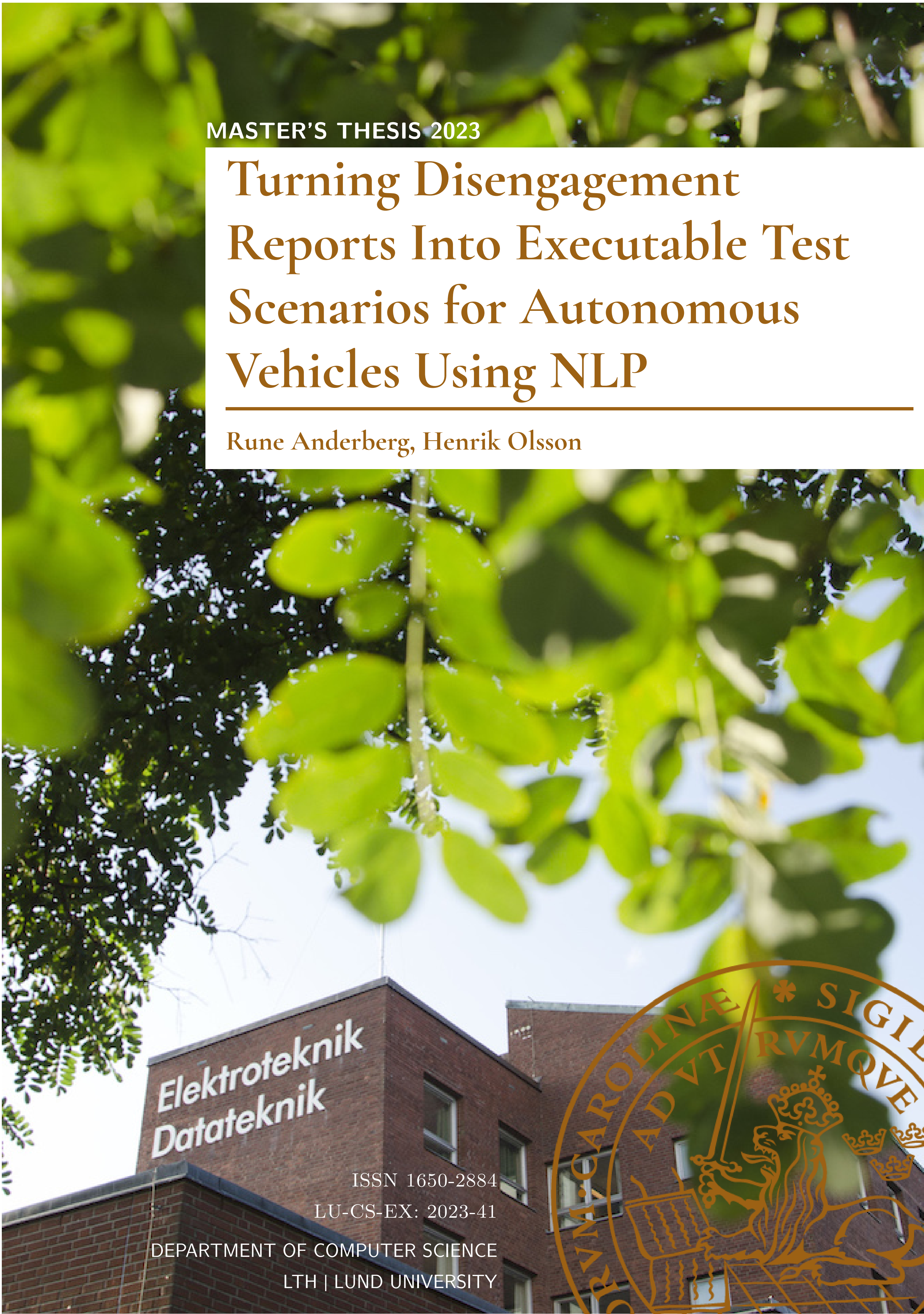
MASTER'S THESIS 2023

Turning Disengagement Reports Into Executable Test Scenarios for Autonomous Vehicles Using NLP

Rune Anderberg, Henrik Olsson



ISSN 1650-2884
LU-CS-EX: 2023-41

DEPARTMENT OF COMPUTER SCIENCE
LTH | LUND UNIVERSITY

EXAMENSARBETE
Datavetenskap

LU-CS-EX: 2023-41

Turning Disengagement Reports Into Executable Test Scenarios for Autonomous Vehicles Using NLP

Skapande av exekverbara testscenarier för självkörande bilar utifrån frånkopplingsrapporter med hjälp av NLP

Rune Anderberg, Henrik Olsson

Turning Disengagement Reports Into Executable Test Scenarios for Autonomous Vehicles Using NLP

Rune Anderberg
ru5122an-s@student.lu.se

Henrik Olsson
he0755ol-s@student.lu.se

15th September 2023

Master's thesis work carried out at
the Department of Computer Science, Lund University.

Supervisor: Qunying Song, qunying.song@cs.lth.se

Examiner: Per Runeson, per.runeson@cs.lth.se

Abstract

Autonomous vehicle development is rapidly growing, and occasionally, testing occurs on public roads. Although valuable, such testing can also be expensive and irregular. To be allowed to test on public roads in California, USA, manufacturers must submit reports containing each instance the autonomous driving mode had been disengaged, either by the driver or the vehicle, to the California Department of Motor Vehicles. This master's thesis has studied using these disengagement reports as a basis when creating testing scenarios for simulators to ease and lower the cost of testing to increase the safety of autonomous vehicles. We developed a concept application that automatically creates test scenarios in OpenSCENARIO from disengagement reports using Natural Language Processing. Due to qualitative issues with the disengagement reports, mainly a lack of detailed information, the application can only create concrete test scenarios for a small subset of the available disengagement reports in its current state. However, the project demonstrates the feasibility of this approach and proposes future work through suggested application improvements and ideas for other opportunities that utilise the disengagement reports.

Keywords: Autonomous vehicle, Natural Language Processing, Disengagement report, Test generation, OpenSCENARIO, CARLA

Acknowledgements

We would like to thank Qunying Song for the base idea of this project as well as for all the guidance we have received throughout the project. We would also like to thank Per Runeson for helping us find the opportunity to do this project and for the valuable input he provided in the early phases of the project.

Contents

Chapter 1
Introduction

1.1 Background

Many modern motor vehicles already come equipped with features that allow them to be partially autonomous under certain conditions such as lane assist and self-parking, and even though several challenges exist before full autonomy can become reality [1], Ahmed, Iqbal, Amin *et al.* [1] and Petrović, Mijailović and Pešić [2] among many others believe that autonomous vehicles (AVs) will be the future of transport. However, there is scepticism when it comes to adoption, where the safety, security and reliability of autonomous vehicles are some of the areas that future users are concerned with [1].

Assuring 'safety, reliability, resilience, and performance of the vehicles in diverse road conditions, transportation infrastructures, as well as environmental conditions' is the primary objective of quality validation for autonomous vehicles according to Gao, Wu and Aktouf [3]. This quality assurance process with validation activities is also referred to as testing. However, testing is an expensive part of autonomous vehicle development, with costs making up over 50% of the overall engineering budget [4]. As a way to reduce cost and to increase testing efficiency when it comes to testing autonomous vehicles, conducting the testing in simulation is an area that has received increased focus [3]. In addition to creating the complex software that is needed to simulate the environment, the autonomous vehicles and all of their sensors etc., research efforts are also being put into improving the testing processes itself [3].

Three perspectives of work that are relevant in the test processes for autonomous vehicles are to [3]:

- Use adequate test models and methods to support the creation of high-quality test cases and scenarios.
- Speeding up and reducing the cost of a test process by using tools to reduce manual operations and automate different parts such as planning, generation and selection.
- Applying data-driven AI-powered techniques to improve an already existing process.

Using real-world data as a basis for test generation is not a novel idea. Two examples of this are the approaches presented by Gambi, Huynh and Fraser [5], [6] and Xinxin, Fei and Xiangbin [7]. They used police reports on car crashes and traffic accident videos respectively to generate test scenarios for autonomous vehicles based on these critical situations. Another source of real-world data is disengagement and collision reports, for example, those collected by California Department of Motor Vehicles (DMV). These have been studied before, for example, the collision reports by Petrović, Mijailović and Pešić [2] and Favarò, Nader, Eurich *et al.* [8] and the disengagement reports by Lv, Cao, Zhao *et al.* [9].

However, not all real-world data is equal. One downside with the data used in [5]–[7] is that an incident is not guaranteed to have involved an autonomous vehicle as the main vehicle, meaning that the incident could have been caused due to the fault of a human driver, such as driving under influence or distractions from a mobile phone. The collision reports studied in [2], [8] are at least guaranteed to involve an autonomous vehicle however the disengagement reports studied in [9] provide two additional aspects that make them particularly interesting as a basis for test generation. Firstly, a disengagement that has been reported has occurred either because of an error in the autonomous mode or because the test operator intervened for safety reasons. In both of these cases, the autonomous mode has come across a situation it was unable to handle properly, making the situation a good candidate for testing. Secondly, although using situations that lead to an incident is valuable, it also limits the available data to generate tests from as it does not contain situations not leading to an incident. The disengagement reports however contain situations that may or may not have led to an incident if the test operator did not take over the control of the vehicle.

1.2 Research Goals and Questions

The goal of this project is to explore and study the possibilities of automating the process of producing test scenarios from disengagement reports to facilitate autonomous vehicle testing. There are two primary outcomes from this project, with the first one being an application capable of producing test scenarios from disengagement reports. As the disengagement reports are written in regular human-readable language the aim is to use natural language processing (NLP) to be able to extract the information embedded in a disengagement report and to then programmatically produce executable test scenarios from the extracted information. The second primary outcome is this report which serves as the main documentation artefact of the application development.

What this project aims to study can be proposed as a technological rule:

> To automatically generate test scenarios for autonomous vehicles given disengagement reports use NLP.

To aid in this study, the report will also answer three research questions, which are as follows:

- RQ1 How usable are the disengagement reports, released by the California DMV from 2015 to 2022, for generating test scenarios?
- RQ2 How can NLP help to create test scenarios from current autonomous vehicle disengagement reports?
- RQ3 How usable are the generated test scenarios for testing autonomous driving systems?

These research questions are answered through either direct analysis and attempted usage of the data in the application (for RQ1 and RQ3), as well as through the knowledge gained from the process of trying to develop the application (for RQ2 and RQ3). RQ2 and RQ3 have a strong connection to the technological rule and their answers should therefore be able to support the proposed technological rule

1.3 Research Method

The main purpose of this project was to be a combination of both a problem-solving and an exploratory project. It is exploratory in the sense that there exists all this public disengagement data from the California DMV, and the project tries to explore its usability within a certain scope. As for problem-solving, the project attempts to use all of this data to provide one solution to the problem of generating test scenarios for autonomous vehicle testing. To conduct this project, design science was chosen as the primary methodology to be used. According to Runeson, Engström and Storey [10], the principles of design science match well to the aims of and research practice in software engineering. The aim is to follow the three fundamental elements of design science, problem conceptualisation, solution design and validation, in an iterative manner.

1.4 Result and Contributions

Although the total number of disengagement reports for the years 2015 to 2022 is high, 183,182, the absolute vast majority of them were not very usable in this project. For the comparatively few that were usable, the developed application is able to reconstruct correct and concrete scenarios for around half of them, whereas it produces non-complete scenarios or manages to extract some of the details for the other half. This project contributes insight into the disengagement report data and its usability within the scope of the project, as well as a concept application that recreates test scenarios for autonomous vehicles from disengagement reports.

1.5 Structure of the Report

The rest of the report is structured as follows. First, terms and concepts relevant to the project, as well as related work, are presented in Chapter 2. Thereafter, Chapter 3, describes the steps and work that has been done during the project and how the created application artefact functions. Chapter 4 contains the results of investigative work regarding the disengagement reports, the results of evaluating the application artefact as well as a general presentation on the application's performance. Discussion of the results, findings during the project and answers to the research questions are presented in Chapter 5. Finally, the report is concluded in Chapter 6 where further work is also pointed out.

Chapter 2
Terms and Related Work

2.1 Terms and Concepts

2.1.1 Natural Language Processing

Natural language (NL) is unstructured and because of that it is hard for a computer to process it directly. Over the years better ways to process NL into a more structured data form have been developed and also made it possible to understand complex relationships within NL texts.

Natural language processing (NLP) uses many techniques to be able to understand NL. One essential part of NLP is text pre-processing, with common steps of it being tokenization, removing stop words and stemming or lemmatisation [11]. Tokenization splits the text into a vector of words so they can be processed. Stop words, which are common words used in texts that do not have a significant impact on the text (i.e. an, is, the), are typically removed. After this, what is left is a reduced text but with most of the substance still there. Generally, the next step is stemming or lemmatisation. Both are ways to distil a word down to its root, for example, 'learning' comes from the root word 'learn'. Stemming takes a word and removes its ending, so 'learning' would be 'learn' and 'university' would be 'universe'. For simple words stemming works but since it is based on rules some words are not handled correctly. To remedy that it is common to instead use a lemmatiser. A lemmatiser has knowledge about words and maps them back to the actual root word, a good example here is 'went' which has the root word 'go' [11], [12].

After the text has been pre-processed it is ready to be analysed. There are many ways to analyse and extract information out of a NL text. Generally, they start with part-of-speech tagging which is a way to determine what part of speech a word is. While some words are always used as a verb and some are always used as a noun there are some that can be used as both depending on context. Part-of-speech tagging understands this and tags each word as it sees fit. For instance 'run' is a verb in 'run and fetch the doctor' while it is a noun in 'a mile run' [13]. After part-of-speech tagging comes dependency parsing. Within a sentence words grammatically depend on each other and knowing what that dependency graph looks like is useful for understanding a sentence. Each word can have one parent and multiple children, and each dependency is tagged with information about what the dependency is. For instance 'I saw a fox', 'I' would be a nominal subject to the verb 'saw' which would be the root of the graph [14]. The last common part is Named Entity Recognition (NER), understanding what a named part is referring to. For instance, for the sentence 'The Eiffel Tower is a tall tower in Paris, France.' NER will understand that 'The Eiffel Tower' refers to a building [13]. Further NER could also tag 'Paris' and 'France' as places. NER can also be trained to handle other types of entities depending on what is needed.

While not a common part, co-reference resolution is another tool that can be used in NLP. It identifies the connections between words which are referring to the same entity. For instance 'Anna was very sick; she needed her medicine.' would contain a connection between 'Anna', 'she' and 'her' as all of them are referring to 'Anna'. These connections make it easier to understand how different parts in more complex sentences are connected [13].

All of these parts are useful for understanding the meaning of a sentence and what is being spoken about. While these could be static logic, that has proven to be unstable to changes and a mess of edge cases. Instead, most of the new models doing the above analysis are based on trained statistical models, mostly using machine learning [13].

2.1.2 Disengagement Reports

A disengagement is the transfer of the control of a vehicle from the autonomous mode to a test driver/operator, during a test, either due to a technology failure or a situation that requires intervening for safety reasons [15]. Manufacturers of autonomous technology may conduct testing on public roads in California given that a certain set of requirements are met, with a subsection of them being the requirement to save data related to disengagements of the autonomous mode as well as submitting an annual report summarising this data [15] to the California DMV. Some of the data that the summary of disengagements should contain is the circumstances of disengagement such as the location where it occurred (interstate, freeway/motorway, highway, rural road, street or parking facility) and a description of the cause of disengagement (weather conditions, road surface, traffic conditions and similar) written in plain language so that a non-technical person can understand the circumstances [15].

These disengagement reports are publicly available at California DMV [16] and have been collected yearly over 8 years (2015-2022), though a small amount of disengagement reports exist for the year 2014 as well (but were included with 2015's year's reports). The total number of reported disengagements for each year of submittal as well as the number of unique manufacturers that submitted a non-zero amount of disengagement reports that year can be seen in Table 2.1. The exact information contained in a disengagement report, as well as the exact format the California DMV provide them in, has changed a bit over the years. For the years 2015-2018, the disengagement reports are provided individually per manufacturer in Portable Document Format (PDF) files, either as images of scanned documents or as tables. As for the years 2019-2022, the data is supplied individually per manufacturer in table format in PDF and Excel files, but it is also aggregated into Comma-separated Values (CSV) files combining all manufacturers' data. The year 2018 certainly stands out when it comes

Table 2.1: The total number of reported disengagements each year as well as the number of unique manufacturers that submitted a non-zero amount of disengagement reports each year.

Year	Disengagement count	Unique manufacturer count
2015	2,719	6
2016	1,503	9
2017	2,308	12
2018	152,459	31
2019	9,339	35
2020	3,726	29
2021	2,708	26
2022	8,420	25
Sum	183,182	52

to total disengagement count, however, the vast majority of them are either from Apple or UATC, LLC which submitted 76,585 and 70,165 respectively, with repetitive descriptions with a max length of three and five words.

The quality and quantity of information contained within different disengagement reports also vary greatly, mainly in terms of the description. Some of the disengagement reports give a rather clear understanding of the whole scenario that caused the autonomous driving mode to be disengaged whereas others are vague or lack significant details about the disengagement or, in some cases, have not been properly reported at all. Besides the description, what additional information was available also varied between years and manufacturers.

From the year 2018 and onwards the information to be included and how it was presented has been more standardised. In addition to information regarding the disengagement itself it now also includes information such as permit number, VIN, if the AV is capable of operating without a driver, if a driver was present during the disengagement and who initiated the disengagement. From the year 2019 and onwards, the information is presented in standardised CSV files but for the year 2018, the disengagement reports were presented in a standardised table format embedded in PDF files per manufacturer. As for the earlier years, these are also in PDF files per manufacturer, in a varying table format, but often with information such as permit number, VIN and driver-related facts missing. A few examples of individual simplified disengagements are presented in Table 2.2.

2.1.3 Scenario Standards: OpenDRIVE and OpenSCENARIO

To be able to represent the possibly complex scenarios that could be embedded in a disengagement report, and in a way that is easily integrable and usable by others, the description and formatting of the scenario need to be standardised. The Association for Standardization of Automation and Measuring Systems (ASAM) is a standardisation organisation for automotive development that 'focuses on standardizing the data exchanges between the many tools used in the process to develop and validate vehicles, their components, and their control systems' [17]. Two of these standards that are particularly relevant in this project are

Table 2.2: Examples of disengagement reports of varying quality and quantity.

Manufacturer	Year	Disengagement location	Description of disengagement
Nissan	2016	City Street	A large/tall trailer was stopping, the driver stepped on the brake because the driver felt a deceleration delay and driver safely disengaged and resumed manual control.
Apple	2017	Highway	Manual Takeover
Mercedes Benz	2019	Street	Driver performed steering maneuver because the vehicle didn't drive on the expected path. Vehicle not in an active construction zone. No emergency vehicles or collisions present in the vicinity. Weather and/or road conditions dry in the area.
Ridecell Inc	2020	Street	AV at a busy stop sign intersection. AV starts to move after the first cross vehicle turns in, however it comes to a stop when another vehicle approaches the cross stop sign. The second vehicle also turns in ahead of AV, as the AV response is slow and conservative. Safety driver intervenes once the 2nd vehicle took AV's right of way.
Apple	2021	Street	Undesirable motion plan resulted in incorrect vehicle position on roadway

OpenDRIVE and OpenSCENARIO. OpenDRIVE defines a file format used to describe road networks and surrounding objects [18] whereas OpenSCENARIO defines a file format used to describe the dynamic content such as vehicles and other traffic participants, how the aforementioned road users move around, weather and other conditions [19].

OpenDRIVE

The OpenDRIVE standard specifies a format using the Extensible Markup Language (XML) syntax, that can be used to describe the static parts of a scenario. It has the ability to describe regular roads and their lanes for cars as well as other road elements such as bike paths, tunnels and bridges, crossings etc. [18], [20]. Additionally, it can also describe other surrounding objects typically found along roads such as signs, traffic lights and parking spaces [18], [20].

Roads are described using reference lines, two-dimensional lines defined along a plane that can be made up of multiple line segments (i.e. a straight line followed by parts of a spiral followed by an arc) [20]. This practice of combining line segments is essential when creating more complex road networks. Reference lines are not allowed to have any leaps in them, and they are recommended to be kink-free [20], that is, the function describing the reference line should be differentiable. Using a reference line, a non-negative amount of lanes are then specified on either side of it to make up a road [20]. Roads can be elevated both in terms of height and roll, have non-flat shapes and complex surfaces and are linked together either through simpler predecessor/successor connections or more advanced junctions [20].

The other objects that can be described are either the type that changes the physical nature of the road or the type that controls traffic, called signals. The objects that change the roads are bridges and tunnels, parking spaces and traffic islands [20]. Signals are used for controlling road traffic and can be both static and dynamic, signs and road markings are static whereas traffic lights are dynamic [20].

OpenSCENARIO

The OpenSCENARIO standard specifies a file format that is used to describe the dynamic parts of a scenario [19]. There exist two major versions, v1.x (v1.2.0 latest) and v2.x (v2.0 latest). The core concept is the same in both versions but there are differences in how they are implemented. The standard that describes v1.x files, which are written in the XML syntax [19], requires all default values assigned, so the described scenarios are always concrete [21]. The standard that describes v2.x files are expressed in a domain-specific language (DSL), contains all the functionality of the v1.x standard, as well as additional functionality with the main feature being support for higher levels of scenario abstraction [21].

The core concept that is shared between both versions is the ability to describe the dynamic parts of a scenario, the complex movement of and interaction between multiple entities such as vehicles, pedestrians and other possible traffic participants as well as external conditions such as weather [19], [21]. Scenarios are built up in a storyboard, which consists of stories that are built up by acts [22]. By using this structure, simple scenarios such as one car overtaking another, or more complex scenarios involving multiple vehicles and their movement can be represented [22]. The movement of an entity can either be defined mathematically with trajectories or as an action, such as changing lanes [19].

2.1.4 Road and Scenario Viewers, and Simulators

After a road network or scenario has been defined in OpenDRIVE and OpenSCENARIO respectively, being able to view and run them and visually see what happens is important, mainly for the purpose of validation in this project. Tools and programs to aid in this process do exist, and a few of the available ones that have been relevant will be presented in this section.

esmini

One choice of scenario viewer is Environment Simulator Minimalistic (esmini), an open-source basic OpenSCENARIO player originally stemming from a Swedish research project called Simulation Scenarios [23]. It has support for v1.0 and v1.1 of OpenSCENARIO, although not all features defined in these versions are currently covered, for example, the support for traffic signals is lacking [24], as its further development after the research project finished is based on its users' needs [23].

CARLA and Scenario Runner

Another choice is Car Learning to Act (CARLA), an open-source simulator for the research and development of autonomous driving [25]. It is designed as a server-client system, where the server, implemented using Unreal Engine 4, is responsible for simulating the world, whereas the client, built using Python, operates as an actor within the world [25]. Additionally, to make CARLA run scenarios in the OpenSCENARIO format, the open-source extension ScenarioRunner [26] (which acts as a client) is needed. Similarly to esmini, CARLA also does not have perfect feature coverage with its main pitfall being that ScenarioRunner only supports v1.0 of the OpenSCENARIO standard [27].

OpenDRIVE viewer

A helpful tool when creating road networks and scenarios is OpenDRIVE viewer [28], a web browser-based tool that shows a simple view of a road network with details such as roads and lanes, and how they are connected.

2.2 Related Work

One essential part of this project is to create test scenarios for autonomous vehicles for simulators using real-world data. Two previously presented novel approaches that focus on generating critical test scenarios for simulators are Automatic Crash Constructor from Crash Reports (AC3R) by Gambi, Huynh and Fraser [5], [6] and Critical Scenario Generation (CSG) by Xinxin, Fei and Xiangbin [7]. Gambi, Huynh and Fraser [5], [6] present their approach, called AC3R, that reproduces real car crashes and generates test cases for autonomous vehicles based on police reports. The police reports are full of relevant details about the car crashes and contributing factors written by experts, but they are written in a mix of natural and structured language. By using NLP, AC3R is able to extract information and create accurate reconstructions of these real car crashes.

The other approach, CSG, that is presented by Xinxin, Fei and Xiangbin [7], is to use computer vision to extract a critical scenario from a video of a real-world traffic accident, and then reconstruct it as a scenario for a simulator. The toolkit they created spans the complete processing flow from scenario extraction to test scenario generation. Their data source consisted mostly of roadside surveillance cameras and in-vehicle-mounted cameras and almost only human-driven vehicles, although the additional sensors present in autonomous vehicles can present even more data that can be used in this approach.

Another essential part of this project is the disengagement reports that are published by California DMV. Lv, Cao, Zhao *et al.* [9] have analysed a subset of them, from the years 2014 and 2015. They looked at reasons for the disengagements and the surrounding conditions. In both disengagements initiated by the system and the driver, software issues and limitations were the main causes of disengagement. Their recommendations include for original equipment manufacturers (OEMs) to employ robust software design and validation processes to increase the safety of highly autonomous vehicles as well as analysing and utilising disengagements by for example running the scenario in simulation.

One challenge that exists in the space of testing autonomous vehicles is knowing how to measure the quality of driving of autonomous vehicles. Jahangirova, Stocco and Tonella [29] claim that the often-used metric of frequency of human intervention per distance driven is insufficient and can be misleading as it does not give a representative picture of the actual driving quality. Their contribution is a set of fine-grained metrics for determining driving qualities that can be used to create oracles for simulators. The metrics are based on literature and align well with what humans consider high-quality driving, and an oracle derived from them was able to differentiate between weak and robust driving models with little to no false alarms.

Wen, Park and Cho [30] propose a pipeline for generating scenarios that can be used for autonomous vehicle testing. Using a convolutional neural network, their generation pipeline is able to select appropriate agents and create realistic scenarios involving vehicles, pedestrians and animals, with high accuracy. The scenarios are generated around the autonomous vehicle using a scenario map consisting of scenario nodes that contain actions that must be invoked when the autonomous vehicle enters the area in the simulation. This means that, unlike [5]–[7], they create new theoretical test scenarios rather than reconstruct them from another source of data.

One other relevant project is the Carsim system described by Johansson, Williams, Berglund *et al.* [31]. It is a system that visualises car accidents by recreating them as 3D scenes from written texts about them in Swedish such as news reports. It extracts information, creates a structured representation and then generates a visual animation of the scene. NLP is used to extract information, although in a slightly older approach due to the age of the project, however, parts of it are still present in modern NLP. It was able to extract all relevant information correctly from about half of the texts it was evaluated on, showing that the approach was valid.

Chapter 3
Research Method

This project aimed to follow the design science research paradigm and three of its fundamental principles, problem conceptualisation, solution design and validation. Design science is a research paradigm that has the goal of solving practical problems by the creation and scientific study of artefacts, objects that are made with the intention of addressing these practical problems, as they are being developed and used by people [32]. The outcome of design science research is, in addition to the artefacts themselves, to also have the contextual knowledge about the artefacts as well as improving practices [32]. Problem conceptualisation is the activity of describing the problem, moving from the practical domain to the design domain. Solution design is the activity of designing and implementing a solution to the problem, moving back to the practical domain where the solution can then be validated [10]. After validation, the knowledge gained can then be used when doing problem conceptualisation again, starting a new iterative cycle. In this chapter there are step identification numbers, explained in 3.4, referencing different steps of the project.

3.1 Preparation

3.1.1 Importing Disengagement Reports

One large task at the beginning of the project was to take the disengagement reports, extract them from the file formats they were provided in and put them in a format that was easier to work with, constituting step P.1. Since the formatting differed slightly between years this process was not the same for all the disengagement reports. As for the target format, we opted for storing the disengagement reports in a database as it made it simple to interact with programmatically. The database of choice was SQLite, a self-contained and serverless SQL database engine that stores information directly on disk files [33], which allowed for high portability and simplicity as no external database server had to be set up.

Since the disengagement reports from the year 2019 and onwards were stored in CSV files, they were simply imported into the database (2019–2021 initially, and 2022 when it became available) using a program called DB Browser for SQLite, a visual tool for SQLite-compatible database file management [34]. The preceding years however required additional steps to be imported, and since they took significantly more effort to import, this is when the first filtering of disengagement reports took place. Before any of the ones from the year 2018 and before were imported, all the source files were looked through and each manufacturer was put into one of three categories based on the amount of information available in their disengagement reports and how useful their descriptions were. The primary focus was on whether the description contained information about the traffic scenario itself, what the autonomous vehicle was doing, what the conditions were like and how much information there was about external actors and their actions. The three separate categories a manufacturer could be assigned to were as follows:

1. Lack of data or no disengagements reported.
2. Not useful data (few word descriptions and/or mainly irrelevant).
3. Mostly very basic but sometimes usable.

The full notes on this process can be found in Appendix A.

Based on the investigation, only the disengagement reports from manufacturers that had been assigned category three were deemed worth spending time on importing. The PDF files that contained tables in non-image form were converted into Excel files using Adobe's tool for converting PDF to Excel [35]. After the conversion to Excel, they were double-checked and corrected so that there were the correct numbers of columns and content, followed by being converted into CSV files and lastly imported into the database. The process for the PDFs that were scanned documents was mainly the same as the Adobe tool can perform Optical Character Recognition (OCR), although the double-checking and correcting phase required significantly more manual labour since the tool performed less optimally at times. In total, 25,653 disengagement reports out of the possible 182,183 were imported into the database.

3.1.2 Evaluation of Potential Solution

The first major choice in this project was which solution candidate to use when it came to processing the descriptions. We identified three possible candidates, which were as follows:

1. Use the disengagement reports to try training an NLP model from scratch.
2. Use a prebuilt NLP model.
3. Try utilising the quite recently released GPT-3 model from OpenAI.

These candidates were evaluated in step P.3, and the solution candidate we chose was the second one, using a prebuilt NLP model, and this was because of two reasons.

The first reason was due to the available disengagement report dataset. The initial plan was to use the dataset to train an NLP model from scratch, however, doing this puts some requirements on the dataset. Although the imported dataset contained 25,653 disengagement reports there were only 2084 unique ones when considering all the fields present in a disengagement report, and out of those only 969 had a description that was 100 characters or longer. So the corpus would be quite small and make the training process hard. This made the option of training an NLP model infeasible. The next logical step was to use a prebuilt NLP model to process the text and use the output from that to build the scenarios. This was going to be faster and it has already been trained to a large corpus for multiple standard processing steps. A prebuilt model could have issues with the text if it was written in a more domain-specific way, like law text, however as most of the descriptions are written in simple English this should not lead to any issues with prebuilt models.

There was also the option to use another language model, GPT-3 from OpenAI. We tried that as well but the output was not much better than what we were already able to produce with a prebuilt model. While the natural language generation is nice, it still produces natural language and we would then need to process that again. We also tried to have OpenAI produce the OpenSCENARIO XML directly but it showed to not be a valid option. While GPT-3 was not able to do this at the point of testing we do believe that future models from OpenAI will be able to do this in the future. The infeasibility of using this approach is the second reason why using a prebuilt NLP model was determined to be the best approach.

3.1.3 Scenario Frameworks and Tooling

The next major choice was the scenario format, which standard/specification to conform to, also part of step P.3. Our research into the topic revealed that there is a de facto standard for cross compatibility, OpenSCENARIO accompanied by OpenDRIVE. Initially, we looked at the latest version of both, but as we looked closer at the simulators it became apparent that most of them do not support the latest versions. So to be able to run our scenarios, we decided to go with OpenSCENARIO v1.0 and OpenDRIVE v1.5, and the reason for that is *CARLA*.

CARLA (and in conjunction with the extension *ScenarioRunner*) was not our original choice of simulator for validating our generated scenarios but since *esmini* has a limited feature set *CARLA* seemed like a good alternative. Although it supports both traffic lights and signs, which *esmini* barely does, *CARLA* has its own problems regarding feature support with the biggest one being the lack of support for newer OpenSCENARIO versions, hence we generate scenarios targeting v1.0. Although we technically can generate scenarios targeting a newer version, it would mean that we have no visual tools to help validate the scenarios we create. So the primary choice of simulator was *CARLA*, but *esmini* has still been used occasionally when *CARLA* has presented issues. An upside to using the open standards OpenSCENARIO and OpenDRIVE is that we are not locked to using a certain simulator, if the ones we use receive updates or a new one is developed, switching between them is not an issue.

3.2 Implementation

3.2.1 Additional filtering

Before starting the development part of the implementation, the dataset required some additional filtering, step I.1 of the implementation phase. As stated before, some rough filtering was done previously when importing the dataset but even after that the disengagement report dataset still has some shortcomings. The main one is that there are quite a few rather short descriptions. It is quite clear that the shorter the description is, the less information it can contain. Two other shortcomings of the reports are that some do not contain any explanation of why the disengagement occurred, only that it did, and that some descriptions only mentioned what the testing vehicle was doing but nothing about the surrounding environment. These two shortcomings often mean that there are no concrete scenarios that can be produced from those descriptions.

To make the dataset easier for us to work with, the descriptions were extracted from the disengagement reports and duplicates were removed. Having two or more identical descriptions gives nothing of importance to our project, as the output for duplicates would be exactly the same. After this step, we could then move on to trying to deal with the shortcomings. Firstly, to deal with the main shortcoming that many of the descriptions were rather short, they were filtered based on their length in number of characters. A somewhat arbitrary limit of 100 characters was used to filter the descriptions. This was selected based on looking at a large number of descriptions and seeing how many words were needed to describe a scenario well, which was around 20 words in our findings. Since in the English language, the average length of a word is around 5 characters [36] that gives us around 100 characters. Describing the reason for a disengagement, especially with enough details to be able to construct a complete scenario from it, in only a few words, is rather difficult. Using this character limit, the set of descriptions was split into two sets. The set that contained descriptions with 100 or more characters was then used in further filtering.

To deal with the two other shortcomings, the lack of explanations about the disengagement reasons and the lack of information about the surroundings, we took another measure to further filter the set of descriptions. A set of criteria was created and used to grade the descriptions, from zero to three (with three being the best), on how well each description could be converted into a test scenario. The descriptions that were given grade three then became the new main target dataset to be used when developing the application, as they were known to be of comparatively high quality.

3.2.2 Road Sections

From the beginning of the development, it was clear that it would be infeasible to produce custom OpenDRIVE road sections for each scenario as the information given in the description most often is not enough to create a specific road section. Instead, a small set of predefined road sections were created, in step I.2, as the majority of the disengagement reports would be representable on these. The ones created in the beginning were a straight bidirectional one-lane road, a four-way junction, and a four-way junction with traffic lights. More were added as the project progressed but these were the base that we started from. Each of these OpenDRIVE files is accompanied by a small data file in JSON format that describes some aspects of the road section, such as possible start and end points for vehicles, positions and order of traffic lights and other points of interest, such as certain spots to stop at in junctions, that are useful when building the scenarios. Although this information can be read and calculated from an OpenDRIVE file itself, it simplifies the process as, for example, possible start positions can just be read from a JSON file rather than having to be calculated based on the information about the road section each time.

3.2.3 Application

The application developed for the project was written in the programming language Python and makes heavy use of the NLP framework *spaCy*. *spaCy* has multiple prebuilt components for NLP processing but also allows for building and training new models [37]. Alongside *spaCy*, another library, *textacy*, which is an extension to *spaCy*, was used for preprocessing and a small amount of analysis [38]. Lastly, to produce the XML for OpenSCENARIO and OpenDRIVE the library *scenariogeneration* was used, a library that supports both parsing and generating the XML according to the standards through an intuitive Python API [39].

Parsing the Description

While some pre-processing of the disengagement reports happened earlier in the process, there is one last bit that is done right before analysing a description. The last pre-processing is done with *textacy*, and it includes the normalisation of spaces, hyphens and quotation marks. Additionally, two other checks have been added to deal with some minor issues. The first one is replacing 'AV' with 'autonomous vehicle' as 'AV' sometimes causes issues for the NLP model. The second one is normalising the spelling of 'oncoming', as it is spelt a bit differently between texts. It is spelt as one word, as two words, or with a hyphen, which sometimes causes issues, especially with the co-reference resolution.

After this pre-processing, the description is ready to be processed using the NLP model using *spaCy*. With *spaCy*, five tasks are executed: part-of-speech tagging, lemmatisation, NER, dependency parsing, and co-reference resolution. The first four are common tasks that help to understand most of the text and how it all connects. The last one, co-reference resolution is a less common part that finds which words are referencing the same entity within the text, it allows the application to understand what entities exist without duplication.

After the description has been processed by *spaCy* it is returned as a list of tokens with data from each task attached. The tokens are wrapped in a main object called a document which contains some data regarding the whole text. Our parser then begins by reading the root token for each span of tokens, a group of tokens that *spaCy* has considered to be a sentence, and uses those as the starting points of the parsing. From these starting points, we can traverse the dependency tree. Traversal is done by iterating over the children of each token and calling the corresponding method of their general part-of-speech tagging value (i.e. noun, verb, adj). Each of these methods starts off by traversing its own sub-tree before handling the current token, causing the data to flow up towards the root of the tree. By letting the data flow upwards, the parser can build concrete manoeuvres based on sub-trees and have fewer issues figuring out the relationships between the identified data. The data being returned from each method can be data for manoeuvres like 'Creep' and 'DriveToEndpoint', data of interest like 'Numbers', 'Side' and 'Direction', or surrounding information like 'ParkedVehicles'. So using the manoeuvre 'DriveToEndpoint' with the added information of 'Direction' being 'Left' means that the entity that has this manoeuvre should drive from its starting point and take the left endpoint direction. For a junction, we would then start at one side of it and make a left turn. Entities like pedestrians, vehicles and cyclists are also handled and returned by the methods to be used for later manoeuvres. This approach is the one used in the second iteration of the application which is the result of step I.5.

The initial iteration, the result of step I.4, did most of this process differently, mainly the start points of traversal which ended up being only the verbs with a directly connected noun subject. It seemed that in most cases one could figure out the manoeuvres of each entity by just looking at those verbs, but as more cases were considered, two main points were discovered that led to the implementation in the current iteration. Firstly, some descriptions contained information such as location and entity descriptions written in a way where they were not connected to the verbs being in focus. Secondly, we quite often failed to traverse large parts of descriptions as the verbs' sub-trees rarely covered a lot. When the start point changed to be the root of each sentence more of the data had to be processed, but it resulted in a lot more information being extracted, and more reliably at that.

When a data part like 'DriveToEndpoint' has been found and there is an entity and a possible direction, all of these parts can be combined into a manoeuvre. For instance, we might have 'DriveToEndpoint', 'Direction(left)' and an entity called 'ego', which represents the need to create a manoeuvre for the entity 'ego' making a left turn.

Generating the Scenarios

This intermediate data structure that the parser produces, i.e. the manoeuvres above, is then used when generating the OpenSCENARIO files containing the output scenario. To simplify this process, we are using the *scenariogeneration* library that lets us define the scenario we want to reconstruct in an object structure using their defined Python classes. The classes represent different aspects of the OpenSCENARIO specification such as entities and manoeuvres. The object structure can then be converted into an OpenSCENARIO XML file by the library.

Each of the manoeuvres and entities that are contained in our intermediate data structure are then added to this object structure through a few different steps. Firstly, the generator ensures that there is an ego vehicle, which is the vehicle that can be controlled externally and is the main vehicle in the scenario. Since some descriptions do not describe the car to be autonomous or as a testing car when it is the only one being mentioned, we consider a lone car in a description to be the ego car.

After this, the generator figures out the rough location type, be it a junction, a straight road or a highway. Knowing this early on is important as all the positions used by entities depend on the road section that will be used in the scenario. When it is known what road section needs to be used then all of the metadata that accompanies the OpenDRIVE file can be loaded in and used. When checking the location, the generator also handles vehicles that have custom starting positions, such as 'StartPositionRelative', which means that the start position of the entity should be relative to another car.

Entities, which can be both vehicles and pedestrians, are then created next and have their initial positions set. Currently, all entities, even the ego vehicle, receive a basic controller that listens to everything described in the scenario and follows it. Lastly, the generator iterates over each manoeuvre and produces OpenSCENARIO manoeuvres for the actions and conditions that the parser has produced. This includes: driving to an endpoint, creeping, halting for entities, and waiting for traffic signals. On top of actions and conditions, there is also extra data such as cars being parked at the endpoint, these cars are not moving entities and are thus not represented with their own manoeuvres.

As the descriptions are not always very detailed there are some fallback default values for when information is missing, as the older version of OpenSCENARIO we are using is not able to create abstract scenarios; they have to be very concrete. If there is no mention of a location type the generator considers it a straight road. No weather information is currently handled so the default is a sunny day. One major default value that is implemented is that if a vehicle has no actions attached to it, it gets a default action of driving to its forward endpoint.

3.3 Testing and Evaluation

During the earlier stages of the implementation, we did not conduct any strict and formal testing and evaluation of the application. Since it was developed by trying to get one new description to work one after the other there was no point in evaluating the application's performance on a larger scale using a part of the disengagement report dataset as the result of it would just confirm what was already known, the application was unable to handle any general disengagement reports. So the general approach to testing was to just run the application on a new description that had not been tested before, check why it failed and then use this new description as the main focus of further development.

For instance, when the application was fed a new description we were able to compare its output scenario with the scenario we expected to see, as we can understand the descriptions ourselves. If there was a problem, it often had to do with missing entities and actions. We would then use a set of debug outputs to see what information the application was able to extract and why it was unable to extract the rest of it. The application would then be changed and tested again, and this iterative process would continue until the scenario was fully functioning. During this, all previous scenarios were also tested to make sure that no current functionality broke.

However, when the development had reached a point where the application could now handle a few different disengagement reports the first proper round of evaluation was performed in step E.1. The application was tested on the set of descriptions that had received a scenario grade of three. The evaluation aimed to investigate how well the application was able to handle any given general description, which made certain aspects interesting to look at. Firstly, we looked at whether the application was able to generate a correct scenario for any of the descriptions that have not been part of the development set. Secondly, we investigated how close to a correct scenario the generated output of the application was and what issues prevented a correct scenario from being generated. Lastly, we checked for issues that were common and occurred for a large part of the testing data.

The knowledge gained in the evaluation was then used in the next iteration of application development where the approach to the natural language parser part of the application was fundamentally changed. After this major change, the performance of the application was evaluated again, in step E.2, using the same method as the first evaluation. Additionally, in the second evaluation, some extra focus was put on trying to identify general issues with the application, and seeing if they were present in the first evaluation as well. This was done to try to figure out how complicated they would be to fix, and what prevented us from fixing them while developing the second iteration.

3.4 Summary of Research Method

The steps that were taken in this project can be classified into either one of three different categories, preparation, implementation or evaluation. The focus of the preparation steps was to lay a foundation for future work in the project and try to conceptualise and understand the problem we were working on. The implementation steps constituted the work of trying to design and implement a solution to the problem that could then be validated and evaluated in the evaluation steps. A flowchart that shows the approximate steps that have been taken in this project and their relation to each other can be seen in Figure 3.1. In the top left corner of each rectangular box representing a step in the project, there is a step identification number. P.1-3 is preparation steps one to three, I.1-5 is implementation steps one to five, and E.1-2 is evaluation steps one and two. These identification numbers are used in Chapter 3 and 4 with the intent to give a clearer mapping as to which step resulted in which presented result.

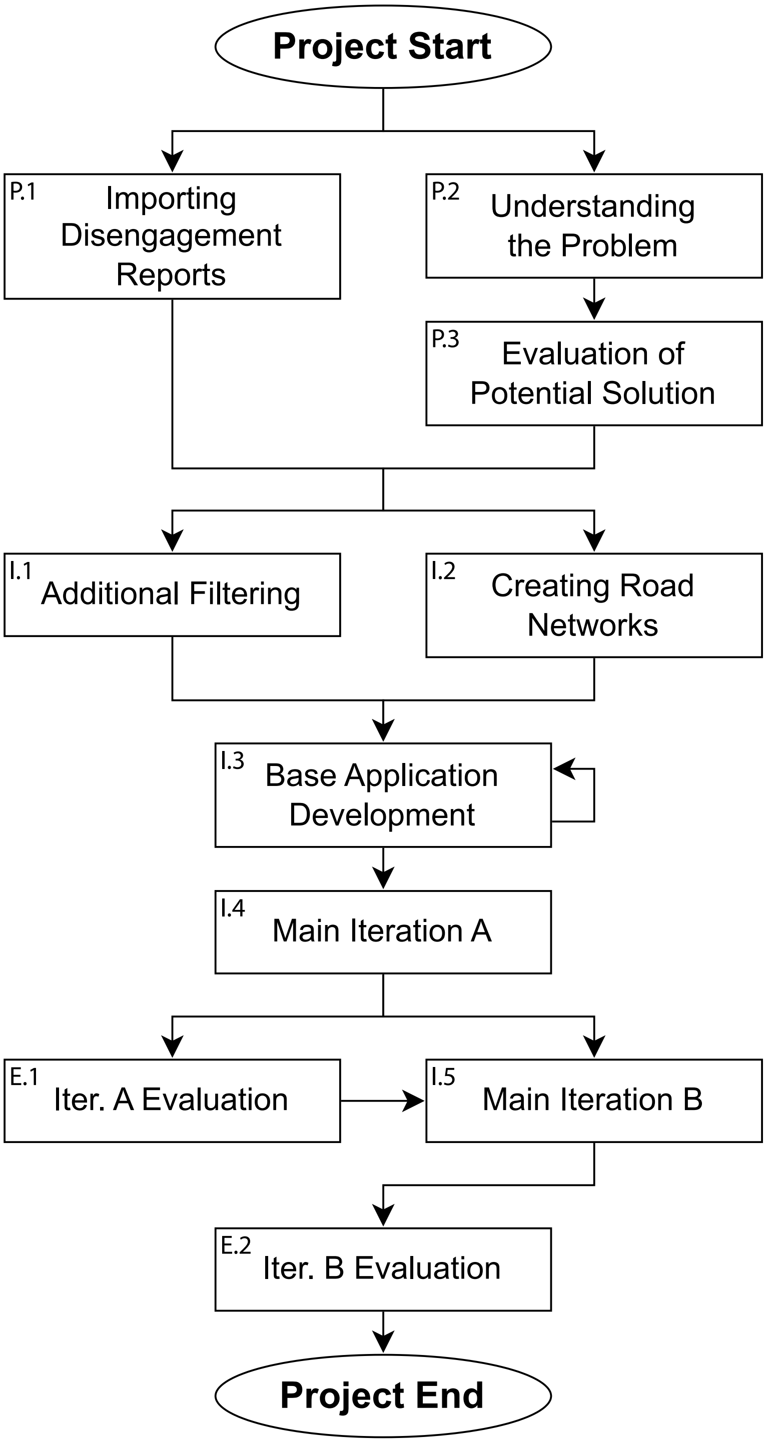


Figure 3.1: A flowchart showing the approximate steps that were taken during the project.

Chapter 4
Results and Analysis

4.1 Results

The first result that we present in this section is the grading of the disengagement report descriptions as it gave an important and quantifiable insight into the quality of the available data. After that, the results of the iteration evaluations and the general issues that we have been able to identify are presented. Lastly, two examples of descriptions that have been converted into scenarios are shown. To aid in showing which result comes from which step in the research method, the step identification numbers, explained in 3.4, are used.

4.1.1 Disengagement Report Description Grading

The work of extracting descriptions and removing duplicates, which was part of step I.1, resulted in a total of 1832 unique descriptions. Out of these, 872 were 100 characters or longer while the remaining 960 were shorter than 100 characters. The criteria that were specified in step I.1 to be able to consistently grade a description's so-called scenario score are shown in Table 4.1. The grade is used as an indication of how well the description describes a traffic scenario, as well as if there is enough information available for the scenario to be reconstructed as a concrete OpenSCENARIO scenario. Furthermore, one example description from each grade along with a motivation as to why it has received that grade can be found in Table 4.2. The result of manually grading those descriptions that were 100 characters or longer, part of step I.1, showed that the vast majority of them did not have enough information in them to represent a concrete traffic scenario as seen in Table 4.3, as only 36 out of the 872 were given grade 3.

Table 4.1: The criteria used to grade a description's scenario score.

Grade	Description of Criteria
0	Does not represent a traffic scenario at all or is missing all/majority of relevant information.
1	Contains some of the information needed to get a basic understanding of why the disengagement occurred but lacks further information to represent a concrete traffic scenario.
2	Contains information to get a somewhat clear representation of what the scenario would entail but still lacks details that are necessary to make the scenario more concrete.
3	Contains enough information to extract a mostly complete and concrete scenario where multiple actors' locations and actions are known or extractable from the description.

Table 4.2: Examples of given grade and motivation for a few descriptions.

Description	Grade	Motivation
Invalid perception result: AV stopped since invalid size of object was detected. Weather: Sunny. Road Condition: Dry.	0	Know it failed to detect the size of something, but no other information than that so impossible to create a scenario from it. Conditions are not relevant if other is not relevant.
Control Issue: Lateral Control performance was not ideal at the time, causing Test Vehicle to oscillate within lane and needed fine tuning. Test Driver disengaged to log issue for further examination.	1	Knowing the car oscillates within its lane but not anything else, would mean introducing the fault into the scenario itself.
Planning discrepancy; system planned incorrect trajectory based on a pedestrian making an illegal crossing.	2	The AV couldn't plan its route due to the pedestrian, but since we have little knowledge about how/where the pedestrian made an illegal crossing (at intersection or not etc) it's somewhat hard to know what concrete scenario to create.
Planning; as a precaution, the safety operator intervened to avoid a vehicle that illegally ran a red light from a blind spot occluded by a truck in Autonomous Driving Vehicle's neighbor lane.	3	Clear information that we are at an intersection with lights and that the AV could not see the vehicle making the illegal manoeuvre (which would be on a perpendicular road from the AV) due to a truck in the neighbour lane blocking the view. The only missing information is which side of the car the truck is.

Table 4.3: The number of disengagement report descriptions that received each grade.

Grade	Number of Descriptions
0	164
1	430
2	242
3	36
Sum	872

4.1.2 OpenAI Results

When using the GPT-3 model from OpenAI, we found that it was not able to produce correct OpenSCENARIO files in an XML format. Generally, it had issues that the syntax of the generated XML was incorrect and not according to the OpenSCENARIO standard as well as missing various elements that would be required to get a concrete and executable test scenario. In the evaluation of it as a solution candidate, step P.3, it was shown not to be a valid option as its abilities were lacking in multiple areas. Examples of prompts and the issues that were found in its responses can be seen in Table 4.4.

4.1.3 Main Iteration Results

Iteration A

After the first main iteration, step I.4, the application was able to correctly convert six descriptions into scenarios out of the 36 that were the main focus. Out of the remaining 30, the application generates executable scenarios on a straight road (either correctly or incorrectly) for around one-half of them and it fails to generate executable scenarios for the other half. In those that the application does generate a scenario for, the main issue is that the parser fails to extract a lot of relevant information correctly, such as manoeuvres, parked cars, other entities and even the AV itself if it is not clearly mentioned in the description. One specific issue that has been identified is that when it fails to detect the AV in the sentence but another vehicle has been identified, it converts the other vehicle into the ego vehicle of the scenario as that is its defined default behaviour, even though the other vehicle has been correctly identified and received its manoeuvres.

As for the descriptions where the application fails to generate executable scenarios the main issue is that the parser fails to extract the AV's manoeuvre, often because it is mostly implicit in the description rather than explicit. Other common issues are that the parser fails to detect entities and their actions/manoeuvres and that the application tries to create manoeuvres for turning left or right on a straight road in cases where the location of the scenario has not been correctly identified. So generally, the parser is unable to extract quite a lot of the information that is embedded in the description. To which degree it is unable to vary, but it is with the descriptions that are most unlike the six main target descriptions it fails the most. The full evaluation comments on all of the 36 scenarios that are the result of step E.1 can be seen in Table B.1 in Appendix B.

Table 4.4: Three examples of OpenAI's GPT-3 being prompted to create OpenSCENARIO files from a given description.

Prompt	Notes & Issues
Given the following description, generate an OpenSCENARIO version 1.0 XML file describing the scenario. Description: AV at a junction where AV has the right of way to move forward. Oncoming car cutting across does not come to a complete stop. Prediction module appears to cause AV to stop. Safety driver does a preventive intervention to continue flow of traffic.	The output is missing *FileHeader* with attributes, also *Actors* is not a keyword. It looks like it has some understanding of the scenario but is missing knowledge about OpenSCENARIO. Full output in Appendix E.
Given the following description, generate an OpenSCENARIO version 1.0 XML file describing the scenario. Include Act, ManueverGroup, Manuever, Entities, FileHeader, ScenarioObject, Storyboard, Story and other required parts. Description: AV at a junction where AV has the right of way to move forward. Oncoming car cutting across does not come to a complete stop. Prediction module appears to cause AV to stop. Safety driver does a preventive intervention to continue flow of traffic.	Tried to give it some more hints of types and required things. Still not using the header correctly and the syntax is still incorrect. Full output in Appendix F.
Given the following description, generate an OpenSCENARIO version 1.0 XML file describing the scenario. Consider the OpenSCENARIO 1.0 specification when generating the XML. Description: AV needs to make a left turn in narrow street with parked vehicles on right side after the turn. The planner is not able to create a consistent path around parked vehicles in to the turn. It keeps switching between finding a path and requesting AV to come to complete stop, at edge of decision boundary, resulting in AV to start slowing down. Safety Driver takes over to maintain a smooth traffic flow.	Syntax missing and continually missing the header. Full output in Appendix G.

Iteration B

After the second main iteration, step I.5, the application was now able to correctly convert 15 descriptions into scenarios, and additionally two semi-correct ones, out of the 36 that were the main focus. For the semi-correct scenario conversions, one is being reconstructed correctly except for an additional vehicle being added due to co-reference resolution issues, and the other would be correctly converted if the needed road section was available. As for the remaining 19 descriptions, the application generates executable but more or less incorrect scenarios for 15 of them, often containing many of the default assumptions, with three of them being incorrect due to co-reference resolution issues. For the last four descriptions, the application fails to generate executable scenarios due to failure to identify and extract any vehicles and their manoeuvres. Many of the incorrectly converted descriptions result in straight road scenarios, although some that need to be on a junction road section get correctly identified when they previously did not.

Similarly to the previous iteration, the application still has the general issue in that it struggles to identify actions performed by either the autonomous vehicle or other actors when they are rather implicitly defined, i.e. not clearly explained in the descriptions. Another general issue is that it still sometimes misses that the scenario occurred in a junction and that traffic lights are involved. In some cases, the parser part of the application correctly identifies the important information in the description for recreating the scenario, however, the rest of the application lacks proper support for reconstructing the scenario, with one example being pedestrians. No new issues have arisen in this second iteration; some of the ones from the earlier iteration have been fixed while some still persist. The full evaluation comments on all of the 36 scenarios that are the result of step E.2 can be seen in Table B.2 in Appendix B.

4.1.4 Natural Language Processing Issues

Co-reference Resolution Issues

One general issue the application struggles with occasionally is co-reference resolutions. Most of the time it performs adequately, but there are some descriptions it has shown to have trouble with. The description

> **AV** at a busy stop sign intersection. **AV** starts to move after the first cross vehicle turns in, however **it** comes to a stop when **another vehicle** approaches the cross stop sign. The **second vehicle** also turns in ahead of **AV**, as the **AV** response is slow and conservative. Safety driver intervenes once the **2nd vehicle** took **AV**'s right of way.

is one example of a description that is problematic. The blue and green colouring in the description shows what the application believes are co-references to the same entity reference. The section marked in red is what the co-reference fails to detect, which is meant to be grouped as the green entity.

A second example of a co-reference resolution failure is

> **AV** has just made a right turn and there is an **oncoming vehicle** in the opposite lane, resulting in a narrow path between parked car(on the right) and **on coming**

> **vehicle**(on the left) for the **AV** to follow. **Safety driver** initiated the take over, when **he** noticed the steering move to the left. Cause: Wrong Prediction of **oncoming vehicle** trajectory for a small time instant causing **the vehicle** to move in the wrong direction for that time instant.

where it fails to detect the oncoming vehicle as the same entity. The co-reference of entities marked by the blue and green colour is correct, however, the entity marked in red is what the application fails to understand are co-references.

Lemmatiser Issues

The lemmatiser is supposed to produce the root of a word and in most cases it does. However, at some point, the lemmatiser acted more like a stemming parser as it took words like 'oncoming' and converted them to 'oncome'. For the application, it was not much of an issue as it just needs to consider both of those lemmas as correct, but it does make it uncertain if other words will have a similar problem later and make the application fail to extract information.

4.1.5 Example Scenarios

Example 1 For the given description,

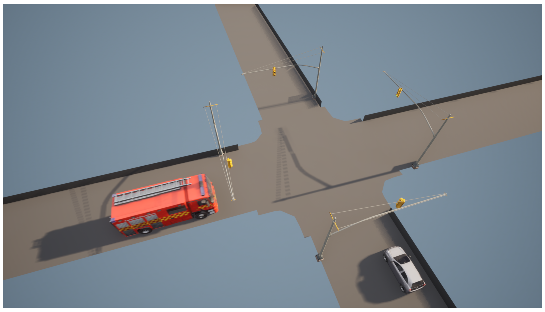
> While the testing car was waiting for green light at an intersection, a long truck blocked both traffic lights at the same time when passing through. The testing car failed to detect any red light and launched forward. The driver took over.

the application is able to extract that the AV is waiting for a green light at a junction but its view of the traffic lights gets blocked by a truck passing through. It can deduce that the truck is driving on a road that is perpendicular to the one the AV is on and it generates a scenario based on this information. It is also able to disregard the last two sentences in the description; since they are not relevant to the scenario construction. The OpenSCENARIO file that is generated contains two vehicles, the ego car and a fire truck (used due to lack of options in regards to *CARLA*), an act consisting of two manoeuvre groups, one controlling each vehicle. The OpenDRIVE file that is referenced is a four-way junction with traffic lights, one of the prebuilt ones the application selects from.

The scenario starts out with both vehicles at the junction (shown in Figure 4.1a). The traffic signals first show green on the road the truck is on, causing it to start going straight through the junction (Figure 4.1b). Meanwhile, the ego car is waiting for the traffic lights to turn green (Figure 4.1b), and its vision of them (as they are on the opposite side of the junction) is obscured as the truck drives through the junction (Figure 4.1c). Finally, as the truck has passed through the junction the car regains its vision of the traffic signals again (Figure 4.1d). The complete XML file that has been generated can be found in Appendix C.

Example 2 For the given description,

> AV at a junction where AV has the right of way to move forward. Oncoming car cutting across does not come to a complete stop. Prediction module appears to cause AV to stop. Safety driver does a preventive intervention to continue flow of traffic.



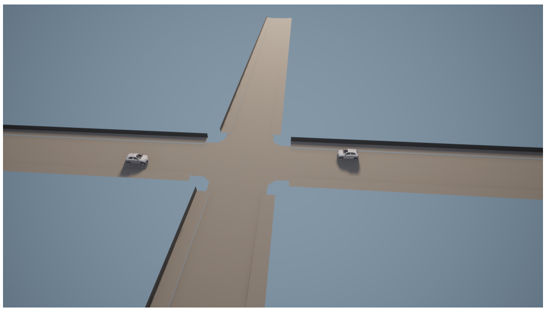
(a) Starting position of both vehicles in the scenario.

(b) The traffic signal shows green on the road the truck is on.

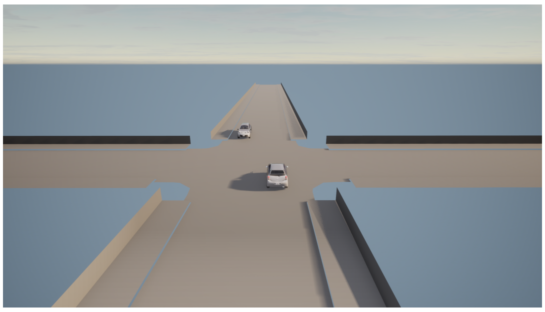
(c) The car's vision of the traffic signal becomes obstructed by the passing truck.

(d) The truck has passed and the car is able to see the traffic signal again.

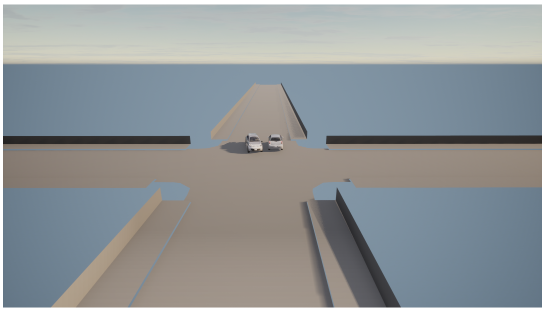
Figure 4.1: Images showing an overview of the steps that happen in the example scenario 1.



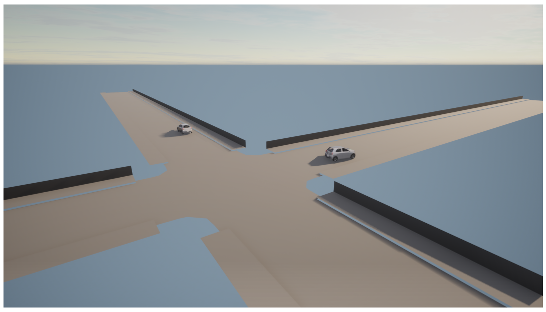
(a) Starting position of both vehicles in the scenario.



(b) Both vehicles drive towards and enter the junction.



(c) Oncoming car does not come to a complete stop.



(d) Both vehicles leave the junction after the interaction.

Figure 4.2: Images showing an overview of the steps that happen in the example scenario 2.

the application now produces a scenario involving two cars, at a junction without any special signs or signals, another available preset. The cars start at opposite roads at the junction (shown in Figure 4.2a). Both cars drive into the junction (Figure 4.2b), the ego car continues straight forward while the other car slows down to turn after the ego car has passed, but without stopping (Figure 4.2c). Afterwards, both the vehicles leave the junction as expected (Figure 4.2d). Again, the application is able to disregard the last two sentences that are irrelevant in constructing the scenario. The complete XML file that has been generated can be found in Appendix D.

4.2 Analysis

When looking at the available disengagement report data it becomes clear that it is very repetitive and in the vast majority of cases does not contain enough information to be able to reconstruct the situation causing the disengagement. In the cases where there is enough information, a somewhat accurate test scenario can be reconstructed, although with some simplifications due to the lack of information regarding the surroundings or by using inaccurate vehicle models due to what is available in the base catalogue in *CARLA*. The lack of information also affects more direct parts of the scenario, for example, the number of lanes on roads and junctions is typically not mentioned. A lot of assumptions regarding such parts of the scenarios sometimes have to be made to be able to create concrete scenarios.

Some of the descriptions that are not able to be accurately reconstructed as a scenario are not due to missing information or difficulties extracting the information from the description, but rather a lack of support in the application. This is mainly because of either two reasons. The first is missing support in the scenario-generating part of the application, and the second is the lack of correct road sections that match the disengagement's needs. Both of these reasons are down to time limitations when conducting the project rather than it not being a possibility.

Although the latest iteration still has issues with extracting all relevant information in a description, it has become a bit better and is able to find some additional information compared to previously. However, descriptions where various information is implicitly expressed rather than explicitly have been shown hard to deal with. When using either of the approaches to extracting information from the two application iterations, both of them struggle. They are very direct in their approaches since they primarily focus on the words that are present in the description and the literal information they provide rather than what is implicitly understandable.

Chapter 5

Discussion

As evident in the result section, the vast majority of the unique descriptions do not contain enough information to generate concrete test scenarios. An important aspect to consider when discussing the usability of the disengagement reports is that the evaluation was done on the set of unique descriptions, and that selection is not based on the complete dataset either. The available data is very repetitive, although some manufacturers are better than others. The investigation into the dataset revealed that those that produce the most disengagement reports also tend to be the most repetitive. This leads us to the first research question of this project.

RQ1 How usable are the disengagement reports, released by the California DMV from 2015 to 2022, for generating test scenarios?

When it comes to creating test scenarios for autonomous vehicles, although we have been able to reconstruct some concrete scenarios from the disengagement reports, this is only from a very small percentage of them. So the vast majority of them are not usable from this perspective.

Creating a concrete scenario when information is missing can be difficult as assumptions will need to be made. Certain parts are easier than others, such as if the description does not mention the location where the disengagement occurred, and there are no other hints about it, it was probably on a straight road, or if the location is mentioned, but not the number of lanes, the possibilities of how many lanes there could be are still few. On the other hand, if the description for example mentions that there is surrounding traffic but no further details, that still leaves large room for different possibilities and interpretations. If it is light or heavy traffic, where exactly in relation to the autonomous vehicle the rest of the traffic is, and so on, is now unclear information.

However, this lack of information can give a natural push towards being more abstract when recreating the test scenarios, by creating multiple scenarios from the same disengagement report by varying the unknown parameters. For example, instead of just assuming a certain number of lanes, create separate test scenarios for different amounts of lanes, or change how light or heavy the traffic is. Additionally, the generated scenarios that are not fully complete can still be used as simple templates serving as a basis when creating more refined scenarios manually. Using such techniques would probably increase the usability of the disengagement reports that are lacking information with respect to this project.

RQ2 How can NLP help to create test scenarios from current autonomous vehicle disengagement reports?

Looking purely at the results, the use of NLP seems like a very helpful tool to produce scenarios from autonomous vehicle disengagement reports. Although the useful part of the dataset is currently not as large as one might expect based on the raw numbers, NLP has still shown to be a key tool in generating scenarios from the small subset of the dataset that was deemed usable. There are still parts which can be better, for instance, the co-reference resolution did not always find the correct connections and the lemmatiser was not always perfect. One way to improve on these issues would be to train the language model on an additional dataset of annotated disengagement reports to allow it to learn more about the specific style and norms in this context. We do not think it would change the outcome significantly in the general case as the text is normally written without the use of domain-specific language. However, in more specific parts, such as co-reference resolution, the benefit of doing this additional training could be larger. Additionally, training separate categorisers for certain aspects, such as locations and entities, could be helpful in extracting specific information which would simplify the necessary tasks the main parser has to complete.

NLP has been a great tool for our use in this project, and as the field keeps progressing, the precision and quality will only become higher. Looking further, even if it is not fully usable in the current state, a tool like OpenAI's GPT-3, and by extension ChatGPT, could hopefully make parts of this process easier as those types of NL processing and generation tools improve further. Until then, the base of NLP output of structured data can be used effectively to generate scenarios if the current application is developed further.

RQ3 How usable are the generated test scenarios for testing autonomous driving systems?

As we have shown earlier, the generated scenarios are executable and produce fairly simple traffic interactions that manage to represent what is described by the disengagement reports. However, the scenarios are probably not complex enough to be considered adequate for more rigorous autonomous vehicle testing as there is not enough happening around the vehicle and the surroundings are missing a lot of details. Thus, the scenarios are not able to fully represent the real-world traffic scenarios that the vehicle would have been experiencing. In other types of testing, when the complexity is not as relevant, where you might want a more controlled environment within a specific test, the generated scenarios should be adequately usable. The project has not included any autonomous driving system that is used for autonomous vehicle testing. Instead, only the built-in simple controller in *CARLA* has been used when evaluating the test scenarios. This controller only serves the purpose of making a vehicle drive from point A to B by following the road network while ignoring other aspects such as traffic rules unless specified. Due to this, it is difficult to provide a complete answer to this research question as we have not been able to evaluate the generated scenarios' usability in a more complex testing environment.

As mentioned in the answer to RQ2 above, some of the scenarios that are generated by the application might not be fully concrete and lack details. However, they could still be used as a starting point when creating more complex scenarios as the basis of the complex scenario is already generated. This could help reduce the resources needed to manually create test scenarios and make it more efficient if there is a large number of disengagement reports that need to be converted into scenarios manually.

Further Discussion

The data that is used as input to the application is the description of each disengagement, which is not all of the data that is available. Alongside the description, there is also separate data regarding the location of the disengagement, such as street, freeway, etc., although this was never used in the application as the focus was to develop the functionality to maximise the knowledge extracted from the descriptions. We do consider this data valuable although not always the most useful. In most cases, the context and actions mentioned in a description give the same information or better about the location of the disengagement. Even if it occurred at an intersection, the location field would most times just say 'Street' which is not exact enough to be better than the description nor does it provide much additional information. It might be helpful to take as extra input to the application and subsequently be more constrained in location selection but it did not feel necessary in this current iteration of the application as it only provides a slight improvement. There are some disengagement reports that have additional fields, such as 'Condition', that describe the weather and road conditions. These fields, however, are not standardised in the recent specifications about what data a disengagement report should contain and therefore only exist on some of the earlier disengagement reports. This means that it would not be very impactful to implement the usage of it when reconstructing a general scenario.

Looking further at the parsing of the NLP output we found that with our approach it is hard to connect an entity to its actions when there are many entities and actions in a description. There should be ways to solve this, in some cases we should be able to make the connections further down the grammar tree and not rely on the data to go all the way up the tree before being handled. Additionally, there are cases where the application ends up with multiple entities for a single action where it picks the first entity found rather than necessarily the correct one. To solve this, we could also have metrics of the distance between actions and entities to always pick the closest ones when connecting them. Anything done here could still create issues but proximity should be a good indication of a connection.

Early in this paper, in Section 1.2, we proposed a technological rule that serves to express the aims and contributions of this project:

> To automatically generate test scenarios for autonomous vehicles given disengagement reports use NLP.

The answers to RQ2 and RQ3 provide support for the proposed technological rule. Although having a fully automatic generation of a complete test scenario is not always possible, at least in the current state, automatically generating an incomplete base scenario is still a helpful tool. The use of NLP has shown to be important for the functionality of the application as what it provides would be difficult to create a sufficient replacement for.

Chapter 6
Conclusion

When investigating the task of trying to automatically generate test scenarios from disengagement reports it is evident that the available dataset from the California Department of Motor Vehicles is lacking in terms of quality, mainly when it comes to the information that is presented in the description of a disengagement report. Despite this issue, when the quality of the description is adequate, the process of generating concrete test scenarios based on disengagement reports is able to be automated to some extent, and when the available information is lacking, a simple basis for a test scenario can still be created.

Broadly speaking, the developed application only works effectively on the disengagement reports that were the main focus, or similar to those, during the development process. While it still has issues in its latest iteration, the application is much better at extracting relevant information and attempting to represent it in a scenario than what the penultimate iteration was. There are a lot of traffic scenarios that require functionality currently not implemented in the application and road sections that have not been created and added to its library, but extending the application and its library is one rather straightforward aspect that could be improved upon in future work. Improving certain aspects of the NLP being used, such as better co-reference resolution and using classifiers to extract information in different ways is another aspect of the application that can be worked on. With a more refined implementation, that has been improved upon further, this type of approach should be a viable option for automatically recreating test scenarios from disengagement reports.

Based on this project, we see many opportunities for future work. One opportunity is to improve the application further, refining it and its abilities. Another opportunity would be to explore the idea of using a similar approach to create novel test scenarios based on natural language input from a human user theorising new possible scenarios an autonomous vehicle could find itself in. There is also an opportunity for additional research into projects such as OpenAI and its GPT models as they improve to see if they can be used as is or fine-tuned into understanding and producing adequate test scenarios from natural language and disengagement reports. These ideas for future work should aid in the goal of creating rigorous tests for autonomous vehicles to increase their safety in the future.

References

[1] M. Ahmed, R. Iqbal, S. Amin, O. Alhabshneh and A. Garba, 'Autonomous vehicle and its adoption: Challenges, opportunities, and future implications', in *2022 International Conference on Emerging Trends in Computing and Engineering Applications (ETCEA)*, IEEE, Nov. 2022. DOI: 10.1109/etcea57049.2022.10009804.

[2] Đ. Petrović, R. Mijailović and D. Pešić, 'Traffic accidents with autonomous vehicles: Type of collisions, manoeuvres and errors of conventional vehicles' drivers', *Transportation Research Procedia*, vol. 45, pp. 161–168, 2020. DOI: 10.1016/j.trpro.2020.03.003.

[3] J. Gao, W. Wu and O.-E.-K. Aktouf, 'Adequate testing unmanned autonomous vehicle systems - infrastructures, approaches, issues, challenges, and needs', in *2022 IEEE International Conference on Service-Oriented System Engineering (SOSE)*, IEEE, Aug. 2022. DOI: 10.1109/sose55356.2022.00025.

[4] A. Abraham, S. C. Nagavarapu, S. Prasad, P. Vyas and L. K. Mathew, 'Recent trends in autonomous vehicle validation ensuring road safety with emphasis on learning algorithms', in *2022 17th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, IEEE, Dec. 2022. DOI: 10.1109/icarcv57592.2022.10004304.

[5] A. Gambi, T. Huynh and G. Fraser, 'Automatically reconstructing car crashes from police reports for testing self-driving cars', in *2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*, IEEE, May 2019, pp. 290–291. DOI: 10.1109/icse-companion.2019.00119.

[6] A. Gambi, T. Huynh and G. Fraser, 'Generating effective test cases for self-driving cars from police reports', in *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, ACM, 12th Aug. 2019, pp. 257–267. DOI: 10.1145/3338906.3338942.

[7] Z. Xinxin, L. Fei and W. Xiangbin, 'Csg: Critical scenario generation from real traffic accidents', in *2020 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 19th Oct. 2020, pp. 1330–1336. DOI: 10.1109/iv47402.2020.9304609.

[8] F. Favarò, N. Nader, S. Eurich, M. Tripp and N. Varadaraju, 'Examining accident reports involving autonomous vehicles in california', *PLOS ONE*, vol. 12, no. 9, e0184952, 20th Sep. 2017. DOI: 10.1371/journal.pone.0184952.

[9] C. Lv, D. Cao, Y. Zhao *et al.*, 'Analysis of autopilot disengagements occurring during autonomous vehicle testing', *IEEE/CAA Journal of Automatica Sinica*, vol. 5, no. 1, pp. 58–68, Jan. 2018. DOI: 10.1109/jas.2017.7510745.

[10] P. Runeson, E. Engström and M.-A. Storey, 'The design science paradigm as a frame for empirical software engineering', in *Contemporary Empirical Methods in Software Engineering*, Springer International Publishing, 2020, pp. 127–147. DOI: 10.1007/978-3-030-32489-6_5.

[11] C. P. Chai, 'Comparison of text preprocessing methods', *Natural Language Engineering*, vol. 29, no. 3, pp. 509–553, Jun. 2022. DOI: 10.1017/s1351324922000213.

[12] T. Korenius, J. Laurikkala, K. Järvelin and M. Juhola, 'Stemming and lemmatization in the clustering of finnish text documents', in *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, ACM, Nov. 2004. DOI: 10.1145/1031171.1031285.

[13] P. Nadkarni, L. Ohno-Machado and W. Chapman, 'Natural language processing: An introduction', *Journal of the American Medical Informatics Association*, vol. 18, no. 5, pp. 544–551, Sep. 2011. DOI: 10.1136/amiajnl-2011-000464.

[14] S. Jaf and C. Calder, 'Deep learning for natural language parsing', *IEEE Access*, vol. 7, pp. 131 363–131 373, 2019. DOI: 10.1109/access.2019.2939687.

[15] *Article 3.7. Testing of Autonomous Vehicles*, State of California, 13th Apr. 2022. [Online]. Available: https://www.dmv.ca.gov/portal/file/adopted-regulatory-text-pdf/.

[16] California Department of Motor Vehicles. 'Disengagement Reports'. (n.d.), [Online]. Available: https://www.dmv.ca.gov/portal/vehicle-industry-services/autonomous-vehicles/disengagement-reports/ (accessed Mar. 9, 2023).

[17] ASAM e. V. 'About ASAM'. (n.d.), [Online]. Available: https://www.asam.net/about-asam/ (accessed Feb. 14, 2023).

[18] ASAM e. V. 'ASAM OpenDRIVE®'. (n.d.), [Online]. Available: https://www.asam.net/standards/detail/opendrive/ (accessed Feb. 14, 2023).

[19] ASAM e. V. 'ASAM OpenSCENARIO®, V1.2.0'. (n.d.), [Online]. Available: https://www.asam.net/standards/detail/openscenario/ (accessed Feb. 17, 2023).

[20] ASAM e. V., *ASAM OpenDRIVE, Version: 1.7.0, 2021-08-03*, 3rd Aug. 2021. [Online]. Available: https://www.asam.net/index.php?eID=dumpFile&t=f&f=4422&token=e590561f3c39aa2260e5442e29e93f6693d1cccd.

[21] ASAM e. V. 'ASAM OpenSCENARIO®, V2.0.0'. (n.d.), [Online]. Available: https://www.asam.net/standards/detail/openscenario/v200/ (accessed Feb. 17, 2023).

[22] ASAM e. V., *ASAM OpenSCENARIO: User Guide, Version 1.2.0, 2022-05-13*, 13th May 2022. [Online]. Available: https://www.asam.net/index.php?eID=dumpFile&t=f&f=4908&token=ae9d9b44ab9257e817072a653b5d5e98ee0babf8.

[23] esmini. 'esmini'. (n.d.), [Online]. Available: https://github.com/esmini/esmini (accessed May 2, 2023).

[24] esmini. 'osc_coverage.txt'. (n.d.), [Online]. Available: https://github.com/esmini/esmini/blob/master/osc_coverage.txt (accessed May 2, 2023).

[25] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez and V. Koltun, 'CARLA: An open urban driving simulator', in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, pp. 1–16.

[26] CARLA. 'Scenariorunner'. (n.d.), [Online]. Available: https://github.com/carla-simulator/scenario_runner (accessed May 2, 2023).

[27] CARLA. 'OpenSCENARIO Support'. (n.d.), [Online]. Available: https://carla-scenariorunner.readthedocs.io/en/latest/openscenario_support/ (accessed May 2, 2023).

[28] S. Pagel. 'Opendrive viewer'. (n.d.), [Online]. Available: https://odrviewer.io (accessed May 2, 2023).

[29] G. Jahangirova, A. Stocco and P. Tonella, 'Quality metrics and oracles for autonomous vehicles testing', in *2021 14th IEEE Conference on Software Testing, Verification and Validation (ICST)*, IEEE, Apr. 2021. DOI: 10.1109/icst49551.2021.00030.

[30] M. Wen, J. Park and K. Cho, 'A scenario generation pipeline for autonomous vehicle simulators', *Human-centric Computing and Information Sciences*, vol. 10, no. 1, Jun. 2020. DOI: 10.1186/s13673-020-00231-z.

[31] R. Johansson, D. Williams, A. Berglund and P. Nugues, 'Carsim: A system to visualize written road accident reports as animated 3d scenes.' *2nd Workshop on Text Meaning and Interpretation,Barcelona, Spain,– Proceedings of the 2nd Workshop on Text Meaning and Interpretation*, pp. 57–64, 2004. [Online]. Available: https://ludwig.lub.lu.se/login?url=https://search.ebscohost.com/login.aspx?direct=true&AuthType=ip,uid&db=edsswe&AN=edsswe.oai.lup.lub.lu.se.e15cf8c7.416a.4783.8f0a.80e3b17ed8de&site=eds-live&scope=site.

[32] P. Johannesson and E. Perjons, *Introduction to Design Science*. Springer International Publishing AG, 2021, p. 185, ISBN: 9783030781316.

[33] SQLite. 'About sqlite'. (n.d.), [Online]. Available: https://sqlite.org/about.html (accessed Apr. 18, 2023).

[34] DB Browser for SQLite. 'Db browser for sqlite'. (n.d.), [Online]. Available: https://sqlitebrowser.org/ (accessed Apr. 18, 2023).

[35] Adobe. 'Convert pdf to excel'. (n.d.), [Online]. Available: https://www.adobe.com/acrobat/online/pdf-to-excel.html (accessed Apr. 19, 2023).

[36] V. V. Bochkarev, A. V. Shevlyakova and V. D. Solovyev, 'Average word length dynamics as indicator of cultural changes in society', *Social Evolution & History. Volume 14, number 2, p. 153-175 (2015)*, 30th Aug. 2012. DOI: 10.48550/ARXIV.1208.6109. arXiv: 1208.6109 [cs.CL].

[37] spaCy. 'Industrial-Strength Natural Language Processing, IN PYTHON'. (n.d.), [Online]. Available: https://spacy.io/ (accessed May 9, 2023).

[38] B. DeWilde. 'textacy: NLP, before and after spaCy'. (2023), [Online]. Available: https://textacy.readthedocs.io/en/latest/ (accessed May 9, 2023).

[39] pyoscx. 'scenariogeneration'. (2023), [Online]. Available: https://github.com/pyoscx/scenariogeneration (accessed May 9, 2023).

Appendices

Appendix A
Overview of Disengagement Reports

A.1 Data usefulness criteria

The primary focus is if the description contains data about the scenario itself such as what the AV was doing, what the conditions were like, what other vehicles there were and what they were doing.

A.2 Analysis

A.2.1 2015 folder

Lack of data/No disengagements reported

Bosch, Tesla

Not useful data (few word descriptions and/or mainly irrelevant)

Google (Waymo), Mercedes-Benz, Volkswagen

Mostly very basic but sometimes usable

Delphi (Aptiv), Nissan

A.2.2 2016 folder

Lack of data/No disengagements reported

Bosch, Honda, Mercedes-Benz, Volkswagen

Not useful data (few word descriptions and/or mainly irrelevant)

GM Cruise, Google (Waymo), Tesla

Mostly very basic but sometimes usable

BMW (one disengagement), Delphi (Aptiv), Ford (scanned document), Nissan (scanned document)

A.2.3 2017 folder

Lack of data/No disengagements reported

BMW, Bosch, Faraday And Future, Ford, Honda, Mercedes-Benz, Nio, Nvidia, Tesla, Valeo, Volkswagen, Wheego

Not useful data (few word descriptions and/or mainly irrelevant)

Delphi (Aptiv), GM Cruise, Waymo, Zoox

Mostly very basic but sometimes usable

Baidu (in the supplemental material, scanned document), Drive.ai, Nissan (scanned document), Telenav Inc (scanned document)

A.2.4 2018 folder

Lack of data/No disengagements reported

Bosch, Changan, Continental, Cyngn, Delphi (Aptiv), Faraday and Future Inc, Ford, Lyft, Navya, Nio, Samsung, Subaru, Tesla, Udacity Inc, Valeo, Volkswagen, Voyage

Not useful data (few word descriptions and/or mainly irrelevant)

AImotive, Apple (a large number of entries, max three-word descriptions), Aurora Innovation, AutoX, CarOne (Udelv), GM Cruise, Honda, Mercedes-Benz, Nullmax, Nvidia, Phantom AI Inc, PlusAi, Qualcomm, Renovo, Roadstar.ai, SAIC, Toyota, TuSimple, UATC LLC - UBER (a large number of entries, max five-word descriptions), Waymo, WeRide

Mostly very basic but sometimes usable

aiPod, Baidu, BMW (scanned document), drive.ai (scanned document), Nissan (scanned document), Nuro (scanned document), Pony.ai (scanned), SF Motors (scanned document), Telenav Inc (scanned document), Zoox

A.2.5 2019–2021 folder

All have been parsed to the DB already. About 1300 unique descriptions of varying quality. Have been divided into tables per year of disengagement occurrence meaning first-time filers have been added to the correct year even though it was in the following year's folder etc. 333 lines parsed to the 2018 table in DB from first-time filers in 2019 data, mostly repetitive and not very useful data but seems to not be overlapping with data available in 2018-folder.

Appendix B
Main Iteration Evaluations

The complete results from the main iterations A and B are presented in Table B.1 and B.2 respectively.

Table B.1: All disengagements report descriptions with a grade of 3 and the result or (main) issues when trying to create a scenario from them using the first main iteration of the application.

Id	Description	Result or (Main) Issues
1	Accelerator pedal pressed. False positive yielding to pedestrian near crosswalk during left turn causing braking.	Fails to find ego and any manoeuvre.
2	AV at a busy stop sign intersection. AV starts to move after the first cross vehicle turns in, however it comes to a stop when another vehicle approaches the cross stop sign. The second vehicle also turns in ahead of AV, as the AV response is slow and conservative. Safety driver intervenes once the 2nd vehicle took AV's right of way.	Co-reference resolution failure.
3	AV at a junction where AV has the right of way to move forward. Oncoming car cutting across does not come to a complete stop. Prediction module appears to cause AV to stop. Safety driver does a preventive intervention to continue flow of traffic.	Correct scenario (one of the development descriptions).
4	AV has just made a right turn and there is an oncoming vehicle in the opposite lane, resulting in a narrow path between parked car(on the right) and on coming vehicle(on the left) for the AV to follow. Safety driver initiated the take over, when he noticed the steering move to the left. Cause: Wrong Prediction of on-coming vehicle trajectory for a small time instant causing the vehicle to move in the wrong direction for that time instant.	Co-reference resolution failure.
5	AV needs to make a left turn in narrow street with parked vehicles on right side after the turn. The planner is not able to create a consistent path around parked vehicles in to the turn. It keeps switching between finding a path and requesting AV to come to complete stop, at edge of decision boundary, resulting in AV to start slowing down. Safety Driver takes over to maintain a smooth traffic flow.	Correct scenario (one of the development descriptions).
6	AV not able to plan a path through a narrow street with parked cars on either side, causing it to come to a stop. Driver takes over. Cause: Localization error in position causing issues in motion planning module.	Correct scenario (one of the development descriptions).

Id	Description	Result or (Main) Issues
7	Box truck parked on right side of AV. Slight bend on the road of traversal created a limited view of oncoming lane to the AV. AV decides to nudge left, but gets too close to the on coming vehicle from the opposite side. Safety driver initiates a take over to make get through the narrow zone between Box truck and the on coming vehicle.	Correct scenario (one of the development descriptions).
8	Car stopped in middle of intersection and did not proceed after pedestrian was finished crossing crosswalk	Correct scenario (one of the development descriptions).
9	Control Issue: Test Vehicle was going at speed limit but slower than traffic behind. Test Driver took over as a precaution and sped up to avoid causing unnecessary traffic.	Straight road default scenario as traffic is not added.
10	Driver makes a preventive intervention, when he sees an on-coming vehicle on a narrow street with parked cars on either sides leaving narrow gap for both vehicles to pass each other.	Straight road default scenario, does not detect on-coming vehicle and does not place parked cars.
11	During an exit/merge the test vehicle was going the "correct" speed as posted by road signs, but was going too slow or too fast given the traffic and road conditions. Conditions: Non-inclement weather, dry roads, no other factors involved	Straight road default scenario due to no exit/merge road network, no speed limit logic and failure to detect and create traffic.
12	During the lane change maneuver the test vehicle was going the "correct" speed as posted by road signs, but was going too slow or too fast given the traffic and road conditions. Conditions: Non-inclement weather, dry roads, no other factors involved	Straight road default scenario due to no speed limit logic and failure to detect and create traffic.
13	Narrow path in front of the AV with parked cars on both sides. Another vehicle coming in the on coming lane. AV yields to the on coming vehicle and starts coming to a stop. Safety driver initiates take over to make sure the encounter is safe.	Detects the AV and oncoming vehicle and places them in a straight road scenario but fails to parse parked cars from the sentence.
14	Narrow street with parked vehicles on either side. Gap on either side is less than AV Planner's safety thresholds, resulting in planner stopping the vehicle. Driver takes over.	Unable to produce an executable scenario due to failure to identify manoeuvres for ego vehicle.
15	Narrow street with parked vehicles on one side and on-coming vehicle on the other side. Gap on either side is less than AV Planner's safety thresholds, resulting in planner stopping the vehicle. Driver takes over to maneuver the AV through the narrow gap.	Unable to produce an executable scenario due to failure to identify manoeuvres for ego vehicle.
16	Narrow street with two large parked vehicles (mail van and pickup truck) on either side. Gap on either side is less than AV Planner's safety thresholds, resulting in planner stopping the vehicle. Driver takes over.	Unable to produce an executable scenario due to failure to identify manoeuvres for ego vehicle.

Id	Description	Result or (Main) Issues
17	Parked vehicle at the corner blocking path of AV while taking a right turn. AV perception appears to prevent vehicle from moving forward. Safety driver took over to continue the flow of traffic.	Fails to detect correct location causing default straight road and parser not able to identify the parked vehicle.
18	Parker cars on both sides of a narrow street, with another vehicle approaching from oncoming lane. AV yields to the oncoming vehicle and starts coming to a stop. Driver takes over to ensure encounter is safe.	Parser fails to detect the oncoming vehicle and parked cars resulting in default straight road scenario.
19	Planning Logic: while approaching right turn, planner inadequately yields for cyclist approaching from behind	Unable to create an executable scenario, fails to identify ego manoeuvre and cyclist.
20	Planning, driver precautionarily intervened for a reckless left vehicle makes a right turn from a go straight lane	Unable to create an executable scenario due to the application trying to generate a turning right action on an incorrectly selected straight road. Also fails to parse manoeuvres.
21	Planning; as a precaution, the safety operator intervened to avoid a vehicle that aggressively cut into the Autonomous Driving Vehicle's traveling lane from left to enter the parking lot on the right side of the road.	Unable to create an executable scenario due to missing ego manoeuvre and reckless vehicle manoeuvre.
22	Planning; as a precaution, the safety operator intervened to avoid a vehicle that did not follow the appropriate traffic procedure and made a left turn into Autonomous Driving Vehicle's traveling lane from a median crossover while Autonomous Driving Vehicle was going straight.	Unable to create an executable scenario due to the application trying to generate a turning left action on an incorrectly selected straight road. Also fails to parse other manoeuvres.
23	Planning; as a precaution, the safety operator intervened to avoid a vehicle that did not yield and made an unprotected left turn into Autonomous Driving Vehicle's traveling lane in intersection while Autonomous Driving Vehicle had the right of way.	Unable to create an executable scenario due to the application trying to create a manoeuvre representing a turn with the direction 'unprotected' as well as other missing manoeuvres.
24	Planning; as a precaution, the safety operator intervened to avoid a vehicle that illegally ran a red light from a blind spot occluded by a truck in Autonomous Driving Vehicle's neighbor lane.	Unable to create an executable scenario due to missing information after parsing.
25	Planning; as a precaution, the safety operator intervened to avoid the risk of being cut off by a vehicle that had suddenly pulled away from the side of the road.	Unable to create an executable scenario due to missing information after parsing.

Id	Description	Result or (Main) Issues
26	Planning; as a precaution, the safety operator intervened to make space between the ADV, which was going to make a left turn, and a vehicle traveling straight through the intersection at high-speed.	Unable to create an executable scenario due to failure to recognise the need for a junction as well as mislabelling entities.
27	Prediction discrepancy. False positive parked vehicle classification of a vehicle that was attempting to pull in front of us from a parking lane.	Unable to create an executable scenario due to mislabelling of entities and missing manoeuvres from the parser.
28	Prediction discrepancy. Incorrect trajectory estimation based on another occluded vehicle pulling out of a private driveway without stopping for Zoox's lane of traffic.	Unable to create an executable scenario due to missing manoeuvres and entities from the parser.
29	Safety driver brought AV to a faster stop because leading vehicle was stopping for a running pedestrian, leaving insufficient braking time.	Default straight road scenario with missing leading vehicle and pedestrian entities.
30	Safety driver brought AV to a full stop because a Pedestrian ran in front of AV while was crossing intersection on a Full Green traffic light.	Unable to create an executable scenario due to missing manoeuvres.
31	Safety Driver disengaged autonomous mode due to cyclist in crosswalk while test vehicle had green light.	Default straight road scenario without a crossing, traffic lights and a cyclist.
32	The AV was block by a car double park on the street, the AV as not able to go around due to incoming traffic. As a result, Driver safely disengaged and resumed manual control.	Default straight road scenario missing parked cars and incoming traffic due to the parser not correctly identifying them.
33	The testing car did not yield to the oncoming car while executing a right turn without traffic light. The driver took over.	Default straight road scenario due to missing information from the parser.
34	Weather(Sunny), Daylight: ODD - Car double parked on our lane - Test car not allowed to go over the double yellow lane marking.	Default straight road scenario missing parked cars.
35	Weather(Sunny), Daylight: ODD - Car on our lane going on reverse for parking - Collision risk with the Stopped AV - Test car not allowed to go over the double yellow lane marking.	Default straight road scenario missing the reversing car.
36	While the testing car was waiting for green light at an intersection, a long truck blocked both traffic lights at the same time when passing through. The testing car failed to detect any red light and launched forward. The driver took over.	Correct scenario (one of the development descriptions).

Table B.2: All disengagements report descriptions with a grade of 3 and the result or (main) issues when trying to create a scenario from them using the second main iteration of the application.

Id	Description	Result or (Main) Issues
1	Accelerator pedal pressed. False positive yielding to pedestrian near crosswalk during left turn causing braking.	Unable to generate a scenario, fails to find ego and any manoeuvre.
2	AV at a busy stop sign intersection. AV starts to move after the first cross vehicle turns in, however it comes to a stop when another vehicle approaches the cross stop sign. The second vehicle also turns in ahead of AV, as the AV response is slow and conservative. Safety driver intervenes once the 2nd vehicle took AV's right of way.	Co-reference resolution failure.
3	AV at a junction where AV has the right of way to move forward. Oncoming car cutting across does not come to a complete stop. Prediction module appears to cause AV to stop. Safety driver does a preventive intervention to continue flow of traffic.	Correct scenario (one of the development descriptions).
4	AV has just made a right turn and there is an oncoming vehicle in the opposite lane, resulting in a narrow path between parked car(on the right) and on coming vehicle(on the left) for the AV to follow. Safety driver initiated the take over, when he noticed the steering move to the left. Cause: Wrong Prediction of on-coming vehicle trajectory for a small time instant causing the vehicle to move in the wrong direction for that time instant.	Co-reference resolution failure.
5	AV needs to make a left turn in narrow street with parked vehicles on right side after the turn. The planner is not able to create a consistent path around parked vehicles in to the turn. It keeps switching between finding a path and requesting AV to come to complete stop, at edge of decision boundary, resulting in AV to start slowing down. Safety Driver takes over to maintain a smooth traffic flow.	Correct scenario (one of the development descriptions).
6	AV not able to plan a path through a narrow street with parked cars on either side, causing it to come to a stop. Driver takes over. Cause: Localization error in position causing issues in motion planning module.	Correct scenario (one of the development descriptions).

Id	Description	Result or (Main) Issues
7	Box truck parked on right side of AV. Slight bend on the road of traversal created a limited view of oncoming lane to the AV. AV decides to nudge left, but gets too close to the on coming vehicle from the opposite side. Safety driver initiates a take over to make get through the narrow zone between Box truck and the on coming vehicle.	Correct scenario (one of the development descriptions).
8	Car stopped in middle of intersection and did not proceed after pedestrian was finished crossing crosswalk	Correct scenario (one of the development descriptions).
9	Control Issue: Test Vehicle was going at speed limit but slower than traffic behind. Test Driver took over as a precaution and sped up to avoid causing unnecessary traffic.	Straight road default scenario as other traffic is not recognised and added.
10	Driver makes a preventive intervention, when he sees an on-coming vehicle on a narrow street with parked cars on either sides leaving narrow gap for both vehicles to pass each other.	Correct scenario (one of the development descriptions).
11	During an exit/merge the test vehicle was going the “correct” speed as posted by road signs, but was going too slow or too fast given the traffic and road conditions. Conditions: Non-inclement weather, dry roads, no other factors involved	Straight road default scenario as other traffic is not recognised and added and no support for exits/merges exists currently.
12	During the lane change maneuver the test vehicle was going the “correct” speed as posted by road signs, but was going too slow or too fast given the traffic and road conditions. Conditions: Non-inclement weather, dry roads, no other factors involved	Straight road default scenario as other traffic is not recognised and added.
13	Narrow path in front of the AV with parked cars on both sides. Another vehicle coming in the on coming lane. AV yields to the on coming vehicle and starts coming to a stop. Safety driver initiates take over to make sure the encounter is safe.	Correct scenario (one of the development descriptions).
14	Narrow street with parked vehicles on either side. Gap on either side is less than AV Planner’s safety thresholds, resulting in planner stopping the vehicle. Driver takes over.	Correct scenario (one of the development descriptions).
15	Narrow street with parked vehicles on one side and on-coming vehicle on the other side. Gap on either side is less than AV Planner’s safety thresholds, resulting in planner stopping the vehicle. Driver takes over to maneuver the AV through the narrow gap.	Correct scenario (one of the development descriptions).
16	Narrow street with two large parked vehicles (mail van and pickup truck) on either side. Gap on either side is less than AV Planner’s safety thresholds, resulting in planner stopping the vehicle. Driver takes over.	Correct scenario (one of the development descriptions).

Id	Description	Result or (Main) Issues
17	Parked vehicle at the corner blocking path of AV while taking a right turn. AV perception appears to prevent vehicle from moving forward. Safety driver took over to continue the flow of traffic.	Semi-correct scenario (one of the development descriptions), co-reference resolution issues cause an additional unwanted vehicle to be created.
18	Parker cars on both sides of a narrow street, with another vehicle approaching from oncoming lane. AV yields to the oncoming vehicle and starts coming to a stop. Driver takes over to ensure encounter is safe.	Co-reference resolution failure.
19	Planning Logic: while approaching right turn, planner inadequately yields for cyclist approaching from behind	Correct scenario (one of the development descriptions).
20	Planning, driver precautionarily intervened for a reckless left vehicle makes a right turn from a go straight lane	Correct scenario (one of the development descriptions).
21	Planning; as a precaution, the safety operator intervened to avoid a vehicle that aggressively cut into the Autonomous Driving Vehicle's traveling lane from left to enter the parking lot on the right side of the road.	Correct scenario (one of the development descriptions).
22	Planning; as a precaution, the safety operator intervened to avoid a vehicle that did not follow the appropriate traffic procedure and made a left turn into Autonomous Driving Vehicle's traveling lane from a median crossover while Autonomous Driving Vehicle was going straight.	Semi-correct scenario, AV and other traffic is created correctly but uses default junction as no median crossover road networks are available currently.
23	Planning; as a precaution, the safety operator intervened to avoid a vehicle that did not yield and made an unprotected left turn into Autonomous Driving Vehicle's traveling lane in intersection while Autonomous Driving Vehicle had the right of way.	Correct scenario (one of the development descriptions).
24	Planning; as a precaution, the safety operator intervened to avoid a vehicle that illegally ran a red light from a blind spot occluded by a truck in Autonomous Driving Vehicle's neighbor lane.	Straight road default scenario as other traffic and that it happened in a junction is not extracted by the parser.
25	Planning; as a precaution, the safety operator intervened to avoid the risk of being cut off by a vehicle that had suddenly pulled away from the side of the road.	Straight road default scenario as other traffic is not recognised and added.

Id	Description	Result or (Main) Issues
26	Planning; as a precaution, the safety operator intervened to make space between the ADV, which was going to make a left turn, and a vehicle traveling straight through the intersection at high-speed.	Scenario with only the ego vehicle as the other vehicle is missing, however, the ego vehicle and its manoeuvre is the result of failure to correctly extract information about the two vehicles and is a mix-up of both of them.
27	Prediction discrepancy. False positive parked vehicle classification of a vehicle that was attempting to pull in front of us from a parking lane.	Unable to generate a scenario as both vehicles and their manoeuvres are unidentified.
28	Prediction discrepancy. Incorrect trajectory estimation based on another occluded vehicle pulling out of a private driveway without stopping for Zoox's lane of traffic.	Unable to generate a scenario as both vehicles and their manoeuvres are unidentified.
29	Safety driver brought AV to a faster stop because leading vehicle was stopping for a running pedestrian, leaving insufficient braking time.	Straight road default scenario with duplicate hero vehicle and manoeuvre (why?). Fails to identify both the other vehicle and the pedestrian.
30	Safety driver brought AV to a full stop because a Pedestrian ran in front of AV while was crossing intersection on a Full Green traffic light.	Creates a scenario at a junction with duplicate hero vehicle and manoeuvre (why?), but it fails to identify the traffic lights. Identifies the pedestrian but fails to add it to the scenario.
31	Safety Driver disengaged autonomous mode due to cyclist in crosswalk while test vehicle had green light.	Standard straight road scenario as the application fails to identify both the cyclist in the crossing and the junction with traffic lights itself.
32	The AV was block by a car double park on the street, the AV as not able to go around due to incoming traffic. As a result, Driver safely disengaged and resumed manual control.	Straight road default scenario as the application fails to identify both parked and incoming traffic.
33	The testing car did not yield to the oncoming car while executing a right turn without traffic light. The driver took over.	Identifies and creates two vehicles, but places them incorrectly in relation to one another and fails to correctly map each one's manoeuvre to the right vehicle.
34	Weather(Sunny), Daylight: ODD - Car double parked on our lane - Test car not allowed to go over the double yellow lane marking.	Unable to generate a scenario as both vehicles and their manoeuvres are unidentified.

Id	Description	Result or (Main) Issues
35	Weather(Sunny), Daylight: ODD - Car on our lane going on reverse for parking - Collision risk with the Stopped AV - Test car not allowed to go over the double yellow lane marking.	Straight road default scenario as the application fails to identify the other vehicle.
36	While the testing car was waiting for green light at an intersection, a long truck blocked both traffic lights at the same time when passing through. The testing car failed to detect any red light and launched forward. The driver took over.	Correct scenario (one of the development descriptions).

Appendix C

Example Scenario 1


```xml
<?xml version="1.0" encoding="UTF-8"?>
<OpenSCENARIO xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="OpenScenario.xsd">
  <FileHeader description="While the testing car was waiting for green light at an intersection, a long truck blocked both traffic lights at the same time when passing through. The testing car failed to detect any red light and launched forward. The driver took over." author="Henrik Olsson &amp; Rune Anderberg" revMajor="1" revMinor="0" date="2023-05-12T14:29:49.890332" />
  <ParameterDeclarations />
  <CatalogLocations />
  <RoadNetwork>
    <LogicFile filepath="roadnetworks/signal_junction_4.xodr" />
  </RoadNetwork>
  <Entities>
    <ScenarioObject name="hero">
      <Vehicle name="vehicle.nissan.micra" vehicleCategory="car">
        <ParameterDeclarations />
        <BoundingBox>
          <Center x="1.5" y="0.0" z="0.9" />
          <Dimensions width="2.1" length="4.5" height="1.8" />
        </BoundingBox>
        <Performance maxSpeed="69.444" maxDeceleration="10.0" maxAcceleration="200.0" />
        <Axles>
          <FrontAxle maxSteering="0.5" wheelDiameter="0.6" trackWidth="1.8" positionX="3.1" positionZ="0.3" />
          <RearAxle maxSteering="0.0" wheelDiameter="0.6" trackWidth="1.8" positionX="0.0" positionZ="0.3" />
        </Axles>
        <Properties>
          <Property name="type" value="ego_vehicle" />
        </Properties>
```

```xml
      </Vehicle>
    </ScenarioObject>
    <ScenarioObject name="Entity_2">
      <Vehicle name="vehicle.carlamotors.firetruck" vehicleCategory="truck">
        <ParameterDeclarations />
        <BoundingBox>
          <Center x="1.5" y="0.0" z="0.9" />
          <Dimensions width="3.6" length="9.5" height="8.8" />
        </BoundingBox>
        <Performance maxSpeed="69.444" maxDeceleration="10.0" maxAcceleration="200.0" />
        <Axles>
          <FrontAxle maxSteering="0.5" wheelDiameter="0.6" trackWidth="1.8" positionX="3.1" positionZ="0.3" />
          <RearAxle maxSteering="0.0" wheelDiameter="0.6" trackWidth="1.8" positionX="0.0" positionZ="0.3" />
        </Axles>
        <Properties />
      </Vehicle>
    </ScenarioObject>
  </Entities>
  <Storyboard>
    <Init>
      <Actions>
        <GlobalAction>
          <EnvironmentAction>
            <Environment name="SetTimeOfDay">
              <TimeOfDay animation="true" dateTime="2020-01-01T12:00:00" />
              <Weather cloudState="free">
                <Sun azimuth="0" intensity="0.85" elevation="1.31" />
                <Fog visualRange="100000.0" />
                <Precipitation precipitationType="dry" intensity="0.9" />
              </Weather>
              <RoadCondition frictionScaleFactor="1.0" />
            </Environment>
          </EnvironmentAction>
        </GlobalAction>
        <Private entityRef="hero">
          <PrivateAction>
            <TeleportAction>
              <Position>
                <LanePosition roadId="1" laneId="1" s="10.0" offset="0.0" />
              </Position>
            </TeleportAction>
          </PrivateAction>
          <PrivateAction>
            <ControllerAction>
              <OverrideControllerValueAction>
                <Throttle active="false" value="0.0" />
                <Brake active="false" value="0.0" />
                <Clutch active="false" value="0.0" />
                <ParkingBrake active="false" value="0.0" />
```

```
                <SteeringWheel active="false" value="0.0" />
                <Gear active="false" number="0.0" />
              </OverrideControllerValueAction>
              <AssignControllerAction>
                <Controller name="HeroAgent">
                  <ParameterDeclarations />
                  <Properties>
                    <Property name="module" value="simple_vehicle_control" />
                  </Properties>
                </Controller>
              </AssignControllerAction>
            </ControllerAction>
          </PrivateAction>
        </Private>
        <Private entityRef="Entity_2">
          <PrivateAction>
            <ControllerAction>
              <OverrideControllerValueAction>
                <Throttle active="false" value="0.0" />
                <Brake active="false" value="0.0" />
                <Clutch active="false" value="0.0" />
                <ParkingBrake active="false" value="0.0" />
                <SteeringWheel active="false" value="0.0" />
                <Gear active="false" number="0.0" />
              </OverrideControllerValueAction>
              <AssignControllerAction>
                <Controller name="Entity_2Agent">
                  <ParameterDeclarations />
                  <Properties>
                    <Property name="module" value="simple_vehicle_control" />
                  </Properties>
                </Controller>
              </AssignControllerAction>
            </ControllerAction>
          </PrivateAction>
          <PrivateAction>
            <TeleportAction>
              <Position>
                <LanePosition roadId="0" laneId="-1" s="90.0" offset="0.0" />
              </Position>
            </TeleportAction>
          </PrivateAction>
        </Private>
      </Actions>
    </Init>
    <Story name="Main Story">
      <ParameterDeclarations />
      <Act name="First Act">
        <ManeuverGroup name="ManueverGroup0" maximumExecutionCount="1">
          <Actors selectTriggeringEntities="true">
            <EntityRef entityRef="hero" />
          </Actors>
          <Maneuver name="ManeuverGroup0_Manuever">
```

```
            <Event name="Test Event" priority="overwrite" maximumExecutionCount="1">
              <Action name="Drive Test">
                <PrivateAction>
                  <RoutingAction>
                    <AcquirePositionAction>
                      <Position>
                        <LanePosition roadId="3" laneId="-1" s="50.0" offset="0.0" />
                      </Position>
                    </AcquirePositionAction>
                  </RoutingAction>
                </PrivateAction>
              </Action>
              <Action name="Speed">
                <PrivateAction>
                  <LongitudinalAction>
                    <SpeedAction>
                      <SpeedActionDynamics dynamicsShape="step" value="2.0" dynamicsDimension="distance" />
                      <SpeedActionTarget>
                        <AbsoluteTargetSpeed value="8.0" />
                      </SpeedActionTarget>
                    </SpeedAction>
                  </LongitudinalAction>
                </PrivateAction>
              </Action>
              <StartTrigger>
                <ConditionGroup>
                  <Condition name="Test" delay="0.0" conditionEdge="none">
                    <ByValueCondition>
                      <TrafficSignalCondition name="pos=112,-10" state="green" />
                    </ByValueCondition>
                  </Condition>
                </ConditionGroup>
              </StartTrigger>
            </Event>
          </Maneuver>
        </ManeuverGroup>
        <ManeuverGroup name="ManueverGroup1" maximumExecutionCount="1">
          <Actors selectTriggeringEntities="true">
            <EntityRef entityRef="Entity_2" />
          </Actors>
          <Maneuver name="ManeuverGroup1_Manuever">
            <Event name="Test Event" priority="overwrite" maximumExecutionCount="1">
              <Action name="Drive Test">
                <PrivateAction>
                  <RoutingAction>
                    <AcquirePositionAction>
                      <Position>
                        <LanePosition roadId="2" laneId="-1" s="50.0" offset="0.0" />
                      </Position>
```

```
                    </AcquirePositionAction>
                  </RoutingAction>
                </PrivateAction>
              </Action>
              <Action name="Speed">
                <PrivateAction>
                  <LongitudinalAction>
                    <SpeedAction>
                      <SpeedActionDynamics dynamicsShape="step" value="2.0" dynamicsDimension="distance" />
                      <SpeedActionTarget>
                        <AbsoluteTargetSpeed value="8.0" />
                      </SpeedActionTarget>
                    </SpeedAction>
                  </LongitudinalAction>
                </PrivateAction>
              </Action>
              <StartTrigger>
                <ConditionGroup>
                  <Condition name="Test" delay="0.0" conditionEdge="none">
                    <ByValueCondition>
                      <TrafficSignalCondition name="pos=118,4" state="green" />
                    </ByValueCondition>
                  </Condition>
                </ConditionGroup>
              </StartTrigger>
            </Event>
          </Maneuver>
        </ManeuverGroup>
        <StartTrigger>
          <ConditionGroup>
            <Condition name="act_start" delay="0.0" conditionEdge="none">
              <ByValueCondition>
                <SimulationTimeCondition value="0.0" rule="greaterThan" />
              </ByValueCondition>
            </Condition>
          </ConditionGroup>
        </StartTrigger>
        <StopTrigger />
      </Act>
    </Story>
    <StopTrigger />
  </Storyboard>
</OpenSCENARIO>
```

Appendix D

Example Scenario 2

```xml
<?xml version="1.0" encoding="UTF-8"?>
<OpenSCENARIO xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="OpenScenario.xsd">
  <FileHeader description="Autonomous vehicle at a junction where Autonomous vehicle has the right of way to move forward. Oncoming car cutting across does not come to a complete stop. Prediction module appears to cause Autonomous vehicle to stop. Safety driver does a preventive intervention to continue flow of traffic." author="Henrik Olsson &amp; Rune Anderberg" revMajor="1" revMinor="0" date="2023-05-11T10:00:00" />
  <ParameterDeclarations />
  <CatalogLocations />
  <RoadNetwork>
    <LogicFile filepath="roadnetworks/junction_4.xodr" />
  </RoadNetwork>
  <Entities>
    <ScenarioObject name="hero">
      <Vehicle name="vehicle.nissan.micra" vehicleCategory="car">
        <ParameterDeclarations />
        <BoundingBox>
          <Center x="1.5" y="0.0" z="0.9" />
          <Dimensions width="2.1" length="4.5" height="1.8" />
        </BoundingBox>
        <Performance maxSpeed="69.444" maxDeceleration="10.0" maxAcceleration="200.0" />
        <Axles>
          <FrontAxle maxSteering="0.5" wheelDiameter="0.6" trackWidth="1.8" positionX="3.1" positionZ="0.3" />
          <RearAxle maxSteering="0.0" wheelDiameter="0.6" trackWidth="1.8" positionX="0.0" positionZ="0.3" />
        </Axles>
        <Properties>
          <Property name="type" value="ego_vehicle" />
```

```xml
        </Properties>
      </Vehicle>
    </ScenarioObject>
    <ScenarioObject name="Entity_4">
      <Vehicle name="vehicle.nissan.micra" vehicleCategory="car">
        <ParameterDeclarations />
        <BoundingBox>
          <Center x="1.5" y="0.0" z="0.9" />
          <Dimensions width="2.1" length="4.5" height="1.8" />
        </BoundingBox>
        <Performance maxSpeed="69.444" maxDeceleration="10.0" maxAcceleration="200.0" />
        <Axles>
          <FrontAxle maxSteering="0.5" wheelDiameter="0.6" trackWidth="1.8" positionX="3.1" positionZ="0.3" />
          <RearAxle maxSteering="0.0" wheelDiameter="0.6" trackWidth="1.8" positionX="0.0" positionZ="0.3" />
        </Axles>
        <Properties />
      </Vehicle>
    </ScenarioObject>
  </Entities>
  <Storyboard>
    <Init>
      <Actions>
        <GlobalAction>
          <EnvironmentAction>
            <Environment name="SetTimeOfDay">
              <TimeOfDay animation="true" dateTime="2020-01-01T12:00:00" />
              <Weather cloudState="free">
                <Sun azimuth="0" intensity="0.85" elevation="1.31" />
                <Fog visualRange="100000.0" />
                <Precipitation precipitationType="dry" intensity="0.9" />
              </Weather>
              <RoadCondition frictionScaleFactor="1.0" />
            </Environment>
          </EnvironmentAction>
        </GlobalAction>
        <Private entityRef="hero">
          <PrivateAction>
            <TeleportAction>
              <Position>
                <LanePosition roadId="1" laneId="1" s="10.0" offset="0.0" />
              </Position>
            </TeleportAction>
          </PrivateAction>
          <PrivateAction>
            <ControllerAction>
              <OverrideControllerValueAction>
                <Throttle active="false" value="0.0" />
                <Brake active="false" value="0.0" />
                <Clutch active="false" value="0.0" />
                <ParkingBrake active="false" value="0.0" />
```

```
            <SteeringWheel active="false" value="0.0" />
            <Gear active="false" number="0.0" />
          </OverrideControllerValueAction>
          <AssignControllerAction>
            <Controller name="HeroAgent">
              <ParameterDeclarations />
              <Properties>
                <Property name="module" value="simple_vehicle_control" />
              </Properties>
            </Controller>
          </AssignControllerAction>
        </ControllerAction>
      </PrivateAction>
    </Private>
    <Private entityRef="Entity_4">
      <PrivateAction>
        <ControllerAction>
          <OverrideControllerValueAction>
            <Throttle active="false" value="0.0" />
            <Brake active="false" value="0.0" />
            <Clutch active="false" value="0.0" />
            <ParkingBrake active="false" value="0.0" />
            <SteeringWheel active="false" value="0.0" />
            <Gear active="false" number="0.0" />
          </OverrideControllerValueAction>
          <AssignControllerAction>
            <Controller name="Entity_4Agent">
              <ParameterDeclarations />
              <Properties>
                <Property name="module" value="simple_vehicle_control" />
              </Properties>
            </Controller>
          </AssignControllerAction>
        </ControllerAction>
      </PrivateAction>
      <PrivateAction>
        <TeleportAction>
          <Position>
            <LanePosition roadId="3" laneId="1" s="10.0" offset="0.0" />
          </Position>
        </TeleportAction>
      </PrivateAction>
    </Private>
  </Actions>
</Init>
<Story name="Main Story">
  <ParameterDeclarations />
  <Act name="First Act">
    <ManeuverGroup name="ManueverGroup1" maximumExecutionCount="1">
      <Actors selectTriggeringEntities="true">
        <EntityRef entityRef="hero" />
      </Actors>
      <Maneuver name="ManeuverGroup1_Manuever">
```

```
        <Event name="Test Event" priority="overwrite" maximumExecutionCount="1">
          <Action name="Drive Test">
            <PrivateAction>
              <RoutingAction>
                <AcquirePositionAction>
                  <Position>
                    <LanePosition roadId="3" laneId="-1" s="50.0" offset="0.0" />
                  </Position>
                </AcquirePositionAction>
              </RoutingAction>
            </PrivateAction>
          </Action>
          <Action name="Speed">
            <PrivateAction>
              <LongitudinalAction>
                <SpeedAction>
                  <SpeedActionDynamics dynamicsShape="step" value="2.0" dynamicsDimension="distance" />
                  <SpeedActionTarget>
                    <AbsoluteTargetSpeed value="8.0" />
                  </SpeedActionTarget>
                </SpeedAction>
              </LongitudinalAction>
            </PrivateAction>
          </Action>
          <StartTrigger>
            <ConditionGroup>
              <Condition name="InstantStart" delay="1.0" conditionEdge="none">
                <ByValueCondition>
                  <SimulationTimeCondition value="0.0" rule="greaterThan" />
                </ByValueCondition>
              </Condition>
            </ConditionGroup>
          </StartTrigger>
        </Event>
      </Maneuver>
    </ManeuverGroup>
    <ManeuverGroup name="ManueverGroup2" maximumExecutionCount="1">
      <Actors selectTriggeringEntities="true">
        <EntityRef entityRef="Entity_4" />
      </Actors>
      <Maneuver name="ManeuverGroup2_Manuever">
        <Event name="Test Event" priority="overwrite" maximumExecutionCount="1">
          <Action name="Drive Test">
            <PrivateAction>
              <RoutingAction>
                <AcquirePositionAction>
                  <Position>
                    <LanePosition roadId="2" laneId="-1" s="50.0" offset="0.0" />
                  </Position>
```

```
                  </AcquirePositionAction>
                </RoutingAction>
              </PrivateAction>
            </Action>
            <Action name="Speed">
              <PrivateAction>
                <LongitudinalAction>
                  <SpeedAction>
                    <SpeedActionDynamics dynamicsShape="step" value="2.0" dynamicsDimension="distance" />
                    <SpeedActionTarget>
                      <AbsoluteTargetSpeed value="8.0" />
                    </SpeedActionTarget>
                  </SpeedAction>
                </LongitudinalAction>
              </PrivateAction>
            </Action>
            <StartTrigger>
              <ConditionGroup>
                <Condition name="InstantStart" delay="1.0" conditionEdge="none">
                  <ByValueCondition>
                    <SimulationTimeCondition value="0.0" rule="greaterThan" />
                  </ByValueCondition>
                </Condition>
              </ConditionGroup>
            </StartTrigger>
          </Event>
        </Maneuver>
      </ManeuverGroup>
      <ManeuverGroup name="ManueverGroup3" maximumExecutionCount="1">
        <Actors selectTriggeringEntities="true">
          <EntityRef entityRef="Entity_4" />
        </Actors>
        <Maneuver name="ManeuverGroup3_Manuever">
          <Event name="Test Event" priority="overwrite" maximumExecutionCount="1">
            <Action name="SpeedSync">
              <PrivateAction>
                <SynchronizeAction masterEntityRef="hero">
                  <TargetPositionMaster>
                    <LanePosition roadId="104" laneId="-1" s="14.0" offset="0.0" />
                  </TargetPositionMaster>
                  <TargetPosition>
                    <LanePosition roadId="105" laneId="1" s="10.0" offset="-2.0" />
                  </TargetPosition>
                  <FinalSpeed>
                    <AbsoluteSpeed value="8.0" />
                  </FinalSpeed>
                </SynchronizeAction>
              </PrivateAction>
            </Action>
            <StartTrigger>
```

```
                <ConditionGroup>
                  <Condition name="InstantStart" delay="1.0" conditionEdge="none">
                    <ByValueCondition>
                      <SimulationTimeCondition value="0.0" rule="greaterThan" />
                    </ByValueCondition>
                  </Condition>
                </ConditionGroup>
              </StartTrigger>
            </Event>
          </Maneuver>
        </ManeuverGroup>
        <StartTrigger>
          <ConditionGroup>
            <Condition name="act_start" delay="0.0" conditionEdge="none">
              <ByValueCondition>
                <SimulationTimeCondition value="0.0" rule="greaterThan" />
              </ByValueCondition>
            </Condition>
          </ConditionGroup>
        </StartTrigger>
        <StopTrigger />
      </Act>
    </Story>
    <StopTrigger />
  </Storyboard>
</OpenSCENARIO>
```

Appendix E

Example OpenAI Scenario 1

```xml
<?xml version="1.0" encoding="UTF-8"?>
<OpenSCENARIO xmlns="http://www.openscenario.org/file/v1.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.openscenario.org/file/v1.0 OpenSCENARIO-v1.0.xsd" version="1.0">
  <Actors>
    <Actor id="AV">
      <Behavior>
        <Actuator Steering="1" Acceleration="1" Brake="1" Gear="1" />
      </Behavior>
    </Actor>
    <Actor id="SD" />
  </Actors>
  <Environment>
    <Time Date="2020-10-07T00:00:00Z">
      <PredefinedDateTime Date="2020-10-07T00:00:00Z" />
    </Time>
    <RoadNetwork Source="File" File="myRoadNetwork.xodr" />
    <RelativeLanePosition pos="-0.5" laneId="laneId" />
  </Environment>
  <Storyboard>
    <Act Actors="AV" />
    <PreCondition>
      <TriggerAction name="oncomingVehicle" />
    </PreCondition>
    <Action>
      <TrajectoryFollow AllowCorrections="true" />
      <SpeedByTime Curve="constant" />
      <Event ev="oncomingVehicle" type="other" subType="intersection_crossing">
        <RelativeLanePosition pos="-0.5" laneId="laneId" />
        <Controller actuator="Brake" value="1" />
      </Event>
```

```
    </Action>
    <Act Actors="SD" />
    <Action>
      <StandStill />
      <Event ev="preventIntervention" type="anticipated" subType="obstacle">
        <RelativeLanePosition pos="-0.5" laneId="laneId" />
        <Controller actuator="Brake" value="0" />
        <Controller actuator="Acceleration" value="1" />
      </Event>
    </Action>
  </Storyboard>
</OpenSCENARIO>
```

Appendix F

Example OpenAI Scenario 2

```xml
<?xml version="1.0" encoding="UTF-8"?>
<OpenSCENARIO xmlns="http://openscenario.org/file/v1.0" xmlns:xsi="http
    ://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://
    openscenario.org/file/v1.0 OpenSCENARIO-v1.0.xsd">
  <FileHeader>
    <Author>Example Author</Author>
    <Date>05/20/2020</Date>
    <Category />
    <Description>AV at a junction where AV has the right of way to move
     forward. Oncoming car cutting across does not come to a complete
    stop. Prediction module appears to cause AV to stop. Safety driver
    does a preventive intervention to continue flow of traffic.</
    Description>
  </FileHeader>
  <Storyboard>
    <Story>
      <Act>
        <Name>First Act</Name>
        <ScenarioObjects>
          <RoadNetwork />
          <TrafficDefinition />
          <ControllerDefinition />
        </ScenarioObjects>
      </Act>
    </Story>
  </Storyboard>
  <Entities>
    <ObjectController type="prediction" name="predictionModule" />
    <PedestrianController type="safetyDriver" name="safetyDriver" />
    <Vehicle name="AV" controller="predictionModule" catalogueName="
    vehicleCatalog" />
    <Vehicle name="oncomingCar" controller="predictionModule"
    catalogueName="vehicleCatalog" />
```

```xml
  </Entities>
  <ManeuverGroups>
    <ManeuverGroup name="proactiveAIManeuver">
      <Maneuver name="ProactiveAction">
        <Act>
          <Name>FirstAct</Name>
        </Act>
        <Entities>
          <ObjectController ref="predictionModule" />
          <PedestrianController ref="safetyDriver" />
          <Vehicle ref="AV" />
          <Vehicle ref="oncomingCar" />
        </Entities>
      </Maneuver>
    </ManeuverGroup>
  </ManeuverGroups>
</OpenSCENARIO>
```

Appendix G

Example OpenAI Scenario 3

```xml
<?xml version="1.0" encoding="UTF-8"?>
<OpenSCENARIO xmlns="http://openscenario.xsd" xmlns:xsi="http://www.w3.
    org/2001/XMLSchema-instance" version="1.0" xsi:schemaLocation="http
    ://openscenario.xsd ../xsd/OpenSCENARIO.xsd">
  <Storyboard>
    <Actors>
      <Actor entityRef="a0" />
      <Actor entityRef="a1" />
    </Actors>
    <Action id="a0" name="AV" type="private">
      <SpeedAction goal="25.0" speed="25.0" />
      <ManeuverAction>
        <LaneChangeAction goal="true" />
        <DirectionAction closed="true" leader_link="L1" />
        <LaneOffsetAction laneOffset="-2" />
        <LongitudinalRuleAction cruiseControl="true" />
        <SpeedAction goal="18.0" speed="18.0" />
      </ManeuverAction>
    </Action>
    <Action ref="a0" id="a1" name="Safety Driver" type="private">
      <SpeedAction goal="25.0" speed="25.0" />
      <ReplaceActionReference ref="a0" />
      <SpeedAction goal="18.0" speed="18.0" />
    </Action>
  </Storyboard>
</OpenSCENARIO>
```

EXAMENSARBETE Turning Disengagement Reports Into Executable Test Scenarios for Autonomous Vehicles Using NLP
STUDENTER Rune Anderberg, Henrik Olsson
HANDLEDARE Qunying Song (LTH)
EXAMINATOR Per Runeson (LTH)

Testscenarier för självkörande bilar med hjälp av naturlig språkbehandling

POPULÄRVETENSKAPLIG SAMMANFATTNING **Rune Anderberg, Henrik Olsson**

Det är väldigt viktigt med ordentlig testning av självkörande bilar för att kunna garantera deras säkerhet på vägen. Vårt arbete har undersökt ett sätt att skapa testscenarier för användning i simulator genom att använda naturlig språkbehandling för att plocka ut information från beskrivningar av problematiska situationer i verkligheten.

I vårt examensarbete har vi utvecklat en applikation som kan underlätta i arbetet att skapa testscenarier för självkörande bilar som testas i simulator. Det gör den genom att använda naturlig språkbehandling (eng. Natural Language Processing) på frånkopplingsrapporter. Frånkopplingsrapporter är beskrivningar av situationer där autopiloten i en självkörande bil var tvungen att kopplas ifrån, skrivna på naturligt språk. Applikationen plockar ut information från dem, för att sedan försöka återskapa situationen som ett körbart testscenario.

Naturlig språkbehandling är precis vad det låter som, behandling av naturligt språk, alltså det språk som vi människor använder oss av. Området är inget nytt, men på senare tid har det utvecklats mycket. De tekniker och tillvägagångssätt som används har ändras genom tiden, och nu är det väldigt vanligt att man använder sig av maskininlärning på ett eller annat sätt.

Utvecklingen av självkörande bilar är något annat som också har utvecklats mycket på senaste tiden, men för att kunna garantera deras säkerhet på vägen krävs noggrann testning och validering. Att testa dem direkt på offentliga vägar är värdefullt då det representerar den miljö där bilarna ska användas. Dock finns det vissa problem med det också. Det är ofta är dyrt att genomföra och svårt att kontrollera de förhållanden bilen befinner sig i. Ett sätt att minska kostnaderna och göra testningen mer kontrollerad är att genomföra den i simulator istället, programvara som simulerar den självkörande bilen, vägarna den befinner sig på och alla andra medtrafikanter.

När man testar i simulator är det vanligt att i förväg definiera ett scenario, till exempel att svänga vänster i en korsning med många andra bilar i, som man låter den självkörande bilen genomföra. Så att kunna generera realistiska testscenarier automatiskt är något som kan va till stor nytta. Möjligheten att ta ett scenario som visade sig vara problematiskt när en självkörande bil testades i verkligheten, för att sen testa ny programvara på samma scenario i simulator är hjälpsamt.

Att använda sig av frånkopplingsrapporter är något som kan leda till testscenarier som man aldrig hade kunna tänka sig. Tänk själv, om du hade jobbat med att försöka skapa nya testscenarier, hur lång tid hade det tagit innan du hade kommit på ett scenario skapat utifrån att ett barn åker sparkcykel i en cirkel på vägen? Det är ingen situation som är enkel att förutspå, men det är ändå väldigt viktigt att en självkörande bil har förmågan att hantera situationen på ett säkert sätt.