelay)*: This attribute defines the average time taken for the message to be sent successfully from a source node to the specific destination
- Channel Utilization *(channelUtilization)*: This attribute defines the percentage of time a channel is being used.
- Updating frequency *(updatingFrequency)*: This attribute defines the frequency at which a communication data is sent
- Resending frequency *(resendingFrequency)*: This attribute defines the frequency at which data is sent.
- Position accuracy *(positionAccuracy)*: This attribute defines the position accuracy for the vehicle the domain can range from [0 – 100%]
- System reliability *(systemReliability)*: This attribute defines the reliability that a system performs correctly during a specific time duration. The system reliability could be affected due to radio coverage, bit error rate and blackzone etc.
- Message Delivery Success Rate *(messageDeliverySuccessRate)*: This attribute defines the percentage of the total number of packets sent by a source node to the number of data packets successfully received by a destination node.

- Overhead *(overhead)*: This attribute defines the total number of message transmissions divided by the number of delivered messages
- Delivered messages *(deliveredMessage)*: This attribute defines the total number of messages delivered.
- Request time *(requestTime)*: This attribute defines the request time for a message.
- Processing time *(processingTime)*: The attribute processing time is the time taken from receiving the last byte of the request and returning the first byte of the response. This does not include request or response time.
- Response time *(responseTime)*: This attribute defines the total time it takes to respond to a sent request; this includes all network latencies. Therefore, the response time is the sum of processing time and encountered latencies.

  Response Time = time of response – time of request

This can be represented as follows:

$$C=\{ SystemPerformance \} \rightarrow A= \{ latency, updatingFrequency, positionAccuracy, messageDeliverySuccessRate, overhead, responseTime \} \rightarrow D= \{ D_{latency}, D_{updatingFrequency}, D_{positionAccuracy}, D_{messageDeliverySuccessRate}, D_{overhead}, D_{responseTime} \}$$

$$\omega (SystemPerformance) = (latency, updatingFrequency, positionAccuracy, messageDeliverySuccessRate, overhead, responseTime, \{ D_{latency}, D_{updatingFrequency}, D_{positionAccuracy}, D_{messageDeliverySuccessRate}, D_{overhead}, D_{responseTime} \})$$

4.5. Security

Concept: security *(Security)*

Definition:

This concept discusses Vehicle communication security capabilities. The attribute for this concept is as follows:

- Privacy *(privacy)*: respect of privacy and anonymity
- Integrity *(integrity)*: integrity and confidentiality
- Resistance *(resistance)*: resistance to external security attacks
- Authenticity *(authenticity)*: authenticity of received data
- Data Integrity *(dataIntegrity)*: data and system integrity
- performance of security operations, such as performance of signing and verifying messages and certificates

This can be represented as follows:

$$C=\{\ NetworkCommunication\ \} \rightarrow A=\{modeOfDissemination,\ dataAggregation,\ congestionControl,\ messagePriority,\ support,\ mobility\ \} \rightarrow D=\{\ D_{modeOfDissemination},\ D_{dataAggregation},\ D_{congestionControl},\ D_{messagePriority},\ D_{support},\ D_{mobility}\ \}$$

$$\omega\ (NetworkCommunication) = (modeOfDissemination,\ dataAggregation,\ congestionControl,\ messagePriority,\ support,\ mobility,\{\ D_{modeOfDissemination},\ D_{dataAggregation},\ D_{congestionControl},\ D_{messagePriority},\ D_{support},\ D_{mobility}\ \})$$

Section 5: Scene Layer

5.1. Scene

Concept: scene *(Scene)*

Definition

The concept *(C)* scene *(Scene)* describes a snapshot of the environment including, fixed, stationary and dynamic objects

This concept has the following attributes:

- Scene Number *(sceneNumber)*: This attribute defines the scene number
- Road Network id *(roadNetwork)*: This attribute represents the road network id within a scene.
- Number of Junction: This attribute defines the number of junctions within a scene. The domain for this attribute must be equal to or greater than one 1
- Number of Ego *(numberOfEgo)*: This attribute defines the number of ego vehicle in a scene. The domain for this attribute must be equal to 1.
- Number of connectedAV *(numberOfconnectedAV)*: This attribute defines the number of ConnectedAV in a scene. The domain for this attribute must be equal or greater than 1.
- Number of road user *(numberOfRoadUser):* This attribute defines the total number of road user per scene the domain for this must be greater than or equal to 2
- Number of fixed object *(numberOfFixedObject)*: This attribute defines the number of fixed objects within a scene.
- Number of stationary objects *(numberOfStationaryObjects)*: This attribute defines the number of stationary objects within a scene.
- Number of dynamic objects *(numberOfDynamicObjects)*: This attribute defines the number of dynamic objects within a scene.

$$C=\{Scene\} \rightarrow A=\{ sceneNumber, roadNetwork\ numberOfEgo, numberOfconnectedAV, numberOfRoadUser, numberOfFixedObject, numberOfStationaryObjects, numberOfDynamicObjects \} \rightarrow D=\{ D_{sceneNumber}, D_{roadNetwork}\ D_{numberOfEgo}, D_{numberOfConnectedAV}, D_{numberOfRoadUser}, D_{numberOfFixedObject}, D_{numberOfStationaryObjects}, D_{numberOfDynamicObjects} \}$$

Hence ω can be defined as follows:

$$\omega(Scene) = (sceneNumber, roadNetwork\ numberOfEgo, numberOfconnectedAV, numberOfRoadUser, numberOfFixedObject, numberOfStationaryObjects, numberOfDynamicObjects\ \{ D_{sceneNumber}, D_{roadNetwork}\ D_{numberOfEgo}, D_{numberOfConnectedAV}, D_{numberOfRoadUser}, D_{numberOfFixedObject}, D_{numberOfStationaryObjects}, D_{numberOfDynamicObjects}\ \})$$

Section 6: Scenario Layer

6.1. Scenario

Concept: scenario *(Scenario)*

Definition: The 'Scenario' Layer represents the temporal development or relationship between several scenes in a sequence of scenario. As shown in Figure 4.29

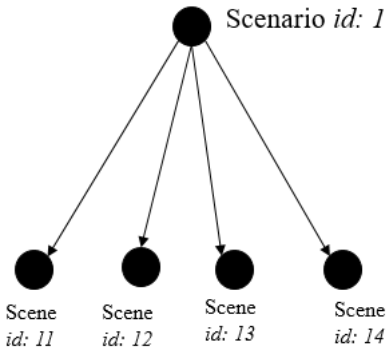


Figure 4.29: Scenario

Definition

This concept (c) has the following attribute:

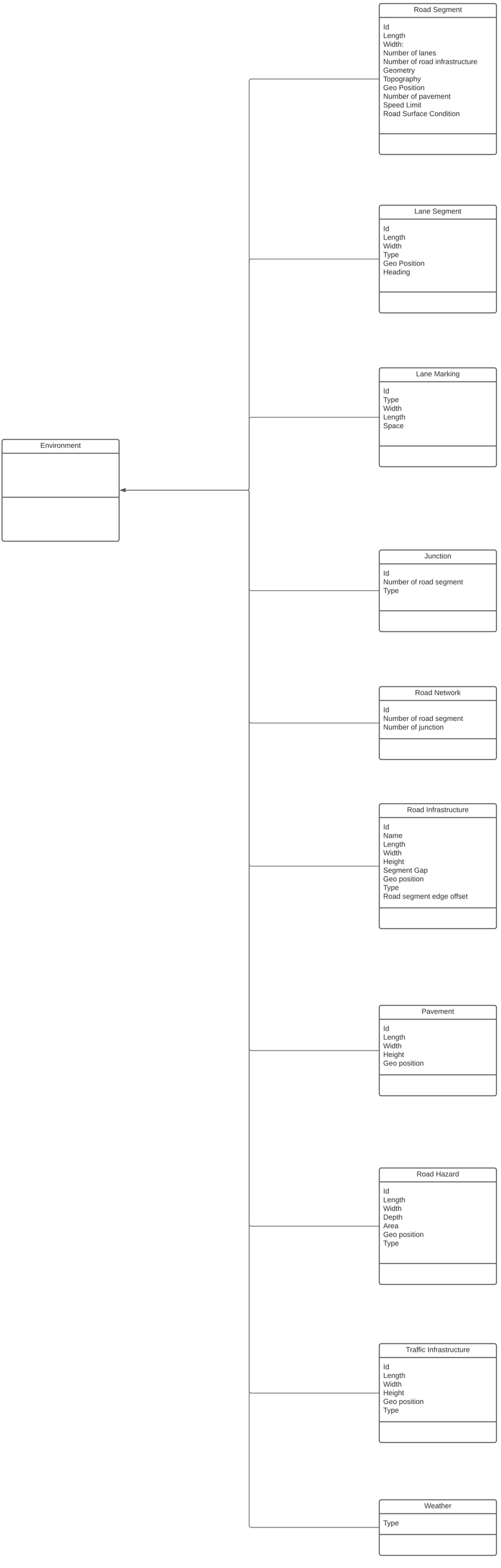
- Scenario id *(scenarioId)*: This attribute uniquely identifies each scenario.
- Time *(time)*: This attribute defines the duration of the scenario
- Current time *(currentTime)*: This attribute defines the current time at an instance with reference to t_0 at the first scene.
- Initial scene *(initialScene)*: This attribute defines the id of the initial scene
- Final scene *(finalScene)*: This attribute defines the id of the final scene
- Trajectory *(trajectory)*: This attribute defines the trajectory of the scenes. so, a vector going from the initial scene to the final scene. The trajectory is a thread that connects waypoints consecutive scenes and speed.

$$C=\{Scenario\} \rightarrow A=\{scenarioId, time, currentTime, weather, intialScene, finalScene\ trajectory\} \rightarrow D=\{ D_{scenarioId}, D_{time}, D_{currentTime}, D_{weather}\ D_{intialScene}, D_{finalScene}, D_{trajectory} \}$$

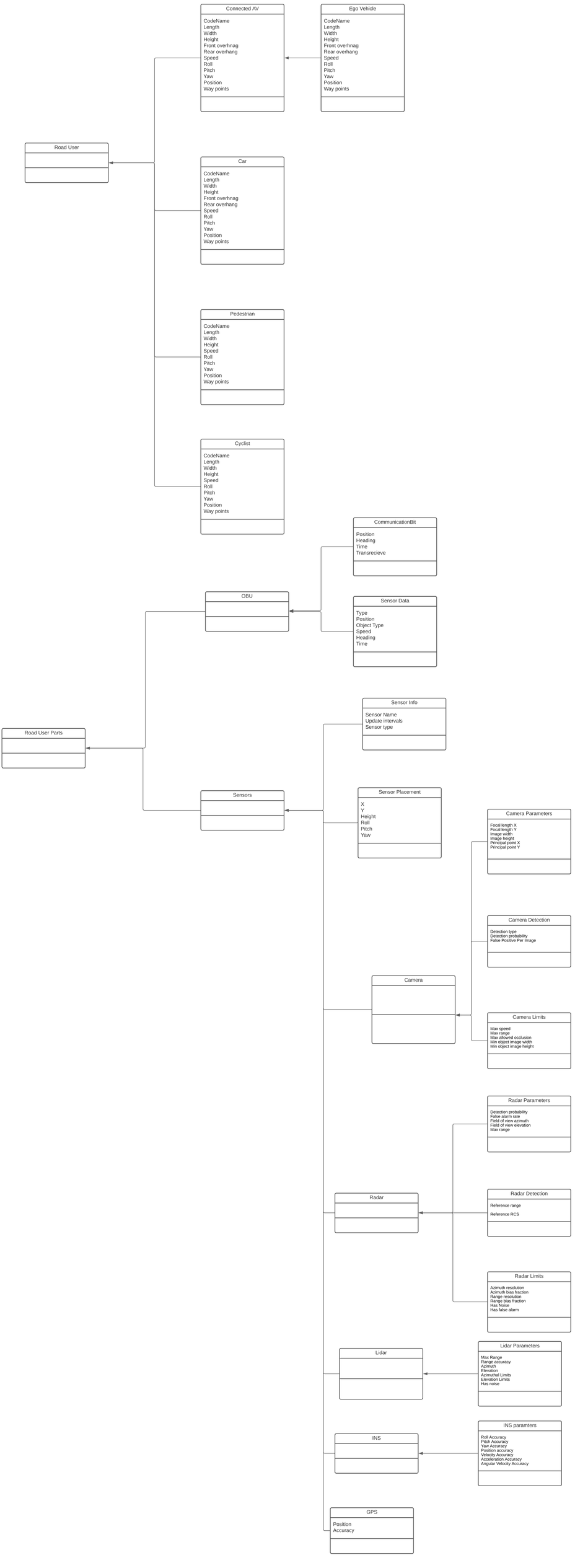
Hence ω can be defined as follows:

$$\omega(Scenario) = (scenarioId, time, currentTime, weather, intialScene, finalScene, trajectory, \{ D_{scenarioId}, D_{time}, D_{currentTime}, D_{weather} D_{intialScene}, D_{finalScene} D_{trajectory} \})$$

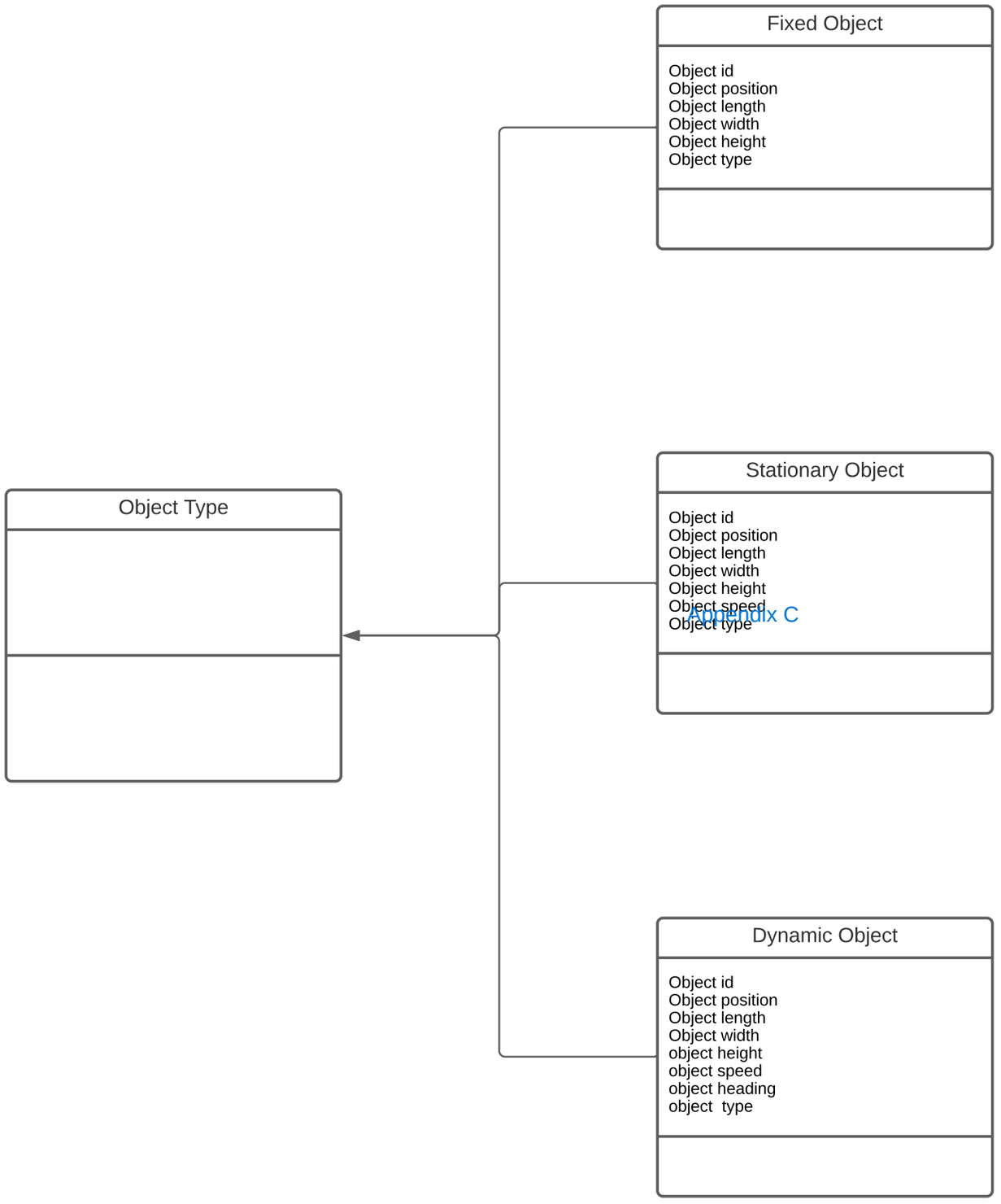
Appendix B


Road Segment
Id
Length
Width:
Number of lanes
Number of road infrastructure
Geometry
Topography
Geo Position
Number of pavement
Speed Limit
Road Surface Condition
Lane Segment
Id
Length
Width
Type
Geo Position
Heading
Lane Marking
Id
Type
Width
Length
Space
Environment
Junction
Id
Number of road segment
Type
Road Network
Id
Number of road segment
Number of junction
Road Infrastructure
Id
Name
Length
Width
Height
Segment Gap
Geo position
Type
Road segment edge offset
Pavement
Id
Length
Width
Height
Geo position
Road Hazard
Id
Length
Width
Depth
Area
Geo position
Type
Traffic Infrastructure
Id
Length
Width
Height
Geo position
Type
Weather
Type

Appendix C

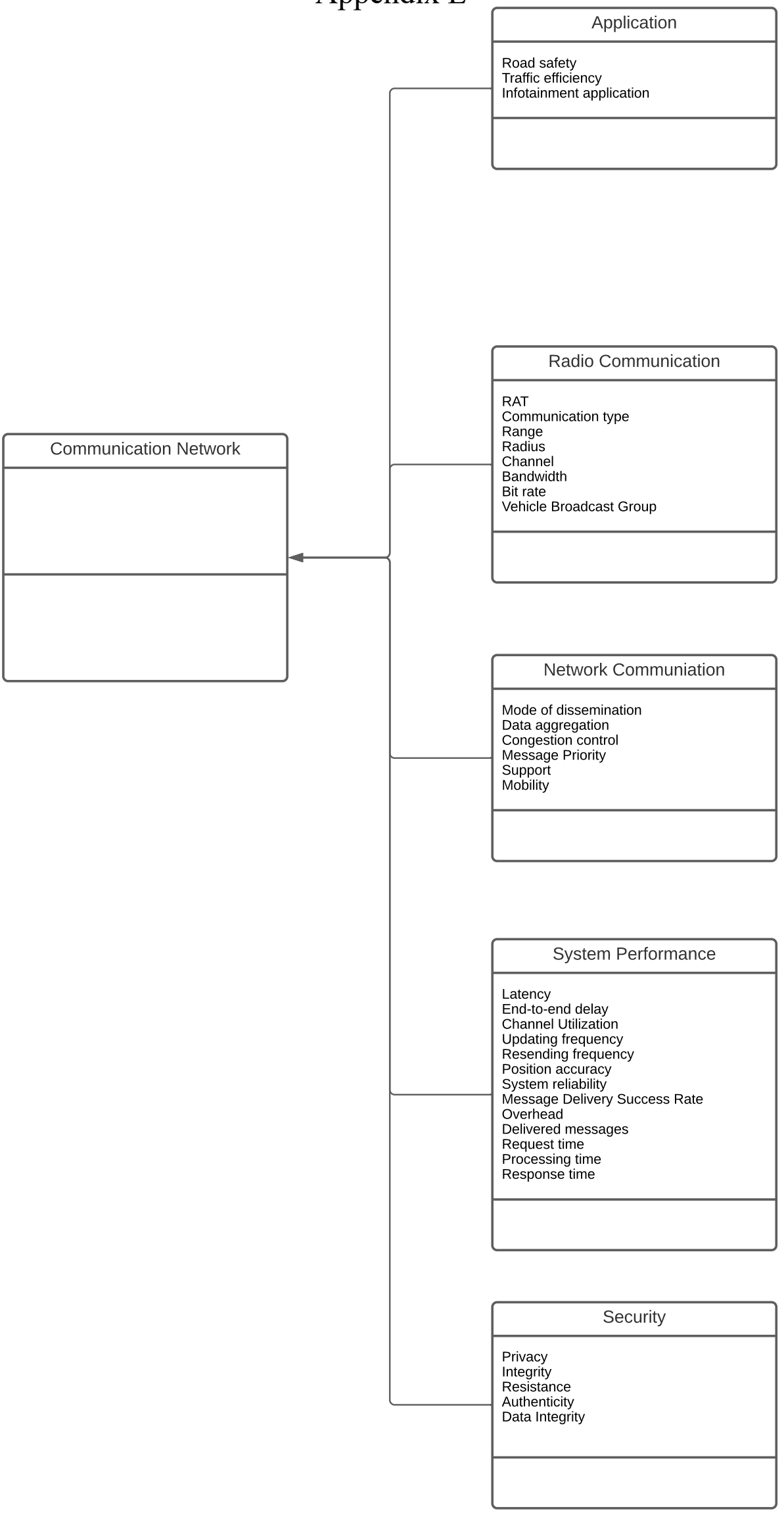

Connected AV
CodeName
Length
Width
Height
Front overhnag
Rear overhang
Speed
Roll
Pitch
Yaw
Position
Way points
Ego Vehicle
CodeName
Length
Width
Height
Front overhnag
Rear overhang
Speed
Roll
Pitch
Yaw
Position
Way points
Road User
Car
CodeName
Length
Width
Height
Front overhnag
Rear overhang
Speed
Roll
Pitch
Yaw
Position
Way points
Pedestrian
CodeName
Length
Width
Height
Speed
Roll
Pitch
Yaw
Position
Way points
Cyclist
CodeName
Length
Width
Height
Speed
Roll
Pitch
Yaw
Position
Way points
CommunicationBit
Position
Heading
Time
Transrecieve
OBU
Sensor Data
Type
Position
Object Type
Speed
Heading
Time
Sensor Info
Sensor Name
Update intervals
Sensor type
Road User Parts
Sensors
Sensor Placement
X
Y
Height
Roll
Pitch
Yaw
Camera Parameters
Focal length X
Focal length Y
Image width
Image height
Principal point X
Principal point Y
Camera Detection
Detection type
Detection probability
False Positive Per Image
Camera
Camera Limits
Max speed
Max range
Max allowed occlusion
Min object image width
Min object image height
Radar Parameters
Detection probability
False alarm rate
Field of view azimuth
Field of view elevation
Max range
Radar
Radar Detection
Reference range
Reference RCS
Radar Limits
Azimuth resolution
Azimuth bias fraction
Range resolution
Range bias fraction
Has Noise
Has false alarm
Lidar
Lidar Parameters
Max Range
Range accuracy
Azimuth
Elevation
Azimuthal Limits
Elevation Limits
Has noise
INS
INS paramters
Roll Accuracy
Pitch Accuracy
Yaw Accuracy
Position accuracy
Velocity Accuracy
Acceleration Accuracy
Angular Velocity Accuracy
GPS
Position
Accuracy

Appendix D

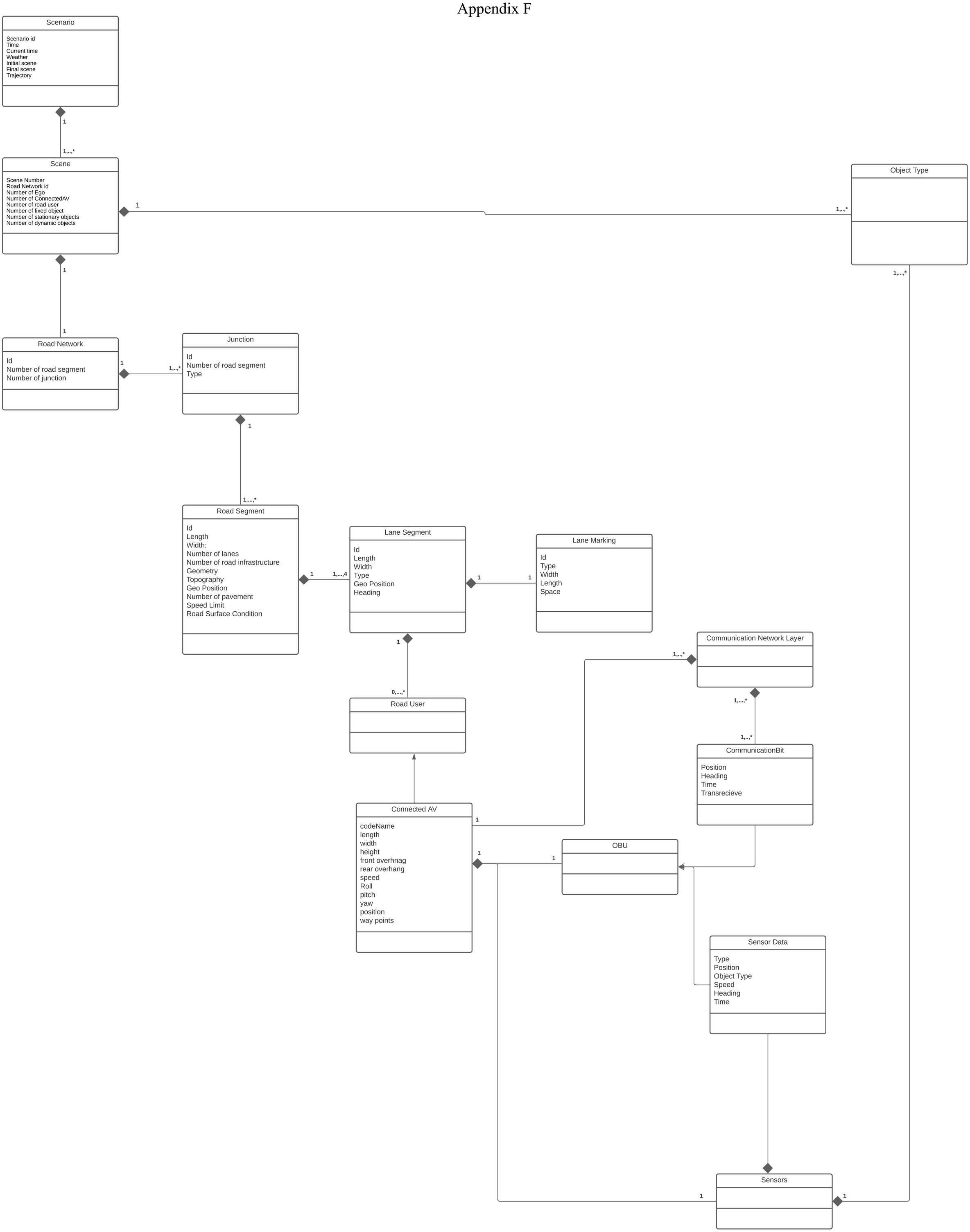


Appendix C

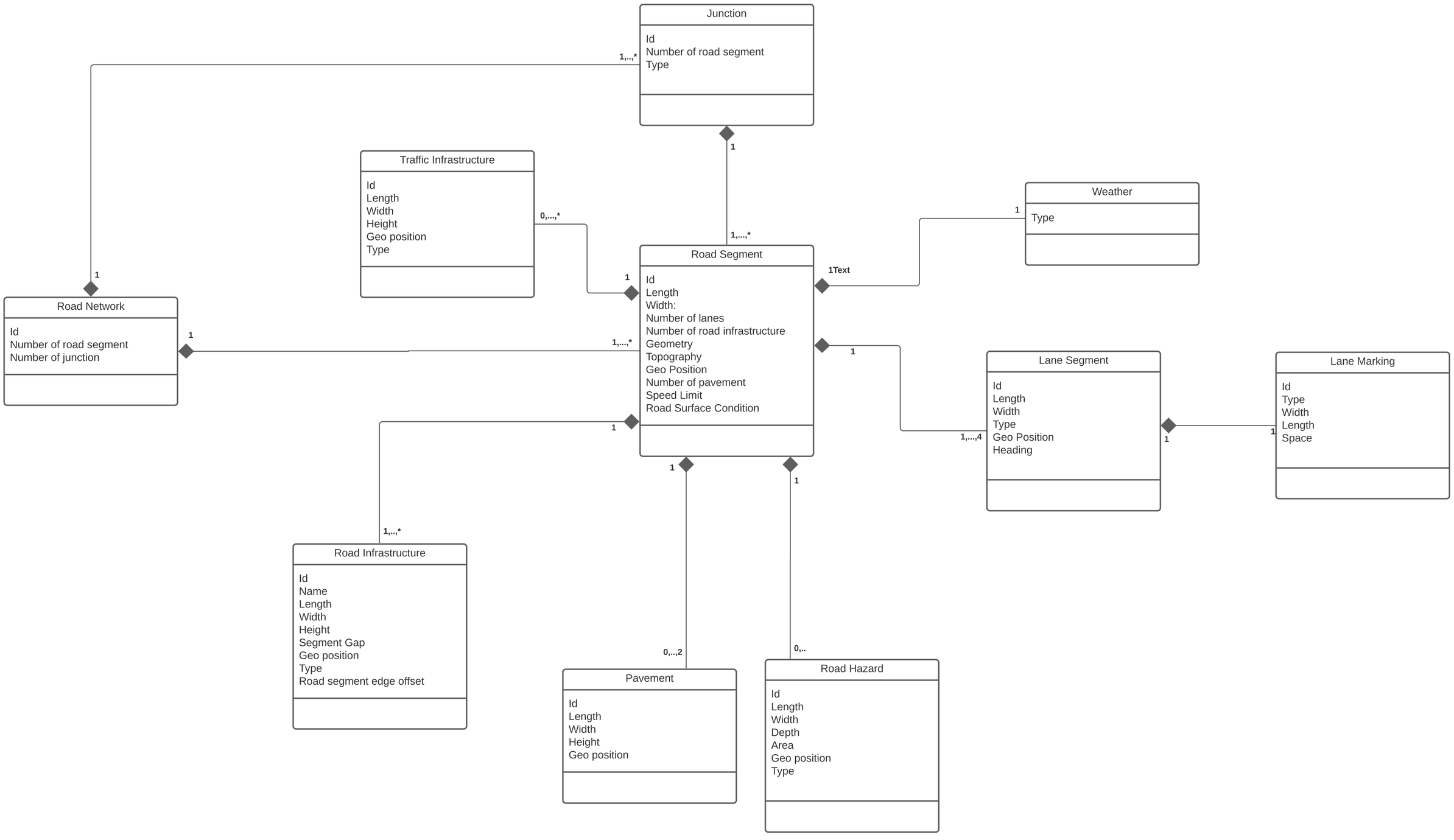
Appendix E


Application
Road safety
Traffic efficiency
Infotainment application
Radio Communication
RAT
Communication type
Range
Radius
Channel
Bandwidth
Bit rate
Vehicle Broadcast Group
Communication Network
Network Communiation
Mode of dissemination
Data aggregation
Congestion control
Message Priority
Support
Mobility
System Performance
Latency
End-to-end delay
Channel Utilization
Updating frequency
Resending frequency
Position accuracy
System reliability
Message Delivery Success Rate
Overhead
Delivered messages
Request time
Processing time
Response time
Security
Privacy
Integrity
Resistance
Authenticity
Data Integrity

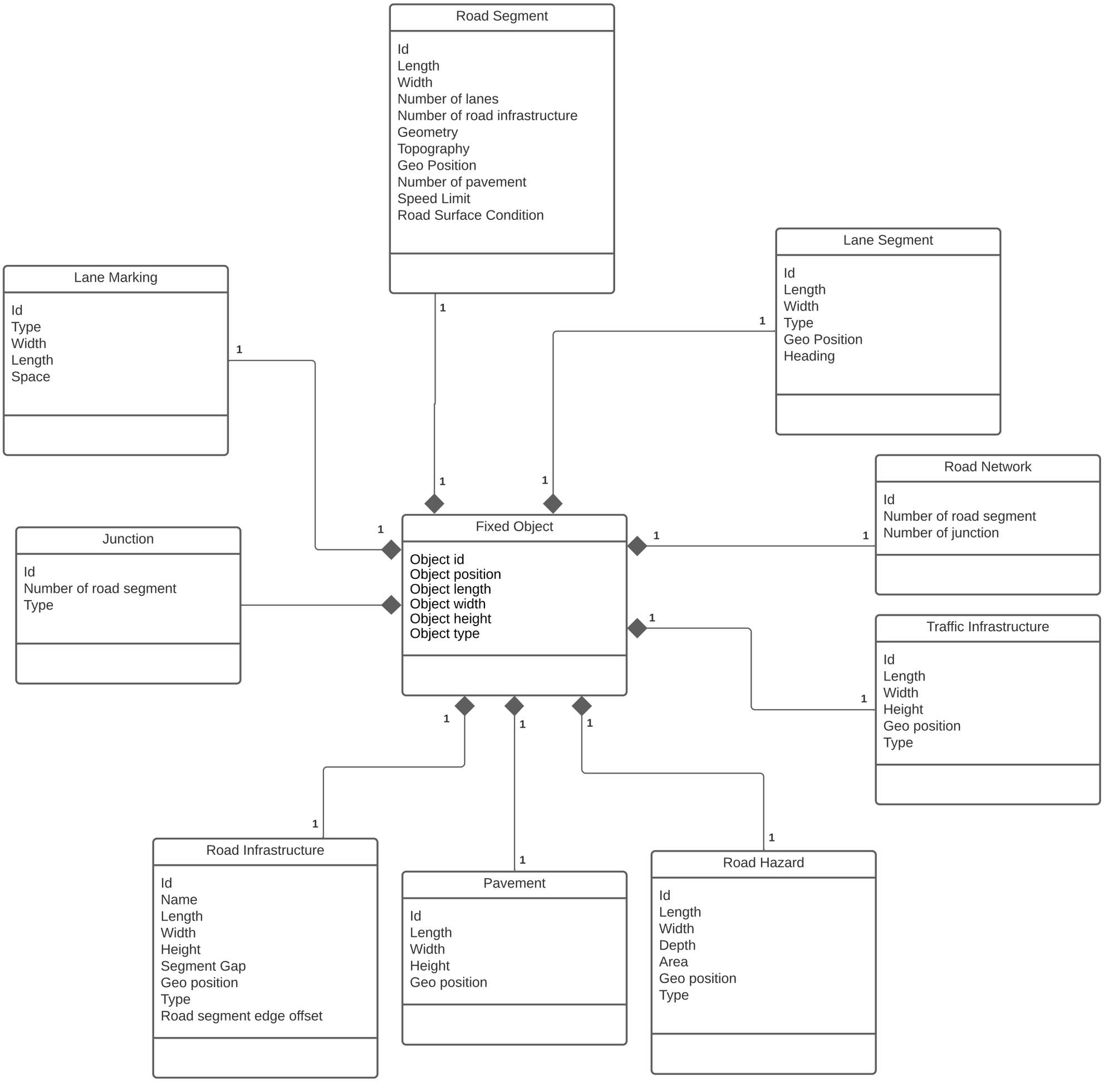
Appendix F


Scenario
Scenario id
Time
Current time
Weather
Initial scene
Final scene
Trajectory
1
1,..,*
Scene
Scene Number
Road Network id
Number of Ego
Number of ConnectedAV
Number of road user
Number of fixed object
Number of stationary objects
Number of dynamic objects
1
1,..,*
Object Type
1,...,*
1
1
Road Network
Id
Number of road segment
Number of junction
1
1,..,*
Junction
Id
Number of road segment
Type
1
1,...,*
Road Segment
Id
Length
Width:
Number of lanes
Number of road infrastructure
Geometry
Topography
Geo Position
Number of pavement
Speed Limit
Road Surface Condition
1
1,...,4
Lane Segment
Id
Length
Width
Type
Geo Position
Heading
1
1
Lane Marking
Id
Type
Width
Length
Space
1
0,...,*
Road User
1,..,*
Communication Network Layer
1,...,*
1,..,*
CommunicationBit
Position
Heading
Time
Transrecieve
Connected AV
codeName
length
width
height
front overhnag
rear overhang
speed
Roll
pitch
yaw
position
way points
1
1
OBU
Sensor Data
Type
Position
Object Type
Speed
Heading
Time
1
1
Sensors

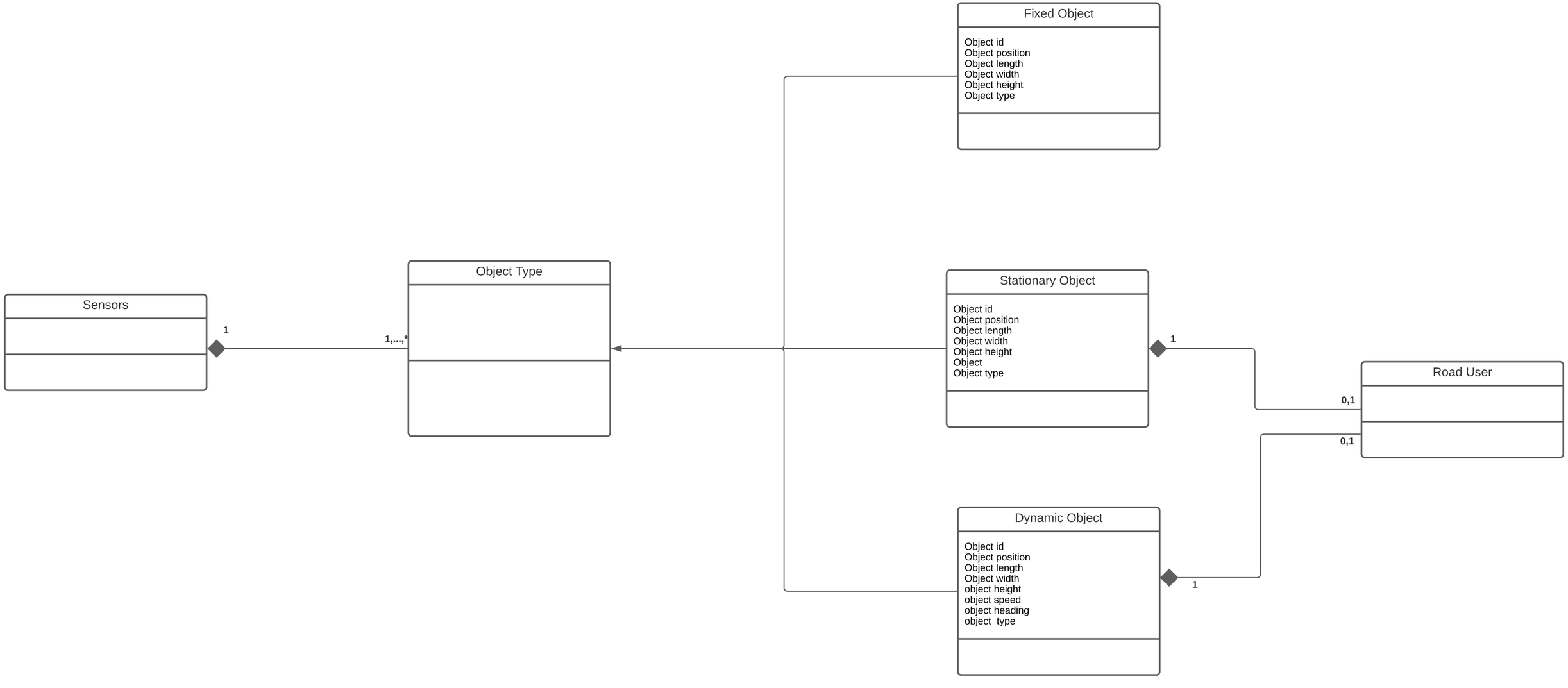
Appendix G


Junction
Id
Number of road segment
Type
1,..,*
1
Traffic Infrastructure
Id
Length
Width
Height
Geo position
Type
0,...,*
1,...,*
Weather
Type
1
1Text
Road Network
Id
Number of road segment
Number of junction
1
1,...,*
Road Segment
Id
Length
Width:
Number of lanes
Number of road infrastructure
Geometry
Topography
Geo Position
Number of pavement
Speed Limit
Road Surface Condition
1
1,...,4
Lane Segment
Id
Length
Width
Type
Geo Position
Heading
1
1
Lane Marking
Id
Type
Width
Length
Space
1
1,..,*
Road Infrastructure
Id
Name
Length
Width
Height
Segment Gap
Geo position
Type
Road segment edge offset
0,..,2
Pavement
Id
Length
Width
Height
Geo position
0,..
Road Hazard
Id
Length
Width
Depth
Area
Geo position
Type

Appendix H


Road Segment
Id
Length
Width
Number of lanes
Number of road infrastructure
Geometry
Topography
Geo Position
Number of pavement
Speed Limit
Road Surface Condition
Lane Marking
Id
Type
Width
Length
Space
Lane Segment
Id
Length
Width
Type
Geo Position
Heading
Road Network
Id
Number of road segment
Number of junction
Junction
Id
Number of road segment
Type
Fixed Object
Object id
Object position
Object length
Object width
Object height
Object type
Traffic Infrastructure
Id
Length
Width
Height
Geo position
Type
Road Infrastructure
Id
Name
Length
Width
Height
Segment Gap
Geo position
Type
Road segment edge offset
Pavement
Id
Length
Width
Height
Geo position
Road Hazard
Id
Length
Width
Depth
Area
Geo position
Type
1

Appendix I


Fixed Object
Object id
Object position
Object length
Object width
Object height
Object type
Object Type
Sensors
1
1,...,*
Stationary Object
Object id
Object position
Object length
Object width
Object height
Object
Object type
1
Road User
0,1
0,1
Dynamic Object
Object id
Object position
Object length
Object width
object height
object speed
object heading
object type
1