Szenariobasierte simulationsgestützte funktionale Absicherung hochautomatisierter Fahrfunktionen durch Nutzung von Realdaten

zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

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

genehmigte

Dissertation

von

Dipl.-Wi.-Ing. Raphael Pfeffer, M.Sc.

aus Frankfurt am Main

Tag der mündlichen Prüfung: 08.10.2020
Erster Gutachter: Prof. Dr.-Ing. Eric Sax
Zweiter Gutachter: Univ.-Prof. Dr.-Ing. Martin Fellendorf

Kurzfassung

Das autonome Fahren ist gegenwärtig einer der wesentlichen Innovations- und Forschungstreiber in der Automobilindustrie. Entwicklung und Test hochautomatisierter Fahrfunktionen als Vorstufe des autonomen Fahrens stehen vor großen Herausforderungen. Einerseits sind herkömmliche Testmethoden wie X-in-the-Loop zum Nachweis der funktionalen Sicherheit zwar grundsätzlich weiterhin anwendbar, beantworten jedoch andererseits per se nicht die Frage, welche Testfälle aus den a priori teilweise unbekannten Anforderungen an das System abgeleitet werden müssen. Während Ansätze der Breitenerprobung durch Prototypenfahrzeuge zwar theoretisch einen statistischen Nachweis der Sicherheit für diese Funktionen führen können, sind diese in der Praxis aus ökonomischen Gründen nicht durchführbar. Auf der anderen Seite stehen Ansätze des szenariobasierten Testens und Simulationsmethoden, die aber isoliert betrachtet keine Aussage über die erforderliche Testabdeckung liefern können.

Ein möglicher Lösungsansatz für dieses Dilemma wird in dieser Arbeit mit einem neuen Prozess zur realdatenbasierten simulationsgestützten Absicherung entwickelt und prototypisch umgesetzt, um die grundsätzliche Anwendbarkeit des Prozesses zu bewerten. Nach diesem Prozess stützen sich die erforderlichen Testfälle auf Szenarien, die in realen Situationen entstehen und aufgezeichnet werden. Statt dem Einsatz von teuren und spät verfügbaren Prototypen können diese Daten aber auch durch Serien- oder Flottenfahrzeuge aufgezeichnet werden, die mit einer entsprechenden Sensorik ausgestattet sind oder es kann auf Daten-(banken) aus früheren Aufzeichnungen zurückgegriffen werden. Die Szenarien werden mit verschiedenen Ansätzen unter anderem des maschinellen Lernens aus den vorverarbeiteten Sensordaten (wie beispielsweise Objektlisten) extrahiert. Um eine konsistente Darstellung der enthaltenen Szenarien zu ermöglichen, wird ein Szenario-Metamodell für die dynamischen Zusammenhänge mit den Verkehrsteilnehmern entwickelt und verwendet, das es ermöglicht, die extrahierten Szenarien auf Basis der enthaltenen Manöver und Zustände zu clustern und damit in Katalogen zusammen-

zuführen. Die so gebildeten und stetig erweiterbaren Kataloge bilden, mit Auswertekriterien versehen, die Grundlage für einen automatisch ableitbaren Test der zu entwickelnden Fahrfunktion nach dem X-in-the-Loop Vorgehen. Bei beziehungsweise nach der Closed-Loop Testdurchführung kann durch Abgleich der abgeleiteten Szenarien mit den Originaldaten sichergestellt werden, dass diese dem Originalszenario hinsichtlich der zugrundeliegenden Fragestellung entsprechen.

Im Rahmen dieser Arbeit wird das Szenario-Metamodell anhand von simulierten Daten auf Autobahnumgebungen entwickelt und angewandt. Dabei wird auf die erforderlichen Schritte eingegangen, die notwendigen Bestandteile des Modells aus aufgezeichneten Daten zu extrahieren. Schlussendlich wird das Konzept anhand der Anwendung von Realdatensätzen evaluiert.

Abstract

Autonomous driving is currently one of the main drivers of innovation and research in the automotive industry. The development and testing of highly automated driving functions as a preliminary stage of autonomous driving is facing major challenges. On the one hand, conventional test methods such as X-in-the-Loop for proving functional safety are still applicable in principle, but on the other hand they do not by themselves answer the question of which test cases have to be derived from the a priori partially unknown requirements for the system. While approaches of broad testing by prototype vehicles can theoretically lead to a statistical proof of safety for these functions, in practice they are not feasible for economic reasons. Alternatively, there are approaches of scenario-based testing and simulation methods, which, however, when viewed in isolation, cannot provide any statement about the required test coverage.

A possible solution for this dilemma is developed in this thesis with a new process for real data-based and simulation supported validation and prototypically implemented to evaluate the basic applicability of the process. Following this process the required test cases are based on scenarios that are created and recorded in real situations. Instead of using expensive and late available prototypes, these data can also be recorded by series or fleet vehicles equipped with appropriate sensor technology or even by using data or databases from previous recordings. The scenarios are extracted from the pre-processed sensor data (such as object lists) using various approaches, including machine learning. In order to enable a consistent representation of the contained scenarios, a scenario metamodel for the dynamic relationships with the road users is developed and used, which allows to cluster the extracted scenarios on the basis of the contained maneuvers and states and thus to combine them in catalogues. The resulting catalogues, which can be continuously updated, are provided with evaluation criteria and form the basis for an automatically derivable test of the driving function to be developed according to the X-in-the-Loop procedure. During or after the closed-loop test, the derived scenarios can be compared

with the original data to ensure that they correspond to the original scenario with regard to the underlying problem.

In this thesis the scenario metamodel is developed and applied to highway environments using simulated data. The necessary steps to extract the necessary components of the model from recorded data are described. Finally, the concept of this thesis is evaluated by using real data sets.

Inhaltsverzeichnis

Vorwort

Ich bedanke mich an dieser Stelle sehr herzlich bei meinem Doktorvater Prof. Dr.-Ing. Eric Sax für die Übernahme der Betreuung meiner Dissertation, der jederzeit und stets zeitnah mit konstruktiven Rückmeldungen und Anregungen zur wissenschaftlichen Arbeit und darüber hinaus zur Verfügung stand. Ein weiterer Dank geht an Univ.-Prof. Dr.-Ing. Martin Fellendorf von der TU Graz für die Übernahme des Korreferats und die weitere Prüfungskommission, bestehend aus Prof. Dr.-Ing. Sören Hohmann, Prof. Dr. Ivan Peric und Prof. Dr.-Ing. Laurent Schmalen.

Ein besonderer Dank geht auch an die Firma IPG Automotive GmbH, namentlich und allen voran Steffen Schmidt und Dr. Alexander Schmidt, die mir dieses Projekt ermöglicht und die Umsetzung unterstützt haben, ebenso wie Josef Henning und viele weitere Kollegen aus dem Team und darüber hinaus, die stets Rücksicht und Verständnis für meine in dieser Zeit begrenzte Anwesenheit zeigten.

Des Weiteren bedanke ich mich bei den Kollegen am Institut für Technik der Informationsverarbeitung (ITIV) für die zahlreichen wertvollen und unzähligen Gespräche und Diskussionen, die einen wertvollen Beitrag für meine wissenschaftliche Arbeit darstellten und bei meinen zahlreichen Masteranden, Bacheloranden und Hiwis, die dieses Projekt intensiv begleitet und mitgestaltet haben.

Zu guter Letzt möchte ich mich bei meiner Familie bedanken, deren stetige und unbedingte Unterstützung auf meinem bisherigen Lebensweg dieses Projekt erst möglich gemacht hat, sowie bei meinen Freunden, die sich auch in Phasen größter Arbeitsintensität verständnisvoll und geduldig zeigten.

Karlsruhe, im Oktober 2020

Raphael Pfeffer

1 Einleitung

1.1 Ausgangssituation und Motivation

Das automatisierte Fahren ist aktuell der zentrale Innovationstreiber für die Automobilindustrie. Die Entwicklung und Einführung des automatisierten Fahrens erfolgt dabei nicht schlagartig, sondern evolutionär durch die sukzessive Umsetzung von Fahrerassistenzfunktionen und –systemen, die einen immer höheren Grad an Automatisierung aufweisen. Nach der SAE-Norm J3016 werden diese Systeme dabei in sechs Stufen klassifiziert, wobei die höchste Stufe die tatsächliche Vollautomatisierung, also das eigentliche vollständige automatisierte Fahren unter allen Umgebungsbedingungen ohne notwendige Fahrerüberwachung und -eingriff, beschreibt [1].

Trotz der Strategie einer schrittweisen Entwicklung bis zur Einführung des automatisierten Fahrens stellt dies die Branche vor große Herausforderungen. Dies umfasst neben den technischen Aspekten auch unter anderem rechtliche, ethische und soziologische Fragestellungen. Eine immer wesentlichere Anforderung ist dabei die Gewährleistung der funktionalen Sicherheit dieser Systeme, um ein fehlerfreies Verhalten garantieren zu können (vgl. ISO 26262) [2]. Die funktionale Sicherheit von hochautomatisierten Fahrfunktionen, und letztendlich dem automatisierten Fahren, kann dabei nur durch effiziente Testverfahren und eine Anpassung der konventionellen Entwicklungsprozesse in der Industrie sichergestellt werden.

Herkömmliche Test- und Absicherungsmethoden stoßen sowohl an ökonomisch sinnvolle als auch technische Grenzen. So steigt mit dem zunehmenden Automatisierungsgrad auch die Anzahl der zu berücksichtigenden Fahrsituationen. Die Exponentialität dieses Zusammenhangs mündet schließlich in einer sogenannten Freigabefalle. So zeigen Wachenfeld und Winner in [3], dass sta-

tistisch betrachtet knapp 7 Milliarden Testkilometer auf der Autobahn gefahren werden müssten, um allein die Funktion “Autobahnpilot” in Deutschland freigeben zu können.[1]

Die schrittweise Einführung von immer höher automatisierten Fahrfunktionen auf Basis der Übernahme von bereits abgesicherten Komponenten bedeutet dabei nur den ersten Schritt, um dieser Herausforderung zu begegnen. Das Ziel muss es sein, zum einen die für die Freigabe relevanten Testfälle zu identifizieren, um die Zahl redundanter Testszenarien reduzieren zu können, und zum anderen alternative Testmethoden zu finden, die in der Lage sind, Realfahrten oder Teile davon valide zu ersetzen.

In der Vergangenheit haben sich simulationsgestützte Testverfahren wie Model-in-the-Loop (MiL), Software-in-the-Loop (SiL), Hardware-in-the-Loop (HiL) oder auch Vehicle-in-the-Loop (ViL) als erfolgreich erwiesen, um den realen Testaufwand für eingebettete elektronische und mechatronische Steuergeräte und Regelsysteme zu reduzieren und sind feste Bestandteile in den Fahrzeugentwicklungsprozessen vieler OEMs und Zulieferer [4]. Für einzelne Systeme wie beispielsweise die Fahrdynamikregelung (Electronic Stability Control, ESC) können bereits seit einigen Jahren validierte Umgebungen eingesetzt werden, die es ermöglichen, die Freigabe dieser Systeme für Fahrzeugderivate sogar vollständig ohne den Einsatz des realen Fahrversuchs durchzuführen (simulationsbasierte Homologation [5]).

Darüber hinaus begeben sich die langjährig-etablierten Marktakteure in der Automobilindustrie auch technologisch auf neues Terrain. So stellen die damit einhergehenden Anforderungen an das Knowhow und existierende Personal- und Prozessstrukturen diese Akteure vor neue Herausforderungen. Auch wenn Standards wie Automotive SPICE[2] dabei den Stand der Technik für die Prozesse und die Umsetzung von Methoden in der Fahrzeugentwicklung bedeu-

[1] Bei Nutzung einer Prototypen-Fahrzeugflotte würden hierfür 70.000 Fahrzeuge mit einer durchschnittlichen Laufleistung von 1 Mio. km notwendig.

[2] http://www.automotivespice.com/

ten, werden durch diese derzeitigen Trends in der Automotive Softwareentwicklung Ergänzungen durch neue Prozesse und Methoden notwendig sein, um die Konformität auch in Zukunft sicherzustellen.

1.1.1 Marktsituation und Trends in der Automobilbranche

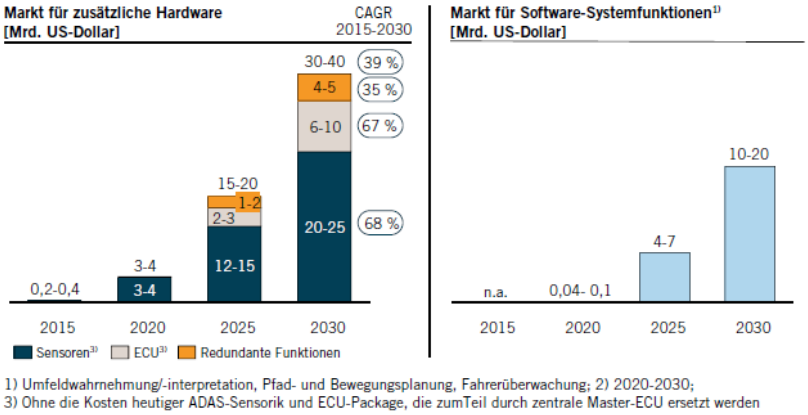
Der globale Markt der Automobilbranche befindet sich branchenübergreifend langzeitlich in einem kontinuierlichen Wachstum. So stieg die Zahl der weltweit produzierten Fahrzeuge beständig von 2001 von 39,8 Millionen Personenkraftwagen (16,5 Millionen Nutzfahrzeuge) bis 73,5 Millionen Personenkraftwagen (23,8 Millionen Nutzfahrzeuge) im Jahr 2017[3].

Zusätzlich ermöglicht aber die Technologie „Autonomes Fahren“ neue Geschäftsfelder. So könnte nach Schätzungen alleine der weltweite Markt für Sensoren bis ins Jahr 2030 auf 20-25 Mrd. US-Dollar anwachsen [6]. Auch der Markt für Software wächst in diesem Kontext in ähnliche Dimensionen (vgl. Abbildung 1.1). Nach einer Studie von McKinsey&Company von 2017 wurden allein in dem Zeitraum ab 2010 Unternehmensakquisitionen im Zusammenhang mit künstlicher Intelligenz für autonomes Fahren mit einer Investitionssumme von 33,5 Mrd. $ getätigt [7].

Darüber hinaus initiiert diese Technologie auch gänzlich neue Geschäftsmodelle. Im Nutzfahrzeugumfeld und Transportwesen werden die Fahrer vollautomatisierter Fahrzeuge obsolet. Aber auch für den Privat-PKW-Bereich stehen den Nutzern durch vollautomatisiertes Fahren plötzlich die bisher „unproduktive“ Zeit des Autofahrens zur Verfügung, die beispielsweise durch den Gebrauch des Internets genutzt werden kann. In [6] wird allein daraus ein Gegenwert des automatisierten Fahrens innerhalb der Fahrzeuglebensdauer von 300 – 400 $ errechnet, was wiederum die Aktivitäten von Unternehmen wie Google / Waymo als neue Marktakteure in der Automobilbranche in diesem Zusammenhang in ganz anderem Licht erscheinen lässt. In einer Studie des Fraunhofer IAO von 2016 wurde eine monatliche Zahlungsbereitschaft

[3] http://www.oica.net/category/production-statistics/2017-statistics/

von über 100 € pro Fahrer für die definierten Bedürfniskategorien „Kommunikation, Produktivität, Grundbedürfnisse, Wohlfühlen, Information und Unterhaltung“ für die frei gewordene Zeit im Fahrzeug ermittelt [8].



1) Umfeldwahrnehmung/-interpretation, Pfad- und Bewegungsplanung, Fahrerüberwachung; 2) 2020-2030;
3) Ohne die Kosten heutiger ADAS-Sensorik und ECU-Package, die zumTeil durch zentrale Master-ECU ersetzt werden

Abbildung 1.1: Erwartetes Marktwachstum aus automatisiertem Fahren (nur “On-Board”-Systeme) [6]

1.1.2 Testkomplexität und Freigabefalle für hochautomatisiertes Fahren

Dem großen marktwirtschaftlichen Potential für automatisiertes Fahren stehen enorme technische Herausforderungen gegenüber. Die Integration neuer Technologien erfordert wiederum einen größeren Testaufwand in der Entwicklung neuer Funktionen. Der insgesamt zunehmende Testbedarf bei modernen Fahrzeugen kann durch verschiedene Ursachen begründet werden. Ein Grund liegt in dem Trend der zunehmenden Anzahl an Steuergeräten und Funktionen pro Fahrzeug, der sich kontinuierlich über die letzten Jahrzehnte beobachten lässt. Dies kann im Wesentlichen durch die gestiegene Leistungsfähigkeit der verwendeten Mikrocontroller erklärt werden, die viele neue Funktionen und Anwendungen erst ermöglichen. So haben Fahrzeuge bereits heute bis 100 Steuergeräte und hunderte von Sensoren/Aktoren [9]. Allgemein wird erwartet, dass die Anzahl der Funktionen im Fahrzeug auch in Zukunft weiter zunehmen

wird, auch wenn die Anzahl der Steuergeräte dafür nicht notwendigerweise steigen muss. Die Anzahl der Steuergeräte lässt sich sogar reduzieren, wenn zukünftig vermehrt Software-Funktionen, die bisher durch separate Steuergeräte realisiert wurden, nun auf leistungsfähigeren Steuergeräten zusammengefasst werden. Die beschriebenen Zusammenhänge nach [10] sind in Abbildung 1.2 dargestellt.

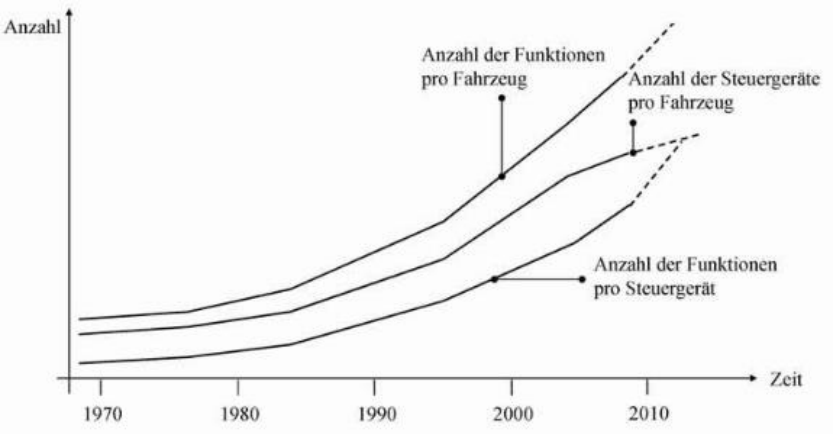


Abbildung 1.2: Funktionen und Steuergeräte pro Fahrzeug [10]

Neben der rein quantitativen Zunahme der Funktionen und dem damit einhergehenden Aufwand für den Test der Funktionen und ihrer Vernetzung, ist es sinnvoll, die Eigenschaften neuer Funktionen hinsichtlich ihres Testaufwands genauer zu betrachten. Funktionen des automatisierten Fahrens stützen sich dabei auf eine Vielzahl an Sensoren, die eingesetzt werden, um die Umwelt wahrzunehmen und zu interpretieren.

Die zu realisierenden Funktionen in Software und Hardware werden in diesem Kontext betrachtet zunehmend komplexer, was sich wiederum im zu leistenden Testaufwand widerspiegelt. Bereits seit einigen Jahren kann man moderne Fahrzeuge auch als komplexe Softwareprodukte betrachten. So befanden sich bereits 2010 in einem Oberklasse-Fahrzeug über 100 Millionen Zeilen Code und damit um Faktoren mehr als beispielsweise im Boeing 787 Dreamliner

(6,5 Millionen Zeilen Code) oder dem Betriebssystem Windows 7 (40 Millionen Zeilen Code) [11].

Bei der Betrachtung aktuell verfügbarer Fahrerassistenzsysteme und –funktionen werden die Herausforderungen für den Test und die Absicherung deutlich. Vergleicht man die Einflussparameter auf das zu betrachtende System, so zeichnen sich aktuelle Systeme wie beispielsweise ein Spurhalteassistenzsystem im Vergleich zu herkömmlichen Elektronikregelsystemen durch eine deutlich höhere Anzahl verschiedener Einflüsse aus, die in großen Teilen auf die Umweltkomplexität zurückzuführen sind. Die dadurch entstehende Vielfalt verschiedenartiger Szenarien gilt es letztendlich zu beherrschen und die Beherrschbarkeit durch geeignete Testmethoden und –verfahren nachzuweisen. Tabelle 1 zeigt zur Verdeutlichung eine einfache Gegenüberstellung möglicher Einflussparameter für ein generisches Spurhalteassistenzsystem zu einem herkömmlichen Elektronikregelsystem ohne Anspruch auf Vollständigkeit:

Querdynamikregler / Electronic Stability Control (ESC)	**Spurhalteassistenzsystem (Lane Keeping Assist)**
- Fahrbahneigenschaften (z.B. Längs-, Querneigung) - Fahrereigenschaften / Wechselwirkung mit dem System - Wetterbedingungen (als Einfluss auf Fahrbahnreibwert)	- Fahrbahneigenschaften (z.B. Längs-, Querneigung) - Fahrereigenschaften / Wechselwirkung mit dem System - Wetterbedingungen (wie bei ESC aber plus weitere wie z.B. Nebelsichtweite, Tageszeit, Belichtungsverhältnisse) - Umweltbedingungen - Verkehr und Hindernisse - Fahrbahnmarkierungen / Fahrstreifen und Fahrstreifenbreiten / Bebauungen - Verkehrszeichen

Tabelle 1: Einflussgrößen auf elektronische Regelfunktionen

Geht man davon aus, dass herkömmliche Elektronikregelsysteme (wie hier am Beispiel des Electronic Stability Controls) unabdingbar für die Sicherheit des

Fahrzeugs sind, ist naheliegend, dass ihre Funktion auch unter Zunahme weiterer (Assistenz)-Funktionen wie dem Spurhalteassistenzsystem gewährleistet sein muss. Zumindest unter dem Blickwinkel einer fehlerlosen Kombination der Systeme, kann man also die Einflussparameter für das Verhalten einer ESC-Funktion als Teilmenge für das erwähnte Spurhalteassistenzsystem betrachten.

Verfolgt man die schrittweise Entwicklung bis hin zum vollautomatisierten Fahren als einen evolutionären Ansatz und damit als Synthese verschiedener Systeme, nimmt somit die Komplexität als Funktion der Einflussparameter mit jeder weiteren Stufe der Automatisierung zu.

Zielt man auf eine möglichst vollständige Testabdeckung ab, wird die Herausforderung für das Testen mit dem Ziel der Freigabe hochautomatisierter Fahrfunktionen deutlich. Durch die Menge verschiedener Einflussparameter auf die Fahrfunktion bedingt durch Umwelteigenschaften erhält man durch einfache Kombinatorik eine große Anzahl notwendiger Testfallvariationen. Ein Beispiel zeigt Abbildung 1.3.

Regen / Schnee / Klar:	3 Variationen	6.561 Variationen zu einem einzigen Szenario
Nebelsichtweite:	3 Variationen	
Tag / Dämmerung / Nacht:	3 Variationen	
Sonnenstand:	3 Variationen	
Temperatur:	3 Variationen	
Luftdruck:	3 Variationen	
GPS-Empfang:	3 Variationen	
Fahrbahnreibwert:	3 Variationen	

Abbildung 1.3: Kombinatorische Testfallexplosion für ein fiktives Szenario einer hochautomatisierten Fahrfunktion

Dieses Beispiel zeigt, dass selbst bei einer einfachen Diskretisierungsbetrachtung von nur drei Ausprägungen einer eigentlich kontinuierlichen Variablen wie der Temperatur in Verbindung mit der Vielzahl der zu betrachtenden Variablen bereits eine große Variationsvielfalt möglicher Situationen allein durch die Umweltbedingungen entsteht. Kombiniert man dies darüber hinaus mit den

möglichen Verkehrssituationen durch andere Verkehrsteilnehmer, entsteht ein kaum abzudeckender Raum an Kombinationen (vgl. Kapitel 4.1.2). Eine vollständige Testabdeckung ist damit für derartige Funktionen kaum realisierbar und kann als Methode zur Bewertung der Funktionsgüte nicht angewendet werden.

Um derartige Funktionen schlussendlich als Produkte freigeben und zulassen zu können, muss deren ausreichende Sicherheit nachgewiesen werden können. In der Vergangenheit konnte der Nachweis für die funktionale Sicherheit für herkömmliche Systeme durch reale Erprobungsfahrten erbracht werden, da diese die höchste Validität bei bisher vertretbarem ökonomischen Aufwand aufwiesen. Für das (hoch)automatisierte Fahren ist diese Methode allerdings nicht mehr ökonomisch sinnvoll. Eine mögliche Metrik zur Bewertung der Sicherheit des automatisierten Fahrens ist die Betrachtung, ob durch den Einsatz des Systems der Schaden, gemessen an der Anzahl der Verletzten und Getöteten, nicht größer ist als ohne den Einsatz der automatisierten Fahrfunktionen.[4] Anhand der Betrachtung des Status Quo nach der durchschnittlich zwischen zwei Unfällen mit Getöteten in Deutschland 210 Millionen Kilometer Fahrleistung liegen, konnte in [3] statistisch gezeigt werden, dass 6,62 Milliarden Testkilometer auf der Autobahn abgeleistet werden müssten, um signifikant nachweisen zu können, dass die oben dargestellte Bedingung erfüllt wird. Winner bezeichnet dieses ökonomische Dilemma als Freigabefalle für das automatisierte Fahren.

1.2 Fragestellung, Ziel und Abgrenzung der Arbeit

Zusätzlich zur realen, physischen Breitenerprobung sind für Entwicklung und Test hochautomatisierter Fahrfunktionen herkömmliche Methoden wie X-in-the-Loop zwar einerseits grundsätzlich weiterhin anwendbar, beantworten je-

[4] Ob die Erfüllung dieser Hypothese für eine allgemeine Akzeptanz der Systeme durch die Gesellschaft bereits ausreicht, muss separat betrachtet werden.

doch andererseits per se nicht die Frage, welche Testfälle aus den a priori teilweise unbekannten Anforderungen an das System abgeleitet werden müssen. Ein möglicher Lösungsansatz für dieses Dilemma wird in dieser Arbeit mit einem neuen Prozess zur realdatenbasierten simulationsgestützten Absicherung entwickelt und prototypisch umgesetzt.

Auch wenn eine statistische Absicherung auf Basis realer Testfahrten aufgrund des ökonomischen Aufwands nicht zielführend scheint, stellt sich die Frage ob die Menge an Fahrszenarien, die im Laufe dieser Tests gefahren würden, nicht auch anderweitig gesammelt und analysiert werden können – beispielsweise durch den Einsatz von Flotten- oder Serienfahrzeugen, die zwar nicht mit der zu testenden Funktion ausgestattet sind, aber bereits eine vergleichbare Sensorkonfiguration und die Möglichkeit zur Datenaufzeichnung haben.

Es wird ein Ansatz entwickelt, der es ermöglicht, repräsentative Fahrszenarien aus Aufzeichnungen mit Serienfahrzeugen zu extrahieren, um sie anschließend zeit- und kosteneffizienteren Testmethoden im Labor und in der Simulation zur Verfügung stellen zu können. Exemplarisch wird vorgestellt, wie eine Umsetzung ausgehend von einer Datenaufzeichnung bis zur Generierung des Testfalls in einer Simulation aussehen kann.

Explizit nicht Bestandteil sind Validierungsmethoden, mit dem Ziel einen Nachweis über die Güte von Simulationen, Modellierungen von Komponenten oder ihrer Parametrierungen zu führen.

1.3 Gliederung der Arbeit

Die vorliegende Dissertation ist wie folgt gegliedert:

Nach diesem Einleitungsteil schließt das Kapitel zwei mit für das Verständnis dieser Arbeit notwendigen Grundlagen zum automatisierten Fahren, Elektronik im Automobil sowie Entwicklung und Test derselben an. Anschließend folgt die Analyse des aktuellen Stands der Technik zum Test von hochautomatisierten Fahrfunktionen und potentiellen Freigabemethoden sowie dem Stand der Forschung. Kapitel vier leitet die zentrale Forschungsfrage für diese Arbeit

ab und erläutert die zentralen Annahmen, die ihr zugrunde liegen. In Kapitel fünf wird das Grobkonzept zur Lösung der Forschungsfrage vorgestellt, bevor Kapitel sechs auf die Detailmodellierung und verwendete Methoden eingeht. Anschließend folgen als Proof-of-Concept die Umsetzung zunächst anhand synthetisch erzeugter Daten aus einer Simulation in Kapitel sieben und schließlich exemplarisch auf Realdatenebene in Kapitel acht. Die Dissertation wird mit einer kurzen Zusammenfassung und einem Ausblick auf zukünftige Forschungsthemen im letzten Kapitel abgerundet.

2 Grundlagen

Nachfolgend werden allgemeine Grundlagen erläutert, die für das Verständnis dieser Arbeit notwendig sind.

2.1 Fahrerassistenz und automatisiertes Fahren

Wie bereits in der Einleitung beschrieben, nimmt die Anzahl und Relevanz der Fahrerassistenzsysteme und Systeme automatisierter Fahrfunktionen stetig zu. Im gleichen Zuge steigt der Automatisierungsgrad der verfügbaren Systeme. So übernehmen moderne Systeme nicht nur den Fahrer assistierend warnende Funktionen, sondern mehr und mehr Teilfunktionen der Fahraufgabe selbst (beispielsweise hinsichtlich der Längs- oder Querführung des Fahrzeugs) oder führen das Fahrzeug selbstständig und aktiv in ausgewählten und definierten Situationen (z. B. Staupilot). In diesem Abschnitt werden die Begrifflichkeiten abgegrenzt, Möglichkeiten zur Klassifikation dieser Systeme aufgeführt und die aktuelle Gesetzeslage vorgestellt.

2.1.1 Klassifikation der Systeme

Der sprachliche Begriff „Fahrerassistenzsystem“ ist ohne weitere Einschränkung zunächst umfassend und allgemein. Wie beispielhaft von Maurer dargestellt wird, kann unter einem „technisches System“, das den „Fahrer des Kraftfahrzeugs“ in Form von „Beistand“ oder „Mithilfe“ unterstützt, bereits ein Hilfsmittel wie ein Tachometer oder eine automatische Blinkerrückstellung verstanden werden [12].

Je nach Fokus in der Fachliteratur werden Assistenzsysteme in verschiedene Kategorien eingeteilt. So lassen sich Assistenzsysteme beispielsweise danach kategorisieren, ob sie unfallvermeidend oder die Unfallfolgen abmildernd ausgelegt sind (aktive vs. passive Sicherheit) oder eher einer Sicherheits- bzw.

Komfortfunktion zugeordnet werden können. In [13] findet sich eine Übersicht über verwendete Kategorien wieder und fasst die unterschiedlichen Ansätze allgemein in fünf übergeordnete Klassen für Assistenzsysteme zusammen:

1. Navigationssysteme: Auf Kartendaten und Telematik basierend informierende Komfortfunktionen ohne aktiven Eingriff in das Fahrgeschehen

2. Stabilisierungssysteme: Stabilisierende Systeme auf Basis interner Sensoren (z. B. Gieraten- oder Drehzahlsensor) mit aktiven Eingriffen zur Erhöhung der Fahrstabilität (z. B. Electronic Stability Control, ESC)

3. Collision Mitigation System: CMS sind Systeme, die die Unfallfolgen abmildern und werden erst unmittelbar vor einem nicht mehr zu verhindernden Unfall aktiv. Systeme, die Rettungsmaßnahmen einleiten oder vereinfachen, fallen ebenfalls in diese Klasse.

4. Driver Assistance System: Ein DAS ist ein System, das auf Basis von Sensordaten aus dem Fahrzeugumfeld den Fahrer warnt oder informiert.

5. Advanced Driver Assistance System (ADAS): Diese Systeme greifen als Untermenge der DAS zusätzlich aktiv in die Fahrzeuglängs- oder Querführung ein.

Da in dieser Arbeit im Wesentlichen Fahrfunktionen behandelt werden, sind nach dieser Klassifikation insbesondere die letzten beiden Klassen von Relevanz und werden im Folgenden genauer beschrieben.

Eine spezifischere Definition von Fahrerassistenzsystemen (Driver Assistance Systems) wird in den Code-of-Practice im RESPONSE 3 Projekt aufgestellt [14], nach dem es die Aufgabe des Fahrerassistenzsystems ist, den Fahrer in der primären Fahraufgabe zu unterstützen. Die Systeme können dabei informierend, warnend, Komfort erhöhend oder Feedback gebend wirken. Sie übernehmen niemals vollständig die Fahraufgabe und die Verantwortung bleibt stets beim Fahrer. Das auch im deutschen Sprachgebrauch verbreitete Akro-

nym „ADAS“ (Advanced Driver Assistance System) wird als Teilmenge dieser Fahrerassistenzsysteme betrachtet, bei denen folgende Eigenschaften erfüllt sein müssen [14]:

- Unterstützung des Fahrers bei der primären Fahraufgabe
- aktive Unterstützung der Quer- und/oder Längsführung mit oder ohne Warnungen
- Erfassung und Bewertung der Fahrzeugumgebung
- Nutzung komplexer Signalverarbeitung
- direkte Interaktion zwischen Fahrer und System

Zusammengefasst sind damit Funktionen beschrieben, die mittels maschineller Wahrnehmung die Umwelt des Fahrzeugs erfassen und interpretieren, sowie auf Basis dieser Information eine primäre Fahraufgabe oder einen Teil dieser Fahraufgabe zeitlich begrenzt oder in bestimmten Situationen ausführen. Sie erfüllen diese Aufgabe redundant-parallel zum menschlichen Fahrer, wobei dieser die Ausführung zu überwachen hat.

Neben den vorgestellten Ansätzen zur Klassifikation der Fahrerassistenzsysteme definiert die SAE J3016 sechs Stufen für den Grad der Automatisierung [1] (siehe Abbildung 2.1). Ähnlich bestimmt auch die Bundesanstalt für Straßenwesen (BASt) in [15] analog dazu den Grad der Automatisierung in fünf Stufen und lässt dabei die letzte Stufe „Vollautomatisierung unter jeder Bedingung“ aus. Die zuvor beschriebenen Definitionen für DAS beziehungsweise ADAS lassen sich nach diesem Schema je nach Ausprägung in die SAE-Automatisierungsstufen 0-3 einordnen. Die Stufen 4 (hochautomatisiertes Fahren) und 5 (vollautomatisiertes Fahren) können in diesem Sinne nicht mehr als ADAS bezeichnet werden, da hier der menschliche Fahrer nicht mehr als Rückfallebene vorhanden sein muss.

SAE level	Name	Narrative Definition	Execution of Steering and Acceleration/ Deceleration	*Monitoring* of Driving Environment	Fallback Performance of *Dynamic Driving Task*	System Capability *(Driving Modes)*
***Human driver* monitors the driving environment**						
0	**No Automation**	the full-time performance by the *human driver* of all aspects of the *dynamic driving task*, even when enhanced by warning or intervention systems	Human driver	Human driver	Human driver	n/a
1	**Driver Assistance**	the *driving mode*-specific execution by a driver assistance system of either steering or acceleration/deceleration using information about the driving environment and with the expectation that the *human driver* perform all remaining aspects of the *dynamic driving task*	Human driver and system	Human driver	Human driver	Some driving modes
2	**Partial Automation**	the *driving mode*-specific execution by one or more driver assistance systems of both steering and acceleration/ deceleration using information about the driving environment and with the expectation that the *human driver* perform all remaining aspects of the *dynamic driving task*	**System**	Human driver	Human driver	Some driving modes
***Automated driving system* ("system") monitors the driving environment**						
3	**Conditional Automation**	the *driving mode*-specific performance by an *automated driving system* of all aspects of the dynamic driving task with the expectation that the *human driver* will respond appropriately to a *request to intervene*	System	**System**	Human driver	Some driving modes
4	**High Automation**	the *driving mode*-specific performance by an automated driving system of all aspects of the *dynamic driving task*, even if a *human driver* does not respond appropriately to a *request to intervene*	System	System	**System**	Some driving modes
5	**Full Automation**	the full-time performance by an *automated driving system* of all aspects of the *dynamic driving task* under all roadway and environmental conditions that can be managed by a *human driver*	System	System	System	**All driving modes**

Abbildung 2.1: Level der Automatisierung (Quelle: SAE J3016 [1])

Stand Ende 2019 befindet sich eine Vielzahl von Systemen verschiedener Hersteller bis Level 2 auf dem Markt. Freigegebene Level 3 Funktionen existieren derzeit noch nicht, auch wenn Systeme wie Audis Staupilot bereits technisch nahe an der Serienreife sind [16]. Ähnliche Systeme sind beispielsweise bei BMW, Volvo oder auch Tesla zu finden [17] [18]. Aktuell existieren zumindest in Europa weiterhin gesetzliche Hürden, die die Fahrzeugzulassung von Fahrzeugen mit Level 3 Funktionen (und höher) abseits von dezidierten Testfeldern erschweren.

2.1.2 Gesetzeslage

Geeignete rechtliche Rahmenbedingungen sind eine Grundvoraussetzung, um das automatisierte Fahren vorantreiben zu können. Das betrifft sowohl die nationale als auch internationale Gesetzgebung. Die Herausforderung besteht vor allem darin, dass historisch betrachtet, eine Übernahme von Fahraufgaben

durch ein automatisiertes System nie von Relevanz war und daher auch gesetzlich nicht betrachtet wurde. So entwickelte sich ein wesentlicher Bestandteil der Gesetzgebung auf Basis des „Wiener Übereinkommen für den Straßenverkehr“ von 1968. In diesem Vertrag wird in seiner ursprünglichen Form geregelt, dass der Fahrer zu jedem Zeitpunkt und in jeder Situation die Kontrolle über sein Fahrzeug haben muss. Teilt man die Fahrerassistenzsysteme nach ihrer Wirkweise ein, kann man drei verschiedene Kategorien von Systemen abgrenzen [19] (siehe Abbildung 2.2).

Kategorie A: **Informierende und warnende Funktionen**	**Kategorie B:** **Kontinuierlich automatisierende Funktionen**	**Kategorie C:** **Eingreifende Notfallfunktionen (unfallgeneigte Situation)**
Wirken ausschließlich „mittelbar“ über den Fahrer auf die Fahrzeugführung	Haben unmittelbaren Einfluss auf die Fahrzeugsteuerung (bewusste Übertragung durch den Fahrer – arbeitsteilige Ausführung). Immer übersteuerbar, i.d.R. Komfortfunktionen	Haben unmittelbaren Einfluss auf die Fahrzeugsteuerung in unfallgeneigten Situationen, die der Fahrer faktisch nicht mehr kontrollieren kann (i.d.R: Sicherheitsfunktionen)
Gestaltungsbeispiele: • Verkehrszeichenassistenz (bspw. Anzeige der Geschwindigkeitsbegrenzung) • Spurverlassenswarnung (bspw. Vibration am Lenkrad)	Gestaltungsbeispiele: • Adaptive Geschwindigkeitsregelung (ACC) • Spurhalteassistenz (über Lenkeingriffe)	Gestaltungsbeispiele: • Automatisches Notbremssystem (systeminitiiert) • Ausweichsystem • Nothaltesystem (Fahrer handlungsunfähig)

Abbildung 2.2: Einteilung von Fahrerassistenzsystemen nach Wirkweisen [19]

Systeme der Kategorie A zeichnen sich dadurch aus, dass sie nur informierend oder warnend Informationen an den Fahrer weitergeben. Sie greifen nicht aktiv in die Ausführung der Fahraufgabe ein. Wirken diese Systeme redundant-parallel, liefern dem Fahrer also Informationen, die dieser selbst wahrnehmen kann, stehen sie nicht im Widerspruch zur oben genannten Straßenver-

kehrskonvention. Assistenzsysteme der Kategorie C, eingreifende Notfallsysteme, beeinflussen zwar durch ihr Eingreifen in kritischen Situationen aktiv die Fahrzeugsteuerung, werden aber als übersteuerbar ausgelegt. Des Weiteren sind teilweise Systeme dieser Kategorie wie der Notbremsassistent für Nutzfahrzeuge mittlerweile durch den Gesetzgeber vorgeschrieben (vgl. General Safety-Regulation (GSR) 661/2009). Systeme der Kategorie B sind, solange sie durch den Fahrer übersteuerbar sind, widerspruchsfrei gegenüber der Straßenverkehrsordnung. Dies gilt jedoch nicht für Systeme des hochautomatisierten oder vollautomatisierten Fahrens (SAE Level 3-5), da der Fahrer nach der Definition dieser Systeme nicht mehr in jeder Situation das System und die Fahrsituation zu überwachen hat. Zumindest temporär[5] wurde diese rechtlichen Unvereinbarkeit mit der aktuellen Gesetzeslage in Deutschland durch die Änderung des Straßenverkehrsgesetzes (StVG), Paragraph 1 vom 17. August 2017 aufgehoben. Paragraph 1a Absatz 1 erlaubt nun explizit den Betrieb eines Fahrzeugs mittels einer hoch- oder vollautomatisierten Fahrfunktion. Der Fahrzeugführer ist jedoch weiterhin verpflichtet, bei Aufforderung durch das System, die Fahrzeugsteuerung unverzüglich zu übernehmen. Damit wirkt sich diese Gesetzesänderung vorerst lediglich auf Systeme nach SAE Level 3 aus.

2.2 Elektronik im Automobil

In diesem Abschnitt werden die erforderlichen Grundlagen dargestellt, die aus Komponentensicht für die Realisierung von Regelfunktionen im Fahrzeug notwendig sind.

2.2.1 E/E-Architektur

Die E/E-Architektur umfasst die eingebetteten Systeme im Fahrzeug in ihrer Gesamtheit. Neben den verschiedenen Soft- und Hardwarekomponenten be-

[5] § 1c sieht die Evaluierung dieser Gesetzesänderung nach Ablauf des Jahres 2019 vor

schreibt die E/E-Architektur auch die verschiedenen Betriebsarten, Abhängigkeiten und das dynamische Verhalten des Systems. Darüber hinaus werden auch die Bauraumgestaltung und die Vernetzung der Komponenten definiert.

Abbildung 2.3 zeigt nach [20] ein Modell für die vier Systemebenen einer E/E-Architektur, die unterschiedliche Relevanz für die verschiedenen Stakeholder innehaben (Kunden, Funktionsentwickler, Zulieferer oder Produktstrategen) und die Abhängigkeiten zwischen den Ebenen.

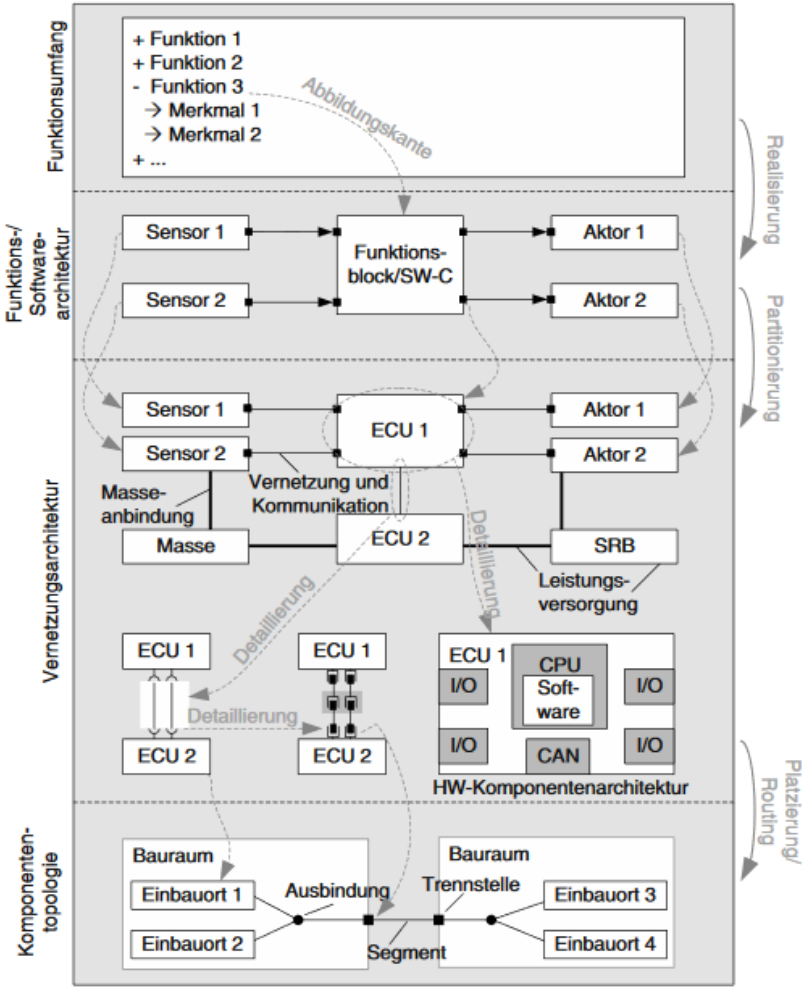


Abbildung 2.3: Vier Systemebenen einer E/E-Architektur nach [20]

In diesem Modell beschreibt die oberste Ebene den Funktionsumfang, der aus den (Kunden-)Anforderungen für das finale Produkt abgeleitet wird, in seinen einzelnen Funktionen und Merkmalen und bildet diese hierarchisch ab. In der

zweiten Ebene wird die Funktions-/ Softwarearchitektur dargestellt. Hier erfolgt die Aufteilung der Funktionsblöcke nach dem EVA-Prinzip (siehe Abbildung 2.4).

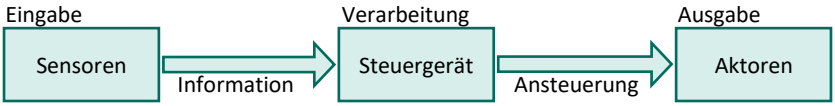


Abbildung 2.4: EVA-Grundprinzip

Die Abhängigkeiten werden in dieser Ebene noch technologieunabhängig beschrieben. Der Umfang der einzelnen Software-Komponenten inklusive ihrer Schnittstellen wird in den Funktionsblöcken definiert. Die dritte Ebene verteilt die Software-Komponenten auf die verschiedenen Hardware-Teilsysteme (Sensoren, ECUs, Aktoren). Neben der Vernetzungsarchitektur, die die erforderlichen Kommunikations- und Datenabhängigkeiten abbildet, wird an dieser Stelle auch das Modell zur Leistungsversorgung der einzelnen Komponenten beschrieben. Die Hardware-Komponentenarchitektur beschreibt darüber hinaus die Zusammensetzung einzelner Steuergeräte (ECUs), Sensoren oder Aktoren. Schlussendlich findet sich auf der vierten und untersten Ebene die Komponententopologie wieder. Hier werden die Einzelkomponenten auf vorgegebene Bauräume aufgeteilt und Leitungswege für Kommunikation, Vernetzung und die Leistungsversorgung geroutet.

Für das hochautomatisierte Fahren ergeben sich darüber hinaus weitere Anforderungen an die E/E-Architektur. So müssen derartige Systeme in der Lage sein große (Sensor-)Datenmengen schnell zu verarbeiten und darüber hinaus eine hohe Sicherheit gegenüber Angriffen von außen bieten. Dabei müssen die Architekturen statt fehlertolerant nun fehleroperativ ausgelegt werden [21], also im Fehlerfall weiter aktiv sein, wenn ein sofortiger Wechsel in den Ruhezustand nicht möglich ist, weil beispielsweise die aktuelle Fahrsituation dies nicht zulässt. Um dies zu gewährleisten sind verschiedene Topologiekonzepte denkbar, die sich beispielsweise in Redundanzen der Hardware oder Software auswirken. Diese Konzepte sind jedoch nicht Fokus dieser Arbeit.

2.2.2 Sensorik und Sensorsysteme

Ein wesentlicher Aspekt für die Realisierung von Fahrerassistenzsystemen und hochautomatisierten Fahrfunktionen stellt die Verwendung geeigneter Sensoren und Sensorsysteme dar. Insbesondere die Umfeldwahrnehmung stützt sich dabei je nach System auf eine oder auch mehrere fusionierte Sensortechnologien. Die Verwendung verschiedener Sensortechnologien hilft, die Unsicherheit hinsichtlich der Komplexität der Umwelt zu reduzieren, indem gezielt die Stärken der verschiedenen Technologien für bestimmte Anwendungen genutzt werden. Gleichzeitig können mehrere Sensoren einer Technologie redundant eingesetzt werden, um beispielsweise die Ausfallsicherheit zu erhöhen. Abbildung 2.5 zeigt gängige Sensortechnologien und ihre schematischen Wirkbereiche im Kontext von Fahrerassistenzsystemen und hochautomatisiertem Fahren.

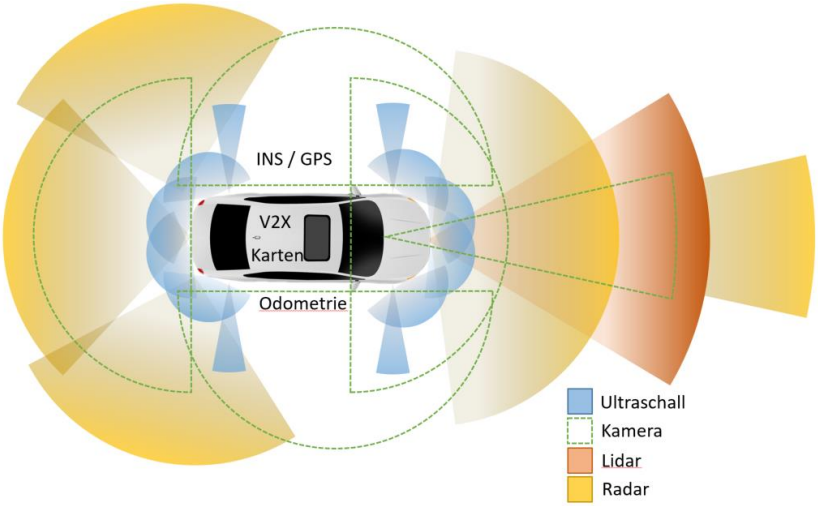


Abbildung 2.5: Sensoren im Automobil (Quelle: eigene Darstellung)

Nachfolgend werden die einzelnen Sensortechnologien detaillierter erläutert:

INS / GPS:

Das GPS (Global Positioning System) ist die Basis für die Positionsbestimmung des Fahrzeugs. Die dreidimensionale geografische Position des Fahrzeugs wird mittels Signallaufzeitenmessung zu mindestens vier Satelliten in der Erdumlaufbahn bestimmt. Das ursprünglich für militärische Zwecke geschaffene System ist seit dem Jahr 2000 für die zivile Nutzung vollständig freigegeben. Seitdem lassen sich mit dieser Technologie Positionsgenauigkeiten in der Ebene von ca. 3-5m erzielen. Die Genauigkeit in der Höhenbestimmung beträgt 10-20 m [22]. Durch Hinzunahme von Differential-GPS (DGPS), einer lokalen Referenzstation, und die Nutzung von Inertialsensorik auf Grundlage fahrzeuginterner Beschleunigungs- und Drehratensensoren lässt sich die Positionsgenauigkeit auf bis zu 10 cm in der Ebene verbessern [19]. So kann beispielsweise auch der GPS-Signalverlust während Tunneldurchfahrten temporär kompensiert werden [23]. Das vollständige System bestehend aus Inertialsensorik und GPS kann als INS (Inertiales Navigationssystem) bezeichnet werden.

Karten und Odometrie:

Zusätzlich zum inertialen Navigationssystem werden für die absolute Positionsbestimmung digitale Straßenkarten verwendet. Die Karten liefern je nach Ausprägung Informationen über die Route, Fahrbahnen, Schilder etc. und können somit als Sensor betrachtet werden. Insbesondere entscheidend für den Einsatz für sicherheitsrelevante Funktionen ist die Gewährleistung der Aktualität der Kartendaten [19]. Für die sogenannte Koppelnavigation kann sich die Ortsbestimmung, zusätzlich zum oben vorgestellten GPS / INS, auf odometrische Sensoren wie den Raddrehzahlsensor oder Lenkradwinkelsensor stützen.

Kamera:

Kameras sind Sensoren für die bildgebende Wahrnehmung der Verkehrsumgebung. Abgesehen von der „Nachbildung“ des menschlichen Sehens bei der Umfelderfassung sind weitere Funktionen möglich, wie die Entfernungsmessung oder Nachtsicht (Infrarot). Darüber hinaus werden Kameras für die Innenraumüberwachung eingesetzt [19]. Für die Umfelderfassung kommen

Frontview, Rückfahrkameras, Surround View-Kameras oder auch Kameras als Spiegelersatz zum Einsatz. Je nach Anwendung unterscheiden sich die Kameras in Parametern wie Blickfeld (horizontal, vertikal), Auflösung, Farbempfindlichkeit und Dynamikumfang. Damit spielen Kameras für die Objekterkennung und –klassifizierung eine bedeutende Rolle.

Ultraschall:

Ultraschall kommt aufgrund seiner technologischen Eigenschaften hauptsächlich im Nahbereich (bis ca. 5 m) zum Einsatz [24]. Mittels Ultraschall werden Entfernungen zu umgebenden Hindernissen durch die Anwendung des Laufzeitprinzips ermittelt. Durch den Einsatz mehrerer Sensoren ist damit auch die Objektlokalisierung möglich. Die Sensorik eignet sich vor allem zum Erfassen statischer Umgebungen bei langsamer Fahrt. Die Hauptanwendungen liegen damit im Bereich der Einparkassistenz [24]. Objekterkennung ist im Gegensatz zur Kamera mit Ultraschallsensorik nur auf kurze Distanzen (ca. 2 m) möglich [25].

Lidar:

Lidar (Light Detection and Ranging) ist ebenfalls ein Verfahren zur Entfernungsmessung und Ortung von Objekten. Im Fahrzeug wird dafür meist, wie beim Ultraschall, das „Time of Flight“ Prinzip angewandt. Beim Lidar werden für die Laufzeitmessung elektromagnetische Wellen im Infrarot, Ultraviolett oder sichtbaren Bereich eingesetzt. Die örtliche Auflösung ist im Vergleich zum Radar sehr gut und kann wenige Zentimeter betragen. Nachteilig ist jedoch der Dämpfungseffekt bei Nebel und schlechter Sicht, wodurch sich die Messreichweite erheblich reduzieren kann. Damit ist der Lidar in Sicherheitsanwendungen nur eingeschränkt verwendbar [26].

Radar:

Beim Radar (Radio Detection and Ranging) werden elektromagnetische Wellen als charakteristische Pulse im Bereich von 24 GHz oder zunehmend 77 – 79 GHz ausgesendet [23]. Durch Verknüpfung der Echo-Ergebnisse der Mo-

dulationszyklen lassen sich Abstands- und Relativgeschwindigkeitsinformationen der Objekte ermitteln. Der Radar eignet sich zur Objektdetektion und –lokalisierung (horizontal) auch auf große Distanzen (im Fernbereich bis 200 m [27]). Der Radar ist auch unter unterschiedlichen Umgebungsbedingungen einsetzbar, für die Objekterkennung und –klassifikation jedoch eher schlechter geeignet.

V2X:

Eine immer größere Bedeutung kommt der Vernetzung der Fahrzeuge untereinander (Vehicle-to-Vehicle – V2V) sowie der Vernetzung der Fahrzeuge mit der Infrastruktur, beispielsweise mit Verkehrszeichen oder Ampelanlagen, zu (Vehicle-to-X – V2X). Dies gilt sowohl für moderne Assistenzsysteme als auch für das automatisierte Fahren und insbesondere im „Mischverkehr“ mit autonomen Fahrzeugen und Fahrzeugen mit menschlichen Fahrern [28]. Mit den über diese Systeme übertragenen Daten werden zusätzliche Informationen zur Umfelderfassung des Fahrzeugs gewonnen, die sich auch außerhalb des Sichtbereichs der im Fahrzeug integrierten Sensoren (wie Radar, Lidar oder Kamera) befinden können und ermöglichen eine umfassendere Abdeckung der Umwelt. Im Sinne der Umfelerfassung kann damit die V2X-Kommunikation wie beispielsweise auch die Kartendaten als weiterer Sensor des Fahrzeugs betrachtet werden.

2.2.3 Aktorik

Für Fahrzeuge die gemäß der in Abschnitt 2.1.1 vorgestellten Klassifikation mit ADAS Funktionen ausgerüstet sind oder Systeme des (teil)automatisierten Fahrens, ist die direkte Ansteuerung der Aktoren für Gas, Bremse und Lenkung wesentlicher Bestandteil des Funktionsprinzips.

Seit einigen Jahren gibt es bereits Systeme wie das elektronische Gaspedal, die elektrohydraulische Bremse oder die elektromechanische Lenkung, die die Grundlage für den direkten Zugriff der Steuergeräte auf die Fahrzeugaktorik schaffen. Dieser sogenannte X-by-Wire Ansatz wird in herkömmlichen Fahrzeugen jedoch in der Regel durch mechanische Komponenten ergänzt, die es

dem Fahrer im Fall des Systemversagens ermöglichen, das Fahrzeug ohne Hilfe des elektronischen Systems sicher zu kontrollieren. Für (hoch)automatisierte Fahrzeuge ist der Fahrer als Rückfallebene nicht vorgesehen. Daher sind Konzepte notwendig, die die Ansteuerung der Aktorik mehrfach redundant erlauben [29]. Als Vorbild kann beispielsweise die Luftfahrtindustrie dienen, die ähnliche Architekturen, wenn auch unter anderen ökonomischen Voraussetzungen, schon seit Jahrzehnten einsetzt.

2.3 Entwicklungsprozess und Testmethoden

In diesem Abschnitt werden der übergeordnete Produktentstehungsprozess, Vorgehensmodelle und Testmethoden erläutert, die für die Entwicklung und den Test insbesondere der elektronischen Komponenten und Systeme im Fahrzeug in der Industrie etabliert sind.

2.3.1 Produktentstehungsprozess

Der Produktentstehungsprozess (PEP) bildet das Rahmenwerk für den übergeordneten Ablauf in der Fahrzeugentwicklung von der Forschung bis zur Serienbetreuung bei zahlreichen Automobilherstellern. In der Gesamtheit müssen oftmals über 1000 Beteiligte in einem Simultaneous-Engineering-Prozess miteinander vernetzt werden, in dem durch die globalen Organisationsstrukturen darüber hinaus häufig Kultur- und Sprachbarrieren zu überwinden sind [30].

Das Vorgehen nach dem PEP ist in getrennte Phasen, die sogenannten Stages, unterteilt und begleitet die Produktentstehung vom Projektbeginn bis zum Start-of-Production (SOP) in herstellerabhängigen Zeiträumen von drei bis fünf Jahren. Vor dem eigentlichen Projektbeginn kann eine Strategie-, Marktanalyse- und Vorentwicklungsphase stehen. Im PEP selbst werden die Stages durch Quality Gates voneinander getrennt, die als fest definierte und verbindliche Termine für die einzelnen Teams und die definierten Arbeitspakete dienen. Die Bewertung der erbrachten Ergebnisse zu diesen Stichtagen ist die Grundlage für die letztendliche Managemententscheidung, ob mit der nächsten

Phase (gegebenenfalls unter Auflagen) begonnen werden kann. Wird die Freigabe verweigert, kann dies unter Umständen zur Gefährdung der Einhaltung des Zieltermins für den SOP führen [31].

Die tatsächlichen Phasen werden je nach Hersteller unterschiedlich bezeichnet. Ein Referenzmodell für den PEP unterscheidet nach Göpfert vier Hauptphasen (vergleiche Abbildung 2.6). Die Phase der Zieldefinition beschreibt die Anforderungen an das zu entwickelnde und zu fertigende Produkt aus der Kundenperspektive. Die Anforderungen werden iterativ konkretisiert, bis ein vollständiger Zielkatalog oder Rahmenheft vorliegt. In der nachfolgenden Phase der Konzeptentwicklung wird produktspezifisch das Lastenheft für das vollständige Fahrzeug entwickelt und dient als Vorgabe für die anschließende Serienentwicklung. Diese beginnt zwei bis drei Jahre vor SOP und wird von den sogenannten Package und Design Freezes begleitet, bei denen das Produkt bis auf die Bauteilebene final definiert wird und als grundlegend für die Lieferantenbeauftragung gilt. Erfolgt am Ende dieser Phase die Freigabe, beginnt mit dem Serienanlauf die Vorbereitung auf den eigentlichen SOP als Startpunkt für die Massenproduktion [31].

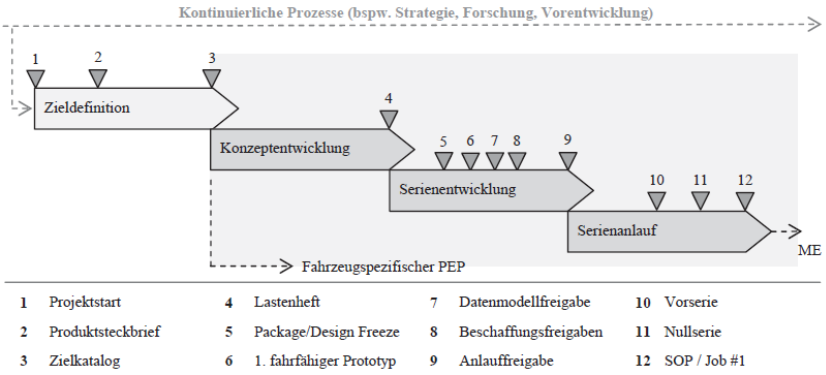


Abbildung 2.6: Referenzmodell für den Produktentstehungsprozess nach Göpfert [25].

2.3.2 V-Modell

Das Vorgehen in der Produktentwicklung nach dem allgemeinen PEP wird durch verschiedene untergeordnete Prozesse und Methoden ergänzt. Diese Vorgehensmodelle dienen dazu, das Entwicklungsvorhaben übersichtlich, planbar und letztendlich beherrschbar zu gestalten [32].

Ein verbreitetes Vorgehensmodell in diesem Kontext, das vor allem im Rahmen der Entwicklung von Hardware-, Elektronik- und Software-Komponenten im Fahrzeug eingesetzt wird, ist das sogenannte V-Modell. Dieses Modell wurde ursprünglich als Prozessmodell für die Software-Entwicklung definiert und 1992 vorgestellt [33]. Der Standard Automotive SPICE referenziert die Implementierung des V-Modells in der Automobilindustrie und die Integration in den übergeordneten Produktentstehungsprozess.[6]

Das V-Modell stellt ein Framework für einen durchgängigen Prozess dar, der alle notwendigen Schritte von der Analyse der Kunden- beziehungsweise Benutzeranforderungen bis hin zum Akzeptanztest des vollständigen Systems beinhaltet und repräsentiert einen vollständigen Zyklus für die Systementwicklung. Die graphische Repräsentation (vgl. Abbildung 2.7) zeigt die zwei „Äste“ des „V“.

Auf der linken Seite, dem absteigenden Ast, werden in einem top-down Prozess zunächst die Systemanforderungen definiert und nachfolgend auf verschiedenen Abstraktionsebenen implementiert. So verbindet das Modell konsekutiv im Durchlaufen des Prozesses die System- mit der Komponentenebene.

Analog folgt für die rechte Seite, dem aufsteigenden Ast, ein bottom-up Prozess, bei dem die zuvor entwickelten Komponenten zunächst einzeln und später im System integriert und getestet werden. Jeder Schritt des Integrierens und

[6] http://www.automotivespice.com/

Testens auf dem aufsteigenden Ast ist dabei einem Schritt auf dem absteigenden Ast zugeordnet (waagerechte Pfeile). Im letzten Schritt wird das Gesamtsystem im Freigabetest gegen die Systemanforderungen getestet.

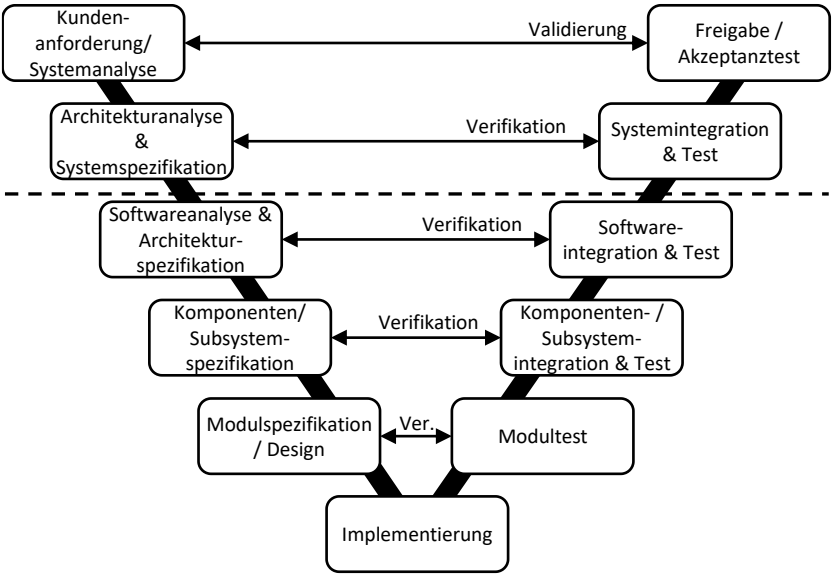


Abbildung 2.7: V-Modell (eigene Darstellung nach [4], [34] und [35])

Durch die Trennung der System- und Komponentenebene kann man das V-Modell auch als Zusammenarbeitsmodell zwischen dem Automobilhersteller (OEM – Original Equipment Manufacturer) im oberen Teil des „V“ und dem System-Zulieferer (Tier-1) im unteren Teil des „V“ betrachten (Trennung beispielsweise an der gestrichelten Linie in Abbildung 2.7). So hat zwar der OEM die Gesamtverantwortung für das Fahrzeug, die Zuständigkeit auf der Komponentenebene kann jedoch auch häufig beim Zulieferer liegen. Für die Zusammenarbeit ist sowohl die Spezifikationsschnittstelle im linken Ast, als auch die Integrationsschnittstelle im rechten Ast des V-Modells zu definieren [10].

Bei Anwendung des V-Modells wird implizit davon ausgegangen, dass die Kunden- und Benutzeranforderungen am Anfang des Entwicklungsprojekts

vollständig und korrekt vorliegen. Durch den chronologischen Aufbau des Vorgehensmodells ist auch nur dann ein reibungsloser Ablauf gewährleistet, zumal erst beim korrespondierenden Schritt in der Integration festgestellt werden kann, ob die jeweilige Spezifikation vollständig und korrekt ist [34]. In der Praxis sind allerdings die Anforderungen zu Beginn nicht immer vollständig erfassbar, sondern werden erst im Laufe des Projekts detailliert erarbeitet. Um dem entgegenzuwirken wird daher ein iteratives oder inkrementelles Vorgehen innerhalb einzelner Schritte des V-Modells gelebt oder das vollständige V-Modell sogar mehrfach durchlaufen [10]. Diese Anwendung des V-Modells wird auch als Multi-Phasen Implementierung bezeichnet. Die einphasige Implementierung sollte hingegen nur zur Anwendung kommen, wenn folgende Bedingungen erfüllt sind [36]:

- Alle Anforderungen sind bekannt und über die Zeitdauer des Projekts stabil.
- Der beste Lösungsansatz kann einfach aus allen Alternativen ausgewählt werden, die darüber hinaus alle bekannt sind.
- Die ausgewählte Lösung ist bekannt und anwendbar (auf allen Hierarchieebenen).
- Die vollständige Systemlösung wird als „first-time-right“ angenommen.

2.3.3 Agile Modelle

Wie zuvor erläutert, ist ein grundsätzlicher Nachteil am V-Modell, dass die Korrektheit der Spezifikation erst spät im Entwicklungsprozess in der Phase der Integrationstests überprüft werden kann [34] [36]. Agile Modelle oder Me-

thoden zielen darauf ab, diese Schwäche zu überwinden und haben insbesondere in den letzten Jahren zunehmend an Aufmerksamkeit und Verbreitung gewonnen.[7]

Ähnlich wie das V-Modell stammen auch die aktuell am meisten verbreiteten Ansätze agiler Modelle wie „Scrum“ oder „Kanban“ aus dem Projektmanagement in der Softwareentwicklung. Sie eignen sich vor allem dann, wenn auf sich ändernde Anforderungen reagiert werden muss, was durch die meist zeitlich sehr kurz ausgelegten Entwicklungszyklen agiler Modelle begünstigt wird. Damit sehen sich agile Modelle als Gegenbewegung zu den eher traditionellen und „schwergewichtigen“ (Software-)entwicklungsprozessen wie dem V-Modell [37].

Inzwischen existieren unzählige Vertreter agiler Modelle, deren Grundidee und Werte auf das Agile Manifest (Manifesto for Agile Software Development) von 2001 zurückgehen.[8] Aus dem Manifest folgend wird Zusammenarbeit und Kommunikation nach innen und außen als essentiell für den Projekterfolg angesehen. Die Prozesse agiler Modelle fokussieren darüber hinaus auf „Ergebnisorientierung, Angemessenheit, Flexibilität und kontinuierliches Lernen“ und folgen einem „iterativ-inkrementellen Entwicklungsprozess“ [38].

Bei der aktuell am weitesten verbreiteten agilen Methode „Scrum“ werden im Sinne der Anforderungsanpassung und Kundenorientierung User Stories abgeleitet, für die der sogenannte Product Owner verantwortlich ist. Das Entwicklungsteam (Scrum Team) entwickelt daraus iterativ und in kurzen Zyklen neue Produktinkremente, die dem Kunden jeweils anschließend zum Testen zur Verfügung gestellt werden. Das Ziel nach diesem Vorgehen ist es, zum Ende jeder dieser Iterationen ein dem aktuellen Stand der Entwicklung entsprechendes funktionsfähiges Produkt vorweisen zu können. Dabei können insbesondere in der frühen Phase der Entwicklung diese Prototypen noch eine sehr einfache Form annehmen und beispielsweise aus Modellen, Mock-ups oder

[7] https://explore.versionone.com/state-of-agile/versionone-12th-annual-state-of-agile-report
[8] https://agilemanifesto.org/iso/de/manifesto.html

Funktionsmustern bestehen. Erst mit der Detaillierung der Anforderungen werden schrittweise Funktionalitäten durch neue Features hinzugefügt [39].

Agile Modelle besitzen ihre Stärke vor allem bei der Anwendung in softwarelastigen Domänen. Methoden wie Scrum sind zwar grundsätzlich auch auf Bereiche wie Hardware-Entwicklung oder die Entwicklung mechanischer Bauteile anwendbar, stehen aber dort vor der Herausforderung, dass kaum innerhalb der kurzen Sprintdauern jeweils funktionsfähige Baugruppen erzeugt werden können, die einen integrierten Test der Gesamtfunktionalität zulassen. Auch herrschen gerade abseits der reinen Software-Entwicklung weitere Randbedingungen vor, die sich beispielsweise durch Designvorgaben, Bauraumbeschränkungen oder die notwendige Integration von Sensor- und Aktorkomponenten ergeben. Neben Verfahren wie Rapid Prototyping sind daher Anpassungen im Prozess erforderlich, wie die Verlängerung der Sprint-Zyklen in diesen Domänen, die Integration in den bestehenden Produktentstehungsprozess und der richtigen Umgang mit Quality Gates [40].

Eine weitere Herausforderung für agile Modelle besteht in der Zusammenarbeit mit Zulieferern oder Dienstleistern. Da zu Beginn des Projektes der vollständige Funktionsumfang des finalen Produktes kaum festgelegt oder dieser durchaus von den eingangs geplanten Vorstellungen abweichen kann, erfordert der Umgang mit Lieferanten einen Paradigmenwechsel in der Zusammenarbeit sowie der vertraglichen Gestaltung und muss im Vorfeld des Projekts klar kommuniziert und abgestimmt werden [40].

2.3.4 Eingesetzte Methoden

Obgleich des zunehmenden Einzughaltens agiler Modelle im Zuge der Herausforderungen im Automotive-Entwicklungsprozesses, wird das V-Modell nach wie vor bei vielen OEMs und Tier-1 Zulieferern in Entwicklungsprojekten im E/E-Umfeld als abteilungs- und domänenübergreifendes Vorgehensmodell verwendet. Agile Vorgehensweisen finden sich vor allem für Teilaufgaben zum Beispiel für die Entwicklung der enthaltenen Softwarekomponenten wieder [35].

Formal betrachtet kann bei strikter Anwendung des V-Modells erst am Ende, beim Freigabetest, überprüft werden, ob die Spezifikation des Gesamtsystems erfüllt wird. Um im Rahmen der Vorgehensweise des V-Modells zu möglichst frühen Zeitpunkten Aussagen über die Qualität der Entwicklung treffen zu können, werden in der Automotive-Entwicklung verschiedene Methoden eingesetzt. Die nachfolgend vorgestellten Methoden sind jedoch nicht nur auf das V-Modell in ihrer Anwendung beschränkt, sondern können auch im Kontext agiler Modelle eingesetzt werden.

Die sogenannte X-in-the-Loop (XiL) Methode [41] (vgl. auch Kapitel 3.1.3) ermöglicht es, die zum jeweiligen Entwicklungszeitpunkt bereits vorhandenen Modelle, Komponenten oder Systeme bereits in diesem Stadium im Kontext des Gesamtsystems zu bewerten. Grundlage dieses Ansatzes ist die Ergänzung beziehungsweise Nachbildung der jeweilig fehlenden restlichen Systeme durch eine Simulationsumgebung. Im linken Ast des V-Modells kommen hierbei Model-in-the-Loop (MiL) und Software-in-the-Loop (SiL) zum Einsatz.

Im Falle von MiL wird die logische Architektur der zu entwickelnden Funktion getestet. Der Entwurf der Funktion ist zu diesem Zeitpunkt noch unabhängig von der verwendeten Hardware im Zielsystem. Um diese Modelle der logischen Architektur zu testen, werden sie in eine vollständige und virtuelle Umgebung des Gesamtsystems eingebettet. Diese Umgebung muss modular sämtliche benötigte Komponenten abbilden, wie beispielsweise Fahrdynamik, Antriebsstrang, Sensoren, Fahrer, Umwelt oder Verkehr. Der Test erfolgt entsprechend den Spezifikationsanforderungen in einem virtuellen Fahrversuch (vgl. auch Abschnitt 3.1.2). Auf diese Weise können frühzeitig Fehler in Bezug auf die Spezifikation aufgedeckt und korrigiert werden, ohne dass der vollständige Prozess durchlaufen werden muss [42]. Analog dazu werden mittels der SiL-Methode bereits implementierte Software-Module getestet.

Auf der rechten Seite des V-Modells wird die Softwarefunktion als reale Komponente abgebildet und auf der Zielhardware implementiert. Die Testumgebung in dieser Phase wird als Hardware-in-the-Loop (HiL) bezeichnet. In diesem Fall muss die Simulationsumgebung die Schnittstellen zu der Hardware, beispielsweise über entsprechende I/O-Messtechnik, aus der virtuellen Umgebung heraus bedienen können (Komponenten-HiL). Nachdem die Integration

und der Test der Einzelkomponenten abgeschlossen ist, werden diese zum Gesamtsystem integriert und ebenfalls mit der HiL-Methode auf das korrekte Zusammenwirken getestet (Integrations-HiL).

Eine neuere Methode in dem XiL-Ansatz mit besonderer Relevanz für Fahrerassistenzsysteme und automatisiertes Fahren stellt die Vehicle-in-the-Loop (ViL) Methode dar. Bei dieser Methode wird das vollständige und reale Versuchsfahrzeug in die virtuelle Umgebung eingebettet. Hierbei wird die Sensorik des Fahrzeugs über eine geeignete Schnittstelle zur Simulation stimuliert oder vollständig ersetzt. Verkehr und weitere Komponenten der Umwelt des Fahrzeugs werden vollständig simuliert. Diese Methode schließt die Lücke zwischen dem Integrations-HiL und der realen Fahrerprobung [43].

Die nachfolgende Abbildung zeigt zusammenfassend die Zuordnung der XiL Methode zu den entsprechenden Entwicklungsphasen im V-Modell:

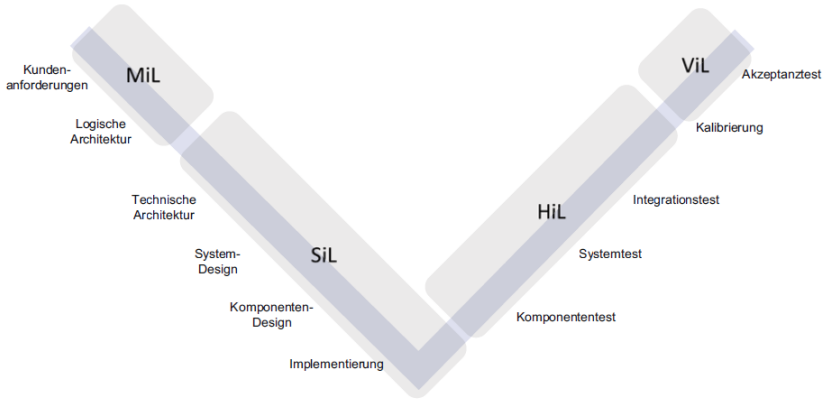


Abbildung 2.8: In-the-Loop Methoden im V-Modell [34]

3 Stand der Technik und Forschung

Der Stand der Technik und Forschung wird hier im Kontext des betrachteten Themengebiets in drei wesentliche Bereiche unterteilt, auf die nachfolgend im Detail eingegangen wird. Erstens die Umsetzung aktueller Testverfahren für hochautomatisierte Fahrfunktionen in der Industrie, zweitens die Darstellung Stand heute angewandter oder diskutierter Ansätze für die Absicherung und Freigabe dieser Funktionen und schließlich drittens, der aktuelle Stand der Forschung in diesen Bereichen.

3.1 Testverfahren für hochautomatisierte Fahrfunktionen

Eine reine physische Fahr- oder Breitenerprobung der Fahrzeuge reicht für einen vollständigen Test hochautomatisierter Fahrfunktionen nicht mehr aus. Bereits für die Mercedes E-Klasse (W212)[9] mussten insgesamt 36 Millionen Testkilometer absolviert werden [3]. Die Anwendung alternativer Testmethoden ist jedoch nicht grundsätzlich neu und wird bereits heute in der Entwicklung zur effizienten Fehlersuche und Testdurchführung in verschiedenen Bereichen praktiziert.

Im Folgenden werden relevante Testmethoden und -konzepte zusammengefasst, die von ihrem Prinzip her auch für den Test von hochautomatisierten Fahrfunktionen eingesetzt werden können:

[9] Dieses Fahrzeug weist nur Systeme nach Level 0 bzw. 1 der SAE-Klassifikation (vgl. Kapitel 2.1.1) auf.

3.1.1 Physische Fahrerprobung

Die physische Fahrerprobung ist, neben dem zuvor erläuterten hohen Testaufwand, mit weiteren Nachteilen verbunden. Aus Betrachtung der Prozessperspektive können die erforderlichen Tests erst relativ spät in der Entwicklung durchgeführt werden – erst dann, wenn die ersten Prototypen zur Verfügung stehen. Des Weiteren sind die Tests an Echtzeitbedingungen geknüpft. Eine zeitliche Raffung der Tests ist nicht möglich, es kann nur über die Anzahl der (teuren) Prototypen skaliert werden. Zudem sind die gefahrenen Tests durch die Vielzahl der Einflussparameter nur bedingt vollständig reproduzierbar und die Sicherheit des beteiligten Personals und auch des Materials muss zu jedem Zeitpunkt sichergestellt sein [44] [45].

3.1.2 Simulationsbasiertes Testen

Simulationsbasierte Testmethoden versuchen, diese Nachteile zu eliminieren (siehe Abschnitt 3.1.1). Das Ziel der Simulation besteht darin, die zu betrachtenden Systeme durch (mathematisch-physikalische) Modelle so abzubilden, dass valide Rückschlüsse auf das Verhalten der Systeme durch die Beobachtung des Modellverhaltens möglich werden [46].

Gelingt die repräsentative Abbildung der betrachteten Systeme, ergeben sich für die Simulation zusammengefasst folgende Vorteile gegenüber der physischen Fahrerprobung (Tabelle 2):

Vorteile simulationsbasierter Methoden	Vorteile der physischen Fahrerprobung
- Vollständige Reproduzierbarkeit der einzelnen Testfälle möglich - Effizienz in der Testdurchführung (Automatisierung, zeitliche Raffung, Parallelisierung) - Verfügbarkeit in jeder Phase des Entwicklungsprozesses - Sichere Durchführung auch kritischer Testfälle - Nahezu vollständige Gestaltungsfreiheit in den Testfällen	- Valides Systemverhalten - Keine Modellierung/Modellparametrierung erforderlich - Sämtliche Einflussfaktoren (bekannte + unbekannte) des Gesamtsystems werden automatisch berücksichtigt - Zusätzliche Berücksichtigung des subjektiven Testeindrucks möglich

Tabelle 2: Simulationsbasierte Methoden und physische Fahrerprobung im Vergleich

Für viele Anwendungen, wie die Auslegung, Bewertung und Erprobung der Kinematik und Dynamik des Fahrwerks und der Lenkung, der Fahrdynamik des Gesamtfahrzeugs, des Fahrkomforts oder der Analyse von Unfallvorgängen sind Methoden der Simulation von Mehrkörpersystemen bereits seit einiger Zeit der Stand der Technik und im Entwicklungsprozess fest verankert [46].

Je nach zu testendem System (System-under-Test bzw. SUT) müssen in der Simulation unterschiedliche Elemente abgebildet werden (vgl. hierzu auch die XiL-basierte Methoden in Abschnitt 3.1.3). Für eine vollständige Testumgebung werden nicht nur Modelle für das Fahrzeug selbst, sondern auch für die Umwelt um das Fahrzeug sowie für den Fahrer benötigt (siehe grüne Elemente in Abbildung 3.1). Nur dann kann die Umgebung des SUT vollständig als Regelkreis abgebildet werden.

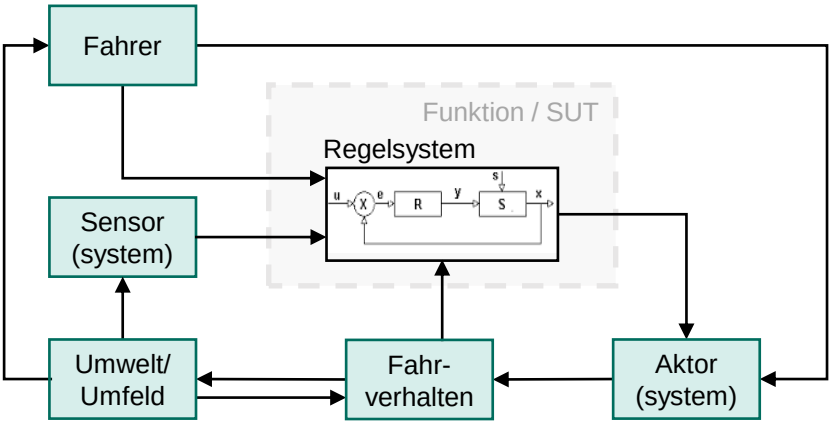


Abbildung 3.1: Schema zur Einbettung eines exemplarischen System-under-Tests in einer Simulation

Die Modellierung der Komponenten des Fahrzeugs und seiner Systeme (z.B. Reifen, Radaufhängung, Bremse, Antriebsstrang, Regelsysteme, Lenkung, Sensoren, Aerodynamik usw.) sowie der Umwelt (z.B. Straße und Verkehr) und des Fahrers erfolgt in der Regel anwendungsfallgetrieben. Dabei muss ein Kompromiss zwischen dem für die Anwendung notwendigen Grad der Detaillierung bei gleichzeitig benötigtem Aufwand für die Berechnung bei der Simulationsdurchführung erreicht werden: So einfach wie möglich, so detailliert wie nötig [47][10]. Aus diesem Grund sind verschiedene Abstraktionsgrade notwendig, die beispielsweise in [48] beschrieben werden (vgl. auch Abbildung 3.2).

[10] Welcher Grad der Detaillierung für welche Komponente in der Simulation erreicht werden muss, kann beispielsweise über eine Sensitivitätsanalyse [214] bestimmt werden.

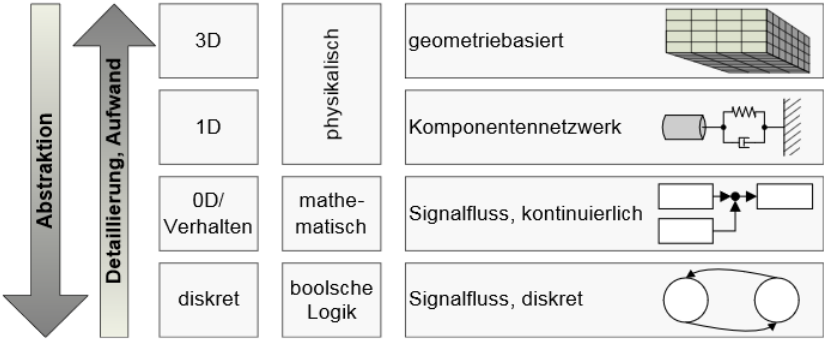


Abbildung 3.2: Abstraktionsebenen der Modellierung [48]

Mit einem steigenden Grad der Detaillierung geht oftmals auch ein erhöhter Aufwand bei der Parametrierung der Modelle einher, die wiederum gesonderte Prozesse und Vorgehensweisen bei der Extraktion dieser Größen aus Messungen am Realsystem oder auch aus CAD-Daten erfordern [49] [47].

Ein weiterer Aspekt ist die Validierung der erstellten Simulationsmodelle und der Parameterkonfigurationen. Nur durch die Durchführung der Modelvalidierung kann die Glaubwürdigkeit und damit die Nutzbarkeit der Modelle als Testumgebungen im Prozess sichergestellt werden. Insbesondere für den Bereich der Fahrdynamiksimulation sind in der Vergangenheit eine Vielzahl an Verfahren für diesen Nachweis untersucht und entwickelt worden. Der aktuelle Stand der Technik der Verfahren wird in [50] untersucht und zusammengefasst. Auch ist Stand heute die Fahrdynamiksimulation der einzige bekannte Automotive-Anwendungsbereich, bei dem die Modellgüte und Validierung eine Genauigkeit erreicht, mit der für Regelsysteme eine Freigabe auf Basis simulationsbasierter Tests erfolgen kann (simulationsbasierte Homologation [5]).

Eine vollständige Testumgebung für den simulationsbasierten Test von Fahrerassistenzsystemen oder Systeme des hochautomatisierten Fahrens fokussiert jedoch nicht nur auf das Fahrzeug, seiner Komponenten oder das fahrdynamische Verhalten allein. Es müssen weitere Komponenten modelliert und

abgebildet werden, die in Interaktion mit dem Fahrzeug stehen. Dazu gehören unter anderem das Fahrerverhalten, statische Objekte aus der Umgebung (z.B. Fahrbahn mit Markierungen, Verkehrsschilder, Ampeln, Bebauungen), dynamische Objekte (z.B. Verkehrsteilnehmer wie PKWs, Nutzfahrzeuge, Fußgänger, Radfahrer) und das unterschiedliche Verhalten der Sensoren (vgl. Abschnitt 2.2.2), mit denen das Fahrzeug respektive das Fahrzeugsystem die Umgebung wahrnimmt.

Insbesondere die Abbildung von Sensoreigenschaften führt zu einem hohen Modellierungs- und Ressourcenaufwand, wenn beispielsweise raytracing-basierte Techniken zum Einsatz kommen, bei denen die Ausbreitung der elektromagnetischen Wellen (Radar) in der virtuellen Umgebung durch physikalisch approximierte Modelle simuliert wird [51]. Cao et al. beschreiben verschiedene Prinzipien der Sensormodellierung von der dort sogenannten White-Box-Modellierung, die versucht jedes Detail des Detektions- und Messprinzips des jeweiligen Sensors zu berücksichtigen, bis hin zur Black-Box-Modellierung, die auf statistischen Repräsentationen und Wahrscheinlichkeitsverteilungen der Messungen beruht. Black-Box-Modellierungen zeichnen sich durch eine vergleichsweise hohe Recheneffizienz bei geringerer Genauigkeit aus [52]. Daher werden heute häufig die aufwändigen aber hochgenauen physikalisch-basierten Simulationen von Sensormodellen auf Grafikprozessor-Architekturen ausgelagert, die sich durch hohe Parallelisierbarkeit auszeichnen und damit eine höhere Simulationsgeschwindigkeit erreichen [53]. Die meisten kommerziellen als auch Open-Source Simulationswerkzeuge wie IPG CarMaker[11], VIRES VTD[12], TASS PreSCAN[13], TESIS DYNA4[14], dSPACE ASM[15], Microsoft AirSim [16] oder Gazebo [17] verwenden eine oder mehrere Abstraktionsebenen in ihren Modellbibliotheken um Sensoren abzubilden

[11] https://ipg-automotive.com/products-services/simulation-software/carmaker/
[12] https://vires.com/vtd-vires-virtual-test-drive/
[13] https://tass.plm.automation.siemens.com/prescan
[14] https://www.tesis.de/dyna4/
[15] https://www.dspace.com/de/gmb/home/products/sw/automotive_simulation_models.cfm
[16] https://github.com/microsoft/AirSim
[17] http://gazebosim.org/

[53]. Zusammenfassend ist am Beispiel des Tools IPG CarMaker ein mehrstufiges Sensorkonzept dargestellt (Abbildung 3.3). Während die „Raw Signal Interfaces“ für die verschiedenen Sensortypen physikalische Reflektionen in Abhängigkeit der modellierten Geometrien in der Umgebung bestimmen (vgl. auch Abbildung 3.2 oben), ermitteln die „idealen Sensoren“ auf Basis stark vereinfachender Modellannahmen Ground Truth Informationen (wie Relativabstand oder –geschwindigkeit). Für diese Modelle mit hohem Abstraktionsgrad werden keine sensorspezifischen Unterschiede berücksichtigt.

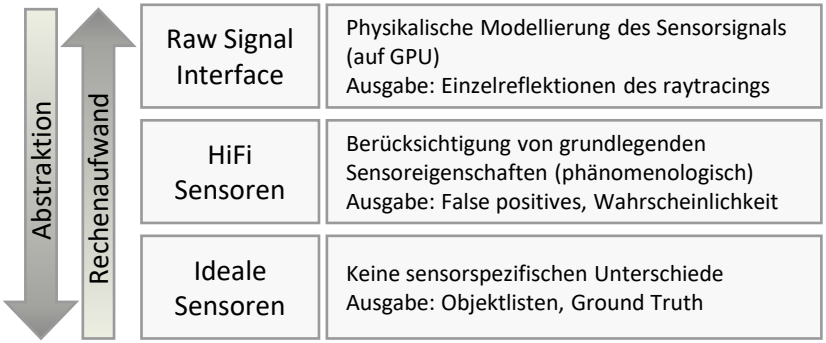


Abbildung 3.3: Abstraktionsebenen für Sensoren im Tool IPG CarMaker, zusammengestellt nach [54]

Auch für die Modellierung des dynamischen Umfelds beziehungsweise des Verkehrs stehen verschiedene Ansätze zur Verfügung. Schuldt kategorisiert in diesem Zusammenhang Verkehrssimulationen ebenfalls anhand der Abstraktionsebenen [55]:

Die makroskopische Verkehrssimulation legt den Fokus auf die Abbildung des Gesamtverhaltens des Verkehrsflusses. Hierbei werden die Fahrzeuge nicht als einzelne Elemente simuliert, sondern auf Basis von Fahrzeugkollektiven modelliert. Die mikroskopische Verkehrssimulation hingegen modelliert das Verhalten einzelner Fahrzeuge. Fahrzeuge, Fahrer und potentielle Assistenzsysteme stellen eine Einheit dar. In der Simulation selbst werden unterschiedliche Geschwindigkeits- und Wegverläufe zwischen den Verkehrsteilnehmern beobachtbar. Mesoskopische Verkehrssimulationen können als Verbindung der

beiden zuvor genannten Verkehrssimulationen genannt werden. Die Modellierung erfolgt auf Basis der Fahrzeugkollektive, es können jedoch auch einzelne Fahrzeuge simuliert werden. Bei der submikroskopischen Verkehrssimulation werden Umgebung und Fahrzeug noch in weitere Teilkomponenten zerlegt. So können beispielsweise Interaktionen zwischen Fahrzeug, Fahrer und Assistenzsystem in der Simulation dargestellt werden. Eine Verbindung von mikroskopischen und submikroskopischen Verkehrssimulationen ist in der Praxis möglich[18]. Die nachfolgende Tabelle fasst die wesentlichen Unterschiede bei den Verkehrssimulationen zusammen (Tabelle 3):

Makroskopisch	**Mesoskopisch**	**Mikroskopisch**	**Submikroskopisch**
Schrittweite: ~10s	Schrittweite: ~1-10s	Schrittweite: ~1s	Schrittweite: < 100 ms
Modellierung: Verkehrsfluss auf Basis von Kollektiven	Modellierung: Kollektiven mit Einzelfahrzeugen	Modellierung: Einzelfahrzeuge als Gesamtsysteme	Modellierung: Fahrzeuge mit Teilkomponenten
Fokus: z.B. Planung neuer Straßen, Staus	Fokus: Fahrstreifengenaue Simulation z.B. für Knotenpunkte	Fokus: z.B. Längs- und Querdynamische Effekte der Teilnehmer	Fokus: z.B. Test von Assistenzfunktionen, Verbrauchsanalysen

Tabelle 3: Klassifikation von Verkehrssimulationen nach [55]

3.1.3 XiL-basierte Methoden

Bereits in Kapitel 2.3.4 wurde die XiL Methode im Kontext des Entwicklungsprozesses eingeführt und anhand des V-Modells dargelegt. Simulationen oder einzelne Simulationsmodelle sind wesentlicher Teil dieser Methode. Nachfolgend werden wichtige Elemente simulationsbasierter Methoden nach dem XiL Ansatz aus Sicht der Testmethodik dargestellt.

[18] Vgl. https://ipg-automotive.com/de/news/article/ptv-vissim-interface-fuer-die-carmaker-produktfamilie/

Closed-Loop und Open-Loop Tests:

Grundlegend kann, nahezu unabhängig von dem letztendlich eingesetzten XiL-Verfahren, zwischen sogenannten Closed-Loop und Open-Loop Tests unterschieden werden. Die Begrifflichkeiten selbst stammen aus dem Fahrversuch, bei dem Open-Loop Tests Versuche beschreiben, bei denen der Testfahrer das Lenkrad fixiert oder gar die Hände („hands off“) komplett vom Lenkrad nimmt (z.B. stationäre Kreisfahrt, Lenkwinkelsprung…). Im Closed-Loop Fall hingegen regelt der Fahrer aktiv eine gewünschte Trajektorie oder allgemein ein bestimmtes Verhalten ein (z.B. Slalom, Doppelter Fahrstreifenwechsel…) [56].

Analog dazu werden in einer Open-Loop Testumgebung im XiL-Kontext die zu testenden Modelle oder Algorithmen mit geeigneten Input-Daten stimuliert. Die vom Testobjekt erzeugten Output-Daten werden analysiert und mit zu erwartenden Ergebnissen abgeglichen. Dieser Ansatz ist häufig einfach zu implementieren und benötigt wenig Aufwand in der Testdurchführung [57]. Open-Loop Verfahren eignen sich vor allem für reaktive Systeme, die keine oder nicht Teil von schnellen Regelkreisen sind und sich im Idealfall vollständig durch Zustandsautomaten beschreiben lassen (z.B. Systeme im Fahrzeug-Innenraum oder Karosserie-Elektronik) [58]. Dem gegenüber steht das Closed-Loop Testen, bei dem die Output-Daten an die simulierten Umgebungsmodelle zurückgespeist werden, um so auch Wechselwirkungen in beide Richtungen des Testobjekts mit der Umgebung untersuchen zu können. Dies erfordert nicht nur einen in der Regel höheren Implementierungsaufwand, sondern stellt auch Anforderungen an die Simulationsmodelle, da diese nun sobald Testkomponenten in Hardware vorliegen, in Echtzeit[19] berechnet werden müssen, um rechtzeitig die erforderlichen Stimuli zu liefern. Für den Open-Loop Test können des Weiteren auch aufgezeichnete Daten, beispielsweise auch aus realen Messungen verwendet werden, was für einen Closed-Loop Test kaum möglich ist, da die Input-Daten zur Testlaufzeit vom Modell in Abhängigkeit vom Test-

[19] Unter Echtzeit wird der Betrieb eines Rechensystems verstanden, bei dem die Datenverarbeitung der anfallenden Daten derart durchgeführt wird, dass die Ergebnisse innerhalb einer vorgegebenen Zeitspanne verfügbar sind [213]

objekt-Output berechnet und bereitgestellt werden müssen. Eine mögliche Lösung für dieses Dilemma wird mit der Reactive-Replay Methode beschrieben, die unter bestimmten Randbedingungen für ausgewählte Anwendungsfälle eine messdatenbasierte Closed-Loop Simulation erlaubt [44].

Model/Software-in-the-Loop:

Model-in-the-Loop kommt in einer frühen Phase des Fahrzeugentwicklungsprozesses zum Einsatz [59][20]. Zur Vorbereitung für diese Testmethode muss nicht nur für die Umgebungskomponenten des zu entwickelnden Systems eine Modellbildung erfolgen, sondern auch das Testobjekt selbst liegt zu diesem Entwicklungszeitpunkt nur als Modell vor [60]. Die Model-in-the-Loop Simulation können häufig auf handelsüblichen PCs durchgeführt werden und können darüber hinaus auch für die Entwicklung von Funktionen eingesetzt werden, die später nicht durch Software realisiert werden [60]. In der nächsten Phase wird der aus den Modellen (automatisch) generierte Code auf einer targetnahen Plattform getestet und von den Umgebungsmodellen getrennt. Diese Stufe wird als Software-in-the-Loop bezeichnet [61] auch wenn im deutschen Sprachgebrauch Software-in-the-Loop als Oberbegriff für die beiden genannten Verfahren verbreitet verwendet wird [62]. Im Gegensatz zur Hardware-in-the-Loop Simulation werden bei MiL/SiL keine realen Steuergeräte oder mechanische Komponenten eingesetzt, sodass die eingesetzten Simulationsmodelle für die Umgebung per se keine Echtzeitanforderungen erfüllen müssen [62]. Dadurch ist es einerseits möglich genaue und rechenzeitintensive Modelle zu verwenden oder aber andererseits den Testdurchsatz durch schnelle Simulationen in mehrfacher Echtzeit zu erhöhen.

Hardware-in-the-Loop:

Hardware-in-the-Loop stellt die zentrale Methode im Stand der Technik für Test und Absicherung von Regler- und Steuergerätealgorithmen dar [63]. Ein Hardware-in-the-Loop Simulator wird dabei als ein Aufbau verstanden, bei dem das zu testende System (System-under-test) oder Subsystem als reale

[20] Amper stellt mit Concept-in-the-Loop ein Verfahren vor, das im Absicherungsprozess noch vor MiL durchgeführt werden kann [209].

Komponente vorliegt und in eine echtzeitfähige Simulationsumgebung integriert wird. Dabei können Teile der Umgebung, wie Aktoren oder Sensoren ebenfalls als Realkomponenten vorliegen, wenn beispielsweise die Modellierung dieser Komponenten zu aufwändig oder eine echtzeitfähige Simulation der Modelle nicht gewährleistet werden kann [64]. Im Automotive-Bereich wurden erste HiL Simulatoren bereits Ende der 1980er für den Test von Regelsystemen wie ABS eingesetzt [65] und sind heute in den Entwicklungsprozessen wesentlicher Bestandteil. Der Einsatz der HiL-Methode bietet dabei zahlreiche Vorteile [66]:

- Kosteneffizienz durch weniger benötigte Hardware verglichen mit Gesamtfahrzeugprototyen
- Schnellere Verfügbarkeit als Fahrzeugprototypen (rapid prototyping)
- Höheres Vertrauen in die Testresultate (verglichen mit reiner Simulation) insbesondere für Komponenten die nicht vollständig oder nur schwer modelliert werden können
- Höhere Simulationsgeschwindigkeit bei Systemen mit komplexen physikalischen Phänomenen als bei reinen Simulationen
- Reproduzierbarkeit der Testergebnisse durch kontrollierbare Testbedingungen
- Abbildbarkeit von Situationen wie Unfälle ohne Zerstörung des Materials
- Größere Bandbreite an Betriebszuständen und Randbedingungen abbildbar
- Keine Gefährdung der Sicherheit der „Testfahrer“ auch beim Test von sicherheitskritischen Systemen

Hinzu kommen die Möglichkeiten für die automatische Erzeugung von Variationen, den automatisierbaren und damit dem Dauerbetrieb (24/7) der Simulatoren beziehungsweise der Testdurchführung [67].

Nachteilig sind hingegen die Aufwände, die notwendig sind, um ein HiL System vollständig einsatzfähig aufzubauen und um sicherzustellen, dass die im HiL Test eingesetzten Modelle und damit das erzielte Ergebnis repräsentativ für den eigentlichen Untersuchungsgegenstand ist.

Für Fahrerassistenzsysteme oder Systeme hochautomatisierten Fahrens stehen HiL Simulatoren vor der Herausforderung, Lösungen für die verschiedenen Sensoren und deren Test bieten zu müssen. Die zu testenden Steuergeräte erfassen die Umgebungen und benötigen dafür im HiL Testsystem eine realistische Simulation der Umwelt, um entsprechende Daten für die Stimuli zu extrahieren.

Für den Test eines radarbasierten Systems beispielsweise müssen dafür in der Umweltsimulation Modelle für Materialeigenschaften, für die Berechnung von Reflexionen oder für die Interpolation von Detektion in Echtzeit vorhanden sein [68]. Für die Schnittstelle zur Signaleinspeisung existieren verschiedene Konzepte mit unterschiedlichen Vor- und Nachteilen. So können zum Beispiel Radarrohdaten oder Objektlisten an geeigneten Einsprungpunkten (Design-for-Test Schnittstellen) im Sensor selbst eingespeist werden (vergleiche Abbildung 3.4) [68]. Dies hat den Vorteil, dass über diese Schnittstellen eine nahezu beliebig große Anzahl an Detektionen oder Zielen übertragen werden kann, sofern die Simulationsberechnung diese in Echtzeit ausgibt. Alternativ dazu kann der vollständige Radarsensor in ein HiL-System eingebunden werden, um das gesamte Sensorverhalten real abzubilden. Dazu sendet der Radarsensor seine Signale aus, die von den Antennenkonstruktionen des HiL-Systems „over-the-air“ absorbiert und entsprechend der Testszenarien in der Simulation reflektiert werden. Nachteilig ist, dass die Realisierung eines derartigen Systems in der Regel deutlich aufwändiger ist und durch baulich/mechanische Randbedingungen nur auf die Simulation einer kleinen Zahl an Radarzielen beschränkt ist und nur eine beschränkte Mindestentfernung darstellen kann [69]. Ähnliche Lösungskonzepte werden in [70] und [71] vorgestellt.

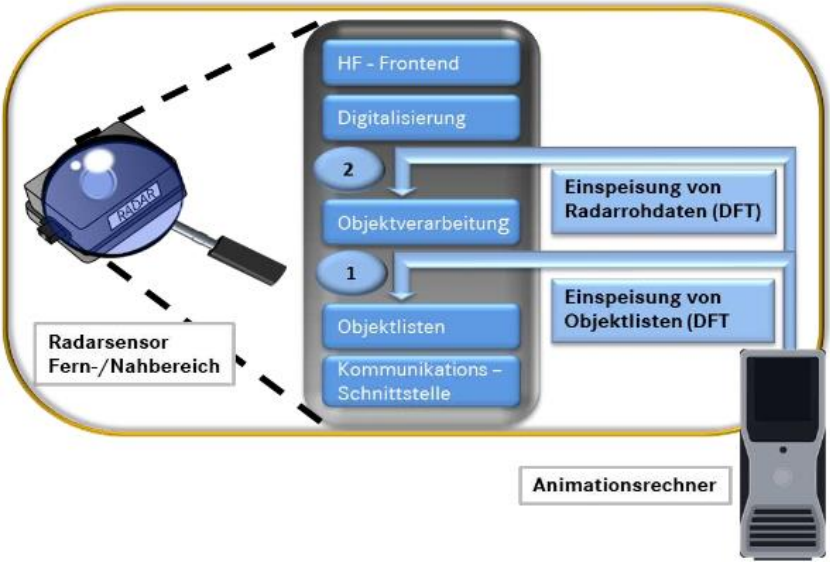


Abbildung 3.4: Design-for-Test Schnittstellen beim Radar [68]

Für ultraschall-, lidar- und kamerabasierte Systeme, existieren vergleichbare Lösungsprinzipien für HiL Simulatoren:

Für den Test ultraschallbasierter (Park-)assistenzsysteme wird in [72] ein Ansatz beschrieben, bei dem der Ultraschallsensoroutput durch Nachbildung der elektrischen Signale emuliert wird. Die künstlich erzeugten Signale werden direkt an Stelle der Sensoren an das Steuergerät weitergegeben. Auch für den Ultraschall ist eine Lösung möglich, bei dem der vollständige Sensor als reales Testobjekt mit integriert wird. Dafür werden, ähnlich wie dem Prinzip beim Radar „over-the-air“ Test System, Ultraschallwandler gegenüber den Ultraschallsensoren platziert, die die ausgesendeten Ultraschallsignale detektieren und dann über eine Laufzeitberechnung in der Simulationsumgebung die entsprechenden Reflektionssignale an die Sensoren zurücksenden.[21]

[21] https://ipg-automotive.com/de/produkte-services/test-systems/vil-systeme/

Auch für den HiL-Test kamerabasierter Systeme sind unterschiedliche Schnittstellen zwischen System-under-Test und Simulationsumgebung möglich. Bereits seit einigen Jahren gehört die Methode zum Stand der Technik, das vollständige Kamerasystem als Testobjekt vor einen Monitor zu platzieren, auf dem synthetisch erzeugte Bildsequenzen als Stimuli erzeugt werden (vgl. [73], [74]). In [75] wird ein Framework vorgestellt, in dem auch mit synthetisch generierten Bildern verschiedene Effekte wie Sonne, Regen, Schnee oder Nebel erzeugt werden können. Diese Methode weist allerdings auch einige Nachteile auf. So ist sie nicht ohne weiteres auf Kamerasysteme anwendbar, die Weitwinkel- oder Stereoobjektive verwenden, da der darstellbare Bildbereich auf dem Monitor in der Regel nicht ausreicht. Auch entspricht die darstellbare Lichtstärke für bestimmte Anwendungen (z.B. Nachtfahrten) nicht den realen, nachgestellten Kontrastverhältnissen. Schlussendlich muss sichergestellt sein, dass der Bildaufbau im Monitor mit der Bilderfassung der Kamera synchronisiert wird, um zerrissene und damit fehlerhafte Bilder zu vermeiden [76] [77] [78]. Aus diesen Gründen wird eine Methode eingesetzt, die die Bildeinspeisung durch Emulation des Bildsensors direkt am Kamerasteuergerät durchführt („video injection“) und damit den kompletten optischen Pfad im System-under-Test umgeht [78] [79]. Die Methode der direkten Bildeinspeisung ist darüber hinaus auch mit dem Vehicle-in-the-Loop Ansatz kombinierbar (vgl. Abschnitt „Vehicle-in-the-Loop“ in diesem Kapitel) [79].

Zusammenfassend lässt sich festhalten, dass es je nach verwendeter Sensortechnologie beim System-under-Test verschiedene Ansätze gibt, um nach dem HiL Prinzip eine Schnittstelle zur Simulationsumgebung darzustellen. Diese Ansätze unterscheiden sich nicht nur in der technischen Realisierbarkeit und unterschiedlichen Vor- und Nachteilen für den HiL Simulator, sondern zielen auch auf unterschiedliche Anwendungsfälle beim HiL Test ab. Schematisch lässt sich die Wirkkette vom Sensor bis zur Regelfunktion mit den möglichen Schnittstellen des System-under-Test für HiL wie folgt darstellen (Abbildung 3.5):

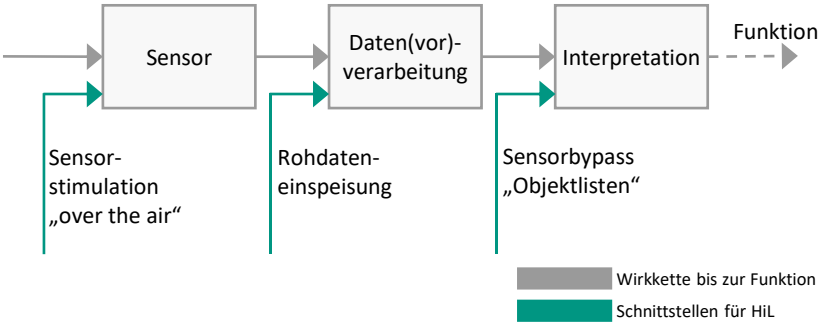


Abbildung 3.5: Wirkkette sensorbasierter Regelfunktionen mit potentiellen Schnittstellen für HiL (eigene Darstellung, angelehnt an [73])

Ein vollständiger Test der Wirkkette inklusive der realen Sensorfunktion ist somit nur über die Sensorstimulation möglich. Liegt der Fokus hingegen auf dem Test der nachgelagerten Regelfunktion, z.B. einem Spurhalteassistenten, kann der komplette Sensor umgangen werden und das Ergebnis der Sensordatenverarbeitung, z.B. einer Linienerkennung, als Objektliste direkt in die Funktion eingespeist werden. Einen Mittelweg stellt die Rohdateneinspeisung dar, die zwar den Test der Datenverarbeitung, z.B. der Objekterkennung, ermöglicht, aber im Gegensatz zum erstgenannten Ansatz keinen vollständigen physikalischen Test des Sensors ermöglicht.

Driver-in-the-Loop:

Die Interaktion zwischen Fahrer und Fahrerassistenzsystem und ihre Auslegung in Form der Mensch-Maschine-Schnittstelle (engl. human-machine interface, HMI) stellt für Fahrerassistenzsysteme bis zur Stufe des hochautomatisierten Fahrens eine wichtige Komponente dar. Daher ist insbesondere für diese Systeme auch der Test dieser Schnittstellen von Bedeutung. Bei Driver-in-the-Loop Test Systemen wird das Fahrermodell durch einen realen, menschlichen Fahrer ersetzt. Durch dieses Verfahren kann das Probandenverhalten beispielsweise hinsichtlich Aufmerksamkeit, Ermüdung oder Übergabe- und Reaktionsgeschwindigkeit zwischen System und Fahrer in einem Laborumfeld bestimmt und getestet werden. Eine Übersicht über Test- und Messtechniken,

sowie Metriken zur Auswertung in diesem Kontext wird beispielsweise in [80] gegeben. Grundsätzlich sind zwei Formen von Simulatoren denkbar, in denen die Probanden oder Testfahrer platziert werden können: stationäre und bewegliche [55] (vgl. Abbildung 3.6). Während die dynamischen Fahrsimulatoren ein realistischeres Empfinden des Fahrverhaltens für den Fahrer auch bei dynamischen Fahrszenarien bieten, ist die Realisierung dessen mit erhöhten Aufwänden verbunden, da die Beschleunigungen im Fahrzeug nur für eine begrenzte Zeitdauer im Simulator in Abhängigkeit der Konstruktion der Bewegungsplattform abgebildet werden können [81]. Stationäre Simulatoren hingegen bieten keinerlei Bewegung des „Fahrzeuginnenraums“. Die Umgebung ist vollständig auf den Test der Bedienelemente in der Interaktion zwischen Fahrer und System und Wahrnehmungsaufgaben des Fahrers ausgelegt.

Abbildung 3.6: Links: Dynamischer Fahrsimulator (Toyota) [81], rechts: stationärer Fahrsimulator (IPG Automotive)[22]

Beiden Simulatorvarianten ist gemein, dass das Umfeld, andere Verkehrsteilnehmer, Systeme des (Rest-)fahrzeugs, die (Rest-)wahrnehmung und Teile der Fahrdynamik simuliert werden [55].

Der Einsatz von Driver-in-the-Loop fokussiert damit hauptsächlich nicht auf den Test der Funktion als solche, sondern auf den Test der Schnittstellen zum Fahrer.

[22] https://ipg-automotive.com/de/produkte-services/test-systems/fahrsimulatoren/

Vehicle-in-the-Loop:

Mit der Vehicle-in-the-Loop Methode wurde 2008 erstmals ein Ansatz für Fahrerassistenzsysteme vorgestellt, der versucht, Vorteile der physischen Fahrerprobung mit den Vorteilen aus Simulationsumgebungen zu verbinden [82] [83]. Bei ViL wird das vollständige Fahrzeug mit dem integrierten zu testenden System in eine modellierte Simulationsumgebung eingebettet. Die ursprüngliche Motivation für Vehicle-in-the-Loop bestand darin, eine Testumgebung zu finden, die Fahrerassistenzsysteme unter möglichst realistischen Randbedingungen für den (Test)-fahrer erlebbar und bewertbar macht. Dies wurde später dahingehend erweitert, dass ViL auch als Alternative für HiL Tests angewendet kann, indem es de facto eine HiL-Lösung darstellt, bei der das vollständige Restfahrzeug als reale Komponente vorliegt und sich in einer freien Umgebung bewegen kann. Der Vorteil liegt darin, dass keine Modellierung und Validierung für das fahrdynamische Verhalten erforderlich ist, da dieses in Form des Realfahrzeugs verfügbar ist, und damit der Testaufbau verhältnismäßig einfach zu realisieren ist [84]. Die simulierten Komponenten beschränken sich in diesem Fall auf Teile der Umwelt wie beispielsweise Verkehrsobjekte oder Fahrbahnmarkierungen. Dennoch muss sichergestellt sein, dass die Messung der Bewegung und des Verhaltens des Fahrzeugs in der Realität mit einer ausreichenden Genauigkeit erfasst wird, da diese Signale wiederum als kontinuierlicher Input für die Simulationsumgebung zur Verfügung stehen müssen [84].

Mit Vehicle-in-the-Loop werden in der Literatur darüber hinaus auch Testkonstellationen beschrieben, bei denen sich das Fahrzeug nicht, wie zuvor beschrieben, auf einer Freifläche bewegt, sondern fest auf einem Rollenprüfstand platziert ist, sodass sich die Räder des Fahrzeugs entsprechend des Rollwiderstands und weiterer Fahrwiderstände zwar drehen können, der Gesamtfahrzeugaufbau aber keinen größeren Bewegungen ausgesetzt ist. Die Umgebung kann nun simuliert und in das Fahrzeug und seine Systeme über geeignete Schnittstellen (vergleiche Hardware-in-the-Loop) eingespeist werden [85]. Mit VEHIL wurde bereits 2002 ein Verfahren vorgestellt, bei dem das Fahrzeug ebenfalls fest auf einer Rolle platziert wird, sämtliche Umgebungsobjekte aber über ein Schienensystem relativ zum Testfahrzeug bewegt werden [86].

Zusammenfassung:

In den vorangegangenen Abschnitten wurden mit Model-, Software-, und Hardware-in-the-Loop die etabliertesten Verfahren nach dem XiL-Ansatz für den Test von Steuergeräten und Regelsystemen im Automotiveumfeld vorgestellt. Für Fahrerassistenzsysteme und hochautomatisiertes Fahren sind insbesondere die Umfeld- und Sensorsimulation und die Realisierung entsprechender Simulation zu System-under-Test Schnittstellen für diese Testmethoden eine zentrale Herausforderung. Vehicle-in-the-Loop in seinen verschiedenen Ansätzen und Driver-in-the-Loop stellen darüber hinaus weitere Testmethoden dar, die speziell für den Test von Systemen mit Umfeldwahrnehmung konzipiert wurden.

Wie bereits in Kapitel 2.3.2 dargestellt wurde, lassen sich XiL-Methoden im Entwicklungsprozess mit beispielsweise dem Vorgehen nach V-Modell kombinieren. Dabei ist charakteristisch, dass entlang des Prozesses der Anteil verfügbarer realer Komponenten immer weiter zunimmt. Aus Sicht der Testmethodik bedeutet das, dass in früheren Phasen mehr Komponenten simuliert werden müssen, um so zu jedem Zeitpunkt einen Test gegen Gesamtfahrzeuganforderungen zu ermöglichen. Auf diese Weise werden im Prozess verschiedene Integrationsstufen realer und virtueller Komponenten und Bestandteile durchlaufen (Abbildung 3.7) [34]:

	MiL	SiL	ECU-HIL	System-HIL	Rollenprüfstand	ViL	Fahrversuch
Funktions-Code	V	R	R	R	R	R	R
Steuergerät	V	V	R	R	R	R	R
System	V	V	V	R	R	R	R
Fahrzeug	V	V	V	V	R	R	R
Fahrer	V	V	V	V	V/R	V/R	R
Fahrdynamik	V	V	V	V	V	R	R
Erlebbarkeit	V	V	V	V	V	R	R
Fahrbahn	V	V	V	V	V	R	R
Verkehr/Umfeld	V	V	V	V	V	V	R

V: virtuell, R: real

Abbildung 3.7: Virtuelle und reale Komponenten verschiedener Testmethoden [34]

3.1.4 Szenariobasiertes Testen

Das szenariobasierte Testen ist ein zentrales Konzept im Kontext der simulationsbasierten XiL-Methoden und darüber hinaus. Insbesondere für den Test von Fahrerassistenzsystemen und hochautomatisierten Fahrfunktionen bilden definierte Szenarien die Grundlage, um daraus relevante Testfälle für das System-under-Test abzuleiten [87]. Im Nachfolgenden werden, dem Stand der Technik entsprechend, wesentliche Bestandteile des szenariobasierten Testens erläutert.

Szenario:

Das Szenario stellt den zentralen Begriff in diesem Konzept dar. Nach Bach bildet das Szenario die Grundlage für die Beschreibung aller Aspekte einer Simulation. Das Szenario wird als die zeitliche Entwicklung von Szenenelementen definiert. Die Szenen bestehen innerhalb des Szenarios aus einer vordefinierten Sequenz, die mit einer Startszene beginnt. Die verschiedenen Aktionen von Verkehrsteilnehmern stellen die Verknüpfung zwischen den Szenen

dar. Ein Szenario dauert im Gegensatz zu einer Szene eine bestimmte Zeitspanne an, während eine Szene eher eine Momentaufnahme darstellt [44]. Ebner ordnet dem Szenario drei beschreibende Komponenten zu [88]:

- Ego-Fahrzeug: Betrachtung der Fahrmanöver und der Eigenschaften des Ego-Fahrzeugs sowie des Fahrers
- Verkehrsteilnehmer: Beschreibung der Eigenschaften und des Verhaltens durch Parameter
- Umwelt: Bestimmt durch die Straßeninfrastruktur, Objekte (wie parkende Fahrzeuge, Bäume oder Leitplanken), Wetter und Lichtbedingungen

In den Forschungsarbeiten im und um das PEGASUS-Projekt wurde der Szenariobegriff weiter detailliert und konkretisiert [89] (siehe auch Abschnitt 3.3). Um den unterschiedlichen Anforderungen im Laufe des Entwicklungsprozesses (beispielsweise Konzept, Entwicklung, Implementierung und Test) an die Eigenschaften und Darstellungen von Szenarien gerecht zu werden, wurde ein Konzept erarbeitet, das Szenarien auf drei Abstraktionsebenen abbildbar macht. Es werden funktionale, logische und konkrete Szenarien unterschieden [90]. Während die funktionalen Szenarien eine Ebene darstellen, bei der relevante Eigenschaften sprachlich festgehalten und beschrieben werden, definieren logische Szenarien Parameterbereiche im Zustandsraum für diese Eigenschaften in Form von Entitäten und Beziehungen des Szenarios. Konkrete Szenarien sind letztlich tatsächliche Ausprägungen eines logischen Szenarios, stellen also ein festes ausgewähltes Parameterset dar. Ein logisches Szenario kann daher in mindestens ein konkretes Szenario überführt werden [90] (vergleiche Abbildung 3.8).

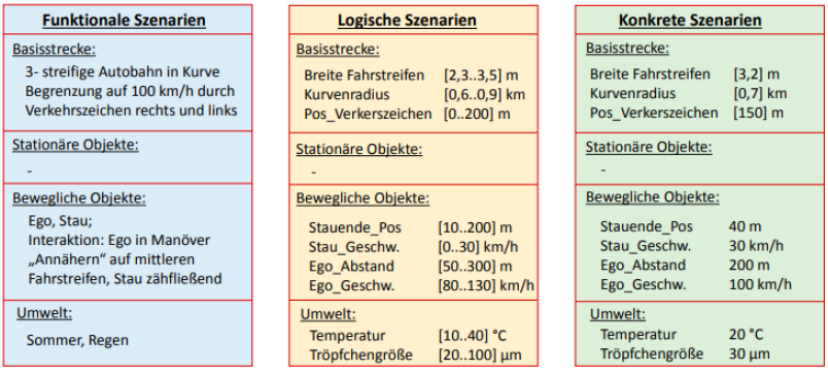


Abbildung 3.8: Abstraktionsebenen von Szenarien [90]

Die Überführung der funktionalen Szenarien in logische und schlussendlich konkrete Szenarien und der Auswahl einer Parameterraumdarstellung kann in einem wissensbasierten und systematischen Prozess der Szenariengenerierung erfolgen. Am Ende stehen sowohl eine menschenlesbare als auch maschinenlesbare Darstellung der einzelnen Szenarien [91].

Szene und Situation:

Die Szene ist, gemäß der zuvor vorgestellten Begriffsklärung des Szenarios, ein elementarer Bestandteil desselben. Eine frühere Definition verwendet den Szenebegriff in einer Analogie zum Theater: Die Szene besteht demnach aus einer (statischen) Szenerie, enthaltenen dynamischen Elementen und optionalen Fahranweisungen. Die Szenen schließen nahtlos aneinander an. Die Startszene definiert alle Elemente und ihr Verhalten [92]. Nach Ebner beinhaltet die Szene die Umwelt und Randbedingungen einer Verkehrssituation und definiert damit die Situation als Bestandteil der Szene [88]. Ulbrich und Bach stimmen den Definitionen grundsätzlich zu, betrachten aber die Szene lediglich als Momentaufnahme mit seinen Bestandteilen ohne zeitliche Dauer und damit als Teil einer Situation [93] [44]. Reschka liefert ebenfalls auf Basis der zuvor genannten Quellen eine ausführliche Definition:

Definition 3.1: *Eine Szene beschreibt eine Momentaufnahme des Umfelds, welche die beweglichen und unbeweglichen Elemente des Umfelds, die Selbstrepräsentation aller Akteure und Beobachter und die Verknüpfung dieser Entitäten umfasst […]* [94].

Die Elemente einer Szene werden genauer beschrieben als [94]:

- Ein Akteur ist ein selbsthandelndes Element.
- Ein Beobachter ist ein wahrnehmendes Element und ist Element innerhalb der Szene oder betrachtet diese als Ganze.
- Ein Element kann Akteur und Beobachter zur selben Zeit sein.
- Bewegliche Elemente haben die Fähigkeit, sich zu bewegen und können aktuell statisch oder aktuell dynamisch sein.
- Unbewegliche Elemente sind alle räumlich stationären Elemente und werden als Szenerie bezeichnet.
- Es existiert eine Selbstrepräsentation der Akteure (Beschreibung der Fähigkeiten).

Zusammenfassend sind die Bestandteile einer Szene nach den genannten Definitionen gegenübergestellt (Tabelle 4):

Aspekt	**Ebner (2014)** [88]	**Geyer et al. (2014)** [92]	**Ulbrich et al. (2015)** [93]	**Reschka (2016)** [94]
Fahrzeugumgebung und Randbedingungen	✓	✓		✓
Dynamische Elemente (Objekte)		✓		✓
Optionale Fahranweisungen		✓	✓	
Momentaufnahme des Umfelds			✓	✓
Situation ereignet sich in der Szene	✓			

Szene ereignet sich in der Situation		✓	✓	✓

Tabelle 4: Vergleich der Bestandteile und Eigenschaften einer Szene

Im Gegensatz zur Szene beschreibt die Situation die Repräsentation des Blicks eines Teilnehmers auf die jeweilige Szene [44]. Alle möglichen Verkehrssituationen werden dabei aus der Fahrersicht erfasst. Die Verkehrssituation beschreibt einen erlebbaren, zeitlich und räumlich begrenzten Ausschnitt aus dem Verkehrsablauf [95]. Nach Bubb lässt sich die Situation hinsichtlich der objektiven und subjektiven Aspekte weiter differenzieren in erstens die Verkehrssituation, die eine objektive, räumliche und zeitliche Konstellation der verkehrsbezogenen Einflussgrößen darstellt, zweitens die Fahrsituation, welche die sich daraus ergebende allgemein wahrnehmbare Fahrersicht beschreibt und drittens, die eigentlich subjektive, vom Fahrer tatsächlich wahrgenommene Situation [96].

Ulbrich und Reschka fassen die Definition der Situation folgendermaßen zusammen:

Definition 3.2: *Eine Situation beschreibt die Gesamtheit der Umstände, die für die Auswahl geeigneter Verhaltensmuster zu einem bestimmten Zeitpunkt zu berücksichtigen sind. Eine Situation wird aus der Szene durch einen Prozess der Informationsauswahl und –augmentierung abgeleitet, basierend auf transienten (z.B. missionspezifischen) wie auch permanenten Zielen und Werten. Folglich ist eine Situation immer subjektiv, indem sie die Sicht eines Elements repräsentiert* [93] [94].

Manöver:

Der Manöverbegriff wird in der Literatur umfangreich und vielseitig betrachtet. Nach [97] ist ein Manöver allgemein definiert als zeitlicher Verlauf von Steuersignalen, die gewünschte Übergängen von einem stationären Verhalten zu einem anderen abbilden. Ein Manöver kann dabei eine Zeitspanne von mehreren Minuten andauern [97] [94].

Mit Bezug auf die Fahraufgabe kann ein Fahrmanöver als das Verhalten eines dynamischen Verkehrsteilnehmers in Form einer Aktion oder Aktionsfolge gesehen werden, welche die aktuelle Fahrsituation in eine andere Fahrsituation überführt. Ein Fahrmanöver wiederum kann sich wiederum aus untergeordneten Fahrmanövern zusammensetzen. Beispielsweise kann ein Überholmanöver aus einem Fahrstreifenwechsel nach links, einer Vorbeifahrt und einem abschließenden Fahrstreifenwechsel nach rechts bestehen [98]. Bach definiert zusammenfassend das Manöver als „die abstrakte Beschreibung des Verhaltens eines Teilnehmers während eines Akts [44]“.

Je nach Forschungs- oder Anwendungskontext existieren verschiedene Klassifikationen von Manövern, die in unterschiedlichen Manöverkatalogen vorgestellt werden. Tölle [99] leitet aus den von Nagel und Enkelmann [100] bereits im Jahr 1991 vorgestellten 17 Manövern eine Basismenge von neun Manövern ab. Schreiber et al. untersuchen, welcher der von Nagel und Enkelmann definierten Manöver in Versuchsreihen durch Probanden wieder identifiziert werden können [101] und schließen in ihrem Manöverkatalog beispielsweise das Manöver „Überholen“ unter anderem aus der oben genannten Zusammensetzbarkeit aus anderen Manöver aus [102]. Franz et al. wiederum greifen für ihr Conduct-by-Wire Konzept auf die von Schreiber definierten Manöver zurück und nehmen für diesen Anwendungsfall eine Selektion impliziter und expliziter Manöver vor [103].

Zusammenfassend wird die Zuordnung der Manöver zu den verschiedenen hier erläuterten Konzepten dargestellt (Tabelle 5):

Manöver	**Nagel (1991)** [100]	**Bajcy (1996)** [104]	**Tölle (1996)** [99]	**Schreiber (2012)** [102]	**Franz (2015)** [103]
Anfahren	✓	✓	✓	✓	
Straßenzug folgen	✓	✓	✓	✓	✓
Kreuzung überqueren	✓	✓	✓		
Fahrstreifenwechsel links/rechts	✓	✓	✓	✓	✓
Links/Rechts abbiegen	✓	✓		✓	✓

Am rechten Rand anhalten	✓	✓			
Rückwärts fahren	✓	✓			
Kehre links/rechts	✓	✓			
Richtung umkehren	✓	✓	✓		
An einen sich vor dem Fzg. befindenden Gegenstand annähern	✓	✓	✓		
Überholen	✓	✓	✓		
Vor einem Gegenstand anhalten	✓	✓			
Links/rechts an einem Gegenstand vorbeifahren	✓	✓			
Hinter einem anfahrenden Fzg. anfahren	✓	✓			
Einem Fzg. folgen	✓	✓			
In eine Parklücke einfahren	✓	✓	✓		
Aus einer Parklücke herausfahren	✓	✓			
Anhalten		✓		✓	
Bremsen				✓	
Geradeaus fahren					✓

Tabelle 5: Übersicht über Fahrmanöverzuordnungen nach verschiedenen Literaturquellen

Testfall:

Die Generierung von Testfällen bildet einen wesentlichen Bestandteil in einem Testkonzept für die Validierung und Verifikation von Fahrerassistenzsystemen oder auch Systemen hochautomatisierten Fahrens [105] (vgl. auch Kapitel 3.2.3). Der Testfall dient zur Überprüfung konkreter Eigenschaften des System-under-Tests anhand von Randbedingungen, wie Eingabedaten, erwartete Reaktionen oder Prüfanweisungen, die vor, während oder nach der Testdurchführung überprüft werden. Nach Spillner und Linz kann die Spezifikation von Testfällen in zwei Schritten erfolgen. Zunächst werden, ähnlich der Szenario-

definition, logische Testfälle erstellt, bei denen beispielsweise die Randbedingungen als Bereiche definiert sind. Bei der anschließenden Ableitung der konkreten Testfälle werden konkrete Parameter bestimmt, die bei der eigentlichen Testausführung zum Tragen kommen [106]. Im Kontext des szenariobasierten Testens kann unter einem Testfall also die Konkretisierung eines Szenarios verstanden werden, die als weitere Bestandteile Bewertungskriterien beinhaltet, die entscheiden, unter welchen Bedingungen ein Testfall bestanden wurde [55]. Des Weiteren gehören zu einem Testfall die Beschreibung der Anforderungen an die Testdurchführung, wie beispielsweise die Anforderung an die Genauigkeit oder die benötigte Zeit zur Durchführung der Testfälle [55]. Die Zusammenhänge zwischen Szenarien, Testfällen, –durchführung und –zielen können auch in Form von UML-Diagrammen dargestellt werden [107].

Szenariodatenbank/-katalog:

Die Erstellung von Szenariokatalogen oder auch Datenbanken ist ein effizientes Mittel, um die verschiedenen Szenarien und die daran gekoppelten Testfälle zu verwalten. Sie ermöglichen die Sammlung von relevanten Verkehrssituationen, die aus verschiedenen Quellen entstammen können. Eine Datenquelle für eine Szenariodatenbank zum Test hochautomatisierter Fahrfunktionen stellen Unfalldatenbanken dar. Da diese Funktionen darauf ausgelegt sind, Unfälle zu vermeiden, sind vor allem die Szenarien einer Unfalldatenbank relevant, die als Unfallursache auf die Verkehrskonstellation an sich, aber eher nicht auf menschliches Fehlverhalten zurückzuführen sind [108]. Für hochautomatisierte Fahrfunktionen können jedoch auch Ursachen zu kritischen Szenarien führen, die sich nicht durch Rekonstruktion aus den Unfalldatenbanken ermitteln lassen. So existieren beispielsweise noch keine umfangreichen Datenbanken, bei denen eine Sensorfehldetektion zu einem potentiellen Unfall führt, da diese Ursachen für menschliche Fahrer keine oder eine nicht vergleichbare Rolle spielen [109]. Als weitere Quellen für die Datenbank können daher Realverkehrsdaten aus dem Feld oder dem Prüfgelände (vgl. [110] [111] [112]), ebenso wie Daten aus der Verkehrssimulation oder dem Fahrsimulator als Eingang dienen (vgl. beispielsweise [113]). Als letzte Möglichkeit können kritische und relevante Szenarien aus Expertenwissen abgeleitet werden und damit

ebenfalls als Quelle für die Szenariodatenbank fungieren [108]. Die Zusammenfassung möglicher Quellen wird in nachfolgender Tabelle (Tabelle 6) dargestellt:

Kategorie	Quelle
virtuell	Verkehrssimulation Fahrsimulator
real	Realverkehrsdaten Felddaten Field Operational Test (FOT) Naturalistic Driving Study (NDS) Prüfgelände Unfalldatenbanken
verbal	Expertenwissen

Tabelle 6: Datenquellen als Eingang für die Szenariodatenbank [108]

Zusammenfassung:

In den vorigen Abschnitten wurden wesentliche Elemente des szenariobasierten Testens dargestellt. Folgende Darstellung (Abbildung 3.9) zeigt zusammenfassend die Zusammenhänge der zuvor hier dargelegten Definitionen für Szenarien, Szenen und Testfällen auf. Ein Szenario besteht demnach aus verschiedenen konsekutiven Szenen, die jeweils Repräsentationen für das Ego-Objekt mit der zu testenden Funktion, der statischen Objekte (Szenerie) und den dynamischen Objekten beinhalten. Aktionen und Events stellen die Transitionen zwischen verschiedenen Szenen dar. Insbesondere die Manöver[23] des Ego-Objekts und der dynamischen Elemente sind dabei wesentlicher Bestandteil. Die Ausprägung des konkreten Szenarios leitet sich aus den Anforderungen der Use-Case Definitionen ab, die zusammen mit den Bewertungskriterien („Pass/Fail“) einen zugehörigen Testfall bilden.

[23] Für eine Schärfung des Begriffs „Manöver“ über den Stand der Technik hinaus, wird auf Kapitel 6 verwiesen.

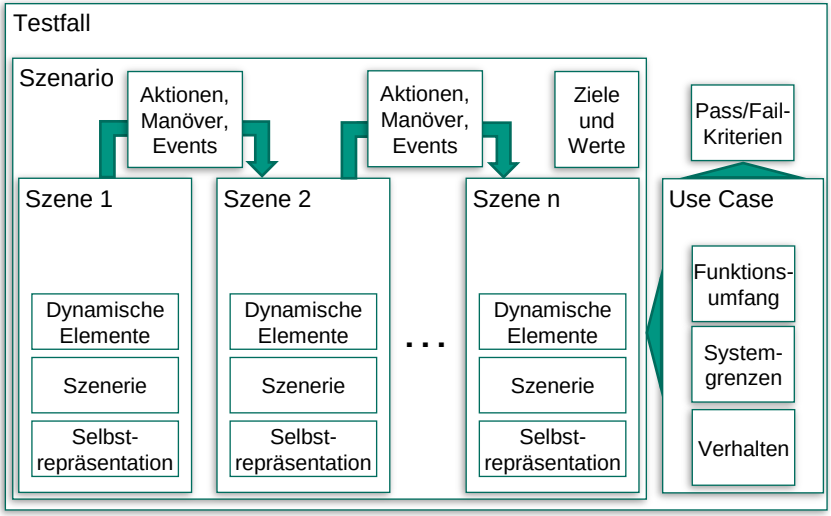


Abbildung 3.9: Zusammenhänge zwischen Szenario, Szene und Testfall (Eigene Darstellung mit Elementen aus [93])

Standardisierung:

Viele der neueren Modelle und Definitionen stammen aus dem Umfeld des PEGASUS-Projekts (siehe auch Abschnitt 3.3), welches allein schon aufgrund der Reichweite hinsichtlich Teilnehmerkreis des Projekts und Publikationen aus dem Projektkontext als aktuell maßgeblich, zumindest für den deutschen Raum, angenommen werden kann. Dennoch existieren in der Umsetzung für den szenariobasierten Test bislang erst wenige Standards. Hervorzuheben sind die Aktivitäten von ASAM[24] (Association for Standardization of Automation and Measuring Systems), die insbesondere mit OpenDRIVE und OpenSCENARIO, als geplantes vollständiges Beschreibungsformat für das Szenario, Ansätze zur Standardisierung zu schaffen.

[24] https://www.asam.net/

Sowohl OpenDRIVE als auch OpenSCENARIO erheben den Anspruch als toolunabhängiges Format eingesetzt werden zu können. OpenDRIVE konzentriert sich auf die Abbildung von Staßen(netzwerken) und ihrer Geometrien, sowie der Eigenschaften, die die Logik beeinflussen (z.B. Fahrstreifen, Markierungen, Verkehrszeichen)[25]. OpenSCENARIO verwendet ebenfalls eine XML-basierte Syntax und setzt auf OpenDRIVE als Grundlage auf. In dem Beschreibungsformat wird das zeitabhängige Verhalten der Entitäten eines Szenarios festgehalten. Dazu dienen die Definitionen von Aktionen und (Trigger)-Bedingungen um verschiedenste Verkehrsszenarien in einer Simulation beschreiben zu können.[26] Während OpenDRIVE bereits vollständig verfügbar ist (Version 1.5), ist die Definition von OpenSCENARIO mit Stand Ende 2019 noch nicht abgeschlossen.

Aktuelle alternative Beschreibungsformate sind beispielsweise für Staßennetzwerke das auf Open Street Map basierte Lanelets [114] oder GeoScenario für die Darstellung des Szenariorepräsentation [115].

3.2 Aktuelle Ansätze für die Absicherung und Freigabe

Stand heute ist kein allgemein anerkanntes und dokumentiertes Vorgehen bekannt, das die vollständige Freigabe für hochautomatisierte Fahrfunktionen, bei der die Rückfallebene „Fahrer“ nicht mehr existiert, in Deutschland zielführend erlaubt. Allein die statistische Validierung ist aufgrund der zu fahrenden Menge an Testkilometern in der Felderprobung zwar theoretisch möglich, aber in der Praxis kaum durchführbar (vgl. Kapitel 1.1). Durch die in der Validierung zu berücksichtigende vergleichsweise große Zahl an Fahrsituationen, wird mit einem um den Faktor 10^6 bis 10^7 höheren Validierungsaufwand für das autonome Fahren im Vergleich zu einfachen Fahrerassistenzfunktionen gerechnet [6]. Auch sollte im ursprünglichen Gedanken der Erkenntnisgewinn

[25] http://www.opendrive.org/docs/OpenDRIVEFormatSpecRev1.5M.pdf

[26] https://www.pegasusprojekt.de/files/tmpl/Pegasus-Abschlussveranstaltung/14_Scenario-Formats.pdf

durch die Felderprobung eher auf die Evaluation des tatsächlichen Nutzens des Systems abzielen, während hingegen die Lastenheftkonformität nach der ISO 26262 (funktionale Sicherheit) bereits per Auslegung sicherzustellen beziehungsweise fatale Eingriffsfolgen auszuschließen sind [116].

In den folgenden Abschnitten werden daher alternative Ansätze beschrieben, wie sie aktuell in Industrie und Forschung diskutiert werden.

3.2.1 Anforderungen an das Testkonzept

Wachenfeld und Winner beschreiben grundsätzliche Anforderungen, die ein Testkonzept erfüllen muss, um den Herausforderungen für die Freigabe[27] zu begegnen. Dabei wird zwischen Effektivitäts- und Effizienzkriterien unterschieden [3]:

Effektivitätskriterien umfassen die Repräsentativität / Validität, die Variierbarkeit und die Beobachtbarkeit des Testkonzepts. Damit wird gefordert, dass die Testfallgenerierung eine erforderliche Testabdeckung sicherstellt (Repräsentativität) und die Testdurchführung einen hinreichenden Grad an Realität (Validität) aufweist. Variierbarkeit fordert, dass die Testduchführung in der Lage ist, alle definierten Testfälle umzusetzen, während für die Testauswertung alle Parameter der Testdurchführung bekannt sein müssen (Beobachtbarkeit) [3].

Effizienzkriterien hingegen sind Ökonomie in der Testduchführung und -vorbereitung, Reproduzierbarkeit der Tests, Frühzeitigkeit im Entwicklungsprozess und die Gewährleistung der Sicherheit aller Testbeteiligten, insbesondere bei Realfahrten [3].

[27] Hier wird unter Freigabe die SOP-Freigabe durch den OEM im Sinne der funktionalen Sicherheit (ISO 26262) verstanden. Die behördliche Zulassung des Gesamtfahrzeugs für den Straßenverkehr erfolgt nach national geregelten Vorschriften (Homologation) [215] und wird hier nicht betrachtet.

3.2.2 Wiederverwendung validierter Funktionen

Einem evolutionären Ansatz folgend, ist der einfachste Weg die Sicherheit eines automatisierten Fahrzeugs zu bewerten, auf bereits bekannte und validierte Funktionen zu setzen und die Funktionalität sukzessive und inkrementell zu erweitern [117]. Auch wenn dieser Ansatz die Validierungsaufgabe erleichtert, beantwortet er per se nicht das grundsätzliche Validierungsproblem, da zwar das „was-muss-validiert-werden“ eingeschränkt wird, aber nicht das „wie“. Für das „wie“ existiert kein bekanntes allgemein akzeptiertes Vorgehen, zumal sichergestellt werden muss, dass durch Wechselwirkungen der neuen Funktionen mit den existierenden Funktionen die Validität dieser weiterhin gegeben ist. Dennoch schreibt der Freigabeprozess vor, Prüfkriterien im Sinne von „bestanden“ respektive „nicht bestanden“ für den Sicherheitsnachweis der Funktionen vorzusehen. Dies gilt unabhängig davon, welche Methoden schlussendlich für die Freigabe herangezogen werden und sollte durch ein Expertengremium bestimmt werden. Für die Kriterien sind allgemein anerkannte Werte auszuwählen, die die (gewünschte) Fahrzeugreaktion beschreiben und die geeignet sind, das Systemversagen oder Fehlfunktionen in bestimmten Situationen zu beschreiben [118]. Sowohl für die Auswahl der Kriterien als auch der zugehörigen Tests und Testfälle lassen sich verschiedene Ansätze wiederfinden, die seit einigen Jahren diskutiert werden.

3.2.3 Ansätze für den Absicherungsprozess

In Winner [117] und Wachenfeld [3] werden einige aktuelle Ansätze zur Beschleunigung des Validierungsprozesses und der Freigabe zusammengefasst, die hier ausführlich dargestellt werden:

Eckstein und Zlocki [119] gehen nicht explizit auf die Validierung von hochautomatisierten Fahrfunktionen ein, schlagen aber einen effizienten Prozess für ADAS-Funktionen mit Umweltwahrnehmung vor. In ihrem Prozess werden vier Methoden für die Evaluierung eingesetzt: Klassische Felderprobung (field operational test), Fahrversuch in kontrollierter Umgebung (controlled field), Fahrsimulator (Dynamic Driving Simulator) und Simulation. Diese Methoden

unterscheiden sich im Wesentlichen im Grad der Virtualisierung der drei Bestandteile Fahrzeug, Fahrer und Umwelt (Abbildung 3.10 links).

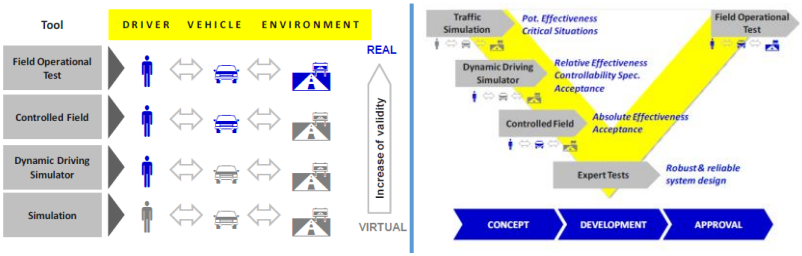


Abbildung 3.10: Evaluierungsmethoden und -prozess nach Eckstein und Zlocki [119]

Im Prozess (Abbildung 3.10 rechts) werden diese Methoden wie folgt verwendet: Wird ein Konzept für eine neue Funktion entworfen, können bereits in dieser Phase kritische Situationen, die bereits zuvor in Feldtests gesammelt wurden, effizient in einer Simulation erprobt werden. Im weiteren Verlauf wird durch die Hinzunahme weiterer realer Komponenten im Fahrsimulator und Fahrversuch der Grad der Validität dieser Testumgebungen für die ausgewählten kritischen Situationen weiter erhöht. Damit werden gezielt Situationen unter effizienten Rahmenbedingungen erprobt und der Einfluss auf die zu entwickelnde Funktion untersucht. Dieser Ansatz liefert allerdings kein Maß für die Repräsentativität der kritischen Situationen und gibt damit auch keine Antwort auf die Frage der erforderlichen Testabdeckung.

Ein weiterer Ansatz zur Testfallreduzierung wird von Schuldt et al. vorgeschlagen. Für ein effizientes Testkonzept (im Sinne einer Reduktion von Aufwand und Kosten für das Testen) sind demnach vier konsekutive Stufen notwendig [120]:

1. Analyse des Systems auf Einflussfaktoren
2. Effiziente und systematische Testfallgenerierung
3. Testdurchführung

4. Testauswertung

Der Fokus in [120] liegt auf dem zweiten Punkt, der Testfallgenerierung. Das Ziel ist es, möglichst effizient redundanzfreie Tests zu finden und orientiert sich am szenariobasierten Testen (vgl. Kapitel 3.1.4). Dabei kommen Methoden des Black-Box-Testens (ohne Kenntnis der inneren Struktur des Testobjekts) und kombinatorischer Parametervariation zum Einsatz. Die Generierung der Testfälle erfolgt dabei aus einer zuvor durchgeführten Anforderungs- und Schnittstellenbeschreibung. Durch eine Äquivalenzklassenbildung und Grenzwertbetrachtung werden schließlich redundante Testfälle ausgeschlossen [105]. Der szenariobasierte Ansatz ermöglicht ebenfalls die Zuordnung der Testfälle zu X-in-the-Loop Methoden [121].

Tatar und Mauss beschreiben ein Verfahren, das es ermöglicht, Eingangsparameter von Testfällen zu variieren, sodass eine zu definierende Bewertungsfunktion für den Test optimiert wird. Damit hat auch dieser Ansatz das Ziel, effizient relevante Testfälle zu generieren, die ebenfalls im Rahmen von X-in-the-Loop einsetzbar sind [122]. Sowohl dieser als auch der zuvor genannte Ansatz ermöglichen eine Reduktion der Testfälle für ein Black-Box-Verfahren, allerdings ohne eine mögliche Aussage über einen Abdeckungsgrad zu treffen.

Gegensätzlich zu den Black-Box-Tests, bei denen die Testfälle in der Regel ausgehend von einer Anforderungsanalyse abgeleitet, und ohne Kenntnis über das zu testende, implementierte System entwickelt werden, stehen beispielsweise White-Box-Tests oder sogar formale Methoden, die das Ziel haben, einen mathematischen Nachweis der Fehlerfreiheit des zu testenden Systems zu erbringen (vgl. [123], [32]). Am Beispiel eines “obstacle avoidance control algorithms” konnte der Sicherheitsnachweis des Systems mittels formaler Methoden erbracht werden [124]. Zu berücksichtigen ist allerdings, dass die Gültigkeit des Nachweises durch formale Methoden die Validität der eingesetzten Modelle voraussetzt, die als weitere Herausforderung betrachtet werden muss. Auch ist zumindest fraglich, ob mit einer steigenden Komplexität der zu testenden Systeme, der Einsatz formaler Methoden für den Sicherheitsnachweis überhaupt noch möglich oder zumindest mit leistbarem Aufwand durchführbar ist.

Weitzel und Lotz stellen die Übertragbarkeit des Use-Case- und anforderungsfallgetriebenen Absicherungsansatzes, wie er beispielsweise nach dem V-Modell für herkömmliche Fahrerassistenzsysteme praktiziert wird, auf Funktionen des teil- und hochautomatisierten Fahrens dar [21]. Der Vorteil des Use-Case getriebenen Ansatzes ist, dass durch ein systematisches Vorgehen prinzipiell alle potentiellen Lösungen für das funktionale Verhalten durch die zuvor bestimmten Merkmale gefunden und überprüft werden können. Merkmale und die jeweiligen Merkmalsausprägungen beschreiben in diesem Kontext für den Anwendungsfall relevante Situationsparameter wie beispielsweise die Anzahl der Fahrstreifen in einer Verkehrssituation. Nachteilig ist, dass durch eine Erhöhung der Merkmalsanzahl und deren Ausprägungen schnell eine so große Zahl an Anwendungsfällen erzeugt wird, dass die Durchführung der zugehörigen Testfälle und deren Auswertung nicht mehr mit vertretbarem Aufwand möglich ist. Auch der Ansatz ein Relevanzmaß für die Anwendungsfälle einzuführen, ist nur bedingt hilfreich, da für ein dauerhaft aktives System für die Fahrzeuglängs- oder -querführung (z.B. Autobahnpilot) sämtliche, auch unbekannte, Situationen innerhalb des Anwendungsbereiches beherrscht werden müssen. Damit greift das allgemeine Prinzip für die Testfallermittlung nicht mehr, nach dem bei der lastenheftbasierten Testspezifikation für jede definierte Anforderung mindestens ein Testfall erstellt wird, da eine vollständige Systemspezifikation nicht sichergestellt werden kann. Auch für die beiden weiteren nach Horstmann existierenden Möglichkeiten zur Testfallermittlung, die risikobasierte Testspezifikation und die schnittstellenbasierte Testspezifikation [125], ist die Anwendbarkeit für die Absicherung von hochautomatisierten Fahrfunktionen nach dem aktuellen Stand der Technik noch unklar beziehungsweise bislang nicht umfassend beantwortet [21].

Einen integrierenden Ansatz beschreibt Reschka mit der Einführung von Fertigkeitengraphen als Werkzeug zur Modellierung und Erstellung der Item-Definitionen entlang des Entwicklungsprozesses. Der resultierende Anforderungskatalog hat den Anspruch, funktional sicheres Verhalten umfassend zu beschreiben. Das Sicherheitskonzept baut darüber hinaus auf einer Gefährdungsanalyse und Risikobewertung auf und leitet Sicherheitsziele für identifizierte sogenannte “pathologische Szenarien” ab, um die jeweilig aktuelle Leistungsfähigkeit des automatisierten Fahrzeugs zu bewerten. Die pathologischen

Szenarien werden in diesem Ansatz als Ausgangspunkt betrachtet. Der Ansatz liefert keine Aussage zur Menge der benötigten Szenarien, um eine ausreichende Sicherheit zu gewährleisten [94].

Fast alle bis hier dargestellten Ansätzen orientieren sich an folgender Herangehensweise [126]:

1. Eine schrittweise Einführung der neuen Technologien bis hin zum Endzustand des vollautonomen Fahrens

2. Raffung der notwendigen Testfälle, basierend auf Felderfahrung und statistischen Verfahren, mit der Herausforderung eine Metrik für die Sicherheit des Systems zu bestimmen

3. Einsatz zur Realfahrt alternativer Testwerkzeuge mit virtuellen (Teil)-komponenten, um die Tests abhängig von der verfügbaren Rechenleistung beschleunigen und parallelisieren zu können

Zusammengefasst zeigt nachfolgende Tabelle diese und weitere Ansätze (vgl. auch Methoden aus Kapitel 3.1.3) in einer kompakten Darstellung (Tabelle 7):

Ansatz	**Vorteil**	**Nachteil**
Physischer Dauerlauf	Validität	Anzahl Testkilometer → „Freigabefalle“
Open-Loop an Realdaten	Optimierung insbesondere der Perzeption	Szenarioauswahl, Testfälle, auf Teilfunktionen (z.B. Perzeption) beschränkt
X-in-the-Loop / V-Model	Testautomatisierung / Testvarianz	Ableitung der Testfälle aus Spezifikation, Nachweis der Validität
Evolutionäre Einführung / Kundentest	Kilometerzahl, Updatefähigkeit	Begrenzte Einsetzbarkeit, Risiko

Verfahren aus Luftfahrt und Schienenverkehr (erläutert in [3])	Automatisierte Systeme seit Jahren eingesetzt und etabliert	Nicht übertragbar → andere Rahmenbedingungen
Trojanisches Pferd (erläutert in [127])	Schnelle Verfügbarkeit der Testdaten	Kein vollständiger Test der Zielfunktion

Tabelle 7: Übersicht von Absicherungsansätzen, angelehnt an [127]

3.3 Stand der Forschung

Für den Test und die Absicherung hochautomatisierter Fahrfunktionen existieren weiterhin für Industrie und Forschung ungelöste Fragestellungen (siehe Abschnitt 3.2). Nachfolgend wird explizit der Stand der Forschung dargestellt und ein Überblick über aktuell laufende oder kürzlich abgeschlossene Projekte und Initiativen gegeben.

Das autonome Fahren gilt für die Bundesregierung Deutschlands als eine der Zukunftstechnologien im Mobilitätsbereich. Bereits 2015 wurde die „Strategie automatisiertes und vernetztes Fahren (Strategie AVF)“ der Bundesressorts BMBF, BMWi und BMVI definiert, um Innovationen in diesem Zusammenhang zu fördern und die Technologieführerschaft für den Automobilstandort Deutschland zu sichern [128]. Der Fokus der Förderungen liegt nicht nur in der Förderung der Innovation neuer Technologien, sondern auch in Forschungen zu neuen Test- und Absicherungsmethoden.

Grundlagen sollten mit dem im Juni 2019 abgeschlossenen Forschungsprojekt PEGASUS geschaffen werden, das inhaltlich insbesondere auf das hochautomatisierte Fahren auf Autobahnen abzielt. In diesem Projekt wurden Testmethoden und -prozesse entwickelt, die basierend auf einer durchgängigen und flexiblen Werkzeugkette, den übergreifenden Einsatz von definierten Szenarien und Testfällen als einheitliche Methode bewirken sollen. Die Methode

wird in 20 Punkten zusammengefasst, die in [89] näher erläutert werden. Ein Schwerpunkt des Projekts lag im Rahmen der Methodik auf der Schaffung einer einheitlichen Szenario- und Testfalldefinition (siehe rot markierte Elemente in Abbildung 3.11), die bereits in Abschnitt 3.1.4 dargestellt wurde.

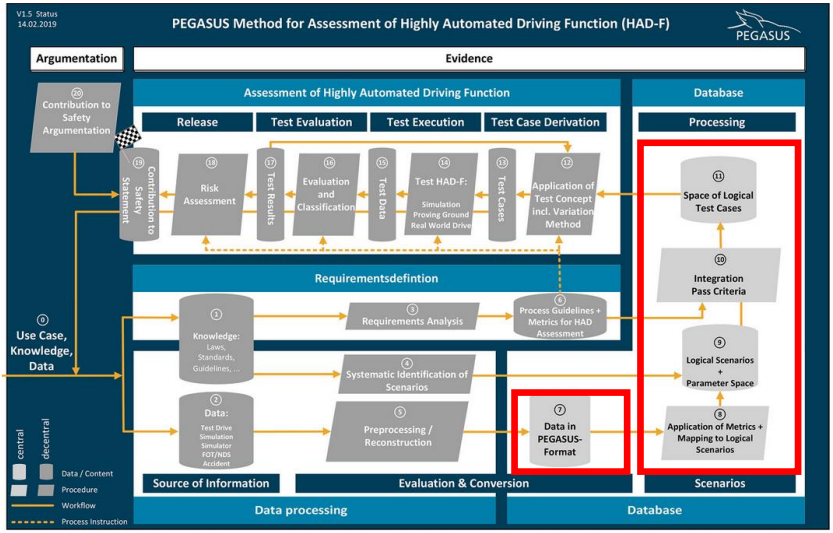


Abbildung 3.11: Die 20 Punkte der PEGASUS-Methode [89]

Das seit März 2019 laufende Projekt SETLevel4to5 fokussiert auf die Simulationsumgebungen, die Erhöhung in der Testeffizienz und versucht die gewonnenen Erkenntnisse aus dem PEGASUS Projekt für die „Autobahn“ auf andere Verkehrsbereiche zu übertragen [89].

Das auf vier Jahre ausgelegte Projekt UNICARagil beschäftigt sich seit Anfang 2018 mit neuen modularen und skalierbaren Fahrzeugarchitekturen und –plattformen. Im Rahmen des Projektes werden insgesamt vier voll-automatisierte und fahrerlose Prototypen entwickelt und aufgebaut, um die Architekturen zu demonstrieren [129]. Ein Schwerpunkt des Projekts liegt auch in der Validierung der Funktionen und der Entwicklung eines Konzepts zur modularen Absicherung der Softwarekomponenten.

Das Projekt ENABLE-S3, beendet im April 2019, beschäftigte sich ebenfalls mit Test und Validierungsmethoden, wenn auch für allgemein hochautomatisierte Systeme. Neben der Entwicklung von domänenübergreifenden Testverfahren zur Überprüfung von funktionalen und nicht-funktionalen Eigenschaften lag ein besonderer Fokus ebenfalls auf szenariobasierten Ansätzen, Simulationsplattformen für die Systemvalidierung, Technologien für HiL und der Schaffung und Integration von Standards für Datenstrukturbeschreibungen wie dem Open Simulation Interface (OSI) für virtuelle Sensoren [130]. Die Validierungsmethodologie nach dem ENABLE-S3 Projekt sieht als Teilaspekte ebenfalls die Realdatenanalyse zur Extraktion von Szenarien und die Selektion relevanter Testfälle vor (vergleiche rot markierte Elemente in Abbildung 3.12).

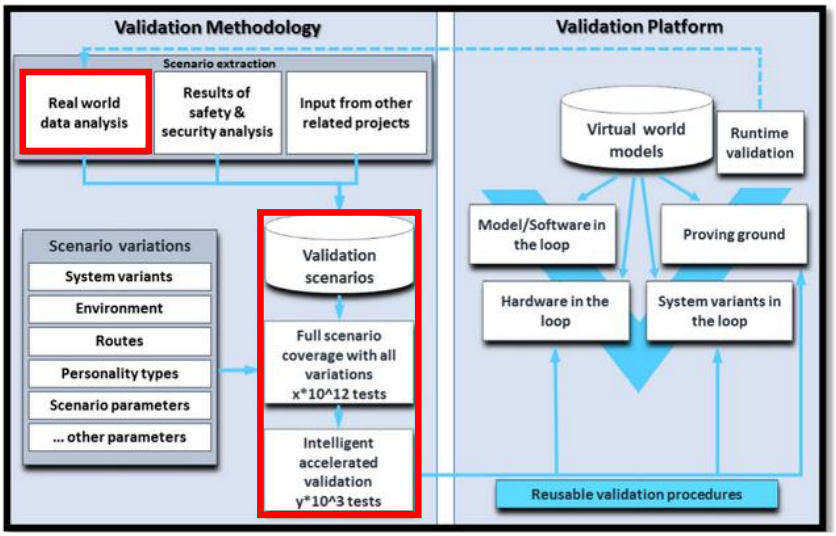


Abbildung 3.12: Validierung hochautomatisierter Systeme nach ENABLE-S3 [130].

Das 2016 gestartete Projekt Imagine befasst sich mit Lösungen für kooperatives Fahren. Das Ziel ist es, neue Assistenzsysteme für das vernetzte und kooperative Fahren zu entwickeln. Das Projekt stützt sich auf fünf Kerninnovationen. Neben der Entwicklung der kooperativen Funktionen selbst, den

Mensch-Maschine-Schnittstellen für diese Funktionen, dem fahrzeugübergreifenden Umfeldmodell und den Kommunikationsmechanismen, liegt ein Schwerpunkt auf der Abbildung der Sensorik in einer virtuellen Simulationsumgebung, um eine nutzbare Testplattform für diese Funktionen zu schaffen [131].

Das Forschungsprojekt Providentia beschäftigt sich ebenfalls mit vernetzten kooperativen Systemen in hochautomatisierten Fahrzeugen und hat neben den Entwicklungen das Ziel Kommunikations- und Backendinfrastrukturtests für diese Systeme auf dem Testfeld A9 zu erforschen und zu erproben. Ein Fokus liegt bei der Testumgebung auf der Virtualisierung der erforderlichen Verkehrsszenarien.[28]

Ein weiteres Testfeld speziell zur Erprobung für das vernetzte und autonome Fahren wurde 2018 in der Region Karlsruhe in Baden-Württemberg eingerichtet. Das Testfeld umfasst im Gegensatz zu anderen Projekten alle Arten von öffentlichen Straßen, von der Tempo-30-Zone bis zur Autobahn, und soll mit seiner Infrastruktur sowohl zukünftige Lösungen für den Individualverkehr als auch den öffentlichen Personennahverkehr fördern.[29]

[28] http://testfeld-a9.de/projekt/
[29] https://taf-bw.de

4 Herausforderung und Ansatz für den Test automatisierter Fahrfunktionen

Der dargestellten Stand der Technik (Kapitel 3) zeigt, dass aktuell keine zielführende Lösung für die Absicherung und Freigabe (hoch)automatisierter Fahrfunktionen bekannt ist. Die grundsätzliche Herausforderung stützt sich auf Annahmen, die im nachfolgenden erläutert sowie begründet werden und für die Ableitung des Lösungskonzepts essentiell sind:

4.1 Annahmen

4.1.1 A1 – Nachweis durch statistische Breitenerprobung

Annahme A1: Eine vollständige Testabdeckung auf Basis einer rein physischen Breitenerprobung ist nicht möglich oder zumindest nicht ökonomisch sinnvoll.

Die wichtigste Anforderung an hochautomatisierte Fahrfunktionen ist die Gewährleistung der Sicherheit der beteiligten Verkehrsteilnehmer. Die Herausforderung für Funktionen ab Level 3 (vgl. Abschnitt 2.1.1) besteht indes darin, dass sie sämtliche Situationen innerhalb ihres Einsatzbereichs beherrschen müssen. Bedingt durch die Tatsache, dass die möglichen Situationen zum Zeitpunkt der Entwicklung und des Tests nur teilweise bekannt sein können, ist ein vollständiger Sicherheitsnachweis durch eine 100%-Testabdeckung unmöglich. Ein Rückschluss auf die Beherrschbarkeit der nicht bekannten Situationen ist aus der Analyse des Verhaltens bei bekannten Situationen ebenfalls nicht

oder nur bedingt möglich. In Anlehnung an die aus der formalen Wissensrepräsentation stammende Systematik kann diese Herausforderung als „Open-World-Problem“ bezeichnet werden.[30]

Wie bereits in Abschnitt 1.1.2 erläutert, ist jedoch ein statistischer Nachweis der Sicherheit zumindest theoretisch möglich. Nach dem von Winner in [132] präsentierten Ansatz, kann die zu leistende Referenzfahrstrecke über eine Bestimmung des Erwartungswertes für die Sicherheit der Fahrfunktion abgeleitet werden. Als Maß für die Sicherheit kann die Wahrscheinlichkeit bestimmt werden, wie oft das Fahrzeug mit aktivierter Funktion in einen Unfall, beispielsweise mit Personenschaden, verwickelt ist.

Als Referenz kann die Unfallwahrscheinlichkeit für nicht automatisierte, also herkömmlich vom Fahrer geführte, Fahrzeuge herangezogen werden. Auf deutschen Autobahnen wurden im Jahr 2018 beispielsweise 20.537 Verkehrsunfälle mit Personenschaden registriert[31]. Bezogen auf eine Gesamtfahrleistung von ca. 250 Mrd. km auf den Autobahnen im Jahr, entspricht dies im Mittel etwa einem Unfall pro 12,2 Mio. km. Besteht nun beispielsweise die (Kunden-)Anforderung, dass die Fahrfunktion, hier beispielsweise ein Autobahnpilot, „doppelt“ so sicher ist, wie der menschliche Referenzfahrer, erhöht sich der Erwartungswert der Strecke pro Unfall für das zu testende System auf 24,4 Mio. km. Bezieht man darüber hinaus noch ein, dass im Mittel nur 50% der Unfälle selbst verursacht sind, steigt der Erwartungswert (Fehler des Systems mit Unfallfolge) auf einen selbstverschuldeten Unfall pro 48,4 Mio. km.

Für die Bestimmung der Referenzstrecke wird nach [132] die Poisson-Verteilung (vgl. [133]) herangezogen, mit der die Wahrscheinlichkeit berechnet wird, mit der k Ereignisse bei einem Erwartungswert λ auftreten:

[30] Bei der formalen Wissensrepräsentation kann zwischen der Open-World und der Closed-world Annahme unterschieden werden. Eine Modellierung unter der Closed-World Annahme geht davon aus, dass alles was nicht im Modell existiert und unbekannt ist, zwingend als falsch abgeleitet wird. Bei einer Modellierung unter der Open-World Annahme gilt dieser Rückschluss nicht (vgl. [210]).

[31] Vgl. https://www.dvr.de/unfallstatistik/de/autobahnen/

$$P_\lambda(k) = \frac{\lambda^k}{k!} e^{-\lambda} \tag{4.1}$$

Der Erwartungswert λ ergibt sich aus Multiplikation der Unfallwahrscheinlichkeit *p* (entspricht dem Kehrwert der Distanz zwischen zwei selbstverschuldeten Unfällen) und dem Stichprobenumfang *n* (also hier der zurückgelegten Distanz im Test).

In der Argumentation bedeutet das, tritt kein Ereignis (k = 0) während des Tests auf, wird bei einer 5%-Irrtumswahrscheinlichkeit die 3-fache Referenzstrecke ($\lambda = 3$) benötigt. Leider ist dieser Fall für eine Funktion, die „nur" doppelt so gut ist wie das Referenzsystem, recht unwahrscheinlich (p = 22,3%, für k=0 und $\lambda = 1{,}5$), sodass dieser Versuch entweder misslingt oder aber kaum wiederholbar ist. In [132] wird daher ein Streckenfaktor von 10 bis 20 der Referenzstrecke abgeleitet, damit der Nachweis trotz potentieller Ereignisse zu mindestens 50% bei einer 5%-Irrtumswahrscheinlichkeit gelingt.

Für das Zahlenbeispiel des Autobahnpiloten entspricht dies einem Testbedarf von 244 Mio. bis 488 Mio. Kilometern. Anzumerken ist, dass etwaige zwischenzeitliche Änderungen in der Funktion zu einem erneuten benötigten Nachweis führen.

Nimmt man für die 488 Mio. Kilometer zum Beispiel eine auf der Autobahn erzielbare Durchschnittsgeschwindigkeit von 100 km/h an und setzt 200 mit der zu testenden Funktion ausgerüstete Prototypen 250 Tage pro Jahr und 12 Stunden pro Tag ein, werden über 8 Jahre benötigt, um den Nachweis zu erzielen.

4.1.2 A2 – Das Dilemma des zufälligen Testens

Annahme A2: Ein zufälliges Testen im Rahmen einer physischen Breitenerprobung führt aus Sicht eines szenariobasierten Ansatzes zu einer Vielzahl an redundanten Szenarien und ist damit ineffizient.

Eine wesentliche Ursache für die Ineffizienz der physischen Breitenerprobung wird transparent, wenn man die Breitenerprobung aus Sicht des szenariobasierten Ansatzes betrachtet: Über große zurückgelegte Strecken machen die kritischen relevanten Situationen nur einen kleinen Anteil der insgesamt auftretenden Szenarien aus. Darüber hinaus tritt eine Vielzahl der auftretenden Szenarien mehrfach redundant auf. Beispielsweise kann man annehmen, dass ein Szenario „freie Fahrt“ (keine weiteren Verkehrsteilnehmer in Sichtweite), auf Autobahnen durchaus vor allem abseits der Verkehrsstoßzeiten relativ häufig und mit vergleichbaren Szenarioeigenschaften vorkommt.

Dieses *Dilemma des zufälligen Testens* wird als Begründung für diesen Zusammenhang wie folgt hergeleitet.

Die Grundlage bildet ein einfaches Gedankenexperiment. Bereits in der Einleitung (vgl. Kapitel 1.1.2) wurde ein Beispiel zur Testfallkombinatorik dargestellt, bei dem allein durch Variation der Wetter- und Umgebungsbedingungen 6.561 Varianten eines Szenarios erzeugt wurden. Erweitert man dieses Beispiel um typische mögliche Startparameter (hier: Startgeschwindigkeit des Überholers, Startgeschwindigkeit des Überholenden, Startabstand) eines einfachen Überholszenarios auf der Autobahn, erhält man bei der dargestellten Diskretisierung (siehe Abbildung 4.1) 7.600 Varianten für die Startszene des Szenarios.

Kombiniert man die hier dargestellten Varianten für die Sensorfunktion mit den Startvarianten für die Regelfunktion, ergeben sich für dieses Szenario schon ungefähr mögliche 50 Mio. unterschiedliche Variationen (7.600 x 6.561).

Unter der Annahme, dass die 50 Mio. Variationen des Überholszenarios den vollständigen Ereignisraum darstellen, stellt sich die Frage, wie lange es dau-

ert, respektive wie viele Szenarien im Rahmen einer Breitenerprobung (zufällige Fahrt in der realen Welt) voraussichtlich durchfahren werden müssten, um sämtlichen 50 Mio. Variationen zu begegnen.

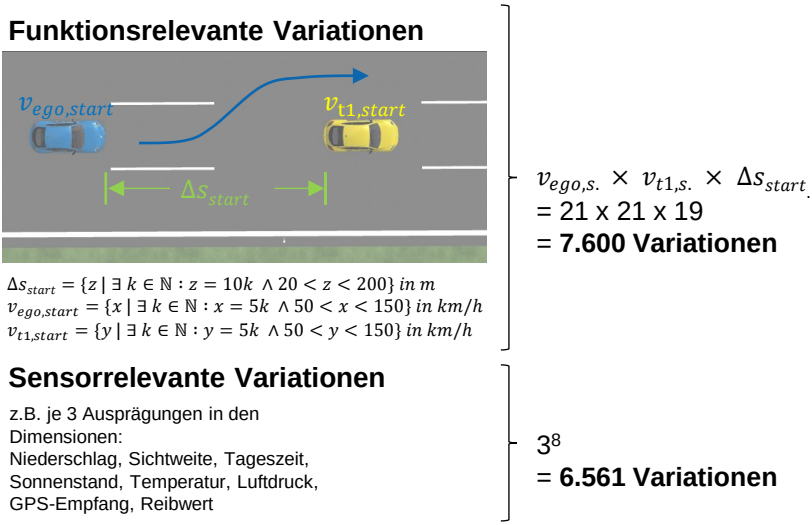


Abbildung 4.1: Mögliche Variationen für die Startszene eines Überholszenarios

Das sogenannte Coupon Collector Problem[32] beschreibt einen mathematischen Modellierungsansatz, der sich für diese Fragestellung anwenden lässt. Der Erwartungswert für die Anzahl der zu fahrenden Szenarien, damit eine Anzahl von *m* gleichwahrscheinlichen Szenarien vollständig gefunden wird, ergibt sich zu (vgl. [134]):

$$E(m) = m\left(\frac{1}{m} + \frac{1}{m-1} + \frac{1}{m-2} + \cdots + \frac{1}{1}\right) = mH_m \qquad (4.2)$$

[32] Das Coupon Collector Problem (dt.: Sammlerproblem) beschreibt ein klassisches Problem der Wahrscheinlichkeitstheorie, das Fragestellungen wie beispielsweise "Eine Firma legt ihrem Produkt die Bilder von Spielern der Nationalmannschaft bei. Wie viele Produkte müssen im Mittel gekauft werden um alle Bilder zu erhalten?" [211] (S.118) beantwortet.

Mit H_m als m-te harmonische Zahl.

Es kann gezeigt werden, dass die Differenz $H_m - ln(m)$ für große m gegen die Euler-Mascheroni-Konstante γ ($\gamma \approx 0{,}5772\ldots$) konvergiert (vgl. [135]).

Damit ergibt sich für $E(m)$ folgende Approximation:

$$E(m) \approx m \cdot (\ln(m) + \gamma) \tag{4.3}$$

Mit $m = 5 \cdot 10^7$ ergibt sich $E(m) \approx 9{,}15 \cdot 10^8$. Es müssten also über 900 Mio. Szenarien gefahren werden, um erwartungsgemäß eine Abdeckung der gesuchten 50 Mio. Szenarien (Abdeckungsraum) zu erreichen.

Allerdings gilt dies nur unter der vereinfachenden Annahme, dass die Szenarien gleichverteilt sind, also das Auftreten jedes einzelnen Szenarios gleichwahrscheinlich ist.

Die tatsächliche Verteilung der Szenarien ist nicht bekannt. Eine realistischere Annahme als die Gleichverteilung der Szenarien ist eine Verteilung der Szenarien nach dem Zipf'schen Gesetz: Ordnet man die Szenarien nach ihrer Häufigkeit des Auftretens, soll die Wahrscheinlichkeit des Auftretens eines Szenarios genau umgekehrt proportional zur Position innerhalb der Ordnung sein.[33]

Unter der Anwendung der Zipfverteilung, ergibt sich der neue Erwartungswert zu (vgl. [134]):

$$E_{zipf}(m) \approx m \cdot \ln(m)^2 \tag{4.4}$$

Wiederum mit $m = 5 \cdot 10^7$ ergibt sich $E_{zipf}(m) \approx 1{,}57 \cdot 10^{10}$. Demnach müssten bereits über 15 Mrd. Szenarien gefahren werden, um erwartungsgemäß eine Abdeckung der 50 Mio. Szenarien zu erreichen. Der reine Erwartungswert ist allerdings nicht belastbar für eine Absicherung, sagt er schließlich nur aus, dass

[33] Die Zipfverteilung entstand ursprünglich aus der Analyse der Verteilung von Wörtern in einer Sprache. Verallgemeinerte Formen dieser Potenzverteilung lassen sich in der Beschreibung der Verteilungen vieler natürlicher Prozesse, wie der Verteilung von Stadtgrößen oder des Einkommens wiederfinden [212].

durchschnittlich diese Zahl an Szenarien gefahren werden müsste, wenn das Experiment oft wiederholt würde. Um eine belastbare Aussage im „ersten" Versuch der Erprobung treffen zu können, ist die Wahrscheinlichkeit für die Erreichung dieses Ergebnisses mit einzubeziehen.

In [136] wird der Grenzwert für das verallgemeinerte Coupon Collector Problem für beliebige Verteilungen bestimmt:

$$P\left\{\frac{T_m - b_m}{k_m} \leq y\right\} \rightarrow exp(e^{-y}), \quad \text{für } alle\ y \in \mathbb{R}\ und\ m \rightarrow \infty \qquad (4.5)$$

mit

$$b_m = A_m f(m)\left[\ln\left(\frac{f(m)}{f'(m)}\right) - \ln\ln\left(\frac{f(m)}{f'(m)}\right)\right] \text{ und } k_m = A_m f(m) \qquad (4.6)$$

Unter der Annahme der Zipfverteilung und des hier dargestellten Problems gilt nach [136]: m = 50 Mio. (Anzahl der gesuchten Szenarien), $f(m) = m$, $ln\ (f(m)/f'(m)) = ln\ m$ und $A_m = H_{50*10\wedge6} \approx 18{,}3047$ (mit Approximation vgl. (4.3)).

Für hier somit $b_{50*10\wedge6} = 1{,}3593 \cdot 10^{10}$ und $k_{50*10\wedge6} = 9{,}1523 \cdot 10^{8}$ ergibt sich beispielsweise für eine 99% Wahrscheinlichkeit $y = -\ln[-\ln(0{,}99)] = 4{,}6001$. Mit (4.5) ergibt sich für die benötigte Anzahl der Szenarien $T_{50*10\wedge6} = y \cdot k_{50*10\wedge6} + b_{50*10\wedge6} = 1{,}78 \cdot 10^{10}$. Durch den exponentiellen Zusammenhang belegt dieses Modell damit, dass eine 100% Abdeckung der gesuchten Szenarien durch einen zufälligen Test in der physikalischen Breitenerprobung nicht garantiert werden kann (vgl. Abbildung 4.2). Nimmt man eine 99% Wahrscheinlichkeit als ausreichend an, werden bereits über Faktor 300 redundante Überholszenarien ($1{,}78 \cdot 10^{10} / 50 \cdot 10^{6}$) benötigt, als letztendlich abgetestet werden.

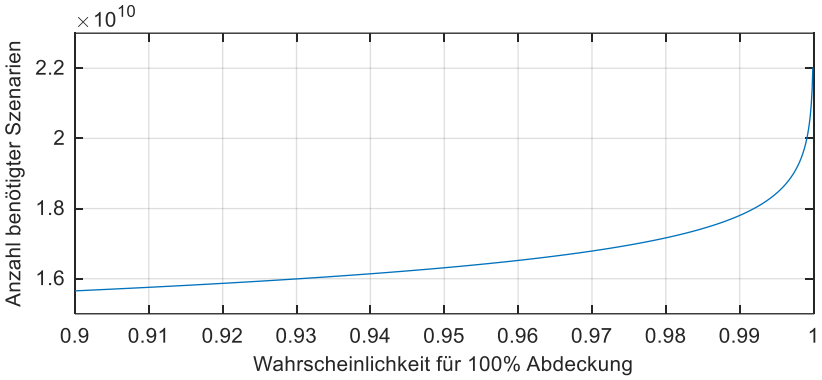


Abbildung 4.2: Anzahl benötigter Szenarien für 50 Mio. gesuchte Szenarien (Abdeckungsraum) über der Wahrscheinlichkeit einer 100% Abdeckung unter der Annahme einer Zipfverteilung

4.1.3 A3 – Testabdeckung in der Simulation

Annahme A3: Eine vollständige Testabdeckung kann in der Simulation nicht (ausschließlich) systematisch durch rein künstliche Verfahren wie Parametervariation und –kombinatorik sichergestellt werden.

Bereits das unter Annahme A2 gezeigte Beispiel legt dar, dass eine Variation von bereits wenigen Szenarioparametern zu einer Explosion an konkreten Szenarien führt. Auch effiziente Simulationsumgebungen, stoßen mit ihrer Leistungsfähigkeit schnell an Grenzen, wenn die Parameter potentieller Szenarien rein kombinatorisch und ohne weitere Restriktionen variiert werden. Geht man beispielsweise von den 50 Mio. Szenarien aus Annahme A2 aus, würde eine Simulationsumgebung, selbst wenn sie vollständig automatisiert und 24 Stunden pro Tag betrieben wird, bereits über 47 Jahre benötigen, nimmt man eine durchschnittliche Simulationszeit von 30 s pro Szenario an:

$$50 \cdot 10^6 \cdot \frac{30s}{1} \cdot \frac{1h}{3600s} \cdot \frac{1d}{24h} \cdot \frac{1a}{365d} = 47{,}6\text{a} \tag{4.7}$$

Erweitert man das dargestellte Überholszenario um weitere potentielle Verkehrsteilnehmer, die ebenfalls mit variierbaren Parametern verknüpft sind, erhält man nahezu beliebig hohe Aufwände, die auch durch Möglichkeiten der Parallelisierung oder Erhöhung des Echtzeitfaktors in der Simulation (beispielsweise für Model-in-the-Loop oder Software-in-the-Loop) nicht mehr realisierbar sind.

4.1.4 A4 – Szenarienvielfalt und Testfälle

Annahme A4: Eine bestimmte notwendige Menge an real gefahrenen Kilometern enthält eine hinreichende Vielfalt an Testfällen, um statistisch die Güte einer Funktion belegen zu können.

Die statistische Herleitung einer Referenzstrecke (vgl. Annahme A1) zeigt bereits, dass eine entsprechende Menge an zufälligen, real gefahrenen Kilometern bestimmt werden kann, die die Sicherheit einer Funktion unter einer gewissen Irrtumswahrscheinlichkeit nachweisen kann. Betrachtet man Testfälle als Ausprägungen konkreter Szenarien mit entsprechenden „Pass/Fail“-Kriterien (vgl. Kapitel 3.1.4) und berücksichtigt, dass mit jedem gefahrenen (Test)kilometer auch eine Zahl an durchfahrenen Szenarien einhergeht, enthalten diese zwangsweise auch daraus ableitbare Testfälle.

Mit dem Beispiel aus Annahme A2 wird gezeigt, dass eine bestimmte Anzahl an Szenarien nur mit einer hohen Sicherheit gefunden werden kann, wenn der Einsatz der zu fahrenden Szenarien die zu suchende Menge um Größenordnungen übersteigt. Allerdings gilt auch der Umkehrschluss: Eine bestimmte Zahl an gefahrenen Szenarien enthält statistisch mit einer bestimmten Wahrscheinlichkeit auch einen Anteil *verschiedener* Szenarien. Wird eine ausreichende Strecke gefahren, beispielsweise die Referenzstrecke aus Annahme A1, ist damit davon auszugehen, dass auch ausreichend verschiedene Szenarien und damit Testfälle in dieser Strecke enthalten sind.

4.2 Ableitung der Leitfrage für das Lösungskonzept

Aus den vorangegangen Abschnitten lassen sich folgende Schlussfolgerungen ziehen, die im Kern die Unterschiede für die Absicherung von Funktionen des hochautomatisierten Fahrens im Vergleich zu herkömmlichen Funktionen in der Automobilentwicklung beschreiben:

1. Die Absicherungsaufgabe ist ein „Open-World-Problem“. Eine systematische Ableitung aller Systemgrenzen und der damit verbundenen Testfälle ist, zumindest mit herkömmlichen Methoden, nicht möglich.

2. Aus den fehlenden Anforderungen entsteht eine „Lücke“ im Testprozess. Es ist unklar, wie getestet werden muss und wie die Vollständigkeit des Tests nachgewiesen werden kann.

3. Der menschliche Fahrer kann hinsichtlich seiner Leistungsfähigkeit als Referenz für eine hochautomatisierte Fahrfunktion herangezogen werden. Zumindest kann davon ausgegangen werden, dass eine Akzeptanz für das System nur dann möglich wird, wenn nachgewiesen werden kann, dass das System hinsichtlich seiner Leistungsfähigkeit (beispielsweise als Maß der verschuldeten Unfälle pro Kilometer) mindestens genauso gut wie der Mensch abschneidet.

4. Auch wenn die Leistungsfähigkeit grundsätzlich theoretisch-statistisch über eine entsprechende Breitenerprobung nachgewiesen werden könnte, ist die Umsetzung, zumindest mit teuren Prototypen-Fahrzeugen aufgrund des Dilemmas des zufälligen Testens nicht nur ineffizient, sondern auch keine ökonomisch sinnvolle Möglichkeit.

Hieraus ergibt sich für ein Lösungskonzept folgende Forschungsfrage:

Können relevante Szenarien und Testfälle aus dem Referenzsystem „menschlicher Fahrer“ abgeleitet, effizient dargestellt und mit herkömmlichen Testmethoden durchgeführt werden, ohne dass für den Test voll-funktionale Prototypenfahrzeuge eingesetzt werden müssen (vgl. auch Abbildung 4.3)?

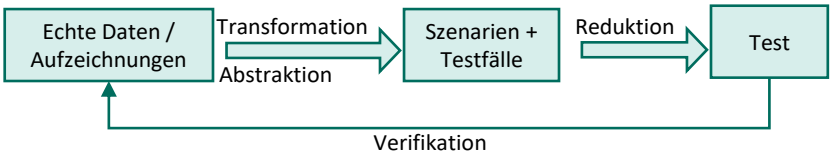


Abbildung 4.3: Prinzipskizze zur Leitfrage

5 Konzept: Real-szenariobasierte simulationsgestützte Absicherung

Auf Basis der in Kapitel 1 aufgezeigten Motivation, den vorgestellten Grundlagen aus Kapitel 2, dem Stand der Technik und Forschung aus Kapitel 3 und der Forschungsfrage aus Kapitel 4, wird in diesem Kapitel ein Lösungsvorschlag dargestellt, wie der Test und die Absicherung für eine automatisierte Fahrfunktion simulationsgestützt auf Basis von realen Szenarien, die beispielsweise von einer Fahrzeugflotte aufgezeichnet werden, erfolgen kann.

5.1 Ziele und Anforderungen

Das übergeordnete Ziel ist es, ein vollständiges Vorgehensmodell zu entwickeln, das es ermöglicht, Szenarien in real gefahrenen Situationen zu identifizieren, ihre Elemente zu klassifizieren und in der Simulation abbildbar zu machen, um sie damit herkömmlichen Methoden im Rahmen des Entwicklungsprozesses (vergleiche X-in-the-Loop Methode) zur Verfügung stellen zu können. Diesem Ziel werden folgende Unterziele und Anforderungen untergeordnet:

Z1: Daten aus realen (Sensor-)Aufzeichnungen können auf einen, jeweils für den Anwendungsfall passenden, Abstraktionsgrad reduziert werden, um als Grundlage für die zu extrahierenden Szenarien zur Verfügung zu stehen.

Z2: Aus den realen Daten können charakteristische Elemente wie Manöver identifiziert und abgeleitet werden, die wiederum die Grundlage für daraus zusammengesetzte Szenariorepräsentationen bilden.

Z3: Die ermittelten Szenarien aus den realen Daten können in einem stetig erweiterbaren Szenariokatalog gespeichert werden, aus dem Testfälle für die Anwendung der herkömmlichen X-in-the-Loop Methodik abgeleitet werden können.

Z4: Die Abstraktion kann in der Simulation hinreichend ab- beziehungsweise nachgebildet werden, damit sichergestellt ist, dass die Simulation die ursprünglich aufgezeichneten Situationen mit einer ausreichenden Genauigkeit repräsentiert.

5.2 Annahmen im Konzept

Dem Konzept und seinen Zielen liegen, basierend auf den Ausführungen der vorigen Kapitel, die folgenden Annahmen zugrunde:

Z1: Daten aus realen (Sensor-)Aufzeichnungen können auf einen, jeweils für den Anwendungsfall passenden, Abstraktionsgrad reduziert werden, um als Grundlage für die zu extrahierenden Szenarien zur Verfügung zu stehen.

Z1-A1: Sensoren schaffen Abbildungen der Realität. Sie weisen Schwankungen und Grenzen hinsichtlich (Mess-)Genauigkeit, Auflösung und Zuverlässigkeit auf [137]. Mit herkömmlicher Seriensensorik ist daher eine vollständige, fehlerlose Erfassung der Umgebung nicht möglich. Auch der Einsatz präziserer Referenzsensorik kann diese Problematik nicht vollständig lösen nur verbessern. In Zusammenhang mit Z1-A2 wird davon ausgegangen, dass trotz dieser Einschränkungen Sensoraufzeichnungen zum Test der nachgelagerten Funktionen geeignet sind.

Z1-A2: Die Wirkkette einer Fahrfunktion, von der Umweltwahrnehmung bis zur Umsetzung der Fahraufgabe, lässt sich in sinnvolle, voneinander auch getrennt testbare, Teilschritte zerlegen.

Das Konzept der funktionalen Dekomposition [138] beschreibt einen Ansatz, in dem diese Trennung der Fahraufgabe in sechs aufeinanderfolgende Layer

der Wirkkette erfolgt (vergleiche Abbildung 5.1). Das Ziel dieses Ansatzes ist es, die Anzahl der Szenarien und Testfälle reduzieren zu können, indem nur die jeweils relevanten Tests für einen Layer durchgeführt werden müssen. Zum Beispiel macht es für das potentielle Szenario „an einem statischen Objekt vorbeifahren“ für die Planungs- oder Ausführungsaufgabe keinen Unterschied, um was für eine Art Objekt es sich handelt. Für die Umweltwahrnehmung und –interpretation kann es jedoch entscheidend sein, ob es sich bei dem statischen Objekt beispielsweise um eine Hecke oder einen Leitpfosten handelt [138].

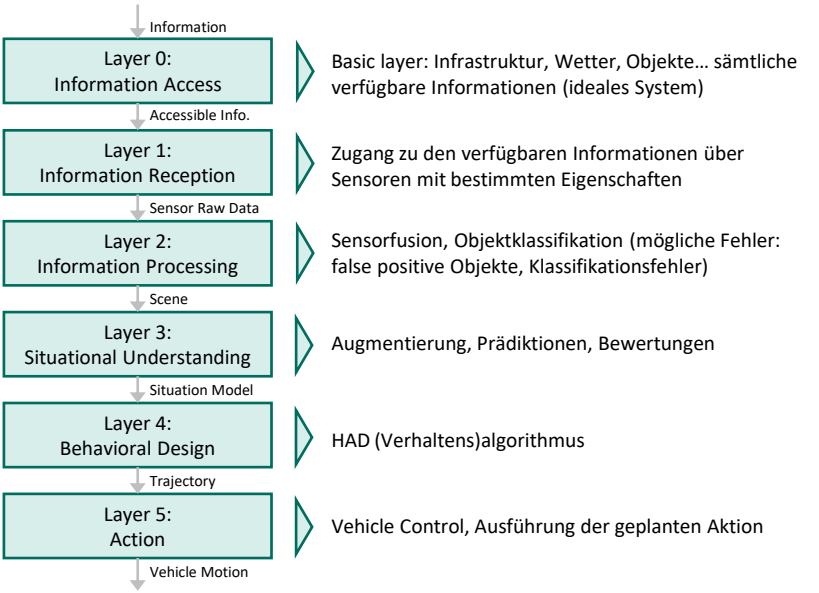


Abbildung 5.1: Layer der funktionalen Dekomposition nach [138] (eigene Darstellung)

Mit Realdaten aus Sensoraufzeichnungen ist die vollständige Kette nach dem Konzept der funktionalen Dekomposition nicht abdeckbar, da dies für die Quelle der Realdaten ideale, perfekte Sensoren voraussetzen würde (beispielsweise für den Layer 0). Dennoch bietet der Ansatz eine Möglichkeit sinnvolle Teilaspekte der Wirkkette untersuchbar zu machen.

Z2: Aus den realen Daten können charakteristische Elemente wie Manöver identifiziert und abgeleitet werden, die wiederum die Grundlage für daraus zusammengesetzte Szenariorepräsentationen bilden.

Z2-A1: Geeignete formale Beschreibungsverfahren ermöglichen es, Szenarien aus einer Metaebene heraus zu definieren.

Verschiedene Arbeiten beschäftigen sich bereits mit der Definition des Szenariobegriffs oder schaffen Beschreibungen für die relevanten Eigenschaften von Szenarien (vergleiche Kapitel 3.1.4 und [88] [94] [93] [139]). Im Rahmen des Forschungsprojekts PEGASUS werden Szenarien auf verschiedenen Abstraktionsebenen definiert und der Unterscheidung zwischen funktionalen, logischen und konkreten Szenarien [140]. Die Beschreibung der Szenarien stützt sich dabei auf fünf Ebenen, die die Strukturierung des Szenarios in seinen Bestandteilen darstellen (Abbildung 5.2). Die Ebene 1 deckt das Straßennetz ab, definiert Fahrbahnen und Verläufe sowie Fahrstreifen und Eigenschaften der Topologie. In der Ebene 2 wird die Straßenausstattung bestimmt. Dazu gehören beispielsweise Verkehrszeichen mit ihren Eigenschaften und Positionen. In der Ebene 3 werden temporäre Beeinflussungen oder Veränderungen wie beispielsweise durch Baustellen abgebildet. Während die Ebenen 1 bis 3 eher auf die statischen Eigenschaften des Szenarios konzentrieren, bilden die Ebenen 4 und 5 die dynamischen Bestandteile ab. Hierzu gehören bewegliche Objekte wie Verkehrsteilnehmer mit ihren Eigenschaften und Aktionen (Ebene 4) aber auch Umweltbedingungen (Ebene 5).

Formate wie OpenDRIVE (Ebene 1 bis 3) und OpenSCENARIO (Ebene 4 und 5) stellen aktuelle Bestreben dar, standardisierte Beschreibungsformen zu schaffen, die diese Ebenen abdecken [141], fokussieren allerdings nicht explizit auf den Prozess der Gewinnung der benötigten Informationen aus Datenaufzeichnungen.

Abbildung 5.2: Ebenen-Modell für die Repräsentation von Szenarien, in blau exemplarische Eigenschaften, in gelb exemplarische Parameter [142]

Z2-A2: Die Grundbausteine für die dynamischen Elemente der Szenarien stellen die darin enthaltenen Objekte und ihre Manöver dar.

Nach Ulbrich [93] beschreibt ein Szenario die zeitliche Entwicklung von Folgen von Szenen, die über Aktionen und Ereignisse miteinander verknüpft sind. Folglich stellen die dynamischen Elemente (Manöver der Objekte und des Egofahrzeugs) den Kern für die zeitliche Entwicklung des Szenarios dar, während die statischen Komponenten, wie die Fahrbahn, Bebauungen, Schilder etc. hinsichtlich der zeitlichen Betrachtung des Szenarios immer gleich bleiben. In diesem Sinne ist ein einheitliches Manöverkonzept notwendig, welches in der Lage ist, diese Bestandteile der Szenarien auf einer geeigneten Abstraktionsebene abzubilden.

Z2-A3: Es existieren Methoden oder es lassen sich Verfahren finden, die es ermöglichen, die benötigten Elemente zur Beschreibung der Szenarien aus den Datenaufzeichnungen eines Fahrzeugs zu extrahieren.

Damit das realdatengestütztes Konzept tatsächlich anwendbar ist, müssen Verfahren gefunden werden, die es ermöglichen, die Szenarioabbildung aus den aufgezeichneten Daten zu generieren.

In [143] wird bereits ein Ansatz vorgestellt, der es ermöglicht, Daten aus real gefahrenen Szenarien über eine geeignete Referenzsensorik in eine Simulation zu überführen. Dies entspricht einer reinen Transformation der Daten in die Simulationsumgebung (vergleiche unterer Pfad in Abbildung 5.4). Für den hier vorgestellten Zweck müssen die einzelnen Szenarioelemente wie die enthaltenen Objekte und ihre Manöver in den Daten mit einer ausreichenden Genauigkeit erkannt und extrahiert werden können. Betrachtet man die Szenarien als Zeitreihenabschnitte aus den aufgezeichneten Daten, sind Verfahren der Zeitreihenanalyse anwendbar. In [144] werden aktuelle Verfahren beziehungsweise Techniken der Zeitreihenanalyse und –klassifikation zusammengefasst. Im Implementierungsteil werden verschiedene Verfahren wie Dynamic Time Warping und künstliche neuronale Netze zur Anwendung auf die Manöverextraktion aus den Daten untersucht.

Z3: Die ermittelten Szenarien aus den realen Daten können in einem stetig erweiterbaren Szenariokatalog gespeichert werden, aus dem Testfälle für die Anwendung der herkömmlichen X-in-the-Loop Methodik abgeleitet werden können.

Z3-A1: Um die Szenarien zu katalogisieren, müssen sie nach bestimmten Kriterien in Klassen einteilbar sein.

Wendet man ein geeignetes Abstraktionsmodell für die Szenariorepräsentation und bildet seine Elemente ab, lassen sich die Sequenzen der Elemente in Klassen abbilden. Ein Beispiel stellt die wörterbuchbasierte Klassifikation dar (siehe Abbildung 5.3) [144]. Identifizierte Abschnitte wie die Manöver aus der Zeitreihe einer Aufzeichnung werden einzelnen Buchstaben zugeordnet, sodass sich für dieses einfache Beispiel für das Szenario „Überholen links“ die

Wortgruppe „abac“ hinsichtlich der zugrundliegenden Manöversequenz der Querführung (Fahrstreifen folgen – Fahrstreifenwechsel links – Fahrstreifen folgen – Fahrstreifenwechsel rechts) des Egofahrzeugs ergibt. Auf diese Weise erzeugte Wortgruppen können schematisch abgespeichert werden.

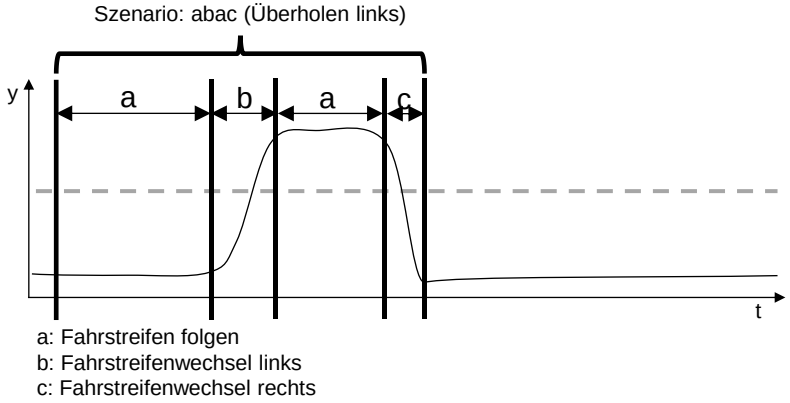


Abbildung 5.3: Prinzip der wörterbuchbasierten Klassifikation zur Beschreibung des Szenarios “Überholen links” anhand der zugrundeliegenden Manöversequenz

Z3-A2: Es existieren Methoden oder es lassen sich Verfahren finden, die eine Vergleichbarkeit zwischen Szenarien nach zu bestimmenden Kriterien ermöglichen.

Das zuvor in Z3-A1 gezeigte Beispiel verdeutlicht, wie die Anwendung eines geeigneten Abstraktionsmodells auf den aufgezeichneten Daten helfen kann, die klassifizierten Szenarien miteinander vergleichbar zu machen. So kann beispielsweise die Wortgruppe „abac“ als Repräsentant für das Szenario „Überholen links“ betrachtet werden. Wendet man das Abstraktionsmodell sowohl auf die generierten Szenarien aus dem Katalog als auch auf die originalen Daten an, können die simulierten Szenarien aus dem Katalog auf diese Weise verifiziert werden (vergleiche Abbildung 5.4).

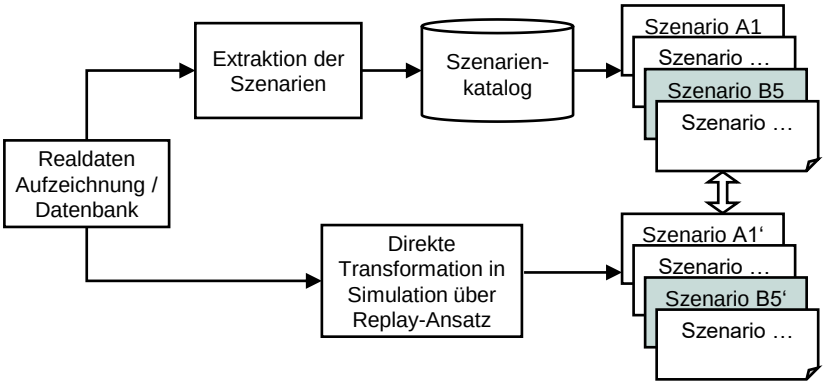


Abbildung 5.4: Prinzip des Abgleichs der erzeugten Szenarien mit den ursprünglichen Daten

Z4: Die Abstraktion kann in der Simulation hinreichend ab- beziehungsweise nachgebildet werden, damit sichergestellt ist, dass die Simulation die ursprünglich aufgezeichneten Situationen mit einer ausreichenden Genauigkeit repräsentiert.

Z4-A1: Eine Aufteilung der Wirkkette der Fahrfunktion in sinnvolle Teilschritte vereinfacht die erforderliche Modellrepräsentationen in der Simulation.

Simulationsmodelle stellen stets eine Repräsentation und Abstraktion des realen Systems dar. Modellvalidierungstechniken überprüfen die Korrelation der Modelausgaben O_k^m mit den Systemausgaben O_k^s unter Verwendung des gleichen Sets an Model- und Systemeingaben I_k^m bzw. I_k^s (vergleiche Abbildung 5.5). Je komplexer also das reale Vorbild ist, desto aufwändiger gestaltet sich auch das erforderliche Modell und damit auch die Herausforderung an die Modellvalidierung [145]. Das Konzept der funktionalen Dekomposition kann hier als Ansatz dienen, die Gesamtkomplexität zu reduzieren, da die Betrachtung auf der Ebene einzelner Layer die Verwendung einfacherer Modelle zulässt.

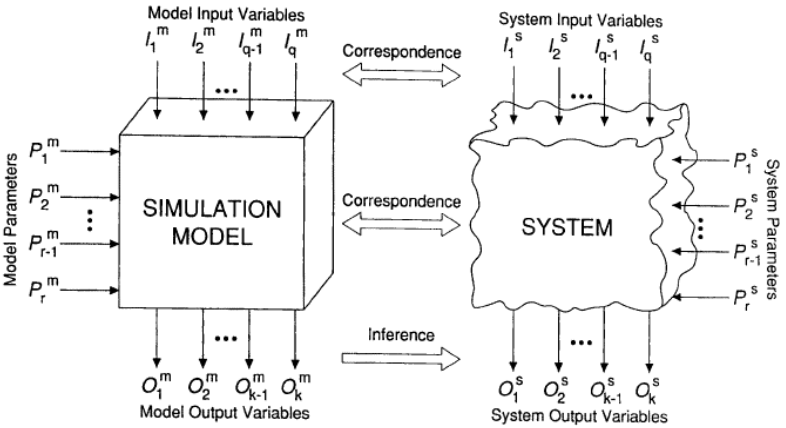


Abbildung 5.5: Modell und Systemeigenschaften [145]

Z4-A2: Für Teile der Wirkkette lassen sich Modelle finden, die einen hinreichenden Genauigkeitsgrad besitzen.

Betrachtet man die einzelnen Layer der funktionalen Dekomposition (vgl. Abbildung 5.1) auf der Abstraktionsebene des Informationsaustauschs zwischen den Layern, ergibt sich folgendes Bild ohne Anspruch auf Vollständigkeit (Abbildung 5.6):

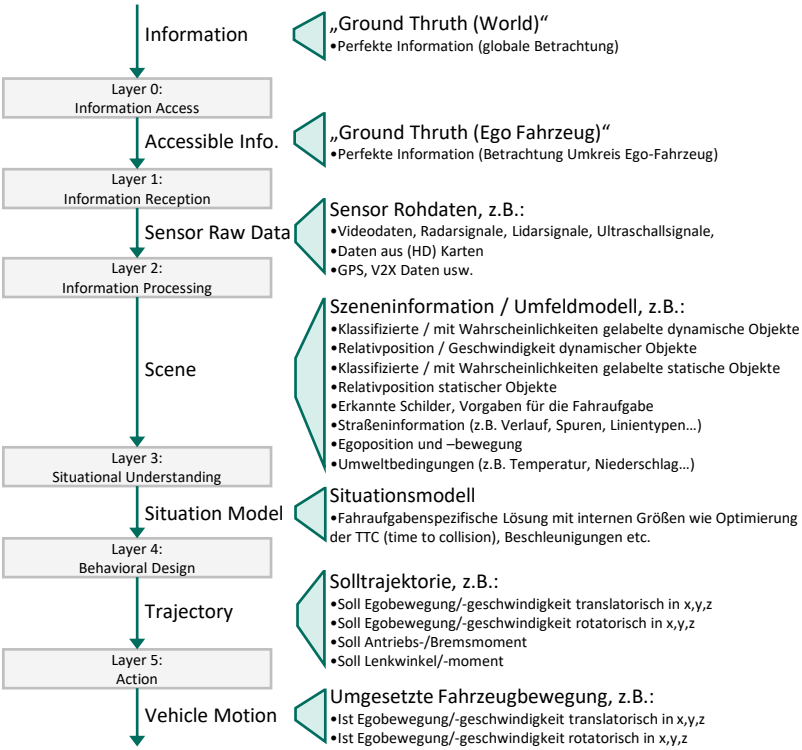


Abbildung 5.6: Informationsschnittstellen zwischen den Layern

Für die Umsetzung in der Simulation stellt es also einen Unterschied dar, welche Layer im Simulationsmodell berücksichtigt bzw. abgebildet werden. Dies zeigt sich auch im Reifegrad bereits vorhandener Simulationslösungen. So benötigt eine Abbildung der Eigenschaften der Layer 0,1 und 2 Modelle, die Sensoreigenschaften auf der Ebene des jeweiligen physikalischen Verhaltens darstellen können. Während die Abbildung des Fahrverhaltens (Layer 3 – 5) bereits seit Jahren im Fokus der Simulation stand und für viele Anwendungen bereits eine hohe Güte erreicht hat (vergleiche beispielsweise „simulationsbasierte Homologation“ [5]), befindet sich die Entwicklung geeigneter und vor allem validierter Modelle für Sensoreigenschaften noch relativ am Anfang.

Des Weiteren stützt sich die Umsetzung in der Simulation nach dem hier vorgeschlagenen Vorgehen auf die Daten aus der Sensoraufzeichnung. Diese enthalten zwangsweise „Fehler“ in den ersten Layern (vergleiche auch Z1). Aus diesen Gründen fokussiert sich der Ansatz dieser Arbeit auf geeignete Daten der Layer 3 – 5.

5.3 Prozess

Das zuvor beschriebene Konzept lässt sich mit seinen Zielen in folgende Prozessdarstellung überführen (Abbildung 5.7):

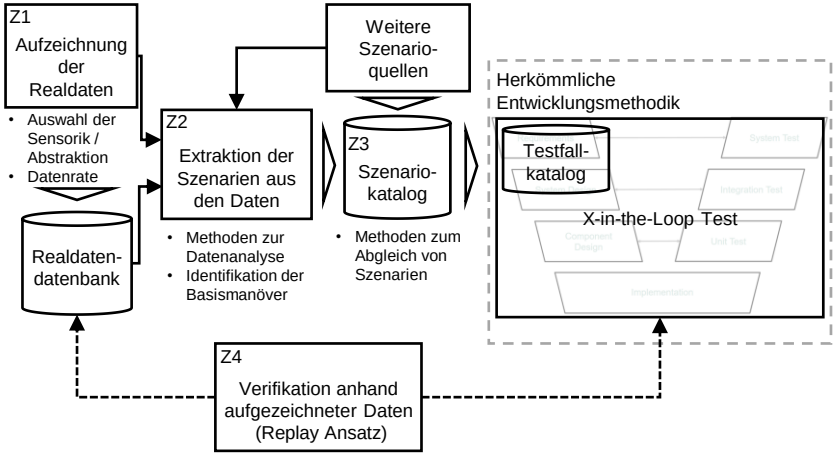


Abbildung 5.7: Prozess zur real-szenariobasierten simulationsgestützte Absicherung

Aus Prozesssicht werden zunächst Daten aus realen Fahrsituationen mittels geeigneter Sensoren aufgezeichnet und gespeichert. Auf den vorverarbeiteten Daten werden auf den Zeitreihen der Objektinformationen zu den Verkehrsobjekten und des Egofahrzeugs Verfahren der Zeitreihenanalyse eingesetzt, um die enthaltenen Manöver zu identifizieren und die Szenarien daraus zu bestimmen (vergleiche Z1 / Z2). Anschließend werden die extrahierten Szenarien mit

einem bestehenden Szenarienkatalog abgeglichen und gegebenenfalls in diesen aufgenommen (Z3). Nach der Definition von geeigneten Pass- / Failkriterien werden aus den Szenarien konkrete Testfälle [55], die anschließend für das Testen einer hochautomatisierten Fahrfunktion herangezogen werden können. Die Verwendung der Testfälle kann nach herkömmlichem Vorgehen im Entwicklungsprozess erfolgen (vergleiche X-in-the-Loop Methode). Dabei können zu jedem Zeitpunkt die verwendeten Szenarien anhand der originalen aufgezeichneten Daten auf verschiedenen Betrachtungsebenen verifiziert werden (Z4).

5.4 Einschränkungen

Der Prozess und die Herangehensweise weisen Einschränkungen auf, die im Folgenden erläutert werden:

E1: Fehler in den Sensorlayern werden nicht entdeckt und führen unter Umständen zu „falschen“ Szenarien und Testfällen.

Der Schritt der Szenarioextraktion basiert auf den Aufzeichnungen aus der Sensorik. Entstehen bei der Aufzeichnung sensorbedingte Fehler, wie beispielsweise nicht erkannte Objekte, können diese auch bei der Extraktion nicht berücksichtigt werden. Das kann unter Umständen zu Szenarien führen, die von den ursprünglichen realen Szenarien abweichen. Für den Test der nachgelagerten Regelfunktion (Layer 3 bis 5) entstehen damit dennoch nutzbare und vollständige Testfälle.

E2: Der Prozess ist nicht oder nur bedingt für die Absicherung der Sensorfunktion geeignet.

Der Fokus hier liegt auf der Absicherung der Funktion für die nachgelagerten Layer. Als Input dienen die Aufzeichnungen, die die eingesetzten Sensoren liefern. Soll auch die korrekte Sensorfunktion oder Sensordatenvorverarbeitung geprüft werden, muss dies in einem separaten Prozess erfolgen. Dies ist nicht Untersuchungsgegenstand dieser Arbeit.

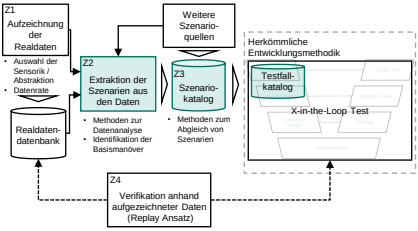
E3: Der Abstraktionsschritt hat immer einen Fokus bezogen auf die zu testende Funktion und ihre Umgebung. Eine Übertragung auf andere Funktionen ist nicht automatisch zulässig und erfordert einen erneuten Durchlauf des Prozesses.

Die Extraktion der Szenarien erfolgt stets im Hinblick auf ein konkretes Ziel. Das Ziel wird durch die zu testende hochautomatisierte Fahrfunktion bestimmt. Beispielsweise macht eine Abstraktion von Abbiegevorgängen aus einer Kreuzungssituation keinen Sinn, wenn die zu testende Funktion einen Autobahnpiloten darstellt. Dementsprechend muss vor dem Durchlaufen des Prozesses die zu testende Funktion feststehen, um geeignete Abstraktionsschritte festlegen zu können. Dennoch können konsequenterweise einmal aufgezeichnete Datensätze für verschiedene Funktionen zur Verfügung stehen. Der Prozess oder zumindest der Abstraktionsschritt muss in diesem Fall separat durchgeführt werden.

6 Modellierung und Methoden

Die Ausarbeitung des zuvor vorgestellten Konzepts erfolgt entlang des Prozessbildes (vgl. Abbildung 5.7). Als Anwendungsbeispiel wird eine Testumgebung für einen fiktiven Autobahnpiloten herangezogen und gestaltet sich wie folgt:

6.1 Entwicklung eines geeigneten Szenario-Metamodells



Zentrales Element des Konzepts ist der Entwurf eines geeigneten Szenario-Metamodells mit dem Ziel eine adäquate Abstraktion für die aufgezeichneten Daten zu bilden, die gleichzeitig als Beschreibungsgrundlage für die späteren Testfälle herangezogen werden kann. Der Fokus dieses Modells liegt vor allem auf der Abbildung der dynamischen Objekte in den Fahrszenarien. An dieses Modell werden folgende Detailanforderungen gestellt:

- Erweiterbarkeit: Die Szenarioextraktion erfolgt anwendungsfallgetrieben. Wie bereits zuvor beschrieben, sind je nach Anwendungsfall unterschiedliche Dimensionen (z.B. der Umwelteigenschaften) von Relevanz für die zu testende Funktion. Dementsprechend muss das Szenario-Metamodell entsprechend der zu betrachtenden Dimensionen anpassbar und erweiterbar sein. Auch können somit statische Elemente der Szenarien, die nicht der Hauptfokus des hier entwickelten Modells sind, mitberücksichtigt werden (vergleiche auch Abschnitt 6.1.6).

- Überführbarkeit: Mit Formaten wie OpenDRIVE und OpenSCENARIO existieren bereits Standardisierungsansätze für die generische Szenariobeschreibung im Simulationsumfeld. Idealerweise sieht das Konzept für das Szenario-Metamodell eine grundsätzliche Kompatibilität und Erweiterbarkeit dieser Formate vor.

- Eignung des Abstraktionsgrades: Das Szenario-Metamodell muss in der Lage sein, relevante Eigenschaften mit einer ausreichenden Genauigkeit abzubilden (vergleiche Z2 aus Kapitel 5.1).

- Anwendbarkeit: Zusätzlich zu den vorherigen Anforderungen muss das Szenario-Metamodell in der Form anwendbar sein, dass die entsprechenden Elemente des Modells aus den aufgezeichneten Daten mit geeigneten Methoden überhaupt ableitbar sind. Andernfalls ist ein Einsatz des Modells für den beschriebenen Prozess nicht praktikabel.

6.1.1 Präzisierung des Manöverbegriffs

Im Stand der Technik wurde der Manöverbegriff als Bestandteil des szenariobasierten Testens eingeführt (vgl. Abschnitt 3.1.4). In der Fachliteratur existieren verschiedene, teils auch widersprüchliche Konzepte, um „Manöver“ als Bestandteil der Verhaltensweisen von Verkehrsteilnehmern zu beschreiben. Je nach Quelle werden nicht nur kontextabhängig verschiedene Manöver als relevant eingestuft(vgl. Tabelle 5: Übersicht über Fahrmanöverzuordnungen nach verschiedenen LiteraturquellenTabelle 5), sondern diese weisen teilweise auch grundsätzlich unterschiedliche Eigenschaften auf. So werden beispielsweise bei manchen Manövern Relationen mit anderen Verkehrsteilnehmern mit einbezogen (z.B. „einem Fahrzeug folgen“), während dies bei anderen Manövern nicht der Fall ist (z.B. „dem Fahrstreifen folgen“). Auch können Manöver teilweise aus anderen Manövern zusammensetzbar sein (z.B. „überholen“ im Vergleich zu „Fahrstreifen wechseln“ und „vorbeifahren“).

Hier wird daher zunächst eine Vereinheitlichung und Definition von Manövern hinsichtlich ihrer Eigenschaften wie folgt vorgenommen:

Definition 6.1: *Ein Manöver ist die Abstraktion des Verhaltens eines Verkehrsteilnehmers zwischen zwei stationären Zuständen, Szenen oder Beziehungen. Je nach Abstraktionsgrad kann zwischen „atomaren Manövern", „Basismanövern" oder „Kompositionsmanövern" unterschieden werden.*

Atomare Manöver beschreiben nach dieser Definition grundsätzliche Aktionen zur Längs- und Querführung eines Fahrzeugs zwischen zwei stationären Zuständen des Fahrverhaltens. Dazu gehören Lenkradwinkelvorgaben oder Brems- und Gaspedalbetätigungen. Eine Manöverbeschreibung auf dieser Ebene besitzt einen vergleichsweise minimalen Abstraktionsgrad und hat das Ziel, das Fahrverhalten mit maximaler Genauigkeit darzustellen. Basismanöver werden als Grundeinheiten der Aktionen zur Längs- und Querführung des Fahrzeugs verstanden, die zu einem Übergang zwischen verschiedenen Szenen eines Szenarios im Kontext einer konkreten Anwendungsumgebung führen. Für einen potentiellen Autobahnpiloten sind dies beispielsweise in Querrichtung die Manöver

- „Fahrstreifen folgen",
- „Fahrstreifenwechsel links" und
- „Fahrstreifenwechsel rechts",

sowie in Längsrichtung die Manöver

- „Zielgeschwindigkeit halten",
- „Zielgeschwindigkeit erhöhen" und
- „Zielgeschwindigkeit verringern".

Kompositionsmanöver stellen Beschreibungen mit dem höchsten Abstraktionsgrad dar, sind aus verschiedenen Basismanövern eines oder auch mehrerer Verkehrsteilnehmer zusammensetzbar und können damit auch Beziehungen zwischen den Verkehrsteilnehmern auf der Verhaltensebene abbilden. Kompositionsmanöver sind beispielsweise „überholen", „folgen" oder „annähern". Auch interaktive Vorgänge zwischen Verkehrsteilnehmern wie „einscheren"

werden als eine Form der Kompositionsmanöver beschrieben. Beziehen sich die verwendeten Abstraktionen der Manöver ausschließlich auf ein von außen beobachtbares Verhalten der Verkehrsteilnehmer, umfassen die Kompositionsmanöver in beschreibender Weise auch wechselwirkende Vorgänge mit Verhandlungscharakter zwischen Verkehrsteilnehmern (beispielsweise das Einfädeln auf der Autobahn). Abfolgen und Auswirkungen möglicher Verhandlungssituationen zwischen den Teilnehmern können als weitere Charakteristika einer erweiterten Manöverbeschreibung aufgefasst werden, die hier nicht weiterverfolgt werden.

Nach dieser Manöverdefinition weisen die Basismanöver einen vergleichsweise mittleren Abstraktionsgrad auf (vgl. Abbildung 6.1).

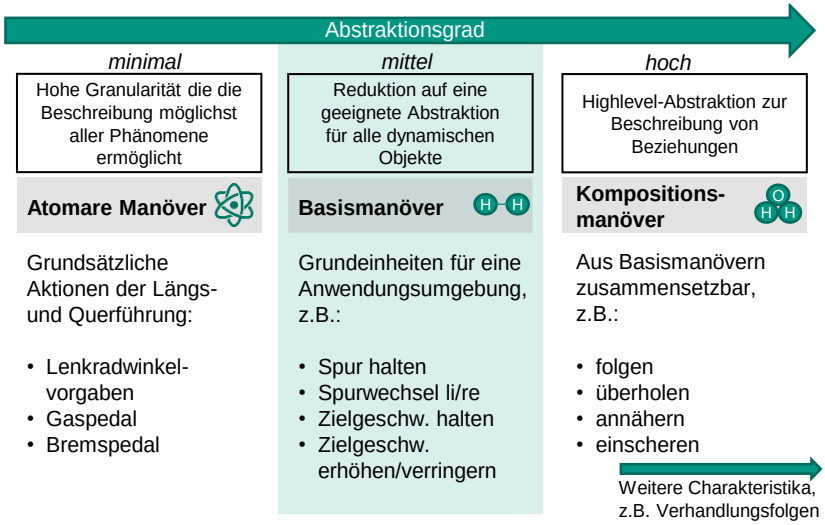


Abbildung 6.1: Erweiterung des Manöverbegriffs

Integriert in die Szenariomodellierung nach Steimle et al. [107] stellen sich die Manöverdefinition und –beziehungen wie folgt dar (Abbildung 6.2, Erweiterungen in grün):

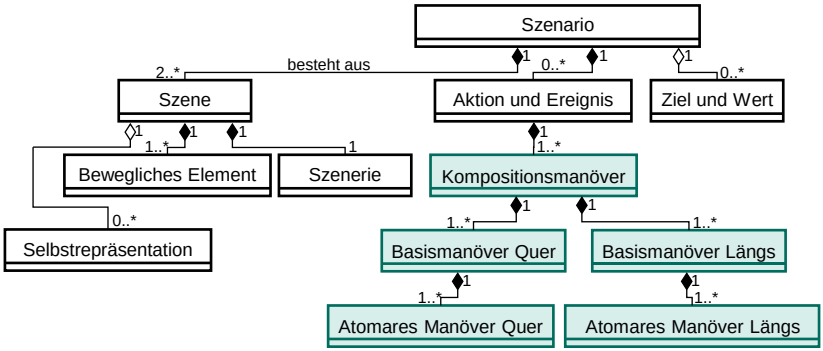


Abbildung 6.2: UML-Diagramm zum Szenario mit Manövern, erweitert nach [107]

6.1.2 Kinematische Modelle für die Longitudinaldynamik

Nachfolgend wird am Beispiel von Autobahnszenarien gezeigt, wie die Modellierung auf dem Abstraktionslevel der Basismanöver erfolgen kann.

Die Fahrzeugdynamik kann klassischerweise getrennt nach den drei translatorischen Bewegungsfreiheitsgraden (longitudinal, lateral und vertikal) betrachtet werden [146]. Auch wenn das Schwingungsverhalten des Aufbaus entlang der Hochachse (Vertikaldynamik) neben dem Einfluss auf den Fahrkomfort ebenfalls relevant für die Fahrsicherheit sein kann, wird die Vertikaldynamik hier nicht weiter berücksichtigt, da sie für die Fahrzeugführung im Sinne der Bahnplanung keine oder nur eine untergeordnete Bedeutung innehat. Darüber hinaus wird die Vertikaldynamik nur bedingt aktiv durch das Verhalten der Verkehrsteilnehmer beeinflusst, sondern ist das Resultat, abgesehen von etwaigen Regelsystemen, externer Anregungen aus der Fahrbahn.

Das Ziel des Szenario-Metamodells ist die Abbildung der in den Ausgangsdaten enthaltenen Manöver der Verkehrsteilnehmer auf einer Abstraktionsebene, die es erlaubt, die Szenarien aus den Metainformationen heraus in einer beliebigen X-in-the-Loop Simulation möglichst originalgetreu wiederherzustellen.

Für die Abbildung der Longitudinaldynamik existiert in der Literatur bereits eine Vielzahl von Modellen. Bham und Benekohal motivieren aus Felddaten der Ohio State University [147] ein dualphasiges Beschleunigungsmodell, das für niedrige Fahrzeuggeschwindigkeiten eine höhere Beschleunigungsrate annimmt [148]. Die Parametrierung wird empirisch aus den Felddaten abgeleitet. Die Fahrzeuggeschwindigkeit zu einem Zeitpunkt *t* ergibt sich nach diesem Modell zu:

$$v(t) = \begin{cases} v(t-1) + a_1 \cdot \Delta t, & falls\ 0 \leq v < 12{,}19\,\frac{m}{s^2} \\ v(t-1) + a_2 \cdot \Delta t, & falls\ v \geq 12{,}19\,\frac{m}{s^2} \end{cases} \tag{6.1}$$

mit *v(t-1)* als Geschwindigkeit zum Zeitpunkt *t-1* und *Δt* als Zeitdauer. Empirisch wurden für $a_1 = 1{,}1\ m/s^2$ und für $a_2 = 0{,}37\ m/s^2$ bestimmt. Mit den beiden Phasen wird dem Umstand Rechnung getragen, dass das maximale Beschleunigungsvermögen eines Fahrzeugs bei höheren Gängen in Abhängigkeit der Übersetzung konstruktiv bedingt eher abnimmt.

Auf dieser Prämisse setzen auch die linearen und quadratischen Regressionsmodelle von Wang et al. auf [149]. Darüber hinaus werden in diesem Modell Beschleunigungsvorgänge während der Geradeaus-Fahrt und auch bei Abbiegevorgängen unterschieden. Für die Beschleunigung ergibt sich nach [149]:

Für die Geradeaus-Fahrt:

$$a(t) = \begin{cases} 1{,}883 - 0{,}021 \cdot v(t), lineares\ Modell \\ [1{,}381 - 0{,}011 \cdot v(t)]^2, quadratisches\ Modell \end{cases} \tag{6.2}$$

Für das Abbiegen (links):

$$a(t) = \begin{cases} 1{,}646 - 0{,}017 \cdot v(t), lineares\ Modell \\ [1{,}289 - 0{,}009 \cdot v(t)]^2, quadratisches\ Modell \end{cases} \tag{6.3}$$

Deligianni et al. analysieren Fahrdaten hinsichtlich des Bremsverhaltens [150]. In Abhängigkeit der Fahrsituation schlagen sie drei verschiedene mathematische Ansätze zur Bestimmung des Bremswerts *d* vor:

- $d = a_1 \cdot t$, für sanftes Bremsen im normalen Verkehr

- $d^2 = 2 \cdot a_2 \cdot t$, starkes Bremsen zu Beginn aufgrund eines Hindernisses, gefolgt von sanfterem Bremsen

- $t^2 = 2 \cdot a_3 \cdot d$, starkes Bremsen erst am Ende aufgrund einer Fehleinschätzung durch den Fahrer

Darüber hinaus analysieren die Autoren Schwellenwerte für die Verzögerungsrate, um Bremsvorgänge vom „normalen" Fahren und nicht intendierten Bremsen zu unterscheiden. Sie stellen fest, dass Verzögerungsraten von 0,3 m/s² in den meisten Fällen durch Rauschen dem normalen Fahren zugeordnet werden müssen und klassifizieren damit Bremsevents als solche, wenn Verzögerungen größer oder gleich 0,5 m/s² erreicht werden [150].

Von Akcelik und Biggs wird ein polynomiales Modell vorgeschlagen, das sowohl für die Abbildung des Beschleunigungs- als auch des Verzögerungsverhaltens konzipiert ist [151]. Das Modell ist durch folgende allgemeine Form der polynomialen Beschleunigungsgleichung definiert:

$$a(t) = r \cdot a_m \cdot \theta^n \cdot (1 - \theta^m)^2, mit\ n > 0\ und\ m > -0{,}5n \qquad (6.4)$$

mit:

- *m, n:* freieinstellbare Kalibrierungsparameter (empirisch)

- *r:* Modellparameter in Abhängigkeit von *m, n*

- *a(t):* Beschleunigungsrate zum Zeitpunkt *t*

- a_m*:* maximale Beschleunigung

- θ*:* Zeitverhältnis t/t_a mit t_a als gesamte Beschleunigungszeit

Mit dem empirischen Parameterwert für n=1 und Integration der Gleichung (6.4) ergibt sich die Geschwindigkeitsgleichung [151]:

$$v(t) = v_i + t_a \cdot r \cdot a_m \cdot \theta^2 \cdot \left(0{,}5 - \frac{2 \cdot \theta^m}{m-2} + \frac{\theta^{2m}}{2m+2}\right) \qquad (6.5)$$

Im Vergleich mit anderen mathematischen Modellen wie zwei- oder dreiphasigen Sinusmodellen weist das polynomiale Modell eine verhältnismäßig hohe Güte und hohe Flexibilität für den Einsatzbereich auf.

Weitere Modelle berücksichtigen explizit für die Darstellung der Längsdynamik Fahrzeugfolgesituationen in verschiedenen Anwendungsumgebungen wie in Stadt- oder Autobahnsituationen und weitere Aspekte, die auf den umgebenden Verkehr zurückzuführen sind. So wird die Beschleunigung in unterschiedlicher Weise in Abhängigkeit beispielsweise eines Abstands zum Vorderfahrzeug oder der Geschwindigkeit desselben bestimmt. Auf Basis des Verhaltens des Vorderfahrzeugs bestimmt das Gipps-Modell eine „sichere Geschwindigkeit“ für die Longitudinaldynamik in Abhängigkeit des Bremswegs des Vorderfahrzeugs, eines Minimalabstands zu diesem und einer Reaktionszeit des Fahrers [152]. Das Intelligent-Driver-Model (IDM) versucht ein realistisches Beschleunigungsverhalten in allen Verkehrssituationen abzubilden und bezieht die Zeitlücke und einen Mindestabstand zum Vorderfahrzeug, ruckfreie (stetig differenzierbare) Beschleunigungsübergänge bei der Folgefahrt und dem Annähern zum Vorderfahrzeug und eine Bremsbeschleunigungsstrategie in Abhängigkeit der Kritikalität der Situation zum Vorderfahrzeug mit ein [152].

Auf der Betrachtungsebene des Einzelfahrzeugs kann, grundsätzlich unabhängig von der Fahrzeugfolgesituation, abgeleitet aus dem polynomialen Modell, und ohne Widerspruch zu den linearen oder quadratischen Modellen, ein Beschleunigungs- bzw. Verzögerungsvorgang durch folgende fünf physikalische Größen hinsichtlich der Randbedingungen im Weg/Zeit-Verlauf eines Szenarios beschrieben werden:[34]

- Zeitpunkt des Beginns eines neuen Longitudinalmanövers t_{long}
- Anfangsgeschwindigkeit v_s

[34] Die ausführliche Herleitung der nachfolgenden Abstraktionsmodelle für die Längs- und Lateraldynamik und des Tensormodells (siehe Abschnitt 6.1.5) wurden in der Masterthesis (Khalef, Mohamed Amine; 2019) dargestellt.

- Endgeschwindigkeit v_e

- Dauer (des Longitudinalmanövers) Δt_{long}

- Zurückgelegte Distanz Δx

Entlang des zeitlichen Verlaufs eines beliebigen Szenarios lässt sich damit die Folge der enthaltenen Manöver in Longitudinalrichtung eines Verkehrsobjekts in folgende Matrixform überführen:

$$M_{long} = \begin{pmatrix} \overrightarrow{t_{long}} \\ \overrightarrow{v_s} \\ \overrightarrow{v_e} \\ \overrightarrow{\Delta t_{long}} \\ \overrightarrow{\Delta x} \\ \overrightarrow{L_{long}} \end{pmatrix} = \begin{pmatrix} t_{long,1} & t_{long,2} & \cdots & t_{long,n} \\ v_{s,1} & v_{s,2} & \cdots & v_{s,n} \\ v_{e,1} & v_{e,2} & \cdots & v_{e,n} \\ \Delta t_{long,1} & \Delta t_{long,2} & \cdots & \Delta t_{long,n} \\ \Delta x_1 & \Delta x_2 & \cdots & \Delta x_n \\ L_{long,1} & L_{long,2} & \cdots & L_{long,n} \end{pmatrix} \quad (6.6)$$

Während der Vektor $\overrightarrow{t_{long}}$ die Zeitpunkte der Manöverwechsel in Longitudinalrichtung beschreibt, werden diesen Zeitpunkten die entsprechenden Größen zugewiesen und das entsprechend klassifizierte Basismanöver unter dem Labelvektor $\overrightarrow{L_{long}}$ erfasst. Das Label kann folgende Ausprägungen annehmen und wird wie folgt codiert:

- a > 0,5m/s²: Basismanöver „Zielgeschwindigkeit erhöhen“, Code 1

- a < -0,5m/s²: Basismanöver „Zielgeschwindigkeit verringern“, Code -1

- |a| ≤ 0,5m/s²: Basismanöver „Zielgeschwindigkeit halten“, Code 0

Die Festlegung auf den Schwellenwert von 0,5m/s² als Unterscheidungsmerkmal wurde dabei aus den Ergebnissen der Untersuchungen nach Deligianni übernommen [150].

6.1.3 Kinematische Modelle für die Lateraldynamik

Die Lateraldynamik für das hier exemplarisch betrachtete System eines Autobahnpiloten lässt sich, wie in Abschnitt 6.1.1 dargelegt, in die Basismanöver „Fahrstreifen folgen“, „Fahrstreifenwechsel links“ sowie „Fahrstreifenwechsel rechts“ zerlegen.

Nach DIN 75204 wird ein Fahrstreifenwechsel wie folgt definiert [153]:

Definition 6.2: *Durch gewolltes Lenken eingeleitetes Querversetzen des Fahrzeugs. Beim Überholvorgang werden sowohl Ausscher- als auch Einschervorgang jeweils als Fahrstreifenwechsel bezeichnet.*

Bereits in den 60er Jahren wurden Modelle zur Beschreibung von Fahrstreifenwechseln im Rahmen der Unfallrekonstruktion entwickelt. Spindler beschreibt Fahrstreifenwechsel 1963 in einer empirischen Analyse durch Wendeklothoiden, während Runkel 1969 zwei zueinander entgegengesetzte Kreisbögen zur Modellierung verwendet [154].

Sledge et al. vergleichen verschiedene Kreisbogen-, Polynomial-, Sinus und Bezierkurvenmodelle für die Trajektorienbeschreibung von Fahrstreifenwechseln nach unterschiedlichen Kriterien [155]. Durchweg gute Ergebnisse liefert das Polynomial-Modell 5. Grades nach Nelson [156]:

$$y(x) = y_e \left[10 \left(\frac{x}{x_e}\right)^3 - 15 \left(\frac{x}{x_e}\right)^4 + 6 \left(\frac{x}{x_e}\right)^5 \right] \tag{6.7}$$

zur Bestimmung des Querversatzes y in Abhängigkeit des Weges in Längsrichtung x mit y_e als lateraler Endposition der Querabweichung, x_e als entsprechender longitudinaler Endposition des Fahrstreifenwechsels.

Als weiterer Einflussfaktor kann die Querbeschleunigung betrachtet werden. Diese wird beispielsweise im Modell nach Chovan et al. mit einbezogen und geht nach Integration als Quergeschwindigkeit mit ein. Als Beschreibung des Querversatzes in Abhängigkeit der Zeit ergibt sich [157]:

$$y(t) = \frac{y_e}{T_{sw}} t - \frac{y_e}{2\pi} \cdot \sin\left(\frac{2\pi}{T_{sw}} t\right) + v_{y,0} \cdot t + y_0 \tag{6.8}$$

mit T_{sw} als Gesamtdauer des Fahrstreifenwechsels und $v_{y,0}$ und y_0 als anfängliche Quergeschwindigkeit beziehungsweise anfängliche seitliche Position.

Gegensätzlich dazu kommt nach Analyse realer Fahrdaten Freyer zu dem Schluss, dass keine relevante Abhängigkeit von der Querbeschleunigung beim Fahrstreifenwechsel auch bei unterschiedlichen Geschwindigkeiten oder Fahrsituationen gegeben ist. Als durchschnittliche Approximation der Messdaten bestimmt er die Parameter *n=0,0504* und *z=6,88* für folgende Cosinusfunktion [158]:

$$y(t) = -\frac{1}{2}\left(\cos\left(\frac{\pi}{\sqrt{n \cdot T_{sw} + z}} t\right) - 1\right) \tag{6.9}$$

Analog zur Bestimmung der Längsdynamik werden hier auf Basis der vorgestellten Modelle folgende physikalische Größen als Randbedingungen im Weg/Zeit-Verlauf eines beliebigen Szenarios für die Lateraldynamik als maßgeblich angenommen. Die Querbeschleunigung wurde nicht berücksichtigt:

- Zeitpunkt des Beginns eines neuen Lateralmanövers t_{lat}
- Lateralposition zu Beginn des Manövers y_s
- Lateralposition am Ende des Manövers y_e
- Dauer (des Lateralmanövers) Δt_{lat}

Die Überführung in die Matrixform erfolgt wie hier dargestellt:

$$M_{lat} = \begin{pmatrix} \overrightarrow{t_{lat}} \\ \overrightarrow{y_s} \\ \overrightarrow{y_e} \\ \overrightarrow{\Delta t_{lat}} \\ \overrightarrow{L_{lat}} \end{pmatrix} = \begin{pmatrix} t_{lat,1} & t_{lat,2} & \cdots & t_{lat,n} \\ y_{s,1} & y_{s,2} & \cdots & y_{s,n} \\ y_{e,1} & y_{e,2} & \cdots & y_{e,n} \\ \Delta t_{lat,1} & \Delta t_{lat,2} & \cdots & \Delta t_{lat,n} \\ L_{lat,1} & L_{lat,2} & \cdots & L_{lat,n} \end{pmatrix} \tag{6.10}$$

Das entsprechend klassifizierte laterale Basismanöver wird unter dem Labelvektor $\overrightarrow{L_{lat}}$ erfasst. Die Codierung -1 bezeichnet einen „Fahrstreifenwechsel links“, während 0 beziehungsweise 1 für „Fahrstreifen folgen“ respektive „Fahrstreifenwechsel rechts“ stehen.

Die Klassifikation der lateralen Basismanöver ist im Vergleich zu den longitudinalen Basismanövern aufwändiger. Während für die Longitudinaldynamik mit der Längsbeschleunigung ein eindeutiges Signal zur Verfügung steht, ist Vergleichbares für die Lateraldynamik nicht gegeben. Zwar kann der Moment des Übergangs von einem Fahrstreifen zum anderen beispielsweise durch den Zeitpunkt des Kreuzens des Fahrzeugmittelpunkts über die Fahrstreifentrennlinie eindeutig bestimmt werden, allerdings ermöglicht dies keine Aussage über den Beginn (oder Ende) des Basismanövers „Fahrstreifenwechsel“. Hier wird daher zur eindeutigen Bestimmung des Beginns und Endes eines Fahrstreifenwechsels dieser wie folgt definiert:

Definition 6.3: *Ein Fahrstreifenwechsel ist durch die vollständige Überschreitung der Fahrstreifenbegrenzung durch den Fahrzeugmittelpunkt gekennzeichnet. Der Beginn des Fahrstreifenwechsels wird ab der ersten Überschreitung eines geeigneten Schwellenwertes für die Ableitung des lateralen Querversatzes des Fahrzeugs bestimmt. Analog dazu wird für das Ende des Fahrstreifenwechsels die erstmalige Unterschreitung des Schwellenwertes nach dem Kreuzen der Fahrstreifenbegrenzung angenommen.*

Folgendes Schema zeigt den in der Definition beschriebenen Zusammenhang (Abbildung 6.3).

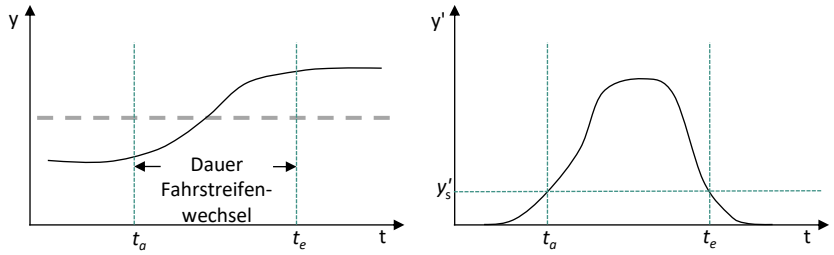


Abbildung 6.3: Schema zur Bestimmung des Anfangzeitpunktes t_a und Endzeitpunkts t_e des Basismanövers " Fahrstreifenwechsel". Aus der lateralen Querabweichung entlang des Fahrstreifens (links) folgt die Bestimmung eines geeigneten Schwellenwertes y'_s aus der Ableitung der Querabweichung (rechts).

Die Bestimmung des Schwellenwerts von 0,05 m/s erfolgte hier aus der empirischen Analyse einer Simulationsreihe.

Im Gegensatz zur Longitudinaldynamik ist die Klassifikation der Basismanöver für die Lateraldynamik damit zwar ebenfalls zur Laufzeit der Aufzeichnung der Signale möglich, der Beginn eines Fahrstreifenwechsels kann allerdings erst als solcher bestimmt werden kann, wenn dieser tatsächlich auch erfolgt ist.

6.1.4 Abstraktion der Zustände als Szenenrepräsentation

Entsprechend der in Kapitel 3.1.4 vorgestellten Elemente eines Szenarios, kann dieses in eine Sequenz von Szenen und Situationen zerlegt werden, die durch dazwischenliegende Aktionen verbunden werden. Bezogen auf die dynamischen Elemente des Szenarios, stellen die in den vorherigen Abschnitten hergeleiteten Basismanöver den Aktionsraum des hier entwickelten Szenario-Metamodells dar. Um die Szenariodarstellung für die dynamischen Elemente zu vervollständigen, muss darüber hinaus ein entsprechendes Abbild für die Szenenrepräsentation dieser Elemente erfolgen.

Im Rahmen der Manöverplanung für ADAS existieren bereits Ansätze zur Beschreibung von Verkehrsszenen in Form semantischer Modelle. Kohlhaas et al. präsentieren beispielsweise ein Konzept, um Relationen zwischen dem Ego-Fahrzeug und anderen Verkehrsteilnehmern sowie der Umwelt strukturiert abbilden zu können (siehe Abbildung 6.4). Das Konzept versucht dabei auch die Dynamik der Zusammenhänge zu berücksichtigen und formalisiert Regeln für die Übergänge der dynamischen Objekte zwischen den verschiedenen Zuständen, die zu neuen möglichen Konfigurationen führen können oder nichtzulässige Konfigurationen ausschließen [159].

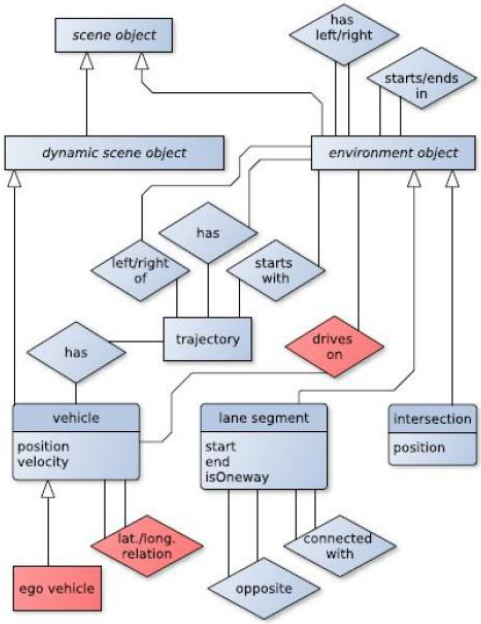


Abbildung 6.4: Ontologieschema für die Generierung des Zustandsraums des Ego-Fahrzeugs [159]

Petrich et al. erweitern dieses Konzept und entwerfen eine kompakte semantische Tensordarstellung zur paarweisen Beschreibung der Beziehungen aller Verkehrsteilnehmer eines Szenarios (Abbildung 6.5). Darüber hinaus bietet es

die Möglichkeit, Unsicherheiten (beispielsweise Sensordaten bedingt) durch die Integration einer Likelihood-Funktion aufzunehmen [160].

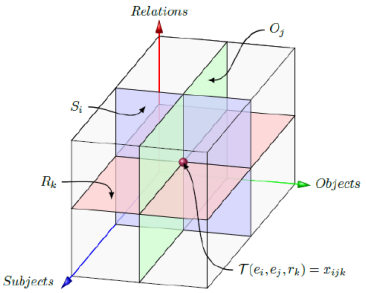


Abbildung 6.5: Prinzip der Struktur des semantischen Tensors zur Darstellung der Relationen zwischen den Szenarioentitäten [160]

Abgeleitet aus diesen Ansätzen wird in dieser Arbeit ebenfalls ein Modell für die statische Zustands- und Szenenrepräsentation des Ego-Fahrzeugs (betrachtetes System) im Verhältnis zu den umgebenden Verkehrsobjekten verwendet. Die räumlichen Relationen ergeben sich in einer möglichen Diskretisierung für Szenarien auf der Autobahn zu jeweils drei Zustandsmöglichkeiten in longitudinaler und lateraler Richtung:

Longitudinal:

- Das Objekt befindet sich vor dem Ego-Fahrzeug, Code 1
- Das Objekt befindet sich in Längsrichtung auf der Höhe des Ego-Fahrzeugs. Hier wird einem Objekt, das einen relativen Längsabstand zum Egofahrzeug <|2,5m| aufweist, dieser Zustand zugewiesen, Code 0.
- Das Objekt befindet sich hinter dem Ego-Fahrzeug, Code -1

Lateral[35]:

- Das Objekt befindet sich auf dem ersten benachbarten Fahrstreifen rechts vom Ego-Fahrzeug, Code 1
- Das Objekt befindet sich auf dem gleichen Fahrstreifen wie das Ego-Fahrzeug, Code 0
- Das Objekt befindet sich auf dem ersten benachbarten Fahrstreifen links vom Ego-Fahrzeug, Code -1

Aus der Kombination dieser longitudinalen und lateralen räumlichen Relationszustände ergibt sich die Menge aller möglichen Zustände nach diesem Modell zu der Menge aller 2-Tupel in Codierungsschreibweise (vergleiche auch schematisch Abbildung 6.6):

$$Q = \{(1,1),(1,0),(1,-1),(0,1),(0,0),(0,-1),(-1,1),(-1,0),(-1,-1)\}$$

[35] Hier werden zur Vereinfachung der Darstellung nur die jeweils benachbarten Fahrspuren berücksichtigt. Das Modell lässt sich aber durch einfache Anpassung der Codierung auf Umgebungen auf eine beliebige Anzahl an Fahrspuren erweitern.

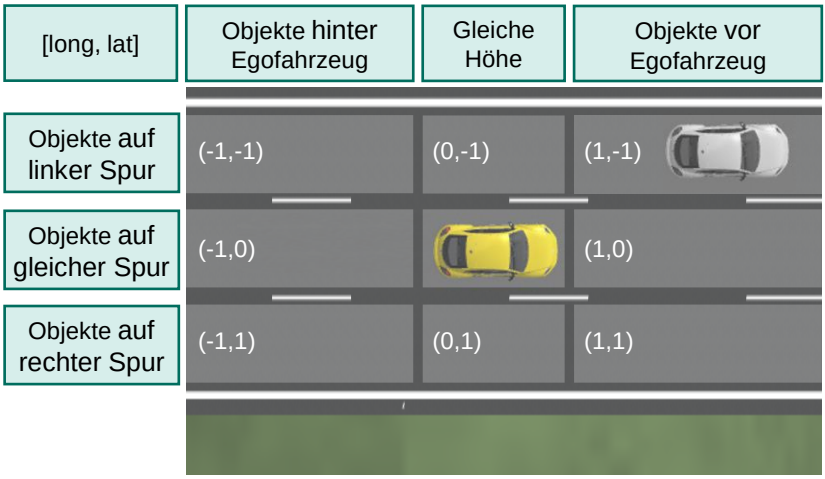


Abbildung 6.6: Schema des Zustandsmodells und Codierung der räumlichen Zustände

Die Übergänge und die Übergangsfunktion $\delta: Q \times E \rightarrow Q$, die einen beliebigen Zustand $z_0 \in Q$ in einen neuen Zustand $z_1 \in Q$ überführt, werden bestimmt von der Menge aller Eingabewerte *E*, die in diesem Fall aus den im vorigen Abschnitt eingeführten Basismanövern besteht.

Beispielsweise kann der zeitliche Übergang des Zustands Q_1 = (1,-1) (dargestellter Zustand in Abbildung 6.6) in den Zustand Q_4 = (1,0) durch verschiedene Basismanöverkombinationen herbeigeführt werden. So kann beispielsweise das Ego-Fahrzeug einen Fahrstreifenwechsel nach links (abgekürzt LCL) durchführen, während das Verkehrsobjekt seinem Fahrstreifen folgt (LK) oder umgekehrt das Verkehrsobjekt einen Fahrstreifenwechsel nach rechts (LCR) durchführen, während das Ego-Fahrzeug seinem Fahrstreifen folgt (LK). Zusätzlich kann in Längsrichtung das Ego-Fahrzeug, ebenso wie vice versa das Verkehrsobjekt, die Geschwindigkeit verringern, konstant fahren oder (bis zu einem gewissen Grade) beschleunigen. Für den Übergang *T* dieses Beispiels gilt also formal:

$$T_{Q_1 \rightarrow Q_4} = \{(LCL_{Ego}, LK_{Obj,}), \{(LK_{Ego}, LCR_{Obj,})\}$$

Daraus folgt indirekt, dass für eine Auflösung der Mehrdeutigkeit des Übergangs zusätzlich zu den erfassten räumlichen Relationen für die Szenenrepräsentation, das Erfassen der Basismanöver ein wesentlicher Bestandteil für die Modellierung darstellt. Darüber hinaus kann die Formalisierung für die (interne) Verifikation des Modells eingesetzt werden (siehe hierzu Abschnitt 6.4.1).

6.1.5 Zusammenführung in einer Tensordarstellung

Um die einzelnen Elemente des in den vorherigen Abschnitten entwickelten Szenario-Metamodells im Sinne des in dieser Arbeit vorgestellten Prozesses nutzen zu können, müssen sie in einer einheitlichen Darstellung verankert werden.

Die Abstraktionen der Longitudinal- und Lateraldynamik lassen sich mit der Darstellung der Basismanöver in die jeweiligen Matrixformen überführen (vgl. Abschnitte 6.1.2 und 6.1.3). Verknüpft man diese Matrixdarstellungen für die Kinematik mit der in Abschnitt 6.1.4 dargestellten Zustandsrepräsentation, lässt sich die Szenarioabstraktion für ein beliebiges dynamisches Objekt Obj_i in folgender Objektmatrix abbilden:

$$M_{Obj_i} = \begin{pmatrix} \overrightarrow{t_i} \\ \overrightarrow{v_s} \\ \overrightarrow{v_e} \\ \overrightarrow{\Delta t} \\ \overrightarrow{\Delta x} \\ \overrightarrow{y_s} \\ \overrightarrow{y_e} \\ \overrightarrow{ds_x} \\ \overrightarrow{ds_y} \\ \overrightarrow{L_{long}} \\ \overrightarrow{L_{lat}} \\ \overrightarrow{L_{st,long}} \\ \overrightarrow{L_{st,lat}} \end{pmatrix} = \begin{pmatrix} t_{i,1} & t_{i,2} & \cdots & t_{i,n} \\ v_{s,1} & v_{s,2} & \cdots & v_{s,n} \\ v_{e,1} & v_{e,2} & \cdots & v_{e,n} \\ \Delta t_1 & \Delta t_2 & \cdots & \Delta t_n \\ \Delta x_1 & \Delta x_2 & \cdots & \Delta x_n \\ y_{s,1} & y_{s,2} & \cdots & y_{s,n} \\ y_{e,1} & y_{e,2} & \cdots & y_{e,n} \\ ds_{x,1} & ds_{x,2} & \cdots & ds_{x,n} \\ ds_{y,1} & ds_{y,2} & \cdots & ds_{y,n} \\ L_{long,1} & L_{long,2} & \cdots & L_{long,n} \\ L_{lat,1} & L_{lat,2} & \cdots & L_{lat,n} \\ L_{st,long,1} & L_{st,long,2} & \cdots & L_{st,long,n} \\ L_{st,lat,1} & L_{st,lat,2} & \cdots & L_{st,lat,n} \end{pmatrix} \quad (6.11)$$

In der Objektmatrix werden im Vergleich zur Kinematikabstraktion die Vektoren $\overrightarrow{L_{state,long}}$ und $\overrightarrow{L_{state,lat}}$ neu eingeführt, welche die zuvor vorgestellten Codierungen der räumlichen Zustände enthalten. Des Weiteren werden die Vektoren $\overrightarrow{ds_x}$ und $\overrightarrow{ds_y}$ ergänzt, die die relativen Abstände des Objekts in Längs- und Querrichtung zum Ego-Fahrzeug beschreiben. Dies ist notwendig, um die räumliche Relation nicht nur qualitativ (nach dem zuvor vorgestellten Zustandsmodell), sondern auch quantitativ zu beschreiben.

Entsprechend der dynamischen Objekte wird die Abstraktionsmatrix für das Ego-Fahrzeug definiert. Sie enthält analog zu den Matrizen für die dynamischen Objekte die vollständige Abbildung für die Kinematik, besitzt aber keine Einträge für die räumlichen Relationen, da diese in Bezug auf das Ego-Fahrzeug bereits in den jeweiligen Matrizen der Objekte abgebildet sind. Damit ergibt sich die Abstraktionsmatrix für das Ego-Fahrzeug wie folgt:

$$M_{Ego} = \begin{pmatrix} \overrightarrow{t_{ego}} \\ \overrightarrow{v_s} \\ \overrightarrow{v_e} \\ \overrightarrow{\Delta t} \\ \overrightarrow{\Delta x} \\ \overrightarrow{y_s} \\ \overrightarrow{y_e} \\ \overrightarrow{L_{long}} \\ \overrightarrow{L_{lat}} \end{pmatrix} = \begin{pmatrix} t_{Ego,1} & t_{Ego,2} & \cdots & t_{Ego,n} \\ v_{s,1} & v_{s,2} & \cdots & v_{s,n} \\ v_{e,1} & v_{e,2} & \cdots & v_{e,n} \\ \Delta t_1 & \Delta t_2 & \cdots & \Delta t_n \\ \Delta x_1 & \Delta x_2 & \cdots & \Delta x_n \\ y_{s,1} & y_{s,2} & \cdots & y_{s,n} \\ y_{e,1} & y_{e,2} & \cdots & y_{e,n} \\ L_{long,1} & L_{long,2} & \cdots & L_{long,n} \\ L_{lat,1} & L_{lat,2} & \cdots & L_{lat,n} \end{pmatrix} \quad (6.12)$$

Ordnet man die Abstraktionsmatrizen der Objekte und des Ego-Fahrzeugs anhand der Dimensionen *Zeit*, *Objekte* und *Metainformation* an, erhält man einen dreidimensionalen Tensor (genauer: ein Tensor mit Rang 3). Entlang des *Zeitvektors* werden alle Zeitpunkte erfasst, bei denen das Ego-Fahrzeug eine Manöveränderung durchführt, eines der erfassten dynamischen Objekte eine Manöveränderung durchführt oder sich die räumliche Relation eines der Objekte zum Ego-Fahrzeug verändert. Für jeden dieser Zeitpunkte wird die jeweils relevante *Metainformation* für das betreffende Objekt oder Ego-Fahrzeug gemäß den vorgestellten Abstraktionsmatrizen gespeichert. In der dritten Dimension werden die *Objekte* angeordnet, wobei das erste Objekt dem Ego-Fahrzeug entspricht. Das Schema des Tensormodells ist nachfolgend dargestellt (Abbildung 6.7).

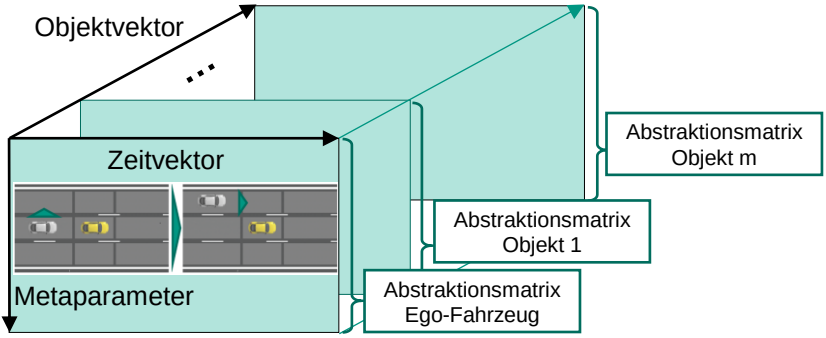


Abbildung 6.7: Schema des Tensormodells

6.1.6 Erweiterungsmöglichkeiten des Tensormodells

Der Fokus in dieser Arbeit liegt auf der Modellierung der dynamischen Komponenten eines Szenarios. Dennoch gehören zu einer vollständigen Szenariobeschreibung auch statische Elemente, wie der Fahrbahnverlauf, Bebauungen oder Schilder[36]. Aus diesem Grund muss das Tensormodell dahingehend erweiterbar sein, dass auch diese Komponenten bei Bedarf mitberücksichtigt werden können. Ein möglicher Ansatz wird nachfolgend skizziert.

Eine Eigenschaft des Tensormodells nach dem bisher vorgestellten Vorgehen ist die *zeitbasierte* Erfassung der Ereignisse (Manöver und Zustände der dynamischen Objekte des Szenarios). Dies ist insbesondere bedingt durch die Tatsache, dass das Modell vor allem nach dem hier verfolgten Konzept für die Anwendung von aufgezeichneten Daten vorgesehen ist.

Für die Erfassung der statischen Elemente oder Szenarioeigenschaften ist hingegen ein *ortsbasierter* Ansatz zu bevorzugen. Während statische Eigenschaften zu den dynamischen Objekten (beispielsweise die Objektklasse wie PKW, LKW etc., oder auch die erfassten Dimensionen der dynamischen Objekte) noch durch eine einfache Erweiterung der Metainformationsdimension in jeder Objektmatrix, und damit weiterhin zeitbasiert, gespeichert werden können, ist dies für andere statische Elemente nicht sinnvoll. Sollen beispielsweise Verkehrsschildinformationen in der Repräsentation mit abgebildet werden, ist nicht der Zeitpunkt von Bedeutung, sondern der Ort, an dem das entsprechende Schild erkannt wurde.

Ortsbasierte Komponenten lassen sich in das bislang vorgestellte Tensormodell durch das Hinzufügen einer weiteren Dimension ergänzen. Dieser *Ortsvektor* muss mit den bisherigen Dimensionen verknüpft werden. Das Ego-Fahrzeug stellt dabei die Schlüsselkomponente dar. Geht man davon aus, dass bei der Aufzeichnung der Daten auch die globale Position des Ego-Fahrzeugs,

[36] Anzumerken ist, dass Schilder und andere Elemente auch dynamische Eigenschaften aufweisen können (z.B. Wechselzeichenanlagen). Da aber zum Zeitpunkt der jeweiligen Erfassung nur genau ein Informationszustand vorliegt, werden diese Elemente in der Modellierung als statisch betrachtet.

beispielsweise via GPS, erfasst werden kann, kann auch die Position der statischen Elemente bestimmt werden. Die statischen Elemente können in einer weiteren Matrix M_{stat} gespeichert werden, die entlang des Ortsvektors $\vec{s}$ dargestellt ist:

$$M_{stat} = \begin{pmatrix} \vec{s} \\ \overrightarrow{t_{ego}} \\ \overrightarrow{Koord_x} \\ \overrightarrow{Koord_y} \\ \overrightarrow{Typ} \\ \overrightarrow{Wert} \\ \vdots \end{pmatrix} = \begin{pmatrix} s_1 & s_2 & \dots & s_m \\ t_{ego,1} & t_{ego,2} & \dots & t_{ego,m} \\ Koord_{x,1} & Koord_{x,2} & \dots & Koord_{x,m} \\ Koord_{y,1} & Koord_{y,2} & \dots & Koord_{y,m} \\ Typ_1 & Typ_2 & \dots & Typ_m \\ Wert_1 & Wert_2 & \dots & Wert_m \\ \vdots & \vdots & \dots & \vdots \end{pmatrix} \quad (6.13)$$

$\vec{s}$ entspricht dem Weg entlang der gefahrenen Route. Für jedes statische Element können beim Passieren desselben sowohl die globalen Koordinaten, als auch der entsprechende Zeitstempel aufgezeichnet werden. Über diesen Ansatz erfolgt die Zuordnung der wegbasierten Dimension zur bislang verwendeten zeitbasierten Dimension des Tensors. Wird der Tensor einer Aufzeichnung entlang der Zeitdimension (oder der Wegdimension) geteilt, um die enthaltenen Szenarien zu extrahieren (vergleiche auch Abschnitt 6.3.2), können über diese eindeutige Weg-Zeit Beziehung alle relevanten Elemente für das betreffende Szenario bestimmt werden.

Schlussendlich können in der Matrixform, analog zu den dynamischen Objekten, alle beliebig erforderlichen (Meta-)Informationen zu den statischen Elementen gespeichert werden. Im hier dargestellten Beispiel könnte dies für eine Verkehrsschilderfassung beispielsweise der Typ des Schilds (Geschwindigkeitsbegrenzung, Überholverbot etc.) und ein jeweiliger Wert (z.B. 100 km/h für die Geschwindigkeitsbegrenzung) sein. Je nach zu erfassendem Element können beliebig weitere Informationsvektoren eingeführt werden, die weitere Eigenschaften beschreiben.

Schematisch zeigt Abbildung 6.8 die Erweiterung des Tensormodells um statische Komponenten mit einem ortsbasierten Bezug.

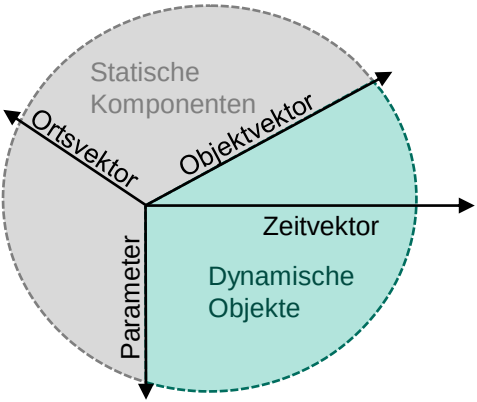
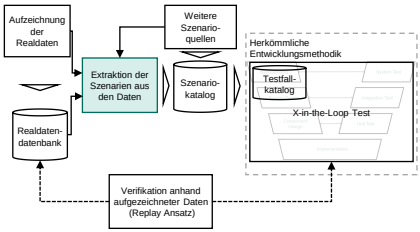


Abbildung 6.8: Schema zur Erweiterung des Tensormodells um statische Komponenten

6.2 Verfahren für die Szenarioextraktion



Ein wesentlicher Bestandteil des Prozesses ist die Extraktion der Szenarien aus den (aufgezeichneten) Daten und damit die Überführung in die Form des im vorherigen Abschnittes dargestellten Meta-Modells.

Dem Modell entsprechend, wird damit der Schritt der Szenarioextraktion auf folgende Elemente heruntergebrochen werden:

1. Bestimmung der Manöver des Ego-Fahrzeugs
2. Bestimmung der Manöver aller dynamischen Objekte (im Sensorsichtbereich) und
3. Bestimmung der Zustände als Objekt-Ego-Fahrzeug-Relationen ebenfalls für jedes sichtbare Objekt

6.2.1 Manöverextraktion für Ego-Fahrzeug und Objekte

Um die Manöver für das Ego-Fahrzeug in longitudinaler und lateraler Richtung extrahieren zu können, müssen diese zunächst in den Daten korrekt klassifiziert werden. Insbesondere für die Manöverklassifikation für das Ego-Fahrzeug existieren bereits unterschiedliche Ansätze, die auf verschiedene Datengrundlagen (Sensoren) zurückgreifen. In mehreren Arbeiten werden hierfür Beschleunigungs-, Gyroskop-, GPS- und Magnetometer-Daten verwendet, die teilweise (kostengünstig) aus einem Smartphone ausgelesen werden [161] [162] [163] [164] [165]. Weitere Arbeiten verwenden, unter direktem Zugriff via Fahrzeug-CAN, Daten wie Lenkwinkel, Fahrzeuggeschwindigkeit, laterale Geschwindigkeit, Giergeschwindigkeit oder die Gas- beziehungsweise Bremspedalstellung [166] [167] [168] [169]. Des Weiteren können die Manöver aus Daten einer Fahrsimulatorfahrt [170] [171] oder realen Testfahrten [172] gewonnen werden. Hier werden Daten des lateralen Abstands zwischen Fahrzeug und Fahrbahnmarkierung, Spurabfahrtsbetrag, Beschleunigung, Lenkwinkel, -geschwindigkeit und Moment, sowie Position, Ausrichtung und Geschwindigkeit des Ego-Fahrzeugs sowie benachbarter Objekte verwendet. In diesen Arbeiten wird nicht näher dargestellt, wie die Daten ausgelesen wurden.

Für die Auswertung kommen verschiedene Klassifikatoren zum Einsatz. So werden die Methoden

- Support Vector Machine (SVM) [170] [162] [163] [164] [171] [168],
- Random Forest (RF) [161] [162] [171],
- k-Nearest-Neighbor (kNN) [166] [164] [171],
- Hidden Markov Model (HMM) [169] [166],
- Fuzzy Rule-Based Classifier (FRC) [165] [162],
- Bayesian Inference Model [170],

- Convolutional Neural Network (CNN) [172],
- Decision Tree [167] und
- Naive Bayes [164]

angewandt.

Diese Arbeiten konzentrieren sich dabei auf die Klassifikation unterschiedlicher Fahrmanöver wie beispielsweise Fahrstreifenwechsel, Abbiegevorgänge, Kurvenfahrt, Beschleunigen, Abbremsen, Ein-/Ausfahrt in einen Kreisverkehr. Gruner et al. betrachten darüber hinaus auch Manöver, die durch andere Objekte bedingt werden wie beispielsweise Überholmanöver oder Querverkehr vor dem Ego-Fahrzeug [172].

Den zuvor genannten Ansätzen ist gemein, dass sie zwar gute Ergebnisse in der Klassifikation erzielen, aber jeweils auf Basis der dabei konkreten, definierten Datengrundlagen. Ein Übertrag auf potentiell andere zur Verfügung stehende Signale oder Sensoren ist nur bedingt mit diesen Verfahren möglich beziehungsweise wird nicht dargestellt.

In dieser Arbeit werden als Datengrundlage Objektlistensignale verwendet, wie sie typischerweise von Sensoren wie Radar, Kamera, Lidar nach der Rohdatenverarbeitung sowie deren Fusion beispielsweise über das Fahrzeugbussystem zur Verfügung stehen. Eine Herausforderung dabei ist, dass auf dieser Ebene verschiedene Konzepte je nach Hersteller oder Systemzulieferer existieren, sodass nicht von einem standardisierten Setup hinsichtlich der verfügbaren Signale ausgegangen werden kann. Aus diesem Grund werden hier Konzepte verwendet, die sich möglichst einfach adaptieren lassen, je nachdem welche Signale für eine Auswertung überhaupt zur Verfügung stehen.

Objektlisten sind Signale beziehungsweise Daten, die in Form einer Zeitreihe vorliegen. Angelehnt an die Definitionen aus der Statistik kann hier die Zeitreihe wie folgt bezeichnet werden [173]:

Definition 6.4: *Eine Zeitreihe ist eine zeitlich geordnete Folge von numerischen Merkmalswerten, die im Zeitablauf zu bestimmten Zeitpunkten über einen Beobachtungszeitraum erfasst wurden.*

Für die Auswertung der Zeitreihen in Form einer Zeitreihenanalyse kommen grundsätzlich verschiedene Verfahren in Frage. Für die Klassifikation von Zeitreihen unterscheiden Bagnall et al. dabei sechs Kategorien von Methoden [144]:

- Klassifikation der vollständigen Zeitreihe
- Intervallbasierte Klassifikation
- Formbasierte Klassifikation
- Wörterbuchbasierte Klassifikation
- Kombinierte Klassifikation
- Modellbasierte Klassifikation

Für jede dieser Kategorien existiert eine Vielzahl von Algorithmen oder Algorithmuskombinationen. Bei der formbasierten und wörterbuchbasierten Klassifikation liegt der Fokus eher darauf, einzelne oder mehrere markante Teilsequenzen („Shapelets“) in einer Zeitreihe wiederzufinden. Ein bekannter Vertreter hierfür ist beispielsweise die Symbolic Aggregate Approximation (SAX), die symbolische Repräsentationen in Form von Buchstaben/Strings eines definierten Alphabets für die Transformation der Abschnitte einer Zeitreihe verwendet [174]. Weitere Vertreter sind die teilweise darauf aufsetzenden wörterbuchbasierten Klassifikatoren wie Bag of Patterns, der wiederum aus den identifizierten Strings Wörter bildet und auf Ähnlichkeiten zwischen mehreren Zeitreihen oder Ähnlichkeiten innerhalb einer Zeitreihe vergleicht [175].

Diese Verfahren zielen typischerweise nicht darauf ab, die Zeitreihe vollständig zu klassifizieren, sodass diese hier nicht weiter untersucht und berücksichtigt werden.

Ein vielseitiger Vertreter für die erstgenannten Kategorien ist nach Bagnall et al. hingegen das Dynamic Time Warping (DTW) mit seinen Abwandlungen Weighted Dynamic Time Warping (WDTW) und Derivative Dynamic Time Warping (DDTW), die auch in dieser Arbeit angewandt wurden (vgl. Abschnitt 7.3.1) [144].

Grundsätzlich sind der DTW-Algorithmus und seine Varianten Methoden zur Bestimmung der Distanz zwischen zwei Zeitreihen und gehen ursprünglich auf eine Methode in der Spracherkennung von Sakoe und Chiba aus dem Jahr 1978 zurück [176]. Im Gegensatz zu anderen Verfahren verwendet der DTW jedoch kein euklidisches Distanzmaß, bei dem der Abstand zweier Punkte der Zeitreihen paarweise verglichen wird (Abbildung 6.9 links oben), sondern ein elastisches Distanzmaß (Abbildung 6.9 links unten). Damit ist es möglich, bei der Berechnung des Abstands zweier Zeitreihen auch Streckungen oder Stauchungen entlang einer Dimension (der Zeit) zu berücksichtigen. Beispielsweise kann, bezogen auf den hier betrachteten Anwendungsfall, ein Fahrstreifenwechsel eine unterschiedliche zeitliche Länge aufweisen. Bestimmt man ein Referenzsignal für diesen Fahrstreifenwechsel, kann mittels DTW diese Sequenz auch in den zu analysierenden Zeitreihen wiedergefunden werden, wenn sie dort entsprechend kürzer oder länger auftritt.

Sei das Referenzsignal *C* eine Zeitreihe mit *m* Elementen und die zu analysierende Zeitreihe *Q* eine Sequenz mit *n* Elementen:

$$C = c_1, c_2, \dots, c_i, \dots, c_m$$
$$Q = q_1, q_2, \dots, q_j, \dots, q_n \tag{6.14}$$

Werden die Sequenzen *C* und *Q* in einer *m·n* Distanzmatrix angeordnet, in der jeder Punkt *(i,j)* die (euklidische) Distanz zwischen den Elementen der Zeitreihen c_i und q_j angibt, lässt sich eine Warping-Kurve W bestimmen, die die Distanz zwischen den Zeitreihen minimiert (roter Pfad in Abbildung 6.9 rechts) [177] [178].

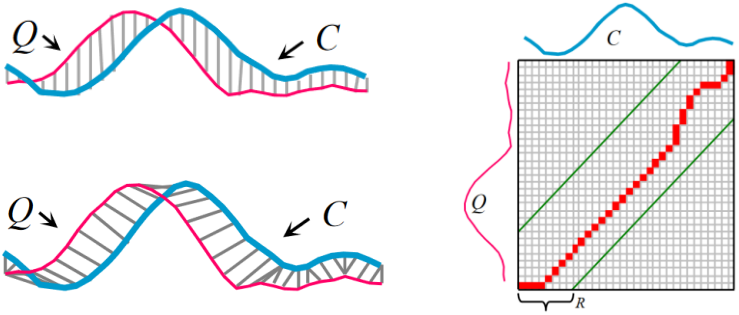


Abbildung 6.9: Die euklidische Distanz (links oben) führt bei phasenverschobenen Signalpeaks zu großen Werten. Dies kann durch das nonlineare Matching im DTW aufgelöst werden (links unten). Die Warping Kurve (rechts) beschreibt den minimalen Abstand. In grün ist zusätzlich das Sakoe-Chiba-Band mit Abstand R angegeben, das als zusätzliche Randbedingung für die Warping Kurve gelten kann [177].

Wendet man das Verfahren auf das dargestellte Beispiel des Fahrstreifenwechsels an, lassen sich mit einem Referenzsignal für einen typischen Fahrstreifenwechsel (beispielsweise Lenkradwinkelverlauf während des Fahrstreifenwechsels) entlang einer beliebigen Aufzeichnung für den Lenkradwinkel potentielle Fahrstreifenwechsel bestimmen.

Eine weitere Möglichkeit für die Musterkennung in den Daten und damit der Klassifikation der Manöver ist der Einsatz künstlicher neuronaler Netze (KNN)[37]. Während die vorgenannten Verfahren vornehmlich darauf abzielen, *eine* Zeitreihe oder *ein* bestimmtes Signal zu analysieren, können mit künstlichen neuronalen Netzen verhältnismäßig einfach auch mehrere Signale gleichzeitig untersucht werden. Geht man beispielsweise davon aus, dass ein charakteristischer Lenkradwinkelverlauf zwar prinzipiell als guter Indikator für einen Fahrstreifenwechsel herangezogen werden kann, lässt sich dieser Verlauf ohne Zuhilfenahme weiterer Signale nicht von einem Verlauf unterscheiden, bei

[37] Für die Grundlagen zu KNN sei hier auf den Anhang dieser Arbeit verwiesen (Anhang A).

dem das Fahrzeug einem Fahrstreifen folgt, die durch die Charakteristik im Streckenverlauf zu einer fahrstreifenwechselähnlichen Trajektorie führt.

Für die Aufgabe der Manöverklassifikation, sowohl des Ego-Fahrzeugs als auch der Objekte, können also KNN-Architekturen entworfen werden, die sich je nach verfügbaren Signalen anpassen und entsprechend trainieren lassen. Beispielsweise könnte ein KNN für die Erkennung der lateralen Basismanöver eine Ausgangsschicht mit drei Perzeptronen innehaben (für die drei Klassen „Fahrstreifenwechsel links“, „Fahrstreifenwechsel rechts“ und „Fahrstreifen folgen“), während die Größe der Eingangsschicht abhängig von der Zahl der zur Verfügung stehenden Signale gestaltet wird. Im Gegensatz zum DTW-Verfahren und seinen Varianten, die für die Anwendung lediglich den Referenzsignalabschnitt benötigen, basiert die erzielbare Genauigkeit der KNN-Algorithmen auf Trainingsdaten, die zunächst in ausreichendem Umfang und mit korrekten Labelannotationen zur Verfügung stehen müssen, bevor diese Algorithmen eingesetzt werden können.

Die Ergebnisse der Anwendung der DTW-Verfahren und KNN finden sich in den Abschnitten 7.3.1 und 7.3.2.

6.2.2 Bestimmung der Zustände

Die Aufgabe zur Bestimmung der Zustände gemäß des in Abschnitt 6.1.4 vorgestellten Modells reduziert sich im Wesentlichen auf zwei Aufgaben: Die Auswertung des longitudinalen Abstands des betrachteten Objektes zum Ego-Fahrzeug und die Auswertung des lateralen Abstands und damit die Zuordnung des Objekts zum entsprechenden Fahrstreifen.

Während der longitudinale Abstand direkt über die Entfernungsmessung eines entsprechenden Sensors verfügbar ist und demnach direkt über die Auswertung des Signals dem jeweiligen Zustand zugewiesen werden kann, gestaltet sich die Bestimmung des lateralen Zustands aufwändiger, da die laterale Abstands- und Winkelmessung für die genaue Fahrstreifenzuordnung alleine nicht ausreicht, sofern nicht auch der Fahrbahnverlauf mitberücksichtigt wird.

Bereits heutige Serien-Assistenzsysteme wie Adaptive Cruise Control (ACC) setzen Techniken zur Fahrstreifenzuordnung und Kursprädiktion ein. Dies erfolgt beispielsweise mittels Schätzung der Krümmungsparameter der Fahrbahn und anschließender Fusion mit den Messungen der videobasierten Fahrstreifenerkennung (vergleiche [179] [180] [181]).

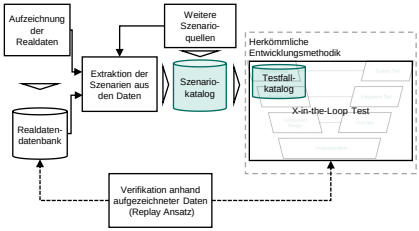
Folglich wird in dieser Arbeit davon ausgegangen, dass auf der Objektlistenebene eines entsprechenden Fahrzeuges Signale für die Fahrstreifenzuordnung der detektierten Objekte bereits zur Verfügung stehen, die direkt für die Bestimmung der Zustände in lateraler Richtung herangezogen werden können.

6.2.3 Bestimmung statischer Szenarioelemente

Das hier vorgestellte Szenario-Metamodell fokussiert auf die Abstraktion dynamischer Szenarioelemente. In Abschnitt 6.1.6 wurde die grundsätzliche Erweiterbarkeit des Modells um statische Elemente dargestellt. Die Umsetzung oder die Betrachtung entsprechender Methoden für die Extraktion dieser Komponenten sind jedoch nicht Bestandteil dieser Dissertation und stellen auch nur bedingt einen Neuigkeitswert dar. Zum einen ist die Erstellung hochauflösender Karten (HD Maps) zur Speicherung einer Vielzahl statischer Szenarioelemente, wie beispielsweise Fahrstreifenverläufe, Linien, Markierungen oder Schilder, längst nicht mehr nur Stand der Forschung, sondern wird von einigen Unternehmen bereits auch kommerziell betrieben (vgl. beispielsweise here[38], TomTom[39], NVIDIA[40]). Zum anderen existieren auch bereits kommerzielle Lösungen für den Erstellungsprozess dieser Karten aus Aufzeichnungen von Realfahrten (vgl. Atlatec[41] oder 3D-Mapping[42]).

[38] https://www.here.com/products/automotive/hd-maps
[39] https://www.tomtom.com/products/hd-map/
[40] https://www.nvidia.com/de-de/self-driving-cars/hd-mapping/
[41] https://www.atlatec.de/
[42] https://www.3d-mapping.de/home/

6.3 Szenariokatalog und Simulation



Sind alle Elemente gemäß den Erläuterungen in den vorherigen Abschnitten extrahiert, stellt sich die Frage, wie diese letztendlich effizient aufbereitet und gespeichert werden können, sodass sie später für die eigentliche Testdurchführung im X-in-the-Loop zur Verfügung stehen.

6.3.1 Speicherbedarf

Eine große Herausforderung im Absicherungsprozess ist der Umgang mit großen Datenmengen. Eine Aufzeichnung der Sensorrohdaten für freigaberelevante Strecken ist kaum praktikabel und ökonomisch schwer durchführbar. Nach Schätzungen des Unternehmens Intel produzieren die Sensoren eines autonomen Fahrzeugs eine Datenmenge von ca. 100 Megabyte/s.[43] Würde man alleine die Sensordaten einer Strecke von 1 Mrd. km bei Autobahnfahrten für die Absicherung (vgl. auch Abschnitt 4.1.1) mit einer angenommenen Durchschnittsgeschwindigkeit von 100 km/h aufzeichnen, kommt man auf einen Speicherbedarf von 3,6 Mio. Terabyte.

Im Vergleich dazu ist das vorgestellte Abstraktionsmodell extrem speichereffizient. Eine Analyse der im Rahmen dieser Arbeit in Matlab erstellten Tensoren ergab einen Speicherbedarf von durchschnittlich ca. 20 Kilobyte pro 60 Sekunden – ohne weitere Optimierungsmaßnahmen. Unter den oben getroffenen Annahmen benötigte man für die Strecke von 1 Mrd. km im Vergleich einen Speicherbedarf von nur 12 Terabyte. Selbst bei einer Erweiterung des

[43] https://newsroom.intel.com/editorials/krzanich-the-future-of-automated-driving/#gs.pjrs7q

Modells um statische Elemente (vgl. Abschnitt 6.1.6), die hier nicht berücksichtigt wurden, und deren Speicherung bleibt der Bedarf der benötigten Speicherkapazitäten ökonomisch realisierbar.

6.3.2 Schneiden der Tensoren

Das in Abschnitt 6.1 entwickelte Szenario-Metamodell stellt einen Ansatz dar, Szenarien aus realen Fahrsituationen zu extrahieren, beantwortet aber per se nicht die Frage, wie diese Szenarien geordnet oder katalogisiert werden können, da eine Tensorrepräsentation zunächst „nur“ die vollständige Abstraktion einer aufgezeichneten Messfahrt beliebiger Länge enthält. Dies ist unabhängig davon, ob die Aufzeichnung wenige Sekunden, eine Stunde oder länger andauert, in der folglich eine Vielzahl an Szenarien enthalten sein können.

Darum ist es notwendig, die Tensoren der Aufzeichnungen in sinnvolle Abschnitte zu zerlegen. Dabei stellt sich die Frage: Welche Länge weist ein zu betrachtendes Szenario auf?

Der Stand der Technik liefert auf diese Fragestellung keine Antwort, da zwar für ein Szenario allgemein eine Zeitspanne angenommen wird, diese aber nicht klar definiert ist (vgl. Abschnitt 3.1.4). Eine Einteilung nach einer zu bestimmenden fixen Zeitdauer kann nicht zielführend sein, da ein und das gleiche funktionale oder logische Szenario wie beispielsweise „Überholen“ je nach Parameterausprägung (z.B. der Differenzgeschwindigkeit der beiden Objekte) eine deutlich unterschiedliche Zeitspanne einnehmen kann.

Hier wird daher ein alternativer Ansatz verfolgt: Ein Szenario besteht aus einer Anzahl von n Szenen, die jeweils durch Aktionen, Manöver oder Events miteinander verbunden sind. Das n ist unbestimmt, beziehungsweise ein Szenario mit größeren n kann wiederum kleinere Teilszenarien mit kleineren n enthalten, wie beispielsweise ein Szenario „Überholen“ aus „Ausscheren“, „Vorbeifahrt“ usw. besteht (vgl. Abbildung 6.10). Als kleinstmöglichste Ausprägung kann für ein Szenario $n=2$ angenommen werden, da eine einzelne Szene keine zeitliche Ausdehnung besitzt.

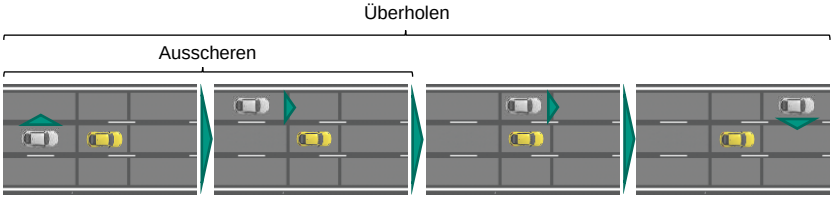


Abbildung 6.10: Szenarien als Folgen von Zuständen

Im Tensor sind die Szenenrepräsentationen in Form des in Abschnitt 6.1.4 beschriebenen Zustandsmodells mit ihrem zugehörigen Zeitstempel enthalten. Daraus ergibt sich die Möglichkeit, die Tensoren aus den Aufzeichnungen entlang dieser Zustandswechsel zu schneiden, um die Teiltensoren zu erhalten, die exakt ein Szenario der Länge n, unabhängig von der tatsächlichen zeitlichen Ausprägung, enthalten. Schneidet man jeweils nach zwei aufeinanderfolgenden Szenen, erhält man die kleinstmöglichen Teilszenarien, die wiederum für komplexere Szenarien wieder zusammengesetzt werden können. Darüber hinaus kann sich der nun ein spezifisches Szenario repräsentierende Teiltensor weiter vereinfachen, da nicht zwangsweise alle Objekte, die in einer Aufzeichnung erfasst wurden, auch in diesem Szenario enthalten sind (Abbildung 6.11).

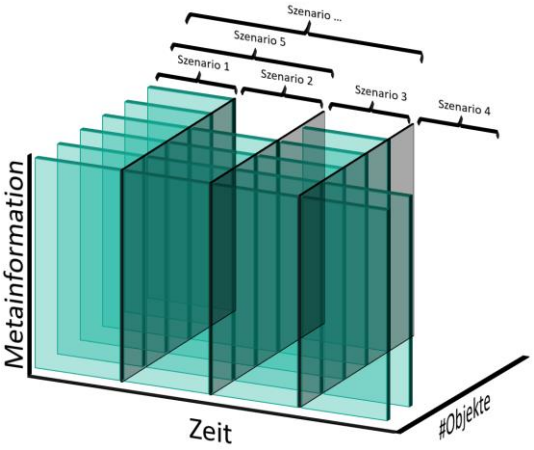


Abbildung 6.11: Prinzip des Schneidens der Tensoren aus den ursprünglichen Aufzeichnungen. Das Schneiden erfolgt anhand der Szenenwechsel und ist nicht notwendig zeitlich äquidistant.

Die so ermittelten Teiltensoren können nun gespeichert werden und stehen für weitere Datenanalysen zur Verfügung. Beispielsweise können an diese Menge der Tensoren mit einfachen Mitteln Abfragen gestellt werden, wie das Selektieren nur bestimmter Szenarien wie „Überholen“, die anschließend getestet werden sollen. Exemplarische Anwendungsmöglichkeiten wurden bereits in [182] veröffentlicht und näher beschrieben.

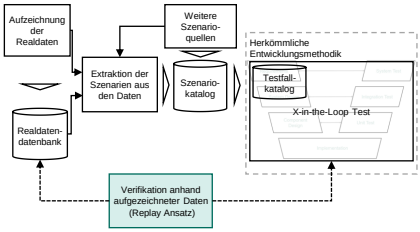
6.3.3 Tensor als Eingangsformat für einen X-in-the-Loop Test

Schlussendlich besteht die Anforderung an das Szenario-Metamodell, dass es in der Lage sein muss, als Beschreibungsformat für eine X-in-the-Loop-Simulation zu dienen oder (automatisch) in ein entsprechendes Format überführt werden kann. Nur dann ist eine Testfallerzeugung und Durchführung im Sinne des dargestellten Prozesses möglich.

Wie bereits im Stand-der-Technik-Teil dargelegt wurde, existiert heute kein vollständiges Standardformat für die Beschreibung von Szenarien und wird je nach verwendeter Toolumgebung unterschiedlich gehandhabt.

Aus diesem Grund wurde hier aus den Tensoren des Szenario-Metamodells heraus direkt das spezifische verwendete Beschreibungsformat der Simulationsumgebung erzeugt. Die Umsetzung ist in Abschnitt 7.2.3 beschrieben.

6.4 Ebenen der Verifikation



Der dargestellte Prozess erlaubt gleich mehrere Möglichkeiten zur Verifikation auf unterschiedlichen Ebenen. Grundsätzlich wird, gemäß der Definition für die Verifikation im Testprozess (vergleiche zum Beispiel [183]), mit dieser sichergestellt, dass die erzeugten Szenarien und Testfälle auch geeignet sind, die Anforderungen zu prüfen. Da in der hier vorgestellten Testmethodik die Szenarien und Testfälle aus den aufgezeichneten Daten abgeleitet werden, muss in diesem Fall mit der Verifikation dargestellt werden, dass die nach dem Prozess erzeugten Szenarien und Testfälle in der Anwendung im X-in-the-Loop Verfahren auch ausreichend die originalen Aufzeichnungen repräsentieren.

Hierbei spielen folgende Einflussfaktoren eine Rolle:

1. Die Abbildungsgenauigkeit der eingesetzten Simulationsmethoden im X-in-the-Loop Vorgehen (vgl. hierzu auch die Abschnitte 3.1.2 und 3.1.3).

2. Der Fehler, der bereits bei der Aufzeichnung der Daten entsteht (wie bereits in den Einschränkungen zum Konzept erläutert (vgl. Abschnitt 5.4).

3. Der Fehler beziehungsweise die Ungenauigkeit, die durch die Abstraktion im Meta-Modell für die Szenariorepräsentation entsteht.

Die ersten beiden Faktoren werden nicht weiter betrachtet, da sie nicht grundlegend mit der hier entwickelten Methode zusammenhängen oder zumindest nicht durch diese bedingt werden.[44] Der dritte Faktor hingegen ist der eigentliche (potentielle) Fehler, der durch die Modellierung bedingt ist und den es für eine erfolgreiche Verifikation zu eliminieren oder zumindest für eine Güteaussage zu bestimmen gilt.

Hierfür werden drei verschiedene Ebenen konzeptionell untersucht und nachfolgend skizziert: die interne Verifikation (Abschnitt 6.4.1), die Verifikation auf der Abstraktionsebene (Abschnitt 6.4.2) und die Verifikation auf der Datenebene (Abschnitt 6.4.3).

6.4.1 Interne Verifikation

Mittels der internen Verifikation kann zum einen überprüft werden, ob das Szenario-Metamodell für die Anwendung korrekt implementiert wurde, zum anderen können aus der Überprüfung heraus auch fehlerhafte Extraktionen aus den Daten (oder Fehler in den Ausgangsdaten) bestimmt werden, die sich im Nachgang gesondert überprüfen lassen.

Ein einfacher Schritt zur internen Verifikation kann durchgeführt werden, indem sämtliche Wertebereiche der Vektoren des ermittelten Tensors auf jeweilige Plausibilität überprüft werden. Beispielsweise könnte ein Eintrag von „500,34 km/h“ für den Vektor $\overrightarrow{dv_e}$ auf einen etwaigen Sensorfehler bei der Datenaufzeichnung hinweisen. Aufwändiger hingegen ist die Überprüfung auf Plausibilität der Einträge, die innerhalb des zulässigen Wertebereichs liegen. Eine Möglichkeit ergibt sich hier aus der vollständigen Formalisierung der Übergänge zwischen den Szenenrepräsentationen durch die Abbildung der Ba-

[44] Dennoch können durch die hier vorgestellten Methoden auch teilweise Fehler dieser beiden Ursachen „aufgedeckt“ werden.

sismanöver in der Übergangsfunktion (vergleiche Abschnitt 6.1.4). Vereinfacht ausgedrückt: Zwischen zwei konkreten Szenenrepräsentationen sind nur bestimmte Basismanöverkombinationen der beteiligten Objekte möglich, andere nicht. Bildet man diese ab und untersucht anschließend die entsprechenden Codierungen in der Tensorrepräsentation für eine Aufzeichnung, lassen sich durch implementierte Abfragen unzulässige Übergänge finden, die wiederum auf die zuvor genannten fehlerhaften Extraktionen oder Fehler in den Ausgangsdaten schließen lassen.

Exemplarisch ist hier dargestellt, welche Übergänge von einem möglichen Zustand Q_1 in den Zustand Q_2 möglich sind, beziehungsweise welche Manöverkombinationen definitiv ausgeschlossen werden können (Abbildung 6.12). Das betrachtete graue Objektfahrzeug befindet sich zunächst im grün dargestellten Bereich (Zustand Q_1) „links vor“ dem Ego-Fahrzeug und anschließend in der nächsten Zustandsrepräsentation im blauen Bereich (Zustand Q_2) „links neben“ dem Ego-Fahrzeug. Unzulässige Manöverkombinationen für diesen Übergang sind in Codierungsschreibweise in der Abbildung rechts in Form der roten Pfade dargestellt. Bremst beispielsweise das Ego-Fahrzeug („-1“) ist dieser Zustandsübergang definitiv nicht möglich, wenn das Objektfahrzeug gleichzeitig konstant fährt („0“) oder sogar beschleunigt („1“).

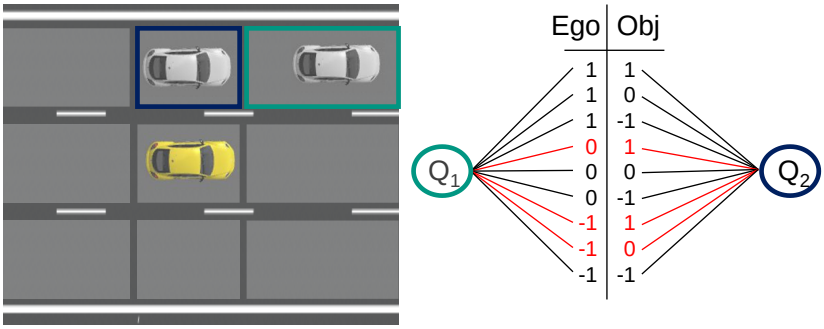


Abbildung 6.12: Übergang zwischen zwei exemplarischen aufeinanderfolgenden Zuständen bezogen auf die Longitudinalmanöver

Auf diese Weise lassen sich sämtliche unzulässigen Übergänge zwischen allen Zuständen anhand der entsprechenden Basismanöverkombinationen in longitudinaler und lateraler Richtung der jeweilig betroffenen Objekte bestimmen. Schematisch sind alle Zustände und Übergänge nachfolgend dargestellt (Abbildung 6.13). Die farblich markierten Elemente kennzeichnen den Ausschnitt, der zuvor exemplarisch dargestellt wurde (vergleiche Abbildung 6.12).

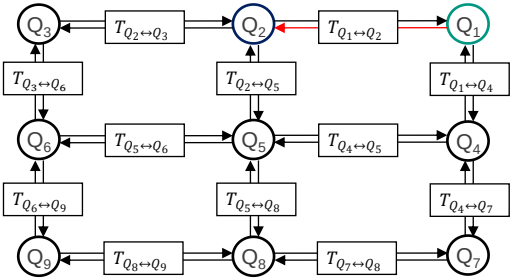


Abbildung 6.13: Schematische Übersicht über alle Zustände und Übergänge

Im Schema sind alle neun möglichen Zustände dargestellt, wie sie bereits schon in Abschnitt 6.1.4 hergeleitet wurden. Bereits das Schema zeigt darüber hinaus, dass nicht jeder Zustand aus einem konkreten Zustand Q_i erreicht werden kann.[45] Die Analyse auf diese ungültigen Zustandsfolgen kann somit ebenso wie die Überprüfung der Wertebereiche für die interne Verifikation herangezogen werden.

6.4.2 Verifikation auf der Abstraktionsebene

Während die interne Verifikation vor allem die Überprüfung der Plausibilität der Werte im Tensormodell zum Ziel hat, zielt die eigentliche Verifikation im Sinne des Prozesses darauf ab, zu prüfen, ob die letztendlich erstellten Szena-

[45] Beispielsweise sind aus dem Zustand Q_1 nur die Zustände Q_2 und Q_4 erreichbar. Alle anderen Zustände können nur über den „Umweg“ weiterer Zustände erreicht werden.

rien und Testfälle ausreichend genau den ursprünglichen Szenarien in den Ausgangsdaten entsprechen. Eine Möglichkeit dies zu überprüfen, stellt die Verifikation auf der Abstraktionsebene dar. Hiermit wird der Abstraktionsgrad des verwendeten Szenario-Metamodells geprüft.

Dafür kann für diesen Verifikationsschritt der Prozess dahingehend erweitert werden, dass nach der Erstellung des eigentlichen Tensors für die Abstraktion nach der Testdurchführung in der X-in-the-Loop Simulation aus dieser ein weiterer Tensor (Replay-Tensor) nach dem gleichen Prinzip erzeugt wird. Geht man davon aus, dass das im X-in-the-Loop Test erzeugte Szenario möglichst dem ursprünglichen Datensatz entspricht, sollten die jeweiligen Tensorrepräsentationen ebenfalls vergleichbar sein (Abbildung 6.14).

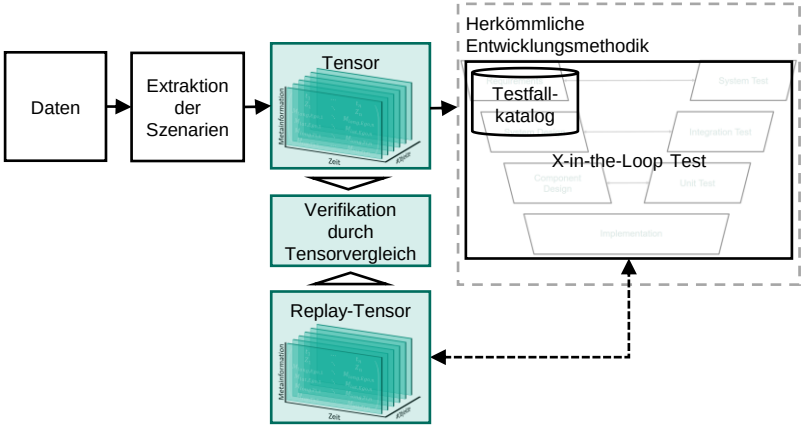


Abbildung 6.14: Prinzip des Vergleichs der erzeugten Tensoren aus den Originaldaten und aus der Testdurchführung für die Verifikation

Eine Möglichkeit für den Tensorvergleich wäre, den Unterschied anhand statistischer Metriken für den gesamten Tensor zu ermitteln. Zu berücksichtigen ist allerdings, dass der Replay-Tensor nicht zwangsweise die gleiche Dimension wie der Ausgangstensor aufweisen muss. Auch werden im Tensor sowohl nominalskalierte (beispielsweise die Codierungen für die Manöver), als auch metrische Variablen (die Parameter der Manöver) erfasst, was bei Anwendung einer gleichen Metrik zu Verzerrungen führen kann. Schlussendlich können

sich Fehler in der Resimulation im zeitlichen Verlauf fortsetzen und verstärken, sodass grade bei größeren Tensoren insbesondere die Einträge für höhere Zeitwerte nicht mehr sinnvoll verglichen werden können.

Infolgedessen ist ein mehrstufiges Vorgehen erforderlich:

1. Zerteilung des Gesamttensors in den relevanten Abschnitt / Repräsentation des relevanten Szenarios (vergleiche auch Abschnitt 6.3.2)
2. Erstellung des entsprechenden Replay-Tensors zu diesem Abschnitt
3. Überprüfung auf Übereinstimmung der Dimensionalität der beiden Tensoren
4. Überprüfung auf Identität für die Einträge der nominalskalierten Elemente
5. Überprüfung auf Abweichung für die Einträge der metrischen Elemente

Mit der abschnittsweisen Erstellung des Replay-Tensors zum jeweilig betrachteten Szenario oder Testfall (Schritt 1 und 2) ist sichergestellt, dass potentielle Fehler aus dem vorherigen Szenarioverlauf keinen Einfluss auf den betrachteten Abschnitt haben. Die nachfolgenden drei Schritte überprüfen stufenweise die Genauigkeit des abgebildeten Szenarios im X-in-the-Loop Vorgehen. Mit dem Abgleich der Dimensionalität im Schritt 3 wird sichergestellt, dass das erstellte Szenario grundsätzlich dem Szenario aus den Daten entspricht (beispielsweise hinsichtlich der Anzahl der beteiligten Objekte). Im vierten Schritt wird geprüft, ob die Sequenz der Szenen und Manöver tatsächlich vollständig übereinstimmen, das heißt das gleiche logische Szenario vorliegt. Mit dem Vergleich der metrischen Elemente im letzten Schritt wird die tatsächliche Abweichung vom letztendlich konkreten Szenario bestimmt. Zumindest für den vierten und fünften Schritt ist zu beachten, dass eine möglichst genaue Abbildung nur gelingen kann, wenn die verwendeten Modelle im X-in-the-Loop Vorgehen hinsichtlich ihrer Parametrierung und ihres Verhaltens ausreichend

validiert die Originalumgebung abbilden können. Für die Bestimmung der Abweichung wird in dieser Arbeit als Metrik der *mittlere absolute prozentuale Fehler* (engl. *MAPE*) verwendet (siehe beispielsweise [184]):

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{T_i - RT_i}{T_i}\right| \tag{6.15}$$

T_i bezeichnet dabei die Einträge des Tensors, die metrischen Variablen angehören. Mit RT_i wird die jeweilig zugehörige Größe des Replay-Tensors bezeichnet. Da die Abweichung bei *MAPE* über alle Größen gemittelt wird, ist diese Metrik für beliebige Tensoren beliebiger Größe anwendbar (für die Ergebnisse der Anwendung siehe Abschnitt 7.4.2).

6.4.3 Verifikation auf der Datenebene

Die letzte Ebene der Verifikation stellt die Verifikation auf der Datenebene dar. Dies bedeutet, die Ergebnisse aus der Testdurchführung (der Simulation) mit den Daten aus der Originalaufzeichnung auf der Signalebene zu vergleichen. Geht man davon aus, dass die interne Verifikation und die Verifikation auf der Abstraktionsebene (weitestgehend) erfüllt sind, können Abweichungen auf der Datenebene „nur“ noch durch den Einsatz nicht-validierter Simulationsmodelle oder –parametrierungen entstehen. Eine vollständige Validierung von Simulationsumgebungen ist nicht Bestandteil der Untersuchung an dieser Stelle, daher wird hier nur ein Überblick über mögliche Validierungstechniken gegeben, wie sie beispielsweise aus der Fahrdynamiksimulation bereits bekannt und etabliert sind:

Subjektive Verfahren	Objektivierbare Validierungstechniken	
- Selbstprüfung - Augenscheinvalidierung / grafischer Vergleich - Visualisierung, Animation - Turing-Test	- Vergleichstest - Fehlerimplementierungstest - Feldtest - Funktionaler Test - Ereignisvalidierung - Submodelltest	- Prädiktive Validierung - Regressionstest - Modellanpassung - Test spezieller Eingangsgrößen - Sensitivitätsanalyse - Statistische Techniken

Tabelle 8: Übersicht über Validierungstechniken, zusammengestellt nach [50]

Zu nahezu jeder dieser Techniken existieren umfangreiche Arbeiten. Balci fasst allein für die statistischen Techniken folgende Methoden zusammen [145]: Varianzanalyse, Konfidenzintervall /-region, Faktoranalyse, Hotellings T^2 Test, Multivariate Varianzanalyse (Standard MANOVA, Permutationsmethoden, nichtparametrische Rangmethoden), nichtparametrische „Goodness-of-fit“ Tests (Kolmogorov-Smirnov Test, Cramer-Von Mises Test, Chi-Quadrat Test), nichtparametrische Tests der Mittelwerte (Mann-Whitney-Wilcoxon Test, Analyse paarweiser Beobachtungen), Regressionsanalyse, Theils Ungleichheitskoeffizient, Zeitreihenanalysen (Spektralanalyse, Korrelationsanalyse, Fehleranalyse) und t-Test.

In dieser Arbeit wurden im Rahmen der Implementierung einzelne der genannten statistischen Methoden eingesetzt, ohne Anspruch auf einen vollständigen Validierungsnachweis (vgl. Abschnitt 7.4).

7 Umsetzung und Ergebnisse aus der Modellierung

Im ersten Schritt wird für die Umsetzung des vorgestellten Detailkonzepts eine dahingehend vereinfachte Umgebung verwendet, dass die Ausgangsdaten statt aus einer realen Aufzeichnung synthetisiert durch eine Simulation erzeugt wurden. Dies hat folgende Vorteile:

1. Keine Abhängigkeit von einem Datensatz und seinen Eigenschaften: Für industrielle Anwendungen und Forschungszwecke stehen grundsätzlich verschiedene Datensätze aus Fahrzeugaufzeichnungen aus unterschiedlichen Quellen zur Verfügung (siehe z.B. KITTI-Datensatz [185]). Diese weisen aber in der Regel Einschränkungen hinsichtlich Umfang und Qualität auf. Beispielsweise waren zum Zeitpunkt der Entstehung dieser Arbeit keine Datensätze bekannt, die die benötigten sensorfusionierten Objektlisten aus Aufzeichnungen auf Autobahnen bereitstellen oder die verfügbaren Autobahnabschnitte hatten nur eine Gesamtlänge von wenigen Kilometern.

2. Durch Simulation erzeugte Daten können mit umfangreichen Meta- und Label-Informationen ausgestattet werden, die als „Ground Truth“ verfügbar sind. Reale Datensätze beinhalten diese Informationen je nach Konfiguration nur eingeschränkt und fehlerbehaftet, beispielsweise für in Bildern enthaltene kleine Objekte oder Objekte in großer Entfernung, insbesondere wenn die Daten in diesen Datensätzen manuell annotiert wurden.

7.1 Workflow mit synthetischen Daten

Der in Kapitel 5 entwickelte Prozess verändert sich unter Verwendung von synthetischen Daten wie folgt und dient hier als Überblick über die in den nächsten Abschnitten dargestellten Ergebnisse. Als Repräsentant für den X-in-

the-Loop Test wird eine closed-loop Simulation verwendet. Es kommen folgende Tools zum Einsatz (Abbildung 7.1):

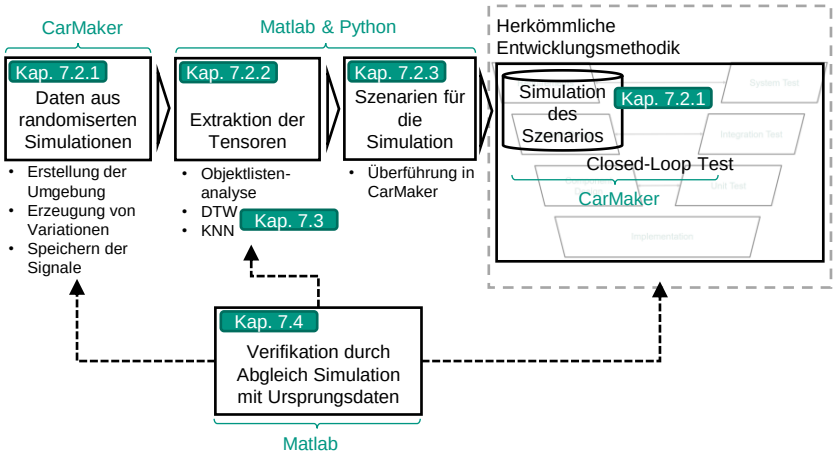


Abbildung 7.1: Workflow und Tools zur Evaluierung mit synthetisch erzeugten Ausgangsdaten

7.2 Umsetzung zur Modellbildung

In den nächsten Abschnitten wird die Umsetzung der Teile des Modells zum Prozess zur Szenarioextraktion beschrieben – von der Erstellung der Daten bis hin zur Simulation der gewonnenen Szenarien.

7.2.1 Simulationsumgebung CarMaker

Sowohl die Datenerzeugung (vgl. links oben in Abbildung 7.1) als auch die eigentliche Simulation (vgl. rechts oben) der extrahierten Szenarien wurden hier mit der für diese Dissertation von der Firma IPG Automotive GmbH zur

Verfügung gestellten Simulationsumgebung CarMaker durchgeführt.[46] CarMaker dient als kommerzielles Tool für den virtuellen Fahrversuch als eine simulative Testumgebung in der Fahrzeugentwicklung. Es besteht aus einer Vielzahl parametrierbarer Komponenten, wie dem virtuellen Prototypen als detailliertes Fahrzeugmodell, sowie Modellen für Fahrer, Verkehr, Straße und Umgebung. Alle Modelle sind echtzeitfähig, und damit prinzipiell für den X-in-the-Loop Test einsetzbar, parametrierbar und über Schnittstellen austauschbar. Neben Modellierungsmöglichkeiten für die Umwelt über einen speziellen Editor zeichnet sich das Tool durch eine manöver- und eventbasierte Ablaufsteuerung aus, die sowohl die Erzeugung von closed-loop als auch open-loop Testfällen erlaubt.

Für die Datengenerierung wurde eine Reihe von Simulationen auf zwei- und dreispurigen Autobahnabschnitten durchgeführt. Dazu wurde ein Oval mit einem Umfang von ca. 10 km erstellt, um sowohl Gradenabschnitte als auch Bereiche mit Kurvenkrümmungen abzudecken.

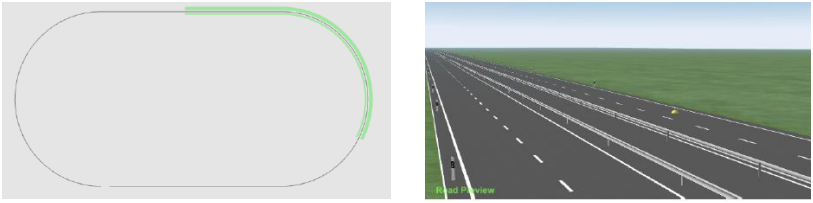


Abbildung 7.2: Design des in CarMaker erstellten Autobahnabschnitts

Auf dieser Strecke wurde autonomer, stochastisch verteilter Verkehr erzeugt, der zu 80% aus PKWs und zu 20% aus sonstigen Objekten (LKWs und Motorrädern) bestand. Auf diese Weise entstanden durchschnittlich 130 Verkehrsobjekte auf der virtuellen Autobahn, deren Anfangspositionen und Geschwindigkeiten zufällig im Abschnitt der ersten 2km Strecke initialisiert wurden.

[46] Grundsätzlich kämen hier für die Anwendung auch andere Simulationsumgebungen infrage (vgl. auch die Tools aus Abschnitt 3.1.2). Die vollintegrierte und frei parametrierbare Manöverablaufsteuerung, sowohl für das Ego-Fahrfahrzeug als auch die anderen Verkehrsobjekte, ist aber im Vergleich zu CarMaker bei den anderen Tools nur teilweise gegeben.

Diese Fahrzeuge fahren autonom unter Beachtung von Sicherheitsabständen auf ihren Fahrstreifen, bremsen und beschleunigen also selbstständig. Zusätzlich wurden in einigen Varianten manuelle Fahrstreifenwechsel für die Objekte getriggert, um auch komplexere Fahrsituationen zu erzeugen. Die durch die Objekte erzielte Verkehrsstärke (gemessen nach 1.000 m vom Startpunkt) lag in Bereichen von 500 bis 3.000 Fahrzeugen pro Stunde auf den beiden betrachteten Fahrstreifen.

Darüber hinaus wurde ein Ego-Fahrzeug mit Fahrermodell parametriert, das selbständig, je nach Verkehrssituation, überholt. Dieses Fahrzeug wurde mit virtuellen Sensoren ausgestattet, die sowohl nach vorne, zur Seite, als auch nach hinten ausgerichtet wurden, um damit ein Fahrzeug mit Radar-, Lidar- und Kamerasensoren beziehungsweise den Ergebnissen der Sensorfusion zu repräsentieren. Um eine möglichst breite Datenbasis zu erreichen, wurden automatische Variationen des Ursprungssetups erzeugt und durchgeführt, bei denen Parameter wie die grundsätzliche Ego-Geschwindigkeit, das maximale Beschleunigungs- und Bremsverhalten in Längs- und Querrichtung oder die Überholwahrscheinlichkeit variiert wurden.

Mit der Durchführung der Simulationen zu dem vorgestellten Setup wurden eine Vielzahl von Signalen für die spätere Auswertung aufgezeichnet. Die Auswahl der Größen erfolgte anhand typischerweise vergleichbarer zur Verfügung stehender Größen im realen Fahrzeug. Neben Größen des Ego-Fahrzeugverhaltens (Geschwindigkeit, Gierwinkel, Beschleunigung, Lenkradwinkel, Position usw.) wurden auch Größen aus der Sensorik (wie Relativgeschwindigkeit, -abstand, -winkel zum Objekt, Objekt ID usw.) verwendet. Neben diesen für reale Aufzeichnungen repräsentativen Größen wurden auch Ground Truth Größen zu den Objekten (Absolutposition, Geschwindigkeit, Fahrstreifenzuordnung, Abstand zu den Fahrstreifenmarkierungen usw.) für die Validierung der Ergebnisse gespeichert. Alle Signale wurden mit einer Frequenz von 50 Hz aufgezeichnet und im .csv Format gespeichert.

Auf diesen Abschnitten wurden ca. 200 simulierte Fahrten mit unterschiedlichen Anzahlen an Verkehrsteilnehmern im Sichtbereich des Ego-Fahrzeugs durchgeführt. Die durchschnittliche Szenariolänge betrug ca. 30 Sekunden.

7.2.2 Erzeugung der Tensoren

Um das Szenario-Metamodell aus Abschnitt 6.1 mit den Ergebnissen aus den Aufzeichnungen zu befüllen, sind eine Reihe von Teilaufgaben notwendig.

Bestimmung der Longitudinalmanöver für das Ego-Fahrzeug:

Die definierten longitudinalen Basismanöver lassen sich direkt aus der Auswertung des Beschleunigungssignals (bzw. Differenzierung des Geschwindigkeitssignals nach der Zeit) ableiten. Die Unterscheidung nach den in Abschnitt 6.1.2 definierten Schwellenwerten entlang der Zeit ermöglicht die direkte Zuordnung zu den Basismanövern „Zielgeschwindigkeit erhöhen/verringern/halten".

Bestimmung der Longitudinalmanöver für die Objekte:

Nutzt man die Ground Truth Informationen der Objekte, erfolgt die Bestimmung der Manöver analog zum Vorgehen für das Ego-Fahrzeug. Da diese Informationen in der Realität nicht vorliegen, ist ein alternativer Ansatz, das Geschwindigkeitssignal eines Objekts aus der Addition der Ego-Geschwindigkeit und der Relativgeschwindigkeit zum Objekt aus dem Sensorsignal zu ermitteln. Anschließend werden die gleichen Schwellenwerte für die Unterscheidung der Basismanöver wie für das Ego-Fahrzeug herangezogen.

Bestimmung der Lateralmanöver für das Ego-Fahrzeug:

Die definierten lateralen Basismanöver „Fahrstreifenwechsel rechts", „Fahrstreifenwechsel links" und „Fahrstreifen folgen" lassen sich aus einer Analyse des Signals „Lateraler Abstand zur nächsten linken Fahrstreifenmarkierung"[47] bestimmen, das typischerweise auch in einem realen kamerabasierten Sensorsystem bei einer Linien- oder Fahrstreifenerkennung vorliegt. Dieses Signal

[47] Alternativ kann natürlich gleichermaßen die rechte Fahrstreifenmarkierung herangezogen werden.

bildet bei Durchführung eines Fahrstreifenwechsels den Verlauf einer typischen Sprungfunktion (siehe Abbildung 7.3).

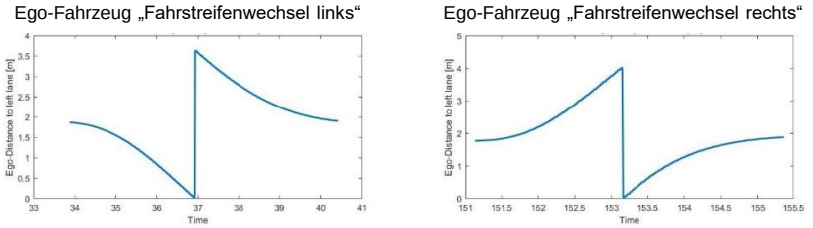


Abbildung 7.3: Typischer Signalverlauf bei Fahrstreifenwechselmanövern

Durch die Bestimmung der Extremstellen der Ableitung des Signals können die Fahrstreifenwechselmanöver eindeutig bestimmt werden. Der genaue Beginn beziehungsweise das Ende des Fahrstreifenwechsels gemäß Definition 6.3 (vergleiche Abschnitt 6.1.3) bestimmt sich zusätzlich aus der Analyse des lateralen Querversatzes des Fahrzeugs, wie in Abschnitt 6.1.3 beschrieben.

Bestimmung der Lateralmanöver für die Objekte:

Wenn nicht auf die Ground Truth Information der Objekte zurückgegriffen wird, können im Gegensatz zu den Lateralmanövern des Ego-Fahrzeugs die Lateralmanöver der Objekte nur auf Basis von Berechnungen aus mehreren Signalen des Egofahrzeugs und seiner Sensoren geschätzt werden. Eine typische Sensorfusion liefert Relativwinkel α und Relativabstand ds zum Objekt, woraus sich der relative Querabstand ds_y des Objekts zum Ego-Fahrzeug bestimmen lässt. Dies lässt allerdings keinen Rückschluss auf die Fahrstreifenzuordnung des Objekts zu, sofern die befahrene Strecke nicht eine Gerade darstellt. Steht auch keine direkte Information über den Streckenverlauf zur Verfügung, kann zumindest der Kurvenradius r aus dem Fahrverhalten des Ego-Fahrzeugs (Geschwindigkeit v und Querbeschleunigung a_y) approximiert werden:

$$r = \frac{v^2}{|a_y|} \tag{7.1}$$

- r (gefahrener) Kurvenradius
- ds gemessener Relativabstand
- ds_x Längskomponente des Abstands
- ds_y Querkomponente des Abstands
- α gemessener Relativwinkel
- y_{off} imaginärer Fahrzeugversatz an der Objektposition
- l_{off} Fahrstreifenbreite / 2

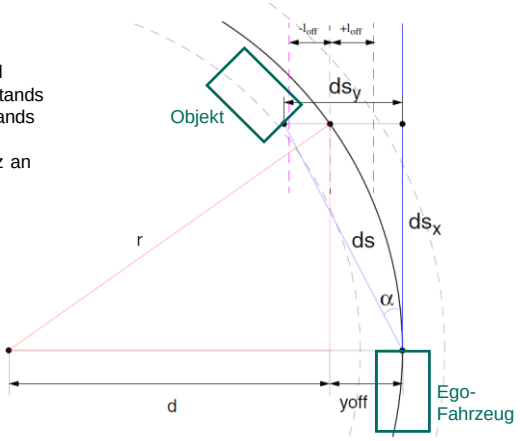


Abbildung 7.4: Bestimmung der Fahrstreifenzuordnung des Objekts nach [186]

Gemäß der Darstellung in Abbildung 7.4 lässt sich der imaginäre Fahrzeugversatz des Ego-Fahrzeugs auf Höhe des Objekts zur Spur (in der Abbildung angedeutet durch die gestrichelten Linien) bestimmen zu:

$$y_{off} = r - d = r - \sqrt{r^2 - {ds_x}^2} \tag{7.2}$$

Das Objekt befindet sich approximativ nicht in der gleichen Fahrspur des Ego-Fahrzeugs, wenn gilt:

$$|ds_y| > y_{off} + l_{off} \tag{7.3}$$

Ist Ungleichung 7.3 erfüllt und weist ds_y ein negatives Vorzeichen auf, befindet sich das Objekt auf dem Fahrstreifen links des Ego-Fahrzeugs, bei einem positiven Vorzeichen entsprechend auf dem Fahrstreifen rechts. Die Übergänge kennzeichnen folglich die Fahrstreifenwechsel und können damit zur Bestimmung der lateralen Basismanöver der Objekte herangezogen werden.

Bestimmung der kinematischen Größen und der Zustände:

Für die Vervollständigung der Tensoren werden neben den identifizierten Manövern weitere kinematische Größen benötigt, wie in den Abschnitten 6.1.2 und 6.1.3 dargestellt.

Folgende Tabelle zeigt die hierfür heranzuziehenden Simulationsgrößen in CarMaker für jeweils das Ego-Fahrzeug und die Objekte (über den Sensor des Ego-Fahrzeugs) für die benötigten kinematischen Größen in longitudinaler Richtung (Tabelle 9):

Größe im Tensor	**Ego-Fahrzeug**	**Objekte**
t_{long} - Zeitpunkt des Beginns eines neuen Longitudinalmanövers	Wert der Größe *Time* zu Beginn eines neuen Manövers	Wert der Größe *Time* zu Beginn eines neuen Manövers
v_s - Anfangsgeschwindigkeit	*Car.v* zu Beginn eines neuen Manövers	Differenz aus: *Sensor.Object.*.relvTgt.RefPnt.dv.x* und *Car.v* zu Beginn eines neuen Manövers
v_e - Endgeschwindigkeit	*Car.v* am Ende des Manövers	Differenz aus: *Sensor. Object.*.relvTgt.RefPnt.dv.x* und *Car.v* am Ende des Manövers
Δt_{long} - Dauer	Differenz aus *Time* zu Beginn und Ende des Manövers	Differenz aus *Time* zu Beginn und Ende des Manövers
Δx - Zurückgelegte Distanz	Integral aus *Car.v* im entsprechenden Zeitabschnitt	Integral aus o.g. Differenz im entsprechenden Zeitabschnitt

Tabelle 9: Kinematische longitudinale Größen im Tensor und zugehörige Simulationsgrößen

Analog zeigt nachfolgende Übersicht die Größen für die laterale Richtung (Tabelle 10):

Größe im Tensor	**Ego-Fahrzeug**	**Objekte**
t_{lat} - Zeitpunkt des Beginns eines neuen Lateralmanövers	Wert der Größe *Time* zu Beginn eines neuen Manövers	Wert der Größe *Time* zu Beginn eines neuen Manövers
y_s - Lateralposition zu Beginn des Manövers	*Sensor.Road.*.ty* zu Beginn eines neuen Manövers	Schätzung gemäß Formeln 7.1 bis 7.3 und zugehöriger Simulationsgrößen[48] zu Beginn des Manövers
y_e - Lateralposition am Ende des Manövers	*Sensor.Road.*.ty* am Ende des Manövers	Schätzung gemäß Formeln 7.1 bis 7.3 und zugehöriger Simulationsgrößen am Ende des Manövers
Δt_{lat} - Dauer (des Lateralmanövers)	Differenz aus *Time* zu Beginn und Ende des Manövers	Differenz aus *Time* zu Beginn und Ende des Manövers

Tabelle 10: Kinematische laterale Größen im Tensor und zugehörige Simulationsgrößen

Schlussendlich müssen noch die Zustände für das in Abschnitt 6.1.4 eingeführte Zustandsmodell erfasst werden. Während die Fahrstreifenzuordnung, wie zuvor dargestellt, bereits die Zustandsbestimmung in lateraler Richtung beschreibt, kann die Auswertung in longitudinaler Richtung durch die einfache Auswertung des Relativabstandssignals in Längsrichtung durchgeführt werden.

[48] Es werden folgende Simulationsgrößen verwendet: *Car.v* (für v), *Car.ay* (für a_y), *Sensor.Object.*.relvTgt.RefPnt.ds.x* (für ds_x) und *Sensor.Object.*.relvTgt.RefPnt.ds.y* (für ds_y). l_{off} als halbe Fahrstreifenbreite wird als konstant angenommen.

7.2.3 Generierung der (X-in-the-Loop) Simulationsbeschreibung

Die in den Testdaten aufgezeichneten Signale wurden entsprechend dem im vorherigen Abschnitt erläuterten Vorgehen analysiert. Für diesen Schritt wurde in Matlab eine Umgebung implementiert, die die .csv Dateien auf die enthaltenen Signale analysiert und automatisiert zu jeder .csv Datei, als Datenaufzeichnung für eine (virtuelle) Testfahrt, den entsprechenden Tensor mit den jeweiligen Einträgen generiert. Hierzu können sowohl die aus der Simulationsumgebung vorhandenen Ground Truth Signale verwendet werden, als auch die Aufzeichnungen aus den virtuellen Fahrzeugsensoren. Der gewonnene Tensor kann wiederum in seine enthaltenen Szenarien zerlegt werden, sodass zu jedem Szenario ein repräsentierender Teiltensor erzeugt und gespeichert wird (vergleiche Abschnitt 6.3.2). Der Teiltensor beinhaltet letztendlich die Abstraktion des so gewonnenen Szenarios. Für die X-in-the-Loop Simulation muss aus der Tensorbeschreibung ein Format gewonnen werden, das es wiederum ermöglicht, die Simulation (automatisiert) auszuführen.

Zu diesem Zweck wurde für das Simulationstool CarMaker ein Konverter in Matlab implementiert, der das in CarMaker verwendete Beschreibungsformat für eine Simulation aus den Tensordaten befüllt. CarMaker verwendet für die Beschreibung ein eigenes Format („TestRun“), bei dem die Komponenten einer Simulation in sogenannten Infofiles definiert werden. Das übergeordnete TestRun-Infofile beinhaltet alle Informationen und Verweise auf weitere Files oder Modelle, die benötigt werden, um eine X-in-the-Loop Simulation zu instanziieren. Dazu gehören die Referenzierung der zu verwendenden Modelle (beispielsweise des Ego-Fahrzeugs), Parameter für die Ablaufsteuerung, Fahrer, Umgebung, Verkehrsobjekte und weitere Bestandteile. Modelle und Parameter werden dabei vom Programm definierten Strings als Keys zugewiesen, die von der Simulation bei der Initialisierung ausgelesen werden.

Mittels des Konverters werden die Einträge aus den Tensoren transformiert und auf die entsprechenden Keys im TestRun-Infofile geschrieben.

Im Rahmen dieser Arbeit stand keine konkrete zu testende Funktion als Modell zur Verfügung. Daher wurde das in CarMaker integrierte Fahrermodell *IPGDriver* herangezogen, um die im Tensor gespeicherten Manöver für das Ego-Fahrzeug umzusetzen beziehungsweise zu reproduzieren. An dieser Stelle ist zu beachten, dass das Modell zwar grundsätzlich Manövervorgaben in Closed-Loop umsetzen kann, dabei aber eine getrennte, unabhängige Betrachtung der Längs- und Querführung, wie es das zuvor eingeführte Basismanöverkonzept erfordert, nicht möglich ist. Gleiches gilt für die Manöversteuerung der Verkehrsobjekte. Daher mussten die in den Tensoren erfassten Basismanöver zunächst in eine für CarMaker ausführbare Manöverdefinition überführt werden. Zu diesem Zweck werden die Basismanöver, sofern eine zeitliche Überlappung gegeben ist, entlang der Zeitachse geschnitten, sodass jeder Manöverwechsel in Längs und Querführung synchronisiert und damit CarMaker-kompatibel vorgegeben werden kann (Abbildung 7.5).

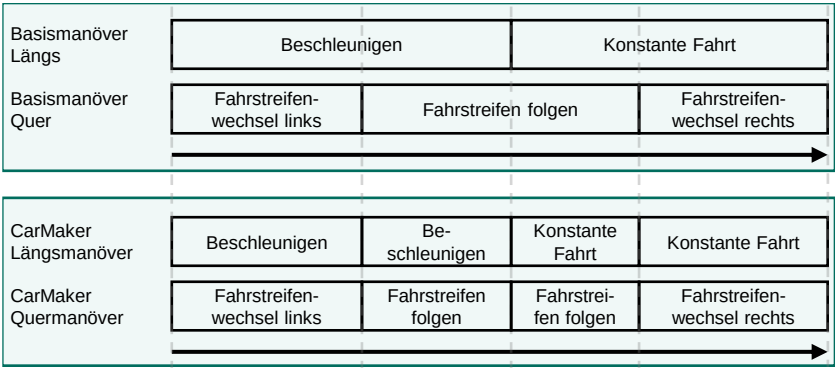


Abbildung 7.5: Prinzip der Überführung der Basismanöver aus dem Tensor in die CarMaker Manöverdefinition

Um dies zu ermöglichen, müssen für die zu schneidenden Manöver die neuen entsprechenden Zielgrößen für die Simulation bestimmt werden. Wird beispielsweise ein Beschleunigungsmanöver, das durch die Anfangsgeschwindigkeit v_s=50 km/h und Endgeschwindigkeit v_e=100 km/h bei einer Dauer von Δt_{long}=10 s definiert ist, nach t=6 s in zwei Manöver geschnitten, muss für das erste neue Manöver eine neue plausible Endgeschwindigkeitsvorgabe ermittelt

werden (die gleichzeitig die neue Anfangsgeschwindigkeit für das anschließende Manöver darstellt). Zur Bestimmung der Zielgrößen für die auf diese Weise geteilten Manöver wurden die Schätzmodelle eingesetzt, wie sie in den Abschnitten 6.1.2 und 6.1.3 vorgestellt wurden. Sowohl für das Ego-Fahrzeug als auch für die jeweils enthaltenen dynamischen Verkehrsobjekte wurden die Manöver auf diese Weise in der Simulationsumgebung definiert.

Für die Definition der statischen Umgebung (z.B. Straßenverlauf, Verkehrsschilder) wurden in der Simulation dahingehend eine vereinfachte Annahme getroffen, dass hierfür erneut die bereits modellierte Umgebung für die Erstellung der Ursprungsdaten herangezogen wurde.

7.3 Ergebnisse zu den Extraktionsverfahren

In den vorherigen Abschnitten wurde eine Vorgehensweise aufgezeigt, wie aus aufgezeichneten Daten (hier aus einer quasi-zufälligen Simulation), mittels der Analyse geeigneter Signale die Tensordaten extrahiert werden und anschließend wiederum aus dem Tensor eine Closed-Loop Simulation erstellt wird.

Um die Informationen für die Tensordaten zu gewinnen, wurden alternativ zu den in der Simulation vorhandenen Ground Truth Daten explizit Signale ausgewählt, die typischerweise nach einer Sensorfusion erzeugt oder aus dieser gewonnen werden. Für diese Objektlistensignale existiert allerdings kein Standard, sodass nicht gewährleistet ist, dass diese Signale in dieser oder ähnlicher Form überhaupt im Fahrzeug zur Verfügung stehen.

Aus diesem Grund wurden weitere Verfahren untersucht, mit dem Ziel, die Basismanöver auch mit alternativen Methoden zu identifizieren, die idealerweise auch auf andere beziehungsweise variierbare Signalkombinationen anwendbar sind. Da die longitudinalen Basismanöver verhältnismäßig einfach identifiziert werden können, beschränken sich die nachfolgend erläuterten Verfahren auf die Identifikation der lateralen Basismanöver.

7.3.1 Zeitreihenanalysen

Zunächst wurden unter anderem die Zeitreihenanalyseverfahren DTW, DDTW und WDTW für die Bestimmung der Basismanöver aus Sensorsignalen in der Simulation angewandt (für die Verfahren vergleiche Abschnitt 6.2.1). Dafür wurden Referenzsignalabschnitte für eine durchschnittliche Länge eines Basismanövers, beispielsweise eines Fahrstreifenwechsels, erzeugt (siehe schwarze Kurve in Abbildung 7.6).

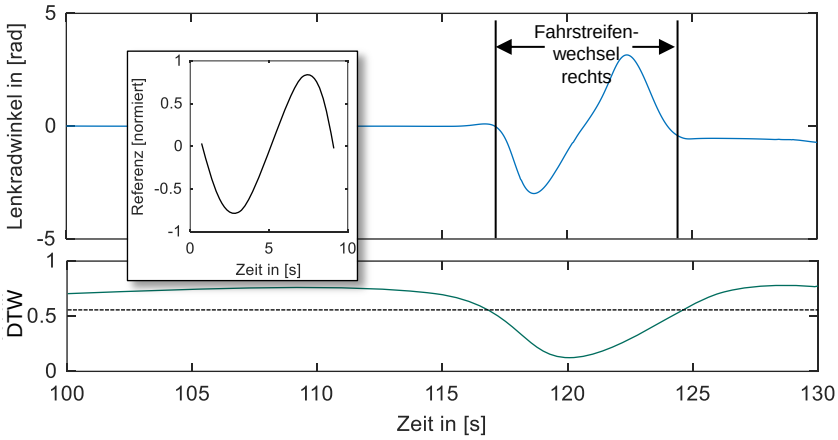


Abbildung 7.6: Anwendung des DTW-Algorithmus anhand eines Referenzsignalabschnitts für den Lenkradwinkel zur Bestimmung des Manövers "Fahrstreifenwechsel rechts"

Diese Abschnitte wurden unter Anwendung der genannten Verfahren nach dem Sliding Window Prinzip[49] über eine Messreihe verschoben. Aus der Warping-Kurve ergibt sich die jeweilige minimale Distanz im betrachteten Fensterabschnitt der Messreihe (blaue Kurve) zum Referenzsignal. Stellt man auf diese Weise die Funktion des minimalen Abstands des DTW-Algorithmus über

[49] Nach dem Sliding Window Prinzip wird sequentiell der Referenzsignalabschnitt mit einem Abschnitt gleicher Länge aus der Messreihe verglichen. Anschließend wird der nächste Abschnitt aus der Messreihe zum Vergleich herangezogen usw.

die vollständige Messreihe dar (grüne Kurve), erhält man potentielle „Kandidaten“ für die gesuchten Manöver an den Stellen der lokalen Minima dieser Funktion. In der Abbildung ist in schwarz gestrichelt der Schwellenwert für die Distanzfunktion dargestellt, der für diesen Fall die Genauigkeit bei Anwendung des DTW für die Bestimmung des gesuchten Manöverabschnitts maximiert.

Wendet man den DTW und seine Varianten anhand der Einzelsignale zur Bestimmung der Quermanöver nach dem vorgestellten Prinzip an, lassen sich folgende Ergebnisse erzielen: Auf den Testdaten wurden für das Ego-Fahrzeug mit dem DTW ca. 92% der lateralen Basismanöver richtig erkannt (mit DDTW 90% bzw. WDTW 93%). Auch für die Objekte wurden ca. 92% der lateralen Basismanöver richtig erkannt (mit DDTW 93% bzw. WDTW 84%)[50]. Es ist allerdings einzuschränken, dass diese Ergebnisse nur auf den Einzelsignalen erzielbar sind, die den optimalen Informationsgehalt zur Bestimmung der Manöver innehaben. Für die Quermanöver des Ego-Fahrzeugs gilt dies beispielsweise für das Signal „Lateraler Abstand zur nächsten linken Fahrbahnmarkierung“.

Damit stellt sich in Anbetracht der eingangs genannten Motivation die Frage, ob und wie diese Verfahren auch auf unbekannte oder zumindest potentiell variierende zur Verfügung stehende Signale und Signalkombinationen angewandt werden können.

Zu diesem Zweck wird ein schrittweises Vorgehen angewandt:

1. Bestimmung der Korrelation zwischen den insgesamt zur Verfügung stehenden Signalen, um ähnliche oder auch redundante Signale im ersten Schritt ausschließen zu können. Dies erfolgt durch die Clusterbildung der Signale aus denen jeweils ein zufälliges Signal für die weiteren Schritte ausgewählt wird. Es werden acht Cluster gebildet und

[50] Für die Anwendung auf Einzelsignalen wurden diese Ergebnisse in der Masterthesis (Wang, Jiawei; 2019) erzielt.

folglich acht Signale ausgewählt.[51] Je nach Durchführung können somit unterschiedliche Signale ausgewählt werden. Mit diesem Schritt wird sichergestellt, dass die acht herangezogenen Signale wenig miteinander korrelieren und damit insgesamt einen hohen Informationsgehalt innehaben.

2. Bestimmung der Referenzsignalabschnitte zu jedem der acht Signale anhand eines repräsentativen Beispiels zu dem die Ground Truth Informationen der zu bestimmenden Manöver vorhanden oder ableitbar sind (im realen Fall beispielsweise aus einer zeitsynchronen Kameraaufzeichnung).

3. Anwendung des DTW für jedes der acht Signale an den zu analysierenden aufgezeichneten Messdaten analog zum zuvor beschriebenen Vorgehen.

4. Bestimmung der drei Signale mit der höchsten Genauigkeit in der Manövererkennung und „Komposition" einer neuen gewichteten Distanzfunktion aus den einzelnen Distanzfunktionen der drei Signale. Damit wird sichergestellt, dass das Signal, das am Besten als Klassifikator geeignet ist, den größten Einfluss auf die so designte Distanzfunktion hat.

5. Anwendung der neuen Distanzfunktion über die Messdaten zur Bestimmung der Manöver.

In der Umsetzung gestaltet sich dies am Beispiel der lateralen Basismanövererkennung für die Objekte wie folgt:

Schritt 1: Die paarweise Korrelation zwischen zwei beliebigen Signalen *x* und *y* kann nach dem Pearson-Koeffizienten *r* bestimmt werden zu:

$$r = \frac{cov(x,y)}{s_x \cdot s_y} \tag{7.4}$$

[51] Die Festlegung auf genau acht Cluster erfolgt hier willkürlich, um den Implementierungsaufwand für die nachfolgenden Schritte zu verringern. Grundsätzlich kann und sollte die optimale Clusteranzahl auch methodisch bestimmt werden (vgl. z.B. Elbow-Methode).

mit *cov* als Kovarianz und s_x bzw. s_y als jeweilige Standardabweichung.

Berechnet man die Korrelation zu allen möglichen paarweisen Kombinationen verfügbarer Signale, lassen sich die Ergebnisse in einer Heatmap darstellen (siehe Abbildung 7.7). Je heller die Markierungen, desto ähnlicher im Sinne der Korrelation sind die Signale zueinander.

Abbildung 7.7: Exemplarische Heatmap zur Korrelation paarweiser Signalkombinationen einer Datenaufzeichnung

Anschließend kann die Clusterbildung nach dem Ward-Verfahren bestimmt werden, um die Distanzen der Korrelationen der Signale zueinander hierar-

chisch abzubilden [187]. Dabei werden „ähnliche“ Signale schrittweise vereinigt und zu Clustern zusammengefasst, deren Abstand zueinander über die vertikale Achse im Dendrogram kenntlich gemacht wird (Abbildung 7.8).

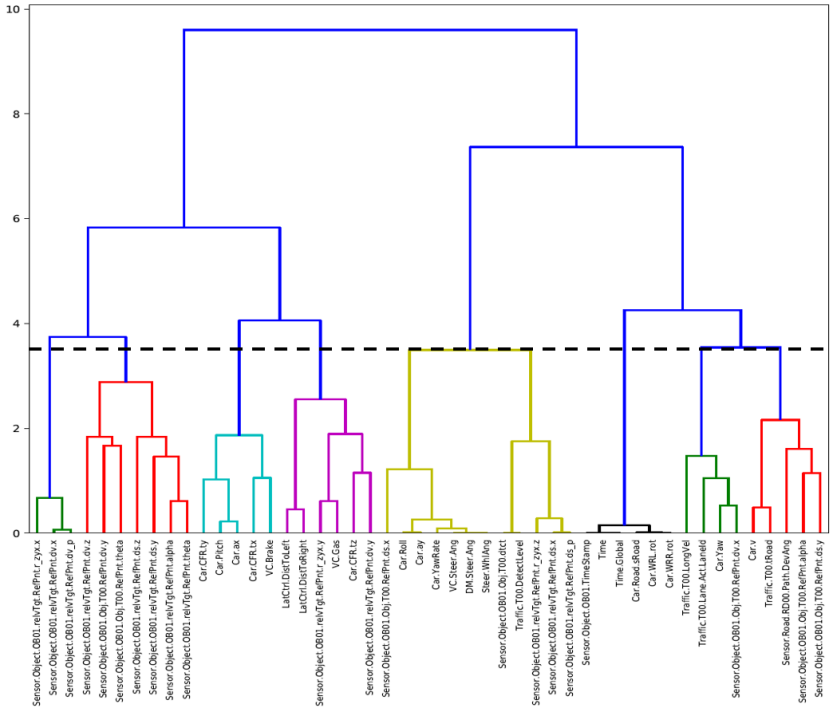


Abbildung 7.8: Dendrogram als Ergebnis des Ward-Verfahrens

Je nach Wahl eines festen Abstandes lassen sich unterschiedliche Zahlen an Clustern bilden. Hier wird der Abstand so gewählt (gestrichelte Linie), dass genau acht, maximal verschiedene, Cluster entstehen. Signale im gleichen Cluster sind an der gleichen Farbgebung zu erkennen. Aus diesen acht Clustern wird, wie zuvor beschrieben, jeweils ein Signal ausgewählt. Die Auswahl erfolgt entweder zufällig, wenn die Signale nicht bekannt sind, oder unter Anwendung von Vorwissen, falls bekannt.

Schritt 2 und 3: Anschließend werden zu den acht Signalen die Referenzsignalabschnitte für die drei lateralen Basismanöver bestimmt. In der Simulationsumgebung geschieht dies unter Zuhilfenahme des Ground Truth Signals für die Fahrstreifenzuordnung der Objekte und der Extraktion der Referenzsignale in den entsprechenden Abschnitten. Im Anschluss wird der DTW nach dem eingangs vorgestellten Sliding Window Verfahren für die acht Referensignale auf die Testdaten angewendet (siehe auch Abbildung 7.6). Aufgrund des für diese Verfahren verhältnismäßig hohen benötigten Rechenaufwands, wurde an dieser Stelle lediglich eine Teilmenge der Testdaten, bestehend aus 80 Testfahrten mit jeweils mindestens einem enthaltenen Fahrstreifenwechsel für die Verkehrsobjekte, ausgewählt.

Schritt 4: Nach dem Durchlauf kann die Genauigkeit für jedes Signal als Klassifikator für die lateralen Basismanöver der Objekte bestimmt werden. Eine Klassifikation für ein Basismanöver wird dabei hier als korrekt angesehen, sobald die Fensterbreite des Referenzsignalabschnitts zum Minimum der Distanzfunktion aus dem DTW zu mindestens 50% überlappend zur Breite des tatsächlichen Fahrstreifenwechselabschnitts anhand der Ground Truth Daten ist. Nach dieser Metrik ergab sich auf den Testdaten eine Genauigkeit von 81,1% für das „beste“ Signal und 73,3% respektive 46,6% für das zweit- und drittbeste Signal.

Schritt 5: Für diese drei Signale wird eine neue Distanzfunktion aus der gewichteten Summe der drei ursprünglichen Distanzfunktionen erstellt. In den Untersuchungen zeigte sich eine Gewichtung von 50% Signal 1, 30% Signal 2 und 20% für Signal 3 als zielführend. Tatsächlich konnte mit Anwendung dieser neuen Distanzfunktion eine Genauigkeit von 93,3% korrekt klassifizierter lateraler Basismanöver für die Objekte erreicht werden.

7.3.2 Künstliche neuronale Netze

Eine weitere Möglichkeit zur Analyse der Zeitreihensignale stellt der Einsatz künstlicher neuronaler Netze dar (vgl. auch Abschnitt 6.2.1 und Anhang A). Insbesondere rekurrente Netzarchitekturen (engl. Recurrent Neural Networks) sind zur Verarbeitung dieser kontinuierlichen Datenströme geeignet, da sie

Rückkopplungen enthalten, die die Speicherung von internen Zuständen ermöglichen [188] [189]. Eine Variante dieser Architekturen stellen Long Short-Term Memory (LSTM) Netze dar (vgl. auch Anhang B).

Diese Eigenschaft der rekurrenten Architekturen der LSTMs kann hier genutzt werden, um künstliche neuronale Netze über mehrere Signale und gleichzeitig über einen zeitlichen Abschnitt einer Zeitreihe anzuwenden. Je nach Anzahl der jeweils zu verwendenden Signale definiert sich dabei korrespondierend die entsprechende Größe der Eingangsschicht des neuronalen Netzes. Es wurden folgende Signale verwendet und untersucht (zur Beschreibung der Signale vergleiche auch Abbildung 7.4):

- ds gemessener Relativabstand zum Objekt
- α gemessener Relativwinkel zum Objekt
- β gemessener Gierwinkel des Objekts um seine Hochachse
- ds_y Querkomponente des gemessenen Relativabstands
- dv_y gemessene relative Quergeschwindigkeit des Objekts
- φ Gierwinkel des Ego-Fahrzeugs
- φ' Gierrate (Ableitung des Gierwinkels) des Ego-Fahrzeugs
- a_y Querbeschleunigung des Ego-Fahrzeugs
- dL Abstand des Ego-Fahrzeugs zur nächsten Fahrstreifenmarkierung links
- ω Lenkradwinkel des Ego-Fahrzeugs

Während die ersten fünf Signale sich auf durch einen Sensor gemessene Objektgrößen relativ zum Ego-Fahrzeug beziehen, stellen die übrigen Signale Größen des Ego-Fahrzeugs dar, die keinen Bezug zum Objekt aufweisen. Auf Basis dieser Signale wurden verschiedene Signalkombinationen untersucht, zu

denen entsprechende Netze *a* bis *i* mit passenden Eingangsschichten entworfen wurden (Abbildung 7.9):

	Sensor (Objektlistensignale)					Ego-Fahrzeug Signale				
#	ds	α	β	ds_y	dv_y	φ	φ'	a_y	dL	ω
a	x	x		x						
b	x	x		x	x					
c	x	x	x	x	x					
d	x	x	x	x	x	x		x	x	
e	x	x	x	x	x		x	x	x	
f	x	x	x	x	x	x		x	x	x
g	x	x	x	x	x		x	x	x	x
h	x	x	x	x	x	x	x	x	x	x
i						x	x	x	x	x

Abbildung 7.9: Untersuchte Signalkombinationen als Eingangsvektor für die künstlichen neuronalen Netze

Während das Netz *i*, das nur Ego-Fahrzeug Signale enthält, dafür vorgesehen ist, die Basismanöver in Querrichtung für das Ego-Fahrzeug zu bestimmen, sind die Netze *a* bis *h* dafür konzipiert, die Basismanöver in Querrichtung für das jeweilige dynamische Objekt zu erkennen. Im Gegensatz zu den Netzarchitekturen *a*, *b* und *c*, greifen die Netze *d* bis *h* neben den Objektlistensignalen noch auf zusätzliche Informationskombinationen des Ego-Fahrzeugs zurück. Die größte untersuchte Eingangsarchitektur weist demnach Netz *h* auf, dass alle 10 dargestellten Signale verwendet und demnach eine Eingangsschicht mit 10 Neuronen innehat.

Die verschiedenen Eingangsschichtarchitekturen lassen sich mit wiederum mit einer unterschiedlichen Zahl und Konfiguration an Hidden Layer Architekturen kombinieren. Nachfolgend werden die Resultate verschiedener empirisch getesteter Hidden Layer Architekturen vorgestellt:[52]

- Architektur 1: Bestehend aus der Eingangsschicht in Abhängigkeit von *a* bis *i*, einer versteckten LSTM-Schicht mit 256 Einheiten und der Ausgabeschicht mit den drei Neuronen für die drei zu bestimmenden Basismanöver.

- Architektur 2: Wie Architektur 1 aber mit zwei versteckten LSTM-Schichten a 100 Einheiten.

- Architektur 3: Wie Architektur 1 aber mit drei versteckten LSTM-Schichten a 100 Einheiten.

Um die Netze als Manöverklassifikatoren einsetzen zu können, erfolgt zunächst das Training der Netze nach dem Prinzip des überwachten Lernens (siehe auch Anhang A). Für diesen Zweck müssen neben den Trainingsdaten auch die zugehörigen Label-Informationen im Sinne der Output-Klassen (hier also „Fahrstreifenwechsel rechts“, „Fahrstreifenwechsel links“ und „Fahrstreifen folgen/halten“ der Objekte zur Verfügung stehen. Dafür wurde ein Trainingsdatensatz in der Simulation erstellt, der als Annotationen neben der Erzeugung der genannten Signale auch die Fahrstreifenwechselinformationen für das Ego-Fahrzeug und die enthaltenen Verkehrsobjekte anhand der Ground Truth Informationen enthält. Um die Datenbasis für das Training zu vergrößern wurden die in Abschnitt 7.2.1 erzeugten Szenarien verlängert und insgesamt 963 km simuliert. Auf dieser Strecke wurden in Summe 3,4 Mio. Signalpunkte erzeugt und mit den annotierten Klasseninformationen gespeichert. Der so erzeugte Datensatz wurde nach dem üblichen Vorgehen aufgeteilt, sodass 70% der Daten für das Training, und je 15% für Validierung und Test verwendet

[52] Ein Großteil der Kombinationen mit den Architekturen 1 und Architekturen 2 wurden in der Masterthesis (Chen, Jiaxi; 2019) umgesetzt.

wurden. Mit diesem Datensatz wurden alle Netzvarianten trainiert und evaluiert.

In nachfolgender Abbildung sind die Ergebnisse der Netze *a* bis *i* in den jeweiligen vorgestellten Architekturvarianten nach dem Training bei Evaluierung auf den Testdaten dargestellt (Abbildung 7.10):

	Architektur 1			Architektur 2			Architektur 3		
	Links	Halten	Rechts	Links	Halten	Rechts	Links	Halten	Rechts
a	58,2	86,6	42,0	69,0	90,0	89,1	71,3	83,3	91,2
b	86,0	93,8	61,7	91,2	95,8	92,5	93,4	95,6	92,8
c	84,8	94,4	53,7	93,8	95,6	92,2	95,1	95,3	93,7
d	90,5	92,8	84,9	94,6	96,4	94,7	96,8	95,7	95,1
e	88,9	93,4	70,7	96,7	96,7	92,7	95,0	96,0	95,7
f	90,5	92,3	84,2	93,9	96,2	95,7	95,7	96,7	93,5
g	88,9	90,3	79,7	97,3	96,0	95,2	96,5	96,2	93,7
h	90,8	91,7	78,8	94,7	96,2	94,6	95,1	96,1	96,3
i	72,1	96,2	48,9	92,4	98,8	95,3	97,2	98,6	97,3

Abbildung 7.10: Erzielte Genauigkeit (true positive Rate in %) verschiedener Netzarchitekturen für die Manövererkennung bei unterschiedlichen Eingangssignalkonfigurationen *a* bis *i*

Die erzielten Genauigkeiten für die jeweilige Netzarchitektur / Eingangssignalkombination sind aufgeteilt nach den drei Manöverklassen und als true positive Rate angegeben. Die höchsten insgesamt pro Manöverklasse erzielten Werte sind grün markiert, die schlechtesten entsprechend rot.

Die Ergebnisse zeigen, dass über alle Signalkombinationen die tieferen Netze (Architektur 2 und 3) deutlich besser abschneiden, als die Architektur 1 mit nur einer versteckten Schicht. Besonders deutlich wird dies bei den erreichten Genauigkeiten für die Fahrstreifenwechsel links und rechts. Hier können mit den Architekturen 2 und 3 fast durchweg Ergebnisse über 90% bis hin zu 97,3% erzielt werden. Eine Ursache kann sein, dass für die Erkennung der

Spurwechsel im Gegensatz zur Klasse „Fahrstreifen halten“ der zeitliche Verlauf der Signale eine größere Rolle spielt. Da bei tieferen LSTM-Netzen größere Sequenzen durch die Struktur der Netze länger gespeichert werden können, kann dies damit begründet werden.

Hinsichtlich der verschiedenen Eingangssignalkonfiguration zeigt sich in den Ergebnissen ein Unterschied zwischen den Kombinationen *a* bis *c* und *d* bis *h*. Die Kombinationen *d* bis *h* schneiden über alle Architekturen und alle Manöverklassen besser ab. Dies ist plausibel, da in diesen Kombinationen zusätzlich zu den Objektlistensignalen die Signale des Ego-Fahrzeugs genutzt werden, die somit das Ergebnis verbessern. Des Weiteren ist innerhalb der Kombinationen *d* bis *h* kein signifikanter Unterschied in den erzielten Ergebnissen zu beobachten. Dies zeigt, dass die künstlichen neuronalen Netze grundsätzlich nach entsprechendem Training auch auf potentiell unterschiedlich vorliegende Signalkonfigurationen und –kombinationen adaptierbar und einsetzbar sind.

Schlussendlich wird mit der Konfiguration *i* gezeigt, dass auch die Manöver des Ego-Fahrzeugs erkannt werden können. Hier wird eine Genauigkeit von mindestens 97,2% für alle drei Manöver erreicht (Architektur 3).

7.4 Ergebnisse zur Verifikation

Nachdem in den vorigen Abschnitten die Umsetzungen und Methoden vorgestellt wurden, die notwendigen Szenario-Metainformationen aus den aufgezeichneten Daten zu extrahieren und in die Simulation zu überführen, stellt sich die Frage, wie gut die Simulation letztendlich im X-in-the-Loop in der Lage ist, die Ausgangsdaten wieder abzubilden (siehe Abbildung 7.1).

Die Ergebnisse zur Verifikation werden anhand der drei Ebenen (vergleiche Abschnitt 6.4) dargestellt.

7.4.1 Interne Verifikation

Die interne Verifikation dient der Überprüfung des angewandten Modells und der Plausibilisierung der erfassten und im Tensor gespeicherten Werte. Während unrealistische Wertebereiche für Einzeleinträge durch Filterabfragen identifiziert werden können, können auch erfasste Zustandsübergänge auf Plausibilität geprüft werden. Dafür wurden Funktionen implementiert, die für jeden Zustand alle möglichen Folgezustände ermitteln und den gespeicherten Tensor systematisch auf diese untersuchen. Exemplarisch ist nachfolgend die Umsetzung für den Zustand Q_1 dargestellt (Abbildung 7.11). Aus diesem Zustand sind, wie bereits als Beispiel in Abschnitt 6.4.1 erläutert, lediglich die Zustände Q_2 und Q_4 erreichbar.

```
// Überprüfung auf plausible Zustandsübergänge für Zustand Q1

for "jede Objektmatrix i im Tensor T"
    for "jede Spalte s in Objektmatrix i"
        //wenn Zustand Q1, dann sind Folgezustände Q1, Q2 und Q4 zulässig
        if "Zeile Label_State_Long = 1 und Zeile Label_State_Lat = -1"
            Überprüfe, ob in Spalte s+1 einer der folgenden Einträge enthalten ist:
            (Zeile Label_State_Long = 1 und Zeile Label_State_Lat = -1) //erneut Zustand Q1
            oder
            (Zeile Label_State_Long = 0 und Zeile Label_State_Lat = -1) //Zustand Q2
            oder
            (Zeile Label_State_Long = 1 und Zeile Label_State_Lat = 0) //Zustand Q4
        else
            Fehler
        end if
    end for
end for
```

Abbildung 7.11: Pseudocode für die Untersuchung möglicher Folgezustände aus Zustand Q_1 in der Tensorrepräsentation

Analog zu diesem Vorgehen können sowohl sämtliche anderen Zustandsübergänge als auch die Manöverfolgen zwischen den Zuständen überprüft werden.

7.4.2 Verifikation auf der Abstraktionsebene

Durch die Verifikation auf der Abstraktionsebene kann abgeglichen werden, mit welcher Genauigkeit das ursprüngliche Szenario durch das Metamodell wiederhergestellt werden kann. Dazu werden die Tensorrepräsentationen aus dem originalen Szenario mit einem gebildeten Tensor aus der Resimulation verglichen (siehe Abschnitt 6.4.2).

Für sämtliche Testszenarien stimmte die Dimensionalität des erzeugten Replay-Tensors mit dem Originaltensor überein. Im nachfolgenden Schritt wurden die nominalskalierten Einträge (Codierungen der Zustände und Basismanöver) überprüft. Übergreifend über den Testdatensatz wurde für das Ego-Fahrzeug eine Übereinstimmung von über 99% bei den durchgeführten Manövern erreicht. Dies setzt sich zusammen aus einer Quote von 100% für die Basismanöver in lateraler Richtung und 99,2% für die Basismanöver in longitudinaler Richtung. Die entsprechende Fehlerquote von 0,8% für die Longitudinalmanöver entstand überwiegend bei den (seltenen) Szenarien, bei denen eine „Schwingung“ um die definierten Schwellenwerte (schwarz gestrichelte Linien in Abbildung 7.12) für die Längsbeschleunigung zur Unterscheidung der Manöver zu beobachten war. Dieses verhältnismäßig hochfrequente Wechseln zwischen den Manövern konnte in der Resimulation nur bedingt wiederhergestellt werden (vergleiche insbesondere den Bereich zwischen 10 und 16 Sekunden in Abbildung 7.12). Es ist anzunehmen, dass dieser Effekt durch die Ergänzung einer zeitbasierten Hysteresefunktion um den Schwellenwert für die Bestimmung der Longitudinalmanöver zumindest verringert werden kann.

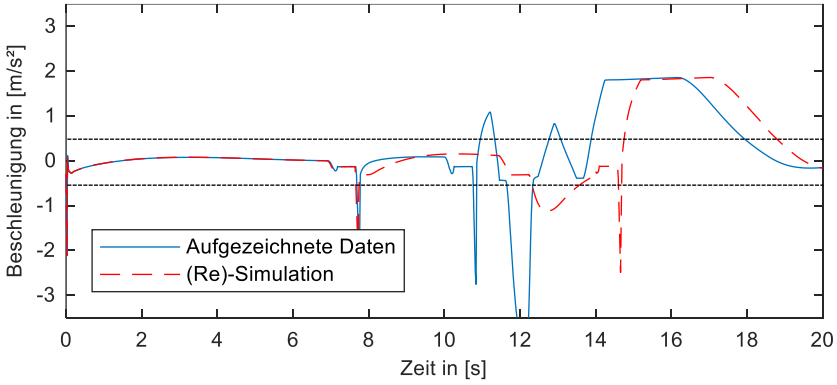


Abbildung 7.12: Beschleunigungsverlauf eines Szenarioausschnitts mit künstlichen hochfrequenten Manöverwechseln und Wiederherstellung in der Simulation

Auch für die Objekte wurden ähnliche Ergebnisse erzielt. Übergreifend über den Testdatensatz wurden für die Objekte eine Übereinstimmung von über 98% bei den durchgeführten Manövern erreicht. Dies setzt sich zusammen aus einer Quote von 99,3% für die lateralen Basismanöver sowie 98,4% für die Basismanöver in longitudinaler Richtung. Die etwas schlechtere Quote für die longitudinalen Manöver kann analog zum zuvor erläuterten Effekt des Ego-Fahrzeugs erklärt werden. Darüber hinaus ist die Fehlerquote für die Manöver bei den Objekten insgesamt etwas höher als beim Ego-Fahrzeug. Ursächlich dafür sind Manöver, die im originalen Datensatz im Grenzsichtbereich der Ego-Fahrzeugsensorik stattfinden. Hier kann es sein, dass zuvor erfasste Manöver bei der Resimulation beispielsweise durch eine leichte Verschiebung in longitudinaler Richtung gerade außerhalb der Sensorreichweite stattfinden und demzufolge nicht mehr erfasst werden.

Die Tensoren zur Repräsentation der Szenarien enthalten über die nominalskalierten Größen für die Manöver und Zustandsbeschreibung auch metrische Größen, die die kinematischen Zusammenhänge quantitativ beschreiben. In Abschnitt 6.4.2 wurde als statistisches Vergleichsmaß für diese Größen der *mittlere absolute prozentuale Fehler* (engl. *MAPE*) vorgestellt. Die Anwendung dieses Maßes über jeweils vollständige Tensoren aus den Testdaten und der Resimulation führte zu stark variierenden Ergebnissen. Bei genauerer Analyse stellte sich jedoch dieses Maß bei Anwendung über die vollständigen Tensoren als ungeeignet heraus, was an folgendem Beispiel erläutert wird (Abbildung 7.13):

In der Closed-Loop Resimulation stellt sich das Verhalten der Objekte entsprechend der hinterlegten Modelle ein. Das bedeutet, dass die konkrete Umsetzung der Manövervorgaben (aus dem Tensor) je nach Modell beziehungsweise Parametrierung zu Unterschieden innerhalb des Manövers führen kann. Dies wiederum kann zu deutlichen Verschiebungen in den Manöverübergängen führen. In Abbildung 7.13 ist als Ausschnitt ein einfacher Beschleunigungsvorgang von 100 km/h auf ca. 110 km/h dargestellt. Es werden sowohl Anfangs-, als auch Endgeschwindigkeit in der Resimulation korrekt abgebildet. Der Verlauf selbst unterscheidet sich lediglich darin, dass zu Beginn minimal

stärker beschleunigt wird und zum Ende hin etwas weniger als in den aufgezeichneten Daten. Für die Manöverextraktion führt dies jedoch in diesem Fall zu einem deutlichen Unterschied in der Repräsentation im Tensor. In der Resimulation wird der Schwellenwert von 0,5 m/s² deutlich früher erreicht und damit das Manöver „Zielgeschwindigkeit erhöhen“ bereits nach ca. 21,8 s begonnen. Im Tensor zu den originalen Daten wurde dieses Manöver erst knapp 2 s später definiert und mit einer höheren Startgeschwindigkeit v_s bedatet. Die Anwendung von *MAPE* für die Tensoreinträge $v_{s,data}$, $v_{s,resim}$, $t_{s,data}$ und $t_{s,resim}$ ergibt einen Fehler von über 5%, trotz der guten Übereinstimmung im Geschwindigkeitsverlauf.

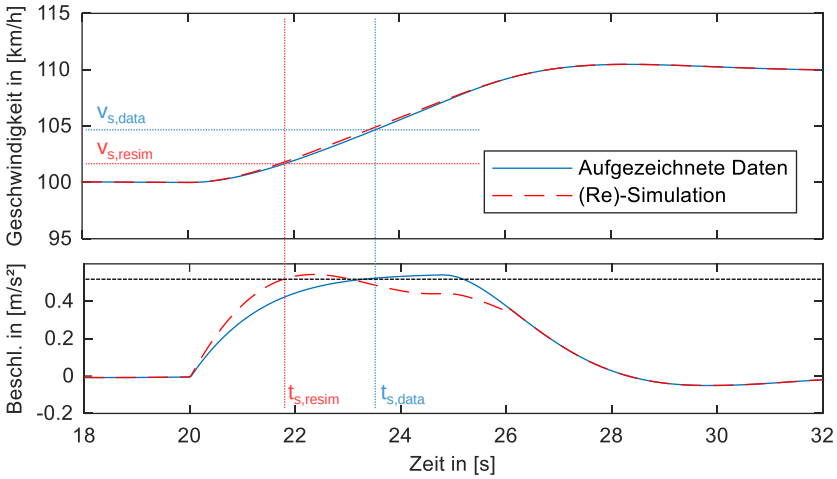


Abbildung 7.13: Verschiebungen in der Erfassung der Manöverwechsel

Daraus lässt sich schließen, dass die metrischen Größen im Tensor im Gegensatz zu den nominalskalierten Größen zur Beschreibung der Manöver und Zustände nur eine bedingte Aussagekraft hinsichtlich der Repräsentationsfähigkeit des Tensors aufweisen. Diese Einträge dienen vornehmlich zur eindeutigen Beschreibung und Vorgabe des kinematischen Verhaltens wie in

den Abschnitten 6.1.2 und 6.1.3 erläutert, wobei Abweichungen in diesen Werten wie zuvor dargestellt nicht unbedingt im gleichen Maße für ein schlechteres Abstraktionsverhalten des Tensors sprechen müssen.

7.4.3 Verifikation auf der Datenebene

Die Verifikation auf der Datenebene stellt die letzte Stufe der Verifikation dar, die gleichzeitig die höchste Präzision bei der Beurteilung der Wiedergabegenauigkeit des Originalszenarios durch die Simulation erlaubt. Um das Verhalten der Resimulation zu beurteilen, wurden für die Verifikation sechs Kennzahlen gebildet, die auf folgenden Größen in den Originaldaten und der Resimulation beruhen:

- Gefahrene Distanz in longitudinaler Richtung über die Zeit des Ego-Fahrzeugs (abgekürzt: Long Ego)
- Gefahrene Distanz in longitudinaler Richtung über die Zeit der dynamischen Objekte (Long Objekt)
- Seitliche Querabweichung zu einer Referenzlinie der Fahrbahn über die Zeit des Ego-Fahrzeugs (Lateral Ego)
- Seitliche Querabweichung zu einer Referenzlinie der Fahrbahn über die Zeit der dynamischen Objekte (Lateral Objekt)
- Geschwindigkeitsverlauf über die Zeit des Ego-Fahrzeugs (Geschw. Ego)
- Geschwindigkeitsverlauf über die Zeit der dynamischen Objekte (Geschw. Objekte)

Diese Größen wurden sowohl in den Originaldaten als auch in der Resimulation mit einer Frequenz von 50 Hz aufgezeichnet. Für diese Analyse wurden zweimal zehn repräsentative Szenarien untersucht, die typische Überholvorgänge auf der Autobahn darstellen. In den ersten zehn Szenarien überholt das Ego-Fahrzeug ein Objekt mit verschiedenen Differenzgeschwindigkeiten,

während in den zweiten zehn Szenarien entsprechend das Ego-Fahrzeug durch ein Objekt überholt wird. In allen Szenarien finden also verschiedene Fahrstreifenwechsel, sowie Beschleunigungs- und Abbremsmanöver statt, je nach Konstellation. Alle Szenarien weisen eine Länge von 25 Sekunden auf. Dies entspricht also für jede Größe einer Menge von 1250 aufgezeichneten Datenpunkten pro Szenario.

Auf jeder dieser Größen wurde der *MAPE* als Kenngröße gebildet. In Abbildung 7.14 sind die so ermittelten Kenngrößen als Durchschnittswert über die jeweils zehn Szenarien dargestellt. Die gestrichelte Darstellung entspricht den Szenarien mit dem überholenden Objekt, während bei der Darstellung mit der durchgezogenen Linie das Ego-Fahrzeug überholt. Für alle Größen ist zu sehen, dass der durchschnittliche Fehler unter 1% liegt, in den meisten Fällen sogar unter 0,5%.

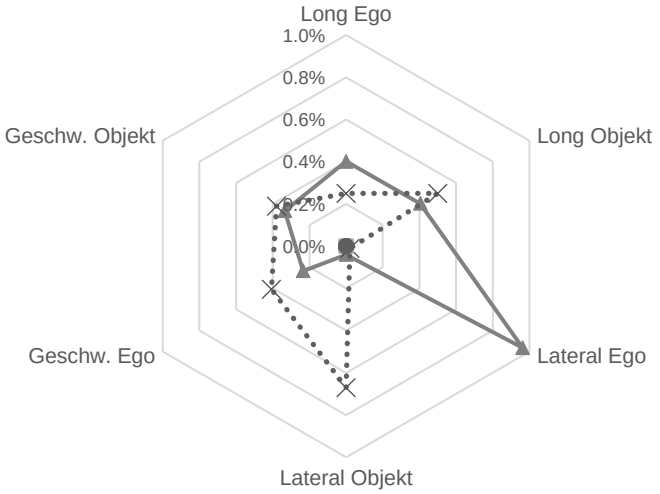


Abbildung 7.14: Durchschnittlicher Fehler über die Kenngrößen bei den Szenarien "Ego-Fahrzeug überholt" (durchgezogene Linie) und "Objekt überholt Ego-Fahrzeug" (gestrichelte Linie)

Lediglich die beiden Kennzahlen für die seitliche Querabweichung stellen leichte Ausreißer dar. Der größere Fehler kommt insbesondere dann zustande,

wenn in den Szenarien mehrere Fahrstreifenwechsel gefahren werden. In den Szenarien, in denen durch Ego-Fahrzeug oder Objekte keine Fahrstreifenwechsel gefahren werden, liegt der durchschnittliche Fehler jeweils im Bereich von absolut betrachtet weniger als 1 cm in Querrichtung. Eine Detailanalyse zeigt, dass der Fehler bei den Fahrstreifenwechseln insbesondere dann entsteht, wenn eine leichte Phasenverschiebung im Manöververlauf gegeben ist. Da die Messung des Fehlers stets zur gleichen Zeitreferenz durchgeführt wird, führt selbst ein exakt gleicher Trajektorienverlauf in der Resimulation im Vergleich zu den Originaldaten zu einem Fehler, wenn dieser zeitlich verschoben durchgeführt wird. Ähnliches gilt auch für Beschleunigungs- und Abbremsmanöver und ihrer Auswirkung auf die longitudinalen Kenngrößen.

Insgesamt lässt sich jedoch festhalten, dass die Closed-Loop Resimulation die Originaldaten mit einer übergreifend hohen Genauigkeit wiederherstellt. Zum gleichen Ergebnis führt auch ein Abgleich der insgesamt gefahrenen Distanz pro Szenario zwischen den Originaldaten und der Resimulation. Die durchschnittlich vom Ego-Fahrzeug gefahrene Distanz pro Szenario beträgt knapp 1.000 m. Der maximal gemessene Unterschied am Ende eines jeweiligen Szenarios zwischen Originaldaten und Resimulation betrug 4,5 m, die durchschnittliche Abweichung dabei lediglich 0,23%, was ungefähr gerademal einer halben Fahrzeuglänge entspricht.

In Abschnitt 6.1.1 wurde mit der Präzisierung des Manöverbegriffs und der Einführung des *Basismanövers* ein für die Verfolgung der Fragestellung (siehe Abschnitt 4.2) notwendiger Kompromiss im Abstraktionsgrad der Szenariobeschreibung geschaffen. Die Analyse im Rahmen der Verifikation zeigte jedoch, dass bestimmte Phänomene eigentlich *atomarer Manöver* aus den Originaldaten in der Closed-Loop Resimulation nachgebildet werden können, obwohl diese auf dem Abstraktionslevel der Basismanöver und dem Meta-Modell nicht beschrieben werden. Voraussetzung hierfür ist eine ausreichend detaillierte Parametrierung des Fahrzeugmodells in der Simulationsumgebung entsprechend des Verhaltens mit dem die Originaldaten erzeugt wurden.

Ein Beispiel für diese „Wiederherstellung“ *atomarer* Manöverbestandteile (vgl. auch Abbildung 6.1) zeigt sich in der Analyse eines einfachen Beschleunigungsvorgangs. Das Basismanöver „Zielgeschwindigkeit erhöhen“ ist durch

eine Anfangs- und Endgeschwindigkeit sowie eine Zeitspanne gekennzeichnet. Der tatsächliche Geschwindigkeitsverlauf innerhalb dieses Manövers lässt sich durch ein kinematisches Approximationsmodell beispielsweise polynomial schätzen (siehe zum Beispiel nach Akcelik und Biggs [151], wie in Abschnitt 6.1.2 dargestellt). Der tatsächliche Geschwindigkeitsverlauf kann jedoch je nach Getriebe- und Antriebsstrangkonfiguration durch deutliche Knicke im Kurvenverlauf geprägt sein, die durch den Momenteneinbruch in den Schaltvorgängen ausgelöst werden (Abbildung 7.15). Die Abstraktion dieser Informationen auf der atomaren Manöverebene sind nicht im hier verwendeten Szenario-Metamodell enthalten. Dennoch können sich diese Effekte in der Resimulation originalgetreu wieder einstellen, die entsprechende Fahrzeugmodellparametrierung vorausgesetzt.

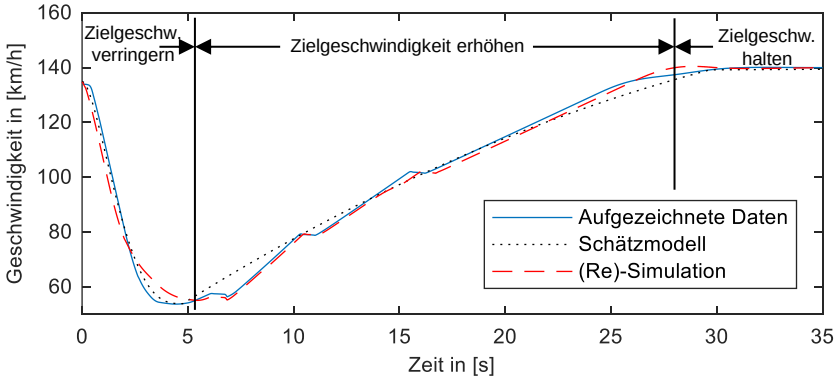


Abbildung 7.15: Reproduktion von atomaren Manövereigenschaften in der Resimulation (erstmalig veröffentlicht in [190])

7.5 Zusammenfassung der Ergebnisse

Die in diesem Kapitel vorgestellte Evaluierung an simulierten, aufgezeichneten Ausgangsdaten zeigt, dass das entwickelte Szenario-Metamodell im Sinne des in Kapitel 5 vorgestellten Prozesses anwendbar ist. Diese These stützt sich

auf folgende Teilergebnisse, die in den vorherigen Abschnitten ausführlich dargestellt wurden:

- Das Szenario-Metamodell erlaubt die Abstraktion eines (aufgezeichneten) Szenarios im Sinne der enthaltenen Verkehrsteilnehmer und des Ego-Fahrzeugs und bietet dabei ein vergleichsweise effizientes Speicherformat.

- Das Szenario-Metamodell bildet das Verhalten der dynamischen Verkehrsteilnehmer und des Ego-Fahrzeugs mit einer hohen Genauigkeit ab. Insbesondere auf der Datenebene wurde für alle betrachteten Signale in der Resimulation stets ein durchschnittlicher Fehler von unter 1% erreicht.

- Das Schneiden des repräsentierenden Tensors einer Aufzeichnung ermöglicht das Herauslösen (und den Vergleich) der enthaltenen Szenarien anhand der Folgen von Szenenwechseln – unabhängig von der zeitlichen Länge der jeweiligen Szenarien.

- Höherfrequente Manöverwechsel können zu abweichenden Tensorrepräsentationen führen. Für diese Fälle stößt der hier gewählte Abstraktionsgrad mit den Basismanövern an seine Grenzen. Dennoch können auch durch diesen Abstraktionsgrad implizit einzelne Phänomene wieder abgebildet werden, die gemäß der hier entwickelten Manövereinteilung eigentlich eher den atomaren Manövern zuzuordnen sind (siehe Beispiel Verzögerungen durch Schaltvorgänge).

- Die Extraktion der für die Befüllung des Metamodells erforderlichen Daten kann aus typischen Sensoraufzeichnungen des Ego-Fahrzeugs abgeleitet werden. Mit den Zeitreihenanalyseverfahren Dynamic Time Warping und künstlichen neuronalen Netzen wurden zwei vielversprechende Methoden angewandt, die in der Lage sind, auf Sensor-Objektlistendaten auch unterschiedlicher Konfigurationen eingesetzt werden zu können und die enthaltenen Manöverbestandteile der Objekte zu bestimmen. Die Erstellung der Objektlisteninformationen aus beispielsweise Sensorrohdaten sind nicht Bestandteil.

8 Evaluierung an Realdatensätzen

Der in Abschnitt 5.3 dargestellte Prozess zur Extraktion der Szenarien stützt sich, entsprechend der Leitfrage aus Abschnitt 4.2, auf aufgezeichnete Daten aus realen Situationen.

Die Auswahl geeigneter Datensätze für diesen Anwendungsfall wird durch drei Anforderungen bedingt, die nachfolgend als Kriterium a), b) und c) bezeichnet werden:

a) Um für das in den Kapiteln 6 und 7 entwickelte und für synthetische Eingangsdaten bereits umgesetzte Modell anwendbar zu sein, müssen die Daten von Autobahnabschnitten oder zumindest von autobahn-ähnlichen Umgebungen stammen. Als autobahn-ähnlich werden mehrspurige Straßen ohne Kreuzungen und mit geringen Kurvenradien angenommen.

b) Die Ground Truth Positionsdaten des Ego-Fahrzeugs und der Verkehrsobjekte müssen (als Referenz) enthalten oder zumindest aus den Daten ableitbar sein, um die Güte der hier verwendeten Verfahren beurteilen zu können.

c) Um vollwertige Szenarien beschreiben und extrahieren zu können, müssen die Daten in zeitlich zusammenhängenden Sequenzen über mindestens 5-10 Sekunden vorliegen. Sammlungen oder Sets von Einzelbildern oder annotierten Einzelframes sind für diese Anwendung daher nicht geeignet.

Aktuell stehen im Forschungs- und Industrieumfeld mehrere Datensätze zur Verfügung, die jedoch unterschiedliche Eigenschaften hinsichtlich der enthaltenen Daten oder Aufzeichnungstechniken aufweisen. Darüber hinaus sind die Datensätze mit unterschiedlichen Meta-Informationen und Daten-Annotation ausgestattet, die jeweils unterschiedliche Anwendungsfelder ermöglichen. In Tabelle 11 sind Stand Anfang 2020 eine Reihe bekannter und frei verfügbarer

Datensätze mit ihren wichtigsten Eigenschaften sowie den Bewertungen hinsichtlich der Kriterien a), b) und c) zusammengetragen.

			Kriterium		
Datensatz	**Eigenschaften**	**Meta-Informationen**	**a)**	**b)**	**c)**
Audi A2D2[53]	6x Kamera, 5x Lidar + Bus-Daten Umfang: 40.000 Bilder	3D Bounding Boxen Semantische Segmentierung	-	+	o
BDD 100k[54]	Kamera, GPS und IMU Daten Umfang: 1.100 h	100.000 Bilder mit 2D Bounding Boxen, 10.000 Bilder mit semantischer Segmentierung	+	-	+
Citys-capes[55]	Hauptsächlich Stereo-Kameradaten, Videosequenzen Umfang: 20.000 Bilder	Label für jedes 20. Bild, Objektklassen und semantische Segmentierung	-	-	+
comma.ai[56]	GPS/IMU Daten, Kamera, CAN-Daten + Thermometer Umfang: 33 h	Überwiegend Rohdaten	+	-	+
highD[57]	Kameraaufzeichnungen von Drohnen auf Autobahnen Umfang: 16,5 h	Objektinformationen und Trajektorien	+	+	+
KITTI MOTS[58]	GPS/IMU Daten, 1x Lidar, 2x Graustufenkamera, 2x Farbkamera Umfang: 10.000 Bilder in 50 Sequenzen	Label für 4 Objektklassen 2D und 3D Bounding Boxen mit Tracking IDs	-	o	+

[53] https://www.audi-electronics-venture.de/aev/web/de/driving-dataset/download.html
[54] https://bdd-data.berkeley.edu/
[55] https://www.cityscapes-dataset.com/
[56] https://github.com/commaai/comma2k19
[57] https://www.highd-dataset.com/
[58] http://www.cvlibs.net/datasets/kitti/

Lyft Level 5[59]	3x Lidar, 7x Kamera + HD-Karte Umfang: >55.000 Bilder in Sequenzen	3D Bounding Boxen Semantische Informationen in der Karte	o	+	+
Mapil-lary[60]	Sammlung von 1 Mrd. Bildern verschiedener Kameras	Objektklassifikationen und Rohdaten	o	-	-
nuScenes[61]	6x Kamera, Lidar, 5x Radar, GPS und IMU-Daten Umfang: 15 h	3D Bounding Boxen	-	+	+
udacity[62]	Kameradaten + Logs zu Position, Gas, Bremse, Lenkwinkel, Geschw. Umfang: 70 min	2D Bounding Boxen	+	-	-
Waymo[63]	Synchronisierte Daten von: 1x Lidar (mid-range), 1x Lidar (short-range), 5x Kamera Umfang: 200.000 Bilder in 2.950 Sequenzen	Label für 4 Objektklassen 2D und 3D Bounding Boxen mit Tracking IDs	-	+	+
inD[64]	Kameraaufzeichnungen von Drohnen auf Kreuzungen Umfang: 10 h	Objektinformationen und Trajektorien	-	+	+

Tabelle 11: Vergleich frei für Forschung verfügbarer Datensätze (Stand Anfang 2020)

[59] https://level5.lyft.com/dataset/
[60] https://www.mapillary.com/
[61] https://www.nuscenes.org/data-collection
[62] https://github.com/udacity/self-driving-car/tree/master/datasets
[63] https://waymo.com/open/
[64] https://www.ind-dataset.com/

Entsprechend der Kriterien wird die Anwendbarkeit des Prozesses aus Abschnitt 5.3 anhand von zwei verschiedenen Datensätzen demonstriert:

- highD-Datensatz (Abschnitt 8.1)
- Lyft Level 5-Datensatz (Abschnitt 8.2).

Bei allen anderen aufgeführten Datensätzen wird jeweils mindestens eins der genannten Kriterien nicht erfüllt.

8.1 highD-Datensatz

Der highD-Datensatz [65] wurde 2018 vom Institut für Kraftfahrzeuge der RWTH Aachen erstellt und enthält 16,5 Stunden aufgezeichnete Verkehrsszenarien von deutschen Autobahnen sechs verschiedener Abschnitte mit ca. 110.000 Fahrzeugen. Insgesamt wurde im Datensatz eine zurückgelegte Distanz von 45.000 km mit über 11.000 enthaltenen Fahrstreifenwechseln erfasst [191]. Die Besonderheit des highD-Datensatz im Gegensatz zu anderen verfügbaren Datensätzen liegt darin, dass die Aufzeichnungen nicht von einer typischen Fahrzeugsensorik stammen, sondern aus Auswertungen von Drohnenkameras, die über den jeweiligen Streckenabschnitten platziert wurden. Die Streckenabschnitte im Datensatz weisen eine Länge von ca. 420 m auf und stellen jeweils Autobahngeraden mit zwei bis vier Fahrstreifen in beiden Fahrtrichtungen dar (vgl. auch Abbildung 8.1).

[65] Verfügbar unter: http://www.highD-dataset.com

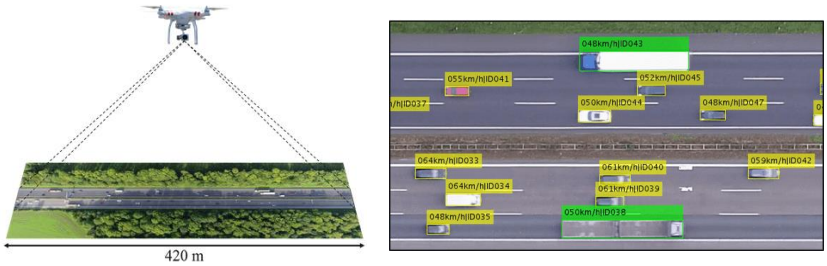


Abbildung 8.1: Prinzip der Erfassung der Daten (links) und der Datenauswertung (rechts) aus dem highD-Datensatz[66]

Aus den Auswertungen der Drohnendaten stehen im Datensatz folgende Informationen zu den Streckenabschnitten und den enthaltenen Verkehrsobjekten als .csv-Daten zur Verfügung:

Größe im Datensatz	**Beschreibung**	**Einheit**
frame	Aktuelle Bildnummer	[-]
id	Kennung der Trajektorie	[-]
x	x-Position der oberen linken Ecke der Fahrzeug Bounding-Box im zugehörigen Bild	[m]
y	y-Position der oberen linken Ecke der Fahrzeug Bounding-Box im zugehörigen Bild	[m]
width	Länge des Fahrzeugs	[m]
height	Breite des Fahrzeugs	[m]
initialFrame	Erste Bildnummer an der das getrackte Fahrzeug sichtbar ist	[-]
finalFrame	Letzte Bildnummer an der das getrackte Fahrzeug sichtbar ist	[-]
class	Fahrzeugklasse des getrackten Objekts (PKW oder Truck)	[-]

[66] Quelle: https://www.highd-dataset.com/#about

drivingDirection	Flag für die Fahrtrichtung (1 für rechts nach links d.h. obere Fahrstreifen, 2 für links nach rechts d.h. untere Fahrstreifen)	[-]

Tabelle 12: Verwendete Informationen aus dem highD-Datensatz

Darüber hinaus enthält der Datensatz weitere (Meta-)Informationen, wie die Geschwindigkeiten oder Beschleunigungen der getrackten Objekte, sowie abgeleitete Kenngrößen wie die Time-to-Collision, die hier nicht weiter betrachtet werden.

8.1.1 Umsetzung

Um den Datensatz gemäß des Prozesses aus Abschnitt 5.3 in eine X-in-the-Loop Umgebung überführen zu können (vgl. Abbildung 5.7), werden die Daten zunächst als reines Replay in der Simulation abbildbar gemacht. Hierfür sind mehrere Teilschritte notwendig, die nachfolgend erläutert werden[67]:

1. Modellierung und Abbildung des Straßenverlaufs aus der jeweiligen Aufzeichnung im Datensatz

2. Generierung passender Objekte zu den jeweiligen aufgezeichneten Objektdimensionen und –klassen

3. Transformation der Positionsdaten der Objekte in die Simulation und Generierung entsprechender Trajektorien

4. Erzeugung des Replays für verschiedene (fiktive) Ego-Fahrzeuge durch Durchführung der Simulation nach iterativer Selektion der Objekte als Ego-Fahrzeug

Schritt 1: Die insgesamt 60 Aufzeichnungen im highD-Datensatz wurden an sechs verschiedenen Orten durchgeführt. Dementsprechend müssen die sechs

[67] Die Implementierung des Replays der highD-Daten in CarMaker wurde in der Bachelorthesis (Sauerer, Fabian; 2020) durchgeführt.

verschiedenen Autobahnabschnitte in der Simulationsumgebung modelliert werden. Aus den Meta-Daten der Aufzeichnungen werden folgende Informationen verwendet:

- Anzahl und Breite der Fahrstreifen in beide Richtungen
- Position der Fahrbahnmarkierungen
- Länge des betrachteten Abschnitts

Aus diesen Informationen werden in der Simulationsumgebung entsprechende Geradenabschnitte mit Fahrstreifen erstellt. Zu Steigung, Wölbung und Krümmung der Fahrbahn liegen im Datensatz keine Informationen vor und werden dementsprechend nicht betrachtet beziehungsweise zu 0 gesetzt.

Schritt 2: Im highD-Datensatz liegen zu jedem aufgezeichneten Fahrzeug die Maße für die Länge und Breite, sowie die Fahrzeugklasse (PKW oder LKW) vor. Informationen über die Höhe der Objekte sind nicht enthalten. Jedes detektierte Fahrzeug im Datensatz wird getrackt und einer ID zugewiesen. Für das Replay wird in der Simulationsumgebung zu jeder ID ein Verkehrsobjekt erzeugt. In CarMaker steht eine umfangreiche Bibliothek von mehreren 100 3D-Modellen für Verkehrsobjekte zur Verfügung. Um ein realistisches Abbild in der Simulation zu erzeugen, wird für jede ID im Datensatz das 3D-Modell aus der Bibliothek ausgewählt, welches hinsichtlich der Differenz in den Maßen für die Länge und Breite die geringste Abweichung zu dem getrackten Objekt aufweist. Durch die Auswahl des 3D-Modells in CarMaker wird automatisch eine entsprechende Höhe für das jeweilige Objekt angenommen.

Schritt 3: Im highD-Datensatz liegen frameweise die Positionsdaten zu jedem sichtbaren Objekt vor. Die Positionsdaten müssen entsprechend auf die erzeugten Objekte in der Simulationsumgebung übertragen werden.

In den Daten werden die Positionspunkte als x- und y-Koordinaten angegeben und markieren jeweils aus Sicht der Drohne den linken oberen Eckpunkt des detektierten Objekts. In der Simulationsumgebung wird hingegen der mittige Punkt am Fahrzeugende als Referenzpunkt angenommen (siehe Markierungen

in Abbildung 8.2). Auch sind die Koordinatensysteme in Orientierung und Ursprung verschieden. Daher wird jeder Positionspunkt zu jedem Objekt entsprechend transformiert, um eine äquivalente Darstellung in der Simulation zu erreichen. Im Datensatz stehen die Positionspunkte mit der Aufzeichnungsfrequenz der Drohnenkamera zur Verfügung. Für die Realisierung der Trajektorien in der Simulation werden die Positionspunkte passend zur Simulationsschrittweite von 1 ms interpoliert und über die Zeit dargestellt.

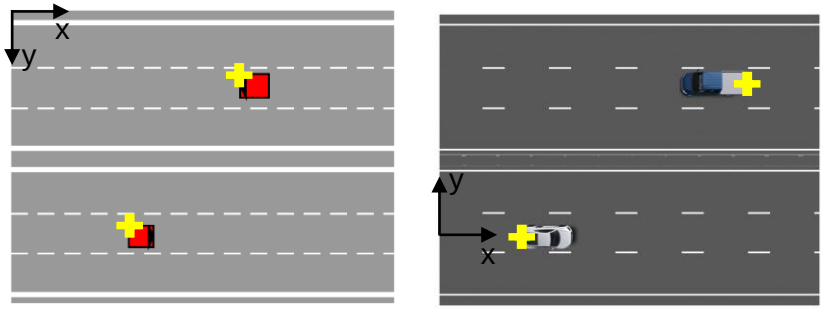


Abbildung 8.2: Referenzpunkte (gelb) für Objekte im highD-Datensatz (links) und in der Simulation (rechts)

Schritt 4: Im highD-Datensatz werden die Daten von einer Drohne aufgezeichnet. Im Gegensatz zu anderen Datensätzen, bei denen die Aufzeichnung aus der Fahrzeugsensorik stammen, liegen damit die Daten aus einer für den jeweiligen Abschnitt globalen Perspektive vor. Die hier entwickelte Methodik zielt auf Daten ab, die aus einer Ego-Fahrzeug Perspektive gewonnen werden. In der Simulationsumgebung steht mit dem Ego-Fahrzeug ein Objekt zur Verfügung, das wiederum mit Sensorik ausgestattet werden kann, um Szene und Szenerie aus seiner Perspektive zu erfassen (vgl. auch Abschnitt 7.2.1). Um die Daten aus dem highD-Datensatz und die Trajektorien für die Ego-Fahrzeug Perspektive nutzbar zu machen, wird iterativ jedes Objekt im Datensatz in der Simulationsumgebung einmal als Ego-Fahrzeug mit Sensorik deklariert und die Simulation aufgezeichnet. Lediglich für Objekte deren Aufzeichnungszeitspannen weniger als 2 s betragen, wird dieser Schritt übersprungen, da nur resultierende Szenarien von mindestens dieser Zeitdauer in Betracht gezogen

werden. Schematisch ist dies in nachfolgender Abbildung dargestellt. Zu jedem Objekt der Aufzeichnung wird ein Replay durchgeführt, außer für das Objekt links am Bildrand. Dieses erfüllt die oben genannte Bedingung für die Dauer des Szenarios nicht und wird daher nicht als Ego-Fahrzeug dargestellt (Abbildung 8.3).

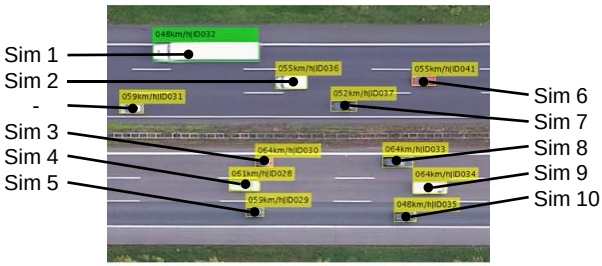


Abbildung 8.3: Erzeugung von Ego-Fahrzeug Replay zu den aufgezeichneten Objekten

Insgesamt werden nach diesem Vorgehen über 100.000 Replay-Simulationen mit den zugehörigen Signalen erzeugt. Die damit generierte Gesamtdauer an Daten aus Ego-Perspektive beträgt ca. 350h.

Mit der Erzeugung der Daten als Replay-Simulationen werden sie direkt für den in Abbildung 5.7 dargestellten Prozess für den Schritt der „Extraktion der Szenarien (Z2)“ anwendbar. Die Vorgehensweise von den originalen highD-Datenaufzeichnungen bis hin zur vollständigen Closed-Loop-Simulation gestaltet sich wie folgt (Abbildung 8.4). Die vier Teilschritte zur Generierung der Replay-Simulation sind grün dargestellt:

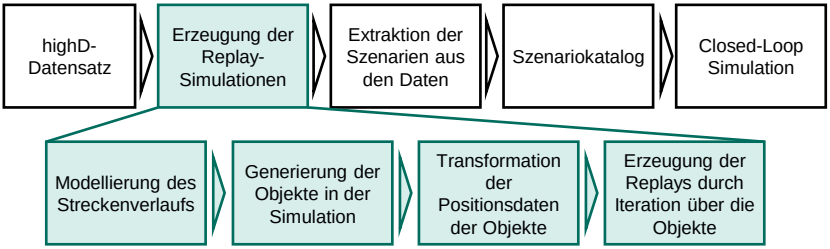


Abbildung 8.4: Schritte der Überführung des highD-Datensatzes in die Closed-Loop Simulation

Nach der wie zuvor beschriebenen Umwandlung in die Replay-Simulationen (ersten beiden Phasen in Abbildung 8.4), können die weiteren Schritte zur Extraktion der Szenarien, der Bildung des Szenariokataloges und die daraus erstellten Closed-Loop Simulationen durchgeführt werden. Der Workflow erfolgt ab dem Schritt der Szenarioextraktion analog zu dem in Kapitel 7 dargestellten Vorgehen mit synthetischen Eingangsdaten.

8.1.2 Ergebnisse

Nach dem zuvor dargestellten Vorgehen wurde zunächst aus dem Datensatz eine Replay-Simulation erstellt. Der Vorteil liegt darin, dass für die anschließende Extraktion der Daten die Ground Truth Information aus der Simulation herangezogen werden kann. Unter der Annahme, dass die Abbildung der Trajektorien der Objekte aus dem Datensatz korrekt in der Simulation wiedergegeben wird, bildet diese das Verhalten der Objekte im Datensatz 1:1 ab. Demzufolge wird das Vorgehen für die Verifikation anwendbar (vgl. „Z4“ in Abbildung 5.7, sowie die Abschnitte 6.4 und 7.4), mit dem Unterschied, dass die dort verwendeten Eingangsdaten durch die Replay-Simulation des highD-Datensatzes ersetzt werden. Gegenüber den zuvor künstlich generierten Szenarien wird das Tensormodell nun auf Anwendbarkeit auf reale Fahrsituationen überprüft.

Zur Beurteilung wurden die sechs bereits in Abschnitt 7.4.3 eingeführten Kenngrößen herangezogen. Damit ist darüber hinaus die Vergleichbarkeit der

Anwendung des Tensormodells sowohl auf synthetische als auch die realen Eingangsdaten gewährleistet.

Die zuvor generierten Replay-Simulationen aus dem highD-Datensatz entsprechen bereits einer gefahrenen Gesamtstrecke für das Ego-Fahrzeug von über 20.000 km. Um den Aufwand für die Analyse zu verringern, wurden aus den über 100.000 generierten Replay-Sequenzen zufällig 500 für die Untersuchung ausgewählt. Die einzelnen Replay-Sequenzen wiesen, repräsentativ für die Verteilung der Sequenzen im vollständigen Datensatz, eine Länge von 2,5 s bis ca. 15 s auf. Aus diesen Replay-Sequenzen wurden nach dem Vorgehen aus den vorigen Kapiteln die enthaltenen Szenarien extrahiert, die Tensorrepräsentation gebildet und die Closed-Loop Simulation der Ursprungsszenarien durchgeführt (Abbildung 8.5). Die nachfolgend präsentierten Ergebnisse beziehen sich auf die Verifikation auf der Datenebene (siehe Abschnitt 7.4.3).

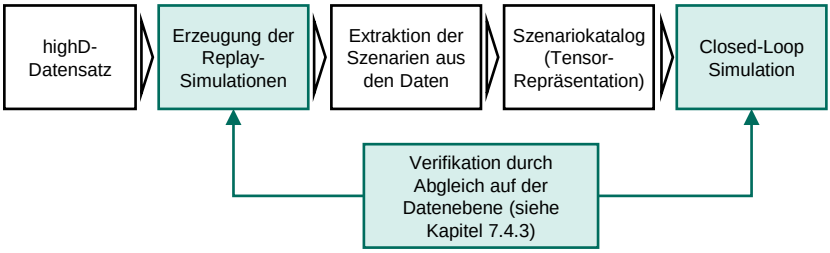


Abbildung 8.5: Verifikation der Closed-Loop Simulation der highD-Datensatz Szenarien

Über die sechs verwendeten Kenngrößen wurde der *MAPE* (vgl. Abschnitt 6.4.2) als durchschnittlicher Fehler über alle 500 Sequenzen berechnet. Analog zu der Darstellung mit rein synthetischen Eingangsdaten ergibt sich folgendes Bild (Abbildung 8.6). Sämtliche Kenngrößen liegen in Größenordnungen, die auch mit synthetischen Eingangsdaten erzielt werden konnten. Für fünf von sechs Kenngrößen wurden sogar bessere Werte erzielt.

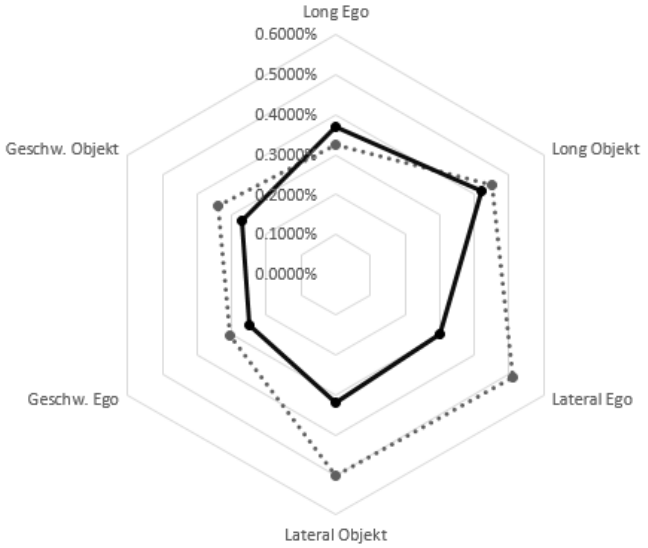


Abbildung 8.6: Durchschnittlicher Fehler über die Kenngrößen aus Abschnitt 7.4.3. für die Closed-Loop Simulationen aus highD-Datensatz (durchgezogene Linie) und simulierten Eingangsdaten aus Kapitel 7 (gestrichelte Linie)

Im Einzelnen betrachtet bedeutet dies:

- Seitliche Querabweichung zu einer Referenzlinie der Fahrbahn über die Zeit des Ego-Fahrzeugs (abgekürzt: Lateral Ego):

  Bei dieser Größe wurden bessere Ergebnisse als mit den synthetischen Eingangsdaten erzielt. Dies kommt dadurch zustande, dass bei den synthetischen Eingangsdaten Szenarien verwendet wurden, die viele Fahrstreifenwechsel-Manöver beinhalten. Diese wirken sich im Vergleich zu konstanter Fahrt auf dem Fahrstreifen negativ auf die Abbildungsgenauigkeit in der Closed-Loop Simulation aus (vgl. Kapitel 7). Im highD-Datensatz ist der Anteil der Fahrstreifenwechselmanöver relativ betrachtet geringer. Demzufolge wird durchschnittlich eine höhere Genauigkeit erzielt.

- Seitliche Querabweichung zu einer Referenzlinie der Fahrbahn über die Zeit der dynamischen Objekte (Lateral Objekt):

  Auch bei dieser Größe wurden für die Closed-Loop Simulationen der highD-Datensatz Eingangsdaten deutlich bessere Ergebnisse als mit synthetischen Eingangsdaten erzielt. Dieses Ergebnis ist analog zu der Ursache für *Lateral Ego* zu bewerten.

- Gefahrene Distanz in longitudinaler Richtung über die Zeit des Ego-Fahrzeugs (Long Ego):

  Diese Größe schneidet bei Verwendung der highD-Daten als einzige Größe schlechter ab (um ca. 0,05%), als unter Nutzung der synthetischen Eingangsdaten. Ursächlich hierfür ist die Parametrierung des Fahrzeugmodells. Für die Replay-Simulation des highD-Datensatz wird über alle enthaltenen Objekte iteriert, wodurch das Ego-Fahrzeug mit jeder Iteration ein anderes Fahrzeug repräsentiert. Die jeweils repräsentierten Fahrzeuge im Datensatz sind unbekannt, wodurch demzufolge auf eine Default-Parametrierung zurückgegriffen werden muss. Während bei der Nutzung der synthetischen Eingangsdaten für die Closed-Loop Simulation exakt die gleichen Fahrzeugparameter verwendet werden können, wirken sich hier die unvermeidbaren Abweichungen in der Parametrierung auf das sich einstellende Fahrverhalten aus.

- Gefahrene Distanz in longitudinaler Richtung über die Zeit der dynamischen Objekte (Long Objekt):

  Die erzielten Ergebnisse dieser Größe für die Closed-Loop Simulation der highD-Daten liegen im Bereich bzw. etwas über den Ergebnissen aus den Simulationen mit synthetischen Eingangsdaten. Die Ursache für die besseren Resultate ist durch die kürzere Länge der Sequenzen bei den highD-Daten begündet. Wie bereits in Kapitel 7 gezeigt, können grundsätzlich längere Szenario-Sequenzen zu schlechteren Ergebnissen in der Closed-Loop Simulation führen, da sich die sich einstellenden Fehler über die Zeit aufsummieren.

- Geschwindigkeitsverlauf über die Zeit des Ego-Fahrzeugs (Geschw. Ego):

  Analog zur Begründung für *Long Objekt*

- Geschwindigkeitsverlauf über die Zeit der dynamischen Objekte (Geschw. Objekte):

  Analog zur Begründung für *Long Objekt*

Insgesamt zeigen die Ergebnisse, dass das in Kapitel 6 entwickelte Modell in der Lage ist, die Szenarien aus dem high-Datensatz sehr gut in der Closed-Loop Simulation abzubilden. Hervorzuheben ist, dass abseits der zuvor vorgestellten Bewertung anhand der Kenngrößen sämtliche in den Daten enthaltenen Fahrstreifenwechsel abstrahiert und in der Closed-Loop Simulation nachgebildet werden konnten. Auch die Betrachtung auf der Ebene der Kenngrößen zeigt, dass das Modell die Daten mit einer durchweg hohen Genauigkeit abbildet. Im Durchschnitt liegt diese dabei sogar höher, als bei Verwendung von synthetischen Eingangsdaten.

8.2 Lyft Level 5-Datensatz

Der Lyft Level 5-Datensatz ist ein Datensatz des gleichnamigen Fahrdienst-Vermittlers und beinhaltet Fahrtaufzeichnungen von Fahrzeugen, die mit Kameras und Lidar-Sensoren ausgerüstet wurden. Im Gegensatz zum highD-Datensatz liegen die Daten im Lyft-Datensatz damit aus einer Ego-Fahrzeug-Perspektive vor. Die Aufzeichnungen entstanden sowohl auf mehrspurigen Schnellstraßen, als auch auf kleineren Nebenstraßen und innerstädtischen Kreuzungsumgebungen. Insgesamt umfasst der Datensatz über 55.000 Frames die mit Annotationen versehen sind und auf jeweils ca. 20-45 Sekunden lange Aufzeichnungssequenzen verteilt sind. Zusätzlich zu den Sensoraufzeichnungen stehen Bounding-Box Informationen zu den Objekten als hinzugefügte Ground Truth Annotationen zur Verfügung, um den Datensatz für die Entwicklung von Objekt-Detektoren und Verfahren des maschinellen Lernens anwendbar zu machen. Im Datensatz liegt die Umfeldaufzeichnung in Form von Daten

aus drei Lidarsensoren, sechs Kameras für 360° Rundumsicht, sowie einer weiteren Kamera speziell für die Ampelerkennung vor. Die Lidardaten sind mit einer Frequenz von 10 Hz aufgezeichnet und sowohl untereinander als auch mit den Kameras zeitlich synchronisiert.

Der Lyft-Datensatz verwendet als Speicherformat das Nuscenes-Format [192], in dem sowohl die Klassen als auch Positionen, Orientierungen sowie weitere beschreibende Attribute für alle Objekte in den Sensordaten gespeichert werden. Neben den annotierten Informationen auf der Ebene einzelner Frames, werden sowohl detektierte als auch Ground Truth Fahrzeuge über den vollständigen sichtbaren Fahrtverlauf mit IDs versehen und stehen damit auch Entwicklungs- und Testumgebungen für Tracking-Algorithmen zur Verfügung. Des Weiteren werden Parameter wie die Orientierung und Position der Sensoren in einem Fahrzeug-Koordinatensystem definiert und gespeichert und können damit als weitere Meta-Informationen genutzt werden.

8.2.1 Umsetzung

Die Vorgehensweise für den Lyft-Datensatz erfolgt analog zu den zuvor gezeigten Schritten im highD-Datensatz (siehe Abbildung 8.4). Zunächst wird eine Replay-Simulation der Daten erzeugt.[68] Der anschließende Ansatz zur Extraktion der Szenarien bis zur Closed-Loop-Simulation entspricht wiederum dem Vorgehen aus Kapitel 7. Das Vorgehen zur Erzeugung der Replay-Simulation ergibt sich wie folgt:

Für die Verarbeitung von Sensorrohdaten wie Kamerabilder oder Lidarpunktwolken existieren eine Vielzahl an Verfahren für die Objektdetektion und –klassifikation. Einige der in Tabelle 11 genannten Datensätze stehen zur Verfügung, um die Leistungsfähigkeit entwickelter Verfahren und Algorithmen untereinander vergleichen zu können. Zu diesen Datensätzen gehört auch der KITTI-Datensatz. Der Datensatz umfasst GPS-, Stereokamera- und Lidardaten

[68] Die Implementierung des Replays der Lyft-Daten in CarMaker wurde in der Bachelorthesis (Jost, Daniel; 2020) durchgeführt.

(360°-Velodyne-Laserscanner) von Fahrten eines mit dieser Sensorik ausgestatteten Fahrzeugs. Auf insgesamt knapp 40 km Streckenlänge wurden die aufgezeichneten Sensordaten und die enthaltenen Objekte mit den Ground Truth Informationen, wie beispielsweise 2D- oder 3D Bounding Boxen annotiert. In den letzten Jahren wurden viele in der Forschung entstandene Verfahren für die Objekterkennung, -klassifikation und –tracking am KITTI-Datensatz evaluiert und können so miteinander verglichen werden.

Ein Objektdetektor, der für Lidardaten einsetzbar ist und im KITTI-Benchmark zum einen überdurchschnittlich gut abschneidet, zum anderen offen verfügbar und damit anwendbar ist, ist der SECOND-Detektor (Sparsely-Embedded-Convolutional-Detection, siehe Abbildung 8.7) [193].

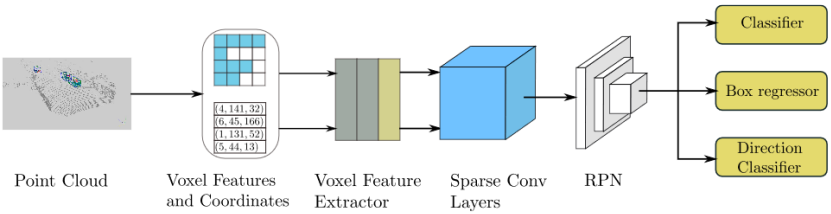


Abbildung 8.7: Struktur des SECOND-Detektors [193]

Der Detektor gliedert sich in drei Abschnitte zur Verarbeitung der Eingabedaten (als Lidarpunktwolken) bis hin zur Ausgabe (Klasse der identifizierten Objekte, 3D-Bounding Box und Orientierung der Objekte).

Zunächst erfolgt die Überführung der Punktwolke in ein Voxel-Gitter. Dazu wird der betrachtete Raum in die sogenannten Voxel unterteilt, die Volumenelemente darstellen, welche neben der Position zusätzlich die Anzahl der in der Punktwolke enthaltenen Punkte beinhalten. Für die Merkmalsextraktion kommt eine Voxel-Feature-Extractor-Schicht zum Einsatz (siehe auch [194]). Nachgelagert wird ein Convolutional-Neural-Network[69] verwendet, welche

[69] Genauer: Spatially-Sparse-Convolutional-Neural-Network (siehe [216])

die dreidimensionalen Daten in eine zweidimensionale Darstellung überführen. Es entsteht in der Vogelperspektive eine Dichte-Feature-Map, die verschiedene Höhenebenen repräsentiert. Schlussendlich werden mit einer Regional-Proposal-Network-Schicht (RPN) aus diesen Dichte-Feature-Maps Detektionen erzeugt, die die Klasse der detektierten Objekte, Position und Orientierung der Bounding-Box beinhalten.

Für die Anwendung auf das Datenformat des Lyft-Datensatzes existiert eine angepasste Version des SECOND-Detektors, die jedoch kein vortrainiertes Modell zur Verfügung stellt.[70]

Für die Übertragung der Lyft-Daten in die Replay-Simulation sind somit folgende Teilschritte notwendig:

1. Trainieren des Detektors mit einem Teil der Lyft-Daten anhand der im Datensatz enthaltenen Ground Truth Annotationen zu den Lidarpunktwolken

2. Anwendung des trainierten Detektors auf den restlichen Datensatz zur Extraktion der Objekte

3. Erstellung und Überführung der Trajektorien für die detektierten Objekte und das Ego-Fahrzeug

4. Erstellung der Umgebung für die Szenarien in der Simulation

Schritt 1: Die Variante des SECOND-Detektors für den Lyft-Datensatz enthält kein vortrainiertes Modell für die Lyft-Daten und ist deshalb nicht direkt einsetzbar. Die direkte Verwendung des vortrainierten Modells aus dem KITTI-Datensatz ist nicht möglich, da sich beide Datensätze hinsichtlich der verwendeten Sensoren unterscheiden und damit Abweichungen bezogen auf Reichweite, Auflösung und somit der Dichte der resultierenden Punktwolke aufweisen. Zunächst wird daher ein Training des Detektors mit den Lyft-Daten

70 Rishabh Agrahari. SECOND for Lyft 3d object detection challenge. https://github.com/pyaf/second.pytorch

durchgeführt. Dazu wird der Datensatz geteilt. 60% der Daten werden als Trainingsdaten verwendet, je 20% für Validierung und Test. Obwohl ein Großteil des Datensatzes nicht auf Autobahnen oder autobahnähnlichen Abschnitten aufgezeichnet wurde (vergleiche Kriterium a) zu Beginn von Kapitel 8), kann dennoch der vollständige Datensatz für das Training des Detektors genutzt werden, da dieser unabhängig von den später betrachteten Szenarien ist.

Nach dem durchgeführten Training steht der Detektor für die Anwendung zur Verfügung.

Schritt 2: Im nächsten Schritt kann der trainierte Detektor eingesetzt werden, um die Objektinformationen aus den Lidarpunktwolken des Datensatzes zu gewinnen. Für jeden Sensorframe liefert der Detektor zu jedem detektierten Objekt folgende Informationen:

- Klasse der detektierten Objekte (z.B. PKW oder LKW)
- Position der Bounding Boxen für jedes detektierte Objekt
- Orientierung der Bounding Boxen für jedes detektierte Objekt

Schritt 3: Der Detektor liefert die zuvor genannten Informationen für die Objekte einzeln für jeden Sensorframe. Um eine plausible Trajektorie der Objekte über die Zeit zu gewinnen, muss ein Tracking der Einzeldetektionen über die Zeit erfolgen. Hierfür wird ein Kalman-Filter-Ansatz verwendet, der die frameweisen Detektionen verbindet und in ein globales Koordinatensystem überführt. Für diese Aufgabenstellung steht mit der in [195] vorgestellten Tracking-Methode eine Implementierung zur Verfügung, die auch hier eingesetzt wird. Als Metrik für die Abweichung zwischen den Prädiktionen des Kalman-Filters und der tatsächlichen Detektionen wird die Mahalanobis-Distanz verwendet (siehe auch Abbildung 8.8).

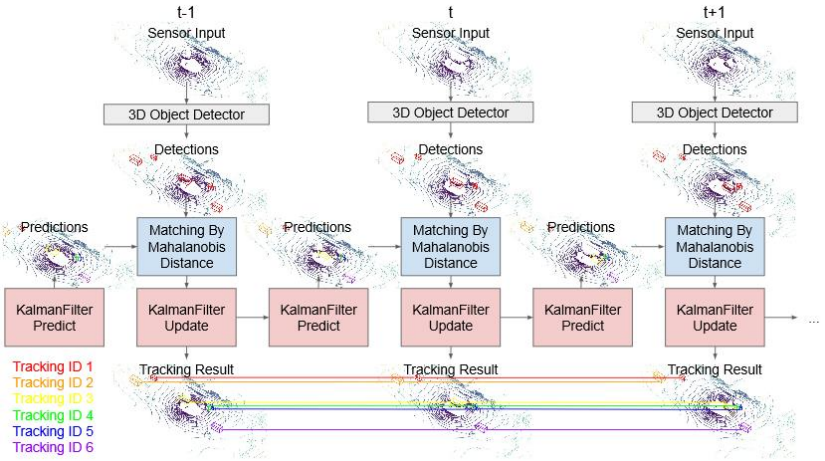


Abbildung 8.8: Architektur-Übersicht des Multi-Objekt-Trackers aus [195]

Die mit dem Tracker ermittelten Positionswerte über die Zeit für die detektierten Objekte können direkt als Trajektorien in die Simulationsumgebung überführt werden. Dort wird für jede Trajektorie ein Verkehrsobjekt der entsprechenden Klasse angelegt, dem die jeweilige Trajektorie übergeben wird.

Für die Trajektorie des Ego-Fahrzeugs werden die ebenfalls im Lyft-Datensatz gespeicherten GPS-Daten herangezogen und verwendet. Um der von der Simulationsumgebung geforderten Zykluszeit von 1 ms gerecht zu werden, wird zwischen den Werten der GPS-Aufzeichnung interpoliert.

Schritt 4: Für eine vollständige Replay-Simulation wird neben dem Verhalten der Verkehrsobjekte und des Ego-Fahrzeugs über die Zeit auch eine Beschreibung für die Umgebung benötigt. Da die Erstellung von (virtuellen) Kartendaten einen Aufgabenbereich für sich darstellt (siehe auch Abschnitt 6.2.3), wird für den hier verfolgten Anwendungsfall mit Fokus auf der Beschreibung des Verhaltens der Verkehrsteilnehmer eine vereinfachte Umgebung in der Simulation herangezogen. Dazu wird ein Fahrbahnstreckenverlauf entlang der

(GPS)-Trajektorie des Ego-Fahrzeugs angenommen. Zusätzlich werden weitere, dazu parallele, Fahrstreifen angelegt. Randbebauungen oder Verkehrszeicheninformationen werden in dieser Umsetzung nicht berücksichtigt.

Zusammengefasst lässt sich analog zur Darstellung für den highD-Datensatz (vgl. Abbildung 8.4) auch die Überführung des Lyft-Datensatzes mit den zuvor gezeigten Schritten (grün) in den Workflow zur Erzeugung von Closed-Loop Simulationen abbilden (Abbildung 8.9Abbildung 8.7).

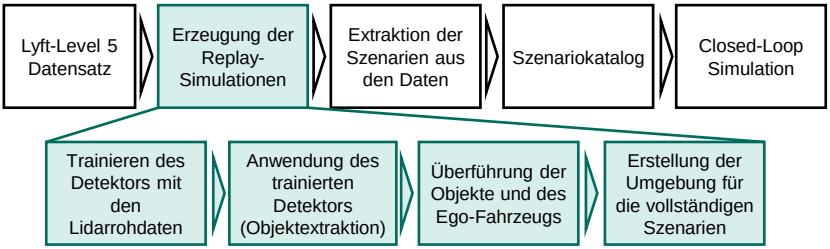


Abbildung 8.9: Schritte der Überführung des Lyft-Datensatzes in die Closed-Loop Simulation

8.2.2 Ergebnisse

Das Ziel ist es, analog zum Vorgehen für den highD-Datensatz (vgl. Abschnitt 8.1.2), die Genauigkeit zu bewerten, mit der der Lyft-Level 5 Datensatz mit Hilfe des in Kapitel 6 entwickelten Modells automatisiert in eine Closed-Loop Simulation überführt werden kann.

Im Gegensatz zum highD-Datensatz muss allerdings berücksichtigt werden, dass, bedingt durch die Anwendung auf den Sensorrohdaten im Lyft-Datensatz, der dabei entstehende Fehler in der Replay-Simulation durch das zuvor vorgestellte Verfahren berücksichtigt werden muss. Für den highD-Datensatz ergibt sich diese Fragestellung nicht, da von diesem nur die ausgewerteten Bilder der Drohnenkameras zur Verfügung gestellt werden. Etwaige Fehler die in diesem Schritt erfolgen, sind im Rahmen des Datensatzes nicht einsehbar und damit nicht bewertbar.

Für die Darstellung der Ergebnisse für den Lyft-Datensatz ergibt sich damit folgendes Schema (Abbildung 8.10):

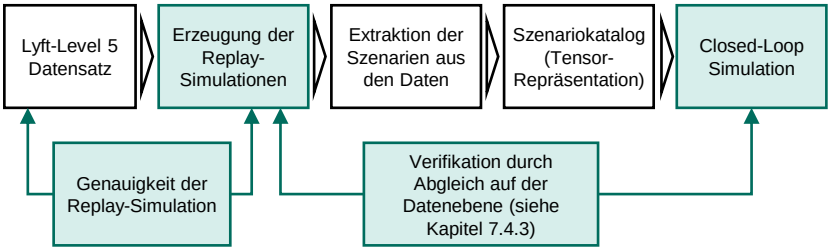


Abbildung 8.10: Ergebnissdarstellung für die Abbildung des Lyft-Datensatzes in der Closed-Loop Simulation

Die erzielte Genauigkeit der Replay-Simulation lässt sich wie folgt bewerten:

Im KITTI-Benchmark schneidet das verwendete SECOND-Netz für die Objektdetektion überdurchschnittlich ab. Für die hier hauptsächlich relevante Objektklasse „Car“ erreicht das Netz eine Genauigkeit von 75,96%[71] für die Benchmark-Metrik der 3D Objektdetektion aus [196], basierend auf den drei Teilen 2D Objektdetektion (anhand der Average Precision – AP [197]), 3D Orientierung (anhand Average Orientation Similarity – AOS [196]) und der Richtigkeit der Klassifikation (nach [196]). Zum Vergleich: Das aktuell „führende“ Framework im KITTI-Benchmark *MMLab PV-RCNN* erreicht in der gleichen Kategorie eine Genauigkeit von ca. 81% (Stand Anfang 2020).

In der hier durchgeführten Adaption des SECOND-Netzes auf den Lyft-Datensatz konnte eine Genauigkeit für diese Metrik von ca. 68% erzielt werden. Hierfür sind zwei Aspekte zu berücksichtigen:

[71] Diese Genauigkeit wird in der Schwierigkeitsstufe „moderate“ erzielt: Die zu erkennenden Bounding Boxen weisen dabei eine Mindesthöhe von 25 Pixeln auf und können einen Verdeckungsgrad von bis zu 30% innehaben.

1. Das SECOND-Netz wurde mit Benchmark-Ergebnissen für den KITTI-Datensatz veröffentlicht und demnach auf diesen Datensatz optimiert.

2. Der Lyft-Datensatz weist Ground Truth-Annotationen in einem größeren Bereich aus. Objekte in größerer Entfernung sind tendenziell „schwieriger“ zu detektieren.

Die Aussagekraft der zuvor vorgestellten Metrik ist allerdings nur bedingt geeignet, um über die Genauigkeit des Detektors hinaus die Güte der Replay-Simulation in Bezug auf die Sensorrohdaten zu bewerten. Eine weitere, ebenfalls verbreitete Metrik zur Bewertung stellen die Kenngrößen Multiple-Object-Tracking-Accuracy (MOTA) und Multiple-Object-Tracking-Precision (MOTP) dar [198]. Diese Kennzahlen weisen gegenüber den detektionsbezogenen Metriken den Vorteil auf, dass sie zum einen die Präzision des Trackings der Objekte hinsichtlich der durch den Tracker geschätzten Abweichung zu den tatsächlichen, exakten Positionen mit einbeziehen und zum anderen die korrekte Zuordnung eines Objekts zu einer ID über die Zeit berücksichtigen.

MOTP definiert sich wie folgt [198]:

$$MOTP = \frac{\sum_{i,t} d_t^i}{\sum_t c_t} \tag{8.1}$$

MOTP beschreibt damit die Summe aller Fehler der geschätzten Positionen der gematchten Objekte über die Zeit t durch die Summe aller Matches die durch den Tracker erkannt werden. Als „MOTP overlap“-Index wird die relative Abweichung von einer vollständigen Übereinstimmung der geschätzten zu den tatsächlichen Bounding Boxen bestimmt, sodass sich hier bei vollständiger Übereinstimmung ein Maximalwert von 100% ergibt.

MOTA ergibt sich zu [198]:

$$MOTA = \frac{\sum_t (m_t + fp_t + mme_t)}{\sum_t g_t} \tag{8.2}$$

Diese Kennzahl besteht aus drei Teilen, der Summe aller nicht zugeordneten zu matchenden Objekte („miss“, m_t), der false positives fp_t und der fälschlich

vertauscht zugeordneten zu matchenden Objekte („missmatch“, mme_t) über der Summe aller Objekte über die Zeit.

In der Anwendung dieser Kennzahlen auf die Objekte in der Replay-Simulation des Lyft-Datensatzes ergibt sich ein MOTP-Wert von 78% und ein MOTA-Wert von 82%. Zum Vergleich: Der Stand Anfang 2020 beste veröffentlichte Tracking Algorithmus nach dem KITTI-Datensatz Benchmark erzielt einen MOTP-Wert von 84% und einen MOTA-Wert von 90%.

Insgesamt lassen sich aus den ermittelten Werten für die Genauigkeit der Replay-Simulation zwei Implikationen ableiten:

1. Die Implementierung des Detektors und des Tracking Algorithmus erreicht für den Lyft-Datensatz eine Performance, die, mit leichten Abstrichen, mit der Performance vergleichbar ist, die mit Stand-der-Technik-Verfahren im KITTI-Benchmark erzielt wird.

2. Die insgesamt erreichbare Genauigkeit für die vollständig automatisch generierte Replay-Simulation auf Basis von Sensorrohdaten (Lidarpunktwolken) ist mit den verwendeten Stand-der-Technik-Verfahren nicht ausreichend, um für den hier verfolgten Anwendungsfall der Generierung der Closed-Loop Simulation ohne weitere Korrektur verwendet werden zu können (vergleiche hierzu auch die nachfolgenden Ergebnisse).

Das Vorgehen zur Bewertung der Genauigkeit der Closed-Loop Simulation im Vergleich zur Replay-Simulation (siehe Abbildung 8.10) erfolgt analog zur Herangehensweise für den highD-Datensatz (vergleiche Abschnitt 8.1.2):

In den Testdaten zum Lyft-Datensatz wurden acht Sequenzen identifiziert, die als autobahnähnlich angenommen werden können, damit das Kriterium a) (vergleiche Anfang Kapitel 8) erfüllen und somit für die Evaluierung herangezogen werden. Damit weisen die nachfolgend präsentierten Ergebnisse im Vergleich zu den 500 untersuchten Sequenzen des highD-Datensatz eine geringere statistische Signifikanz auf. Für die sechs Kenngrößen erzielt der *MAPE* für die Testdaten des Lyft-Datensatzes folgende Ergebnisse (Abbildung 8.11):

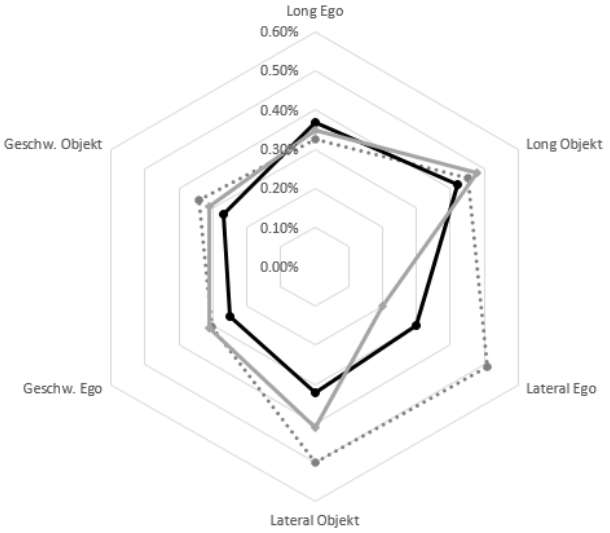


Abbildung 8.11: Durchschnittlicher Fehler über die Kenngrößen aus Abschnitt 7.4.3. für die Closed-Loop Simulationen im Lyft-Datensatz (grau) im Vergleich zum highD-Datensatz (schwarz) und simulierten Eingangsdaten aus Kapitel 7 (gestrichelte Linie)

Die Abbildung der Kenngrößen zeigt, dass übergreifend die Werte für den Lyft-Datensatz in gleichen Bereichen wie im highD-Datensatz liegen. Auch im Vergleich zu den simulierten Eingangsdaten zeigen beiden Studien ähnliche Werte und die gleichen, bereits in Abschnitt 8.1.2 diskutierten Effekte auf.

Setzt man allerdings die Kenngrößen für die Abbildungsgenauigkeit der Closed-Loop Simulation statt in Bezug zur Replay-Simulation zu den Ursprungsdaten (Ground Truth der Sensorrohdaten), erhält man um Faktor 5 (Ego-Fahrzeug) bis 50 (Objekte) höhere Werte (Abbildung 8.12) .

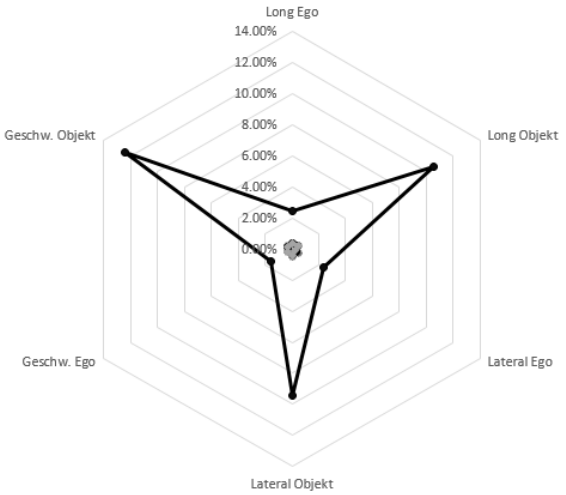


Abbildung 8.12: Durchschnittlicher Gesamtfehler der Closed-Loop Simulation in Relation zu den Ground Truth Informationen des Lyft-Datensatzes

Zusammenfassend lassen sich diese Ergebnisse wie folgt interpretieren:

- Die Eignung des in Kapitel 6 entwickelten Modells für die Abstraktion der Szenarien kann auch unter Verwendung von Realdaten bestätigt werden. Es können automatisiert Closed-Loop Simulationen aus Realdatenaufzeichnungen erstellt werden.

- Die unter Verwendung von Realdaten erzielten Ergebnisse übertreffen sogar durchschnittlich die Ergebnisse unter Verwendung von synthetischen Eingangsdaten aus Kapitel 7.4.

- Diese Ergebnisse gelten nur, wenn im Zwischenschritt eine Replay-Simulation der Originaldaten erstellt wird und gegen diese validiert wird.

- Die automatische Erstellung der Replay-Simulation ist mit aktuellen Methoden der Sensorrohdatenauswertung fehlerbehaftet. Der hier erzeugte Fehler liegt für die Objekte um mindestens eine Größenordnung höher als der Fehler der durch das Abstraktionsmodell erzeugt wird.

9 Zusammenfassung und Ausblick

Hochautomatisierte Fahrfunktionen erfordern neue Methoden für den Test und Nachweis der funktionalen Sicherheit. Herkömmliche Ansätze wie die physische Breitenerprobung durch Prototypenfahrzeuge stellen keine ökonomisch sinnvoll durchführbare Möglichkeit dar, um den erforderlichen Abdeckungsgrad im Test dieser Funktionen zu erreichen. Im Gegensatz dazu stehen simulationsbasierte Methoden (X-in-the-Loop), die sich durch eine hohe Effizienz und Variabilität in der Testdurchführung auszeichnen. Um diese Methoden freigaberelevant einsetzen zu können, ist zum einen der Nachweis der Validität der eingesetzten Modelle und Umgebungen erforderlich, zum anderen muss die Repräsentativität der im Labor erstellten Testfälle gewährleistet sein. Für Funktionen des hochautomatisierten Fahrens können diese Testfälle nicht mehr vollständig systematisch aus den Anforderungen abgeleitet werden, sodass alternative Ansätze herangezogen werden müssen.

Auf Basis dieser Motivation wurde in dieser Dissertation ein Prozess entwickelt, der zum Ziel hat, die relevanten Testfälle als Szenarien aus aufgezeichneten Daten zu extrahieren und sie so aufzubereiten, dass sie simulationsbasierten Methoden zur Verfügung stehen. Damit können im Gegensatz zur klassischen Breitenerprobung statt teuren Prototypenfahrzeugen auch Serienfahrzeuge herangezogen werden, die nicht mit der zu testenden Funktion ausgerüstet sein müssen, sofern sie mit einer entsprechenden Umfeldsensorik ausgestattet sind. Die auf diese Weise gesammelten Fahrsituationen können als Referenzfälle für eine zu testende Funktion herangezogen werden. So kann sichergestellt werden, dass die Funktion nicht nur alle im Labor erdenklichen Situationen beherrscht, sondern auch die unbekannten Fälle Eingang in den Testprozess finden. Auf diese Weise wird es möglich, die Vorteile des realen Fahrversuchs hinsichtlich der auftretenden Situationsvielfalt mit den Vorteilen der Simulationsmethoden hinsichtlich der Testeffizienz zu verbinden. Damit soll letztendlich eine erforderliche Testabdeckung auch durch die Gewährleistung der ökonomischen Durchführbarkeit der Testfälle erzielt werden.

In Abbildung 9.1 sind in Anlehnung an den in Kapitel 5 vorgestellten Prozess die erforderlichen Schritte vorgestellt, auf die im Rahmen dieser Arbeit konkret eingegangen wurde.

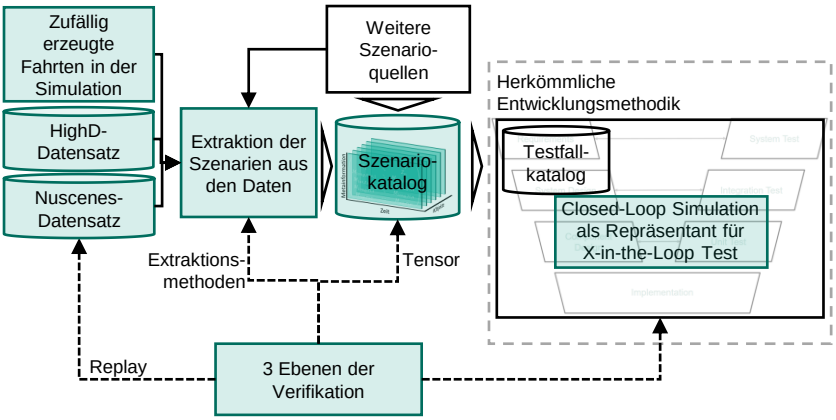


Abbildung 9.1: Prozess zur Extraktion von Szenarien und Testfällen aus Realdaten und die in dieser Arbeit betrachteten Aspekte (grün)

Ein szenariobasierter Testansatz stellt die zentrale Komponente in diesem Prozess dar. In dieser Arbeit wird die dem Stand der Technik entsprechende Definition des Szenarios mit der Schärfung und Differenzierung des Manöverbegriffs erweitert. Die sogenannten Basismanöver bilden dabei ein wesentliches Element zur konsistenten Beschreibung der Zusammenhänge zwischen den dynamischen Objekten in einem Szenario. Auf dieser Grundlage erfolgte die Entwicklung eines Szenario-Metamodells zur Beschreibung des zeitlichen Verlaufs auf einer für diesen Prozess geeigneten Abstraktionsebene. Als potentielle Anwendungsumgebung wurde ein Autobahnpilot als zu entwickelnde Funktion beziehungsweise System-under-Test angenommen. Eine wesentliche Eigenschaft des Szenario-Metamodells ist die Sicherstellung der Anwendbarkeit auf den entwickelten Prozess.

Das entwickelte Modell fokussiert auf die Beschreibung des Verhaltens der dynamischen Verkehrsteilnehmer. Um der Tatsache gerecht zu werden, dass auch statische Elemente zu einer vollständigen Szenariobeschreibung gehören,

wurden Ansätze präsentiert, wie auch diese Elemente potentiell in das Modell integriert werden können. Die Informationen des Szenario-Metamodells werden in einer Tensordarstellung gespeichert, die zum einen speichereffizient genutzt werden kann und die zum anderen Szenarien beliebiger Länge aufnehmen kann. Szenarien können durch das Schneiden dieser Tensoren beliebig „herausgelöst“ werden, was eine Katalogisierung der Szenarien und damit gezielte Analysen erst ermöglicht.

Für die Extraktion der Szenarienbestandteile wurden verschiedene Verfahren vorgestellt und angewandt. Auf Grundlage simulierter Daten konnte gezeigt werden, dass Verfahren der Zeitreihenanalyse grundsätzlich anwendbar sind, um die Basismanöver zu identifizieren. Auf der Objektlistenebene konnte sowohl für die Erkennung der Manöver des Ego-Fahrzeugs als auch für die der detektierten Verkehrsobjekte eine hohe Genauigkeit erreicht werden. Insbesondere die künstlichen neuronalen Netze erreichten eine Genauigkeit von 95% und darüber, auch bei potentiell unterschiedlichen zur Verfügung stehenden Signalen auf der Objektlistenebene. Es ist anzunehmen, dass die Genauigkeit durch eine weitere Optimierung der Hyperparameter der Netze, wie beispielsweise das Hinzufügen zusätzlicher versteckter Schichten, noch weiter erhöht werden kann.

Es wurde am Beispiel des Tools CarMaker dargestellt, wie die in das Szenario-Metamodell überführten extrahierten Informationen automatisiert in eine Simulationsumgebung transformiert werden können. Da im Rahmen der Arbeit keine tatsächliche hochautomatisierte Fahrfunktion zur Verfügung stand, wurde die Abbildung auf die reine Closed-Loop Simulation ohne Hardwarekomponenten beschränkt. Die Simulationsumgebung steht dabei stellvertretend für den X-in-the-Loop Test.

Es konnte gezeigt werden, dass die Bedatung der Simulation aus der Tensordarstellung heraus und die anschließende Closed-Loop Simulation repräsentativen Ergebnissen bezogen auf die Ausgangsdaten führt. Anhand der drei Ebenen der Verifikation konnten folgende Ergebnisse erzielt werden:

- Die Analyse der Codierungen der Zustands- und Manöversequenzen können zur Plausibilisierung der Ausgangsdaten herangezogen werden (um beispielsweise Sensorfehler in den Daten zu ermitteln).

- Bildet man die jeweiligen Tensorrepräsentationen zu den Ausgangsdaten und den daraus erstellten Resimulationen, zeigt der Vergleich eine gute Übereinstimmung. Insbesondere die Zustands- und Manöverfolgen werden nahezu fehlerlos wiederhergestellt.

- Auf der Datenebene gelingt in der Closed-Loop Simulation die „Wiederherstellung“ der Ausgangsdaten auf einem sehr guten Niveau. Der durchschnittliche Fehler in den untersuchten kinematischen Signalen des Ego-Fahrzeugs und der Verkehrsobjekte war stets unter 1%.

- In der Simulation können Effekte reproduziert werden, die auf der verwendeten Ebene der Basismanöver nicht explizit beschrieben werden und eigentlich den atomaren Manövern zuzuordnen sind.

Unter der Verwendung von Realdaten anhand des highD- und des Lyft-Datensatzes konnten die Evaluierungsergebnisse aus der Simulation grundsätzlich bestätigt werden. Für einen voll-automatischen Einsatz des Prozesses aus Kapitel 5 bedeuten die teils hohen Fehlerquoten in der Sensorrohdatenverarbeitung noch eine Herausforderung. Zumindest für die Anwendung ab der Objektlistenebene zeigt sich aber auch für Verwendung von Realdaten eine sehr gute Übereinstimmung zwischen den Daten und den Ergebnissen der Closed-Loop Simulation.

Insgesamt stellt damit das entwickelte Szenario-Metamodell einen erfolgsversprechenden Ansatz dar, die Szenarien im Sinne des Prozesses auf einer geeigneten Abstraktionsebene, zumindest für die nicht sensorbezogenen Teile der Wirkkette einer hochautomatisierten Fahrfunktion, abbilden zu können (vergleiche Konzept der funktionalen Dekomposition [138]). Die Abstraktion bildet dabei einen Kompromiss zwischen hoher Abbildungsgenauigkeit und der Möglichkeit, die Szenarien miteinander vergleichbar zu machen. Für den Ab-

sicherungsprozess bedeutet das, dass grundsätzlich die Voraussetzung geschaffen wird, Situationen und Szenarien im Feld zu sammeln und sie den Simulationsmethoden repräsentativ zur Verfügung stellen zu können.

9.1 Ausblick

Diese Arbeit liefert einen Beitrag zu einer neuartigen Absicherungsmethodik für hochautomatisierte Fahrfunktionen. Es wird ein Prozess vorgestellt und prototypisch auf Umsetzbarkeit überprüft, bei dem die erforderlichen Testfälle aus aufgezeichneten realen Situationen abgeleitet werden können und für die Simulation aufbereitet werden. Folgende zukünftige Forschungsfelder knüpfen direkt an die hier erzielten Erkenntnisse an:

9.1.1 Vervollständigung des Szenario-Metamodells

Der Schwerpunkt für die Gestaltung des Szenario-Metamodels lag auf der Abbildung der dynamischen Komponenten eines Szenarios. Zur Vervollständigung ist die Ergänzung um die statischen Bestandteile notwendig, die hier nur konzeptionell und exemplarisch angerissen wurde. Erst mit der vollständigen Integration aller Szenario-Elemente wird das Szenario-Metamodell im Sinne des hier dargestellten Prozesses nutzbar.

Insbesondere die Abbildung gewonnener Karteninformationen stellt dabei einen wesentlichen Faktor dar. Für viele der in der Industrie verbreiteten Simulationstools existieren bereits Mechanismen und Schnittstellen um digitale Karten oder Elemente der Karten in die Simulation überführen zu können. Das Format OpenDRIVE stellt in diesem Zusammenhang ein verbreitetes Standardformat für die einheitliche Beschreibung statischer Szenario-Informationen dar und dient damit auch dazu, den Austausch zwischen Tools und Umgebungen voranzutreiben. Eine zusätzliche Erweiterungsmöglichkeit besteht darin, Formate wie OpenDRIVE oder auch OpenSCENARIO in das Szenario-Metamodell überführen zu können und vice versa.

Aus Sicht der Fahrdynamik wurden im hier entwickelten Szenario-Metamodell das longitudinale und laterale Verhalten der Objekte berücksichtigt, die Vertikaldynamik hingegen außen vorgelassen. Für eine vollständige Beschreibung der Fahrdynamik muss das Szenario-Metamodell um diese Komponente ebenfalls ergänzt werden.

9.1.2 Formalisierung der Katalogisierung

Im Hauptteil wurden bereits Ansatzmöglichkeiten zur Katalogisierung der (extrahierten) Szenarien aufgezeigt. Das vorgestellte Schneiden der Tensorrepräsentationen entlang der enthaltenen Szenarien einer Datenaufzeichnung ist die Grundvoraussetzung. Durch die im Szenario-Metamodell gewählte Szenenrepräsentation und die Codierung der Szenensequenzen im Tensor können beliebige zu analysierende funktionale oder logische Szenarien wie beispielsweise Überholvorgänge bestimmt werden. Anschließend kann die Menge aller Aufzeichnungen auf diese Szenarien automatisiert durchsucht werden und die so gefundenen Szenarien für weitere Untersuchungen herangezogen werden (siehe Abbildung 9.2).

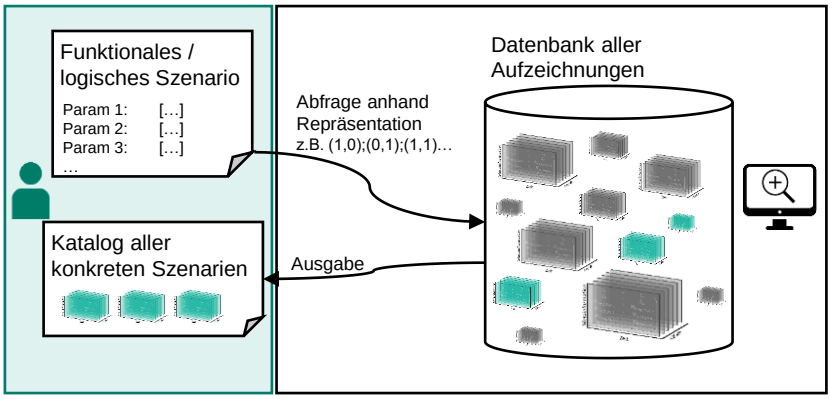


Abbildung 9.2: Schema der Katalogerzeugung von Szenarien aus der Datenbank

Die grundsätzliche Machbarkeit dieses Vorgehens konnte bereits demonstriert werden.[72] Des Weiteren eröffnen sich durch diesen Ansatz weitere Möglichkeiten, die Eigenschaften der Menge der konkreten Szenarien zu den bestimmten funktionalen oder logischen Szenarien zu untersuchen. Insbesondere die Methoden des maschinellen Lernens bieten hier vielversprechende Verfahren, um beispielsweise anhand von statistischen Verteilungen oder Clusterbildungen seltene oder auch kritische Szenarien zu identifizieren und anschließend gezielt untersuchen zu können. Erste Arbeiten, Analysen und Ergebnissen zu Clusterverfahren für diesen Anwendungsfall wurden in [182] vorgestellt.

Neben der Formalisierung des grundsätzlich in Abbildung 9.2 vorgestellten Prinzips, ergibt sich somit insbesondere ein weiterer Forschungsbedarf hinsichtlich den Anwendung weiterer Analysemethoden und –verfahren für die Bewertung und Untersuchung der nach diesem Vorgehen gewonnen Szenarien.

9.1.3 Entwicklung einer Abdeckungsmetrik

Das ultimative Ziel dem sich letztendlich alle Testverfahren in diesem Kontext und auch der hier vorgestellte Prozess unterordnen, ist der Nachweis der funktionalen Sicherheit verbunden mit der Freigabe der konkreten zu entwickelnden Fahrfunktion. In dieser Arbeit beschränken sich sämtliche Untersuchungen im Rahmen des Prozesses auf die Resimulation der ursprünglich aufgezeichneten Daten nach dem Closed-Loop Prinzip, um die grundsätzliche Anwendbarkeit darzustellen. Es stand keine explizite Fahrfunktion als System-under-Test zur Verfügung. Unter Anwendung einer konkreten Fahrfunktion ergibt sich ein wesentlicher Unterschied, da im zeitlichen Verlauf der Entwicklung eines Szenarios dieses durch die Rückkopplung mit der Funktion einen wesentlich anderen Verlauf nehmen kann, als das ursprünglich aufgezeichnete Szenario - selbst bei exakt gleichen Startbedingungen. Wie kann also unter diesen Umständen eine Absicherung der Funktion erreicht und durch den Nachweis einer Testabdeckung gezeigt werden?

[72] Vgl. hierzu Masterarbeit (Schweizer, Dennis; 2019).

Ein möglicher auf die hier präsentierten Ergebnisse aufbauender Forschungsansatz kann wie folgt skizziert werden:

Folgt man der Argumentation zur Ableitung der Forschungsfrage und geht man davon aus, dass das menschliche Fahrverhalten zum einen die (Mindest)-Referenz für ein funktional sicheres System ist und zum anderen mit einer ausreichend großen Menge an realen Fahrsituationen ein statistischer Nachweis für die Sicherheit erfolgen kann, stellt der Raum dieser Fahrsituationen den Raum der erforderlichen Testabdeckung dar. Das hier entwickelte Szenario-Metamodell kann in diesem Zusammenhang zur „Diskretisierung“ dieses eigentlich kontinuierlichen Raumes herangezogen werden. Eine Metrik zur Testabdeckung könnte also darauf basieren, die erforderliche Menge an Fahrsituationen für einen statistischen Nachweis über eine Fahrzeugflotte zu sammeln und entsprechend des Metamodells in seine enthaltenen (Teil)-szenarien und Szenen zu zerlegen. Diese dienen als Input für Startbedingungen in einer X-in-the-Loop Umgebung mit der zu prüfenden Fahrfunktion. Die X-in-the-Loop Simulation erzeugt unter Verwendung der Fahrfunktion wiederum neue Szenarien, die analog zu den originalen Szenarien gespeichert und abstrahiert werden können (siehe Abbildung 9.3).

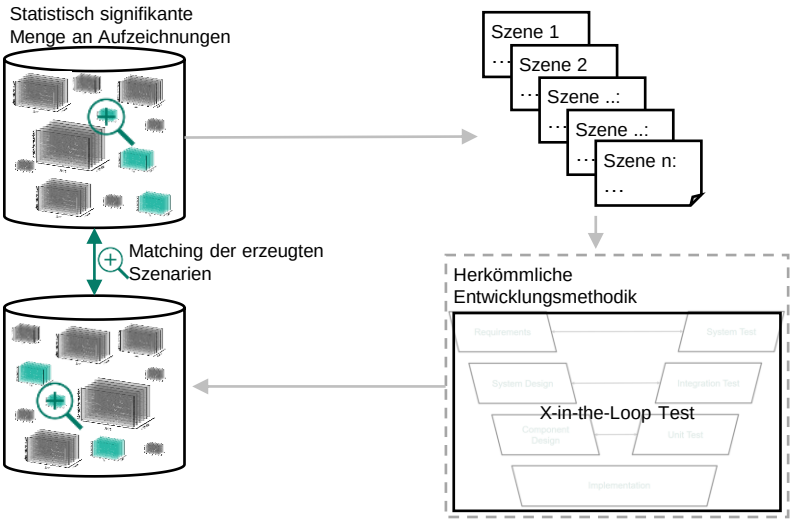


Abbildung 9.3: Schema zur Bestimmung der Testabdeckung für eine hochautomatisierte Fahrfunktion

Unter der Voraussetzung, dass sich ein valides Maß für den Vergleich der Szenen oder Szenarien bilden lässt, könnte eine Abdeckungsmetrik aus dem Verhältnis der aus dem X-in-the-Loop Test erzeugten Einzelszenen, Szenensequenzen oder Szenarien mit den originalen Einzelszenen, Szenensequenzen oder Szenarien als Grundgesamtheit gebildet werden.

Ein möglicher Ansatz für den Vergleich von Szenen und Szenensequenzen auf Basis des Szenario-Metamodells unter der Verwendung von Autoencodern wurde bereits in [199] präsentiert.

9.1.4 Erweiterung auf andere Teile der Wirkkette

Betrachtet man die Wirkkette einer hochautomatisierten Fahrfunktion, beispielsweise anhand des Konzepts der funktionalen Dekomposition (vergleiche Kapitel 5 und [138]), lässt sich diese in verschiedene funktionale und zu testende Teile zerlegen - vom Sensor und der Informationsgewinnung hin zum

Aktor und der Umsetzung der Fahrzeugbewegung. Der Ansatz dieser Arbeit bezieht sich auf die Teile ab der Schnittstelle nach der Sensordatenverarbeitung. Das Szenario-Metamodell zielt damit konkret auf die Abbildung der erforderlichen Szenarien zur Prüfung von Testfällen der eigentlichen Fahrfunktion und ihre Umsetzung hinsichtlich des Fahrverhaltens ab. Im Gegensatz zu den sensorbezogenen Funktionen, wie der jeweiligen Sensorsignalverarbeitung oder der Objektklassifikation, ist es für die Fahrfunktion beispielsweise irrelevant, welche Farbe ein detektiertes Objekt hat. Da der hier vorgestellte Prozess auf einer Aufzeichnung der Daten durch die (fehlerbehafteten) Fahrzeugsensoren beruht, können die vorderen Teile der Wirkkette bis zur Sensorrohdatenverarbeitung nicht berücksichtigt werden und es müssen, zumindest für diese Teile, alternative und zusätzliche Absicherungsverfahren erforscht werden (siehe Abbildung 9.4).

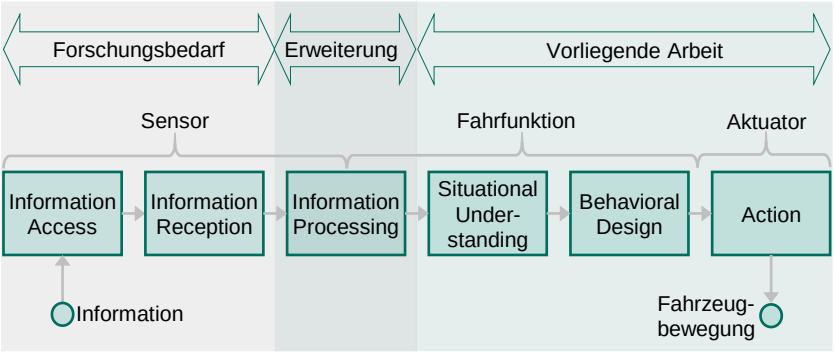


Abbildung 9.4: Forschungsbedarf am Schema der Wirkkette der funktionalen Dekomposition nach [138]

Bezieht man die sensordatenverarbeitenden Schritte mit ein und geht man davon aus, dass eine Realdatenspeicherung und –gewinnung im Feld im Rahmen des vorgestellten Prozesses auch auf der Rohsignalebene möglich ist, kann das Konzept wie dargestellt erweitert werden. Grundlage hierfür ist die Annahme, dass die Sensorrohdaten im Labor im Gegensatz zu den vergleichsweise be-

schränkten Ressourcen im Fahrzeug und den dort herrschenden Echtzeitbedingungen mit geringer fehlerbehafteten Methoden analysiert und ausgewertet werden können.

Gleichwohl lässt sich die Grenze der Betrachtung damit zwar verschieben, ermöglicht aber nicht die vollständige Berücksichtigung der Wirkkette. Hier sind alternative Forschungsansätze gefordert, um die Lücke für die Absicherung zukünftig weiter zu schließen.

Anhang

A: Künstliche neuronale Netze

Künstliche neuronale Netze (KNN) stammen aus dem Bereich der maschinellen Lernverfahren und sind als Informationsverarbeitungsmodelle der Funktionsweise menschlicher Gehirnen nachempfunden [200]. In einem Teilbereich des maschinellen Lernens, dem überwachten Lernen, können KNN als Klassifikatoren eingesetzt werden, wenn sie zuvor mittels geeigneter Trainingsalgorithmen und Trainingsdaten, die sowohl Eingangsvektoren als auch die zugehörigen Zielvektoren beinhalten, trainiert wurden [201]. Grundeinheiten der KNN sind die einzelnen Neuronen (Perzeptronen) als Klassifikatoren, die sich wiederum aus einem Eingangsvektor $x_{1,\dots,}x_n$, einem Gewichtsvektor $w_{1,\dots,}w_n$, einer Summenfunktion Σ, einer Aktivierungsfunktion φ mit einem Schwellenwert θ und einem Aktivierungswert $o(\vec{x})$ zusammensetzen (Abbildung 9.5) [202]. Das Ausgangssignal (Aktivierungswert) ergibt sich in Abhängigkeit der Aktivierungsfunktion φ zu:

$$o = \varphi\left(\sum_{i=1}^{n} x_i\, w_i\right) \tag{A.1}$$

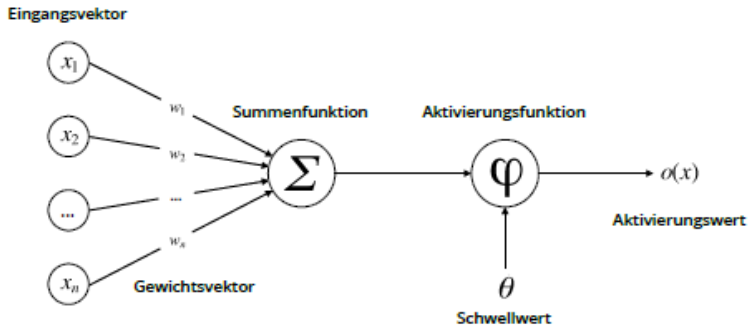


Abbildung 9.5: Modell eines Perzeptrons [202]

Als Klassifikator liefert das Perzeptron das richtige Ergebnis für ein Objekt einer Klasse *t*, wenn das Ergebnis *o* der Aktivierungsfunktion $\varphi(\vec{x};\vec{w})$ der tatsächlichen Klasse des Objekts *t* entspricht. Wenn die Klasse nicht richtig erkannt wurde $o(\vec{x}) \neq$ t, werden die Gewichte $\vec{w}$ entsprechend einer Lernregel angepasst.

Das KNN besteht aus mehreren Perzeptronen und ist in mehreren Schichten angeordnet (vergleiche Abbildung 9.6). Die Eingangsschicht nimmt den Inputvektor $\vec{I}$ auf. Nachfolgend können ein oder mehrere versteckte Schichten angeordnet sein, die miteinander verbunden sind. Die Ausgangsschicht liefert letztendlich den Outputvektor $\vec{O}$ und damit das Ergebnis des KNN als Klassifikator.

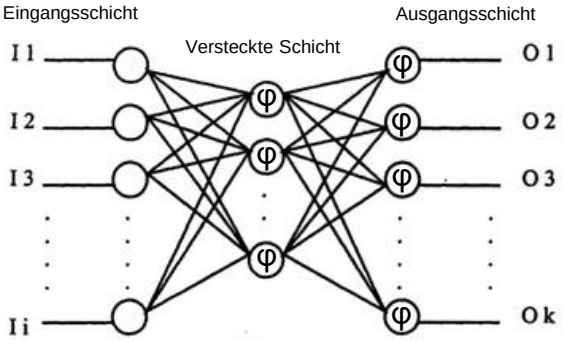


Abbildung 9.6: KNN als Multilayer Perzeptron mit einer versteckten Schicht [203]

Um ein KNN als Klassifikator einsetzen zu können, muss zunächst ein Lernprozess durchgeführt werden, bei dem Trainingsdaten in Form von Inputvektoren $\vec{I}$ mit ihren zugehörigen Outputvektoren $\vec{O}$ verwendet werden. Dieser Prozess wird als *überwachtes Lernen* (engl. supervised learning) bezeichnet [201]. Soll beispielsweise ein KNN als Klassifikator für die Erkennung von Bildern eingesetzt werden, muss bei den Trainingsdaten zusätzlich zum eigent-

lichen Bild (als Inputvektor) auch die zugehörige Klasse (z.B. „PKW“) vorliegen. Der Prozess zur Anreicherung der Inputdaten um ihre Klasseninformation wird auch als Annotation (engl. labeling) bezeichnet.

Analog zum Training des einzelnen Perzeptrons werden für das Training des KNN die Gewichte $\overrightarrow{w}$ initialisiert (beispielsweise mit zufälligen Werten). Bei jedem Trainingsschritt wird überprüft, ob die berechnete Klasse $o(\vec{x})$ der tatsächlichen Klasse *t* entspricht. Ist dies nicht der Fall, werden die Gewichte $\overrightarrow{w}$ unter Anwendung einer Lernregel aktualisiert.

B: Rekurrente neuronale Netze und LSTMs

Eine Variante von KNN stellen die sogenannten rekurrenten neuronalen Netze (RNN) dar. RNN eignen sich für sequentielle Daten, wie beispielsweise Zeitreihen, da beim Training auch Werte von Neuronen derselben Schicht berücksichtigt werden. Dies wird im Gegensatz zum klassischen Multilayer Perzeptron durch zyklische Verbindungen innerhalb der Schichten ermöglicht. Damit können RNN nicht nur einen Input(vektor) auf einen Output abbilden, sondern speichern auch die Historie vorheriger Inputs. Je nach Tiefe der Architektur können unterschiedlich lange Eingabesequenzen gespeichert werden [204].

Eine Herausforderung für die RNN besteht darin, dass, bedingt durch das Training, der Einfluss eines bestimmten Inputs auf die versteckten Schichten entweder abklingt oder exponentiell wächst, wenn es die zyklischen Verbindungen des Netzes durchläuft [204]. Dieser Effekt wird in der Literatur als *Problem des verschwindenden Gradienten* (engl. vanishing gradient problem) beschrieben. Damit sind die klassischen RNN in der Praxis für längere Inputsequenzen nur schwer anwendbar. Eine spezielle Variante der RNN zur Lösung des Problems stellen die Long Short-Term Memorys (LSTMs) dar, die erstmalig von Hochreiter und Schmidhuber vorgestellt wurden [205].

LSTM-Netze bestehen in den versteckten Schichten aus LSTM-Blöcken, die aus drei Toren, dem *Eingangstor* (engl. input gate), dem *Merk-* und *Vergesstor* (engl. forget gate) und dem *Ausgangstor* (engl. output gate) aufgebaut sind.

Um das Problem des verschwindenden oder explodierenden Gradienten zu verhindern, wird der Zellenzustand innerhalb eines LSTM-Blocks gespeichert. Dazu werden sogenannte Peepholes verwendet, die den Zugriff auf den Zellenzustand ermöglichen (siehe Abbildung 9.7).

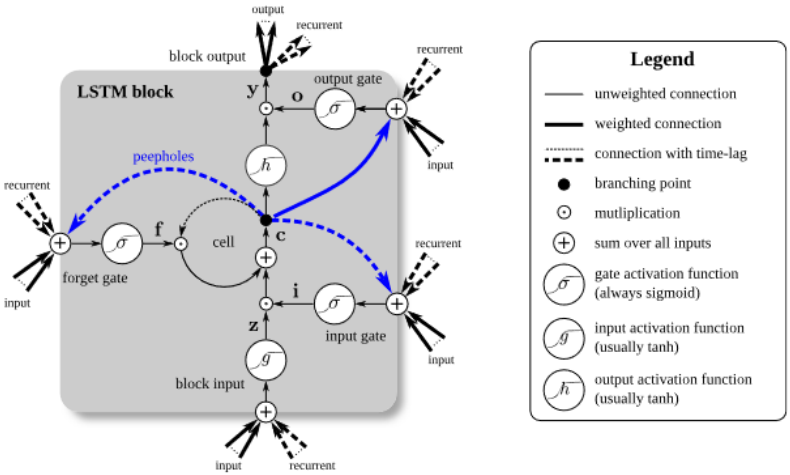


Abbildung 9.7: LSTM-Block als Bestandteil der Hidden Layer [206]

Für einen definierten Inputvektor x^t zum Zeitpunkt t ergeben sich für eine Architektur mit N LSTM-Blöcken und M Inputs nach [206] folgende Gewichte:

1. Input-Gewichte: $W_z, W_s, W_f, W_o \in R^{N \times M}$
2. Rekurrente Gewichte: $R_z, R_s, R_f, R_o \in R^{N \times N}$
3. Peephole-Gewichte: $p_s, p_f, p_o \in R^N$
4. Bias-Gewichte: $b_z, b_s, b_f, b_o \in R^N$

Mit den in Abbildung 9.7 dargestellten Konventionen ergeben sich folgende Formeln für die Komponenten [206]:

$$
\begin{aligned}
\bar{z}^t &= W_z x^t + R_z y^{t-1} + b_z \\
z^t &= g(\bar{z}^t) && \text{block input} \\
\bar{\imath}^t &= W_i x^t + R_i y^{t-1} + p_i \odot c^{t-1} + b_i \\
i^t &= \sigma(\bar{\imath}^t) && \text{input gate} \\
\bar{f}^t &= W_f x^t + R_f y^{t-1} + p_f \odot c^{t-1} + b_f \\
f^t &= \sigma\left(\bar{f}^t\right) && \text{forget gate} \\
c^t &= z^t \odot i^t + c^{t-1} \odot f^t && \text{cell} \\
\bar{o}^t &= W_o x^t + R_o y^{t-1} + p_o \odot c^t + b_o \\
o^t &= \sigma(\bar{o}^t) && \text{output gate} \\
y^t &= h(c^t) \odot o^t && \text{block output}
\end{aligned}
\tag{A.2}
$$

mit σ, g und h als jeweilige, nichtlineare Aktivierungsfunktionen (Sigmoid oder Tangens Hyperbolicus).

Weitere Grundlagen zu den LSTM-Netzen und ihre Trainingsmethoden (z.B. via Backpropagation) sind in [205] und [206] beschrieben.

Literaturverzeichnis

[1] SAE INTERNATIONAL, „Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles,“ 2016.

[2] ISO 26262:2018, Road vehicles – Functional safety, 2018.

[3] W. Wachenfeld und H. Winner, „Die Freigabe des autonomen Fahrens,“ in *Autonomes Fahren*, Berlin, Springer Vieweg, 2015, pp. 439 - 464.

[4] E. Sax, Automatisiertes Testen Eingebetteter Systeme in der Automobilindustrie, München: Carl Hanser Verlag, 2008.

[5] A. Lutz, B. Schick, H. Holzmann, M. Kochem, H. Meyer-Tuve, O. Lange, Y. Mao und G. Tosolin, „Simulation methods supporting homologation of Electronic Stability Control in vehicle variants,“ *Vehicle System Dynamics*, Bd. 55, Nr. 10, p. 1432 – 1497, 1 Juni 2017.

[6] W. Bernhart, „Automatisiertes Fahren – Evolution statt Revolution,“ *ATZextra Sonderheft 7/2015*, 1 April 2015.

[7] McKinsey&Company, „Smart Moves Required – Der Weg zu künstlicher Intelligenz im Mobilitätssektor,“ *McKinsey Center for Future Mobility*, 2017.

[8] Fraunhofer IAO - Horvath&Partners, „The Value of Time - Nutzerbezogene Service-Potenziale durch autonomes Fahren,“ 2016.

[9] H. Richter, „Elektronik und Datenkommunikation im Automobil,“ *IfI Technical Report Series (IfI-09-15), Institut für Informatik, Technische Universität Clausthal*, 2009.

[10] J. Schäuffele und T. Zurawka, Automotive Software Engineering, Wiesbaden: Springer Fachmedien Wiesbaden, 2016.

[11] R. N. Charette, „This Car Runs on Code,“ [Online]. Available: https://spectrum.ieee.org/transportation/systems/this-car-runs-on-code. [Zugriff am 25 3 2018].

[12] M. Maurer, „Entwurf und Test von Fahrerassistenzsystemen,“ in *Handbuch Fahrerassistenzsysteme*, Wiesbaden, Vieweg+Teubner, 2009, pp. 43 - 54.

[13] F. Schmidt, Funktionale Absicherung kamerabasierter Aktiver Fahrerassistenzsysteme durch Hardware-in the-Loop-Tests, Bd. 39, Stuttgart: Fraunhofer Verlag, 2012.

[14] A. Knapp, M. Neumann, M. Brockmann, R. Walz und T. Winkle, „RESPONSE 3 – Code of Practice for the Design and Evaluation of ADAS,“ 2009. [Online]. Available: https://www.acea.be/uploads/publications/20090831_Code_of_Practice_ADAS.pdf. [Zugriff am 1 April 2020].

[15] BASt, „CEDR Call 2014 DoRN Mobility & ITS. CEDR Transnational Road Research Programme,“ *Call 2014: Mobility & ITS' - Description of Research Needs (DoRN). Dezember 2014*, Bd. 2014.

[16] A. Herrmann, W. Brenner und R. Stadler, Autonomous Driving: How the Driverless Revolution will Change the World, Emerald Group Publishing, Hrsg., Bingley: emerald publishing, 2018.

[17] M. T. Oge, Driving the Future: Combating Climate Change with Cleaner, Smarter Cars, A. Publishing, Hrsg., New York, 2015.

[18] M. Aeberhard, S. Rauch, M. Ardelt und N. Kämpchen, „Autonomes Fahren auf der Autobahn – Eine Potentialstudie für zukünftige Fahrerassistenzsysteme,“ 2012. [Online]. Available: https://mediatum.ub.tum.de/doc/1142101/1142101.pdf. [Zugriff am 1 April 2020].

[19] H. Winner, S. Hakuli, F. Lotz und C. Singer, Hrsg., Handbuch Fahrerassistenzsysteme, 3., überarb. und erg. Aufl. Hrsg., Wiesbaden: Springer Vieweg, 2015.

[20] T. Streichert und M. Traub, Elektrik/Elektronik-Architekturen im Kraftfahrzeug, Berlin, Heidelberg: Springer Berlin Heidelberg, 2012.

[21] A. Weitzel, H. Winner, C. Peng, S. Geyer, F. Lotz und M. Sefati, Absicherungsstrategien für Fahrerassistenzsysteme mit Umfeldwahrnehmung, Bd. 98, Bremen: Fachverlag NW in der Carl Schünemann Verlag GmbH, 2014.

[22] K. Reif, Sensoren im Kraftfahrzeug, Wiesbaden: Vieweg+Teubner, 2010.

[23] K. Borgeest, Elektronik in der Fahrzeugtechnik: Hardware, Software, Systeme und Projektmanagement, 3. Auflage Hrsg., Wiesbaden: Springer Fachmedien, 2014.

[24] H. Krimmel und M. Ersoy, „Fahrwerkelektronik," in *Fahrwerkhandbuch: Grundlagen, Fahrdynamik, Fahrverhalten, Komponenten, Elektronische Systeme, Fahrerassistenz, Autonomes Fahren, Perspektiven*, 5. Auflage Hrsg., Wiesbaden, Springer Fachmedien, 2017, pp. 747 - 792.

[25] S. Beiker, „Implementierung eines selbstfahrenden und individuell abrufbaren Personentransportsystems," in *Autonomes Fahren: Technische, rechtliche und gesellschaftliche Aspekte*, Wiesbaden, Springer Fachmedien, 2015, pp. 287 - 312.

[26] H. Wallentowitz und K. Reif, Handbuch Kraftfahrzeugelektronik, Wiesbaden: Vieweg, 2006.

[27] K. Reif, „Fahrerassistenzsysteme," in *Fahrstabilisierungssysteme und Fahrerassistenzsysteme*, Wiesbaden, Springer Fachmedien, 2010, pp. 104 - 121.

[28] M. Maurer, Autonomes Fahren, M. Maurer, J. C. Gerdes, B. Lenz und H. Winner, Hrsg., Berlin: Springer Vieweg, 2015.

[29] P. Bergmiller, M. Maurer und B. Lichte, „Probabilistic Fault Detection and Handling Algorithm for Testing Stability Control Systems With a Drive-by-Wire Vehicle," in *IEEE International Symposium on Intelligent Control (ISIC),*, Piscataway, NJ, 2011.

[30] H.-H. Braess, „Produktentstehungsprozess," in *Handbuch Kraftfahrzeugtechnik*, Wiesbaden, Springer Fachmedien, 2016, pp. 1257-1369.

[31] I. Göpfert und M. Schulz, „Zukünftige Neuprodukt- und Logistikentwicklung am Beispiel der Automobilindustrie," in *Logistik der Zukunft - Logistics for the Future*, Wiesbaden, Springer Gabler, 2016, pp. 269-291.

[32] F. Wolf, Fahrzeuginformatik: Eine Einführung in die Software- und Elektronikentwicklung aus der Praxis der Automobilindustrie, Wiesbaden: Springer Fachmedien, 2018.

[33] S. H. Reinhard Höhn, „Das V-Modell XT - Grundlagen, Methodik und Anwendungen," 2008. [Online]. Available: http://www.springer.com/de/book/9783540302490. [Zugriff am 31 3 2018].

[34] S. Hakuli und M. Krug, „Virtuelle Integration," in *Handbuch Fahrerassistenzsysteme*, Wiesbaden, Springer-Fachmedien, 2015, pp. 125-138.

[35] R. Pfeffer, G. Basedow, N. Thiesen, M. Spadinger, A. Albers und E. Sax, „Automated Driving - Challenges for the Automotive Industry in Product Development with Focus on Process Models and Organizational Structure," in *13th Annual IEEE International Systems Conference, SYSCON*, Orlando, USA, 2019.

[36] H. Palm, J. Holzmann, R. Klein, S.-A. Schneider und D. Gerling, „A Novel Approach on Virtual SystemsPrototyping Based on a Validated, Hierarchical, Modular Library," in *Embedded World 2013*, Nürnberg, 2013.

[37] M. Eigner, „Überblick Disziplin-spezifische und -übergreifende Vorgehensmodelle," in *Modellbasierte virtuelle Produktentwicklung*, Wiesbaden, Springer Vieweg, 2014, pp. 15-52.

[38] S. Höppner, „Vorgehensmodelle für Systementwicklungund Business Engineering," in *Das V-Modell XT*, Wiesbaden, Springer Verlag, 2008, pp. 275 - 406.

[39] P. Link, „Agile Methoden im Produkt-Lifecycle-Prozess – Mit agilen Methoden die Komplexität im Innovationsprozess handhaben,“ in *Komplexitätsmanagement in Unternehmen*, Wiesbaden, Springer Verlag, 2014, pp. 65 - 92.

[40] U. Dombrowski, „Gesaltungsprinzipien,“ in *Lean Development - Aktueller Stand und zukünftige Entwicklungen*, WIesbaden, Springer Verlag, 2015, pp. 21 - 138.

[41] C. Zachäus, B. Müller und G. Meyer, Advanced Microsystems for Automotive Applications 2017, Cham: Springer International Publishing, 2018.

[42] H. Shokry und M. Hinchey, „Model-Based Verification of Embedded Software,“ *Computer*, Bd. 42, Nr. 4, p. 53–59, 2009.

[43] C. Gühmann, J. Riese und K. Rüden, Simulation and Testing for Vehicle Technology, Cham: Springer International Publishing, 2016.

[44] J. Bach, Methoden und Ansätze für die Entwicklung und den Test prädiktiver Fahrzeugregelungsfunktionen, Karlsruhe: Dissertation, Karlsruhe Institut für Technologie, 2018.

[45] J. Schäuffele und J. Zurawka, „Kernprozess zur Entwicklung von elektronischen Systemen und Software,“ in *Automotive Software Engineering - Grundlagen, Prozesse, Methoden und Werkzeuge effizient einsetzen*, Wiesbaden, Springer Vieweg, 2016, pp. 155 - 206.

[46] D. Schramm, M. Hiller und R. Bardini, Modellbildung und Simulation der Dynamik von Kraftfahrzeugen, Wiesbaden: Springer Vieweg, 2013.

[47] F. Matthies, Beitrag zur Modellbildung von Antriebsträngen für Fahrbarkeitsuntersuchungen, Berlin: epubli GmbH, 2013.

[48] C. Günther, Beitrag zur Co–Simulation in derGesamtsystementwicklung des Kraftfahrzeugs, München: Dissertation, TU München, 2017.

[49] M. Hirz, W. Dietrich, A. Gfrerrer und J. Lang, Integrated Computer-Aided Design in Automotive Development - Development Processes, Geometric Fundamentals, Methods of CAD, Knowledge-Based Engineering Data Management, Berlin: Springer-Verlag Berlin Heidelberg, 2013.

[50] M. Viehof und H. Winner, Stand der Technik und der Wissenschaft: Modellvalidierung im Anwendungsbereich der Fahrdynamiksimulation, Darmstadt: Technische Universität Darmstadt, 2017.

[51] M. Holder, P. Rosenberger, H. Winner, V. Makkapati, M. Maier, H. Schreiber, Z. Magosi, T. D'hondt, Z. Slavik, O. Bringmann und W. Rosenstiel, „Measurements revealing Challenges in RadarSensor Modeling for Virtual Validation ofAutonomous Driving," in *21st International Conference on Intelligent Transportation Systems (ITSC)*, Hawaii, USA, 2018.

[52] P. Cao, W. Wachenfeld und H. Winner, „Perception sensor modeling for virtual validation of automated driving," *it - Information Technology*, Bd. 57(4), 2015.

[53] J. Thieling und J. Roßmann, „Highly-Scalable and Generalized Sensor Structures for Efficient Physically-Based Simulation of Multi-Modal Sensor Networks," in *Twelfth International Conference on Sensing Technology (ICST)*, Limerick, Irland, 2018.

[54] IPG Automotive GmbH, CarMaker Reference Manual 9.0, Karlsruhe, 2020.

[55] F. Schuldt, Ein Beitrag für den methodischen Test von automatisierten Fahrfunktionen mit Hilfe von virtuellen Umgebungen, Braunschweig: Dissertation, TU Braunschweig, 2016.

[56] W. D. Käppler, „Testing Roadmap: Durchführung, Analyse und Bewertung von Fahrversuchen," in *Smart Vehicle Handling - Test und Evaluation in der Fahrzeugtechnik*, Wiesbaden, Springer Vieweg, 2015, pp. 49 - 102.

[57] A. G. Ulsoy, H. Peng und M. Cakmakci, Automotive Control Systems, New York, USA: Cambridge University Press, 2012.

[58] G. Baumann und M. Brost, „Testverfahren für Elektronik und Embedded Software in der Automobilentwicklung," in *MBEES: Modellbasierte Entwicklung eingebetteter Systeme IV*, Dagstuhl, Deutschland, 2008.

[59] S. Schwab, T. Leichsenring, M. Zofka und T. Bär, „Durchgängige Testmethode für Assistenzsysteme," *ATZ - Automobiltechnische Zeitschrift,* Bd. Volume 116, Nr. 9, pp. 54 - 59, August 2014.

[60] J. Schäuffele und T. Zurawka, „Methoden und Werkzeuge in der Entwicklung," in *Automotive Software Engineering - Grundlagen, Prozesse, Methoden und Werkzeuge effizient einsetzen*, Wiesbaden, Springer Fachmedien, 2016, pp. 207 - 326.

[61] H. Schlingloff, M. Conrad, H. Dörr und C. Sühl, „Modellbasierte Steuergeräteentwicklung für den Automobilbereich," in *Automotive - Safety & Security 2004: Sicherheit und Zuverlässigkeit für automobile Informationstechnik*, Universität Stuttgart, 2004.

[62] R. Isermann, „Anwendungsorientierte Übersicht kommerzieller Fahrzeug-Simulations-Systeme," in *Fahrdynamik-Regelung*, Wiesbaden, Friedr. Vieweg & Sohn Verlag, 2006, pp. 93 - 116.

[63] P. Reinold, N. Meyer, D. Buse, F. Klingler, C. Sommer, F. Dressler, M. Eisenbarth und J. Andert, „Verkehrssimulation im Hardware-in-the-Loop-Steuergerätetest," in *Simulation und Test 2018*, Wiesbaden, Springer Vieweg, 2019, pp. 253 - 269.

[64] R. Isermann, J. Schaffnit und S. Sinsel, „Hardware-in-the-loop simulation for the design and testing of engine-control systems," *Control Engineering Practice,* Bd. 5, Nr. 7, pp. 643 - 654, Mai 1999.

[65] D. Kempf, L. Bonderson und L. Slafer, „Real time simulation for application to ABS development," *SAE International Congress and Exposition,* 1 Februar 1987.

[66] H. Fathy, Z. Filipi, J. Hagena und J. Stein, „Review of hardware-in-the-loop simulation and its prospects in the automotive area," in *Proceedings of SPIE - The International Society for Optical Engineering* , Orlando, USA, 2006.

[67] F. Schmidt und E. Sax, „Funktionaler Softwaretest für aktive Fahrerassistenzsysteme mittels parametrierter Szenario-Simulation," in *Informatik 2009: Im Focus das Leben, Beiträge der 39. Jahrestagung der Gesellschaft für Informatik e.V.* , Lübeck, 2009.

[68] M. Weiskopf, C. Wolhfahrt und A. Schmidt, „Absicherung eines Radarsensors im Systemverbund mit der Hardware-in-the-Loop Testtechnologie," in *Automotive - Safety & Security 2014*, Bonn, Gesellschaft für Informatik e.V., 2015, pp. 29 - 40.

[69] S. Gowdu, M. E. Asghar, R. Stephan, M. Hein, J. Nagel und F. Baumgärtner, „System architecture for installed-performance testing of automotive radars over-the-air," in *2018 IEEE MTT-S International Conference on Microwaves for Intelligent Mobility (ICMIM)*, München, 2018.

[70] M. E. Gadringer, F. M. Maier, H. Schreiber, V. Makkapati, A. Gruber, M. Vorderderfler, D. Amschl, S. Metzner, H. Pflügl, W. Bösch, M. Horn und M. Paulweber, „Radar target stimulation for automotive applications," *IET Radar, Sonar & Navigation,* Bd. 12, Nr. 10, pp. 1096 - 1103, Oktober 2018.

[71] T. Dallmann, J.-J. Mende und S. Wald, „ATRIUM: A Radar Target Simulator for Complex Traffic Scenarios," in *IEEE MTT-S International Conference on Microwaves for Intelligent Mobility (ICMIM)*, München, 2018.

[72] J. Vejlupek, R. Grepl, P. Krejčí, F. Lesák und K. Matouš, „Hardware-In-the-Loop Simulation for Automotive Parking Assistant Control Units," in *Proceedings of the 16th International Conference on Mechatronics - Mechatronika 2014*, Brno, Tschechien, 2014.

[73] M. Feilhauer und J. Haering, „Current Approaches in HiL-Based ADAS Testing," *SAE Int. J. Commer. Veh.*, pp. 63 - 69, 27 September 2016.

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

[75] S. Mueller, D. Hospach, O. Bringmann, J. Gerlach und W. Rostenstiel, „Robustness Evaluation and Improvement for Vision-based Advanced Driver Assistance Systems," in *Proceedings IEEE 18th International Conference on Intelligent Transportation Systems (2015), Bd. 18*, 2015.

[76] B. Schick und S. Schmidt, „Evaluation of Video-Based Driver Assistance Systems with Sensor Data Fusion by Using Virtual Test Driving," in *Proceedings of the FISITA 2012 World Automotive Congress - Volume 8: Vehicle Designand Testing (II)*, Heidelberg, Springer, 2013, pp. 1363 - 1375.

[77] R. Pfeffer und S. Schmidt, „Durchgängige Werkzeugkette zum Testen kamerabasierter Fahrerassistenzsysteme," *ATZelektronik*, Bd. 10, Nr. 6, pp. 74 - 97, November 2015.

[78] M. Haselhoff und S. Hakuli, „ECUs für kamerabasierte Fahrerassistenzsysteme im Closed-Loop-Verfahren," *ATZextra*, Bd. 20, Nr. Supplement 7, pp. 30 - 33, April 2015.

[79] R. Pfeffer und M. Haselhoff, „Video Injection Methods in a Real-world Vehicle for Increasing Test Efficiency," *ATZ worldwide*, pp. 44 - 49, Juli 2016.

[80] A. Riener, M. Jeon, I. Alvarez und A. Frison, „Driver in the Loop: Best Practices in Automotive Sensing and Feedback Mechanisms," in *Automotive User Interfaces*, Cham, Springer, 2017, pp. 295 - 323.

[81] T. Murano, T. Yonekawa, M. Aga und S. Nagiri, „Development of High-Performance Driving Simulator," in *SAE Int. J. Passeng. Cars – Mech. Syst. 2(1):661-669*, 2009.

[82] T. Bock, „Bewertung von Fahrerassistenzsystemen mittels der Vehicle in the Loop-Simulation,“ in *Handbuch Fahrerassistenzsysteme*, Vieweg+Teubner, 2009, pp. 76 - 83.

[83] T. Bock, „Vehicle in the Loop,“ in *ATZ - Automobiltechnische Zeitschrift*, 2008, pp. 10 - 16.

[84] C. Miquet, S. Schwab, R. Pfeffer, M. Zofka, T. Bär, T. Schamm und J. Zöllner, „New test method for reproducible real-time tests of ADAS ECUs: “Vehicle-in-the-Loop” connects real-world vehicles with the virtual world,“ in *5th International Munich Chassis Symposium 2014*, München, 2014.

[85] T. Düser, X-in-the-Loop –ein durchgängiges Validierungsframework für die Fahrzeugentwicklung am Beispiel von Antriebsstrangfunktionen und Fahrerassistenzsystemen, Karlsruhe: IPEK Institut für Produktentwicklung, KIT, 2010.

[86] D. J. Verburg, A. C. M. van der Knaap und J. Ploeg, „VEHIL: developing and testing intelligent vehicles,“ *Intelligent Vehicle Symposium*, pp. 537 - 544, 2002.

[87] S. Surmund, C. Walkowiak, D. Nguyen und M. Beck, „Neue Szenarien für autonome Fahrsysteme,“ *ATZextra*, pp. 42 - 45, Mai 2018.

[88] A. Ebner, Referenzszenarien als Grundlage für die Entwicklung und Bewertung von Systemen der Aktiven Sicherheit, Berlin: Dissertation, TU Berlin, 2014.

[89] PEGASUS Project, „PEGASUS METHOD - An Overview,“ 2019. [Online]. Available: https://www.pegasusprojekt.de/files/tmpl/Pegasus-Abschlussveranstaltung/PEGASUS-Gesamtmethode.pdf. [Zugriff am 4 Januar 2020].

[90] G. Bagschik, T. Menzel, A. Reschka und M. Maurer, „Szenarien für Entwicklung, Absicherung und Test von automatisierten Fahrzeugen,“ in *11. Workshop Fahrerassistenzsysteme*, 2017.

[91] G. Bagschik, T. Menzel, C. Körner und M. Maurer, „Wissensbasierte Szenariengenerierung für Betriebsszenarien auf deutschen Autobahnen," in *12. Workshop Fahrerassistenzsysteme und automatisiertes Fahren, vol. 12*, Walting im Altmühltal, 2018.

[92] S. Geyer, M. Baltzer, B. Franz, S. Hakuli, M. Kauer, M. Kienle, S. Meier, K. Weissgerber, K. Bengler, R. Bruder, F. Flemisch und H. Winner, „Concept and development of a unified ontology forgenerating test and use-case catalogues for assistedand automated vehicle guidance," *IET Intelligent Transport Systems*, Bd. 8, Nr. 3, pp. 183 - 189, 2013.

[93] S. Ulbrich, T. Menzel, A. Reschka und F. Schuldt, „Definition der Begriffe Szene, Situation und Szenario für das automatisierte Fahren," *10. Workshop Fahrerassistenzsysteme FAS*, 2015.

[94] A. Reschka, Fertigkeiten- und Fähigkeitengraphen als Grundlage des sicheren Betriebs von automatisierten Fahrzeugen im öffentlichen Straßenverkehr in städtischer Umgebung, Braunschweig: Dissertation, TU Braunschweig, 2017.

[95] W. Fastenmeier, Autofahrer und Verkehrssituation : neue Wege zur Bewertung von Sicherheit und Zuverlässigkeit moderner Straßenverkehrssysteme, Bonn: Verl. TÜV Rheinland, 1995.

[96] H. Bubb, M. Vollrath, K. Reinprecht, E. Mayer und M. Körber, „Der Mensch als Fahrer," in *Automobilergonomie*, Wiesbaden, Springer Fachmedien, 2015, pp. 67 - 162.

[97] E. Dickmanns, Dynamic Vision for Perceptionand Control of Motion, London: Springer Science+Business Media, 2007.

[98] H. Schneider, Modellierung und Erkennung von Fahrsituationen und Fahrmanövern für sicherheitsrelevante Fahrerassistenzsysteme, TU Chemnitz: Dissertation, 2009.

[99] W. Tölle, Ein Fahrmanöverkonzept für einen maschinellen Kopiloten, Fortschritt Berichte-VDI Reihe 12 Verkehrstechnik Fahrzeugtechnik, 1996.

[100] H.-H. Nagel und W. Enkelmann, „Generic road traffic situations and driver support system," in *5th PROMETHEUS Workshop*, München, 1991.

[101] M. Schreiber, M. Kauer, D. Schlesinger, S. Hakuli und R. Bruder, „Verification of a Maneuver Catalog for a Maneuver-Based Vehicle Guidance System," in *IEEE International Conference on Systems, Man and Cybernetics*, Istanbul, Türkei, 2010.

[102] M. Schreiber, Konzeptionierung und Evaluierung eines Ansatzes zu einer manöverbasierten Fahrzeugführung im Nutzungskontext Autobahnfahrten, TU Darmstadt: Dissertation, 2012.

[103] B. Franz, M. Kauer, S. Geyer und S. Hakuli, „Conduct-by-Wire," in *Handbuch Fahrerassistenzsysteme*, Wiesbaden, Springer Vieweg, 2015, pp. 1111 - 1122.

[104] R. Bajcsy und H.-H. Nagel, „Descriptive and prescriptive languages for mobility tasks. Are they different?," in *Advances in Image Understanding - A Festschrift for Azriel Rosenfeld*, Los Alamitos, Kalifornien, IEEE Computer Society Press, 1996, pp. 28 - 300.

[105] F. Schuldt, F. Saust, B. Lichte, M. Maurer und S. Scholz, „Effiziente systematische Testgenerierung für Fahrerassistenzsysteme in virtuellen Umgebungen," in *AAET - Automatisierungssysteme, Assistenzsysteme und eingebettete Systeme für Transportmittel*, Braunschweig, 2013.

[106] A. Spillner und T. Linz, Basiswissen Softwaretest, Heidelberg: dpunkt Verlag, 2012.

[107] M. Steimle, G. Bagschik, T. Menzel, J. Wendler und M. Maurer, „Ein Beitrag zur Terminologie für den szenarienbasierten Testansatz automatisierter Fahrfunktionen," in *AAET - Automatisiertes und vernetztes Fahren*, 2018.

[108] A. Pütz, A. Zlocki und L. Eckstein, „Absicherung hochautomatisierter Fahrfunktionen mithilfe einer Datenbank relevanter Szenarien," in *11. Workshop Fahrassistenzsysteme und automatisiertes Fahren*, Walting, 2017.

[109] M. Büker, B. Kramer, E. Böde, S. Maelen und M. Fränzle, „Identifikation von Automationsrisiken hochautomatischer Fahrfunktionen in PEGASUS," in *AAET Automatisiertes und vernetztes Fahren* , Braunschweig, ITS mobility e. V. , 2019, pp. 315 - 329.

[110] P. Minnerup, T. Kessler und A. Knoll, „Collecting Simulation Scenarios by Analyzing Physical Test Drives," in *IEEE 18th International Conference on Intelligent Transportation Systems*, Las Palmas, Spanien, 2015.

[111] C. Wolschke, T. Kuhn, D. Rombach und P. Liggesmeyer, „Observation Based Creation of Minimal Test-Suites for Autonomous Vehicles," in *IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, Toulouse, Frankreich, 2017.

[112] U. Lages, M. Spencer und R. Katz, „Automatic Scenario Generation based on Laserscanner Reference Data and Advanced Offline Processing," in *IEEE Intelligent Vehicles Symposium Workshops (IV Workshops)*, Gold Coast, Australien, 2013.

[113] C. Sippl, F. Bock, D. Wittmann, H. Altinger und R. German, „From Simulation Data to Test Cases for Fully Automated Driving and ADAS," in *Testing Software and Systems*, Cham, Schweiz, Springer Nature, 2016, pp. 191 - 208.

[114] P. Bender, J. Ziegler und C. Stiller, „Lanelets: Efficient Map Representation for Autonomous Driving," in *IEEE Intelligent Vehicles Symposium (IV)*, Dearborn, Michigan, USA, 2014.

[115] R. Queiroz, T. Berger und K. Czarnecki, „GeoScenario: An Open DSL for Autonomous Driving Scenario Representation," in *IEEE Intelligent Vehicles Symposium (IV)*, Paris, Frankreich, 2019.

[116] P. Glauner, A. Blumenstock und M. Haueis, „Effiziente Felderprobung von Fahrerassistenzsystemen," in *8. Workshop Fahrerassistenzsysteme*, Walting im Altmühltal, Uni-DAS e.V., 2012, pp. 5 - 14.

[117] H. Winner, W. Wachenfeld und P. Junietz, „Validation and Introduction of Automated Driving," in *Automotive Systems Engineering II*, Cham, Schweiz, Springer International Publishing, 2018, pp. 177 - 196.

[118] T. Winkle, „Entwicklungs- und Freigabeprozess automatisierter Fahrzeuge: Berücksichtigung technischer, rechtlicher und ökonomischer Risiken," in *Autonomes Fahren*, Berlin, Springer Vieweg, 2015, pp. 612 - 635.

[119] L. Eckstein und A. Zlocki, „Safety Potential of ADAS – Combined Methods for an Effective Evaluation," in *23rd International Technical Conference on the Enhanced Safety of Vehicles (ESV)*, Seoul, Südkorea, 2013.

[120] F. Schuldt, A. Reschka und M. Maurer, „A Method for an Efficient, Systematic Test Case Generation for Advanced Driver Assistance Systems in Virtual Environments," in *Automotive Systems Engineering II*, Cham, Schweiz, Springer International Publishing, 2018, pp. 147 - 175.

[121] F. Schuldt, T. Menzel und M. Maurer, „Eine Methode für die Zuordnung von Testfällenfür automatisierte Fahrfunktionen auf X-in-the-Loop Verfahren im modularen virtuellen Testbaukasten," in *10. Workshop Fahrerassistenzsysteme*, Walting, 2015.

[122] M. Tatar und J. Mauss, „Systematic Test and Validation of Complex Embedded Systems," in *Embedded Real Time Software and Systems (ERTS 2014)*, Toulouse, Frankreich, 2014.

[123] D. Hoffmann, Software-Qualität, Heidelberg: Springer Vieweg, 2013.

[124] S. Mitsch, K. Ghorbal und A. Platzer, „On Provably Safe Obstacle Avoidance forAutonomous Robotic Ground Vehicles," in *Robotics: Science and Systems*, Berlin, 2013.

[125] M. Horstmann, Verflechtung von Test und Entwurf für eine verlässliche Entwicklung eingebetteter Systeme im Automobilbereich, Braunschweig: Dissertation, 2005.

[126] E. Minx und R. Dietrich, Autonomes Fahren - Wo wir heute stehen und was noch zu tun ist, Ladenburg: Axel Springer SE, Corporate Solutions, 2015.

[127] H. Winner und W. Wachenfeld, „Absicherung automatischen Fahrens," 2013. [Online]. Available: http://tubiblio.ulb.tu-darmstadt.de/63810/. [Zugriff am 30 August 2019].

[128] BMBF, BMWi und BMVI, „Aktionsplan Forschung für autonomes Fahren," 2019. [Online]. Available: https://www.bmbf.de/upload_filestore/pub/Aktionsplan_Forschung_fuer_autonomes_Fahren.pdf. [Zugriff am 9 September 2019].

[129] T. Woopen, Lampe, B., T. Böddeker und etal., „UNICARagil - Disruptive Modular Architectures for Agile, Automated Vehicle Concepts," in *27th Aachen Colloquium Automobile and Engine Technology 2018*, Aachen, 2018.

[130] ENABLE S3, „Testing & Validation of Highly Auomated Systems - Summary of Results," 2019. [Online]. Available: https://www.tugraz.at/fileadmin/user_upload/Institute/IHF/Projekte/ENABLE-S3_SummaryofResults_May2019.pdf. [Zugriff am 12 Mai 2019].

[131] IMAGINE, „Fact Sheet zum Verbundprojekt Imagine," 2017. [Online]. Available: https://imagine-online.de/fileadmin/user_upload/pdf/IMAGinE_factsheet_171017_druck.pdf. [Zugriff am 5 Mai 2019].

[132] H. Winner, „Quo vadis, FAS?," in *Handbuch Fahrerassistenzsysteme*, Wiesbaden, Springer, 2015, pp. 1167 - 1186.

[133] J. Bortz und C. Schuster, „Wahrscheinlichkeitsverteilungen," in *Statistik für Human- und Sozialwissenschaftler*, Heidelberg, Springer Medizin Verlag, 2005, pp. 61 - 78.

[134] P. Flajolet, D. Gardy und L. Thimonier, „Birthday paradox, coupon collectors, caching algorithms and self-organizing search,“ *Discrete Applied Mathematics*, Bd. 39, Nr. 3, pp. 207 - 229, 11 November 1992.

[135] G. Walz, Lexikon der Mathematik: Band 2, Berlin: Springer Spektrum, 2017.

[136] A. Doumas, „How many trials does it take to collect all different types of a population with probability p?,“ *Journal of Applied Mathematics & Bioinformatics*, Bd. 5, Nr. 3, pp. 1 - 14, 2015.

[137] K. Reif, Sensoren im Kraftfahrzeug, Wiesbaden: Springer Verlag, 2016.

[138] C. Amersbach und H. Winner, „Functional Decomposition: An Approach to Reduce the Approval Effort for Highly Automated Driving,“ *8. Tagung Fahrerassistenz*, 2017.

[139] M. Kienle, B. Franz, H. Winner, K. Bengler, M. Baltzer, F. Flemisch, M. Kauer, T. Weißgerber, S. Geyer, R. Bruder, S. Hakuli und S. Meier, „Concept and development of a unified ontology for generating test and use-case catalogues for assisted and automated vehicle guidance,“ *IET Intelligent Transport Systems*, Bd. 8, Nr. 3, p. 183–189, 2014.

[140] G. Bagschik, T. Menzel, A. Reschka und M. Maurer, „Szenarien für Entwicklung, Absicherung und Test von automatisierten Fahrzeugen,“ in *Workshop Fahrerassistenzsysteme und automatisiertes Fahren, 2017*, p. 17–124.

[141] T. Menzel, G. Bagschik, L. Isensee, A. Schomburg und M. Maurer, „Szenariobeschreibung für die Durchführung in der Simulation am Beispiel von Szenarien auf deutschen Autobahnen,“ in *Zivilprozessordnung und Nebengesetze*, 4., neu bearb. Aufl., Stand der Bearb.: Januar 2015 Hrsg., Berlin, DE GRUYTER, 2015, p. 1–42.

[142] T. Menzel, G. Bagschik, L. Isensee, A. Schomburg und M. Maurer, „Detaillierung einer stichwortbasierten Szenariobeschreibung für die Durchführung in der Simulation am Beispiel von Szenarien auf deutschen Autobahnen," *12. Uni-DAS e.V. Workshop Fahrerassistenz und automatisiertes Fahren,* pp. 15 - 26, 2018.

[143] A. Wagener und R. Katz, „Automated scenario generation for testing advanced driver assistance systems based on post-processed reference laser scanner data," [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-658-21444-9_12#citeas. [Zugriff am 29 9 2018].

[144] A. Bagnall, J. Lines, A. Bostrom, J. Large und E. Keogh, „The great time series classification bake off," *Data Mining and Knowledge Discovery,* Bd. 31, Nr. 3, p. 606–660, 2017.

[145] O. Balci, „Validation, verification, and testing techniques throughout the life cycle of a simulation study," *Annals of Operations Research,* Bd. 53, Nr. 1, p. 121–173, 1994.

[146] M. Ersoy, C. Elbers, D. Wegener, J. Lützow, C. Bachmann und C. Schimmel, „Fahrdynamik," in *Fahrwerkhandbuch - Grundlagen · Fahrdynamik · Fahrverhalten · Komponenten · Elektronische Systeme · Fahrerassistenz · Autonomes Fahren · Perspektiven*, Wiesbaden, Springer Fachmedien , 2017, pp. 51 - 170.

[147] J. Treiterer, D. Clear, D. Howarth, J. Lee und J. Myer, „Investigation of traffic dynamics by aerial photogrammetry techniques," in *Federal Highway Administration*, 1975.

[148] G. Bham und R. Benekohal, „A high fidelity traffic simulation model based on cellular automata and car-following concepts," in *Transportation Research Part C: Emerging Technologies*, Urbana, USA, 2004.

[149] J. Wang, K. Dixon, H. Li und J. Ogle, „Normal Acceleration Behavior of Passenger Vehicles Starting from Rest at All-Way Stop-Controlled Intersections," in *Transportation Research Record Traffic flow theory and highway capacity and quality of service 2004*, Washington, USA, 2004.

[150] S. Deligianni, M. Quddus, A. Morris, A. Anvuur und S. Reed, „Analyzing and Modeling Drivers' Deceleration Behavior from Normal Driving," in *Deligianni, S. P., Quddus, M., Morris, A., Anvuur, A., & Reed, S. (2017). Analyzing and Modeling Drivers' Deceleration Behavior from Normal Driving. Transportation Research Record: Journal of the Transportation Research Board, 2663*, 2017.

[151] R. Akcelik und D. C. Biggs, „Acceleration profile models for vehicles in road traffic," in *Transportation Science*, 1987.

[152] M. Treiber und A. Kesting, Verkehrsdynamik und -simulation: Daten, Modelle und Anwendungen der Verkehrsfiussdynamik, Heidelberg: Springer Verlag, 2010.

[153] H. Burg und A. Moser, „D1 Fachbegriffe nach DIN 75204 Straßenfahrzeuge," in *Handbuch Verkehrsunfallrekonstruktion - Unfallaufnahme · Fahrdynamik · Simulation*, Wiesbaden, Springer Vieweg, 2017, pp. 941 - 970.

[154] A. Weiser, Probabilistische Vorhersage von Fahrstreifenwechseln für hochautomatisiertes Fahren auf Autobahnen, Karlsruhe: Dissertation, KIT Scientific Publishing, 2017.

[155] N. Sledge und K. Marshek, „Comparison of ideal vehicle lane-change trajectories," in *SAE Transactions*, 1997.

[156] W. Nelson, „Continuous-Curvature Paths for Autonomous Vehicles," in *International Conference on Robotics and Automation*, Scottsdale, USA, 1989.

[157] J. D. Chovan, L. Tijerina, G. Alexander und D. L. Hendricks, „Examination of lane change crashes and potential IVHS countermeasures," in *National Highway Traffic Safety Administration*, Cambridge, USA, 1994.

[158] J. Freyer, Vernetzung von Fahrerassistenzsystemen zur Verbesserung des Spurwechselverhaltens von ACC, Göttingen: Cuvillier Verlag, 2008.

[159] R. Kohlhaas, T. Bittner, T. Schamm und M. Zöllner, „Semantic State Space for High-level Maneuver Planning in Structured Traffic Scenes," in *17th International Conference on Intelligent Transportation Systems (ITSC)*, Qingdao, China, 2014.

[160] D. Petrich, D. Azarfar, F. Kuhnt und M. Zöllner, „The fingerprint of a traffic situation: A semantic relationship tensor for situation description and awareness," in *21st International Conference on Intelligent Transportation Systems (ITSC)*, Maui, USA, 2018.

[161] J. Xie, A. R. Hilal und D. Kulić, „Driving Maneuver Classification: A Comparison of Feature Extraction Methods," *IEEE Sensors Journal*, Bd. 18, pp. 4777-4784, 6 2018.

[162] J. Cervantes-Villanueva, D. Carrillo-Zapata, F. Terroso-Saenz, M. Valdes-Vela und A. F., „Vehicle Maneuver Detection with Accelerometer-Based Classification," *Sensors (Basel, Switzerland)*, 2016.

[163] C. Woo und D. Kulić, „Manoeuvre segmentation using smartphone sensors," in *2016 IEEE Intelligent Vehicles Symposium (IV)*, 2016.

[164] Z. Camlica, A. Hilal und D. Kulić, „Feature abstraction for driver behaviour detection with stacked sparse auto-encoders," in *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2016.

[165] C. Arroyo, L. M. Bergasa und E. Romera, „Adaptive fuzzy classifier to detect driving events from the inertial sensors of a smartphone," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, 2016.

[166] Y. Zheng und J. Hansen, „Lane-Change Detection From Steering Signal Using Spectral Segmentation and Learning-Based Classification," *IEEE Transactions on Intelligent Vehicles*, Bd. PP, pp. 1-1, 5 2017.

[167] Y. Zheng, A. Sathyanarayana und J. H. L. Hansen, „Threshold based decision-tree for automatic driving maneuver recognition using CAN-Bus signal," in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2014.

[168] Y. Zheng, A. Sathyanarayana und J. Hansen, „Non-Uniform Time Window Processing of In-Vehicle Signals for Maneuvers Recognition and Route Recovery," *SAE Technical Papers*, Bd. 2015, 4 2015.

[169] S. Li, Y. Liao, W. Wang, B. Cheng und F. Chen, „Lane change maneuver recognition via vehicle state and driver operation signals—Results from naturalistic driving data," in *IEEE Intelligent Vehicles Symposium (IV)*, Seoul, 2015.

[170] H. Sun, W. Deng, C. Su und J. Wu, „Robust Traffic Vehicle Lane Change Maneuver Recognition," in *SAE Technical Paper 2017-01-0110*, 2017.

[171] Z. Zheng, P. Li, M. Hu, W. Zhang und Y. Li, „Drivers' Lane-Changing Maneuvers Detection in Highway," in *MMESE 2016: Man-Machine-Environment System Engineering*, Bd. 406, 2016, pp. 21-29.

[172] R. Gruner, P. Henzler, G. Hinz, C. Eckstein und A. Knoll, „Spatiotemporal representation of driving scenarios and classification using neural networks," in *2017 IEEE Intelligent Vehicles Symposium (IV)*,, Los Angeles, CA, 2017.

[173] P. Eckstein, Repetitorium Statistik: Deskriptive Statistik. Stochastik. Induktive Statistik. Mit Klausuraufgaben und Lösungen, Wiesbaden: Springer Verlag, 2013.

[174] J. Lin, E. Keogh, L. Wei und S. Lonardi, „Experiencing SAX: a Novel Symbolic Representation of Time Series," *Data Min Knowl Disc* , p. 107–144, 2007.

[175] J. Lin, R. Khade und Y. Li, „Rotation-invariant similarity in time series using bag-of-patterns representation," *Journal of Intelligent Information Systems*, pp. 287 - 315, 2012.

[176] H. Sakoe und S. Chiba, „Dynamic programming algorithm optimization for spoken word recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* pp. 43 - 49, 1978.

[177] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria und E. Keogh, „Searching and mining trillions of time series subsequences under dynamic time warping," *18th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '12). Association for Computing Machinery,* pp. 262 - 270, 2012.

[178] D. Berndt und J. Clifford, „Using Dynamic Time Warping to Find Patterns in Time Series," *Computer Science. KDD Workshop,* 1994.

[179] K. Dietmayer, A. Kirchner und N. Kämpchen, „Fusionsarchitekturen zur Umfeldwahrnehmung für zukünftige Fahrerassistenzsysteme," in *Fahrerassistenzsysteme mit maschineller Wahrnehmung*, Springer-Verlag, 2005, pp. 59 - 88.

[180] H. Winner und M. Schopper, „Adaptive Cruise Control," in *Handbuch Fahrerassistenzsysteme*, Springer-Verlag, 2015, pp. 851 - 892.

[181] A. Weiser, Probabilistische Vorhersage von Fahrstreifenwechseln für hochautomatisiertes Fahren auf Autobahnen, Karlsruhe Institut für Technologie: KIT Scientific Publishing, 2019.

[182] R. Pfeffer, F. Pistorius und E. Sax, „Anwendung von Clusteringverfahren für die Extraktion und Reduktion von Testfällen aus aufgezeichneten Fahrszenarien," in *CVT Symposium*, Kaiserslautern, 2020.

[183] F. Witte, „Teststufen," in *Testmanagement und Softwaretest: Theoretische Grundlagen und praktische Umsetzung*, Wiesbaden, Springer Verlag, 2015, pp. 67 - 80.

[184] T. O'Connell und A. Koehler, Forecasting, Time Series, and Regression: An Applied Approach, Thomson Brooks/Cole, 2005.

[185] A. Geiger, P. Lenz, C. Stiller und R. Urtasun, „Vision meets robotics: the KITTI dataset," *The International Journal of Robotics Research,* pp. 1231-1237, 2013.

[186] IPG Automotive GmbH, „Sensors," in *Reference Manual CarMaker Version 8.1,* Karlsruhe, 2020, pp. 621 - 723.

[187] P.-N. Tan, M. Steinach, A. Karpatne und V. Kumar, Introduction to Data Mining, Pearson Education Limited, 2014.

[188] Z. Lipton, J. Berkowitz und C. Elkan, „A Critical Review of Recurrent Neural Networks for Sequence Learning," *arXiv:1506.00019,* 2015.

[189] M. Weber, Untersuchungen zur Anomalieerkennung in automotive Steuergeräten durch verteilte Observer mit Fokus auf die Plausibilisierung von Kommunikationssignalen, Dissertation, Karlsruhe Institut für Technologie, 2019.

[190] R. Pfeffer, S. Schmidt und E. Sax, „Realdatenbasierte simulationsgestützte Absicherung hochautomatisierter Fahrfunktionen," *ATZ Elektronik,* 11 2019.

[191] R. Krajewski, J. Bock, K. L. und L. Eckstein, „The highD Dataset: A Drone Dataset of Naturalistic Vehicle Trajectories on German Highways for Validation of Highly Automated Driving Systems," in *IEEE 21st International Conference on Intelligent Transportation Systems (ITSC,* Maui, Hawaii, USA, 2018.

[192] H. Caesar, V. Bankiti, A. Lang, S. Vora, V. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan und O. Beijbom, „A multimodal dataset for autonomous driving," in *arXiv preprint arXiv:1903.11027,* 2019.

[193] Y. Yan, Y. Mao und B. Li, „SECOND: Sparsely EmbeddedConvolutional Detection," in *Sensors 18(10):3337,* 2018.

[194] Y. Zhou und O. Tuzel, „VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection," in *arXiv Computer Vision and Pattern Recognition,* 2017.

[195] H. Chiu, A. Prioletti, J. Li und J. Bohg, „Probabilistic 3D Multi-Object Tracking for Autonomous Driving," in *arXiv: Computer Vision and Pattern Recognition*, 2020.

[196] A. Geiger, P. Lenz und R. Urtasun, „Are we ready for autonomous driving? The KITTI vision benchmark suite," in *IEEE Conference on Computer Vision and Pattern Recognition*, Providence, Rhode Island, USA, 2012.

[197] M. Everingham und J. Winn, „The PASCAL Visual Object Classes Challenge," in *VOC2011*, 2011.

[198] K. Bernardin und R. Steifelhagen, „Evaluating Multiple Object Tracking Performance: The CLEAR MOT Metrics," in *EURASIP Journal on Image and Video Processing*, 2008.

[199] R. Pfeffer, J. He und E. Sax, „Development and implementation of a concept for the meta description of highway driving scenarios with focus on interactions of road users," in *6th International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS)*, Prag, Tschechien, 2020.

[200] W. McCulloch und W. Pitts, „A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, pp. 115 - 133, 1943.

[201] C. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.

[202] F. Rosenblatt, „The perceptron: a probabilistic model for information storage and organization in the brain," *Psychological Review*, 1958.

[203] E. Tzanakou, Supervised and Unsupervised Pattern Recognition: Feature Extraction and Computational Intelligence, CRC Press, 2017.

[204] A. Graves, Supervised Sequence Labelling with Recurrent Neural Networks, Springer Verlag, 2012.

[205] S. Hochreiter und J. Schmidhuber, „Long Short-Term Memory," *Neural computation*, Bd. 9, pp. 1735-80, 1997.

[206] K. Greff, R. Srivastava, J. Koutnik, B. Steunebrink und J. Schmidhuber, „LSTM: A Search Space Odyssey," in *IEEE Transactions on neural networks and learning systems, Vol. 28., No. 10*, 2017.

[207] H. Winner, G. Prokop und M. Maurer, Automotive Systems Engineering II, Cham: Springer International Publishing, 2018, p. 196.

[208] W. Gerke, Technische Assistenzsysteme, Berlin u.a.: De Gruyter Oldenbourg, 2015, pp. X, 372 S.

[209] F. Amper, „Von Model-in-the-Loop zu Concept-in-the-Loop," *ATZextra*, Bd. 23, Nr. 2, pp. 38 - 41, Mai 2018.

[210] R. Reiter, „On closed world data bases," in *Logic and Data Bases*, Boston, Springer, Boston, MA, 1978, pp. 55 - 76.

[211] H.-O. Georgii, Statistik: Einführung in die Wahrscheinlichkeitstheorie und Statistik, Walter de Gruyter, 2008.

[212] I. Stewart, Größen der Mathematik: 25 Denker, die Geschichte schrieben, rowohlt, 2018.

[213] P. Scholz, „Echtzeit, Echtzeitsysteme, Echtzeitbetriebssysteme," in *Softwareentwicklung eingebetteter Systeme: Grundlagen, Modellierung, Qualitätssicherung*, Heidelberg, Springer-Verlag, 2006, pp. 39 - 74.

[214] B. Stegmann, Ein Beitrag zur Modellierungsgenauigkeit im Bereich eindimensionaler Simulation von Pkw-Kühlsystemen, Wiesbaden: Springer Verlag, 2016.

[215] P. Löw, R. Pabst und E. Petry, Funktionale Sicherheit in der Praxis: Anwendung von DIN EN 61508 und ISO/DIS 26262 bei der Entwicklung von Serienprodukten, Heidelberg: dpunkt Verlag, 2011.

[216] B. Graham, „Spatially-Sparse-Convolutional-Neural-Network,“ in *arXiv: Computer Vision and Pattern Recognition*, 2014.

[217] R. Agrahari, „SECOND for Lyft 3d object detection challenge,“ in *https://github.com/pyaf/second.pytorch*, 2018.

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

ACC	Adaptive Cruise Control
ADAS	Advanced Driver Assistance Systems
CAN	Controller Area Network
DAS	Driver Assistance Systems
DDTW	Derivative Dynamic Time Warping
DTW	Dynamic Time Warping
ECU	Electronic Control Unit
ESC	Electronic Stability Control
HiL	Hardware-in-the-Loop
KNN	Künstliches neuronales Netz
LSTM	Long Short-Term Memory
MiL	Model-in-the-Loop
OEM	Original Equipment Manufacturer
PEP	Produktentstehungsprozess
SiL	Software-in-the-Loop
SOP	Start of Production
ViL	Vehicle-in-the-Loop
XiL	X-in-the-Loop

WDTW Weighted Dynamic Time Warping

Eigene Veröffentlichungen

Pfeffer, R.; Ukas, P.; Sax, E.: „Potential of virtual test environments for the development of highly automated driving functions using neural networks“ in: Fahrerassistenzsysteme 2018: Von der Assistenz zum automatisierten Fahren 4. Internationale ATZ-Fachtagung Automatisiertes Fahren. Ed.: T. Bertram, 203–211, Springer Fachmedien Wiesbaden, 2019.

Pfeffer, R.; Basedow, G. N.; Thiesen, N. R.; Spadinger, M.; Albers, A.; Sax, E.: „Automated Driving - Challenges for the Automotive Industry in Product Development with Focus on Process Models and Organizational Structure“ in 13th Annual IEEE International Systems Conference (SysCon 2019), Orlando, FL, USA, 2019.

Ahn, N.; Pfeffer, R.: „Artificial Intelligence in the Development of Autonomous Driving Functions“ in ATZ Worldwide 121, 42–45, 2019.

Pfeffer, R.; Ahn, N.: „Potential of Training Neural Networks Using Virtual Environments“ Internationaler VDI-Kongress ELIV (Electronics in Vehicles), Bonn, 2019. (1. Platz VDI Auto Electronic Excellence Award)

Pfeffer, R.: „Training and validation of neural networks in virtual environments“ in: Automatisiertes Fahren 2019: Von der Fahrerassistenz zum autonomen Fahren 5. Internationale ATZ-Fachtagung. Ed.: T. Bertram, 135–144, Springer Fachmedien Wiesbaden, 2020.

Pfeffer, R.; Schmidt, S.; Sax, E.: „Real Data Based Validation of Highly Automated Driving Functions Using Simulation Methods“ in ATZelectronics worldwide, 14 (11), 24–29, 2019.

Pfeffer, R.; Bredow, K.; Sax, E.: „Trade-Off Analysis Using Synthetic Training Data for Neural Networks in the Automotive Development Process“ in Proceedings of IEEE Intelligent Transportation Systems Conference - ITSC 2019, Auckland, Neuseeland, 2019.

Pfeffer, R.; Pistorius, F.; Sax, E.: „Anwendung von Clusteringverfahren für die Extraktion und Reduktion von Testfällen aus aufgezeichneten Fahrszenarien," in Proceedings of CVT Symposium, Kaiserslautern, 2020 (eingereicht).

Pfeffer, R.; He, J.; Sax, E.: „Development and implementation of a concept for the meta description of highway driving scenarios with focus on interactions of road users" in Proceedings of 6th International Conference on Vehicle Technology and Intelligent Transport Systems – VEHITS, Prag, Tschechien, 2020.

Betreute Master- und Bachelorarbeiten

Pokrandt, Steffen: *Konzept und Entwicklung einer intelligenzten adaptiven Fahrzeuglängsregelung,* Masterthesis, Karlsruhe Institut für Technologie, 2018

Franz, Dennis: *Untersuchung des Potentials einer Synergie aus realem Fahrversuch und Simulation am Beispiel einer visionsbasierten Fusionsmethode zur Fahrspurzuordnung eines Fahrzeugs,* Masterthesis, Karlsruhe Institut für Technologie, 2018

Kaiser, Manuel: *Künstliches Lernen: Trainingsdaten aus dem virtuellen Fahrversuch für Szenarienklassifizierung mit Deep Learning Algorithmen*, Masterthesis, Karlsruhe Institut für Technologie, 2018

Nägele, Ann-Therese: *Framework zur Testfallklassifikation für hochautomatisierte Fahrfunktionen,* Bachelorthesis, Karlsruhe Institut für Technologie, 2018

Bredow, Kai: *Daten aus dem virtuellen Fahrversuch für das Training neuronaler Netze zur Objekterkennung,* Masterthesis, Karlsruhe Institut für Technologie, 2018

Thiesen, Nina: *Die Entwicklungsherausforderungen des hochautomatisierten Fahrens mit Fokus auf die deutsche Automobilindustrie,* Masterthesis, Karlsruhe Institut für Technologie, 2018

Wang, Jiawei: *Implementierung und Evaluierung von Verfahren zur Zeitreihenanalyse und Szenarioklassifikation aus Fahrzeugdaten*, Masterthesis, Karlsruhe Institut für Technologie, 2019

Chen, Jiaxi: *Evaluierung von Deep-Learning-Algorithmen zur Identifikation von Fahrmanövern in aufgezeichneten Fahrdaten*, Masterthesis, Karlsruhe Institut für Technologie, 2019

He, Jingyu: *Clustering of Scenarios from Vehicle Data using Machine Learning Methods*, Masterthesis, Karlsruhe Institut für Technologie, 2019

Khalef, Mohamed Amine: *Entwicklung und Implementierung eines Konzepts zur Metabeschreibung von Fahrszenarien mit Fokus auf die Interaktion mit anderen Verkehrsteilnehmern*, Masterthesis, Karlsruhe Institut für Technologie, 2019

Schweizer, Dennis: *Konzeptionierung und Umsetzung eines Clusteringverfahrens für Fahrszenarien basierend auf Machine Learning*, Masterthesis, Karlsruhe Institut für Technologie, 2019

Baumann, Daniel: *Entwurf und Implementierung eines Reinforcement Algorithmus für die Generierung von Testszenarien für das hochautomatisierte Fahren*, Masterthesis, Karlsruhe Institut für Technologie, 2019

Heumesser, Bettina: *Entwicklung und Implementierung eines szenarienbasierten Testkonzeptes für einen Autobahnpiloten in einer Simulationsumgebung*, Bachelorthesis, Karlsruhe Institut für Technologie, 2020

Sauerer, Fabian: *Klassifikation von Fahrmanövern in resimulierten aufgezeichneten Zeitreihensignalen unter Nutzung von Deep Learning*, Bachelorthesis, Karlsruhe Institut für Technologie, 2020

Mersch, Simon: *Extraktion von Fahrmanövern aus aufgezeichneten Fahrdaten für automatisierte Fahrfunktionen mittels künstlicher neuronaler Netze*, Masterthesis, Karlsruhe Institut für Technologie, 2020

Jost, Daniel: *Überführung von aufgezeichneten Verkehrsszenarien in die Replay-Simulation mit Machine-Learning-Algorithmen*, Bachelorthesis, Karlsruhe Institut für Technologie, 2020

Metzger, Philipp: *Erweiterung eines Datenmodells zur Übertragung von Fahrdaten eines automatisierten Fahrzeugs in eine virtuelle Testumgebung um Informationen der statischen Umgebung mittels Machine-Learning-Methoden*, Bachelorthesis, Karlsruhe Institut für Technologie, 2020

Tong, Huimin: *Development of a Testing Environment for Advanced Driver Assistance Systems in HiL*, Masterthesis, Karlsruhe Institut für Technologie, 2020