

# **Manövergestütztes Testen für das durchgängige Erstellen, Auswählen und Ausführen von Testszenarien von Hochautomatisierten Fahrfunktionen**

Zur Erlangung des akademischen Grades eines

**DOKTORS DER INGENIEURWISSENSCHAFTEN  
(Dr.-Ing.)**

von der KIT-Fakultät für Elektrotechnik und Informationstechnik  
des Karlsruher Instituts für Technologie (KIT)

angenommene

**DISSERTATION**

von

**M.Sc. Thilo Braun**

geb. in Heidelberg

Tag der mündlichen Prüfung:

10.09.2025

Hauptreferent:

Prof. Dr.-Ing. Eric Sax

Korreferent:

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



# Kurzfassung

Automatisierte Fahrfunktionen haben das Potenzial, den Verkehr durch verbesserte Sicherheit, erhöhten Komfort, neue Mobilitätskonzepte und wirtschaftliche Vorteile im Güterverkehr grundlegend zu verändern. Aber insbesondere bei Interaktionen zwischen Verkehrsteilnehmern im urbanen Raum sind Verifikations- und Validierungsstrategien noch in der Entwicklung. Aktuelle Forschungsprojekte und die Industrie fokussieren dafür das szenariobasierte Testen. Dabei werden die Fahrszenarien in (un-)bekannte und (un-)sichere Szenarien eingeteilt. Für die Sicherheitsargumentation wird dann nachgewiesen, dass die Zahl an unsicheren und unbekannten Szenarien ausreichend klein ist.

Bisherige Arbeiten befassen sich aber entweder mit dem Gesamtprozess auf einer abstrakten Ebene oder fokussieren sich auf einzelne Problemstellungen ohne Berücksichtigung der übrigen Prozessschritte. Dies führt zu einer Vielzahl von Insellösungen, die nicht miteinander verknüpft sind und eine nahtlose Integration in den Entwicklungsprozess erschweren.

Eine wesentliche Lücke besteht in der uneinheitlichen Szenariobeschreibung, die die Effizienz der Methode beeinträchtigt. Diese Variation resultiert aus der isolierten Betrachtung der Problemstellungen, die jeweils unterschiedliche Arten der Szenariobeschreibung verwenden. Die Motivation dieser Arbeit ist die Entwicklung einer durchgängigen Methode, die eine einheitliche Form der Szenarien im gesamten Prozess unterstützt. Dazu wird spezifisch untersucht, wie Verkehrsszenarien formalisiert beschrieben werden können, um unbekannte Szenarien zu finden, repräsentative Testszenarien auszuwählen und diese auszuführen.

Ausgehend von der Betrachtung des Standes der Wissenschaft und Technik wird eine Liste von acht Anforderungen an eine Szenariobeschreibung hergeleitet. Darauf aufbauend wird das Konzept des manövergestützten Testens ausgearbeitet. Die Szenariobeschreibung, welche das zentrale Element in dieser Methode darstellt, wird dabei auf Basis von Manövern aufgebaut, welche

---

das Verhalten der Verkehrsteilnehmer im Szenario abstrahiert und formalisiert beschreibt. Grundsatz im manövergestützten Testen ist, dass die einzelnen Prozessschritte des Testprozesses (Erstellen der Szenarien sowie Ableiten, Ausführen und Evaluieren der Testfälle) so umgesetzt werden, dass sie alle mit dieser manöverbasierten Beschreibungsform arbeiten. Für das Erstellen der Szenarien wird dazu eine Manöver-Erkennung durchgeführt, die Auswahl der Testfälle erfolgt durch ein dafür entwickeltes Ähnlichkeitsmaß als Basis für einen Clusteralgorithmus und die Ausführung wird durch eine teilautomatisierte Resimulation realisiert.

Das entwickelte Konzept wird prototypisch für alle diese Prozessschritte umgesetzt. Für eine datenbasierte Evaluation wird eine mehrwöchige Verkehrsaufzeichnung einer Kreuzung in Mannheim genutzt. Dabei wird erläutert, wie das manövergestützte Testen die zuvor erstellten Anforderungen erfüllt. Somit wird gezeigt, dass der durchgängige Einsatz einer manöverbasierten Szenariobeschreibung grundsätzlich funktioniert und die Effizienz des gesamten Testprozesses steigert.

Diese Dissertation leistet somit einen Beitrag zum durchgängigen Testen von Automatisierten Fahrfunktionen, indem sie eine systematische und effiziente Validierung ermöglicht, um die Potenziale der Fahrfunktionen im Straßenverkehr zeitnah auszuschöpfen.



# Abstract

Automated driving functions have the potential to fundamentally change traffic through improved safety, increased comfort, new mobility concepts and economic benefits in freight transport. However, verification and validation strategies are still being developed, particularly for interactions between road users in urban areas. Current research projects and the industry are focusing on scenario-based testing. This approach classifies driving scenarios into (un)known and (un)safe scenarios. For the safety argumentation, it is then demonstrated that the number of unsafe and unknown scenarios is sufficiently small.

However, previous work either deals with the overall process at an abstract level or focuses on individual problems without taking the other process steps into account. This leads to numerous isolated solutions that are not linked to each other and make seamless integration into the development process more difficult.

A significant gap is the inconsistent scenario description, which impairs the efficiency of the method. This variation results from the isolated consideration of the problems, each of which uses different types of scenario description. The motivation of this work is to develop a consistent method that supports a uniform form of scenarios throughout the process. For this purpose, it is specifically investigated how traffic scenarios can be described in a formalized way in order to find unknown scenarios, select representative test scenarios and execute them.

Based on the state of the art in science and technology, a list of eight requirements for a scenario description is derived. Based on this, the concept of maneuver-guided testing is developed. The scenario description, which is the central element of this method, is based on maneuvers that describe the behavior of road users in the scenario in an abstract and formalized way. The basic principle of maneuver-guided testing is that the individual process steps

---

of the test process (creating the scenarios and deriving, executing and evaluating the test cases) are implemented in such a way that they all work with this maneuver-guided form of description. For the creation of the scenarios, a maneuver identification is carried out, the test cases are selected using a similarity measure developed for this purpose as the basis for a cluster algorithm and the execution is realized using a partially automated resimulation.

The developed concept will be implemented as a prototype for all of these process steps. A traffic recording of an intersection in Mannheim over several weeks is used for a data-based evaluation. It is explained how the maneuver-guided testing fulfills the previously created requirements. This demonstrates that the consistent use of a maneuver-based scenario description works in principle and increases the efficiency of the entire test process.

This dissertation thus makes a contribution to the seamless testing of automated driving functions by enabling systematic and efficient validation in order to exploit the potential of driving functions in road traffic in a timely manner.

# Danksagung

Mein besonderer Dank gilt meinem Doktorvater Prof. Dr.-Ing. Eric Sax für die exzellente Betreuung, die sich durch kontinuierliche Unterstützung und vertrauensvollen Austausch auszeichnete. Ich danke ihm dafür, dass er sich stets Zeit genommen hat, meine Anliegen ausführlich zu besprechen, und dass ich aus jedem unserer Gespräche einen klaren Plan sowie wertvolle Impulse mitgenommen habe. Hierdurch konnte ich mich sowohl fachlich als auch persönlich stark weiterentwickeln.

Ebenso möchte ich Prof. Dr.-Ing. Dr. h. c. Michael Weyrich für die Übernahme des Korreferats und seine wertvollen Anmerkungen danken, die zur Weiterentwicklung dieser Dissertation beigetragen haben.

Mein Dank gilt meinen Kollegen am FZI, ITIV und in der Industrie für das gemeinsame Brainstorming, das konstruktive Hinterfragen von Ideen, die anregenden Gespräche in der Kaffeeküche und insgesamt für die sehr angenehme und inspirierende Zusammenarbeit während meiner Promotionszeit. Besonders hervorheben möchte ich hier Lennart Ries, Markus Lemmer, Christian Steinhauser, Philipp Rigoll, Christian King, Barbara Schütt, Hubert Padusinski, Julian Fuchs, Laurenz Adolph, Felix Reisgys, Johannes Plaum und Stefan Otten.

Teile dieser Arbeit entstanden in enger Zusammenarbeit mit der ZF Friedrichshafen AG. Hier möchte ich mich für die jahrelange, sehr gute und produktive Kooperation insbesondere bei Franziska Körtke, Johannes Daube, Lara Turner, Wolfgang Scheffer, Thomas Schulik und Martin Sedlacek bedanken.

Ein weiterer Dank gilt meiner Familie, meinen Eltern Volker und Inge sowie meinem Bruder Kai, die mir stets mit Rat und Tat zur Seite stehen und mich auch während der Promotion jederzeit bestärkt haben. Mein herzlichster Dank gilt schließlich meiner Frau Janina für ihr Verständnis, ihre Geduld und ihre unerschütterliche Unterstützung, mit der sie mich in allen Phasen dieser Arbeit begleitet hat.



# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>1</b>
1.1	Automatisierte Fahrfunktionen	1
1.1.1	Ausgangssituation und Potenziale	1
1.1.2	Definitionen und Klassifikation der Systeme	3
1.1.3	Herausforderung Verifikation und Validierung	3
1.2	Motivation und Forschungsfragen	6
<b>2</b>	<b>Entwicklungsprozesse und Testplattformen</b>	<b>9</b>
2.1	Prozesse für die Entwicklung von Hochautomatisierten Fahrfunktionen	9
2.1.1	Produktenstehungsprozess	9
2.1.2	Musterphasen	10
2.1.3	Reifegradbestimmung	11
2.1.4	V-Modell	12
2.1.5	Verknüpfung von Produktenstehungsprozess, Musterphasen, Reifegraden und dem V-Modell	14
2.1.6	Agile Methoden	16
2.2	Testplattformen	17
2.2.1	Definitionen	17
2.2.2	Gängige Testplattformen	19
<b>3</b>	<b>Stand der Wissenschaft und Technik</b>	<b>23</b>
3.1	Manöver	23
3.1.1	Definition und Einordnung	23
3.1.2	Manöverkataloge	24
3.1.3	Kategorien von Manövern	26
3.1.4	Manöver-Identifikation	28

3.1.5	Anwendung von Manövern im szenariobasierten Testen	30
3.2	Szenariobasiertes Testen . . . . .	31
3.2.1	Begriffsdefinitionen . . . . .	32
3.2.2	Strukturierung und Kategorisierung des Verkehrs und der Umgebung . . . . .	34
3.2.3	Abstraktionsebenen während des Entwicklungsprozesses . . . . .	35
3.2.4	(Un-)bekannte und (un-)sichere Szenarien . . . . .	36
3.3	Der SBT-Prozess . . . . .	38
3.3.1	Erstellen der Szenarien . . . . .	38
3.3.2	Erstellen der Anforderungen . . . . .	39
3.3.3	Ableiten der Testfälle . . . . .	40
3.3.4	Ausführen der Testfälle . . . . .	42
3.3.5	Evaluation der Testfälle . . . . .	43
3.3.6	Sicherheitsargumentation . . . . .	43
3.3.7	Verknüpfung mit dem V-Modell . . . . .	43
3.4	Eigenschaften von Szenarien . . . . .	45
3.4.1	Generieren von Szenarien . . . . .	45
3.4.2	Simulation von Verkehrsszenarien . . . . .	47
3.4.3	Ähnlichkeiten und Distanzen zwischen Szenarien . . . . .	48
3.4.4	Aufgezeichnete und ausführbare Szenarien . . . . .	52
3.4.5	Beschreibungsform, intuitive Nutzbarkeit und automa- tisierte Anwendung . . . . .	53
<b>4</b>	<b>Anforderungen an die Szenariobeschreibungsform . . . .</b>	<b>57</b>
4.1	Ableiten der Anforderungen . . . . .	57
4.2	Abgleich des Stands der Wissenschaft und Technik . . . . .	62
4.2.1	Abgleich der Eigenschaften von Szenarien mit den An- forderungen . . . . .	62
4.2.2	Durchgängigkeit bestehender modellbasierter Beschrei- bungsformen . . . . .	64
<b>5</b>	<b>Konzept des manövergestützten Testens . . . . .</b>	<b>67</b>
5.1	Beitrag dieser Dissertation . . . . .	67
5.1.1	Einordnung . . . . .	67

5.1.2	Grundgedanke des manövergestützten Testens . . . .	67
5.2	Das Vorgehensmodell im manövergestützten Testen . . . .	69
5.2.1	Übersicht des Vorgehensmodells . . . . .	69
5.2.2	Manöver-Identifikation und Szenario-Extraktion . .	70
5.2.3	Ähnlichkeitsorientierte Szenarien-Selektion . . . .	74
5.2.4	Manöver als Anweisungen für die Ausführung von Testszenarien . . . . .	85
5.2.5	Validierung der ausgeführten Szenarien . . . . .	87
5.2.6	Sicherheitsargumentation durch Abdeckung des Sze- narienraums . . . . .	89
5.2.7	Architektur der manöverbasierten Szenariobeschrei- bung . . . . .	90
5.2.8	Zusammenfassung des Vorgehensmodells . . . . .	94
<b>6</b>	<b>Prototypische Umsetzung . . . . .</b>	<b>97</b>
6.1	Umsetzung der Manöver-Identifikation und Szenario-Extraktion	97
6.1.1	Manöver-Identifikation . . . . .	97
6.1.2	Szenario-Extraktion . . . . .	103
6.2	Umsetzung der ähnlichkeitsorientierten Szenarien-Selektion	104
6.2.1	Erstellen logischer Szenarien . . . . .	104
6.2.2	Berechnen der Ähnlichkeiten . . . . .	105
6.2.3	Umsetzung des Szenariographen und des Clusterings	106
6.3	Umsetzung der Ausführung von Testszenarien . . . . .	107
6.4	Umsetzung der Validierung der Testszenarien . . . . .	110
6.5	Verwendete Manöver-, Kontext-, Teilnehmer-, Spur- und Objekt- Typen . . . . .	110
6.5.1	Teilnehmer . . . . .	110
6.5.2	Szenerie . . . . .	111
6.5.3	Manöver . . . . .	112
6.5.4	Kontexte . . . . .	114
<b>7</b>	<b>Evaluation . . . . .</b>	<b>117</b>
7.1	Proof of Concept . . . . .	117
7.1.1	Verwendeter Datensatz . . . . .	117
7.1.2	Manöver-Identifikation und Szenario Extraktion . . .	121

7.1.3	Ähnlichkeitsorientierte Szenarien-Selektion . . . . .	124
7.1.4	Ausführung . . . . .	132
7.1.5	Validierung der ausgeführten Szenarien . . . . .	134
7.1.6	Laufzeit . . . . .	135
7.2	Diskussion zur Abdeckung der Anforderungen . . . . .	136
7.2.1	Anforderungen . . . . .	136
7.3	Limitierungen . . . . .	140
7.4	Diskussion der Forschungsfragen . . . . .	143
7.4.1	1: Durchgängige Beschreibungsform . . . . .	143
7.4.2	2: Herausforderungen im szenariobasierten Testen . . . . .	144
<b>8</b>	<b>Zusammenfassung und Ausblick . . . . .</b>	<b>147</b>
8.1	Zusammenfassung . . . . .	147
8.2	Ausblick . . . . .	149
<b>A</b>	<b>Anhang . . . . .</b>	<b>151</b>
A.1	Algorithmische Grundlagen . . . . .	151
A.1.1	Sequenzalignment . . . . .	151
A.1.2	k-Medoids-Clustering . . . . .	152
A.1.3	Visualisierung und Einbettung von Graphen . . . . .	153
A.2	Manöver-Matrix und Beispielszenarien . . . . .	155
A.3	Beispielszenario zur Manöver-Identifikation . . . . .	158
A.4	Repräsentanten der Beispielszenarien . . . . .	159
	<b>Abkürzungen . . . . .</b>	<b>161</b>
	<b>Abbildungsverzeichnis . . . . .</b>	<b>163</b>
	<b>Tabellenverzeichnis . . . . .</b>	<b>167</b>
	<b>Literaturverzeichnis . . . . .</b>	<b>171</b>
	<b>Eigene Veröffentlichungen . . . . .</b>	<b>189</b>
	<b>Betreute studentische Arbeiten . . . . .</b>	<b>191</b>



# 1 Einleitung

## 1.1 Automatisierte Fahrfunktionen

### 1.1.1 Ausgangssituation und Potenziale

Automatisierte Fahrfunktionen (AF) haben das Potenzial, den Verkehr durch eine verbesserte Verkehrssicherheit, erhöhten Komfort, das Ermöglichen neuer Mobilitätskonzepte und wirtschaftliche Vorteile im Güterverkehr grundlegend zu verändern.

Die Anzahl der Verkehrstoten in Deutschland ist in den vergangenen Jahrzehnten trotz gestiegenen Verkehrsaufkommens kontinuierlich gesunken. Sind 1970 noch über 21.000 Menschen durch Verkehrsunfälle ums Leben gekommen, sind dies 2021 nur noch ca. 2.500 [1]. Die Ursachen dafür liegen in der Einführung neuer Gesetze wie Gurtpflicht, Tempolimit und Promillegrenze, technischen Weiterentwicklungen sowie infrastrukturellen und medizinischen Maßnahmen [2]. Während auf technischer Seite hauptsächlich passive Sicherheitssysteme und nicht automatisierte Systeme, wie Airbags, Antiblockiersystem (ABS) und Elektronisches Stabilitätsprogramm (ESP), (weiter-)entwickelt wurden, richtet sich der Fokus nun auf AF.

Nach Daten des Statistischen Bundesamts wurden 2021 88% der Unfälle mit Personenschaden durch menschliche Fehler verursacht [1]. AF ermöglichen es, diese Unfälle zu reduzieren und damit die Sicherheit auf öffentlichen Straßen weiter zu erhöhen. Laut EU-Kommission können damit bis 2038 25.000 Verkehrstote und mindestens 140.000 Schwerverletzte vermieden werden<sup>1</sup>. Um dieses Ziel zu erreichen, sind in der EU einige AF seit 2022 für die Zulassung neuer Fahrzeugtypen vorgeschrieben [3]. Dazu gehören z.B. der Notbremsassistent und der Notfall-Spurhalteassistent. AF mit höherem Automatisie-

---

<sup>1</sup> [https://ec.europa.eu/commission/presscorner/detail/en/IP\\_22\\_4312](https://ec.europa.eu/commission/presscorner/detail/en/IP_22_4312)

rungsgrad (s. Abschnitt 1.1.2) bieten zukünftig weiteres Potenzial Unfälle zu vermeiden [2].

AF erhöhen den Komfort des Fahrers bzw. des Insassen, indem sie ihn unterstützen und Teile der Fahraufgabe übernehmen. Wird die Fahraufgabe komplett an eine AF übergeben, kann sich der Fahrer währenddessen anderen Tätigkeiten widmen, wie Nachrichten schreiben, im Internet surfen oder Filme schauen. 2021 wurde mit dem Mercedes-Benz DRIVE PILOT das erste System freigegeben, das unter bestimmten Umgebungsbedingungen auf der Autobahn die Fahraufgabe vollständig übernimmt<sup>2</sup>.

Der motorisierte Individualverkehr machte 2019 ca. 83% des Gesamtverkehrs aus<sup>3</sup>. Doch vor allem im urbanen Bereich steht er zunehmend in der Kritik: Zugeparkte Flächen und Stau durch die ineffiziente Flächeninanspruchnahme [4] sowie ausgestoßene Schadstoffe<sup>4</sup> und Treibhausgase<sup>4,3</sup> führen zur Suche nach neuen Mobilitätskonzepten und dem Ausbau des öffentlichen Nahverkehrs. Ist die Distanz zwischen dem eigentlichen Aufenthaltsort und dem nächsten Zugangspunkt zum öffentlichen Nahverkehr zu lang, ist häufig dennoch das eigene Auto das Verkehrsmittel der Wahl<sup>5</sup>.

Die Integration von fahrerlosen Fahrzeugen in den öffentlichen Nahverkehr trägt dazu bei, dieses Problem der sogenannten ersten und letzten Meile zu lösen [5]. Fahrerlose Shuttles können als On-Demand-Fahrzeuge gebucht oder auf Routen eingesetzt werden, auf denen aus wirtschaftlichen Gründen kein klassischer öffentlicher Nahverkehr möglich ist. In Deutschland sind inzwischen über 60 solcher Shuttles als Pilotprojekte geplant oder im Einsatz<sup>6</sup>.

Im Güterverkehr können AF dem aufkommenden Fahrermangel entgegenwirken [6] und zu wirtschaftlichen Vorteilen führen. Die Personalkosten lassen sich durch fahrerlose Fahrzeuge um bis zu 90% reduzieren [7], bei den Gesamtkosten wird eine Reduktion um bis zu 35% prognostiziert [6, 8].

Im Fernverkehr sind bereits Testflotten von fahrerlosen LKW auf Autobahnen

---

<sup>2</sup> <https://group.mercedes-benz.com/innovation/produktinnovation/autonomes-fahren/systemgenehmigung-fuer-hochautomatisiertes-fahren.html>

<sup>3</sup> <https://www.dlr.de/content/de/artikel/forschung/personenverkehr-in-deutschland-verkehrsmittel-im-vergleich.html>

<sup>4</sup> <https://www.umweltbundesamt.de/bild/vergleich-der-durchschnittlichen-emissionen-0>

<sup>5</sup> <https://blog.tkelevator.com/de/mobilitat-der-zukunft-auf-der-letzten-meile/>

<sup>6</sup> <https://www.vdv.de/liste-autonome-shuttle-bus-projekte.aspx>

in den USA unterwegs<sup>7</sup>. Im Nahverkehr wurde 2021 die erste kommerzielle Lizenz für Fahrten ohne Sicherheitsfahrer auf einer festgelegten Route von 11 Kilometern erteilt<sup>8</sup>. Für den Personentransport wurde 2022 in San Francisco die erste kommerzielle Lizenz für fahrerlose Taxis gewährt<sup>9</sup>. Weitere Pilotprojekte laufen u.a. in mehreren Städten in den USA<sup>10</sup>, Paris<sup>10</sup>, München<sup>10</sup> und Seoul<sup>11</sup>.

### 1.1.2 Definitionen und Klassifikation der Systeme

**Definition 1.1 (ODD).** *Die **Operational Design Domain (ODD)** sind Betriebsbedingungen, unter denen ein bestimmtes Fahrautomatisierungssystem oder ein Merkmal davon funktionieren soll, einschließlich, aber nicht beschränkt auf umweltbedingte, geografische und tageszeitliche Einschränkungen und/oder das erforderliche Vorhandensein oder Nichtvorhandensein bestimmter Verkehrs- oder Fahrbahnmerkmale. [9]*

Die SAE J3016 [9] unterteilt AF nach dem Grad ihrer Automatisierung in die Stufen 0 bis 5 (s. Abbildung 1.1). Auf Stufe 0 ist keine Automatisierung vorhanden, auf den Stufen 1 und 2 hat der Fahrer die volle Verantwortung für das Fahren und wird durch Fahrerassistenzsysteme (FA) unterstützt. Auf Stufe 3 wird die Fahraufgabe innerhalb einer ODD von der Fahrfunktion übernommen, der Fahrer muss beim Verlassen der ODD das Steuer übernehmen. Ab Stufe 4 fällt der Mensch als Rückfallebene weg und Hochautomatisierte Fahrfunktionen (HAF) übernehmen die Fahraufgabe vollständig.

### 1.1.3 Herausforderung Verifikation und Validierung

Trotz der vielseitigen Vorteile von HAF (s. Abschnitt 1.1.1) und den bereits erreichten technischen Fortschritten sind HAF bisher nur in eng beschränk-

---

<sup>7</sup> <https://torc.ai/arizona-self-driving-trucks-torc/>

<sup>8</sup> <https://www.autoweek.com/news/technology/a38198243/walmart-autonomous-delivery-trucks-gatik/>

<sup>9</sup> <https://docs.cpuc.ca.gov/PublishedDocs/Published/G000/M481/K896/481896441.PDF>

<sup>10</sup> <https://mobility-talk.com/erste-robotaxis-deuschlands-fahren-in-muenchen>

<sup>11</sup> <https://42dot.ai/tap/locations>



## SAE J3016™ LEVELS OF DRIVING AUTOMATION™

Learn more here: [sae.org/standards/content/j3016\\_202104](https://sae.org/standards/content/j3016_202104)

Copyright © 2021 SAE International. The summary table may be freely copied and distributed AS-IS provided that SAE International is acknowledged as the source of the content.

	SAE LEVEL 0™	SAE LEVEL 1™	SAE LEVEL 2™	SAE LEVEL 3™	SAE LEVEL 4™	SAE LEVEL 5™
What does the human in the driver's seat have to do?	You <b>are driving</b> whenever these driver support features are engaged – even if your feet are off the pedals and you are not steering			You are <b>not driving</b> when these automated driving features are engaged – even if you are seated in “the driver's seat”		
	You must constantly supervise these support features; you must steer, brake or accelerate as needed to maintain safety			When the feature requests, you must drive	These automated driving features will not require you to take over driving	
Copyright © 2021 SAE International.						
	These are driver support features			These are automated driving features		
What do these features do?	These features are limited to providing warnings and momentary assistance	These features provide steering <b>OR</b> brake/acceleration support to the driver	These features provide steering <b>AND</b> brake/acceleration support to the driver	These features can drive the vehicle under limited conditions and will not operate unless all required conditions are met	This feature can drive the vehicle under all conditions	
	<ul style="list-style-type: none"><li>• automatic emergency braking</li><li>• blind spot warning</li><li>• lane departure warning</li></ul>	<ul style="list-style-type: none"><li>• lane centering <b>OR</b> adaptive cruise control</li></ul>	<ul style="list-style-type: none"><li>• lane centering <b>AND</b> adaptive cruise control at the same time</li></ul>	<ul style="list-style-type: none"><li>• traffic jam chauffeur</li></ul>	<ul style="list-style-type: none"><li>• local driverless taxi</li><li>• pedals/steering wheel may or may not be installed</li></ul>	<ul style="list-style-type: none"><li>• same as level 4, but feature can drive everywhere in all conditions</li></ul>

Abbildung 1.1: Automatisierungsgrade des automatisierten Fahrens nach SAE J3016 [9]

ten ODDs unterwegs. Nur 62% der Deutschen können sich aktuell vorstellen, in einen fahrerlosen PKW, Shuttle oder Taxi zu steigen<sup>12</sup>. Größte Bedenken bei der Einführung von HAF sind die Angst vor technischem Versagen und der Zweifel an der technischen Zuverlässigkeit [8]. Angesichts dieser gesellschaftlichen Bedenken über die Auswirkungen von HAF muss nachgewiesen werden, dass solche Systeme sicherer sind als menschliche Fahrer [10].

Die Verifikation und Validierung (V&V) von HAF birgt mehrere Herausforderungen: Fahrerlose Fahrzeuge sind insofern ein Novum, als sie sicherheitskritische Systeme sind, die sich im Open-World Kontext bewegen. Durch den

<sup>12</sup> <https://www.bitkom-research.de/de/pressemitteilung/wachsendes-vertrauen-autonome-fahrzeuge>

Verlust des Fahrers als Rückfallebene muss nachgewiesen werden, dass HAF auch in selten vorkommenden und kritischen Situationen die richtigen Entscheidungen treffen und unfallfrei durch den Verkehr navigieren können. Weil der Verkehr aber keine abgeschlossene Umgebung ist, ist es a priori nicht möglich, alle dieser kritischen Situationen zu kennen und zu testen.

Zusätzlich führen die steigende Funktionalität von HAF und die Erweiterung der ODD zu zunehmend größer werdender Software im Auto, sodass über 10 Millionen Zeilen Programmcode in einem Auto stecken [11]. In einem modernen Auto sind weitergehend bis zu 150 miteinander interagierende Steuergeräte<sup>13</sup> mit unterschiedlichen Mikrocontrollern, Betriebs- und Bussystemen verbaut [11]. Die Menge und die Verknüpfung von Software- und Hardware-Elementen führen zu etlichen Fehlerpotenzialen, die durch Testen identifiziert werden müssen. Dabei steigen mit dieser wachsenden Systemkomplexität auch die Anforderungen an kontinuierliches und automatisiertes Testen [12].

Diese Kombination aus Open-World-Kontext, Erweiterung der ODD und ansteigender Menge an Software- und Hardware im Fahrzeug führt dazu, dass bisherige Testmethoden aus der Fahrzeugentwicklung nicht ausreichend sind, um die Herausforderungen der V&V von HAF vollständig zu lösen:

Beim anforderungsbasierten Testen, dem Standardtestverfahren für die Fahrzeugentwicklung [11], werden Testfälle anhand der Systemspezifikation erstellt und in Testkatalogen gesammelt. Als Richtlinien gelten dabei: “Kein Test ohne Anforderung” und “Zu jeder Anforderung mindestens ein Test” [13]. Somit entsteht eine Systematik beim Erstellen der Testfälle, die sicherstellt, dass alle Anforderungen mindestens einmal getestet werden. Hervorzuheben ist, dass die Systemspezifikation der Ausgangspunkt dieser Testfälle ist. Dadurch können aber nur bereits bekannte Risiken berücksichtigt werden [14]. Für die Absicherung von HAF, welche in dem beschriebenen Open-World-Kontext operieren und daher auch bisher unbekannte Situationen meistern müssen, ist das anforderungsbasierte Testen daher nicht ausreichend.

Eine Alternative bietet die distanzbasierte Validierung von HAF. Dabei wird die Sicherheit über einen statistischen Nachweis mit einer zugehörigen Konfidenz gezeigt. Wachenfeld et al. [15] und Kalra et al. [16] schätzen in ihren Arbeiten die notwendigen Testumfänge, um nachzuweisen, dass ein HAF weniger tödliche Unfälle als ein menschlicher Fahrer verursacht. Als Grundlage

---

<sup>13</sup> <https://www.ept.ca/2019/05/sales-of-automotive-electronic-control-units-soar/>

wird angenommen, dass das Auftreten sicherheitskritischer Situationen der Poisson-Verteilung unterliegt. Mit unterschiedlichen Annahmen ergeben sich bei den Schätzungen 2,3 Milliarden [15] bzw. 11 Milliarden [16] Testkilometer. Auch wenn sich die Ergebnisse unterscheiden, ist die Aussage, dass Testkampagnen in dieser Größenordnung ökonomisch nicht leistbar sind: Für ein neues Fahrzeugmodell werden ca. 5.000 Prototypen gebaut. Wenn 1.000 davon täglich 1.000 km für den Sicherheitsnachweis fahren, werden selbst für die optimistischeren Annahmen 2.300 Tage bzw. über 6 Jahre benötigt. Zudem muss bei jeder Änderung des Systems, z. B. nach einem behobenen Fehler, die Testkampagne erneut gestartet werden.

Shalev et al. [17] schlagen alternativ eine formale Verifikation der Fahrfunktion vor. Dabei werden mithilfe von statischen Tests Grundregeln wie “Ein Unfall muss verhindert werden, solange dadurch kein anderer Unfall verursacht wird” oder “Vorfahrt wird gewährt, nicht genommen” formal nachgewiesen. Die Sensorik wird bei diesem Vorgehen getrennt von der Fahrfunktion statistisch abgesichert. Ansätze für die formale Verifikation wurden bereits in der Robotik verfolgt [18]. Die erfolgreiche Übertragung auf den Open-World-Kontext von HAF wurde aber bisher nicht durchgeführt.

Insbesondere bei Interaktionen zwischen Verkehrsteilnehmern im urbanen Raum sind daher Verifikations- und Validierungsstrategien noch in der Entwicklung. Aktuelle Forschungsprojekte und die Industrie fokussieren dabei das szenariobasierte Testen (s. Abschnitt 3.2). Dabei werden die Fahrscenarien in (un-)bekannte und (un-)sichere Szenarien eingeteilt. Für die Sicherheitsargumentation der HAF wird dann nachgewiesen, dass die Zahl an unsicheren und unbekannten Szenarien ausreichend klein ist. Eine intelligente Testauswahl, um geeignete Fahrscenarien zu identifizieren, reduziert dabei aus statistischer Sicht die zum Testen erforderlichen Kilometer [19].

## 1.2 Motivation und Forschungsfragen

Trotz des Potenzials des szenariobasierten Testens ist die Methode bisher nicht ausgereift und es gibt noch einige offene Punkte, die für eine breite Anwendung in der Industrie gelöst werden müssen. So befassen sich bisherige Arbeiten entweder mit dem Gesamtprozess auf einer abstrakten Ebene oder fokussieren sich auf einzelne Problemstellungen ohne Berücksichtigung der sonstigen Aufgaben. Dies führt zu einer Vielzahl von Insellösungen, die nicht miteinan-

der verknüpft sind und eine nahtlose Integration in den Entwicklungsprozess erschweren.

Eine wesentliche Lücke ist eine fehlende Harmonisierung im Bereich der Szenariobeschreibung, wodurch die Effizienz der gesamten Methode verringert wird. Die fehlende Verknüpfung äußert sich vorrangig durch unterschiedliche Arten der Beschreibung von Szenarien, die jeweils auf das isolierte Problem zugeschnitten sind. Die Motivation der Arbeit ist es daher, eine durchgängige Methode zu entwickeln, die eine einheitliche Form der Szenarien im gesamten Prozess unterstützt. Dabei werden die Aspekte des automatisierten und manuellen Aufbaus eines Szenarienkatalogs sowie das Auswählen, Ausführen und Evaluieren der Testszenarien genauer untersucht.

Dies führt zu der ersten Forschungsfrage:

**Forschungsfrage 1.** *Wie lassen sich Verkehrsszenarien für das szenariobasierte Testen formalisiert beschreiben, um durchgängig im V&V-Prozess genutzt werden zu können und dessen Effizienz zu steigern?*

Anschließend ergibt sich mit den zentralen Herausforderungen des szenariobasierten Testens (s. Abschnitt 3.2) die zweite Forschungsfrage:

**Forschungsfrage 2.** *Wie lassen sich die vier zentralen Herausforderungen des szenariobasierten Testens mithilfe der Ergebnisse von Forschungsfrage 1 meistern?*





## **2 Entwicklungsprozesse und Testplattformen**

### **2.1 Prozesse für die Entwicklung von Hochautomatisierten Fahrfunktionen**

#### **2.1.1 Produktenstehungsprozess**

Der Produktenstehungsprozess (PEP) ist ein Geschäftsprozess, der die Arbeitsabläufe von der Idee über das Design und Testen bis zur Produktion und den Vertrieb eines Produkts beinhaltet. Er prägt maßgeblich die Entwicklung von Produkten im Automotive-Bereich und die Zusammenarbeit zwischen Herstellern und Lieferanten. Während die übergeordneten Phasen Konzeptentwicklung, Produktentwicklung, Produktionsvorbereitung und Serienproduktion (s. Abbildung 2.1) ein etabliertes Muster im PEP sind, unterscheiden sich die detaillierten Ausprägungen und Begriffe von Unternehmen zu Unternehmen und sind aus Wettbewerbsgründen nicht öffentlich [20]. Eine Übersicht und ein Vergleich unterschiedlicher PEPs in der Literatur, auch außerhalb des Automotive-Bereichs, findet sich in [21], konkrete Beispiele für die Fahrzeugentwicklung in [20, 22, 23].

Im Weiteren wird der PEP nach dem Verband der Automobilindustrie e. V. (VDA) [22] und Kerber [21] betrachtet (s. Abbildung 2.1): In der Konzeptentwicklung wird ein technisch abgesichertes Produktkonzept entwickelt, welches die Grundlage für die weitere Ausgestaltung ist. Während der Produktentwicklung werden die technischen Spezifikationen festgelegt. Die Produktentwicklung wird üblicherweise durch den Bau von Prototypen begleitet und es werden entwicklungsbegleitende Tests durchgeführt (s. Abschnitt 2.1.5). Während der Produktionsvorbereitung wird die Infrastruktur zur Produktion geplant und vorbereitet. In der Serienproduktion findet die Produktfreigabe

statt, anschließend ist der Start of Production (SOP) und das Produkt wird der Serienbetreuung übergeben.

### 2.1.2 Musterphasen

Bis zum SOP durchläuft die Entwicklung vier Musterphasen: Das A-, B-, C- und D-Muster. Die Muster dienen "zur Bestätigung des Funktionsprinzips, zur Überprüfung von Bauraum und Montage-Fähigkeit, der Erprobung und zur Erreichung der technischen Freigabe" [22] und sind durch folgende Charakteristiken definiert [22]:

A-Muster repräsentieren erste, bedingt fahrtaugliche Funktionsmuster, die zur Bestätigung des Funktionsprinzips und zur grundlegenden Funktionsdarstellung dienen. Sie sind bei fahrsicherheitsrelevanten Funktionen nicht im öffentlichen Straßenverkehr zulässig und werden daher hauptsächlich im Labor, in der Simulation und auf dem Prüfstand genutzt. A-Muster ermöglichen keine Lebensdaueraussagen.

Im Unterschied dazu sind B-Muster funktionsfähige und fahrtaugliche Prototypen, die eine ausreichende Betriebssicherheit für erste Erprobungsphasen gewährleisten. Die Nutzung im öffentlichen Straßenverkehr ist mit einer Sonderzulassung möglich. Dadurch können die technischen Anforderungen des gesamten Funktionsumfangs abgesichert werden. Eine Dauererprobung ermöglicht zudem das Ermitteln und Beseitigen von verbliebenen Schwachstellen. Alle Hardware- und Funktionsanforderungen sind erfüllt, auch wenn die Muster bisher nicht in allen Punkten serientauglich sind. Die Herstellung des B-Musters erfolgt nicht unter Serienbedingungen und -prozessen, zudem sind Einzelteile oder Komponenten mit Prototypenwerkzeugen hergestellt.

Die Serienwerkzeuge werden ab dem C-Muster für die Herstellung der Komponenten unter seriennahen Bedingungen verwendet. C-Muster sind, wie B-Muster, mit einer Sonderzulassung im öffentlichen Straßenverkehr nutzbar und werden verwendet, um das Erreichen der technischen Freigabe zu erproben. Es sind keine technischen Einschränkungen zugelassen, die Auswirkungen auf die Fahrsicherheit haben. Die Prüfung der Komponenten erfolgt mit Serienprüfmitteln oder gleichwertigen Absicherungsmaßnahmen.

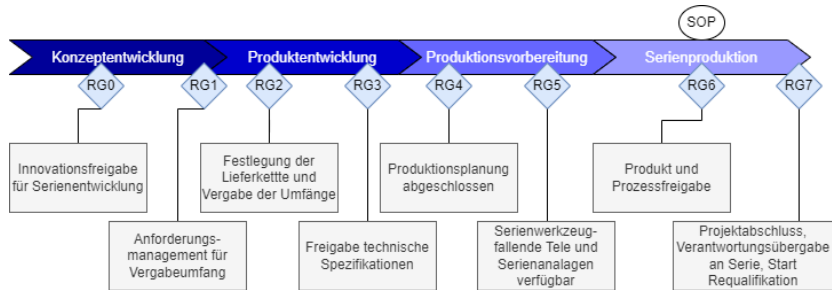


Abbildung 2.1: Der Produktentstehungsprozess, die zugehörigen Phasen und mögliche Reifegrade nach VDA [22]

Bei den D-Mustern erfolgt die Fertigung vollständig mit Serienwerkzeugen und unter Serienbedingungen. Die Serienfunktionalität ist vollständig vorhanden und erfüllt alle Spezifikationsanforderungen. Alle Qualitätsanforderungen werden gleichbleibend sichergestellt. Mit D-Mustern wird die Produktionsprozessfreigabe und Produktfreigabe durchgeführt.

### 2.1.3 Reifegradbestimmung

Um Qualitätsproblemen im PEP vorzubeugen, ist eine kontinuierliche Schätzung und präventive Erkennung von Qualitätsrisiken notwendig.

Für diese fortlaufende Beurteilung der Projektreife definiert der VDA das Reifegrad-Absicherungs-Modell [22] mit acht Reifegraden, RG0 bis RG8, die sich zeitlich in den PEP einordnen (s. Abbildung 2.1).

**Definition 2.1** (Reifegrad). Ein *Reifegrad* bestimmt die Projektreife über quantifizierbare Kenngrößen (Reifegrad-Messkriterien) zu einem vereinbarten Zeitpunkt.

Das Bestimmen der Reifegrade während der Entwicklung liefert Updates über die Qualität des Produkts. Diese bilden eine Grundlage, um eventuell notwendige Maßnahmen zur Qualitätssicherung zu identifizieren und durchzuführen.

Die Bewertung der Reifegrade erfolgt über einen Satz an spezifischen Messkriterien, welche mit eindeutig "Ja" oder "Nein" beantwortet werden können. Jedes Messkriterium wird über eine Ampellogik (rot, gelb, grün) bewertet: Bei rot wird das Messkriterium mit "Nein" beantwortet, mindestens ein Projektziel ist nicht erreichbar und eine Maßnahme zur Zielanpassung ist notwendig. Bei gelb wird das Messkriterium ebenfalls mit "Nein" beantwortet, eine Maßnahme ist erforderlich und vereinbart und alle Projektziele werden mit den festgelegten Maßnahmen erreicht. Bei grün wird das Messkriterium mit "Ja" beantwortet und keine zusätzlichen Aktivitäten sind notwendig.

### 2.1.4 V-Modell

Das V-Modell ist ein Entwicklungsprozess, der Anfang der 1990er-Jahre vom Bundesministerium des Inneren für die Softwareproduktion veröffentlicht wurde [24]. In der Automobilentwicklung ist das V-Modell der am weitesten verbreitete Referenzprozess zur Entwicklung, Validierung und Verifikation von Systemen [13]. Frühe Versionen des V-Modells wurden von Boehm et al. [25] und Rook et al. [26] veröffentlicht. Weitläufig genutzt und Kern des aktuellen Entwicklungsprozesses ist das V-Modell 97 [24], das zusätzlich zur Software auch Prozesse für die Hardware-Entwicklung beinhaltet. Über die Software- bzw. Hardwareentwicklung hinaus betrachtet das Modell auch Projektmanagement, Qualitätssicherung und Konfigurationsmanagement.

Das V-Modell teilt sich in zwei Äste auf (s. Abbildung 2.2): Auf dem linken Ast wird das System in mehreren Phasen im Top-Down-Prinzip entworfen und anschließend implementiert. In einer Analyse werden die Anforderungen der Kunden definiert und festgelegt, was das fertige Produkt können soll. Im anschließenden Systementwurf wird das Gesamtsystem in Subsysteme aufgeteilt und deren Schnittstellen spezifiziert. Für diese werden jeweils eigene Anforderungen definiert. Die anschließenden Grobentwurf- und Feinentwurf-Phasen unterteilen diese Subsysteme weiter in kleinere Module und Units, die in der Implementierung erstellt werden.

**Definition 2.2.** *Validierung ist der Prozess zur Beurteilung eines Systems oder einer Komponente mit dem Ziel festzustellen, ob der Einsatzzweck oder die Benutzererwartungen erfüllt werden [27].*

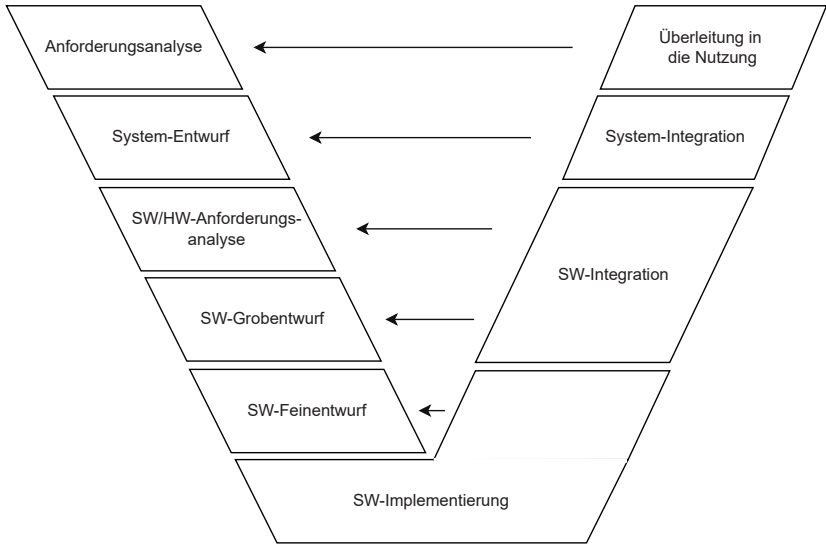


Abbildung 2.2: Vereinfachte Darstellung des V-Modells basierend auf [24]

**Definition 2.3. Verifikation** ist der Prozess zur Beurteilung eines Systems oder einer Komponente, mit dem Ziel festzustellen, ob die Resultate einer gegebenen Entwicklungsphase den Vorgaben für diese Phase entsprechen [27].

Auf dem rechten Ast wird die erstellte Software zum Gesamtsystem integriert, verifiziert und validiert. Die Tests und die Integration werden, im Kontrast zum rechten Ast, Bottom-Up realisiert. Dabei werden Prüfschritte auf der entsprechenden Abstraktionsebene, also horizontal im V-Modell, umgesetzt: Die Software-Implementierung wird dem zugehörigen Feinentwurf gegenübergestellt. Abschließend existiert ein Akzeptanztest, der validiert, ob das System den in den Kundenanforderungen festgelegten Spezifikationen genügt.

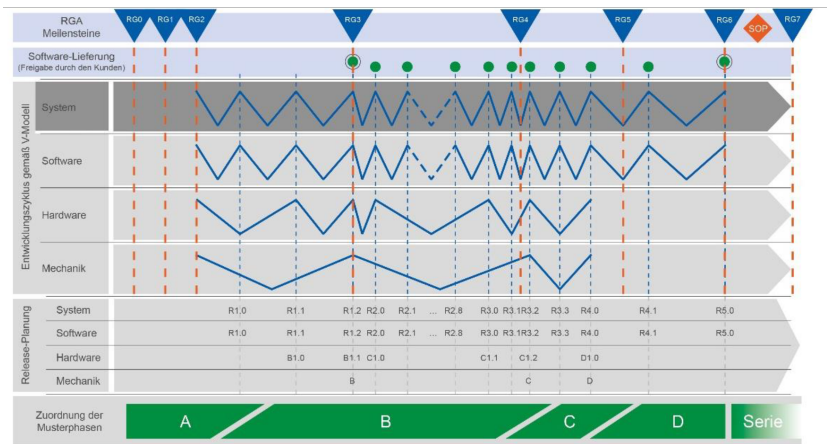


Abbildung 2.3: Beziehungen zwischen PEP, Musterphasen, Reifegraden und V-Modell mit beispielhafter Taktung von Hardware, Software und Mechanik [22]

### 2.1.5 Verknüpfung von Produktenstehungsprozess, Musterphasen, Reifegraden und dem V-Modell

Während der Entwicklung, die sich über mehrere Jahre erstreckt [13], werden mehrere Stufen an Prototypen bzw. Mustern mit zugehörigen Tests benötigt, um die Reifegrade zu erreichen und zu validieren. Jede Prototypen-Erstellung ist wiederum eine verkleinerte Produktentwicklung, die alle Stufen des V-Modells durchläuft. Das V-Modell in der Automobilentwicklung beinhaltet somit viele verkleinerte, untergeordnete V-Modelle [20], die bei Abschluss einen neuen Release erzeugen.

Diese untergeordneten V-Modelle unterteilen sich in die Bereiche Software, Hardware und Mechanik und gliedern sich sequenziell in den PEP und die Reifegradbestimmung ein (s. Abbildung 2.3). Dies führt dazu, dass Tests schon entwicklungsbegleitend für die Releases und zum Erreichen der Reifegrade und nicht erst zur Freigabe des Produkts durchgeführt werden.

Die Software hat dabei die höchste Taktung an Releases, die Mechanik die niedrigste. Der steigende Softwareanteil im Automobil (s. Abschnitt 1.1.3) führt damit auch zu einem überproportionalen Anstieg an Tests, da Software-

tests häufiger durchgeführt werden als Tests der Hardware oder Mechanik. Ab dem D-Muster sind keine Entwicklungsaktivitäten mehr für Hardware und Mechanik geplant, die Software wird bis zum Serienstart optimiert.

Die Synchronisierung der Entwicklung in den unterschiedlichen Bereichen wird zu definierten Zeitpunkten durch die Musterstände (s. Abschnitt 2.1.2) erreicht. Somit werden die einzelnen Komponenten des Fahrzeugs schon während der Entwicklung aufeinander abgestimmt und Fehler und Probleme bei Schnittstellen werden möglichst früh erkannt.

Zusätzlich zu den Tests, in denen spezifisch Komponenten und Funktionen unter kontrollierten Bedingungen mit definierten Testeingängen und erwartbaren Ergebnissen geprüft werden, finden auch Erprobungsfahrten statt.

**Definition 2.4.** *Die **Erprobung** ist eine Analyse des Gesamten, der eine vergleichsweise geringe Systematik zugrunde liegt. In Erproben steckt das Ausprobieren in einer unbestimmten Situation ([13], zitiert nach [28]).*

Wegen der geringen Systematik und den unbestimmten Situationen besitzen Erprobungsfahrten auch eine höhere Redundanz an ähnlichen Fahrsituationen [29].

Erprobungen finden in unterschiedlichen Varianten statt. Eine Art ist die Dauererprobung, in der das Testobjekt durch Langzeitbelastungstests auf seine Zuverlässigkeit geprüft wird. Winter- und Sommererprobung wiederum finden unter, teilweise extremen, klimatischen Umgebungsbedingungen statt, um deren Einflüsse auf das Testobjekt zu beurteilen.

Im V-Modell (s. Abschnitt 2.1.4) ordnen sich Erprobungsfahrten auf dem rechten Ast bei den Validierungs- und Verifikations-Aktivitäten ein. Erste Erprobungsfahrten werden ab dem B-Muster (s. Abschnitt 2.1.2) durchgeführt, da hier eine Straßenzulassung vorhanden ist. Ab dann werden die Erkenntnisse aus der Erprobung durchgängig bis zur finalen Freigabe genutzt, um das Fahrzeug zu validieren und zu verifizieren.

## 2.1.6 Agile Methoden

Der wachsende Anteil an Software im Fahrzeug (s. Abschnitt 1.1.3) führt auch dazu, dass weitere Prozesse aus der Software-Entwicklung in der Fahrzeugentwicklung etabliert werden. Dazu gehören die Agilen Methoden, die durch vergleichsweise kurze Entwicklungszyklen auf sich ändernde Anforderungen reagieren können. Damit sehen sie sich als Gegenbewegung zu den traditionellen, eher „schwergewichtigen“ Entwicklungsprozessen, wie z.B. dem V-Modell [30]. Die meisten der Methoden bauen auf dem “Agile Manifest” von 2001 [31] auf, das vier Werte und zwölf Prinzipien für die agile Softwareentwicklung vorstellt. Die populärste Ausgestaltung des Manifests in ein Framework zur Produktentwicklung ist Scrum [32], in dem feste Rollen, die Teamgröße, die Dauer eines Entwicklungszyklus von zwei bis vier Wochen, genannt Sprint, und regelmäßige, zweckgebundene Meetings innerhalb des Sprints formalisiert werden.

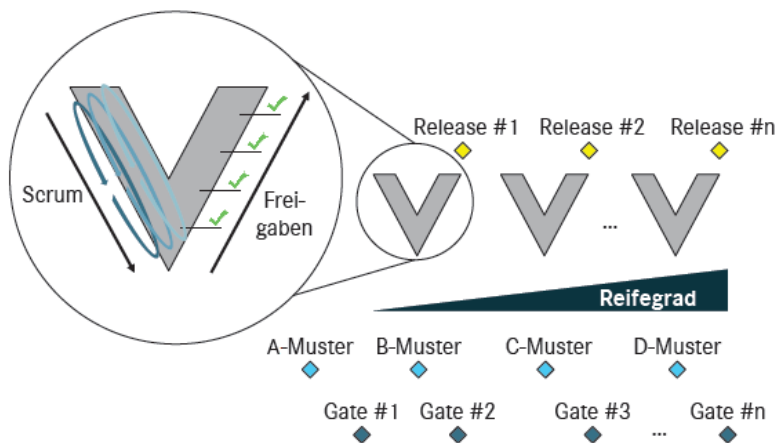


Abbildung 2.4: Mögliche Integration von agiler Softwareentwicklung ins V-Modell nach Reisgys et al. [33]

Da Agile Methoden wie Scrum ursprünglich für einzelne Teams und eigenständige Software-Projekte ohne Schnittstellen entwickelt wurden, lassen sie sich nicht uneingeschränkt auf die Fahrzeugentwicklung übertragen [34, 35]. Für die Hardware und Mechanik ist z.B. die Entwicklung funktionsfähiger Bau-



gruppen in Sprints von maximal vier Wochen kaum möglich [36]. Um Agile Methoden in diesen Domänen dennoch anzuwenden, ist eine Verlängerung der Sprint-Dauer und eine Synchronisierung der Soft- und Hardware-Releases notwendig. Zudem muss die Qualitätssicherung sicherheitskritischer Systeme berücksichtigt werden [34]. Um Elemente der Agilen Methoden in der Fahrzeugentwicklung zu verwenden, werden sie in die bereits bestehenden Entwicklungsprozesse integriert (s. Abbildung 2.4) [34, 33]. Z.B. können einzelne Releases (s. Abbildung 2.3) innerhalb eines Sprints agil entwickelt und in das V-Modell bzw. den PEP eingegliedert werden.

Für die Koordination mehrerer Entwicklungsteams, wie sie für die Fahrzeugentwicklung benötigt werden, existieren Erweiterungen des klassischen Scrum-Frameworks, wie z.B. Scrum@Scale [37], Large Scale Scrum (LeSS) [38] und Scaled Agile Framework (SAFe) [39].

Die Anwendungen von schnelleren Iterationszyklen haben auch Auswirkungen auf den Testaufwand, da idealerweise am Ende jedes Zyklus Tests durchgeführt werden. Die Zahl der entwicklungsbegleitenden Tests steigt also zusätzlich durch die Anwendung der Agilen Methoden an.

## 2.2 Testplattformen

### 2.2.1 Definitionen

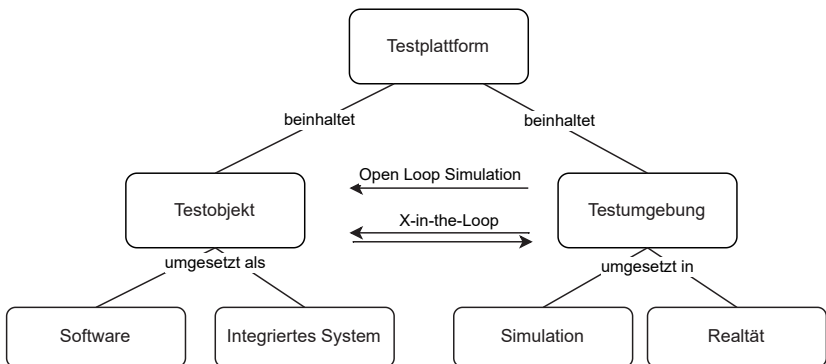


Abbildung 2.5: Überblick der relevanten Begriffe und Beziehungen für Testplattformen

Zur Verifikation und Validierung eines Testobjekts lassen sich Tests auf verschiedenen Plattformen durchführen.

**Definition 2.5** (Testplattform, -objekt und -umgebung). *Auf einer **Testplattform** wird ein **Testobjekt**<sup>1</sup> durch eine **Testumgebung** stimuliert, mit dem Ziel, die Funktionalität des Testobjekts nachzuweisen.*

Das Testobjekt ist als reine Software oder als integriertes System auf der Ziel-Hardware umgesetzt (s. Abbildung 2.5) [13]. Je nach Entwicklungsphase ist das Testobjekt eine Unit, ein oder mehrere Subsysteme oder das Gesamtsystem (s. Abschnitt 2.1.4). Die Testumgebung ist entweder die Realität oder wird durch eine Simulation abgebildet (s. Abbildung 2.5) [13].

Werden Informationen nur von der Testumgebung an das Testobjekt gesendet, wird dies als Open-Loop-Simulation bezeichnet [40]. Beeinflussen sich Testumgebung und Testobjekt durch eine zusätzliche Rückführung, wird dies als X-in-the-Loop (XiL) bezeichnet [41] (s. Abbildung 2.6), wobei das X die Art repräsentiert, in der das Testobjekt eingebunden ist (s. Abschnitt 2.2.2). Die Erfassung der Umgebung findet über Sensoren statt, über Aktoren beeinflusst es die Umgebung wie z. B. andere Verkehrsteilnehmer. Vorteile der XiL-Plattformen im Vergleich zu realen Testfahrten sind die Reduktion von Aufwänden für die Implementierung und das Ausführen von Testfällen, ein verringertes Unfallrisiko durch Einsatz virtueller Umgebungen sowie die Reproduzierbarkeit der Testfälle [42]. Die maßgebliche Herausforderung ist die Sicherstellung einer ausreichenden Glaubwürdigkeit der simulierten Elemente, die für belastbare Testergebnisse notwendig ist [42].

Die Testplattformen lassen sich nach Sax et al. [13] anhand der Umsetzung des Testobjekts und der Umgebung in Realität und Simulation kategorisieren (s. Abbildung 2.7). Jede dieser Testplattformen hat spezifische Vor- und Nachteile. Die Auswahl bzw. Kombination der Testplattformen hat somit einen Einfluss auf die Qualität der Testergebnisse sowie den Aufwand für das Erstellen und Durchführen der Tests. Einen detaillierten Überblick über mögliche Plattfor-

---

<sup>1</sup> In der Literatur sind unterschiedliche Begriffe für das Testobjekt gebräuchlich. Dazu zählen u.a. System under Test (SuT), Device under Test (DuT) und Unit under Test (UuT). Im Rahmen dieser Thesis wird der Begriff Testobjekt verwendet.

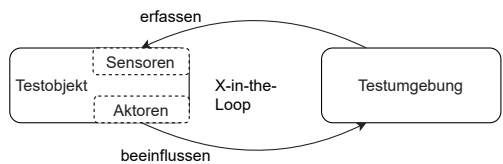


Abbildung 2.6: Gegenseitiger Einfluss von Testobjekt und Testumgebung bei X-in-the-Loop Plattformen

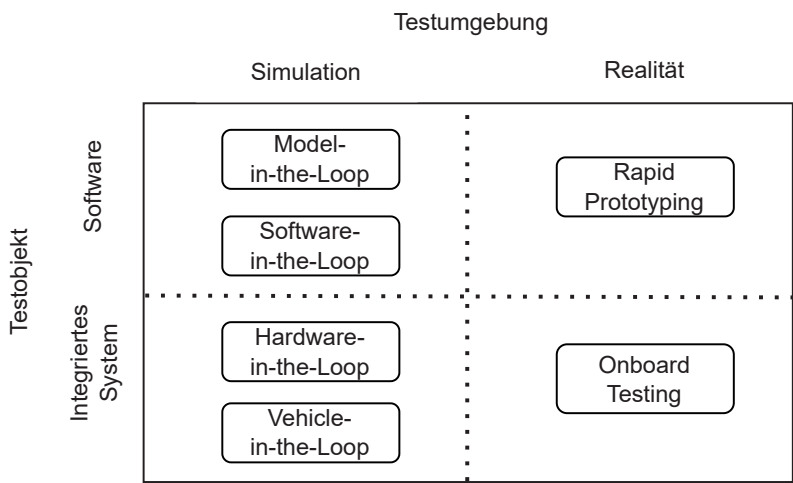


Abbildung 2.7: Kategorisierung der Testplattformen über die Umsetzung der Testumgebung und des Testobjekts basierend auf Sax et al. [13]

men, deren Kategorisierung und Eigenschaften findet sich in King et al. [43] und Steimle et al. [44].

### 2.2.2 Gängige Testplattformen

Beim *Software-in-the-Loop* (SiL) wird ein reines Software-Testobjekt in einer simulierten Umgebung getestet. Ein Sonderfall des SiL ist Model-in-the-Loop (MiL), bei dem kein ganzes System, sondern nur einzelne Software-

module getestet werden. Bei SiL-Systemen ist es möglich, die Testfälle in einem Vielfachen der realen Geschwindigkeit zu simulieren. Zudem ist eine Parallelisierung auf mehrere Rechner bzw. in die Cloud möglich. Diese Eigenschaften führen dazu, dass SiL-Systeme u.a. dazu eingesetzt werden, den erhöhten Testaufwand durch steigende Software im Produkt (s. Abschnitt 1.1.3) entgegenwirken zu können. SiL-Systeme werden vor allem für Unit- und Modultests (s. Abschnitt 2.1.4) eingesetzt. Sie können in allen Musterphasen (A-D, s. Abschnitt 2.1.2) verwendet werden.

Im Vergleich zum SiL läuft das Testobjekt am *Hardware-in-the-Loop (HiL)* als integriertes System auf der Zielhardware bzw. Steuergeräten. Die Umgebung des Testobjekts wird weiterhin simuliert. Die Nutzung realer Steuergeräte begrenzt die Ausführungsgeschwindigkeit der Testfälle, da die Geräte die Daten in Echtzeit verarbeiten. Eine Parallelisierung der Tests auf mehreren HiL-Systeme, vergleichbar zum SiL, ist aufgrund der damit verbundenen Kosten und Auf- und Umrüstzeiten nicht einfach möglich [28]. HiL-Plattformen werden vor allem für Modul- und Integrationstests (s. Abschnitt 2.1.4) verwendet und können für alle Musterphasen (s. Abschnitt 2.1.2) eingesetzt werden. Im Vergleich zum SiL können am HiL Fehler gefunden werden, die durch Hardware oder reale Schnittstellen zwischen den Modulen verursacht werden [11]. Dennoch ist weiterhin eine Lücke zwischen der Realität und der simulierten Umgebung vorhanden.

Während beim HiL Teile des Fahrzeugs als Testumgebung simuliert werden, wird beim *Vehicle-in-the-Loop (ViL)* das ganze Fahrzeug in Realität aufgebaut und nur die Fahrzeugumgebung simuliert. Auch am ViL kann nur in realer Geschwindigkeit getestet werden und es ist keine Parallelisierung möglich. ViL wird für System- und Akzeptanztests (s. Abschnitt 2.1.4) und für die Musterphasen B-D (s. Abschnitt 2.1.2) verwendet. ViL eignet sich zudem für Studien mit Menschen als Fahrer, mit dem Vorteil, dass das Unfallrisiko durch die simulierte Umgebung minimiert ist. Die simulierte Umgebung kann z.B. in Augmented- bzw. Virtual-Reality-Brillen für den Fahrer angezeigt werden. Da im Vergleich zum HiL das gesamte Fahrzeug aufgebaut ist, wird die Fahrdynamik zusätzlich real abgebildet.

Beim *Rapid Prototyping (RP)* ist das Testobjekt als Software umgesetzt und wird in der Realität getestet. Rapid Prototyping wird dann eingesetzt, wenn die Hardwareentwicklung für das Testobjekt bisher nicht abgeschlossen ist, dieses aber schon in realer Umgebung getestet wird. RP kann bereits im linken Ast

des V-Modells angewendet werden, um konzeptionelle Tests durchzuführen. Es wird für die Musterphasen A-B (s. Abschnitt 2.1.2) verwendet, wenn noch keine seriennahen Produktionswerkzeuge existieren. Durch die reale Umgebung können Konzepte bereits validiert werden, es sind aber keine Aussagen über die Zielhardware möglich.

Beim *Onboard Test (OT)* ist das Testobjekt als integriertes System umgesetzt, das in der Realität getestet wird. Die benötigten Fahrzeug-Prototypen sind eine teure und knappe Ressource [13, 11], weil sie für den aktuellen Stand der Entwicklung neu aufgebaut werden müssen. Ein zusätzlicher Aufwand für Testfälle für HAF besteht darin, dass die Testumgebung, also Straßen, Infrastruktur und andere Verkehrsteilnehmer, aufgebaut bzw. aufgesucht werden muss. Zudem ist die Abdeckung der ODD nicht ohne Weiteres möglich, wenn relevante Aspekte des Tests wetterabhängig oder ortsabhängig sind: Tests mit Regen, Nebel, Schneefall oder niedrigen Temperaturen können z. B. nicht jederzeit an einem beliebigen Ort durchgeführt werden. OT wird für die Systemtests und Akzeptanztests (s. Abschnitt 2.1.4) sowie in den Musterphasen B-D (s. Abschnitt 2.1.2) verwendet. Die Testergebnisse des OT haben die höchste Glaubwürdigkeit unter den vorgestellten Testplattformen, da keine simulierten Anteile vorhanden sind. Deshalb spielt das Onboard Test in der Absicherung von HAF immer noch eine wichtige Rolle [45, 33].



## 3 Stand der Wissenschaft und Technik

### 3.1 Manöver

#### 3.1.1 Definition und Einordnung

Manöver beschreiben das Verhalten und die Interaktion von Verkehrsteilnehmern. Sie lassen sich im 6-Layer-Modell (s. Abschnitt 3.2.1) auf Layer 4 einordnen und sind daher Bestandteil einer Szenariobeschreibung (s. Definition 3.1.1). Zusätzlich existieren bei einigen Manövern Beziehungen zu anderen Layern.

In der Literatur werden verschiedene Definitionen für den Manöverbegriff eingeführt [46, 29, 47, 48, 49], wobei folgende Gemeinsamkeiten existieren: Manöver haben eine räumliche bzw. zeitliche Ausdehnung, die eine Abstraktion des Verhaltens ermöglicht. Zudem bilden Manöver im Allgemeinen häufig einen Übergangsprozess zwischen zwei Zuständen ab. Für diese Dissertation wird die Definition für Manöver nach Bach [40] verwendet:

**Definition 3.1** (Manöver). *Ein **Manöver** ist eine abstrakte Definition des Verhaltens eines Verkehrsteilnehmers innerhalb einer Zeitspanne ([40]).*



Das Abbilden des Verhaltens von Verkehrsteilnehmern durch Manöver kann als klassische Modellierung betrachtet werden, in der sich Modelle nach Stachowiak [50] durch drei Merkmale auszeichnen: Abbildung, Verkürzung und Pragmatik.

Abbildung bedeutet, dass das Modell ein Original nachahmt bzw. repräsentiert. Es kann von einem Original auch mehrere Abbilder bzw. Modelle geben. Manöver bilden in diesem Zusammenhang das Verhalten von Verkehrsteilnehmern in der realen Welt ab.

Verkürzung bedeutet, dass im Modell nicht alle Attribute des Originals erfasst werden. Zum einen können nur bekannte Eigenschaften in die Modellierung einfließen, zum anderen werden nur die für den Verwendungszweck wichtigen, nützlichen oder notwendigen Eigenschaften betrachtet [51]. Die Verkürzung bei der Beschreibung durch Manöver ist durch die Abstraktion gegeben, indem das exakte Verhalten durch ein kompakt definiertes Manöver repräsentiert wird. Beispielhaft kann, anstatt den kompletten Geschwindigkeitsverlauf eines Bremsvorgangs zu beschreiben, eine Verkürzung auf ein Bremsmanöver mit Start- und Endgeschwindigkeit sowie durchschnittlicher Bremsverzögerung erfolgen.

Pragmatik bedeutet, dass Modelle keinen Selbstzweck erfüllen, sondern für einen Verwendungszweck designet werden. Die Pragmatik steht in engem Zusammenhang zu den ausgewählten Eigenschaften. Eigenschaften des Originals können z.B. für einen Verwendungszweck wichtig sein, für einen anderen aber nicht. Daraus ergibt sich auch, dass es a priori keine richtigen oder falschen Modelle gibt [51]. Die Auswahl der Manöver und deren Eigenschaften hängt mit deren Verwendung im SBT-Prozess zusammen (s. Abschnitt 3.3), und damit auch mit ihrer Art der Beschreibung (s. Abschnitt 3.4.5) und den enthaltenen Informationen (s. Abschnitt 3.4.4). Werden sie z. B. für ausführbare Szenarien in der Testfallausführung verwendet, sind andere Manöver und Eigenschaften nützlich, wie beim Vergleich von beobachteten Szenarien zur Auswahl der Testfälle.

### 3.1.2 Manöverkataloge

Für das Verwenden von Manövern in einer Szenariobeschreibung werden alle modellierten Manöver definiert und in einem Manöverkatalog festgehalten.

**Definition 3.2** (Manöverkatalog). *Ein **Manöverkatalog** enthält alle für die Beschreibung des Verhaltens der Verkehrsteilnehmer verwendbaren Manöver.*



Nagel et al. [46]	Bagschik et al. [48]	Hartjen et al. [49]
Start und Fortfahren	Fortfahren	Beschleunigen
Straße folgen		Spur halten
Annäherung an vorausliegendes Objekt	Annähern	Objekt annähern
Überholen		Überholen
Vor Hindernis anhalten		
Hindernis seitlich umfahren	Vorbeifahren	
Hinter vorausfahrendem Fahrzeug starten		
Fahrzeug folgen	Folgen	Objekt folgen
Kreuzung überqueren		Kreuzung überqueren
In rechte/linke Fahrspur einordnen	Spurwechsel	Spurwechsel
Rechts/Links abbiegen	Abbiegen	Rechts/Links abbiegen
Verlangsamen und am Straßenrand halten	Sicher Anhalten	Bremsen
Rückwärts fahren		Rückwärts fahren
U-turn links/rechts	Umdrehen	
Richtung wechseln		
In Parklücke einfahren		Parken
Parklücke verlassen		
		Geschwindigkeit halten
		Zebrastreifen annähern / überqueren

Tabelle 3.1: Ausgewählte Manöverkataloge. Manöver, die ähnliches Verhalten beschreiben, sind auf der gleichen Ebene in der Tabelle dargestellt.

In der Literatur und im praktischen Gebrauch existieren je nach Umfeld und Ziel unterschiedliche Kataloge an Manövern, welche jeweils eine Modellierung mit unterschiedlichem Verwendungszweck darstellen (s. Abschnitt 3.1.1). Ein Manöverkatalog erhebt damit keinen Anspruch auf universelle Vollständigkeit. Tabelle 3.1 zeigt die Manöver ausgewählter Manöverkataloge. Ein Katalog findet sich in Nagel et al. [46], der durch Tolle et al. [52] und Reschka et al. [53] erweitert wurde. Dieser zielt auf eine Abdeckung des Verkehrs für Kraftfahrzeuge ab und enthält z.B. die Manöver “Anfahren”, “Kreuzung überqueren”, “rückwärts fahren” und “in eine Parklücke einfahren”. Pfeffer [36] definiert einen Katalog an Manövern für das Verhalten auf Autobahnen, mit dem Fokus auf die Resimulation aufgezeichneter Szenarien. Bagschik et al. [48] entwickeln einen Satz an Manövern, um wissensbasierte Szenarien aus Ontologien herzuleiten. Hartjen et al. [49] erstellen einen Manöverkatalog für die Beschreibung von urbanem Verkehr, in dem die Manöver auch Verknüpfungen zur Infrastruktur (Layer 1) beinhalten können. Für die Ausführung von Szenarien bieten Simulationstools jeweils eigene Manöverkataloge, z. B. der Action Cataloge aus OpenSCENARIO XML [54], welche mit den Simulationsmodellen des jeweiligen Tools kompatibel sind.

### 3.1.3 Kategorien von Manövern

Als weiteren Schritt können die Manöver eines Manöverkatalogs anhand ihrer Eigenschaften in mehrere Kategorien eingeteilt werden. Wie bei der Modellierung der Manöver (s. Abschnitt 3.1.1), gibt es auch hierbei a priori keine richtigen oder falschen Kategorien. Vielmehr müssen diese dem Verwendungszweck entsprechen. Abbildung 3.1 zeigt in der Literatur vorgestellte Kategorien für Manöver und ihre Ausprägungen.

**Definition 3.3** (Manöverkategorie). *Eine **Manöverkategorie** beinhaltet verschiedene Manöver mit gleichen Eigenschaften.*

Eine verbreitete Einteilung ist in die *Bewegungsrichtungen* längs (longitudinal) und quer (lateral). Sie ist besonders für die Steuerung von Fahrzeugen geeignet, weil laterale Manöver über die Lenkung und longitudinale Manöver über den Antriebsstrang geregelt werden. Weitergehend ist diese Kategorie bei der Beschreibung ausführbarer Szenarien, wie z. B. den in Abschnitt 3.4.2

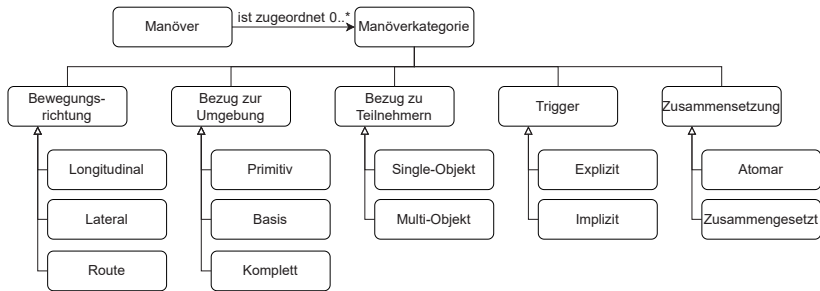


Abbildung 3.1: Manöverkategorien und ihre Ausprägungen

vorgestellten Tools, etabliert. Eine Erweiterung durch “Routing”-Manöver, wie z. B. in OpenSCENARIO XML [54], ist möglich. Diese beziehen sich nicht auf die Spurhaltung, sondern auf die Navigation auf der Karte.

Nagel et al. [55] kategorisiert die Manöver über die Sicherheitsrisiken und *Bezug zur Umgebung*. Primitive Manöver haben keinen Bezug zur Szenerie oder zu anderen Verkehrsteilnehmern. Dazu gehören z. B. die Manöver Beschleunigen, Abbremsen und eine Kreisfahrt mit einem konstanten, vorgegebenen Gierwinkel. Basis-Manöver haben einen Bezug zu anderen Verkehrsteilnehmern und/oder zur Szenerie, der sich aber während des Manövers nicht ändert. Beispiele für Basis-Manöver sind eine Folgefahrt und Spurhalten. Bei kompletten Manöver ändert sich der Bezug zu anderen Elementen während des Manövers. Dies ist z. B. bei einem Überholmanöver der Fall, bei dem das eigene Fahrzeug zu Beginn hinter einem anderen Fahrzeug und am Ende vor einem anderen Fahrzeug ist.

Firl et al. [56] charakterisiert auf Basis der *Interaktion* mit anderen Teilnehmern. Single-Object Manöver haben keinen Bezug zu anderen Teilnehmern. Dazu gehören z. B. Spurwechsel und Einparken. Bei Multi-Object Manöver findet Interaktion zu anderen Teilnehmern statt. Beispiele sind Folgefahrten und Überholmanöver.

Schreiber et al. [57] kategorisiert über den *Trigger* für den Start bzw. Ende des Manövers. Wird ein Manöver aktiv gestartet, z. B. ein Spurwechsel, ist dies ein explizites Manöver. Ein auf diesen Spurwechsel folgendes Spurhalten-Manöver ist ein implizites Manöver, weil es automatisch startet, sobald der

Spurwechsel abgeschlossen ist. Analog dazu teilen sich Manöver in selbstbeendende Manöver (z. B. Spurwechsel) und ohne Einfluss von außen immer weiter ausführende Manöver (z. B. Spurhalten) ein.

Eine weitere Einteilung ist in *atomare* und *zusammengesetzte* Manöver, wie z. B. in Bagschik et al. [48], Pfeffer [36] und OpenSCENARIO DSL [58]. Dabei sind atomare Manöver eine Grundklasse an Manövern, die sich zu zusammengesetzten Manövern kombinieren lassen. Z.B., ein Überholmanöver, das aus den atomaren Manövern Ausscheren, an Fahrzeug vorbeifahren und Einscheren zusammengesetzt wird.

### 3.1.4 Manöver-Identifikation

In den aufgezeichneten Daten sind die Manöver der Teilnehmer nicht direkt festgehalten, sondern nur indirekt über die konkret erfassten Größen, wie z.B. die Geschwindigkeitsverläufe und Positionen, zu ermitteln. Um die Manöver der Teilnehmer darin zu erkennen, ist die Manöver-Identifikation notwendig.

**Definition 3.4.** *Die Manöver-Identifikation erfasst die ausgeführten Manöver der Teilnehmer in aufgezeichneten Daten.*

Es stehen verschiedene Methoden der Klassifikation zur Verfügung, um diese Aufgabe zu lösen (s. Tabelle 3.2): Bei der *regelbasierten Identifikation* werden Regeln erstellt, die auf logischen Annahmen und Systemwissen basieren. Anhand dieser Regeln werden die aufgezeichneten Daten klassifiziert.

Vorteile der regelbasierten Identifikation sind die Interpretierbarkeit und Nachvollziehbarkeit der Ergebnisse, welche durch die bekannten Regeln erzeugt werden. Dadurch kann auch eine gezielte Fehlerkorrektur durchgeführt werden, da die Gründe für Fehlerkennungen eindeutig im Regelwerk bestimmbar sind. Nachteile sind eine beschränkte Übertragbarkeit auf unbekannte Manöver und Situationen, die beim Erstellen des Regelwerks nicht berücksichtigt wurden. Zudem können das Erstellen und die Wartung solcher Regelwerke umfangreich und zeitaufwendig sein.

Regelbasierte Klassifikation wird u.a. von Hartjen et al. [49] verwendet, um Manöver in urbanen Verkehrssituationen zu identifizieren. Roesener et al. [59] erfassen die Manöver in aufgezeichneten Realdaten über einen Entscheidungs-

baum in Realdaten. Montanari et al. [60] nutzen einen erstellten Satz an Regeln, um Manöver für die Resimulation zu extrahieren.

Beim *unüberwachten maschinellen Lernen* ist der Fokus auf dem Erkennen von Mustern, ohne dass vorherige Kenntnisse oder explizite Anweisungen vorhanden sind. Die Methodik identifiziert selbstständig Strukturen in den Daten als Manöver. Vorgegebene Manöver, z. B. in Form eines Manöverkatalogs, werden nicht benötigt und können auch nicht festgelegt werden. Durch dieses Vorgehen können Manöver ohne Expertenwissen und manuellen Modellierungsschritt identifiziert werden. Das unüberwachte Lernen kann zudem unbekannte Muster erkennen und sich an diese anpassen. Allerdings besteht dabei die Gefahr, dass unerwünschte oder für den Anwender unlogische Muster erlernt werden. Auch der Entscheidungsprozess des Modells kann nur implizit und damit schwer nachvollziehbar sein.

Da kein Manöverkatalog vorgegeben werden kann, ist eine Klassifikation von modellierten Manövern mit unüberwachtem maschinellen Lernen nicht möglich. Das Vorgehen wird daher zur allgemeinen Verhaltenserkennung von Verkehrsteilnehmern verwendet. Z.B. nutzen Li et al. [61] unüberwachtes Clustering für die Analyse von Fahrstilen von Probanden einer ViL-Simulation. Zhu et al. [62] clustern das Verhalten von Teilnehmern in Spurwechselmanövern für das Parametrisieren eines personalisierten Spurwechselassistenten.

Das *überwachte maschinelle Lernen* trainiert ein Klassifikationsmodell auf bekannten Datensätzen, in denen für jede Aufzeichnung die entsprechenden Manöver gekennzeichnet sind. Durch das Training erlernt das Modell, wie es Manöver in unbekannten Daten nach denselben Mustern der gelabelten Aufnahmen identifizieren kann. Überwachtes Lernen ermöglicht es einem Modell, Muster in den Trainingsdaten zu erkennen und diese Muster auf neue, nicht gesehene Daten zu generalisieren. Dadurch können solche Modelle die höchste Genauigkeit der vorgestellten Methoden erreichen. Die Ergebnisse hängen allerdings stark von der Qualität und Quantität der Trainingsdaten ab. Der Aufbau eines hochwertigen und ausreichend großen Datensatzes ist die größte Herausforderung im überwachten maschinellen Lernen und kann zum Teil nur durch manuellen Aufwand gelöst werden.

Neuronale Netze zur Klassifikation werden u.a. von Aboah et al. [63] und Erdogan et al. [64] verwendet. Barbier et al. [65] verwenden den Random-Forest-Algorithmus, um Manöver an einer urbanen Kreuzung in einem hybriden Datensatz aus realen und simulierten Daten zu identifizieren. In Al et al.

[66] wird gezeigt, wie Manöver in aufgezeichneten Mobiltelefonaten mittels Random Forest und Support Vector Machines erkannt werden können.

Kategorie	Manöver-katalog	Vorteile	Nachteile
Regelbasiert	vorgegeben	Interpretierbarkeit, gezielte Fehlerkorrektur	Unbekannte Daten, Aufwändige Regelwerke
Unüberwachtes Lernen	nicht vorgegeben	Anpassungsfähigkeit, Kein vorheriges Wissen erforderlich	Mangelnde Kontrolle, Schwierige Interpretierbarkeit
Überwachtes Lernen	vorgegeben	Generalisierbarkeit, Genauigkeit	Abhängigkeit von Trainingsdaten, Aufwändiges Labeln

Tabelle 3.2: Vor- und Nachteile der Methoden zur Manöver-Identifikation

### 3.1.5 Anwendung von Manövern im szenariobasierten Testen

Manöver werden vielfältig in der Beschreibung von Szenarien angewendet und an unterschiedlichen Stellen im szenariobasierten Testen verwendet.

So werden Manöver genutzt, um Szenarien zu erstellen (s. Abschnitt 3.4.1): In Pfeffer [36] und Montanari [67] bilden Manöver die Grundlage, um beobachtete Szenarien zu resimulieren. Auch Bach et al. [29] erstellen eine modellbasierte Beschreibung auf der Basis von Manövern. Der zeitliche Verlauf eines Szenarios wird dabei in Akte unterteilt, die sich durch wechselnde Manöver voneinander abgrenzen. Erdogan et al. [64] extrahieren Manöver aus Realdaten, um Testszenarien für die virtuelle Absicherung zu definieren. Krajewski et al. [68] trainieren ein Generative Adversarial Network und einen Variational Autoencoder als Spurwechselmodell, mit dem sich künstlich Trajektorien für Spurwechsel generieren lassen. Während die bisher genannten Ansätze Manöver aus den Daten extrahieren, nutzt Bagschik et al. [48] Manöver in einer Ontologie für die wissensbasierte Generation von Szenarien, welche in der Absicherung zum Einsatz kommen.

In ausführbaren Szenarien sind die Manöver die Anweisungen an die Teilnehmer, wie z.B. in OpenSCENARIO [54]. In aufgezeichneten Szenarien beschreiben Manöver das aufgezeichnete Verhalten, wie z.B. in Hartjen et al. [69]. Der Unterschied zwischen beiden Arten kann durch die sprachliche Ausdrucksweise verdeutlicht werden: In ausführbaren Szenarien ist der Imperativ als Anweisung passend (Beschleunige, Überhole), in aufgezeichneten Szenarien das dazugehörige substantivierte Verb (Beschleunigung, Überholvorgang).

Nach Cai et al. [70] können Manöver auch verwendet werden, um manöverbasierte Kritikalitätsmetriken zu definieren, wie z.B. Time-to-Steer, Time-to-Break oder Time-to-Maneuver. Die Kritikalität der Szenarien ist relevant für die Auswahl der Testfälle (s. Abschnitt 3.3.3).

Zudem nutzen gängige Simulationstools (s. Abschnitt 3.4.2) Manöver als Anweisungen in ausführbaren Szenarien beim Ausführen der Testfälle (s. Abschnitt 3.3.4).

In der Sicherheitsargumentation (s. Abschnitt 3.3.6) werden Manöver für die Abschätzung der Testabdeckung verwendet. In Hartjen et al. [71] sind Manöver die Basis, um urbane Verkehrsszenarien in Realdaten zu vergleichen. Das erklärte Ziel dabei ist, die Gesamtmenge an unterschiedlichen Szenarien im urbanen Raum zu schätzen und Sättigungseffekte beim Sammeln von Szenarien zu erkennen. Als wissensbasierter Ansatz berechnet Nanri et al. [72] über die Kombinatorik der gegebenen Manöver die Anzahl möglicher Szenarien innerhalb einer ODD, welche als vollständige Menge der zu beherrschenden und zu testenden Szenarien interpretiert wird.

## 3.2 Szenariobasiertes Testen

Aktuelle Forschungsprojekte und die Industrie konzentrieren sich auf Szenariobasiertes Testen (SBT) als zentrale Testmethode für die Verifikation und Validierung von Automatisierte Fahrfunktionen [73]. Hauer et al. [74] beschreiben die Idee des szenariobasierten Testens als „automatisches oder manuelles Identifizieren eines vernünftig kleinen Satzes von relevanten dynamischen Verkehrssituationen oder Szenarioarten“. Dazu werden Szenarien gesammelt, relevante und repräsentative Szenarien aus diesem ausgewählt und mit Anforderungen an das Testobjekt kombiniert. Das szenariobasierte Testen kann im

V-Modell (s. Abschnitt 2.1.4) sowohl auf niedrigen Stufen entwicklungsbegleitend als auch für die Validierung des Gesamtsystems verwendet werden.

### 3.2.1 Begriffsdefinitionen

Diese Ausarbeitung verwendet die Begriffsdefinitionen der ISO/PAS 21448 [75], welche auf den Begriffen der PEGASUS-Projektfamilie aufbauen [76][53].

**Definition 3.5.** *Die **Sicherheit** eines Systems ist definiert durch das Fehlen unangemessener Risiken aufgrund von Gefahren, die aus Funktionsmängeln der vorgesehenen Funktionalität entstehen [75].*

Diese Definition aus der ISO 21448 wird auch als “Safety of the Intended Functionality” (SOTIF) bezeichnet. Sie bezieht folgende Ursachen von Risiken mit ein: funktionale Mängel, einschließlich Algorithmen basierend auf künstlicher Intelligenz, fehlerhafte oder unzureichende Gestaltung der Mensch-Maschine-Schnittstelle, vorhersehbarer fehlerhafter Gebrauch sowie externe Einflüsse durch Kommunikationsschnittstellen oder Umwelteinflüsse [42].

**Definition 3.6 (Szene).** *Eine **Szene** beschreibt eine Momentaufnahme des Umfelds, welche die Szenerie, dynamische Elemente, die Selbstrepräsentation aller Akteure und Beobachter wie auch die Verknüpfung dieser Entitäten umfasst.*

Die Szenerie beschreibt in diesem Kontext alle räumlich stationären Aspekte einer Szene. Diese sind metrische, semantische und topologische Informationen, z.B. Fahrstreifen, Fahrstreifenmarkierungen, Fahrbahnoberflächen oder der Typ der Verkehrsdomäne. Zudem sind quasi-stationäre Wetter- und Lichtbedingungen Teil der Szenerie. Für dynamische Elemente bzw. Verkehrsteilnehmer wird angenommen, dass sie sich durch kinetische Energie bewegen oder sich durch zur Bewegung vorhandene Energie oder Fähigkeiten bewegen könnten. Die Selbstrepräsentation der Akteure (selbst handelndes Element) und Beobachter (wahrnehmendes Element) umfasst das aktuelle Fertigkeiten-Level sowie generelle Systemfähigkeiten und Fertigkeiten (s. Reschka et al. [53]).



**Definition 3.7** (Szenario). Ein **Szenario** beschreibt die zeitliche Entwicklung von Szenenelementen innerhalb einer Folge von Szenen, welche mit einer Start-szene beginnt.

Im Gegensatz zur Szene, die nur einen Zeitpunkt darstellt, deckt ein Szenario eine bestimmte Zeitspanne ab (s. Abbildung 3.3). Die Szenen innerhalb eines Szenarios werden durch Aktionen und Ereignisse verknüpft, welche zuvor spezifiziert werden können.

Szenarien allein sind für eine Sicherheitsargumentation (s. Abschnitt 3.3.6) nicht ausreichend, da kein Bezug zu einem Testobjekt oder Anforderungen an dieses enthalten sind. So könnte man ein Szenario auch ohne den V&V-Hintergrund ausführen. Um die spezifischen Eigenschaften eines Testobjekts im Szenario nachzuweisen, werden daher aus Anforderungen abgeleitete Pass-Fail-Kriterien hinzugefügt, die im Szenario zu erfüllen sind.

**Definition 3.8** (Testfall). Ein **Testfall** im szenariobasierten Testen beschreibt eine Menge von Bedingungen, um festzustellen, ob ein Testobjekt entsprechend seiner beabsichtigten Funktionalität operiert. Ein Testfall umfasst ein Szenario (inkl. Variationen, die alle Aspekte des Szenarios abdecken) und Pass-Fail-Kriterien, anhand derer es bewertet wird (s. Abbildung 3.2).

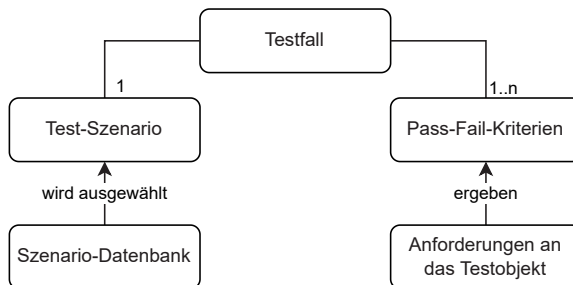


Abbildung 3.2: Überblick der Begriffe und Beziehungen für Testfälle im szenariobasierten Testen

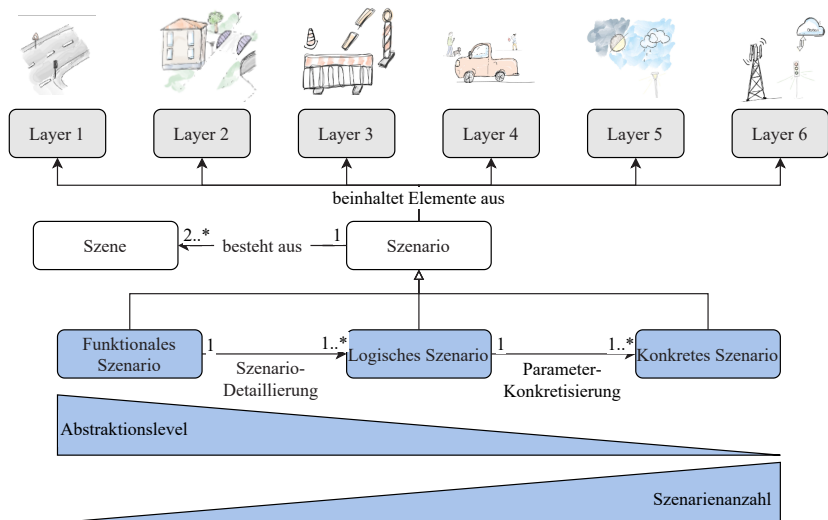


Abbildung 3.3: Beziehungen von Szenarien zu Szenen [53], Abstraktionsebenen [77, 78] und Strukturierung der Elemente [79].

### 3.2.2 Strukturierung und Kategorisierung des Verkehrs und der Umgebung

Zur detaillierteren Kategorisierung der Elemente eines Szenarios wurde im Rahmen der PEGASUS-Projekte iterativ ein Layer-Modell entwickelt [80][48][81]. In der aktuellen Version von Scholtes et al. [79] werden 6 Layer verwendet (s. Abbildung 3.3):

*Layer 1* beschreibt die Straßentopologie und den Verlauf der Fahrspuren. *Layer 2* beschreibt die Infrastruktur und statischen Elemente wie Verkehrsschilder, Leitplanken und Bäume. *Layer 3* beschreibt zeitliche Veränderungen der Layer 1 und 2, wie z.B. Baustellen. *Layer 4* beschreibt die dynamischen Objekte (Verkehrsteilnehmer) des Szenarios, inklusive deren Verhalten. *Layer 5* beschreibt die Umweltbedingungen wie Wetter, Wind oder Helligkeit. *Layer 6* beschreibt Car-to-X Kommunikation (Car2X) und Datenübertragung zwischen der Infrastruktur und den Teilnehmern.

### 3.2.3 Abstraktionsebenen während des Entwicklungsprozesses

Während des Entwicklungsprozesses (s. Kapitel 2) muss die Beschreibung des Szenarios unterschiedlich detaillierte Informationen enthalten. In den PEGASUS-Forschungsprojekten werden sie dazu in die Abstraktionsebenen funktional, logisch und konkret aufgeteilt (s. Abbildung 3.3) [78], um die Anforderungen an das Szenario im jeweiligen Prozessschritt zu erfüllen (s. Abbildung 3.4):

*Funktionale Szenarien* basieren auf einer sprachlichen, informellen Beschreibung, können Expertenwissen abbilden und sind intuitiv lesbar. *Logische Szenarien* ergeben sich durch eine Detaillierung und Formalisierung des funktionalen Szenarios in den physikalischen Zustandsraum, indem Wertebereiche für im Szenario enthaltene Parameter aufgestellt werden. Die Parameter können als Wahrscheinlichkeitsverteilung (Normalverteilung, Gleichverteilung, etc.) dargestellt werden, die der Realität nachempfunden sind. *Konkrete Szenarien* entstehen durch eine Konkretisierung der Parameterbereiche logischer Szenarien auf feste Werte. Zwischen den logischen und konkreten Szenarien besteht damit eine 1-zu-n-Beziehung.

Die Abstraktionsformen können in das V-Modell (s. Abschnitt 2.1.4) eingeordnet werden [78] (s. Abbildung 3.4). Während des Entwicklungsprozesses werden die Szenarien von abstrakt zu konkret ausdetailliert.

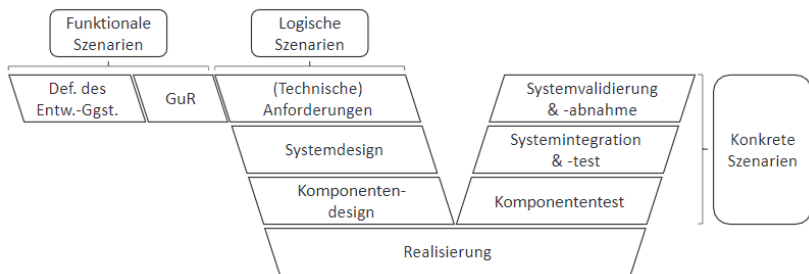


Abbildung 3.4: Einordnung der funktionalen, logischen und konkreten Szenario-Darstellungsformen im V-Modell nach [78]

### 3.2.4 (Un-)bekannte und (un-)sichere Szenarien

**Definition 3.9** (Sicheres Szenario). Ein *sicheres Szenario* ist ein Szenario, in dem das Testobjekt unfallfrei operiert und alle Anforderungen der zugehörigen Testfälle erfüllt (s. Abbildung 3.2).

**Definition 3.10** (Bekanntes Szenario). Ein *bekanntes Szenario* kann für Testfälle genutzt werden (s. Abbildung 3.2).

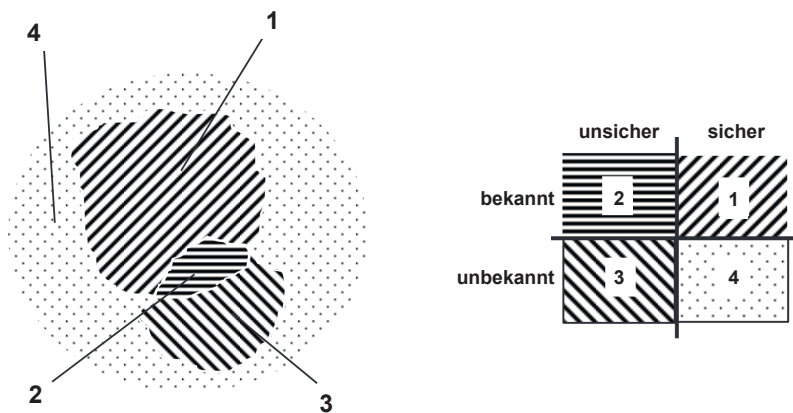


Abbildung 3.5: Klassifikation der Szenarien in sicher/unsicher und bekannt/unbekannt nach ISO/PAS 21448 [75]

In der ISO/PAS 21448 [75] werden Szenarien während des Entwicklungsprozesses über die zwei Klassifikatoren zur Bekanntheit (s. Definition 3.10) und Sicherheit (s. Definition 3.9) in vier Bereiche eingeteilt (s. Abbildung 3.5):

- Bereich 1: Bekannte, sichere Szenarien
- Bereich 2: Bekannte, unsichere Szenarien
- Bereich 3: Unbekannte, unsichere Szenarien
- Bereich 4: Unbekannte, sichere Szenarien

Die Größe der Bereiche repräsentiert den Anteil jeder Art von Szenario innerhalb der ODD. Während die Häufigkeit bekannter Szenarien, also die Szenarien in den Bereichen 1 und 2, bestimmbar ist, kann sie für die unbekannten Szenarien der Bereiche 3 und 4 nur geschätzt werden. Da unbekannte Szenarien aber auch unsicher sein können (Bereich 3), ist diese Schätzung für eine vollständige Sicherheitsargumentation notwendig (s. Herausforderung 2).

Während der Entwicklung von HAF ist es das Ziel, die Bereiche mit unsicheren Szenarien (Bereich 2 und 3) ausreichend zu verringern, um das Risiko auf ein akzeptables Maß zu reduzieren (s. Abschnitt 1.1.1 und Abbildung 3.6). Ein sicheres Szenario wird über Testfälle, die dieses Szenario beinhalten, nachgewiesen.

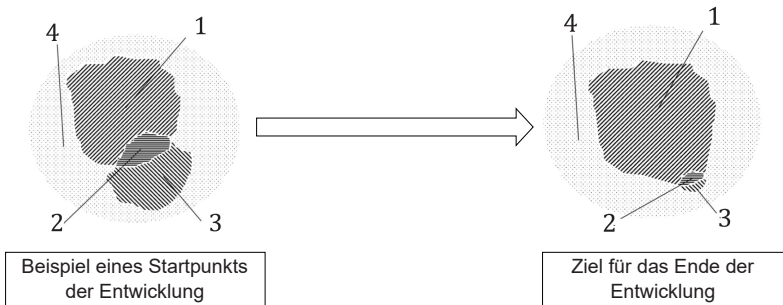


Abbildung 3.6: Evolution der Bereiche während des Entwicklungsprozesses durch Anwendung der ISO/PAS 21448 Aktivitäten [75]

Für die einzelnen Bereiche ergeben sich daraus folgende Ziele: Bereich 1 maximieren, während Bereiche 2 und 3 minimiert werden. Bereich 2 minimieren, potenzielle Risiken evaluieren und, falls notwendig, risikoreiche Szenarien durch technische Verbesserungen in Bereich 1 verschieben. Bereich 3 so weit wie möglich unter akzeptablem Aufwand minimieren. Bereich 4 ist für die Risikoabschätzung nicht relevant, daher sind keine Ziele definiert.

Aus diesen Zielen lassen sich zusammenfassend drei zentrale Herausforderungen ableiten:

**Herausforderung 1.** *Exploration des Szenarienraums, um unbekannte Szenarien zu finden und den Anteil an bekannten Szenarien zu erhöhen.*

**Herausforderung 2.** *Abschätzen der Häufigkeit unbekannter, potenziell unsicherer Szenarien für eine vollständige Sicherheitsargumentation.*

**Herausforderung 3.** *Nachweis der Sicherheit des Testobjekts in bekannten Szenarien über zugehörige Testfälle, um den Anteil an sicheren Szenarien zu erhöhen.*

### 3.3 Der SBT-Prozess

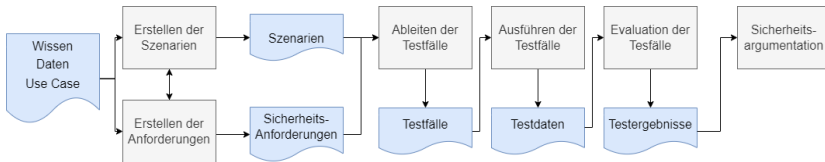


Abbildung 3.7: Darstellung des SBT-Prozesses für das szenariobasiertes Testen nach [82]

Abbildung 3.7 zeigt ein Prozessbild für das Szenariobasiertes Testen (SBT) nach Neurohr et al. [82] basierend auf den PEGASUS-Projekten und dem Projekt ENABLE-S3. Graue Rechtecke symbolisieren Methoden, blaue Blöcke Artefakte, die zwischen den Methoden entstehen.

#### 3.3.1 Erstellen der Szenarien

Das Erstellen der Szenarien beinhaltet den Aufbau einer Szenariendatenbank, die in Abbildung 3.7 als Artefakt “Szenarien” gekennzeichnet ist. Die Gesamtheit der erstellten Szenarien ist die Basis für die Testabdeckung der ODD und muss diese daher möglichst gut repräsentieren (s. Herausforderung 1).

Bei der wissensbasierten Szenario-Erstellung wird Expertenwissen formalisiert und festgehalten. Ausgangspunkte sind z.B. bestehende Normen und Gesetze, Systemspezifikationen und bereits bekannte Szenarien [83]. Die Formalisierung kann z.B. über Ontologien durchgeführt werden, in denen die Elemente eines Szenarios und deren logische Beziehungen festgehalten sind [48]. Bei der datenbasierten Szenario-Erstellung werden Szenarien aus aufgezeichneten

Daten extrahiert und kategorisiert [84][69][78]. Datengrundlage können durch Testfahrzeuge, Verkehrsinfrastruktur oder von Drohnen aufgezeichnete Realdaten, Verkehrssimulationsmodelle oder Unfalldatenbanken sein. Erweiterte Methoden zur Erstellung werden im Abschnitt 3.4.1 vorgestellt.

Je höher der Grad der Automatisierung (s. Abschnitt 1.1.2), desto schwieriger ist es, eine vollständige rein wissensbasierte Szenariendatenbank zu erstellen, weil es „Experten an kombinatorischer Leistung fehlt, den offenen Kontext [der realen Fahrumgebung] vollständig zu untersuchen, was das Risiko eines unvollständigen Satzes an Szenarioklassen erhöht“ [82]. Deshalb können Kombinationen aus beiden Methoden verwendet werden [74, 77]: Top-down können Szenarien von Experten erstellt und deren Häufigkeit über Daten ermittelt werden [85]. Bei Bottom-up-Ansätzen analysieren Experten aus Daten erzeugte konkrete Szenarien und klassifizieren diese zu logischen und funktionalen Szenarien [86][87].

### **3.3.2 Erstellen der Anforderungen**

Das Erstellen der Anforderungen beinhaltet das Erkennen von Gefährdungen und Risiken, die Ableitung von Sicherheitszielen und das Ausdetaillieren dieser Sicherheitsziele in Anforderungen auf System- und Komponenten-Ebene.

Ausgangspunkt für das Erstellen der Anforderungen ist nach der PEGASUS-Methode [77] der Wissenscontainer. Dieser beinhaltet als relevante Quellen für die Anforderungen Gesetze und Normen (z.B. ISO 26262 [45] und ISO/PAS 21448 [75]), die Leitlinien der Ethik-Kommission [88], die ODD und Ergebnisse aus vorherigen Schritten des Entwicklungsprozesses wie durchgeführte Risikoabschätzungen und die Systemspezifikation. Dazu zählt z.B. Automotive Safety Integrity Level (ASIL), eine Klassifikationsmethode, mit der potenzielle Gefahren und deren Auswirkungen bewertet werden, Failure Mode and Effects Analysis (FMEA), eine Methode zur systematischen Analyse von potenziellen Ausfallmodi, und Threat Analysis and Risk Assessment (TARA), ein Vorgehensmodell zur Identifikation von IT-Sicherheitsrisiken und Schwachstellen. In den meisten Fällen ist das Wissen in abstrakter Form, wie Text, abgelegt und damit nicht direkt für einen technischen Prozess nutzbar. Aus dem Inhalt des Wissenscontainers werden dann technisch nutzbare Sicherheitsanforderungen erstellt und abgelegt.

### 3.3.3 Ableiten der Testfälle

Die in den vorherigen Schritten erstellten Szenarien und Anforderungen werden nach Definition 3.8 zu Testfällen zusammengefügt. Dabei wird festgelegt, ab wann ein Szenario als sicher gilt (s. Herausforderung 3 und Definition 3.5), indem Pass-Fail-Kriterien für das Szenario aus den zugehörigen Anforderungen abgeleitet werden. Zudem werden die für die Sicherheitsargumentation (s. Abschnitt 3.3.6) relevanten und repräsentativen Testfälle ausgewählt und parametrisiert.

Für die Auswahl der Testszenarien werden in einem ersten Schritt die ODD relevanten und repräsentativen logischen Szenarien aus der Szenariendatenbank ermittelt.

**Definition 3.11** (Relevantes Szenario). *Relevante Szenarien befinden sich innerhalb oder am Grenzbereich der ODD und weisen Merkmale auf, die einen Einfluss auf die HAF haben.*

**Definition 3.12** (Repräsentatives Szenario). *Repräsentativ zu sein bedeutet, eine Menge stellvertretend unter Berücksichtigung von deren Merkmalen darzustellen. Ein **repräsentatives Szenario** ist damit typisch für eine Gruppe an Szenarien und beinhaltet in Bezug auf Stichproben typische Fälle hinsichtlich mehrerer oder aller Merkmale (vgl. [89]).*

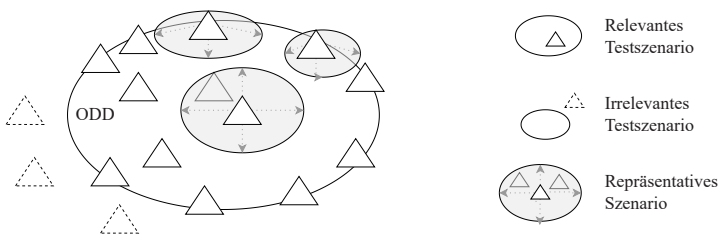


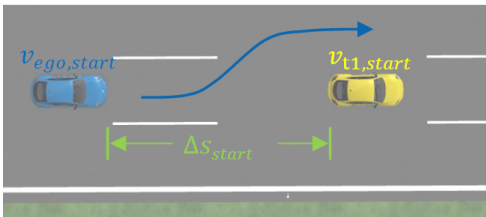
Abbildung 3.8: Repräsentative Szenarien als Stützstellen für die Testabdeckung innerhalb der ODD nach [90]

Um Redundanz im Testen zu vermeiden und die Effizienz zu erhöhen, werden die repräsentativen Szenarien stellvertretend für eine Szenariengruppe mit ähnlichen Merkmalen getestet (s. Abbildung 3.8).



In einem zweiten Schritt werden die Parameter der ausgewählten logischen Szenarien ermittelt und konkrete Szenarien erstellt. Jedoch kann selbst unter dem Einsatz von Simulationstools keine vollständige Testabdeckung (ausschließlich) systematisch durch rein künstliche Verfahren wie Parametervariation und -kombinatorik sichergestellt werden, wie folgende Beispielrechnung zeigt (s. Abbildung 3.9) [36]:

### Funktionsrelevante Variationen



$$\begin{aligned}\Delta s_{start} &= \{z \mid \exists k \in \mathbb{N} : z = 10k \wedge 20 < z < 200\} \text{ in m} \\ v_{ego,start} &= \{x \mid \exists k \in \mathbb{N} : x = 5k \wedge 50 < x < 150\} \text{ in km/h} \\ v_{t1,start} &= \{y \mid \exists k \in \mathbb{N} : y = 5k \wedge 50 < y < 150\} \text{ in km/h}\end{aligned}$$

$$\begin{aligned}v_{ego,s.} \times v_{t1,s.} \times \Delta s_{start} \\ = 21 \times 21 \times 19 \\ = \mathbf{7.600 \text{ Variationen}}\end{aligned}$$

### Sensorrelevante Variationen

z.B. je 3 Ausprägungen in den Dimensionen:  
Niederschlag, Sichtweite, Tageszeit,  
Sonnenstand, Temperatur, Luftdruck,  
GPS-Empfang, Reibwert

$$\begin{aligned}3^8 \\ = \mathbf{6.561 \text{ Variationen}}\end{aligned}$$

Abbildung 3.9: Mögliche Variationen für die Startszene eines Überholszenarios [36]

Für ein einfaches Überholszenario werden die Start-Parameter für den Abstand zweier Fahrzeuge  $\Delta s_{start}$  und deren Geschwindigkeit  $v_{ego,start}$  und  $v_{t1,start}$  variiert. Mit den in Abbildung 3.9 gezeigten Diskretisierungen für funktionsrelevante Parameter erhält man bereits 7.600 Variationen. Für jeweils drei Ausprägungen für Umgebungs- bzw. Wetterbedingungen erhält man 6.561 Variationen. Kombiniert man beide Variationen, ergeben sich ungefähr 50 Mio. Variationen ( $7.600 \times 6.561$ ) des Szenarios. Mit der Annahme, pro Szenario 30 Sekunden für die Ausführung zu benötigen, ist für dieses einfache Autobahnszenario bereits ein Rechenaufwand von über 47 Jahren notwendig [36]. Für urbane Szenarien mit mehreren Verkehrsteilnehmern, Interaktionen und Variationen lässt sich der Aufwand weiter steigern.

Da also nicht alle Szenarien mit allen Parametervariationen ausgeführt werden können, der Anteil sicherer Szenarien aber maximiert wird (s. Herausforderung 3), müssen die auszuführenden Szenarien so gewählt werden, dass die Testabdeckung bei einem Testfall effizient erhöht wird bzw. Redundanz in den Ausführungen vermieden wird.

**Herausforderung 4.** *Die Anzahl der auszuführenden Testszenarien muss bei möglichst hoher Testabdeckung reduziert werden.*

Um effizient eine hohe Testabdeckung zu erreichen, ist ein Ziel der Testfall-Ableitung, die für die Sicherheitsargumentation relevanten Szenarien und deren Parameter zu ermitteln. So kann die mehrfache Ausführung von Testfällen mit redundanten bzw. ähnlichen Szenarien vermieden werden. Für die Auswahl der Szenarien bzw. Parametervariationen eines logischen Szenarios existieren zwei Vorgehensweisen [91]:

Bei der Abtastung des Parameterraums werden die zuvor definierten Parameterbereiche bzw. -verteilungen der logischen Szenarien genutzt, um über Methoden aus der Statistik (z.B. N-wise Sampling, Monte-Carlo-Simulation, Extremwerttheorie) ein Set an konkreten Szenarien zu definieren. Das Ziel dieser Methoden ist es, den Szenarienraum möglichst breit und realitätsnah abzutasten und einen Überblick über die Testabdeckung zu erhalten.

Im Gegensatz dazu werden bei der Auswahl der Szenarien über Falsifizierung gezielt besonders kritische Szenarien gesucht, die die Sicherheitsanforderungen verletzen. Dazu wird entweder Expertenwissen genutzt, der Inhalt von Unfalldatenbanken ausgewertet, ein konkretes Szenario ausgewählt und die Kritikalität für dieses erhöht oder ein logisches Szenario ausgewählt und kritische Parametervariationen über einen Optimierungsalgorithmus gefunden.

#### 3.3.4 Ausführen der Testfälle

Die Testfälle werden auf die in Abschnitt 2.2 vorgestellten Testmittel verteilt, dort ausgeführt und die für die Auswertung notwendigen Daten gesammelt. Die Aufteilung erfolgt unter der Berücksichtigung der Eigenschaften der Testmittel [43] [92]. Virtuelle Testmittel und Modelle müssen zuvor auf ihre Validität überprüft werden [93] [94].

### 3.3.5 Evaluation der Testfälle

Bei der Test-Evaluation werden beide Elemente des Testfalls (s. Abbildung 3.2), das Szenario und die Pass-Fail-Kriterien, evaluiert, um die Sicherheit des Testobjektes im Szenario nachzuweisen (s. Herausforderung 3). Bei der Szenario-Evaluation wird überprüft, ob das Szenario wie spezifiziert ausgeführt wurde. Gründe für nicht valide ausgeführte Szenarien können ein frühzeitiger Abbruch der Ausführung sein (z.B. Programmabsturz des Simulationstools) oder eine Parameterkombination, die zu ungewünschtem Verhalten führt. Über die Pass-Fail-Kriterien wird evaluiert, ob das Testobjekt die an es gestellten Anforderungen (s. Abschnitt 3.3.2) erfüllt. Die so erzeugten Testergebnisse bilden die Grundlage für die Sicherheitsargumentation.

### 3.3.6 Sicherheitsargumentation

Für den Open World Kontext lässt sich keine finale Sicherheitsaussage treffen (s. Abschnitt 1.1.3), deshalb wird eine Argumentation über die „absence of unreasonable risk“ geführt [90]: Wie in Abschnitt 3.2.4 beschrieben, wird dabei argumentiert, dass die Wahrscheinlichkeit für neue unbekannte Szenarien gering ist (s. Herausforderung 2) und dass das Testobjekt in den bekannten Szenarien sicher agiert (s. Herausforderung 3). Durch Kombination beider Aussagen lässt sich die Wahrscheinlichkeit für unsichere Szenarien nach oben abschätzen. Diese muss für das Erreichen eines zuvor definierten Sicherheitsziels unter einem Schwellwert liegen. Mögliche Sicherheitsziele sind z.B. Akzeptanzkriterien der Gesellschaft, wie seltener auftretende tödliche Unfälle im Vergleich zu menschlichen Fahrern (s. Abschnitt 1.1.1).

### 3.3.7 Verknüpfung mit dem V-Modell

Wird das szenariobasierte Testen mit dem V-Modell verknüpft (s. Abbildung 3.10), bilden Szenarien dabei wiederkehrende Bezugspunkte und ein Gerüst, das den Entwicklungsprozess systematisch begleitet. Somit bieten sie eine gemeinsame Sicht für die an der Entwicklung beteiligten Personen, wie Entwickler, Tester und Kunden.

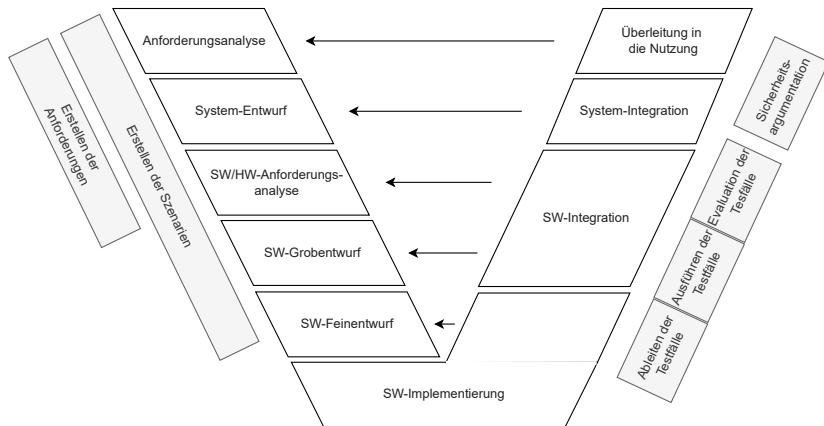


Abbildung 3.10: Verknüpfung des SBT-Prozesses (s. Abschnitt 3.3) mit dem V-Modell (s. Abschnitt 2.1.4)

Die Schritte “Erstellen der Szenarien und Anforderungen” starten im Rahmen der Anforderungsanalyse. Dazu werden die Anforderungen an das System und dessen ODD mithilfe von Szenarien erfasst. Sie beschreiben die erwarteten Anwendungsfälle des Systems aus der Sicht der Benutzer. Diese werden als funktionale Szenarien (s. Abschnitt 3.2.3) in natürlicher Sprache formuliert.

In den Entwurfsphasen dienen die Szenarien als Orientierung, um die Systemarchitektur auszulegen. Parallel zur technischen Detaillierung des Systems werden auch die Szenarien weiter formalisiert und in logische Szenarien überführt. Verschiedene Design-Optionen werden anhand der Szenarien eingeschätzt und ihre Auswirkungen auf diese analysiert.

Bei der Software- und Systemintegration werden die Szenarien zum Testen verwendet. Dazu werden relevante, konkrete Szenarien für die Ausführung abgeleitet und mit zugehörigen Anforderungen zu Testfällen kombiniert (s. Definition 3.8). Die Szenarien eignen sich dabei sowohl für die Komponenten- als auch für die Systemtests.

Der abschließende Akzeptanztest (s. Abschnitt 2.1.4) wird durch die gesamte Szenariendatenbank mit der Sicherheitsargumentation (s. Abschnitt 3.3.6) verknüpft, um das System validiert in die Nutzung überzuleiten. Es wird da-

bei überprüft, ob die Anforderungen in den Szenarien der definierten ODD gesamtheitlich erfüllt werden.

Für die iterative Nutzung des V-Modells im Rahmen einer agilen Entwicklung (s. Abbildung 2.4) können sich die Szenariendatenbank und die zugehörigen Tests bei jeder Iteration ändern. So werden neue, bisher unbekannte Szenarien in den Prozess integriert, um schrittweise den Reifegrad des Systems zu steigern.

## 3.4 Eigenschaften von Szenarien

Szenarien sind das Kernelement des szenariobasierten Testens. Im Absicherungsprozess (s. Abschnitt 3.3) werden sie vielseitig eingesetzt und unterscheiden sich darin in der Art der Beschreibung, im Ursprung, Verwendungszweck, dem Detailgrad und der enthaltenen Information.

### 3.4.1 Generieren von Szenarien

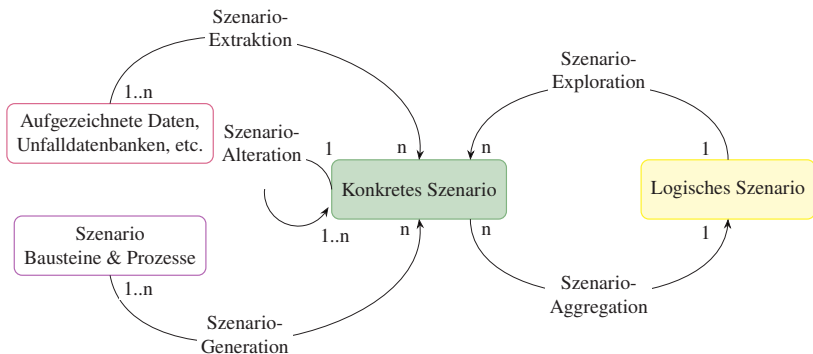


Abbildung 3.11: Input, Output und Beziehungen der Kategorien zum Erstellen von Szenarien nach Schütt et al. [TB11].

Das Generieren von neuen Szenarien lässt sich in 6 Kategorien einteilen (s. Abbildung 3.11)<sup>1</sup>:

Bei der *Szenario-Alteration* werden die Parameter eines konkreten Szenarios verändert, um ein Ziel zu erreichen, z. B. die Kritikalität zu maximieren. Dazu werden vielfältige Algorithmen verwendet, u.a. Bayessche Optimierung in Abeyisirigoonawardena et al. [95], Wang et al. [96] und Gangopadhyay et al. [97], evolutionäre Optimierung in Klischat et al. [98] und Wang et al. [96] sowie Reinforcement Learning in Karunakaran et al. [99].

Auch bei der *Szenario-Exploration* werden neue Szenarien durch Parametervariation erstellt. Ausgangspunkt ist, im Gegensatz zur Szenario-Alteration, ein logisches Szenario. Ein Optimierungsverfahren zieht dabei Werte aus den Parameterverteilungen des logischen Szenarios, um unterschiedliche und neue konkrete Szenarien zu finden. Auch bei der Szenario-Exploration werden Bayessche Optimierung in Schütt et al. [100], evolutionäre Optimierung in Bussler et al. [101] und Abdessalem et al. [102] sowie Reinforcement Learning in Ding et al. [103], Baumann et al. [104] und Kuutti et al. [105] eingesetzt.

Bei der *Szenario-Generation* werden existierende konkrete Szenario-Bausteine manuell oder algorithmisch ausgewählt, kombiniert und modifiziert, um neue Szenarien zu erstellen. Die Bausteine können entweder manuell aufgebaut sein oder datengetrieben aus beobachteten Szenarien erzeugt werden. Bei der manuellen Szenario-Erstellung werden Szenarien von Experten aufgebaut. Menschliche Experten können Szenarien auf allen Abstraktionsebenen (s. Abschnitt 3.2.3) erstellen. Insbesondere werden funktionale Szenarien, deren Fokus auf menschlichem Verständnis und weniger auf maschineller Anwendung liegt, manuell erstellt [106]. Die Beschreibung der erstellten Szenarien variiert von einfachen Skizzen über formalisierte grafische Darstellungen bis zu einer Domain Specific Language (DSL) (s. Abschnitt 3.4.5). Um komfortabel manuell ausführbare Szenarien zu erstellen, bieten die meisten Simulationswerkzeuge (s. Abschnitt 3.4.2) grafische Modellierungswerkzeuge an.

Im Gegensatz zu den anderen Kategorien wird die Szenario-Generation auch eingesetzt, um statische Elemente eines Szenarios zu verändern. Verwendete Algorithmen sind u. a. prozedurale Ansätze von Li et al. [61] und Paranjape et al. [107], Reinforcement Learning (RL) von Rietsch et al. [108], und

---

<sup>1</sup> Diese Kategorisierung wurde in Schütt et al. [TB11] veröffentlicht.

Ontologien in Bagschik et al. [48]. Datengetriebene Methoden sind Encoder-Decoder-Architekturen von Feng et al. [109] und Rekombination extrahierter atomarer Bausteine von Goss et al. [110] bzw. Manöver in Ghodsi et al. [111].

Die *Szenario-Extraktion* extrahiert konkrete Szenarien aus Realdaten (vgl. beobachtete Szenarien in Abschnitt 3.4.4). Die Extraktion erfolgt z.B. durch Abstraktion der beobachteten Trajektorien zu Manövern (s. Abschnitt 3.1), Occupancy Grids in Weber et al.[112] und Balasubramanian et al. [113] oder Betrachten der statischen Umgebung in Langner et al. [84]. Eine weitere Quelle für die Szenario-Extraktion sind Unfalldatenbanken, z.B. in Xinxin et al. [114], Gambi et al. [115] und Cao et al.[116]

Die *Szenario-Aggregation* gruppiert ähnliche konkrete Szenarien zu logischen Szenarien. Sie wird in der Regel im Anschluss an eine Szenario-Extraktion ausgeführt, um mehrfach beobachtete Szenarien und deren Häufigkeit zu identifizieren. Die Aggregation kann mittels verschiedener Methoden erfolgen: z.B. über das Clustern erkannter Features in Langner et al.[84], Weber et al. [112] Hauer et al. [117], und Kerber et al. [118], über Abstraktion zu Manövern in King et al. [TB7] und Hartjen et al. [71] bzw. eines Occupancy Grids in Weber et al.[112], Kerber et al. [118] und Ries et al. [119], oder über neuronale Netze in Balasubramanian et al. [113].

### 3.4.2 Simulation von Verkehrsszenarien

Das Testen in einer Simulation bietet Vorteile gegenüber der Ausführung in der realen Welt (s. Abschnitt 2.2): Es lassen sich die Kosten für den Bau von physischen Prototypen vermeiden und die Ausführung der Szenarien kann flexibler und schneller durchgeführt werden. Zudem können potenziell gefährliche Situationen risikofrei in der virtuellen Umgebung getestet werden. Allerdings besteht bei einer Simulation immer eine Lücke zur Realität, weil nicht alle Aspekte perfekt nachgebildet werden. Unerwartete Ereignisse und wichtige Details werden so nicht komplett über eine Simulation abgedeckt. Zudem ist der initiale Aufwand für die Entwicklung und Validierung von Modellen vorhanden.

Die Simulation des Verkehrs wird dabei in die drei Kategorien makroskopisch, mikroskopisch und mesoskopisch eingeteilt [120]. Die *makroskopische Simulation* fokussiert die Verkehrsströme von größeren Mengen an Fahrzeugen.

Diese werden durch über mehrere Verkehrsteilnehmer aggregierte Parameter wie Durchschnittsgeschwindigkeiten und Dichten charakterisiert. Die mesoskopische Betrachtung eignet sich daher für Verkehrsplanung in Städten oder Regionen, bei denen Engpässe identifiziert oder der Einfluss von geplanter Infrastruktur simuliert werden kann. Da das Verhalten einzelner Verkehrsteilnehmer nicht betrachtet wird, eignet sich diese Art nicht für das Testen von HAF.

In der *mikroskopischen Simulation* wird das Verhalten und die Interaktion einzelner Verkehrsteilnehmer berücksichtigt, indem spezifische Modelle für diese eingesetzt werden. Diese ermöglichen die explizite Beschreibung von z.B. Beschleunigungsvorgängen, Spurwechseln und weiteren Manövern. Durch die detailliertere Modellierung ist der Bedarf an Rechenleistung höher, was die Betrachtung ganzer Regionen oder Städte wie bei der makroskopischen Simulation erschwert. Die mikroskopische Betrachtung erfüllt dafür die Bedingungen für das Entwickeln, Optimieren und Testen von HAF.

Die *mesoskopische Simulation* kombiniert Modelle aus der mikroskopischen und makroskopischen Simulation miteinander. Die relevanten Bereiche werden durch detaillierte mikroskopische Modelle simuliert, die weiteren Bereiche durch die größeren makroskopischen Modelle. Dadurch können für spezifische Einsatzzwecke individuelle Kompromisse aus Detailgrad und Rechenleistung erstellt werden.

Für diese Ausarbeitung wird weitergehend nur die mikroskopische Simulation betrachtet, welche die Anforderungen für das szenariobasierten Testen von HAF erfüllt. Etablierte kommerzielle Tools für die Ausführung sind z. B. CarMaker von IPG [121], Virtual Test Drive von Vires [122] und ControlDesk bzw. ASM von dSpace [123]. Auf der Open-Source Seite existieren u. a. die Tools CARLA [124] und esmini [125].

### 3.4.3 Ähnlichkeiten und Distanzen zwischen Szenarien

Ähnlichkeiten und Distanzen zwischen Szenarien werden im szenariobasierten Testen an verschiedenen Stellen genutzt. Beide Begriffe beschreiben das gleiche Prinzip, eine hohe Ähnlichkeit bedeutet eine geringe Distanz und umgekehrt.



Eine allgemeine Definition für Distanzmaße liefert Backhaus et al. [126], welche von Braun et al. [TB2] für den Kontext Verkehrsszenarien konkretisiert wurde.

**Definition 3.13.** Ein *Szenario-Distanzmaß*  $d$  quantifiziert die Unähnlichkeit zweier Szenarien  $S^a$  und  $S^b$  basierend auf einer Auswahl von Merkmalen  $F^a$  und  $F^b$ . Identische Szenario-Merkmale implizieren ein Distanzmaß von null.

$$d(S^a, S^b) = f_d(F^a, F^b) \quad (3.1)$$

Nach dieser Definition muss  $d$  die folgenden Bedingungen erfüllen [127]:

$$d(S^a, S^a) = 0 \quad (3.2)$$

$$d(S^a, S^b) \geq 0 \quad (3.3)$$

$$d(S^a, S^b) = d(S^b, S^a) \quad (3.4)$$

Weitergehend wird, aufbauend auf der allgemeinen Definition nach Backhaus et al. [126], ein Maß für die Ähnlichkeit von Szenarien definiert:

**Definition 3.14.** Ein *Szenario-Ähnlichkeitsmaß*  $\sigma$  quantifiziert die Ähnlichkeit zweier Szenarien  $S^a$  und  $S^b$  basierend auf einer Auswahl von Merkmalen  $F^a$  und  $F^b$ .

$$\sigma(S^a, S^b) = f_\sigma(F^a, F^b) \quad (3.5)$$

Ein Ähnlichkeitsmaß für Szenarien muss folgende Bedingungen erfüllen [127]:

$$\sigma(S^a, S^a) \geq \sigma(S^a, S^b) \quad (3.6)$$

$$\sigma(S^a, S^b) = \sigma(S^a, S^b) \quad (3.7)$$

$$\sigma(S^a, S^b) \geq 0 \quad (3.8)$$

$$\sigma(S^a, S^a) = 1 \quad (3.9)$$

In Braun et al. [TB2] werden fünf Einsatzmöglichkeiten für Szenarioähnlichkeiten bzw. -distanzen ermittelt und in den SBT-Prozess für szenariobasiertes Testen (s. Abschnitt 3.3) eingeordnet. Das *unüberwachte Sammeln von Szenarien* deckt sich mit der Szenario-Extraktion und -Aggregation und wird beim Erstellen der Szenarien eingesetzt. Das Ähnlichkeitsmaß wird dabei verwendet, um die Gruppierung der extrahierten Szenarien zu erstellen. Auch die *Anomalie-Detektion* wird in diesem Zusammenhang eingesetzt, dabei werden in aufgezeichneten Daten unbekannte und seltene Szenarien gefunden, welche sich durch eine niedrige Ähnlichkeit zu allen bisher bekannten Szenarien auszeichnen. Bei der *Szenario-Identifikation* werden von Experten spezifizierte Szenarien in aufgezeichneten Datensätzen ermittelt, die eine hohe Ähnlichkeit zum spezifizierten Szenario aufweisen. Die *Reduktion auszuführender Szenarien* ermittelt im Schritt Ableiten der Testfälle ähnliche Szenarien, um die Anzahl auszuführender Szenarien zu reduzieren. Dabei wird angenommen, dass Tests in einem Testszenario repräsentativ (s. Definition 3.12) für weitere, ähnliche Szenarien ausgeführt werden können. Die *Validierung von Simulationsumgebungen* überprüft, wie ähnlich ausgeführte Szenarien einer Simulationsumgebung zu real aufgezeichneten Fahrten sind, um somit die Simulationsumgebung zu bewerten. Diese Validität ist Voraussetzung, um Simulationsergebnisse nach Ausführung nutzen zu können.

Weitergehend werden in [TB2] drei Kategorien zur Berechnung von Ähnlichkeiten vorgestellt. Bei der *Berechnung durch Zeitreihen* werden die Zeitreihen aufgezeichneter Szenarien, z.B. Trajektorien, direkt miteinander verglichen. Dazu eignen sich Zeitreihendistanzmaße wie die Hausdorff-, Euklidische- oder Frechet-Distanz und das Dynamic Time Warping (DTW). Die *Berechnung über Abstraktion* verdichtet die relevanten Aspekte der Szenarien durch eine Modellierung. Die Berechnung der Distanz bzw. Ähnlichkeit erfolgt dann auf der Modellinstanz. Das Modell kann manuell als semantisches Modell oder automatisch durch Methoden des maschinellen Lernens abgeleitet werden. Die

*Berechnung über Kennzahlen* kondensiert die relevante Information noch stärker auf skalare Kennzahlen, welche dann miteinander verglichen werden. Typische Kennzahlen sind die Kritikalität des Szenarios und die Performance des Testobjekts.

	Zeitreihen	Abstraktion	Kennzahlen
Sammeln von Szenarien	[TB9, 128, 129, 130, 89, 117, 131]	[TB7, 71, 132, 133, 112, 134, 118, 135, 117, 136]	[137, 89, 138]
Szenario-Identifikation		[139, 140, 141]	
Anomaliedetektion	[128, TB9]	[TB7, 142]	
Reduktion auszuführen- der Szenarien		[143, 144]	[137, 145, 146, 147]
Validierung von Simulati- onsumgebungen	[94, 148]	[148]	[149, 150, 151, 148, 152]

Tabelle 3.3: Überblick über Literatur zu Ähnlichkeiten von Szenarien, deren Anwendungsmöglichkeiten im szenariobasierten Testen und Berechnungsmethodik.

Tabelle 3.3 zeigt bestehende Ansätze zu Distanzen bzw. Ähnlichkeiten und ihre Einordnung in die vorgestellten Kategorien zur Anwendung und Berechnung. Zeitreihenvergleiche werden dabei typischerweise für die unüberwachte Sammlung von Szenarien verwendet und können direkt mit den aufgezeichneten Daten ohne weitere Verarbeitung arbeiten.

Ein etabliertes Maß ist z.B. die DTW-Distanz, die von Ries et al. [TB9] in Kombination mit Metadaten der Szenarien verwendet wird. Watanabe [89] nutzt ebenfalls DTW, um Einscher-Szenarien anhand der Geschwindigkeiten der Teilnehmer zu clustern. Eine Alternative ist die Hausdorff-Distanz, die von Siebinga et al. [128] angewendet wird, um die Trajektorien des highD-Datensatzes zu vergleichen.

Abstraktionsverfahren sind nach dieser Auswertung vor allem für das Sammeln von Szenarien weit verbreitet. Die Erkennung von Anomalien durch Abstraktion ist prinzipiell ebenso möglich, doch durch die Reduktion auf ein abstraktes Modell können Eigenschaften von Szenarien verloren gehen, die

für deren Erkennung relevant sind. Dies gilt insbesondere für Anomalien auf Zeitreihenebene.

Mögliche Arten für die Abstraktion sind u. a. Manöver, welche von Braun et al. [TB3], King et al. [TB7] und Hartjen et al. [71] für den Vergleich von Szenarien verwendet werden. Eine Alternative ist das Occupancy Grid. Dabei wird die unendliche Anzahl von (relativen) Fahrzeugpositionen auf eine endliche Anzahl von Zuständen in einem Belegungsgitter reduziert, das um ein Fahrzeug gelegt wird. Dieser Ansatz wird z.B. von Balasubramanian et al. [113] in Kombination mit einem neuronalen Netzwerk genutzt. Weber et al. [112] erstellen mit einem vergleichbaren Prinzip Cluster anhand von Momentaufnahmen des Occupancy Grids.

Ähnlichkeit über Kennzahlen wird zur Testreduzierung und XiL-Validierung verwendet, da sie nicht nur auf Repräsentativität, sondern auch auf die Kritikalität von Szenarien abzielen kann. Auch wenn es nur wenig Literatur über die Identifikation spezifizierter Szenarien und die Erkennung von Anomalien mithilfe von Kennzahlen gibt, werden diese Maße implizit und häufig verwendet, wenn z. B. die kritischsten Szenarien gesucht werden.

Rasch et al. [145] verwendet etwa die Kritikalität von Szenarien, um die Anzahl an Testfällen zu reduzieren. Reisgys et al. [151] nutzt eine Kombination aus mehreren Kennzahlen, um die Validität von Testplattformen zu bestimmen.

### 3.4.4 Aufgezeichnete und ausführbare Szenarien

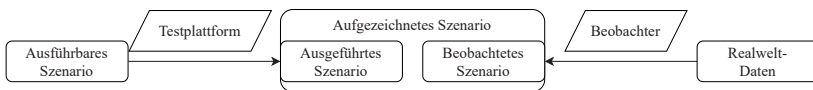


Abbildung 3.12: Unterschiedliche Typen von Szenarien nach Quelle und enthaltenen Informationen

Je nachdem, aus welcher Quelle das Szenario erstellt ist und welche Informationen enthalten sind, lassen sich Szenarien als ausführbar, aufgezeichnet und Realwelt-Daten kategorisieren. Eine Möglichkeit, Szenarien für Tests zu verwenden, besteht darin, sie auf Testplattformen auszuführen (Abschnitt 2.2). In diesem Zusammenhang enthält ein *ausführbares Szenario* alle für die Ausführung erforderlichen Informationen, insbesondere die Anweisungen für das Verhalten jedes Verkehrsteilnehmers und der Straßeninfrastruktur. *Ausgeführte*

*Szenarien* enthalten die aufgezeichneten Ergebnisse der ausgeführten Szenarien, wie z.B. die Trajektorien der Teilnehmer. Alternativ zur gezielten Ausführung können Szenarien durch Daten der Verkehrsbeobachtung, die von Testfahrzeugen, Drohnen, spezieller Infrastruktur oder Verkehrssimulationen gesammelt wurden, erstellt werden. Im Gegensatz zu den ausführbaren Szenarien ist das Verhalten der Teilnehmer dabei nicht explizit definiert. Szenarien, die durch Realwelt-Daten erstellt wurden, sind als *beobachtete Szenarien* definiert. Sie enthalten die gleiche Art von Informationen wie ausgeführte Szenarien, der einzige Unterschied zwischen ausgeführten und beobachteten Szenarien ist deren Ursprung. Daher werden sie unter dem Begriff *aufgezeichnete Szenarien* zusammengefasst.

Beispiele für ausführbare Szenarien sind die Testszenarien für die Zulassung, wie z. B. in der ISO 15622 [153] für Adaptive Cruise Control (ACC)-Systeme, und manuell erstellte Szenarien für Simulationstools. Sie werden im Schritt Ausführung der Testfälle (s. Abschnitt 3.3.4) eingesetzt. Beispiel für beobachtete Szenarien sind der GIDAS-Unfallkatalog, in dem gezielt vorgefallene Unfälle aus Realwelt-Daten festgehalten werden, von Drohnen aufgezeichnete Trajektorien-Datensätze wie der Interaction-Datensatz [154], levelxdata [155, 156, 157, 158] und von Verkehrsinfrastruktur aufgezeichnete Daten [159, 160]. Sie werden im Schritt Erstellen von Szenarien (s. Abschnitt 3.3.1) als datenbasierte Methode eingesetzt.

### 3.4.5 Beschreibungsform, intuitive Nutzbarkeit und automatisierte Anwendung

**Definition 3.15.** *Die Szenariobeschreibungsform beschreibt die strukturierte Darstellungsweise bzw. das verwendete Format um Informationen in einem Szenario zu dokumentieren.*

Szenarien lassen sich in verschiedenen Formen und Formalisierungen darstellen. Ein Spannungsfeld ergibt sich dabei zwischen der intuitiven Nutzbarkeit und Interpretierbarkeit für Menschen (im Weiteren verkürzt als Intuitivität bezeichnet) und der automatisierten Anwendbarkeit für z.B. Computer. Während intuitive Szenarien vorwiegend beim Erstellen von Szenarien, Anforderungen und der Sicherheitsargumentation verwendet werden, ist die automatisierte

Anwendbarkeit bei der Ausführung und Evaluation nötig, um die notwendige Anzahl an Szenarien effizient testen zu können (s. Abschnitt 3.3).

Die *verbale Beschreibung* verwendet natürliche Sprache, um Szenarien zu beschreiben. Ein Beispiel dafür sind erste Entwürfe von Szenarien und funktionalen Testfällen (s. manuelle Szenario-Erstellung in Abschnitt 3.4.1). Größter Vorteil dieser Art ist die hohe Intuitivität. Die fehlende Formalisierung natürlicher Sprache führt aber auf der anderen Seite dazu, dass sie nicht ohne Weiteres maschinell verarbeitet werden können und somit die Szenarien z. B. von Simulationstools nicht ausführbar sind. Die verbale Beschreibung wird daher in den frühen Phasen des Entwicklungsprozesses verwendet, wenn der Fokus auf dem Inhalt oder dem Austausch von Szenarien und nicht auf deren Ausführung liegt.

*Visuelle Beschreibung* verwendet bildliche Darstellungen des Szenarios, z. B. Momentaufnahmen in der Vogelperspektive. Ähnlich wie die verbale Form ist sie intuitiv verständlich, aber mangels Formalisierung für Maschinen nicht automatisierbar nutzbar. Sie eignet sich als Ergänzung zur verbalen Beschreibung, um einen grundlegenden Überblick über ein Szenario festzuhalten und um diesen mit anderen zu teilen. Diese Form wird z. B. in der ISO 15622 [153] und der ISO/PAS 21448 [75] für die Spezifikation von ACC-Szenarien genutzt.

*Formalisierte verbale und visuelle Beschreibung* systematisieren die verbalen und visuellen Formen durch ein definiertes Vokabular und Regeln. Diese Methode ist ein Kompromiss zwischen menschlicher und maschineller Anwendbarkeit: Es sind nur noch Szenarien beschreibbar, die innerhalb des Vokabulars auch abbildbar sind, und ein Erlernen der Regeln und des Vokabulars ist notwendig. Dafür ist eine automatisierte Anwendung (teilweise) möglich.

Menzel et al. [161] schlagen eine Ontologie als Beschreibungsform vor, um die Umwandlung von verbalen funktionalen Szenarien in logische und konkrete zu vereinfachen. In dieser Ontologie sind verwendbare Vokabeln, Parameter, deren Zusammenhänge und einzuhaltende Rahmenbedingungen festgehalten. Damm et al. [162] definieren sogenannte Traffic Sequence Charts, um Szenarien und deren Logik visuell abzubilden. Ein Szenario besteht darin aus einer Reihe von Momentaufnahmen und relevanten Informationen über die Teilnehmer und Events. Für den Aufbau dieser Momentaufnahmen werden ausschließlich Elemente in einem zuvor definierten Symbollexikon verwendet. Bauer et al. [163] beschreiben ausführbare Szenarien visuell über Behaviour

Trees. Das Verhalten der Teilnehmer wird dabei als Knoten eines mathematischen Baums definiert. Durch die Kompositionsmöglichkeiten parallel und sequenziell lässt sich der Ablauf des Szenarios im Baum genauer definieren.

Bei *modellbasierter Beschreibung* und *Domain Specific Language* liegt der Fokus auf der automatisierten maschinellen Anwendbarkeit. Bei der modellbasierten Form wird ein technisches Metamodell erstellt, das alle relevanten Entitäten und Beziehungen abbildet [164]. Bei Beschreibungen über eine DSL werden die Szenarien über eine eigens dafür entwickelte Programmiersprache beschrieben. Beide Beschreibungsarten zeichnen sich durch eine enge Kopplung mit den auszuführenden Programmen aus, haben aber eine niedrige Intuitivität. Diese kann durch eine grafische Schnittstelle zur manuellen Erstellung und Bearbeitung erhöht werden.

OpenSCENARIO XML[54] verwendet das Format XML, um ausführbare Szenarien zu definieren. Die Struktur und die verwendbaren Elemente werden in einem zugehörigen XML-Schema definiert. Darin sind alle verwendbaren Teilnehmer, Aktionen und deren Parameter aufgeführt. Zusätzlich wird der Ablauf des Szenarios über sogenannte Stories, Acts, Maneuver Groups und Maneuver definiert. Bach et. al [29] nutzen für die Resimulation von aufgezeichneten Verkehrsszenarien ein Ecore-Modell. Verwendbare Applikationen bauen direkt auf diesem Modell auf, was eine nahtlose Nutzung zwischen Szenarien und Tools ermöglicht. Der Ablauf der Szenarien wird über Manöver und Akte beschrieben. OpenSCENARIO DSL [58] verwendet für die Simulation von Szenarien eine eigens für diesen Zweck definierte Programmiersprache. Der Fokus liegt auf der Wiederverwendbarkeit von bereits definierten Szenarien bzw. von einzelnen Teilen der Szenarien. Die Semantik der Sprache verwendet u.a. Manöver, die parallel und sequenziell verschaltet werden können. Die Syntax ist an Python angelehnt.





## 4 Anforderungen an die Szenariobeschreibungsform

### 4.1 Ableiten der Anforderungen

Wie in den vorherigen Kapiteln gezeigt, werden Szenarien in allen Prozessschritten des szenariobasierten Testens verwendet (s. Abschnitt 3.3). Je nach Prozessschritt werden dabei verschiedene Anforderungen an die Beschreibungsform gestellt, was zu den in Abschnitt 3.4 gezeigten unterschiedlichen Eigenschaften und Formen der beschriebenen Szenarien führt. Zusammenfassend lassen sich folgende acht Anforderungen an die Szenariobeschreibungsform herleiten. Abbildung 4.1 zeigt zudem, in welchen Prozessschritten und für welche Herausforderungen die Anforderungen jeweils am relevantesten sind.

**Anforderung 1.** *Die beschriebenen Szenarien müssen **vergleichbar** sein. Zwei Szenarien müssen als gleich oder ungleich gekennzeichnet bzw. über eine Ähnlichkeit miteinander in Beziehung gebracht werden können.*

Wie in Abschnitt 3.4.3 beschrieben, werden Szenarien in mehreren Prozessschritten miteinander verglichen: Beim datenbasierten Erstellen der Szenarien und bei der Exploration des Szenarienraums müssen gleiche Szenarien in den Daten erkannt werden, um diese als bekannt oder unbekannt zu klassifizieren. Für bekannte Szenarien gibt es dabei mindestens ein gleiches Szenario in den bisher analysierten Daten, für unbekannte Szenarien ist dies nicht der Fall. Die Menge verbleibender unbekannter Szenarien lässt sich dann durch eine Sättigungsanalyse der Auftretenswahrscheinlichkeit unbekannter Szenarien abschätzen (s. Abschnitt 3.1.5).

Für das Ableiten von Testfällen und für eine möglichst effiziente Testabdeckung muss die Ausführung redundanter Szenarien vermieden werden. Diese Redundanz kann über eine Ähnlichkeit zwischen den Szenarien beschrieben

werden. Bei der Evaluation der Testfälle wird das ausgeführte mit dem zuvor beschriebenen Testszenario verglichen, um zu überprüfen, ob dieses wie spezifiziert ausgeführt wurde.

**Anforderung 2.** *Die beschriebenen Szenarien müssen **ausführbar** sein. Die Beschreibung muss dafür Anweisungen an die Verkehrsteilnehmer beinhalten.*

Um Szenarien auf einer Testplattform auszuführen, müssen sie alle dafür erforderlichen Informationen enthalten. Dies beinhaltet insbesondere die Anweisungen an die Teilnehmer in einer ausführbaren Form (s. Abschnitt 3.4.4). Es müssen Testplattformen, die in Abschnitt 2.2 beschrieben wurden, unterstützt werden. Das bedeutet, dass sowohl Simulationsprogramme als auch menschliche Testfahrer in der Lage sind, die Anweisungen in der Szenariobeschreibung zu verarbeiten.

Die Ausführbarkeit von Szenarien wird beim Ausführen der Testfälle und damit für den Nachweis der Sicherheit des Testobjekts in einem Szenario benötigt.

**Anforderung 3.** *Die Szenariobeschreibungsform muss **erweiterbar** sein. Bisher irrelevante oder unbekannte Szenarioelemente müssen integrierbar sein.*

Zu Beginn der Entwicklung sind noch nicht alle Szenarien und deren Elemente bekannt (s. Abschnitt 3.2.4). Zusätzlich kann sich die Struktur des Verkehrs generell ändern und damit auch die ODD. Wenn neue, relevante Elemente gefunden werden, muss die Beschreibungsform die neuen Elemente integrieren können. Dies schließt sowohl neue Typen von Verkehrsteilnehmern als auch neue oder neu entdeckte Verhaltensweisen ein. Ein Beispiel für neue Teilnehmer sind E-Scooter, die seit 2019 zugelassen werden dürfen und damit in Szenarien vertreten sein können. Durch die Erweiterbarkeit wird sich die Beschreibungsform im Laufe der Zeit weiterentwickeln und an die neuen Herausforderungen anpassen.

Die Erweiterbarkeit wird für die Exploration des Szenarienraums und beim Erstellen der Szenarien benötigt.

**Anforderung 4.** *Der Inhalt der beschriebenen Szenarien muss für Menschen **intuitiv** verständlich sein. Zudem muss das manuelle Erstellen der Szenarien intuitiv sein.*

Die Zielgruppe soll dabei möglichst viele Nutzer beinhalten, z.B. Testingenieure, Personen mit und ohne Programmierkenntnisse und ebenfalls Personen ohne Fachkenntnisse im Automotive- bzw. Testing-Bereich.

Die Intuitivität (s. Abschnitt 3.4.5) ist für manuelle Schritte, wie die manuelle Szenario-Erstellung, das Erstellen der Anforderungen und die Sicherheitsargumentation, erforderlich.

**Anforderung 5.** *Die Szenarien müssen **formalisiert** beschrieben und damit maschinenlesbar sein.*

Während bei manuellen Schritten und funktionalen Szenarien auch eine natürlichsprachliche Beschreibung möglich ist, benötigen automatisierte Prozesse und Programme einen Formalismus, um selbstständig abzulaufen (s. Abschnitt 3.4.5).

Automatisierte Prozessschritte und die damit verbundene formalisierte Beschreibung sind für eine effiziente Testabdeckung nötig. Dies gilt insbesondere für die Ausführung von Testfällen auf SiL-Plattformen.

**Anforderung 6.** *Einzelne Elemente der Szenarien müssen **parametrisierbar** sein. Es muss möglich sein, sowohl Parameterbereiche als auch konkrete Werte zu beschreiben.*

Während des szenariobasierten Testens werden die Szenarien als funktionale, logische und konkrete Szenarien abgebildet. Dies ermöglicht es, mehrere konkrete Szenarien zusammenzufassen und die Unterschiede nur durch die Parameter auszudrücken (s. Abschnitt 3.2.3). Für die logischen Szenarien muss die Szenariobeschreibung eine Möglichkeit bieten, die Parameter und deren Bereiche zu beschreiben.

Diese werden beim Ableiten und Ausführen der Testfälle genutzt, um eine effiziente Testabdeckung zu erreichen.

**Anforderung 7.** *Die Szenarien müssen möglichst **kompakt** beschrieben werden.*

Der Umfang der Beschreibung muss so gering wie möglich und so genau wie nötig sein. Zudem sollen Redundanzen vermieden werden.

Diese Anforderung ist nicht zwingend für einen Prozessschritt notwendig, fördert aber die Effizienz des gesamten Testprozesses.

**Anforderung 8.** *Die beschriebenen Szenarien müssen **durchgängig** im gesamten Testprozess verwendet werden können.*

Im Testprozess werden verschiedene Tools und Methoden angewendet, welche die erläuterten Anforderungen an die Szenariobeschreibung stellen. Nicht jede Anforderung wird dabei in jedem Prozessschritt erfordert (s. Abbildung 4.1). Die Verwendung von mehreren Beschreibungsformaten, die jeweils nur für einzelne Prozessschritte relevante Anforderungen erfüllen, soll vermieden werden. Eine durchgängige Szenariobeschreibung erhöht die Interoperabilität zwischen den verschiedenen Schritten und Tools. Zudem fördert sie die Zusammenarbeit zwischen den Teams und kann den Grad der Automatisierung erhöhen.

Die Durchgängigkeit ist somit zwar nicht für einen einzelnen Prozessschritt notwendig, ermöglicht aber eine effiziente Testabdeckung. Die erstellten Anforderungen hängen mit den Prozessschritten des SBT-Prozess (s. Abschnitt 3.3) und den hergeleiteten Herausforderungen des szenariobasierten Testens (s. Abschnitt 3.2) zusammen (s. Abbildung 4.1). Durch die Vielfalt an Einsatzzwecken von Szenarien über den gesamten SBT-Prozess hinweg entsteht auch die Vielfalt an hergeleiteten Anforderungen. Teilweise widersprechen sich diese untereinander, z.B. wenn Szenarien in der manuellen Erstellung intuitiv und am besten in natürlicher Sprache beschrieben werden sollen, für eine automatisierte Ausführung aber ein Formalismus notwendig ist.

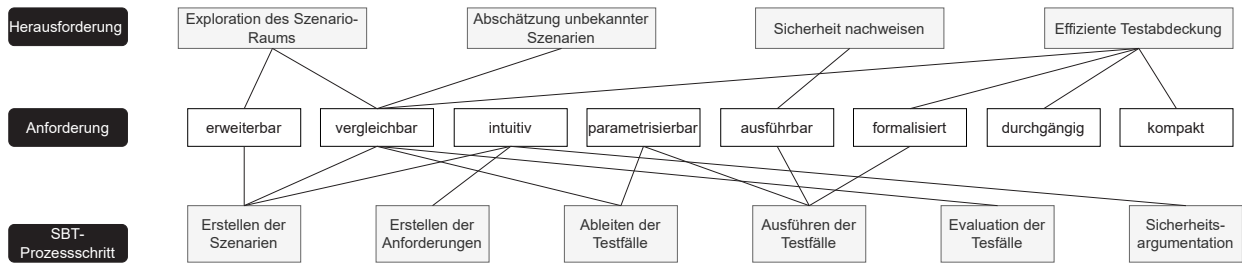


Abbildung 4.1: Verknüpfung zwischen den Anforderungen, Herausforderungen (s. Abschnitt 3.2) und Prozessschritten (s. Abschnitt 3.3)

## 4.2 Abgleich des Stands der Wissenschaft und Technik

Um solche Widersprüche aufzulösen, werden bisherige Szenariobeschreibungsformen im Stand der Wissenschaft und Technik auf einzelne Prozessschritte und deren Anforderungen fokussiert und optimiert, ohne auf die Durchgängigkeit und Effizienz im gesamten Prozess zu achten. Dadurch entstehen Insellösungen, bei denen ein Szenario im Laufe des SBT-Prozesses (s. Abschnitt 3.3) in mehreren Beschreibungsformen vorliegt.

### 4.2.1 Abgleich der Eigenschaften von Szenarien mit den Anforderungen

Bestehende Ansätze werden in Tabelle 4.1 mit den eingeführten Anforderungen in Bezug gesetzt. Zur kompakteren Darstellung und Erläuterung werden die Ansätze über die vorgestellten Eigenschaften von Szenarien (s. Abschnitt 3.4.4 und Abschnitt 3.4.5) gruppiert. Eine erfüllte Anforderung wird durch einen Haken “✓” markiert, eine nicht erfüllte Anforderung durch ein Kreuz “×”. Hängt das Erfüllen der Anforderungen von der konkreten Ausgestaltung ab, ist dies durch ein “-” gekennzeichnet.

1) Für ausführbare Szenarien wird stellvertretend der Industriestandard OpenSCENARIO XML [54] betrachtet. Dieser ist ausführbar, über ein XML-Schema formalisiert und ab Version 1.2 auch parametrisierbar. Bisher existieren aber keine Arbeiten, OpenSCENARIO-Dateien miteinander zu vergleichen. Dies liegt auch daran, dass die in OpenSCENARIO verwendeten Manöver Anweisungen an die Teilnehmer darstellen und keine beobachteten Verhaltensweisen sind. Ein durchgängiger Einsatz im SBT-Prozess ist daher nicht möglich. Die gleichen Einschränkungen gelten für andere proprietäre ausführbare Beschreibungsformen gängiger Simulationstools (s. Abschnitt 3.4.2).

2) Dagegen ist die meistgenutzte Verwendung von aufgezeichneten Szenarien das Vergleichen und Gruppieren dieser, um die Menge der bekannten Szenarien zu ermitteln und zu strukturieren. So nutzen z.B. Siebinga et al. [128] und Hauer et al. [117] aufgezeichnete Trajektorien als Beschreibungsform. Hartjen et al. [71] verwenden eine eigens dafür entwickelte Manöver-Beschreibungsform.

	Vergleichbar	Ausführbar	Erweiterbar	Intuitiv	Formalisiert	Parametrisierbar	Kompakt	Durchgängig	Literatur
1) Ausführbar	×	✓	-	-	✓	✓	-	×	[54]
2) Aufgezeichnet	✓	×	-	-	-	-	-	×	[128] [117] [71]
3) Verbal	×	×	✓	✓	×	-	×	×	[106]
4) Visuell	×	×	-	✓	×	-	✓	×	[153] [75]
5) Modellbasiert	-	-	✓	-	✓	✓	-	✓	[29][54] [71] [36]

Tabelle 4.1: Vergleich der Anforderungen mit bestehenden Ansätzen aus der Literatur, gruppiert nach den Eigenschaften aus Abschnitt 3.4.4 und Abschnitt 3.4.5

Während die Vergleichsberechnungen ausführlich beschrieben werden (s. Abschnitt 3.4.3), wird in keiner der Arbeiten weiter darauf eingegangen, wie die Szenarien im fortlaufenden Prozess verwendet werden, um sie z.B. als Testszenario auszuführen. So bieten Trajektorien keine notwendige Abstraktion für Closed-Loop-Simulationen und KPIs nicht ausreichende Informationen über den Ablauf der Szenarien. Ein durchgängiger Einsatz der vorgestellten Beschreibungen ist daher nicht möglich.

3) Verbal beschriebene Szenarien (s. Abschnitt 3.4.5), zu welchen die von Menzel et al. [106] vorgestellten funktionalen Szenarien gehören, sind aufgrund der verwendeten natürlichen Sprache intuitiv von Menschen verwendbar. Da der gesamte Wortschatz der natürlichen Sprache verwendet wird, ist die Beschreibung auch erweiterbar. Neue Elemente werden durch neue Wörter hinzugefügt. Natürliche Sprache ist nicht technisch formalisiert, wodurch diese Anforderung von verbalen Beschreibungsformen nicht erfüllt wird. Diese Eigenschaft verhindert auch eine Ausführbarkeit auf den Testplattformen, wie z.B. SiL-Systemen. Ebenso ist ein automatisierter Vergleich von Szenarien ohne Formalisierung nicht möglich. Diese Einschränkungen hemmen auch einen durchgängigen Einsatz verbaler Beschreibung. Natürliche Sprache ist nicht kompakt, da sie aufgrund von Redundanzen, Mehrdeutigkeiten und Kontextabhängigkeiten oft mehr Worte benötigt, um Ideen und Konzepte auszudrücken.

4) Visuelle Beschreibungen vermitteln im Gegensatz dazu kompakt Informationen, indem sie auf einen Blick erfassbar sind und weniger sprachliche Nuancen erfordern. Diese werden z.B. in der ISO 15622 [153] und der ISO/PAS 21448 [75] für die Spezifikation von Folgefahrtszenarien verwendet, wobei sie durch sprachliche Beschreibungen ergänzt werden. Sie sind zusätzlich intuitiv nutzbar, weil Menschen in der Lage sind, visuelle Informationen gut zu verarbeiten. Wie bei der verbalen Beschreibung fehlt bei rein visueller Beschreibung ein Formalismus, wodurch die Anforderungen der automatisierten Ausführbarkeit, Vergleichbarkeit und damit auch Durchgängigkeit nicht erfüllt werden.

5) Modellbasierte Beschreibung baut dagegen definitionsgemäß auf so einem Formalismus auf, wodurch diese Anforderung erfüllt ist. Wird der Modellierungsprozess iterativ angewendet, ist das so erstellte Modell auch um neue Elemente erweiterbar. Die Vergleichbarkeit und Ausführbarkeit hängen dabei von der konkreten Ausgestaltung des Modells ab. Für die Vergleichbarkeit muss zusätzlich ein Ähnlichkeitsmaß modelliert werden. Für die Ausführbarkeit müssen alle dafür relevanten Informationen im Modell enthalten sein. Dies ist z.B. für OpenSCENARIO [54] gegeben, allerdings existiert kein Ähnlichkeitsmaß. Bei Hartjen et al. [71] dagegen steht die Vergleichbarkeit und nicht die Ausführbarkeit im Fokus. Modellbasierte Beschreibungen sind als einzige der vorgestellten Methoden durchgängig im SBT-Prozess nutzbar, wenn alle Prozessschritte bei der Modellierung mit betrachtet werden. So nutzen Bach et al. [165] und Pfeffer [36] die gleiche Beschreibungsart für das Erstellen aus Daten und Ausführen von Szenarien.

### **4.2.2 Durchgängigkeit bestehender modellbasierter Beschreibungsformen**

Dennoch existiert bisher keine Beschreibung, die bereits in allen Prozessschritten des SBT-Prozesses durchgängig verwendet wird (s. Tabelle 4.2).

OpenSCENARIO XML ist als ausführbares Format entwickelt und erfüllt daher nicht die Anforderungen der Vergleichbarkeit und Durchgängigkeit (s. Tabelle 4.1). Durch die XML-Syntax ist es zudem nicht intuitiv für Menschen lesbar. Überdies ist das Format nicht kompakt. So hat das einfache Beispielszenario für einen Doppelspurwechsel bereits 178 Zeilen.



Methode	Erstellen (manuell)	Erstellen (Daten)	Ableiten (Vergleichen)	Ausführen	Evaluation
OSC XML [54]	✓			✓	
Pfeffer [36]	(✓)	✓		✓	✓
Montanari [67]		✓		✓	
Hartjen [166]		✓	✓		
King [28]					✓

Tabelle 4.2: Einsatz von modellbasierten Szenariobeschreibungen bezogen auf den SBT-Prozess (s. Abschnitt 3.3)

Pfeffer [36] stellt ein Metamodell zur manöverbasierten Szenariobeschreibung vor, das für eine datenbasierte Resimulation von Szenarien auf der Autobahn eingesetzt wird. Dafür werden Manöver in den Daten identifiziert, als Szenarien abgespeichert und anschließend in ein proprietäres Simulationsformat konvertiert. Eine Evaluation der ausgeführten Szenarien über die beschriebenen Manöver ist ebenfalls vorgestellt. Manuelles Erstellen der Szenarien wird konzeptionell als Möglichkeit erwähnt, allerdings nicht tiefergehend vorgestellt. Fehlend für einen durchgängigen Einsatz im Sinne des SBT-Prozesses ist ein Ähnlichkeitsmaß, mit dem die Szenarien verglichen, geclustert und ausgewählt werden. Eine weitere Einschränkung ist, dass die Arbeit nur für die Autobahn entwickelt und evaluiert wurde. So wurden auch keine Karteninformationen verwendet, welche für den vielschichtigen urbanen Bereich notwendig sind.

Montanari [67] erzeugt ebenfalls eine Resimulation von Szenarien aus aufgezeichneten Daten, die direkt in einem Fahrzeug aufgenommen werden. Über eine Abstraktion zu Manövern wird anschließend eine OpenSCENARIO-Datei erstellt, die als Eingang für ein Simulationstool dient. Der Ablauf dabei ist zwar vollständig automatisiert, allerdings nur auf einzelne Szenarien bezogen. Ein Vergleich von Szenarien, um z.B. die Gesamtheit der Szenarien zu erfassen oder eine Auswahl repräsentativer Szenarien zu ermitteln, findet nicht statt.

Der Fokus bei der semantischen Szenariobeschreibung von Hartjen [166] liegt auf dem datenbasierten Erstellen und Vergleichen der Szenarien. Es wird vor allem die Vollständigkeit einer Szenariodatenbank untersucht, indem Sättigungseffekte betrachtet werden. Dazu werden ein Manöverkatalog und eine

passende Manöver-Identifikation verwendet. Der Vergleich der Szenarien erfolgt über sogenannte “Parallel Maneuver Combinations”, welche den gleichzeitigen Zustand von Manövern unterschiedlicher Kategorien beschreiben. Ein Beispiel ist ein gleichzeitiges Spurhalten, Abbremsen und Objekt folgen. Neben der Analyse der aufgezeichneten Daten wird aber weder auf die Ausführung der erstellten Szenarien noch auf eine Auswahl repräsentativer Testfälle eingegangen.

Die Notwendigkeit, neben dem Testobjekt auch das ausgeführte Szenario zu validieren, wird von King [28] festgehalten. Hierfür wird für jedes Szenario ein Satz an Merkmalen definiert, welche es beinhalten muss. Für das Verhalten der Teilnehmer sind dies u. a. deren ausgeführten Manöver. Die so definierten Manöver werden aber nur für diese Validierung und nicht in vorherigen Prozessschritten genutzt. Auch muss für jedes ausführbare Szenario die Spezifikation der zu erfüllenden Merkmale manuell erstellt werden und kann nicht aus vorherigen Schritten automatisiert genutzt werden.

Die erläuterte Fragmentierung der Szenariobeschreibungsformen auf einzelne Anwendungsschritte lässt sich zusätzlich in der wissenschaftlichen Literatur beobachten. Neurohr et al. [82] ordnen in ihrem Übersichtspaper zu szenariobasierten Testen jedem Prozessschritt die relevante Literatur zu. Mehr als 90% der Veröffentlichungen sind dabei nur einem Prozessschritt zugeordnet, keine einzige behandelt alle Schritte. Dies unterstreicht noch einmal, dass bisher viele Insellösungen erarbeitet wurden, aber eine Durchgängigkeit der Ergebnisse aus den einzelnen Prozessschritten nicht im Fokus der Forschung liegt. Gerade dies würde aber eine Grundlage schaffen, um ein effizientes szenariobasiertes Testen zu ermöglichen.

## **5 Konzept des manövergestützten Testens**

### **5.1 Beitrag dieser Dissertation**

#### **5.1.1 Einordnung**

Um diese Forschungslücke zu schließen, wird eine Methode für die in Abschnitt 3.3 gezeigten Prozessschritte entwickelt, die eine manövergestützte Beschreibungsform verwendet, welche die in Kapitel 4 erläuterten Anforderungen erfüllt. Besonders betont wird die Anforderung der Durchgängigkeit, weil diese bisher von keiner Beschreibungsform erfüllt wird (s. Abschnitt 4.2). Das Konzept bezieht sich nur auf die Szenarien der Testfälle (s. Definition 3.8). Die Anforderungen an das Testobjekt sind nicht Teil des Konzepts. Zudem werden nur die dynamischen Elemente des Szenarios (Layer 4 aus Abschnitt 3.2.2), sowie deren Verknüpfung zu statischen Objekten und zur Straßeninfrastruktur (Layer 1–3 aus Abschnitt 3.2.2) betrachtet.

#### **5.1.2 Grundgedanke des manövergestützten Testens**

Die Szenariobeschreibungsform, welche das verbindende Element darstellt, wird auf Basis von Manövern aufgebaut. Zur Veranschaulichung dieser Verhaltensbeschreibung zeigt Abbildung 5.1 beispielhaft den schematischen Ablauf eines Kreuzungsszenarios mit einem kreuzenden Fußgänger und die zugehörigen Manöver des Pkw in einer Matrixform<sup>1</sup>.

---

<sup>1</sup> Erläuterungen zur Darstellung sowie ein weiteres Beispiel findet sich in Abschnitt A.2.

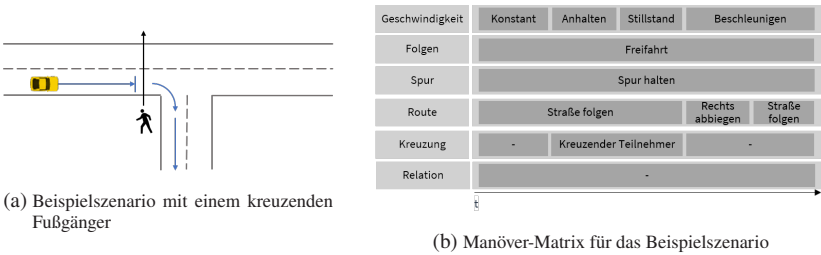


Abbildung 5.1: Beispielszenario zur Veranschaulichung der Verhaltensbeschreibung über Manöver

Manöver als Beschreibung werden bereits erfolgreich in verschiedenen Anwendungsfällen eingesetzt (s. Abschnitt 3.1.5). Sie werden bei unterschiedlichen Typen von Szenarien (s. Abschnitt 3.4.4) genutzt, um das Verhalten der Teilnehmer zu beschreiben: Manöver lassen sich auch in den Abstraktionsebenen funktional, logisch und konkret (s. Abschnitt 3.2.3) verwenden. Für logische und konkrete Szenarien muss jedem Manöver eine Menge an Parametern zugeordnet werden, z.B. die Ziel-Geschwindigkeit bei einem Beschleunigungsmanöver oder der Abstand zur Spurmitte bei einem Spurhalte­manöver. Logische Szenarien zeichnen sich dann durch Parameterverteilungen der Manöver aus, konkrete Szenarien durch konkrete Werte der Parameter [78]. Für funktionale Szenarien ist ein eindeutiger Satz an Parametern nicht notwendig. Der Einsatz von Manövern erstreckt sich zudem über die Prozessschritte im SBT-Prozess (s. Abschnitt 3.1.5).

Bisher wurden die entwickelten Manöverkataloge, die Eigenschaften der Manöver und die zugehörige Szenariobeschreibungsform allerdings nur isoliert für den jeweiligen Anwendungsfall betrachtet und optimiert (s. Abschnitt 3.1 und Abschnitt 4.2.2) und nicht für den durchgängigen Prozess verwendet. Mit dem folgenden Konzept zum manövergestützten Testen wird gezeigt, wie Manöver auch für einen durchgängigen Einsatz verwendet werden können.

## 5.2 Das Vorgehensmodell im manövergestützten Testen

### 5.2.1 Übersicht des Vorgehensmodells

Grundsatz im manövergestützten Testen ist, dass die einzelnen Prozessschritte des SBT-Prozesses (s. Abschnitt 3.3) so umgesetzt werden, dass sie alle mit denselben manöverbasierten Beschreibungsformen arbeiten (s. Abbildung 5.2).

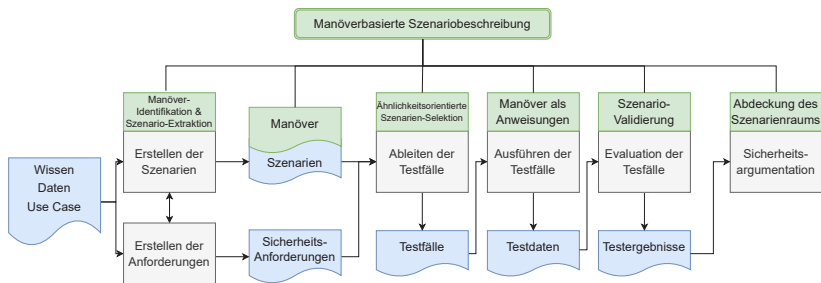


Abbildung 5.2: Übersicht zum Vorgehensmodell des manövergestützten Testens auf Basis des Prozesses zum szenariobasierten Testen (s. Abbildung 3.7)

Die erstellten Manöver werden, mit den zugehörigen Szenarien verknüpft, in einer Datenbank gespeichert. Die zugehörige Struktur wird in Abschnitt 5.2.7 vorgestellt.

Bei der datenbasierten Szenario-Erstellung werden die Manöver durch eine Identifikation in den aufgezeichneten Daten erkannt und gespeichert (s. Abschnitt 3.3.1). Werden die Szenarien expertenbasiert von Menschen erstellt, werden Manöver manuell generiert, um die Szenarien zu beschreiben.

Für das Ableiten der Testfälle wird ein auf Manövern basierendes Ähnlichkeitsmaß entwickelt. Darauf aufbauend wird ein Clustering und eine Visualisierung der Ähnlichkeit verwendet, um repräsentative Szenarien für die Testfälle auszuwählen (s. Abschnitt 5.2.3). Werden mehrere Testplattformen verwendet, wird dieser Schritt für jede Plattform einzeln durchgeführt.

Für die Ausführung auf den Testplattformen werden die Manöver als Anweisungen für die Teilnehmer interpretiert (s. Abschnitt 5.2.4).

Folgend werden die ausgeführten Szenarien validiert, indem die durchgeführten Manöver mit den zuvor spezifizierten verglichen werden (s. Abschnitt 5.2.5).

Abschließend bilden die Szenarien und erstellten Cluster einen Teil der Sicherheitsargumentation (s. Abschnitt 5.2.6).

Das Vorgehen ist hier linear aufgezeichnet, kann im Entwicklungsprozess aber auch iterativ durchlaufen werden. Je nach Grund der Iteration variiert dabei der Einstiegspunkt: Werden bisher unbekannte Szenarien gefunden, werden diese im Prozessschritt Erstellen der Szenarien erzeugt und dem Szenarienkatalog hinzugefügt. Beim Ableiten der Testfälle kann z.B. gestartet werden, wenn sich Testkapazitäten ändern oder eine neue Testkampagne geplant wird, um repräsentative Testszenarien auszuwählen. Und das Ausführen der Testfälle auf den bereits ausgewählten Szenarien ist Einstiegspunkt, wenn das Testobjekt weiterentwickelt wird, z.B. um neue Features hinzuzufügen und diese zu untersuchen.

Die Verknüpfung der einzelnen Prozessschritte über die Manöver ermöglicht damit erstmals einen durchgängigen Prozess, ohne die Art der Szenariobeschreibung zu verändern. Ein Kernelement dafür ist die Entwicklung eines neuen, auf Manövern basierenden Ähnlichkeitsmaßes (s. Abschnitt 5.2.3), das für das Clustering und die Visualisierung der Vielfalt der Szenarien sowie für deren Validierung verwendet wird.

5.2.2 Manöver-Identifikation und Szenario-Extraktion

Ausgangspunkt für das Vorgehen sind aufgezeichnete Sequenzen von Trajektorien in Verkehrsszenarien, die z. B. von Drohnen oder der Verkehrsinfrastruktur erfasst werden. Die Daten beinhalten noch keine Informationen über Manöver und werden daher für die weitere Verarbeitung mit diesen angereichert. Die Manöver-Identifikation ist sowohl für ein Ego-Fahrzeug als auch für andere Teilnehmer anwendbar.

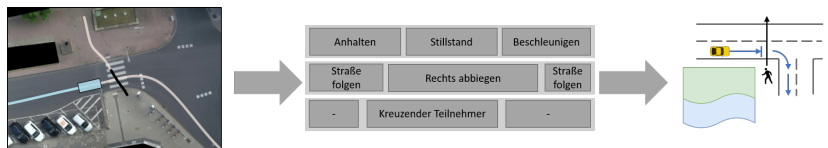


Abbildung 5.3: Vorgehen zum Erkennen von Manövern in aufgezeichneten Realdaten und Speichern dieser in der Manöverdatenbank.



Abbildung 5.4: Aktivitätsdiagramm für die Manöver-Identifikation und Szenario-Extraktion

Für das Vorgehensmodell wird folgende Notation verwendet:

- $S^a$ : Szenario mit der ID  $A$
- $\vec{z}^a$ : Aufgezeichnete Zeitreihen der Teilnehmer im Szenario  $S^a$  (Z.B. Geschwindigkeit und Position)
- $k$ : Manöverkategorie (Z.B. Spurmanöver oder Abbiegemanöver, s. Abschnitt 5.2.7)
- ${}^i m_k^a$ : Manöver der Kategorie  $k$  am Index  $i$  des Szenarios  $S^a$
- $M_k^a = \{{}^k m_1^a, {}^k m_2^a, \dots, {}^k m_n^a\}$ : Sequenz der Manöver der Kategorie  $k$  im Szenario  $S^a$
- $s^a$ : Statische Elemente des Szenarios  $S^a$

## Manöver-Identifikation

Die Manöver-Identifikation befasst sich mit dem Erkennen der Manöver in aufgezeichneten Daten (s. Abschnitt 3.1.4). Als Input für die Erkennung werden

die Zeitreihen der Teilnehmer  $\vec{z}^a$ , ihre Verknüpfung zur Karte und weiteren statischen Elementen  $s^a$  verwendet. Output sind die erkannten Manöver  $M_k^a$  sowie deren zugehörige Parameter, wie z.B. Dauer und Durchschnittsgeschwindigkeit. Die Manöver-Identifikation lässt sich also als folgende Funktion beschreiben:

$$M_k^a = f_{mi}(\vec{z}^a, s^a) \quad (5.1)$$

Mögliche Ansätze zur Manöver-Identifikation sind regelbasiert sowie unüberwachtes und überwachtes maschinelles Lernen (s. Abschnitt 3.1.4). Aufgrund der Modellierung des Manöverkatalogs durch Expertenwissen ist die Anwendung eines unüberwachten maschinellen Lernalgorithmus ausgeschlossen. Es eignen sich sowohl die überwachten maschinellen Lernmethoden als auch die regelbasierte Erkennung. Regelbasierte Methoden sind für die Identifikation von Manövern etabliert (s. Abschnitt 3.1.4), benötigen keine gelabelten Daten und erzeugen nachvollziehbare Ergebnisse. Die Ergebnisse von überwachten maschinellen Lernmethoden hängen von der Qualität und Quantität der Trainingsdaten ab. Öffentliche Trajektorien-Datensätze mit entsprechenden Labels der für dieses Konzept spezifizierten Manöverkategorien und -typen (s. Abschnitt 5.2.7) sind aber nicht vorhanden: Entweder sind keine Manöver gelabelt [154, 167, 155, 157], oder es werden andere Manöverkategorien - und Typen verwendet, wie beim highD-Datensatz [156] und dem exiD-Datensatz [158], deren Fokus auf der Autobahn und den Auf- bzw. Abfahrten dazu liegt. Um keinen eigenen Datensatz erstellen zu müssen, wird eine regelbasierte Methode angewendet. Die Erkennung von Manövern durch überwachte maschinelle Lernmethoden wird im Ausblick (Abschnitt 8.2) betrachtet.

Für jeden Manövertyp wird in diesem Rahmen ein eigener Klassifikator erstellt, der die Zeitreihen der aufgezeichneten Daten als Input verwendet (s. Abbildung 5.4). Je nach Struktur der Eingangsdaten sind zusätzliche Vorverarbeitungsschritte notwendig, um die Klassifikation durchzuführen. Beispiele für solche Regeln sind:

- *Anhalten*: Wenn die Beschleunigung negativ ist und die Geschwindigkeit dadurch "0" erreicht, befindet sich der Teilnehmer im Anhalten-Manöver (s. Abbildung 5.5).



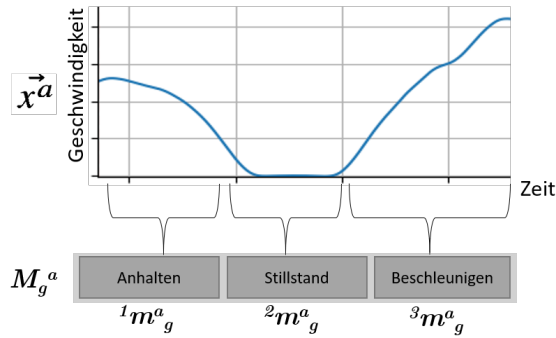


Abbildung 5.5: Beispiel einer Manöver-Identifikation für die Kategorie “Geschwindigkeit” anhand des Geschwindigkeitsverlaufs für das Szenario in Abbildung 5.3

- *Stillstand*: Wenn die Geschwindigkeit “0” ist, befindet sich der Teilnehmer im Manöver Stillstand (s. Abbildung 5.5).
- *Spurwechsel*: Überquert der Teilnehmer eine Spur, findet ein Spurwechselmanöver statt.
- *Folgefahrt*: Befindet sich ein anderer Teilnehmer vorausfahrend mit gleicher Geschwindigkeit auf derselben Spur, ist eine Folgefahrt aktiv.

## Szenario-Extraktion

Die so identifizierten Manöver bilden den essenziellen Input für die entwickelte Beschreibungsform (s. Abschnitt 5.2.7). Um die Szenarien zu beschreiben, werden die Manöver der im Szenario agierenden Teilnehmer zu Sequenzen zusammengesetzt und eine Instanz des Szenario-Metamodells (s. Abbildung 5.19) erzeugt. Somit erfolgt die Verknüpfung zur Szenariendatenbank, in der das Szenario und dessen Verbindungen zu den Manövern abgespeichert werden (s. Abbildung 5.4):

$$S^a = f_{se}(M_k^a, s^a) \quad (5.2)$$

### 5.2.3 Ähnlichkeitsorientierte Szenarien-Selektion

Beim Ableiten der Testszenarien werden die zu testenden Szenarien aus der Szenariendatenbank bestimmt. Abbildung 5.6 zeigt das Zielbild für diesen Prozessschritt, in dem repräsentative Szenarien (schwarz umrandete Dreiecke) ausgewählt werden, um die Testabdeckung der ODD zu erhöhen (vgl. Abschnitt 3.3.3). Für die Auswahl wird die ähnlichkeitsorientierte Szenarien-Selektion entworfen (s. Abbildung 5.7), mit dem Ziel, effizient eine umfassende Abdeckung zu erreichen.

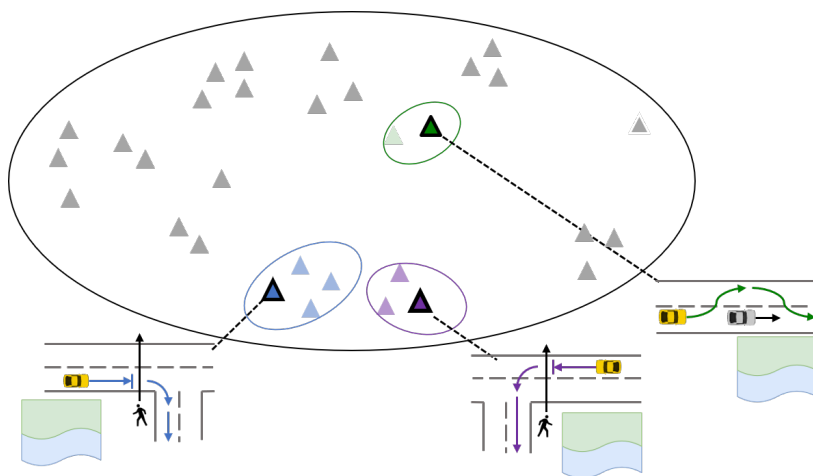


Abbildung 5.6: Zielbild für das Ableiten der Testszenarien innerhalb einer ODD am Beispiel von drei repräsentativen Szenarien

Die Basis für diesen Prozessschritt ist die Vergleichbarkeit (s. Anforderung 1), um ähnliche Szenarien zu identifizieren. Diese werden aus Sicht des Testens als redundant betrachtet, da der Nachweis der Sicherheit in dem repräsentativen Szenario (s. Definition 3.12) auf die zu ihm ähnlichen erweitert werden kann.

Für die Auswahl der Testszenarien durch Manöver wird ein neues Distanzmaß entwickelt (s. Abschnitt 5.2.3), welches die Ähnlichkeit zwischen zwei Szenarien quantifiziert. Dieses baut auf den zuvor definierten Begriffen zu Distanz und Ähnlichkeit auf (s. Abschnitt 3.4.3), wobei die Manöver als Merkmale  $F$  eingesetzt werden.

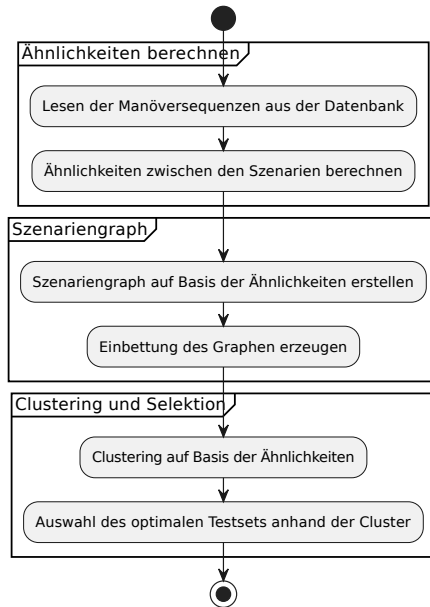


Abbildung 5.7: Aktivitätsdiagramm für die ähnlichkeitsorientierte Selektion von Testscenarien

**Definition 5.1.** Die (Manöver-)Distanz  $d(S^a, S^b)$  zwischen den Szenarien  $S^a$  und  $S^b$  quantifiziert deren Unähnlichkeit aufgrund der Manöver ihrer Teilnehmer. Identische Manöverfolgen implizieren damit ein Distanzmaß von null.

Auf dieser Grundlage wird eine grafische Repräsentation des Szenarienraums erstellt, in dem ähnliche Szenarien räumlich nahe beieinander liegen, während unähnliche Szenarien eine höhere Distanz zueinander aufweisen. Innerhalb dieser Visualisierung wird, entweder manuell oder mit Optimierungsalgorithmen, ein Set an Testscenarien ausgewählt, um eine effiziente und zielgerichtete Testabdeckung der ODD sicherzustellen. Unterstützt wird die Auswahl durch einen Clusteralgorithmus (s. Abschnitt 5.2.3), der ähnliche Szenarien in Gruppen einteilt.

## Ähnlichkeiten von Szenarien durch Manöversequenzen

Zur Berechnung der Ähnlichkeit bzw. Distanz zwischen zwei Szenarien (s. Abbildung 5.7) wurden in Abschnitt 3.4.3 vielfältige Methoden vorgestellt. In diesem Konzept wird ein Distanzmaß über die modellierten Manöver berechnet. Der Ansatz gliedert sich somit in die Ähnlichkeitsbestimmung durch Abstraktion ein. Prinzipiell besteht die Möglichkeit, das erstellte Distanzmaß mit weiteren, beispielsweise auf Zeitreihen oder Kennzahlen basierenden Maßen, zu kombinieren. Dies wird im Ausblick betrachtet (vgl. Abschnitt 8.2).

Es ist möglich, die Manöver aller Teilnehmer oder nur die des Ego-Fahrzeugs für die Berechnung zu verwenden. Werden die Manöver aller Teilnehmer betrachtet, tritt folgende Schwierigkeit auf: Das unwesentliche Verhalten von Teilnehmern, das nicht relevant für das Ego-Fahrzeug ist, beeinflusst fälschlicherweise die Ähnlichkeit zwischen den Szenarien. So können z.B. Beschleunigungs- oder Abbiegemanöver, die keinen Einfluss auf das Ego-Fahrzeug haben, die Ähnlichkeit fälschlicherweise verringern.

Um nur die relevanten Interaktionen des Ego-Fahrzeugs zu betrachten, werden diese durch dessen Manöver mit Bezug zu anderen Objekten (z.B. Folgen) und Kontexte (z.B. kreuzender Teilnehmer) modelliert (s. Definition 5.8) und fließen somit in die Ähnlichkeitsberechnung ein. Das Betrachten der Manöver anderer Teilnehmer ist aber eine mögliche Ergänzung, die im Ausblick erläutert wird (s. Abschnitt 8.2).

Für die Berechnung werden die Sequenzen der einzelnen Manöverkategorien miteinander verglichen. Dabei werden sowohl der Typ als auch die Reihenfolge betrachtet. Zudem wird nicht nur eine binäre Unterteilung in gleiche und ungleiche Szenarien durchgeführt, sondern ein kontinuierliches Ähnlichkeitsmaß berechnet.

Um diese Eigenschaften zu erreichen, werden die Sequenzen der Manöver verwendet. Jedes Manöver wird, unabhängig von der Dauer, als ein Element dieser Sequenz betrachtet. Diese lassen sich als multivariate, kategorische Zeitreihen beschreiben. Multivariat bedeutet, dass mehrere Zeitreihen betrachtet werden, in diesem Fall mehrere Kategorien an Manövern. Kategorisch bezeichnet die Eigenschaft dieser Zeitreihen, keine numerischen Werte, sondern ausschließlich diskrete Werte in den zuvor definierten Kategorien anzunehmen.

Um den multivariaten Aspekt in der Berechnung der Distanz zu erfassen, werden die Sequenzen der Manöver jeder Kategorie einzeln miteinander verglichen. Die Distanz der Szenarien ist dann das gewichtete Mittel der so ermittelten Teildistanzen. Dabei wird folgende Notation verwendet (s. Abbildung 5.8):

- $d_k(M_k^a, M_k^b)$ : Distanz der Kategorie  $k$  zwischen den Szenarien  $S^a$  und  $S^b$
- $c_k$ : Gewichtungsfaktor für verschiedene Kategorien

Darauf aufbauend wird die Distanz zwischen den Szenarien wie folgt berechnet:

$$d(S^a, S^b) = \sum_k c_k \cdot d_k(M_k^a, M_k^b) \quad (5.3)$$

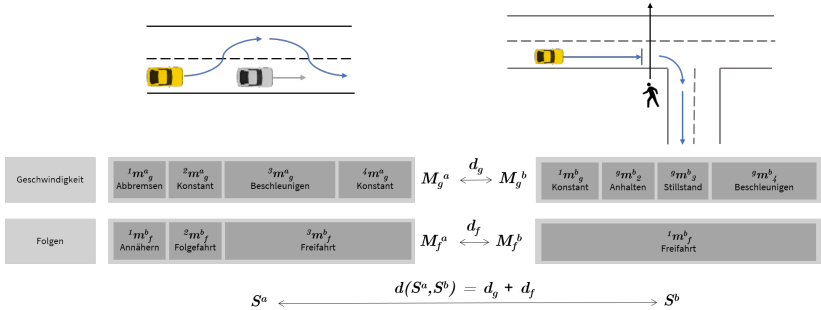


Abbildung 5.8: Veranschaulichung der Notation zur Distanzberechnung für die Szenarien  $S^a$  und  $S^b$ , beispielhaft anhand der Distanz der Manöversequenzen für die Kategorien “Geschwindigkeit” (g) und “Folgen” (f). Für die vollständige Berechnung werden alle Manöverkategorien verwendet.

Der verbleibende Vergleich der einzelnen, kategorischen Zeitreihen  $M_k^a$  wird durch Sequenzalignment gelöst (s. Abschnitt A.1.1). Die Distanz der Manöversequenzen wird dabei durch die Operationen “Match” für Übereinstimmungen, “Mismatch” für Unterschiede und “Gap” für eingefügte Lücken bestimmt. Durch das Einfügen der Gaps können auch unterschiedlich lange Sequenzen verglichen werden. Jede der Operationen hat zugehörigen Kosten, z. B. “0” für Match (gleiche Manövertypen), “1” für Gap (ein Manöver in einer Sequenz

einfügen) und “2” für Mismatch (unterschiedliche Manövertypen). Um die Distanz der beiden Sequenzen zu bestimmen, werden die Operationen zwischen den beiden Sequenzen auf die geringsten Kosten optimiert. Durch das Verwenden der Gaps ist es auch möglich, Sequenzen unterschiedlicher Länge miteinander zu vergleichen. Das Ergebnis des Sequenzalignment für Manöver kann in Form einer Matrix dargestellt werden, in der Unterschiede und Lücken zwischen den Sequenzen markiert sind (s. Abbildung 5.9). Mit den zuvor erläuterten Kosten haben diese Beispielsequenzen für die Manöver der Kategorie “Geschwindigkeit” die Distanz  $d_g = 3$ , zusammengestellt aus den Kosten für zwei Matches (0), einem Gap (1) und einem Mismatch (2).

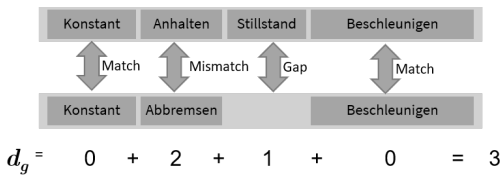


Abbildung 5.9: Veranschaulichung der Methodik Sequenzalignment für die Berechnung der Distanz von zwei Manöversequenzen

Die Auswahl der verwendeten Manöver- und Kontextkategorien stellt einen Trade-Off zwischen Abstraktionsgrad und Genauigkeit dar. Werden zu wenige Kategorien verglichen, besteht die Gefahr, dass relevante Aspekte des Szenarios nicht berücksichtigt werden. Werden z.B. keine Kontexte zu anderen Teilnehmern betrachtet, haben diese keinen Einfluss auf die Ähnlichkeitsberechnung. Andererseits erzeugen mehr Kategorien auch einen größeren Parameterraum, der eventuell nicht mehr von den erstellten Szenarien ausreichend abgedeckt wird. Zudem ergibt sich ein Optimierungsproblem, bei dem die Gewichtungsfaktoren  $c_k$  aufeinander abgestimmt werden, die Rechenzeit erhöht sich und die Kompaktheit der Beschreibungsform sinkt. Für dieses Konzept werden alle in Abschnitt 3.1.3 vorgestellten Kategorien als Eingangsgröße für den Vergleich genutzt. Diese werden zweckmäßig so gewählt, dass die Anzahl möglichst gering ist und dennoch die relevanten Aspekte der Szenarien beschrieben und verglichen werden.

Weiterhin werden die absoluten Werte der Manöverparameter, wie z.B. die Geschwindigkeit, im Rahmen der Vergleichsmethodik nicht betrachtet. Durch das Identifizieren der Manöver wird das Szenario semantisch abstrahiert. Die

Ähnlichkeitsberechnung findet rein auf dieser Abstraktionsebene statt. Eine feinere Distanzberechnung durch das Hinzufügen der Parameter ist möglich und wird im Ausblick betrachtet (s. Abschnitt 8.2).

### Visualisieren der Ähnlichkeiten als Szenariengraph

Die so errechneten Distanzen erzeugen Informationen über die Ähnlichkeiten zwischen jeweils zwei Szenarien. Um die gesamte Menge an Szenarien zu analysieren und repräsentative Testfälle auszuwählen, müssen aber die Ähnlichkeiten aller Szenarien aggregiert betrachtet werden. Dafür wird eine Methode zur Visualisierung entwickelt, die eine intuitive Einschätzung und Analyse des kompletten Szenarienraums ermöglicht (s. Abbildung 5.7).

Ausgangspunkt ist das Berechnen einer Distanzmatrix  $D$ , in der von jedem in der Datenbank enthaltenen Szenario zu jedem anderen Szenario die Distanz gespeichert wird. Bei  $n$  betrachteten Szenarien enthält die so erzeugte Matrix also  $n^2$  Einträge. Da das verwendete Distanzmaß symmetrisch ist, ist auch die zugehörige Distanzmatrix symmetrisch. Durch dieses Vorgehen steigt auch die Rechenzeit quadratisch mit der Anzahl der betrachteten Szenarien, in Abschnitt 7.1.6 wird aber gezeigt, dass die gesamte Rechendauer trotzdem unproblematisch ist, weil die einzelnen Schritte nicht rechenintensiv sind.

Die Distanzmatrix wird weitergehend als Adjazenzmatrix eines vollständig verbundenen, gewichteten Graphen interpretiert. Die Knoten des Graphen entsprechen den Szenarien, das Gewicht der Kanten der Ähnlichkeit zwischen ihnen. Der so erstellte Graph wird im Folgenden als Szenariengraph bezeichnet.

**Definition 5.2.** *Der **Szenariengraph** bildet die Szenarien der Datenbank und ihre Ähnlichkeiten als vollständig verbundenen, gewichteten Graphen ab. Die Szenarien sind die Knoten des Graphen, die gewichteten Kanten enthalten, deren Ähnlichkeit.*

Die Interpretation der Szenario-Datenbank als Graph ermöglicht eine visuelle Darstellung, in der die Zusammenhänge der enthaltenen Szenarien analysiert werden. Dafür wird eine Abbildung erzeugt, in der die euklidische Distanz zwischen Knoten möglichst proportional zum Kantengewicht ist. Diese Methodik wird als Einbettung eines Graphen bezeichnet (s. Abschnitt A.1.3).

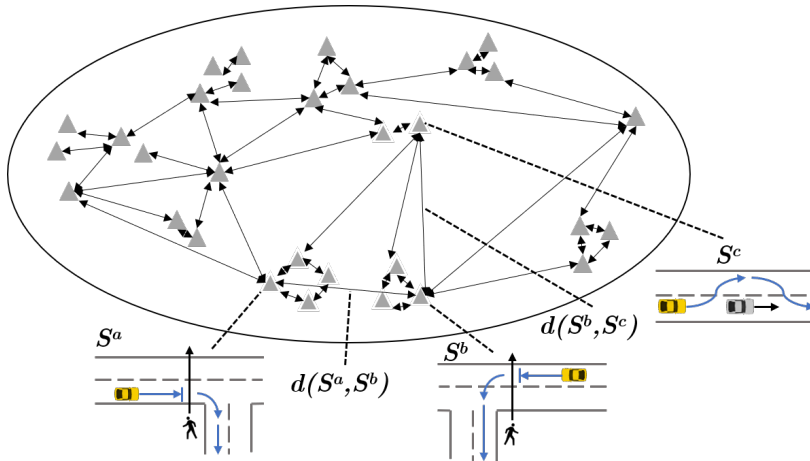


Abbildung 5.10: Visualisierung der Ähnlichkeit der Szenarien über das manöverbasierte Distanzmaß und die Einbettung des Szenariengraphen

Das Ergebnis der Einbettung ist die zweidimensionale Abbildung der enthaltenen Szenarien (s. Abbildung 5.10), in der jedem Szenario die Koordinaten  $(x, y)$  zugeordnet werden. Ein Dreieck repräsentiert hierbei ein Szenario, doppelte Pfeile sind als Distanz interpretierbar. Zur besseren Übersichtlichkeit werden nicht alle Distanzen abgebildet. Basis für die Ähnlichkeitsberechnung ist das entwickelte manöverbasierte Distanzmaß (s. Abschnitt 5.2.3). Ähnliche Szenarien liegen nahe beieinander, unähnliche Szenarien haben einen großen Abstand zueinander. So liegen z. B. die Gruppen der beiden Abbiege-Szenarien näher beieinander als das Doppelspurwechsel-Szenario.

Die Visualisierung der Szenarien lässt sich dabei als ODD interpretieren (s. Abbildung 3.8), in denen die Fahrfunktion operiert. Die inneren Strukturen, wie z. B. Gruppen ähnlicher Szenarien und Ausreißer, lassen sich so erkennen.

Für die Umsetzung der Einbettung wurden in Abschnitt A.1.3 verschiedene Methoden vorgestellt. Sowohl die Dimensionsreduktion als auch kraftbasierte Einbettungen lassen sich konzeptionell für das manövergestützte Testen verwenden. Die Auswahl der geeigneten Methode richtet sich in diesem Fall nach der Menge der zu visualisierenden Szenarien: Da kraftbasierte Layouts aufgrund des Rechenaufwands nur für mehrere hunderte bis tausende Szenarien



einsetzbar sind und nur eine lokale Optimierung vornehmen (s. Tabelle A.1), fällt diese Möglichkeit bei einer entsprechend hohen Anzahl an Szenarien weg. Ist die Anzahl der zu visualisierenden Szenarien geringer oder wird nur eine gefilterte Untergruppe der Szenarien als Datensatz verwendet, sind kraftbasierte Layouts eine Alternative, da sie unter Umständen eine intuitivere Darstellung erzeugen (s. Tabelle A.1).

Die vorgestellten Layouts wurden in unterschiedlichen Domänen mit bis zu ca. 25.000 Datenpunkten [168] verwendet, dennoch reduziert sich die Anschaulichkeit und damit auch die Stärke dieser Darstellungsform mit steigender Anzahl an Datenpunkten. Durch das vorgestellte Ähnlichkeitsmaß wird diesem Effekt entgegengewirkt, da Szenarien mit denselben Manöversequenzen eine Distanz von 0 besitzen und daher auf denselben Punkt im Layout projiziert werden. Die Anzahl an effektiv dargestellten Datenpunkten verringert sich dadurch. Eine weitere Möglichkeit zur Reduktion der angezeigten Szenarien bieten Filter, um nur spezifische Kategorien von Szenarien anzuzeigen.

### **Clustering und Selektion der Szenarien**

Gruppen ähnlicher Szenarien sind in der so erzeugten Darstellung zwar visuell erkennbar, für eine automatisierte Auswertung werden zusätzlich algorithmisch Cluster gebildet (s. Abbildung 5.7). Abbildung 5.11 zeigt beispielhaft drei Cluster, die Farbe der Dreiecke gibt die Zugehörigkeit zu den Clustern an.

Die Cluster bilden die Grundlage, um Testfälle zur Ausführung auszuwählen. Pro Cluster wird ein Szenario ausgewählt, welches als repräsentatives Szenario (s. Definition 3.12) für umliegende Szenarien genutzt wird.

Das entwickelte Vorgehen betrachtet die Situation, dass eine feste Anzahl an auszuführenden Testfällen zur Verfügung steht, welche als Testkapazitäten bezeichnet werden. Diese sind durch vorhandene Prüfstände, Simulationsrechner und Versuchsfahrzeuge vorgegeben und müssen optimal genutzt werden, indem redundante Testfälle vermieden werden. Algorithmisch betrachtet ist die Auswahl des optimalen Testsets (s. Abbildung 5.7) damit ein Optimierungsproblem, bei dem die Repräsentanz des ausgewählten Sets an Szenarien maximiert wird.

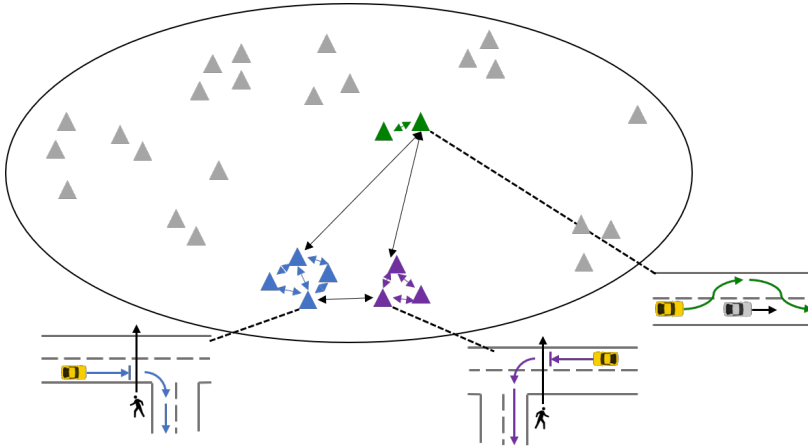


Abbildung 5.11: Clustering der Szenarien für drei Gruppen ähnlicher Szenarien

**Definition 5.3.** Die *Repräsentanz eines Testsets* beschreibt die Fähigkeit von ausgewählten Szenarien, wesentliche Eigenschaften der Grundgesamtheit der Szenarien abzubilden.

Eine quantifizierbare Repräsentanz wird auf Basis des entwickelten Distanzmaßes (s. Abschnitt 5.2.3)  $d$  erstellt:

- $n_t$ : Anzahl auszuführender Testfälle
- $n_c$ : Anzahl der erkannten Cluster
- $n_s$ : Anzahl der Szenarien im Szenarienkatalog
- $T = \{S^{t1}, S^{t2}, \dots\}$ : Set an ausgewählten Testszenarien, wobei gilt:  $len(T) = n_t$

Diese Distanz eines Szenarios  $S^a$  zum Testset  $T$  entspricht der Distanz  $d$  des Szenarios  $S^a$  zum nächstliegenden Szenario des Sets  $T$ :

$$d(S^a, T) = \min_i (d(S^a, S^{ti})) \quad (5.4)$$

Die Summe der Distanzen aller Szenarien zum nächstliegenden ausgewählten Szenario wird als Gesamtdistanz  $\Delta_T$  des Testsets bezeichnet und folgendermaßen berechnet:

$$\Delta_T = \sum^S d(S, T) \quad (5.5)$$

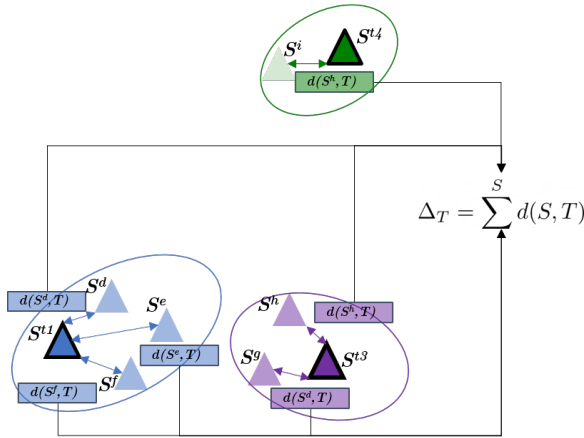


Abbildung 5.12: Veranschaulichung der Notationen zu den berechneten Distanzen für die Repräsentanz eines Testsets

Die Gesamtdistanz steht somit in direktem Zusammenhang mit der Repräsentanz des Testsets. Eine hohe Repräsentanz ist dann erreicht, wenn eine niedrige Gesamtdistanz gegeben ist. In diesem Fall ist, durchschnittlich betrachtet, jedes Szenario durch einen naheliegenden ausgewählten Testfall repräsentiert. Für hohe Gesamtdistanzen dagegen ist die Repräsentanz niedrig.

Es sind zwei unterschiedliche Prinzipien zum Erstellen der Cluster möglich: Eine Möglichkeit nutzt den zuvor erstellten dimensionsreduzierten Raum der Einbettung, in dem jedem Szenario ein Punkt  $(x, y)$  zugeordnet wird. Das für das Clustering verwendete Distanzmaß  $d$  ist dann die euklidische Distanz der Koordinaten der Szenarien. Vorteil dieses Vorgehens ist es, dass etablierte Algorithmen, wie z.B. k-means auf diese Struktur von Eingangsdaten optimiert sind und dadurch verwendet werden können. Zusätzlich sind Ergebnisse aus Visualisierung und Clustering direkt miteinander gekoppelt, weil die Visuali-

sierung als Input für das Clustering verwendet wird. Das Vorgehen baut aber auf der Annahme auf, dass die Einbettung die Ähnlichkeiten ausreichend genau abbildet.

Um diese Ungenauigkeit zu vermeiden, wird das Clustering nicht auf der Projektion des Graphen, sondern unmittelbar mit den berechneten Distanzen durchgeführt. Dadurch schränkt sich die Wahl der möglichen Algorithmen ein, da als Eingangsdaten keine Koordinaten in einem Raum, sondern eine nicht-euklidische Distanzmatrix vorhanden ist.

Für die Auswahl des Clusteralgorithmus ergeben sich somit folgende Anforderungen:

- *Vorgegebene Clusteranzahl:* Die Anzahl auszuführender Testfälle  $n_t$  ist vorgegeben, da jedes Cluster von genau einem Testszenario repräsentiert wird. Dadurch muss dem Algorithmus die Anzahl der Cluster als Parameter vorgegeben werden.
- *Distanzmatrix als Eingang:* Die berechneten Distanzen werden direkt als Eingang für das Clustering verwendet, der Algorithmus muss eine Distanzmatrix als Eingang unterstützen.
- *Nicht-euklidische Distanzen:* Das erstellte Ähnlichkeitsmaß ist keine euklidische Distanz. Daher muss der Algorithmus in der Lage sein, mit allgemeinen paarweisen Distanzen zu rechnen.

Algorithmus	Vorgegebene Clusteranzahl	Distanzmatrix	Nicht-euklidische Distanzen
k-Means [169]	✓	×	×
k-Medoids [170]	✓	✓	✓
DBSCAN [171]	×	✓	✓
Gaussian Mixture Models [172]	✓	×	×
Hierarchisch agglomerativ [173]	✓	✓	✓
Spektrales Clustering [174]	✓	✓	✓

Tabelle 5.1: Anwendbarkeit verschiedener Clustering-Algorithmen für die gegebenen Anforderungen

Durch die Anforderungen scheiden einige bewährte Algorithmen aus (s. Tabelle 5.1): k-Means und Gaussian Mixture Models benötigen die Koordinaten von Punkten als Eingangsgröße und sind nur mit euklidischen Distanzen nutzbar. Bei DBSCAN ist es nicht möglich, die Anzahl der Cluster a priori anzugeben. k-Medoids, hierarchisch agglomeratives und spektrales Clustering dagegen erfüllen die Anforderungen und sind somit nutzbar. Im Vergleich dieser drei Verfahren hat das spektrale Clustering die aufwändigste Berechnung und eignet sich nur für eine vergleichsweise geringe Anzahl von Clustern. Da aber viele unterschiedliche Szenarien erwartet werden, wird für dieses Konzept eine andere Methode bevorzugt. K-Medoids und hierarchisch agglomeratives Clustering sind beide verwendbar. Da k-Medoids robuster gegen Ausreißer ist, ist es für den Anwendungsbereich dieses Konzepts besser geeignet.

### 5.2.4 Manöver als Anweisungen für die Ausführung von Testszenarien

Die Szenarien des erzeugten Testsets werden im nächsten Prozessschritt “Ausführen der Testszenarien” (s. Abbildung 5.2) ausgeführt. Dies ist der praktische Kern der Testdurchführung, in dem die Anforderungen an das Testobjekt in den Testszenarien in realer oder simulierter Umgebung überprüft werden. Durch die Ausführung der Szenarien werden aus der manöverbasierten Beschreibungsform des Szenarios  $S$  Zeitreihen  $\vec{z}$  erstellt.

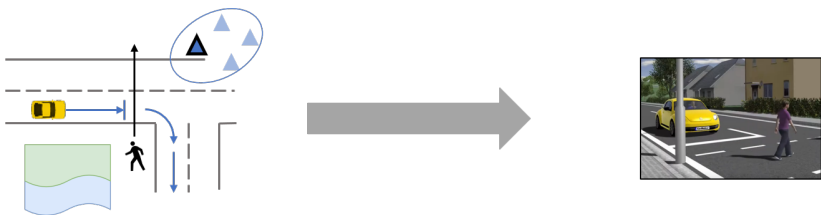


Abbildung 5.13: Beispiel für Ausführung eines repräsentativen Szenarios auf der SiL-Plattform IPG CarMaker [121].

Grundvoraussetzung für diesen Prozessschritt ist die Anforderung der Ausführbarkeit (s. Anforderung 2). Eine automatisierte Ausführung, z. B. in SiL- und HiL-Systemen, erfordert weitergehend eine formalisierte Beschreibung (s.

Anforderung 5), die für eine erhöhte Effizienz ebenfalls parametrisierbar sein muss (s. Anforderung 6). Wenn Testfahrer auf realen Testfahrten involviert sind, ist zusätzlich eine intuitive Darstellung der Szenarien notwendig (s. Anforderung 4). Dafür werden die visuellen Darstellungen der Manöver-Matrizen genutzt (s. Abschnitt A.2).

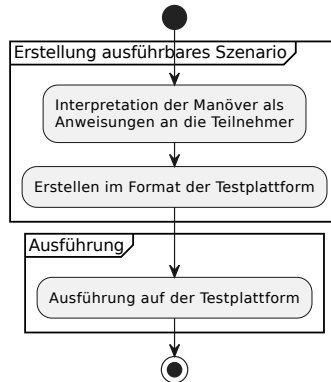


Abbildung 5.14: Aktivitätsdiagramm für das Ausführen der Szenarien auf einer Testplattform

Stellvertretend für das gesamte Cluster wird für die Ausführung dessen repräsentatives Szenario ausgewählt (s. Abbildung 5.13).

Während die Manöver in den vorherigen Prozessschritten beschreibenden Charakter hatten, werden sie für die Ausführung der Szenarien als Anweisungen an die Teilnehmer interpretiert (s. Abbildung 5.14). Die Verwendung von Manövern in dieser Form ist als Eingang für Simulationstools verbreitet (s. Abschnitt 3.4.2). Auch der Industriestandard OpenSCENARIO [54] baut auf Manövern auf. Die entworfene manöverbasierte Beschreibungsform (s. Abschnitt 5.2.7) eignet sich konzeptionell ebenso für die Ausführung, auch wenn das technische Format nicht direkt von den bestehenden Tools unterstützt wird. Deshalb muss für das Szenario eine Instanz im proprietären Format der Testplattform erstellt werden (s. Abbildung 5.14). Für SiL-Prüfstände ist dies OpenSCENARIO, in Abschnitt 6.3 wird daher gezeigt, wie aus der entwickelten Szenariobeschreibung eine standardisierte OpenSCENARIO-Datei erstellt werden kann. Anschließend wird das Szenario mit dieser Datei ausgeführt.

### 5.2.5 Validierung der ausgeführten Szenarien

Im Anschluss an die Ausführung wird überprüft, ob im ausgeführten Szenario  $S$  tatsächlich der erwünschte Verlauf des spezifizierten Testszenarios  $S^t$  durchgeführt wurde und ob es die beschriebenen Inhalte besitzt. Dieser Vorgang wird, analog zur Validierung des Testobjekts, als Szenariovalidierung bezeichnet und im Prozessschritt “Evaluation der Testfälle” durchgeführt.

**Definition 5.4.** Die *Szenariovalidierung* bezeichnet die Überprüfung eines ausgeführten Szenarios im Vergleich zu dem im Voraus spezifizierten Szenario.

Mögliche Gründe für eine Abweichung sind z. B. die fehlende Interaktion von Verkehrsteilnehmern oder invalide Parameterkombinationen beim Ableiten von konkreten Szenarien aus logischen Szenarien.

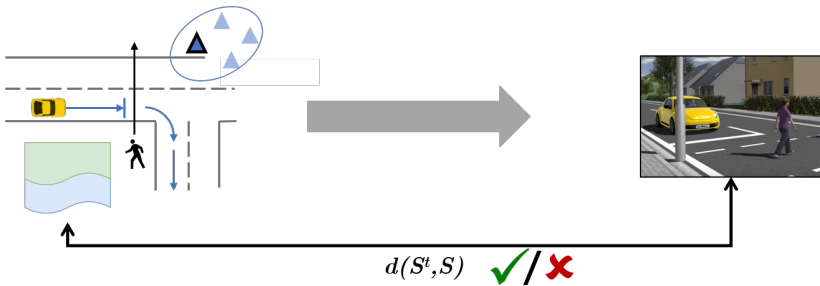


Abbildung 5.15: Validierung der ausgeführten Szenarien über das entwickelte manöverbasierte Distanzmaß

Eine Validierung der ausgeführten Szenarien wird über die Betrachtung der Ähnlichkeiten der Szenarien (s. Abschnitt 5.2.3) durchgeführt (s. Abbildung 5.16). Dazu wird das ausgeführte Szenario mit dem spezifizierten Szenario verglichen. Eine Herausforderung dabei ist, dass das ausgeführte Szenario durch Zeitreihen  $\vec{x}$  beschrieben ist, das spezifizierte Szenario im Gegensatz dazu durch Manöver. Um die Manöver des ausgeführten Szenarios zu bestimmen, werden die entwickelte Manöver-Identifikation und Szenario-Extraktion (s. Abschnitt 5.2.2) angewendet. Somit haben ausgeführtes und spezifiziertes Szenario dieselbe Beschreibungsform. Dies ermöglicht einen Vergleich durch

das zuvor erstellte Ähnlichkeitsmaß (s. Abschnitt 5.2.3). Ist die Ähnlichkeit der beiden Szenarien zu gering, bzw. Distanz der beiden Szenarien zu hoch, wird die Durchführung als invalide klassifiziert (s. Abbildung 5.16). Invalide Ausführungen dürfen nicht direkt für die Sicherheitsargumentation verwendet werden, da das Testobjekt nicht im gewünschten Umfeld überprüft wurde.

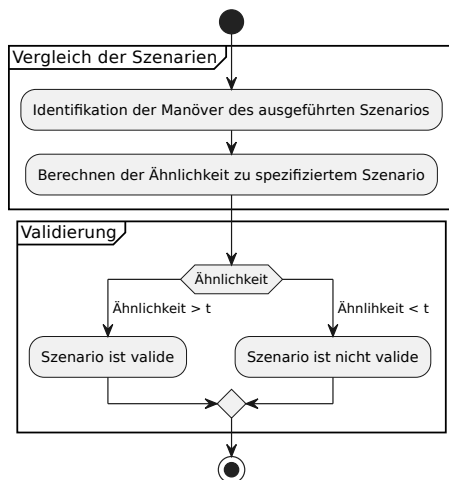


Abbildung 5.16: Aktivitätsdiagramm für die manöverbegünstigte Validierung von ausgeführten Test-szenarien

Abbildung 5.17 zeigt beispielhaft ein valides und ein invalides ausgeführtes Szenario. Dabei wird im invaliden Durchgang die Kombination aus Geschwindigkeit des Fahrzeugs, Geschwindigkeit des Fußgängers und Zeitpunkt, zu dem der Fußgänger seine Bewegung startet, so gewählt, dass der Fußgänger erst nach dem Fahrzeug über die Straße läuft. Ein Bremsen des Fahrzeugs ist also nicht mehr notwendig, obwohl dies, z.B. bei einem Testfall für einen Notbremsassistenten, ein relevanter Anteil ist.

Während beim validen Szenario die Manöver von ausgeführtem und spezifiziertem Szenario gleich sind, unterscheiden sich die Manöver des invaliden Szenarios von diesen. Durch die errechnete Distanz wird dies quantifiziert und das Szenario als invalide erkannt. Das genaue Festlegen des Parameters  $t$  muss im Rahmen des Absicherungsprozesses durch Experten erfolgen. Ein Indikator ist die maximale Distanz der Szenarien innerhalb des Clusters des repräsentativen Szenarios.



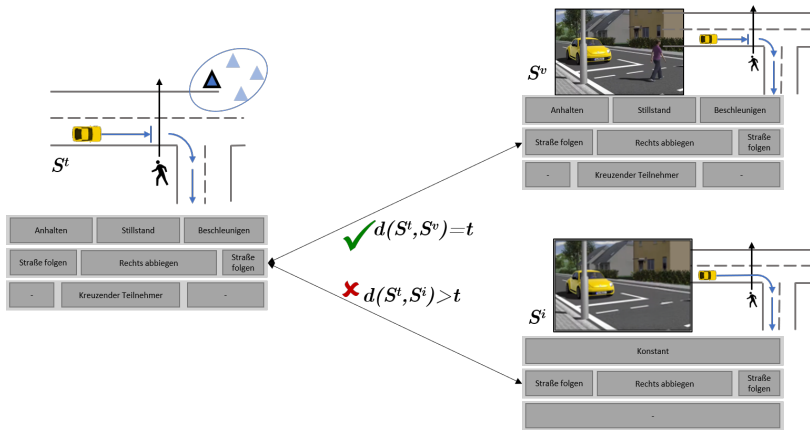


Abbildung 5.17: Beispiel für ein valide ausgeführtes Szenario  $S^v$  und ein invalide ausgeführtes Szenario  $S^i$ . Durch Vergleich der manöverbasierte Distanzen zum Testszenario  $S^t$  wird das invalide Szenario erkannt.

## 5.2.6 Sicherheitsargumentation durch Abdeckung des Szenarienraums

Eine vollständige Sicherheitsargumentation betrachtet neben den Szenarien auch die Sicherheitsanforderungen. Sie ist nicht Fokus dieser Dissertation (s. Abschnitt 5.1.1), es wird aber eine denkbare Vorgehensweise vorgestellt: Dazu wird jedes Szenario durch ein repräsentatives Szenario vertreten (s. Abbildung 3.8). Somit wird sichergestellt, dass der gesamte Szenarienraum durch die repräsentativen Szenarien abgedeckt ist. Es wird dabei angenommen, dass ein Testobjekt die Sicherheitsanforderungen in einem Szenario erfüllt, wenn dies im zugehörigen repräsentativen Szenario zutrifft.

Im manövergestützten Testen ist dieses Vorgehen durch den Szenariengraphen (s. Abschnitt 5.2.3), das Clustering und die Repräsentanz des Testes (s. Definition 5.3) abgebildet. Übertragen auf das vorgestellte Ähnlichkeitsmaß muss die Distanz von jedem Szenario  $S$  zu einem Szenario des Testsets  $T$  kleiner als ein definierter Schwellenwert  $d_{max}$  sein. Folglich ist die Distanz in jedem Cluster des Repräsentanten zu allen anderen Szenarien kleiner als  $d_{max}$ . Zusätzlich

muss das Testobjekt in jedem repräsentativen Szenario die an es gestellten Sicherheitsanforderungen erfüllen.

Weiterer Untersuchung bedarf u.a. das Festlegen des Maximalwertes  $d_{max}$ . Dafür muss auch der Zusammenhang zwischen den Sicherheitsanforderungen und dem Ähnlichkeitsmaß untersucht werden. Zusätzlich ist eine Abschätzung notwendig, ab wann ausreichend viele Szenarien aufgezeichnet wurden, sodass die Menge an unbekannten Szenarien ausreichend klein ist (s. Herausforderung 2). Diese Themen bieten Ausgangspunkte für zukünftige Arbeiten (s. Abschnitt 8.2).

### **5.2.7 Architektur der manöverbasierten Szenariobeschreibung**

Die entwickelte Szenariobeschreibungsform baut auf den in Abschnitt 3.1 erläuterten Definitionen und Einordnungen zu Manövern auf. Die Manöver bilden dabei ein abstraktes Modell des Verhaltens der Verkehrsteilnehmer ab. Das Ziel der erstellten Szenariobeschreibungsform ist es, die Modellierung so durchzuführen, dass die in Kapitel 4 vorgestellten Anforderungen erfüllt werden.

Die Struktur des Szenarios wird durch ein Metamodell definiert, in dem Beziehungen zwischen den zuvor definierten Elementen dargestellt werden. Damit ordnet sich die Beschreibungsform in den Typ der modellbasierten Szenariobeschreibungen ein (Abschnitt 3.4.5). Ein Szenario ist dann eine Instanz dieses Metamodells.

#### **Elemente der Szenariobeschreibung**

Die Elemente der Szenariobeschreibung sind der Baukasten, mit dem die einzelnen Szenarien erstellt und beschrieben werden. Jedes Element bildet einen bestimmten Aspekt des Szenarios ab. Abbildung 5.18 zeigt die verwendeten Elemente des Konzepts und, falls vorhanden, deren übergeordnete Klassen. Die Beziehungen dieser Klassen untereinander sind in Abbildung 5.19 dargestellt. Im Folgenden werden die Definitionen für die Elemente gegeben:

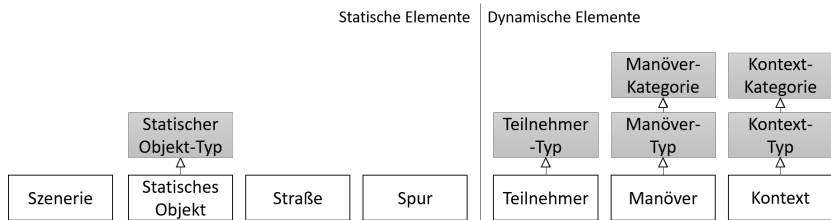


Abbildung 5.18: Elemente der Szenario-Beschreibungsform (weiß) und übergeordnete Gruppierungen (grau).

Die Begriffe Szenario, Szenerie und Teilnehmer wurden bereits in Abschnitt 3.2.1 erläutert und deren Definitionen werden übernommen. Die erweiterten Elemente der Szenerie orientieren sich an den Inhalten des Standards ASAM OpenDRIVE [175].

**Definition 5.5.** Das Straßennetz wird durch **Straßen-Elemente** dargestellt. Jede Straße verläuft entlang einer Referenzlinie. Eine Straße enthält mindestens eine Fahrspur mit einer Breite größer als 0. (nach [175])

**Definition 5.6.** Fahrspuren werden durch das Element **Spur** innerhalb eines Straßen-Elements dargestellt. Alle Straßen besitzen Spuren. Die Anzahl der Spuren ist nicht limitiert. (nach [175])

**Definition 5.7. Statische Objekte** sind Gegenstände, die eine Straße beeinflussen, indem sie ihren Verlauf erweitern, begrenzen und ergänzen. (nach [175])

Im Gegensatz zu Straßen besitzen Spuren einen definierten Typ. Mögliche Typen sind z.B. Fahrspur, Radweg und Fußweg. Auch statische Objekte gehören einem bestimmten Typ an. Die häufigsten Beispiele sind Fußgängerüberwege und Verkehrshindernisse. Die für das Konzept verwendeten Typen werden in Abschnitt 6.5 definiert.

Die für Manöver eingeführte Definition 3.1 wird auch für dieses Konzept verwendet. Zusätzlich werden den Manövern Typen und Kategorien zugeordnet. Ein Manöver ist immer von einem definierten Typ. Jeder Typ ist wiederum

genau einer Kategorie zugeordnet (s. Abbildung 5.18). Für jeden Zeitpunkt und jede Kategorie ist immer genau ein Manöver aktiv. Beispiele für Manöver-Kategorien sind “Spur” und “Geschwindigkeit” (s. Abschnitt 3.1.3). Beispiele für Manöver-Typen sind Spurwechsel (Kategorie “Spur”) und Anhalten (Kategorie “Geschwindigkeit”). Eine Auflistung der verwendeten Manöver-Typen und deren Kategorien findet sich in Abschnitt 6.5.

Da sich durch die Manöver nicht alle relevanten Umstände eines Teilnehmers beschreiben lassen, wird zusätzlich das Element “Kontext” eingeführt:

**Definition 5.8.** *Kontexte beschreiben im Gegensatz zu Manövern Umstände, welche nicht oder nur teilweise durch den Verkehrsteilnehmer beeinflusst werden können, jedoch einen Einfluss auf den Gesamtzustand der Situation haben. Sie können Beschreibungen der umliegenden Verkehrsinfrastruktur sowie Beschreibungen des Verhaltens anderer Verkehrsteilnehmer umfassen (nach [S1]).*

Beispiele für Kontexte sind ein vorausfahrender oder ein kreuzender Teilnehmer an einer Kreuzung. Beide sind kein aktives Manöver, das durchgeführt werden kann, beeinflussen aber das Verhalten und werden daher auch in der Szenariobeschreibung aufgenommen.

Auch jeder Kontext ist immer von einem definierten Typ. Jeder Typ ist wiederum genau einer Kategorie zugeordnet. Beispiele für Kategorien von Kontexten sind die räumlichen Relationen zu anderen Teilnehmern oder die Konstellation der Teilnehmer an Kreuzungen. Beispiele für Kontext-Typen sind vorausfahrendes Fahrzeug (Relation-Kontext) und kreuzendes Fahrzeug (Kreuzungskategorie). Eine Auflistung der verwendeten Kontext-Typen und deren Kategorien findet sich in Abschnitt 6.5.

### Beziehungen der Elemente

Abbildung 5.19 zeigt die Beziehungen der Elemente in der entwickelten Szenariobeschreibungsform.

Jedem Szenario ist genau eine Szenerie zugeordnet. In einer Szenerie können wiederum mehrere Szenarien stattfinden. Dies ist der Normalfall, wenn z.B. auf derselben Kreuzung unterschiedliche Szenarien nacheinander ablaufen. Eine

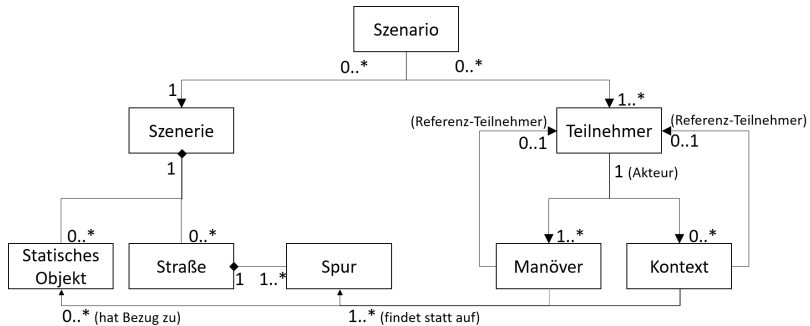


Abbildung 5.19: Metamodell der Szenariobeschreibungsform mit den Elementen und deren Beziehungen als UML-Klassendiagramm

Szenerie kann mehrere statische Objekte und Straßen beinhalten, die jeweils genau einer Szenerie zugeordnet sind. Eine Straße hat dabei mindestens eine Spur, kann aber auch mehrere Spuren haben.

Der dynamische Teil des Szenarios wird über die Teilnehmer beschrieben. Jedes Szenario besitzt mindestens einen Teilnehmer. Derselbe Teilnehmer kann in mehreren Szenarien vertreten sein. Das Verhalten der Teilnehmer wird über Manöver und Kontexte beschrieben. Jeder Teilnehmer führt definitionsgemäß zu jedem Zeitpunkt ein Manöver pro Kategorie aktiv aus (s. Abschnitt 5.2.7). Zusätzlich kann jeder Teilnehmer Kontexte haben. Jedem Manöver und Kontext wird ein Akteur zugeordnet. Dies ist der Teilnehmer, der das Manöver ausführt, bzw. auf den der Kontext eine Auswirkung hat. Einige Manöver und Kontexte besitzen zusätzlich einen Referenz-Teilnehmer, auf den sich das Manöver bzw. der Kontext bezieht. Dies ist z.B. das vorausfahrende Fahrzeug bei einer Folgefahrt.

Der Zusammenhang zwischen Szenerie und dynamischem Verhalten wird folgendermaßen abgebildet: Jedes Manöver und jeder Kontext finden auf mindestens einer Spur statt. Somit sind sie auch den Straßen zugeordnet, auf denen sie stattfinden. Es können auch mehrere Spuren und Straßen dem Manöver bzw. Kontext zugeordnet werden, wenn sich die Ausführung über verschiedene Spuren bzw. Straßen erstreckt. Zusätzlich können Manöver und Kontexte auch Bezug auf statische Objekte haben.

## Parameter der Elemente

Jedes Element kann mehrere Parameter besitzen, welche es genauer beschreiben. Der Wert eines Parameters kann sich von Szenario zu Szenario unterscheiden. Beispiele sind die Geschwindigkeit und Dauer eines Manövers oder die Länge und Breite eines Fahrzeugs. Die jeweiligen Parameter der Elemente werden in den entsprechenden Abschnitten (s. Abschnitt 6.5) eingeführt.

Die Parameter erhöhen die Flexibilität und Wiederverwendbarkeit der Szenarien. Durch Ändern eines oder mehrerer Parameterwerte kann das Szenario leicht variiert werden, ohne die zugrunde liegende abstrakte Beschreibung anzupassen. Sie ermöglichen es zudem, logische und konkrete Szenarien darzustellen. Für logische Szenarien werden dazu Parameterverteilungen aus den konkreten Parametern erstellt.

## 5.2.8 Zusammenfassung des Vorgehensmodells

Die vorgestellten Methoden ergeben zusammengefügt ein Vorgehensmodell, um die Prozessschritte des szenariobasierten Testens mit einer manöverbasierten Szenariobeschreibungsform durchzuführen (s. Abbildung 5.20).

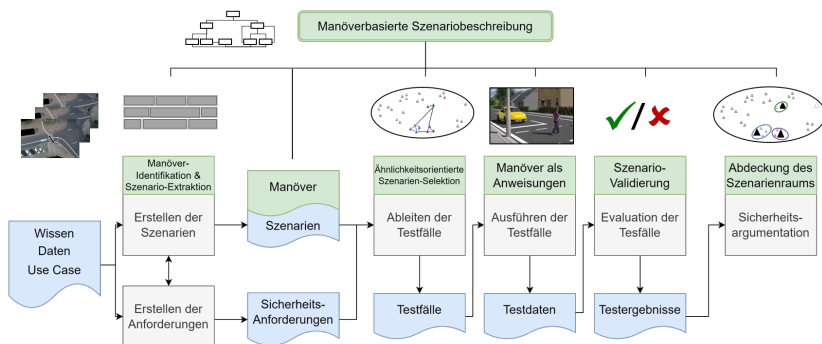


Abbildung 5.20: Zusammenfassung des Modells zum manövergestützten Testen

Die manöverbasierte Szenariobeschreibung ist das Bindeglied der einzelnen Prozessschritte und definiert die Struktur zum Speichern der Manöver und

Szenarien in der Datenbank. Ausgangspunkt sind aufgezeichnete Szenarien, welche die Zeitreihen der Teilnehmer beinhalten. Durch eine Manöver-Identifikation und Szenario-Extraktion werden die ausgeführten Manöver erkannt und zu Sequenzen zusammengefügt. Ein neu entwickeltes Maß zur Ähnlichkeitsberechnung von Szenarien basierend auf den Manövern dient zur optimalen Auswahl von repräsentativen Testszenarien. Diese werden anschließend auf einer Testplattform ausgeführt, wobei die Manöver als Anweisungen an die Teilnehmer interpretiert werden. Um sicherzustellen, dass das ausgeführte Szenario dem spezifizierten entspricht, werden deren Manöver miteinander verglichen. Abschließend bilden die Szenarien und erstellten Cluster einen Teil der Sicherheitsargumentation, indem sie eine Abdeckung des Szenarienraums darstellen.





## 6 Prototypische Umsetzung

### 6.1 Umsetzung der Manöver-Identifikation und Szenario-Extraction

#### 6.1.1 Manöver-Identifikation

Für die Identifikation der Manöver wird ein regelbasierter Ansatz umgesetzt (s. Abschnitt 5.2.2). Als Eingang werden die aufgezeichneten Zeitreihen  $\vec{z}^a$  verwendet, welche folgende Signale beinhalten:

- Geschwindigkeit  $v$
- Beschleunigung  $a$
- Position  $x, y$  und Orientierung  $\phi$
- Zuordnung zur Straße  $id_{str}$  und Spur  $id_{spur}$
- Straßenposition in Längsrichtung  $s$ , Querrichtung  $t$  und Kurswinkel  $\theta$
- Relative Positionen  $pos_{rel}$  und Geschwindigkeiten  $v_{rel}$  zu anderen Teilnehmern

Als Format für die Karte wird der Industriestandard OpenDRIVE [175] genutzt. Je nach Ausgangsdatensatz müssen einige der definierten Signale im Rahmen einer Vorverarbeitung berechnet werden. Die Signale mit Bezug auf die Straße lassen sich z.B. aus den  $x, y$ -Koordinaten und der zugehörigen OpenDRIVE-Datei herleiten. Relative Positionen  $pos_{rel}$  ergeben sich aus den Positionen der Teilnehmer.

Die Klassifikation der unterschiedlichen Kategorien an Manövern wird, mit Ausnahme der Folgemanöver, unabhängiger voneinander berechnet.

Manöver der Kategorie “Geschwindigkeit”

Manöver der Kategorie “Geschwindigkeit” werden klassifiziert, indem die Signale für Geschwindigkeit ( $v$ ) und Beschleunigung ( $a$ ) betrachtet werden. Die Kategorisierung ist als Kette von Wenn-Dann-Befehlen umgesetzt (s. Abbildung 6.1).

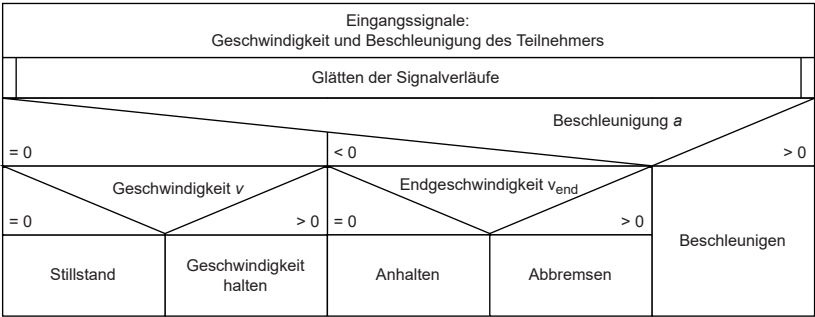


Abbildung 6.1: Struktogramm für die Manöver-Identifikation der Kategorie “Geschwindigkeit”

In einem ersten Schritt wird die Beschleunigung betrachtet, anschließend die Geschwindigkeit (s. Abbildung 6.1). Sind z. B. beide Werte 0, befindet sich der Teilnehmer im Manöver *Stillstand*. Für die anderen Manöver sind weitere passende Regeln umgesetzt.

Um die Stabilität der Klassifikation zu erhöhen und das Rauschen in den Signalen zu reduzieren, werden diese vor der Verarbeitung geglättet. Da die Signale nicht immer genau 0 sind, wird zusätzlich ein Toleranzbereich um diesen Wert festgelegt, sodass auch relativ kleine Werte als 0 betrachtet werden. Zur vereinfachten Darstellung ist dies nicht im Struktogramm gezeigt. Die Größe des Toleranzbereichs erfolgt durch manuelle Optimierung.

Manöver der Kategorie “Folgen”

Um die Manöver der Kategorie *Folgen* zu identifizieren, werden die relativen Positionen  $pos_{rel}$  und Geschwindigkeiten  $v_{rel}$  zu vorausfahrenden Teilnehmern betrachtet. Die Erkennung, ob es einen vorausfahrenden Teilnehmer gibt, wird durch den zugehörigen Kontext (s. Abschnitt 6.1.1) bereitgestellt.

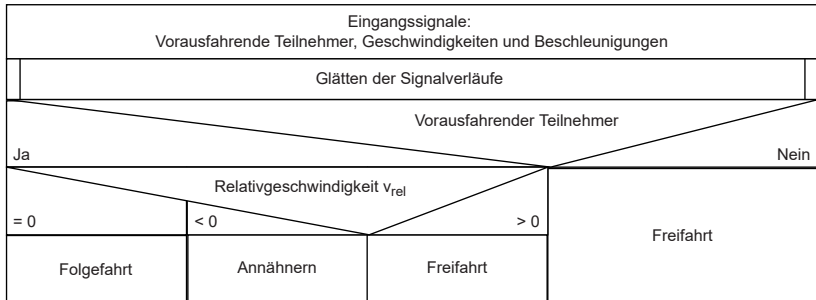


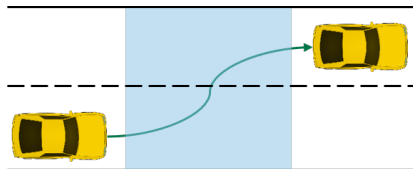
Abbildung 6.2: Struktogramm für die Manöver-Identifikation der Kategorie “Folgen”

Existiert kein vorausfahrender Teilnehmer (s. Abbildung 6.2), ist das Manöver *Freifahrt* aktiv. Andernfalls wird über die relative Geschwindigkeit zu diesem Teilnehmer klassifiziert, ob eine *Folgefahrt* ( $v_{rel} = 0$ ) oder ein *Annähern* ( $v_{rel} < 0$ ) durchgeführt wird. Ist die Relativgeschwindigkeit  $v_{rel}$  größer als 0, wird dies ebenfalls als *Freifahrt* betrachtet, da der vorausfahrende Teilnehmer keinen Einfluss auf die eigene Geschwindigkeit hat.

Wie bei der Klassifikation zu Manövern der Kategorie “Geschwindigkeit”, werden die Signale vorab geglättet und ein Toleranzbereich um den Wert “0” angelegt, um die Stabilität der Identifikation zu erhöhen.

### Manöver der Kategorie “Spur”

In der Kategorie “Spur” werden zwei Manöver unterschieden: *Spurwechsel* und *Spur halten*.

Abbildung 6.3: Darstellung der Ausdehnung eines Manövers *Spurwechsel* (blau)

Um die Spurwechsel zu identifizieren, wird ein zweistufiges Verfahren angewandt:

1. Identifizieren der Zeitpunkte von Spurwechseln:  
Eine Änderung in den Werten des Spursignals  $id_{spur}$  zeigt einen Spurwechsel an. Ausnahmen bilden Zeitpunkte, in denen sich zusätzlich das Straßensignal  $id_{str}$  ändert.
2. Ermittlung der Ausdehnung des Manövers für jeden so identifizierten Spurwechsel:  
Ausgehend vom Zeitpunkt der Änderung im Spursignal ist der Spurwechsel in beide Richtungen so lange aktiv, bis eine der folgenden Bedingungen falsch wird:
  - a) Ableitung der  $t$ -Koordinate kleiner als ein Schwellenwert
  - b) Winkel zwischen Fahrtrichtung und Spurrichtung größer als ein Schwellenwert

Durch dieses Vorgehen werden die *Spurwechsel*-Manöver inklusive deren Ausdehnungen ermittelt (s. Abbildung 6.3). Wird dabei mehr als eine Spur überschritten, wird dies als ein einziges Manöver betrachtet. Die Anzahl der überquerten Spuren geht als Parameter in dieses ein.

Zu jedem Zeitpunkt, in dem kein Manöver *Spurwechsel* aktiv ist, wird dann das Manöver *Spur halten* gesetzt.

### **Manöver der Kategorie “Route”**

Um die Manöver der Kategorie “Route” zu identifizieren, werden die Informationen zur Karte, insbesondere zu den Kreuzungen, und die Orientierung des Teilnehmers analysiert.

In einem ersten Schritt wird überprüft, ob sich der Teilnehmer auf einer Kreuzung befindet (s. Abbildung 6.4). Ist dies nicht der Fall, ist das Manöver *Straße folgen* aktiv.

Um zwischen den Manövern *Rechts abbiegen*, *Kreuzung überqueren*, *Links abbiegen* und *Wende* zu unterscheiden, wird die Richtungsänderung  $\Delta\Phi$  während der Überfahrt der Kreuzung betrachtet. Der mögliche Winkel wird dabei auf

Eingangssignale: Karteninformationen und Orientierung des Teilnehmers				
Befindet sich auf Kreuzung				
Ja	Richtungsänderung $\Delta\Phi$ während Kreuzungsüberahrt			Nein
$225^\circ < \Delta\Phi < 315^\circ$	$\Delta\Phi < 45^\circ$ $\Delta\Phi > 315^\circ$	$45^\circ < \Delta\Phi < 135^\circ$	$135^\circ < \Delta\Phi < 225^\circ$	Straße Folgen
Rechts abbiegen	Kreuzung überqueren	Links Abbiegen	Wende	

Abbildung 6.4: Struktogramm für die Manöver-Identifikation der Kategorie "Route"

vier Bereiche aufgeteilt, welche den Manövern passend zugeordnet werden (s. Abbildung 6.4).

### Kontexte der Kategorie "Relation"

Für die Kontexte *Vorausfahrender Teilnehmer* und *Folgender Teilnehmer* werden die relativen Positionen mit Bezug zur Karte verwendet. Eine Klassifikation anhand der relativen  $x$ ,  $y$ -Koordinaten ist nicht möglich, weil dann Kurvenfahrten nicht korrekt berücksichtigt werden. Um zu erkennen, ob ein anderer Teilnehmer vorausfährt, wird aus den Straßen und Spuren der OpenDRIVE-Karte eine Graphstruktur erstellt (s. Abbildung 6.5). Die Sektionen der Straße und deren Spuren sind als Knoten modelliert. Wenn eine Verbindung zwischen zwei dieser Sektionen besteht, wird dafür eine Kante erstellt. Somit ist der mögliche Fahrverlauf entlang der Straßen und Spuren abgebildet.

Kandidaten für vorausfahrende Teilnehmer sind solche, die vom Akteur aus in diesem Graphen erreicht werden können. Gibt es zu einem Teilnehmer keine Verbindung (oberes Fahrzeug in Abbildung 6.5), fällt dieser als Kandidat aus. Weitergehend werden alle so gefundenen Kandidaten mit der Route des Akteurs verglichen. Da das Szenario aufgezeichnet ist, ist diese bekannt. Weichen die Routen voneinander ab (unteres Fahrzeug in Abbildung 6.5), wird der Kandidat nicht weiter betrachtet. Die verbleibenden Kandidaten werden als *vorausfahrende Teilnehmer* klassifiziert, wenn der Abstand entlang der Route unter einem Schwellwert ist (rechtes Fahrzeug in Abbildung 6.5).

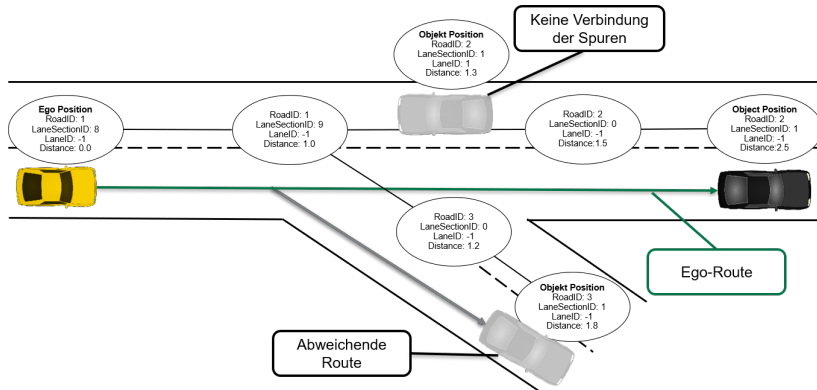


Abbildung 6.5: Finden des nächsten vorausfahrenden Fahrzeugs in einem aufgezeichneten Szenario anhand der OpenDRIVE-Karte

Dieselbe Methodik wird auch eingesetzt, um *Folgende Teilnehmer* zu identifizieren. Als Variation wird die Suche im Graphen dabei rückwärts durchgeführt.

## Kontexte der Kategorie “Kreuzung”

Um die Kontexte auf den Kreuzungen zu erkennen, werden zwei Schritte durchgeführt (s. Abbildung 6.6). Als erster Schritt wird überprüft, ob gleichzeitig zum Akteur ein weiterer Teilnehmer an der Kreuzung anwesend ist und diese überquert.

Für die so ermittelten Teilnehmer wird dann festgestellt, ob er die Bedingungen für einen der modellierten Kontexte erfüllt (s. Abschnitt 6.5.4). Dazu wird verglichen, von welcher zur Kreuzung angrenzenden Straße und die Teilnehmer kommen ( $Ego^{-1}$  und  $Ref^{-1}$ ) und auf welche sie die Kreuzung verlassen ( $Ego^{+1}$  und  $Ref^{+1}$ ). Kommen z.B. beide Fahrzeuge von derselben Straße ( $Ego^{-1} = Ref^{-1}$ ), aber biegen unterschiedlich ab ( $Ego^{+1} \neq Ref^{+1}$ ), ist ein *Ausscherender Teilnehmer* detektiert. Entsprechende Regeln gelten auch für die Kontexte *Kreuzender Teilnehmer* und *Einscherender Teilnehmer* (s. Abbildung 6.6).

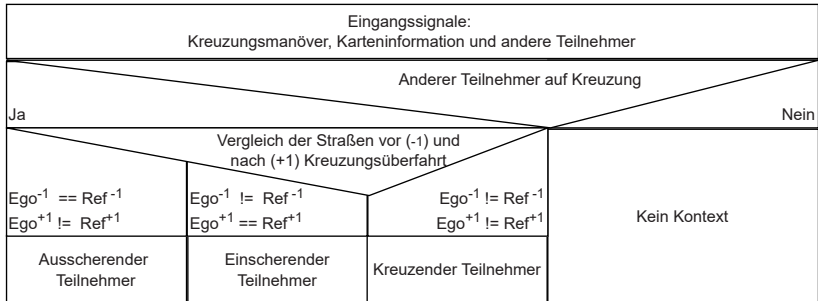


Abbildung 6.6: Struktogramm für die Kontext-Identifikation der Kategorie “Kreuzung”

### 6.1.2 Szenario-Extraktion

Ausgangspunkte für die Szenario-Extraktion sind zum einen die erkannten Manöver, zum anderen das entworfene Szenario-Metamodell (s. Abbildung 5.19), in dem die Elemente und deren Verbindungen definiert sind. Nach den Regeln des Metamodells werden anschließend jedem Szenario die Manöver zugeordnet, die innerhalb der Zeitspanne des Szenarios stattfinden.

Technologisch umgesetzt wird die Datenbank in der Structured Query Language (SQL). Die im Metamodell definierten Klassen werden darin durch eigene Tabellen repräsentiert. Die Attribute der Szenarien, Manöver und Kontexte werden so als Spalten definiert. Verknüpfungen zwischen den Elementen sind durch “Associations” abgebildet. Zum Beispiel werden alle Manöver des Typs *Beschleunigen* in einer Tabelle gespeichert, zusätzlich werden die Parameter wie Start- und Endgeschwindigkeit, Dauer oder durchschnittliche Beschleunigung dort gesichert. Associations sind etwa gleichzeitige Manöver oder Kontexte, wie ein *rechts abbiegen* während der Beschleunigung.

Da jedes Szenario mehrere Teilnehmer und jeder Teilnehmer mehrere Manöver und Kontexte beinhalten kann, steigt die Anzahl der Einträge in den Datenbanktabellen nichtlinear an. SQL ist ein weitverbreiteter Standard und skalierbar, um auch große Datenmengen effizient zu verwalten. Zusätzlich sind durch die Speicherform als Tabellen aggregierte Analysen der Manöver unmittelbar durchführbar, z.B. können alle Manöver *Spurwechsel* mit einer bestimmten Dauer durch eine einfache Query abgerufen werden.

## 6.2 Umsetzung der ähnlichkeitsorientierten Szenarien-Selektion

### 6.2.1 Erstellen logischer Szenarien

Im Verlauf des manövergestützten Testens werden die aufgezeichneten Szenarien durch ihre Manöversequenzen abstrahiert und repräsentiert. Szenarien mit identischen Manöversequenzen sind also in der weiteren Betrachtung identisch. Das bedeutet auch, dass sie immer eine maximale Ähnlichkeit bzw. eine Distanz von 0 besitzen.

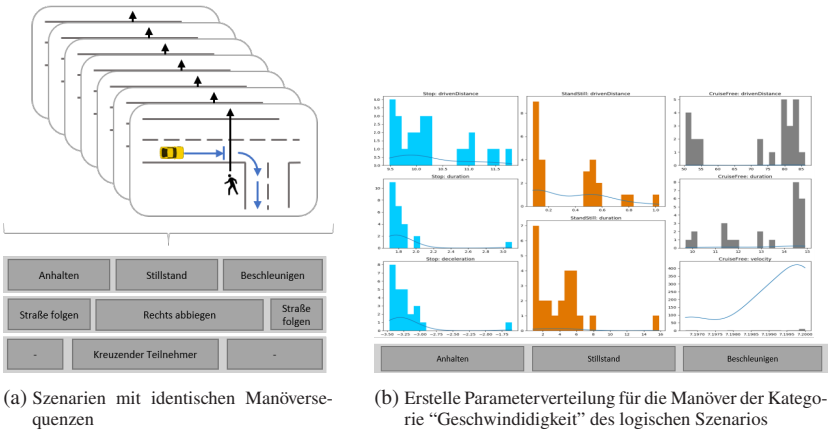


Abbildung 6.7: Erstellen eines logischen Szenarios aus aufgezeichneten Szenarien

Um dies abzubilden, werden Szenarien mit identischen Manöversequenzen zu logischen Szenarien zusammengefasst (s. Abbildung 6.7). Die Anzahl der enthaltenen Szenarien wird als Parameter des logischen Szenarios modelliert.

Für jedes Manöver innerhalb eines logischen Szenarios ist dann nicht nur ein einzelner Wert für ein Attribut vorhanden, sondern es werden Verteilungen aus den im logischen Szenario enthaltenen konkreten Szenarien erstellt.



## 6.2.2 Berechnen der Ähnlichkeiten

In diesem Abschnitt wird eine initiale Parametrisierung der Ähnlichkeitsberechnung (s. Abschnitt 5.2.3) für den Proof-of-Concept (s. Abschnitt 7.1) vorgestellt. Für den Vergleich werden alle modellierten Manöver- und Kontextkategorien (s. Abschnitt 6.5) betrachtet, nur der Kontext “Relation” geht nicht mit. Dessen Auswirkungen auf das Ego-Fahrzeug werden durch die Manöver der Kategorie “Folgen” abgebildet. Die initialen Gewichtungsfaktoren  $c_k$  (s. Gleichung 5.3) für jede Kategorie sind 1, d.h. jede Kategorie geht zu gleichen Teilen in die Gesamtdistanz ein:

$$c_{geschw} = c_{folgen} = c_{route} = c_{spur} = c_{kreuzung} = 1 \quad (6.1)$$

Für die weiteren Prozessschritte wird die Ähnlichkeit zwischen jedem Szenarienpaar berechnet. Die Ergebnisse werden zur Weiterverarbeitung in einer Distanzmatrix  $D$  gespeichert.

### Normierung

Die Sequenzen der unterschiedlichen Manöverkategorien variieren in ihrer Länge. Während die Anzahl an Manövern in der Kategorie “Route” z.B. durch die Anzahl an Kreuzungen im Szenario limitiert ist und Manöver der Kategorie “Spur” vergleichsweise selten vorkommen, können die Manöver der Kategorie “Geschwindigkeit” innerhalb eines Szenarios häufiger wechseln.

Da längere Sequenzen bei der Berechnung über das Sequenzalignment auch höhere Distanzen erzeugen, gehen diese dann stärker in die Gesamtdistanz ein. Deshalb wird eine Normierung durchgeführt. Sie bewirkt, dass die Distanzen unabhängig von der absoluten Länge der Manöversequenzen dennoch vergleichbar bleiben. Zu diesem Zweck wird die ermittelte Distanz für jede Kategorie durch die Summe der Manöver dividiert:

$$^{norm}d_k = \frac{d_k}{len(M_k^a) + len(M_k^b)} \quad (6.2)$$

## Parametrisierung des Sequenzalignments

Als Implementierung der Distanzberechnung wird auf die Bibliothek BioPython zurückgegriffen<sup>1</sup>. Für die Parametrisierung des Sequenzalignments (s. Abschnitt A.1.1) müssen die Kosten für Match  $d_{match}$ , Mismatch  $d_{mismatch}$  und Gap angegeben werden. Gap wird dabei in zwei Varianten aufgeteilt, das Öffnen eines Gaps  $d_{open\_gap}$  und das Verlängern von Gaps  $d_{gap}$ .

Für die Berechnung werden folgende Werte verwendet:

$$d_{match} = 0 \quad (6.3)$$

$$d_{mismatch} = 2 \quad (6.4)$$

$$d_{open\_gap} = 1 \quad (6.5)$$

$$d_{gap} = 0,5 \quad (6.6)$$

Durch die gewählten Werte erzeugt ein Match keine Kosten, identische Manöversequenzen haben also eine Distanz von 0. Mismatches erzeugen höhere Distanzen als Gaps, wobei das Verlängern von Gaps weniger Kosten erzeugt als das Öffnen von neuen.

### 6.2.3 Umsetzung des Szenariographen und des Clusterings

Da die Visualisierung und das Clustering verbreitete und domänenunabhängige Methoden sind, existieren etablierte Implementierungen dafür. Auf diese wird im Rahmen der Umsetzung zurückgegriffen.

Basis für den Szenariengraphen ist die zuvor berechnete Distanzmatrix  $D$ . Diese beschreibt die Gewichte der Kanten zwischen den Knoten, die die Szenarien abbilden. Für die Umsetzung der Berechnung der Einbettung wird der Algorithmus zur Dimensionsreduktion Uniform Manifold Approximation and Projection (UMAP) eingesetzt. Als Implementierung wird auf das Python-

---

<sup>1</sup> <https://biopython.org>

Paket `umap-learn`<sup>2</sup> zurückgegriffen. Die Koordinaten der Einbettung  $(x_e, y_e)$  werden darin direkt aus der vollständig gefüllten Distanzmatrix  $D$  berechnet.

$$(x_e, y_e) = UMAP(D) \quad (6.7)$$

Auch das Clustering nutzt die berechnete Distanzmatrix  $D$  als Eingang (s. Abschnitt 5.2.3). Als Implementierung wird das Python-Paket `SciPy`<sup>3</sup> genutzt, das effiziente Clusteralgorithmen zur Verfügung stellt. Der ausgewählte Algorithmus `k-Medoids Clustering` (s. Abschnitt 5.2.3) ist in dieser Bibliothek enthalten. Die Zuordnung der Szenarien  $S$  zu den Clustern  $C$  wird anschließend in der Szenariendatenbank gespeichert.

Das `k-Medoids Clustering` wählt automatisch aus den Datenpunkten des Clusters Mittelpunkte aus, welche die Repräsentanten  $R_C$  des Clusters darstellen. Sollte im Rahmen einer anderen Realisierung ein Algorithmus ausgewählt werden, der dies nicht tut (z.B. hierarchisch-agglomerativ oder `k-Means`), werden die Repräsentanten nachträglich ermittelt. Dafür wird in jedem Cluster  $C$  das Szenario ausgewählt, welches den minimalen kumulierten Abstand zu allen anderen Szenarien innerhalb des Clusters besitzt, also die beste Repräsentanz  $\Delta_C$  (s. Definition 5.3) innerhalb des Clusters aufweist. Mit Gleichung 5.4 wird damit das Szenario ausgewählt, das folgende Summe minimiert:

$$\Delta_C = \sum_{S \in C} d(R, S) \quad (6.8)$$

## 6.3 Umsetzung der Ausführung von Testszenarien

Der Industriestandard für das Ausführen von Szenarien auf Prüfständen und Simulationsumgebungen ist `OpenSCENARIO XML` [54]. Um die weitverbreitetsten Tools (s. Abschnitt 3.4.2) zu unterstützen, sieht die Umsetzung

---

<sup>2</sup> <https://pypi.org/project/umap-learn/>

<sup>3</sup> <https://scipy.org/>

ein teilautomatisiertes Konvertieren der entwickelten Beschreibungsform in OpenSCENARIO XML vor.

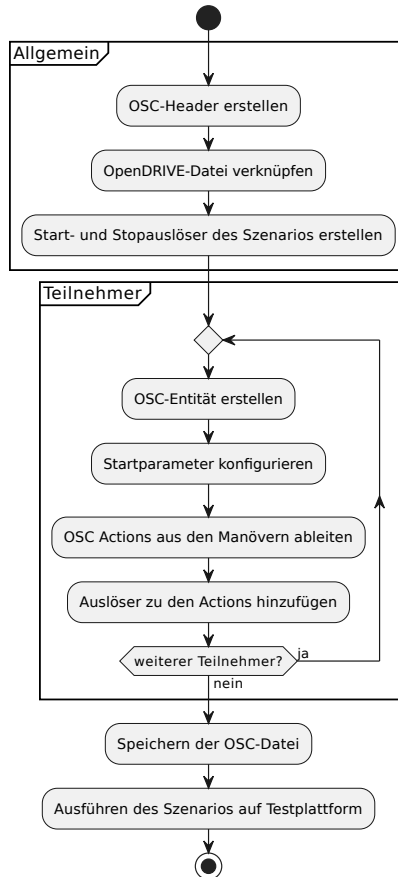


Abbildung 6.8: Aktivitätsdiagramm zur Erstellung von OpenSCENARIO Dateien für die Ausführung der Testszenarien

Die Konvertierung beinhaltet die in Abbildung 6.8 gezeigten Schritte. Für den allgemeinen Teil der Umwandlung werden OpenSCENARIO-spezifische Elemente, wie z.B. der OSC-Header, gesetzt. Der statische Teil des Szenarios wird über die Verknüpfung zu einer OpenDRIVE-Karte abgebildet. Start-

und Stopppauslöser des Szenarios werden aus den Manövern der Teilnehmer erstellt. Zusätzlich wird für jeden Teilnehmer eine OSC-Entität erstellt, in der Typ und Verhalten beschrieben werden. Da OpenSCENARIO nur für die Ausführung optimiert ist, unterscheidet sich der Manöverkatalog von dem, der in diesem Prototyp verwendet wird. Die einzelnen Manöver werden daher in OSC-Äquivalente umgewandelt.

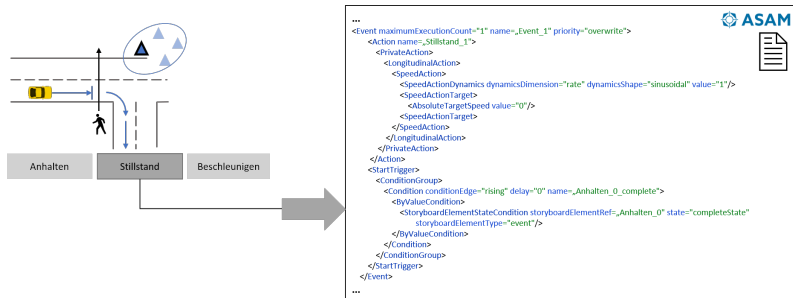


Abbildung 6.9: Beispiel eines konvertierten Stillstandmanövers in den Industriestandard OpenSCENARIO XML

Abbildung 6.9 zeigt die Konvertierung eines Manövers in eine OpenSCENARIO Action am Beispiel *Stillstand* und gibt einen Einblick in die Beschreibung dieses Formats. Das Setzen der Geschwindigkeit auf “0” wird über das Element *SpeedAction* erreicht, welches Anweisungen für die Geschwindigkeitsregelung eines Teilnehmers gibt. *AbsoluteTargetSpeedValue* wird die Zielgeschwindigkeit, im Fall des Stillstandmanövers “0”, vorgegeben. Zusätzlich werden über die *SpeedActionDynamics* Eigenschaften mitgegeben, wie diese Geschwindigkeit erreicht werden soll. Über den “StartTrigger” wird die Action gestartet, sobald das vorherige Anhalten-Manöver abgeschlossen ist. Dazu wird der Status der zugehörigen Action überwacht und der Trigger wird aktiviert, sobald dieser den Status “complete” erreicht hat. Die weiteren Zeilen der Datei bilden die allgemeine OpenSCENARIO-Struktur ab und werden im Handbuch [54] erläutert.

Die so erzeugten Dateien werden von gängigen Simulationstools unterstützt und die Szenarien sind somit für diese ausführbar.

## 6.4 Umsetzung der Validierung der Testszenarien

Der Algorithmus für die Validierung der Testszenarien ist in Abbildung 5.16 festgehalten. Die dafür notwendigen Umsetzungen zur Manöver-Identifikation und Ähnlichkeitsberechnung wurden in den vorherigen Abschnitten beschrieben und werden hier wiederverwendet.

Der elementare Parameter für die Validierung ist die maximale Distanz  $d_{max}$ , die das ausgeführte Szenario vom spezifizierten Szenario haben darf, um als valide zu gelten. Die für diese Parametrisierung notwendigen Annahmen müssen aus der Sicherheitsargumentation kommen (s. Abschnitt 5.2.5). Das exakte Festlegen des Wertes für  $d_{max}$  ist damit nicht Teil dieser Dissertation (s. Abschnitt 5.1.1). Eine Orientierung für die Parametrisierung können die Distanzwerte innerhalb des Clusters des Szenarios geben.

Für diese Umsetzung wird initial eine Distanz von 0 gefordert, es müssen also alle Manöver des Ego-Fahrzeugs im ausgeführten Szenario identisch mit dem definierten Szenario sein.

## 6.5 Verwendete Manöver-, Kontext-, Teilnehmer-, Spur- und Objekt-Typen

Tabelle 6.1 bis Tabelle 6.4 zeigen die modellierten Typen für die jeweiligen Elemente, die für das manövergestützte Testen genutzt werden.

### 6.5.1 Teilnehmer

Die Teilnehmertypen umfassen *PKW*, *Truck*, *Motorrad*, *Fahrrad* und *Fußgänger* (s. Tabelle 6.1). Die Typen orientieren sich an üblicherweise genutzten Taxonomien und verfügbaren Datensätzen. Eine vertiefende Kategorisierung, z.B. von PKW in SUV, Kombi oder Kleinwagen, wäre konzeptionell möglich. Im Rahmen dieses Prototyps wird darauf verzichtet, da dies nicht der Fokus der Arbeit ist.

Typ	Beschreibung
PKW	Auto für den Personenverkehr
Truck	Lastkraftwagen für den Gütertransport
Motorrad	Zweirad mit Motorantrieb
Fahrrad	Zweirad ohne Motorantrieb
Fußgänger	Mensch zu Fuß unterwegs

Tabelle 6.1: Teilnehmertypen und Beschreibungen

Abbildung 6.10 zeigt die Teilnehmertypen inklusive deren Parameter als UML-Klassendiagramm.

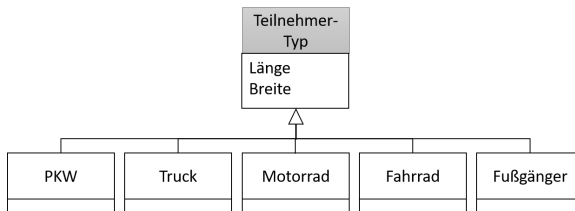


Abbildung 6.10: UML-Klassendiagramm der Teilnehmer inklusive deren Parameter

## 6.5.2 Szenerie

Für die Spurtypen werden *Fahrspur*, *Radweg*, *Gehweg* und *Parkplatz* verwendet (s. Tabelle 6.2). Die Spurtypen sind eine Untermenge der in OpenDRIVE definierten Spuren und bilden den urbanen Bereich ab. Auch hier ist konzeptionell eine Erweiterung möglich, um z.B. den Standstreifen auf Autobahnen beschreiben zu können. Der einzige statische Objekttyp ist der *Zebrastreifen*, welcher relevant für den urbanen Bereich ist (s. Tabelle 6.2).

Abbildung 6.11 bietet einen Überblick der statischen Elemente inklusive deren Parameter als UML-Klassendiagramm.

Typ	Beschreibung
Fahrspur	Fahrstreifen auf der Straße für Fahrzeuge
Radweg	Separate Spur für Fahrräder
Gehweg	Separate Spur für Fußgänger
Parkplatz	Bereich zum Abstellen von Fahrzeugen
Zebrastreifen	Markierung für Fußgängerübergänge

Tabelle 6.2: Spurtypen und statische Objekttypen sowie Beschreibungen

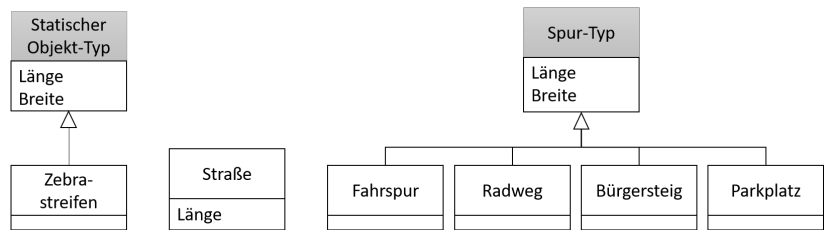


Abbildung 6.11: UML-Klassendiagramm der statischen Elemente inklusive deren Parameter

6.5.3 Manöver

Die modellierten Manöver werden in vier Kategorien eingeteilt (s. Tabelle 6.3).

Die Manöver der Kategorie “Geschwindigkeit” beschreiben die Geschwindigkeit des Akteurs (ausführender Teilnehmer) sowie deren Änderung. Die Kategorie ist unabhängig von der Infrastruktur und anderen Teilnehmern und bezieht sich nur auf den Akteur. Die Manöver der Kategorie sind *Geschwindigkeit halten*, *Beschleunigen*, *Abbremsen*, *Anhalten* und *Stillstand*.

Die Manöver der Kategorie “Folgen” beschreiben die Geschwindigkeit des Akteurs in Bezug auf einen anderen Teilnehmer. Im Manöver *Freifahrt* hat kein anderer Teilnehmer einen Einfluss auf die Geschwindigkeit des Akteurs. Beim Manöver *Annähern* passt der Akteur seine Geschwindigkeit an einen vorausfahrenden Teilnehmer an. Im Manöver *Folgefahrt* sind die Geschwindigkeiten des Akteurs und des vorausfahrenden Teilnehmers identisch.



Manöver-Kategorie	Manöver-Typ	Beschreibung
Geschwindigkeit	Geschwindigkeit halten	Akteur bewegt sich mit konstanter Geschwindigkeit
Geschwindigkeit	Beschleunigen	Akteur erhöht seine Geschwindigkeit
Geschwindigkeit	Abbremsen	Akteur verringert seine Geschwindigkeit
Geschwindigkeit	Anhalten	Akteur verringert seine Geschwindigkeit auf null
Geschwindigkeit	Stillstand	Akteur steht vollständig
Folgen	Freifahrt	Akteur ohne Bezug zu vorausfahrendem Teilnehmer
Folgen	Annähern	Akteur passt seine Geschwindigkeit einem vorausfahrenden Teilnehmer an
Folgen	Folgefahrt	Akteur folgt einem anderen Teilnehmer mit angepasster Geschwindigkeit
Spur	Spur halten	Akteur hält seine aktuelle Fahrspur
Spur	Spurwechsel	Akteur wechselt von einer Fahrspur zur anderen
Route	Straße folgen	Akteur folgt Straße ohne Kreuzung
Route	Rechts abbiegen	Akteur biegt nach rechts ab
Route	Links abbiegen	Akteur biegt nach links ab
Route	Kreuzung überqueren	Akteur überquert eine Kreuzung gerade
Route	Wende	Akteur macht eine Kehrtwende

Tabelle 6.3: Manöver-Kategorien, -Typen und Beschreibungen

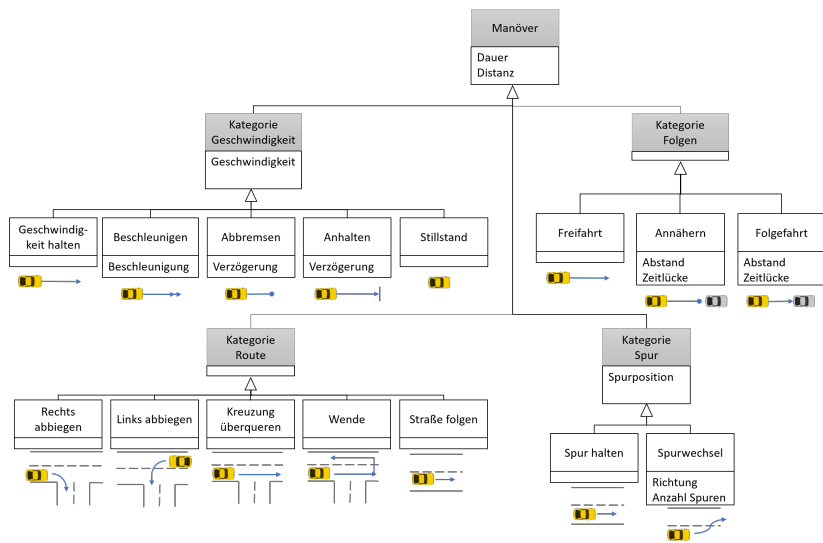


Abbildung 6.12: UML-Klassendiagramm der Manöver inklusive deren Parameter und Visualisierungen

Die Manöver der Kategorie “Spur” beschreiben das Verhalten des Akteurs in Bezug auf die Fahrspuren. Beim *Spur halten* fährt der Akteur innerhalb seiner Spur. Der *Spurwechsel* beschreibt den Wechsel von einer Spur auf eine andere.

Die Manöver der Kategorie “Route” beschreiben das Verhalten des Akteurs in Bezug auf Straßen und Kreuzungen. Auf Kreuzungen ist die Abbiege-Richtung ausschlaggebend für das jeweilige Manöver: *Rechts abbiegen*, *Links abbiegen*, *Kreuzung überqueren* und *Wende*. Befindet sich der Akteur nicht auf einer Kreuzung, ist er im Manöver *Straße folgen*.

Abbildung 6.12 bietet einen Überblick über die Manöver, deren Kategorien, Typen und Parameter als UML-Klassendiagramm. Die Manöver werden zusätzlich durch Visualisierungen veranschaulicht.

6.5.4 Kontexte

Die modellierten Kontexte werden in zwei Kategorien unterteilt (s. Tabelle 6.4).

Kontext-Kategorie	Kontext-Typ	Beschreibung
Relation	Vorausfahrender Teilnehmer	Teilnehmer vor dem Akteur
Relation	Folgender Teilnehmer	Teilnehmer hinter dem Akteur
Relation	Rechter Teilnehmer	Teilnehmer rechts des Akteurs
Relation	Linker Teilnehmer	Teilnehmer links des Akteurs
Kreuzung	Kreuzender Teilnehmer	Teilnehmer, der die Kreuzung vor dem Akteur kreuzt
Kreuzung	Einscherender Teilnehmer	Teilnehmer, der vor dem Akteur einschert
Kreuzung	Ausscherender Teilnehmer	Teilnehmer, der vor dem Akteur auf eine andere Spur abbiegt

Tabelle 6.4: Kontext-Kategorien, -Typen und Beschreibungen

Die Kontexte der Kategorie “Relation” beschreiben die relativen Positionen des Akteurs und eines weiteren Teilnehmers in Bezug auf die Spuren der Straßeninfrastruktur. Beim Kontext *vorausfahrender Teilnehmer* befindet sich der Teilnehmer vor dem Akteur in gleicher Fahrtrichtung. Dabei ist nicht relevant, ob der Teilnehmer einen Einfluss auf die Geschwindigkeit des Akteurs hat. Diese Eigenschaft wird über die Manöver der Kategorie Folgen modelliert. Ein *vorausfahrender Teilnehmer* ist aber Voraussetzung für eine Folgefahrt oder ein Annähern. Ein *folgender Teilnehmer* ist das Pendant zum *vorausfahrenden Teilnehmer*, bei dem sich der Teilnehmer auf der Spur hinter dem Akteur be-

findet. Bei den Kontexten *rechter Teilnehmer* und *linker Teilnehmer* befindet sich ein Teilnehmer auf der jeweils entsprechenden angrenzenden Spur.

Kontexte der Kategorie “Kreuzung” beschreiben das Verhalten anderer Teilnehmer in Bezug auf den Akteur an Kreuzungen. Ein *kreuzender Teilnehmer* fährt vor dem Akteur über die Kreuzung. Der kreuzende Teilnehmer kommt weder von der gleichen Straße wie der Akteur, noch verlässt er sie in die gleiche Straße wie der Akteur. Ein *einscherender Teilnehmer* kommt von einer anderen Straße als der Akteur und ordnet sich vor diesem ein. Akteur und einscherender Teilnehmer verlassen die Kreuzung auf derselben Straße hintereinander. Der *ausscherende Teilnehmer* fährt vor dem Akteur und kommt von derselben Straße an die Kreuzung, verlässt diese aber in eine andere.

Abbildung 6.13 bietet einen Überblick über die Kontexte, deren Kategorien, Typen als UML-Klassendiagramm. Die einzelnen Kontexte sind durch Visualisierungen veranschaulicht.

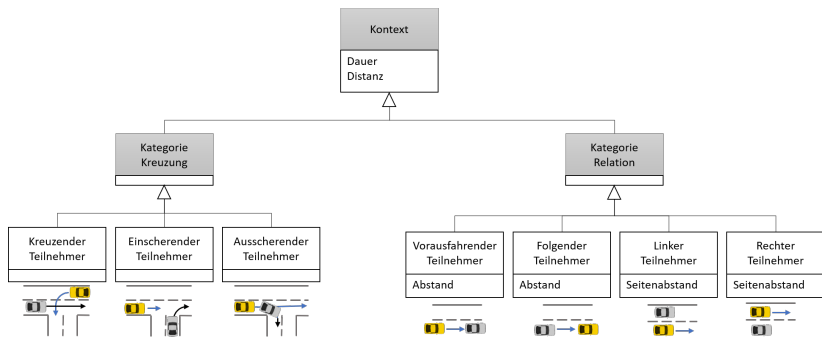


Abbildung 6.13: UML-Klassendiagramm der Kontexte und Visualisierungen



## 7 Evaluation

### 7.1 Proof of Concept

#### 7.1.1 Verwendeter Datensatz

Für den Proof-of-Concept (PoC) werden reale, aufgezeichnete Szenarien verwendet, die an einer T-Kreuzung (s. Abbildung 7.1) in einem Wohngebiet in Mannheim aufgenommen wurden<sup>1</sup>. Auf dieser Kreuzung ist der Testbetrieb für ein autonomes Shuttle geplant<sup>2</sup>. Die Sensorik zur Aufzeichnung ist an Masten in 10 Metern Höhe angebracht. Die erzeugten Daten enthalten die Trajektorien der Teilnehmer im Kartenkoordinatensystem mit einer zeitlichen Auflösung von 10 Hz. Zusätzlich liegt eine zugehörige Karte der Strecke im OpenDRIVE-Format vor.

Die Aufzeichnungen enthalten Teilnehmer der Typen PKW, Truck, Fahrrad und Fußgänger. Als Ego-Fahrzeuge werden die Teilnehmer vom Typ PKW betrachtet, die entlang der für das Shuttle definierten Route für diese Kreuzung fahren. Sie beinhalten damit sowohl Rechts- als auch Linksabbiegevorgänge des Ego-Fahrzeugs. Die aufgezeichneten Szenarien beinhalten zusätzlich alle weiteren Teilnehmer, die gleichzeitig im Erfassungsbereich der Sensorik liegen. Das Geschwindigkeitslimit im Wohngebiet, in dem die Daten aufgezeichnet wurden, beträgt 30 km/h.

Um aus den kontinuierlich aufgezeichneten Daten die Start- und Endzeitpunkte der aufgezeichneten Szenarien zu ermitteln, werden die statischen Eigenschaften des Streckensegments betrachtet. Ein Szenario startet, sobald das Ego-

---

<sup>1</sup> Dieser Datensatz wurde von der ZF Friedrichshafen AG aufgezeichnet und darf im Rahmen einer Kooperation für wissenschaftliche Zwecke verwendet werden.

<sup>2</sup> <https://www.projekt-rabus.de/de/Die-Reallabore/>



Abbildung 7.1: Satellitenansicht der Datensatz-Kreuzung (<https://earth.google.com>)

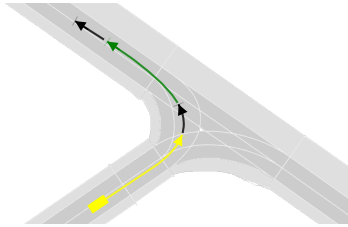
Fahrzeug ca. 20 m vor der Kreuzung ist und endet ebenfalls ca. 20 m nach dieser. Somit wird auch das Verhalten bei der Anfahrt betrachtet. Es wurden durch dieses Vorgehen 135.647 Szenarien aus 9 Monaten Aufzeichnung extrahiert. Der Datensatz hat eine Größe von 96 GB.

### Beispielszenarien

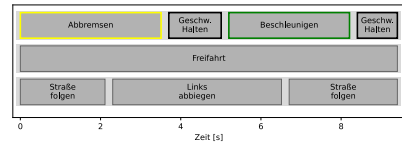
Folgende vier Beispielszenarien werden im PoC aus dem Datensatz ausgewählt und über die Prozessschritte hinweg betrachtet, um das manöverbegünstigte Testen zu untersuchen. Die Fahrzeuge sind als Rechtecke dargestellt, ihre Trajektorien als Linien. Zur vereinfachten Darstellung werden zusätzliche Teilnehmer, die keinen Einfluss auf das Ego-Fahrzeug haben, nicht betrachtet.

#### Beispielszenario 1: Alleine abbiegen

Im ersten Beispielszenario (s. Abbildung 7.2) führt das Ego-Fahrzeug ein Linksabbiegemanöver durch. Es sind keine weiteren Teilnehmer vorhanden. Der Topview zeigt die Fahrzeugposition und die Fahrspur, während die zugehörige Manövermatrix die durchgeführten Manöver zeigt. *Abbremsen* (gelb) vor der Kreuzung, *Beschleunigen* (grün) nach der Kreuzung und *Geschwindigkeit halten* (schwarz) sind farblich markiert. Dieses einfache Szenario wird im PoC verwendet, um grundlegende Aspekte zu betrachten.



(a) Topview auf Beispielszenario 1

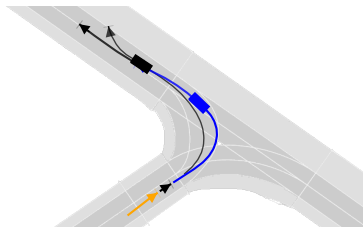


(b) Manövermatrix für Beispielszenario 1

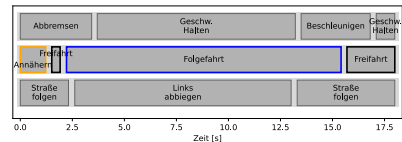
Abbildung 7.2: Beispielszenario 1

### Beispielszenario 2: Folgefahrt

Im zweiten Beispielszenario (s. Abbildung 7.3) wird eine Folgefahrt über die Kreuzung durchgeführt. Hier passt das Ego-Fahrzeug seine Geschwindigkeit und seinen Abstand an ein vorausfahrendes Fahrzeug an. Der Topview veranschaulicht die relative Positionierung beider Fahrzeuge, die Trajektorien der Manöver *Annähern* (orange), *Folgen* (blau) und *Freifahrt* (schwarz) sind farblich eingezeichnet. Das Szenario wird im PoC verwendet, um den Einfluss von Manövern der Kategorie “Folgen” zu zeigen.



(a) Topview auf Beispielszenario 2



(b) Manövermatrix für Beispielszenario 2

Abbildung 7.3: Beispielszenario 2

### Beispielszenario 3: Kreuzender Fußgänger

Im dritten Beispielszenario (s. Abbildung 7.4) begegnet das Ego-Fahrzeug einem Fußgänger (lila Viereck), der quer über die Kreuzung läuft. Der Topview zeigt die kreuzenden Trajektorien, die Farben zeigen die Abfolge der Manöver *Anhalten* (rot), *Stillstand* und *Warten*, bis der Fußgänger die Kreuzung überquert hat (blau) und anschließendes *Beschleunigen* (grün). Diese sind auch in der zugehörigen Manövermatrix zu finden. Zusätzlich ist dort zu erkennen,

wie der Fußgänger als Kontext *Kreuzender Teilnehmer* modelliert ist. Dieses Szenario wird im PoC verwendet, um die im urbanen Bereich auftretenden Fußgänger zu betrachten.

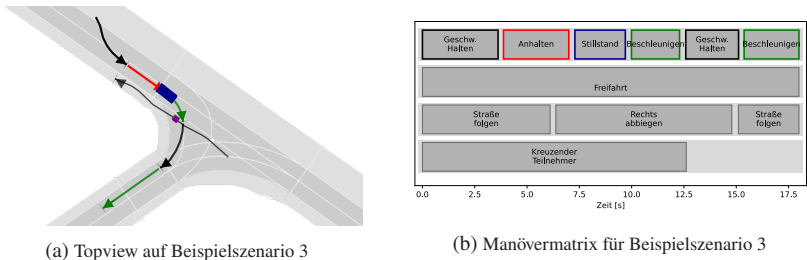


Abbildung 7.4: Beispielszenario 3

**Beispielszenario 4: Kreuzeunde und einscherende Fahrzeuge**

Im vierten Beispielszenario (s. Abbildung 7.5) sind neben dem Ego-Fahrzeug drei weitere Verkehrsteilnehmer involviert, die nacheinander in unterschiedlichen Richtungen über die Kreuzung fahren und dabei die geltenden Vorfahrtsregeln beachten müssen. Die Manöver der Kategorie “Geschwindigkeit” sind wie in den anderen Beispielszenarien farblich markiert, die Kontexte *kreuzender* und *einscherender Teilnehmer* sind in der Manövermatrix abgebildet. Dieses Szenario wird im PoC verwendet, um mehrere miteinander interagierende Teilnehmer und Kontexte zu analysieren.

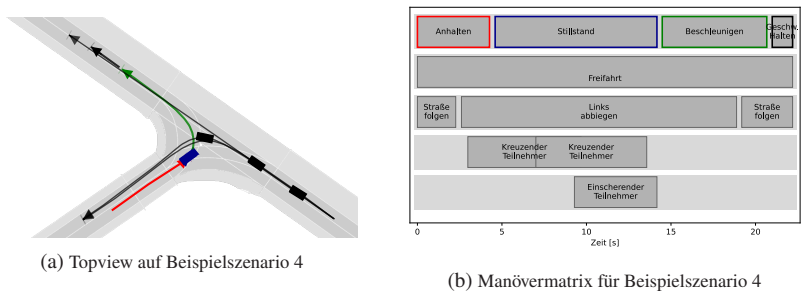


Abbildung 7.5: Beispielszenario 4



### 7.1.2 Manöver-Identifikation und Szenario Extraktion

Die Manöver-Identifikation und Szenario-Extraktion wird anhand von Beispielszenario 4 erläutert. Dazu wird dessen Ablauf über die vier abgebildeten Szenen (s. Abbildung 7.6a – Abbildung 7.6d) veranschaulicht: (a) die Startszene, in der das Ego-Fahrzeug die Kreuzung anfährt, (b) das Anhalten des Ego-Fahrzeugs an der Kreuzung, um Vorfahrt zu gewähren, (c) das Abwarten, in dem zwei kreuzende und ein einscherender Verkehrsteilnehmer die Kreuzung überqueren, und (d) das Ende des Manövers, in dem das Ego-Fahrzeug seine Geschwindigkeit erhöht, um die Kreuzung zu verlassen.

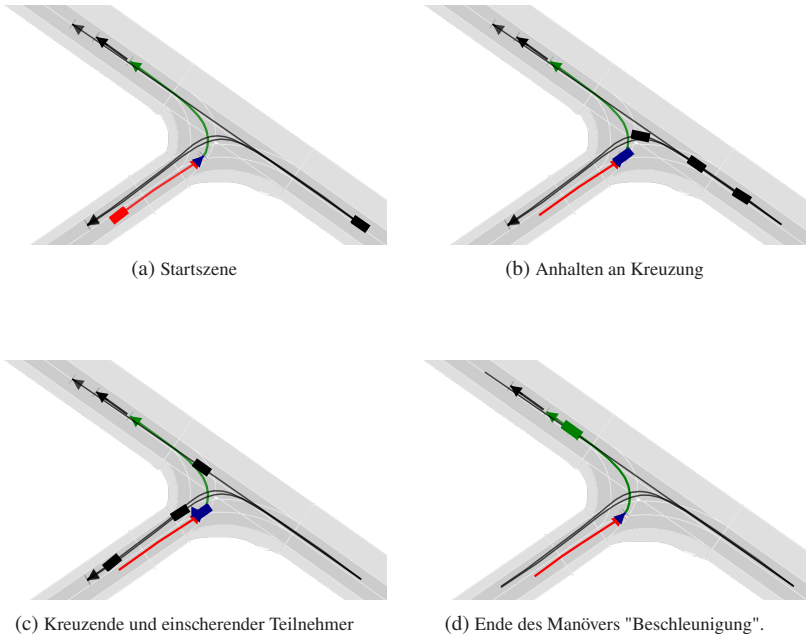


Abbildung 7.6: Ablauf von Beispielszenario 4 mit Ego-Fahrzeug und drei weiteren Teilnehmern an einer Kreuzung. Die Farben zeigen die Manöver der Kategorie "Geschwindigkeit" des Fahrzeugs: Anhalten (rot), Stillstand (blau), Beschleunigen (grün) und Geschwindigkeit halten (schwarz).

### Identifikation der Manöver der Kategorie “Geschwindigkeit”

Die Manöver der Kategorie “Geschwindigkeit” ergeben sich aus dem Signalverlauf der Geschwindigkeit (s. Abbildung 7.7) und den zuvor definierten Regeln zur Klassifikation (s. Abschnitt 6.1.1). Die Farben sind auch in der Übersicht des Szenarios (s. Abbildung 7.6) sichtbar.

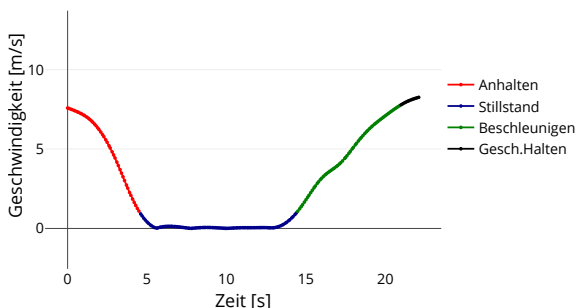


Abbildung 7.7: Geschwindigkeitsverlauf in Beispielszenario 4

Das Manöver *Anhalten* (rot) ist durch eine kontinuierliche Verzögerung charakterisiert, die bis zum Stillstand geht. Das *Stillstand*-Manöver (blau) hat einen konstanten Wert nahe null. Hier lässt sich auch erkennen, dass ein Messrauschen vorhanden ist, da die Kurve leichte Schwingungen zeigt. Das Manöver *Beschleunigen* (grün) beschreibt das Verlassen der Kreuzung mit kontinuierlich steigender Geschwindigkeit, bevor die Beschleunigung geringer wird und in *Geschwindigkeit halten* (schwarz) übergeht.

### Identifikation der Manöver der Kategorie “Route”

Die Manöver für die “Route” ergeben sich aus Signalverläufen der Spur-ID, (s. Abbildung 7.8a) und dem Yaw-Winkel (s. Abbildung 7.8b). Über die Spur-ID lässt sich ermitteln, ob sich das Fahrzeug auf einer Kreuzung befindet. Hierzu wird die hinterlegte OpenDRIVE-Karte verwendet. Für die Richtung des Abbiegemanövers (blau) wird die Abweichung des Yaw-Winkels verwendet (s. Abschnitt 6.1.1), in diesem Szenario  $110^\circ$ , wodurch ein *Links abbiegen* indiziert ist.

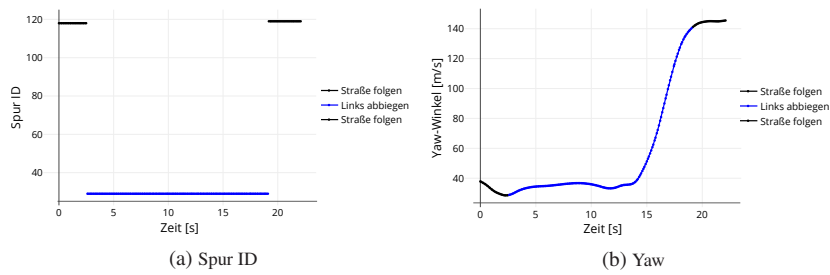


Abbildung 7.8: Signalverläufe für Manöver der Kategorie “Route”

Identifikation der Kontexte der Kategorie “Kreuzung”

Für die Erkennung der Kontexte an der Kreuzung wird zuerst überprüft, ob sich Ego und andere Teilnehmer zur gleichen Zeit auf der Kreuzung befinden (s. Abschnitt 6.1.1). Im Beispielszenario ist dies für alle Teilnehmer gegeben. Anschließend wird durch Fallunterscheidung ermittelt, welche Interaktion zwischen den Teilnehmern besteht, indem die Straße vor ( $Ego^{-1} == Ref^{-1}$ ) und nach der Kreuzung ( $Ego^{+1} == Ref^{+1}$ ) verglichen werden (s. Abbildung 6.6). Die ersten beiden Teilnehmer des Beispielszenarios fahren die Kreuzung von einer anderen Straße als das Ego-Fahrzeug an und verlassen sie auch auf einer anderen als das Ego-Fahrzeug. Sie sind damit *kreuzende Teilnehmer*. Teilnehmer 3 verlässt die Kreuzung auf der gleichen Straße wie das Ego-Fahrzeug und ist damit ein *einscherender Teilnehmer* (s. Tabelle 7.1).

	$Ego^{-1} == Ref^{-1}$	$Ego^{+1} == Ref^{+1}$	Kreuzungs-Kontext
Teilnehmer 1	False	False	Kreuzender Teilnehmer
Teilnehmer 2	False	False	Kreuzender Teilnehmer
Teilnehmer 3	False	True	Einscherender Teilnehmer

Tabelle 7.1: Ermittlung der Kreuzungskontexte für die Teilnehmer des Beispielszenarios

Diskussion der Ergebnisse

Zusammenfassend lässt sich durch die regelbasierte Klassifikation (s. Abschnitt 5.2.2) in den Daten des PoC Manöver und Kontexte identifizieren. Die

Ergebnisse sind anhand der Signalverläufe nachvollziehbar. Das Vorgehen lässt sich auch auf weitere Manöver- und Kontextkategorien ausweiten. Zudem sind die Regeln bei erkannten Fehlklassifikationen anpassbar.

Es treten aber auch Klassifikationsfehler auf (z.B. Abschnitt A.3). Optimierungspotentiale bieten verbesserte Regeln, Parameteroptimierung oder der Einsatz maschinellen Lernens (s. Abschnitt 8.2).

### **7.1.3 Ähnlichkeitsorientierte Szenarien-Selektion**

#### **Logische Szenarien**

Szenarien mit identischen Manöverfolgen werden für die Szenarien-Selektion als identisch angesehen (Distanz = 0, s. Abschnitt 6.2.1). Dieser Zwischenschritt ermöglicht die Erstellung von logischen Szenarien aus den aufgezeichneten Szenarien. Die Anzahl der konkreten Szenarien im logischen Szenario wird weiterhin als Größe des logischen Szenarios bezeichnet.

Im Beispieldatensatz wurden mit diesem Vorgehen aus 135.647 aufgezeichneten Szenarien 9.555 logische Szenarien gefunden. Die Größe variiert dabei von 1 bis 12.890 (s. Abbildung 7.9). Dies entspricht bereits einer Reduktion von 93 % durch den modellbasierten Ansatz der Manöverbeschreibung. Folgende Prozessschritte rechnen dann mit dem reduzierten Datensatz weiter und sind dadurch deutlich schneller durchzuführen (s. Abschnitt 7.1.6).

Das häufigste logische Szenario ist ein einfaches Linksabbiegen ohne weitere Teilnehmer und entspricht dem Beispielszenario 1 (s. Abbildung 7.9). Da die Manöverfolgen identisch sind, können Parameterverteilungen ermittelt werden. Z.B. sind die Geschwindigkeitsverteilungen der Manöver für das logische Szenario wie folgt (s. Abbildung 7.10): Beim Start des Szenarios und zu Beginn des Abbrems-Manövers fährt das Ego-Fahrzeug meistens zwischen 20 km/h und 30 km/h, was durch das Geschwindigkeitslimit von 30 km/h auf der Strecke bedingt ist. In der Kreuzungsüberfahrt, während des Geschwindigkeit-Halten-Manövers, verringert sich die Geschwindigkeit auf unter 20 km/h. Am Ende des Szenarios, im zweiten Geschwindigkeit-Halten-Manöver, fährt das Ego-Fahrzeug in den meisten Aufzeichnungen wieder um die 30 km/h.

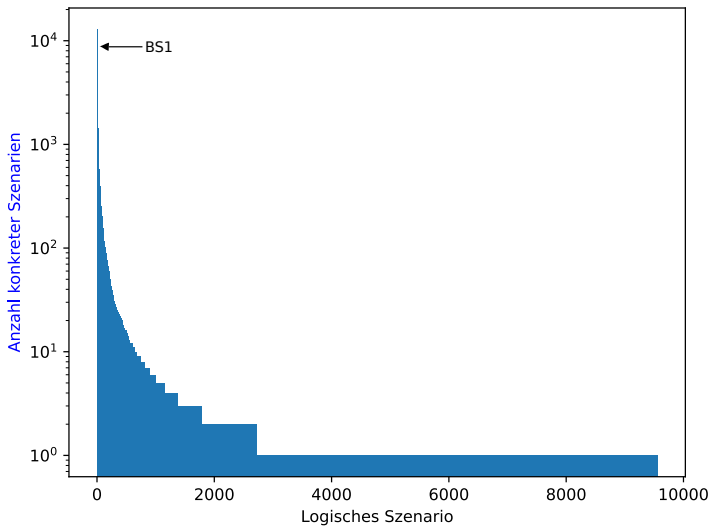


Abbildung 7.9: Größe der logischen Szenarien im Datensatz auf logarithmischer Skala

Logische Szenarien der Größe 1 kommen 6.834 Mal vor und enthalten hauptsächlich Szenarien mit mehreren Teilnehmern und vergleichsweise langen Manöverfolgen: Beispielhaft hier ein Szenario, in dem das Ego-Fahrzeug am Straßenrand anhält und mehrere Teilnehmer passieren lässt (s. Abbildung 7.11).

### Ähnlichkeiten und Distanzen

Die im Realisierungskapitel vorgestellte Parametrisierung (s. Abschnitt 6.2.2) wird für den PoC verwendet, es werden also Kategorien der Manöver und Kontexte mit dem Wert “1” gleich gewichtet.

Für die vier Beispielszenarien ergibt sich damit eine reduzierte Distanzmatrix (s. Tabelle 7.2) mit folgenden Eigenschaften: Die höchste Ähnlichkeit bzw. geringste Distanz besteht zwischen BS 1 und BS 2. Ein Blick auf die zugehörigen Manövermatrizen zeigt, dass hier der einzige Unterschied in den Manövern der Kategorie “Folgen” zu finden ist. (BS 1: *Freifahrt*, BS 2: zusätzlich *Annähern*

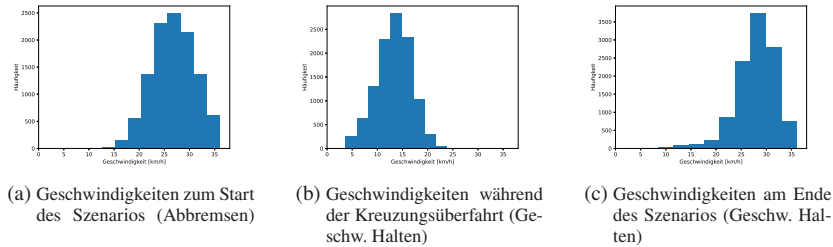


Abbildung 7.10: Parameterverteilungen der Geschwindigkeit für das größte logische Szenario

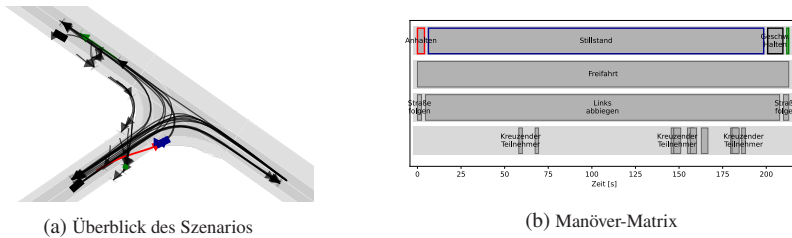


Abbildung 7.11: Ein einzelnes vorkommendes Szenario im PoC mit mehreren (kreuzenden) Teilnehmern und langen Manöverfolgen

und *Folgefahrt*). Die hohe Ähnlichkeit ist also semantisch nachvollziehbar. Beispielszenario 1 hat in Summe die geringste Distanz zu den anderen Szenarien. Das simple Szenario bildet damit eine Art neutralen Mittelpunkt ohne Besonderheiten. Die höchste Distanz findet sich zwischen BS 2 und BS 4. Hier unterscheiden sich sowohl die Art und Anzahl der anderen Teilnehmer, als auch die Manöver der Kategorie “Geschwindigkeit” und “Folgen”. Abbildung 7.12 zeigt zusätzlich die Visualisierung der Ähnlichkeiten als Graph.

Die Distanzmatrix für den gesamten Datensatz des PoC, also die 9.555 aufgezeichneten logischen Szenarien, berechnet sich nach derselben Methodik und besitzt insgesamt 45.644.235 Einträge. Die Berechnung dauert 2 h 45 min.

	BS 1	BS 2	BS 3	BS 4
BS 1 (Nur Ego)	0	0,73	1,68	1,71
BS 2 (Annähern und Folgen)	0,73	0	2,25	2,44
BS 3 (Kreuzender Fußgänger)	1,68	2,25	0	1,37
BS 4 (Vier Fahrzeuge)	1,71	2,44	1,37	0

Tabelle 7.2: Distanzmatrix für die vier Beispielszenarien (BS 1-4)

## Clustering

Für das Clustering wird das k-Medoids-Clustering (s. Abschnitt A.1.2) verwendet, für die Erstellung des Szenariographen UMAP (s. Abschnitt 6.2.3).

### *Beispielszenarien*

Für den einfachen Fall der vier Beispielszenarien sind im Graphen (s. Abbildung 7.12) die entsprechenden Ähnlichkeiten (s. Tabelle 7.2) visualisiert. BS 1 und BS 2, die mit 0,73 die geringste berechnete Distanz haben, werden nah zusammen abgebildet. BS 3 und BS 4 mit größerem Abstand zu diesen, der größte Abstand ist zwischen BS 1 und BS 4, auch dies ist in der Abbildung korrekt dargestellt. BS 1 liegt in der Mitte, da die Ähnlichkeiten zu allen anderen Szenarien vergleichsweise einheitlich sind. Auch ist zu erkennen, dass der Abstand zu BS 2 und BS 4 etwas geringer ist als zu BS 3. Die anderen Werte der Distanzmatrix sind für dieses Minimalbeispiel ebenfalls sinnvoll dargestellt. Es wurden zudem zwei Cluster erstellt und farblich markiert, bestehend aus BS 1 und BS 2, sowie BS 3 und BS 4. Diese stimmen mit den ermittelten Distanzen überein.

### *Cluster im gesamten Datensatz*

Zudem wurden die Cluster für die 9.555 logischen Szenarien des PoC bestimmt. Die Anzahl der Cluster ist ein Parameter, der sich aus der Testkapazität ergibt (s. Abschnitt 5.2.3) wurde für den PoC beispielhaft auf 100 gesetzt. Die Größen der damit errechneten Cluster besitzen folgende Eigenschaften: größtes Cluster: 328, kleinstes Cluster: 11, durchschnittliche Größe: 95, Median 79,50.

### *Cluster der Beispielszenarien im gesamten Datensatz*

Um die Ergebnisse zu plausibilisieren, werden die Cluster betrachtet, die die

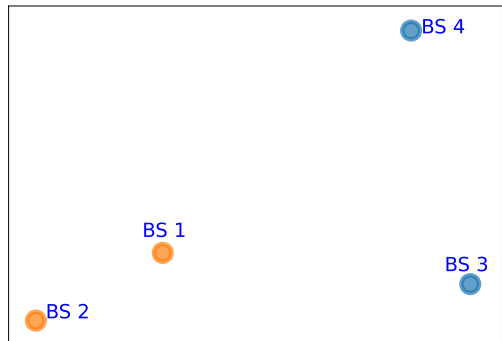


Abbildung 7.12: Szenariengraph für die Beispielszenarien (BS 1-4)

Beispielszenarien enthalten. Dafür werden für jedes Cluster folgende Eigenschaften berechnet (s. Tabelle 7.3): Anzahl der Szenarien im Cluster (Größe), die durchschnittliche Anzahl anderer Teilnehmer an der Kreuzung, die als Kontext klassifiziert wurden (Kreuzungsteilnehmer), der Anteil an Folgen-Manövern im Szenario (Folgen) und die Abbiegerichtung des Ego-Fahrzeugs (Abbiegerichtung). Die Werte der Beispielszenarien sind in Klammern hinzugefügt.

Cluster	Größe	Kreuzungsteilnehmer	Folgen	Abbiegerichtung
Cluster BS 1	248	0,0 (0,0)	0,0 (0,0)	100% (l)
Cluster BS 2	67	0,0 (0,0)	<b>1,18</b> (1,0)	99% (l)
Cluster BS 3	49	<b>1,18</b> (1,0)	0,04 (0,0)	<b>90%</b> (r)
Cluster BS 4	211	<b>2,73</b> (3,0)	0,12 (0,0)	92% (l)

Tabelle 7.3: Eigenschaften für die Cluster der Beispielszenarien

Basierend auf den vorgestellten Beispielszenarien sollen folgende Annahmen erfüllt sein: Die Anzahl an Kreuzungsteilnehmern ist für BS 4 am höchsten. Der Anteil an Folgen-Manövern ist für BS 2 am höchsten. Der Anteil an Rechtsabbiegern ist für BS 3 am höchsten.



Betrachtet man die fett markierten Maximalwerte, sind alle Annahmen erfüllt. Zusätzlich bestehen bei den betrachteten Merkmalen keine oder nur geringe Abweichungen zwischen den Beispielszenarien und den zugehörigen Mittelwerten der Cluster. Die höchsten Abweichungen betreffen die Folge-Manöver bei BS 2 (1,18 zu 1,0) und die Kreuzungsteilnehmer von BS 3 (ebenfalls 1,18 zu 1,0). Auch die Abbiegerichtung innerhalb der Cluster stimmt mit über 90% mit den Beispielszenarien überein. Insgesamt finden sich also innerhalb der Cluster Szenarien, die bezüglich der betrachteten Merkmale semantisch mit den entsprechenden Beispielszenarien übereinstimmen.

### *Visualisierung als Szenariengraph*

Um einen Überblick über die 9.555 logischen Szenarien zu bekommen, werden diese als Szenariengraph dargestellt (s. Abbildung 7.13). Die errechneten Cluster sind in der Abbildung farblich markiert. Da 100 Cluster erstellt wurden, wiederholen sich die Farben für mehrere Cluster. Repräsentanten sind durch einen schwarzen Rand gekennzeichnet.

Die Szenarien sind nicht gleichmäßig verteilt, einige Regionen sind dicht besetzt und bilden optisch erkennbare Gruppen von Szenarien mit hoher Ähnlichkeit. Die charakteristischen Eigenschaften der Regionen sind annotiert. Zwischen diesen Gruppen gibt es teils klare Trennungen, teilweise gehen diese fließend ineinander über. Zudem sind einzelne Ausreißer zu erkennen.

Kompakte Gruppen bilden häufige Standardfälle ab, wie z.B. BS 1. Diffuse Gruppen enthalten Szenarien mit mehr Varianz. Z.B. finden sich BS 3 und BS 4, die durch einscherende und kreuzende Teilnehmer charakterisiert sind, in einer Gruppe mit den Szenarien, in denen weitere Teilnehmer über die Kreuzung fahren. Ausreißer bilden seltene und ungewöhnliche Szenarien ab.

Die Szenarien innerhalb eines Clusters werden auch im Graphen nahe beieinander abgebildet und liegen innerhalb einer der optischen Gruppen. Die Cluster gehen fließend ineinander über. So zeigt sich, dass eine exakte Darstellung aller Distanzen in zwei Dimensionen nicht möglich ist und diese Visualisierung nur eine Näherung darstellt (s. Abschnitt A.1.3). Der Szenario-Graph ist also in der Lage, semantisch ähnliche Szenarien nahe beieinander zu visualisieren, aber nicht exakt alle Distanzen abzubilden.

Die gewählten 100 Cluster sind mehr als die optischen Gruppen im Graphen. Diese Gruppen werden daher auf mehr als ein Cluster aufgeteilt. Für den

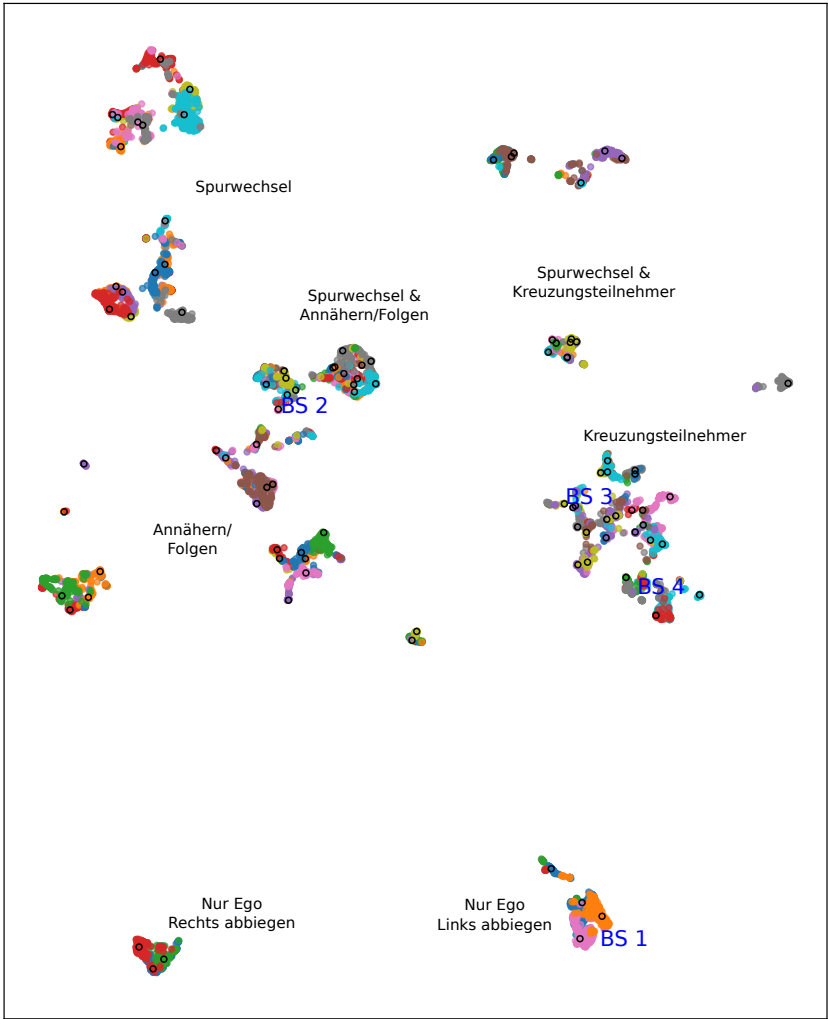


Abbildung 7.13: Visualisierung als Graph mit farblicher Markierung der Cluster inkl. Annotation der Beispielszenarien und Charakteristika von Regionen.

Testprozess bedeutet dies, dass diese Gruppen mit mehreren unterschiedlichen Testfällen abgedeckt werden.

Die Beispielszenarien 1–4 sind im Graphen markiert. Die Positionierung ist allerdings nur teilweise wie im zuvor reduzierten Beispiel (s. Abbildung 7.12). BS 3 und BS 4 sind weiterhin nah zusammen abgebildet, allerdings liegt BS 1 nicht mehr zentral zwischen den anderen, sondern weit von ihnen entfernt.

Zusammenfassend bietet der Szenarien-Graph einen intuitiven Überblick über die Verteilung der aufgezeichneten Szenarien und ermittelten Cluster. Er bleibt aber eine Näherung, eine exakte Darstellung der Ähnlichkeiten nimmt mit der Menge an Datenpunkten ab. Daher unterliegt er gewissen Limitierungen im Einsatz (s. Abschnitt 7.3).

## Auswahl der Repräsentanten

### *Repräsentanten der Beispielszenarien*

Um die Ergebnisse der Selektion zu überprüfen, werden die Distanzen der Beispielszenarien und die Repräsentanten (s. Abbildung A.6) der zugehörigen Cluster betrachtet. Für ein erfolgreiches Ergebnis muss die Distanz zwischen einem Beispielszenario und dem zugehörigen Repräsentanten geringer sein als die Distanz zu den anderen Repräsentanten.

	Rep 1	Rep 2	Rep 3	Rep 4
BS 1	<b>0.11</b>	0.82	2.29	1.38
BS 2	0.84	<b>0.25</b>	2.03	2.11
BS 3	1.70	2.24	<b>0.44</b>	1.20
BS 4	1.72	2.43	2.08	<b>0.33</b>

Tabelle 7.4: Distanzwerte der Beispielszenarien (BS 1–4) zu deren Repräsentanten (Rep 1–4)

Dies ist für alle Beispielszenarien erfüllt (s. Tabelle 7.4). Während die Werte zu den Repräsentanten durchschnittlich 0,18 betragen, ist die zu Repräsentanten der anderen Beispielszenarien durchschnittlich 1,3 und damit mehr als 7 Mal höher.

### *Repräsentanz für den gesamten Datensatz*

Nach Gleichung 5.4 lässt sich die Gesamtdistanz eines Testsets  $\Delta_T$  berechnen, um die Qualität der ausgewählten Szenarien zu quantifizieren. Für den PoC beläuft sich diese auf 3.081.

Als Vergleich hätten 100 zufällig ausgewählte Szenarien eine Gesamtdistanz von ca. 3.753.

## 7.1.4 Ausführung

Die tatsächliche Ausführung wird im PoC über vier Artefakte (s. Abbildung 7.14) durchgeführt und über das Simulationsprogramm IPG CarMaker realisiert: Ausgehend von der Manövermatrix (Abbildung 7.14a) wird die OpenSCENARIO-Datei erstellt (Abbildung 7.14b). Die Länge der Datei für dieses einfache Szenario beträgt bereits 240 Zeilen und 369 Wörter, deshalb wird im Bild nur das Manöver *Abbremsen* dargestellt. Hier wird noch einmal deutlich, warum OpenSCENARIO nicht die Anforderung der Kompaktheit erfüllt.

Im nächsten Schritt wird die Datei in einen IPG CarMaker “Testrun” umgewandelt. Das Simulationsprogramm bietet dafür eine Importfunktion an. Die erstellten Manöver sind im Scenario Editor, einer GUI des Programms, dargestellt und können hier bei Bedarf angepasst werden. In Abbildung 7.14c sind die vier Manöver der Kategorie “Geschwindigkeit” zu sehen: *Bremsen* (Decelerate\_1552498), *Geschwindigkeit halten* (Keep\_1552499), *Beschleunigen* (Accelerate\_1552500) und nochmals *Geschwindigkeit halten* (Keep\_1552501). Als Konvention zur Benennung sind der Manövertyp und die ID aus der Manöverdatenbank gewählt, um den Ursprung der Manöver nachzuverfolgen.

Abschließend wird das Szenario simuliert. Abbildung 7.14d zeigt einen Ausschnitt aus der Visualisierung während der Kreuzungsüberfahrt.

Das Konvertieren in eine OpenSCENARIO-Datei dauert 0,5 Sekunden, das Importieren dieser in CarMaker 4 Sekunden und die tatsächliche Simulation, inkl. Laden und Vorbereitung der Simulationsumgebung für das Szenario, ebenfalls 4 Sekunden.

Sowohl die erstellte Szenariobeschreibungssprache, als auch OpenSCENARIO und die Beschreibung in IPG CarMaker bauen auf Manövern auf. Wie in Abschnitt 5.2.4 und Abschnitt 6.3 erläutert, ist die Umwandlung daher konzeptio-

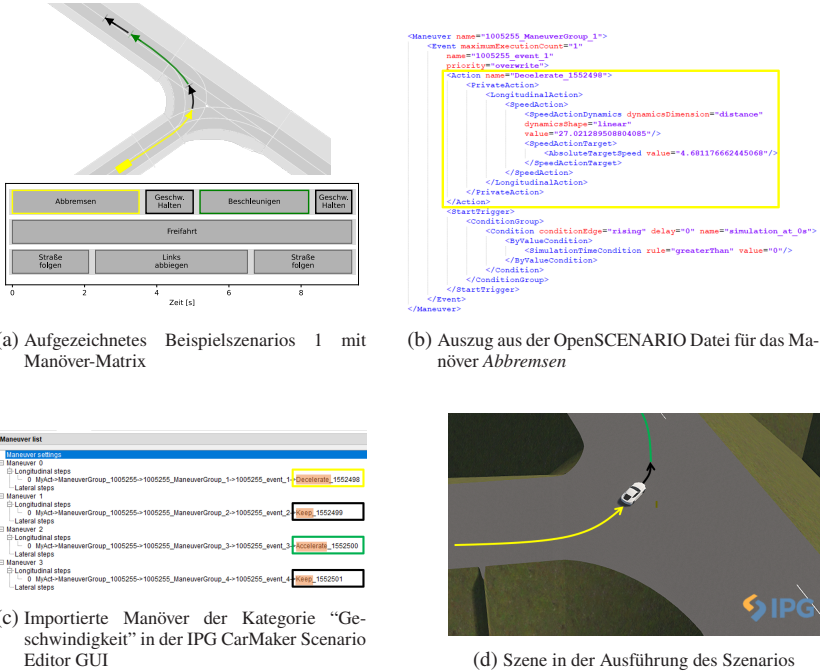
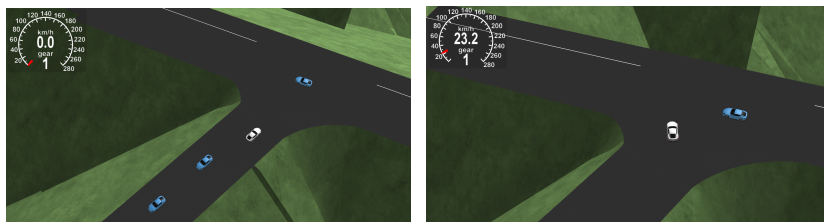


Abbildung 7.14: Darstellung der Artefakte zur Ausführung von Beispielszenario 1 am SiL

nell möglich. Vor allem Szenarien mit vielen Interaktionen zwischen den Teilnehmern sind aber schwer umzusetzen, weil die Gründe für die Manöverwechsel aus den Daten interpretiert werden müssen. Es ist z.B. nur aufgezeichnet, dass ein Teilnehmer ein Stop-Manöver durchführt, aber nicht immer eindeutig zu erkennen, warum er dies tut. In der Implementierung dieser Arbeit wird daher nur von einer teilautomatisierten Umsetzung gesprochen, die anschließend manuell überprüft und bei Bedarf angepasst werden muss. Aber selbst die Teilautomatisierung, also das Anlegen der Straße aus der OpenDRIVE-Karte, das Positionieren der Teilnehmer zu Beginn des Szenarios auf dieser sowie das Erstellen von initialen Manöverlisten und Übergangsbedingungen, die noch einmal manuell überprüft werden, führt zu einem Vorteil gegenüber einer komplett manuellen Erstellung. Dieses Thema wird in Abschnitt 7.3 und Abschnitt 8.2 weiter diskutiert.

### 7.1.5 Validierung der ausgeführten Szenarien

Für die Validierung der ausgeführten Szenarien werden die Manöver und Kontexte des Ausgangsszenarios mit dem ausgeführten Szenario miteinander verglichen. Dazu wird das entwickelte Ähnlichkeitsmaß verwendet. Für den PoC ist das Szenario nur valide, wenn alle Manöver und Kontexte gleich sind, die Distanz also 0 beträgt.



(a) Valide Variation: Ego-Fahrzeug (weiß) lässt alle Teilnehmer (blau) passieren (b) Invalide Variation: Ego-Fahrzeug (weiß) fährt vor Teilnehmer (blau) über die Kreuzung

Abbildung 7.15: Valide und invalide ausgeführte Variationen von Beispielszenario 4 erzeugt durch Parameteränderungen

Es werden Variationen des Beispielszenario 4 betrachtet (s. Abbildung 7.15), in dem das Ego-Fahrzeug und drei weitere Fahrzeuge über die Kreuzung fahren. In der validen Ausführung (s. Abbildung 7.15a) hält das Ego-Fahrzeug an und lässt alle anderen Teilnehmer passieren. Durch Parametervariationen der Geschwindigkeiten der Teilnehmer entstehen invalide Variationen des Szenarios:

In einer dieser Variationen fährt das Fahrzeug, das als letztes die Kreuzung passiert, so langsam, dass das Ego-Fahrzeug vor diesem die Kreuzung überquert (s. Abbildung 7.15b). Dieser wird somit nicht mehr als *einscherender Teilnehmer* modelliert, die Kontextbeschreibung ändert sich und das ausgeführte Szenario wird als invalide klassifiziert. Weitere invalide Variationen entstehen zum Beispiel, wenn das Ego-Fahrzeug vor allen Teilnehmern über die Kreuzung fährt. In diesem Fall ändern sich zusätzlich zu den fehlenden Kontexten auch die Manöver der Kategorie “Geschwindigkeit”, da kein *Anhalten* stattfindet. Vergleichbare Variationen entstehen auch bei Beispielszenario 2, wenn die Parameter für den kreuzenden Fußgänger variiert werden. Diese Klassifikation von diesen invaliden Szenarien ist notwendig, wenn dieses Szenario z. B. als Testszenario für einen Kreuzungsassistenten verwendet wird,

um zu überprüfen, ob die Vorfahrtregeln für einscherende Teilnehmer richtig eingehalten werden.

### 7.1.6 Laufzeit

Die Berechnung für den PoC wurde auf einem Desktop-PC durchgeführt und die einzelnen Prozessschritte haben dafür folgende Zeiten benötigt (s. Tabelle 7.5):

Prozessschritt	Laufzeit	Anzahl Szenarien
Manöver-Identifikation	46h	135.647
Szenarien-Selektion - Logische Szenarien	15h	135.647
Szenarien-Selektion - Ähnlichkeitsmatrix	2h 45m	9.555
Szenarien-Selektion - Visualisierung	34m	9.555
Szenarien-Selektion - Clustering	1m 42s	9.555
Szenarien-Selektion - Repräsentanten	10m 12s	9.555
Ausführen der Testfälle	16 m <sup>3</sup> / 50 m <sup>4</sup>	100
Evaluation der Testfälle	8 m <sup>5</sup>	100

Tabelle 7.5: Übersicht über die Laufzeit der Prozessschritte und die Anzahl der prozessierten Szenarien

Die komplette Durchführung bis zur Auswahl der Repräsentanten für 9 Monate Daten ist also auf einem Desktop-PC innerhalb von Tagen durchführbar. Größter Aufwand zu Beginn der Berechnung ist die Manöver-Identifikation. Da durch die Modellierung auf Manöverbeschreibung die Information kondensiert wird, profitieren die nachfolgenden Prozessschritte in Bezug auf die Laufzeit von der Reduktion zu Beginn des Prozesses. Dies ist auch am benötigten Speicherplatz zu sehen, die 96 GB Ausgangs-Dateien werden auf 17 GB für Szenario- und Manöver-Datenbank verringert.

<sup>3</sup> Für SiL, geschätzt unter der Annahme 10s/Szenario (s. Abschnitt 7.1.4)

<sup>4</sup> Für Ausführung in Echtzeit

<sup>5</sup> Für SiL mit 5s/Szenario

Für die Ausführung gehen die Zahlen für das Verwenden von SiL und Testplattformen, die nur in Echtzeit funktionieren (s. Abschnitt 2.2) stark auseinander: Während am SiL nur ca. 10 Minuten benötigt werden, werden für die reine Ausführung der 100 Szenarien in Echtzeit 50 Minuten gebraucht. Je nach Testplattform erhöht sich diese Zeit noch, da Aufbau und Initialisierung des Testszenarios hier nicht betrachtet sind. Das Validieren ausgeführter Testszenarien benötigt ca. 5 Sekunden pro Szenario und beinhaltet eine weitere Manöver-Identifikation für dieses inkl. Szenario-Extraktion und den Vergleich mit dem ursprünglichen Szenario.

Weitere Verbesserungen in Bezug auf die Laufzeit sind durch Multiprocessing möglich. Die Dateien sind parallel auf mehreren Rechnern verarbeitbar, und sie werden abschließend in einer zentralen Datenbank zusammengeführt.

## **7.2 Diskussion zur Abdeckung der Anforderungen**

### **7.2.1 Anforderungen**

#### **Anforderung 1: Vergleichbar**

Die entwickelte Manöverbeschreibung ermöglicht den Vergleich von Szenarien auf zwei Arten: Zum einen werden die Manöverfolgen zweier Szenarien direkt miteinander verglichen, um binär gleiche und ungleiche Szenarien zu identifizieren (s. Abschnitt 5.2.3). Einen detaillierteren, quantitativen Vergleich wird durch das entwickelte Ähnlichkeitsmaß (s. Gleichung 5.3) ermöglicht, das auch Werte zwischen 0 und 1 annimmt.

Die gewünschte Ähnlichkeit zwischen Szenarien variiert in Abhängigkeit vom Testobjekt sowie von der ODD. Die Ähnlichkeitsberechnung auf Basis von Manövern und Kontexten ermöglicht es, durch unterschiedliche Gewichtung der Kategorien auf diese Variationen einzugehen. Während z.B. für einen ACC auf der Autobahn die longitudinalen Manöver eine hohe Gewichtung erhalten, sind für einen urbanen Kreuzungsassistenten ebenfalls Manöver der Kategorie “Route” und Kontexte der Kategorie “Kreuzung” relevant.



Das entwickelte Ähnlichkeitsmaß wurde im PoC sowohl für die ähnlichkeitsorientierte Szenarien-Selektion (s. Abschnitt 7.1.3), als auch für die Szenario-Validierung (s. Abschnitt 7.1.5) erfolgreich angewendet.

### **Anforderung 2: Ausführbar**

Für eine Ausführung der Szenarien auf einer Testplattform werden die Manöver als Anweisungen an die Teilnehmer interpretiert. Da Manöver die etablierte Form der Beschreibung für ausführbare Szenarien sind (s. Abschnitt 3.1.5), fügt sich dies nahtlos in die manövergestützte Methodik dieser Arbeit ein. Eine technisch breite Abdeckung an Testplattformen wird durch die Konvertierung in den Standard OpenSCENARIO erreicht. Limitierungen dieses Ansatzes werden in Abschnitt 7.3 erläutert. Herausforderungen dabei sind die unterschiedlichen Manöverkataloge und die Beschreibung der Interaktion der Teilnehmer, die eine vollständig automatisierte Umwandlung erschwert.

Im PoC wird eine teilautomatisierte Übertragung gezeigt, mit der manueller Aufwand für die Erstellung eingespart wird (s. Abschnitt 7.1.4). Es existieren weitere Potenziale im Übertragungsalgorithmus, um den manuellen Aufwand in der Szenarioerstellung zu reduzieren. Die anschließende Ausführung auf den Testplattformen kann vollautomatisiert erfolgen, falls diese den OpenSCENARIO-Standard unterstützen.

### **Anforderung 3: Erweiterbar**

Die modellbasierte Szenariobeschreibung (s. Abbildung 5.19) definiert zuerst nur übergeordnete Elemente wie Szenario, Teilnehmer und Manöver. Die tatsächlichen Kategorien und Typen dieser Elemente werden erst in der Umsetzung (s. Abschnitt 6.5) definiert und können damit auch im Laufe des Absicherungsprozesses nachträglich erweitert werden, ohne die zugrunde liegende Struktur zu verändern.

Das Modell ist nach dem Open-Close-Prinzip erstellt: Es ist offen für Erweiterungen wie neue Kategorien und Typen für Manöver, Kontexte und Teilnehmertypen, aber geschlossen für Modifikationen. Diese Erweiterungen haben also weder einen Effekt auf bereits bestehende Modellteile, wie z.B. schon vorhandene Manövertypen, noch auf die übergeordneten Elemente im Metamodell

wie Szenerie, Teilnehmer, Manöver und Kontext. Technisch erreicht wird dies durch abstrakte Superklassen und deren Vererbung (s. Abbildung 5.19).

#### **Anforderung 4: Intuitiv**

Manöver haben ihren Ursprung in der natürlichen Sprache und sind damit für Menschen verständlich formuliert. Die auf Manövern aufbauende Szenariobeschreibung übernimmt diese Eigenschaft. Da zusätzlich auch eine Formalisierung gefordert ist (s. Anforderung 5), kann keine komplett freie natürliche Sprache verwendet werden. Die Kombination aus Manövern mit einem technischen Metamodell (s. Abbildung 5.19) ist damit ein Kompromiss zwischen Formalismen und intuitiver Nutzbarkeit.

Eine intuitive visuelle Darstellung dieser formalisierten Form erfolgt durch die Manöver-Matrix (Abschnitt A.2). Sind die Trajektorien bekannt, wird sie noch mit dem Top-View kombiniert, um einen Überblick über das Szenario zu geben (s. Abschnitt 7.1.2). Diese Darstellungsformen ermöglichen es auch nicht-technischen Nutzern, die Inhalte der Szenarien einfach zu erfassen.

#### **Anforderung 5: Formalisiert**

Die Szenariobeschreibung ist als eindeutig definiertes Modell (s. Abbildung 5.19) realisiert und somit formal beschrieben. Als Modellierungssprache ist ein UML-Klassendiagramm gewählt, das in etablierte Programmiersprachen übertragbar ist.

Während des PoC wird erfolgreich gezeigt, wie die Beschreibung aller automatisierten Prozessschritte algorithmisch verarbeitet wird.

#### **Anforderung 6: Parametrisierbar**

Die einzelnen Elemente der Szenariobeschreibung, wie Manöver, Kontexte und statische Elemente, besitzen jeweils Parameter, die ihre genauere Ausprägung beschreiben (s. Abschnitt 5.2.7). Darüber hinaus sind die Szenarien im Ganzen parametrisierbar. Die objektorientierte Modellierung und Vererbung der Elemente binden deren Parameter harmonisch in der Beschreibung ein.

Da logische Szenarien (s. Abschnitt 6.2.1) dieselben Elemente und damit auch Parameter besitzen, lassen sich für diese Verteilungen erstellen.

Im PoC sind die Parameterverteilungen beispielhaft für ein logisches Szenario gezeigt (s. Abbildung 7.10).

### **Anforderung 7: Kompakt**

Eines der drei wesentlichen Merkmale der Modellierung ist die Verkürzung (s. Abschnitt 3.1.1). Dadurch wird die Szenariobeschreibung im Modellierungsprozess (s. Abschnitt 5.2.7) auf ihre wesentlichen Eigenschaften und Zusammenhänge reduziert und eine kompakte Repräsentation erzeugt.

Auch die Parametrierbarkeit erhöht die Kompaktheit der Beschreibung: Unterschiedliche Ausprägungen eines Elements werden über einzelne Werte, in Form der Parameter, erzeugt und müssen nicht als ganzes neu beschrieben werden. Zusätzlich sorgt die Vererbungsstruktur der Elemente dafür, dass die Parameter nur einmal in der Elternklasse und nicht in allen abgeleiteten Typen definiert sind.

Im Gesamtprozess sorgt die Beschreibung als Manöver direkt zu Beginn für eine Informationsreduktion im Vergleich zur Beschreibung als Zeitreihen. Dies zeigt sich direkt im benötigten Speicherplatz im PoC (s. Abschnitt 7.1.6), der sich von 96 GB auf 17 GB reduziert.

### **Anforderung 8: Durchgängig**

Aufbauend auf Forschungsfrage 1 ist diese Anforderung Grundlage für den Entwurf der Beschreibungsform. In Kapitel 5 wird dargelegt, wie sie in den einzelnen Prozessschritten verwendet wird: Die datenbasierte Szenario-Erstellung nutzt die Manöver-Identifikation und Szenario-Extraktion, das Ableiten der Testfälle, die auf Manövern basierende ähnlichkeitsorientierte Szenarien-Selektion, das Ausführen der Testfälle, die teilautomatisierte Umwandlung in OpenSCENARIO, die Evaluation, die Szenario-Validierung anhand des Ähnlichkeitsmaßes und die Sicherheitsargumentation, die Abdeckung des Szenarienraums über Clustering und Szenario-Graph.

Im PoC wird anhand eines Beispieldatensatzes diese Durchgängigkeit erfolgreich nachgewiesen.

## 7.3 Limitierungen

Auch wenn die Anforderungen alle erfüllt sind, existieren einige Limitierungen im manövergestützten Testen:

Grundlegende konzeptionelle Limitierungen ergeben sich durch die Beschreibung des Verhaltens als Modell. Nur modellierte Manöver und Kontexte werden identifiziert und beschrieben. Sind relevante Elemente nicht enthalten, geht diese Information bei der Abstraktion verloren. Werden solche Fälle während des Absicherungsprozesses entdeckt, löst eine Erweiterung des Modells um das fehlende Element dieses spezifische Defizit. Da das urbane Fahren aber einen Open-World-Kontext darstellt (s. Abschnitt 1.1.3), ist die Vollständigkeit des Modells nicht nachzuweisen [119].

Auch die Abbildung der Welt auf genau definierte Manöver führt zu Schwierigkeiten: Ein Teilnehmer ändert seine laterale Position auf der Straße, um ein parkendes Auto zu umfahren. Wird dies als Spurwechsel modelliert oder ändert sich nur der Abstand zur Spurmitte im Spurhalten-Manöver? Für beide Varianten gibt es schlüssige Argumentationen mit Vor- und Nachteilen. Um das Modell zu erstellen, muss dann eine Entscheidung getroffen werden. In der praktischen Umsetzung tritt diese Situation vermehrt auf, wenn mehrere Personen mit unterschiedlichen Blickwinkeln an der Modellierung beteiligt sind.

Eine weitere Limitierung entsteht durch das Klassifizieren von diskreten Klassen aus kontinuierlichen Merkmalen, wie den Signalverläufen aufgezeichneter Szenarien. Dieser Effekt ist mit einer Quantifizierungsabweichung vergleichbar. Dies tritt z.B. bei der Definition bzw. Erkennung eines Folgemanövers auf, wenn folgende oder ähnliche Fragen beantwortet werden müssen: Ab welchem Abstand unterstellt man einem vorausfahrenden Teilnehmer Einfluss auf die eigene Geschwindigkeit? Auch bei Kreuzungsmanövern und Kontexten stellte sich u. a. die Frage: Ab wann beginnt eine Kreuzung bzw. ab wann hat sie Einfluss auf das Verhalten der Teilnehmer? Das Festlegen auf einen exakten Wert, z.B. 20 m, ist zumindest in dieser Genauigkeit nicht exakt herleitbar.

Szenarien mit 19,9 m werden dann einer anderen Klasse zugeordnet als solche mit 20,1 m, obwohl die Änderung zu 19,8 m genauso groß ist. Eine eigentlich vernachlässigbare Änderung von 10 cm führt also einmal zu einer Änderung des Manövers, und das andere Mal nicht.

Zu erwähnen ist auch, dass diese Quantifizierung mit ihren Problemen auch bei nicht-modellbasierten Methoden spätestens beim Clustern von Szenarien durchgeführt werden muss. Im manövergestützten Testen wird die Problematik also direkt zu Beginn des Prozesses und in Kombination mit der Modellierung angegangen und nicht auf spätere Prozessschritte verschoben.

Die bisher genannten Limitierungen ergeben sich durch die modellbasierte Betrachtung des Verhaltens der Teilnehmer und gelten für das gesamte manövergestützte Testen. Weitergehend finden sich auch in den einzelnen Prozessschritten Limitierungen: Die Manöver-Identifikation bildet den Startpunkt und damit die Grundlage für das datengetriebene Vorgehen. Fehlklassifikationen in diesem Schritt werden durch den gesamten Absicherungsprozess weitergegeben. Im PoC wurden einige dieser Fehlklassifikationen bereits gezeigt (s. Abschnitt A.3). Durch ausgereifere Klassifikationsmethoden, wie den Einsatz von neuronalen Netzen, ist die Klassifikationsrate weiter zu steigern. Eine komplett fehlerfreie Identifikation kann aber nicht sichergestellt werden.

Eine weitere Limitierung betrifft das Ähnlichkeitsmaß und alle damit verbundenen Schritte, aber vor allem das Finden von Ausreißern beim Erstellen des Szenario-Katalogs. Es wird ausschließlich das modellierte Verhalten berücksichtigt. Verhaltensweisen, die nicht in die vordefinierten Manöverkategorien klassifiziert werden, werden dabei ignoriert. Dadurch fließen potenziell relevante oder ungewöhnliche Verhaltensweisen, die außerhalb der modellierten Manöver liegen, nicht adäquat in die Analyse ein. Dies schränkt die Aussagekraft der Methode in Bezug auf die Erfassung der gesamten Bandbreite möglicher Szenarien ein. Abhilfe schafft hier die Kombination mit einem zusätzlichen Ähnlichkeitsmaß, das ohne Abstraktion direkt auf den Daten rechnet und so diese Ausreißer findet.

Im Szenario-Graphen können aufgrund der Reduktion auf zwei Dimensionen die exakten Abstände zwischen den Szenarien nur näherungsweise abgebildet werden. Dies bedeutet, dass die in der Darstellung gezeigten Distanzen lediglich Annäherungen an die tatsächlichen Beziehungen sind und somit nicht exakt interpretiert werden dürfen. Dies wird durch die Kombination mit dem Clustering sichtbar, wenn die Cluster ineinander übergehen. Das Erhöhen von

zwei auf drei Dimensionen verringert diesen Effekt theoretisch, inwieweit die Graphen aber dadurch insgesamt besser werden, müsste weiter untersucht werden. Die Übersichtlichkeit der Visualisierung als Graph sinkt mit der Menge der Szenarien, die zuvor beschriebenen Probleme bei der Abbildung nehmen zu. Abhilfe schaffen hier Filter bzw. das Unterteilen in mehrere Visualisierungen (z.B. nur Szenarien mit Rechts-Abbiege-Manövern in einer Abbildung).

Eine weitere Limitierung betrifft die technische Realisierung zur teilautomatisierten Erstellung von OpenSCENARIO-Dateien. Konzeptionell ist es möglich, die in dieser Dissertation entwickelte Beschreibungsform direkt an Testplattformen zur Ausführung zu übergeben. Dafür müsste aber jede Testplattform einen eigenen Importer oder Konverter implementieren. Der Weg über einen bestehenden Standard ermöglicht es, eine breite Menge an Plattformen mit einem einzigen Konverter zu schreiben. Im PoC wurde gezeigt, dass dies prinzipiell möglich ist, weitere Verbesserungen, um den Grad der Automatisierung zu erhöhen und die Effizienz weiter zu steigern, sind noch erstrebenswert.

Zusammenfassend lässt sich sagen, dass Limitierungen im manöverbasierten Testen existieren, das Vorgehen im Ganzen aber nicht blockiert wird. Einige Limitierungen lassen sich durch Weiterentwicklung verringern oder beheben (Manöver-Identifikation, automatisierte Ausführung, Übersichtlichkeit des Clusterings), oder durch gezielten Einsatz von ergänzenden Methoden (Kombination mit nicht-manöverbasierten Ähnlichkeitsmaßen) abfangen. Andere davon liegen in der grundlegenden Methodik der Modellierung, ein Zitat von George Box fasst diese sehr passend zusammen: “Im Prinzip sind alle Modelle falsch, aber manche sind nützlich.”<sup>6</sup> Auf das manövergestützte Testen lässt sich dies so übertragen, dass zwar nicht alle Verhaltensweisen exakt abgebildet werden, das Verfahren aber trotzdem dazu beiträgt, das Ziel der Absicherung von HAF zu erreichen.

---

<sup>6</sup> [https://de.wikipedia.org/wiki/George\\_Box](https://de.wikipedia.org/wiki/George_Box)

## 7.4 Diskussion der Forschungsfragen

### 7.4.1 Forschungsfrage 1: Durchgängige Beschreibungsform

Diese Forschungsfrage befasst sich mit der fehlenden Harmonisierung der Szenariobeschreibung im Rahmen des Absicherungsprozesses. Um herauszufinden, was eine Szenariobeschreibungsform leisten muss, um durchgängig anwendbar zu sein, wurden acht Anforderungen erstellt. Wie in Abschnitt 5.1.2 dargestellt, wurde eine Beschreibung durch Manöver gewählt, um diese Anforderungen zu erfüllen.

Zwei Gründe haben überwiegend zu den Insellösungen in den Szenariobeschreibungen geführt: Zum einen der Konflikt zwischen intuitiver Nutzbarkeit für menschliche Anwender und Formalismus für automatisierte Anwendungen, zum anderen die Trennung zwischen vergleichenden Prozessschritten, wie dem Auswählen der Testszenarien, und ausführenden Prozessschritten, wie dem Testen des Testobjekts.

Die Brücke zwischen intuitiver Nutzbarkeit und automatisierter Verarbeitung wird geschlagen, indem die Manöver und Kontexte aus natürlicher Sprache abstammen, deren Zusammenhänge und Ausprägungen aber zusätzlich in einem Metamodell formal definiert sind (s. Abschnitt 5.1.2).

Für die Trennung zwischen vergleichenden und ausführenden Anwendungen bildet die manöverbasierte Ähnlichkeitsberechnung (s. Abschnitt 5.2.3) den zentralen innovativen Teil dieser Arbeit. Für die Ausführung waren Manöver bereits die etablierte Lösung, durch das entwickelte Ähnlichkeitsmaß werden sie nun auch verwendet, um Szenarien zu clustern und repräsentative Szenarien auszuwählen.

Die entworfene Form zur Beschreibung von Szenarien erfüllt die aus einem ganzheitlichen Einsatz abgeleiteten Anforderungen, wie in Abschnitt 7.2 dargestellt. Somit wurde nachgewiesen, dass ihr durchgängiger Einsatz möglich ist.

## **7.4.2 Forschungsfrage 2: Herausforderungen im szenariobasierten Testen**

Diese Forschungsfrage befasst sich mit den zentralen Herausforderungen des szenariobasierten Testen. Zusätzlich zur durchgängigen Verwendbarkeit aus Forschungsfrage 1 wird dabei untersucht bzw. sichergestellt, dass der Absicherungsprozess zielführend durchgeführt wird. Auf Basis der Szenariobeschreibungsform wurde dazu das Vorgehensmodell des manöverbasierten Testens erstellt. Ebenso wurden die Anforderungen den einzelnen Herausforderungen zugeordnet (s. Abbildung 4.1).

### **Herausforderung 1: Exploration des Szenario Raums**

Um die Exploration erfolgreich zu bewältigen, müssen Anforderung 3, erweiterbar, und Anforderung 1, vergleichbar, erfüllt sein. In Abschnitt 7.2 wurde erläutert, wie das manövergestützte Testen diese Anforderungen erfüllt.

Neue Szenarien werden dann klassifiziert, wenn die Ähnlichkeit zu allen bisher bekannten Szenarien unter einem Schwellenwert liegt. So wird der Raum an Szenarien iterativ erweitert. Finden sich dabei relevante, aber bisher unbekannte oder nicht modellierte Elemente des Szenarios, wird das Modell der Szenariobeschreibung entsprechend erweitert.

### **Herausforderung 2: Abschätzung der Häufigkeit**

Die einzige Anforderung für das Abschätzen der Häufigkeiten ist Anforderung 1, die Vergleichbarkeit. Existierende Methoden zur Abschätzung sind unabhängig von der zugrunde liegenden Ähnlichkeitsberechnung. Ob diese über Manöver, andere Abstraktionsmodelle, direkt auf den Daten oder über KPIs berechnet werden, ist nicht entscheidend. Da die Anforderung der Vergleichbarkeit durch das entwickelte Ähnlichkeitsmaß erfüllt ist, kann auch eine Abschätzung zur Häufigkeit der Szenarien durchgeführt werden. Vorgehen zu dieser Thematik sind z.B. in [71, 119] enthalten und auf manöverbasierten Ähnlichkeitsmaße übertragbar.



Eine Limitierung für die rein manöverbasierte Abschätzung ist, dass nur verglichen werden kann, was auch im Modell abgebildet ist (s. Abschnitt 7.3). Da ein Modell auch immer eine (gewollte) Reduktion darstellt, gibt es seltene Verhaltensweisen, die nicht durch einen dafür definierten Manövertyp dargestellt werden. Um dieses Problem zu lösen, lässt sich das manöverbasierte Ähnlichkeitsmaß mit anderen, z.B. trajektorienbasierten Maßen verknüpfen. Nur wenn beide Maße das zu untersuchende Szenario als bereits bekannt klassifizieren, wird dies auch als bekannt betrachtet [TB1].

### **Herausforderung 3: Nachweis der Sicherheit**

Für den Nachweis der Sicherheit in einem Szenario ist Anforderung 2, die Ausführbarkeit, die einzige Anforderung an die Beschreibungsform. Diese ist konzeptionell durch die manöverbasierte Beschreibung erfüllt, da Manöver bereits die etablierte Form sind, um die Anweisungen an die Teilnehmer im Szenario zu geben (s. Abschnitt 7.2.1).

Der zweite zentrale Punkt beim Nachweis der Sicherheit sind die Anforderungen an das Testobjekt, die aber in dieser Dissertation nicht betrachtet werden (s. Abschnitt 5.1.1). Diese definieren, ob das Testobjekt das Szenario erfolgreich durchgeführt hat.

### **Herausforderung 4: Effizienz des Testprozesses**

Den größten Anteil zur Effizienzsteigerung hat die bereits in Abschnitt 7.4.1 diskutierte Anforderung 8, die Durchgängigkeit. Sie realisiert ein nahtloses Zusammenspiel der einzelnen Prozessschritte.

Durch Anforderung 5, Formalismen, wird die Effizienz weiter erhöht. Sie ist Grundvoraussetzung für alle automatisierten Vorgänge und ermöglicht eine standardisierte und einheitliche Absicherung.

Ein vergleichbarer Effekt ergibt sich aus Anforderung 1, die die Auswahl repräsentativer Testszenarien erlaubt. Dadurch kann mit einer minimalen Anzahl ausgeführter Testszenarien eine möglichst hohe Testabdeckung erzielt werden. Besonders bei Testplattformen mit nicht-simulativen Anteilen, wie reale Erprobungsfahrten oder HiL-Systemen, führt dies zu Kosten- und Zeitersparnissen.

Anforderung 7, Kompaktheit, ist ein weiterer Faktor zur Steigerung der Effizienz. Im manövergestützten Testen führt die Reduktion der Information direkt zu Beginn des Prozesses dazu, dass alle Schritte auf der kompakten Form arbeiten. Vor allem die vergleichenden Prozessschritte zu Beginn des Vorgehensmodells, deren Laufzeit nichtlinear mit der Anzahl der Szenarien ansteigt, profitieren so von einer beschleunigten Berechnung. So ist es möglich, dass z. B. der Beispieldatensatz mit neun Monaten aufgezeichneter Daten auf einem Desktop-PC innerhalb von einigen Tagen verarbeitet wird (s. Abschnitt 7.1.6).

## 8 Zusammenfassung und Ausblick

### 8.1 Zusammenfassung

Automatisierte Fahrfunktionen besitzen das Potenzial, den Verkehr grundlegend zu verändern, indem sie die Verkehrssicherheit verbessern, den Komfort steigern, neue Mobilitätskonzepte ermöglichen und wirtschaftliche Vorteile im Gütertransport bieten. Ein zentraler Punkt, warum dennoch weiterhin Menschen am Steuer sitzen, ist der fehlende Nachweis der Sicherheit für automatisierte Fahrsysteme. Ein signifikantes Defizit in der aktuellen Umsetzung des Absicherungsprozesses ist die fehlende Harmonisierung im Bereich der Szenariobeschreibung, wodurch die Effizienz der gesamten Methode verringert wird.

Um dieses Thema zu untersuchen, wurden in dieser Dissertation vier zentrale Herausforderungen des szenariobasierten Testens herausgearbeitet: die Exploration des Szenarienraums, das Abschätzen der Häufigkeit unbekannter Szenarien, der Nachweis der Sicherheit des Testobjekts im Szenario und die Effizienz des Testprozesses. Auf diesen aufbauend wurden acht Anforderungen an Szenariobeschreibungsformen hergeleitet. Die Anforderungen wurden sowohl den zentralen Herausforderungen des szenariobasierten Testens als auch dessen einzelnen Prozessschritten zugeordnet.

Als Konzept wurden die modellbasierte Beschreibungsform und das zugehörige Vorgehensmodell, das manövergestützte Testen, entworfen. Der Kern ist eine formalisierte Beschreibung der Szenarien durch Manöver, welche durchgängig in den einzelnen Prozessschritten verwendet wird, um bestehende, auf einzelne Aufgaben zugeschnittene Insellösungen zu ersetzen. Manöver wurden ausgewählt, da sie bereits in unterschiedlichen Typen von Szenarien erfolgreich eingesetzt werden, um das Verhalten der Verkehrsteilnehmer zu beschreiben. Ein weiterer zentraler innovativer Anteil ist die manöverbasierte Ähnlichkeitsberechnung, durch die Szenarien quantitativ miteinander vergleichbar werden.

Durch diese ist es nun möglich, Manöver auch für vergleichende Prozessschritte, wie den Aufbau des Szenarienkatalogs und die Auswahl der Testszenarien, zu verwenden.

Weitergehend wurde eine konkrete Realisierung vorgestellt. Dazu wurden Kategorien für Manöver und Kontexte im urbanen Raum erstellt, Regeln für deren Identifikation ausgearbeitet, die Parametrisierung für das Ähnlichkeitsmaß durchgeführt, passende Implementierungen für Clustering und Einbettung ausgewählt und eine teilautomatisierte Erstellung von ausführbaren OpenSCENARIO-Dateien entwickelt.

Ein Proof-of-Concept wurde mit einem repräsentativen Datensatz von 135.647 Szenarien an einer urbanen Kreuzung durchgeführt. Dabei wurde die Beschreibungsform in allen Prozessschritten des manövergestützten Testens durchgängig eingesetzt. Anhand der Ergebnisse wurden die zuvor erstellten Anforderungen diskutiert und dabei erläutert, wie die Abstraktion des Verhaltens zu Manövern diese Anforderungen erfüllt.

Weitergehend wurden Limitierungen erörtert, die sich in zwei Kategorien aufteilen: Zum einen entstehen diese durch die Informationsreduktion der Modellierung, wodurch vor allem seltene Verhaltensweisen nicht exakt abgebildet werden. Zum anderen gibt es Limitierungen, die spezifisch aus den Methoden im Vorgehensmodell bzw. aus der konkreten Realisierung hervorgehen, z. B. Fehler in der datenbasierten Manöverklassifikation.

Zusammenfassend bildet die folgende Erkenntnis den wissenschaftlichen Beitrag dieser Dissertation: Eine Beschreibung von Szenarien durch Abstraktion zu Manövern kann einheitlich und durchgängig im Testprozess eingesetzt werden. Sie ersetzt bestehende Insellösungen und erhöht so signifikant die Effizienz des Absicherungsprozesses.

Die Dissertation erzeugt so einen wertvollen Beitrag zur Weiterentwicklung von Testmethoden für Automatisierte Fahrfunktionen. Der manöverbasierte Ansatz zeigt das Potenzial, die bestehenden Herausforderungen im Bereich Verifikation und Validierung zu bewältigen und eine effizientere Entwicklung sicherheitskritischer Systeme zu ermöglichen.

## 8.2 Ausblick

### Maschinelles Lernen zur Manöver-Identifikation

Der Einsatz von maschinellem Lernen hat das Potenzial die Identifikation von Manövern weiter zu optimieren. Dabei bieten sich sowohl überwachte als auch semi-überwachte Lernmethoden an, um die Klassifikationsgenauigkeit zu erhöhen. Ausgangspunkt dafür ist ein ausreichend großer Datensatz, in dem die zu erkennenden Manövertypen gelabelt sind.

### Kombination mit anderen Ähnlichkeitsmaßen

Die Integration von manöverbasierter Ähnlichkeitsberechnung mit zusätzlichen Metriken, die zum Beispiel direkt mit Trajektoriendaten rechnen, kann die Schwächen der modellbasierten Ähnlichkeitsberechnung ausgleichen und so einen vollständigeren Szenarienkatalog erstellen. Erste Konzepte und Ergebnisse dazu wurden bereits in [TB1] veröffentlicht.

### Nutzung von Manöverparametern

Eine weitere Verfeinerung der Ähnlichkeitsberechnung kann durch die Berücksichtigung spezifischer Parameter der Manöver erfolgen, wie Geschwindigkeit oder Dauer. Dies erhöht die Genauigkeit beim Vergleich der Szenarien und gibt weitere Möglichkeiten, die Ähnlichkeitsberechnung an die ODD und das Testobjekt anzupassen.

### Ähnlichkeitsberechnung für alle Verkehrsteilnehmer

In dieser Arbeit wird die Ähnlichkeit zwischen Szenarien nur durch das Ego-Objekt bestimmt, der Einfluss anderer Teilnehmer wird über Kontexte abgebildet. Als Erweiterung können alle beteiligten Verkehrsteilnehmer betrachtet werden, um eine umfassende Analyse zu ermöglichen. Eine Schwierigkeit in diesem Ansatz besteht dann darin, die für das Szenario wirklich relevanten Teilnehmer bzw. Manöver zu bestimmen.

### Optimierungsalgorithmen für Testszenarien

Der Einsatz von Optimierungsalgorithmen, wie genetischen Algorithmen, zur Auswahl von Testszenarien könnte die Effizienz der Szenario-Selektion steigern und die Testabdeckung weiter verbessern. Erste Konzepte und Ergebnisse dazu finden sich in [S7].

### Erweiterung des Automatisierungsgrads zur OpenSCENARIO-Erstellung

Es ist möglich, den Automatisierungsgrad bei der Erstellung von OpenSCENARIO-

Dateien zu erhöhen, um die Generierung von Szenarien mit weniger manuellem Aufwand zu verbessern. Somit kann der Prozess noch kosten- und zeiteffizienter gestaltet werden.

Erweiterungen über die Themen dieser Arbeit hinaus

Themen wie Sicherheitsargumentation und die Verknüpfung mit spezifischen Anforderungen an das Testobjekt wurden in der vorliegenden Arbeit nicht vertieft behandelt. Diese Aspekte könnten jedoch in zukünftigen Arbeiten genauer untersucht werden, um eine breitere Basis für die Validierung automatisierter Fahrfunktionen zu schaffen.

Einsatz von Large Language Models (LLMs)

Der aktuelle Fortschritt bei LLMs kann im Rahmen des manövergestützten Testens vielfältig genutzt werden, beispielsweise um Szenarien in natürlicher Sprache zu spezifizieren (“Schreibe ein Szenario, das...”), zusammenzufassen (“Erkläre den Ablauf folgendes Szenarios in drei Sätzen...”) oder zu variieren (“Erstelle mir 100 Variation des Szenarios mit unterschiedlichen Parametern...”). Da die manöverbasierte Szenariobeschreibung auf Elementen natürlicher Sprache aufbaut, sind LLMs in der Lage, nahtlos mit ihr zu arbeiten.

# A Anhang

## A.1 Algorithmische Grundlagen

### A.1.1 Sequenzalignment

Sequenzalignment wurde ursprünglich in der Bioinformatik entwickelt, um die Ähnlichkeit zwischen biologischen Sequenzen wie DNA, RNA oder Proteinen zu analysieren. Mittlerweile findet es aber auch in anderen Bereichen Anwendung, um gemeinsame Muster und strukturelle Eigenschaften in Sequenzen zu identifizieren. Ein wesentliches Merkmal des Sequenzalignments ist, dass die Reihenfolge der beiden Sequenzen erhalten bleibt. Auch der Vergleich von Sequenzen unterschiedlicher Länge ist somit möglich.

Ein Alignment ist als eine Reihe von Editierschritten definiert, die eine Transformation zwischen zwei Sequenzen beschreiben. Mögliche Editierschritte sind Einfügen, Löschen und Ersetzen eines Wertes in einer Sequenz.

Um die Qualität eines so erzeugten Alignment zu bestimmen, werden folgende Elemente verwendet (s. Abbildung A.1).

- *Match*: Positionen in den Sequenzen, an denen die gleichen Elemente auftreten.
- *Mismatch*: Positionen in den Sequenzen, an denen unterschiedliche Elemente auftreten.
- *Gap*: Einfügungen von Lücken in einer oder beiden Sequenzen, um optimale Übereinstimmungen zu erreichen.

Die Identifikation und Bewertung dieser Elemente helfen dabei, die Güte des Alignments zu bewerten und somit die Ähnlichkeit der Sequenzen zu beschreiben.

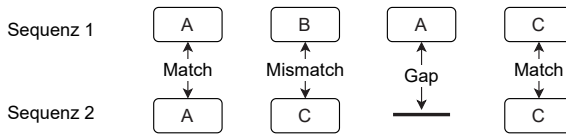


Abbildung A.1: Beispiel für ein Alignment zwischen zwei Sequenzen

Es existieren verschiedene Algorithmen, um optimale Alignments zu finden. Der am weitesten verbreitete ist der Needleman-Wunsch-Algorithmus [176]. Er basiert auf dynamischer Programmierung und findet so den optimalen Pfad durch eine zugehörige Kostenmatrix. Die Kostenfunktionen sind dabei frei konfigurierbar, d. h. es sind unterschiedliche Kosten für Gap, Mismatch und Match als Parameter wählbar. Somit werden unterschiedliche Aspekte der Sequenzähnlichkeit betont und priorisiert, was die Anpassbarkeit des Algorithmus an verschiedene Anwendungsgebiete erhöht.

## A.1.2 k-Medoids-Clustering

k-Medoids-Clustering ist ein partitionierendes Verfahren zur Gruppierung von Datenpunkten basierend auf ihrer Ähnlichkeit. Es basiert auf dem Prinzip, jedes Cluster durch einen repräsentativen Punkt, dem Medoid, zu beschreiben.

Zu Beginn des Algorithmus wird eine feste Anzahl  $k$  von Medoids zufällig aus den Datenpunkten ausgewählt. Jeder weitere Datenpunkt wird dem Medoid zugeordnet, zu dem er die geringste Distanz aufweist. Anschließend wird iterativ die Auswahl der Medoids verbessert: Für jeden Medoid wird geprüft, ob ein Tausch mit einem nicht ausgewählten Punkt zu einer Verbesserung des Ergebnisses führt.

Ein zentraler Vorteil von k-Medoids ist, dass das Verfahren nicht anfällig gegenüber Ausreißern ist. Da nur reale Datenpunkte als Repräsentanten verwendet werden, sind die Clustermitten nicht durch Extremwerte verzerrt. Zudem erlaubt k-Medoids die Nutzung nicht-euklidischer Distanz- oder Ähnlichkeitsmaße, wodurch es flexibel in unterschiedlichen Anwendungsbereichen eingesetzt werden kann.



### A.1.3 Visualisierung und Einbettung von Graphen

Die Visualisierung von Graphen dient dazu, die Struktur und Eigenschaften dieser intuitiv zu erfassen und Muster zu erkennen. Dazu wird eine Projektion der Datenstruktur aus Knoten und Kanten in eine zweidimensionale oder dreidimensionale Ebene durchgeführt, mit dem Ziel eng verbundene Knoten nahe beieinander abzubilden. Die Position der Knoten in dieser Visualisierung wird dann als Einbettung bezeichnet.

**Definition A.1.** Eine *Einbettung eines Graphen* beschreibt dessen Projektion in einen 2- oder 3-dimensionalen Raum. Dabei werden die wesentlichen Eigenschaften des Graphen, wie z. B. lokale und globale Beziehungen der Knoten, so gut wie möglich erhalten.

Mathematisch lässt sich die Einbettung für einen Graphen  $G = (V, E)$  mit

- $V$ : Menge der Knoten des Graphen
- $E$ : Menge der Kanten des Graphen
- $d$ : Dimension des Einbettungsraumes

als eine Funktion  $f$  beschreiben,

$$f : V \rightarrow \mathbb{R}^d \tag{A.1}$$

die jedem Knoten  $v \in V$  einen Vektor  $x_v \in \mathbb{R}^d$  zuordnet.

Visualisierungsmethoden für Graphen finden Anwendung in verschiedenen Bereichen, wie z.B. der Social-Media-Analyse und der Literaturrecherche (s. Abbildung A.2).

Je nach Anwendungsbereich, Ziel und Struktur der Daten, wird die Einbettung durch verschiedene Projektionsmethoden erstellt (s. Tabelle A.1): *Kraftbasierte Algorithmen* basieren darauf, den Graphen als physikalisches System von Punkten zu modellieren, die Kräfte aufeinander ausüben. Die Kräfte hängen von den Kanten des Graphen ab: Knoten, die viele Verbindungen miteinander haben, ziehen sich an, andere stoßen sich ab. So wird eine geeignete Positionierung der Knoten gesucht, um diese im Raum so anzuordnen, dass verwandte

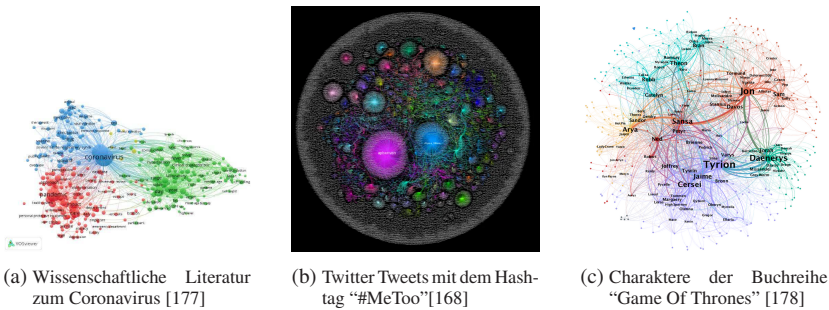


Abbildung A.2: Beispiele für die Verwendung von Graphenvisualisierung in unterschiedlichen Domänen.

Knoten näher zusammen und weniger verwandte weiter auseinander liegen. Eine umfassende Gegenüberstellung existierender kraftbasierter Methoden findet sich in Kobourov et al. [179].

Die Methoden zur *Dimensionsreduktion* dagegen minimieren die Abstände zwischen Knoten im hochdimensionalen Raum und projizieren sie dann in einen niedrigerdimensionalen Raum. Während der Projektion wird so optimiert, dass die Abstände der Knoten so genau wie möglich beibehalten werden. Gansner et al. [180] und Klimenta [181] zeigen, wie die Dimensionsreduktion als Anwendung für die Einbettung von (vollständig verbundenen) Graphen nutzbar ist. Häufig verwendete Algorithmen dieser Kategorie sind Multidimensionale Skalierung (MDS), Principal Component Analysis (PCA) und UMAP.

	Dimensionsreduktion	Kraftbasierte Layouts
Grundprinzip	Mathematische Optimierung	Physikalische Analogie
Visualisierungsziel	Präzise Darstellung der Struktur und Abstände	Intuitive, visuell ansprechende und explorative Darstellung
Globale vs. lokale Optimierung	Globale Optimierung	Lokale Optimierung
Menge der Knoten	Moderate bis große Mengen (bis 100.000)	Geeignet für kleine bis mittelgroße Mengen (bis 1000)

Tabelle A.1: Vergleich von Dimensionsreduktion und kraftbasierten Methoden zur Grapheneinbettung

Während das Visualisierungsziel bei der Dimensionsreduktion eine möglichst präzise Darstellung der tatsächlichen Distanzen ist, liegt der Fokus bei kraftbasierten Layouts auf einer visuell ansprechenden, leicht interpretierbaren Darstellung.

Die Methode der Dimensionsreduktion strebt eine globale Optimierung an, indem sie die Abstände zwischen allen Punkten im Datensatz optimiert. Im Gegensatz dazu führen kraftbasierte Layouts eine iterative Optimierung durch. Dieses Vorgehen kann zu einem Verharren in lokalen Minima führen und gewährleistet somit nicht das Finden des globalen Optimums.

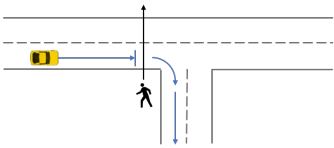
Aufgrund des Rechenaufwands sind kraftbasierte Layouts nur für kleine bis mittelgroße Mengen an Knoten geeignet [182], Dimensionsreduktionsmethoden können dagegen innerhalb weniger Sekunden eine Einbettung von bis zu 100.000 Knoten berechnen [182].

## A.2 Manöver-Matrix und Beispielszenarien

Die Manöver-Matrix dient zur grafischen, intuitiven Darstellung der ausgeführten Manöver eines Teilnehmers in einem Szenario. Jede Spalte bildet eine Manöver- bzw. Kontextkategorie ab. Der zeitliche Verlauf der Manöver bzw. Kontexte wird innerhalb der Spalte angezeigt. Die Zeitleiste verläuft von links nach rechts. Während für Manöver nach deren Definition immer genau ein Manöver pro Kategorie und Zeitpunkt aktiv ist, kann eine Kontext-Spalte auch zeitweise oder komplett leer sein.

Abbildung A.3a zeigt schematisch den Ablauf eines Kreuzungsszenarios mit einem kreuzenden Fußgänger. In Abbildung A.3b ist die zugehörige Manöver-Matrix für den PKW abgebildet. Zu Beginn des Szenarios befindet sich der PKW in einer Fahrt mit konstanter Geschwindigkeit. Sobald der Fußgänger die Straße kreuzt, beginnt ein Anhalte-Manöver und mit einem gleichzeitigen Kontext für einen kreuzenden Teilnehmer. Sobald die Geschwindigkeit des PKW auf null gesunken ist, wird aus dem Anhalte-Manöver ein Stillstand-Manöver. Dieser dauert an, bis der Fußgänger die Straße überquert hat und der zugehörige Kontext inaktiv ist. Sobald die Straße frei ist, beginnt das Beschleunigen-Manöver mit gleichzeitigem Rechts-Abbiege-Manöver über die Kreuzung. Ist die Kreuzung überquert, befindet sich der PKW wieder im

Manöver Straße folgen. Da keine Spurwechsel stattfinden, ist Spur halten das einzige Manöver in der Kategorie “Spur”. Es gibt keinen Relation-Kontext. Zwar befindet sich der Fußgänger zu einer bestimmten Zeitspanne in derselben Spur wie der PKW, allerdings bewegt er sich nicht in die gleiche Richtung und ist damit kein vorausfahrender Teilnehmer.

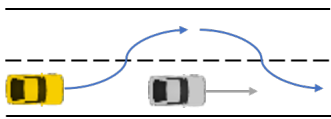


(a) Beispielszenario mit einem kreuzenden Fußgänger

Geschwindigkeit	Konstant	Anhalten	Stillstand	Beschleunigen
Folgen	Freifahrt			
Spur	Spur halten			
Route	Straße folgen		Rechts abbiegen	Straße folgen
Kreuzung	-	Kreuzender Teilnehmer		-
Relation	-			

(b) Manöver-Matrix für das Beispielszenario aus Abbildung A.3a

Abbildung A.4a zeigt schematisch den Ablauf eines Doppelspurwechsel-Szenarios. In Abbildung A.4b ist die zugehörige Manöver-Matrix für den gelben PKW abgebildet. Zu Beginn des Szenarios befindet sich der gelbe PKW (Akteur) hinter dem grauen PKW und hat eine höhere Geschwindigkeit als dieser. Somit ist der Kontext vorausfahrender Teilnehmer aktiv. Gleichzeitig gleicht der Akteur seine Geschwindigkeit an die des anderen PKW an, was durch die Manöver Abbremsen und Annähern beschrieben ist. Sobald die Geschwindigkeit angepasst ist, wechseln diese Manöver in konstante Geschwindigkeit und Folgefahrt. Um den grauen PKW zu überholen, beginnt der Akteur einen Spurwechsel. Sobald er auf der anderen Spur ist, beschleunigt er. Da sich kein vorausfahrendes Fahrzeug auf diese Spur befindet, wechselt das Manöver von Folgefahrt in Freifahrt und der Relation-Kontext wird inaktiv. Während des Überholvorgangs wird der graue PKW zu einem Kontext des Typs rechter Teilnehmer. Dieser Kontext wird nach dem Überholen inaktiv, und der Akteur beginnt einen weiteren Spurwechsel, um wieder auf die ursprüngliche Spur zurückzukehren. Da keine Kreuzung im Szenario vorhanden ist, ist Straße folgen das einzige Manöver der Kategorie “Route” und es gibt keinen Kreuzungs-Kontext.



(a) Beispielszenario mit einem kreuzenden Fußgänger

Geschwindigkeit	Abbremsen	Konstant	Beschleunigen	Konstant	
Folgen	Annähern	Folgefahrt	Freifahrt		
Spur	Spur halten	Spurwechsel	Spur halten	Spurwechsel	Spur halten
Route	Straße folgen				
Kreuzung	-				
Relation	Vorausfahrender Teilnehmer	-	Rechter Teilnehmer	-	

(b) Manöver-Matrix für das Beispielszenario aus Abbildung A.4a

### A.3 Beispielszenario zur Manöver-Identifikation

Der Ablauf des Beispielszenarios für ein Annähern mit anschließender Folgefahrt über die Kreuzung wird über vier Szenen veranschaulicht. (s. Abbildung A.5a – Abbildung A.5d). Das Ego-Fahrzeug startet mit einem Annähern als longitudinales und Abbremsen als Geschwindigkeitsmanöver (s. Abbildung 7.3). Sobald die Geschwindigkeit an das vorausfahrende Fahrzeug angepasst ist, wird die Folgefahrt über die Kreuzung durchgeführt. Durch das Verwenden der Karteninformationen wird das Folgen auch auf der ungeraden Fahrbahn durchgängig erkannt, was durch reine relative Positionen nicht möglich ist.

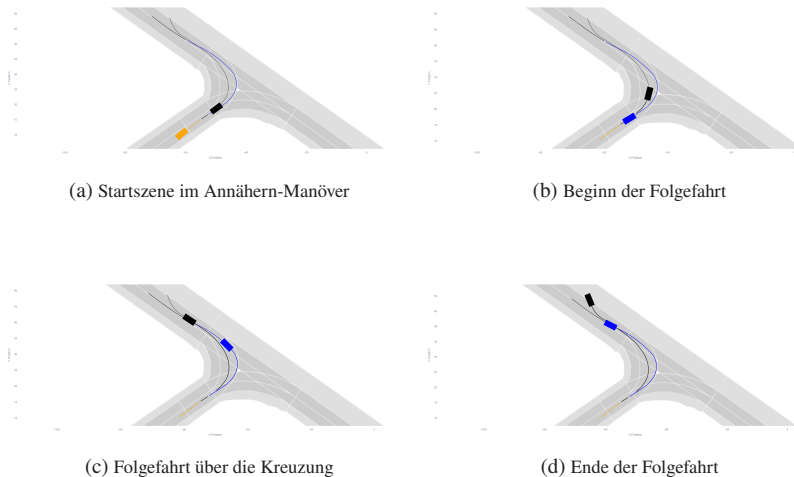
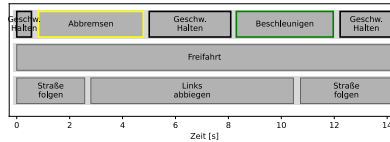


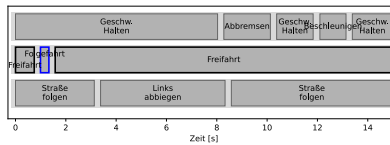
Abbildung A.5: Beispielszenario mit Ego-Fahrzeug und vorausfahrendem Teilnehmer an einer Kreuzung. Die Farben zeigen die longitudinalen Manöver des Fahrzeugs: Annähern (orange), Folgefahrt (blau) und Freifahrt (schwarz).

Weitergehend ist ein Freifahrtmanöver zwischen Annähern und Folgefahrt-Manövern erkannt, welches zwar durch den Signalverlauf plausibel ist, aber das Szenario nicht korrekt semantisch beschreibt. Dies kann in Zukunft durch eine bessere Manövererkennung gelöst werden (s. Abschnitt 8.2). Der Einfluss solcher Fehlklassifikationen wird im Kapitel Limitierungen (s. Abschnitt 7.3) besprochen.

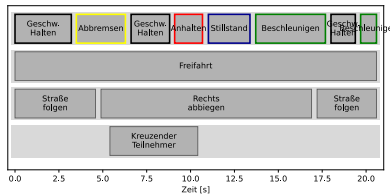
## A.4 Repräsentanten der Beispielszenarien



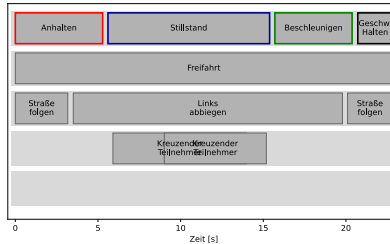
(a) Repräsentant für das Cluster von BS 1



(b) Repräsentant für das Cluster von BS 2



(c) Repräsentant für das Cluster von BS 3



(d) Repräsentant für das Cluster von BS 4

Abbildung A.6: Manövermatrizen Repräsentanten der Cluster der Beispielszenarien





# Abkürzungen

<b>AF</b>	Automatisierte Fahrfunktionen
<b>HAF</b>	Hochautomatisierte Fahrfunktionen
<b>FA</b>	Fahrerassistenzsysteme
<b>VDA</b>	Verband der Automobilindustrie e. V.
<b>ODD</b>	Operational Design Domain
<b>ESP</b>	Elektronisches Stabilitätsprogramm
<b>ABS</b>	Antiblockiersystem
<b>ACC</b>	Adaptive Cruise Control
<b>Car2X</b>	Car-to-X Kommunikation
<b>V&amp;V</b>	Verifikation und Validierung
<b>PEP</b>	Produktenstehungsprozess
<b>SOP</b>	Start of Production
<b>SiL</b>	Software-in-the-Loop
<b>HiL</b>	Hardware-in-the-Loop
<b>ViL</b>	Vehicle-in-the-Loop
<b>OT</b>	Onboard Test
<b>RP</b>	Rapid Prototyping

<b>MiL</b>	Model-in-the-Loop
<b>XiL</b>	X-in-the-Loop
<b>RL</b>	Reinforcement Learning
<b>DSL</b>	Domain Specific Language
<b>DTW</b>	Dynamic Time Warping
<b>ASIL</b>	Automotive Safety Integrity Level
<b>FMEA</b>	Failure Mode and Effects Analysis
<b>TARA</b>	Threat Analysis and Risk Assessment
<b>SBT</b>	Szenariobasiertes Testen
<b>MDS</b>	Multidimensionale Skalierung
<b>UMAP</b>	Uniform Manifold Approximation and Projection
<b>PCA</b>	Principal Component Analysis
<b>SQL</b>	Structured Query Language
<b>PoC</b>	Proof-of-Concept
<b>LLM</b>	Large Language Model

# Abbildungsverzeichnis

1.1	Automatisierungsgrade des automatisierten Fahrens nach SAE J3016 [9] . . . . .	4
2.1	Der Produktenstehungsprozess, die zugehörigen Phasen und mögliche Reifegrade nach VDA [22] . . . . .	11
2.2	Vereinfachte Darstellung des V-Modells basierend auf [24] . . .	13
2.3	Beziehungen zwischen PEP, Musterphasen, Reifegraden und V-Modell mit beispielhafter Taktung von Hardware, Software und Mechanik [22] . . . . .	14
2.4	Mögliche Integration von agiler Softwareentwicklung ins V-Modell nach Reisgys et al. [33] . . . . .	16
2.5	Überblick der relevanten Begriffe und Beziehungen für Testplattformen . . . . .	17
2.6	Gegenseitiger Einfluss von Testobjekt und Testumgebung bei X-in-the-Loop Plattformen . . . . .	19
2.7	Kategorisierung der Testplattformen über die Umsetzung der Testumgebung und des Testobjekts basierend auf Sax et al. [13] . . .	19
3.1	Manöverkategorien und ihre Ausprägungen . . . . .	27
3.2	Überblick der Begriffe und Beziehungen für Testfälle im szenariobasierten Testen . . . . .	33
3.3	Beziehungen von Szenarien zu Szenen [53], Abstraktionsebenen [77, 78] und Strukturierung der Elemente [79]. . . . .	34
3.4	Einordnung der funktionalen, logischen und konkreten Szenariodarstellungsformen im V-Modell nach [78] . . . . .	35
3.5	Klassifikation der Szenarien in sicher/unsicher und bekannt/unbekannt nach ISO/PAS 21448 [75] . . . . .	36

3.6	Evolution der Bereiche während des Entwicklungsprozesses durch Anwendung der ISO/PAS 21448 Aktivitäten [75] . . . . .	37
3.7	Darstellung des SBT-Prozesses für das szenariobasiertes Testen nach [82] . . . . .	38
3.8	Repräsentative Szenarien als Stützstellen für die Testabdeckung innerhalb der ODD nach [90] . . . . .	40
3.9	Mögliche Variationen für die Startszene eines Überhol szenarios [36] . . . . .	41
3.10	Verknüpfung des SBT-Prozesses (s. Abschnitt 3.3) mit dem V-Modell (s. Unterabschnitt 2.1.4) . . . . .	44
3.11	Input, Output und Beziehungen der Kategorien zum Erstellen von Szenarien nach Schütt el al. [TB11]. . . . .	45
3.12	Unterschiedliche Typen von Szenarien nach Quelle und enthaltenen Informationen . . . . .	52
4.1	Verknüpfung zwischen den Anforderungen, Herausforderungen (s. Abschnitt 3.2) und Prozessschritten (s. Abschnitt 3.3) . . . . .	61
5.1	Beispielszenario zur Veranschaulichung der Verhaltensbeschreibung über Manöver . . . . .	68
5.2	Übersicht zum Vorgehensmodell des manövergestützten Testens auf Basis des Prozesses zum szenariobasierten Testen (s. Abbildung 3.7) . . . . .	69
5.3	Vorgehen zum Erkennen von Manövern in aufgezeichneten Realdaten und Speichern dieser in der Manöverdatenbank. . . . .	70
5.4	Aktivitätsdiagramm für die Manöver-Identifikation und Szenario-Extraktion . . . . .	71
5.5	Beispiel einer Manöver-Identifikation für die Kategorie “Geschwindigkeit” anhand des Geschwindigkeitsverlaufs für das Szenario in Abbildung 5.3 . . . . .	73
5.6	Zielbild für das Ableiten der Testszenarien innerhalb einer ODD am Beispiel von drei repräsentativen Szenarien . . . . .	74
5.7	Aktivitätsdiagramm für die ähnlichkeitsorientierte Selektion von Testszenarien . . . . .	75

5.8	Veranschaulichung der Notation zur Distanzberechnung für die Szenarien $S^a$ und $S^b$ , beispielhaft anhand der Distanz der Manöversequenzen für die Kategorien “Geschwindigkeit”(g) und “Folgen”(f). Für die vollständige Berechnung werden alle Manöverkategorien verwendet. . . . .	77
5.9	Veranschaulichung der Methodik Sequenzalignment für die Berechnung der Distanz von zwei Manöversequenzen . . . . .	78
5.10	Visualisierung der Ähnlichkeit der Szenarien über das manöverbasierte Distanzmaß und die Einbettung des Szenariengraphen .	80
5.11	Clustering der Szenarien für drei Gruppen ähnlicher Szenarien .	82
5.12	Veranschaulichung der Notationen zu den berechneten Distanzen für die Repräsentanz eines Testsets . . . . .	83
5.13	Beispiel für Ausführung eines repräsentativen Szenarios auf der SiL-Plattform IPG CarMaker [121]. . . . .	85
5.14	Aktivitätsdiagramm für das Ausführen der Szenarien auf einer Testplattform . . . . .	86
5.15	Validierung der ausgeführten Szenarien über das entwickelte manöverbasierte Distanzmaß . . . . .	87
5.16	Aktivitätsdiagramm für die manövergestützte Validierung von ausgeführten Testszenarien . . . . .	88
5.17	Beispiel für ein valide ausgeführtes Szenario $S^v$ und ein invalide ausgeführtes Szenario $S^i$ . Durch Vergleich der manöverbasierte Distanzen zum Testszenario $S^t$ wird das invalide Szenario erkannt.	89
5.18	Elemente der Szenario-Beschreibungsform (weiß) und übergeordnete Gruppierungen (grau). . . . .	91
5.19	Metamodell der Szenariobeschreibungsform mit den Elementen und deren Beziehungen als UML-Klassendiagramm . . . . .	93
5.20	Zusammenfassung des Modells zum manövergestützten Testen .	94
6.1	Struktogramm für die Manöver-Identifikation der Kategorie “Geschwindigkeit” . . . . .	98
6.2	Struktogramm für die Manöver-Identifikation der Kategorie “Folgen” . . . . .	99
6.3	Darstellung der Ausdehnung eines Manövers <i>Spurwechsel</i> (blau)	99

6.4	Struktogramm für die Manöver-Identifikation der Kategorie “Route” . . . . .	101
6.5	Finden des nächsten vorausfahrenden Fahrzeugs in einem aufgezeichneten Szenario anhand der OpenDRIVE-Karte . . . . .	102
6.6	Struktogramm für die Kontext-Identifikation der Kategorie “Kreuzung” . . . . .	103
6.7	Erstellen eines logischen Szenarios aus aufgezeichneten Szenarien	104
6.8	Aktivitätsdiagramm zur Erstellung von OpenSCENARIO Dateien für die Ausführung der Testszenarien . . . . .	108
6.9	Beispiel eines konvertierten Stillstandmanövers in den Industriestandard OpenSCENARIO XML . . . . .	109
6.10	UML-Klassendiagramm der Teilnehmer inklusive deren Parameter	111
6.11	UML-Klassendiagramm der statischen Elemente inklusive deren Parameter . . . . .	112
6.12	UML-Klassendiagramm der Manöver inklusive deren Parameter und Visualisierungen . . . . .	113
6.13	UML-Klassendiagramm der Kontexte und Visualisierungen . . .	115
7.1	Satellitenansicht der Datensatz-Kreuzung ( <a href="https://earth.google.com">https://earth.google.com</a> )	118
7.2	Beispielszenario 1 . . . . .	119
7.3	Beispielszenario 2 . . . . .	119
7.4	Beispielszenario 3 . . . . .	120
7.5	Beispielszenario 4 . . . . .	120
7.6	Ablauf von Beispielszenario 4 mit Ego-Fahrzeug und drei weiteren Teilnehmern an einer Kreuzung. Die Farben zeigen die Manöver der Kategorie “Geschwindigkeit” des Fahrzeugs: Anhalten (rot), Stillstand (blau), Beschleunigen (grün) und Geschwindigkeit halten (schwarz). . . . .	121
7.7	Geschwindigkeitsverlauf in Beispielszenario 4 . . . . .	122
7.8	Signalverläufe für Manöver der Kategorie “Route” . . . . .	123
7.9	Größe der logischen Szenarien im Datensatz auf logarithmischer Skala . . . . .	125
7.10	Parameterverteilungen der Geschwindigkeit für das größte logische Szenario . . . . .	126

---

7.11	Ein einzelnes vorkommendes Szenario im PoC mit mehreren (kreuzenden) Teilnehmern und langen Manöverfolgen . . . . .	126
7.12	Szenariengraph für die Beispielszenarien (BS 1-4) . . . . .	128
7.13	Visualisierung als Graph mit farblicher Markierung der Cluster inkl. Annotation der Beispielszenarien und Charakteristika von Regionen. . . . .	130
7.14	Darstellung der Artefakte zur Ausführung von Beispielszenario 1 am SiL . . . . .	133
7.15	Valide und invalide ausgeführte Variationen von Beispielszenario 4 erzeugt durch Parameteränderungen . . . . .	134
A.1	Beispiel für ein Alignment zwischen zwei Sequenzen . . . . .	152
A.2	Beispiele für die Verwendung von Graphenvisualisierung in unterschiedlichen Domänen. . . . .	154
A.5	Beispielszenario mit Ego-Fahrzeug und vorausfahrendem Teilnehmer an einer Kreuzung. Die Farben zeigen die longitudinalen Manöver des Fahrzeugs: Annähern (orange), Folgefahrt (blau) und Freifahrt (schwarz). . . . .	158
A.6	Manövermatrizen Repräsentanten der Cluster der Beispielszenarien	159





# Tabellenverzeichnis

3.1	Ausgewählte Manöverkataloge. Manöver, die ähnliches Verhalten beschreiben, sind auf der gleichen Ebene in der Tabelle dargestellt.	25
3.2	Vor- und Nachteile der Methoden zur Manöver-Identifikation . .	30
3.3	Überblick über Literatur zu Ähnlichkeiten von Szenarien, deren Anwendungsmöglichkeiten im szenariobasierten Testen und Berechnungsmethodik. . . . .	51
4.1	Vergleich der Anforderungen mit bestehenden Ansätzen aus der Literatur, gruppiert nach den Eigenschaften aus Unterabschnitt 3.4.4 und Unterabschnitt 3.4.5 . . . . .	63
4.2	Einsatz von modellbasierten Szenariobeschreibungen bezogen auf den SBT-Prozess (s. Abschnitt 3.3) . . . . .	65
5.1	Anwendbarkeit verschiedener Clustering-Algorithmen für die gegebenen Anforderungen . . . . .	84
6.1	Teilnehmertypen und Beschreibungen . . . . .	111
6.2	Spurtypen und statische Objekttypen sowie Beschreibungen . . .	112
6.3	Manöver-Kategorien, -Typen und Beschreibungen . . . . .	113
6.4	Kontext-Kategorien, -Typen und Beschreibungen . . . . .	114
7.1	Ermittlung der Kreuzungskontexte für die Teilnehmer des Beispielszenarios . . . . .	123
7.2	Distanzmatrix für die vier Beispielszenarien (BS 1-4) . . . . .	127
7.3	Eigenschaften für die Cluster der Beispielszenarien . . . . .	128
7.4	Distanzwerte der Beispielszenarien (BS 1-4) zu deren Repräsentanten (Rep 1-4) . . . . .	131
7.5	Übersicht über die Laufzeit der Prozessschritte und die Anzahl der prozessierten Szenarien . . . . .	135

A.1 Vergleich von Dimensionsreduktion und kraftbasierten Methoden  
zur Grapheneinbettung . . . . . 154

# Literaturverzeichnis

- [1] S. B. (Destatis), “Verkehrsunfälle 2021,” *Fachserie 8 Reihe 7*, 2022.
- [2] M. Maurer, J. C. Gerdes, B. Lenz, and H. Winner, *Autonomes Fahren: technische, rechtliche und gesellschaftliche Aspekte*. Springer Nature, 2015.
- [3] E. Union, “Regulation (eu) 2019/2144 of the european parliament and of the council of ministers,” 2019.
- [4] M. Randelhoff, “Vergleich unterschiedlicher flächeninanspruchnahmen nach verkehrsarten (pro person),” *Zukunft Mobilität*. Online verfügbar unter <https://www.zukunft-mobilitaet.net/78246/analyse/flaechenbedarf-pkw-fahrrad-bus-strassenbahn-stadtbahn-fussgaenger-metro-bremsverzoeigerung-vergleich/>, zuletzt aktualisiert am, vol. 10, p. 2019, 2014.
- [5] M. Ohnemus and A. Perl, “Shared autonomous vehicles: Catalyst of new mobility for the last mile?,” *Built Environment*, vol. 42, no. 4, pp. 589–602, 2016.
- [6] F. Zohm, “Automation verändert das transportwesen schon heute,” *ATZ-extra*, vol. 27, no. 1, pp. 50–50, 2022.
- [7] R. Berger, “Automated trucks-the next big disruptor in the automotive industry,” *Roland Berger*, 2016.
- [8] M. Krail, “Energie-und treibhausgaswirkungen des automatisierten und vernetzten fahrens im straßenverkehr,” 2019.
- [9] SAE International, “Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles (SAE J3016),” 2021.
- [10] C. Ebert and M. Weyrich, “Validation of autonomous systems,” *IEEE Software*, vol. 36, no. 5, pp. 15–23, 2019.
- [11] S. Schmerler, *Softwaretest in der Praxis: Grundlagen, Methoden und Technologien*. epubli, 2020.

- [12] C. Ebert, D. Bajaj, and M. Weyrich, “Testing software systems,” *IEEE Software*, vol. 39, no. 4, pp. 8–17, 2022.
- [13] E. Sax, *Automatisiertes Testen Eingebetteter Systeme in der Automobilindustrie*. Hanser, 2008.
- [14] E. Triou, Z. Abbas, and S. Kothapalle, “Declarative testing: A paradigm for testing software applications,” in *2009 Sixth International Conference on Information Technology: New Generations*, pp. 769–773, IEEE, 2009.
- [15] W. Wachenfeld and H. Winner, “The release of autonomous vehicles,” *Autonomous Driving: Technical, Legal and Social Aspects*, pp. 425–449, 2016.
- [16] N. Kalra and S. M. Paddock, “Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability?,” *Transportation Research Part A: Policy and Practice*, vol. 94, pp. 182–193, 2016.
- [17] S. Shalev-Shwartz, S. Shammah, and A. Shashua, “On a formal model of safe and scalable self-driving cars,” *arXiv preprint arXiv:1708.06374*, 2017.
- [18] S. Mitsch, K. Ghorbal, and A. Platzer, “On provably safe obstacle avoidance for autonomous robotic ground vehicles,” in *Robotics: Science and Systems IX, Technische Universität Berlin, Berlin, Germany, June 24-June 28, 2013*, 2013.
- [19] C. Ebert, M. Weyrich, B. Lindemann, and S. P. Chandrasekar, “Systematic testing for autonomous driving,” *ATZelectronics worldwide*, vol. 16, no. 3, pp. 18–23, 2021.
- [20] J. Weber, *Automotive development processes: Processes for successful customer oriented vehicle development*. Springer Science & Business Media, 2009.
- [21] S. Kerber, *Prozessgestaltung zum Einsatz digitaler Fabrikgesamtmodelle*. Springer, 2016.
- [22] Qualitäts Management Center, *Das gemeinsame Qualitätsmanagement in der Lieferkette*. Verband der Automobilindustrie VDA e.V. (VDA), 2021.
- [23] S. Rudert and J. Trumpheller, “Der Produktentstehungsprozess - Grundlage für den Erfolg eines Produktes,” *Porsche Engineering Magazin*, 2015.
- [24] W. Dröschel and M. Wiemers, *Das V-Modell 97: der Standard für die Entwicklung von IT-Systemen mit Anleitung für den Praxiseinsatz*. Walter de Gruyter GmbH & Co KG, 2015.

- [25] B. Boehm, “Verifying and validating software requirements and design specifications,” *IEEE Software*, vol. 1, no. 1, pp. 75–88, 1984.
- [26] P. Rook, “Controlling software projects,” *Software Engineering Journal*, vol. 1, pp. 7–16(9), January 1986.
- [27] J. Schäuuffele and T. Zurawka, *Automotive software engineering*. Springer, 2010.
- [28] C. King, *Bewertung von Fahrerassistenzsystemen im Umfeld des szenariobasierten Testens*. PhD thesis, Karlsruher Institut für Technologie (KIT), 2023.
- [29] J. Bach, S. Otten, and E. Sax, “Model based scenario specification for development and test of automated driving functions,” in *2016 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1149–1155, IEEE, 2016.
- [30] M. Eigner, “Überblick disziplin-spezifische und-übergreifende vorgehensmodelle,” in *Modellbasierte virtuelle Produktentwicklung*, pp. 15–52, Springer, 2014.
- [31] K. Beck, M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, J. Kern, B. Marick, R. C. Martin, S. Mellor, K. Schwaber, J. Sutherland, and D. Thomas, “Manifesto for agile software development,” 2001.
- [32] K. Schwaber and J. Sutherland, “The scrum guide,” *Scrum Alliance*, vol. 21, no. 19, p. 1, 2011.
- [33] F. Reisgys, J. Plaum, A. Schwarzhaupt, and E. Sax, “Scenario-based x-in-the-loop test for development of driving automation,” *14. Workshop Fahrerassistenz und automatisiertes Fahren*, 2022.
- [34] P. Hohl, J. Ghofrani, J. Münch, M. Stupperich, and K. Schneider, “Searching for common ground: Existing literature on automotive agile software product lines,” in *Proceedings of the 2017 International Conference on Software and System Process, ICSSP 2017*, (New York, NY, USA), p. 70–79, Association for Computing Machinery, 2017.
- [35] S. Weber, “Agile in automotive—state of practice 2015,” *Study, Kornwestheim*, vol. 58, 2015.
- [36] R. Pfeffer, *Szenariobasierte simulationsgestützte funktionale Absicherung hochautomatisierter Fahrfunktionen durch Nutzung von Realdaten*. PhD thesis, Dissertation, Karlsruhe, Karlsruher Institut für Technologie (KIT), 2020, 2020.

- [37] J. Sutherland *et al.*, “The scrum@ scale guide,” *The definitive guide to Scrum@ Scale: scaling that works*. Scrum Inc, 2019.
- [38] C. Larman and B. Vodde, *Large-scale scrum: More with LeSS*. Addison-Wesley Professional, 2016.
- [39] D. Leffingwell, *SAFe 4.5 reference guide: scaled agile framework for lean enterprises*. Addison-Wesley Professional, 2018.
- [40] J. Bach, *Methoden und Ansätze für die Entwicklung und den Test prädiktiver Fahrzeugregelungsfunktionen*. PhD thesis, Karlsruher Institut für Technologie (KIT), 2018.
- [41] K. von Neumann-Cosel, *Virtual Test Drive*. Dissertation, Technische Universität München, München, 2014.
- [42] F. M. Reisgys, *Glaubwürdigkeit und Einsatz des szenariobasierten X-in-the-Loop-Tests für Fahrerassistenzsysteme*. PhD thesis, Karlsruher Institut für Technologie (KIT), 2024.
- [43] C. King, L. Ries, J. Langner, and E. Sax, “A taxonomy and survey on validation approaches for automated driving systems,” in *2020 IEEE International Symposium on Systems Engineering (ISSE)*, pp. 1–8, IEEE, 2020.
- [44] M. Steimle, T. Menzel, and M. Maurer, “A method for classifying test bench configurations in a scenario-based test approach for automated vehicles,” *arXiv preprint arXiv:1905.09018*, 2019.
- [45] International Organization for Standardization, “Road vehicles – Functional safety,” 2011.
- [46] H.-H. Nagel, W. Enkelmann, and G. Struck, “Fhg-co-driver: From map-guided automatic driving by machine vision to a cooperative driver support,” *Mathematical and Computer Modelling*, vol. 22, no. 4-7, pp. 185–212, 1995.
- [47] J. Kramer, *Kollektive Energieeffizienz auf mehrspurigen Straßen*. PhD thesis, KIT, 05 2018.
- [48] G. Bagschik, T. Menzel, and M. Maurer, “Ontology based scene creation for the development of automated vehicles,” in *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1813–1820, IEEE, 2018.
- [49] L. Hartjen, R. Philipp, F. Schuldt, B. Friedrich, and F. Howar, “Classification of driving maneuvers in urban traffic for parametrization of test scenarios,” in *9. Tagung Automatisiertes Fahren*, 2019.

- [50] H. Stachowiak, "Allgemeine modelltheorie," (*No Title*), 1973.
- [51] M. Glinz, "Einführung in die modellierung," 2005.
- [52] W. Tölle, "Ein fahrmanöverkonzept für einen maschinellen kopiloten," *FORTSCHRITT BERICHTE-VDI REIHE 12 VERKEHRSTECHNIK FAHRZEUGTECHNIK*, 1996.
- [53] A. Reschka, *Fertigkeiten-und Fähigkeitengraphen als Grundlage des sicheren Betriebs von automatisierten Fahrzeugen im öffentlichen Straßenverkehr in städtischer Umgebung*. PhD thesis, Universität Carolo-Wilhelmina zu Braunschweig, 2017.
- [54] ASAM, "OpenSCENARIO 1.0 user guide." <https://www.asam.net/index.php?eID=dumpFile&t=f&f=3496&token=df4fdaf41a8463e585495001cc3db3298b57d426>. Accessed: 2020-08-12.
- [55] H. Nagel and W. Enkelmann, "Generic road traffic situations and driver support systems," in *PROMETHEUS (PROGRAM) WORKSHOP (5TH: 1991:)*, 1991.
- [56] J. Firl and Q. Tran, "Probabilistic maneuver prediction in traffic scenarios.," in *ECMR*, pp. 89–94, 2011.
- [57] M. Schreiber, *Konzeptionierung und Evaluierung eines Ansatzes zu einer manöverbasierten Fahrzeugführung im Nutzungskontext Autobahnfahrten*. PhD thesis, Technische Universität, 2012.
- [58] ASAM, "OpenSCENARIO 2.0 concept paper." <https://www.asam.net/index.php?eID=dumpFile&t=f&f=3460&token=14e7c7fab9c9b75118bb4939c725738fa0521fe9>. Accessed: 2020-08-12.
- [59] C. Roesener, F. Fahrenkrog, A. Uhlig, and L. Eckstein, "A scenario-based assessment approach for automated driving by using time series classification of human-driving behaviour," in *2016 IEEE 19th international conference on intelligent transportation systems (ITSC)*, pp. 1360–1365, IEEE, 2016.
- [60] F. Montanari, C. Stadler, J. Sichermann, R. German, and A. Djanatljev, "Maneuver-based resimulation of driving scenarios based on real driving data," in *2021 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1124–1131, IEEE, 2021.
- [61] Q. Li, Z. Peng, L. Feng, Q. Zhang, Z. Xue, and B. Zhou, "Metadrive: Composing diverse driving scenarios for generalizable reinforcement learning."

- ning,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.
- [62] B. Zhu, S. Yan, J. Zhao, and W. Deng, “Personalized lane-change assistance system with driver behavior identification,” *IEEE Transactions on Vehicular Technology*, vol. 67, no. 11, pp. 10293–10306, 2018.
- [63] A. Aboah, Y. Adu-Gyamfi, S. V. Gursoy, J. Merickel, M. Rizzo, and A. Sharma, “Driver maneuver detection and analysis using time series segmentation and classification,” *Journal of transportation engineering, Part A: Systems*, vol. 149, no. 3, p. 04022157, 2023.
- [64] A. Erdogan, B. Ugranli, E. Adali, A. Sentas, E. Mungan, E. Kaplan, and A. Leitner, “Real-world maneuver extraction for autonomous vehicle validation: A comparative study,” in *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 267–272, IEEE, 2019.
- [65] M. Barbier, C. Laugier, O. Simonin, and J. Ibanez-Guzman, “Classification of drivers manoeuvre for road intersection crossing with synthetic and real data,” in *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 224–230, IEEE, 2017.
- [66] M. S. N. Al-Din, “Driving maneuvers recognition and classification using a hybrid pattern matching and machine learning,” *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 2, 2023.
- [67] F. Montanari, *Automatic Generation of Test Scenarios and Simulation Scenarios based on Real-World Driving Data*. PhD thesis, Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), 2023.
- [68] R. Krajewski, T. Moers, D. Neger, and L. Eckstein, “Data-driven maneuver modeling using generative adversarial networks and variational autoencoders for safety validation of highly automated vehicles,” in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 2383–2390, Nov 2018.
- [69] L. Hartjen, F. Schuldt, and B. Friedrich, “Semantic classification of pedestrian traffic scenarios for the validation of automated driving,” in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 3696–3701, IEEE, 2019.
- [70] J. Cai, W. Deng, H. Guang, Y. Wang, J. Li, and J. Ding, “A survey on data-driven scenario generation for automated vehicle testing,” *Machines*, vol. 10, no. 11, p. 1101, 2022.



- [71] L. Hartjen, R. Philipp, F. Schuldt, and B. Friedrich, “Saturation effects in recorded maneuver data for the test of automated driving,” *Highway Res. Rec.*, 2020.
- [72] T. Nanri, F. Fang, and A. Khiat, “Use-case generation and analysis for autonomous driving in urban areas,” *International journal of automotive engineering*, vol. 12, no. 2, pp. 54–61, 2021.
- [73] COMMISSION IMPLEMENTING REGULATION (EU) 2022/1426, “COMMISSION IMPLEMENTING REGULATION (EU) 2022/1426 of 5 August 2022 laying down rules for the application of Regulation (EU) 2019/2144 of the European Parliament and of the Council as regards uniform procedures and technical specifications for the type-approval of the automated driving system (ADS) of fully automated vehicles.” <https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32022R1426#d1e41-20-1>, 2022. [Online; accessed 17-October-2022].
- [74] F. Hauer, T. Schmidt, B. Holzmüller, and A. Pretschner, “Did we test all scenarios for automated and autonomous driving systems?,” in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 2950–2955, IEEE, 2019.
- [75] International Organization for Standardization, “ISO/PAS 21448:2019 - Road vehicles — Safety of the intended functionality,” 2019.
- [76] S. Ulbrich, T. Menzel, A. Reschka, F. Schuldt, and M. Maurer, “Defining and substantiating the terms scene, situation, and scenario for automated driving,” in *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, pp. 982–988, 2015.
- [77] PEGASUS, “Pegasus method - an overview.” <https://www.pegasusprojekt.de/files/tmpl/Pegasus-Abschlussveranstaltung/PEGASUS-Gesamtmethode.pdf>, 2019. Accessed: 2022-03-01.
- [78] G. Bagschik, T. Menzel, A. Reschka, and M. Maurer, “Szenarien für entwicklung, absicherung und test von automatisierten fahrzeugen,” in *11. Workshop Fahrerassistenzsysteme. Uni-DAS e. V.*, pp. 125–135, 2017.
- [79] M. Scholtes, L. Westhofen, L. R. Turner, K. Lotto, M. Schuldes, H. Weber, N. Wagener, C. Neurohr, M. H. Bollmann, F. Körtke, *et al.*, “6-layer model for a structured description and categorization of urban traffic and environment,” *IEEE Access*, vol. 9, pp. 59131–59147, 2021.

- [80] F. Schuldt, *Ein Beitrag für den methodischen Test von automatisierten Fahrfunktionen mit Hilfe von virtuellen Umgebungen*. PhD thesis, TU Braunschweig, 2017.
- [81] J. Bock, R. Krajewski, L. Eckstein, J. Klimke, J. Sauerbier, and A. Zlocki, “Data basis for scenario-based validation of had on highways,” in *27th Aachen colloquium automobile and engine technology*, pp. 8–10, 2018.
- [82] C. Neurohr, L. Westhofen, T. Henning, T. de Graaff, E. Möhlmann, and E. Böde, “Fundamental considerations around scenario-based testing for automated driving,” *arXiv preprint arXiv:2005.04045*, 2020.
- [83] T. Ponn, C. Gnandt, and F. Diermeyer, “An optimization-based method to identify relevant scenarios for type approval of automated vehicles,” in *Proceedings of the ESV—International Technical Conference on the Enhanced Safety of Vehicles, Eindhoven, The Netherlands*, pp. 10–13, 2019.
- [84] J. Langner, H. Grolig, S. Otten, M. Holzäpfel, and E. Sax, “Logical scenario derivation by clustering dynamic-length-segments extracted from real-world-driving-data.,” in *VEHITS*, pp. 458–467, 2019.
- [85] T. Menzel, G. Bagschik, L. Isensee, A. Schomburg, and M. Maurer, “From functional to logical scenarios: Detailing a keyword-based scenario description for execution in a simulation environment,” in *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 2383–2390, IEEE, 2019.
- [86] H. Watanabe, L. Tobisch, J. Rost, J. Wallner, and G. Prokop, “Scenario mining for development of predictive safety functions,” in *2019 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*, pp. 1–7, IEEE, 2019.
- [87] J. Bach, J. Langner, S. Otten, E. Sax, and M. Holzäpfel, “Test scenario selection for system-level verification and validation of geolocation-dependent automotive control systems,” in *2017 International Conference on Engineering, Technology and Innovation (ICE/ITMC)*, pp. 203–210, IEEE, 2017.
- [88] U. Di Fabio, M. Broy, R. J. Brünger, U. Eichhorn, A. Grunwald, D. Heckmann, E. Hilgendorf, H. Kagermann, A. Losinger, M. Lutz-Bachmann, et al., “Ethik-kommission automatisiertes und vernetztes fahren,” 2017.
- [89] H. Watanabe, *Methodik zur Determinierung repräsentativer und relevanter Testszenarien für prädiktive Sicherheitsfunktionen*, vol. 21. Cuvillier Verlag, 2022.

- [90] V. M. Projekt, “VVM main approach - How to systematically release AD systems?.” [https://www.vvm-projekt.de/fileadmin/user\\_upload/Mid-Term/Presentations/VVM\\_HZE\\_MS3\\_20220315\\_VVMmaInApproach.pdf](https://www.vvm-projekt.de/fileadmin/user_upload/Mid-Term/Presentations/VVM_HZE_MS3_20220315_VVMmaInApproach.pdf). Accessed: 2023-01-31.
- [91] S. Riedmaier, T. Ponn, D. Ludwig, B. Schick, and F. Diermeyer, “Survey on scenario-based safety assessment of automated vehicles,” *IEEE access*, vol. 8, pp. 87456–87477, 2020.
- [92] F. Schuldt, T. Menzel, and M. Maurer, “Eine methode für die zuordnung von testfällen für automatisierte fahrfunktionen auf x-in-the-loop verfahren im modularen virtuellen testbaukasten,” in *Workshop Fahrerassistenzsysteme*, pp. 171–182, 2015.
- [93] S. Riedmaier, J. Nesensohn, C. Gutenkunst, T. Düser, B. Schick, and H. Abdellatif, “Validation of x-in-the-loop approaches for virtual homologation of automated driving functions,” in *11th Graz Symposium Virtual Vehicle*, 2018.
- [94] B. Schütt, M. Steimle, B. Kramer, D. Behnecke, and E. Sax, “A taxonomy for quality in simulation-based development and testing of automated driving systems,” *IEEE Access*, vol. 10, pp. 18631–18644, 2022.
- [95] Y. Abeyesirigoonawardena, F. Shkurti, and G. Dudek, “Generating adversarial driving scenarios in high-fidelity simulators,” in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 8271–8277, IEEE, 2019.
- [96] J. Wang, A. Pun, J. Tu, S. Manivasagam, A. Sadat, S. Casas, M. Ren, and R. Urtasun, “Advsim: Generating safety-critical scenarios for self-driving vehicles,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9909–9918, 2021.
- [97] B. Gangopadhyay, S. Khashtgir, S. Dey, P. Dasgupta, G. Montana, and P. Jennings, “Identification of test cases for automated driving systems using bayesian optimization,” in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 1961–1967, IEEE, 2019.
- [98] M. Klischat and M. Althoff, “Generating critical test scenarios for automated vehicles with evolutionary algorithms,” in *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 2352–2358, IEEE, 2019.
- [99] D. Karunakaran, S. Worrall, and E. Nebot, “Efficient statistical validation with edge cases to evaluate highly automated vehicles,” in *2020 IEEE 23rd*

- International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–8, IEEE, 2020.
- [100] B. Schütt, M. Heinrich, S. Marahrens, J. M. Zöllner, and E. Sax, “An Application of Scenario Exploration to Find New Scenarios for the Development and Testing of Automated Driving Systems in Urban Scenarios,” in *Proceedings of the 8th International Conference on Vehicle Technology and Intelligent Transport Systems*, pp. 338–345, 2022.
  - [101] A. Bussler, L. Hartjen, R. Philipp, and F. Schuldt, “Application of evolutionary algorithms and criticality metrics for the verification and validation of automated driving systems at urban intersections,” in *2020 IEEE Intelligent Vehicles Symposium (IV)*, pp. 128–135, IEEE, 2020.
  - [102] R. B. Abdessalem, S. Nejati, L. C. Briand, and T. Stifter, “Testing vision-based control systems using learnable evolutionary algorithms,” in *2018 IEEE/ACM 40th International Conference on Software Engineering (ICSE)*, pp. 1016–1026, IEEE, 2018.
  - [103] W. Ding, B. Chen, M. Xu, and D. Zhao, “Learning to collide: An adaptive safety-critical scenarios generating method,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2243–2250, IEEE, 2020.
  - [104] D. Baumann, R. Pfeffer, and E. Sax, “Automatic generation of critical test cases for the development of highly automated driving functions,” in *2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*, pp. 1–5, IEEE, 2021.
  - [105] S. Kuutti, S. Fallah, and R. Bowden, “Training adversarial agents to exploit weaknesses in deep control policies,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 108–114, IEEE, 2020.
  - [106] T. Menzel, G. Bagschik, and M. Maurer, “Scenarios for development, test and validation of automated vehicles,” in *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1821–1827, IEEE, 2018.
  - [107] I. Paranjape, A. Jawad, Y. Xu, A. Song, and J. Whitehead, “A modular architecture for procedural generation of towns, intersections and scenarios for testing autonomous vehicles,” in *2020 IEEE Intelligent Vehicles Symposium (IV)*, pp. 162–168, IEEE, 2020.
  - [108] S. Rietsch, S.-Y. Huang, G. Kontes, A. Plinge, and C. Mutschler, “Driver dojo: A benchmark for generalizable reinforcement learning for autonomous driving,” *arXiv preprint arXiv:2207.11432*, 2022.

- [109] L. Feng, Q. Li, Z. Peng, S. Tan, and B. Zhou, “Trafficgen: Learning to generate diverse and realistic traffic scenarios,” *arXiv preprint arXiv:2210.06609*, 2022.
- [110] Q. Goss, Y. AlRashidi, and M. İ. Akbaş, “Generation of modular and measurable validation scenarios for autonomous vehicles using accident data,” in *2021 IEEE Intelligent Vehicles Symposium (IV)*, pp. 251–257, IEEE, 2021.
- [111] Z. Ghodsi, S. K. S. Hari, I. Frosio, T. Tsai, A. Troccoli, S. W. Keckler, S. Garg, and A. Anandkumar, “Generating and characterizing scenarios for safety testing of autonomous vehicles,” in *2021 IEEE Intelligent Vehicles Symposium (IV)*, pp. 157–164, IEEE, 2021.
- [112] N. Weber, C. Thiem, and U. Konigorski, “Unscene: Toward unsupervised scenario extraction for automated driving systems from urban naturalistic road traffic data,” *arXiv preprint arXiv:2202.06608*, 2022.
- [113] L. Balasubramanian, J. Wurst, M. Botsch, and K. Deng, “Traffic scenario clustering by iterative optimisation of self-supervised networks using a random forest activation pattern similarity,” in *2021 IEEE Intelligent Vehicles Symposium (IV)*, pp. 682–689, IEEE, 2021.
- [114] Z. Xinxin, L. Fei, and W. Xiangbin, “Csg: Critical scenario generation from real traffic accidents,” in *2020 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1330–1336, IEEE, 2020.
- [115] A. Gambi, T. Huynh, and G. Fraser, “Generating effective test cases for self-driving cars from police reports,” in *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pp. 257–267, 2019.
- [116] Y. Cao, L. Xiao, H. Dong, Y. Wang, X. Wu, P. Li, and Y. Qiu, “Typical pre-crash scenarios reconstruction for two-wheelers and passenger vehicles and its application in parameter optimization of aeb system based on nais database,” in *26th International Technical Conference on the Enhanced Safety of Vehicles (ESV)*, 2019.
- [117] F. Hauer, I. Gerostathopoulos, T. Schmidt, and A. Pretschner, “Clustering traffic scenarios using mental models as little as possible,” in *2020 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1007–1012, IEEE, 2020.
- [118] J. Kerber, S. Wagner, K. Groh, D. Notz, T. Kühbeck, D. Watenig, and A. Knoll, “Clustering of the scenario space for the assessment of automated

- driving,” in *2020 IEEE Intelligent Vehicles Symposium (IV)*, pp. 578–583, IEEE, 2020.
- [119] L. Ries, *Fahrsequenz-Clustering zur datenbasierten Szenarienanalyse für die Validierung urbaner automatisierter Fahrfunktionen*. PhD thesis, Karlsruher Institut für Technologie (KIT), 2025.
- [120] S. Hoogendoorn and P. Bovy, “State-of-the-art of vehicular traffic flow modeling,” *J. Syst. Cont. Eng.*, vol. 215, pp. 283–303, 06 2001.
- [121] IPG, “Ipg - automotive gmbh - everything about virtual test driving.” Accessed: Nov. 11, 2022.
- [122] MSC Software, “Virtual test drive (vtd) - complete tool-chain for driving simulation.” Accessed: Nov. 11, 2022.
- [123] dSPACE GmbH, “dSPACE - SIMPHERA Web-based solution for simulation and validation in autonomous driving development.”
- [124] 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.
- [125] esmini, “Environment simulator minimalistic.” Accessed: 11.02.2024.
- [126] K. Backhaus, B. Erichson, S. Gensler, R. Weiber, and T. Weiber, *Multivariate Analysemethoden*. Springer-Verlag, 2021.
- [127] L. Fahrmeir, *Multivariate statistische Verfahren*. De Gruyter, 2. überarb. aufl. ed., 1996.
- [128] O. Siebinga, A. Zgonnikov, and D. Abbink, “Automatic extraction of similar traffic scenes from large naturalistic datasets using the Hausdorff distance,” 2022.
- [129] Z. Zhang, K. Huang, and T. Tan, “Comparison of similarity measures for trajectory clustering in outdoor surveillance scenes,” in *18th International Conference on Pattern Recognition (ICPR’06)*, vol. 3, pp. 1135–1138, 2006.
- [130] S. Atev, G. Miller, and N. P. Papanikolopoulos, “Clustering of vehicle trajectories,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 3, pp. 647–657, 2010.
- [131] Q. Lin, W. Wang, Y. Zhang, and J. M. Dolan, “Measuring similarity of interactive driving behaviors using matrix profile,” in *2020 American Control Conference (ACC)*, pp. 3965–3970, 2020.

- [132] L. Ries, M. Stumpf, J. Bach, and E. Sax, "Semantic comparison of driving sequences by adaptation of word embeddings," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–7, IEEE, 2020.
- [133] L. Ries, J. Langner, S. Otten, J. Bach, and E. Sax, "A driving scenario representation for scalable real-data analytics with neural networks," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 2215–2222, IEEE, 2019.
- [134] R. Gruner, P. Henzler, G. Hinz, C. Eckstein, and A. Knoll, "Spatiotemporal representation of driving scenarios and classification using neural networks," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1782–1788, IEEE, 2017.
- [135] J. Lenard, R. Danton, M. Avery, A. Weekes, D. Zubay, and M. Kühn, "Typical pedestrian accident scenarios for the testing of autonomous emergency braking systems," in *Enhanced Safety of Vehicles (ESV) Conference, Washington, USA*, pp. 1–9, 2011.
- [136] M. Zipfl and J. M. Zöllner, "Towards traffic scene description: The semantic scene graph," in *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 3748–3755, 2022.
- [137] M. Winkelmann, M. Kohlhoff, H. H. Tadjine, and S. Müller, "Probabilistic metamodels for an efficient characterization of complex driving scenarios."
- [138] Y. Zheng, J. Wang, X. Li, C. Yu, K. Kodaka, and K. Li, "Driving risk assessment using cluster analysis based on naturalistic driving data," in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pp. 2584–2589, IEEE, 2014.
- [139] H. Elrofai, J.-P. Paardekoooper, E. de Gelder, S. Kalisvaart, and O. op Den Camp, "Streetwise: Scenario-based safety validation of connected and automated driving," *Netherlands Organization for Applied Scientific Research, TNO*, p. 28, 2018.
- [140] "Leveraging Regular Expressions for Flexible Scenario Detection in Recorded Driving Data," *ISSE 2020 - 6th IEEE International Symposium on Systems Engineering, Proceedings*, 2020.
- [141] P. Elspas., Y. Klose., S. Isele., J. Bach., and E. Sax., "Time series segmentation for driving scenario detection with fully convolutional networks," in *Proceedings of the 7th International Conference on Vehicle*

- Technology and Intelligent Transport Systems - VEHITS*, pp. 56–64, INSTICC, SciTePress, 2021.
- [142] N. Weber, C. Thiem, and U. Konigorski, “A needle in a haystack—how to derive relevant scenarios for testing automated driving systems in urban areas,” *arXiv preprint arXiv:2109.03648*, 2021.
  - [143] F. Gao, J. Duan, Z. Han, and Y. He, “Automatic virtual test technology for intelligent driving systems considering both coverage and efficiency,” p. 1, 2020.
  - [144] Y. Qi, Y. Luo, K. Li, W. Kong, and Y. Wang, “A trajectory-based method for scenario analysis and test effort reduction for highly automated vehicle,” SAE International, 2019.
  - [145] M. Rasch, P. Ubben, T. Most, V. Bayer, and R. Niemeier, “Safety Assessment and Uncertainty Quantification of Automated Driver Assistance Systems using Stochastic Analysis Methods,” in *NAFEMS World Congress 2019*, 2019.
  - [146] S. Hallerbach, Y. Xia, U. Eberle, and F. Koester, “Simulation-based identification of critical scenarios for cooperative and automated vehicles,” vol. 1, no. 2, pp. 93–106, 2018.
  - [147] M. Nabhan, M. Schoenauer, Y. Tourbier, and H. Hage, “Optimizing coverage of simulated driving scenarios for the autonomous vehicle,” in *2019 IEEE International Conference on Connected Vehicles and Expo (ICCVE)*, pp. 1–5, IEEE, 2019.
  - [148] C. Stadler, F. Montanari, W. Baron, C. Sippl, and A. Djanatljev, “A credibility assessment approach for scenario-based virtual testing of automated driving functions,” *IEEE Open Journal of Intelligent Transportation Systems*, vol. 3, pp. 45–60, 2022.
  - [149] S. Riedmaier, J. Nesensohn, C. Gutenkunst, T. Düser, B. Schick, and H. Abdellatif, “Validation of X-in-the-Loop Approaches for Virtual Homologation of Automated Driving Functions,” in *11th Graz Symposium Virtual Vehicle*, 2018.
  - [150] K. Groh, S. Wagner, T. Kuehbeck, and A. Knoll, “Simulation and Its Contribution to Evaluate Highly Automated Driving Functions,” in *WCX SAE World Congress Experience*, SAE (Soc. Automotive Eng.), 2019.
  - [151] F. Reisgys, M. Elgharbawy, A. Schwarzhaupt, and E. Sax, “Argumentation on ADAS Simulation Validity using Aleatory and Epistemic Uncertainty Estimation,” in *Proceedings of the Driving Simulation Conference*



- 2021 *Europe VR* (A. Kemeny, J.-R. Chardonnet, and F. Colombet, eds.), pp. 25–32, Driving Simulation Association, 2021.
- [152] F. Reisgys, J. Plaum, A. Schwarzhaupt, and E. Sax, “Validation and plausibilization of x-in-the-loop tests for driving automation,” in *27th International Technical Conference on the Enhanced Safety of Vehicles (ESV)*, 2023.
- [153] International Organization for Standardization, “ISO15622: Intelligent transport systems — Adaptive cruise control systems — Performance requirements and test procedures,” 2019.
- [154] W. Zhan, L. Sun, D. Wang, H. Shi, A. Clausse, M. Naumann, J. Kummerle, H. Konigshof, C. Stiller, A. de La Fortelle, *et al.*, “Interaction dataset: An international, adversarial and cooperative motion dataset in interactive driving scenarios with semantic maps,” *arXiv preprint arXiv:1910.03088*, 2019.
- [155] J. Bock, R. Krajewski, T. Moers, S. Runde, L. Vater, and L. Eckstein, “The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections,” in *2020 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1929–1934, 2020.
- [156] R. Krajewski, J. Bock, L. Kloecker, and L. Eckstein, “The highd dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems,” in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 2118–2125, 2018.
- [157] R. Krajewski, T. Moers, J. Bock, L. Vater, and L. Eckstein, “The round dataset: A drone dataset of road user trajectories at roundabouts in germany,” in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–6, 2020.
- [158] T. Moers, L. Vater, R. Krajewski, J. Bock, A. Zlocki, and L. Eckstein, “The exid dataset: A real-world trajectory dataset of highly interactive highway scenarios in germany,” in *2022 IEEE Intelligent Vehicles Symposium (IV)*, pp. 958–964, 2022.
- [159] C. Systematics, “Ngsim i-80 data analysis, summary reports,” *Federal Highway Administration*, available at, 2005.
- [160] M. Zipfl, T. Fleck, M. R. Zofka, and J. M. Zöllner, “From traffic sensor data to semantic traffic descriptions: The test area autonomous driving

- baden-württemberg dataset (taf-bw dataset),” in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, 2020.
- [161] T. Menzel, G. Bagschik, L. Isensee, A. Schomburg, and M. Maurer, “Detailing a Keyword Based Scenario Description for Execution in a Simulation Environment Using the Example of Scenarios on German Highways,” in *Workshop Fahrerassistenzsysteme und automatisiertes Fahren*, vol. 12, pp. 15–26, 2018.
- [162] W. Damm, E. Möhlmann, T. Peikenkamp, and A. Rakow, “A formal semantics for traffic sequence charts,” *Principles of Modeling: Essays Dedicated to Edward A. Lee on the Occasion of His 60th Birthday*, pp. 182–205, 2018.
- [163] M. P. Bauer, A. Ngo, and M. Resch, “The yase framework: Holistic scenario modeling with behavior trees,” in *2021 IEEE 94th Vehicular Technology Conference (VTC2021-Fall)*, pp. 1–7, IEEE, 2021.
- [164] T. Wuellner, S. Feuerstack, and A. Hahn, “Clustering environmental conditions of historical accident data to efficiently generate testing sceneries for maritime systems,” in *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11842 LNCS, pp. 349–362, Springer, oct 2019.
- [165] J. Bach, “Vehicle Simulations From in-Field Measurements,” 2016.
- [166] L. Hartjen, *Semantic classification of urban traffic scenarios for the validation of automated driving systems*. PhD thesis, Technische Universität Braunschweig, 2023.
- [167] T. Fleck, K. Daaboul, M. Weber, P. Schörner, M. Wehmer, J.-D. Doll, S. Orf, N. Sußmann, C. Hubschneider, M. Zofka, F. Kuhnt, R. Kohlhaas, I. Baumgart, R. Zoellner, and J. Zöllner, “Towards large scale urban traffic reference data: Smart infrastructure in the test area autonomous driving baden-württemberg,” 06 2018.
- [168] E. Gallagher, “Metoo hashtag network visualization.” <https://erin-gallagher.medium.com/metoo-hashtag-network-visualization-960dd5a97cdf>. Accessed: 2024-04-25.
- [169] J. MacQueen *et al.*, “Some methods for classification and analysis of multivariate observations,” in *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, pp. 281–297, Oakland, CA, USA, 1967.

- [170] H.-S. Park and C.-H. Jun, "A simple and fast algorithm for k-medoids clustering," *Expert systems with applications*, vol. 36, no. 2, pp. 3336–3341, 2009.
- [171] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *kdd*, vol. 96, pp. 226–231, 1996.
- [172] C. Rasmussen, "The infinite gaussian mixture model," *Advances in neural information processing systems*, vol. 12, 1999.
- [173] S. C. Johnson, "Hierarchical clustering schemes," *Psychometrika*, vol. 32, no. 3, pp. 241–254, 1967.
- [174] U. Von Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, pp. 395–416, 2007.
- [175] ASAM, "OpenDRIVE 1.7 user guide." [https://www.asam.net/fileadmin/Standards/OpenDRIVE/ASAM\\_OpenDRIVE\\_BS\\_V1-7-0.html#top-263d2bb0-9c50-4953-ae25-3ea2b94d0fdc](https://www.asam.net/fileadmin/Standards/OpenDRIVE/ASAM_OpenDRIVE_BS_V1-7-0.html#top-263d2bb0-9c50-4953-ae25-3ea2b94d0fdc). Accessed: 2020-08-12.
- [176] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of Molecular Biology*, vol. 48, no. 3, pp. 443–453, 1970.
- [177] I. Hamidah, S. Sriyono, M. nur hudha, and I. Ijost, "A bibliometric analysis of covid-19 research using vosviewer," *Indonesian Journal of Science and Technology*, vol. 5, pp. 209–216, 09 2020.
- [178] A. Beveridge, "Network of thrones." <https://networkofthrones.wordpress.com/>. Accessed: 2024-04-25.
- [179] S. G. Kobourov, "Spring embedders and force directed graph drawing algorithms," 2012.
- [180] E. R. Gansner, Y. Koren, and S. North, "Graph drawing by stress majorization," in *International Symposium on Graph Drawing*, pp. 239–250, Springer, 2004.
- [181] M. Klimenta, *Extending the usability of multidimensional scaling for graph drawing*. PhD thesis, Citeseer, 2012.
- [182] H. Gibson, J. Faith, and P. Vickers, "A survey of two-dimensional graph layout techniques for information visualisation," *Information visualization*, vol. 12, no. 3-4, pp. 324–357, 2013.



## Eigene Veröffentlichungen

- [TB1] Laurenz Adolph, Lennart Ries, Thilo Braun, Johannes Bernhard, Vishnu Priya Kumar, and Eric Sax. A Combined Maneuver-Trajectory Approach to Cluster Real-World Driving Data. In *2024 International Conference on Electrical and Computer Engineering Researches (ICECER)*, pages 1–8, 2024.
- [TB2] Thilo Braun, Julian Fuchs, Felix Reisgys, Lennart Ries, Johannes Plaum, Barbara Schütt, and Eric Sax. A Review of Scenario Similarity Measures for Validation of Highly Automated Driving. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 689–696, 2023.
- [TB3] Thilo Braun, Lennart Ries, Moritz Hesche, Stefan Otten, and Eric Sax. Maneuver-based Visualization of Similarities between Recorded Traffic Scenarios. In *Proceedings of the 11th International Conference on Data Science, Technology and Applications - Volume 1: DATA*, pages 236–244. INSTICC, SciTePress, 2022.
- [TB4] Thilo Braun, Lennart Ries, Franziska Körtke, Lara Turner, Stefan Otten, and Eric Sax. Collection of Requirements and Model-based Approach for Scenario Description. In *Proceedings of the 7th International Conference on Vehicle Technology and Intelligent Transport Systems - VEHITS*, pages 634–645. INSTICC, SciTePress, 2021.
- [TB5] Jannis Erz, Barbara Schütt, Thilo Braun, Houssem Guissouma, and Eric Sax. Towards an Ontology That Reconciles the Operational Design Domain, Scenario-based Testing, and Automated Vehicle Architectures. In *2022 IEEE International Systems Conference (SysCon)*, pages 1–8. IEEE, 2022.
- [TB6] Jacqueline Henle, Laurenz Adolph, Thilo Braun, Eric Sax, Lukas Erlichshagen, Christoph Stiller, Martin Lauer, Peter Vortisch, Sebastian Buck, and Claude Weyland. Kombination von Oberleitungssystemen und Elektrischen, Autonomen Nutzfahrzeugen, 2021.

- [TB7] Christian King, Thilo Braun, Constantin Braess, Jacob Langner, and Eric Sax. Capturing the Variety of Urban Logical Scenarios from Bird-view Trajectories. In *Proceedings of the 7th International Conference on Vehicle Technology and Intelligent Transport Systems - VEHITS*,, pages 471–480. INSTICC, SciTePress, 2021.
- [TB8] Hubert Padusinski, Christian Steinhauser, Thilo Braun, Lennart Ries, and Eric Sax. The Machine Vision Iceberg Explained: Advancing Dynamic Testing by Considering Holistic Environmental Relations. In *2024 IEEE 27th International Conference on Intelligent Transportation Systems (ITSC)*, pages 70–77, 2024.
- [TB9] Lennart Ries, Philipp Rigoll, Thilo Braun, Thomas Schulik, Johannes Daube, and Eric Sax. Trajectory-Based Clustering of Real-World Urban Driving Sequences with Multiple Traffic Objects. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 1251–1258. IEEE, 2021.
- [TB10] Barbara Schütt, Thilo Braun, Stefan Otten, and Eric Sax. SceML: A graphical modeling framework for scenario-based testing of autonomous vehicles. In *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*, pages 114–120, 2020.
- [TB11] Barbara Schütt and Thilo Braun. 1001 Ways of Scenario Generation for Testing of Self-driving Cars: A Survey. In *2023 IEEE International Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2023.

## Betreute studentische Arbeiten

- [S1] Constantin Braess. Extraktion von manöverbasierten logischen Szenarien im urbanen Verkehr. Bachelorarbeit, 2021.
- [S2] Moritz Hesche. Entwicklung intuitiver Visualisierungs- und Gruppierungsmethoden von Szenarioclustern im szenariobasierten Testen. Bachelorarbeit, 2022.
- [S3] Xiaoman Liu. Maneuver identification in urban traffic using Machine Learning. Masterarbeit, 2020.
- [S4] Luyi Pan. Similarity analysis of concrete scenarios in the context of virtual scenario-based validation of vehicles. Masterarbeit, 2022.
- [S5] Marc Schindewolf. Prüfung der möglichen Nutzung von autonomen Fahren in einem Fahrgeschäft. Masterarbeit, 2019.
- [S6] Lennard Sommer. Entwicklung und Evaluation eines Szenariodistanzmaßes für den Test automatisierter Fahrzeuge. Masterarbeit, 2022.
- [S7] Alexandros Tsiknas. Selektion und Priorisierung von Top-View Fahraufnahmen im szenario-basierten Testen hochautomatisierter Fahrfunktionen. Masterarbeit, 2023.