An exploration of methods for functional safety assessment of autonomous vehicles

Singh, T.S.

Award date:
2021

Technische Universiteit Eindhoven University of Technology
Department of Mathematics and Computer Science

SIEMENS

An exploration of methods for functional safety assessment of autonomous vehicles

Master Thesis
MSc Automotive Technology

Tajinder Singh - 1436295

Tutor: Sangeeth Kochanthara MSc (TU/e)

Supervisors:

dr.ir. Loek G.W.A. Cleophas (TU/e)
dr. Alexandru Forrai (Siemens Digital Industries Software Netherlands)

Eindhoven, 12th Oct, 2021

Declaration concerning the TU/e Code of Scientific Conduct for the Master's thesis

I have read the TU/e Code of Scientific Conduct[i].

I hereby declare that my Master's thesis has been carried out in accordance with the rules of the TU/e Code of Scientific Conduct

Date

5th October 2021

Name

Tajinder Singh

ID-number

1436295

Signature

Submit the signed declaration to the student administration of your department.

[i] See: https://www.tue.nl/en/our-university/about-the-university/organization/integrity/scientific-integrity/
The Netherlands Code of Conduct for Scientific Integrity, endorsed by 6 umbrella organizations, including the VSNU, can be found here also. More information about scientific integrity is published on the websites of TU/e and VSNU

Abstract

Autonomous vehicles have entered the phase of public road testing in recent years. Assessing and ensuring the functional safety of autonomous vehicles is key for their safe operation and for societal acceptance of the technology. Two challenges to autonomous vehicle functional safety are: (i) complexity and diversity of real-world driving conditions and (ii) safety concerns related to the use of artificial intelligence (AI) based components. This thesis investigates methods to assess the functional safety of an autonomous vehicle for the real-world driving conditions of a Dutch highway setting, with a focus on the vehicle's AI-based sensing and perception subsystem.

We assess the functional safety of an industrial-grade autonomous vehicle software stack, Baidu Apollo, for the context of operating conditions in the A270 Dutch highway. The functional safety assessment is performed against functional safety requirements regarding the vehicle's AI-based sensing and perception subsystem. The functional safety requirements are elicited according to the automotive safety standards ISO 26262 and DIS/ISO 21448. We perform the functional safety assessment against the functional safety requirements at (i) design level by checking for safety-related design practices and (ii) implementation level by simulation-based testing.

The performed functional safety assessment reveals potential gaps in the functional safety of Apollo. These gaps point to the need for additional safety mechanisms related to the AI-based sensing and perception subsystem in the context of operating conditions in the A270 Dutch highway.

Acknowledgements

First of all, I would like to thank Loek, Alexandru, and Sangeeth for guiding me throughout this journey. It has been a pleasure to work with the three of you. Your words of encouragement, friendliness, insights, and enthusiasm inspired me to do my best. Loek, thank you for your quick email responses and your thoughtfulness in making sure I had the help I needed. Alexandru, thank you for taking extra time out to support me and for always checking in with me to see how I am doing. Sangeeth, thank you for the many extra discussions we had, for your efforts in making sure I was on track, and for being available for a chat whenever I needed it.

I would also like to thank Siemens DISW Helmond for providing me with this opportunity, and the Amandus H. Lundqvist scholarship program and the Holland scholarship program for providing financial support for my master studies at TU/e.

I am grateful for many other things at TU/e - the beautiful campus, the community, support facilities for students, organizations such as tint and studium generale, and the opportunities we were given to make special memories (like the ice rink outside Metaform for Christmas 2019).

This thesis has been a challenging journey and I am proud of the resilience I showed and the hard work I have put into this project. I had to learn along the way (and I am still learning) that good work comes out of taking care of my well-being first, by setting realistic goals, not being afraid to make mistakes, and asking for help.

I would like to dedicate this thesis to my loving parents, sister, and extended family. A special thanks also goes to my friends for walks, video calls, dinners, laughter, encouragement, and much more during this journey.

Contents

Chapter 1

Introduction

Autonomous vehicles (AVs), also referred to as self-driving cars, promise many societal benefits including reduced road fatalities by elimination of human error [1], and lower energy consumption with initiatives such as truck platooning [2]. Technologies for AV have undergone extensive research and development in recent years. In March 2021, the first partially autonomous vehicle was certified for commercial use on highways [3]. Fully autonomous vehicles have also been certified for public road testing in several countries, including the Netherlands [1] and the United States [4].

A major challenge facing the deployment of AVs on public roads is demonstrating the safety of the technology. Compared to manually driven vehicles, AVs face new safety challenges due to human-automation interaction, complex driving conditions, and the lack of human intervention or supervision. Assessing and ensuring safety is important both for public acceptance [5] and for regulatory bodies. For example, the German ethics commission states that AVs should demonstrate a "positive risk balance compared to human driving" [6].

This thesis addresses a specific area of AV safety known as *functional safety*; the absence of unreasonable risk due to failures of AV components. The scope of failures related to functional safety are outlined in the two automotive safety standards: ISO 26262 and the *safety of the intended functionality* (SOTIF). The ISO 26262 standard, first published in 2011 [7], and revised in 2018 [8], tackles automotive safety against malfunctioning behavior, or faults, in the vehicle's software and hardware components. Although ISO 26262 was designed with manually driven vehicles in mind, it is widely used in AV development [5], [9].

The SOTIF standard covers AV safety against functional insufficiencies of the technology and issues related to human-automation interaction. The *draft international standard* (ISO/DIS 21448) [10] version of the standard was released in 2020. Thus, based on these two standards, we may further describe functional safety as the absence of unreasonable risk due to malfunctioning behavior (faults) or functional insufficiencies in the AV components.

Although the two safety standards provide guidance for AV functional safety, the use of artificial intelligence (AI) based components in AVs remains a challenge for functional safety [6]. AI technology is crucial for AVs [6] and is state-of-practice and state-of-the-art for the 3 dimensional object detection functionality of the AV [11]. AI-based components are developed using datasets instead of formal specifications, leading to safety concerns such as difficulty of risk assessment [12], unpredictable behavior in novel or rare driving conditions [13, 14] and low robustness to input distribution shifts [15].

This project investigates methods to assess the functional safety of an AV for a Dutch highway setting, with a focus on the AI-based sensing and perception subsystem of the AV. The rest of the chapter motivates and outlines the graduation project. Section 1.1 describes the motivation

of the research. Section 1.2 presents the research questions and the scope of the project. Finally, Section 1.3 describes the structure of this thesis.

1.1 Motivation

This graduation project is initiated in collaboration with Siemens Digital Industries Software Netherlands. Within the framework of EU-funded research projects, Siemens aims to perform public road testing and demonstration of AVs in the Netherlands. Ensuring the functional safety of an AV is essential to safety of public road testing. In this project, we assess the functional safety of an AV for Dutch highway settings.

Safety systems in AVs should be designed to satisfy functional safety requirements (FSRs), including the detection of problems that prevent safe operation. The AV may be fault-tolerant to such problems, i.e. operate nominally in their presence, or may require a subsequent fallback response. It turns out that the state-of-the-art on safety systems for AVs, presented in Chapter 2, does not fully consider safety requirements associated with an operational design domain in the design or assessment of the safety systems.

The Society of Automotive Engineers (SAE) International J3016B standard [16] defines the operational design domain (ODD) of an AV as the "*operating conditions under which a given driving automation system or feature thereof is specifically designed to function (...)*"[16]. The chosen ODD impacts SOTIF safety requirements, as functional insufficiencies of the AV may be with respect to certain operating conditions, e.g., the vehicle is unable to perceive its surroundings in heavy rainfall. The ODD also influences ISO26262 safety requirements, as it impacts the level of risk posed by malfunctions in vehicle components [17]. Furthermore, an appropriate fallback response also differs by ODD. For example, stopping the vehicle in its driving path as a fallback response may be more dangerous on an active highway lane with sharp curves than on a low-speed urban road [18]. Given the impact of the ODD on the safety requirements and appropriate fallback behavior, it is important to assess the safety system of an AV for a given ODD.

This thesis focuses on a specific level of automation, SAE L4, of the AV. J3016B defines six levels of driving automation, ranging from no automation (SAE L0) to full automation (SAE L5). At L4, the AV is fully autonomous but restricted to a given ODD. If needed, the AV performs a fallback response to reach a safe state without human intervention. J3016B defines a safe state as follows: *"A condition to which a user or an ADS (automated driving system) may bring a vehicle after performing the DDT (dynamic driving task) fallback in order to reduce the risk of a crash when a given trip cannot or should not be completed"* [16].

1.2 Research questions

We tackle two aspects of functional safety for a SAE L4 AV in a Dutch highway setting, with an emphasis on the AI-based sensing and perception subsystem. The first aspect addresses the elicitation of FSRs and appropriate fallback behavior. The second aspect focuses on the assessment of functional safety. The research questions are specified as follows:

RQ1: What functional safety requirements shall be fulfilled by the safety system of a SAE L4 autonomous vehicle in a Dutch highway setting, regarding faults and functional insufficiencies in the AI-based sensing and perception subsystem?

To answer RQ1, we follow the safety processes for elicitation of FSRs outlined in the ISO 26262 and SOTIF safety standards. The safety requirements are derived with respect to faults and functional insufficiencies of the AI-based sensing and perception subsystem in the context of operating conditions in a defined Dutch highway ODD.

RQ2: How to assess the functional safety of a SAE L4 autonomous vehicle regarding faults and functional insufficiencies in the AI-based sensing and perception subsystem?

To answer RQ2, we perform assessment at two levels; design level and implementation level. Design assessment investigates the fulfillment of FSRs by checking if safety-related architectural tactics and AI-safety related best practices are employed in the AV design. At the implementation level, we consider a simulation-based testing approach to assess the fulfillment of FSRs. Simulation-based testing is important for AV testing and validation [5, 9, 19] as it provides a safe, repeatable, and low cost solution for large-scale testing. Siemens has expertise and products related to simulation-based testing of AVs, e.g., Simcenter PreScan [19]. Therefore, it is interesting to Siemens to investigate simulation-based testing methods for functional safety assessment.

The activities for RQ2 ideally require an industrial grade, mature AV software stack which is not available in-house at Siemens or TU/e. Therefore, after comparing three popular open-source industrial-grade AV software stacks; (i) Autoware.Auto [20], (ii) Autoware.AI [21], and (iii) Baidu Apollo [22], we have chosen Apollo for this project. Autoware.Auto is designed for the specific use case of automated valet parking, while Autoware.AI is soon approaching its end-of-life. Apollo does not have such limitations. Furthermore, the software stack has good supporting documentation and an active open-source community.

1.3 Thesis outline

Chapter 2 reviews the state of practice and state-of-the-art of safety systems for AVs. The chapter also explores the state-of-the-art on verification and validation (V&V) methods for the AI-based sensing and perception subsystem of the AV.

Chapter 3 defines the ODD for the Dutch highway setting. The functional concept of the AV and the functional architecture of the Apollo software stack are also presented. The specification in this chapter is used further in the derivation of FSRs in RQ1 and the assessment of functional safety in RQ2.

Chapter 4 describes the elicitation of FSRs for the Dutch highway ODD, answering RQ1. FSRs are derived regarding failures in the sensing and perception components of the AV based on the safety processes outlined in the ISO 26262 and SOTIF standards. The elicited FSRs are refined for the safety safety in Apollo.

Chapter 5 presents the method and results of assessing the functional safety of Apollo, answering RQ2. The fulfillment of FSRs is assessed first at the design level by checking for the use of safety tactics and AI-safety related best practices in the Apollo design. The fulfillment of FSRs is then assessed at the implementation level through simulation based testing.

Chapter 6 presents conclusions and future work.

Chapter 2

Related work

This chapter presents related work to our research activities. Section 2.1 reviews the state-of-practice and state-of-the-art of safety systems for highly autonomous vehicles. Section 2.2 then explores the state-of-the-art on verification and validation methods for the AI-based sensing and perception subsystem of the AV.

2.1 Safety systems for highly autonomous vehicles

We review the state of practice and the state-of-the-art on safety systems designed for highly autonomous vehicles corresponding to SAE automation levels 3 and higher. Although extensive detail on safety systems is not disclosed by AV manufacturers, publicly available safety reports and publications are examined to understand the state-of-practice. The state-of-the-art is more extensive and we categorize it by driver involvement and type of fallback response the safety system provides.

2.1.1 State of practice

We examine public safety reports of AV manufacturers, Waymo [5] and General Motors (GM) [9]. Both manufacturers test their systems on public roads [23] and are therefore assumed to employ proper safety systems. The manufacturers highlight the importance of "safety by design", which incorporates safety into every development stage.

Safety mechanisms in Waymo and GM AVs include redundancy in safety-critical hardware components including sensors, actuators, networks, power systems, and computation. Diversity in environment sensing and localization methods are also highlighted due to limitations of individual modalities. In addition, an independent collision detection and avoidance system is present as a backup for the main functionality. During operation, the system status is continually monitored to detect faults in components. GM has a *vehicle health monitor* module which manages diagnostics for all AV components. Waymo states the vehicle runs "thousands of checks per second, looking for faults" [5]. In addition the AV automatically detects scenarios which it may not be able to handle safely, for example, environmental conditions outside of its ODD. Upon such a fault or complex scenario, the vehicle switches to degraded operation, or performs a fail-safe maneuver and deactivates the vehicle.

The publication *Safety first for automated driving* (SFAD) [6] summarizes safety by design methods for AVs. The publication is by a collaboration of major OEMs, tiered suppliers, and AV technology providers. It systematically translates safety principles into necessary AV capabilities, components, and architecture. The safety principles are drawn from the ISO 26262 and SOTIF standards among

other sources.

Two key safety-related capabilities of the AV are identified in SFAD. The first is to determine when nominal operation of the AV is not being achieved. Several factors which influence nominal operation are highlighted including technological limitations (functional insufficiencies), systematic and random failures (faults) and human misuse. The second capability is responding to a failure of nominal operation such that safety is not compromised. For example, upon a failure, the AV may degrade its operation to a lower driving speed. A generic functional architecture is proposed, incorporating these capabilities. The *monitoring* functionality to detect failures is included as "sub-elements [embedded within a component] or as a separate element [component]" [6]. The monitoring functionality detects insufficient nominal performance at component or system-level. A coordinating component switches between nominal and degraded operation modes.

2.1.2 State-of-the-art

The state-of-the-art on safety systems for AVs is identified using a google scholar search with the following search string:

(26262 OR "functional safety") AND (21448 OR SOTIF) AND (fail-operational OR "degraded" OR fail-safe OR fail-degraded OR limp-home OR "fault tolerant") AND ("autonomous driving" OR "autonomous vehicle" OR "self driving")

We first discuss safety systems which consider driver involvement as part of the fallback response. Next, safety systems which achieve a fallback response without driver intervention are discussed. Here, we distinguish between systems that attain a fail-safe response and those which include degraded operation modes. A fail-safe response transitions the vehicle in a short time frame to a state where the AV can deactivate safely. Instead, in degraded operation, the AV may continue operating with some constraints for a limited duration. All types of safety systems should include monitoring and detection of situations which prevent safe operation e.g., a fault in a AV component.

Driver involvement in a fallback response may be considered for a SAE L3 AV. According to the J3016b standard [16], a SAE 3 AV has two additional responsibilities once it has identified a situation which requires a fallback response. Firstly, it must warn the driver to take-over control of the vehicle. Then, the AV must sustain safe operation for several seconds to allow the driver sufficient time to take-over vehicle control.

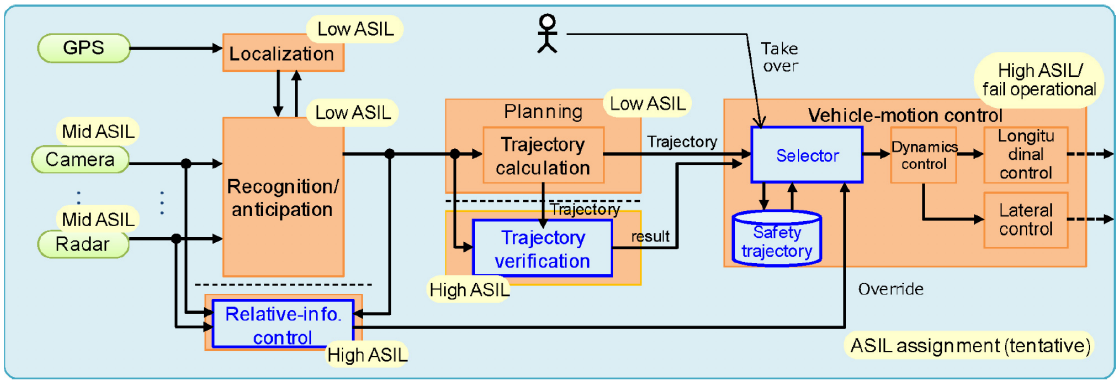

Figure 2.1: Proposed architecture by Otsuka et al. [24] for a SAE L3 AV. Relative-info-control and trajectory verification are monitors for perception and planning subsystems respectively. The selector component chooses whether nominal or safety trajectory should be used for actuation.

Otsuka et al. [24] propose monitor/actuator pairs for the detection of random and systematic failures of sensing, perception and planning components, as shown in Figure 2.1. Upon a failure,

a warning is issued for the driver to take over vehicle control. To sustain safe operation following a failure till the driver takes over, a *safety trajectory buffer* component is proposed, as shown in the figure. The component stores ego vehicle trajectory of several seconds length which has been computed taking into account predicted future states of other road users. However, Otsuka et al. do not take into account challenges related to human-automation interaction, leading to situations where a driver may not respond to the issued take-over warning [25]. Horwick et al. [26] propose transition to a safe state if the driver does not respond to the take-over warning. The safe state considered is "standstill or low speed at a non-hazardous place". Horwick et al. consider several types of failures for their safety system such as driver misuse, faults in components, implausible vehicle behavior, and situations outside of the AV's ODD.

According to the J3016b standard [16], a SAE L4+ AV must be capable of transitioning itself to a safe state, without driver intervention. Here, we cover safety systems capable of providing such a safety response. We first describe systems which aim at fail-safe states and then cover those which include degraded operation. Novickis et al. [27] present a functional architecture and its implementation for a SAE L4+ AV which includes emergency braking upon a critical failure. Critical failures are detected for sensors and the nominal logic controller by sensor data monitoring and a heartbeat safety mechanism respectively.

vom Dorff et al. [28] propose a safety system which enables a more gradual stop of the vehicle. The presented architecture includes an *observer* component which assesses the safety of the planned AV trajectory with respect to its environment. The planned AV trajectory at each timestep includes a gradual stop, stored in a *safety trajectory buffer* component, similar to the safety system of Otsuka et al. If the planned trajectory is considered unsafe, the trajectory in the safety trajectory buffer component brings the vehicle to a stop. The authors claim that the proposed design enables fail-safe operation without costly redundancy in perception and planning components. However, the authors note the the safety trajectory buffer is only suited to low-speed settings, given challenges in behavior prediction of other road users [29, 30].

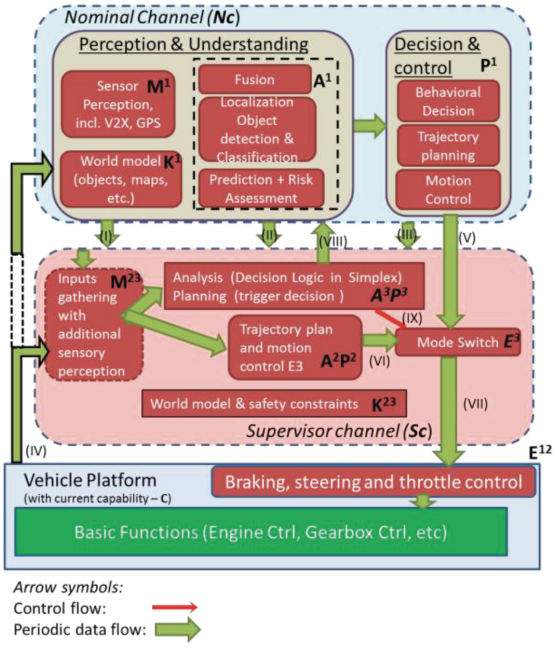

Figure 2.2: Proposed architecture by for a L3+ AV by Torngren et al. [31]. The arrows and red blocks represent dataflows and functional components respectively. The safety channel is termed as supervisor channel (Sc).

Architectures proposed for complex fail-safe maneuvers e.g., parking on the roadside, are observed

to consist of separate channels for nominal functionality and fail-safe functionality. Torngren et al. [31] present a functional architecture with a nominal and a safety channel, shown in Figure 2.2. The safety channel contains two logic loops, one for failure detection and the other to transition the AV to a safe state. The authors consider two failure types: hardware faults and systematic faults. They also indicate the possibility of detecting performance limitations via the safety channel. Ishigaooka et al. [32] present an architecture based on the 1-out-of-2 system with diagnosis pattern, with a primary and secondary control logic. Run-time monitoring on each control output is performed to detect controller failures.

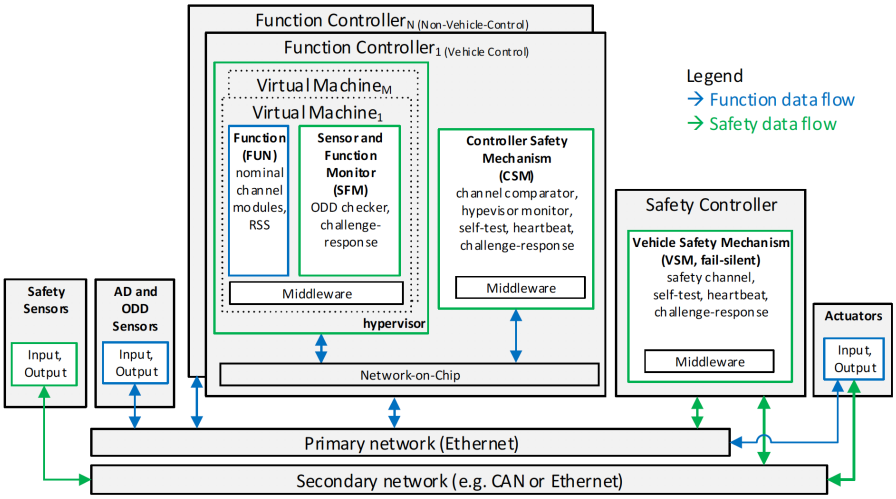

Figure 2.3: The architecture presented by Fu et al. [33] based on their proposed safety pattern for a L3+ AV. The safety sensors and safety controller are used specifically for the fallback operation.

Luo et al. [34] propose a 3-channel safety architecture pattern, inspired by the *Safety Executive* pattern [35]. The pattern includes: (i) nominal channel, (ii) dedicated channel for failure detection, and a (iii) safety channel responsible for transition to a safe state. The pattern covers detection of random HW failures, as well as conflicts in crosschecks between the nominal and safety channel which may indicate other failures. Fu et al. [33] also present a pattern enabling multiple fail-safe modes e.g., park on roadside, gradual stop, and a degraded operation mode if the safety components themselves become faulty. Faults in the controllers, safety monitors, network, sensors are covered as well as the detection of outside ODD conditions. The architecture presented by Fu et al., based on their pattern, is shown in Figure 2.3.

Several of the previous discussed safety systems proposed the use of a simplified safety channel as a secondary channel to the nominal functionality. In contrast, Fruehling et al. [36] claim that safety channels composed of simplified robotic algorithms cannot achieve complex fail-safe strategies "in many conceivable scenarios", although an example of such scenarios was not provided. Furthermore, they suggest the use of artificial intelligence (AI) components in the safety channel. The proposed architecture consists of a primary and secondary controller, each containing two artificial intelligence (AI) based data processing and control units. Within each AI-based unit, validation of individual component outputs through local diagnostics / prognostics monitoring is performed. The two AI-based units are also in "continuous result comparison" with each other, as are the two controllers.

Finally, we discuss the state-of-the-art on safety systems which enable the AV to safely continue operation with degraded vehicle functionality upon failures. Colwell et al. [38] suggest the restriction of the operational design domain (ODD) at runtime in response to failures. This safety strategy maps failures to corresponding restrictions on the ODD. For example, left turns at T-intersections are excluded from the ODD if the right radar on the AV has failed. Subsequently,

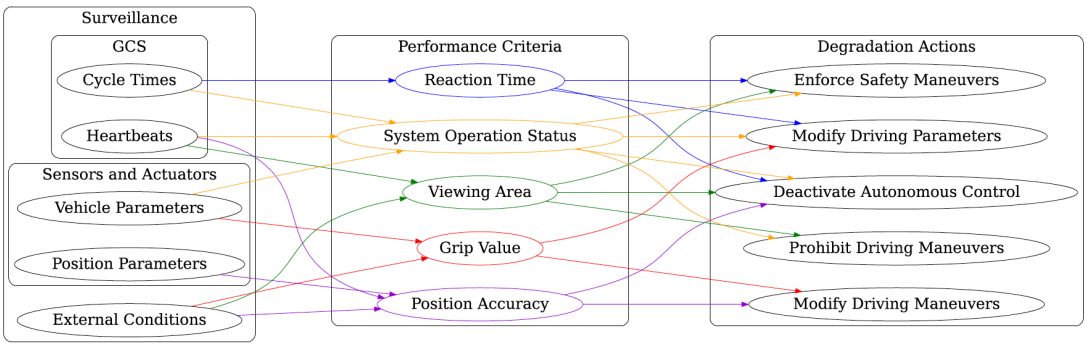

Figure 2.4: Degraded functionality concept by Reschka et al. [37] for their research prototype AV. The target automation level is not mentioned.

the AV modifies its functionality to the restricted ODD (ROD). Covered failure types include hardware, software component failures or performance limitations which may arise, for instance, due to interference of snow in a lidar scan. Reschka et al. [37] also propose functional degradation including the modification of driving parameters, for example vehicle speed, or the restriction of certain driving maneuvers, for instance a lane change maneuver. The degradation concept, as shown in Figure 2.4, is based on performance criteria such as system health status (heartbeat, cycle times of software, hardware modules) or current sensor field-of-view. The safety systems of Colwell et al. and Reschka et al. both include fail-safe measures if degraded operation is not possible.

Molina et al. [39] aim for complete independence of safety modules from the nominal functionality. The safety module detects faults in nominal functionality and also monitors system variables e.g., driving speed, as a check for abnormal behaviors. Upon a problem, the safety module enforces degraded states on the vehicle such as a reduced driving speed.

The state-of-the-art safety systems are summarized in Table 2.1. Here, we present: (1) SAE level of automation as mentioned by the authors, (2) types of failures covered, and (3) the fallback response (safe states).

We note that many of the state-of-the-art safety systems are not designed for a specific ODD. In fact, few related works mention even a target use case - Luo et al. [34] aim at a truck platooning use case, vom Dorff et al. [28] determine their proposal to be useful in low-speed settings, and the AV prototype from Reschka et al. [37] is aimed at urban settings.

The assessment of the proposed safety systems also does not consider an ODD. Proof-of-concept studies are performed by Colwell et al. [38] and Ishigaooka et al. [32]. Colwell et al. [38] test failure scenarios, for example a front facing camera impairment, with a proof-of-concept implementation. Ishigaooka et al. [32] conduct a case study on a simple indoor track, also by injecting failures. Formal verification is used by Fu et al. for safety requirements such as: “the AV successfully transitions to a degraded or emergency mode when necessary” [33]. Many state-of-the-art systems are not assessed at all [36, 26, 34].

Safety system	**SAE level of automation**	**Types of failures**	**Safe state**
Otsuka et. al [24]	SAE L3	random and systematic failures of sensing, perception and planning components	sustain safe operation till driver takes over control
Horwick et. al [26]	undefined	external e.g., driver misuse, ODD exit, faults in individual components, implausible vehicle behavior	fail-safe maneuver if no driver response to takeover request
Norvickis et al. [27]	undefined	sensor data health, controller status, and unavoidable collision	emergency braking
vom Dorff et al. [28]	SAE L4	safety of trajectory given nearby objects and road users	gradual stop in driving path
Torngren et al. [31]	SAE L3+	random HW faults, systematic faults, and "possibly performance limitations"	complex fail-safe manuevers e.g., parking on the roadside
Luo et al. [34]	SAE L3+	randow HW failures, conflicts between nominal and secondary channels	
Fruehling et al. [36]	SAE L4	validation of individual component outputs through local diagnostics / prognostics monitoring, result comparison between redundant system units	
Ishigaooka et al. [32]	SAE L4+	run time monitoring on controller outputs to detect controller failure	low-performance secondary controller e.g., lower driving speed
Fu et al. [33]	SAE L3+	component faults e.g., sensors or controller and ODD exit	multiple fail-safe modes and detection of out-of-ODD conditions
Colwell et al. [38]	SAE L3+	hardware or software component failures and performance limitations	reduction in ODD, fail-safe
Reschka et al. [37]	undefined	system health, performance criteria e.g., sensor field-of-view	modify driving parameters or restrict certain driving maneuvers e.g., restrict lane change, fail-safe
Molina et al. [39]	SAE L5	faults in nominal functionality, improper system and environment variables	degraded operation e.g., reduced driving speed

Table 2.1: Proposed safety systems. The SAE level of automation is as described by the authors themselves.

2.2 V&V methods for the AI-based sensing and perception subsystem

We identify related work on V&V methods for the AI-based sensing and perception subsystem by retrieving secondary studies from google scholar with the following search string:

("neural network" OR "machine learning" OR "deep learning" OR "artificial intelligence") AND (26262 OR 21448) AND (verification OR validation) AND ("systematic literature review" OR "systematic review")

Eight relevant review papers were identified; five specific to V&V of autonomous vehicles, and three related to V&V of AI-based safety-critical systems across domains. The scope of our review is limited to AI safety V&V of the sensing and perception subsystem related to the two automotive safety standards, i.e. faults as covered by ISO 26262 and functional insufficiencies with respect to ODD conditions as covered in SOTIF. We do not address the safety of AI components w.r.t. other challenges such as interpretability, transparency, and robustness to adversarial examples [14].

Furthermore, we aim to identify V&V methods that may be practical for a virtual-world simulation test environment. These methods are interesting to the assessment of functional safety through a simulation-based testing approach, part of our aim to answer RQ2. We thus extract primary studies from the review papers based on these considerations. The following subsections describe our findings. Section 2.2.1 describes and categorizes the test conditions simulated in the studies. Section 2.2.2 then discusses the corresponding safety assessment performed.

2.2.1 Test conditions

The conditions simulated for the testing of the AI-based sensing and perception subsystem of an AV are: (1) faults, (2) environmental conditions, (3) image transformations, and (4) sensor noise. A fault may be injected at the input of the AI-based component, within the neural network (NN), or at the output of the component. Jha et al. [40] inject faults related to corrupted variables (single or multi-bit faults) at the raw sensor data level (input data) and within the NN. They also simulate erroneous outputs of the AI-based object perception component, for example, a false positive detection. Chen et al. [41] simulate hardware transient faults which lead to bit flips within the NN. Rubaiyat et al. [42] inject two types of faults for the AI-based object perception component output: random additions to the outputs, or a disconnected component (no output).

Environmental conditions including rain, fog, and snow are examined by Rubaiyat et al., Zhang et al. [43] and Tian et al. [44]. Rubaiyat et al., and Tian et al. also analyze the impact of image transformations (input data) including contrast, brightness, blur, translation, rotation. Finally, Rao et al. [45] determine sensor noise to be an important influence on object detection performance through a failure mode and effect safety analysis. The authors test the effects of salt and pepper camera sensor noise for their study. This type of noise may occur due to a software or hardware failure of the camera sensor [45].

The test conditions simulated in the studies are summarized in Figure 2.5.

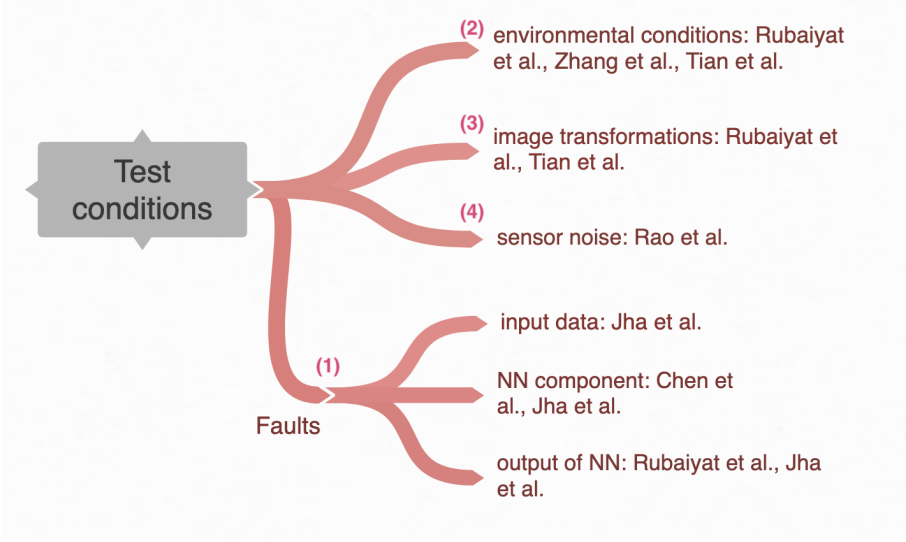

Figure 2.5: AI Safety V&V methods: test conditions

2.2.2 Safety assessment

Several studies test directly on real-world image datasets, or synthesize test conditions from the original image dataset. Virtual-world simulation is employed only by Jha et al. and Rubaiyat et al. Rubaiyat et al. also use an image dataset for their vision module, but evaluate the system under test, Comma.ai OpenPilot [46], in the OpenPilot PC simulation environment. Jha et al. make use of a popular open-source simulation environment, Carla [47], as well as a proprietary simulation environment from Nvidia [48].

Table 2.2 summarizes the evaluation setup of the studies. Some studies perform safety assessment at the subsystem-level of the AV, testing only the object detection functionality or the NN model itself. In this case, the test criteria used relates directly to NN performance such as misclassification rate or precision and recall. In the studies performing safety assessment at system-level, the subjects under test are diverse, ranging from end-to-end NN steering models for AVs, advanced driver assistance systems (ADAS), and complete AV software stacks. For the end-to-end NN models, the test criteria used includes violation of metamorphic relations in the output or deviations from the ground truth output. Finally, the studies conducted on ADAS and AVs use vehicle-level criteria, such as safety distance violation or other hazardous behavior.

V&V method	**Subject under test**	**Safety assessment level**	**Test criteria**
Chen et al. [41]	End-to-end steering models: Nvidia [49], Comma.ai [50], and image classification models: VG11, VG16 [51]	sensing and perception subsystem, system level	deviations from correct steering angle and misclassification of images
Rao et al. [45]	CNN object detection model	sensing and perception subsystem	precision, recall of model
Rubaiyat et al. [42]	Comma.ai OpenPilot (ADAS) [46]	system level	violation of longitudinal safety distance, unnecessary deceleration to a full stop, or not maintaining of lane
Jha et al. [40]	Apollo [22], NVIDIA Drive AV [48]	system level	violation of longitudinal or lateral safety distances
Zhang et al. [43]	End-to-end steering models: Udacity [52]	system level	violation of metamorphic relations on steering angles
Tian et al. [44]	End-to-end steering models: Udacity [52]	system level	violation of metamorphic relations on steering angles

Table 2.2: AI safety V&V methods: test subjects and test criteria

Chapter 3

ODD and functional representation of the AV

This chapter presents the ODD and functional representations of the AV. Section 3.1 specifies a Dutch Highway operational design domain (ODD), representing the driving conditions in which the AV shall safely operate. Section 3.2 presents the functional concept, a functional depiction of the AV operation in that ODD. Section 3.3 describes the functional architecture of the chosen AV software stack, Apollo, and details its sensing and perception components. Finally, Section 3.4 unifies the two representations by describing how the functional concept is achieved in the functional architecture.

The ODD and functional representations developed in this chapter are used in both the derivation of functional safety requirements in Chapter 4 and the assessment of functional safety in Chapter 5.

3.1 Operational design domain (ODD)

The ODD definition is important for the functional safety of the AV as it explicitly describes the operating conditions in which the AV can be expected to safely operate. Real-world operating conditions of the AV contain many variables such as weather conditions, road infrastructure, and traffic conditions. The ODD identifies supported conditions, e.g., the AV may only operate in fair weather conditions. It also captures required conditions for AV operation, such as clear lane markings or availability of GPS signals. Regulatory bodies such as the U.S. Department of Transport (DOT) recommend that AVs be assessed, tested, and validated for a clearly defined ODD prior to deployment [53].

To properly define our ODD, we base our definition on two documents. A document from the SAE Automated Vehicle Safety Consortium (AVSC) [54] outlines best practices for a framework and approach to the ODD definition. The second document is from the U.S. National Highway Traffic Safety Administration (NHTSA) [18], which presents a taxonomy of attributes to define an ODD. The documents are used in a complementary fashion; the approach from the AVSC document is followed to describe the taxonomy of attributes in the NHTSA document.

The ODD is defined by a bottom-up approach, starting from an operational route for the autonomous vehicle. The other variables pertaining to the ODD can then be defined by:

- Characterising the identified route or road network, e.g., the types of intersections or types of road users.
- Identifying operational constraints that are not part of the ODD, e.g., extreme weather conditions or specific times of day such as morning rush-hours.

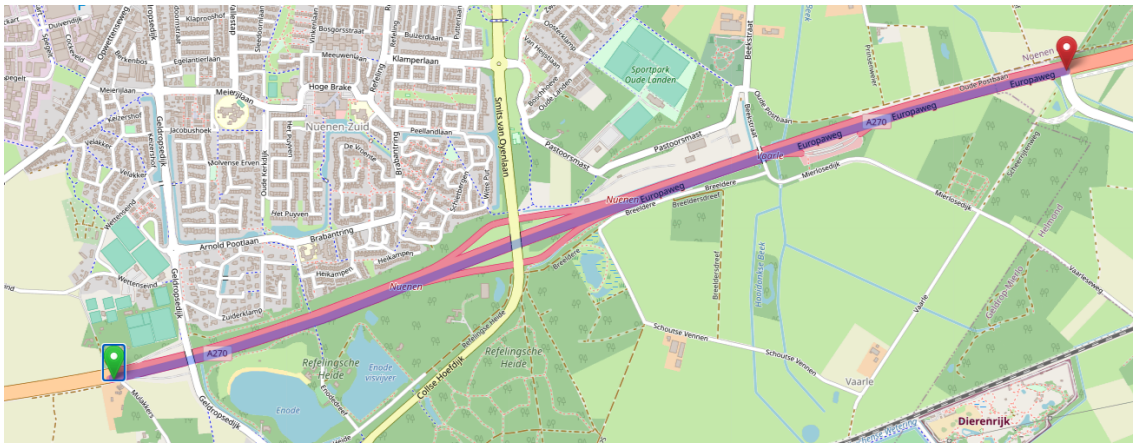

Figure 3.1: Highway A270, Netherlands. The green and red markers on the map correspond to the Eindhoven and Helmond ends of the route respectively.

The chosen route for our ODD is shown in Figure 3.1. The route corresponds to highway A270, which is part of the highway between the cities of Eindhoven and Helmond in North Brabant, The Netherlands. It spans a distance of 3.4km[1]. The complete ODD description is presented in `Description_ODD.pdf` [2].

Most attributes of the route can be deduced from maps, Google Street View, and guides from Dutch authorities [55], [56]. The A270 highway road infrastructure is not uniform in attributes such as speed limit, types and number of lanes. A part of the A270 highway lane schematic is shown in Figure 3.2. Weather conditions in the North Brabant region include rainfall, snow, sleet, wind, and fog. The temperatures in the region are described by a conservative range between -15 and 40 degree celsius [57]. Finally, changes in traffic conditions due to an accident or construction may also occur within the ODD.

Some assumptions made in characterizing the route are outlined as follows:

- Connectivity related attributes are not considered as the AV is not connected, with the exception of GPS signals.
- "Interference zones" and "negative obstacles" are assumed non-existent due to a lack of information. Interference zones are regions where the GPS signal may be obstructed. Negative obstacles refer to road infrastructure, such as ditches or pits, which may lead to false detections [58]. Specialized on-road testing is required to characterize such attributes.
- Other forms of particulate matter besides fog are not considered. Additionally, weather-induced roadway conditions, except small puddles of water from rain, are unlikely in the local climate.
- The attribute "partially occluded" is not defined as its meaning was not clear from the NHTSA document or through other resources.

Operational constraints for the route must be defined based on the current limitations of AV technology. We examine the state-of-practice through safety reports of Waymo [5], General Motors [9], and BMW [59] to understand typical operational conditions for high levels of automation. We observe that the AV is designed to operate in all illumination conditions. Weather conditions are limited to moderate inclement weather, not including heavy amounts of rainfall, snow, sleet, wind

[1] `https://www.openstreetmap.org/directions?engine=fossgis_osrm_car&route=51.4564%2C5.5408%3B51.4657%2C5.5865#map=15/51.4610/5.5636`

[2] `https://surfdrive.surf.nl/files/index.php/s/2RwiL5taL9nTTFj`

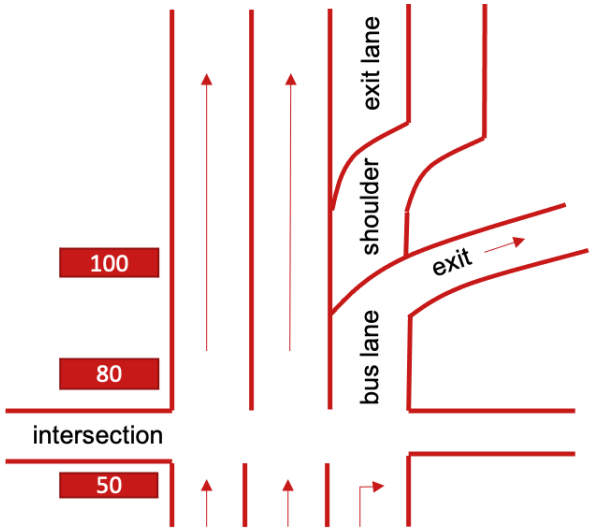

Figure 3.2: Lane schematic for a part of the A270 Highway ODD. The lengths are not representative of actual distances.

or fog which are known to degrade AV performance [60]. We apply the same constraints to our ODD.

3.2 Functional concept

The functional concept of the L4 AV operation in the A270 highway ODD is made up of four *operational modes* and two *categories of functions*. The two categories of functions are (i) avoid collision with other road users and obstacles, and (ii) follow traffic rules in the ODD. The functions within each category are differentiated based on the four operational modes, summarized in Table 3.1. The basic operational mode is the *driving in lane* mode. While driving in a highway lane, the AV must avoid collisions in its driving lane and follow highway traffic rules. The term *object* in the avoid collision functions refers to both obstacles and other road users.

On certain occasions, the AV may change driving lanes on the highway. While doing so, the vehicle must avoid collisions with traffic in the target lane. It must also indicate the lane change, as well as give priority as per highway rules to any passing by vehicles [55]. These functions are part of the *changing lanes* operational mode. Two settings in the ODD also require additional AV functionality to the basic *driving in lane* mode. In the intersection setting, the AV must interact with cross-traffic vehicles and follow traffic lights which dictate traffic flow. The functions specific to the intersection are part of the *crossing an intersection* operational mode. Likewise, a merging point is a designated point for traffic in multiple lanes to converge into a single lane. Here, the AV must avoid collisions with merging vehicles from another lane. This is part of the *crossing a merging point* operational mode.

Note that the functions in the *follow traffic rules* category only cover a small set of traffic rules applicable to the A270 Highway. While this is sufficient for our proof-of-concept study, the AV functional concept can be extended to include the entire set of Dutch highway traffic rules.

Category	**Function**	**Operational modes**
avoid collision with other road users and obstacles	avoid collision with an object in driving lane	all
	avoid collision with an object at an intersection	crossing an intersection
	avoid collision with an object at a merging point	crossing a merging point
	avoid collision with an object during lane change	changing lanes
follow traffic rules in the ODD	drive within speed limit	all
	drive in correct lane	all
	maintain driving lane	all except changing lanes
	indicate lane change	changing lanes
	give priority during lane change	changing lanes
	follow traffic lights	crossing an intersection

Table 3.1: Functional concept of the L4 AV operation in the A270 highway ODD

3.3 Functional architecture

The latest version of Apollo, v6.0, does not include the entire perception functionality[2]. Therefore, the derivation of the Apollo functional architecture is based on the software architecture [61], documentation, and code of Apollo v5.5. The functional architecture is presented in Figure 3.3. For traceability, the naming of the functional components is matched to that of the Apollo software modules.

[2]the camera-based object detection, tracking, and classification is not present in v6.0, see https://github.com/ApolloAuto/apollo/tree/v6.0.0/modules/perception

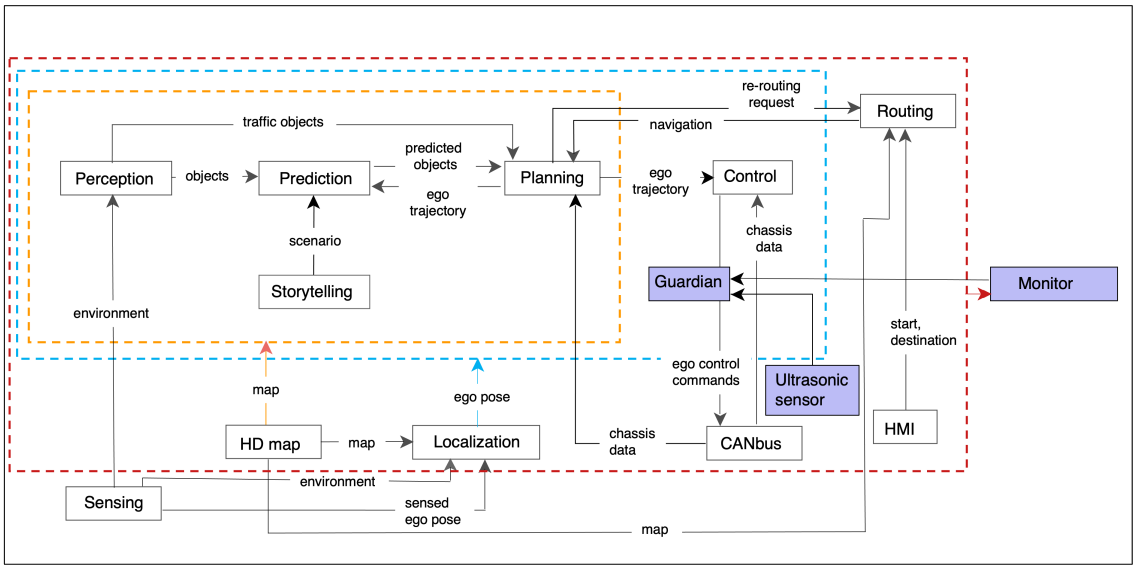

Figure 3.3: Functional architecture of Apollo. Blocks and arrows represent functional components and dataflows respectively. A dashed box indicates data flow to all functional components within the box. The safety components are shown in purple.

The Sensing functional component is external to the Apollo software stack, but is represented here as part of the AV system-level functional architecture. We exclude functions not directly relevant to the driving task e.g., status updates to occupants via the human-machine interface (HMI) component.

Functional component	**Description of functional component**
CANbus	abstracts the vehicle platform; executes control commands and retrieves vehicle chassis data.
Control	generate ego vehicle control (actuation) commands
Planning	generate ego vehicle future trajectory
Routing	generate ego vehicle navigation route
Localization	estimate ego vehicle current pose
Storytelling	manage information on ego vehicle operating scenario
Prediction	predict future trajectory of other road users
Perception	determine state information of other road users and obstacles, and of traffic lights
Sensing	capture raw data information on the environment and the ego vehicle's current pose
HD map	provide map information to other modules
HMI	occupant (user) interface to the AV

Table 3.2: Descriptions of the functional components. The term *ego vehicle* refers to the AV.

Table 3.2 and Table 3.3 elaborate further on the functional components and data flows. Note that two methods for localization are possible in Apollo, i.e., real-time kinematic (RTK) positioning and multi-sensor fusion (MSF) [3]. While both methods use the ego vehicle pose information extracted by the *Sensing* component, MSF additionally uses the sensed environment data. The described functional architecture is based on the MSF localization method. Furthermore, while Apollo

[3]https://github.com/ApolloAuto/apollo/tree/master/modules/localization

Functional component	**Output data flow**	**Description of data flow**
CANbus	chassis data	ego vehicle chassis data e.g., engine rotations per minute
Control	ego control commands	ego vehicle actuation values e.g., steering angle
Planning	ego trajectory	ego vehicle target future position, speed, and acceleration
Planning	re-routing request	request for re-computation of ego vehicle driving route
Routing	navigation	ego vehicle road and lane level navigation information
Localization	ego pose	ego vehicle position, orientation, linear velocity, etc.
Storytelling	scenario	scenario information
Prediction	predicted objects	detected objects with predicted trajectories and priorities
Perception	objects	detected objects with heading, velocity and classification information
Perception	traffic objects	traffic light bounding boxes with color labels
Sensing	environment	raw data of environment perception sensors
Sensing	sensed ego pose	ego vehicle raw data position and orientation
HD map	map	lane information, road infrastructure, traffic signs and lights
HMI	start, destination	start and destination points for navigation

Table 3.3: Descriptions of the output data flows of the functional components. The term *ego vehicle* refers to the AV.

supports interactions between the *storytelling* component and all other modules, it currently only interacts with the prediction module. This is reflected in the functional architecture.

The sensing and perception functional components are detailed further in Figure 3.4. The additional detail identifies functions not captured in the system-level architecture (1), e.g., *Lane detection and tracking*. Furthermore, diverse redundancies are visible in the detailed architecture (2), e.g., the multiple *Sense environment* components. Finally, we obtain more insight into interactions of other components with the sensing and perception components. For example, the *localization* data flow (3) is used within the perception component by the *Traffic light detection and recognition* component, and the *Object detection, classification and tracking (radar)* component. A dataflow introduced in the detailed architecture is *lane* (4), which extracts road lane information.

We finally introduce the three safety components in the Apollo functional architecture. The *monitor* component (1) ensures the integrity of other components and triggers the *guardian* component (2) if a problem is detected. When triggered, the guardian component executes a safety response. The *ultrasonic sensing* component (3) perceives near-distance objects during the safety response.

3.3.1 Comparison against other functional architectures

Several functional architectures for autonomous vehicles have been proposed in literature and industry. Here, we compare the Apollo functional architecture to a state-of-practice architecture from the safety publication, *safety first for automated driving* (SFAD) [6], also described in Section 2.1. Related to sensing and perception components, SFAD has two differences compared to Apollo. Firstly, perception includes traffic sign detection functionality. This adds diverse redundancy to traffic sign information retrieved from HD maps. Secondly, the sensor fusion component creates a comprehensive model of the surroundings, while in Apollo, sensor fusion is limited to dynamic objects. This adds diverse redundancy to the perception of other objects in the environ-

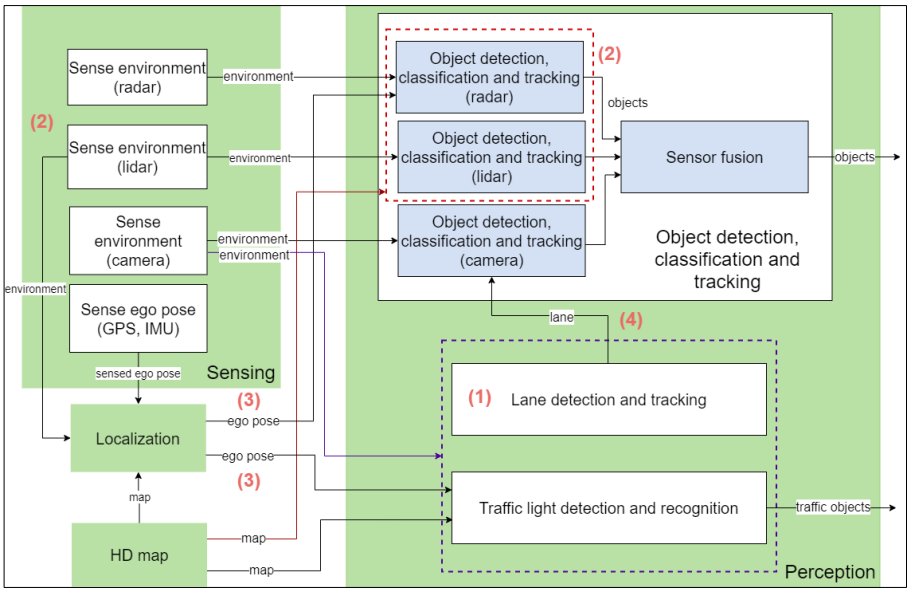

Figure 3.4: Sensing and perception functional architecture. System-level, subsystem-level, and further detailed components are shown in green, white, and blue respectively. A dashed box indicates data flow to all functional components within the box. The architecture is derived mainly from the software architecture of the perception module in Apollo. Some dataflows are derived from other documents at `https://github.com/ApolloAuto/apollo/`:

- `blob/v5.5.0/docs/specs/traffic_light.md`
- `blob/v5.5.0/docs/specs/3d_obstacle_perception.md`
- `tree/v5.5.0/modules/localization`
- `blob/v5.5.0/docs/specs/Apollo_5.5_Software_Architecture.md`

ment, e.g., traffic lights. Thus, the Apollo functional architecture has low diverse redundancy for environment perception, apart from dynamic objects.

Related to safety mechanisms, as per Apollo documentation, the Apollo *monitor* component only performs health monitoring of the AV components. Instead, SFAD discusses monitoring of the ODD, user state and vehicle platform state besides health monitoring of the AV components. A dedicated *guardian* component is present in Apollo for executing the safety response. In SFAD, a central coordinating component handles all operation mode switching, including those for a safety response. The mode switching is communicated to the planning component which acts accordingly. The differences between Apollo and SFAD related to safety mechanisms can be summarized as: (i) the safety related monitoring in Apollo, as apparent from its documentation, is limited to health monitoring while other safety-related monitoring is performed in SFAD, and (ii) the architectures differ in the manner in which the safety response is executed.

3.4 Functional allocation

Functional allocation describes how the functional concept of the AV in Section 3.2 is achieved in the Apollo functional architecture. By way of illustration, we outline the functional allocation for the function *avoid collision with an object in driving lane* as shown in Figure 3.5. Note that the safety components are not shown in the figure, as they do not contribute to the nominal operation of the AV as covered in the functional concept. To achieve the example function, the *Sensing* component (1) captures raw data on the objects in the environment and the ego vehicle's pose.

The *Localization* component (2) uses the raw data to determine the ego vehicle's pose with respect to its environment with the help of map information from the *HD map* component (3).

The *Perception* component (4) extracts state information on objects in the environment from the raw environment data, using the map and ego vehicle pose for filtering and coordinate transformation. The *Prediction* component (5) estimates the future trajectory of the detected objects. It additionally assigns a priority to each detected object, which indicates the importance of the object to the behavior of the ego vehicle. This assignment makes use of the map, ego vehicle pose, the current scenario and the last planned ego vehicle trajectory. The current scenario is determined by the *Storytelling* component (6) based on map and ego vehicle pose.

The *Planning* component (7) generates the ego vehicle trajectory to avoid collisions, using the predicted object trajectories and the ego vehicle's current state. The ego vehicle's current state is determined through vehicle chassis data, ego vehicle pose, and map information. The *Control* component (8) generates the ego vehicle control commands based on the ego vehicle trajectory, chassis data, and ego vehicle pose. Finally, the control commands are executed in the *CANbus* component (9), which additionally retrieves the vehicle chassis data for feedback to the planning and control components.

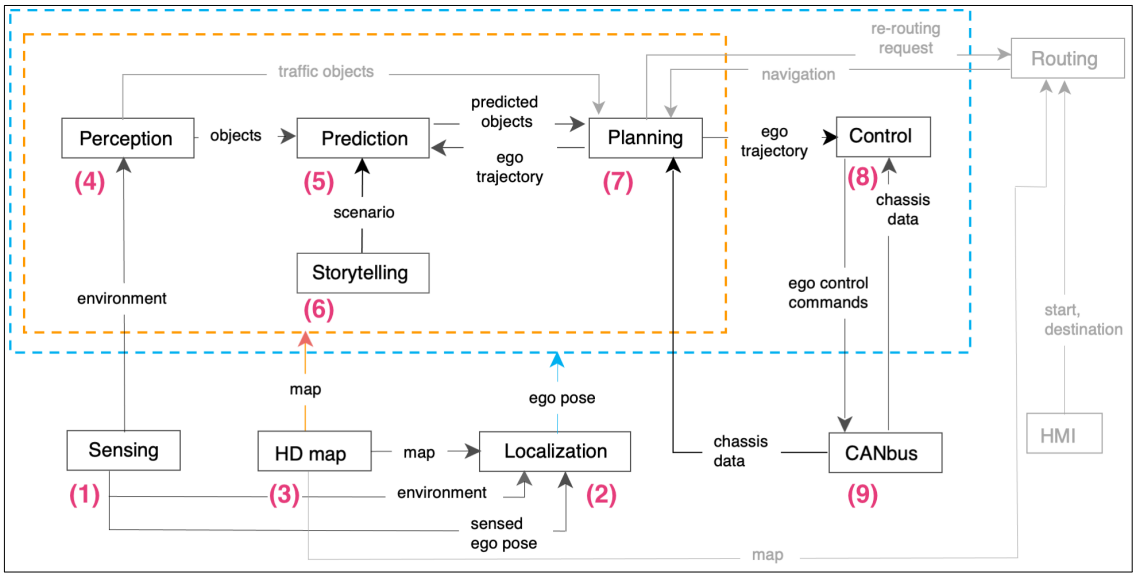

Figure 3.5: Functional allocation for the function *avoid collision with an object in driving lane*. The grayed out components and data flows do not contribute to the function. Note that the safety components are not shown in this figure.

The functional allocation for the complete set of functions is presented in `FunctionalAllocation.pdf` [2]. Due to unclear documentation in Apollo[4] on the dataflow *re-routing request*, it is not considered in the functional allocation.

[2] `https://surfdrive.surf.nl/files/index.php/s/2RwiL5taL9nTTFj`

[4] Apollo documentation states: "Under certain scenarios, the planning module might trigger a new routing computation by sending a routing request if the current route cannot be faithfully followed." (`https://github.com/ApolloAuto/apollo/blob/v5.5.0/docs/specs/Apollo_5.5_Software_Architecture.md#planning`). Although Apollo code for the planning and routing components was briefly examined, we could not determine the scenarios in which the request may be made.

3.5 Summary

This chapter defined the ODD and functional representations of the AV. These artefacts are prerequisites for the derivation of functional safety requirements, detailed in Chapter 4, and the assessment of functional safety in Chapter 5. The A270 Highway route in Netherlands is chosen as the ODD of the L4 AV, and characterized according to the AVSC and NHTSA guidelines. The functional concept of the AV operating in its ODD is depicted by four operational modes: (i) driving in lane, (ii) changing lanes, (iii) crossing an intersection, and (iv) crossing a merging point. Functions within two categories: (i) avoid collision with other road users and obstacles and (ii) follow traffic rules in the ODD, are identified for each operational mode.

The functional architecture of the chosen AV software stack, Apollo, is derived from documentation and code. The sensing and perception components are further detailed, revealing diverse redundancy in the component and offering more insight into their functionality. Finally, the functional concept of the AV is linked to the components and dataflows in the functional architecture through functional allocation.

Chapter 4

Functional safety requirements for a Dutch highway ODD

This chapter derives functional safety requirements (FSRs) regarding faults and insufficient functionality in the sensing and perception components of a L4 autonomous vehicle. The derivation of functional safety requirements is based on the frameworks presented in the automotive safety standards, ISO 26262 and SOTIF. The specific part of ISO 26262 which guides the derivation of FSRs is the concept phase (part 3) of the standard [8]. The concept phase has three steps: (1) *item definition* to gather the representation of the vehicle and its surroundings (2) *hazard analysis and risk assessment* (HARA) for identifying potential safety hazards for the AV and defining corresponding safety goals, and (3) *functional safety concept* where safety goals are mapped into safety requirements which are allocated to specific functional components.

The notion of FSRs is not explicit in the SOTIF standard safety lifecycle shown in Figure 4.1. However, three analysis steps (clauses 5-7), performed prior to the design and V&V phases (clauses 8-11), are identified to be necessary for the derivation of FSRs with the help of related work [62]. The steps of *specification and design* (clause 5) and *SOTIF related hazard identification and risk evaluation* (clause 6) are similar to the first two steps of the ISO 26262 concept phase. The step of *identification and evaluation of functional insufficiencies and triggering conditions* (clause 7) is unique to SOTIF and aims at identifying functional insufficiencies of components that may lead to hazardous behavior of the AV. We aim to define FSRs related to such identified functional insufficiencies.

The functional specification presented in Chapter 3 already covers the representation of the AV and its driving conditions required in the two safety frameworks. Next, Section 4.1 presents the common step of *hazard analysis and risk assessment* (HARA) in the two safety frameworks. To perform HARA, we follow guidelines in the ISO 26262 standard, since the SOTIF standard states: *"The identification and evaluation of hazards caused by the intended functionality is aligned with HARA of ISO 26262"* [10].

The subsequent section, Section 4.2, maps the safety goals to functional safety requirements for sensing and perception components. Here, a fault tree analysis is performed and then followed by steps specific to ISO 26262 and SOTIF to establish safety requirements regarding faults and insufficient functionalities respectively. Section 4.3 then refines the elicited safety requirements based on the Apollo safety concept. The refined safety requirements are used for the assessment of functional safety in Chapter 5. In addition, the section also formulates an ideal safety concept for the A270 highway ODD. Finally, Section 4.4 summarizes and discusses the obtained FSRs regarding faults and insufficient functionality in the sensing and perception components.

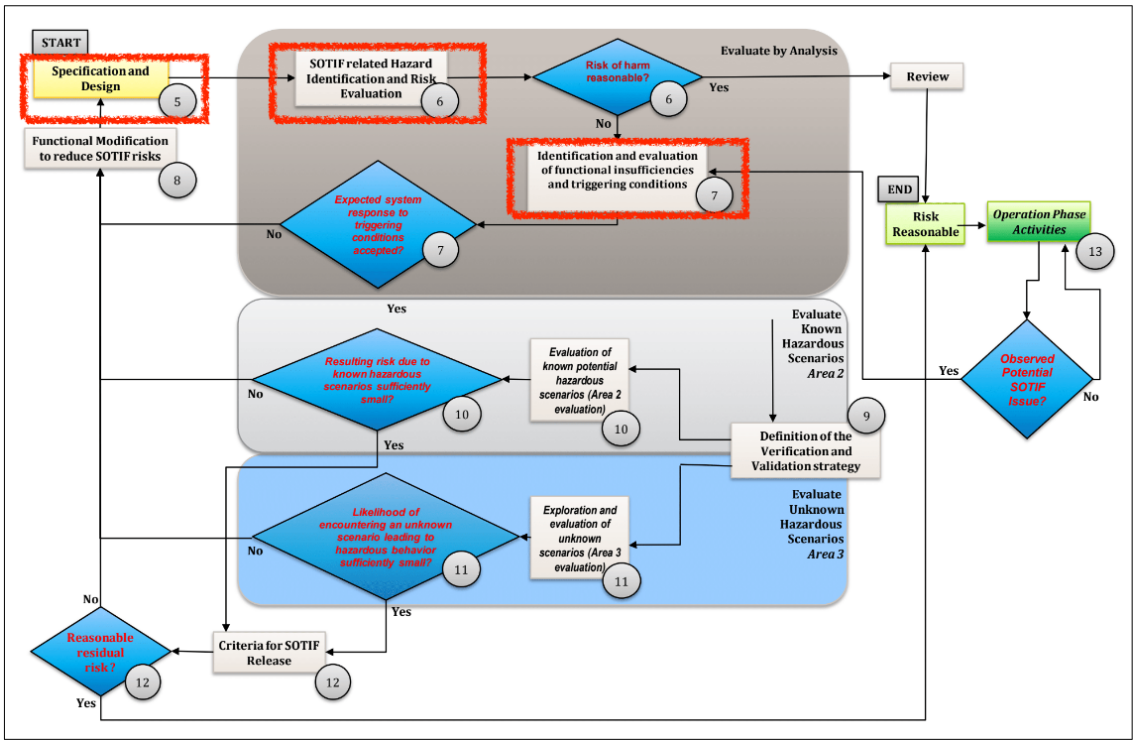

Figure 4.1: Safety processes as outlined in the DIS/ISO 21448 (SOTIF) standard [10]. The clauses relevant for elicitation of FSRs (clauses 5-7) are marked in red.

4.1 Hazard analysis and risk assessment

First, potential hazardous events are identified, i.e., combinations of vehicle-level hazards and operational scenarios which may lead to a harmful consequence. The risk posed by the hazardous events is then assessed, followed by the elicitation of safety goals to mitigate posed risks.

4.1.1 Identification of hazardous events

To identify hazards, the ISO 26262 standard [8] recommends the use of systematic techniques such as failure modes and effects analysis (FMEA) or hazard and operability study (HAZOP). FMEA is a bottom-up approach where failure modes of individual components of a system are analyzed to identify corresponding hazards for the system. A *failure mode* of a component describes a manner in which the component may fail and its resultant consequence. Instead, HAZOP explores deviations in the system functions (according to the functional concept) from their intended design to identify hazards. Due to the complexity of performing FMEA on the many components of the AV, we use HAZOP in this project.

HAZOP utilizes guide words to systematically identify deviations in a function that may result in a hazard. The most common guide words are *no, more, less, as well as, part of, reverse, other than, early, late, before, after* [63]. We observe that related works on AVs perform HAZOP for subsystem-level functions and tailor guide words to the subsystems. The EU ENSEMBLE truck research project on platooning [64] performs analysis on specific categories of functions (communication, acceleration, braking, and human machine interface). The analysis uses guide words *loss, unintended, lack* along with other words for incorrect behavior like *excessive* and *insufficient*. Bagschik et al. [65] also tailor guide words, e.g., ego vehicle planning related functions have guide words such as *physically not possible* and *irrelevant*. Unlike related studies, our functional concept

has functions defined at the vehicle-level, which are relatively abstract. Therefore, we do not customize the common guide words for the HAZOP analysis.

Only certain combinations of guide words and functions results in a hazard. For example, consider the function *avoid collision with an object in driving lane*. While the guide word *no* results in the hazard *does not avoid collision with an object in driving lane*, there is no corresponding hazard for the guide word *more*. We observe that only the guidewords *no, reverse, early* and *late* lead to hazards for the AV functional concept in Section 3.2.

Next, operational scenarios in which the hazards may lead to a potential risk are to be identified. The combination of a hazard, an AV operational mode, and an operational scenario is referred to as a *hazardous event* by the ISO 26262 standard. As the number of possible scenarios in our ODD is large, we refer again to the related work from the ENSEMBLE project [64] and by Bagschik et al. [65] to identify operational scenarios most relevant to the purpose of defining hazardous events.

Both ENSEMBLE and Bagschik et al. consider the following variables: (i) state of the AV, (ii) state of other road users, (iii) road infrastructure such as the slope of the road. ENSEMBLE also considers (iv) special events such as obstacles in the driving path, while Bagschik et al. also consider (v) weather conditions. Multiple variations within these variables may only be considered if they affect the risk posed by the hazardous event. As stated by Bagschik et al.: *"too-detailed scenes can distort the risk assessment (...) and result in a lower exposure rating"* [65].

Variables	**Coverage in hazardous events**			
Environmental conditions	All weather conditions	All illumination conditions		
Traffic infrastructure	Speed signs	Stopped bus region signs	Construction signs	Traffic lights
Road users and obstacles	Vehicles	Pedestrians	Cyclists	Obstacles (animals, debris)
Altered road due to construction or accident	Construction	Accident		
Designated points on route	On-off ramp	Merging point	Signaled intersection	
Vehicles				
Vehicle in-front of AV	driving	accelerate	decelerate	changes lane
Vehicle behind the AV	driving	accelerate	decelerate	changes lane
Vehicle in adjacent lane, ahead of AV	driving	accelerate	decelerate	changes lane
Vehicle in adjacent lane, behind AV	driving	accelerate	decelerate	changes lane
Vehicle merging into AV lane, at merging point	merging into lane			
Cross-traffic vehicle at intersection	Follows traffic light	Does not follow traffic light		

Figure 4.2: Operational scenarios covered in hazardous events. For each variable shown in blue, the variations in yellow are covered

We define operational scenarios using similar variables as presented in the related work. The state of other road users is not directly captured by the ODD. Currently, we focus only on other vehicles, as the primary road users on the highway. Pedestrians and cyclists, especially at intersections, and obstacles such as animals or debris, are to be considered in a future iteration of the safety analysis. The state of other vehicles on the highway may be described by a list of possible behaviors: *decelerate, accelerate, drive in lane, merge lanes, change lanes, and follow traffic lights*. We additionally consider a case of incorrect behavior by another road user; a cross-traffic vehicle at the intersection which does not follow traffic lights.

Then, behaviors relevant to the AV functional concept are extracted, e.g., *front vehicle decelerating* triggers a corresponding *avoid collisions* function. The operational scenarios covered within the performed safety analysis is summarized in Figure 4.2. Certain variables are captured within the AV operational modes and functions, e.g., *traffic lights* and *signaled intersection* in the function *follow traffic light* and operational mode *crossing an intersection*. Hence, we can express the operational scenarios instead through the remaining variables such as road users and obstacles.

All possible *hazardous events* are obtained through a combination of each hazard and operational scenario in an operational mode. The hazards of an operational mode are those which correspond to functions in that operational mode. The potential harmful consequence of the hazardous events is next determined to subsequently assess the risk posed by the event. We filter out invalid hazardous events, for example, the combination of hazard *avoid collision with an object in driving lane* with operational scenario *vehicle approaching from behind in goal lane.* Assumptions are made for when a hazardous event may lead to a harmful consequence. For example, emergency braking at highway speeds (speed limit 100 km/h) is assumed harmful due to limited reaction times for a following vehicle. However, emergency braking at an intersection (speed limit 50km/h) is assumed safe with a following vehicle having sufficient reaction time.

4.1.2 Risk assessment and safety goals

The risk posed by the hazardous event and the resultant consequence is assessed. An automotive safety integrity level (ASIL) of quality management (QM) or ASILs A-D is assigned to each hazardous event. Hazardous events assigned QM pose tolerable risk, while those assigned ASIL D pose high risk. Three factors determine the ASIL level; the *controllability* (C), *severity* (S), and *exposure* (E) of the hazardous event. The controllability level estimates the ability of a driver or vehicle user to mitigate the consequence of a hazardous event. The level ranges from *controllable in general* (C0) to *incontrollable (C3).* At L4 automation, the AV can not rely on driver intervention and all hazardous events are level C3.

Operational mode	**Exposure**	**Operational scenario**	**Exposure**	**Combined exposure**
driving in lane	E4	decelerating vehicle in front	E4	E4
driving in lane	E4	vehicle from side lane which cuts in front	E4	E4
driving in lane	E4	closed lane due to construction	E2	E2
changing lanes	E3	decelerating vehicle in front	E4	E3
changing lanes	E3	vehicle approaching from behind in goal lane	E4	E3
crossing a merging point	E3	decelerating vehicle in front	E4	E3
crossing a merging point	E3	vehicle merging into lane	E4	E3
crossing an intersection	E3	decelerating vehicle in front	E4	E3
crossing an intersection	E3	cross-traffic vehicle which follows traffic light	E4	E3
crossing an intersection	E3	cross-traffic vehicle which skips traffic light	E2	E2

Table 4.1: Exposure level of hazardous events

The severity level is a measure of potential harm, ranging from *no injuries* (S0) to *fatal injuries* (S3). We estimate potential harm of a collision based on speed limits, assigning S2 to the intersection (50 km/h) and S3 to other locations on the highway (100 km/h). Thus, hazardous events which result in a collision have severity S2 if they correspond to the operational mode *crossing an intersection* and severity S3 if they correspond to the other operational modes. S0 is assigned when the hazardous event does not lead to a collision.

As per ISO 26262, exposure level may be estimated either by duration or frequency, ranging from *incredibly low* (E0) to *high probability* (E4). The exposure to an operational scenario is estimated by its frequency of occurrence, based on examples in the concept phase of ISO 26262 [66]. For example, a vehicle decelerating in front of the AV may happen *multiple times during a single*

trip (E4), but a lane closed due to construction may only be encountered *up to once or twice a year* (E2). Although not accounted for explicitly in the ISO 26262 standard, the exposure to a hazardous event should also depend on the exposure of the corresponding operational mode. The exposure to an operational mode can be determined by the proportion of time spent in that mode. The final exposure level is taken as the minimum of the operational mode and operational scenario exposure levels, as a low exposure to either operational mode or operational scenario results in a overall low exposure level. The exposure levels are summarized in Table 4.1.

The final step in HARA is to specify safety goals to mitigate all hazardous events of ASIL A or higher. A safety goal is described as a functional objective for the AV. Once a safety goals is specified for each hazardous event, similar safety goals may be aggregated. Table 4.2 presents the 18 aggregated safety goals from the 69 safety goals originally formulated. The safety goals corresponding to the operational mode *crossing an intersection* have the lowest ASIL levels. This is inherited from the lower severity level assigned to the operational mode compared to other operational modes. SG5, SG6 and SG18 are QM as the corresponding hazardous events do not lead to a harmful consequence. Each hazardous event causes emergency braking at an intersection, which as described earlier in 4.1.1, is assumed safe at the lower intersection driving speeds.

The complete HARA analysis is documented in `HARA.xlsx` [2].

[2] `https://surfdrive.surf.nl/files/index.php/s/2RwiL5taL9nTTFj`

No.	Safety goal: The AV shall...	ASIL
SG1	avoid collision with an object (obstacle or vehicle) in driving lane in all operational modes	D
SG2	prevent unintentional activation of avoid collision with an object (obstacle or vehicle) in driving lane in all operational modes	D
SG3	avoid collision with an object (obstacle or vehicle) in driving lane without emergency braking in all operational modes	D
SG4	avoid collision in scenario: cross-traffic vehicle which skips traffic light in operational mode: crossing an intersection	A
SG5	prevent unintentional activation of avoid collision with an object (obstacle or vehicle) at intersection in operational mode: crossing an intersection	QM
SG6	avoid collision in scenario: cross-traffic vehicle which skips traffic light without emergency braking in operational mode: crossing an intersection	QM
SG7	avoid collision in scenario: vehicle merging into lane in operational mode: crossing a merging point	C
SG8	prevent unintentional activation of avoid collision with an object (obstacle or vehicle) at a merging point in operational mode: vehicle merging into lane	C
SG9	avoid collision in scenario: vehicle merging into lane without emergency braking in operational mode: crossing a merging point	C
SG10	avoid collision in scenario: vehicle approaching from behind in goal lane without emergency braking in operational mode: changing lanes	C
SG11	avoid collision in scenario: vehicle approaching from behind in goal lane without emergency steering in operational mode: changing lanes	C
SG12	drive within speed limit in all operational modes	D
SG13	follow correct lane in all operational modes	D
SG14	maintain driving lane in all operational modes except operational mode: changing lanes	D
SG15	indicate lane change before performing lane change maneuver in scenario: vehicle approaching from behind in goal lane in operational mode: changing lanes	C
SG16	give priority during lane change in scenario: vehicle approaching from behind in goal lane in operational mode: changing lanes	C
SG17	follow red and yellow traffic light in operational mode: crossing an intersection	B
SG18	follow red and yellow traffic light without emergency braking in operational mode: crossing an intersection	QM

Table 4.2: List of aggregated safety goals

4.2 Derivation of functional safety requirements

This section first performs the fault-tree analysis (FTA) to map violations of safety goals at the AV-level (as obtained in Section 4.1) to failure events in the sensing and perception components. FSRs are then derived for the failure events by steps specific to the ISO 26262 and SOTIF standards. Note that throughout this thesis, we use the terms 26262 FSRs and SOTIF FSRs to refer to FSRs w.r.t faults and functional insufficiencies respectively.

FTA is a top-down failure analysis, which determines how component-level failures lead to system-level events. This mapping requires the functional allocation in Section 3.4. As an example, Figure 4.3 presents the FTA for the violation of safety goal *"The AV shall follow red and yellow traffic lights in operational mode: crossing an intersection"*. The semantics of the FTA include two types of events and two boolean gates. A basic event, depicted by a circle, is a primary failure, whereas a secondary event, represented by a rectangle, is a failure consequent of basic events. In our FTA, we focus only on basic events related to the sensing and perception components.

4qopieuriopu2/2n2nn2n2n2nn2nn2/ 2nn2n2n2n2n/2n/ ñ/ nb2 b/ b/ 2nn /b

The boolean "AND" gate signifies that *all input failures* must occur to cause the output failure. Instead, the "OR" gate indicates that *a single input failure* results in the output failure. The boolean gates are primarily determined via the functional architecture. Instances where the boolean gates required other sources are documented in FTA.pdf [2], together with the FTA for the complete set of safety goals. Eleven basic failure events are identified through the FTA, presented in Table 4.3. Each event corresponds to a unique sensing and perception functional component.

ID	**Event**	**Violated safety goals**
E1	Traffic light detection and recognition fails to detect or recognize color of yellow or red traffic light	SG17,SG18
E2	Object detection classification and tracking (lidar) estimates incorrect state of vehicle / obstacle	SG1-SG11,SG16
E3	Object detection classification and tracking (radar) estimates incorrect state of vehicle / obstacle	SG1-SG11,SG16
E4	Object detection classification and tracking (camera) estimates incorrect state of vehicle / obstacle	SG1-SG11,SG16
E5	Lane detection and tracking estimates incorrect lane information	SG1-SG11,SG16
E6	Sensor fusion does not provide correct combined state information of vehicle / obstacle	SG1-SG11,SG16
E7	Sense ego pose does not provide correct sensing of ego pose	all
E8	Sense environment (lidar) does not provide correct sensing of environment	all
E9	Sense environment (radar) does not provide correct sensing of environment	SG1-SG11,SG16
E10	Sense environment (camera) does not provide correct sensing of environment	SG1-SG11 ,SG16,SG17,SG18
E11	HD map does not provide correct map information	all

Table 4.3: The eleven basic events identified in our FTA

[2] https://surfdrive.surf.nl/files/index.php/s/2RwiL5taL9nTTFj

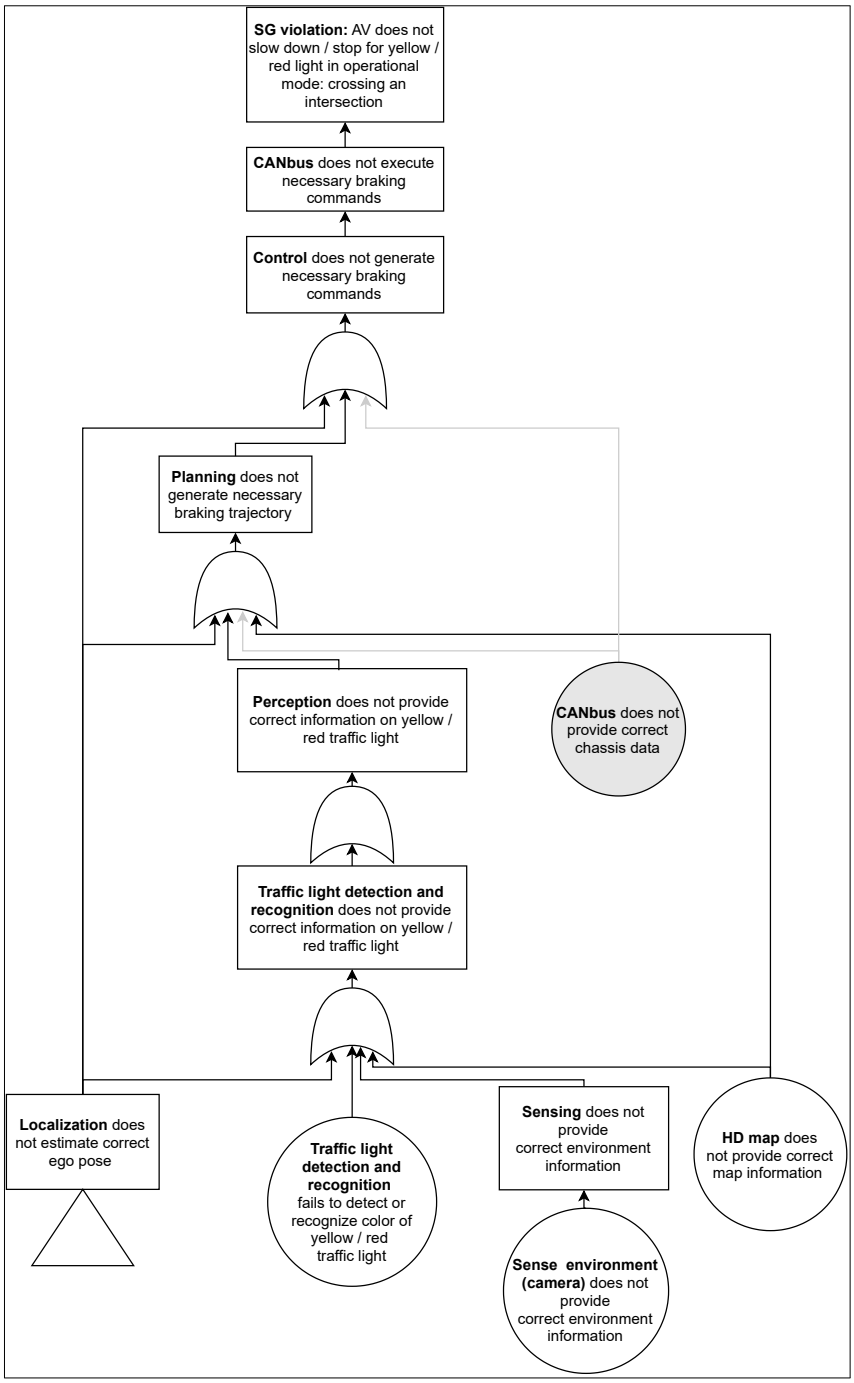

Figure 4.3: FTA for violation of safety goal SG17, "*follow red and yellow traffic light in2nn2nn2n2nnn/ operational mode: crossing an intersection2/ n*". The events not related to sensing and p22n2n2n2n2n2n/ ///n2n/ erception components (marked in gray) are not 2n/ considered in this analysis. The triangle belo¿ /////// 2n22n2n2n/2n2n2n2n/n2n2/ w the localization event indicates the FTA i2n2n /2bnn2n2n2n2n2n2nnn2n/ s continued further in another diagram. n22n2n2n22n/ //n2nn/

4.2.1 ISO 26262 FSRs

The functional safety concept step in the concept phase of ISO 26262 derives FSRs to prevent the violation of the vehicle-level safety goals. With respect to ISO 26262, a basic event corresponding to a functional component may be due to the (i) *component becoming non-operational* or (ii) *component output being lost or corrupted*. FSRs are specified to detect both failures for all 11 basic events and prevent a corresponding violation of safety goals. An example FSR is, *Non-operational state of the traffic light detection and recognition component shall not lead the AV to ignore red or yellow traffic light in operational mode: crossing an intersection.*

FSRs are also allocated to specific components in the functional safety concept step. The allocation is usually an input for part 4 of ISO 26262, which deals with product development, based on the FSRs. In our case, we aim to *assess functional safety of an existing AV*. Thus, we do not assume a FSR is fulfilled by a specific subsystem level component and instead allocate the FSRs to the sensing and perception subsystem as a whole. In addition, the FSRs may also be fulfilled by the safety components in Apollo. Thus, the allocation of FSRs is additionally extended to the safety components in Apollo.

4.2.2 SOTIF FSRs

Here, we describe the derivation of SOTIF FSRs related to the failures identified in the sensing and perception components through the FTA analysis. The relevant section of the SOTIF standard is clause 7: *the identification and evaluation of potential functional insufficiencies and triggering conditions* [10]. In our study, we consider the functionality of *camera based object detection, classification, and tracking* (camera object perception) as a proof-of-concept. The functionality is composed of three functional components within the detailed sensing and perception architecture (shown in Figure 3.4, Section 3.3). The components are: (1) Sense environment (camera), (2) Lane detection and tracking, and (3) Object detection, classification, and tracking (camera).

The term *functional insufficiency* is defined as *a specification of the functionality or a limitation in technical capability which may lead to a hazardous behavior in combination with triggering conditions* [10]. Triggering conditions can be considered as *a-priori worst case conditions within an ODD*; conditions which may invoke a functional insufficiency and lead to potential hazardous behavior.

The SOTIF analysis is performed for functional insufficiencies related to illumination (time-of-day) and weather conditions, using examples in the SOTIF standard for reference. According to the standard, functional insufficiencies of these categories are key for perception-related functionalities. The functional insufficiencies identified to deteriorate performance of *object detection* are low-illumination conditions [62, 67], illumination conditions rarely captured in training data sets such as dusk and dawn [68] and weather conditions, in particular fog [69], rain [70, 71], and snow [72]. The deteriorated performance of object detection further deteriorates performance of *object tracking* [73], but its effect on *object classification* is not found in literature. Thus, the functional insufficiency is specified w.r.t the performance of object detection and object tracking for the camera object perception functionality.

Next, *triggering conditions* are identified for each functional insufficiency. The triggering conditions are represented by three entities; (i) the scene, (ii) the AV behavior, and (iii) the behavior of other road users. The scene characteristic directly relevant to the functional insufficiency, i.e., a weather or illumination condition, is explicitly defined. The illumination conditions corresponding to all times-of-day are part of the ODD (see `Description_ODD.pdf` [2]). The weather conditions include: (ii) up to moderate inclement levels of wind, rain, snow, sleet and fog; (iii) a temperature range of -15 degrees celsius to 40 degrees celsius; and (iv) standing water on the road. We make no assumptions on worst-case variations of other variables are made. Thus, all possible variations

[2] `https://surfdrive.surf.nl/files/index.php/s/2RwiL5taL9nTTFj`

within the ODD of variables other than the scene characteristic directly relevant to the functional insufficiency is considered as part of the triggering conditions.

The vehicle-level effects of the functional insufficiency, the severity rating of the vehicle-level effects (S*), and the occurrence rating (O*) of the associated triggering conditions are to be specified as part of the analysis. The examples in the SOTIF standard present the same criteria for S*, as the severity levels used for risk assessment in HARA: ranging from *no injury* (S*=0) to *fatal injuries* (S*=3). The O* rating ranges from *improbable* (O*=0) to *once or more per driving cycle* (O*=6).

Based on the FTA, the failure of the camera based object detection, classification, and tracking functionality violates safety goals SG1-SG11 related to *avoiding collisions* and SG16 on *giving priority during lane change* from Table 4.2. The two ratings are determined as S*=3 (highest) and O*=5 (second highest) for all functional insufficiencies. The severity criteria in SOTIF are the same as in HARA and S* rating is inherited from the severity of the violated safety goals SG1-11 and SG16. The occurrence criteria is based on frequency of the triggering conditions with O*=5 corresponding to *occur once or more per year*. The highest O* rating of 6 corresponds to *occur each trip* which is not expected for the triggering conditions.

Finally, SOTIF FSRs are specified to prevent violation of safety goals due to the potential functional insufficiency and its associated triggering conditions. An example SOTIF requirement for SG1 (avoid collisions with an object in driving lane) in Table 4.2 is "*deteriorated performance of object detection and tracking at night shall not lead to a collision with an object in driving lane*". SOTIF FSRs are allocated to the sensing and perception subsystem and the safety components in Apollo.

4.3 Safety concept

This section outlines the refining of requirements defined in Section 4.2 for the Apollo safety concept. The Apollo safety concept has vehicle-level safe states which are triggered upon safety-related issues in the AV. We begin this section by discussing appropriate safe states for the A270 Highway ODD. Subsequently, Section 4.3.1 present an ideal safety concept for the A270 highway ODD. Then, Section 4.3.2 discusses the safe states implemented in Apollo, for which we refine our elicited FSRs.

A trade-off exists between the safety or availability provided by a safe state, and the ability of the AV to achieve the state. Possible fail-safe states on the highway include an emergency stop in lane, slow stop in lane, or parking on the roadside. An emergency stop can cause a collision with a following vehicle at highway speeds. A stationery vehicle on an active highway lane can also lead to accidents especially in low-visibility scenarios [18]. However, achieving these safe states requires limited AV ability. Parking on the roadside is a safer final state but requires additional functionality.

Similarly, fail-operational states (degraded mode) offer higher availability of the AV but require additional functionality. Possible fail-operational states include reduced driving speed or automation level, or a restriction of certain operating modes or functionality. A reduction in automation level is not appropriate to our L4 AV, as we do not assume driver availability. Driving at a reduced speed on the highway is also dangerous [74], [75]. However, the AV may drive at a reduced speed on the road shoulder till a highway exit or parking spot [1]. Restricting the operational mode to the basic *driving in lane* operational mode may also be suitable. This may be appropriate if a failure does not impact operation of that mode.

[1] Driving on the road shoulder is permissible for emergencies under Dutch traffic rules [55]

4.3.1 Theoretical safety concept

The trade-off in the choice of a safe state is important to consider upon a failure in the sensing and perception components. Figure 4.4 presents a state machine of our safety concept for the highway ODD related to failure events in sensing and perception components. The safety concept is developed considering the functional architecture of Apollo including the *ultrasonic sensor* safety component which perceives near-distance objects (as introduced in Section 3.3.

The reasoning for a transition to specific safe states for our safety concept is summarized in Table 4.4. The safety concept distinguishes between the failure of a component which has a redundancy or diverse redundancy in the functional architecture (assigned safe state S3 or S4), and a component that does not have such redundancy (assigned safe state S2). The choice of whether the vehicle should park at the roadside (safe state S3) or continue operating on the road shoulder (safe state S4) is made based on the affected functionality.

The AV localization functionality has two diverse redundant inputs, while object perception (object detection, classification and tracking) has three diverse redundant inputs and corresponding components Figure 3.4. For a failure event that affects an input related to localization, safe state S3 is chosen. Safe state S4 is considered dangerous; the failure event may lead to inaccurate localization and cause a hazardous behavior such as steering back into an active lane. In the case of a failure event that affects one input or component related to object perception, safe state S4 is considered safe as (i) two diverse inputs or components of object perception are still operational and (ii) the complexity of the object perception task is lower on the road shoulder.

The safety concept also chooses an appropriate safe state based on the cause of a failure event, i.e., a fault or a functional insufficiency. The *Sensor fusion* functional component and the diverse redundancy in the object perception functionality (see Figure 3.4) are designed at mitigating vehicle-level effects due to functional insufficiencies of individual modalities (e.g., camera-based object perception) for the ODD conditions. This is noted by the state-of-practice [5, 9]. Thus, the satisfaction of SOTIF FSRs for triggering events within the ODD may not necessarily require a safe state transition, and the requirements do not need further refinement.

Faulty components as covered in ISO 26262 FSRs indicate an unexpected failure, which can in turn reduce the diverse redundancy in the system. Thus, a vehicle-level safe state is appropriate for the ISO 26262 FSRs. In addition, SOTIF FSRs which relate to triggering conditions also require a vehicle-level safe state. Conditions outside of the ODD are by definition unsafe. Thus, 26262 FSRs and SOTIF FSRs related to triggering conditions outside of the ODD are to be refined further to specify a transition to the appropriate safe state for the corresponding failure event.

ID	**safe state**	**choice of safe state**
S1	emergency stop in lane	other safe states are not possible
S2	slow stop in lane	no redundancy in architecture for failed component or multiple failed components
S3	park on the roadside	if failed component leads to a failure of the localization component
S4	continue driving at lower speed on road shoulder till parking spot or highway exit	if failed component leads to a failure of an object detection component
S5	continue driving in lane till destination or parking spot or highway exit	not suitable as all events affect the driving in lane operational mode.

Table 4.4: Safe states for the A270 Highway ODD

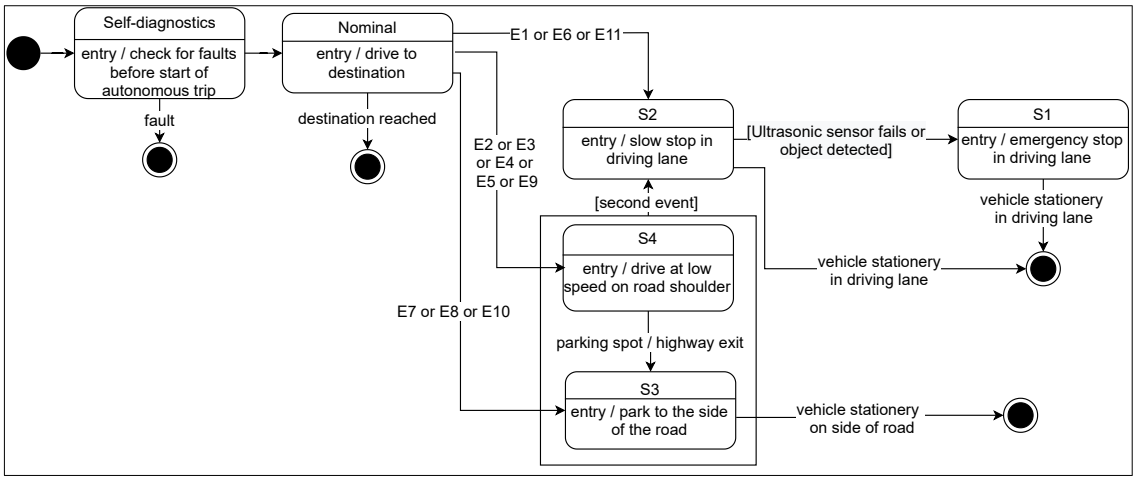

Figure 4.4: State machine of safety concept. The events E1-E12 refer to failure events in Table 4.3. The safe states S1-S4 refer to safe states in Table 4.4.

4.3.2 Apollo safety concept and refinement of safety requirements

The Apollo safety mechanisms, i.e., the monitor, guardian, and ultrasonic sensor are introduced in Section 3.3. The safety response in Apollo constitutes only two of the aforementioned safe states; S1, *an emergency stop in lane,* and S2, *a slow stop in lane.* The choice between S1 and S2 is based on the integrity of the ultrasonic sensing component and the distance to nearby objects [2]. A transition to the safe state S3, *park on the roadside* is also part of the planning component functionality [3]. However, triggering of S3 in response to a failure may not be implemented in Apollo [4].

To obtain testable safety requirements for our assessment of functional safety of Apollo (detailed in Chapter 5), the safety requirements from Section 4.2 are refined based on the Apollo safety concept. Note that we also draw on the discussion in Section 4.3.1 to distinguish which requirements need further refinement. Thus, a transition to a safe state is specified in the safety requirements for the ISO 26262 FSRs as well as SOTIF FSRs related to triggering conditions outside of the ODD. For SOTIF FSRs corresponding to within ODD conditions, no further refinement is necessary at this stage.

Note that unlike our ideal safety concept, a mapping between failure events and safe states is not known for the Apollo safety concept. Therefore, specific safe states are not specified in the safety requirements.

4.4 Results and discussion

The process followed in this chapter for the elicitation of safety requirements for the A270 highway ODD, and the functional architecture of Apollo, is summarized in Figure 4.5. 18 aggregated safety goals are identified from the 69 safety goals originally formulated, out of which 15 are ASIL A or higher (presented in Table 4.2). As discussed earlier in Section 4.1.2, the safety goals corresponding to the operational mode *crossing an intersection* have the lowest ASIL levels. Through the FTA analysis, we identified eleven failures events (shown in Table 4.3), each corresponding to a unique

[2] https://github.com/ApolloAuto/apollo/blob/master/docs/specs/Apollo_5.5_Software_Architecture.md#guardian
[3] https://github.com/ApolloAuto/apollo/tree/v5.5.0/modules/planning#emergency
[4] We have only found evidence of manual triggering of the safe state by a remote user, as in https://github.com/ApolloAuto/apollo/blob/master/modules/dreamview/backend/teleop/README.md

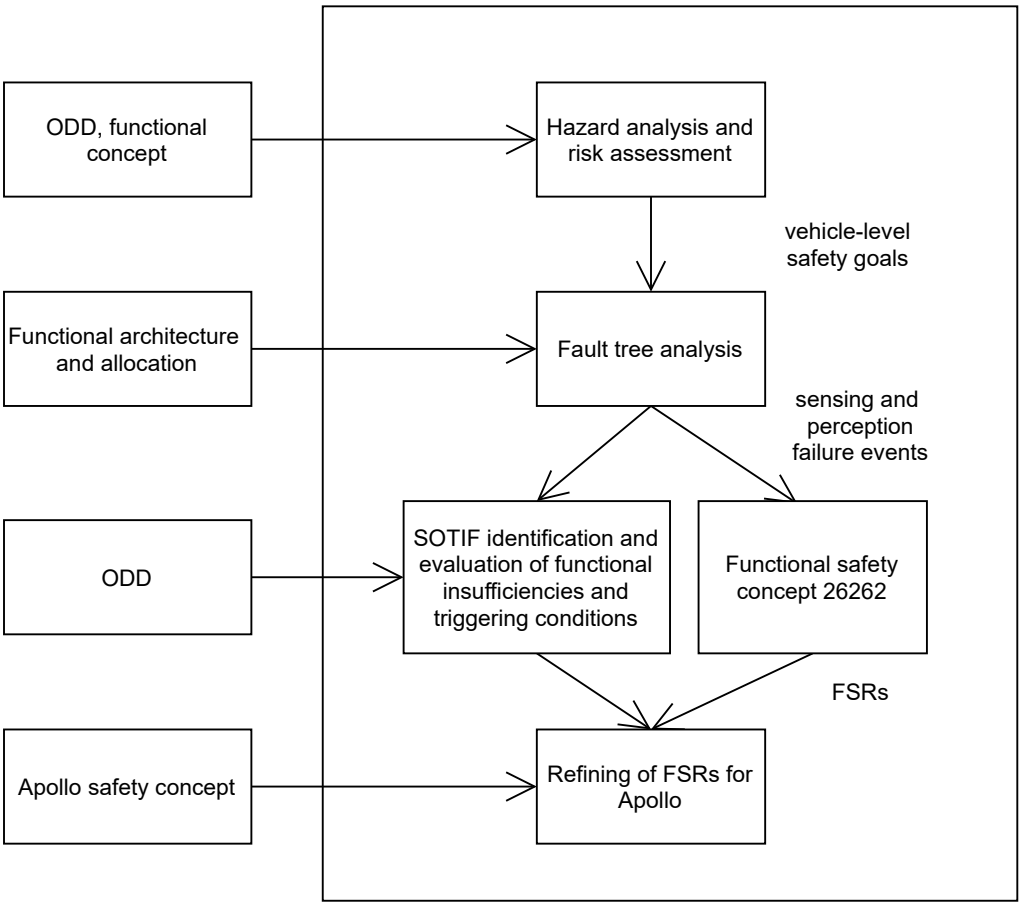

Figure 4.5: Elicitation process for functional safety requirements

sensing and perception functional component. The failure event of functional components and their resultant violation of safety goals is also identified through the FTA. For example, the failure of the camera based object detection, classification, and tracking (object perception) functionality violates safety goals SG1-SG11 related to *avoiding collisions* and SG16 on *giving priority during lane change*.

In the SOTIF and ISO 26262 specific steps, we have elicited safety requirements for the failure events of the sensing and perception components. The ISO 26262 covers two failure types; (i) a non-operational state and (ii) output data corruption or loss. In the SOTIF analysis, we identified the functional insufficiencies related to the camera based object perception functionality. The ratings of the severity of vehicle level effects (S*) of the functional insufficiencies, and the occurrence of the associated triggering conditions (O*) were determined as S*=3 (highest) and O*=5 (second highest). Thus, the identified functional insufficiencies are likely to both appear in the ODD as well as lead to hazardous behavior.

We then present an ideal safety concept for the A270 highway ODD, as well as the Apollo safety concept. The ideal safety concept considers additional safe states compared to the Apollo safety concept. An example of an additional safe state is S3, *parking on the road shoulder*. This may be a safer final state for a vehicle on a highway as discussed in Section 4.3. This points to a potential limitation in Apollo's safety concept for operation in a highway setting.

As the final step of FSR elicitation, the FSRs are refined for the Apollo safety concept. The requirements are documented in FunctionalSafetyRequirements.xlsx [2]. Example safety requirements are illustrated in Table 4.5 for SG1: *avoiding collisions with an object in driving lane* and a failure event of the *camera based object detection, classification, and tracking functionality.* The table presents an example 26262 FSR, SOTIF requirement for triggering conditions within the ODD, and a SOTIF requirement for triggering conditions outside of the ODD corresponding to the failure event.

Safety goal	**Event**	**Example 26262 FSR**	**Example SOTIF requirement (within ODD conditions)**	**Example SOTIF requirement (outside of ODD conditions)**
SG1	Object detection, classification and tracking (camera) estimates incorrect state of vehicle / obstacle (E4)	If the *Object detection, classification and tracking (camera)* functional component becomes non-operational, the vehicle shall transition to a safe state	deteriorated performance of object detection and tracking at night shall not lead to a collision with an object in driving lane	If the performance of object detection and tracking deteriorates due to heavy rainfall, the vehicle shall transition to a safe state

Table 4.5: Example of refined safety requirements for the Apollo safety concept

[2] https://surfdrive.surf.nl/files/index.php/s/2RwiL5taL9nTTFj

Chapter 5

Assessment of functional safety

This chapter presents methods and results for assessment of functional safety in Apollo. The assessment of functional safety is performed against the elicited functional safety requirements (FSRs) (detailed in Chapter 4) related to faults and insufficient functionality of the AI-based sensing and perception subsystem. The fulfillment of FSRs is assessed at both the design level and implementation of Apollo. We focus the application of functional safety assessment methods on FSRs related to two AI-based functionalities; the lidar-based and camera-based object detection, classification, and tracking functionalities. The relevant components are shown in Figure 5.1.

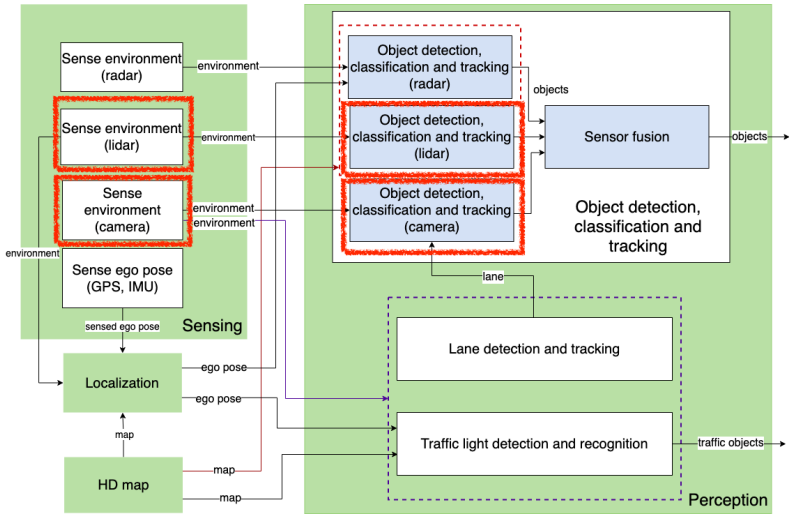

Figure 5.1: The detailed functional architecture of the sensing and perception subsystem of Apollo. We focus the application of functional safety assessment methods on FSRs related to the components marked in red.

First, Section 5.1 uses an existing approach by Kochanthara et al. [76, 77] to assess fulfillment of FSRs in the vehicle's software architecture. As the approach does not take into account FSRs related to AI-based components, we extend the approach further to consider AI-safety related best practices and AI related design artefacts.

Then, Section 5.2 outlines a simulation-based testing approach to assess FSRs in the Apollo implementation. Firstly, Section 5.2.1 describes a systematic approach to derive test cases for the elicited FSRs. Section 5.2.2 then demonstrates the testing of a few example test cases.

Finally, Section 5.3 summarizes the chapter.

5.1 Design assessment

To assess the fulfillment of FSRs at the design level, we use the methodology proposed by Kochanthara et al. [77, 76]. The methodology proposes a design-level assessment of functional safety by checking the application of safety tactics in the vehicle software architecture. An architectural tactic is an abstract design decision that influences non-functional attributes of a system [78]. A safety tactic is an architectural tactic which addresses safety.

The fulfillment of FSRs is systematically assessed in two steps. First, (combinations of) safety tactics which can fulfill each FSR are identified by matching the FSR description to the tactic description. Then, the vehicle software architecture is checked for the utilization of safety tactics within the identified set. In their case study, the authors relied on a set of 13 safety tactics that are most widely employed in design [79, 80] to assess ISO 26262 FSRs for a cooperative autonomous vehicle. The 13 safety tactics are: *heartbeat, simplicity, substitution, sanity check, comparison, replication redundancy, diverse redundancy, condition monitoring, repair, voting, degradation, override* and *barrier*.

Here, we apply the methodology proposed by Kochanthara et al. to our context which includes SOTIF FSRs in addition to ISO 26262 FSRs. Our safety requirements are also targeted at AI-based components who have other artefacts (dataset, neural network (NN) model) besides the functional and technical architecture of the vehicle. In addition, the safety challenges of AI-based components are unique; for example, unpredictable behavior in novel or rare driving conditions [13, 14]. Thus, we explore possible extensions to the set of 13 safety tactics which address the safety of AI-based components. As a preliminary literature search on safety tactics for AI-based components did not result in any findings, we broadened our search to best practices for software engineering of AI-based components.

From the retrieved papers, we extract best practices addressing the safety of AI-based components and further filter them for their relevance to our context. For example, the safety related best practice of *human oversight of AI-based components* is not applicable for our L4 AV. Safety best practices that are relevant in the development of the AV are also out of scope as such information is not provided by Apollo, e.g., practices for training of the neural network. The list of extracted best practices for safety of AI-based components is shown in Table 5.1.

We use both the 13 safety tactics and the 11 best practices derived for safety of AI components to assess our FSRs. The tactics and best practices are matched to FSRs based on their description and aim. For example, the best practice of *uncertainty estimation and monitoring* has the aim: "recognize degradation of model or erroneous outputs". This best practice applies to the elicited SOTIF FSRs, as it enables detection of deteriorated performance of the object detection, classification, and tracking functionality.

As a case study, we assess the employment of best practices and tactics for FSRs related to the camera-based object detection, classification, and tracking functionality for which the proof-of-concept SOTIF analysis has been performed (detailed in Section 4.2.2). The use of safety tactics and AI-safety best practices is checked in the functional architecture, Apollo documentation and Apollo code. We limit the code review to certain components relevant to the target functionality: (i) camera object pipeline, (ii) sensor fusion, and the (ii) two safety components; monitor and guardian. Additionally, some AI-safety best practices are to be assessed in relation to the training dataset, Waymo Open dataset [85], and the NN model [86], SMOKE NN model. We base the assessment criteria for practices related to the dataset or NN model on the documentation of these design entities. We illustrate the assessment criteria for best practices with two examples in Table 5.2.

Several artefacts related to AI-safety best practices are presented in `AI_bestPractices.xlsx` [1]: (i) complete process of extracting and filtering safety-related best practices for AI components from literature, (ii) complete list of assessment criteria to check if the best practices are employed, and (iii) elaborate examples to illustrate the process of identifying relevant best practices for FSRs.

ID	Best practice	Description	Relevant AV entity
1	check that input data is complete, balanced and well Distributed [81]	(our interpretation) dataset used for the NN development covers operating scenarios in the ODD	dataset
2	design specification for the NN [82]	specification of the NN (example techniques: formal specification or design for DNNs, break ML functionality into smaller algorithms in hierarchical structures)	NN model
3	testing and verification for the NN [82]	verifying NN correctness (example techniques: neuron coverage testing, fuzz testing)	NN model
4	uncertainty estimation and monitoring for the NN [82], [83]	estimate and monitor uncertainty including epistemic uncertainty (model confidence on its prediction) and aleatoric uncertainty (uncertainty for unknown samples). Techniques mentioned are deep ensembles and Monte Carlo Dropout	NN model, SW architecture
5	in-distribution error detectors [82]	detect misclassification of in-domain samples (example techniques: selective classification, prediction of failures)	NN model, SW architecture
6	out-of-distribution error detectors [82]	detect inputs outside of normal training distribution (example techniques: reject option in output, measure of class probabilities to detect abnormal samples)	NN model, SW architecture
7	domain generalization [82]	improve difference between real-world performance vs. training data	dataset, NN model
8	robustness to corruption, perturbations [82]	robustness to natural corruptions e.g., weather conditions, and artificial perturbations, e.g., sensor noise	dataset, NN model
9	Use n-versioning [83]	use of ensembles of ML models or back-up with an interpretable or rule based model	SW architecture
10	Use metric monitoring and alerts to detect failure [83]	continuous monitoring of metrics e.g., decrease in NN model accuracy. Subsequent alerting / warning	SW architecture
11	monitor data quality issues [84]	(our interpretation) detect sensor raw data issues such as data corruption	NN model, SW architecture

Table 5.1: Best practices for safety of AI-based components

Similarly for the safety tactics, we document in `Tactics.xlsx` [1]: (i) complete list of assessment criteria if the safety tactics are employed, and (ii) elaborate examples to illustrate the process of identifying relevant safety tactics for FSRs.

ID	Best practice	Description	Assessment criteria: best practice employed in design
7	robustness to corruption, perturbations [82]	robustness to natural corruptions e.g., weather conditions, and artificial perturbations, e.g., sensor noise	The NN model paper / Apollo documentation discusses robustness of model to natural corruptions and perturbations
4	uncertainty estimation and monitoring	estimate and monitor uncertainty including epistemic uncertainty (model confidence on its prediction) and aleatoric uncertainty (uncertainty for unknown samples). Techniques mentioned are deep ensembles and Monte Carlo Dropout	(i) The NN model paper discusses uncertainty estimation for the NN (ii) The uncertainty estimation may rely on one of the described techniques or refer to other state-of-the-art work (iii) The Apollo documentation / architecture / code around the model shows evidence of monitoring NN uncertainty

Table 5.2: Assessment criteria to check if best practices for safety of AI-based components are employed in design artefacts

5.1.1 Results and discussion

The results of the design-level assessment for FSRs related to the camera-based object detection, classification, and tracking functionality are summarized in Table 5.3. Our results show (i) AI-safety related best practices are mainly relevant for SOTIF FSRs and not 26262 FSRs, (ii) AI-safety related best practices are lacking in the Apollo design, (iii) several safety tactics are relevant to FSRs and SOTIF FSRs (iv) some of the relevant safety tactics are employed in the design. The complete results are documented in `Tactics.xlsx` and `AI_bestPractices.xlsx` [1].

We first elaborate on the assessment results (relevance to FSRs, employment in design) for best practices specific to AI-safety. In terms of relevance to the FSRs, only the best practice of monitoring data quality issues (practice 11) is assessed relevant to ISO 26262 FSRs (FSR 26262_19, FSR 26262_20). In contrast, for SOTIF FSRs, most AI-safety related best practices are determined relevant, except the practices of design specification of NN (practice 2), domain generalization (practice 6), and monitor data quality issues (practice 11). Some of the relevant best practices decrease the corresponding functional insufficiency addressed by the FSR e.g., input data characteristics (practice 1). Others prevent the functional insufficiency from leading to a safety goal violation, as specified in the requirement e.g., uncertainty estimation (practice 8).

The employment in design for these best practices is found to be low. Only practice 11 and 9 are found employed in the Apollo design. Practice 11, relevant to FSR 26262_19, FSR 26262_20, is employed in the form of data integrity checks performed by the *monitor* component. Practice 9, relevant to the SOTIF requirements, is employed by the use of diverse redundancy (lidar, radar, camera based) for the object detection, classification, and tracking functionality. The other relevant AI-safety related practices are assessed lacking in the Apollo design.

Next, we discuss the assessment results (relevance to FSRs, employment in design) for the safety tactics. In terms of relevance to FSRs, several relevant safety tactics such as *override, diverse redundancy, and sanity check* are identified for both 26262 FSRs and SOTIF FSRs. Some of these tactics are observed to be employed in the Apollo design. The override tactic, which brings

[1] `https://surfdrive.surf.nl/files/index.php/s/2RwiL5taL9nTTFj`

FSR ID	FSR	Best practices employed in design	Safety tactics employed in design
26262_7	If object detection, classification and tracking (camera) component becomes non-operational, the vehicle shall transition to a safe state	n/a (no practices are applicable)	override (safe state triggered based on checks by monitor component)
26262_8	If object detection, classification and tracking (camera) component output data is corrupted or lost, the vehicle shall transition to a safe state	n/a (no practices are applicable)	diverse redundancy (lidar, radar, camera based), replication redundancy (2 partially overlapping cameras)
26262_18	If sense environment (camera) component becomes non-operational, the vehicle shall transition to a safe state	monitor data quality issues (checks by monitor component)	override (safe state triggered based on checks by monitor component)
26262_19	If sense environment (camera) component output data is corrupted or lost, the vehicle shall transition to a safe state	monitor data quality issues (data integrity checks by monitor component)	sanity check (data integrity checks by monitor component), diverse redundancy (lidar, radar, camera based), replication redundancy (2 partially overlapping cameras), override (safe state triggered based on data integrity checks)
SOTIF_1 to SOTIF_6	(broad description of multiple FSRs) Deteriorated performance of camera based object detection and tracking due to triggering conditions within the ODD shall not violate vehicle-level safety goals	n-versioning: diverse redundancy (lidar, radar, camera based)	diverse redundancy (lidar, radar, camera based)
SOTIF_7	If the performance of camera based object detection and tracking deteriorates due to heavy rainfall, the vehicle shall transition to a safe state	n-versioning (diverse redundancy - lidar, radar, camera based)	diverse redundancy (lidar, radar, camera based)

Table 5.3: Results of design assessment of FSRs

the vehicle to a safe state, is determined to be employed for three of the four 26262 FSRs. The tactics of diverse redundancy and replication redundancy, relevant to both SOTIF and ISO 26262 FSRs, are also determined to be employed. Diverse redundancy is present through the use of diverse modalities for the functionality (lidar, radar, camera based), while replication redundancy is applied through the use of two cameras which have partially overlapping fields-of-view.

The employment of certain tactics in the design is not determined. For example, the use of tactics such as *comparison* or *condition monitoring*, for object level information (as for SOTIF FSRs and 26262_8) is not clear from the functional architecture or our high-level code review of the sensing and perception components. A detailed code review of the components is required to determine if these tactics are utilized, which is outside of the design assessment scope.

5.2 Simulation-based testing

This section presents a simulation-based testing approach to assess the fulfillment of FSRs at the implementation level. Section 5.2.1 presents a generic framework to derive testcases for the elicited FSRs in Chapter 4. Next, Section 5.2.2 presents a demonstration of testing example test cases aimed at FSRs related to the lidar-based object detection, tracking, and classification functionality.

The original choice of the AV simulation tool was the simulator from Siemens, Simcenter PreScan [19]. Unfortunately, the in-house integration at Siemens between PreScan and Apollo omits the perception component in Apollo. Currently, PreScan directly provides ground-truth perception data to the prediction component in Apollo. As the integration modification is non-trivial, we could not utilize PreScan for our testing.

Instead, we opt for the open source AV simulator Lgsvl [87] which supports an out-of-the-box integration with Apollo. However, we ran into several technical challenges related to this integration, which could not be resolved despite attempts on multiple versions of Apollo and contact with the Apollo and Lgsvl teams. The technical challenges faced include: (a) radar-based object detection not working (b) camera-based object detection having an unstable detection output (c) the vehicle motion is unstable at times with sudden braking (possibly due to high resource usage as indicated in [88]). Given these technical challenges, we have scoped down our originally planned testing to only consider a few test cases for demonstration.

5.2.1 Requirements to test cases

We systematically derive test cases for the elicited FSRs by first defining test scenarios, as proposed by Post et al. [89]. A test scenario encompasses the set of test cases required to fulfill a FSR, as shown in Figure 5.2. To systematically explore the test cases space within a test scenario, we employ the category partition method [90] and discuss the choice of partitions for testing FSRs w.r.t Apollo.

A test scenario as used in our context is defined as the *encoding of a functional safety requirement into a pre-condition and post-condition*. In addition, we define a corresponding pass/fail criterion to a test scenario, which can be used for the set of test cases within the test scenario. The test scenarios and pass/fail criteria are shown for example FSRs in Table 5.4. In the case of SOTIF_1 (as corresponding only to safety goal 1 of *"avoid collision with an object in driving lane"*), the pass/fail criteria must be defined with reference to results from an additional (reference) test scenario with pre-condition *day-time conditions*. The comparison of the post-conditions of the pre-condition *day-time conditions* and *night-time conditions* enables us to evaluate the specific functional insufficiency of night time conditions as covered by the SOTIF requirement.

Next, we explore the test cases space within the test scenario of a FSR. There are two aspects of a test case: (i) *inputs*; which correspond to the pre-condition of the test scenario and (ii) *test conditions*; a set of conditions under which the test is performed. To systematically explore the input and test conditions space, we use the concepts of *categories* and *partitions* proposed in the category-partition method [90]. The category-partition method is a common black box testing method used to generate test cases for functional requirements [91]. Categories refer to characteristics of the input space and test conditions. Partitions refer to choices within a category which are expected to be handled differently by the system under test.

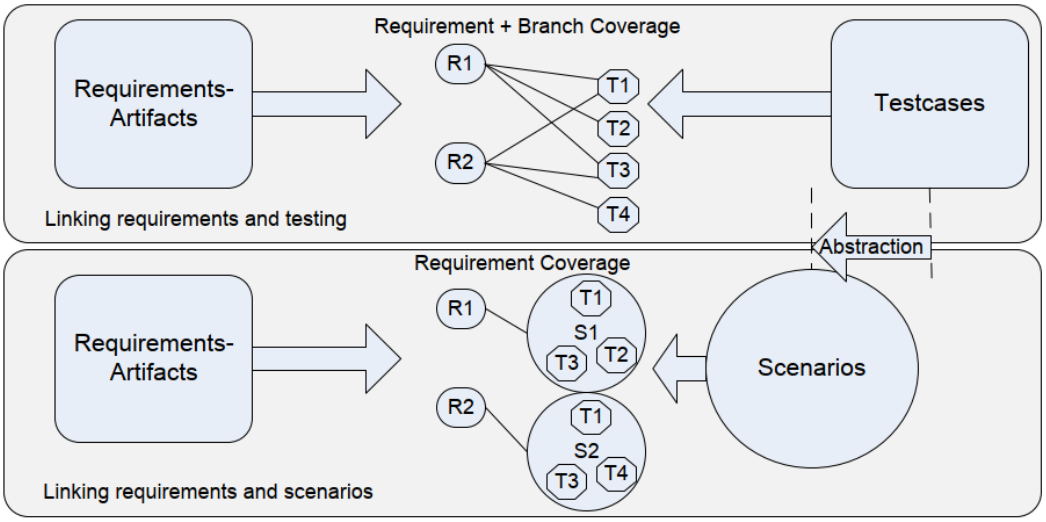

Figure 5.2: Depiction of linking requirements to test cases by the use of test scenarios from Post et al. [89]

Functional safety requirement	**Test scenario: (pre-condition), (post-condition)**	**Pass/Fail criteria for test scenario**
If object detection, classification and tracking (camera) component becomes non-operational, the vehicle shall transition to a safe state (26262_7)	(object detection, classification and tracking (camera) component non-operational), (AV in safe state)	post-condition not met for pre-condition
deteriorated performance of object detection and tracking at night shall not lead to a collision with an object in driving lane (SOTIF_1 for SG1)	(night-time conditions), (AV does not collide with an obstacle or vehicle)	for the same test conditions, the post-condition is not met while it is met for reference pre-condition (day-time conditions)

Table 5.4: Test scenario and pass/fail criterion for an example (i) 26262 FSR and (ii) SOTIF FSR

To illustrate our approach, we consider the same example requirements shown in Table 5.4. For FSR 26262_7, a single category of *duration of non-operational status* is identified for the input space of the pre-condition, *sense environment (camera) module becoming non-operational*. Here, this refers to the duration for which the module is non-operational, from transient failures of a very short duration, e.g., 0.05s, to a permanent failure. For SOTIF_1, the pre-condition can be represented within the test conditions as explained in the following paragraph, and thus no inputs are necessary.

The representation of the test conditions is adopted from that of triggering conditions in the SOTIF standard (as discussed earlier in Section 4.2.2). The test conditions space has three categories: (i) the scene, (ii) the AV status, and (iii) the status of other road users. The *scene* category may be further decomposed into categories representing individual variables in the ODD. In the case of SOTIF_1, the pre-condition of "night-time conditions" is represented using the time-of-day subcategory within the scene category. *The AV status* category also has subcategories such as

operational mode (see Section 3.2) and the health of the system, for example, any active failure events. Finally, *the status of other road users (vehicles)* may be represented by the list of behaviors for other vehicles (detailed for hazardous events in HARA, Section 4.1.1). The status of the AV and other road users should further include subcategories related to other dynamic variables, for instance, driving speed.

The partitions considered for the categories of the input space and test conditions space for each requirement depend on what is expected from the system under test. Based on our code review of Apollo, the partitions that should be considered for test conditions differ for the elicited 26262 requirements and SOTIF FSRs. In the case of 26262 FSRs, a large set of partitions for categories of the test conditions may not be needed. From our code review of the safety components *monitor* and *guardian*, the safety response of the AV (transition to a safe state) as specified in the 26262 FSRs is unaffected by (i) the scene, (ii) parts of the AV status e.g., e.g., driving speed or operational modes as defined in our functional concept (see Section 3.2) (iii) the behavior of other road users. The health of the system, for example, other active failures within the AV, may influence the system under test and therefore should consider multiple partitions. The SOTIF FSRs instead are expected to be significantly affected by external factors and AV status as they relate to safety of vehicle behavior. Thus, either domain knowledge or explorative testing may be used to reason about partitions related to SOTIF FSRs.

5.2.2 Demonstration of testing

Here, we first demonstrate execution of two example test cases for FSRs related to the lidar based object detection, classification, and tracking functionality. Subsequently, we use the framework in Section 5.2.1 to discuss the sufficiency of these test cases in assessing the fulfillment of the FSRs.

26262 FSR	**Test scenario**	**Failure condition injected**
If object detection, classification and tracking (lidar) component output data is corrupted or lost, the vehicle shall transition to a safe state (26262_4)	(object detection, classification and tracking (lidar) output data corrupted or lost), (AV in safe state)	output object data is cleared
If sense environment (lidar) component becomes non-operational, the vehicle shall transition to a safe state (26262_15)	(sense environment (lidar) component non-operational), (AV in safe state)	component is switched off permanently

Table 5.5: Failure conditions (input) for two of the 26262 FSRs related to the lidar-based object detection, classification, and tracking functionality

The FSR, its corresponding test scenario, and the chosen failure condition (input) is shown in Table 5.5. The test conditions chosen are shown in Figure 5.3 and have been setup using the PythonAPI in Lgsvl [2]. In the setup, a stationery vehicle (in white) is positioned in the AV's driving lane. The AV (in blue) is provided a destination in the highway lane such that it encounters the stationery vehicle during the trip. Here, we have limited the AV maximum speed to 20km/h [3].

Next, we describe how the two input failure test conditions are injected. To simulate a non-operational sensor, we intercept the data channel between Lgsvl and Apollo for the lidar sensor data. An intermediate node is setup, which obtains data from Lgsvl, and provides it to Apollo. The non-operational sensor failure can thus be simulated at any time during the AV trip by stopping this intermediate node. Data corruption of missing objects at the output data of the

[2] https://www.svlsimulator.com/docs/python-api/python-api/

[3] Unstable behavior for Apollo was observed during initial tests at high speeds

Figure 5.3: Test setup for the testing of 26262 FSRs. The AV and the stationery vehicle are in blue and white respectively.

camera-based object perception component is simulated by clearing the list of objects generated by the NN. This failure is inserted at compile-time (within Apollo code).

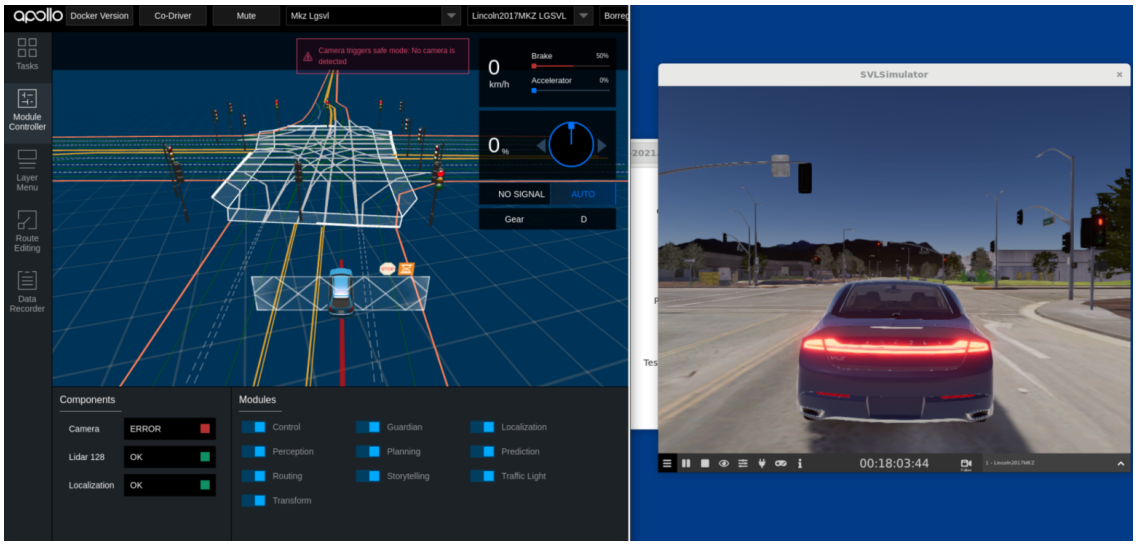

Figure 5.4: Example of a message on the Apollo HMI used to determine testing results. The message describes the triggering of a safe state upon a sensor not being detected.

To evaluate the post-condition of both test scenarios, *AV in safe state*, the corresponding test result to the post-condition is chosen as "a message appears on the HMI screen indicating a safe state has been triggered". The message appears on the HMI when the *monitor* component triggers

the guardian to perform either an emergency stop (safe state S1) or a slow stop (safe state S2) (as described in Section 4.3.2). We choose to check the message on the HMI instead of observing vehicle behavior given the instabilities observed in the vehicle's motion. Figure 5.4 presents an example message on the Apollo HMI indicating a transition to a safe state upon a sensor not being detected.

The test case results are summarized in Table 5.6. TC_0 has no failure condition injection and passes successfully with no error message on the HMI. We observe that TC_1 fails while TC_2 passes. Due to the failing of TC_1, we may directly conclude that the corresponding FSR to TC_1 (26262_4) is not fulfilled in Apollo. We may now consider what insights about FSR 26262_15 may be gained from TC_2, which passes. For this, we consider the framework in Section 5.2.1 to reason about the test cases space (inputs and test conditions) of the corresponding test scenario to FSR 26262_15.

As we discussed in Section 5.2.1, although multiple partitions of scene characteristics, behaviors of other road users, may not be needed for testing of 26262 FSRs w.r.t Apollo, partitions corresponding to the AV internal status - the health of other components in the AV, may be considered. In addition, the possible inputs corresponding to the pre-condition have a category of *duration*, for which multiple partitions may also be considered, i.e. transient failures or permanent failures. Thus, other test cases may be necessary to fully test the fulfillment of the FSR 26262_15 in Apollo.

Test case ID	**Associated FSR ID**	**Failure condition injected**	**Expected test result**	**Actual test result**	**Test case pass/fail**
TC_0	-	-	safe state message not triggered	safe state message not triggered	Pass
TC_1	26262_4	output object data is cleared	safe state message triggered	safe state message not triggered	Fail
TC_2	26262_15	component is switched off	safe state message triggered	safe state message triggered	Pass

Table 5.6: Results for two test cases performed as demonstration for two of the 26262 FSRs related to the lidar-based object detection, classification, and tracking functionality

5.3 Summary

This chapter presented methods and results for the assessment of functional safety in Apollo with respect to functional safety requirements related to the vehicle's AI-based sensing and perception subsystem at the (i) design-level and (ii) implementation level.

The design-level assessment of functional safety is performed based on an existing methodology which assesses the fulfillment of FSRs by checking for application of suitable architectural tactics (safety tactics) in the vehicle's software architecture. We extend this approach further to design artefacts related to AI components (dataset, NN model), and best practices which address the safety of AI components. The list of safety related best practices is extracted from literature. The assessment was applied to FSRs related to the object detection, classification, and tracking functionality. The results demonstrate, in particular, a lack of best practices employed in the Apollo design for the safety of AI-based components, as pertaining to SOTIF FSRs.

The implementation-level assessment of functional safety considered a simulation-based testing approach. We present a systematic derivation of test cases for the elicited FSRs, by first defining a test scenario for each FSR. Then, we explore the test cases space within test scenarios; representing

variations in the inputs and test conditions, through the category-partition method. In this way, we can determine the sufficiency of performed tests, and reason on the types of test cases necessary, for the assessment of the elicited FSRs. We demonstrate test results for example test cases which reveals an unfulfilled FSR in Apollo.

Chapter 6

Conclusions and future work

Technologies for autonomous vehicles (AVs) are rapidly developing and have been certified for public road testing by several countries in recent years. Ensuring the safety of AVs is an important step for public acceptance and regulation of the technology. The area of functional safety, as governed by safety standards ISO 26262 and SOTIF, addresses risks posed due to system faults and functional insufficiencies. This thesis, initiated in collaboration with Siemens Digital Industries Software, is aimed at assessing the functional safety of a SAE L4 AV operating in Dutch highway settings.

The importance of assessing the functional safety of the AV with respect to a well specified operational design domain (ODD) is highlighted by the influence of the ODD on the functional safety requirements (FSRs) and the appropriate fallback response. In addition, the use of AI-based components in the AV poses challenges for functional safety. This motivated the two research questions of this graduation project. For our investigations, we choose an open-source industrial-grade AV software stack, Baidu Apollo.

Section 6.1 concludes our findings on the research questions. Then, Section 6.2 highlights the main contributions of this thesis. Finally, Section 6.3 describes limitations of the study and proposes future research directions.

6.1 Conclusions on research questions

RQ1: *What functional safety requirements shall be fulfilled by the safety system of a SAE L4 autonomous vehicle in a Dutch highway setting, regarding faults and functional insufficiencies in the AI-based sensing and perception subsystem?*

The FSRs are elicited based on the automotive safety standards ISO 26262 and DIS/ISO 21448 for an ODD based on the A270 Dutch highway. The *hazard analysis and risk assessment* (HARA) process, common to both standards, identified 15 safety goals of automotive safety integrity level (ASIL) A or higher. The FSRs are elicited regarding faults and functional insufficiencies in the AI-based sensing and perception subsystem which may lead to violation of the safety goals. The ISO 26262 FSRs address two failure types related to faults: (i) *non-operational status* and (ii) *output data corruption*. The SOTIF FSRs address the functional insufficiencies of the camera-based object detection, classification, and tracking functionality to illumination and weather conditions in the ODD. The safety requirements are finally refined according to the safety concept in Apollo, which activates vehicle-level safe states in response to failures of components.

RQ2: *How to assess the functional safety of a SAE L4 autonomous vehicle regarding faults and functional insufficiencies in the AI-based sensing and perception subsystem?*

The functional safety assessment is performed against FSRs at two levels; the design level and the implementation level. For the design-level assessment, we apply an existing approach which assesses fulfillment of FSRs by checking for relevant safety tactics in the vehicle's software architecture. As the approach and safety tactics do not take into account FSRs related to AI-based components, we further extend the approach to include AI-safety related best practices and AI related design artefacts (dataset, NN model). The application of our approach demonstrates a lack of best practices employed in the Apollo design for the safety of AI-based components. For the implementation-level assessment of the FSRs, a simulation-based testing approach is considered. A systematic approach is outlined to derive test cases for the elicited FSRs. We subsequently demonstrate the execution of example test cases in a simulation test setup for the assessment of FSRs.

6.2 Main contributions

The main contributions of this thesis are summarized as follows:

1. Elicit functional safety requirements for a Dutch highway setting combining safety processes of the automotive industry safety standards: ISO 26262 and DIS/ISO 21448.
2. Adapt an existing approach to assess fulfillment of FSRs at the architecture level to FSRs related to AI-based components by additionally considering AI-safety related best practices and AI related design artefacts (dataset, NN model).
3. Outline a systematic approach to the derivation of test cases for simulation-based testing of functional safety requirements and demonstrate the approach in a simulation test setup.

6.3 Limitations and future work

Here, we outline some of the limitations of the current study and propose future research directions. We first discuss the FSR elicitation process, especially the refining of FSRs for Apollo. Then, we discuss future opportunities for functional safety assessment of the AV w.r.t AI-based components. Finally, we outline possible extensions for the simulation-based testing approach to functional safety assessment.

The final step of FSR elicitation refines the FSRs based on Apollo's safety concept of vehicle-level safe states. For example, the FSR *"Non-operational state of the traffic light detection and recognition component shall not lead the AV to ignore red or yellow traffic light in operational mode: crossing an intersection."* is refined to *"If the traffic light detection and recognition component becomes non-operational, the AV shall transition to a safe state"*. The refining of FSRs in this fashion may not be fully justified as potentially other mechanisms which achieve the former FSR, but not its refined version, may be implemented in Apollo. Thus, a simulation-based testing approach which assesses the original formulation of the FSR (as well as the refined FSR) may be more suited for functional safety assessment.

In addition, the approach used to refine the FSRs also has further limitations such as a lack of consideration for the different severity levels of failures considered in a FSR. For example, the FSR *"Sense environment (camera) module output data corruption or loss shall not lead the AV to collide with an object"* is refined to *"If sense environment (camera) module output data is corrupted or lost, the AV shall transition to a safe state."* The former version of the FSR is appropriate for all severity levels of data corruption or loss as it relates to ensuring the failure does not violate a safety goal. However, the refined FSR may only make sense for a certain threshold severity level of data corruption or loss which does not allow for nominal operation. Thus, further granularity with respect to failure types should be considered in the refining of the FSRs.

The current approach to functional safety assessment only partially addresses functional safety related to AI-based components. The FSR elicitation process was performed using industry safety standards which include AI-based components within their scope. However, the FSR elicitation process does not consider specific characteristics of AI-based components, e.g., unpredictable behavior in novel conditions [13, 14]. The specific characteristics of AI-based components may also be considered in the assessment of functional safety. This was done in our design-level assessment but was not explored in our simulation-based testing approach.

Our simulation-based testing approach is generic and may also include safety considerations for AI-based components. For example, a safety concern of AI-based components is their low robustness to small shifts in input data distribution [15]. Thus, while testing AI-based components, it may be necessary to define test cases considering variations in test conditions at a low level of granularity. Such considerations for simulation-based testing are not defined nor tested in our approach and may be an interesting future research direction. In addition, although we investigated the state-of-the-art of V&V methods for the AI-based sensing and perception subsystem, we did not consider their integration in our approach. The exploration of how these methods may be used to assess FSRs elicited within the framework of the industry safety standards, as in our case, is also an interesting research direction.

Finally, the scope of our simulation-based testing was limited to a demonstration of example test cases due to technical challenges faced in establishing a simulation test setup for Apollo. Thus, the functional safety assessment of the FSRs through simulation-based testing requires further investigation. We also note that the possible test cases space for the FSRs, especially SOTIF requirements, is large. Therefore, research directions related to combinatorial testing techniques or automatic critical test case generation for the assessment of FSRs may be interesting to explore in future work.

Bibliography

[1] "Green light for experimental law for testing self-driving vehicles on public roads." https://www.government.nl/latest/news/2019/07/02/green-light-for-experimental-law-for-testing-self-driving-vehicles-on-public-roads. Accessed: 2021-09-13. 1

[2] "Automated vehicles for safety." https://www.nhtsa.gov/technology-innovation/automated-vehicles. Accessed: 2021-01-24. 1

[3] "Honda launches world's first level 3 self-driving car." https://asia.nikkei.com/Business/Automobiles/Honda-launches-world-s-first-level-3-self-driving-car. Accessed: 2021-03-25. 1

[4] "Waymo opens self-driving car testing to some san francisco residents." https://www.cnbc.com/2021/08/24/waymo-opens-self-driving-car-testing-to-some-san-francisco-residents.html. Accessed: 2021-09-13. 1

[5] "Waymo safety report: On the road to fully self-driving." https://assets.ctfassets.net/sv23gofxcuiz/1xAGjnH0kTxD2Vmvqsv3eS/7da216660cbd9ee7b35eefcac1b28a2f/waymo-safety-report-2018.pdf. Accessed: 2021-01-12. 1, 3, 5, 14, 33

[6] "Safety first for automated driving, 2019." https://www.daimler.com/documents/innovation/other/safety-first-for-automated-driving.pdf. Accessed: 2021-01-11. 1, 5, 6, 18

[7] "ISO 26262-1:2011 road vehicles — functional safety — part 1: Vocabulary." https://www.iso.org/standard/43464.html. Accessed: 2021-05-24. 1

[8] "ISO 26262-3:2018 road vehicles — functional safety — part 3: Concept phase." https://www.iso.org/standard/68385.html. Accessed: 2021-05-24. 1, 23, 24

[9] "Self-driving safety report." https://www.gm.com/content/dam/company/docs/us/en/gmcom/gmsafetyreport.pdf. Accessed: 2021-01-12. 1, 3, 5, 14, 33

[10] "ISO/DIS 21448 road vehicles — safety of the intended functionality." https://www.iso.org/standard/77490.html. Accessed: 2021-08-06. 1, 23, 24, 31

[11] E. Arnold, O. Y. Al-Jarrah, M. Dianati, S. Fallah, D. Oxtoby, and A. Mouzakitis, "A survey on 3d object detection methods for autonomous driving applications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 10, pp. 3782–3795, 2019. 1

[12] R. Salay, R. Queiroz, and K. Czarnecki, "An analysis of ISO 26262: Using machine learning safely in automotive software," *arXiv preprint arXiv:1709.02435*, 2017. 1

[13] J. Serna, S. Diemert, L. Millet, R. Debouk, S. Ramesh, and J. Joyce, "Bridging the gap between ISO 26262 and machine learning: A survey of techniques for developing confidence in machine learning systems," *SAE International Journal of Advances and Current Practices in Mobility*, vol. 2, no. 2020-01-0738, pp. 1538–1550, 2020. 1, 38, 51

[14] M. Borg, C. Englund, K. Wnuk, B. Duran, C. Levandowski, S. Gao, Y. Tan, H. Kaijser, H. Lönn, and J. Törnqvist, "Safely entering the deep: A review of verification and validation for machine learning and a challenge elicitation in the automotive industry," *arXiv preprint arXiv:1812.05389*, 2018. 1, 11, 38, 51

[15] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013. 1, 51

[16] "Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles." https://www.sae.org/standards/content/j3016_202104/. Accessed: 2021-09-14. 2, 6, 7

[17] A. K. Saberi, J. Vissers, and F. P. A. Benders, "On the impact of early design decisions on quality attributes of automated driving systems," in *2019 IEEE International Systems Conference (SysCon)*, pp. 1–6, 2019. 2

[18] E. Thorn, S. C. Kimmel, M. Chaka, B. A. Hamilton, *et al.*, "A framework for automated driving system testable cases and scenarios," tech. rep., US DOT National Highway Traffic Safety, 2018. 2, 13, 32

[19] "Simcenter prescan." https://www.plm.automation.siemens.com/global/en/products/simcenter/prescan.html. Accessed: 2021-09-12. 3, 42

[20] "Autoware.auto." https://autowarefoundation.gitlab.io/autoware.auto/AutowareAuto/index.html. Accessed: 2021-04-03. 3

[21] S. Kato, E. Takeuchi, Y. Ishiguro, Y. Ninomiya, K. Takeda, and T. Hamada, "An open approach to autonomous vehicles," *IEEE Micro*, vol. 35, no. 6, pp. 60–68, 2015. 3

[22] "Baidu apollo." https://github.com/ApolloAuto/apollo. Accessed: 2021-04-03. 3, 12

[23] "Waymo and cruise dominated autonomous testing in california in the first year of the pandemic." https://www.theverge.com/2021/2/11/22276851/california-self-driving-autonomous-cars-miles-waymo-cruise-2020. Accessed: 2021-02-23. 5

[24] S. Otsuka and K. Sakurai, "A safety concept based on a safety sustainer for highly automated driving systems," tech. rep., SAE Technical Paper, 2016. 6, 10

[25] E. Pakdamanian, S. Sheng, S. Baee, S. Heo, S. Kraus, and L. Feng, "Deeptake: Prediction of driver takeover behavior using multimodal data," *arXiv preprint arXiv:2012.15441*, 2020. 7

[26] M. Hörwick and K.-H. Siedersberger, "Strategy and architecture of a safety concept for fully automatic and autonomous driving assistance systems," in *2010 IEEE Intelligent Vehicles Symposium*, pp. 955–960, IEEE, 2010. 7, 9, 10

[27] R. Novickis, A. Levinskis, R. Kadikis, V. Fescenko, and K. Ozols, "Functional architecture for autonomous driving and its implementation," in *2020 17th Biennial Baltic Electronics Conference (BEC)*, pp. 1–6, IEEE, 2020. 7, 10

[28] S. vom Dorff, B. Böddeker, M. Kneissl, and M. Fränzle, "A fail-safe architecture for automated driving," in *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 828–833, IEEE, 2020. 7, 9, 10

[29] S. Lefèvre, D. Vasquez, and C. Laugier, "A survey on motion prediction and risk assessment for intelligent vehicles," *ROBOMECH journal*, vol. 1, no. 1, pp. 1–14, 2014. 7

[30] D. Ridel, E. Rehder, M. Lauer, C. Stiller, and D. Wolf, "A literature review on the prediction of pedestrian behavior in urban scenarios," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 3105–3112, IEEE, 2018. 7

[31] M. Törngren, X. Zhang, N. Mohan, M. Becker, L. Svensson, X. Tao, D.-J. Chen, and J. Westman, "Architecting safety supervisors for high levels of automated driving," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1721–1728, IEEE, 2018. 7, 8, 10

[32] T. Ishigooka, S. Honda, and H. Takada, "Cost-effective redundancy approach for fail-operational autonomous driving system," in *2018 IEEE 21st International Symposium on Real-Time Distributed Computing (ISORC)*, pp. 107–115, IEEE, 2018. 8, 9, 10

[33] Y. Fu, A. Terechko, J. F. Groote, and A. K. Saberi, "A formally verified highly resilient safety concept with degraded modes for automated driving," *arXiv preprint arXiv:2011.00892*, 2020. 8, 9, 10

[34] Y. Luo, A. K. Saberi, T. Bijlsma, J. J. Lukkien, and M. van den Brand, "An architecture pattern for safety critical automated driving applications: Design and analysis," in *2017 Annual IEEE International Systems Conference (SysCon)*, pp. 1–7, IEEE, 2017. 8, 9, 10

[35] B. P. Douglass, *Real-time design patterns: robust scalable architecture for real-time systems*. Addison-Wesley Professional, 2003. 8

[36] T. Fruehling, A. Hailemichael, C. Graves, J. Riehl, E. Nutt, R. Fischer, and A. K. Saberi, "Architectural safety perspectives & considerations regarding the ai-based av domain controller," in *2019 IEEE International Conference on Connected Vehicles and Expo (ICCVE)*, pp. 1–10, IEEE, 2019. 8, 9, 10

[37] A. Reschka, J. R. Böhmer, T. Nothdurft, P. Hecker, B. Lichte, and M. Maurer, "A surveillance and safety system based on performance criteria and functional degradation for an autonomous vehicle," in *2012 15th International IEEE Conference on Intelligent Transportation Systems*, pp. 237–242, IEEE, 2012. 9, 10

[38] I. Colwell, B. Phan, S. Saleem, R. Salay, and K. Czarnecki, "An automated vehicle safety concept based on runtime restriction of the operational design domain," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1910–1917, IEEE, 2018. 8, 9, 10

[39] C. B. S. T. Molina, J. R. de Almeida, L. F. Vismari, R. I. R. González, J. K. Naufal, and J. B. Camargo, "Assuring fully autonomous vehicles safety by design: the autonomous vehicle control (avc) module strategy," in *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*, pp. 16–21, IEEE, 2017. 9, 10

[40] S. Jha, S. Banerjee, T. Tsai, S. K. Hari, M. B. Sullivan, Z. T. Kalbarczyk, S. W. Keckler, and R. K. Iyer, "Ml-based fault injection for autonomous vehicles: A case for bayesian fault injection," in *2019 49th annual IEEE/IFIP international conference on dependable systems and networks (DSN)*, pp. 112–124, IEEE, 2019. 11, 12

[41] Z. Chen, G. Li, K. Pattabiraman, and N. DeBardeleben, "Binfi: an efficient fault injector for safety-critical machine learning systems," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–23, 2019. 11, 12

[42] A. H. M. Rubaiyat, Y. Qin, and H. Alemzadeh, "Experimental resilience assessment of an open-source driving agent," in *2018 IEEE 23rd Pacific rim international symposium on dependable computing (PRDC)*, pp. 54–63, IEEE, 2018. 11, 12

[43] M. Zhang, Y. Zhang, L. Zhang, C. Liu, and S. Khurshid, "Deeproad: Gan-based metamorphic testing and input validation framework for autonomous driving systems," in *2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pp. 132–142, IEEE, 2018. 11, 12

[44] Y. Tian, K. Pei, S. Jana, and B. Ray, "Deeptest: Automated testing of deep-neural-network-driven autonomous cars," in *Proceedings of the 40th international conference on software engineering*, pp. 303–314, 2018. 11, 12

[45] D. Rao, P. Pathrose, F. Huening, and J. Sid, "An approach for validating safety of perception software in autonomous driving systems," in *International Symposium on Model-Based Safety and Assessment*, pp. 303–316, Springer, 2019. 11, 12

[46] "Openpilot." https://github.com/commaai/openpilot. Accessed: 2021-09-12. 12

[47] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*, pp. 1–16, PMLR, 2017. 12

[48] "Nvidia driveworks." https://developer.nvidia.com/drive/driveworks. Accessed: 2021-09-12. 12

[49] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, *et al.*, "End to end learning for self-driving cars," *arXiv preprint arXiv:1604.07316*, 2016. 12

[50] "The comma.ai driving dataset." https://github.com/commaai/research. Accessed: 2021-09-12. 12

[51] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014. 12

[52] "Steering models." https://github.com/udacity/self-driving-car/tree/master/steering-models. Accessed: 2021-09-12. 12

[53] "Automated driving systems 2.0 a vision for safety." https://www.nhtsa.gov/sites/nhtsa.dot.gov/files/documents/13069a-ads2.0_090617_v9a_tag.pdf. Accessed: 2021-01-10. 13

[54] AVSC, "Avsc best practice for describing an operational design domain: Conceptual framework and lexicon," tech. rep., SAE Industry Technologies Consortia, 2020. 13

[55] "Road traffic signs and regulations in the netherlands." https://www.universiteitleiden.nl/binaries/content/assets/customsites/study-abroad-exchange-students/road_traffic_signs_and_regulations_jan_2013_uk.pdf. Accessed: 2021-06-14. 14, 16, 32

[56] "Road design - which road categories are distinguished in the netherlands?." https://www.swov.nl/en/facts-figures/fact/road-design-which-road-categories-are-distinguished-netherlands. Accessed: 2021-03-13. 14

[57] "North brabant climate." https://en.climate-data.org/europe/the-netherlands/north-brabant-334/. Accessed: 2021-06-05. 14

[58] E. Shang, X. An, T. Wu, T. Hu, Q. Yuan, and H. He, "Lidar based negative obstacle detection for field autonomous land vehicles," *Journal of Field Robotics*, vol. 33, no. 5, pp. 591–617, 2016. 14

[59] "Safety assessment report sae level 3 automated driving system." https://www.bmwusa.com/content/dam/bmwusa/innovation-campaign/autonomous/BMW-Safety-Assessment-Report.pdf. Accessed: 2021-03-13. 14

[60] S. Zang, M. Ding, D. Smith, P. Tyler, T. Rakotoarivelo, and M. A. Kaafar, "The impact of adverse weather conditions on autonomous vehicles: how rain, snow, fog, and hail affect the performance of a self-driving car," *IEEE vehicular technology magazine*, vol. 14, no. 2, pp. 103–111, 2019. 15

[61] "Apollo 5.5 software architecture." https://github.com/ApolloAuto/apollo/blob/master/docs/specs/Apollo_5.5_Software_Architecture.md. Accessed: 2021-09-03. 16

[62] C. Becker, J. C. Brewer, L. Yount, *et al.*, "Safety of the intended functionality of lane-centering and lane-changing maneuvers of a generic level 3 highway chauffeur system," tech. rep., US DOT National Highway Traffic Safety, 2020. 23, 31

[63] "Hazard & operability analysis." https://pqri.org/wp-content/uploads/2015/08/pdf/HAZOP_Training_Guide.pdf. Accessed: 2021-05-11. 24

[64] "Hazard analysis and risk assessment and functional safety concept, d2.11 of h2020 project ensemble." https://platooningensemble.eu/storage/uploads/documents/2021/03/24/ENSEMBLE-D2.11-Fist-version-Hazard-Analysis-and-Risk-Assessment-and-Functional-Safety-Concept_FINAL.pdf. Accessed: 2021-04-10. 24, 25

[65] G. Bagschik, A. Reschka, T. Stolte, and M. Maurer, "Identification of potential hazardous events for an unmanned protective vehicle," in *2016 IEEE Intelligent Vehicles Symposium (IV)*, pp. 691–697, IEEE, 2016. 24, 25

[66] "ISO 26262-3:2011 road vehicles — functional safety — part 3: Concept phase." https://www.iso.org/standard/51358.html. Accessed: 2021-05-24. 26

[67] M. Aladem, S. Baek, and S. A. Rawashdeh, "Evaluation of image enhancement techniques for vision-based navigation under low illumination," *Journal of Robotics*, 2019. 31

[68] "ISO/PAS 21448:2019 road vehicles — safety of the intended functionality." https://www.iso.org/standard/70939.html. Accessed: 2021-01-24. 31

[69] S.-C. Huang, T.-H. Le, and D.-W. Jaw, "Dsnet: Joint semantic learning for object detection in inclement weather conditions," *IEEE transactions on pattern analysis and machine intelligence*, 2020. 31

[70] V. A. Sindagi, P. Oza, R. Yasarla, and V. M. Patel, "Prior-based domain adaptive object detection for hazy and rainy conditions," in *European Conference on Computer Vision*, pp. 763–780, Springer, 2020. 31

[71] G. Volk, S. Müller, A. von Bernuth, D. Hospach, and O. Bringmann, "Towards robust cnn-based object detection through augmentation with synthetic rain variations," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 285–292, IEEE, 2019. 31

[72] C. Michaelis, B. Mitzkus, R. Geirhos, E. Rusak, O. Bringmann, A. S. Ecker, M. Bethge, and W. Brendel, "Benchmarking robustness in object detection: Autonomous driving when winter is coming," *arXiv preprint arXiv:1907.07484*, 2019. 31

[73] D. Frossard and R. Urtasun, "End-to-end learning of multi-sensor 3d tracking by detection," in *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 635–642, IEEE, 2018. 31

[74] "The relation between speed and crashes." https://safety.fhwa.dot.gov/speedmgt/ref_mats/fhwasa1304/Resources3/08%20-%20The%20Relation%20Between%20Speed%20and%20Crashes.pdf. Accessed: 2021-07-11. 32

[75] T. Horberry, L. Hartley, K. Gobetti, F. Walker, B. Johnson, S. Gersbach, and J. Ludlow, "Speed choice by drivers: The issue of driving too slowly," *Ergonomics*, vol. 47, no. 14, pp. 1561–1570, 2004. 32

[76] S. Kochanthara, N. Rood, L. Cleophas, Y. Dajsuren, and M. van den Brand, "Semi-automatic architectural suggestions for the functional safety of cooperative driving systems," in *2020 IEEE International Conference on Software Architecture Companion (ICSA-C)*, pp. 55–58, IEEE, 2020. 37, 38

[77] S. Kochanthara, N. Rood, A. K. Saberi, L. Cleophas, Y. Dajsuren, and M. van den Brand, "A functional safety assessment method for cooperative automotive architecture," *Journal of Systems and Software*, vol. 179, p. 110991, 2021. 37, 38

[78] L. Bass, P. Clements, and R. Kazman, "Software architecture in practice," 2013. 38

[79] C. Preschern, N. Kajtazovic, and C. Kreiner, "Building a safety architecture pattern system," in *Proceedings of the 18th European Conference on Pattern Languages of Program*, pp. 1–55, 2015. 38

[80] W. Wu and T. Kelly, "Safety tactics for software architecture design," in *Proceedings of the 28th Annual International Computer Software and Applications Conference, 2004. COMPSAC 2004.*, pp. 368–375, IEEE, 2004. 38

[81] A. Serban, K. van der Blom, H. Hoos, and J. Visser, "Adoption and effects of software engineering best practices in machine learning," in *Proceedings of the 14th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, pp. 1–12, 2020. 39

[82] S. Mohseni, M. Pitale, V. Singh, and Z. Wang, "Practical solutions for machine learning safety in autonomous vehicles," *arXiv preprint arXiv:1912.09630*, 2019. 39, 40

[83] A. Serban and J. Visser, "An empirical study of software architecture for machine learning," *arXiv preprint arXiv:2105.12422*, 2021. 39

[84] "Mlops architecture guide." `https://neptune.ai/blog/mlops-architecture-guide`. Accessed: 2021-09-05. 39

[85] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, *et al.*, "Scalability in perception for autonomous driving: Waymo open dataset," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2446–2454, 2020. 38

[86] Z. Liu, Z. Wu, and R. Tóth, "Smoke: Single-stage monocular 3d object detection via keypoint estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 996–997, 2020. 38

[87] G. Rong, B. H. Shin, H. Tabatabaee, Q. Lu, S. Lemke, M. Možeiko, E. Boise, G. Uhm, M. Gerow, S. Mehta, *et al.*, "Lgsvl simulator: A high fidelity simulator for autonomous driving," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–6, IEEE, 2020. 42

[88] "Running latest apollo with svl simulator." `https://www.svlsimulator.com/docs/system-under-test/apollo-master-instructions/`. Accessed: 2021-09-29. 42

[89] H. Post, C. Sinz, F. Merz, T. Gorges, and T. Kropf, "Linking functional requirements and software verification," in *2009 17th IEEE International Requirements Engineering Conference*, pp. 295–302, IEEE, 2009. 42, 43

[90] T. J. Ostrand and M. J. Balcer, "The category-partition method for specifying and generating fuctional tests," *Communications of the ACM*, vol. 31, no. 6, pp. 676–686, 1988. 42

[91] M. J. Escalona, J. J. Gutierrez, M. Mejías, G. Aragón, I. Ramos, J. Torres, and F. J. Domínguez, "An overview on test generation from functional requirements," *Journal of Systems and Software*, vol. 84, no. 8, pp. 1379–1393, 2011. 42