Glaubwürdigkeit und Einsatz des szenariobasierten X-in-the-Loop-Tests für Fahrerassistenzsysteme

Zur Erlangung des akademischen Grades eines

DOKTORS DER INGENIEURWISSENSCHAFTEN (Dr.-Ing.)

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

angenommene

DISSERTATION

von

M.Sc. Felix Manuel Reisgys
geb. in Weinheim

Tag der mündlichen Prüfung: 07.12.2023

Hauptreferent: Prof. Dr.-Ing. Eric Sax
Korreferent: Prof. Dr. Jürgen Pannek

Kurzfassung

Fahrerassistenzsysteme tragen gerade im Bereich der Nutzfahrzeuge zur Verkehrssicherheit bei. Darüber hinaus bietet das hochautomatisierte Fahren neue Geschäftsmodelle. Eine zentrale Herausforderung bei der Entwicklung dieser Systeme ist die steigende Breite und Tiefe der Testfälle. Bereits in heutigen Entwicklungsprojekten stoßen vorhandene Realtestkapazitäten an ihre Grenzen. Daher sind neue Methoden zum Test von Fahrerassistenzsystemen erforderlich.

Die Kombination aus szenariobasiertem Testen und X-in-the-Loop-Testumgebungen ist ein vielversprechender Ansatz. In dieser Dissertation werden drei Einsatzmöglichkeiten des szenariobasierten XiL-Tests in einem Serien-Entwicklungsprojekt eingeführt und diskutiert. Als besonders geeignet wird hierbei der Einsatz beim Software-Qualifizierungstest bewertet und in einem Prozessentwurf weiter detailliert. Schwerpunkt ist die Zuordnung von Szenarien auf Testumgebungen mit dem Ziel, die Testabdeckung, die Glaubwürdigkeit der Testergebnisse und die Effizienz der Testdurchführung zu optimieren.

Ein entscheidender Prozessschritt ist die sogenannte Glaubwürdigkeitsbewertung. Diese bewertet ein konkretes Szenario in einer spezifischen Testumgebung und besteht je nach Konfidenzanforderung aus den drei Schritten Prädiktion, Plausibilisierung und Validierung. In der Prädiktion werden mithilfe von Unsicherheitsmodellen für drei Subsysteme der XiL-Umgebung und einer Monte-Carlo-Simulation Testergebnis-Verteilungen generiert und mit einem Multinomialansatz Konfidenzintervalle ermittelt. Die Plausibilisierung prüft mithilfe von Pass/Fail-Kriterien und Szenariodistanzmaßen die Äquivalenz einzelner Testausführungen in XiL und Realtest. Bei der Validierung findet ein statistischer Abgleich der Testergebnis-Verteilungen aus XiL und Realtest mithilfe des Barnard-Tests statt.

Die Glaubwürdigkeitsbewertung wird auf Basis von Software-in-the-Loop-Daten eines Entwicklungsprojekts für Fahrerassistenzsysteme für insgesamt sieben konkrete Szenarien evaluiert.

Abstract

Advanced Driver Assistance Systems contribute to traffic safety, especially when it comes to commercial vehicles. Furthermore, highly automated driving offers new business cases. A major challenge in development of such systems is the increasing width and depth of test cases. Even today's development projects reach the limits of existing real world test capacities. Consequently, new methods to test Advanced Driver Assistance Systems are required.

Combining scenario-based testing with X-in-the-Loop test environments is a promising approach. In this dissertation, three deployment options of scenario-based XiL test in a series development project are introduced and discussed. Especially the application in software qualification test is evaluated to be suitable and is further specified. The focus is on the assignment of scenarios to test environments while optimizing test coverage, test result credibility and test execution efficiency.

A decisive process step is so-called credibility evaluation. It evaluates a concrete scenario in a specific test environment and consists of the three steps prediction, plausibilization and validation depending on the confidence requirement. In prediction, uncertainty models of three XiL subsystems and a Monte Carlo simulation yield test result distributions and allow computation of multinomial confidence intervals. Plausibilization evaluates the equivalence of single test executions in XiL and real world using pass/fail criteria and scenario distance measures. In the validation step, a statistical comparison of test results of XiL and real world is performed using the Barnard test.

The credibility evaluation is assessed based on Software-in-the-Loop data of a development project for Advanced Driver Assistance Systems using seven concrete scenarios.

Danksagung

Diese Dissertation ist das Resultat einer intensiven, teils herausfordernden, aber stets spannenden und gewinnbringenden Zeit als Doktorand bei der Daimler Truck AG in Kooperation mit dem Institut für Technik der Informationsverarbeitung (ITIV) am Karlsruher Institut für Technologie. Ohne die Unterstützung von vielen Seiten wäre die Arbeit in dieser Form nicht möglich gewesen.

Zuallererst möchte ich meinem Doktorvater Prof. Dr.-Ing. Eric Sax einen großen Dank für die Möglichkeit zur Promotion sowie die hervorragende Betreuung und die stets sehr hilfreichen Reviews über den gesamten Zeitraum aussprechen. Hierdurch konnte ich mich sowohl fachlich als auch persönlich stark weiterentwickeln.

Ebenfalls danke ich Prof. Dr. Jürgen Pannek für die Übernahme des Korreferats und die wertvollen inhaltlichen Rückmeldungen und Diskussionen zum Ende meiner Promotionszeit. Des Weiteren danke ich dem Prüfungsvorsitzenden Prof. Dr.-Ing. Sören Hohmann sowie den Prüfern Prof. Dr.-Ing. Mike Barth und Prof. Dr.-Ing. Ahmet Cagri Ulusoy.

Ein besonderer Dank geht an meinen Teamleiter Ingo Scherhaufer und an meinen akademischen Betreuer Dr. Andreas Schwarzhaupt, die mich seitens Daimler Truck sehr gut unterstützt haben und so die Anwendung meiner Forschungsansätze im Serienumfeld ermöglicht haben. Gemeinsam konnten wir gerade in der Anfangsphase wichtige Hebel in Bewegung setzen, die für den späteren Erfolg der Dissertation essentiell waren.

Darüber hinaus danke ich allen weiteren Kollegen bei Daimler Truck, die mit ihrer Unterstützung und Expertise zum Gelingen beigetragen haben, darunter Dr. Roland Werner, Dr. Darko Meljnikov und Boas Schray. Für den Beitrag durch studentische Arbeiten danke ich Thomas Müller, Frederik Janßen, Lennard Sommer, Mark Rohrmann und Phillip Freimann.

Ein weiterer großer Dank für wertvolle Gespräche und Impulse geht an die Kolleginnen und Kollegen der Doktorandengruppe Sax und insbesondere an Johannes Plaum, Thilo Braun und Christian Steinhauser.

Zuletzt danke ich ganz besonders meinen Eltern, meinen Geschwistern und Judy für die vielfältige Unterstützung, sei es durch Ermutigung, praktische Hilfe oder Gebete. Ohne euch wäre die Dissertation so nicht möglich gewesen!

Stuttgart, Januar 2024 *Felix Reisgys*

> *Denn Gott ist es, der in euch sowohl das Wollen als auch das Vollbringen bewirkt, nach seinem Wohlgefallen.*
>
> Die Bibel, Philipperbrief, Kapitel 2, Vers 13

Inhaltsverzeichnis

1 Einleitung

1.1 Herausforderungen beim Test von Fahrautomatisierung

Die globale Automobilindustrie sieht sich seit einigen Jahren einem tiefgreifenden technologischen Wandel ausgesetzt. Unter dem Schlagwort „CASE“ werden die vier Megatrends „Connectivity“, „Autonomous“, „Shared & Services“ und „Electric“ zusammengefasst [1]. Einige Autoren stellen die Innovationen der CO2-neutralen Antriebstechnologie mithilfe von E-Mobilität [2] und Brennstoffzelle [3] heraus. Darüber hinaus wird die steigende Digitalisierung und Automatisierung des Fahrzeugs betont [4]. Gerade die Fahrautomatisierung bietet enormes Potential für Gesellschaft, Kunden und die Automobilindustrie. McKinsey geht davon aus, dass sich der wirtschaftliche Umsatz durch Fahrautomatisierung von PKW zwischen 2022 und 2035 mehr als versiebenfacht. Die gleiche Studie prognostiziert, dass bereits im Jahr 2030 zwischen 4 und 20 % der verkauften Fahrzeuge mit Systemen nach SAE Level 3 und höher (siehe 2.1.1.1) [5] ausgestattet sein werden. [6]

Die genannten Trends betreffen die Nutzfahrzeugbranche in gleichem Maße. Insbesondere der steigende Verbreitungsgrad von Fahrerassistenzsystemen (FAS) bietet bei Nutzfahrzeugen signifikantes Potential zur Verringerung von Unfallfolgen. Diese sind aufgrund der höheren Fahrzeugmasse (bis zu 44 t Gesamtmasse in Deutschland [7]) potentiell schwerwiegender als bei Kollisionen zwischen PKW. Daneben sind auch Einsparungen beim Kraftstoffverbrauch mithilfe von FAS möglich [8]. Heutige Serien-LKW sind bereits mit FAS bis SAE Level 2 sowie aktivem Notbrems- und Abbiegeassistenten bestellbar. Darüber hinaus sind weitere warnende bzw. hinweisende Systeme verfügbar [9]. Auch seitens des Gesetzgebers findet eine immer stärkere Priorisierung von FAS statt: Bereits seit 2015 sind Notbremsassistenten für neu zugelassene LKW verpflichtend [10]. Systeme wie Abbiegeassistent oder Verkehrszeichen-

assistent werden im Rahmen der General Safety Regulation (GSR) ab dem Jahr 2024 für alle neu zugelassenen LKW zur Pflicht [11].

Neben der reinen Fahrerassistenz sind bereits in wenigen Jahren serienreife Nutzfahrzeuge höherer Automatisierungslevels zu erwarten. Für Fahrzeuge ab SAE Level 4 ist keine Anwesenheit eines Fahrers mehr erforderlich. Dies ermöglicht neue Geschäftsmodelle in der Transportindustrie, weshalb bei zukünftigen Entwicklungen nicht mehr nur der Sicherheitsgewinn für Kunden im Vordergrund steht. [12]

Unabhängig vom Automatisierungslevel stellt die Absicherung von Fahrautomatisierung bereits heute eine große Herausforderung dar. Einerseits führt die wachsende Anzahl an FAS von sich aus zu einer größeren Anzahl an Testfällen [13]. Zudem existiert im Nutzfahrzeugbereich eine hohe Fahrzeugvarianz, sowohl hinsichtlich der Fahrzeugkonfiguration (z.B. Sattelzug- oder Pritschenfahrzeug) als auch der Einbauposition der Sensorik (z.B. unterschiedliche Einbauhöhen). Gleichzeitig steigen die Anforderungen seitens Kunden und Gesetzgeber an die Leistungsfähigkeit der FAS, sodass diese in einer höheren Tiefe abgesichert werden müssen. Bei der Absicherung von Fahrzeugen ab SAE Level 4 müssen zudem neue, bisher potentiell unbekannte Szenarien getestet werden, da die Rückfallebene des menschlichen Fahrers entfällt [5].

Bestehende Test- und Absicherungsmethoden in der Serienentwicklung von FAS basieren auf manuell definierten Testfallkatalogen [14] [15]. Hierbei spielen reale Testumgebungen (Prüfgelände und Feldtests) eine wichtige Rolle. Heutige Systeme erfordern bereits mehrere Millionen reale Testkilometer [16]. Auch aufgezeichnete Daten realer Fahrer können zur Resimulation genutzt werden (z.B. Shadow Mode [17]). Bereits für die zukünftige Entwicklung von FAS sind diese Ansätze aufgrund der steigenden Testfallanzahl nicht mehr in der aktuellen Form umsetzbar. Der manuellen Definition von Testfällen und deren Durchführung im Realtest sind aus ökonomischen Gründen quantitative Grenzen gesetzt, insbesondere wenn dabei verschiedene Fahrzeugvarianten abgedeckt werden müssen. Auch die Resimulation aufgezeichneter Daten liefert bei eingreifenden Systemen nur einen begrenzten Beitrag zur Absicherung. Somit sind neue Ansätze zur Steigerung der Testeffizienz und Sicherstellung einer ausreichenden Testabdeckung in zukünftigen Entwicklungsprojekten notwendig.

1.2 Forschungsfragen und Lösungsansätze

Um der steigenden Absicherungsbreite und -tiefe bei der Fahrautomatisierung zu begegnen, werden in der Forschung verschiedene Ansätze diskutiert (siehe 3.1). Eine zentrale Rolle spielt dabei der sogenannte **szenariobasierte** Ansatz (siehe 3.1.4), der die Reduktion redundanter Testfälle und Generierung neuer Testfälle mithilfe formalisierter Szenarien ermöglicht. Ein Szenario setzt sich aus einer Abfolge von Szenen zusammen [18] und kann je nach Abstraktionslevel als funktionales, logisches oder konkretes Szenario beschrieben werden [19]. Durch die Variation von Szenarioparametern werden Testfälle automatisiert und systematisch generiert. Mithilfe von logischen Szenarien kann die Testabdeckung optimiert werden, auch wenn eine vollständige Abdeckung hinsichtlich logischer Szenarien und des Parameterraums nicht garantiert werden kann. [20] [21]

Daneben gewinnen **X-in-the-Loop** (XiL)-Umgebungen weiter an Bedeutung und bieten das Potential, Realtests zu ersetzen bzw. zu ergänzen [22]. In XiL-Umgebungen wird die Umwelt des System Under Test (SUT) teilweise oder vollständig virtuell abgebildet, sodass sich ein geschlossener Regelkreis ergibt (siehe 2.5.2). Das „X“ legt die Art der Einbindung des SUT fest. Alle nicht real vorhandenen Komponenten der Umgebung wie z.B. andere Steuergeräte, Fahrzeugmodelle und Sensoren werden in Form von Simulationsmodellen integriert. Typische XiL-Ausprägungen sind Model-in-the-Loop (MiL), Software-in-the-Loop (SiL) und Hardware-in-the-Loop (HiL). [23]

Die Kombination des szenariobasierten Testens mit der stärkeren Nutzung von XiL-Umgebungen stellt somit einen vielversprechenden Weg dar, um Breite und Tiefe zukünftiger Testumfänge effizient zu steigern. Hieraus folgt unmittelbar **Forschungsfrage 1**:

1. *Wie können bestehende Entwicklungsprozesse für Fahrerassistenzsysteme angepasst oder ergänzt werden, um den szenariobasierten XiL-Test optimal einzusetzen?*

Entscheidende Aspekte sind dabei die Generierung von konkreten Szenarien aus logischen Szenarien und deren Zuordnung auf unterschiedliche Testumgebungen wie XiL und Realtest. Für die Szenariogenerierung werden bereits verschiedene Ansätze in der Forschung diskutiert [24]. Hingegen liefern bisherige Ansätze für die Zuordnung von Szenarien auf Testumgebungen erste

qualitative Kriterien, bieten aber keine Strategie zur ganzheitlichen Optimierung der Zuordnung [25] [26].

Es ergibt sich daher die folgende **Forschungsfrage 2**:

2. *Wie können konkrete Szenarien effizient auf unterschiedliche Testumgebungen zugeordnet werden, um eine ausreichende Testabdeckung und Glaubwürdigkeit der Testergebnisse sicherzustellen?*

Da jede XiL-Umgebung die Realität durch simulierte oder emulierte Komponenten vereinfacht repräsentiert [25] [20], ist die Glaubwürdigkeit (siehe 3.3.1) der XiL-Ergebnisse bei der Zuordnungsstrategie zu berücksichtigen. Die Glaubwürdigkeit bewertet dabei die Übertragbarkeit der XiL-Ergebnisse auf die Realität. Bisherige Ansätze fokussieren sich primär auf die Validierung von Sensormodellen [27] oder Fahrdynamikmodellen [28]. Diese lassen sich jedoch nur bedingt auf eine gesamte XiL-Umgebung übertragen. Einige wenige Publikationen liefern Ansätze für die Validierung einer gesamten XiL-Umgebung, wobei eine Glaubwürdigkeitsbewertung noch aussteht [29] [30] [31]. Daher wird als **Forschungsfrage 3** definiert:

3. *Wie kann die Glaubwürdigkeit einer XiL-Umgebung für konkrete Szenarien bewertet werden?*

Sowohl Realtests als auch XiL-Tests beinhalten Unsicherheiten [310] [313], die einen Einfluss auf das Testergebnis haben können. Es ergibt sich somit ein probabilistisches Verhalten im Realtest, welches im XiL-Test mit berücksichtigt werden soll. Somit stellt sich **Forschungsfrage 4**:

4. *Wie kann die Unsicherheit des Realtests im XiL-Test modelliert werden?*

1.3 Umfeld und Anwendung

Die vorliegende Dissertation entstand im Rahmen eines Projekteinsatzes in der Abteilung „Fahrerassistenzsysteme - Aktive Sicherheit“ der Daimler Truck AG. Somit konnten die im Rahmen der Promotion erarbeiteten Ansätze und Methoden unmittelbar innerhalb eines globalen Serien-Entwicklungsprojektes für FAS angewandt und bewertet werden.

1.4 Struktur der Dissertation

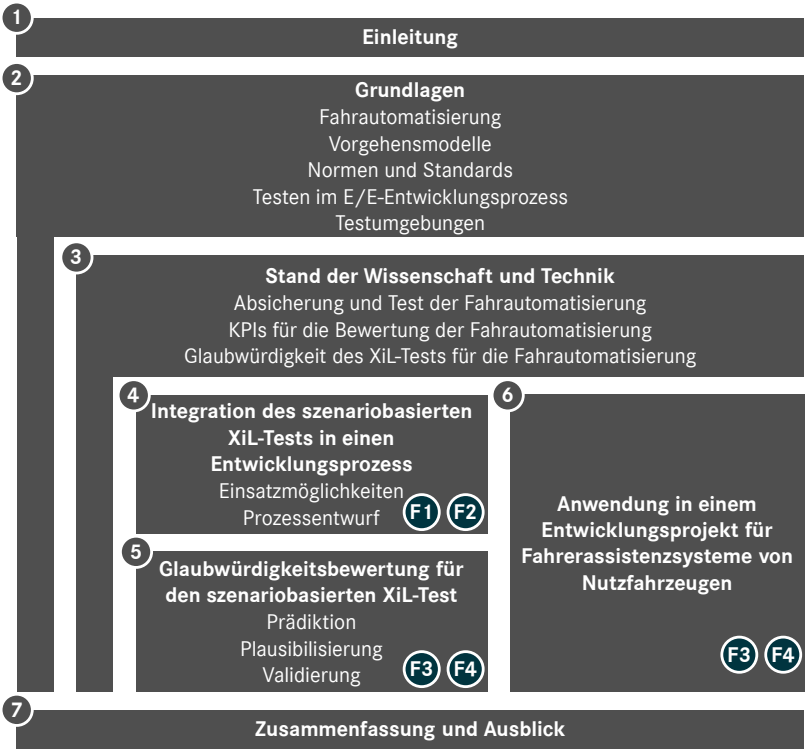

Abbildung 1.1: Struktur der Dissertation und Verknüpfung mit Forschungsfragen

Als Basis für die Erarbeitung von Ansätzen für die in dieser Einleitung vorgestellten Forschungsfragen werden in Kapitel 2 zunächst essentielle Grundlagen eingeführt. Neben der Fahrautomatisierung behandelt das Kapitel eine Auswahl von in der Automobilindustrie verwendeten Vorgehensmodellen sowie Normen und Standards. Detaillierte Grundlagen werden zum Thema Testing und Testumgebungen gegeben. Anschließend wird in Kapitel 3 der Stand der Wissenschaft und Technik diskutiert. Im Fokus stehen dabei mehrere Testansätze für Fahrautomatisierung und KPIs für deren Bewertung. Darüber hinaus

wird die Glaubwürdigkeit, Konfidenz und Validierung von XiL-Umgebungen sowie einzelnen Simulationsmodellen diskutiert.

Kapitel 4 greift die Anwendung des szenariobasierten XiL-Tests im Serienumfeld auf. Anhand von drei Einsatzmöglichkeiten wird **Forschungsfrage 1** diskutiert. Die Detaillierung der dritten Einsatzmöglichkeit in Form eines Prozessentwurfs liefert Antworten für **Forschungsfrage 2**.

In Kapitel 5 wird ein Vorgehen für die im Prozessentwurf vorgesehene Glaubwürdigkeitsbewertung konzeptioniert, was einen unmittelbaren Zusammenhang mit **Forschungsfrage 3** herstellt. Die Glaubwürdigkeitsbewertung gliedert sich je nach erforderlicher Konfidenz in die drei Teile Prädiktion, Plausibilisierung und Validierung. Als Voraussetzung der Prädiktion werden Unsicherheitsmodelle für drei Subsysteme der XiL-Umgebung konzeptioniert und implementiert. Diese stellen den Bezug zu **Forschungsfrage 4** her.

In Kapitel 6 wird die Glaubwürdigkeitsbewertung anhand von sieben Beispielszenarien für drei Fahrerassistenzsysteme evaluiert. XiL-Referenz und Datengrundlage liefert das Entwicklungsprojekt der in Abschnitt 1.3 genannten Abteilung. Abschließend wird die Dissertation in Kapitel 7 zusammengefasst und deren weitere Forschungspotentiale aufgezeigt.

2 Grundlagen zu Entwicklung und Test von Fahrerassistenzsystemen

2.1 Fahrautomatisierung

Die Fahrautomatisierung hat in den vergangenen Jahrzehnten in Form von Fahrerassistenzsystemen eine steigende Marktdurchdringung erfahren. Auch im Bereich der Nutzfahrzeuge haben sie eine wachsende Bedeutung und sind bereits zum Teil gesetzlich vorgeschrieben [32].

2.1.1 Begrifflichkeiten der Fahrautomatisierung

2.1.1.1 Automatisierungslevels nach SAE J3016

In den vergangenen Jahren wurden von verschiedenen Institutionen Standards für die Klassifizierung von Fahrautomatisierung veröffentlicht, die sich voneinander unterscheiden [33] [34]. Dabei hat sich der SAE Standard J3016 „Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles“ [5] etabliert und wird in dieser Dissertation als Referenz herangezogen. Kern des SAE-Standards ist die Definition von sechs Automatisierungslevel, anhand derer sowohl Fahrzeuge mit als auch ohne Automatisierung klassifiziert werden können. Die Definition lehnt sich an davor existierende Definitionen nach [33] und [34] an.

Im Standard werden alle Fahraufgaben in der sogenannten Dynamischen Fahraufgabe (Dynamic Driving Task, DDT) zusammengefasst. Zur Abgrenzung der einzelnen Automatisierungslevels wird diese in die folgenden Unteraufgaben aufgegliedert [5]:

- **Kontinuierliche laterale und longitudinale Regelung**: Dies umfasst das Lenken sowie das Beschleunigen bzw. das Abbremsen. Übernimmt ein System diese Aufgaben nicht kontinuierlich, sondern beispielsweise nur eventbezogen, so wird dieser Teil der DDT vom System nicht erfüllt.
- **Objekt- und Eventdetektion** sowie Reaktion darauf (Object and Event Detection and Response, OEDR): Dies beinhaltet sowohl die Überwachung des Umfelds wie auch die Ausführung von entsprechenden Reaktionen auf Objekte und Events.
- **Rückfallebene**: Tritt ein Systemfehler auf, der Einfluss auf die Leistungsfähigkeit des Systems hat, oder verlässt das Fahrzeug die vorgesehene Betriebsumgebung, so muss entweder der Fahrer die DTT übernehmen oder Fahrer bzw. System einen Zustand minimalen Risikos (Minimal Risk Condition) erreichen.

Level	**Bezeichnung**	**DDT** (lat. & long. Regelung)	**DDT** OEDR	**DDT** Rückfall-ebene	**ODD**	**Anmerkung**
0	No Driving Automation	Fahrer	Fahrer	Fahrer	[-]	
1	Driver Assistance	Fahrer/ System	Fahrer	Fahrer	Ein-geschränkt	System regelt entweder lateral oder longitudinal
2	Partial Driving Automation	System	Fahrer	Fahrer	Ein-geschränkt	
3	Conditional Driving Automation	System	System	Einsatz-bereiter Nutzer	Ein-geschränkt	
4	High Driving Automation	System	System	System	Ein-geschränkt	
5	Full Driving Automation	System	System	System	Unein-geschränkt	

Abbildung 2.1: Fahrautomatisierungslevels nach SAE [5]

In Ergänzung zur Einteilung der DDT definiert der SAE-Standard den Begriff „Operational Design Domain“ (ODD), welche als vorgesehene Betriebsumgebung des Systems aufgefasst werden kann. Dies umfasst alle Betriebsbedingun-

gen, für die das System ausgelegt ist. Die Einschränkungen können sich unter anderem auf den Ort, die Uhrzeit, Verkehrsbedingungen oder Straßenbedingungen erstrecken [5]. Zusammen mit den drei Teilen der DDT ergeben sich somit vier Kriterien für die Definition von Fahrautomatisierung. In Abhängigkeit, ob Fahrer oder System die Teilaufgaben erfüllen, wird ein Automatisierungslevel abgeleitet (siehe Abb. 2.1).

Neben der Definition der Automatisierungslevels nimmt der SAE-Standard J3016 eine Abgrenzung von Begrifflichkeiten vor [5]. Da sich diese nicht immer direkt übersetzen lassen, sollen diese im Kontext dieser Dissertation wie folgt verwendet werden:

- **Fahrautomatisierungssystem** (Driving Automation System, DAS): Kombination aus Hardware und Software, welche die DDT teilweise oder komplett ausführen kann. Dementsprechend bezieht sich dieser Begriff auf alle Automatisierungslevels von 1 bis 5. [5]
- **Automatisiertes Fahrsystem** (Automated Driving System, ADS): Kombination aus Hardware und Software, welche die DDT komplett ausführen kann. Dementsprechend bezieht sich dieser Begriff auf die Automatisierungslevels von 3 bis 5. [5]

Des Weiteren wird im Standard empfohlen, auf den häufig verwendeten Begriff „Autonom“ bzw. „Autonomes Fahren“ zu verzichten, da dies eine Unabhängigkeit von der Umgebung impliziert. Da in der Fahrautomatisierung die Kommunikation zwischen Fahrzeugen als wichtiges Element angesehen wird, ist eine Unabhängigkeit von der Umgebung nur teilweise der Fall. [5]

2.1.1.2 Fahrerassistenzsysteme

Bei Fahrerassistenzsystemen liegt der Fokus nicht darauf, die DDT vollständig zu übernehmen, sondern den Fahrer bestmöglich zu unterstützen. Primäres Ziel ist daher, die Kontrollierbarkeit des Fahrzeugs durch den Fahrer sicherzustellen bzw. wiederherzustellen [35]. Neben der Einstufung als System nach Automatisierungslevel 0 bis 2 nach SAE J3016 lassen sich Fahrerassistenzsysteme daher ergänzend wie folgt kategorisieren [36]:

- **Informierende und warnende Funktionen**: Hier erfolgt kein aktiver Eingriff, es wird lediglich der Fahrer gewarnt. Dementsprechend liegt hier auch keine Automatisierung nach SAE-Definition vor. Beispiele: Spurverlassenswarnung, Verkehrszeichenassistent, Auffahrassistent.
- **Kontinuierlich automatisierende Funktionen**: Diese Systeme entsprechen den SAE Levels 1 oder 2, da sie eine kontinuierliche Regelungsaufgabe übernehmen. Der Fahrer kann hierbei immer übersteuern. Beispiele: Adaptive Cruise Control (ACC), Spurhalteassistent.
- **Eingreifende Notfallfunktionen**: Diese Systeme greifen dann ein, wenn davon auszugehen ist, dass der Fahrer keine ausreichende Kontrolle mehr hat, um einen potentiellen Unfall zu vermeiden. Da es sich um keine kontinuierliche Regelung handelt, entspricht dies dem SAE Level „No Driving Automation“. Beispiele: Notbremsassistent, aktiver Abbiegeassistent, Nothalteassistent.

2.1.2 Systemarchitektur

Unabhängig vom Automatisierungslevel basiert die Systemarchitektur der meisten Systeme auf dem sogenannten „Sense-Plan-Act“-Prinzip [37] [38]. Dieses stellt eine Analogie zum menschlichen Informationsverarbeitungsprozess dar [39]: Zunächst erfolgt eine Informationsaufnahme (Sense), welche beim Menschen durch entsprechende Sinnesorgane erfolgt, während ein Fahrautomatisierungssystem auf einen oder mehrere Sensoren zurückgreift. Für die folgende Informationsverarbeitung (Plan) sind kognitive Fähigkeiten des Fahrers gefragt – oder eine entsprechende Algorithmik des Systems. Letztlich wird in der Informationsabgabe (Act) die Aktion umgesetzt, was entweder über das Hand-Arm-System bzw. Fuß-Bein-System des Fahrers geschieht, oder durch Aktoren des Systems. Der Fahrer wird sowohl bei Informationsaufnahme als auch Informationsabgabe durch das System unterstützt. [39]

Eine weitere Referenz zur Ableitung einer Systemarchitektur ist die Kombination des Drei-Ebenen-Modells für zielgerichtete Tätigkeiten nach Rasmussen mit der Drei-Ebenen-Hierarchie der Fahraufgabe nach Donges [40]. Hierbei wird die Fahraufgabe in die Ebenen Navigation, Führung und Stabilisierung aufgeteilt und mit den Tätigkeiten wissensbasiertes Verhalten, regelbasiertes Verhalten und fertigkeitsbasiertes Verhalten verknüpft (siehe Abb. 2.2).

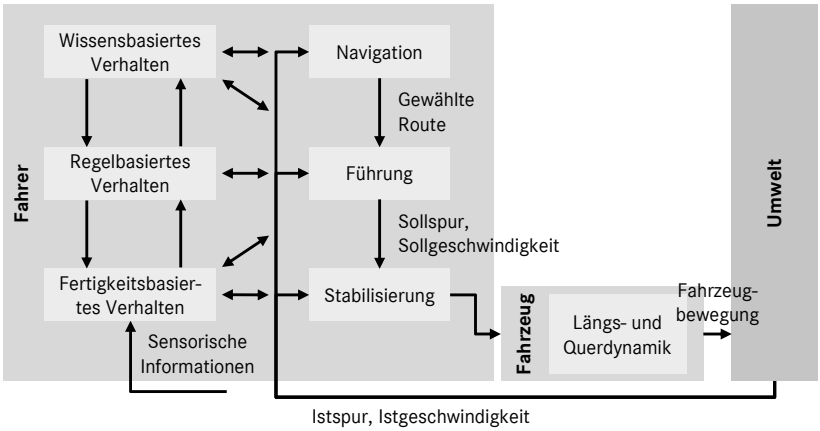

Abbildung 2.2: Kombination aus Drei-Ebenen-Modell nach Rasmussen mit der Drei-Ebenen-Hierarchie nach Donges (vereinfacht) [40]

Gerade an der Schnittstelle der Bereiche „Sense“ und „Plan“ existieren mehrere Varianten, wie die Systemarchitektur definiert werden kann. Häufig werden wie in [41] und [42] Lokalisierung und/oder Wahrnehmung (auch bezeichnet als „Perception“) in unterschiedlicher Form herausgelöst und als separater Bereich definiert. Zum einheitlichen Verständnis wird in dieser Dissertation die in Abb. 2.3 dargestellte Systemarchitektur verwendet, die an [37] angelehnt ist und die drei Bereiche „Sense“, „Plan“ und „Act“ definiert. Lokalisierung und Wahrnehmung werden daher im Bereich „Sense“ verortet.

2.1.2.1 Sense

Sensoren stellen den ersten Schritt in der Prozesskette der Informationsverarbeitung dar (siehe Abb. 2.3). In der Definition nach [43] formt ein Sensor „[. . .] physikalische, chemische oder biologische Messgrößen in elektrische Messsignale um, die mit den Messwerten in eindeutigen oft linearen Zusammenhängen stehen“. Der Begriff Sensor ist demnach sehr weit gefasst und wird für den Anwendungsfall Fahrautomatisierung in die drei Arten Lokalisierungssensoren, Umweltsensoren und Fahrzeugsensoren eingeteilt [41]. Während Lokalisierungssensoren die Position des Ego-Fahrzeug in einem Ko-

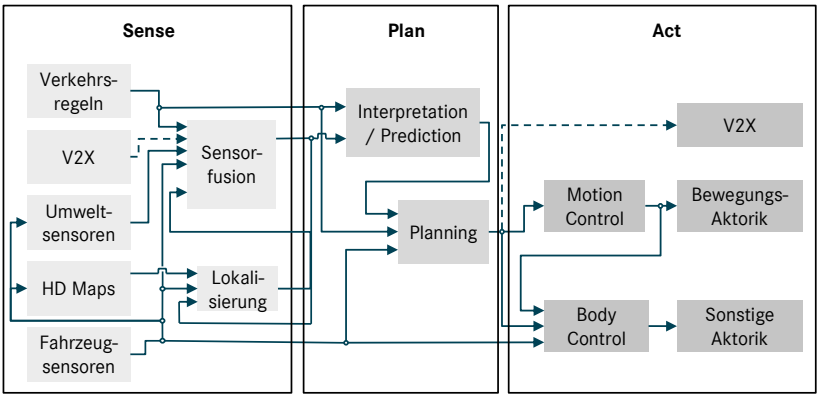

Abbildung 2.3: Systemarchitektur in Anlehnung an [37]

ordinatensystem bereitstellen [41], liefern Umweltsensoren wie Radar- oder Kamerasensoren Informationen über die Umgebung [42]. Fahrzeugsensoren beinhalten alle sonstigen Sensoren, die Daten über das Fahrzeug erfassen [42].

Werden in dieser Dissertation Sensoren erwähnt, so sind Umweltsensoren gemeint, da diese die größte Bedeutung im Bereich der Fahrautomatisierung haben. Passive Sensoren wie beispielsweise Kameras nehmen die natürlich emittierte Energie von Objekten auf und verarbeiten diese als reine Empfänger weiter. Im Gegensatz dazu emittieren aktive Sensoren Energie an die Umgebung und detektieren die dadurch hervorgerufenen Reflexionen. Diese sind in Form von Radarsensoren, Lidarsensoren oder Ultraschallsensoren im Automobilbereich vertreten. [44]

Radarsensoren Ursprünglich als Radio Detection and Ranging bezeichnet, kamen Radarsensoren Ende der 90er Jahre zum ersten Mal in ACC-Systemen in der Automobilindustrie auf den Markt [45]. Das Wirkprinzip des Radars basiert auf der in eine Richtung gebündelten Aussendung elektromagnetischer Wellen, die von Objekten reflektiert werden und von einem Empfänger des Sensors aufgenommen werden. Anhand der Detektionen von Rückstrahlzentren können Objekte der Umwelt identifiziert werden. [45] Radarsensoren können sowohl auf den Nahbereich (Short Range Radar) als auch auf den Fernbereich (Long Range Radar) ausgelegt sein und damit Sichtweiten bis 200 m abdecken [46].

Bei der Erkennung von Objekten durch einen Radarsensor spielt deren Radarquerschnitt (Radar Cross Section, RCS) eine große Rolle. Einfluss auf den Radarquerschnitt hat neben der Querschnittsfläche der Objekte auch die Art der Oberfläche sowie die Ausrichtung und Geometrie der Oberfläche. Dabei sind Oberflächen, die senkrecht zum Sensor positioniert sind, im Allgemeinen durch einen größeren RCS charakterisiert und damit besser erkennbar. Eine ungünstige Ausrichtung hingegen kann die Erkennbarkeit von Objekten signifikant reduzieren. Des Weiteren hat Regen eine dämpfende Wirkung auf die Radarstrahlen und führt zu zusätzlichen Störungen wie Rauschen oder Linseneffekten. Ein weiterer Effekt ist die sogenannte Mehrwegeausbreitung, bei der durch die Reflexion von Radarstrahlen an der Fahrbahnoberfläche oder an Leitplanken Interferenzeffekte die Signalqualität beeinflussen können. Gleichzeitig ist somit aber auch eine Detektion von ursprünglich verdeckten Objekten möglich. [45]

Die Abstands- und Geschwindigkeitsmessung erfolgt bei Radarsensoren über Frequenz- und/oder Amplitudenmodulation. Bei der Geschwindigkeitsmessung kann zusätzlich der Dopplereffekt herangezogen werden [45]. Außerdem wird der Mikrodopplereffekt, welcher den Dopplereffekt bei der Arm- und Beinbewegung beschreibt, bei der Erkennung von Fußgängern und Radfahrern genutzt [47].

Kamerasensoren Kameras kommen der spektralen, räumlichen und temporalen Auflösung des menschlichen Auges von allen Sensortechnologien am nächsten. Sie hatten ihren ersten Serieneinsatz in der Automobilindustrie in Form von Rückfahrkameras, werden aber heutzutage auch von vielen weiteren Fahrerassistenzsystemen in Kombination mit Computer Vision Algorithmen genutzt [48]. Eine weitere Anwendung ist die sogenannte MirrorCam, welche den Seitenspiegel ersetzt und damit durch verbesserte Aerodynamik den Kraftstoffverbrauch reduziert [32]. Daneben kann bei Einsatz von Kameras im Innenraum der Fahrer überwacht werden oder eine Gestensteuerung ermöglicht werden [48].

Die bei Fahrerassistenzsystemen verwendeten Frontkameras sind meist hinter der Windschutzscheibe angebracht und arbeiten im sichtbaren Spektralbereich. Umgesetzte Funktionen sind unter anderem Fernlichtassistent, Verkehrszeichenerkennung, Fahrstreifenerkennung und Objekterkennung. In der Automo-

bilindustrie kommen sowohl Monokameras mit einem Kameramodul als auch Stereokameras mit zwei Kameramodulen zum Einsatz. [48]

Lidarsensoren Sensoren basierend auf Light Detection and Ranging (Lidar) nutzen die Emission von elektromagnetischen Wellen aus dem Bereich des sichtbaren Lichts [49]. Die Reflexionen der Objekte werden vom Sensor als Punktewolke wahrgenommen. [50] Lidarsensoren können eingeteilt werden in Multibeam-Lidare, welche mit mehreren Sender- und Empfängermodulen ausgestattet sind, und Scanning-Lidare, die eine rotierende Sender/Empfänger-Einheit besitzen [51]. Die Distanzmessung von Lidarsensoren geschieht über die Time-of-Flight-Messung, also über die Proportionalitätsbeziehung von Zeit zwischen Emission und Empfang eines Lichtimpulses einerseits und der Distanz zum Objekt bzw. Reflexionspunkt andererseits. Eine Geschwindigkeit kann durch Differentiation der Distanz ermittelt werden und erfordert somit mehrere Messungen. [49] Neben der reinen Entfernungsmessung sind Lidarsensoren auch in der Lage, Objekte zu klassifizieren, wobei die Leistungsfähigkeit hier unterhalb der von Kameras einzuordnen ist. [49]

Ultraschallsensoren Ultraschallsensoren sind aktive Sensoren basierend auf der Emission von Schallwellen. Die Distanzmessung wird dabei auf Basis des Laufzeitprinzips durchgeführt. Hierbei hat die Lufttemperatur einen Einfluss auf die Ausbreitungsgeschwindigkeit und beeinflusst ebenso wie mögliche Einflüsse von externen Schallquellen die Genauigkeit der Messung. Weitere Störfaktoren sind Vereisungen oder Verschmutzungen der Membran. [52] Die Reichweite von Ultraschallsensoren liegt in aktuellen Systemen bei ca. 2,5 m, weshalb ein Einsatz vorrangig in Einparksystemen stattfindet [46] und weniger im Bereich der Fahrautomatisierung.

Eine Übersicht von Vor- und Nachteilen der einzelnen Umweltsensortechnologien ist in Abb. 2.4 dargestellt.

Neben den Umweltsensoren beinhaltet der Bereich „Sense“ unter anderem die folgenden Elemente:

HD Maps/Lokalisierung Durch sogenannte High-Definition Maps (HD Maps), welche einen deutlich höheren Detailgrad als Karten aus dem Bereich der Navi-

Sensor-technologie	Vorteile	Nachteile
Radar	• Hohe longitudinale Auflösung (Distanz und Geschwindigkeit) • Hohe Robustheit gegenüber Wetterbedingungen und Dunkelheit	• Niedrige laterale und vertikale Auflösung • Hohe Abhängigkeit zur Objektausrichtung
Kamera	• Hohe laterale und vertikale Auflösung • Gute Objekterkennung	• Schlechte Geschwindigkeitsmessung • Niedrige Robustheit gegenüber Wetterbedingungen, Dunkelheit und Verschmutzung
Lidar	• Hohe longitudinale Auflösung (Distanz und Geschwindigkeit) • Hohe Robustheit gegenüber Einflüssen von Dunkelheit	• Niedrige laterale und vertikale Auflösung • Niedrige Robustheit gegenüber Wetterbedingungen • Höchste Kosten
Ultraschall	• Hohe Auflösung bei niedrigen Distanzen • Niedrigste Kosten	• Geringe Reichweite • Niedrige Robustheit gegenüber Wettereinflüssen

Abbildung 2.4: Vor- und Nachteile von Umweltsensortechnologien nach [45], [53] [54]

gation aufweisen und in Echtzeit aktualisiert werden können, stehen ergänzend zu den Sensordaten zusätzliche Umgebungsinformationen zur Verfügung [55]. Dabei können Informationen aus HD Maps auch zur Korrektur fehlerhafter Sensorinformationen genutzt werden und bieten somit zusätzliche Redundanz [56]. Erforderlich ist die exakte Lokalisation des Fahrzeugs, wobei GNSS alleine hierfür nicht ausreichend ist [55]. Eine mögliche Alternative, die direkt die Karteninformationen zur Lokalisation nutzt, ist SLAM [57].

V2X Der Begriff V2X steht für die Kommunikation eines Fahrzeugs mit einem anderen Teilnehmer wie beispielsweise ein anderes Fahrzeug oder die Infrastruktur. Die Technologie bietet das Potential, die Wahrnehmung des Fahrzeugs über den durch Umweltsensoren darstellbaren Sichtbereich hinaus zu erweitern. Jedoch besteht eine Abhängigkeit von der Ausstattungsrate aller

Fahrzeuge mit V2X sowie der Verfügbarkeit von V2X. Deshalb sollten die Funktionen von Fahrautomatisierungssystemen unabhängig von V2X sichergestellt sein. [58]

Fusion und Sensordatenverarbeitung Die in aktuellen Serienfahrzeugen verwendeten Sensortechnologien sind hinsichtlich ihrer Stärken und Schwächen komplementär, sodass durch eine Kombination von mehreren Sensortechnologien deren Schwächen gegenseitig kompensiert werden können. Dazu kommt eine Sensorfusion zum Einsatz, welche zudem noch eine Redundanz bei Ausfall oder Fehlinterpretation einzelner Sensoren bietet. [59]

Die Fusion ist optionaler Teil der Sensordatenverarbeitung, welche den kompletten Prozess vom Empfangen der Nutzsignale des Sensors bis zur Klassifikation beinhaltet. Erster Schritt der Sensordatenverarbeitung ist das Messen von Nutzsignalen, welche vom Empfangselement des jeweiligen Sensors bereitgestellt werden, um diese in Rohdaten zu wandeln. Anschließend werden in der Wahrnehmung Merkmale aus den Rohdaten extrahiert, aus denen wiederum Objekthypothesen für die Klassifikation abgeleitet werden. [59]

Findet die Fusion auf Objektebene statt, so wird dies als dezentrale Architektur bezeichnet. Bei einer Fusion auf Rohdaten- oder Merkmalsebene ist von einer zentralen Architektur die Rede. Eine dezentrale Architektur wird bei Sensoren empfohlen, die mit unterschiedlichen Wirkprinzipien arbeiten, während eine zentrale Architektur bei identischen Sensorwirkprinzipien besser geeignet ist. Erforderlich für die Fusion ist, dass sowohl das verwendete Koordinatensystem der zu fusionierenden Sensoren identisch ist als auch, dass über Zeitstempel die zeitliche Synchronisation der Daten möglich ist. [59]

2.1.2.2 Plan

Sowohl Inhalt des Bereichs „Plan“ als auch die Schnittstelle zum Bereich „Sense“ sind nicht einheitlich in der Literatur abgegrenzt. Die funktionale Architektur an dieser Stelle ist zudem stark vom Automatisierungslevel des Systems bzw. der umgesetzten Funktion abhängig. Im Folgenden wird in Anlehnung an die Definition nach [37] auf die Elemente Interpretation/Prediction sowie Planning exemplarisch eingegangen, die vor allem der Architektur von

Systemen ab SAE Level 3 entsprechen. Werden Fahrerassistenzsysteme betrachtet, so ist ein Entfall einzelner Elemente möglich.

Interpretation/Prediction Basierend auf den fusionierten Sensordaten und den Lokalisierungsdaten findet eine Interpretation des aktuellen Fahrszenarios statt, die sowohl den aktuellen Zustand des eigenen Fahrzeugs wie auch das Verhalten der anderen Verkehrsteilnehmer erfasst. In der folgenden Prädiktion wird das Verhalten der anderen Verkehrsteilnehmer für zukünftige Zeitpunkte extrapoliert. [37] Hierbei kommt in der Regel eine probabilistische Betrachtung zum Einsatz, da aus den Sensorinformationen keine deterministischen Aussagen über das Verhalten der anderen Fahrzeuge abgeleitet werden können. In der Literatur genannte Verfahren basieren auf Maschinellem Lernen (Markov Decision Process) [60] oder nutzen Modelle basierend auf physikalischen Gesetzen, Verkehrsregeln oder situationsabhängigem Verhalten [37]. Dabei wird nicht zwingend das Worst-Case-Szenario prädiziert, sondern das in der jeweiligen Situation kritischste erwartbare Verhalten. Die Genauigkeit der Prädiktion nimmt ab, je weiter der prädizierte Zeitpunkt in der Zukunft liegt. [37]

Planning Der Bereich Planning kann entsprechend der Drei-Ebenen-Hierarchie nach Donges [40] (siehe Abb. 2.2) strukturiert werden, sodass sich auf der Navigationsebene der Bereich Route Planning [60] bzw. Global Planning [61] befindet. Ziel ist es, basierend auf dem bekannten Straßennetz, der Mission und der von der Lokalisierung bereitgestellten Position die schnellste oder in anderer Hinsicht optimale Route zu finden. Eine mögliche Methode hierzu ist der Dijkstra-Algorithmus. [60]

Basierend auf den Inputs von Route Planning und Prädiktion wird das Motion Planning ausgeführt, welches sich wiederum in Behavior Planning und Path Planning aufteilen lässt. Im Behavior Planning wird das Verhalten des Systems auf Führungsebene beschrieben, was beispielsweise durch Zustandsautomaten umgesetzt wird [60]. Dabei sind sowohl explizite als auch implizite Verkehrsregeln zu beachten. Implizite Verkehrsregeln stellen ein erwünschtes Verhalten dar, wenn keine expliziten Regeln dies erfordern oder es einen potentiellen Konflikt zwischen expliziten Regeln und der Sicherheit gibt [37]. Das Path Planning entspricht der Trajektorienplanung, welche der Stabilisierungsebene

zuzuordnen ist [62]. Die Umsetzung kann über eine numerische Optimierung einer Kostenfunktion, Random Sampling Algorithmen, Verknüpfung und Fitting von Kurven oder graphenbasierten Ansätzen stattfinden [61]. Ergebnis des kompletten Planungsprozesses ist eine Zieltrajektorie.

2.1.2.3 Act

Nachdem in den Bereichen „Sense“ und „Plan“ eine Zieltrajektorie bestimmt wurde, wird diese im Bereich „Act“ umgesetzt. Dies beinhaltet sowohl die Regelung als auch die tatsächliche Umsetzung mit der entsprechenden Aktorik des Antriebsstrangs, des Bremssystems und des Lenksystems. Die Redundanz bzw. der sichere Betrieb auch im Fehlerfall (Fail-Operational-Betrieb) gewinnt mit steigendem Automatisierungslevel signifikant an Bedeutung. [37]

Motion Control Dieser Schritt beinhaltet die longitudinale und laterale Regelung des Systems auf die Zieltrajektorie [61]. Dabei muss die Regelung robust genug sein, um in unterschiedlichen Umgebungs- und Fahrbedingungen eingesetzt werden zu können [37]. In der Literatur wird zwischen Path Stabilization und Trajectory Stabilization unterschieden. Eine weitere Option ist es, eine modellprädiktive Regelung zu implementieren, um somit besser mit unterschiedlichen Fahrbedingungen zurechtzukommen. Hierbei wird über ein dynamisches Modell des Ego-Fahrzeugs dessen Verhalten zum jeweils zukünftigen Zeitschritt prädiziert und zur Regelung herangezogen. [60]

Aktorik Ein Aktor hat „die Aufgabe, mit steuerungs- und regelungstechnischen Funktionsprinzipien Bewegungen zu erzeugen, Kräfte auszuüben oder mechanische Arbeit zu leisten“ [63]. Die primären Stellgrößen hierbei sind die longitudinale Beschleunigung, welche über Antriebsstrang und Bremssystem umgesetzt wird, und die laterale Beschleunigung bzw. Gierrate, die über das Lenksystem verändert werden können [61].

Je nach Antriebskonzept besteht der Antriebsstrang immer aus einem Motor sowie ggf. Getriebe, Wellen oder Differentiale. [64] Deren Regelung erfolgt über ein oder mehrere Steuergeräte (Definition siehe 2.2.1), welche eine Momentenanforderung basierend auf der gewünschten longitudinalen Beschleunigung

umsetzen. Bei Fahrerassistenzsystemen muss hier neben der Momentenanforderung durch das System auch die Anforderung des Fahrers berücksichtigt und entsprechend priorisiert werden. [65]

Das Bremssystem in aktuellen Serien-PKW wird hydraulisch betrieben und verfügt über zwei redundante Kreise [66]. In Nutzfahrzeugen kommen auch pneumatische Bremssysteme mit zusätzlichen Dauerbremsen zum Einsatz [67]. Über einen Bremskraftverstärker kann einerseits der Fahrer unterstützt werden und andererseits eine externe Bremskraft seitens des Fahrautomatisierungssystems eingeleitet werden. [66] In sogenannten Brake-by-Wire-Konzepten findet die Übertragung des Bremssignals des Fahrers rein elektronisch statt, sodass auch eine rein elektromechanische Bremsaktorik verwendet werden kann. [68]

Die Lenkung leitet über den Schräglaufwinkel des Reifens eine Seitenkraft in das Fahrzeug ein, welche zu einer lateralen Bewegung und einem Giermoment führt [69]. Daneben existieren mehrere Störgrößen wie z.B. Wind, Neigung der Fahrbahn oder unterschiedliche Reibungskoeffizienten, welche Seitenkraft und Giermoment beeinflussen können und bei der Regelung berücksichtigt werden müssen [37]. Auch bei der Lenkung ist eine hydraulische oder elektromechanische Unterstützung des Fahrers in Form von zusätzlichen Aktoren verbaut. In Steer-by-Wire-Systemen existiert keine kraftschlüssige Kopplung von Lenkrad und Rad. Einzelradlenkungen sind damit möglich. [70]

2.1.3 Fahrerassistenzsysteme bei Nutzfahrzeugen

Während in Nutzfahrzeugen grundsätzlich die gleichen Fahrerassistenzsysteme verfügbar sind wie in PKW, gibt es Unterschiede hinsichtlich der Auslegung und auch der Bedeutung dieser Systeme. Durch die deutlich höhere Fahrzeugmasse und ggf. auch der höheren Anzahl an Passagieren bei Bussen sind die Folgen von Unfällen, an denen Nutzfahrzeuge beteiligt sind, in den meisten Fällen schwerwiegender [67]. Des Weiteren handelt es sich bei Nutzfahrzeugen um Arbeitsgeräte, an deren Verfügbarkeit und einwandfreien Funktionserbringung ein höherer Anspruch gestellt wird als bei PKW. Ein Ausfall eines Nutzfahrzeuges ist in der Regel unmittelbar mit hohen Kosten verbunden. [32] Gleichzeitig übersteigt die durchschnittliche Laufleistung eines Nutzfahrzeuges die eines PKW deutlich [71]. Folglich entsprechen die Ansprüche, die an Fahrerassistenzsysteme von Nutzfahrzeugen gestellt werden, mindestens denen

vergleichbarer Systeme im PKW. Im Folgenden wird auf einige der aktuell im Markt verfügbaren Fahrerassistenzsysteme eingegangen:

- **Adaptive Cruise Control** (ACC): Dieses System erweitert den auf die reine Geschwindigkeitsregelung ausgelegten Tempomaten um eine Abstandsregelung. Somit ist das System in der Lage, die Längsregelung des Fahrzeugs zu übernehmen. [72]
- **Spurhalteassistent/Spurverlassenswarnsystem** (Lane Departure Warning System, LDWS): Erste Systeme dieser Art waren reine Warnsysteme im Fall, dass der Fahrer ungewollt die Spur verlässt. Heutige Systeme ergänzen dies mit einer Querregelung, sodass sich in Kombination mit einem ACC-System eine Automatisierung nach SAE Level 2 erreichen lässt. [32] Die Funktionsweise eines Spurhalteassistenten in Kombination mit ACC ist in Abb. 2.5 illustriert.
- **Notbremsassistent** (Advanced Emergency Braking System, AEBS): Dieses System erkennt Objekte in der eigenen Trajektorie und reagiert im Fall einer drohenden Kollision zunächst mit einer optischen und akustischen Warnung. Sofern hierbei keine Reaktion des Fahrers erfolgt, wird zunächst eine Teilbremsung mit ca. 50% der verfügbaren Bremsleistung eingeleitet. Wenn dies ebenfalls keine Reaktion des Fahrers hervorruft, wird über eine Vollbremsung das Fahrzeug zum Stillstand gebracht. AEBS sind in der Lage, sowohl auf andere Fahrzeuge als auch auf Fußgänger zu reagieren. Der Eingriff des Systems wird bei einer Fahrerreaktion abgebrochen. [32] Ein Beispielszenario eines Notbremsassistenten ist in Abb. 2.6 dargestellt.
- **Abbiegeassistent** (Blind Spot Information System, BSIS): Ist dieses System aktiv, so wird der Fahrer bei Abbiegevorgängen gewarnt, wenn sich andere Verkehrsteilnehmer im toten Winkel befinden. Dieses System ist im Speziellen bei Nutzfahrzeugen von Bedeutung, da durch die hohe Sitzposition des Fahrers besonders Fahrradfahrer und Fußgänger unter Umständen nicht erkannt werden, sodass es hier vermehrt zu tödlichen Unfällen kommt. [32] Dieses System wird von bislang einem Hersteller auch als eingreifendes System angeboten, welches im Ernstfall zusätzlich zur Warnung das Fahrzeug mit einer Bremsung zum Stillstand bringt [73]. Abb. 2.7 zeigt den Funktionsraum eines aktiven Abbiegeassistenten in einem Beispielszenario.

- **Verkehrszeichenerkennung**: Dieses rein informierende System ist in der Lage, Verkehrszeichen zu erkennen und dem Fahrer die aktuelle Höchstgeschwindigkeit sowie weitere enthaltene Informationen zurückzumelden. [48]

Abbildung 2.5: Illustration von Spurhalteassistent und ACC in Kombination, Daimler Truck AG

Aufgrund ihres bedeutenden Potentials zur Erhöhung der Verkehrssicherheit [74] sind Notbremsassistent und Spurverlassenswarnsystem seit dem Jahr 2015 für einen Großteil der neu zugelassenen Nutzfahrzeuge in der EU verpflichtend [10]. Im Rahmen der sogenannten General Safety Regulation wird die Anzahl der in Nutzfahrzeugen verpflichtend zu verbauenden Fahrerassistenzsysteme nochmals deutlich erweitert [75]. Damit sind in der EU für Neuzulassungen ab dem Jahr 2022 beispielsweise ACC, Abbiegeassistent und ein deutlich leistungsfähigerer Notbremsassistent verpflichtend [11]. Die Spezifikation und für die Zulassung erforderliche Tests gesetzlich vorgeschriebener Fahrerassistenzsysteme sind in UNECE Regulierungen wie beispielsweise R 130 (LDWS) [76], R 131 (AEBS) [77] und R 151 (BSIS) [78] festgehalten.

Abbildung 2.6: Beispielszenario für den Eingriff eines Notbremsassistenten, Daimler Truck AG

Abbildung 2.7: Beispielszenario für den Eingriff eines Abbiegeassistenten, Daimler Truck AG

2.2 Vorgehensmodelle

Entwicklungsprojekte in der Automobilindustrie erstrecken sich in der Regel über mehrere Jahre und erfordern eine Zusammenarbeit verschiedenster Domänen auf unterschiedlichen Ebenen. Waren Neu- bzw. Weiterentwicklungen der Branche in der Vergangenheit aus dem „klassischen“ Maschinenbau getrieben, so kamen bereits Anfang des Jahrtausends ca. 90% der Innovationen aus dem Bereich der Mechatronik [79]. Dies hat auch unmittelbaren Einfluss auf die Bedeutung entsprechender Entwicklungsprozesse, welche häufig der reinen Softwareentwicklung entlehnt sind und Teil des sogenannten Systems Engineering sind.

2.2.1 Begriffsdefinitionen

Der Standard ISO/IEC/IEEE 24765 („Systems and software engineering“) beinhaltet eine Sammlung an Definitionen aus dem Bereich Systems Engineering. Der Begriff selbst ist hierbei wie folgt definiert:

> **Definition Systems Engineering:** *„Der interdisziplinäre Ansatz, den gesamten technischen und planerischen Aufwand zu beherrschen, welcher notwendig ist, um die Kombination an Bedürfnissen, Erwartungen und Randbedingungen der beteiligten Akteure in eine Lösung umzuwandeln und diese Lösung während des Lebenszyklus zu unterstützen.“* [80]

Der Systembegriff spielt im Systems Engineering wie auch in vielen anderen wissenschaftlichen Disziplinen eine zentrale Rolle. Es werden die Definitionen wie folgt verwendet:

> **Definition System:** *„Eine Kombination von interagierenden Elementen, die so organisiert sind, dass sie ein oder mehrere definierte Zwecke erfüllen.“* [80]

> **Definition Subsystem:** *„Untergeordnetes System innerhalb eines größeren Systems.“* [80]

Die Systemdefinition wird nach DIN IEC 60050-351 wie folgt aus regelungstechnischer Sicht ergänzt:

> **Ergänzung Definition System:** *„Menge miteinander in Beziehung stehender Elemente, die in einem bestimmten Zusammenhang als Ganzes gesehen und als von ihrer Umgebung abgegrenzt betrachtet werden. Das System wird als von der Umgebung und von den anderen äußeren Systemen durch eine gedachte Hüllfläche abgegrenzt betrachtet, durch welche die Verbindungen zwischen diesen Systemen und dem betrachteten System hindurchgehen.“* [81]

Als Sonderfall eines Subsystems wird eine Komponente gesehen und ist wie folgt definiert:

> **Definition Komponente:** *„Eine Entität mit einer eigenständigen Struktur wie beispielsweise eine Baugruppe oder ein Softwaremodul [. . .]; ein funktionaler oder logisch abgetrennter Teil eines Systems.“* [80]

Ein Steuergerät stellt eine mögliche Komponente dar:

> **Definition Steuergerät (Electronic Control Unit, ECU):** *„Ein eingebettetes System aus mindestens einem Prozessor mit entsprechender Peripherie, die in einem Gehäuse platziert sind.“* [82]

Ebenfalls verwendet wird der Begriff Funktion, welcher sich folgendermaßen abgrenzt:

> **Definition Funktion:** *„Umwandlung von Eingängen in Ausgänge mithilfe von Mechanismen und unter Anwendung von Regelungstätigkeit [...].“* [80]

Ziel eines Entwicklungsprozesses ist die Sicherstellung von Qualität, die wie folgt definiert ist:

> **Definition Qualität:** *„Der Grad zu welchem ein System die explizit und implizit formulierten Bedürfnisse der beteiligten Akteure erfüllt und dementsprechend einen Wert bietet.“* [80]

Um die Qualität eines Systems messbar bzw. nachprüfbar zu machen, wird diese in Form einer Spezifikation definiert:

> **Definition Spezifikation:** *„Eine detaillierte Formulierung in Form eines Dokumentes, welche eine maßgebliche Beschreibung eines Systems zum Zweck der Entwicklung oder Validierung bereitstellt.“* [80]

Maßgeblicher Teil einer Spezifikation sind die Anforderungen, welche wie folgt definiert sind:

> **Definition Anforderung:** *„Eine Bedingung oder Fähigkeit, die von einem System, einer Komponente, einem Produkt oder einer Dienstleistung erfüllt werden/vorhanden sein muss, um eine Vereinbarung, einen Standard, eine Spezifikation oder andere formale Dokumente zu erfüllen.“* [80]

Der Nachweis der Qualität eines Produkts erfolgt in der Verifikation und Validierung. In der Praxis werden diese häufig nicht klar voneinander abgegrenzt, sie unterscheiden sich jedoch wie folgt:

> **Definition Verifikation:** *„Prozess der Evaluation eines Systems oder einer Komponente, um zu bestimmen, ob die Produkte einer betrachteten Entwicklungsphase die zu Beginn der Phase gestellten Bedingungen erfüllt.“* [80]

> **Definition Validierung:** *„Prozess der Evaluation eines Systems oder einer Komponente während oder am Ende des Entwicklungsprozesses, um zu bestimmen, ob spezifizierte Anforderungen erfüllt werden.“* [80]

In anderen Worten drückt Boehm treffend aus, was Verifikation und Validierung unterscheidet, indem er die Begriffe mit Fragen verknüpft [83]:

> Verifikation: *„Am I building the product right?“*
> Validierung: *„Am I building the right product?“*

Sowohl Verifikation als auch Validierung sind in einen Entwicklungsprozess eingebettet. Dabei ist ein Prozess wie folgt definiert:

> **Definition Prozess:** *„ Ein Satz von miteinander verbundenen oder agierenden Aktivitäten, welcher Eingänge in Ausgänge umwandelt.“* [80]

Den Rahmen aller Prozesse eines Entwicklungsprojekts bieten Vorgehensmodelle:

> **Definition Vorgehensmodell:** *„Eine Struktur von Prozessen und Aktivitäten mit Bezug zum Lebenszyklus, welche in Etappen strukturiert sein können und als gemeinsame Referenz für die Kommunikation und das Verständnis dienen.“* [80]

2.2.2 Stage-Gate-Modell

Im Stage-Gate-Modell (siehe Abb. 2.8) wird ein Projekt in mehrere Projektphasen (Stages) eingeteilt, die über Meilensteine (Gates) miteinander verknüpft werden. Ein Gate markiert hierbei den Abschluss einer Stage und den Beginn der darauffolgenden Stage. Den Gates sind Zwischenergebnisse zur Messung der Zielerreichung zugeordnet. [84]

Jede Stage des Modells besteht aus parallelen Aktivitäten, einer Analyse der Aktivitäten sowie den daraus resultierenden Deliverables. Ziel ist es, in jeder Stage Informationen zu sammeln, um damit das Projektrisiko zu reduzieren, während die Projektkosten mit jeder Stage ansteigen. [85]

Basierend auf den Deliverables wird an jedem Gate eine Entscheidung getroffen, ob das Projekt weitergeführt, unterbrochen oder abgebrochen wird bzw. die Stage wiederholt wird. Somit findet an jedem Gate eine Qualitätskontrolle statt. Darüber hinaus findet an einem Gate die Planung von Aktivitäten, Deliverables und Terminen der nächsten Stage statt. [85]

Eine häufig verwendete Referenz des Stage-Gate-Modells ist das Modell nach Cooper, welches fünf Gates und Stages beinhaltet und auf welches auch hier Bezug genommen werden soll [85]. In der Automobilindustrie wird das Modell leicht abgewandelt genutzt, indem die serielle Anordnung der Stages aufgeweicht wird und es zu Überlappungen kommt. [84] Eine wichtige Rolle spielen

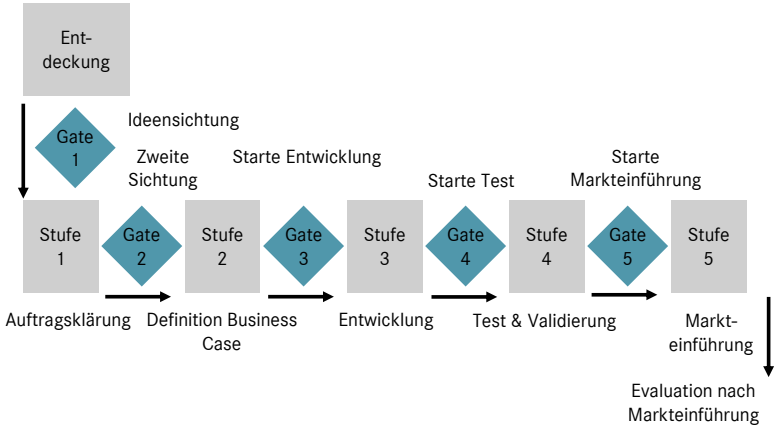

Abbildung 2.8: Stage-Gate-Modell nach Cooper [85]

dabei sogenannte Musterstände, welche als Meilensteine den Reifegrad des zu entwickelnden Systems bzw. der zu entwickelnden Komponente beschreiben [86]:

- **A-Muster**: Da dieser Musterstand lediglich zur Erprobung grundlegender Funktionen unter realen Einsatzbedingungen vorgesehen ist, sind nur Grundfunktionalitäten umgesetzt. Die Hardware unterscheidet sich von der Zielhardware.
- **B-Muster**: Im Unterschied zum A-Muster ist hier die Hardware nahe an der Zielhardware, sodass auch Einbaubedingungen untersucht werden können. Funktional entspricht dieser Musterstand auch noch nicht dem finalen Stand.
- **C-Muster**: Hier ist die Funktionalität vollständig implementiert und die Hardware seriennah umgesetzt, sodass die Erprobung des Gesamtsystems entsprechend der Serienanforderungen durchgeführt werden kann. Lediglich die Fertigung der Hardware unterscheidet sich von der Serie.
- **D-Muster**: Sowohl Funktion als auch Hardware entsprechen dem Serienprodukt und sind somit in vollem Umfang testbar.

2.2.3 V-Modell

Das V-Modell stammt ursprünglich aus dem IT-Bereich und wird im industriellen Umfeld zur Entwicklung mechatronischer Systeme als Referenz verwendet. Es stellt eine Weiterentwicklung des Wasserfallmodells dar [87] und wurde in seiner ersten Form von Boehm publiziert [83]. Eine weitere frühe Form des V-Modells findet sich in der Publikation von Rook [88]. Breite Anwendung fand das V-Modell 97, welches sich aus den vier Submodellen Projektmanagement, Softwareentwicklung, Qualitätssicherung und Konfigurationsmanagement zusammensetzt [89]. Wenn im Folgenden vom V-Modell die Rede ist, entspricht dies dem Teil Softwareentwicklung des V-Modells 97.

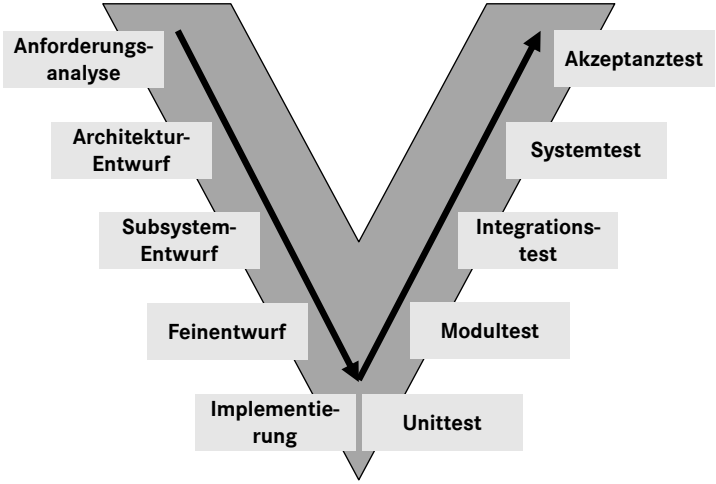

Abbildung 2.9: V-Modell in Anlehnung an Ammann und Offutt [90], korrespondierend mit dem Submodell Systemerstellung des V-Modell 97 [89]

Grundprinzip des V-Modells (siehe Abb. 2.9) ist es, das System zunächst in kleinere Subsysteme bzw. Komponenten zu unterteilen, diese separat und parallel zu entwickeln und anschließend zu einem Gesamtsystem zu integrieren. Die Definition, was als System betrachtet wird, kann vom einzelnen Steuergerät bis zum Gesamtfahrzeug variieren [86]. Die linke Seite des V-Modells (siehe Abb. 2.9) beinhaltet die Definition der Anforderungen sowohl für das Gesamtsystem wie auch der Subsysteme. Im untersten Teil des Modells findet

die Implementierung der Subsysteme statt, welche im rechten Teil des Modells zum Gesamtsystem integriert werden. Dabei werden sowohl die Funktion der einzelnen Subsysteme wie auch deren Zusammenspiel bei der Integration getestet. Der Test der entsprechenden Ebene stellt die Erfüllung der links auf der gleichen Ebene des Modells definierten Anforderungen sicher. [86] Sie sind nach Ammann und Offutt wie folgt definiert [90]:

- **Anforderungsanalyse**: Anforderungen des Kunden werden definiert.
- **Architektur-Entwurf**: Komponenten und deren Schnittstellen werden spezifiziert, sodass das Gesamtsystem die Anforderungen erfüllt.
- **Subsystem-Entwurf**: Struktur und Verhalten der Subsysteme werden spezifiziert, sodass deren Funktion im Kontext der Gesamtarchitektur erfüllt sind.
- **Feinentwurf**: Struktur und Verhalten der einzelnen Module werden definiert. Als Modul wird die Kombination mehrerer Units bezeichnet.
- **Programmierung/Implementierung**: Generierung des eigentlichen Codes und somit einer Unit. Letztere ist die kleinste Funktionseinheit.
- **Unittest**: Überprüfung der Funktionalität der Unit. In der Regel wird dies vom Entwickler vorgenommen.
- **Modultest**: Funktionsprüfung der einzelnen Module unabhängig voneinander. Dies ist auch häufig in der Verantwortung des Entwicklers.
- **Integrationstest**: Überprüfung der Schnittstellen zwischen den Subsystemen unter der Annahme, dass die Subsysteme selbst sich korrekt verhalten. Zum Teil wird dies in der Literatur auch mit dem Systemtest gleichgesetzt.
- **Systemtest**: Überprüfung, ob das zusammengesetzte Gesamtsystem die Spezifikation erfüllt. Dies setzt voraus, dass die einzelnen Module sich korrekt verhalten, da Fehler auf einer tieferen Ebene hier nur aufwändig identifiziert werden können.
- **Akzeptanztest**: Überprüfung, ob das Gesamtsystem den Kundenanforderungen gerecht wird. Hierbei wird Expertenwissen benötigt und Nutzersicht vorausgesetzt.

Im Laufe der Jahre wurden im Auftrag der Bundesrepublik Deutschland neben dem V-Modell 97 weitere Varianten des V-Modells entwickelt [91]. Als eines dieser Modelle ist hier das V-Modell XT zu nennen. Dieses ermöglicht über einen modularen Ansatz die flexible Anpassung des Modells, das sogenannte Tailoring. Dabei wird das Modell in eine Auftraggeber- und Auftragnehmerseite aufgeteilt und entsprechende Produkte, Aktivitäten, Rollen und Abläufe definiert. Die einzelnen Schritte stellen Entscheidungspunkte ähnlich der Meilensteine des Stage-Gate-Modells dar. [92]

2.2.4 Agile Softwareentwicklung und Scrum

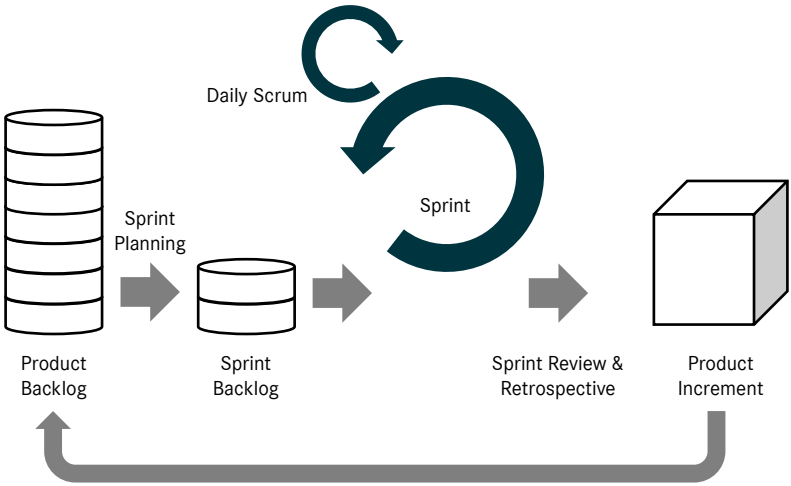

Abbildung 2.10: Übersicht des Scrum-Prozesses [93]

Im Gegensatz zu den zuvor genannten sequentiellen und strukturierten Vorgehensmodellen stellt die Agile Softwareentwicklung einen iterativen und flexiblen Entwicklungsprozess dar. Ziel ist es, die Kompetenzen der Entwickler stärker in den Fokus zu rücken und damit Unsicherheiten besser bewältigen zu können und die Effizienz zu erhöhen. [94] Die beiden am häufigsten vertretenen Methoden der Agilen Softwareentwicklung sind Extreme Programming und Scrum [94], wobei Scrum den größten Verbreitungsgrad hat [95].

Scrum definiert einen Prozess, Rollendefinitionen sowie Artefakte für die Softwareentwicklung [96]. Ein Entwicklungsprojekt wird bei Scrum über sogenannte Sprints strukturiert, welche einen fixen Zeitraum darstellen, in dem die Arbeitsergebnisse inkrementell und iterativ erzielt werden (siehe Abb. 2.10). Ein Sprint beinhaltet neben der Entwicklungstätigkeit folgende Elemente:

- Das **Sprint Planning** stellt den Start eines Sprints und soll die im Sprint zu erreichenden Arbeitsschritte definieren. Arbeitspakete werden quantitativ abgeschätzt und ggf. neu aufgeteilt.
- Im täglich zur gleichen Zeit stattfindenden **Daily Scrum** diskutieren die Entwickler den aktuellen Arbeitsstand, um mögliche Hindernisse zu lösen und kurze Entscheidungen zu treffen.
- Zum Ende des Sprints findet das **Sprint Review** statt, um Arbeitsergebnisse zu präsentieren und zu bewerten. Diese Erkenntnisse fließen in zukünftige Sprints und das Sprint Backlog mit ein.
- Die **Sprint Retrospective** schließt den Sprint ab. Ziel ist, grundlegendes Feedback über den abgelaufenen Sprint zu sammeln und darauf basierend Verbesserungsmaßnahmen abzuleiten.

Ein Scrum Team besteht aus drei Rollen:

- Der **Scrum Master** hat die Aufgabe sicherzustellen, dass die Scrum-Prinzipien umgesetzt werden und somit eine hohe Effektivität des Scrum Teams erreicht wird. Die Rolle existiert einmal pro Scrum Team.
- Ebenfalls einmal gibt es den **Product Owner**. Diese Rolle ist dafür verantwortlich, den Wert des Produkts zu maximieren, welches aus der Arbeit des Scrum Teams entsteht.
- Die **Entwickler** liefern die notwendigen Arbeitsergebnisse zum Erreichen des Sprintziels.

2.2.5 Heutiger Entwicklungsprozess

Als Kombination der drei zuvor vorgestellter Vorgehensmodelle wird ein beispielhafter Entwicklungsprozess für Fahrerassistenzsysteme vorgestellt. Der

Software-Entwicklungsprozess folgt dem im V-Modell (siehe 2.2.3) definierten Vorgehen und wird für jedes Release durchlaufen (siehe Abb. 2.11). Während der Entwicklung (linke Seite des V-Modells) kommt die Scrum-Methodik (siehe 2.2.4) zum Einsatz. Im Absicherungsprozess erfolgt für jede Testebene eine entsprechende Freigabe. Releases stellen Iterationen dar und erfolgen in fest vorgegebenen Zyklen entsprechend Projektplan für das Subprojekt Fahrerassistenzsysteme. Der Gesamtprojektplan basiert auf einem Stage-Gate-Modell (siehe 2.2.2). Releases der Fahrerassistenzsysteme werden auf den Gesamtprojektplan und auf Hardware-Musterstände abgestimmt.

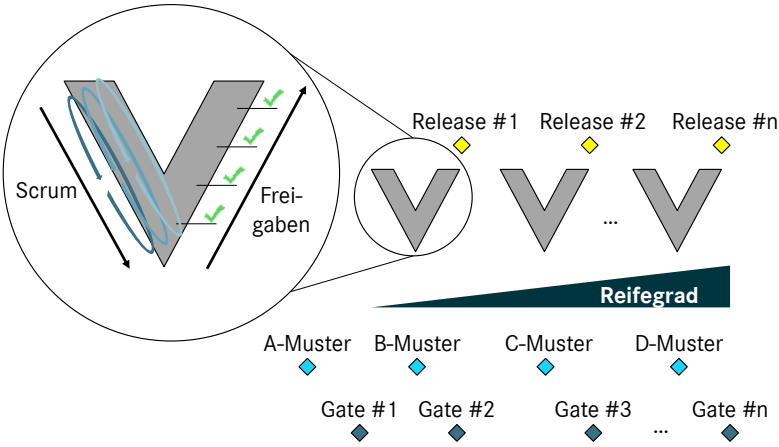

Abbildung 2.11: Übersicht des Entwicklungsprozesses für Fahrerassistenzsysteme

2.3 Normen und Standards

Basierend auf den zuvor eingeführten Vorgehensmodellen haben sich in den vergangenen 20 Jahren mehrere Normen und Standards für die Entwicklung von E/E-Systemen entwickelt. Die folgenden Normen und Standards sind im Bereich der Fahrerassistenzsysteme etabliert.

2.3.1 ISO 26262

Bei der Entwicklung jedes mechatronischen Systems in der Automobilindustrie steht die sogenannte funktionale Sicherheit im Fokus:

> **Definition Funktionale Sicherheit:** *„Die Abwesenheit unzumutbarer Risiken aufgrund von Gefahren, die durch fehlerhaftes Verhalten von elektrischen/elektronischen (E/E) Systemen verursacht werden.“* [15]

Seit dem Jahr 2011 existiert die Norm ISO 26262, welche auf der allgemeinen Norm IEC 61508 zur funktionalen Sicherheit aufbaut und sich speziell auf die funktionale Sicherheit während des kompletten Lebenszyklus von Serien-E/E-Systemen im Automobilbereich bezieht. Nach einer Überarbeitung im Jahr 2018 besteht sie aus nunmehr zwölf Teilen und stellt sowohl eine Referenz für konkrete technische Fragen wie auch den Entwicklungsprozess allgemein bereit. Grundlage ist das V-Modell für die Phasen der Produktentwicklung, welche in die separaten Teile Systementwicklung, Hardware-Entwicklung und Software-Entwicklung aufgeteilt ist. [15]

In der ISO 26262 werden für die entsprechenden Entwicklungsphasen jeweils mehrere Testebenen mit zugehörigen Testmethoden und -umgebungen genannt (siehe Abschnitt 2.5). Im Anschluss an den Entwicklungsprozess findet eine Validierung der funktionalen Sicherheit statt, welche neben statischen und dynamischen Testmethoden (siehe Abschnitt 2.4) auch Langzeit- bzw. Flottentests beinhalten kann. [15]

Ein zentrales Konzept der ISO 26262 ist die Risikobewertung anhand von sogenannten Automotive Safety Integrity Level (ASIL), welche eine Klassifikation eines betrachteten (Sub-)Systems hinsichtlich der benötigten Sicherheitsmaßnahmen und -anforderungen darstellt. Während QM die niedrigsten Sicherheitsanforderungen erfordert, steigen diese von ASIL A bis D an. Die Ermittlung der Einstufung wird in Abhängigkeit möglicher sicherheitskritischer Ereignisse vorgenommen. [15]

Die folgenden Aspekte finden hierbei Berücksichtigung [15]:

- **Schwere der Verletzungen einer beteiligten Person** (S0: keine Verletzungen; S3: lebensgefährliche/tödliche Verletzungen)
- **Eintrittswahrscheinlichkeit der Fahrsituation** (E0: unwahrscheinlich; E4: hohe Wahrscheinlichkeit)
- **Kontrollierbarkeit** (C0: allgemein kontrollierbar; C3: schwer kontrollierbar/unkontrollierbar)

Anhand der in Abb. 2.12 dargestellten Logik wird eine Einstufung des ASIL vorgenommen. Das ASIL hat unmittelbaren Einfluss auf das Vorgehen bei allen Entwicklungsschritten und der dabei eingesetzten Methoden. Insbesondere die empfohlenen Methoden für die Ableitung von Testfällen und die genutzten Testmethoden variieren stark in Abhängigkeit des ASIL. [15]

Schwere	Auftretens-wahrscheinlich-keit	Kontrollierbarkeit		
		C1	C2	C3
S1	E1	QM	QM	QM
	E2	QM	QM	QM
	E3	QM	QM	A
	E4	QM	A	B
S2	E1	QM	QM	QM
	E2	QM	QM	A
	E3	QM	A	B
	E4	A	B	C
S3	E1	QM	QM	A
	E2	QM	A	B
	E3	A	B	C
	E4	B	C	D

Abbildung 2.12: Tabelle zur Ermittlung des ASIL nach ISO 26262 [15]

2.3.2 Automotive SPICE

Ein in Entwicklungsprojekten der Automobilindustrie genutzter Standard ist Automotive SPICE, welcher 2005 vom Verband der Automobilindustrie (VDA) erstmalig herausgegeben und zuletzt 2017 aktualisiert wurde. Zweck des Standards ist es, eine Prüfung („Assessment“) der Entwicklungsprozesse durchführen zu können. Hierfür definiert Automotive SPICE ein sogenanntes Process Assessment Model (PAM) und ein Process Reference Model (PRM). Im PRM werden Entwicklungsprozesse definiert und in die drei Kategorien primäre Prozesse in Anlehnung an das V-Modell (siehe Abb. 2.13), unterstützende Prozesse und organisationsbezogene Prozesse aufgeteilt. Mithilfe der im Standard ISO/IEC 33020 definierten zu messenden Prozesscharakteristiken nimmt das PAM eine Bewertung der im PRM definierten Prozesse vor. [14]

Dabei wird die Prozessfähigkeit in Form eines Levels von 0 (Prozess nicht implementiert oder erfüllt nicht seinen Zweck) bis 5 (Prozess wird kontinuierlich verbessert) bewertet. Die Bewertung der Prozessfähigkeit basiert auf der Erfüllung von neun Prozessattributen, welche von „nicht erreicht“ bis „vollständig erreicht“ in mehreren Abstufungen eingestuft werden können. Um ein entsprechendes Level zu erreichen, müssen die Attribute der vorherigen Levels stets vollständig erreicht werden und die des jeweiligen Levels mindestens mit „größtenteils“ bewertet werden. [14]

Bis zum Level 1 werden sogenannte Prozessperformanceindikatoren bewertet, welche sich auf die Ergebnisse des Prozesses beziehen. Für alle darüber hinausgehenden Levels werden Prozessfähigkeitsindikatoren herangezogen, die die Umsetzung der Prozessattribute bewerten. Die Bewertung erfolgt jeweils auf Basis von Nachweisen wie Arbeitsprodukten oder Aussagen beteiligter Personen. [14]

2.3.3 ISO 21448

In den vergangenen Jahren hat sich gezeigt, dass der in der ISO 26262 definierte Begriff der funktionalen Sicherheit und die angewandten Methoden zur Identifikation von Risiken für Fahrzeuge mit Automatisierungsgrad ab SAE Level 1 nicht ausreichend sind. Hintergrund ist insbesondere die fehlende Berücksichtigung der durch Umweltsensoren hervorgerufenen Fehler. [97] Aus

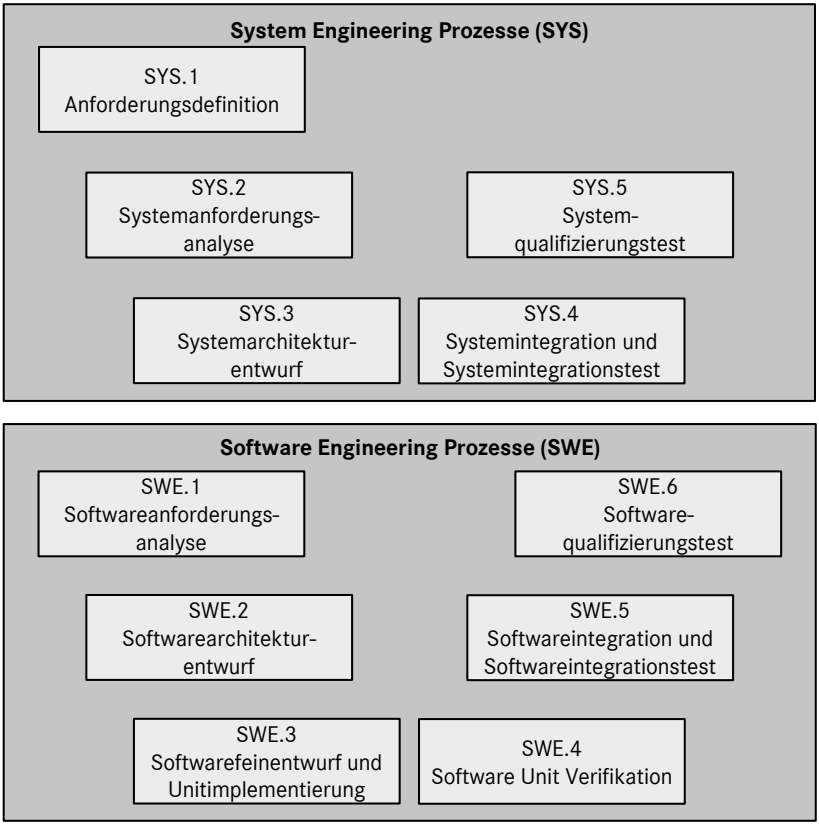

Abbildung 2.13: Auswahl der primären Prozesse des Automotive SPICE Process Reference Models [14]

diesem Grund wurde in Ergänzung zur funktionalen Sicherheit die sogenannte „Safety of the Intended Functionality“ (SOTIF) definiert, welche in dem im Jahr 2022 herausgegebenen Standard ISO 21448 behandelt wird und wie folgt definiert ist:

> **Definition SOTIF:** *„Die Abwesenheit unzumutbarer Risiken aufgrund von Gefahren, die von funktionalen Unzulänglichkeiten der vorgesehenen Funktionalität ausgehen.“* [97]

SOTIF umfasst insbesondere folgende Gründe für Gefahren: Funktionale Mängel (inkl. Algorithmen basierend auf künstlicher Intelligenz), fehlerhafte bzw. unzureichende Gestaltung der Mensch-Maschine-Schnittstelle, voraussehbarer fehlerhafter Gebrauch, externe Einflüsse durch Kommunikationsschnittstellen oder Umwelteinflüsse. Gleichzeitig ist die SOTIF klar abgegrenzt von der funktionalen Sicherheit (siehe 2.3.1), Fehlern die direkt auf Systemfehler zurückzuführen sind und auch Cybersecurity, welche durch den Standard ISO 21434 abgedeckt ist. [97] SOTIF wird ebenfalls im Standard ISO/TR 4804 [98] referenziert, der sich auf die Entwicklung, Verifikation und Validierung von ADS bezieht.

In der ISO 21448 werden die folgenden Definitionen der Begriffe Szene, Szenario und Use Case in Analogie zu [18] vorgenommen:

> **Definition Szene:** *„Momentaufnahme der Umgebung, welche die Szenerie, dynamische Elemente, die Selbstrepräsentation aller Teilnehmer und Beobachter sowie die Beziehung zwischen diesen Entitäten umfasst.“* [97]

> **Definition Szenario:** *„Beschreibung der zeitlichen Entwicklung zwischen mehreren Szenen in einer Sequenz von Szenen.“* [97]

In Abb. 2.14 ist dargestellt, wie die drei definierten Begriffe miteinander zusammenhängen.

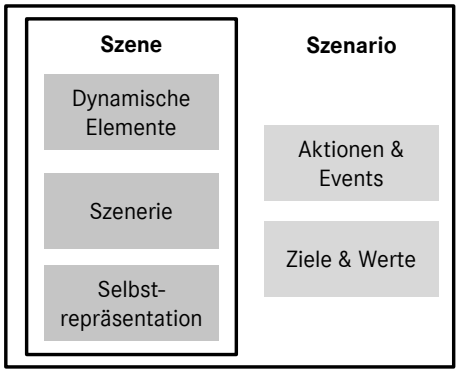

Abbildung 2.14: Schematische Darstellung von Szene und Szenario [18]

Grundsätzliche Annahme der ISO 21448 ist, dass von der SOTIF abgedeckte Gefahren durch ein Auslöseevent inkl. vorhersehbarem fehlerhaftem Gebrauch des Systems ausgelöst werden. In Kombination mit den Szenarioumständen kann es zu einem gefährlichen Event kommen. Ist ein gefährliches Event nicht durch Beteiligte oder andere externe Maßnahmen zu kontrollieren, kann daraus ein Schaden entstehen. [97]

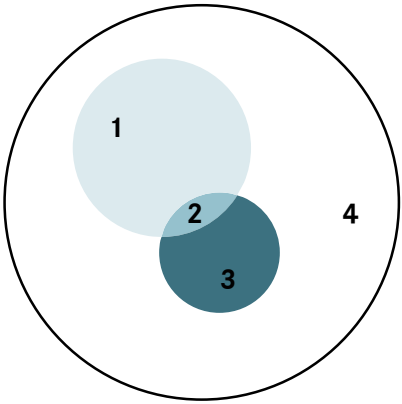

Abbildung 2.15: Kategorien von Szenarien nach ISO/PAS 21448 [97]

Die dafür relevanten Szenarien werden in die folgenden vier Kategorien eingeteilt [97] (siehe Abb. 2.15):

1. Bekannte ungefährliche Szenarien
2. Bekannte gefährliche Szenarien
3. Unbekannte gefährliche Szenarien
4. Unbekannte ungefährliche Szenarien

Ziel der ISO 21448 ist es, die Bereiche 2 und 3 durch einen Prozess zu Risikoevaluation zu identifizieren und zu minimieren [97].

2.4 Testen im E/E-Entwicklungsprozess

Der Test nimmt auf unterschiedlichen Ebenen des V-Modells (siehe 2.2.3) einen wichtigen Teil des Entwicklungsprozesses ein. In dieser Dissertation liegt der Schwerpunkt auf dynamischen Testmethoden, bei denen eine Ausführung des Testobjekts stattfindet.

2.4.1 Begriffsdefinitionen

Auch in diesem Kapitel wird zunächst eine Definition der wichtigsten Begrifflichkeiten vorgenommen. Die zentrale Aktivität des Testens ist wie folgt definiert:

> **Definition Test:** *Testen ist das systematische Prüfen mit a-priori definierten Testeingängen und Ausführungsbedingungen sowie der Vergleich mit erwarteten Ergebnissen.* [86] [80]

Davon abzugrenzen ist das Erproben:

> **Definition Erproben:** *„In Erproben steckt das Ausprobieren in einer unbekannten Situation. [. . .] Erproben [basiert] eher auf der Analyse des Gesamten als dem Auswerten spezieller Messwerte. [. . .] Außerdem liegt einer Erprobung eine vergleichsweise geringe Systematik zur Grunde.“* [86]

Ein Testfall ist definiert wie folgt:

> **Definition Testfall:** *„Ein Satz an Testeingängen, Ausführungsbedingungen und erwarteten Ergebnissen, welcher für ein bestimmtes Ziel entwickelt wurde. Ziel kann unter anderem die Ausführung eines bestimmten Programmpfades oder die Verifikation einer spezifischen Anforderung sein.“* [80]

In Analogie zur oben genannten Definition besteht ein Testfall nach Spillner aus einzuhaltenden Vorbedingungen, Eingabedaten, erwarteten Ergebnissen und erwarteten Nachbedingungen [99].

Zentrales Element eines Tests ist das sogenannte System Under Test (SUT), auch Testobjekt genannt:

> **Definition System Under Test (SUT):** *„Im Zentrum [. . .] des Tests [. . .] steht das Testobjekt [oder SUT]. Abhängig von der Entwicklungsphase einer Komponente oder eines Systems ändert sich die Beschaffenheit des Testobjektes. So kann dieses eine Softwarekomponente, eine Hardware oder auch ein im Fahrzeug verbautes eingebettetes System sein.“* [86]

In dieser Dissertation liegt der Fokus auf dem Test von Software, die optional in ein eingebettetes System integriert sein kann. Tests können anhand unterschiedlicher Aspekte eingeteilt werden. Eine Möglichkeit ist die Gruppierung in funktionale und nichtfunktionale Tests:

> **Definition Funktionaler Test:** *„Alle Testverfahren bzw. -methoden, mittels derer das von außen sichtbare Ein-/Ausgabeverhalten eines Testobjekts geprüft wird. [. . .] als Testbasis bzw. Referenz für das Sollverhalten dienen funktionale Anforderungen.“* [99]

> **Definition Nichtfunktionaler Test:** *„Beschreiben Attribute des funktionalen Verhaltens, also wie gut bzw. mit welcher Qualität das (Teil-)System seine Funktion erbringen soll.“* [99]

Attribute können hierbei unter anderem die Leistungsfähigkeit (Reaktionszeit, Lastabhängigkeit, Verhalten unter Überlast), Stabilität/Robustheit, Benutzerfreundlichkeit, Dokumentation oder Wartbarkeit des Systems sein. Herausforderung ist oft die uneindeutige oder nur implizite Definition der Testanforderungen. [99]

Ein statischer Test ist nach ISO/IEC/IEEE 29119 definiert:

> **Definition Statischer Test:** *„Test, in dem das Testobjekt verglichen wird mit einer Auswahl an Qualitätskriterien oder anderen Kriterien, ohne dass es zu einer Ausführung des Codes kommt.“* [100]

Die Durchführung von statischen Tests kann sowohl manuell als auch mithilfe von Testwerkzeugen (z.B. bei Formaler Verifikation [101]) durchgeführt werden. Außerdem muss der Test nicht auf den Softwarecode beschränkt sein,

sondern auch noch andere Arbeitsprodukte wie beispielsweise die Spezifikation, Testdokumentation oder Projektpläne mit einbeziehen. [99] Da es zu keiner Ausführung des Codes kommt, werden statische Tests in der Regel auf niedrigeren Ebenen des V-Modells für einzelne Subsysteme ausgeführt.

Die Definition eines dynamischen Tests nach ISO/IEC/IEEE 29119 lautet wie folgt:

> **Definition Dynamischer Test:** *„Test, der die Ausführung des Codes erfordert."* [100]

2.4.2 Dynamische Tests

Ein wichtiger Aspekt des dynamischen Testens ist die Testabdeckung a_t, welche den Anteil der tatsächlich getesteten Testfälle an der Gesamtzahl der möglichen Testfälle beschreibt [102]:

$$a_t = (\frac{n_t}{n_{t,ges}} \cdot 100)\% \tag{2.1}$$

n_t: Anzahl ausgeführte Testfälle
$n_{t,ges}$: Anzahl mögliche Testfälle

Nicht immer kann eine vollständige Testabdeckung erreicht werden, ebenso wie die Anzahl möglicher Testfälle bei einigen Testmethoden nur mit hohem Aufwand ermittelt werden kann [103]. Analog wird die Anforderungsabdeckung a_a wie folgt definiert [102]:

$$a_a = (\frac{n_a}{n_{a,ges}} \cdot 100)\% \tag{2.2}$$

n_a: Anzahl durch ausgeführte Testfälle abgedeckte Anforderungen
$n_{a,ges}$: Anzahl Anforderungen gesamt

Dynamische Tests werden anhand der Methode der Testfallgenerierung in Black-Box-Tests und White-Box-Tests eingeteilt.

2.4.2.1 Black-Box-Tests

Ein Black-Box-Test (zum Teil auch als referenziert als anforderungsbasierter Test [99]) nutzt die Spezifikation des extern beobachtbaren Verhalten des SUT und kann sich sowohl auf funktionale als auch nichtfunktionale Eigenschaften beziehen. Dabei bleibt die innere Struktur des SUT unberücksichtigt [99]. Dies entspricht dem Vorgehen auf höheren Testebenen des V-Modells wie Systemtest oder Akzeptanztest. Für die Generierung der Testfälle werden unter anderem die folgenden Methoden genutzt:

- **Äquivalenzklassentest**: Hier werden die Definitionsbereiche so strukturiert, dass für jede Äquivalenzklasse ein ähnliches Systemverhalten erwartet wird, sodass entsprechend auch pro Äquivalenzklasse ein konkreter Testfall bei beliebiger Parameterwahl erstellt wird. Liegen mehrdimensionale Inputs vor, so werden die Definitionsbereiche zunächst unabhängig voneinander strukturiert und anschließend zu Äquivalenzklassen verschmolzen. [103]
- **Grenzwertanalyse**: Grundsätzlich ist diese Methode ähnlich zum Äquivalenzklassentest, allerdings werden speziell die Randbereiche der Parameterverteilungen zur Testfallgenerierung gewählt. Im Fall mehrdimensionaler Inputs werden auch Parameterkombinationen getestet, bei denen nur ein einzelner Input im Randbereich liegt. [103] [101]
- **Use-Case-Test**: Testfälle für den Systemtest werden hier basierend auf möglichen Anwendungsfällen des Systems abgeleitet. Dabei können auch Use-Case-Diagramme zur Beschreibung der Use Cases zum Einsatz kommen. [103]
- **Kombinatorisches Testen**: Liegen Inputs als diskrete Parameterverteilung vor, so ist es möglich, aus deren Kombinationen Testfälle abzuleiten. Eine Erhöhung der Parameterzahl führt zu einem sehr starken Wachstum der Anzahl möglicher Testfälle. Das kombinatorische Testen ermöglicht die Reduktion von Testfällen mithilfe der paarweisen Kombination oder auch der Variation einzelner Parameter. [103] [104]

2.4.2.2 White-Box-Tests

Im Gegensatz zu Black-Box-Tests sind bei White-Box-Tests interne Strukturen des SUT wie beispielsweise der Kontrollfluss oder Aufrufhierarchien bekannt und werden explizit zur Testfalldefinition verwendet [99]. Die Methode bietet sich somit vor allem beim Testen von reinem Softwarecode an, wie er in den unteren Testebenen des V-Modells existiert.

White-Box-Tests lassen sich weiter unterteilen in kontrollflussorientierte und datenflussorientierte Tests. Erstgenannte betrachten nur die internen Berechnungspfade, nicht jedoch die Beschaffenheit der Daten. Der Kontrollfluss wird in Form von Knotenflussdiagrammen modelliert, wobei Knoten Anweisungen darstellen und Kanten Verzweigungsbedingungen entsprechen. Der datenflussorientierte Test betrachtet zusätzlich die Beschaffenheit der Daten und bedarf daher einer tieferen Analyse des SUT. [103]

Daneben wird in der ISO/IEC/IEEE 29119 auch das erfahrungsbasierte Testen in Form von „Error Guessing“ genannt. Hierbei werden basierend auf einer Liste von bekannten möglichen Fehlern Testfälle definiert. [102]

2.4.2.3 Weitere Testarten

Neben dem Black-Box-Test und dem White-Box-Test existieren noch Sonderformen, welche sich nicht direkt einer der beiden Kategorien zuordnen lassen. Eine solche Form ist der **Grey-Box-Test**, welcher auf einem Black-Box-Test basiert, jedoch zusätzliches Wissen über Teile der Programmstruktur zur Testfallgenerierung nutzt.

In einem **Regressionstest** wird das SUT erneut getestet, nachdem eine Modifikation des SUT stattgefunden hat [99]. Der Test soll einen Abgleich der Funktionalität mit der vorherigen Version des SUT schaffen [90]. Da hierbei die gleichen Testfälle wie zuvor verwendet werden sollen, müssen diese wiederholbar sein und im Idealfall auch automatisiert durchführbar. Je nach Modifikation des SUT können auch Anpassungen der Testfälle notwendig sein. In der Regel werden bei einem Regressionstest nicht alle Testfälle wiederholt, sondern es findet eine Priorisierung oder eine Einschränkung auf bestimmte Varianten bzw. Teilsysteme statt. [99]

Im Gegensatz dazu werden bei einem **Back-to-Back-Test** unterschiedliche Implementierungen des gleichen SUT getestet. Hierbei kann es sich einerseits um die gleiche Software handeln, die aufgrund von hohen Sicherheitsanforderungen von unterschiedlichen Teams parallel entwickelt wird [103]. Andererseits ist auch die Ausführung des SUT in unterschiedlichen Testumgebungen wie z.B. MiL und SiL als Back-to-Back-Test denkbar.

Bei einem **Fehlerinjektionstest** werden durch eine Manipulation bewusst Fehler in das SUT eingebracht. Somit kann der Einfluss von möglichen Fehlern im Realbetrieb und die Reaktion des SUT evaluiert werden. [15]

2.5 Testumgebungen

Im Kontext der mechatronischen Systeme in der Automobilindustrie und im Speziellen der Fahrerassistenzsysteme spielen sogenannte X-in-the-Loop (XiL)-Testumgebungen eine bedeutende Rolle. Daneben ist auch der Realtest im Fahrzeug weiterhin von großer Bedeutung, wobei hier der Übergang von XiL-Umgebungen fließend ist.

2.5.1 Realtest/Erprobung

Realtests stellen ein zentrales Element der Absicherungsstrategie von Fahrerassistenzsystemen dar [37] [105], was auch in entsprechenden Normen wie der ISO 26262 [15] und Richtlinien wie dem RESPONSE Code of Practice [35] hinterlegt ist. Die Homologation wird ebenfalls über Realtests vorgenommen [106]. Entscheidender Vorteil im Vergleich zu XiL-Tests (siehe 2.5.2) ist, dass das reale System in einer realen Umgebung getestet wird, sodass keine Abweichungen aufgrund von simulierten oder emulierten Komponenten existieren. Nachteilig ist jedoch, dass aufgrund von wirtschaftlichen Rahmenbedingungen die Quantität an Realtests eingeschränkt ist, sodass keine vollständige Testabdeckung aller Fahrzeugvarianten bzw. Szenarien erreicht wird. Der Begriff Erprobung wird verwendet, wenn es sich um ein gesamtheitliches, weniger systematisches Testen handelt. [86]

Sowohl in Forschungsprojekten wie EuroFOT [107] oder PEGASUS [108] als auch in Serien-Entwicklungsprojekten beim Original Equipment Manufactu-

rer (OEM) befinden sich die eingefahrenen Realtestkilometer im Millionenbereich [109]. Auch neue Hersteller mit dem Ziel, hochautomatisierte Systeme zu entwickeln, führen Feldtests im großen Stil durch. So beläuft sich die gefahrene Gesamtstrecke beispielsweise bei Google-Tochter Waymo auf 5 Millionen Meilen bis zum Jahr 2018. [110]

Allgemein lassen sich Realtests einteilen in Fahrten auf dem Prüfgelände und Feldtests. Diese können mit trainierten oder untrainierten Fahrern stattfinden. Trainierte Fahrer werden normalerweise in früheren Entwicklungsstadien eingesetzt, da sie speziell darin geschult sind, ein mögliches Fehlverhalten des Systems zu erkennen und präzises Feedback zu geben. Zudem haben sie eine große Erfahrung mit unterschiedlichen Systemen und können diese daher gut vergleichen. Untrainierte Fahrer sind dann sinnvoll, wenn das System bereits einen höheren Reifegrad hat. Ziel ist hier, die Interaktion des Systems mit repräsentativen „durchschnittlichen“ Fahrern zu bewerten. Da im Nutzfahrzeugbereich der allergrößte Teil Berufsfahrer ist, werden Kunden gezielt in die Erprobung mit einbezogen. Hierbei wird einer bestimmten Kundengruppe ein Vorseriensystem bereitgestellt, um Feedback hinsichtlich der kundenspezifischen Einsatzbereiche zu erhalten. [35]

2.5.1.1 Prüfgelände

Aktivitäten auf dem Prüfgelände beziehen sich vorwiegend auf den gezielten Test von Szenarien, die in Testfallkatalogen definiert sind. Eine besondere Herausforderung beim Test von Fahrerassistenzsystemen im Vergleich zu anderen mechatronischen Systemen ist die Tatsache, dass neben dem eigenen Zustand zusätzlich der Zustand anderer Objekte bzw. der eigene Zustand in Relation zu diesen betrachtet werden muss. Ebenso ist bei menschlichen Fahrern nur eine begrenzte Reproduzierbarkeit von Szenarien gegeben, da Streuungen bei der Regeltätigkeit des Menschen existieren. Dies ist beim systematischen Test von Szenarien zu berücksichtigen. [108] Ansätze zur koordinierten Steuerung der beteiligten Fahrzeuge über Roboter sollen für eine höhere Reproduzierbarkeit sorgen [111]. Die Aufzeichnung mehrerer Objekte erfordert eine leistungsfähige Referenzmesstechnik, welche heutzutage häufig auf Differential-GPS basiert [108]. Um die Folgen möglicher Kollisionen zu reduzieren, werden sogenannte Dummies bzw. Soft Targets für die Repräsentation anderer Fahrzeuge oder Fußgänger verwendet [107] [111] [112]. Wichtig ist es, dass Soft

Targets möglichst realitätsnahe Optik, Geometrie [112] und Radarreflexionseigenschaften [113] besitzen, um durch alle verwendeten Sensortechnologien korrekt erfasst werden zu können. So werden beispielsweise Radarreflexionsfolien und Eckreflektoren für die Erkennung durch Radarsensoren angebracht. Zur Erkennung durch Kameras hingegen werden Bilder auf die Soft Targets geklebt. [113] Für Fußgänger gibt es spezielle Dummies, die optional für einen höheren Realitätsgrad Arm- und Beinbewegungen ausführen [114]. Abb. 2.16 zeigt eine typische Szene eines Realtests auf dem Prüfgelände.

Abbildung 2.16: Realtest auf dem Prüfgelände, Daimler Truck AG

2.5.1.2 Feldtest

Besonderer Vorteil von Feldtests ist der maximale Realitätsgrad, da alle Wechselwirkungen zwischen SUT und Testumgebung zum Tragen kommen. Dies umfasst insbesondere die Reaktion der Umwelt auf das Verhalten des SUT. [16] Zudem treten in Feldtests Szenarien auf, die über die bereits bekannten Szenarien hinausgehen und mit der systematischen Variation von Parametern nicht generierbar sind [115].

Grundsätzlich werden Feldtests über alle Entwicklungsphasen und Reifegrade des Systems genutzt. In früheren Entwicklungsstadien werden vorrangig spezifische Einzelaspekte des Systemverhaltens im Feld untersucht. Später finden im Rahmen der Erprobung größere Testumfänge im Feld statt, sodass sich im Laufe der Entwicklung einer FAS-Generation mehrere Millionen Testkilometer akkumulieren. [16] Die in Feldtests verwendeten Fahrzeuge benötigen eine spezielle Zulassung und sind meist mit zusätzlicher Messtechnik ausgestattet. [107] Wichtiger Aspekt bei Feldtests in der Erprobung ist die Wahl der Strecke. Diese sollte einen repräsentativen und ausgewogenen Querschnitt über die möglichen Einsatzorte und -szenarien des späteren Serienprodukts darstellen. [16] Dabei kann eine Optimierung basierend auf aggregierten Informationen wie Karten- oder Verkehrsdaten erfolgen [116].

Die Auswertung von Daten aus Feldtests bezieht sich entweder auf die Analyse einzelner Events oder liefert Kennzahlen zur Leistungsfähigkeit des Systems über die gesamte Testkampagne. Das Systemverhalten in einzelnen Events wird durch Funktionsexperten bewertet und kann direkt an die Entwickler zurückgespielt werden. Dies ist insbesondere zur Beantwortung spezifischer Fragestellungen in frühen Entwicklungsphasen notwendig. Hingegen ist es bei größeren Datensätzen möglich, das Systemverhalten gesamtheitlich zu bewerten. Kennzahlen können die Häufigkeit kritischer Szenarien oder der Eingriffe des Fahrerassistenzsystems sein. [16] Die Ermittlung der Kennzahlen findet dabei automatisiert statt [117]. Ebenso können Rückmeldungen von Fahrern über Fragebögen berücksichtigt werden [16]. Neben dem Einsatz von Feldtests im Rahmen von Entwicklungsprojekten werden Felddaten in Forschungsprojekten wie euroFOT [118] oder L3Pilot [119] ausgewertet, um den quantitativen Einfluss von Fahrerassistenzsystemen auf die Verkehrssicherheit zu ermitteln.

2.5.2 X-in-the-Loop

Die Bezeichnung XiL ist ein Sammelbegriff für alle Testumgebungen mit geschlossenem Regelkreis, bei denen das SUT und/oder die Umwelt zumindest teilweise nur in abstrahierter Form existieren. Das „X“ repräsentiert die Art, in der das SUT eingebunden ist. [23] Damit der Regelkreis geschlossen ist, bedarf es stets einer Repräsentation der Umgebung, sodass eine XiL-Umgebung von einer Open-Loop-Simulation abzugrenzen ist [120].

XiL-Umgebungen existieren in verschiedenen Ausprägungen (siehe 2.5.2.2). So kann das SUT sowohl als reine Software wie auch als reales Steuergerät eingebunden sein [25]. Grundsätzlich gilt, dass in höheren Testebenen des V-Modells weniger Simulationskomponenten und mehr reale Komponenten in XiL-Umgebungen zum Einsatz kommen [15]. Dennoch ist der strukturelle Aufbau ähnlich. In Anlehnung an [23] und [25] stellen folgende Komponenten einer XiL-Umgebung die Referenz in dieser Dissertation dar (siehe Abb. 2.17):

- **FAS/ADS-ECU(s)**: Beinhaltet die Software zur Umsetzung der FAS bzw. ADS. Je nach XiL-Ausprägung ist die Software auf Hardware-Muster integriert. Diese Komponente stellt somit SUT und Regler der XiL-Umgebung dar.
- **Andere ECUs**: Alle weiteren mit dem SUT kommunizierenden Steuergeräte bzw. die Restbuskommunikation.
- **Fahrzeug**: Weitere Aktorik und Fahrdynamik des Fahrzeugs.
- **Umwelt**: Straße sowie statische und dynamische Objekte.
- **Sensoren**: Alle Sensoren zur Umweltwahrnehmung.
- **Fahrermodell/Szenariosteuerung**: Modellbasierte oder deterministische Steuerung des Fahrers und anderer Objekte.

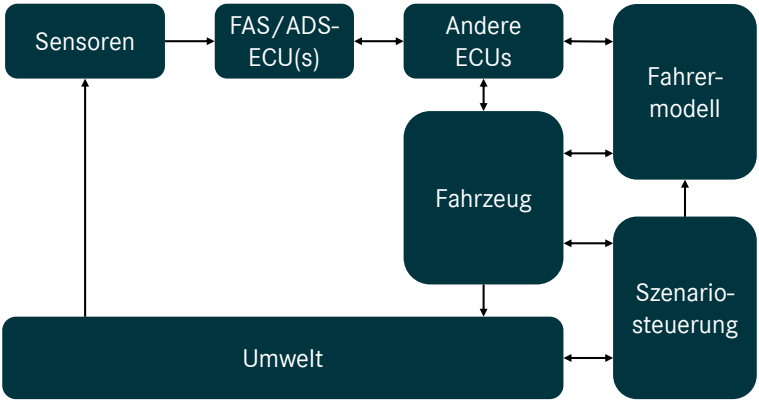

Abbildung 2.17: Übersicht von XiL-Komponenten

Eine Möglichkeit zur strukturierten Einordnung von XiL-Umgebungen ist die Klassifikation der Einbindung von SUT und Umgebungskomponenten nach Schuldt. Den in Abb. 2.17 dargestellten Komponenten werden entsprechend ihrer Ausprägung in der jeweiligen XiL-Umgebung die Diskretisierungsstufen simuliert, emuliert oder real vorhanden zugewiesen [25]. Eine simulierte Komponente besteht aus einem reinen Simulationsmodell, während eine emulierte Komponente durch eine nachahmende Hardware mit entsprechenden Schnittstellen repräsentiert ist. Mit dieser Strukturierung lässt sich grundsätzlich eine Vielzahl möglicher Kombinationen generieren. Abb. 2.18 listet die in der praktischen Anwendung vorwiegend vorkommenden Ausprägungen auf.

	MiL	SiL	PiL	HiL	DiL	ViL
SUT	S (Simulationsmodell)	S (ECU-Code)	E	R	S	S/R
Andere ECUs	S	S	E	E/R	E/R	R
Fahrzeug/ Fahrdynamik	S	S	S	S	S/E	R
Umwelt	S	S	S	S	S	S/R
Sensoren	S	S	S	S	S	S/R
Fahrermodell/ Szenariosteuerung	S	S	S	S	R	R

Abbildung 2.18: Übersicht von XiL-Ausprägungen in der Praxis (S: simuliert, E: emuliert, R: real)

2.5.2.1 Vorteile und Herausforderungen von X-in-the-Loop-Umgebungen

XiL-Umgebungen bieten gegenüber dem Realtest folgende Vorteile:

- **Reduktion Implementierungsaufwand**: Der Aufwand zur Implementierung eines neuen Testfalles ist bei XiL-Umgebungen niedriger als im Realtest, da es sich hierbei in den meisten Fällen um reine Softwareanpassungen handelt, sofern die XiL-Ausprägung kein reales Fahrzeug oder Teststrecke beinhaltet. [121]
- **Reduktion Durchführungsaufwand**: Die Durchführung selbst ist in XiL-Umgebungen mit weniger Aufwand verbunden und kann über ent-

sprechende Tools automatisiert werden [86]. Je nach Repräsentation des SUT ist eine Testgeschwindigkeit schneller als Echtzeit möglich [23]. Der Realtest erfordert den Einsatz von Testfahrern und/oder den Aufbau von Soft Targets. [121] [111]

- **Reduktion Unfallrisiko**: Bei XiL-Tests ohne reale Testumgebung entfällt das Risiko von Kollisionen. [23]
- **Reproduzierbarkeit**: Realtests sind durch Nichtdeterminismus charakterisiert. Tests in XiL-Umgebungen lassen sich hingegen reproduzierbar und annähernd deterministisch[1] durchführen. [23]

Maßgebliche Herausforderung bei der Nutzung von XiL-Umgebungen ist die Sicherstellung der Glaubwürdigkeit (siehe 3.3.1). Ist die Konfidenz der XiL-Umgebung für ein spezifisches Szenario nicht ausreichend, so sind die dafür generierten Testergebnisse nicht belastbar. [25] [26] Für die Validierung kompletter XiL-Umgebungen existieren bislang nur wenige Ansätze (siehe 3.3.5).

2.5.2.2 X-in-the-Loop-Ausprägungen

Model-in-the-Loop (MiL) Ein MiL-Test kommt vor allem bei der modellbasierten Entwicklung zum Einsatz. Hierbei wird der Programmcode nicht von Hand geschrieben, sondern es werden mithilfe eines entsprechenden Modellierungswerkzeuges Simulationsmodelle erstellt, aus denen der Programmcode automatisiert generiert wird. Zum Test wird das Simulationsmodell des SUT als Vorstufe zum Programmcode ausgeführt. Da das Simulationsmodell bereits vorliegt, sind Simulationsschnittstellen einfach zu bedienen und ist die Umgebung schnell und flexibel anpassbar. [86] Es bestehen grundsätzlich keine Echtzeitanforderungen an das Modell und die Ausführung kann auf dem Entwicklungsrechner mit rein simulierten Komponenten stattfinden. Gängige Tools zur Modellierung bzw. Simulation sind beispielsweise MATLAB Simulink [123] oder dSPACE TargetLink [124].

[1] XiL-Umgebungen können auch Nichtdeterminismus beinhalten, der durch Simulationstools oder die Netzwerkkommunikation bedingt ist. Die vorliegenden Abweichungen befinden sich jedoch in einer kleineren Größenordnung als bei Realtests. [122] [310]

Software-in-the-Loop (SiL) SiL-Tests sind grundsätzlich ähnlich zu MiL-Tests, mit dem Unterschied, dass hier bereits der kompilierte Zielcode ausgeführt wird [86]. Es ist möglich, den gleichen Compiler zu verwenden, der auch auf der Ziel-ECU verwendet wird [120]. Neben der Ausführung des reinen Anwendungscodes kann eine SiL-Simulation auch eine Simulation der Laufzeitumgebung beinhalten, welche z.B. im Falle des AUTOSAR-Standards durch die dort existierende Laufzeitumgebung (Run-Time Environment, RTE) spezifiziert ist [125]. Hierbei werden beispielsweise die Funktionsaufrufe einzelner AUTOSAR-Ports bereits mitsimuliert. Die Ausführung kann auch hier auf dem Entwicklungsrechner stattfinden, wobei in Abhängigkeit der verwendeten Tools eine Kopplung mehrerer Rechner zur Co-Simulation erforderlich sein kann. Dies ist insbesondere der Fall, wenn z.B. die Umgebungssimulation auf einem separaten Rechner durchgeführt wird [23]. Auch hier bestehen grundsätzlich keine Echtzeitanforderungen und alle Komponenten werden simuliert, sodass auch schneller als in Echtzeit gerechnet werden kann. Zur Simulation der SUT-Software kommen beispielsweise dSPACE VEOS [126] oder Synopsys Silver [127] zum Einsatz.

Processor-in-the-Loop (PiL) Bei PiL wird die kompilierte Zielsoftware auf einem Emulator des im Steuergerät verwendeten Prozessors simuliert [120] [128]. Schnittstellen des SUT und die Basissoftware werden emuliert, während die restlichen Komponenten in Echtzeit simuliert werden. Ziel von PiL ist es insbesondere, Laufzeiten und Speicherbedarfe zu verifizieren. [120]

Hardware-in-the-Loop (HiL) Im Anwendungsfall HiL ist die Software auf einem Hardware-Muster integriert, sodass das SUT real vorliegt und die Schnittstellen des SUT real bedient werden müssen [120]. Die folgenden zwei HiL-Anwendungsfälle sind in der industriellen Anwendung relevant [120]:

- Im **Komponententest** des SUT werden alle anderen Steuergeräte emuliert. Die entsprechende Netzwerkkommunikation wird entweder aus Signalen von Simulationsmodellen aufgebaut oder durch eine Restbussimulation emuliert. Bei der Restbussimulation liegt der Fokus darauf, dem SUT gültige Bussignale bereitzustellen, ohne dass diese zwingend funktional korrekt sein müssen. [23]

- Beim **HiL-Integrationstest** des gesamten Systems liegen alle Steuergeräte real vor und werden miteinander vernetzt, sodass deren Integration in das gesamte E/E-System getestet werden kann. Weitere Komponenten werden weiterhin simuliert. [120] [25]

Unabhängig vom Anwendungsfall müssen beim HiL-Test eingesetzte Simulationsmodelle echtzeitfähig sein. Dabei sind die Tools für die Simulation der Umgebung sehr ähnlich zum Anwendungsfall SiL.

Driver-in-the-Loop (DiL) Eine DiL-Umgebung wird auch als Fahrsimulator bezeichnet, da hier der Fokus auf der Mensch-Maschine-Interaktion liegt [23]. Unterschied zu den zuvor genannten Testumgebungen ist, dass ein realer Fahrer existiert und das restliche Fahrzeug je nach Ausprägung real oder emuliert ist [25]. Fahrsimulatoren variieren von statischen Aufbauten mit wenigen Cockpit-Elementen bis hin zu dynamischen Fahrsimulatoren, die ein vollständiges Fahrzeug bewegen [120]. Die Umgebung wird stets simuliert, wodurch es möglich ist, die Mensch-Maschine-Interaktion auch in kritischen Szenarien gefahrlos zu simulieren und zu bewerten. Zudem ermöglicht eine DiL-Umgebung eine bessere Steuerbarkeit, Reproduzierbarkeit und Messbarkeit [23].

Vehicle-in-the-Loop (ViL) Bei ViL kommt ein reales Fahrzeug auf einem realen Prüfgelände zum Einsatz, sodass der Fahrer ebenfalls real vorhanden ist. Besonderheit hier ist, dass die Umgebung zum Teil simuliert wird und dem Fahrer die entsprechenden Objekte via Augmented Reality Brille eingeblendet werden. Die simulierten Objekte werden ebenfalls an das SUT weitergegeben [23]. Großer Vorteil dieser Methode ist das verminderte Unfallrisiko, da Objekte nur simuliert werden, während die Fahrdynamik real abgebildet ist [25].

2.5.2.3 Simulationsmodelle

Fahrzeugmodelle Ein Fahrzeugmodell in diesem Kontext umfasst sowohl die Fahrdynamik als auch alle weiteren mechanischen Komponenten, die über unterschiedliche Schnittstellen mit Steuergerätemodellen verbunden sein können. Die Simulation des Fahrzeugs bzw. dessen Subsysteme wird bereits seit

Jahrzehnten in über 50 Anwendungsfällen in der Automobilindustrie genutzt. [129]. Dementsprechend groß ist die Varianz an Simulationsmodellen.

Für den Anwendungsfall Fahrautomatisierung ist hauptsächlich die Fahrphysik relevant, da die Bewegung des Ego-Fahrzeugs in der Umgebung korrekt abgebildet sein muss [23]. Die Fahrdynamik lässt sich in die Längs-, Quer- und Vertikaldynamik einteilen, repräsentiert durch das in Abb. 2.19 dargestellte Koordinatensystem nach ISO 8855 [130]. Die einzelnen Achsen/Winkel sind definiert wie folgt:

- X: Längsachse
- Y: Querachse
- Z: Vertikalachse
- φ: Wankwinkel
- θ: Nickwinkel
- ψ: Gierwinkel

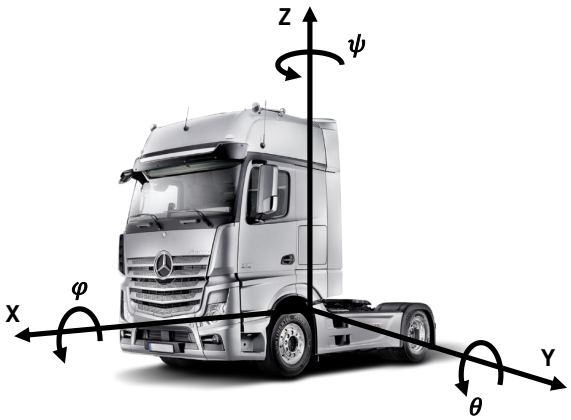

Abbildung 2.19: Fahrzeugkoordinatensystem nach ISO 8855 [130]

Die Längsdynamik ist durch die vier Fahrwiderstände Rollwiderstand, Luftwiderstand, Steigungswiderstand und Beschleunigungswiderstand charakterisiert, deren Summe die am Rad aufzubringende Zugkraft ergibt [131]. Die Wirkkraft am Rad ist einerseits limitiert durch die Reibung an den Reifenaufstandspunkten und ist andererseits abhängig von der Leistung des Antriebsstrangs (incl. Verluste) bzw. des Bremssystems [64]. Fahrzeugmodelle können dabei separate Simulationsmodelle für Motor, Getriebe und Bremssystem enthalten [132].

Maßgeblich für die Querdynamik sind vor allem die am Rad/Reifen umgesetzten Querkräfte, welche aus einem durch Lenken hervorgerufenen Schräglaufwinkel resultieren. Aufgrund seiner Einfachheit wird häufig das Einspurmodell für die Beschreibung der Querdynamik verwendet. [69]

Die Betrachtung der Vertikaldynamik ist vor allem dann relevant, wenn Schwingungen durch Anregungen über die Straße analysiert werden sollen [69], was im Rahmen von Simulationen für die Fahrautomatisierung von untergeordneter Bedeutung ist.

Steuergerätemodelle/Restbusmodelle Für die korrekte Stimulation des SUT müssen Signale bzw. Botschaften von anderen Steuergeräten, die direkte Schnittstellen zum SUT haben, durch die Simulationsumgebung bereitgestellt werden. Hierfür gibt es zwei Möglichkeiten:

- **Source-Code**: Der vollständige Source-Code eines anderen Steuergeräts wird kompiliert und in die Simulationsumgebung integriert. Das Verhalten des Steuergeräts ist damit sehr realitätsnah abgebildet. Wird das Steuergerät nach AUTOSAR-Standard entwickelt, so kann mit entsprechenden Tools eine sogenannte vECU generiert werden, die neben der Anwendungssoftware Teile der AUTOSAR-Basissoftware modelliert. [133]
- **Rebuild-Modell**: Werden nicht alle Funktionen eines anderen Steuergeräts in der realen Tiefe benötigt, so bietet es sich an, ein vereinfachtes Rebuild-Modell zu nutzen. Dieses kann entweder von den jeweiligen ECU-Entwicklern oder den Testingenieuren entwickelt werden. [134] [135]

Liegt das SUT in realer Form mit Hardware vor, so werden für eine Simulation reale Busbotschaften benötigt, welche aus den Signalen der Simulationsmodelle generiert werden können. Eine Ausnahme bildet die reine Restbussimulation:

- **Restbussimulation**: Eine Restbussimulation kommt infrage, wenn für ein Signal keinerlei Funktion benötigt wird, dessen Botschaft jedoch vom SUT im Rahmen der Buskommunikation erwartet wird. Über ein entsprechendes Tool wird die Buskommunikation zeitlich und inhaltlich so abgebildet, dass diese für das SUT gültig erscheint. [86]

Fahrermodelle Fahrermodelle sollen das menschliche Fahrverhalten möglichst realitätsgetreu nachbilden und können sowohl für den Fahrer des Ego-Fahrzeugs als auch für die Fahrer anderer Fahrzeuge eingesetzt werden. Es existieren Folgemodelle für die Längsführung und Spurwechselmodelle für die Querführung [136].

Folgemodelle geben entweder eine direkte Beschleunigungsanforderung oder alternativ eine einzuregelnde Sollgeschwindigkeit aus. Inputs sind die Geschwindigkeiten von Ego-Fahrzeug und vorausfahrendem Fahrzeug sowie der Abstand zum vorausfahrenden Fahrzeug. Analytische Folgemodelle wie das Krauss-Modell [137], das Gipps-Modell [138] oder das Intelligent Driver Model [139] beinhalten eine rein mathematische Beschreibung, die auf kinematischen Betrachtungen des Fahrverhaltens basiert. Im Gegensatz dazu betrachten psycho-physikalische Folgemodelle wie die Modelle nach Wiedemann [140] und Fritzsche [141] die menschliche Wahrnehmung und definieren verschiedene Fahrzustände, welche jeweils ein abweichendes Fahrverhalten zur Folge haben.

Spurwechselmodelle sind normalerweise verknüpft mit Folgemodellen, da sie deren Inputs nutzen, um zu ermitteln, ob eine Spurwechselmotivation vorliegt. Ist dies der Fall, wird der Spurwechsel realisiert über die Prozessschritte Lücke suchen, Spurwechsel vorbereiten und Spurwechsel durchführen. [142]

Auch für andere Verkehrsteilnehmer wie Fußgänger und Radfahrer existieren Simulationsmodelle, welche jedoch stark vom Fokus der jeweiligen Simulation abhängen. Ebenso existieren Fahrermodelle, welche spezielle Aspekte modellieren (z.B. Aggressivität). [143]

Umweltsimulation Im Bereich der Verkehrssimulation werden in Abhängigkeit von der Repräsentation der Verkehrsteilnehmer drei unterschiedliche Abstraktionsgrade unterschieden [142] [143]:

- **Makroskopisch**: Die Modellierung basiert auf fluiddynamischen Modellen und nutzt Durchschnittswerte für die Beschreibung des Verkehrs, wodurch große Mengen an Fahrzeugen effizient modelliert werden können. Da sich die Erkenntnisse jedoch auf den Verkehrsfluss beschränken und das Verhalten einzelner Fahrzeuge außer Acht lassen, sind makroskopische Verkehrssimulationen für XiL-Umgebungen ungeeignet.
- **Mikroskopisch**: Hier werden einzelne Verkehrsteilnehmer und deren Kooperation untereinander modelliert, indem separate Modelle für Fahrzeug und Fahrer zum Einsatz kommen. Dementsprechend ist der Bedarf an Rechenleistung hier zwar höher, der Detailgrad der Simulation ist jedoch deutlich höher und entspricht den Erfordernissen einer XiL-Umgebung.
- **Mesoskopisch**: Dies stellt eine Kombination aus makroskopischer und mikroskopischer Verkehrssimulation dar, indem nur relevante Bereiche mit hohem Detailgrad modelliert werden. Der Rest der Simulation basiert auf einer makroskopischen Simulation. Dies entspricht einem Kompromiss aus benötigter Rechenleistung und modellierbarem Detailgrad.

Für Anwendung in XiL-Umgebungen wird im Weiteren von einer mikroskopischen Verkehrssimulation ausgegangen. Hierfür kommen spezielle Umweltsimulatoren zum Einsatz (siehe 2.5.2), welche als kommerzielles Produkt [144] [145] oder in Open-Source-Form [146] erhältlich sind. Etablierte Tools für die Umgebungssimulation sind VIRES Virtual Test Drive [144], IPG CarMaker [145]/TruckMaker [147] oder Simcenter Prescan [148]. Aber auch neuere Anbieter wie Applied Intuition [149] bieten entsprechende Tools an, genauso wie das Open Source Tool CARLA [146] zum Teil in der Industrie verwendet wird.

Je nach Simulator ist es möglich, sowohl statische Objekte (Gebäude, Schilder, Pflanzen) als auch dynamische Objekte (Fahrzeuge, Fußgänger, Radfahrer) und die Straßengeometrie bzw. -oberfläche zu modellieren. Dabei kann auf hinterlegte Modelle und Elemente zurückgegriffen werden, was den Model-

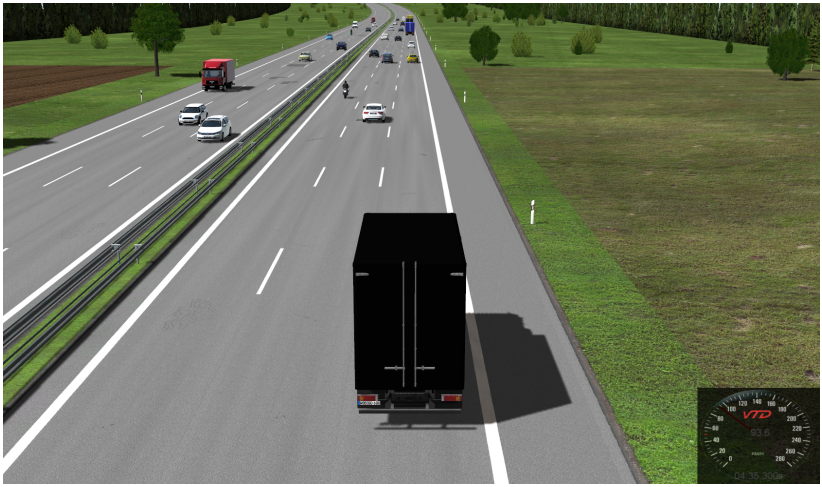

Abbildung 2.20: Szene aus der Umweltsimulation Virtual Test Drive

lierungsaufwand reduziert [144]. Mithilfe von Rendering[2] ist es möglich, das simulierte Szenario visuell auszugeben (siehe Abb. 2.20), was insbesondere für kamerabasierte Systeme von Relevanz ist [151].

Sensormodelle Für den Test von Fahrautomatisierung in einer XiL-Umgebung ist es notwendig, Umweltsensoren durch Modelle abzubilden, welche die Schnittstelle zwischen Umweltsimulation und SUT darstellen [152]. Sensormodelle bilden grundsätzlich alle in 2.1.2 genannten Arten und Wirkprinzipien von Sensoren ab [153]. Auch wenn die Schnittstellen der Sensormodelle zum SUT ähnlich sind, lassen sich hinsichtlich des modellierten Detailgrades drei unterschiedliche Arten von Sensormodellen klassifizieren [154].

Ideale Sensormodelle, zum Teil auch Ground-Truth-Modelle genannt, liefern die in der Simulation existenten Objektattribute direkt als Output, sodass keine Berücksichtigung sensorspezifischer Effekte, Fehler oder Störungen erfolgt [155]. Lediglich das Sichtfeld des Sensors wird berücksichtigt [156]. Somit

[2] Rendering bezeichnet die Methoden, um ein dreidimensionales Objekt auf einem zweidimensionalen Ausgabegerät/Bildschirm darzustellen. [150]

ist bei idealen Sensormodellen im Vergleich zu anderen Sensormodellarten eine größere Abweichung zur Realität zu erwarten [154]. Hingegen ist die benötigte Rechenleistung geringer, sodass Echtzeitfähigkeit gegeben ist und sich die Modelle zur Einbindung in eine XiL-Umgebung eignen. Inputs und Outputs eines idealen Sensormodells basieren jeweils auf Objektlisten. [156]

Phänomenologische Sensormodelle werden auch als High-Level-Sensormodelle [157], Black-Box-Sensormodelle [158] oder probabilistische Sensormodelle [156] referenziert und berücksichtigen sensorspezifische Effekte auf einer phänomenologischen Ebene. Dies bedeutet, dass die Auswirkungen bestimmter Effekte auf den Sensor bekannt sind und im Modell berücksichtigt werden [154]. Es wird allerdings nicht der physikalische Prozess des jeweiligen Effekts exakt modelliert, sondern entweder über probabilistische Ansätze [157] [156] [158] oder vereinfachte physikalische Gleichungen [159] approximiert. Beispiele für modellierte Effekte sind:

- Verlieren von Objekten/False Negatives [159]
- Geisterobjekte/False Positives [160]
- Abhängigkeit des Sichtfeldes von der Objektklasse [160]
- Unsicherheit/Rauschen der Distanz-/Geschwindigkeitssignale in Abhängigkeit von der Position [159]
- Wetterabhängigkeit [159] [154]
- Sonneneinstrahlung [154]

Der benötigte Rechenaufwand von phänomenologischen Sensormodellen ist analog zum abgebildeten Detailgrad größer als bei idealen Sensormodellen. Jedoch sind Echtzeitfähigkeit und Nutzbarkeit in XiL-Umgebungen auch hier gegeben. [154] Ebenfalls basieren Inputs und Outputs von phänomenologischen Sensormodellen auf Objektlisten [156].

Physikalische Sensormodelle, auch bezeichnet als White-Box-Modell [158] oder Low-Level-Sensormodell [157], bilden das Wirkprinzip des jeweiligen Sensors im Detail nach [156] und müssen daher für jeden Sensortyp individuell modelliert werden [158] [156]. Zum Einsatz kommen hierfür insbesondere Ray-Tracing-Ansätze, welche sich beispielsweise zur Modellierung der Wellenausbreitung von Radar- oder Lidarsensoren eignen [154]. Für Kamerasen-

sormodelle findet hingegen ein Rendering der Umgebung statt [161], auf das zusätzlich optische Effekte der Linse angewendet werden können [160].

Erst physikalische Sensormodelle erlauben eine realitätsgetreue Nachbildung von Effekten wie beispielsweise Okklusionen oder die Sensitivität des Radarquerschnitts bezüglich der Objektausrichtung und -position bei Radarsensoren [162]. Kameramodelle erlauben eine Bewertung der Computer Vision Algorithmen des realen Sensors, wenn dem Modell ein gerendertes Bild bereitgestellt wird [161]. Die deutlich erhöhte Leistungsfähigkeit physikalischer Sensormodelle erfordert jedoch eine signifikant höhere Rechenleistung, was in Echtzeitanwendungen einen limitierenden Faktor darstellen kann [154]. Sowohl Input- als auch Outputschnittstelle variieren in Abhängigkeit des Sensormodells. So ist es beispielsweise möglich, statt einer Objektliste auch Sensorrohdaten aus einem physikalischen Sensormodell auszugeben, um Teile der Sensoralgorithmen in einem nachgelagerten Schritt testen zu können [156].

2.5.2.4 Simulationsschnittstellen

Standardisierte Schnittstellen reduzieren den Integrationsaufwand von Simulationsmodellen in unterschiedlichen XiL-Umgebungen [163].

Functional Mock-up Interface Der Functional Mock-up Interface (FMI)-Standard der Modelica Association wurde erstmals im Jahr 2010 veröffentlicht und zuletzt im Jahr 2022 aktualisiert (FMI 3.0). [164] Ziel ist es, eine standardisierte Schnittstelle für Simulationskomponenten unabhängig von der konkreten Simulationsumgebung bereitzustellen. Der Standard erlaubt die Co-Simulation von mehreren Modellen, welche in sogenannte Functional Mock-up Units (FMU) verpackt und als solche ausgeführt werden können. Eine FMU besteht zum einen aus einer XML-Datei, welche Informationen über die Inputs, Outputs und Parameter sowie optional weiterer interner Variablen enthält. Zum anderen beinhaltet eine FMU eine ausführbare Binärdatei (.dll oder .so) und/oder den dazugehörigen C-Code. FMUs können sowohl mit eigenem Solver (FMI for Co-Simulation und FMI for Scheduled Execution) wie auch mit dem Solver der Zielumgebung (FMI for Model Exchange) gerechnet werden. [164] Eine Übersicht der Schnittstellen von FMI 3.0 ist in Abb. 2.21 zu sehen.

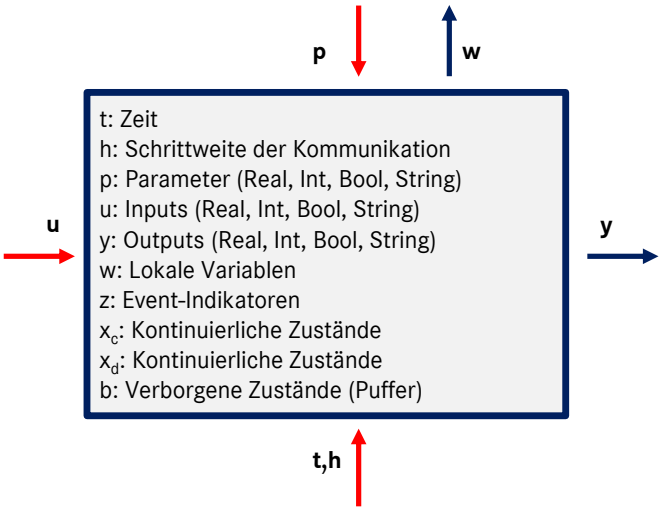

Abbildung 2.21: Schnittstellen einer Functional Mock-up Unit nach FMI 3.0 [164]

Open Simulation Interface Die Simulation von Sensoren und das Verarbeiten von Sensordaten stellen Bereiche von wachsender Bedeutung im Kontext der XiL-Umgebungen dar. Im Rahmen des PEGASUS-Projekts [165] wurde mit dem sogenannten Open Simulation Interface (OSI) ein Standard für die Einbindung von Sensormodellen entwickelt. Grundlage bilden drei Schnittstellendefinitionen, welche für unterschiedliche Stellen der Prozesskette vorgesehen sind (siehe Abb. 2.22) [166]:

- **Ground Truth**: Beinhaltet alle verfügbaren Daten aus Sicht des globalen Koordinatensystems.
- **Sensor View**: Beinhaltet Daten aus Sicht des Sensorkoordinatensystems, welche aus Ground Truth abgeleitet werden. Sensor View dient als Input für physikalische Sensormodelle.
- **Sensor Data**: Beinhaltet Daten aus Sicht des Sensorkoordinatensystems, welche aus Ground Truth abgeleitet werden und dient als Input für phänomenologische oder ideale Sensormodelle bzw. kann auch direkt an das SUT weitergegeben werden.

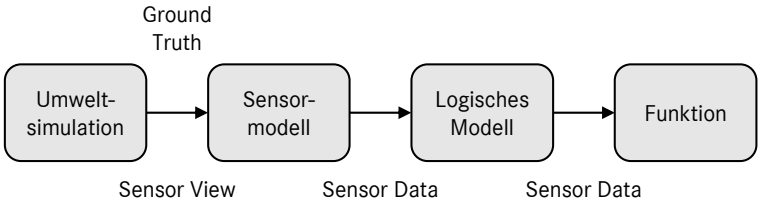

Abbildung 2.22: Prozesskette nach Open Simulation Interface [166]

Für jede Schnittstelle liefert OSI eine objektbasierte Umweltbeschreibung, welche auf Google Protocol Buffers basiert [167]. Dies erlaubt es, Daten hierarchisch zu strukturieren und damit die Verarbeitung in unterschiedlichen Tools zu standardisieren. Die Umwandlung der Daten zwischen den Schnittstellen findet durch entsprechende Sensor- bzw. Logikmodelle statt. [166]

2.5.2.5 Anwendungsfälle von XiL in einem bestehenden Entwicklungsprozess

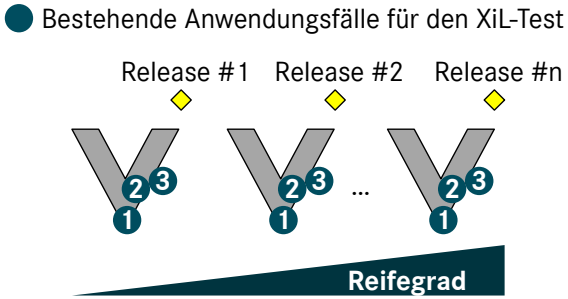

Abbildung 2.23: Einordnung bestehender Anwendungsfälle 1, 2 und 3 für den XiL-Test

Basierend auf dem in 2.2.5 definierten Prozess werden bereits heute Testumfänge in XiL-Umgebungen durchgeführt. Dies beinhaltet insbesondere die folgenden Anwendungsfälle (siehe Einordnung in Abb. 2.23):

1. **Unittests mit Testfallkatalog**: Hierbei kommt eine **SiL**-Umgebung zum Einsatz, welche jedoch nur eine stark vereinfachte Stimulation der Unit bereitstellt und somit nicht vergleichbar ist mit einer Umweltsimulation (siehe 2.5.2). Bei Modellbasierter Entwicklung eignet sich auch eine **MiL**-Umgebung. Eine Durchgängigkeit zwischen verschiedenen Testebenen ist hier nur sehr eingeschränkt gegeben, da Testfälle jeweils nur auf Unit-Ebene definiert sind.

2. **Modultests als manueller Entwicklertest**: In diesem Fall werden sowohl **SiL**- wie auch **HiL**-Umgebungen verwendet. Eine Umweltsimulation (siehe 2.5.2) existiert bereits, bietet aber nur einen eingeschränkten Funktionsumfang und ist somit auf simple Testfälle beschränkt. Neben der manuellen Einregelung von Szenarien via Graphical User Interface (GUI) ist eine teilautomatisierte Testfalldurchführung via Testfallkatalog möglich.

3. **Modultest mit Testfallkatalog**: Hier sind grundsätzlich **SiL**- und **HiL**-Umgebungen einsetzbar. Die Umweltsimulation entspricht dem Aufbau in 2.5.2. Der hier genutzte Testfallkatalog ist konsistent zu dem des Realtests und ermöglicht somit die Durchgängigkeit zu höheren Testebenen.

3 Stand der Wissenschaft und Technik

3.1 Absicherung und Test der Fahrautomatisierung

3.1.1 Feldtests und Extremwerttheorie

Sowohl Wachenfeld [105] als auch Kalra und Paddock [168] kommen basierend auf entsprechenden Abschätzungen in ihren Publikationen zum Schluss, dass alleine über Feldtest während des Entwicklungszeitraumes keine Absicherung von automatisierten Fahrsystemen mit vertretbarem Aufwand realisierbar ist. Beide nehmen an, dass eine Absicherung den statistischen Nachweis mit entsprechender Konfidenz erfordert, dass das System mindestens so sicher ist wie ein menschlicher Fahrer. Auch wenn die Argumentation auf Systeme ab SAE Level 3 abzielt, lässt sie sich auf alle Automatisierungslevel übertragen.

Grundlage der Überlegung in [105] und [168] ist, dass sicherheitskritische Events mit einer Poisson-Verteilung modellierbar sind, da sie unabhängig voneinander mit einer konstanten mittleren Rate eintreten [169]. Betrachtete Größe ist die sogenannte Fehlerrate λ [168]:

$$\lambda = \frac{n_e}{d_{ges}} \tag{3.1}$$

n_e: Anzahl der Events
d_{ges}: gefahrene Distanz

Als Event kommen Unfälle unterschiedlichen Schweregrades infrage, wobei tödliche Unfälle im Fokus stehen, da sie statistisch am besten erfasst sind und damit zur Berechnung der Fehlerrate eines durchschnittlichen menschlichen Fahrers herangezogen werden können. Ziel ist der statistische Nachweis, dass das zu testende System eine geringere Fehlerrate als ein durchschnittlicher menschlicher Fahrer hat. Das hierbei zu wählende Konfidenzniveau entspricht der minimalen Wahrscheinlichkeit, dass das System tatsächlich besser ist als ein menschlicher Fahrer, wenn der Test erfüllt wird. [168]

Gleichzeitig muss der Feldtest so viele Testkilometer sammeln, dass im Fall eines nicht bestandenen Tests das Konfidenzniveau für eine tatsächlich schlechtere Leistungsfähigkeit des Systems im Vergleich zu einem menschlichen Fahrer ausreichend groß ist. Basierend auf beiden Zielwerten für die statistische Sicherheit und Annahmen für die (unbekannte) Leistungsfähigkeit des Systems lässt sich die Anzahl der erforderlichen Testkilometer errechnen. [105]

Beispielrechnungen von Wachenfeld bzw. Kalra und Paddock mit entsprechenden Annahmen für System und durchschnittlichen Fahrer führen zu erforderlichen Testumfängen von 2,3 Milliarden km [105] bzw. 11 Milliarden Meilen [168]. Die beiden Werte sind aufgrund unterschiedlicher zugrunde liegender Daten bzw. Annahmen nicht vergleichbar, zeigen jedoch beide auf, dass der statistische Nachweis zwei bis drei Größenordnungen über aktuellen Testkampagnen liegt [105].

Um dennoch einen statistischen Nachweis für die Sicherheit eines Systems ab SAE Level 3 erbringen zu können, schlägt Wachenfeld eine Abwandlung des heutigen Freigabeprozesses mit drei Phasen vor [105]:

In der Testphase (1) wird darauf verzichtet, Systeme bereits so abzusichern, dass ein statistischer Nachweis erbracht ist, dass das System eine geringere Fehlerrate hat als ein menschlicher Fahrer. Stattdessen wird die Testphase nur dann nicht bestanden, wenn in dieser ein statistischer Nachweis mit entsprechendem Konfidenzniveau erbracht werden kann, dass das System unsicherer ist als ein menschlicher Fahrer. [105] In der Einführungsphase (2) sollen Fahrzeuge kontrolliert in den Realverkehr gebracht werden, ohne dass ein speziell geschulter, sogenannter Sicherheitsfahrer das System überwacht. Somit können schneller Testkilometer gesammelt werden, als dies in der Testphase der Fall ist. Ziel ist es, einen statistischen Nachweis zu erbringen, dass das System entweder eine größere oder kleinere Fehlerrate hat als ein durchschnittlicher menschlicher

Fahrer. Während der anschließenden Überwachungsphase (3) wird die Leistungsfähigkeit des Systems weiterhin überwacht und der finale Nachweis über die Systemsicherheit erbracht. [105]

Eine weitere Möglichkeit, Realdaten weitergehend zu verwerten, ist die Nutzung der Extremwerttheorie. Die Idee dabei ist, basierend auf einer Kritikalitätsmetrik die Wahrscheinlichkeit für eine tatsächliche Kollision zu extrapolieren [170] [171]. Asljung et al. nutzen hierbei die Kritikalitätsmetriken Time-to-Collision (TTC) und Brake Threat Number (BTN). Lokale Maxima von TTC und BTN bzw. die Maxima innerhalb eines diskreten Zeitintervalls werden jeweils durch Poissonverteilungen approximiert. Aus diesen kann anschließend extrapoliert werden, wie groß die zu fahrende Distanz ist, um eine Kollision zu beobachten. [170]

3.1.2 Open-Loop Resimulation

Bereits heute werden Fahrdaten aufgezeichnet, um als Input für die Simulation weiterverarbeitet zu werden. Dies findet sowohl in Serienfahrzeugen statt [17] als auch in Prototypen für höhere Automatisierungslevels [110] [172]. Vorteil dieser Anwendung in der Simulation ist die große Realitätsnähe der Daten.

3.1.2.1 A-posteriori Resimulation

Ein naheliegender Ansatz ist es, die aufgezeichneten Daten direkt in das SUT zur Resimulation einzuspeisen. Vorteil hier ist, dass die Simulationsinputs dabei realistischen Daten entsprechen, aber gleichermaßen reproduzierbare Tests ermöglichen. Dadurch ist es beispielsweise möglich, schnell und einfach Regressionstests durchzuführen. Es ist jedoch zu berücksichtigen, dass es sich um keinen geschlossenen Regelkreis handelt, da die Outputs des SUT keinen Einfluss auf das Fahrverhalten und damit auf die eingespeisten Sensordaten haben. Dementsprechend sind insbesondere bei allen Systemen ab SAE Level 1 Abweichungen zwischen aufgezeichneten Sensordaten und dem tatsächlichen Fahrverhalten zu erwarten. [173] Im Fall von Systemen, die nur warnen oder informieren bzw. nur in Notfallsituationen eingreifen, lässt sich zumindest bis zur Warnung bzw. dem Eingriff annehmen, dass die Sensordaten realitätsgetreu sind. Im Gegensatz zum Test der gesamten Software ist es möglich, eine

Resimulation nur für die Validierung von Subsystemen wie beispielsweise dem Bereich Sense (siehe 2.1.2.1) einzusetzen [37].

Entscheidend für die Möglichkeiten einer Resimulation ist, in welcher Form Daten existieren. Liegen die Daten von Sensoren und anderen Steuergeräten im Bus-Format vor (z.B. CAN, Ethernet), so können diese direkt in das entsprechende SUT bzw. dessen Software zur Resimulation eingespeist werden [174] [175] [176]. Werden hingegen Sensorrohdaten aufgezeichnet, so ist es möglich, die Sensor- bzw. Wahrnehmungsalgorithmen zu evaluieren [177].

Zudem ist es möglich, aus Bus-Daten Trajektorien der beteiligten Fahrzeuge zu extrahieren und zur Szenariogenerierung zu nutzen [176] [178]. Dies kann durch die Aufzeichnung von Ground-Truth-Daten mit entsprechender Referenzmesstechnik ergänzt werden [179] [176]. Dabei ist eine spätere Manipulation der Ground-Truth-Daten durch räumliche und zeitliche Offsets möglich [176].

3.1.2.2 Live Resimulation

Im Gegensatz zur a-posteriori Resimulation der aufgezeichneten Fahrdaten ist es möglich, diese bereits im Fahrzeug einem dort installierten SUT zuzuführen und somit einen direkten Abgleich mit dem Verhalten des menschlichen Fahrers durchzuführen. Die sogenannte VAAFO-Methode greift diesen Ansatz auf, indem während der Fahrt im Abstand von 0,5 s jeweils Simulationen für eine Dauer von 2 s gestartet werden [180]. Ähnlich ist die Methode „Shadow Mode“ von Tesla [17]. Ein weiterer Ansatz dieser Art wird vorgestellt in [181], hier findet die Simulation jedoch nachträglich statt und wird nur durchgeführt, wenn sich das SUT anders verhalten würde als der menschliche Fahrer. Durch die jeweils kurze Simulationsdauer fallen die Abweichungen hinsichtlich des Fahrverhaltens und damit der Simulationsinputs weniger stark aus, als wenn eine kompletter aufgezeichneter Track resimuliert wird. [181]

3.1.2.3 Reactive-Replay

Um die genannten Abweichungen hinsichtlich des Fahrverhaltens zu reduzieren, schlägt Bach die sogenannte Reactive-Replay Methode vor, welche die

aufgezeichneten Daten von der zeitlichen Dimension entkoppelt und stattdessen auf die longitudinale Position des Ego-Fahrzeugs projiziert [182]. Somit entsteht in longitudinaler Richtung ein geschlossener Regelkreis, da die Bewegung anderer Objekte in Abhängigkeit von der Bewegung des Ego-Fahrzeugs stattfindet, während in lateraler Richtung weiterhin ein offener Regelkreis existiert.

3.1.3 Formale Verifikation und Statistische Absicherung

Ein Absicherungsansatz für Systeme ab SAE Level 3 wird in [183] vorgestellt. Durch eine Aufteilung in die Subsysteme Algorithmus und Sensorik können diese getrennt und damit effizient abgesichert werden. Die Absicherung des Algorithmus basiert auf der formalen Verifikation (statischer Test) anhand entsprechender Regeln für das normative Systemverhalten. Es wird davon ausgegangen, dass das Verhalten des Systems so konzipiert ist, dass dieses keine Kollision oder gefährliche Situation verursacht. Entsprechend werden Grundregeln für den Algorithmus definiert [183]:

- Eine Kollision mit einem anderen Verkehrsteilnehmer von hinten (Auffahren) muss ausgeschlossen werden.
- Es darf nicht rücksichtslos eingeschert werden.
- Vorfahrt wird gewährt, nicht genommen.
- Vorsichtiges Verhalten in Bereichen mit eingeschränkter Sicht muss sichergestellt werden.
- Ein Unfall muss verhindert werden, solange dadurch kein anderer Unfall verursacht wird.

Ein ähnlicher Ansatz zur Absicherung des Algorithmus mit formalen Methoden findet sich in [184], wobei hier das Verhalten von Robotern im Fokus steht.

Im Gegensatz zum Algorithmus wird die Sensorik statistisch abgesichert. Dabei wird aus einem Gesamtziel für die Zuverlässigkeit der Sensoren für jeden Sensor ein individuelles Zuverlässigkeitsziel abgeleitet, welches die Redundanz der Sensoren berücksichtigt. Relevante Zielgröße ist einerseits die Häufigkeit sicherheitskritischer Sensorfehler. Andererseits wird aus den Abwei-

chungen bei der Sensorgenauigkeit ein Maß für den Komfortverlust definiert. [183]

3.1.4 Szenariobasiertes Testen

Insbesondere in europäischen Forschungsprojekten wie PEGASUS [108], ENABLE-S3 [185], VVM [186] oder SET Level [187] stellt der szenariobasierte Ansatz die Grundlage der Forschungsaktivitäten dar. Darüber hinaus werden Elemente des Ansatzes bereits in der Industrie genutzt [110].

Grundidee ist es, durch eine systematische Beschreibung der im Realverkehr auftretenden Szenarien eine ausreichend hohe und dennoch realisierbare Testabdeckung in Form von konkreten Szenarien zu erhalten [188]. Aus der Anforderung an die Realisierbarkeit gründet sich die Verwendung von Szenarien, da so redundante Testinputs als Äquivalenzklassen zusammengefasst werden können [151] [25]. Die Verwendung von Szenarien wird bereits in der ISO 26262 [15] sowie der ISO 21448 [97] zur Testfallgenerierung empfohlen.

Besonderer Fokus liegt darauf, diejenigen Szenarien zu testen, welche tatsächlich für die Bewertung des Systems bzw. der Funktion relevant sind. Mögliche Kriterien hierfür sind eine hohe Kritikalität oder ein tatsächlicher oder erwarteter Eingriff eines Fahrerassistenzsystems. [189]

3.1.4.1 Definitionen

Eine häufig zitierte und im Rahmen der ISO 21448 (siehe 2.3.3) eingeführte Referenz sind die Definitionen nach Ulbrich et al. [18], welche auf den Definitionen nach Geyer et al. [190] basieren. Diese werden im Rahmen dieser Dissertation verwendet. Darüber hinaus ist eine Situation wie folgt definiert:

> **Definition Situation:** *„Gesamtheit der Umstände, die berücksichtigt werden müssen, um ein angemessenes Verhaltensmuster zu einem bestimmten Zeitpunkt auszuwählen. Enthält alle für das Verhalten relevanten Bedingungen, Optionen und bestimmenden Faktoren.“* [18]

In Ergänzung wird in Abb. 3.1 und im Folgenden auf den Zusammenhang und die Spezifika der Begriffe eingegangen [18]:

- Da eine Szene eine Momentaufnahme der Umwelt sowie aller relevanten Elemente darstellt, kann diese nur in der Simulation vollständig erfasst werden, während in der Realität durch die eingeschränkte Perspektive der Beobachter nur eine unvollständige Erfassung stattfindet.
- Ähnlich ist es bei einer Situation, bei der Informationen einer Szene ausgewählt und ergänzt werden, sodass es sich um eine subjektive Beschreibung handelt.
- Sowohl Szene als auch Situation beziehen sich stets auf einen diskreten Zeitpunkt, während sich ein Szenario über einen Zeitraum erstreckt. Eine Szene ist damit gleichzeitig Subelement eines Szenarios.

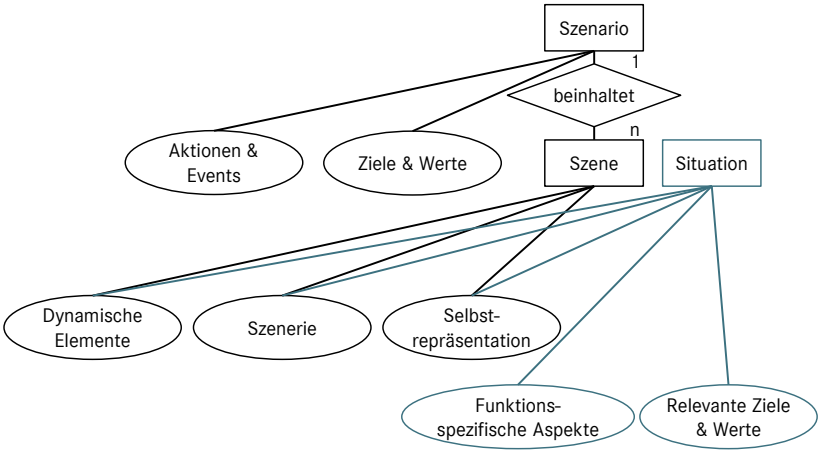

Abbildung 3.1: Zusammenhang der Begriffe Szenario, Szene und Situation nach [18]

3.1.4.2 Szenariobeschreibungen

Auch wenn Szenarien häufig in sprachlicher Form beschrieben werden, ist es für die weitere systematische Verarbeitung unerlässlich, Szenarien formalisiert

zu notieren. Die bestehenden Ansätze berücksichtigen jeweils unterschiedliche Aspekte und ergänzen sich zum Teil untereinander.

Zustandsgraph In Analogie zur Definition eines Szenarios nach Ulbrich modelliert Bach ein Szenario als Abfolge sogenannter Akte, welche als Repräsentation je einer Szene aufgefasst werden können. Ein Übergang zwischen Akten wird von Events ausgelöst, was über einen Zustandsgraphen mit entsprechenden Zuständen und bedingungsgetriggerten Übergängen modelliert ist. So kann eine Fahrzeugfolgefahrt mithilfe von Zustandswechseln modelliert werden, die in Abhängigkeit des Abstandes zum vorausfahrenden Fahrzeug bzw. dessen Existenz aktiviert werden. [191]

6-Ebenen-Modell Das 6-Ebenen-Modell basiert auf dem von Schuldt bzw. Bagschik publizierten 4-Ebenen-Modell [192] bzw. 5-Ebenen-Modell [193] und wurde im Rahmen des PEGASUS-Projekts auf nunmehr sechs Ebenen erweitert [108] [194], welche jeweils unterschiedliche Aspekte eines Szenarios repräsentieren. Hierbei existieren folgende Bezeichnungen für die jeweiligen Ebenen [195]:

1. Fahrbahn-Ebene: Beschreibt die Geometrie und die Oberfläche der Fahrbahn.
2. Straßenausstattung und Regeln: Beinhaltet Verkehrszeichen und Fahrbahnmarkierungen.
3. Zeitliche Änderungen und Ereignisse: modelliert temporäre Objekte und Veränderungen, die Einfluss auf den Fahrraum haben (z.B. Baustellen).
4. Bewegliche Objekte: Beinhaltet alle Verkehrsteilnehmer mit Bewegung und zusätzlichen Abhängigkeiten.
5. Umweltbedingungen: Deckt alle Umwelteinflüsse auf die Systemleistung ab (Licht, Wetter, Temperatur).
6. Digitale Informationen: Beinhaltet die Verfügbarkeit und Qualität der bereitgestellten Informationen (V2X, Digital Maps).

Ontologien Eine ebenfalls systematische Beschreibungsweise für Szenarien bieten Ontologien. In [193] werden beispielsweise drei Ontologie-Darstellungen zur Beschreibung der Ebenen 1 und 4 genutzt, sodass daraus eine Beschreibung einer Szene resultiert. Während die erste Ontologie die vorhandenen Elemente der Ebene 1 beschreibt, stellt die zweite Ontologie deren Beziehung zueinander dar. Die dritte Ontologie beinhaltet die Beziehung der beteiligten Verkehrsteilnehmer. [193] Ähnlichkeit dazu besitzt der Ansatz in [196], der die Ebenen 1 und 4 einer Szene in einer einzelnen Ontologie beschreibt.

Szenarioabstraktion Eine Beschreibung eines Szenarios und damit der Ausprägung der Szenarioinformationen kann auf unterschiedlichen Abstraktionsebenen vorliegen. Dabei existieren die folgenden drei Arten von Szenarien, sortiert von höchstem zu niedrigstem Abstraktionsgrad [19]:

- **Funktionales Szenario**: Das Szenario wird widerspruchsfrei auf einer semantischen Ebene beschrieben, sodass es von Menschen verstanden werden kann. Vokabular und Detailgrad der Beschreibung variieren abhängig vom Anwendungsfall.
- **Logisches Szenario**: Das Szenario wird über die Entitäten und deren Beziehungen in Form von Parameterbereichen quantitativ beschrieben. Optional können für die Parameter statistische Verteilungen sowie Korrelationen und numerische Bedingungen definiert werden.
- **Konkretes Szenario**: Das Szenario wird über die Entitäten und deren Beziehungen in Form von festen Werten beschrieben.

Da funktionale Szenarien nicht in maschinenlesbarer Form vorliegen, ist nicht möglich, sie ohne Weiteres automatisiert in logische Szenarien zu wandeln. Hingegen es möglich, aus logischen Szenarien konkrete Szenarien abzuleiten, da hierfür lediglich ein Parameterset ausgewählt werden muss.

Szenariobeschreibungssprachen Um konkrete Szenarien auf Simulationsplattformen ausführen zu können, sind maschinenlesbare Szenariobeschreibungssprachen unerlässlich. Neben proprietären Beschreibungssprachen, welche von den entsprechenden Simulationstoolherstellern bereitgestellt werden,

gibt es bereits seit einigen Jahren öffentliche Standards, die vom ASAM e.V. verwaltet werden.

Der Standard **OpenDRIVE** basiert auf dem XML (Extensible Markup Language)-Format und wurde in einer Kooperation der VIRES Simulationstechnologie GmbH und der Daimler AG entwickelt [197] und in seiner aktuellsten Fassung 1.7 im Jahr 2021 vom ASAM e.V. veröffentlicht [198]. Er beschreibt ein Straßennetzwerk, was somit den Ebenen 1, 2 und 3 des 6-Ebenen-Modells entspricht. Für eine Straße können Attribute wie Typ, Geometrie und Fahrstreifen spezifiziert werden. Ebenfalls lassen sich Verkehrszeichen und Kreuzungen gesondert definieren. [198]

In Ergänzung zu OpenDRIVE stellt der Standard **OpenSCENARIO** die Beschreibung der dynamischen Elemente (Ebene 4 des 6-Ebenen-Modells) dar, welche auch als sogenannte Aktoren bezeichnet werden. Zu unterscheiden sind hierbei die jeweils im Jahr 2022 veröffentlichten Varianten OpenSCENARIO 1.2 [199] und 2.0 [200]. Während OpenSCENARIO 1.2 analog OpenDRIVE auf dem XML-Format basiert, stellt OpenSCENARIO 2.0 eine eigene Programmiersprache dar. Eine Datei der Variante 1.2 muss mit einer OpenDRIVE-Datei verknüpft werden, bei Variante 2.0 ist dies nur optional. Die Beschreibung des Szenarios in OpenSCENARIO 1.2 wird über eine hierarchische Struktur definiert, welche auf oberster Ebene das sogenannte Storyboard beinhaltet und sich weiter in Story, Act, ManeuverGroup, Maneuver sowie Event untergliedert. Ein Event ist dabei die Kombination aus einer Aktion eines Aktors und einem dazugehörigen Trigger. [199] OpenSCENARIO 2.0 nutzt ebenfalls Aktoren, lässt aber dabei sowohl eine serielle als auch parallele Ausführung von Aktionen zu. Ein weiterer Unterschied ist, dass OpenSCENARIO 2.0 abstraktere maschinenlesbare Szenariobeschreibungen bietet, sodass Parameterwerte nicht explizit spezifiziert werden müssen. [200]

3.1.4.3 Szenariodatenbank

Um das szenariobasierte Testen systematisch durchführen zu können, ist eine Szenariodatenbank notwendig. Diese basiert typischerweise auf logischen Szenarien, da sich diese automatisiert in den Absicherungsprozess einbinden lassen. [108] Für die Extraktion bzw. Generierung der logischen Szenarien kommen die folgenden Methoden in Betracht:

Expertenwissen Die Ableitung von Szenarien aus Expertenwissen ist meist der schnellste und direkteste Weg, um Szenarien zu definieren. Eine Möglichkeit ist es, auf Basis des Wissens über das SUT Szenarien abzuleiten, die von hoher Bedeutung sind, beispielsweise aufgrund ihrer Kritikalität. [201] Dies entspricht dem Vorgehen bei einem White-Box-Test. Alternativ kann das Wissen aus bestehenden Testprozessen und Standards genutzt werden, indem beispielsweise bestehende Testfälle abstrahiert werden [108].

Synthetische/systematische Generierung Ist bereits eine strukturierte Beschreibung von Szenarien vorhanden, so können aus den bestehenden Szenarien neue Szenarien abgeleitet werden, indem Elemente variiert werden. Dies ist beispielsweise mithilfe einer Ontologie möglich [108], welche es außerdem ermöglicht, automatisiert logische Szenarien abzuleiten [202]. Auch das 6-Ebenen-Modell erlaubt die automatisierte Szenariogenerierung [203], genauso wie der Ansatz der Basisszenarien [188] [108].

Felddatenanalyse Da im Realverkehr auch Szenarien auftreten, die sehr selten sind und damit möglicherweise mit wissensbasierten bzw. systematischen Methoden nicht ermittelbar sind, ist die Analyse von Felddaten eine bedeutende Quelle für logische Szenarien. Es lassen sich dabei zwei Anwendungsfälle unterscheiden:

1. Identifikation unbekannter logischer Szenarien, die bislang nicht in der Szenariodatenbank abgebildet sind [21]. Dies können auch Szenarien sein, die erst auftreten, wenn ein SAE Level 4 System zum Einsatz kommt [37]. Ebenso ist es möglich, bislang unbekannte Sonderfälle als Anomalien in den Felddaten zu identifizieren [204].

2. Ermittlung von Parameterverteilungen für bereits bekannte logische Szenarien [201] [108]. Hierfür ist zunächst eine vorgelagerte Szenarioklassifikation erforderlich. Diese kann entweder regelbasiert [205] [206] [191] oder unter Anwendung des maschinellen Lernens [207] [208] [178] [209] durchgeführt werden. Sind die Szenarien a-priori nicht bekannt, kommen Clustering-Verfahren [210] [211] infrage.

Unfalldatenanalyse Da insbesondere kritische Szenarien von Interesse sind, bietet sich die Analyse von Unfalldatenbanken zur Rekonstruktion von Szenarien an [201]. Hierbei ist jedoch zu beachten, dass nicht zwingend alle zur Rekonstruktion erforderlichen Informationen in den Datenbanken enthalten sind [212].

Simulatordatenanalyse Synthetische Daten aus Verkehrssimulationen bieten den Vorteil, dass sie schnell und kostengünstig generierbar sind und gleichzeitig Ground-Truth-Daten darstellen. Quelle kann hierbei sowohl eine reine Verkehrssimulation als auch ein Fahrsimulator mit realem menschlichen Fahrer sein. [201]

3.1.4.4 PEGASUS-Prozess

Das Projekt PEGASUS (Projekt zur Etablierung von generell akzeptierten Gütekriterien, Werkzeugen und Methoden sowie Szenarien und Situationen zur Freigabe hochautomatisierter Fahrfunktionen) war eines der bedeutendsten Forschungsprojekte im Bereich des szenariobasierten Testens und hatte einen Projektzeitraum von 2016 bis 2019. Kern des Projekts war die sogenannte PEGASUS-Methode (siehe Abb. 3.2), welche einen möglichen szenariobasierten Absicherungsprozess für ein SAE Level 3 System beschreibt. [108]

Die PEGASUS-Methode basiert auf dem Input von Wissen und Daten, welche kombiniert verarbeitet werden, sodass daraus eine Datenbank logischer Szenarien entsteht. Aus dem Wissen wird in einem Parallelpfad ein Anforderungskatalog erstellt, welcher zusammen mit den logischen Szenarien den Raum der Testfälle definiert. Für deren Durchführung sind die Testumgebungen Simulation, Prüfgelände und Realverkehr vorgesehen, sodass hier eine Testfallzuordnung stattfindet. Nach der Ausführung der Testfälle werden deren Ergebnisse evaluiert und eine Risikobewertung durchgeführt. Die finale Sicherheitsargumentation basiert auf der Risikobewertung für das System. Der Prozess ist iterativ, sodass Informationen der jeweiligen Schritte als Input des Gesamtprozesses zurückfließen. [108]

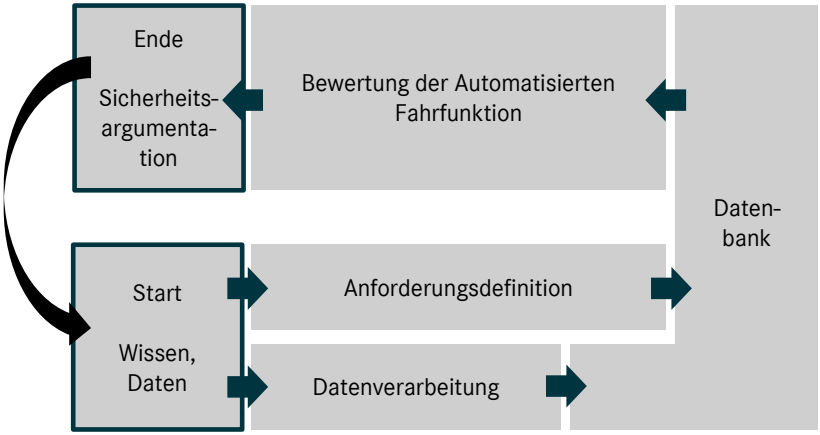

Abbildung 3.2: PEGASUS-Methode [108]

3.1.4.5 Definition Testfall

Existiert eine Datenbank logischer Szenarien, so besteht der nächste Schritt darin, Testfälle zu generieren und den vorhandenen Testumgebungen zuzuordnen. Die klassische Testfalldefinition aus 2.4.1 wird im Kontext des szenariobasierten Testens mit Szenarien als formale Beschreibung der Testeingänge ergänzt (siehe Abb. 3.3) [108] [25]:

- **Logischer Testfall**: Kombination aus logischem Szenario, Bewertungskriterien (KPIs, siehe 3.2.1) und Anforderungen an die Testfalldurchführung (allgemeine Rahmenbedingungen, Genauigkeit, Dauer).
- **Konkreter Testfall**: logischer Testfall mit entsprechendem Parametersatz bzw. konkretem Szenario.

3.1.4.6 Sampling-Strategien für Szenarien

Die Ableitung konkreter Szenarien aus einem logischen Szenario für die Generierung konkreter Testfälle in unterschiedliche Testumgebungen stellt einen zentralen Aspekt des szenariobasierten Testens dar [108] (siehe 3.1.4). Her-

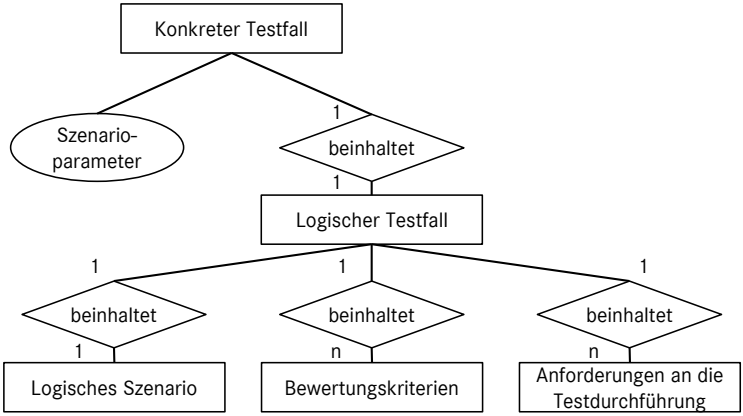

Abbildung 3.3: Übersicht Testfalldefinition nach Schuldt [25]

ausfordernd ist der Umgang mit großen Parameterräumen, bei denen es aus Gründen der Wirtschaftlichkeit bzw. Umsetzbarkeit allgemein selbst bei diskreten Parameterräumen nicht möglich ist, alle denkbaren Kombinationen von Parameterausprägungen (auch Sample genannt) zu testen [213]. Es stellt sich daher die Frage nach einem Sampling-Algorithmus für konkreten Szenarien, sodass weiterhin eine ausreichende Testabdeckung erreicht wird [214]. Existierende Ansätze lassen sich in das deterministische und stochastische Sampling einteilen [24].

Stochastisches Sampling Das stochastische Sampling nutzt Zufallsverteilungen, um Samples zu generieren. Ziel ist es hier beispielsweise, im Rahmen einer Zuverlässigkeitsanalyse Parameterbereiche hoher Kritikalität (siehe 3.2.2) zu identifizieren. Folgende Ansätze werden in der Forschung diskutiert:

Beim klassischen Monte-Carlo-Ansatz wird eine statistische Verteilung, welche beispielsweise auf der realen Parameterverteilung basiert, als Grundlage des Samplings gewählt. Im Vergleich zu anderen Verfahren ist dieser Ansatz jedoch sehr rechenintensiv und damit ineffizient. [215] [216]

Das sogenannte Adaptive Importance Sampling (AIS) passt die Parameterverteilungen aus der Realität [217] basierend auf einer Cross-Entropy-Methode

so an, dass bevorzugt Szenarien mit hoher Kritikalität generiert werden [216]. Dies führt bei Parametern mit diskreten Verteilungen zu Problemen [215].

Ein Importance Sampling-Ansatz kann mit weiteren Methoden kombiniert werden. So ist es möglich, die Beschleunigung des vorausfahrenden Fahrzeugs anzupassen, um so bewusst mehr kritische Szenarien zu simulieren. Aus den daraus generierten Daten kann wiederum eine Aussage über die Kritikalität des Systems in Szenarien mit den real vorkommenden Parametern abgeleitet werden. [218]

Beim sogenannten Importance Sampling using Design Points (ISPUD) werden zunächst Failure Points, also Punkte erwarteter hoher Kritikalität, mithilfe eines lokalen Optimierungsalgorithmus für ein Metamodell ermittelt. Anschließend werden um diese herum Samples generiert. [217]

Weitere Ansätze basieren auf Markov-Chain Monte-Carlo-Methoden oder Kriging-Modellen. [24]

Deterministisches Sampling Beim deterministischen Sampling sind die generierten konkreten Szenarien in jedem Durchlauf des Algorithmus gleich. Hierfür wird im Falle kontinuierlich verteilter Parameter eine Diskretisierung vorgenommen. In der Forschung werden mehrere Methoden diskutiert:

Bei der t-weisen Abdeckung kommt jede mögliche Kombination an Parameterausprägungen von jeweils t Parametern in mindestens einem Sample vor. Dies entspricht dem kombinatorischen Testen (2.4.2). [213] Gegeben sei ein logisches Szenario mit n_{par} Parametern, die jeweils zwischen 1 und k_{par} mögliche Ausprägungen besitzen. Es werden folgende Fälle unterschieden [104]:

- $t = 1$: Jede Parameterausprägung eines einzelnen Parameters kommt mindestens einmal im Datensatz vor. Es werden k_{par} Testfälle generiert.
- $1 < t < n_{par}$: Jede Ausprägung von t Parametern kommt mindestens einmal im Datensatz vor. Die Anzahl der Testfälle ist abhängig von der Anzahl möglicher Ausprägungen der einzelnen Parameter und kann nicht a-priori bestimmt werden.
- $t = n_{par}$: Jede Kombination aller Parameter und somit alle möglichen Testfälle existieren im Datensatz.

Eine Abschätzung für den minimal benötigten Wert von t kann beispielsweise anhand eines Abgleichs mit einer Failure-Triggering Fault Interaction (FTFI), die die Anzahl an zu erfüllenden Bedingungen für die Auslösung eines Fehlers in Abhängigkeit vom jeweiligen System beschreibt, vorgenommen werden. [213] Die abzutestenden Parameterräume wachsen dabei exponentiell mit t. Mithilfe einer funktionalen Dekomposition, also der Aufteilung des Systems in mehrere Subsysteme, können die Parameterräume reduziert und ein kleineres FTFI gewählt werden. [213] [219]

Daneben ist es möglich, über eine Szenarioklassifikation von Realdaten direkt konkrete Szenarien abzuleiten. Ziel ist es, eine möglichst vielfältige Auswahl an Parametern abzuleiten, indem redundante konkrete Szenarien herausgefiltert werden. Eine Möglichkeit dafür ist es, die Verteilung von Parameterkombinationen über ein Histogramm darzustellen und Szenarien nur hinzuzufügen, wenn sie „Lücken“ schließen [220]. Zudem ist es beispielsweise möglich, den Neuheitsgrad eines Szenarios über Auto-Encoder zu bestimmen [221].

Weitere genannte Methoden für das deterministische Sampling kommen aus dem Bereich des Maschinellen Lernens oder stützen sich auf modellbasierte Methoden [24].

3.1.4.7 Zuordnung von Testfällen auf Testumgebungen

Eine zentrale Frage des szenariobasierten Ansatzes ist, welcher Testfall in welcher Testumgebung ausgeführt wird. Dabei kommen neben dem Realversuch diverse XiL-Umgebungen infrage [108]. Bislang spielen in der praktischen Anwendung vor allem qualitative Kriterien bei der Auswahl eine Rolle, während quantitative Aspekte nur Teil theoretischer Betrachtungen sind.

Qualitative Zuordnungsmethoden Als Teil der PEGASUS-Methode stehen bei der Auswahl der Testumgebung vor allem der Neuheitsgrad und die Kritikalität eines Szenarios im Mittelpunkt. Grundsätzlich werden dabei zunächst alle Szenarien simulativ evaluiert und eine Teilmenge dieser Szenarien erneut auf dem Prüfgelände getestet. Letzteres betrifft insbesondere Szenarien, in denen Bestehenskriterien nicht oder nur knapp erfüllt wurden, genauso wie bisher unbekannte Szenarien oder Unfallszenarien. Außerdem wird im Projekt empfohlen, Szenarien im Realtest auszuführen, wenn hohe Anforderungen an

die Genauigkeit der Testumgebung bestehen, beispielsweise bezüglich der Repräsentation des Antriebsstrangs oder der Sensoren. Konkrete Kriterien für die Genauigkeitsanforderungen oder Kriterien zur Auswahl werden jedoch nicht genannt. Letztlich werden in PEGASUS auch manuell ausgewählte Szenarien zur Zulassung bzw. Zertifizierung als Realtests vorgeschlagen. [108]

Daneben existieren systematische Ansätze zur Zuordnung von Testfällen auf XiL-Umgebungen. Dabei kann eine Einordnung nach Vorbild der in 2.5.2 eingeführten Diskretisierungsstufen einzelner XiL-Komponenten vorgenommen werden. Für die jeweilige XiL-Umgebung wird für jede Komponente die entsprechende Ausprägung „real“, „emuliert“ oder „simuliert“ identifiziert. Analog erfolgt die Definition der Anforderungen an die Diskretisierungsstufen für einen zuzuordnenden Testfall. Mithilfe eines Abgleichs von Anforderung des Testfalls und Ausprägungen der XiL-Umgebung wird eine passende Umgebung gewählt. Optional ist es möglich, eine Gütefunktion für die Zuordnung zu berechnen, welche beispielsweise auf einer gewichteten Addition von Gütefunktionalen der einzelnen Bewertungsdimensionen (z.B. Modellgüte, Kosten) basiert. [222] Dabei kann zusätzlich die erwartete Validität der jeweiligen Testumgebung berücksichtigt werden, indem die erforderlichen Betriebsbereiche der verwendeten Simulationsmodelle evaluiert werden [26].

Quantitative Zuordnungsmethoden Eine Zuordnung von Testfällen, welche vollständig auf quantitativen Aspekten der jeweiligen Testumgebungen basiert, existiert bislang nur zu Teilen in Form von theoretischen Ansätzen. Ziel eines von Böde et al. [223] vorgestellten Ansatzes ist es, einen optimalen Tradeoff zwischen Realtests und simulativen Tests hinsichtlich der Gesamtkosten zu erreichen. Kernaspekt ist dabei die Konfidenz der jeweiligen Simulationsumgebung (siehe 3.3.1), welche in diesem Fall die Wahrscheinlichkeit darstellt, dass die Abweichung der Simulation zur Realität unterhalb einer definierten Schranke liegt. Daneben werden Kosten und Sicherheit eines Testfalls ebenfalls quantitativ modelliert. Letztlich ergibt sich ein Optimierungsproblem für die Gesamtkosten mit zwei Nebenbedingungen für das Gesamtkonfidenzlevel und das Gesamtsicherheitslevel. Die Bewertung der Konfidenz und Sicherheit kann auch hinsichtlich einzelner Anforderungen vorgenommen werden. Voraussetzung zur Anwendung dieser Methode ist es, die Konfidenz der verwendeten Simulationskomponenten zu kennen, woraus eine Konfidenz der gesamten Testumgebung ermittelt werden kann. [223]

3.1.5 Diskussion

Während die Ansätze aus 3.1.1 bis 3.1.3 sehr gut für spezifische Teilaspekte der Absicherung geeignet sind, bietet das szenariobasierte Testen ein breiteres Einsatzfeld in verschiedenen Entwicklungsphasen (siehe Abschnitt 4.2).

Das Vorgehen nach Wachenfeld basierend auf **Feldtests** [105] (siehe 3.1.1) eignet sich für den Fall, dass bereits eine ausreichende Datenbasis aus Feldtests verfügbar ist. Die Bewertung kann auch im Kontext einer übergeordneten Sicherheitsargumentation für die Fahrautomatisierung auf alle im Markt verfügbaren ADS vorgenommen werden. Wie von Wachenfeld selbst vorgeschlagen, ist das erst nach der Markteinführung eines ADS realisierbar. Dies ist insbesondere bei der ersten Einführung eines ADS kritisch, da hier keine Referenzdaten von anderen ADS verfügbar sind. Weiterer Kritikpunkt an der Methode ist, dass bei einer Anpassung des SUT (z.B. Software) bereits vorhandene Daten unter Umständen nicht mehr nutzbar sind, da sie mit einem älteren Stand des Systems aufgezeichnet wurden. Die Anwendung der **Extremwerttheorie** kann zwar den Bedarf von Realdaten reduzieren, die genannten Herausforderungen bleiben dennoch bestehen.

Die **Resimulations**-Ansätze (siehe 3.1.2) erlauben sehr effiziente und reproduzierbare Tests, da Daten nur einmal aufgezeichnet werden müssen und anschließend wiederverwendet werden können. Die Möglichkeit der nachträglichen Manipulation erhöht die Flexibilität potentieller Anwendungsfälle weiter. Daneben stellt die Aufzeichnung des Fahrverhaltens des menschlichen Fahrers (sofern vorhanden) eine wertvolle Referenz für die Bewertung des SUT dar (z.B. bei VAAFO [180] und Shadow Mode [17]). Bei variierenden Fahrern sind sogar Rückschlüsse auf das Verhalten einer Fahrerpopulation möglich. Größter Nachteil bei der Anwendung zum Test von Systemen ab SAE Level 1 ist, dass es sich bei der Resimulation um keinen geschlossenen Regelkreis handelt. Dies kann in eingeschränktem Maß durch Ansätze wie Reactive-Replay [182] kompensiert werden, bleibt jedoch auch hier auf Szenarien ähnlich dem aufgezeichneten beschränkt. Somit ist die Resimulation nur für (Sub-)Systeme nutzbar, die keinen kontinuierlichen Eingriff erfordern. Dies können informierende und warnende Funktionen sein oder eingreifende Notfallfunktionen. Letztere sind bis zum Eingriff bewertbar. Auch Teile des Bereichs „Sense“ (z.B. Sensorfusion) sind über die Resimulation testbar. Je nach dem, an welcher Stelle im Gesamtsystem Daten aufgezeichnet werden (Sensorrohdaten

vs. Busdaten) können diese bei Anpassungen vorgelagerter Sense-Subsysteme (z.B. Sensor-Hardware) ggf. nicht mehr wiederverwendet werden. Die Anpassung von Sensor-Software kann jedoch durch eine Resimulation von Rohdaten kompensiert werden.

Die **formale Verifikation** aus [183] ist ein denkbarer Ansatz für die Verifikation des Bereichs „Plan“. Allerdings bleibt die praktische Anwendbarkeit bislang offen, sodass der Ansatz nicht final bewertbar ist. Die separate Betrachtung des Bereichs „Sense“ über eine **statistische Absicherung** entspricht im Grundprinzip bereits dem heutigen Vorgehen bei der Validierung von Umweltsensoren.

Eine Vielzahl möglicher Anwendungsfelder bietet das **szenariobasierte Testen**. Es dient im Kontext dieser Dissertation als Referenz und wird vertieft betrachtet. Im Gegensatz zum bestehenden Vorgehen zur Testfallbeschreibung und -generierung über Testfallkataloge bietet es die folgenden Vorteile:

- **Testfallreduktion**: Indem ähnliche Szenarien in Äquivalenzklassen gruppiert werden, kann die Gesamtanzahl zu testender Szenarien reduziert werden. [25] [20]
- **Synthetische Szenarien**: Auf Basis der systematischen Beschreibung von Szenarien ist es möglich, bislang unbekannte logische [191] [203] [193] oder konkrete [213] [215] Szenarien zu generieren. Dies erhöht die Testabdeckung.
- **Datenextraktion aus Realdaten**: Neben der synthetischen Generierung ist es ebenfalls möglich, Szenarien und deren Parameterverteilungen aus Realdaten zu extrahieren. Dies stellt eine effiziente Repräsentation realer Szenarien dar. [178] [204] Voraussetzung hierfür ist eine ausreichend hohe Qualität und Quantität der Realdaten.
- **Durchgängigkeit**: Aus Szenarien lassen sich Testfälle für unterschiedliche Testebenen und Testumgebungen ableiten, was die Vergleichbarkeit von Tests erhöht. [108] [20]
- **Automatisierung**: Die systematische Szenariobeschreibung erlaubt es, automatisierte Algorithmen zur Testfallgenerierung anzuwenden und somit die Anzahl der Testfälle schnell und flexibel zu skalieren. [215] [24]

Den genannten Vorteilen stehen die folgenden Herausforderungen entgegen:

- Sollen Szenarioparameter und deren Abhängigkeiten aus Realdaten abgeleitet werden, so erfordert dies einen ausreichend großen und repräsentativen Datensatz. Ebenso muss die Datenqualität eine Klassifikation und Parameterextraktion erlauben. [224] [225]
- Werden nichtdeterministische Samplingalgorithmen (siehe 3.1.4.6) genutzt [24], um aus logischen Szenarien konkrete Szenarien zu erzeugen, so ist eine Reproduzierbarkeit nicht mehr gegeben und Back-to-Back-Tests bzw. Regressionstests sind ggf. nicht mehr direkt durchführbar.
- Die Berücksichtigung von Einflüssen auf das Systemverhalten ist auf die in Szenariobeschreibungen (siehe 3.1.4.2) enthaltenen Informationen beschränkt. [226]
- Je nach Anzahl und möglicher Diskretisierung der Szenarioparameter lassen sich nicht alle Parametervariationen testen, da auch in der Simulation Rechenkapazitäten begrenzt sind. Es ist daher notwendig, mithilfe von Sampling-Algorithmen (siehe 3.1.4.6) ein Subset an möglichen Parameterausprägungen zum Test auszuwählen. [214] [226]
- Einige sehr seltene logische Szenarien bleiben bis zu ihrem Auftreten im Feld unbekannt. Es ist möglich, die Auftretenswahrscheinlichkeit eines solchen Szenarios abzuschätzen, jedoch ist dessen Auftreten im Feld nicht auszuschließen. [21]

In der bisherigen wissenschaftlichen Diskussion wurde der Ansatz hauptsächlich für Systeme ab SAE Level 3 konzipiert bzw. angewendet. Nur in wenigen Fällen handelte es sich dabei um Serien-Entwicklungsprojekte [215] [108]. Es besteht somit Bedarf, den Einsatz des szenariobasierten Testens im Kontext Fahrerassistenzsysteme und die Integration in bestehende Entwicklungsprozesse zu evaluieren. Kapitel 4 beleuchtet hierfür drei Einsatzmöglichkeiten.

Die Eignung der genannten **Sampling-Algorithmen** ist stark abhängig vom Ziel der Testfallgenerierung. Deterministische Algorithmen sind insbesondere dann einsetzbar, wenn eine gleichmäßige Testabdeckung angestrebt wird. Für Back-to-back-Tests oder Regressionstests sind sie das Mittel der Wahl, da nur so eine Reproduzierbarkeit der Testfallgenerierung sichergestellt werden kann. Stochastische Algorithmen hingegen sind dann gut geeignet, wenn ge-

zielt kritische Szenarien generiert werden und so die Funktionsgrenze des SUT identifiziert wird. Insgesamt existiert bereits eine größere Auswahl geeigneter Sampling-Algorithmen, sodass in dieser Dissertation auf bestehende Ansätze zurückgegriffen wird. Unabhängig von der Wahl des Sampling-Algorithmus muss sichergestellt werden, dass die generierten Samples gültige Parameterkombinationen aufweisen. Dies kann sich sowohl auf physikalische Grenzen (z.B. maximale Geschwindigkeit) als auch die Einhaltung der vorgesehenen Abfolge von Szenen beziehen. Neben der Berücksichtigung über Abhängigkeiten vor dem Sampling [151] ist eine Überprüfung des Szenarios nach der Durchführung eines Testfalls möglich [227].

Für die **Zuordnung von Testfällen auf Testumgebungen** werden aktuell hauptsächlich qualitative Kriterien betrachtet. Einige Ansätze beinhalten dabei eine Systematik, in der Praxis spielen jedoch weiterhin Expertenwissen und Erfahrungen die größte Rolle. Ein quantitativer Ansatz unter Berücksichtigung der Kosten und der Konfidenz der Testumgebung wurde bislang nur theoretisch evaluiert [223]. Somit besteht Bedarf für einen praktisch anwendbaren Ansatz, der die Kriterien Effizienz und Konfidenz des jeweiligen Szenarios mit berücksichtigt. In Kapitel 4 wird ein möglicher Prozessentwurf vorgestellt.

3.2 KPIs für die Bewertung der Fahrautomatisierung

3.2.1 Definitionen

Zur Bewertung von Fahrautomatisierung kommen verschiedene Schlüsselkennzahlen (Key Performance Indicators, KPIs) zum Einsatz. Um eine eindeutige Abgrenzung sicherzustellen, werden diese hier kategorisiert.

Eine **Systemantwort** (System Response Quantity, SRQ) (siehe [228]) ist eine aus einem XiL-Test direkt messbare Größe. Beispiele sind Ego/Objekt-Trajektorien, Signale des SUT sowie falls messbar SUT-interne Signale. Eine SRQ ist grundsätzlich eine zeitabhängige Größe: $SRQ(t)$.

Metriken werden aus SRQs berechnet und betrachten einen speziellen Aspekt des SUT-Verhaltens (z.B. Kritikalitätsmetriken, siehe 3.2.2).

Eine **zeitreihenbasierte Metrik** (Time-domain Metric, TDM) [229] ist eine aus SRQs abgeleitete Metrik, die eine zeitliche Abhängigkeit besitzt. Beispiele sind Kritikalitätsmetriken (z.B. TTC) oder relative Position zu Objekten.

$$TDM(t) = f_{TDM}(SRQ_1(t), ..., SRQ_n(t)) \tag{3.2}$$

Die Funktion $f_o(i_1, ..., i_n)$ stellt in dieser Dissertation, sofern nicht anders spezifiziert, eine allgemeine Funktion mit Inputs i_1 bis i_n und Output o dar.

Eine **featurebasierte Metrik** (Feature-based Metric, FBM) [229] ist eine aus SRQs und TDMs abgeleitete Metrik, die keine zeitliche Abhängigkeit besitzt und eine skalare Größe darstellt. Dies kann beispielsweise die minimale TTC oder die minimale Relativdistanz zu Objekten sein.

$$FBM = f_{FBM}(SRQ_1(t), ..., SRQ_n(t), TDM_1(t), ..., TDM_m(t)) \tag{3.3}$$

Klassische Testfallkataloge beinhalten Kriterien für das Bestehen bzw. Nichtbestehen eines Testfalls. Daran angelehnt stellen **Pass/Fail-Kriterien** (Pass/Fail-Criterion, PFC) eine binäre Größe dar, die eine qualitative Aussage über den Ausgang des Tests beinhaltet. Beispiele sind das Auftreten einer Kollision oder der Eingriff eines Fahrerassistenzsystems.

$$\begin{aligned} PFC = f_{PFC}(SRQ_1(t), ..., SRQ_n(t), TDM_1(t), ..., TDM_m(t), \\ FBM_1, ..., FBM_p) \end{aligned} \tag{3.4}$$

Die Zusammenfassung von PFCs als Vektor wird als **Testergebnis** T definiert:

$$T = [PFC_1, ..., PFC_q] \tag{3.5}$$

Es gilt dabei, dass unterschiedliche Testergebnisse sich hinsichtlich ihrer PFCs unterscheiden müssen:

$$T_i \neq T_r \tag{3.6}$$

Zudem soll die Gesamtheit aller aus dem Test eines Szenarios messbaren und ableitbaren KPIs wie folgt bezeichnet werden:

$$\begin{aligned} KPI = [&SRQ_1(t), ..., SRQ_n(t), TDM_1(t), ..., TDM_m(t), \\ &FBM_1, ..., FBM_p, PFC_1, ..., PFC_q] \end{aligned} \tag{3.7}$$

3.2.2 Kritikalitätsmetriken zur Bewertung der Sicherheit

Im Rahmen der Absicherung jeglicher Fahrautomatisierung hat die Erfüllung von Sicherheitszielen oberste Priorität, auch wenn weitere Faktoren wie Komfort [230] oder Effizienz [8] eine Relevanz besitzen. Dies kann zum einen mit der benötigten gesellschaftlichen Akzeptanz bei der Einführung hochautomatisierter Systeme begründet werden [37]. Einschlägige Forschungsprojekte wie PEGASUS [108], SET Level [187] oder VVM [186] beschäftigen sich schwerpunktmäßig mit dem Aspekt Sicherheit. Zum anderen war bereits die Entwicklung von Fahrerassistenzsystemen aus Gesichtspunkten der Sicherheit motiviert [36].

Bei der makroskopischen Bewertung der Sicherheit werden typischerweise Unfallstatistiken herangezogen. Analog kann für die mikroskopische Bewertung einzelner Szenarien betrachtet werden, ob es zu einer Kollision kam bzw. wie nah diese war. Beim Vorliegen einer Kollision ist die Kollisionsgeschwindigkeit zur Quantifizierung der erwarteten Unfallschwere eine geeignete Metrik. Da bei der Fahrautomatisierung die Vermeidung von Kollisionen im Vordergrund steht, sind Metriken erforderlich, die die Kritikalität von Szenarien bewerten. Die Metriken unterscheiden sich hinsichtlich ihrer Aussagekraft für bestimmte Unfallkonstellationen. [231]

Sogenannte A-Posteriori-Metriken betrachten für die Bewertung die vollständige Trajektorie und können dementsprechend erst nach deren vollständiger Aufzeichnung bestimmt werden. Trajektorienprädiktionsmetriken hingegen können meist direkt für eine Szene berechnet werden, indem entweder eine deterministische oder probabilistische Prädiktion der Trajektorie erfolgt. [232]

3.2.2.1 Trajektorienprädiktionsmetriken

Metriken basierend auf einer deterministischen Trajektorienprädiktion können in Abhängigkeit des potentiellen Kollisionsmechanismus eingeteilt werden. Folgende Metriken beschreiben den Fall einer longitudinalen Kollision:

- **Time-to-Collision** (TTC): Zeit bis zur Kollision unter der Annahme, dass die beteiligten Fahrzeuge (Ego-Fahrzeug und Objekt q) eine konstante Geschwindigkeit beibehalten. [233]

$$TTC_q = \frac{dx_q}{dv_{q,x}} \tag{3.8}$$

- **Time-to-React** (TTR): Zeitreserve, die bleibt, um eine Kollision zu vermeiden, wenn danach unmittelbar eine Vollbremsung (Time-to-Brake, TTB) oder ein Kickdown (Time-to-Kickdown, TTK) eingeleitet wird. [234] [235]
- **Worst-Time-to-Collision** (WTTC): Zeit bis zu einer Kollision, wenn sich die Fahrzeuge so verhalten, dass es schnellstmöglich zur Kollision kommt, unter Beachtung der kinematischen Startbedingungen und Limitierungen. [236]
- **Required Deceleration** (a_{req}): Die konstante negative Beschleunigung, die erforderlich ist, um eine Kollision zu vermeiden. [237]
- **Brake Threat Number** (BTN): Beschleunigung, die erforderlich ist, um eine Kollision zu vermeiden im Verhältnis zur maximal umsetzbaren Beschleunigung. [238]

$$BTN = \frac{a_{req}}{a_{max}} \tag{3.9}$$

Für laterale Ausweichmanöver existieren unter anderem die Metriken Time-to-Steer (TTS) [235] und Time-to-Closest-Encounter (TTCE) [239]. Beispiele für Metriken beim Verlassen der Fahrbahn sind Time-to-Lane-Crossing (TTLC) und Time-to-Edge-Crossing (TTEC) [240].

Darüber hinaus existieren Erweiterungen auf probabilistischer Basis durch Berücksichtigung der Beschleunigung des vorausfahrenden Fahrzeugs [232] [241] oder mithilfe einer Monte-Carlo-Simulation [242] [243]. Ein weiterer Ansatz ist die Bewertung der Schwierigkeit der Fahraufgabe mithilfe einer modellprädiktiven Regelung [244].

3.2.2.2 A-Posteriori-Metriken

Auch bei einer nachträglichen Auswertung der Trajektorie können bereits eingeführte Metriken wie TTC verwendet werden, jedoch bietet sich hier die Möglichkeit, diese mit zusätzlichen Bedingungen zu verknüpfen. Ebenfalls können weitere Messgrößen wie die tatsächliche Beschleunigung oder die Gierrate berücksichtigt werden. Auch die Fahrerreaktion kann a-posteriori analysiert werden. [206]

Daneben kommt für kreuzende Fahrzeuge Metrik Post-Encroachment-Time (PET) infrage, die die Zeit beschreibt zwischen dem Moment, in dem das erste Fahrzeug den potentiellen Kollisionsort verlässt und dem Moment, in dem das zweite Fahrzeug den potentiellen Kollisionsort erreicht. [245]. Bei lateralen Ausweichmanövern kann die Distance-of-Closest-Encounter (DCE) [239] angewendet werden.

3.2.3 Szenariodistanzmaße

Um aufgezeichnete Daten zweier konkreter Szenarien auf quantitativer Basis vergleichen zu können, sind Distanz- oder Ähnlichkeitsmaße notwendig. Aus einer hohen Distanz folgt einer geringe Ähnlichkeit und umgekehrt [246]. Zur Vereinfachung wird im weiteren der Begriff Distanz gegenüber Ähnlichkeit vorgezogen.

Es gelten die folgenden Bedingungen für ein Szenariodistanzmaß zwischen den Szenarien S_n und S_m [314]:

$$d_{n,m} = f_d(S_n, S_m) \tag{3.10}$$

$$d_{n,n} = 0 \tag{3.11}$$

$$d_{n,m} \geq 0 \tag{3.12}$$

$$d_{n,m} = d_{m,n} \tag{3.13}$$

Szenariodistanzmaße werden bereits im Kontext des szenariobasierten Testens verwendet, jedoch nur implizit als solche bezeichnet. Mögliche Anwendungsfälle umfassen unter anderem:

- Identifikation von zuvor spezifizierten Szenarien [204] [247]
- Clustering ähnlicher unbekannter Szenarien [248] [249]
- Detektion seltener oder unbekannter Szenarien [221] [250] [251]
- Plausibilisierung bzw. Validierung von XiL-Umgebungen [29] [30] [312]

Aktuell diskutierte Ansätze zur Berechnung von Szenariodistanzmaßen fokussieren sich auf Trajektoriendaten des Ego-Fahrzeugs und anderer Verkehrsteilnehmer, was der Ebene 4 des 6-Ebenen-Modells entspricht (siehe 3.1.4.2). Grundsätzlich ist es möglich, darüber hinaus weitere Daten zu verwenden. Dies wird im Rahmen dieser Dissertation jedoch nicht weiter betrachtet. Die Berechnung lässt sich nach [314] in 3 unterschiedliche Ansätze einteilen:

- **Zeitreihenbasiert**: Hier werden die Informationen der aufgezeichneten Zeitreihen direkt verwendet, um eine Szenariodistanz zu berechnen.
- **Manöverbasiert**: Aus den Trajektoriendaten werden Manöver extrahiert, die anschließend in ein Szenariodistanzmaß überführt werden.
- **KPI-basiert**: Es werden KPIs berechnet (siehe 3.2.1), die für die Berechnung der Szenariodistanz verwendet werden.

Im Rahmen dieser Dissertation stehen die zeitreihenbasierten Szenariodistanzmaße im Fokus. Für weitere Details und Berechnungsmethoden von Szenariodistanzmaßen wird auf [314] verwiesen.

3.2.4 Diskussion

Die in 3.2.2 vorgestellten Kritikalitätsmetriken bieten eine Möglichkeit, die Kritikalität eines Szenarios zu quantifizieren. Jedoch müssen dabei folgende Einschränkungen beachtet werden: Kritikalitätsmetriken sind abhängig vom jeweiligen Anwendungsfall zu wählen, hier dem logischen Szenario. Da sie nur eine Teilmenge möglicher Kollisionen abdecken, ist für eine umfassende Bewertung der Kritikalität eines Szenarios ggf. eine Aggregation bzw. Kombination mehrerer Metriken notwendig. Trajektorienprädiktionsmetriken basieren auf Annahmen bezüglich der räumlichen Ausdehnung sowie Bewegung des Ego-Fahrzeugs und anderer Verkehrsteilnehmer, sodass hieraus abgeleitete Aussagen die Unsicherheit der Prädiktion beinhalten. Letzteres wird bei einer probabilistischen Prädiktion berücksichtigt. Da bereits eine Vielzahl an Metriken existiert, wird eine Auswahl dieser im Kontext dieser Dissertation verwendet.

Szenariodistanzmaße stellen ein bislang noch kaum explizit beachtetes Thema in der wissenschaftlichen Diskussion dar. Dennoch werden sie implizit bereits in mehreren Anwendungsfällen verwendet. Synergien zwischen Anwendungsfällen lassen sich insbesondere dort generieren, wo ähnliche Fragestellungen hinsichtlich der Ähnlichkeit von Szenarien bestehen. Da bislang nur eine eingeschränkte Auswahl publizierter Szenariodistanzmaße existiert, werden im Rahmen dieser Dissertation neue Ansätze entwickelt.

3.3 Glaubwürdigkeit des XiL-Tests für die Fahrautomatisierung

3.3.1 Definitionen

Für die Glaubwürdigkeitsbewertung von XiL-Tests ist es vorab notwendig, einige Begrifflichkeiten zu definieren und voneinander abzugrenzen, da diese in der wissenschaftlichen Diskussion nicht eindeutig verwendet werden.

Die Begriffe Simulation und Computersimulation sind wie folgt definiert:

> **Definition Simulation:** *Eine Simulation ist der Prozess des Lösens von Gleichungen eines dynamischen Modells, wodurch ein anderer Prozess imitiert wird.* [252] [253]

> **Definition Computersimulation:** *Eine Computersimulation ist das Ausführen einer Simulation auf einem physischen Computer.* [252] [253]

Da im Kontext dieser Dissertation jede Simulation auf einem physischen Computer ausgeführt wird, soll im folgenden der Begriff „Simulation“ gleichbedeutend mit „Computersimulation“ verwendet werden. Zudem werden die Begriffe konzeptionelles Modell und Simulationsmodell (auch als „Computerized Model“ bezeichnet [254]) eingeführt:

> **Definition konzeptionelles Modell:** *„Ein konzeptionelles Modell beinhaltet die verbale Beschreibung, Gleichungen, maßgebliche Zusammenhänge oder Naturgesetze, mit dem Ziel, die Realität zu beschreiben.“* [254]

> **Definition Simulationsmodell:** *„Ein Simulationsmodell ist ein ausführbares Computerprogramm, das ein konzeptionelles Modell implementiert.“* [254]

Simulationsmodelle können hierarchisch strukturiert und aus mehreren Sub-Modellen aufgebaut sein [255]. Dies ist insbesondere bei XiL-Umgebungen der Fall, da dort Simulationsmodelle verschiedener Komponenten miteinander verbunden werden. Die folgenden Definitionen gelten sowohl für einzelne Komponenten-Simulationsmodelle als auch für eine gesamte XiL-Umgebung.

Die Konfidenz eines Simulationsmodells wird folgendermaßen definiert:

> **Definition Konfidenz:** *Konfidenz ist ein Maß dafür, wie gut aus Sicht des (potentiellen) Anwenders ein Simulationsmodell und daraus abgeleitete Informationen für Entscheidungen nutzbar sind.* [256] [257]

Es sei angemerkt, dass die hier definierte Konfidenz eines Simulationsmodells keinen direkten Zusammenhang zum in 3.1.1 verwendeten Konfidenzniveau hat. In [223] wird die Konfidenz als Wahrscheinlichkeit, dass die Abweichung des Simulationsmodells im Vergleich zur Realität unterhalb einer bestimmten Schranke liegt, definiert (siehe 3.1.4.7). Die Glaubwürdigkeit eines Simulationsmodells ist eng mit der Konfidenz verbunden:

> **Definition Glaubwürdigkeit:** *Ein Simulationsmodell ist glaubwürdig, wenn dieses die aus Sicht des (potentiellen) Nutzers erforderliche Konfidenz hat.* [257]

Aus der Glaubwürdigkeit eines Simulationsmodells folgt die Bereitschaft des Anwenders, Entscheidungen basierend auf Informationen zu treffen, die aus der Simulation generiert wurden [258]. Die Glaubwürdigkeit kann durch eine erfolgreiche Qualifizierung, Verifikation und Validierung des Simulationsmodells argumentiert werden [254] [257]. Deren Zusammenhang ist in Abb. 3.4 dargestellt.

Die Definitionen lauten wie folgt (siehe 2.2.1):

> **Definition Qualifizierung:** *„Nachweis, dass das konzeptionelle Modell geeignet ist, im vorgesehenen Anwendungsbereich eine akzeptable Übereinstimmung zur Realität erreichen.“* [254]

> **Definition Verifikation:** *„Nachweis, dass das Simulationsmodell eine korrekte Implementierung des konzeptionellen Modells ist und das konzeptionelle Modell mit ausreichender Genauigkeit repräsentiert.“* [254] [257]

> **Definition Validierung:** *„Nachweis, dass das Simulationsmodell innerhalb seines Anwendungsbereiches eine zur vorgesehenen Anwendung konsistente und ausreichende Genauigkeit besitzt.“* [254]

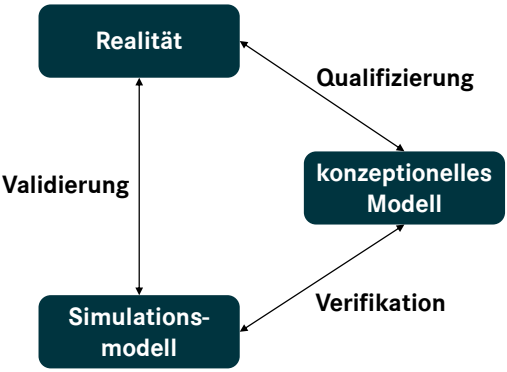

Abbildung 3.4: Zusammenhang zwischen Qualifizierung, Verifikation und Validierung eines Simulationsmodells [254]

Die Validität eines Simulationsmodells wird wie folgt definiert:

> **Definition Validität:** *„Validität ist ein Maß für die Genauigkeit des Simulationsmodells für eine vorgesehene Anwendung innerhalb seines Anwendungsbereichs.“* [254] [257]

Im Kontext der Validierung von Simulationen wird häufig vom Fehler der Simulation gesprochen. Dieser wird wie folgt definiert:

> **Definition Fehler:** *„Ein Fehler ε ist eine wahrnehmbare Ungenauigkeit während der Modellierung oder Simulation, die nicht aufgrund fehlenden Wissens auftritt.“* [259]

Ein Fehler entspricht somit einer messbaren Abweichung von einem tatsächlichen Referenzwert [255]:

$$\gamma = \gamma^{ref} + \varepsilon \tag{3.14}$$

Im konkreten Anwendungsfall stellt die Referenz einen idealen Realtest dar, in dem alle Objekte sich auf ihrer durch die Szenariobeschreibung ergebenden Trajektorie bewegen. Alle Daten des Ego-Fahrzeugs und der Objekte wer-

den dabei ohne Messfehler erfasst. Sowohl die XiL-Umgebung als auch der tatsächlich aufgezeichnete Realtest weichen hiervon ab. Die Fehler von XiL-Umgebung und Realtest sind somit wie folgt definiert:

$$\gamma^x = \gamma^{ref} + \varepsilon^x \tag{3.15}$$

$$\gamma^r = \gamma^{ref} + \varepsilon^r \tag{3.16}$$

Für den daraus resultierenden Gesamtfehler zwischen XiL-Umgebung und Realtest gilt:

$$|\gamma^x - \gamma^r| = \varepsilon^{ges} \leq |\varepsilon^x| + |\varepsilon^r| \tag{3.17}$$

Während ein Fehler eine wahrnehmbare Ungenauigkeit ist, sind die Ungenauigkeiten von Unsicherheiten nicht exakt messbar. Ein Fehler ist somit Resultat einer Unsicherheit. Hierbei werden aleatorische und epistemische Unsicherheiten unterschieden:

> **Definition aleatorische Unsicherheit:** *„Eine aleatorische Unsicherheit U_a ist eine inhärente Varianz, die mit dem betrachteten (physischen) System oder der Umgebung zusammenhängt.“* [259]

Eine aleatorische Unsicherheit kann quantifiziert werden, aber nicht eliminiert [259].

> **Definition epistemische Unsicherheit:** *„Eine epistemische Unsicherheit U_e ist die potentielle Ungenauigkeit während der Modellierung oder Simulation, die aufgrund fehlenden Wissens auftritt.“* [259]

Aleatorische Unsicherheiten können über Wahrscheinlichkeitsdichtefunktionen (Probability Density Function, PDF) oder kumulative Verteilungsfunktionen (Cumulative Distribution Function, CDF) modelliert werden [255]. Diese bieten sich ebenfalls für epistemische Unsicherheiten an [260], neben Intervallen oder P-Boxen [259].

3.3.2 Validierung von Sensormodellen

Bei der Validierung von Sensormodellen liegt der Fokus darauf, zu überprüfen, ob und in welcher Präzision Sensoreffekte und Messungenauigkeiten des realen Sensors im Simulationsmodell abgebildet sind. Grundlage bildet die Aufzeichnung standardisierter Szenarien, in denen neben den Sensordaten mithilfe von Referenzmesstechnik zusätzlich Ground-Truth-Daten aufgezeichnet werden [159] [261]. Es findet stets ein Abgleich von Realdaten mit zugehörigen simulierten Daten statt [262] [261], wofür mehrere Methoden anwendbar sind. So kann ein graphischer Vergleich bestimmter Signale, insbesondere der Distanz zum jeweiligen Objekt, vorgenommen werden [152] [23] [159]. Daneben ist es möglich, einzelne Effekte in speziell dafür abgestimmten Szenarien zu validieren, was vor allem für physikalische Sensormodelle von Bedeutung ist [162]. Für eine gesamtheitliche quantitative Validierung kommen je nach Art des Sensormodells und der zu bewertenden Verarbeitungsebene verschiedene Metriken infrage. Im folgenden einige Beispiele:

Unabhängig von der Art des Sensors bzw. des Sensormodells sind Distanzmaße wie die Hausdorff-Distanz, OSPA, GOSPA oder OSPA-MT für die Bewertung des Trackings nutzbar. [263] [264]

Physikalische Lidar-Sensormodelle können anhand eines Occupancy Grids [265] [263] oder der Wahrheitsmatrix der Objektsegmentierung [266] bewertet werden.

Die Validierung physikalischer Radar-Sensormodelle kann auf Basis des Root Mean Squares der simulierten Detektionen und der über ein Nächste-Nachbar-Verfahren assoziierten realen Detektionen durchgeführt werden. Eder bewertet zudem anhand eines Kolmogorov-Smirnov-Tests die Konvergenz von simulierter und realer Verteilungsfunktion des Rauschens bei Radardetektionen. [262]

3.3.3 Validierung von Fahrdynamikmodellen

Die Fahrdynamik stellt neben den Fahrzeugkomponenten den maßgeblichen Teil der simulativen Repräsentation des Fahrzeugs dar (siehe 2.5.2.3). Die Validierung basiert hier ähnlich wie bei Sensormodellen auf dem Abgleich von Realdaten aus Referenzszenarien bzw. -manövern mit dem simulativen

Verhalten des Modells [28]. Hierbei können sowohl stationäre wie auch transiente Manöver [267] jeweils gezielt für Längs-, Quer- und Vertikaldynamik eingefahren und analysiert werden [28].

Das Vorgehen bei der Analyse und Bewertung der Validität variiert dabei sehr stark, wobei insbesondere die folgenden Methoden genutzt werden:

In einigen Publikationen findet lediglich ein manueller/visueller Abgleich kinematischer Größen wie der Fahrzeuggeschwindigkeit [268] oder Modellcharakteristika wie z.B. dem Lenkverhalten [269] statt.

Das nichtdeterministische bzw. statistische Verhalten der Fahrdynamik in Realtests kann über Konfidenzintervalle [270] [271] oder Toleranzbänder [272] [273] in Abhängigkeit von der Zeit abgebildet werden. Als Ausgangsgrößen kommen ebenfalls kinematische Größen der Fahrdynamik infrage [270] [271]. Durch den Abgleich der Simulation mit dem Konfidenzintervall des Realtests [273] bzw. der Ableitung einer zeitreihenbasierten Metrik (siehe 3.2.1) [272] kann die Validität des Simulationsmodells argumentiert werden. Dies entspricht dem Vorgehen, das in Normen zur Validierung von Fahrdynamikmodellen festgehalten ist. [273]

Basierend auf dem Systemverhalten des Fahrdynamikmodells können zudem featurebasierte Metriken (siehe 3.2.1) definiert werden. Hierbei kommen als Ausgangsgrößen wie zuvor fahrdynamische Charakteristika [28] sowie kinematische Größen [271] [274] infrage.

Eine weitergehende Übersicht von Validierungsverfahren für Fahrdynamikmodelle findet sich in [267]. Neben dem Ego-Fahrzeug kommen Fahrdynamikmodelle auch in der Umweltsimulation zur Simulation anderer Objekte zum Einsatz.

3.3.4 Validierung sonstiger XiL-Komponenten

Für die Validierung von **Steuergeräte-Modellen** sind nur wenige Ansätze publiziert. Analog zur Validierung von Fahrdynamikmodellen ist der rein qualitative visuelle Abgleich des zeitlichen Verlaufs der Ausgangsgrößen eines Simulationsmodells mit dem zugehörigen realen Steuergerät eine Variante [275] [276]. Dies ist auch anwendbar bei einer Kopplung mehrerer Steuergeräte-

Modelle mit weiteren Fahrzeug-Komponentenmodellen [277]. Darüber hinaus ist es auch möglich, einzelne modellierte Effekte (z.B. Übertragungszeiten von Signalen) direkt aus einem realen Steuergerät zu messen und im Modell zu parametrieren [278].

Die Validierung von **Fahrzeugkomponenten** ist stark abhängig vom Ziel der Simulation und der Struktur des modellierten Fahrzeugs. So ist es möglich, den prädizierten Energieverbrauch eines Antriebsstrangmodells mit Realdaten zu vergleichen [279] [280], wenn das Modell diesen Einsatzzweck hat. Die Validierung von Modellen des Bremssystems kann hingegen durch den qualitativen visuellen Vergleich von Ausgangsgrößen (z.B. Bremsdruck oder Fahrzeuggeschwindigkeit) mit Realdaten durchgeführt werden [281] [282].

Bei **Fahrermodellen** findet statt einer Validierung eine Kalibrierung statt, bei der Parameter des jeweiligen Modells auf Basis von Realdaten optimiert werden. Dabei wird zwischen zwei Ansätzen unterschieden: Bei der lokalen Optimierung wird der Output des Fahrermodells zu jedem Zeitschritt auf Basis von Inputs aus Realdaten berechnet und mit dem Verhalten des realen Fahrers verglichen. Im Gegensatz dazu wird bei der Trajektorien-Optimierung eine gesamte Folgefahrt mithilfe des Fahrermodells simuliert und anschließend die simulierte Trajektorie mit der realen Trajektorie verglichen. [283] Die Kalibrierung von Fahrermodellen betrifft neben dem Fahrer des Ego-Fahrzeugs auch Fahrzeuge der Umweltsimulation.

3.3.5 Validierung einer gesamten XiL-Umgebung

Als Kombination verschiedener Komponenten ist die Validität einer XiL-Umgebung grundsätzlich von der Validität der einzelnen Komponenten abhängig. Unsicherheiten bzw. Fehler werden durch andere Komponenten propagiert [284]. Jedoch folgt aus der Validität aller Komponenten nicht zwangsläufig die Validität der gesamten XiL-Umgebung [285] [28]. Im Falle einer Co-Simulation sind Wechselwirkungen zwischen den Komponenten zu berücksichtigen, welche Einfluss auf die Stabilität, die Energieerhaltung und der Synchronität von Events haben können [286]. Dabei wird zwischen einer schwachen und starken Kopplung unterschieden: Komponenten unterliegen einer schwachen Kopplung, wenn deren Austausch (z.B. Informationen, Signale, Energie oder Kräfte) nur in eine Richtung erfolgt. Hingegen existiert eine starke

Kopplung, wenn dies bidirektional zwischen Komponenten stattfindet. [287] Für eine XiL-Umgebung wird aufgrund des geschlossenen Regelkreises von einer starken Kopplung ausgegangen.

Das grundsätzliche Vorgehen bei der Validierung einer XiL-Umgebung ist analog zur Validierung von Sensor- und Fahrdynamikmodellen: Für ein Referenzszenario wird das Systemverhalten der XiL-Umgebung mit dem Verhalten des realen Systems abgeglichen und Metriken abgeleitet. Dafür kommen neben Trajektoriendaten [29] [30] [31] [310] und Kritikalitätsmetriken [29] [30] auch Ausgangssignale des SUT wie beispielsweise Beschleunigungsanforderungen [29] infrage. Trajektoriendaten können sich sowohl auf das eigene Fahrzeug als auch auf andere Objekte des Szenarios beziehen. Somit sind beispielsweise Position, Geschwindigkeit [29] [30] [31] oder Gierwinkel [30] sowohl in absoluten Koordinaten wie auch relativ zum eigenen Fahrzeug von Interesse.

Für den Vergleich von XiL-Umgebung und Realtest wurden bisher verschiedene Metriken vorgestellt:

- Sollen Zeitreihen verglichen werden, so kann durch Subtraktion der Signale aus XiL und Realtest der Fehler des XiL-Tests als zeitreihenbasierte Metrik definiert werden. Anschließend ist es möglich, den mittleren Fehler und die Standardabweichung über die Zeit als featurebasierte Metrik zu berechnen. [30] [31]
- Für den Fall, dass ein identisches Szenario mehrfach im Realtest ausgeführt wird, kann die Standardabweichung der Trajektoriendaten aus XiL und Realtest zusammen zu jedem Zeitpunkt berechnet werden. Dies entspricht ebenfalls einer zeitreihenbasierten Metrik. Eine featurebasierte Metrik kann hier über die mittlere und maximale Standardabweichung abgeleitet werden. [29]
- Eine Alternative zu Trajektoriendaten ist die Repräsentation eines Szenarios als Tensor, in welchem der zeitliche Verlauf des Szenarios in Form von diskreten Zuständen enthalten ist. Durch den Abgleich der Szenariotensoren hinsichtlich deren Ähnlichkeit kann die Validität der XiL-Umgebung bewertet werden. Ein mögliches Verfahren hierfür ist beispielsweise der Mean Absolute Percentage Error. [288]

3.3.6 Diskussion

Grundsätzlich sind alle Ansätze zur Validierung eines Simulationsmodells stark auf den jeweiligen Anwendungsfall abgestimmt. Dies zeigt sich sowohl bei den Ansätzen zur Validierung von Sensormodellen als auch Fahrdynamikmodellen, welche jeweils nur eine Auswahl spezifischer Charakteristika berücksichtigen. Sensormodelle können sowohl hinsichtlich des Trackings als auch der Objektklassifikation oder der Objektsegmentierung validiert werden. In der Fahrdynamik ist die Validierung einzelner fahrdynamischer Phänomene mithilfe von Konfidenzintervallen und Toleranzbändern etabliert. Bei Steuergeräte- und Fahrzeugmodellen wird meist ein visueller qualitativer Abgleich vorgenommen, während Fahrermodelle kalibriert werden.

Jedoch kann aus der Konfidenz bzw. Validität der Sub-Simulationsmodelle nicht auf die Konfidenz des gesamten Simulationsmodells geschlossen werden. Deshalb ist zusätzlich eine separate Betrachtung der Glaubwürdigkeit einer gesamten XiL-Umgebung notwendig. Die hierzu vorgestellten Methoden basieren hauptsächlich auf dem Abgleich aufgezeichneter Zeitreihen von XiL und Realtest. Erste Ansätze berücksichtigen die Unsicherheit in XiL und Realtest hinsichtlich der Trajektorie.

Insgesamt bewerten die bekannten Validierungsansätze zwar die Genauigkeit einer XiL-Umgebung hinsichtlich der Trajektorie oder Kritikalitätsmetriken. Im Kontext der Glaubwürdigkeit einer XiL-Umgebung ist jedoch die Frage offen, inwiefern Aussagen, die aus Tests in XiL-Umgebungen generiert werden, auf den Realtest übertragbar sind. Aus diesem Grund wird in Kapitel 5 eine Glaubwürdigkeitsbewertung vorgestellt, die einen stärkeren Fokus auf Pass/Fail-Kriterien für Testfälle legt und statistische Einflüsse von Testumgebungen mit berücksichtigt.

4 Integration des szenariobasierten XiL-Tests in einen Entwicklungsprozess

4.1 Ausgangssituation

4.1.1 Referenzprojekt und System Under Test

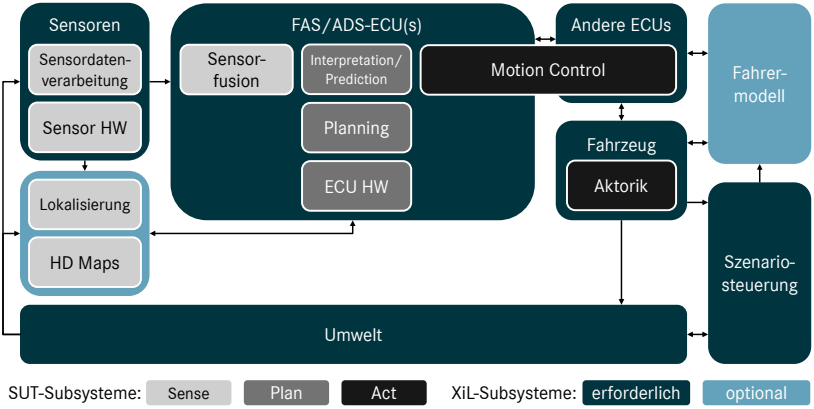

Abbildung 4.1: Übersicht der technischen Systemarchitektur des SUT und der Subsysteme der XiL-Umgebung [311]

Zur Einordnung von Einsatzmöglichkeiten des szenariobasierten XiL-Tests stellt ein Entwicklungsprojekt für Fahrerassistenzsysteme von Nutzfahrzeugen die Referenz dar (siehe Abschnitt 1.3). Entwicklungsgegenstand und damit SUT sind die in 2.1.2 definierten Bereiche „Sense“, „Act“ und „Plan“ der Systemarchitektur. Diese lassen sich in eine technische Architektur entsprechend

Abb. 4.1 in Subsysteme (grau/schwarz) strukturieren, sodass ein Großteil der Funktionen in einem dedizierten Steuergerät für Fahrerassistenzsysteme integriert ist. Das SUT wird durch Umweltsensoren, Modelle anderer Steuergeräte, einem Fahrzeugmodell sowie einer Umweltsimulation in einer XiL-Umgebung (siehe 2.5.2) stimuliert. Für den Test von Fahrerassistenzsystemen wird zusätzlich ein Fahrermodell benötigt. Optional ist die Repräsentation von Lokalisierung und HD Maps. Sofern für die jeweilige Einsatzmöglichkeit relevant, ist grundsätzlich ein Subset der in Abb. 4.1 dargestellten Subsysteme als SUT definierbar (siehe Abb. 4.4). Als konkrete Beispielsysteme zur Illustration der in Abschnitt 4.2 folgenden Einsatzmöglichkeiten werden ein Notbremsassistent und ein Abbiegeassistent herangezogen.

Der Fokus der neuen Testmethode liegt auf dem dynamischen Test des funktionalen Verhaltens, sodass insbesondere folgende Aspekte im Rahmen dieser Dissertation nicht berücksichtigt werden:

- statische Tests wie Reviews, Codeanalyse oder formale Methoden (siehe 2.4.1)
- nichtfunktionale Tests (siehe 2.4.1)
- reine Hardware-Tests (siehe 2.3.1)
- funktionale Sicherheit (siehe 2.3.1)
- Performance-Tests (siehe 2.4.1) mit Fokus auf Hardware-Komponenten (Ausnahme: Sensor-Fehler via Fehlerinjektion)
- alle sonstigen Testfälle, die sich nicht mithilfe eines Szenarios gemäß Definition aus 2.3.3 beschreiben lassen.

4.1.2 Anforderungen an eine neue Testmethode[1]

Realtests stellen ein wesentliches Element bei der Absicherung von Fahrerassistenzsystemen in heutigen Entwicklungsprojekten dar (siehe 2.5.1). Aus ökonomischen und praktischen Gesichtspunkten sind die Umfänge von Real-

[1] Teile dieses Abschnitts wurden bereits in ähnlicher Form vom Autor in [311] publiziert.

tests limitiert [289], während gleichzeitig drei Trends dazu führen, dass die Testumfänge steigen:

1. Die steigende Anzahl an Funktionen von Fahrerassistenzsystemen und der wachsende Einsatzbereich bestehender Fahrerassistenzsysteme führt zu einer höheren Varianz von Einsatzszenarien [13]. Gleiches gilt aufgrund zusätzlicher Sensorik.
2. Die Anforderungen an die Zuverlässigkeit der Systeme steigen, sowohl auf Kundenseite als auch durch den Gesetzgeber [10]. Neben der Kundenzufriedenheit kommt Haftungsfragen eine steigende Bedeutung zu.
3. Die Absicherung von Systemen ab SAE Level 3 beinhaltet im Vergleich zu niedrigeren Automatisierungslevels die zusätzliche Herausforderung, dass das System zumindest temporär die Rückfallebene darstellt [5]. Dies erhöht die Anforderungen an die Systemzuverlässigkeit und führt zu neuen abzusichernden Szenarien.

Dass der steigende Absicherungsaufwand zukünftig mit den bestehenden auf Realversuchen basierenden Absicherungsprozessen nicht mehr realisierbar ist, zeigt beispielsweise die Abschätzung in [288]: Um mit einer Wahrscheinlichkeit von 99% eine vollständige Abdeckung von möglichen Ausprägungen eines logischen Szenarios zu erreichen, müsste bereits mehr als die 300-fache Anzahl der Ausprägungen in Feldtests eingefahren werden. Dies führt bereits bei einer niedrigen zweistelligen Anzahl an Szenarioparametern zu mehreren Milliarden erforderlichen realen konkreten Szenarien für vollständige Testabdeckung.

Zu erwarten ist sowohl ein Anstieg der Breite als auch der Tiefe abzusichernder Szenarien. Die Testbreite t_b wird definiert als die Anzahl logischer Testfälle pro System[2] mit n_s als Anzahl der zu testenden Systeme:

$$t_b = \frac{n_l}{n_s} \tag{4.1}$$

Falls sich Testfälle nicht eindeutig auf Systeme zuordnen lassen, kann für die Quantifizierung der Testbreite auch die Gesamtzahl logischer Testfälle

[2] Der Begriff System bezieht sich hier auf Fahrerassistenzsysteme wie in 2.1.3 beschrieben.

herangezogen werden. Die Testbreite ist somit nicht zum Vergleich des Test- bzw. Absicherungskonzepts unterschiedlicher Systeme geeignet. Als Testtiefe t_t wird die Anzahl nicht redundanter[3] konkreter Testfälle (n_c) pro logischem Testfall (n_l) definiert:

$$t_t = \frac{n_c}{n_l} \tag{4.2}$$

Eine steigende Testtiefe, Testbreite oder Anzahl der Systeme erhöht durch Gl. 4.1 und Gl. 4.2 jeweils die Anzahl konkreter Testfälle.

$$n_c = t_t \cdot t_b \cdot n_s \tag{4.3}$$

Das Ziel ist, den Absicherungsaufwand neuer Fahrerassistenzsysteme und höher automatisierter Systeme in einem realisierbaren Rahmen zu halten. [105] Eine neue Testmethode muss daher die folgenden Anforderungen erfüllen:

1. **Reduktion des Realtest-Anteils**: Realtests stellen die Testumgebung mit dem größten Aufwand dar. Gleichzeitig ist deren Reproduzierbarkeit aufgrund von Unsicherheiten begrenzt (siehe 5.3.1). Daher ist die Verlagerung von Testfällen in XiL-Umgebungen und somit die Reduktion des relativen Anteils von Testfällen im Realtest notwendig. Dabei sollten insbesondere redundante und sicherheitskritische Testfälle im Realtest vermieden werden. Für den Realtest-Anteil gilt:

$$a_r = \frac{n_c^r}{n_c^r + \sum_{\omega=1}^{k} n_{c,\omega}^x} \tag{4.4}$$

   mit n_c^r als Anzahl real durchgeführter konkreter Testfälle und $n_{c,\omega}^x$ als Anzahl der in XiL-Umgebung ω durchgeführter konkreter Testfälle.

[3] Redundant bezieht sich hier auf die Art, wie das SUT stimuliert wird. Ein redundanter Testfall liefert somit keine neuen Erkenntnisse über das SUT-Verhalten.

2. **Erhöhung der Testabdeckung**: War es bislang möglich, Testfallkataloge manuell zu generieren, so wird dies bei steigender Testtiefe herausfordernder und aufwändiger. Ziel ist es, den Anteil der Szenarien zu quantifizieren und zu minimieren, die bekannt [97], ungetestet und nicht redundant sind. Die Testabdeckung ist definiert als (siehe Gl. 2.1):

$$a_t = \frac{n_{c,t}}{n_{c,t} + n_{c,u}} \tag{4.5}$$

   mit der Anzahl bekannter getesteter ($n_{c,t}$) und bekannter ungetesteter ($n_{c,u}$), jeweils nicht redundanter konkreter Szenarien.

3. **Unabhängigkeit von der XiL-Konfidenz**: Ergebnisse aus Tests in XiL-Umgebungen sind aufgrund ihrer eingeschränkten Konfidenz (siehe 3.3.1) nicht immer mit dem Realtest vergleichbar. Um dennoch belastbare Aussagen in allen Testumgebungen zu erhalten, muss die XiL-Unsicherheit U^x (siehe 5.3.1) im Prozess identifiziert und bei der Gesamtaussage berücksichtigt werden. Ebenso trägt eine geringe Sensitivität[4] χ des für die Gesamtaussage relevanten Testergebnisses gegenüber der XiL-Unsicherheit positiv zu dieser Anforderung bei.

   Für die Übertragbarkeit eines XiL-Testergebnisses T_i^x auf die Realität (T_i^r) gilt:

$$P(T_i^r | T_i^x) = f_P(U^x, \chi) \tag{4.6}$$

4. **Identifikation von neuen Szenarien**: Alleine durch manuell definierte Testfallkataloge lassen sich nicht alle in der Realität vorkommenden Szenarien erfassen. Es werden daher Methoden benötigt, um neue Szenarien zu identifizieren und so den Anteil bekannter [97] logischer Szenarien maximieren.

[4] Eine geringe Sensitivität zeichnet sich hier dadurch aus, dass Testergebnisse bzw. daraus abgeleitete Aussagen eine geringe Abhängigkeit gegenüber Unsicherheiten der Testumgebung haben. Die Sensitivität ist abhängig vom Anwendungsfall und hier als theoretische Größe zu verstehen.

Dieser ist wie folgt definiert:

$$a_b = \frac{n_{l,b}}{n_{l,b} + n_{l,u}} \tag{4.7}$$

$n_{l,b}$ stellt die Anzahl bekannter logischer Szenarien dar, $n_{l,u}$ die Anzahl unbekannter logischer Szenarien. Da letztere per Definition unbekannt ist, kann sie nur abgeschätzt werden [21].

5. **Priorisierung von Szenarien**: Da Testkapazitäten unabhängig von der Testumgebung begrenzt sind, ist es wichtig, Testfälle zu priorisieren. Es wird daher das Maß R_i zur Bewertung der Relevanz eines konkreten Szenarios S_i definiert:

$$R_i = f_R(S_i) \tag{4.8}$$

4.1.3 Szenariobasierter XiL-Test als Lösungsansatz

Die Simulation wird als zentrales Element gesehen, um Realtestumfänge zu reduzieren [289] [13]. Bereits heute existieren einzelne Funktionen wie beispielsweise ESP, die auf Basis von Simulation abgesichert werden [121]. Auch im Bereich der Fahrerassistenzsysteme kommt die Simulation in XiL-Umgebungen (siehe 2.5.2) in aktuellen Entwicklungsprojekten zum Einsatz.

Für die Definition von Testfällen ist das in 3.1.4 vorgestellte szenariobasierte Testen ein vielversprechender und in der wissenschaftlichen Diskussion häufig genannter Ansatz [165] [186] [187]. Die Nomenklatur für Szenarien und Testfälle wurde bereits bei der Anforderungsdefinition verwendet.

Die Kombination aus szenariobasiertem Testen und XiL bietet großes Potential, den genannten Herausforderungen zu begegnen und die definierten Anforderungen zu erfüllen [290] [311]. Zu nennen ist insbesondere die Skalierbarkeit und effiziente Testdurchführung im Vergleich zu bestehenden Ansätzen. Aus diesen Gründen wird für alle weiteren Betrachtungen der szenariobasierte XiL-Test als Grundlage weiterer Ansätze verwendet.

4.2 Neue Einsatzmöglichkeiten bei der Entwicklung von Fahrerassistenzsystemen[5]

Als Ergänzung der in 2.5.2.5 skizzierten Anwendungsfälle werden in diesem Abschnitt drei Einsatzmöglichkeiten des szenariobasierten XiL-Tests identifiziert und beschrieben. Die Einordnung findet ebenfalls in den Prozess aus 2.2.5 statt (siehe Abb. 4.2), der eine Kombination aus Scrum, V-Modell und Stage-Gate-Modell repräsentiert. Zur Namensgebung der Einsatzmöglichkeiten werden die in Automotive SPICE verwendeten Bezeichnungen der Software Engineering Prozesse herangezogen (siehe 2.3.2 und Abb. 2.13). Es handelt sich um keine abgeschlossene Auflistung, sodass weitere Einsatzmöglichkeiten denkbar sind.

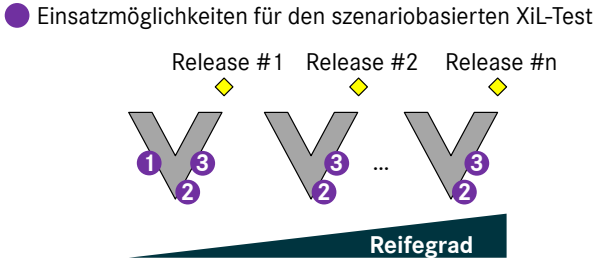

Abbildung 4.2: Einordnung der Einsatzmöglichkeiten 1, 2 und 3 für den szenariobasierten XiL-Test in den bestehenden Entwicklungsprozess

Definiert werden die folgenden neuen Einsatzmöglichkeiten:

1. Software-Anforderungsanalyse
2. Software Unit Verifikation
3. Software-Qualifizierungstest

Abb. 4.3 veranschaulicht die Evolution des SUT über die verschiedenen Einsatzmöglichkeiten und zeigt auf, welche XiL-Umgebungen sich je nach Einsatzmöglichkeit eignen. In Abb. 4.4 ist die Art der Repräsentation der einzelnen

[5] Teile dieses Abschnitts wurden bereits in ähnlicher Form vom Autor in [311] publiziert.

Teile des SUT und der Umgebung für jede Einsatzmöglichkeit basierend auf Abb. 4.1 beschrieben.

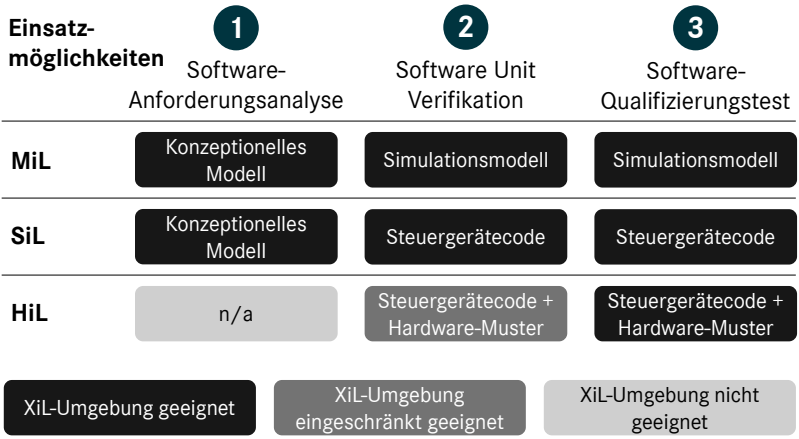

Abbildung 4.3: SUT-Evolution und Eignung von XiL-Umgebungen nach Einsatzmöglichkeit

4.2.1 Einsatzmöglichkeit 1: Software-Anforderungsanalyse

Bereits zur Software-Anforderungsanalyse kann der szenariobasierte XiL-Test einen Mehrwert leisten, indem mithilfe von Szenarien eine systematische und automatisierte Bewertung von Konzepten vorgenommen wird. Es handelt sich dabei, wie in Abb. 4.2 dargestellt, um einen sehr frühen Zeitpunkt im Entwicklungsprojekt. Ziel ist es, die Spezifikation des SUT vor einem initialen Release zu unterstützen. Der Nutzen der Simulation bei der Systemspezifikation wird ebenfalls in [23] erwähnt.

SUT ist noch nicht das tatsächliche Seriensteuergerät mit Seriencode, sondern ein ausführbares Verhaltensmodell, das die Spezifikation repräsentiert (siehe Abb. 4.4). Die Integration des SUT kann entweder als ausführbares Simulationsmodell in einer MiL-Umgebung als auch in Form eines Simulationsmoduls (z.B. als FMU) zusammen mit ggf. schon existierendem Steuergeräte-Code in einer SiL-Umgebung stattfinden (siehe Abb. 4.3). Für die Anwendung auf

Einsatzmöglichkeiten / Subsysteme	1 Software-Anforderungsanalyse	2 Software Unit Verifikation	3 Software-Qualifizierungstest
Sensordatenverarbeitung	simuliert	simuliert	simuliert/real
Sensor HW	simuliert	simuliert	simuliert/real
Lokalisierung	simuliert	simuliert	simuliert/real
Karte	simuliert	simuliert	simuliert/real
Sensorfusion	simuliert	simuliert/real	real
Interpretation/Prediction	simuliert	real	real
Planning	simuliert	real	real
ECU HW	simuliert	simuliert/real	simuliert/real
Motion Control	simuliert	real	real
Aktorik	simuliert	simuliert	simuliert

SUT | optional SUT | nicht SUT

Abbildung 4.4: Repräsentation von SUT und Umgebung je Einsatzmöglichkeit

Serien-Hardware-Mustern ist diese Einsatzmöglichkeit nicht geeignet, da das SUT nicht als Seriencode vorliegt.

Das szenariobasierte Testen erlaubt hier eine automatisierte Generierung konkreter Szenarien über eine Parametervariation. Die dabei verwendeten logischen Szenarien können zur Definition der ODD (siehe 2.1.1.1) verwendet werden. Aufgrund der skalierbaren Menge konkreter Szenarien lässt sich hier bereits eine große Testtiefe t_t erreichen. Sind Parameterverteilungen aus Realdaten bekannt, so können diese bereits zur Gewichtung der Bewertung einzelner konkreter Szenarien verwendet werden. Daneben ist es möglich, über eine systematische Fehlerinjektion in Sensor- oder Fahrzeugmodelle die Robustheit des Konzepts zu testen. Für die Bewertung des Konzepts kommen sowohl Pass/Fail-Kriterien als auch Kritikalitätsmetriken (siehe 3.2.1) in Betracht. Dies erlaubt eine automatisierte Testauswertung.

Die Übertragung dieser Einsatzmöglichkeit auf einen Notbrems- bzw. Abbiegeassistenten zum Test in einer SiL-Umgebung ergäbe folgenden Beispielprozess:

1. Definition logischer Szenarien (incl. Parameterverteilungen) auf Basis von Expertenwissen, Unfalldaten und Analyse von Felddaten. Beispielszenarien sind das Auffahren auf ein Stauende oder das Abbiegen mit einem Fußgänger lateral neben dem Fahrzeug.
2. Definition von Kritikalitätsmetriken (z.B. TTCmin) und Pass/Fail-Kriterien (z.B. Auftreten einer Kollision, Auftreten der Notbremsung).
3. Integration der **konzeptionellen Modelle** in eine MiL/SiL-Umgebung.
4. Generierung konkreter Szenarien über eine **automatisierte** Parametervariation.
5. Test der generierten Szenarien in der MiL/SiL-Umgebung.
6. Berechnung und Visualisierung der Kritikalitätsmetriken und Pass/Fail-Kriterien.
7. Evaluation der Leistungsfähigkeit des Systems basierend auf Kritikalitätsmetriken und Pass/Fail-Kriterien.
8. Ggf. **unmittelbare Anpassung** des konzeptionellen Modells und Wiederholung der MiL/SiL-Tests.

4.2.2 Einsatzmöglichkeit 2: Software Unit Verifikation

Die Software Unit Verifikation findet in Form von Entwicklertests in den unteren Ebenen des V-Modells statt und schließt unmittelbar an die Unit-Implementierung an (siehe 2.3.2). Ziel ist es, mithilfe von logischen und konkreten Szenarien Testfälle standardisiert zu beschreiben, sodass sie skalierbar sind und auf verschiedene Units oder deren Kombinationen anwendbar sind. Im Vordergrund steht primär das Verhalten des SUT in einzelnen konkreten Szenarien und weniger eine vollständige Abdeckung des Parameterraums.

SUT ist somit entweder die gesamte Software des Fahrerassistenz-Steuergeräts oder alternativ ein Subsystem (siehe Abb. 4.4). Auch hier kann entweder

ein ausführbares Simulationsmodell (MiL) oder der reale Steuergerätecode (SiL) in die Testumgebung integriert werden. Grundsätzlich kann eine HiL-Umgebung mit einem realen Steuergerät genutzt werden, sofern ein entsprechendes Hardware-Muster (A- bis D-Muster) verfügbar ist (siehe Abb. 4.3).

Die formalisierte Testfallbeschreibung mithilfe logischer Szenarien bietet Entwicklern ein zusätzliches Werkzeug zur Testfalldefinition als Alternative zu unitspezifischen Testfallkatalogen oder der manuellen Ansteuerung der Testumgebung (siehe 2.5.2.5). Dabei steht vor allem die Flexibilität bei der Generierung neuer konkreter Szenarien durch Parametervariation im Fokus. Dies kann sowohl rein manuell wie auch automatisiert basierend auf einem festen Szenario-Parametersatz erfolgen, um Regressionstests zu ermöglichen. Für nachfolgende Auswertungsschritte werden Pass/Fail-Kriterien und Kritikalitätsmetriken genutzt, die allerdings spezifisch vom Entwickler ergänzt bzw. modifiziert werden können.

Für die Systeme Notbremsassistent und Abbiegeassistent sähe der Beispielprozess bei einem SiL-Test wie folgt aus:

1. Definition logischer Szenarien auf Basis von Expertenwissen bzw. dem Wissen über die Systemstruktur. Beispielszenarien sind ebenfalls das Auffahren auf ein Stauende oder das Abbiegen mit einem Fußgänger lateral neben dem Fahrzeug.
2. Definition von Kritikalitätsmetriken (z.B. TTCmin) und Pass/Fail-Kriterien (z.B. Auftreten einer Kollision, Auftreten der Notbremsung).
3. Integration der **Software-Units** in eine XiL-Umgebung.
4. Generierung konkreter Szenarien über eine **manuelle** Parametervariation.
5. Test der generierten Szenarien in der XiL-Umgebung.
6. Berechnung und Visualisierung der Kritikalitätsmetriken und Pass/Fail-Kriterien.
7. Evaluation der Leistungsfähigkeit des Systems basierend auf Kritikalitätsmetriken und Pass/Fail-Kriterien.
8. Ggf. **unmittelbare Anpassung** der Software-Unit und Wiederholung der XiL-Tests.

4.2.3 Einsatzmöglichkeit 3: Software-Qualifizierungstest

Die Einsatzmöglichkeit Software-Qualifizierungstest ordnet sich in die in Abb. 2.11 dargestellten Freigaben ein und folgt auf den Software-Integrationstest (siehe 2.3.2). Testfälle werden dabei basierend auf logischen Szenarien generiert und nicht wie bisher in statischen Testfallkatalogen (siehe 2.5.2.5). Ziel ist es, mithilfe einer Parametervariation konkrete Szenarien zu generieren und eine hohe Testtiefe t_t zu erreichen. Gleichzeitig muss sichergestellt werden, dass über einen Katalog logischer Szenarien eine ausreichende Testbreite t_b erreicht wird. Ansätze ähnlich dieser Einsatzmöglichkeit finden sich bereits in [108] und [213].

SUT sind bei dieser Einsatzmöglichkeit alle Software-Module des FAS-Steuergeräts (siehe Abb. 4.4). Alternativ können einzelne Subsysteme der Software (z.B. Interpretation/Prediction und Planning) getestet werden, was an dieser Stelle jedoch nicht weiter vertieft wird. Für den Test von Software-Modulen als Simulationsmodell sind grundsätzlich MiL-Umgebungen geeignet. Sofern Software-Module als realer Steuergerätecode zu testen sind, wird dies in einer SiL-Umgebung ausgeführt. Für den Test von auf das reale Steuergerät integrierter Software wird eine HiL-Umgebung und entsprechende Hardware-Muster (A- bis D-Muster) benötigt. In letzterem Fall kann zusammen mit der realen Sensorik (z.B. Kamerasensor) getestet werden, wobei die reale Sensor-Hardware hier Teil der Testumgebung und nicht des SUT ist.

Ähnlich zur bereits genannten Einsatzmöglichkeit in 4.2.1 kommt hier eine Parametervariation für ein logisches Szenario zum Einsatz, welche in diesem Fall jedoch auf den gesamten Parameterraum angewendet wird. Die Parameterverteilungen können optional aus Realdaten abgeleitet sein. Die dabei möglichen Kombinationen an Parameterausprägungen sind jedoch in vielen Fällen zu zahlreich[6], um alle in einer XiL-Umgebung getestet zu werden. Somit sind geeignete Sampling-Algorithmen (siehe 3.1.4.6) notwendig, um den Parameterraum optimal abzudecken. Sofern es sich um einen deterministischen Sampling-Algorithmus handelt, sind Regressionstests für eine Conti-

[6] Nimmt man beispielsweise bei zehn Parametern je fünf mögliche Ausprägungen an, so kommt man auf 5^{10} = 9.765.625 mögliche Szenarien. Bei einer Rechenzeit von 1 min pro Szenario entspräche dies einer gesamten Rechenzeit von mehr als 18 Jahren.

nuous Integration des SUT möglich. Die Auswertung erfolgt in diesem Fall über Pass/Fail-Kriterien sowie Kritikalitätsmetriken.

Über einen Sampling-Algorithmus generierte konkrete Szenarien werden zunächst in einer XiL-Umgebung getestet. Dafür eignen sich vor allem SiL-Umgebungen, da diese schneller als in Echtzeit betrieben werden können. Über die Analyse der Ähnlichkeit der getesteten Szenarien werden aus diesen Cluster gebildet, die jeweils einer Äquivalenzklasse entsprechen. Somit ist es möglich, eine Vorauswahl für Szenarien zu treffen, die im Realtest getestet werden und so die Anzahl redundanter Szenarien im Realtest zu minimieren.

Es ergibt sich folgender Beispielprozess für die Systeme Notbremsassistent und Abbiegeassistent:

1. Definition logischer Szenarien (incl. Parameterverteilungen) auf Basis von Expertenwissen, Wissen über die Systemstruktur, Unfalldaten und Analyse von Felddaten. Beispielszenarien sind auch hier das Auffahren auf ein Stauende oder das Abbiegen mit einem Fußgänger lateral neben dem Fahrzeug.
2. Definition von Kritikalitätsmetriken (z.B. TTCmin) und Pass/Fail-Kriterien (z.B. Auftreten einer Kollision, Auftreten der Notbremsung).
3. Integration der **SUT-Software** in eine XiL-Umgebung.
4. Generierung konkreter Szenarien über eine **automatisierte** Parametervariation.
5. Test der generierten Szenarien in der XiL-Umgebung.
6. Berechnung und Visualisierung der Kritikalitätsmetriken und Pass/Fail-Kriterien.
7. Aggregation der konkreten Szenarien in Cluster.
8. Durchführung Glaubwürdigkeitsbewertung.
9. Ggf. Wiederholung der Tests in weiterer XiL-Umgebung (**keine Anpassung des SUT**).

Für weitere Details des hier skizzierten Prozesses wird auf Abschnitt 4.3 verwiesen.

4.2.4 Diskussion der Einsatzmöglichkeiten

Die Einsatzmöglichkeiten grenzen sich primär durch ihre Integration in den in Abb. 4.2 dargestellten Entwicklungsprozess ab. Während Einsatzmöglichkeit 1 vorrangig zu einem frühen Zeitpunkt im Entwicklungsprozess und mit einem konzeptionellen Modell des SUT zum Einsatz kommt, sind die Einsatzmöglichkeiten 2 und 3 für jeden Reifegrad des SUT geeignet. Hieraus ergeben sich unterschiedliche Repräsentationen des SUT, wie in Abb. 4.3 und Abb. 4.4 aufgeführt. Daneben unterscheiden sich die jeweiligen Einsatzmöglichkeiten hinsichtlich des Automatisierungsgrads bei der Definition bzw. Generierung logischer und konkreter Szenarien.

Einsatzmöglichkeiten / Anforderungen	1 Software-Anforderungsanalyse	2 Software Unit Verifikation	3 Software-Qualifizierungstest
1 Reduktion des Realtest-Anteils	niedrig	niedrig	hoch
2 Erhöhung der Testabdeckung	hoch	mittel	hoch
3 Unabhängigkeit von der XiL-Konfidenz	niedrig	hoch	mittel
4 Identifikation von neuen Szenarien	hoch	niedrig	hoch
5 Priorisierung von Szenarien	hoch	mittel	hoch

Abbildung 4.5: Anforderungserfüllung durch Einsatzmöglichkeiten für szenariobasierten XiL-Test

Die Anforderungserfüllung durch die jeweiligen Einsatzmöglichkeiten ist in Abb. 4.5 qualitativ dargestellt und wird im Folgenden erläutert:

1. **Reduktion des Realtest-Anteils**: Sowohl Einsatzmöglichkeit 1 als auch 2 sind nicht primär dafür vorgesehen, Realtests zu reduzieren. Jedoch lassen sich in beiden Fällen Fehler im SUT früher identifizieren, was eine indirekte Reduktion von n_c^r bewirkt. Die Anzahl der XiL-Tests $n_{c,\omega}^x$ wird in beiden Fällen leicht erhöht, da es sich um zusätzliche Testfälle handelt. Im Gegensatz dazu ist es in Einsatzmöglichkeit 3 möglich, n_c^r

direkt durch den Einsatz von XiL-Tests zu reduzieren. Zudem ist diese Einsatzmöglichkeit auf eine hohe Skalierbarkeit von $n^{x}_{c,\omega}$ ausgelegt, sodass sich insgesamt ein großes Potential zur Reduktion von a_r ergibt.

2. **Erhöhung der Testabdeckung**: Die Testabdeckung lässt sich in allen Einsatzmöglichkeiten mithilfe einer Parametervariation steigern, sodass sich $n_{c,t}$ steigern und $n_{c,u}$ reduzieren lässt. Für die Bewertung der Redundanz von Szenarien eignen sich Szenariodistanzmaße (siehe 3.2.3). Aufgrund der manuellen Parametervariation in Einsatzmöglichkeit 2 ist die Anforderungserfüllung nur zu einem mittleren Grad gegeben.

3. **Unabhängigkeit von der XiL-Konfidenz**: Einsatzmöglichkeit 1 beinhaltet die größte XiL-Unsicherheit U^x, da zu einem frühen Entwicklungszeitpunkt ggf. keine oder wenige Daten zur Validierung von Simulationsmodellen oder zur Bewertung der Konfidenz zur Verfügung stehen. Gleichzeitig ist die Sensitivität χ hoch, da aus den XiL-Tests unmittelbar Rückschlüsse auf die SUT-Spezifikation gezogen werden, sodass sich eine niedrige Anforderungserfüllung für Einsatzmöglichkeit 1 ergibt. Hingegen ist es in Einsatzmöglichkeit 2 und 3 möglich, auf Realdaten zur Validierung der XiL-Umgebung zurückzugreifen, sodass U^x reduziert oder zumindest quantifiziert werden kann. χ ist in Einsatzmöglichkeit 2 niedrig, da für die Software Unit Verifikation vorrangig die Stimulation einzelner Software-Units und die Sicherstellung der grundsätzlichen Funktionalität im Fokus steht. Damit ergibt sich für Einsatzmöglichkeit 2 eine hohe und für Einsatzmöglichkeit 3 eine mittlere Anforderungserfüllung.

4. **Identifikation von neuen Szenarien**: Die Identifikation neuer logischer Szenarien ist in den Einsatzmöglichkeiten 1 und 3 insbesondere durch die Analyse von Feld- und Unfalldaten möglich, sodass sich hier eine hohe Anforderungserfüllung ergibt. Daneben ist es weiterhin möglich, bestehende Prozesse wie beispielsweise die Gefahren- und Risikoanalyse (siehe 2.3.1) zu nutzen bzw. zu ergänzen. Die Definition logischer Szenarien in Einsatzmöglichkeit 2 ist primär manuell und bietet somit nur eingeschränktes Potential, $n_{l,b}$ zu steigern bzw. $n_{l,u}$ zu reduzieren.

5. **Priorisierung von Szenarien**: Die Priorisierung von Szenarien anhand eines Relevanzmaßes R_i kann in den Einsatzmöglichkeiten 1 und 3 mithilfe von Kritikalitätsmetriken (siehe 3.2.2), Szenariodistanzmaßen (sie-

he 3.2.3) oder der Häufigkeitsverteilung von Szenarioparametern vorgenommen werden. Somit ergibt sich hier eine hohe Anforderungserfüllung. Die manuelle Auswahl von Szenarien anhand von Expertenwissen in Einsatzmöglichkeit 2 führt nur zu einer mittleren Anforderungserfüllung, da kein objektives Maß R_i existiert.

Die Einsatzmöglichkeit Software-Qualifizierungstest besitzt großes Potential, die Vorteile des szenariobasierten XiL-Tests zum Tragen zu bringen (siehe Abb. 4.5). Daher wird im weiteren Verlauf dieser Dissertation der Fokus auf diese Einsatzmöglichkeit gelegt und diese in Abschnitt 4.3 weiter detailliert.

4.3 Prozessentwurf für den Software-Qualifizierungstest

4.3.1 Anforderungen und Voraussetzungen

Aufbauend auf die in 4.2.3 eingeführte Einsatzmöglichkeit wird in diesem Abschnitt ein möglicher Prozessentwurf zur Generierung und Zuordnung konkreter Szenarien auf verschiedene XiL-Umgebungen vorgestellt. Der Prozess kombiniert Elemente und Konzepte aus dem Stand der Technik (siehe Kapitel 3) und stellt eine mögliche Konkretisierung der Einsatzmöglichkeit 3 (siehe 4.2.3) dar. Von den definierten Prozessschritten wird lediglich die Glaubwürdigkeitsbewertung (Schritt 9, siehe 4.3.2) im Rahmen dieser Dissertation in Kapitel 5 weiter detailliert. Die in 4.1.2 definierten Anforderungen dienen als Zielgrößen und werden wie folgt erfüllt:

1. **Reduktion des Realtest-Anteils**: Testfälle werden bevorzugt in der hinsichtlich zeitlicher und ökonomischer Gesichtspunkten effizientesten XiL-Umgebung ausgeführt. Die Effizienz wird quantifiziert über die effektiven Kosten pro Testfall $C_{t,\omega}$ für die XiL-Umgebung ω aus der abzählbaren Menge aller XiL-Umgebungen Ω. Diese umfassen sowohl fixe Kosten ($C_{f,\omega}$) als auch variable Kosten pro Testdurchführung ($C_{v,\omega}$). $n^x_{c,\omega}$ stellt die Anzahl durchgeführter konkreter Testfälle in der entsprechenden XiL-Umgebung ω dar.

Es ergibt sich für die effektiven Kosten pro Testfall:

$$C_{t,\omega} = \frac{C_{f,\omega}}{n^x_{c,\omega}} + C_{v,\omega} \tag{4.9}$$

Ziel ist somit die Minimierung von $C_{t,\omega}$ für jedes getestete Szenario:

$$\omega_{min} = \arg\min_{\omega \in \Omega}(C_{t,\omega}) \tag{4.10}$$

Es wird in dieser Dissertation angenommen, $C_{f,\omega}$ und dass $C_{v,\omega}$ quantifizierbar und bekannt sind sowie, dass die XiL-Umgebung ω_{min} mit minimalem $C_{t,\omega}$ eine SiL-Umgebung ist. Realtests werden nur dann durchgeführt, wenn dies für die Plausibilisierung bzw. Validierung (siehe Kapitel 5) zwingend erforderlich ist oder keine weitere XiL-Umgebung bereitsteht (siehe Schritt 11 in 4.3.2). Somit kann n^r_c reduziert werden bei gleichzeitiger Steigerung von $n^x_{c,\omega}$.

2. **Erhöhung der Testabdeckung**: Es kommt ein deterministischer Sampling-Algorithmus zum Einsatz, der die Erfüllung von Abdeckungskriterien (z.B. t-weise Abdeckung [213], siehe 3.1.4.6) sicherstellt [7]. Die Wahl der Parameter des logischen Szenarios und deren möglicher Ausprägungen muss eine möglichst hohe Abdeckung real vorkommender Parameter repräsentieren. Dies kann optional durch Parameterverteilungen aus Realdaten unterstützt werden [201]. Eine ausreichende Testabdeckung ist erreicht, wenn eine t-weise Abdeckung von t_{min} sichergestellt ist:

$$t \overset{!}{\geq} t_{min} \tag{4.11}$$

[7] Grundsätzlich sind auch stochastische Sampling-Algorithmen an dieser Stelle denkbar. In diesem Anwendungsfall stehen jedoch Testabdeckung und Reproduzierbarkeit im Fokus und nicht die Identifikation kritischer Szenarien. Daher wird ein deterministischer Sampling-Algorithmus angewandt. Es ist dennoch auch bei einem solchen Algorithmus möglich, über einen variablen Random-Seed ein nichtdeterministisches Verhalten herbeizuführen.

Für t_{min} werden in [213] mögliche Werte zwischen 3 und 10 genannt. Ein höheres t_{min} stellt eine höhere Anforderung hinsichtlich der Testabdeckung dar. Es wird angenommen, dass der jeweilige Wert für t_{min} mithilfe von Expertenwissen oder eines objektiven Prozesses festgelegt wird. Durch die Anwendung der t-weisen Abdeckung können redundante Szenarien und somit auch $n_{c,u}$ reduziert werden.

3. **Unabhängigkeit von der XiL-Konfidenz**: Die Konfidenz der XiL-Umgebung ω muss durch eine Glaubwürdigkeitsbewertung (siehe Abschnitt 5.1) im Prozess berücksichtigt werden. Eine XiL-Umgebung wird dann als glaubwürdig klassifiziert, wenn eine minimal erforderliche Konfidenzstufe K_{min} erreicht wird:

$$K_\omega \overset{!}{\geq} K_{min} \tag{4.12}$$

   K_{min} stellt hier eine diskrete Stufe zwischen $K1$ und $K3$ dar (siehe Abschnitt 5.2). Die Konfidenzstufe liefert eine qualitative Aussage zu $P(T_i^r|T_i^x)$ und ist unter anderem abhängig von U^x (siehe Abschnitt 5.1).

4. **Identifikation von neuen Szenarien**: Die Erfüllung dieser Anforderung wird durch geeignete Methoden zum Aufbau bzw. der Erweiterung einer Datenbank logischer Szenarien erfüllt. Dies ist jedoch nicht Teil des hier vorgestellten Prozesses und wird daher nicht weiter betrachtet.

5. **Priorisierung von Szenarien**: Testfälle für den Realtest werden so gewählt, dass die Anzahl redundanter Testfälle minimiert ist und sich diese voneinander signifikant unterscheiden. Hierfür wird ein Szenariodistanzmaß d (siehe 3.2.3) in Kombination mit einem Clustering-Algorithmus verwendet, sodass pro Cluster nur noch ein konkretes Szenario im Realtest getestet werden muss. Um dies zu ermöglichen, müssen die gebildeten Cluster einen kompakten und konvexen Parameterraum aufspannen. Darüber hinaus soll für zum Realtest ausgewählte konkrete Szenarien S_i und S_j aus zwei Clustern gelten:

$$d(S_i, S_j) \geq d_{min} \tag{4.13}$$

Der Grenzwert d_{min} stellt die minimale Szenariodistanz zwischen zwei zum Realtest ausgewählten konkreten Szenarien dar. Es wird angenommen, dass dieser Wert auf Basis von Expertenwissen oder eines objektiven datenbasierten Prozesses festgelegt wird.

Ebenso wird die Kritikalität der Szenarien in Form von TDMs, FBMs und PFCs zur Priorisierung herangezogen, wenn nur eine begrenzte Anzahl konkreter Szenarien im Realtest testbar ist. Ziel ist es, Szenarien hoher Kritikalität priorisiert zu testen, sofern dies sicher im Realtest durchgeführt werden kann. Im weiteren wird jedoch davon ausgegangen, dass ausreichende Realtestkapazitäten vorhanden sind, sodass die Kritikalität nicht zur Priorisierung von Testfällen herangezogen wird.

Für das Maß R_i des Szenarios S_i ergibt sich:

$$R_i = f_R(\max(d(S_i, S_j)|j \in N_{c,t}), TDM(S_i), FBM(S_i), PFC(S_i)) \tag{4.14}$$

mit der Menge aller zuvor getesteten Szenarien $N_{c,t}$.

Aus den Anforderungen 1, 2, 3 und 5 ergibt sich ein Optimierungsproblem für die effektiven Kosten pro Testfall (Gl. 4.9) mit drei Nebenbedingungen (Gl. 4.11, Gl. 4.12 und Gl. 4.13), welches durch den hier beschriebenen Prozess gelöst wird. Die Priorisierung durch R_i wird im weiteren Verlauf nicht betrachtet. Folgende Voraussetzungen müssen vorab erfüllt sein:

- Es gibt mindestens eine Methode, um logische Szenarien zu definieren (siehe 3.1.4.3) und in eine Szenariodatenbank zu überführen. Dieser Schritt wird hier jedoch nicht weiter betrachtet.
- Zur Testdurchführung existiert mindestens eine SiL-Umgebung sowie eine weitere XiL-Umgebung. Ebenfalls muss die Möglichkeit für Realtests auf einem Prüfgelände gegeben sein.
- Das SUT ist in Form eines ausführbaren Software-Codes vorhanden. Für HiL-Tests oder Realtests sind entsprechende Musterstände der Zielhardware verfügbar.

4.3.2 Prozessschritte

Entsprechend der in Abb. 4.6 dargestellten Übersicht werden die folgenden Prozessschritte durchlaufen:

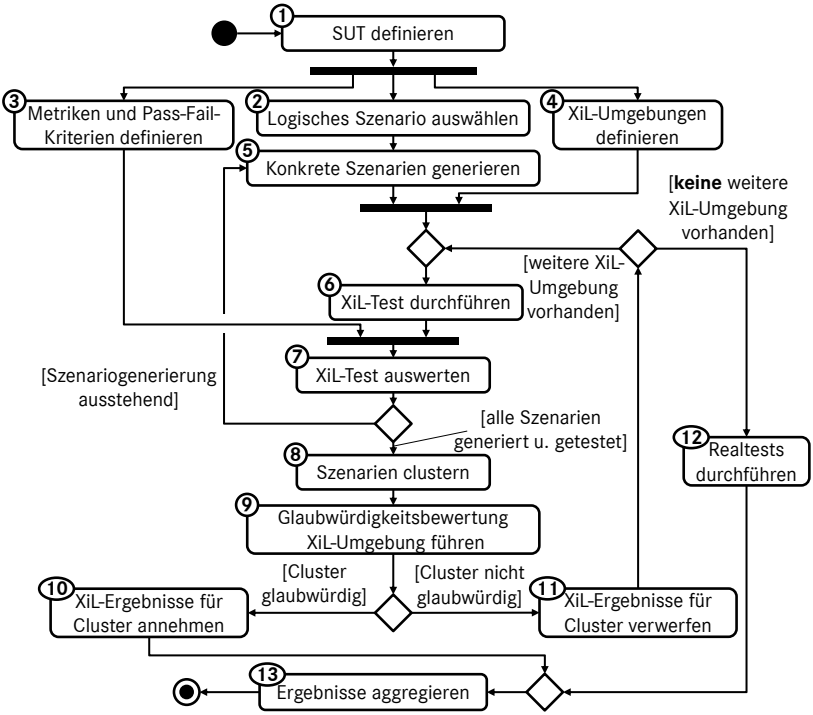

Abbildung 4.6: Aktivitätsdiagramm: Szenariobasierter XiL-Test für den Software-Qualifizierungstest

1. SUT definieren Ziel dieses initialen Schrittes ist es, eine Abgrenzung des SUT zum Rest der XiL-Umgebung vorzunehmen, sodass eindeutig ist, welcher Teil der Software getestet werden soll. Dies erfolgt anhand der Strukturierung aus Abb. 4.1 und Abb. 4.4.

- **Input**: Informationen über Struktur des ausführbaren Software-Codes
- **Output**: Definition des SUT (incl. Schnittstellen)

2. Logisches Szenario auswählen Aus einer Datenbank (siehe 3.1.4.3) wird nun ein logisches Szenario ausgewählt. Für den Test von Fahrerassistenzsystemen wird das Szenario spezifisch für eine zu testende Funktion gewählt. Dies ist für Systeme ab SAE Level 3 nicht mehr möglich, da hier eine Einteilung in einzelne Fahrfunktionen nicht mehr möglich ist. In diesem Fall werden logische Szenarien so selektiert, dass sie die ODD des Systems abdecken.

- **Input**: Definition des SUT
- **Output**: Logisches Szenario (S^l)

3. KPIs definieren Für die Schritte 7 und 9 werden KPIs (siehe 3.2.1) benötigt. Da diese stark vom SUT und dem verwendeten logischen Szenario abhängen, werden sie basierend auf Expertenwissen definiert. Metriken und Pass/Fail-Kriterien müssen in jeder zum Einsatz kommenden XiL-Umgebung sowie im Realtest berechnet werden können.

- **Input**: Definition des SUT, Expertenwissen, S^l
- **Output**: Definition KPIs (KPI)

4. XiL-Umgebungen definieren Die Wahl der XiL-Umgebungen richtet sich primär nach den vorhandenen Ressourcen hinsichtlich Simulationsmodellen, Simulationstools und Hardware. Eine XiL-Umgebung setzt die Existenz dieser Ressourcen voraus und ergibt sich aus deren Kombination. Die Kombination wird so gewählt, dass sich die einzelnen XiL-Umgebungen hinsichtlich $C_{t,\omega}$ differenzieren. Das minimale $C_{t,\omega}$ wird von einer SiL-Umgebung erreicht, die somit die effizienteste verfügbare XiL-Umgebung darstellt. Die Vorgaben für die Testdurchführung sind entsprechend der XiL-Umgebung zu wählen.

- **Input**: Definition des SUT, Sensormodelle, Umgebungssimulation, Fahrzeugmodelle, Restbussimulation (funktionale Modelle, simulierte Netzwerkkommunikation), Simulationstools, Simulationsrechner, Hardware-Muster, sonstige HiL-Hardware
- **Output**: SiL-Umgebung, mindestens eine weitere XiL-Umgebung, Vorgaben für die Testdurchführung, Kosten pro Testfall in XiL-Umgebung ($C_{t,\omega}$)

5. Konkrete Szenarien generieren Mithilfe eines Sampling-Algorithmus werden aus dem logischen Szenario konkrete Szenarien abgeleitet. Jedes konkrete Szenario entspricht einem eindeutigen Set an Parameterausprägungen. In diesem Prozessentwurf kommt ein deterministisches Sampling zum Einsatz, welches eine t-weise Abdeckung des Szenarioparameterraums sicherstellt. Optional kann der Sampling-Algorithmus Ergebnisse von vorherigen Testfällen berücksichtigen. In dieser Dissertation wird das Sampling nicht weiter detailliert.

- **Input**: S^l, Sampling-Algorithmus, Ergebnisse vorherige Testfälle (optional)
- **Output**: konkrete Szenarien (S_i), Szenario-Dateien im Zielformat der Simulationsumgebung

6. XiL-Test durchführen Die vom Samplingalgorithmus generierten konkreten Szenarien werden in diesem Schritt in der effizientesten verfügbaren XiL-Umgebung ausgeführt (minimales $C_{t,\omega}$). Sofern es sich um die erste Iteration handelt, ist dies die SiL-Umgebung (siehe Schritt 4). In folgenden Iterationen kommen weitere XiL-Umgebungen zum Einsatz. Vorgaben für die Testdurchführung werden automatisiert zur Laufzeit der Simulation geprüft.

- **Input**: XiL-Umgebung, Tool zur Testautomatisierung, Vorgaben für die Testdurchführung, S_i, $C_{t,\omega}$
- **Output**: Systemantwort individuell für jedes konkrete Szenario ($SRQ_i(t)$), sonstige Aufzeichnungen (z.B. Video)

7. XiL-Test auswerten Bei der Auswertung werden aus den aufgezeichneten Testdaten die zuvor definierten Metriken und Pass/Fail-Kriterien ermittelt. Diese werden wiederum in einer Datenbank eingetragen und dem jeweiligen konkreten Szenario zugeordnet. Je nach Sampling-Algorithmus ist eine Rückführung der Testergebnisse in Schritt 5 möglich.

- **Input**: KPI, $SRQ_i(t)$, sonstige Aufzeichnungen
- **Output**: KPIs individuell für jedes konkrete Szenario (KPI_i^x)

8. Szenarien clustern Ziel dieses Schritts ist es, basierend auf den Testdaten und der Testergebnisse ähnliche Szenarien zu clustern. Je Cluster wird anschließend ein repräsentatives konkretes Szenario ausgewählt und für die darauf folgende Glaubwürdigkeitsbewertung (Schritt 9) verwendet. Der Clustering-Algorithmus wendet ein Szenariodistanzmaß zur Quantifizierung der Ähnlichkeit zweier konkreter Szenarien an. Somit wird erreicht, dass alle konkreten Szenarien eines Clusters eine äquivalente Stimulation des SUT darstellen und ein repräsentatives konkretes Szenario je Cluster für weitere Betrachtungen herangezogen werden kann. Es gilt die Anforderung der Kompaktheit und Konvexität für Cluster (siehe Anforderung 5 in 4.3.1).

- **Input**: KPI_i^x, sonstige Aufzeichnungen, Szenariodistanzmaß d
- **Output**: Zuordnung jedes konkreten Szenarios zu einem Cluster, Repräsentatives konkretes Szenario für jedes Cluster (S_j)

9. Glaubwürdigkeitsbewertung für XiL-Umgebung führen Für jedes Cluster aus Schritt 8 wird in diesem Schritt die Glaubwürdigkeit bewertet (siehe Abschnitt 5.1). Maßgeblich hierfür ist die Glaubwürdigkeit der XiL-Ergebnisse des jeweiligen repräsentativen konkreten Szenarios. Sofern das Cluster in der einer XiL-Umgebung als glaubwürdig klassifiziert wird, muss aus diesem Cluster maximal ein konkretes Szenario im Realtest getestet werden.

- **Input**: S_j, XiL-Umgebung, Realdaten (wenn vorhanden)
- **Output**: XiL-Konfidenzstufe je Cluster ($K_{\omega,j}$), Glaubwürdigkeitsaussage für das jeweilige Cluster

10. XiL-Ergebnisse für Cluster annehmen XiL-Ergebnisse eines Clusters, das als glaubwürdig klassifiziert wird, werden angenommen und in den nachfolgenden Schritten verwendet.

- **Input**: KPI_i^x, $C_{t,\omega}$, Glaubwürdigkeitsaussage für das jeweilige Cluster
- **Output**: KPI_i^x

11. XiL-Ergebnisse für Cluster verwerfen Für Cluster, die als nicht glaubwürdig klassifiziert werden, findet die Wiederholung der Schritte 5, 7, 8 und 9 statt. Es wird dabei für die folgende Iteration die effizienteste verbleibende XiL-Umgebung (siehe Gl. 4.10) gewählt. Ist keine XiL-Umgebung mehr vorhanden und für die vorhandenen XiL-Umgebungen keine Glaubwürdigkeit gegeben, können die XiL-Testergebnisse nicht bei den nachfolgenden Schritten verwendet werden. Anzumerken ist, dass Cluster sich zwischen Iterationen ändern können, wenn sich Testdaten und -ergebnisse unterscheiden.

- **Input**: $C_{t,\omega}$, $K_{\omega,j}$, Glaubwürdigkeitsaussage für das jeweilige Cluster
- **Output**: Szenario-Dateien im Zielformat der Simulationsumgebung oder Szenariobeschreibungen für Realtests

12. Realtests durchführen Für den Fall, dass die Glaubwürdigkeit eines Clusters in keiner XiL-Umgebung bestätigt werden kann, ist es notwendig, Realtests der jeweiligen konkreten Szenarien auf einem Prüfgelände durchzuführen. Können diese aus Gründen der Sicherheit oder ökonomischer Aspekte nicht stattfinden, bleibt die Leistungsfähigkeit des Systems in diesen Szenarien unbekannt. Auf die Entscheidungsfindung, ob Realtests durchgeführt werden, soll an dieser Stelle nicht weiter eingegangen werden.

- **Input**: Szenariobeschreibungen für Realtests
- **Output**: Realtest-KPIs individuell für jedes konkrete Szenario (KPI_i^r)

13. Ergebnisse aggregieren Aus allen angenommenen XiL- und Realtest-Ergebnissen wird eine gesamtheitliche Bewertung des SUT vorgenommen. Die Aggregation dient als Grundlage für eine Sicherheitsargumentation. Ziel ist es, Parameterbereiche zu identifizieren, in denen das SUT eine kritische Leistungsfähigkeit aufweist.

- **Input**: KPI_i^x, KPI_i^r
- **Output**: SUT-Bewertung gesamtheitlich über die konkreten Szenarien aller Cluster

5 Glaubwürdigkeitsbewertung für den szenariobasierten XiL-Test

5.1 Vorgehen und Einordnung in den Prozessentwurf

Aufbauend auf die in 3.3.1 eingeführten Definitionen wird in diesem Abschnitt eine Methode vorgestellt, die es ermöglicht, die Glaubwürdigkeit einer XiL-Umgebung für den Test von Fahrerassistenzsystemen zu bewerten. Die Methode detailliert Schritt 9 des in Abschnitt 4.3 vorgestellten Prozesses für den Software-Qualifizierungstest (siehe Abb. 5.1). Somit bezieht sich die Argumentation jeweils immer auf ein konkretes Szenario. Voraussetzung ist, dass durch einen geeigneten Clusterprozess je ein repräsentatives konkretes Szenario pro Cluster identifiziert wurde (siehe Schritt 8 in 4.3.2). Darüber hinaus ist die hier vorgestellte Glaubwürdigkeitsbewertung grundsätzlich auf andere Anwendungsfälle im Kontext des szenariobasierten XiL-Test übertragbar.

Für die Glaubwürdigkeitsbewertung wird davon ausgegangen, dass in der XiL-Umgebung geeignete konzeptionelle Modelle verwendet werden, die in jeweils ausreichender Genauigkeit in Simulationsmodellen umgesetzt sind. Annahme ist, dass in Schritt 4 des Referenzprozesses aus Abb. 4.6 eine Qualifizierung und Verifikation der verwendeten Simulationsmodelle erfolgreich stattgefunden hat (siehe Abb. 5.2). Somit ist der Fokus dieses Kapitels die Frage der Validierung.

Ziel ist es, für jeweils ein konkretes Szenario als Repräsentant eines Clusters die Glaubwürdigkeit der XiL-Umgebung zu prüfen (siehe 4.3.2). Da XiL-Szenarien aufgrund der fehlenden Kontrolle über andere Objekte nur sehr eingeschränkt in einem Feldtest reproduzierbar sind, dienen Tests auf einem Prüfgelände als Referenz. Wenn in den folgenden Abschnitten und Kapiteln von einem Realtest die Rede ist, so bezieht sich dies auf Tests auf einem Prüfgelände.

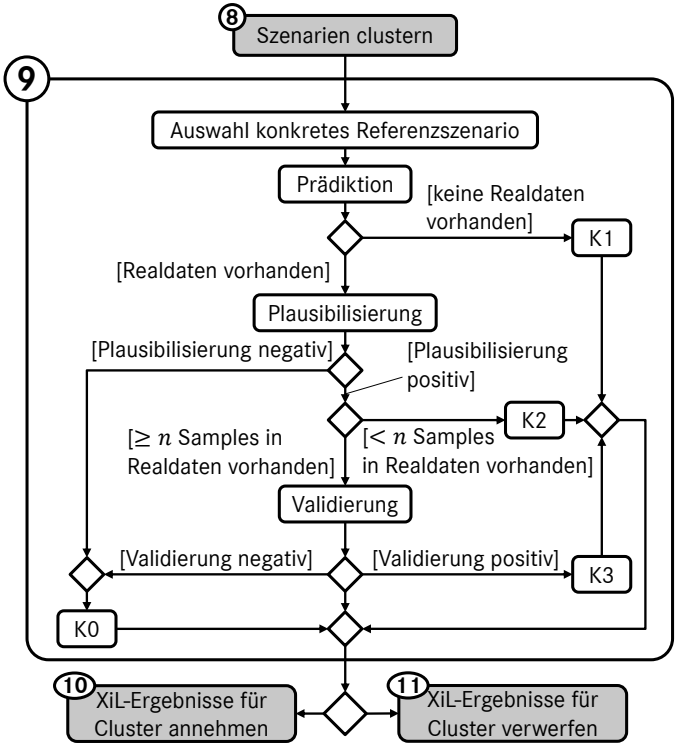

Abbildung 5.1: Vorgehen bei der Glaubwürdigkeitsbewertung

Grundprinzip von XiL- und Realtests ist es, basierend auf quantitativen Größen (SRQs, TDMs, FBMs) eine qualitative Aussage (PFCs) über das SUT abzuleiten. Für die Glaubwürdigkeitsbewertung ist daher die Genauigkeit der XiL-Umgebung hinsichtlich PFCs ein entscheidender Aspekt. Bereits Realtests weisen aleatorische und epistemische Unsicherheiten auf (siehe 5.3.1), die je nach Szenario Auswirkungen auf PFCs haben können. Eine valide XiL-Umgebung sollte diese Unsicherheiten im Idealfall möglichst präzise prädizieren bzw. reproduzieren können. Der hier gewählte Ansatz nutzt dafür ein Monte-Carlo (MC)-Sampling der Unsicherheit einzelner Sub-Simulationsmodelle und wird im folgenden als **Prädiktion** bezeichnet. Wiederholungen eines identischen

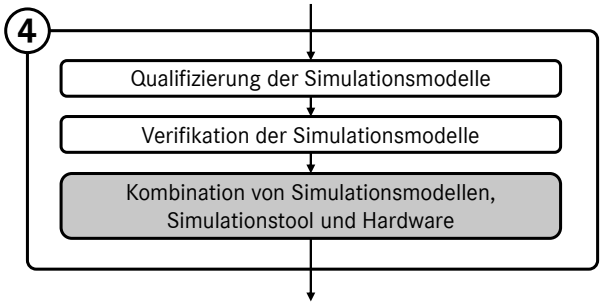

Abbildung 5.2: Qualifizierung und Verifikation der Simulationsmodelle in Prozessschritt 4

konkreten Szenarios für die Prädiktion werden im folgenden als MC-Samples referenziert.

Die **Validierung** der XiL-Umgebung setzt die Quantifizierung der Unsicherheit des Realtests hinsichtlich des Testergebnisses voraus. Dies erfordert die wiederholte Durchführung identischer Testfälle im Realtest, welche im folgenden als R-Samples bezeichnet werden. Anschließend wird die Validierung durchgeführt, welche die Ergebnisse der Prädiktion mit der Unsicherheit des Realtests vergleicht. Daneben wird die sogenannte **Plausibilisierung** als Vorstufe zur Validierung eingeführt, um unabhängig von der Anzahl der R-Samples die Äquivalenz eines konkreten Szenarios zwischen XiL-Test und Realtest zu prüfen. Die Glaubwürdigkeitsbewertung umfasst somit die drei Schritte Prädiktion, Plausibilisierung und Validierung.

5.2 Konfidenzstufen

Aus ökonomischen und organisatorischen Gründen kann nicht für jedes konkrete Szenario die für Validierung erforderliche Anzahl an R-Samples generiert werden. Die vorhandenen Realtestkapazitäten werden daher priorisiert, indem

ähnlich der in ISO 26262 definierten ASILs vier verschiedene Konfidenzstufen definiert werden (siehe Abb. 5.3):

- **K0**: Nicht glaubwürdig
- **K1**: Unbekannt
- **K2**: Plausibel
- **K3**: Valide

Schwere	Auftretens-wahrscheinlichkeit	Kontrollierbarkeit		
		C1	C2	C3
S1	E1	K1	K1	K1
	E2	K1	K1	K1
	E3	K1	K1	K2
	E4	K1	K2	K3
S2	E1	K1	K1	K1
	E2	K1	K1	K2
	E3	K1	K2	K3
	E4	K2	K3	K3
S3	E1	K1	K1	K2
	E2	K1	K2	K3
	E3	K2	K3	K3
	E4	K3	K3	K3

Abbildung 5.3: Erforderliche minimale Konfidenzstufe in Anlehnung an ISO 26262 [15]

K3 stellt die höchste Konfidenzstufe dar und K0 die niedrigste. Ein Cluster ist in einer XiL-Umgebung glaubwürdig dann, wenn mindestens die entsprechende minimal erforderliche Konfidenzstufe (K_{min}) erreicht wird (siehe Gl. 4.12). Wird eine niedrigere Konfidenzstufe erreicht, so ist das Cluster in der XiL-Umgebung als nicht glaubwürdig zu klassifizieren. Um die für das jeweilige

Cluster erforderliche Konfidenzstufe festzulegen, wird ein Vorgehen in Anlehnung der Ermittlung der ASIL in der ISO 26262 (siehe 2.3.1) gewählt. Die Einstufung hinsichtlich Schwere der Verletzungen, Eintrittswahrscheinlichkeit der Fahrsituation und Kontrollierbarkeit werden jeweils für das gesamte Cluster vorgenommen. K0 führt in jedem Fall zu einem nicht glaubwürdigen Cluster und ist daher in Abb. 5.3 nicht aufgeführt.

Die **Schwere der Verletzungen** einer beteiligten Person wird analog ISO 26262 klassifiziert wie folgt [15]:

- **S0**: Keine Verletzungen
- **S1**: Leichte bis moderate Verletzungen
- **S2**: Schwere und lebensgefährliche Verletzungen (Überleben wahrscheinlich)
- **S3**: Lebensgefährliche Verletzungen (Überleben unsicher), tödliche Verletzungen

Die Klasse der Eintrittswahrscheinlichkeit wird basierend auf der Häufigkeit ihres Eintretens (Methode 2 nach ISO 26262) ermittelt [15], da konkrete Szenarien unabhängig von ihrer Dauer bewertet werden. Es ergibt sich die folgende Abschätzung für das mittlere Zeitintervall $t_I(C_j)$ zwischen dem zweifachen Eintreten eines beliebigen konkreten Szenarios $S_{i,j}$ des Clusters C_j aus dem logischen Szenario S^l:

$$t_I(C_j) = t_I(S^l) \cdot \frac{1}{p(C_j)} \tag{5.1}$$

mit der Wahrscheinlichkeit für das Auftreten der Szenarien aus Cluster C_j:

$$p(C_j) = \sum_{i=1}^{m} p(S_{i,j}) \tag{5.2}$$

Hierbei stellt $t_I(S^l)$ das mittlerem Zeitintervall zwischen dem zweifachen Eintreten von S^l dar und $p(S_{i,j})$ repräsentiert die Wahrscheinlichkeit der Parameterkombination von $S_{i,j}$ innerhalb S^l. Die Auswahl der Klasse der **Eintrittswahrscheinlichkeit** wird anschließend aus $t_I(C_j)$ abgeleitet [15]:

- **E1**: Weniger als einmal pro Jahr
- **E2**: Wenige Male pro Jahr
- **E3**: Mindestens einmal pro Monat
- **E4**: Fast auf jeder Fahrt

Für die **Kontrollierbarkeit** wird bei Systemen ab SAE Level 3 die Stufe C3 (schwer kontrollierbar/ unkontrollierbar) angenommen, da der menschliche Fahrer als Rückfallebene entfällt. Ansonsten gilt wie folgt [15]:

- **C0**: Allgemein kontrollierbar
- **C1**: Einfach kontrollierbar
- **C2**: Normalerweise kontrollierbar
- **C3**: Schwer kontrollierbar, unkontrollierbar

Die Ableitung von K_{min} für ein Cluster ergibt sich analog ISO 26262 aus Abb. 5.3. Das Vorgehen zum Erreichen der jeweiligen Konfidenzstufe ist in Abb. 5.1 dargestellt. Aus K_{min} ergibt sich der Bedarf an R-Samples für das repräsentative konkrete Szenario des jeweiligen Clusters:

K1: Es wird nur die Prädiktion durchgeführt und somit keine R-Samples benötigt. Die dabei verwendeten Unsicherheitsmodelle (siehe 5.3.1) sind auf Basis anderer Realtests parametriert. Ein direkter Abgleich zwischen Realtest und XiL-Umgebung für dieses konkrete Szenario findet nicht statt.

K2: Es werden Prädiktion und Plausibilisierung durchgeführt, sodass zwischen 1 und $n^r_{min} - 1$ R-Samples benötigt werden. Der Abgleich zwischen Realtest und XiL-Umgebung ermöglicht noch keine statistisch signifikanten Aussagen.

K3: Es werden Prädiktion, Plausibilisierung und Validierung durchgeführt, wodurch mindestens n^r_{min} R-Samples erforderlich sind. Damit sind statistisch belastbare Aussagen zur Prädiktion der XiL-Umgebung möglich.

Der Parameter n^r_{min} ist so zu wählen, dass die in der Validierung generierten Aussagen eine statistische Signifikanz aufweisen (siehe Abschnitt 5.5). Die Konfidenzstufe **K0** wird dann erreicht, wenn Plausibilisierung oder Validierung mit negativem Ergebnis durchgeführt wurden. Die Definition der Konfidenzstufen erfolgt unabhängig von der in [223] verwendeten Konfidenz (siehe auch 3.1.4.7).

5.3 Prädiktion

Ziel der Prädiktion ist es, die Wahrscheinlichkeit möglicher Testergebnisse T_i des Realtests für ein konkretes Szenario S_j abzuschätzen. Hierfür kommen Unsicherheitsmodelle für die XiL-Umgebung zum Einsatz. Anschließend wird über einen Monte-Carlo-Ansatz und eine Manipulation die Unsicherheit eines konkreten Szenarios abgeschätzt.

5.3.1 Repräsentation von Unsicherheiten in Realtest und XiL-Test

Unsicherheiten werden zunächst als unabhängige Zufallsvariablen modelliert. Aus Gl. 3.17 ergibt sich folgende Ungleichung für die gesamte Unsicherheit zwischen XiL-Umgebung (U^x) und Realtest (U^r):

$$U^{ges} \leq U^x + U^r \tag{5.3}$$

Messungen aus Realtests beinhalten aleatorische Unsicherheiten insbesondere aufgrund von parallelen Prozessen und Kommunikation des SUT sowie stochastischen Einflüssen bei der Testdurchführung auf Fahrdynamik, Sensorik, Fahrer oder Umweltbedingungen. Daneben existiert eine epistemische Unsicherheit durch externe Einflüsse und der möglichen Diskrepanz zwischen Szenariodefinition und -durchführung. [313] Auch die Unsicherheit durch Messfehler kann aleatorisch und epistemisch sein.

Die gesamte Unsicherheit des Realtests als Summe zweier Zufallsvariablen ergibt sich mithilfe der Faltung wie folgt:

$$U^r = U_a^r * U_e^r \tag{5.4}$$

XiL-Tests hingegen weisen primär epistemische Unsicherheiten aufgrund der durch die Simulationsmodelle vereinfachten Repräsentation der realen Welt auf. Ebenso kann die Repräsentation des SUT Unsicherheiten hervorrufen. Aleatorische Unsicherheiten existieren dann, wenn die XiL-Umgebung z.B. aufgrund asynchroner Prozesse oder Kommunikation nichtdeterministisches Verhalten aufweist. [313] Dies soll jedoch im Rahmen dieser Dissertation nicht weiter betrachtet werden. Es ergibt sich somit für die XiL-Umgebung:

$$U^x = U_e^x \tag{5.5}$$

Somit gilt für die gesamte Unsicherheit:

$$U^{ges} = U^x * U^r = U_e^x * (U_a^r * U_e^r) \tag{5.6}$$

Die Verteilungsfunktionen der Unsicherheiten sind nicht a priori bekannt. Da die Unsicherheiten auf einer Überlagerung mehrerer unabhängiger Effekte basieren, kann nach dem zentralen Grenzwertsatz als Annäherung auf eine Normalverteilung zurückgegriffen werden [169]. Darüber hinaus lassen sich bei einer Normalverteilung bewährte statistische Verfahren nutzen, die auch in anderen Feldern Anwendung finden. Es gilt somit:

$$\mu^{ges} = \mu_e^x + \mu_a^r + \mu_e^r \tag{5.7}$$

$$\sigma_{ges}^2 = \sigma_{e,x}^2 + \sigma_{a,r}^2 + \sigma_{e,r}^2 \tag{5.8}$$

$$U^{ges} \sim \mathcal{N}(\mu_e^x + \mu_a^r + \mu_e^r, \sigma_{e,x}^2 + \sigma_{a,r}^2 + \sigma_{e,r}^2) = \mathcal{N}(\mu^{ges}, \sigma_{ges}^2) \tag{5.9}$$

Die normalverteilte Zufallsvariable U^{ges} mit den Parametern μ^{ges} und σ_{ges} kombiniert somit alle Unsicherheiten, die zu Fehlern zwischen XiL-Umgebung und gemessenem Realtest führen.

5.3.2 Unsicherheitsmodelle für XiL-Umgebungen

Ziel der Prädiktion ist es, mithilfe von XiL-Tests und entsprechenden Unsicherheitsmodellen, die Wahrscheinlichkeit von Testergebnissen im Realtest abzuschätzen. Jedoch beschreiben die Ansätze aus 5.3.1 die Unsicherheit zwischen XiL und Realtest lediglich in abstrahierter Form. Es werden die folgenden Randbedingungen definiert, um den Modellierungsaufwand zu reduzieren:

- Es wird angenommen, dass Szenariosteuerung und Umwelt keine Unsicherheit aufweisen und somit deterministisch sind.
- Lokalisierung und HD Maps sind nicht Teil der XiL-Umgebung und werden somit nicht modelliert.
- Unsicherheiten für Fahrzeug und weitere ECUs werden je in einem eigenen Modell erfasst (Unsicherheits-Subsystem 1, siehe Abb. 5.4).
- Unsicherheiten für Sensoren werden in einem eigenen Modell erfasst (Unsicherheits-Subsystem 2, siehe Abb. 5.4).
- Der Fahrer ist in der XiL-Umgebung modelliert, da die Prädiktion für FAS angewendet wird (siehe Kapitel 6). Unsicherheiten des Fahrers werden durch eine Manipulation von Führungsgrößen bzw. Regelgrößen des Fahrerreglers umgesetzt (Unsicherheits-Subsystem 3, siehe Abb. 5.4). Es findet keine Manipulation innerhalb des Fahrerreglers statt.
- Es werden nur Unsicherheiten erfasst, die über eine metrische Skala quantifizierbar sind. Andere Unsicherheiten (z.B. Objektklasse) werden nicht betrachtet.
- Alle Zufallsvariablen werden als normalverteilt modelliert (siehe 5.3.1), sofern die Unsicherheit quantifizierbar ist.
- Für die Modellierung der Unsicherheit werden in der Realität auftretende Effekte auf statistischer Basis berücksichtigt.

Basierend auf der XiL-Struktur aus 4.1.1 werden insgesamt drei Subsysteme voneinander abgegrenzt, deren Unsicherheiten separat modelliert werden (siehe Abb. 5.4). Jedes Unsicherheits-Subsystem beinhaltet ein stochastisches Unsicherheitsmodell mit entsprechenden Zufallsvariablen. Die Modellierung von Unsicherheit wird im folgenden auch als Manipulation bezeichnet.

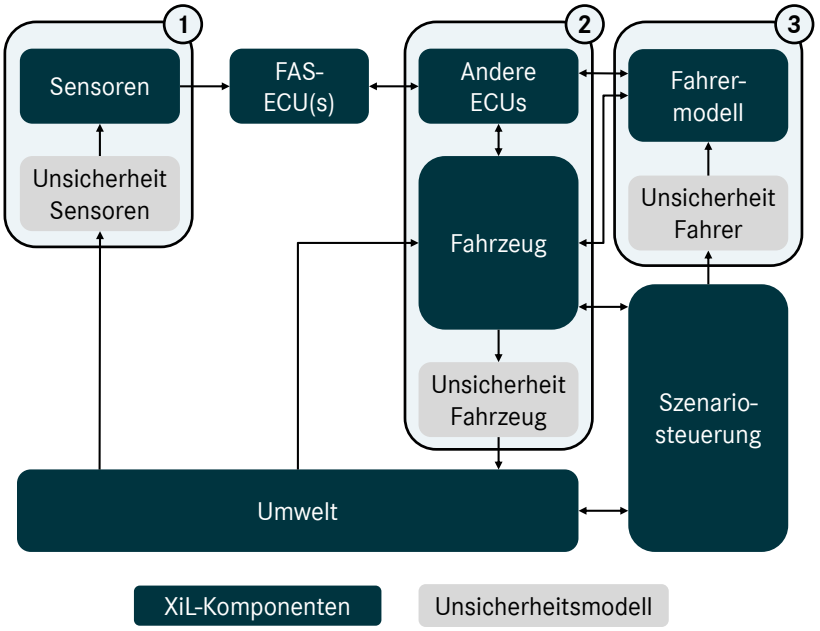

Abbildung 5.4: Unsicherheits-Subsysteme der XiL-Umgebung

5.3.2.1 Unsicherheits-Subsystem 1: Sensoren

Die Unsicherheit von Umweltsensoren lässt sich sowohl auf phänomenologischer als auch physikalischer Ebene modellieren (siehe 2.5.2.3). In diesem Fall wird die phänomenologische Methode genutzt. Hierfür werden interne Größen des Sensormodells manipuliert, was unmittelbar die in der Wirkkette der Sensormodelle folgenden Größen und somit die Ausgangsgrößen der Sensormodelle beeinflusst. Die Manipulation kann sowohl zeitvariant als auch

zeitinvariant erfolgen. Darüber hinaus ist es möglich, zusätzliche ggf. zeitabhängige Eingänge w bei der Manipulation zu berücksichtigen. Eine zeitvariante Größe $m_i(t)$ wird mit vier Unsicherheiten beaufschlagt:

- Skalierungsunsicherheit $S_{err,i}(w)$
- Offsetunsicherheit $O_{err,i}(w)$
- Rauschunsicherheit $R_{err,i}(t, w)$
- Periodische Unsicherheit $P_{err,i}(t, w)$

$$m_i(t, w) = m_{i,pre}(t) \cdot S_{err,i}(w) + O_{err,i}(w) + R_{err,i}(t, w) + P_{err,i}(t, w) \tag{5.10}$$

Für die zeitvarianten Unsicherheiten gilt:

$$R_{err,i}(t, w) \sim \mathcal{GP}(0, \Sigma_{i,r}(w)^2 \cdot I(t)) \tag{5.11}$$

$$P_{err,i}(t, w) = A_i(w) \cdot \sin(\Omega_i(w)t) \tag{5.12}$$

Die Größe $m_{pre,i}$ repräsentiert die jeweilige Größe des Sensormodells vor der Manipulation, während m_i das Resultat der Manipulation darstellt. $R_{err,i}(t, w)$ stellt ein zeitlich veränderliches weißes Rauschen dar, das über einen Gauß-Prozess $\mathcal{GP}$ formalisiert wird. $I(t)$ repräsentiert die Einheitsmatrix mit Kantenlänge des Zeitvektors, während $\Sigma_{i,r}(w)$ die Standardabweichung des weißen Rauschens darstellt. Ebenfalls zeitvariant ist die periodische Unsicherheit $P_{err,i}(t)$, die über eine Sinusfunktion mit Amplitude $A_i(w)$ und Kreisfrequenz $\Omega_i(w)$ modelliert wird. Für die Größe g kommt eine lineare Interpolation mit den Eingangsgrößen w und den Stützstellen $[\hat{w}^1, ..., \hat{w}^n]$ zum Einsatz:

$$f_{int}(w) = f_{int}(w, g(\hat{w}^1), ..., g(\hat{w}^n)) \tag{5.13}$$

Vor jeder Testfallausführung werden die Funktionswerte für $\hat{w}^1$ bis $\hat{w}^n$ neu aus einer Normalverteilung generiert:

$$g(\hat{w}^j) \sim \mathcal{N}(\mu(\hat{w}^j), \sigma(\hat{w}^j)^2) \tag{5.14}$$

Das in Gl. 5.13 und Gl. 5.14 beschriebene Vorgehen wird für folgende Größen bzw. Parameter angewandt:

- $S_{err,i}(w)$
- $O_{err,i}(w)$
- $\Sigma_{i,r}(w)$
- $A_i(w)$
- $\Omega_i(w)$

Es ergibt sich aus den über die Interpolation errechneten Größen bzw. Parameter sowie den zeitvarianten Größen folgender Fehler:

$$m_i(t) = m_{pre,i}(t) \cdot s_{err,i} + o_{err,i} + r_{err,i}(t) + p_{err,i}(t) \tag{5.15}$$

Für zeitinvariante zu manipulierende Größen m_i entfallen die zeitvarianten Terme $r_{err,i}(t)$ und $p_{err,i}(t)$:

$$m_i = m_{pre,i} \cdot s_{err,i} + o_{err,i} \tag{5.16}$$

Basierend auf Expertenwissen, Sensordatenblättern und Realdatenaufzeichnungen wird für die folgenden Größen eine Manipulation konzeptioniert (siehe Abb. 5.5 und Abb. 5.6):

1. Skalierung des longitudinalen/lateralen Sichtfeldes für Objekte ($m_1 = x/y_{fov}$)
2. Relative Objektdistanz longitudinal ($m_2 = dx_q$)
3. Relative Objektdistanz lateral ($m_3 = dy_q$)
4. Relative Objektgeschwindigkeit longitudinal ($m_4 = dv_{q,x}$)
5. Relative Objektgeschwindigkeit lateral ($m_5 = dv_{q,y}$)
6. Verdrehung Objekte ($m_6 = \psi_q$)
7. Spurkrümmung ($m_7 = \kappa_{lane}$)
8. Spurkrümmungsänderung ($m_8 = \dot{\kappa}_{lane}$)
9. Spurdistanz lateral ($m_9 = dy_{lane}$)
10. Spurgeschwindigkeit lateral ($m_{10} = dv_{lane,y}$)
11. Spurgierwinkel ($m_{11} = \psi_{lane}$)

Die Manipulation der Größe m_1 findet zeitinvariant statt, für m_2 bis m_{11} ist dies zeitvariant.

5.3.2.2 Unsicherheits-Subsystem 2: Fahrzeug

Das Unsicherheits-Subsystem Fahrzeug beinhaltet sowohl Modelle für die Fahrdynamik als auch Fahrzeugkomponenten wie z.B. Antriebsstrang oder Lenksystem. Zudem werden die Modelle anderer ECUs berücksichtigt. Das Vorhandensein von Simulationsmodellen wie auch deren Modellierungstiefe können sich je nach XiL-Umgebung unterscheiden. Ausgangsgröße ist der fahrdynamische Zustand des Fahrzeugs, ausgedrückt durch die kinematische Größen Beschleunigung, Geschwindigkeit, Position, Gierwinkel und Gierrate. Es wird ein zusätzliches dem Fahrzeugmodell nachgeschaltetes Unsicherheitsmodell (siehe Abb. 5.8) eingeführt, welches eine Manipulation dieser Aus-

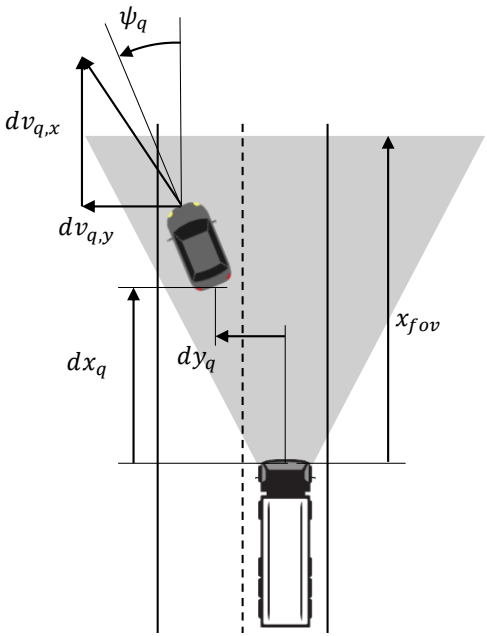

Abbildung 5.5: Skizze zur Manipulation der Sensor-Objektinformationen

gangsgrößen ermöglicht. Das Unsicherheitsmodell ist dabei dreistufig (siehe Abb. 5.7): Zunächst wird in Modul 1 die Unsicherheit des Fahrzeugmodells für die jeweilige Fahrsituation prädiziert [318]. Anschließend werden in Modul 2 daraus zufällige Fehler generiert. Diese Fehler werden in Modul 3 mit den ursprünglichen Ausgangsgrößen des Fahrzeugmodells aggregiert.

Hierfür werden zwei Koordinatensysteme entsprechend Abb. 5.9 definiert:

- Ego-Koordinatensystem: Feststehendes, aber mit der Längsachse des Ego-Fahrzeugs mitdrehendes Koordinatensystem. Index lon bzw. lat.
- Inertialkoordinatensystem: Feststehendes und nicht mitdrehendes Koordinatensystem. Index x bzw. y.

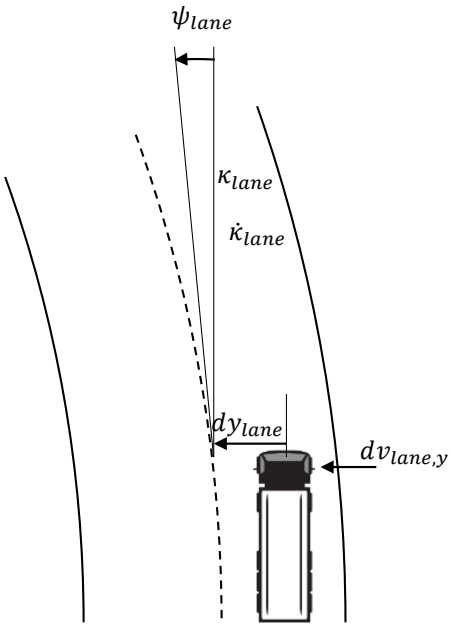

Abbildung 5.6: Skizze zur Manipulation der Sensor-Spurinformationen

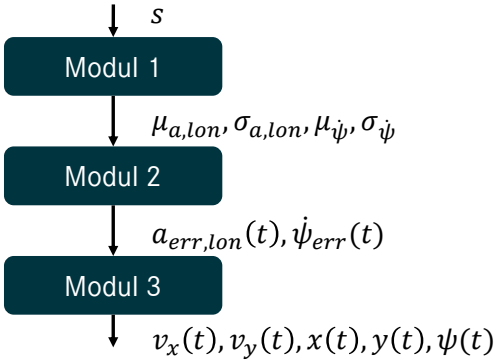

Abbildung 5.7: Module für die Manipulation des Fahrzeugs

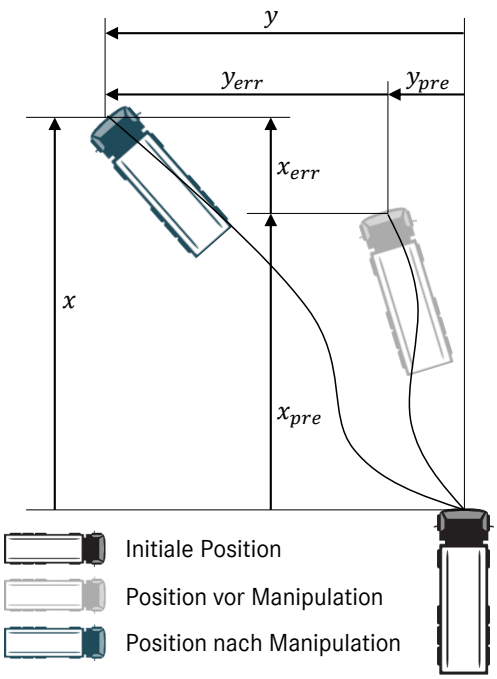

Abbildung 5.8: Skizze zur Manipulation des Fahrzeugs

Die im Ego-Fahrzeug gemessene Geschwindigkeiten und Beschleunigungen werden im Ego-Koordinatensystem aufgezeichnet, während Gierwinkel, Gierrate, Geschwindigkeitsfehler und Pfad des Ego-Fahrzeugs im Inertialkoordinatensystem beschrieben werden.

Grundlage der Manipulation sind Unsicherheiten der folgenden Größen:

- Longitudinale Beschleunigung im Ego-Koordinatensystem $a_{lon}(t)$
- Gierrate im Inertialkoordinatensystem $\dot{\psi}(t)$

Die Unsicherheit beider Größen wird als normalverteilt modelliert (siehe 5.3.1). Modul 1 liefert mithilfe eines Regressionsmodells die entsprechen-

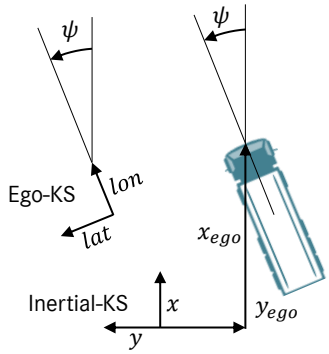

Abbildung 5.9: Koordinatensysteme für das Ego-Fahrzeug

den Parameter in Abhängigkeit von der Fahrsituation, die durch die Inputs $s = [s_1(t), ..., s_j(t)]$ repräsentiert wird:

$$\begin{bmatrix} \mu_{a,lon}(s) \\ \sigma_{a,lon}(s) \\ \mu_{\dot{\psi}}(s) \\ \sigma_{\dot{\psi}}(s) \end{bmatrix} = f_{reg}(s) \tag{5.17}$$

Es wird angenommen, dass bereits ein geeignetes Regressionsmodell mit entsprechender Selektion der Inputs s zur Verfügung steht, sodass hierauf nicht weiter eingegangen wird. [318] Basierend auf dem Regressionsmodell werden die folgenden Zufallsvariablen als Repräsentation der Unsicherheit basierend auf einem Gauß-Prozess definiert:

$$A_{err,lon}(t,s) \sim \mathcal{GP}(\mu_{a,lon}(s), \sigma^2_{a,lon}(s) \cdot C(t)) \tag{5.18}$$

$$\dot{\Psi}_{err}(t,s) \sim \mathcal{GP}(\mu_{\dot{\psi}}(s), \sigma^2_{\dot{\psi}}(s) \cdot C(t)) \tag{5.19}$$

Die verwendete Kovarianzmatrix $C(t)$ ist über die eine quadratisch exponentielle Kovarianzfunktion $k(t,t')$ definiert bzw. leitet sich daraus ab:

$$k(t,t') = e^{r\cdot(t-t')^2} \tag{5.20}$$

Grund für die Wahl einer quadratisch exponentiellen Funktion für $k(t,t')$ ist, dass die Autokorrelation einer durch den Gauß-Prozess generierten Größe zwischen zwei Zeitpunkten sinkt, wenn diese zeitlich weiter auseinander liegen. Für jeden Zeitschritt werden in Modul 2 die Fehler $a_{err,lon}(t)$ und $\dot{\psi}_{err}(t)$ generiert. In Modul 3 werden daraus weitere davon abhängige Fehler unter Anwendung kinematischer Beziehungen abgeleitet. Für den Fehler $\psi_{err}(t)$ und den daraus resultierenden manipulierten Gierwinkel $\psi(t)$ gilt:

$$\psi_{err}(t) = \int_0^t \dot{\psi}_{err}(\hat{t})d\hat{t} \tag{5.21}$$

$$\psi(t) = \psi_{pre}(t) + \psi_{err}(t) \tag{5.22}$$

Der Index „pre“ markiert die jeweiligen Größen vor der Manipulation und somit die ursprünglichen Ausgangsgrößen des Fahrzeugs. Geschwindigkeitsfehler und resultierende manipulierte Geschwindigkeit im Inertialkoordinatensystem berechnen sich wie folgt:

$$a_{lon}(t) = a_{pre,lon}(t) + a_{err,lon}(t) \tag{5.23}$$

$$v_{err,lon}(t) = \int_0^t a_{err,lon}(\hat{t})d\hat{t} \tag{5.24}$$

$$v_{err,x}(t) = \cos(\psi(t)) \cdot v_{err,lon}(t) \tag{5.25}$$

$$v_{err,y}(t) = \sin(\psi(t)) \cdot v_{err,lon}(t) \tag{5.26}$$

$$v_x(t) = v_{pre,x}(t) + v_{err,x}(t) \tag{5.27}$$

$$v_y(t) = v_{pre,y}(t) + v_{err,y}(t) \tag{5.28}$$

Für den Positionsfehler und die resultierende Position im Inertialkoordinatensystem gilt:

$$x_{err}(t) = \int_0^t v_{err,x}(\hat{t})d\hat{t} \tag{5.29}$$

$$y_{err}(t) = \int_0^t v_{err,y}(\hat{t})d\hat{t} \tag{5.30}$$

$$x(t) = x_{pre}(t) + x_{err}(t) \tag{5.31}$$

$$y(t) = y_{pre}(t) + y_{err}(t) \tag{5.32}$$

5.3.2.3 Unsicherheits-Subsystem 3: Fahrer

Für die Modellierung der Unsicherheit des Fahrers wird keine direkte Manipulation innerhalb des Reglers vorgenommen [318]. Stattdessen wird je eine Führungs- bzw. Regelgröße manipuliert. Die Unsicherheit in der Längsregeltätigkeit des Fahrers wird durch Variation der Sollgeschwindigkeit v_{soll} repräsentiert:

$$v_{soll} = \mu(v_{soll,pre}) + \sigma(v_{soll,pre}) \cdot z \tag{5.33}$$

Der Faktor z wird zu Beginn der Testfalldurchführung einmalig aus einer Standardnormalverteilung generiert:

$$z \sim \mathcal{N}(0,1) \tag{5.34}$$

Die Parameter $\mu(v_{soll,pre})$ und $\sigma(v_{soll,pre})$ werden in Abhängigkeit von $v_{soll,pre}$ über lineare Interpolation bestimmt.

Für Szenarien mit einem Einlenkmanöver existiert eine Unsicherheit hinsichtlich der longitudinalen Position x_{ego} des Einlenkmanövers. Die longitudinale

Position stellt eine Regelgröße dar und wird mit einem Offset $x_{ego,of}$ beaufschlagt:

$$x_{ego} = x_{ego,pre} + x_{ego,of} \tag{5.35}$$

Die Größe $x_{ego,of}$ wird zu Beginn der Testfalldurchführung aus einer Normalverteilung generiert. Die entsprechenden Parameter μ und σ werden ebenfalls über eine lineare Interpolation in Abhängigkeit der Ziel-Sollgeschwindigkeit des jeweiligen Szenarios berechnet:

$$x_{ego,of} \sim \mathcal{N}(\mu(v_{soll}), \sigma(v_{soll})) \tag{5.36}$$

5.3.3 Monte-Carlo-Ansatz für die Prädiktion[1]

Grundlage der Prädiktion ist ein Monte-Carlo-Algorithmus [291], welcher über die in 5.3.2 definierten Unsicherheitsmodelle die Unsicherheit U^{ges} zwischen XiL-Umgebung und gemessenem Realtest (siehe 5.3.1) für n^x MC-Samples (S_n^x) variiert. Voraussetzung ist die korrekte Parametrierung der Unsicherheitsmodelle basierend auf Realdaten. Jedes MC-Sample stellt einen Input für die XiL-Umgebung dar, während das Testergebnis T_n^x des MC-Samples S_n^x den entsprechenden Output repräsentiert. Dieser wird in der XiL-Umgebung ermittelt und über die XiL-Transferfunktion f_{XiL} formalisiert:

$$T_n^x = f_{XiL}(S_n^x) \tag{5.37}$$

Der XiL-Test des identischen konkreten Szenarios wird somit n^x Mal durchgeführt. Die Auftretenswahrscheinlichkeit eines Testergebnisses T_i kann mithilfe des Gesetzes der großen Zahlen abgeschätzt werden [292].

[1] Teile dieses Abschnitts wurden bereits in ähnlicher Form vom Autor in [313] publiziert.

Es ergibt sich:

$$p(T_i) \approx \frac{n_i^x}{n^x} \tag{5.38}$$

mit

$$n_i^x = \sum_{n=1}^{n^x} \delta(T_n^x, T_i) \tag{5.39}$$

$$\delta(A, B) = \begin{cases} 1, & A = B \\ 0, & A \neq B \end{cases} \tag{5.40}$$

$p(T_i)$ kann entsprechend Gl. 5.38 für jedes auftretende T_i abgeschätzt werden. Nachteil dieser Abschätzung ist, dass die Unsicherheit aufgrund der Anzahl der MC-Samples (n^x) nicht berücksichtigt wird. Hierfür eignen sich Konfidenzintervalle. Im allgemeinen Fall bzw. bei mehr als zwei auftretenden Testergebnissen liegt eine Multinomialverteilung vor [169]. Mithilfe des Ansatzes nach Goodman [293] können für die Auftretenswahrscheinlichkeiten der einzelnen Testergebnisse Konfidenzintervalle abgeschätzt werden:

$$CI_{i,\omega,j} = [p_{min}(T_i), p_{max}(T_i)] = f_{CI}(n_i^x, n^x) \tag{5.41}$$

Konfidenzintervalle werden jeweils für XiL-Umgebung ω und Cluster j berechnet. Als Konfidenzniveau wird $1 - \alpha = 95\%$ festgelegt.

Für den Parameter n^x ergibt sich ein Zielkonflikt zwischen der statistischen Aussagekraft der Wahrscheinlichkeitsschätzung einerseits und der erforderlichen Rechenzeit andererseits. Zudem ist die Verteilung der Testergebnisse nicht a-priori bekannt. Um dennoch vor dem Start der XiL-Tests eine minimale Zahl für n^x zu definieren, wird eine Worst-Case-Abschätzung vorgenommen. Anforderung soll sein, dass das Konfidenzintervall eines Testergebnisses eine Breite von maximal 0.1 haben darf bzw. der wahre Funktionswert mit einem Konfi-

denzniveau von $1 - \alpha = 95\%$ maximal um 0.05 von $p_{min}(T_i)$ oder $p_{max}(T_i)$ abweichen darf. Für eine Multinomialverteilung ergibt dies nach [294] eine Anzahl von mindestens $n^x_{min} = 510$ Samples.

5.4 Plausibilisierung[2]

Dieser Schritt setzt die vorherige Durchführung der Prädiktion voraus. In der Plausibilisierung findet ein Abgleich zwischen je einem einzelnen R-Sample und allen MC-Samples statt. Ziel ist zu zeigen, dass mindestens ein MC-Sample (S^x_n) äquivalent zum R-Sample (S^r_m) ist und die XiL-Umgebung somit plausible Ergebnisse liefert. Es wird dabei ebenfalls nur ein einziges konkretes Szenario S_j betrachtet. Für die Äquivalenz zweier Samples n und m werden die folgenden Kriterien festgelegt:

1. E_1: Das Testergebnis von XiL-Umgebung und Realtest ist identisch.

$$E_1(n,m) = \begin{cases} 1, & T^x_n = T^r_m \\ 0, & sonst \end{cases} \tag{5.42}$$

2. E_2: Der Testablauf, der im Testergebnis resultiert, ist äquivalent. Dieses Kriterium wird über eine Kombination von k Szenariodistanzmaßen d_k (siehe 5.4.1) geprüft. Äquivalenz ist dann gegeben, wenn gilt:

$$E_2(n,m) = \begin{cases} 1, & d_k(S^x_n, S^r_m) < d_{k,max} \forall k \\ 0, & sonst \end{cases} \tag{5.43}$$

   Als $d_{k,max}$ wird hier ein für das jeweilige Szenariodistanzmaß gültiger Grenzwert bezeichnet, welcher einen zu kalibrierenden Parameter darstellt.

[2] Teile dieses Abschnitts wurden bereits in ähnlicher Form vom Autor in [312] publiziert.

Eine Äquivalenz ($E = 1$) zweier Samples ist dann gegeben, wenn beide Kriterien erfüllt sind:

$$E(n,m) = \begin{cases} 1, & E_1(n,m) \wedge E_2(n,m) \\ 0, & sonst \end{cases} \tag{5.44}$$

Die Prüfung der Äquivalenz zwischen R-Samples (m) und MC-Samples (n) findet sequentiell für alle MC-Samples statt. Sobald die Äquivalenz zum R-Sample für mindestens ein MC-Sample nachgewiesen ist, ist die Plausibilisierung für XiL-Umgebung ω und das konkrete Szenario j erfolgreich beendet ($P_{\omega,j} = 1$):

$$P_{\omega,j} = \begin{cases} 1, & \exists n,m\, E(n,m) = 1 \\ 0, & sonst \end{cases} \tag{5.45}$$

Umgekehrt ist die Plausibilisierung nicht erfolgreich, wenn kein MC-Sample äquivalent zum Realtest ist.

5.4.1 Verwendete Szenariodistanzmaße

Für die Plausibilisierung werden drei Szenariodistanzmaße (siehe 3.2.3) verwendet. Jedes Szenariodistanzmaß bezieht sich auf einen qualitativen Aspekt der Übereinstimmung des Testablaufs zwischen R-Sample S^r und MC-Sample S^x. Grundlage hierbei sind die folgenden SRQs bzw. TDMs:

- Ego-Trajektorie in einem Inertialkoordinatensystem (siehe Abb. 5.9)

$$r_{ego} = [x, y] \tag{5.46}$$

- Relativtrajektorie eines Objekts q zum Ego-Fahrzeug (siehe Abb. 5.5)

$$r_q = [dx_q, dy_q] \tag{5.47}$$

Die Größen dx_q und dy_q stellen jeweils diskrete Zeitreihen dar. Für den Fall, dass mehrere Objekte existieren, wird davon ausgegangen, dass diese jeweils einander zugeordnet wurden und in beiden Szenarien den gleichen Index q haben.

Um die Zeitabhängigkeit der Trajektorien zu eliminieren, wird zur Berechnung aller drei verwendeten Szenariodistanzmaße zunächst der Dynamic Time Warping (DTW)-Algorithmus angewendet. Beim DTW werden zwei zeitabhängige Sequenzen $A = (a_1, ..., a_n)$ und $B = (b_1, ..., b_m)$ mithilfe einer Kostenfunktion miteinander verglichen. Die Indizes der Sequenzen A und B repräsentieren hierbei Zeitschritte. Als Kostenfunktion wird die Euklidische Distanz für einen Vektor mit z Elementen gewählt:

$$c(a_i, b_j) = \sqrt{\sum_{k=1}^{z} (a_{i,k} - b_{j,k})^2} \tag{5.48}$$

Anschließend findet eine Zuordnung aller Elemente von A zu mindestens einem Element von B und umgekehrt statt, sodass die akkumulierten Gesamtkosten minimal werden. Die Indizes der Elemente von A und B müssen dabei monoton steigend sein. Hieraus resultiert ein Zuordnungspfad $[a_u, b_v]$, der durch die Vektoren $u = (u_1, ..., u_p)$ und $v = (v_1, ..., v_p)$ definiert ist. [295] [296]

Bei Anwendung des hieraus ermittelten Zuordnungspfades ist es im allgemeinen Fall aufgrund von mehrfacher Zuordnung eines Elements möglich, dass p im Vergleich zu n und M signifikant größer ist. Somit würden einzelne Zeitschritte eine stärkere relative Gewichtung bekommen. Aus diesem Grund wird ein angepasster Zuordnungspfad definiert, dessen Länge sich an der Zeitreihe mit der größeren Zahl an Elementen orientiert. Es ergibt sich für den Fall $n \leq m$ der Zuordnungspfad $[a_{\hat{u}}, b_{\hat{v}}]$ mit $\hat{u} = (\hat{u}_1, ..., \hat{u}_m)$ und $\hat{v} = (1, ..., m)$. Falls einem Element aus B mehrere Elemente $u_{g,1}, ..., u_{g,k}$ aus A zugeordnet werden, so wird das letzte Element $u_{g,k}$ ausgewählt. Im Fall $m < n$ wird das Verfahren analog angewendet, wobei hier jedem Element aus A genau ein Element aus B zugeordnet wird.

Für die Elimination der Zeitabhängigkeit wird DTW auf r_{ego} bzw. r_q angewendet. Voraussetzung hierfür ist, dass beide Zeitreihen ein Koordinaten-

system verwendet, welches eine Vergleichbarkeit ermöglicht. Dies kann je nach Anwendungsfall durch eine Ausrichtung an externen Referenzpunkten aus Kartendaten oder anderen Objekten sichergestellt werden.

5.4.1.1 Szenariodistanzmaß 1

Das Szenariodistanzmaß d_1 prüft die maximale quantitative Abweichung der Relativdistanzen der Objekte zum Ego-Fahrzeug. Auch wenn Abweichungen nur von kurzer zeitlicher Dauer sind, kann dies bereits eine stark abweichende Systemreaktion (z.B. Notbremsung, Ausweichmanöver) hervorrufen. Solche ggf. kurzfristigen Abweichungen werden durch Analyse der Relativtrajektorie r_q identifiziert. DTW wird auf r_q angewendet, wobei im Falle mehrerer Objekte eine Zuordnung der Objekte notwendig ist.

Basierend auf dem DTW-Pfad ist das Maß wie folgt definiert:

$$\hat{d}_{1,q,o}(S^r, S^x) = |r^x_{q,\hat{u}_o} - r^r_{q,\hat{v}_o}| \tag{5.49}$$

$$d_1(S^r, S^x) = \max_q \max_o \hat{d}_{1,q,o}(S^r, S^x) \tag{5.50}$$

Das Szenariodistanzmaß liefert die maximale Abweichung über alle Zeitschritte und Objekte.

5.4.1.2 Szenariodistanzmaß 2

Durch die Anwendung von DTW wird die Zeitabhängigkeit der Trajektorien eliminiert. DTW wird hier auf r_{ego} angewendet. Um Rückschlüsse auf die Fahrdynamik während des Szenarios zu erhalten, analysiert das Szenariodistanzmaß d_2 die Ego-Geschwindigkeit. Anders als bei d_1 geht es hierbei nicht um die maximale Abweichung, sondern um systematische Abweichungen.

Daher wird hierbei der Mittelwert der Abweichung verwendet:

$$\hat{d}_{2,o}(S^r, S^x) = |\dot{r}^x_{ego,\hat{u}_o} - \dot{r}^r_{ego,\hat{v}_o}| \tag{5.51}$$

$$d_2(S^r, S^x) = \overline{\hat{d}_{2,o}(S^r, S^x)} \tag{5.52}$$

5.4.1.3 Szenariodistanzmaß 3

Ergänzend zu d_2 wird bei d_3 die Ego-Trajektorie in Form des Gierwinkelverlaufs analysiert. DTW wird ebenfalls auf r_{ego} angewendet. Es stehen auch systematische Abweichungen im Mittelpunkt, weshalb der Mittelwert der Abweichung über alle Zeitschritte berechnet wird:

$$\hat{d}_{3,o}(S^r, S^x) = |\psi^x_{ego,\hat{u}_o} - \psi^r_{ego,\hat{v}_o}| \tag{5.53}$$

$$d_3(S^r, S^x) = \overline{\hat{d}_{3,o}(S^r, S^x)} \tag{5.54}$$

5.4.2 Definition von Grenzwerten für Szenariodistanzmaße

Eine zentrale Fragestellung bei der Anwendung von Gl. 5.43 ist die Definition von Grenzwerten für die Prüfung von Äquivalenz zweier konkreter Szenarien. Hierfür wird die folgende Methode angewandt:

Basierend auf den Ergebnissen der Prädiktion für ein konkretes Szenario werden die Szenariodistanzmaße für jedes MC-Sample n zu allen anderen MC-Samples berechnet. Die MC-Samples werden entsprechend ihrer Testergebnisse sortiert. Anschließend wird aus den Szenariodistanzmaßen zwischen MC-Samples mit identischem Testergebnis jeweils ein einseitiges Toleranzintervall mit dem Maximum $d_{k,T_i,max}$ berechnet, sodass gilt:

$$P(F(d_{k,T_i,max}) \geq C) \geq 1 - \alpha \tag{5.55}$$

F stellt hier die kumulierte Verteilungsfunktion dar, während $1-\alpha$ die statistische Konfidenz[3] repräsentiert. Der Parameter C steht für die Coverage, also den Anteil der zu erwarteten zukünftigen Samples im Toleranzintervall. [297] Es wird $1-\alpha = 95\%$ und $C = 95\%$ gewählt und ein Toleranzintervall basierend auf einer Normalverteilung modelliert [297]. Mithilfe eines Toleranzintervalls kann basierend auf einer endlichen Anzahl von Samples die Verteilung einer zu erwartenden Population abgeschätzt werden.

Nachdem für jedes auftretende Testergebnis ein Toleranzintervall bestimmt wurde, wird anschließend das Minimum für das jeweilige Szenariodistanzmaß als Grenzwert verwendet:

$$d_{k,max} = \min_{T_i} d_{k,T_i,max} \tag{5.56}$$

5.5 Validierung

Der Schritt Validierung ist dann möglich, wenn eine Mindestanzahl (n^r_{min}) an R-Samples eines konkreten Szenarios S_j vorhanden sind. n^r_{min} wird so gewählt, dass für alle in der Prädiktion vorliegenden Testergebnisse eine positive und negative Validierung theoretisch möglich ist. Voraussetzung für die Validierung ist ebenfalls die vorherige Durchführung der Prädiktion sowie eine erfolgreiche Plausibilisierung aller n^r R-Samples.

Ziel der Validierung ist es, die Abschätzung der Häufigkeits- bzw. Wahrscheinlichkeitsverteilungen aus dem Schritt Prädiktion zu validieren. Für eine erfolgreiche Validierung der XiL-Umgebung wird die folgenden Nullhypothese definiert:

Es besteht eine statistische Unabhängigkeit zwischen den Merkmalen Testergebnis und Testumgebung (XiL-Umgebung vs. Realtest).

[3] Die statistische Konfidenz ist abzugrenzen von der Konfidenz eines Simulationsmodells bzw. einer XiL-Umgebung (siehe 3.3.1).

Vereinfacht gesagt bedeutet dies, dass es bei einer validen XiL-Umgebung keine statistisch signifikante Abweichung zwischen XiL-Umgebung und Realtest hinsichtlich der Testergebnis-Verteilungen geben darf.

Für den Nachweis der statistischen Unabhängigkeit wird der exakte Test nach Barnard (im folgenden auch als Barnard-Test bezeichnet) [298] herangezogen. Der häufig in diesem Kontext anzutreffende Chi-Quadrat-Unabhängigkeitstest ist grundsätzlich einsetzbar, jedoch ist die zu erwartenden Stichprobengröße in den meisten Fällen zu klein. Typische Realtestumfänge in aktuellen Serien-Entwicklungsprojekten bewegen sich im Rahmen von maximal zehn Wiederholungen pro konkretem Szenario. Da ein Chi-Quadrat-Unabhängigkeitstest mehr als fünf Samples pro Testergebnis benötigen würde [299], kann dies bereits bei zwei möglichen Testergebnissen nicht mehr erfüllt werden. Der ebenfalls denkbare exakte Test nach Fisher [300] erfordert eine konstante Vorgabe der Randsummenpaare der Kontingenztafel [301]. Dies kann im Allgemeinen für den vorliegenden Anwendungsfall nicht gewährleistet werden, sodass dieser Test ebenfalls nicht optimal ist.

Der Barnard-Test wird für jedes in der XiL-Umgebung oder im Realtest auftretende Testergebnis separat durchgeführt. Maßgebliche Eingangsgröße des Barnard-Tests ist eine Kontingenztafel, welche die Häufigkeiten der Merkmalskombinationen enthält. Für den Anwendungsfall der Validierung ergibt sich die Kontingenztafel bei n^x MC-Samples und n^r R-Samples wie folgt:

$$G(T_i) = \begin{pmatrix} n_i^r & n_i^x \\ n^r - n_i^r & n^x - n_i^x \end{pmatrix} \tag{5.57}$$

mit

$$n_i = \sum_{n=1}^{k} \delta(T_n, T_i) \tag{5.58}$$

Die Validierung für Testergebnis T_i ist erfolgreich, wenn die Nullhypothese H_0 bei einer Alternativhypothese H_1 für ein Signifikanzniveau α abgelehnt wird:

$$V(T_i) = \begin{cases} 0, & p_{T_i} \leq \alpha \\ 1, & p_{T_i} > \alpha \end{cases} \tag{5.59}$$

mit dem p-Wert p_{T_i}. Der p-Wert entspricht der Wahrscheinlichkeit, dass die betrachteten Verteilungen der Kontingenztafel bei gültiger Nullhypothese noch stärker in den Randbereichen verteilt sind. Hierfür kommen verschiedene Ordnungskriterien infrage. [302] Sofern nicht anders spezifiziert, wird für α ein Wert von 0.1 gewählt. Für die im Barnard-Test verwendete Wald-Statistik wird davon ausgegangen, dass die Varianzen der beiden Stichproben unabhängig sind [303]. Ist die Validierung für alle Testergebnisse eines konkreten Szenarios S_j erfolgreich, so gilt selbiges für die XiL-Umgebung ω insgesamt im betrachteten konkreten Szenario.

$$V_{\omega,j} = \begin{cases} 1, & V(T_i(S_j)) = 1 \forall i \\ 0, & sonst \end{cases} \tag{5.60}$$

Für die **minimal erforderliche Anzahl der R-Samples** n^r_{min} wird die Anforderung definiert, dass für jedes im XiL-Test auftretende Testergebnis mindestens eine Kombination möglicher Testergebnisse der R-Samples zu einer Ablehnung der Nullhypothese führen können muss. Es muss demnach ausgeschlossen werden, dass aufgrund eines zu kleinen n^r die Nullhypothese nicht abgelehnt werden kann, obwohl dies bei einem größeren n^r der Fall gewesen wäre. Schwierigkeit hierbei ist jedoch, dass die Verteilung der Testergebnisse der R-Samples nie exakt bekannt ist. Daher wird für jedes XiL-Testergebnis als „Worst Case“ angenommen, dass entweder kein R-Sample oder alle R-Samples eine Übereinstimmung mit diesem Testergebnis haben. Es muss damit gelten:

$$p_{T_i} \leq \alpha \forall i \tag{5.61}$$

für den Fall 1:

$$G(T_i) = \begin{pmatrix} 0 & n_i^x \\ n_{min}^r & n^x - n_i^x \end{pmatrix} \tag{5.62}$$

oder den Fall 2:

$$G(T_i) = \begin{pmatrix} n_{min}^r & n_i^x \\ 0 & n^x - n_i^x \end{pmatrix} \tag{5.63}$$

Mit steigendem n_{min}^r reduziert sich p_{T_i}. Daher wird n_{min}^r durch iterative Anwendung des Barnard-Tests ermittelt.

Eine **a-priori Abschätzung** wird über folgende Annahmen vorgenommen:

- Als „Worst Case“ gilt für die Verteilung von n_i^x:

$$n_i^x = \begin{cases} \frac{n^x}{2}, & n^x \; gerade \\ \frac{n^x+1}{2}, & n^x \; ungerade \end{cases} \tag{5.64}$$

- Für n^x wird die Abschätzung aus 5.3.3 übernommen: $n^x = n_{min}^x = 510$.
- Es wird angenommen: $\alpha = 0.1$.

Aus den Annahmen folgt:

$$G(T_i) = \begin{pmatrix} n_{min}^r & 255 \\ 0 & 255 \end{pmatrix} \tag{5.65}$$

In Fall 2 liegen identische Ergebnisse für p_{T_i} vor, sodass für die Bestimmung von n^r_{min} nur Fall 1 betrachtet wird. Die iterative Anwendung des Barnard Tests liefert:

$$p_{T_i}(n^r_{min} = 3) = 0.1294 > \alpha \tag{5.66}$$

$$p_{T_i}(n^r_{min} = 4) = 0.0647 < \alpha \tag{5.67}$$

Somit folgt: $n^r_{min} = 4$. Abweichende Annahmen können zu einem veränderten Wert für n^r_{min} führen.

6 Anwendung in einem Entwicklungsprojekt für Fahrerassistenzsysteme von Nutzfahrzeugen

6.1 Ausgangssituation

Für die Anwendung der in Kapitel 5 vorgestellten Ansätze zur Glaubwürdigkeitsbewertung des szenariobasierten XiL-Tests dient das in Abschnitt 1.3 genannte Entwicklungsprojekt als Referenz. Die Evaluierung wird dementsprechend mit realen Seriendaten vorgenommen.

6.1.1 System Under Test

Entwicklungsgegenstand und SUT ist die **Video Radar Decision Unit** der dritten Generation (**VRDU3**). Hierbei handelt es sich um ein Steuergerät basierend auf AUTOSAR [125], welches Fahrerassistenzsysteme algorithmisch umsetzt. Einsatzgebiet sind sowohl LKW als auch Busse aller globalen Marken der Daimler Truck AG. Das Steuergerät befindet sich während des Entstehungszeitraums dieser Dissertation in der Serienentwicklung. Zur Erfassung der Umwelt greift die VRDU3 auf insgesamt sechs Umweltsensoren (siehe Abb. 6.1) zurück:

- Ein **Long Range Radar (LRR)** an der Fahrzeugfront dient zur Erkennung von Objekten und Fahrbahnbegrenzungen vor dem Ego-Fahrzeug.
- Die **Frontkamera (Multi Purpose Camera, MPC)** erkennt Objekte und Fahrbahnmarkierungen vor dem Ego-Fahrzeug.

- Insgesamt vier **Short Range Radare (SRR)** sensieren Objekte vor, neben und hinter dem Ego-Fahrzeug, sodass mit der Kombination aller Sensoren ein Sichtfeld von ca. 270° erreicht wird.

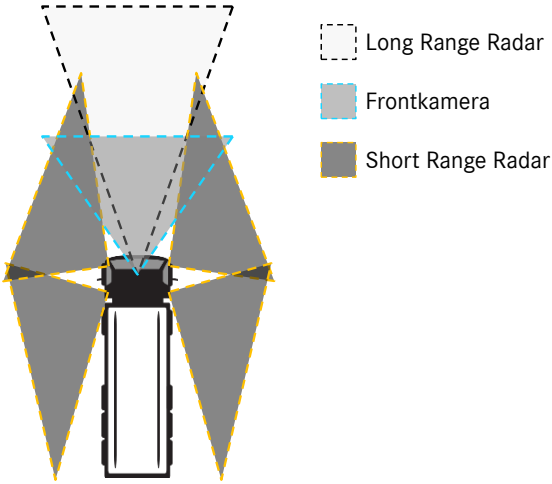

Abbildung 6.1: Übersicht der VRDU3-Umweltsensoren mit Sichtfeld (schematisch)

Die Informationen aller Sensoren werden in der VRDU3 fusioniert (siehe 2.1.2). Daneben sind auch die Subsysteme Interpretation/Prediction und Planning sowie Teile der Subsysteme Motion Control und Body Control als Software in der VRDU3 integriert. Ausgangsseitig kommuniziert die VRDU3 mit mehr als 15 weiteren Steuergeräten wie beispielsweise dem Powertrain Controller, dem Bremsensteuergerät und dem Steuergerät der aktiven Lenkung.

Für die Anwendung der Ansätze aus Kapitel 5 werden folgende Fahrerassistenzsysteme der VRDU3 betrachtet (siehe 2.1.3):

- **Active Brake Assist (ABA)**: Notbremsassistent, der sowohl auf andere Fahrzeuge als auch auf Vulnerable Road Users (VRUs), also Fußgänger und Fahrradfahrer reagiert. Das System verfügt über die in 2.1.3 beschriebene Warnkaskade.
- **Active Sideguard Assist (ASGA)**: Abbiegeassistent für Kreuzungen bei niedrigen Geschwindigkeiten bis 20 km/h. ASGA reagiert auf Fahrzeu-

ge, VRUs und statische Objekte. Neben der reinen Warnung führt dieses System auch eine aktive Bremsung aus.

- **Active Drive Assist (ADA)**: Dieses System stellt eine Kombination aus ACC und Spurhalteassistent dar und entspricht damit SAE Level 2 (siehe 2.1.1.1). Für die Betrachtung in diesem Kapitel wird die longitudinale Regelung des Systems (ACC) betrachtet.

6.1.2 X-in-the-Loop-Umgebung

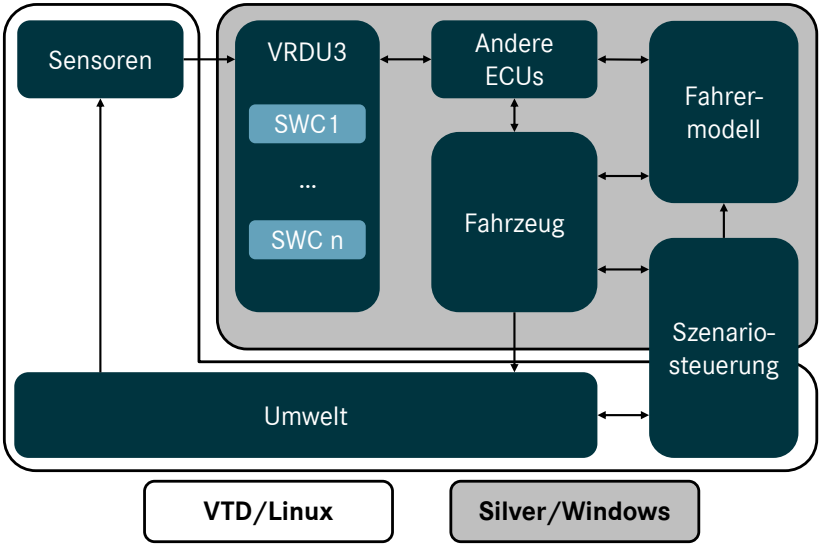

Abbildung 6.2: Übersicht der verwendeten SiL-Umgebung

Als Referenz-XiL-Umgebung dient hier eine SiL-Umgebung (siehe Abb. 6.2), sodass Tests rein virtuell durchgeführt werden. Eine wesentliche Anforderung ist dabei die Möglichkeit, szenariobasiert zu testen. Die SiL-Umgebung ist als Co-Simulation der beiden Tools Synopsys Silver [127] und VIRES Virtual Test Drive (VTD) [144] aufgesetzt. Silver ermöglicht die flexible Einbindung von FMUs, C-Code oder sogenannten Modifiern. Vorteil von VTD ist die Möglichkeit der szenariobasierten Umgebungssimulation und deren Visuali-

sierung. Da beide Tools auf unterschiedlichen Betriebssystemen ausgeführt werden müssen, wird hierfür ein Windows-Rechner (Silver) mit einem Linux-Rechner (VTD) via Netzwerk verbunden. Simulationsmaster ist dabei Silver. In Silver werden die folgenden Komponenten simuliert:

- SUT
- Fahrzeug
- Fahrer (Längs- und Querregelung)
- Szenariosteuerung (Fahrer Ego-Fahrzeug)
- Andere ECUs

Die VTD-Seite beinhaltet die folgenden Komponenten:

- Umwelt
- Sensor
- Szenariosteuerung (andere Objekte)

Grundsätzlich erlaubt eine SiL-Umgebung eine Testausführung schneller als Echtzeit. Der aktuell erreichbare Echtzeitfaktor[1] beträgt ca. 1,4. Die Simulation läuft somit nur unwesentlich schneller als Echtzeit ab. Berücksichtigt man die zur Initialisierung benötigte Zeit, so reduziert sich der Echtzeitfaktor auf ca. 0,5. Der erreichbare Echtzeitfaktor wird signifikant beeinflusst von der verwendeten Hardware sowie den eingesetzten Simulationsmodellen.

6.1.2.1 SUT-Repräsentation

Die VRDU3 ist in der SiL-Umgebung rein virtuell integriert. Da der Fokus auf dem funktionalen Verhalten des SUT liegt, wird lediglich die AUTOSAR Anwendungssoftware getestet (siehe Abb. 6.3). Diese besteht aus einzelnen Softwarekomponenten (Software Components, SWCs), die jeweils als FMUs

[1] Der Echtzeitfaktor stellt das Verhältnis zwischen der simulierten Zeit zur tatsächlich bei der Simulation benötigten Zeit dar.

in die SiL-Umgebung integriert werden. Die Kommunikation zwischen SWCs findet im realen Steuergerät über die AUTOSAR RTE (siehe 2.5.2.2) statt, welche hier über ein Signalmapping von Silver modelliert wird. Durch das FMU-Konzept lassen sich einzelne SWCs unabhängig voneinander austauschen oder auch als ganze Software-Releases in die SiL-Umgebung integrieren.

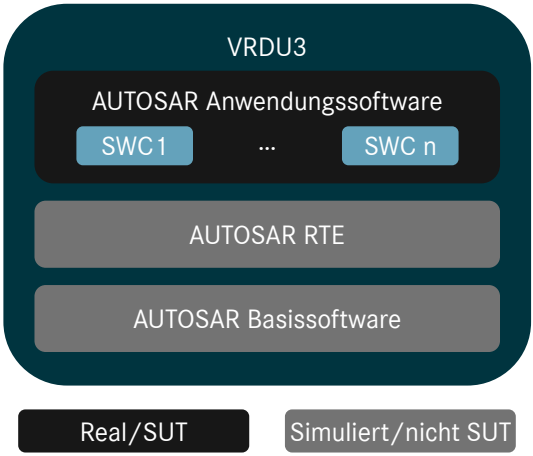

Abbildung 6.3: Repräsentation des SUT VRDU3

6.1.2.2 Umwelt

Die Simulation der Umwelt wird mithilfe der in VTD enthaltenen Modelle für andere Fahrzeuge, VRUs und statische Objekte vorgenommen. VTD lässt sich nur auf einem Linux-Rechner ausführen.

6.1.2.3 Sensoren

In dieser SiL-Umgebung kommen unabhängig von der Sensorart ideale Sensormodelle zum Einsatz, wobei zusätzlich die Möglichkeit besteht, die in 5.3.2.1 vorgestellten Fehler auf deterministischer oder statistischer Basis zu injizieren. Die Sensormodelle sind als eigene Module in VTD integriert und laufen

jeweils parallel in separaten Prozessen. Für die Kommunikation der Sensoren mit der VRDU3 existieren auf der Silver-Seite Empfängermodule, welche die Sensorsignale via Protobuf-Schnittstelle [167] empfangen. Entsprechend der Architektur aus 6.1.1 sind insgesamt sechs Sensormodelle für LRR, MPC und SRR in die SiL-Umgebung integriert.

6.1.2.4 Fahrzeug

Für die Modellierung des Fahrzeugs stehen zwei Modelle zur Verfügung, die sich hinsichtlich Modellierungstiefe und Anforderungen an die Rechenleistung unterscheiden. Fahrzeugmodell 1 basiert auf einer Mehrkörpersimulation und stellt aufgrund der höheren Modellierungstiefe entsprechend höhere Anforderungen an die Rechenleistung. Alternativ ist das auf einem Einspurmodell basierende Fahrzeugmodell 2 nutzbar, welches durch eine niedrigere Modellierungstiefe weniger Rechenleistung erfordert. Die Schnittstellen zu den restlichen ECUs unterscheiden sich je nach verwendetem Fahrzeugmodell. Für die Anwendung in diesem Kapitel wird das Fahrzeugmodell 1 verwendet.

6.1.2.5 Andere ECUs

Bei der Modellierung anderer ECUs wie beispielsweise dem Powertrain Controller und dem Bremsensteuergerät steht neben den genannten Unterschieden bei den Schnittstellen auch der Aspekt der Modellierungstiefe bzw. der Rechenanforderungen im Fokus. Aus diesem Grund existieren analog zu den Fahrzeugmodellen zwei Modellvarianten, die als Rebuild-Modelle (siehe 2.5.2.3) beide nur den Teil der Logik der anderen ECUs nachbilden, die für die VRDU3 SWCs von Bedeutung sind. Es wird die zu Fahrzeugmodell 1 korrespondierende Variante für die hier beschriebene Anwendung verwendet.

6.1.2.6 Fahrer

Auf der Stabilisierungsebene (siehe Abb. 2.2) wird der Fahrer durch einen Längs- und Querregler dargestellt. Die Führungsgröße v_{soll} des Längsreglers wird über eine Manövervorgabe (z.B. Beschleunigen, Geschwindigkeit halten) der Szenariosteuerung vorgegeben. Aus einem vorab definierten Pfad wird die

Führungsgröße des Querreglers abgeleitet. Weitere Fahreraktionen, die nicht direkt in die Fahrdynamik eingreifen (z.B. Blinker aktivieren), können über Bedingungen in der Szenariosteuerung vorgegeben werden.

6.1.2.7 Szenariosteuerung

Neben der Manöver- und Pfadsteuerung des Fahrers ist die Aufgabe der Szenariosteuerung, das Verhalten anderer Objekte zu steuern bzw. zu regeln. Hierfür werden die in VTD lesbaren XML-Szenariodateien und/oder standardisierte Formate wie OpenDRIVE und OpenSCENARIO (siehe 3.1.4.2) verwendet.

6.1.3 Realtest

Die für die Plausibilisierung und Validierung erforderlichen Realdaten werden in Prüfgeländetests für die FAS-Serienentwicklung generiert. Die Testfälle sind in Testfallkatalogen definiert, welche einen Teil der Testspezifikation darstellen (siehe 2.5.1.1). Während der Tests werden die folgenden Daten aufgezeichnet:

- Netzwerk-Inputs der VRDU3 (sensorseitig und fahrzeugseitig), z.B.: Objektlistendaten der Sensoren, Ego-Fahrzeug-Geschwindigkeit, Signale anderer Steuergeräte
- Netzwerk-Outputs der VRDU3, z.B.: Bremsanforderungen, interne Zustände, Signale an andere Steuergeräte
- Differential-GPS (nur bei ABA- und ASGA-Szenarien)

Die Anzahl der generierten R-Samples variiert je nach Testfall zwischen 1 und 5. Mit jedem neuen Software-Release der VRDU3 wird der Testfallkatalog erneut durchlaufen (siehe 2.2.5).

6.1.4 Referenzszenarien

Aufbauend auf der zuvor getroffenen Auswahl der betrachteten Systeme ABA, ASGA und ACC werden Szenarien zur Bewertung der Ansätze aus Kapi-

tel 5 definiert. Diese orientieren sich an in Standards und Forschungsprojekten genannten Testfallkatalogen. Aus einer funktionalen Szenariobeschreibung in verbaler Form sowie konkreten Parametern und Skizzen (Abb. 6.4 bis Abb. 6.6) resultiert eine Beschreibung konkreter Szenarien.

6.1.4.1 ABA

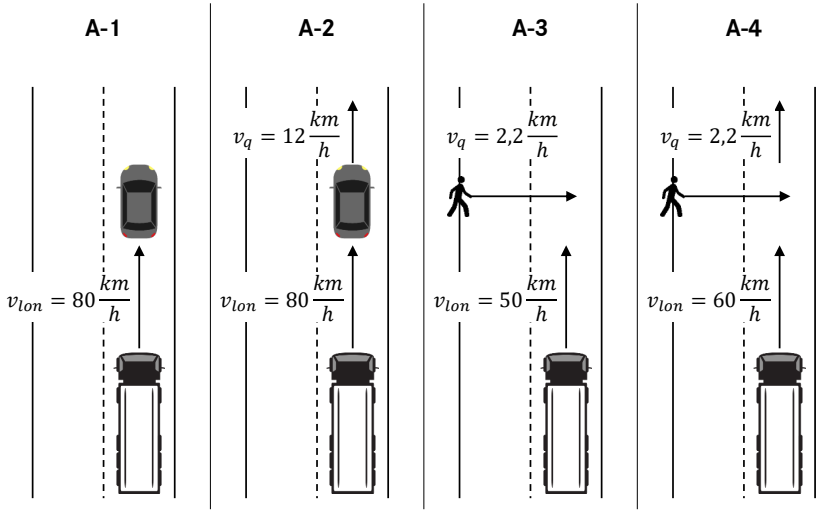

Abbildung 6.4: Skizzen zu Referenzszenarien für ABA

A-1 Das Ego-Fahrzeug fährt mit einer Geschwindigkeit von 80 km/h auf einen stehenden PKW auf. Dies entspricht dem Szenario CCRs der ISO 22733 [304], unter Anpassung der Geschwindigkeit des Ego-Fahrzeugs.

A-2 Das Ego-Fahrzeug fährt mit einer Geschwindigkeit von 80 km/h auf einen fahrenden PKW auf. Der PKW bewegt sich mit 12 km/h in die gleiche Fahrtrichtung wie das Ego-Fahrzeug. Dies entspricht dem Szenario CCRm der ISO 22733 [304], unter Anpassung der Geschwindigkeit des vorausfahrenden PKW.

A-3 Das Ego-Fahrzeug fährt geradeaus mit einer Geschwindigkeit von 50 km/h. Ein Fußgänger läuft mit einer Geschwindigkeit von 2,2 km/h von der linken Seite des Ego-Fahrzeugs kommend senkrecht zum Pfad des Ego-Fahrzeugs. Die Trajektorien von Ego-Fahrzeug und Fußgänger kreuzen sich so, dass bei konstanten Geschwindigkeiten eine Kollision stattfindet. Der Kollisionspunkt befindet sich bei 75 % der Fahrzeugbreite, gemessen von der linken Seite des Ego-Fahrzeugs. Dieses Szenario wird im Forschungsprojekt PROSPECT im Testfallkatalog aufgeführt [305].

A-4 Das Ego-Fahrzeug fährt geradeaus mit einer Geschwindigkeit von 60 km/h. Ansonsten ist der Ablauf identisch zu **A-3**. Dieses Szenario wird im Forschungsprojekt PROSPECT im Testfallkatalog aufgeführt [305], wobei die Geschwindigkeit des Ego-Fahrzeugs angepasst wurde.

6.1.4.2 ASGA

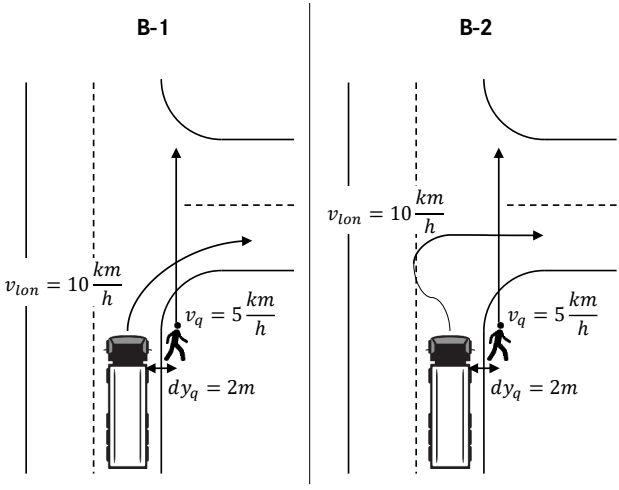

Abbildung 6.5: Skizzen zu Referenzszenarien für ASGA

B-1 Das Ego-Fahrzeug fährt mit einer Geschwindigkeit von 10 km/h und biegt **ohne** Ausholen nach rechts ab. Ein Fußgänger läuft mit einer Geschwindigkeit von 5 km/h rechts neben dem Ego-Fahrzeug in die Fahrtrichtung des Ego-Fahrzeugs vor dem Abbiegen. Der laterale Abstand des Fußgängers beträgt vor dem Abbiegen 2 m, gemessen von der rechten Seite des Ego-Fahrzeugs. Zu Beginn des Abbiegevorgangs befindet sich die Vorderkante des Ego-Fahrzeugs auf Höhe des Fußgängers. Dieses Szenario wird im Forschungsprojekt PROSPECT im Testfallkatalog aufgeführt [305], wobei die Geschwindigkeit des Ego-Fahrzeugs angepasst wurde.

B-2 Das Ego-Fahrzeug fährt mit einer Geschwindigkeit von 10 km/h und biegt **mit** Ausholen nach rechts ab. Ansonsten ist der Ablauf identisch zu **B-1**. Dieses Szenario wird im Forschungsprojekt PROSPECT im Testfallkatalog aufgeführt [305], wobei die Geschwindigkeit des Ego-Fahrzeugs und der Ausholvorgang als LKW-spezifisches Manöver angepasst wurden.

6.1.4.3 ACC

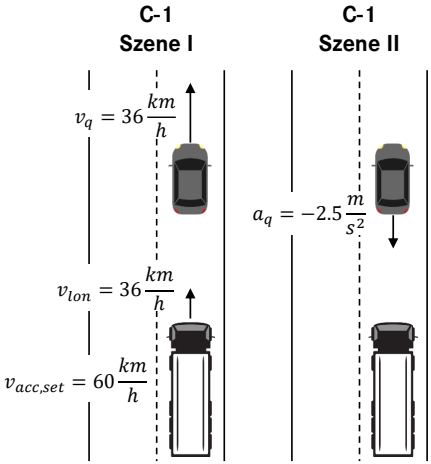

Abbildung 6.6: Skizze zu Referenzszenario für ACC

C-1 Das Ego-Fahrzeug fährt mit einer Geschwindigkeit von 36 km/h und aktivem ACC auf einen fahrenden PKW auf. Die ACC-Setzgeschwindigkeit des Ego-Fahrzeugs beträgt 60 km/h. Der PKW bewegt sich initial mit 36 km/h in die gleiche Fahrtrichtung wie das Ego-Fahrzeug und bremst in Folge des Szenarios mit 2.5 m/s^2 auf Stillstand ab. Das Ego-Fahrzeug bremst aufgrund des ACC-Eingriffs ebenfalls auf Stillstand ab. Das Szenario entspricht dem „Automatic Stop Capability Test“ der ISO 15622 [106].

6.1.5 Betrachtete Pass/Fail-Kriterien

Für die Glaubwürdigkeitsbewertung werden die folgenden PFCs definiert:

- $noColl$: Während des gesamten Szenarios hat keine Kollision zwischen Ego-Fahrzeug und einem anderen Objekt stattgefunden.
- $ttcTh$: Die minimale TTC hat die Schwelle TTC_{min} während des gesamten Szenarios nicht unterschritten.
- $abaW$: ABA hat eine optisch-akustische Warnung ausgelöst.
- $abaPB$: ABA hat eine Teilbremsung ausgelöst.
- $abaFB$: ABA hat eine Vollbremsung ausgelöst.
- $asgaI$: ASGA hat eine optische Information ausgelöst.
- $asgaW$: ASGA hat eine optisch-akustische Warnung ausgelöst.
- $asgaB$: ASGA hat eine Bremsung ausgelöst.
- $vEgoCollTh$: Die Geschwindigkeit des Ego-Fahrzeugs zum Zeitpunkt der Kollision hat die Schwelle $v_{egoCollMax}$ nicht überschritten.
- $aThTimeTh$: Der maximale Betrag der Beschleunigung hat die Schwelle a_{thMax} während des gesamten für eine Dauer von weniger als $t_{thATimeMax}$ während des gesamten Szenarios überschritten.

Für die hier betrachtete Anwendung werden die genannten Parameter wie folgt gewählt:

- $TTC_{min} = 0.2s$
- $v_{egoCollMax} = 0.5\frac{m}{s}$
- $a_{thMax} = 2\frac{m}{s^2}$
- $t_{thATimeMax} = 1s$

Die Auswahl der PFCs und somit die Zusammensetzung des Testergebnisses erfolgt in Abhängigkeit vom zu testenden FAS:

$$T_{ABA} = [noColl, abaW, abaPB, abaFB, ttcTh] \tag{6.1}$$

$$T_{ASGA} = [noColl, asgaI, asgaW, asgaB, vEgoCollTh] \tag{6.2}$$

$$T_{ACC} = [noColl, tthTh, aThTimeTh] \tag{6.3}$$

Die hier getroffene Auswahl stellt nur ein Subset möglicher PFCs da. Für die Bewertung der Ansätze aus Kapitel 5 ist die vollständige Abdeckung der PFCs nicht erforderlich.

Darüber hinaus wird auf Basis der PFCs bewertet, ob die Testdurchführung korrekt war. In Abhängigkeit des zu testenden Systems werden Samples dann als nicht korrekt durchgeführt klassifiziert, wenn die folgenden Bedingungen erfüllt sind:

- ABA: $noColl \wedge \neg abaPB$
- ASGA: $(noColl \wedge \neg asgaB) \vee abaPB$
- ACC: keine

Jedes als nicht korrekt durchgeführt klassifizierte Sample wird in den Schritten Prädiktion, Plausibilisierung und Validierung nicht berücksichtigt.

6.1.6 Datenvorverarbeitung

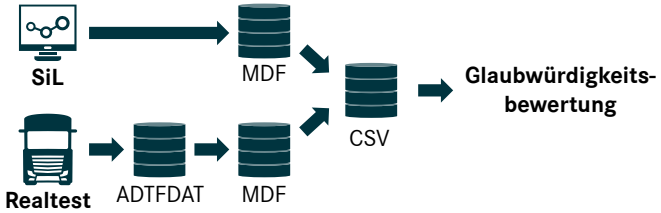

Abbildung 6.7: Datenvorverarbeitung für Realtest und SiL

Um die in XiL und Realtest aufgezeichneten Daten für eine Auswertung nutzbar zu machen, ist jeweils eine Datenvorverarbeitung notwendig (siehe Abb. 6.7). Ziel ist es, die Daten so zu verarbeiten, dass unabhängig von der jeweiligen Testumgebung identische Python-Auswerteskripte der Glaubwürdigkeitsbewertung angewendet werden können.

In der SiL-Umgebung findet die Aufzeichnung im Measurement Data Format (MDF) [306] statt, wobei die gesamte virtuelle Netzwerkkommunikation mit aufgezeichnet wird. Bei der Datenvorverarbeitung werden die für die Glaubwürdigkeitsbewertung erforderlichen Signale selektiert und weitere abgeleitete Signale (z.B. L-Shape Punkte von Objekten) berechnet. Zielformat ist eine Datei im Format Comma-separated values (CSV).

Realdaten werden im sogenannten ADTFDAT Format [307] aufgezeichnet. Über mehrere Schritte wird daraus ebenfalls eine MDF-Datei generiert. Da sich Vorhandensein und Art der verwendeten Referenzmesstechnik zwischen Messungen unterscheiden können, sind für die weitere Verarbeitung unterschiedliche Skripte vorhanden, die letztlich eine CSV mit den gleichen Signalnamen generieren. Die Selektion von Signalen und Generierung abgeleiteter Signale kommt analog zur SiL-Umgebung auch hier zum Einsatz. Je nach Szenario werden die Messdaten manuell zeitlich zugeschnitten, um eine nachfolgende automatisierte Weiterverarbeitung zu ermöglichen.

Nachdem die SiL- bzw. Realtest-Daten in einem konsistenten Format sind, findet ein weiterer automatisierter Zuschnitt statt, um einen möglichst synchronen Startzustand und objektive Endkriterien sicherzustellen. Maßgeblich für das Startkriterium ist die longitudinale Relativposition des Objekts. Das

Endkriterium wird durch Kollision, Stillstand des Ego-Fahrzeugs oder Maximaldauer erreicht. Start- und Endkriterium werden szenarioabhängig definiert.

6.2 Prädiktion

Ziel der Prädiktion ist es, die Auftretenswahrscheinlichkeit von Testergebnissen in der Realität mithilfe von XiL-Tests abzuschätzen, indem die Unsicherheit zwischen XiL-Umgebung und gemessenem Realtest über entsprechende Modelle prädiziert wird (siehe Abschnitt 5.3). Über ein Monte-Carlo-Sampling werden anschließend aus den absoluten Häufigkeiten auftretender Testergebnisse Konfidenzintervalle ermittelt. Hierfür kommt das Python-Package statsmodels zum Einsatz[2] [308]. Die minimale Anzahl der MC-Samples wird wie in 5.3.3 hergeleitet auf 510 festgelegt. Für die Evaluation der Prädiktion werden alle in 6.1.4 definierten konkreten Szenarien betrachtet. Bei Szenario A-1 wird zudem eine Prädiktion mit der Berücksichtigung von Regen im Unsicherheits-Subsystem 2 (Fahrzeug) durchgeführt.

6.2.1 Unsicherheitsmodelle und Parametrierung

Es werden die in 5.3.2 definierten Modelle für die Unsicherheits-Subsysteme Sensoren und Fahrzeug verwendet.

6.2.1.1 Sensoren

Die Unsicherheitsmodelle der Sensoren sind direkt in die in 6.1.2.3 genannten Sensormodelle in VTD integriert. Auf Basis von Analysen der Realdaten werden die Unsicherheitsmodelle parametriert. Entsprechend der Datenverfügbarkeit werden die folgenden Unsicherheiten (siehe 5.3.2.1) parametriert:

Länge des longitudinalen/lateralen Sichtfeldes Für die beiden Sensoren LRR und MPC wird eine Skalierung des Sichtfeldes vorgenommen, die für

[2] `https://www.statsmodels.org/dev/index.html`

jedes MC-Sample konstant ist und zwischen MC-Samples normalverteilt variiert. Ein Offset wird nicht parametriert. Somit gilt für den Sensor LRR:

$$S_{err,1,LRR} \sim \mathcal{N}(\mu_{s,1,LRR}, \sigma^2_{s,1,LRR}) \tag{6.4}$$

$$O_{err,1,LRR} = 0 \tag{6.5}$$

Für den Sensor MPC gilt:

$$S_{err,1,MPC} \sim \mathcal{N}(\mu_{s,1,MPC}, \sigma^2_{s,1,MPC}) \tag{6.6}$$

$$O_{err,1,MPC} = 0 \tag{6.7}$$

Aufgrund der Datenverfügbarkeit wird für die Sensoren SRR keine Unsicherheit des Sichtfeldes parametriert. Es ergibt sich daher:

$$S_{err,1,SRR} = 1 \tag{6.8}$$

$$O_{err,1,SRR} = 0 \tag{6.9}$$

Relative Objektdistanz longitudinal Für die Modellierung der Unsicherheit der relativen Objektdistanz longitudinal und lateral dient als Eingangsgröße der Interpolation nach Gl. 5.13 die Relativdistanz zum Objekt:

$$w = d_q = \sqrt{dx_q^2 + dy_q^2} \tag{6.10}$$

Es werden die Stützstellen $\hat{d_q}^1 = 0\,\text{m}$, $\hat{d_q}^2 = 50\,\text{m}$ und $\hat{d_q}^3 = 100\,\text{m}$ gewählt. Die entsprechenden Funktionswerte $g(d_q)$ werden über Analysen von Realdaten extrahiert.

Der Sensor LRR wird mit einer Offsetunsicherheit, einer Rauschunsicherheit und einer periodischen Unsicherheit beaufschlagt, jeweils für die longitudinale Distanz.

Eine Skalierungsunsicherheit wird nicht parametriert, sodass gilt:

$$O_{err,2,LRR}(d_q) \sim f_{int,o,2,LRR}(d_q) \tag{6.11}$$
$$S_{err,2,LRR}(d_q) = 1 \tag{6.12}$$
$$\Sigma_{2,r,LRR}(d_q) \sim f_{int,\sigma,2,LRR}(d_q) \tag{6.13}$$
$$A_{2,LRR}(d_q) \sim f_{int,A,2,LRR}(d_q) \tag{6.14}$$
$$\Omega_{2,LRR}(d_q) \sim f_{int,\omega,2,LRR}(d_q) \tag{6.15}$$

Für die Sensoren MPC und SRR werden Offsetunsicherheit und Rauschunsicherheit parametriert. Es folgt:

$$O_{err,2,MPC}(d_q) \sim f_{int,o,2,MPC}(d_q) \tag{6.16}$$
$$S_{err,2,MPC}(d_q) = 1 \tag{6.17}$$
$$\Sigma_{2,r,MPC}(d_q) \sim f_{int,\sigma,2,MPC}(d_q) \tag{6.18}$$
$$P_{err,2,MPC}(t, d_q) = 0 \tag{6.19}$$

Für die Sensoren SRR gilt analog:

$$O_{err,2,SRR}(d_q) \sim f_{int,o,2,SRR}(d_q) \tag{6.20}$$
$$S_{err,2,SRR}(d_q) = 1 \tag{6.21}$$
$$\Sigma_{2,r,SRR}(d_q) \sim f_{int,\sigma,2,SRR}(d_q) \tag{6.22}$$
$$P_{err,2,SRR}(t, d_q) = 0 \tag{6.23}$$

Relative Objektdistanz lateral Der Sensor LRR wird nicht mit einer Unsicherheit der lateralen Objektdistanz beaufschlagt. Somit ergibt sich:

$$O_{err,3,LRR}(d_q) = 0 \tag{6.24}$$
$$S_{err,3,LRR}(d_q) = 1 \tag{6.25}$$
$$R_{err,3,LRR}(t, d_q) = 0 \tag{6.26}$$
$$P_{err,3,LRR}(t, d_q) = 0 \tag{6.27}$$

Analog wird beim Sensor MPC verfahren:

$$O_{err,3,MPC}(d_q) = 0 \tag{6.28}$$
$$S_{err,3,MPC}(d_q) = 1 \tag{6.29}$$
$$R_{err,3,MPC}(t, d_q) = 0 \tag{6.30}$$
$$P_{err,3,MPC}(t, d_q) = 0 \tag{6.31}$$

Für die Sensoren SRR wird die Offsetunsicherheit und Rauschunsicherheit parametriert:

$$O_{err,3,SRR}(d_q) \sim f_{int,o,2,SRR}(d_q) \tag{6.32}$$
$$S_{err,3,SRR}(d_q) = 1 \tag{6.33}$$
$$\Sigma_{3,r,SRR}(d_q) \sim f_{int,\sigma,2,SRR}(d_q) \tag{6.34}$$
$$P_{err,3,SRR}(t, d_q) = 0 \tag{6.35}$$

Sonstige Größen Ziel der hier dokumentierten Anwendung ist es, die Einflüsse wesentlicher Unsicherheiten zu untersuchen. Dies umfasst insbesondere Unsicherheiten, deren signifikanter funktionaler Einfluss aus der Realität bekannt ist bzw. zu erwarten ist. Alle weiteren in 5.3.2.1 aufgelisteten Unsicherheiten werden in der hier betrachteten Anwendung nicht parametriert, sodass gilt:

$$O_{err,i}(d_q) = 0 \tag{6.36}$$
$$S_{err,i}(d_q) = 1 \tag{6.37}$$
$$R_{err,i}(t, d_q) = 0 \tag{6.38}$$
$$P_{err,i}(t, d_q) = 0 \tag{6.39}$$

6.2.1.2 Fahrzeug

Für das Subsystem 2 aus Abb. 5.4 wird wie in 5.3.2.2 beschrieben ein dem Fahrzeugmodell nachgelagertes Unsicherheitsmodell implementiert und para-

metriert. Modul 1 wird über ein lineares Regressionsmodell mit den folgenden Inputs $s = [v_{pre,lon}, a_{pre,lon}, \delta_{stw}]$ umgesetzt [318]:

- $v_{pre,lon}$: Geschwindigkeit des Ego-Fahrzeugs vor der Manipulation
- $a_{pre,lon}$: Beschleunigung des Ego-Fahrzeugs vor der Manipulation
- δ_{stw}: Lenkradwinkel

Für $a_{err,lon}$ wird aufgrund der Gegebenheiten beim Realtest (siehe 6.3.2) je eine Modellvariante mit und ohne Regen bei der Testdurchführung parametriert. Ebenso kommen szenarioabhängig zwei abweichende Parametrierungen der Interpolation für $\dot{\psi}_{err}$ zum Einsatz. Maßgeblich ist in letzterem Fall, ob das Ego-Fahrzeug ein Abbiegemanöver durchführt.

Bei der Umsetzung des Gauß-Prozesses in Modul 2 ist die Generierung der Zufallswerte für alle Zeitschritte bereits vor der Simulation erforderlich. Jedoch sind die Parameter μ und σ erst zur Laufzeit der Simulation bekannt, da sie von den Inputs aus der Simulation zum jeweiligen Zeitpunkt abhängen. Aus diesem Grund wird die Generierung der Zeitreihe für einen standardnormierten Gauß-Prozess a-priori durchgeführt und erst zur Laufzeit in eine normalverteilte Größe überführt:

$$z_{a,lon}(t) \sim \mathcal{GP}(0, C_{a,lon}(t)) \tag{6.40}$$

$$z_{\dot{\psi}}(t) \sim \mathcal{GP}(0, C_{\dot{\psi}}(t)) \tag{6.41}$$

Der Parameter r für die Kovarianzfunktionen von $C_{a,lon}(t)$ und $C_{\dot{\psi}}(t)$ wird jeweils auf Basis von Realdaten parametriert. Für die daraus generierten Fehler folgt:

$$a_{err,lon}(t,s) = \mu_{a,lon}(s) + z_{a,lon}(t) \cdot \sigma_{a,lon}(s) \tag{6.42}$$

$$\dot{\psi}_{err}(t,s) = \mu_{\dot{\psi}}(s) + z_{\dot{\psi}}(t) \cdot \sigma_{\dot{\psi}}(s) \tag{6.43}$$

In Modul 3 werden die in 5.3.2.2 erläuterten kinematischen Zusammenhänge in einem eigenen Simulationsmodell umgesetzt und in die SiL-Umgebung

als FMU integriert. Eine zusätzliche Logik ist für die beiden Fälle notwendig, wenn eine negative Beschleunigungsanforderung vorliegt und entweder $v_{lon} = 0$ oder $v_{pre,lon} = 0$ gilt. Beide Fälle sind beispielsweise bei einem Notbremsszenario möglich. Für erstgenannten Fall wird die effektive Beschleunigungsanforderung auf 0 m/s^2 gesetzt, da das Fahrzeug nach der Manipulation bereits steht und nicht weiter verzögern kann. Im zweiten Fall wird die Beschleunigungsanforderung vollständig über Modul 3 umgesetzt, da das Fahrzeugmodell selbst nicht mehr verzögert.

6.2.1.3 Fahrer

Die Unsicherheit des Fahrers wird über die entsprechende Manipulation von v_{soll} und x_{ego} implementiert (siehe 5.3.2.3). Für die Parametrierung der dabei zum Einsatz kommenden Interpolationsfunktionen werden Realdaten herangezogen. Die Stützstellen für $v_{soll,pre}$ bzw. v_{soll} sind 0 km/h, 10 km/h und 20 km/h. [318]

6.2.2 Ergebnisse

Bei Anwendung der Prädiktion treten die in Abb. 6.8, Abb. 6.10 und Abb. 6.12 dargestellten Testergebnisse auf. Die Testergebnisse sind für jedes zu testende FAS individuell auf Basis der in 6.1.5 getroffenen Auswahl an PFCs aufgelistet. Für die Testergebnisse lassen sich aus der Prädiktion absolute Verteilungen sowie die entsprechenden Konfidenzintervalle quantifizieren (siehe Abb. 6.9, Abb. 6.11 und Abb. 6.13). Zusätzlich wird die für die Validierung minimal erforderliche Zahl von R-Samples berechnet (siehe Abschnitt 5.5).

Testergebnis	noColl	abaW	abaPB	abaFB	ttcTh
T1	0	1	0	0	0
T2	0	1	1	1	0
T3	1	1	1	1	0
T4	1	1	1	1	1

Abbildung 6.8: Auftretende Testergebnisse für ABA mit jeweiliger PFC-Ausprägung (siehe 6.1.5)

Szenario		T1	T2	T3	T4	n^x	n^r_{min}
A-1	n^x_i	n/a	10	2	521	533	2
	$CI_{i,\omega,j}$	n/a	0.00898 0.03878	0.00081 0.01726	0.95628 0.98853		
A-1 Regen	n^x_i	n/a	469	6	58	533	2
	$CI_{i,\omega,j}$	n/a	0.84211 0.90965	0.00440 0.02852	0.08059 0.14537		
A-2	n^x_i	n/a	17	3	511	531	2
	$CI_{i,\omega,j}$	n/a	0.01815 0.05588	0.00156 0.02030	0.93712 0.97768		
A-3	n^x_i	2	38	12	476	528	2
	$CI_{i,\omega,j}$	0.00077 0.01839	0.04860 0.10534	0.01126 0.04534	0.86429 0.92937		
A-4	n^x_i	n/a	480	14	46	540	2
	$CI_{i,\omega,j}$	n/a	0.85234 0.91727	0.01388 0.04793	0.06060 0.11848		

Abbildung 6.9: Ergebnisse der Prädiktion für ABA-Szenarien

In den vier getesteten ABA-Szenarien treten bis zu vier unterschiedliche Testergebnisse auf. Das erwünschte Testergebnis ist in allen ABA-Szenarien $T4$ und wird in den Szenarien A-1 (kein Regen), A-2 und A-3 jeweils in der überwiegenden Mehrzahl der Testausführungen erreicht. Basierend auf den zugehörigen Konfidenzintervallen variiert in diesen Szenarien die minimale Wahrscheinlichkeit für $T4$ zwischen 86.429% und 95.628%. Für die Szenarien A-1 (Regen) und A-4 tritt das Testergebnis $T2$ am häufigsten auf.

Insgesamt vier verschiedene Testergebnisse treten in den beiden betrachteten ASGA-Szenarien auf. Das erwünschte Testergebnis ist $T4$, tritt jedoch nur mit sehr geringer Wahrscheinlichkeit auf. Die entsprechenden maximalen Wahrscheinlichkeiten aus den Konfidenzintervallen für dieses Testergebnis betragen 5.250% bzw. 4.093%. Da jedoch bereits eine ASGA-Bremsung zur einer deutlichen Verringerung der Unfallschwere beitragen kann, stellen die Testergebnisse $T2$ bis $T4$ eine ausreichende Systemreaktion dar.

Testergebnis	noColl	asgaI	asgaW	asgaB	vEgoCollTh
T1	0	1	1	0	0
T2	0	1	1	1	0
T3	0	1	1	1	1
T4	1	1	1	1	1

Abbildung 6.10: Auftretende Testergebnisse für ASGA mit jeweiliger PFC-Ausprägung (siehe 6.1.5)

Szenario		T1	T2	T3	T4	n^x	n^r_{min}
B-1	n^x_i	46	419	45	10	520	2
	$CI_{i,\omega,j}$	0.06204 0.12464	0.75892 0.84537	0.06044 0.12244	0.00893 0.04093		
B-2	n^x_i	33	350	132	15	530	2
	$CI_{i,\omega,j}$	0.04080 0.09391	0.60740 0.70963	0.20524 0.29871	0.01508 0.05250		

Abbildung 6.11: Ergebnisse der Prädiktion für ASGA-Szenarien

Testergebnis	noColl	ttcTh	aThTimeTh
T1	1	1	0
T2	1	1	1

Abbildung 6.12: Auftretende Testergebnisse für ACC mit jeweiliger PFC-Ausprägung (siehe 6.1.5)

Szenario		T1	T2	n^x	n^r_{min}
C-1	n^x_i	423	110	533	2
	$CI_{i,\omega,j}$	0.75168 0.83008	0.16992 0.24832		

Abbildung 6.13: Ergebnisse der Prädiktion für ACC-Szenarien

Im Fall des ACC-Szenarios treten zwei Testergebnisse auf, die beide eine ausreichende Systemleistung darstellen. Maßgeblicher Unterschied ist das PFC

$aThTimeTh$, also die Überschreitung des Grenzwerts für die maximale Beschleunigung (siehe 6.1.5). Die minimale Wahrscheinlichkeit für die Überschreitung liegt auf Basis der Konfidenzintervalle bei 75.168%.

6.2.3 Diskussion

6.2.3.1 Einordnung der Ergebnisse

Die in der Prädiktion erreichten Testergebnisverteilungen und Konfidenzintervalle entsprechen in allen Fällen grundsätzlich den in der Realität zu erwartenden Ergebnissen. Aufgrund des in Szenario A-1 in den Unsicherheitsmodellen parametrierten Regens bei der Testdurchführung kommt es in diesem Szenario bei der überwiegenden Mehrzahl der MC-Samples zu einer Kollision. Dies entspricht dem im Realtest beobachtbaren Verhalten (siehe 6.3.2). Ohne Berücksichtigung von Regen ist das Gegenteil der Fall.

Ein weiterer Einfluss ist die Geschwindigkeit des Ego-Fahrzeugs, die sich in den Szenarien A-3 und A-4 unterscheidet. Im Fall der höheren Geschwindigkeit in Szenario A-4 kommt es bei der Mehrheit der Szenarien zu Kollisionen, was bei A-3 nicht der Fall ist. Dies deutet auf eine potentielle Systemgrenze hin.

In den Szenarien B-1 und B-2 wird zwar in wenigen Fällen eine Kollision verhindert, jedoch löst das System in allen Fällen eine Warnung aus und bremst in der überwiegenden Mehrzahl der MC-Samples ($T2$, $T3$ und $T4$). Dies deckt sich mit den in der Realität beobachteten Ergebnissen (siehe 6.3.2). Das Ausholmanöver in Szenario B-2 führt zu weniger MC-Samples mit einer höheren Ego-Geschwindigkeit bei Kollision.

Bei Szenario C-1 kommt es in keinem Fall zu einem kritischen Zustand. Der Einfluss der Unsicherheitsmodelle wird jedoch sichtbar hinsichtlich $aThTimeTh$ (siehe 6.1.5). Ca. 80% der MC-Samples überschreiten die Schwelle a_{thMax}.

Die in 5.3.3 definierte Anforderung zur maximalen Breite der Konfidenzintervalle von 0.1 wird für alle Szenarien und Testergebnisse erreicht. Um die Konfidenzintervalle weiter zu verkleinern, ist eine Erhöhung von n^x notwendig.

Für die statistische Bewertung der Testergebnis-Verteilungen wird auf Abschnitt 6.4 verwiesen.

6.2.3.2 Bewertung der Methode

Insgesamt drei Unsicherheitsmodelle (siehe 5.3.2) kommen bei der Prädiktion zum Einsatz, um neben der Abweichung zwischen Simulationsmodellen und Realität auch den Nichtdeterminismus in Realtests abzubilden. Die Auswahl der Subsysteme erfolgt auf Basis der Struktur der verwendeten SiL-Umgebung sowie Expertenwissen über Einflussquellen möglicher Unsicherheiten. Sowohl die Anzahl als auch die Abgrenzung der Subsysteme in weiteren Anwendungen sind ebenfalls davon abweichend wählbar. Es handelt sich in dieser Anwendung um eine mögliche Lösungsvariante.

Die Unsicherheitsmodelle in dieser Anwendung sind so konzeptioniert, dass sie sich mit den vorhandenen Daten und Ressourcen implementieren und parametrieren lassen. Priorität hat die Integration der Modelle in die SiL-Umgebung. Aus diesem Grund wird nicht für alle in 5.3.2.1 genannten Größen von Unsicherheits-Subsystem 1 (Sensoren) eine Unsicherheit parametriert. Für Unsicherheits-Subsystem 2 (Fahrzeug) erfolgt die Modellierung über eine lineare Regression, was ebenfalls eine Vereinfachung darstellt. Bei Unsicherheits-Subsystem 3 (Fahrer) werden lediglich zwei Führungsgrößen des Reglers manipuliert. Trotz aller Vereinfachungen lässt sich aus den Ergebnissen der Prädiktion schließen, dass Unsicherheiten mit den gewählten Subsystemen und verwendeten Modellen realitätsnah abbildbar sind.

Dennoch besteht weiterer Forschungsbedarf im Bereich der Konzeption von Unsicherheitsmodellen. Alternative Ansätze zur Unsicherheitsmodellierung aus dem Bereich des maschinellen Lernens stellen ein vielversprechendes Forschungsfeld dar. Untersuchungen auf Basis eines neuronalen Netzes [325] zeigen jedoch, dass erheblicher Mehraufwand zum Erreichen einer ausreichenden Genauigkeit im Vergleich zur in dieser Dissertation verwendeten linearen Regression [318] notwendig ist. Ebenfalls weiteres Potential existiert für die Modellierung der Unsicherheit von Sensoren. Relevant ist hier die Auswahl modellierter Effekte sowie der Methode zur Modellierung von Unsicherheit.

Aus der Varianz der betrachteten Szenarien lässt sich ableiten, dass die verwendeten Unsicherheitsmodelle grundsätzlich in unterschiedlichen Szenarien und Manövern des Ego-Fahrzeugs anwendbar sind. Eine Übertragbarkeit der konzeptionellen Modelle ist sowohl innerhalb eines logischen Szenarios als auch zwischen unterschiedlichen logischen Szenarien gegeben.

Die Parametrierung der Unsicherheitsmodelle ist jedoch im Fall von Unsicherheitsmodell 2 szenarioabhängig. Ebenso wird für dieses Subsystem eine abweichende Parametrierung für Szenarien mit und ohne Regen auf der Fahrbahn angewandt. Die zur Parametrierung verwendeten Daten entsprechen konkreten Szenarien, die identisch zu den konkreten Anwendungsszenarien sind oder einem logischen Szenario der Anwendungsszenarien zugeordnet werden können. Daraus ergibt sich, dass die Parametrierung im Allgemeinen nicht auf neue Szenarien übertragbar ist. Es besteht somit Forschungspotential für Unsicherheitsmodelle, die eine Übertragbarkeit der Parametrierung erlauben.

Die Aggregation der Unsicherheitsmodelle in der SiL-Umgebung bringt einen Erkenntnisgewinn, da sich so einerseits Unsicherheiten überlagern und andererseits der Einfluss des SUT in einem geschlossenen Regelkreis mit betrachtet wird. Der qualitative Einfluss von logischem Szenario bzw. Szenarioparametern auf das Testergebnis kann bereits aus dem Vergleich der Testergebnis-Verteilungen der Prädiktion abgeleitet werden. So ist es bereits in der Prädiktion möglich, Szenarien zu identifizieren, die eine große bzw. kleine Varianz hinsichtlich des Testergebnisses haben.

Mithilfe des Monte-Carlo-Ansatzes werden Testergebnis-Verteilungen ermittelt, aus denen mithilfe des Multinomial-Ansatzes Konfidenzintervalle und damit statistische Aussagen abgeleitet werden können. Neben der Bewertung eines einzelnen konkreten Szenarios lässt sich auch ein relativer Vergleich des Einflusses von Parameteränderungen innerhalb eines logischen Szenarios durchführen. So führt beispielsweise die Erhöhung der Ego-Geschwindigkeit zwischen den Szenarien A-3 und A-4 zu einer Steigerung des relativen Anteils von Kollisionen. Die Analyse von Testergebnis-Verteilungen zur Bewertung der Leistungsfähigkeit des SUT ist nicht Schwerpunkt dieser Dissertation.

Die Wahl der PFCs hat entscheidenden Einfluss auf die Granularität der Testergebnisse und damit auch auf die Konfidenzintervalle. Eine nachträgliche Addition der Maxima bzw. Minima mehrerer Testergebnisse zu einem aggregierten Wert ist zwar für eine Abschätzung möglich, liefert aber potentiell unpräzisere Konfidenzintervalle. Ein Beispiel wäre die Betrachtung, ob eine Kollision ($noColl = 0$) stattgefunden hat. Existieren daneben noch weitere zu analysierende PFCs, so führt dies ggf. auch zu mehreren Testergebnissen mit $noColl = 0$. In diesem Fall sollte eine neue Berechnung der Konfidenzintervalle vorgenommen werden, die lediglich das PFC $noColl$ berücksichtigt.

Da die Anzahl der MC-Samples begrenzt ist, wurde als Kriterium die maximale Breite eines Konfidenzintervalls gewählt (siehe 5.3.3). Möglich ist dennoch, dass ein im Realtest mögliches Testergebnis im XiL-Test nicht auftritt. Da $n^x \gg n^r$ und aufgrund der Verwendung des Ansatzes nach Goodman [293] wurde dies hier nicht berücksichtigt. Über die Einführung eines weiteren Testergebnisses mit $n_i^x = 0$ wäre eine Abschätzung der Wahrscheinlichkeit eines solchen Testergebnisses möglich.

6.3 Plausibilisierung

In der Plausibilisierung wird jedes R-Sample mit jedem MC-Sample jeweils einzeln verglichen, um festzustellen, ob diese äquivalent sind (siehe Abschnitt 5.4). Bedingung für eine erfolgreiche Plausibilisierung ist, dass zumindest für eine Kombination von MC-Sample und R-Sample eine Äquivalenz vorliegt. Für die Evaluation der Plausibilisierung werden die Szenarien A-1, A-3 und B-1 herangezogen. Es werden die PFCs aus 6.1.5 und Szenariodistanzmaße wie in 5.4.1 definiert verwendet. Die Implementierung nutzt das Python-Package dtw-python[3] [309].

6.3.1 Grenzwerte für Szenariodistanzmaße

Für die Äquivalenz zweier Szenarien hinsichtlich Szenariodistanzmaßen ($E2$ in Gl. 5.43) werden die Grenzwerte auf Basis der Ergebnisse der Prädiktion ermittelt (siehe 5.4.2 und Abb. 6.14). Ergänzend und zum Vergleich sind die für Szenario A-1 aus den Realtests abgeleiteten Grenzwerte aufgelistet. Diese finden jedoch in der Plausibilisierung keine weitere Verwendung. Bei der Implementierung des Toleranzintervalls kommt das Python-Package toleranceinterval[4] zum Einsatz.

[3] https://dynamictimewarping.github.io/
[4] https://jekel.me/tolerance_interval_py/index.html

Szenario	$d_{1,max}$	$d_{2,max}$	$d_{3,max}$
A-1 SiL	0.9571	1.1694	0.0056
A-1 Real	2.4434	1.0840	0.0063
A-3 SiL	0.6351	0.4940	0.0072
B-1 SiL	0.6667	0.3453	0.0387

Abbildung 6.14: Grenzwerte für Szenariodistanzmaße je Szenario

6.3.2 Ergebnisse

Für die Plausibilisierung werden die in den Realtests erzielten Testergebnis-Verteilungen herangezogen (siehe Abb. 6.15). Für Szenario A-1 wurden die Realtest bei Regen durchgeführt. Daher findet die Plausibilisierung auch anhand der MC-Samples für Regen statt. Die Anzahl plausibler Kombinationen aus R-Sample und MC-Sample ist für jedes R-Sample der genannten konkreten Szenarien in Abb. 6.16 zu finden. Für alle betrachteten Szenarien und zugehörigen R-Samples existiert mindestens ein äquivalentes MC-Sample. Somit ist die Plausibilisierung für alle Szenarien erfolgreich.

Szenario		T1	T2	T3	T4
A-1 Regen	n_i^r	0	5	0	0
A-3	n_i^r	0	0	0	3
B-1	n_i^r	1	2	0	0

Abbildung 6.15: Ergebnisse der Realtests

Exemplarisch sind für das Szenario A-1 in Abb. A.1 und Abb. A.2 Trajektoriendaten sowie daraus resultierende Szenariodistanzmaße für eine plausible Kombination aus R-Sample und MC-Sample geplottet. Analog ist dies für ei-

ne nicht plausible Kombination in Abb. A.3 und Abb. A.4 dargestellt. Das referenzierte R-Sample ist jeweils identisch.

Szenario	R-Sample-Nummer	Anzahl plausibel	n^x
A-1	1	196	533
	2	329	
	3	35	
	4	129	
	5	331	
A-3	1	15	535
	2	20	
	3	18	
B-1	1	8	530
	2	149	
	3	139	

Abbildung 6.16: Ergebnisse der Plausibilisierung

6.3.3 Diskussion

6.3.3.1 Einordnung der Ergebnisse

Auf Basis der für die jeweiligen konkreten Szenarien definierten PFCs und ermittelten Grenzwerte für Szenariodistanzmaße ist die Plausibilisierung für alle R-Samples erfolgreich.

Aufgrund des Regens enthalten im Realtest alle fünf R-Samples in Szenario A-1 eine Kollision. In vergleichbaren Tests unter trockenen Bedingungen ist dies nicht der Fall. Aufgrund der Datenverfügbarkeit und der Möglichkeit, den Einfluss von Regen zu untersuchen, wurde die Plausibilisierung mit den Daten für Regen durchgeführt.

Die für die Szenariodistanzmaße herangezogenen Grenzwerte haben entscheidenden Einfluss auf die Ergebnisse der Plausibilisierung. Ebenso sind sie ein Maß für den durch die Unsicherheitsmodelle in den jeweiligen Szenarien hervorgerufenen Nichtdeterminismus. Der Parameter Ego-Geschwindigkeit auf $d_{1,max}$ und $d_{2,max}$ führt besonders bei einer hohen Geschwindigkeit (80 km/h) zu einem höheren Grenzwert. Daneben steigt durch das Einlenkmanöver in Szenario B-1 der Wert von $d_{3,max}$.

Die beim automatisierten Zuschnitt der aufgezeichneten Daten angewendeten szenarioabhängigen Start- und Endbedingungen sind insbesondere bei $d_{1,max}$ und $d_{2,max}$ von Relevanz, da hier ein zeitlicher Mittelwert berechnet wird. Fahrzustände, die eine geringe Unsicherheit der Fahrdynamik beinhalten (z.B. konstante Geradeausfahrt), werden je nach Zuschnitt unterschiedlich stark gewichtet gegenüber Fahrzuständen mit höherer Unsicherheit (z.B. Notbremsung).

Im Vergleich zu den aus Realdaten abgeleiteten Grenzwerten zeigt sich, dass die über MC-Samples ermittelten Grenzwerte in einer ähnlichen Größenordnung liegen. Die Divergenz bei Grenzwert $d_{1,max}$ kann mit der deutlich kleineren Anzahl der R-Samples und der dementsprechend kleineren Varianz der auftretenden Testergebnisse erklärt werden.

Auch wenn die Plausibilisierung für alle R-Samples erfolgreich ist, lässt die unterschiedliche Anzahl plausibler MC-Samples je R-Sample Rückschlüsse auf die Glaubwürdigkeit der Simulation zu. In den Szenarien A-1 und B-1 existieren jeweils mehrere R-Samples, die eine hohe (dreistellige) Anzahl plausibler MC-Samples besitzen. Für A-3 beträgt der maximale Wert plausibler MC-Samples lediglich 20. Dies deutet darauf hin, dass es eine systematische Abweichung der Trajektorien bei der Simulationsdurchführung geben könnte, welche durch die Unsicherheitsmodelle nicht kompensiert wird.

Ebenfalls auffällig ist die vergleichsweise niedrige Anzahl von 8 plausiblen MC-Samples für R-Sample 1 in Szenario B-1. Der Grund ist, dass im Realtest bei diesem R-Sample Testergebnis $T1$ vorliegt und dieses in der SiL-Umgebung seltener erreicht wird als $T2$. Dies bestätigt den Nichtdeterminismus im Realtest, was durch Unsicherheitsmodelle nachgebildet werden kann.

Die in den Plots Abb. A.1 bis Abb. A.4 dargestellten Kombinationen von R-Sample und MC-Sample weisen in beiden Fällen eine Übereinstimmung hinsichtlich PFCs bzw. Testergebnis auf. Entscheidend dafür, dass die zweite Kom-

bination als nicht plausibel klassifiziert wird, ist die Überschreitung des Grenzwerts für d_2. Letzteres ist auf die Abweichung in der Ego-Geschwindigkeit zurückzuführen. Diese steigt gegen Ende des Szenarios kontinuierlich an (siehe Abb. A.4), sodass sich zum Zeitpunkt der Kollision eine deutlich höhere Ego-Geschwindigkeit ergibt (siehe Abb. A.3). Für das in Abb. A.1 und Abb. A.2 dargestellte Szenario befinden sich alle Szenariodistanzmaße unterhalb der jeweiligen Grenzwerte.

6.3.3.2 Bewertung der Methode

Für die Definition objektiver Grenzwerte für Szenariodistanzmaße existiert kein standardmäßiges Vorgehen, welches unmittelbar angewendet werden kann. Erforderlich hierfür sind objektive quantitative Label für die Ähnlichkeit von Szenarien (siehe auch [326]). Eine manuelle Klassifikation der Szenarioähnlichkeit ist hierfür nur bedingt geeignet. Stattdessen werden hier Testergebnisse als Label verwendet, um ähnliche MC-Samples zu aggregieren (siehe 5.4.2). Folgende Analyse zeigt den Zusammenhang von Szenariodistanzmaß und Testergebnis: Die mittlere Distanz zu MC-Samples von Szenario A-1 mit gleichem Testergebnis ist entweder ähnlich groß (d_3) oder kleiner (d_1 und d_2) als zu allen verbleibenden Szenarien (siehe Abb. 6.17). Eine signifikante Abhängigkeit vom Testergebnis für d_3 liegt beispielsweise bei Szenario B-1 vor.

Szenariodistanzmaß		T2	T3	T4
d_1	$\overline{d(T_i, T_i)}$	0.667	0.638	1.700
	$\overline{d(T_i, T_{j\neq i})}$	3.696	3.490	3.680
d_2	$\overline{d(T_i, T_i)}$	0.503	0.515	0.712
	$\overline{d(T_i, T_{j\neq i})}$	1.465	1.296	1.443
d_3	$\overline{d(T_i, T_i)}$	0.00449	0.00392	0.00268
	$\overline{d(T_i, T_{j\neq i})}$	0.00362	0.00339	0.00358

Abbildung 6.17: Übersicht Mittelwert Szenariodistanzmaße je Testergebnis für Szenario A-1

Mithilfe der vorgestellte Methode ist es möglich, objektive Grenzwerte individuell für jedes Szenario zu ermitteln. Schwerpunkt ist die Auswirkung der Szenariodistanz auf mögliche Testergebnisse. Für andere Anwendungen besteht weiterhin Forschungsbedarf. Aufgrund der Übertragbarkeit in physikalische Größen wie Relativdistanz, Ego-Geschwindigkeit und Ego-Gierwinkel kommt auch eine manuelle Definition von Grenzwerten auf Basis von Expertenwissen infrage.

Bei der Plausibilisierung findet ein kombinierter qualitativer und quantitativer Abgleich zweier Samples eines konkreten Szenarios statt (siehe Abschnitt 5.4). Während E_1 unmittelbar das Testergebnis betrachtet, bezieht sich E_2 auf das Zustandekommen bzw. den zugrundeliegenden Mechanismus, welche durch die analysierten Trajektorieninformationen repräsentiert werden. Für beide Kriterien stellt die Auswahl von PFCs und Szenariodistanzmaßen einen manuellen Prozess dar, der Expertenwissen mit einbezieht. Darüber werden bei jeder Auswahl stets nur Teile der vorhandenen Informationen betrachtet. Von Interesse für zukünftige Forschungsaktivitäten ist die Frage, inwiefern sich die Auswahl standardisieren und objektivieren lässt. Ebenso ist die Validierung von Szenariodistanzmaßen eine noch zu klärende wissenschaftliche Fragestellung ([326]). Die Übertragbarkeit der Plausibilisierung auf andere SUTs bzw. Szenarien ist gegeben, sofern die PFCs und Szenariodistanzmaße entsprechend angepasst werden.

Ein möglicher Schritt wäre die Definition von Szenarien mit PFCs, Szenariodistanzmaßen und entsprechenden Grenzwerten in einem Standard bzw. einer Norm. Auf dieser Basis wäre ein objektiver Nachweis für die Plausibilisierung möglich.

Die Anzahl plausibler MC-Samples wird in der Glaubwürdigkeitsbewertung bislang nur qualitativ berücksichtigt. Für weiterführende Ansätze ist eine quantitative Berücksichtigung möglich.

Hinsichtlich der Übertragbarkeit von Szenariodistanzmaßen auf andere Anwendungsfelder wird auf [314] verwiesen.

6.4 Validierung

Die Validierung wird auf die Szenarien A-1, A-3 und B-1 angewandt. Für A-1 stehen fünf R-Samples zur Verfügung, während es bei A-3 und B-1 drei R-Samples sind. Die erforderliche Mindestanzahl an R-Samples für die Validierung wird somit in allen drei Fällen erreicht (siehe Abb. 6.9). Der exakte Test nach Barnard wird mithilfe des Python-Package scipy[5] [303] umgesetzt.

6.4.1 Ergebnisse

In beiden Szenarien ist der p-Wert des exakten Tests nach Barnard ausreichend groß für alle auftretenden Testergebnisse, sodass die SiL-Umgebung für alle drei betrachteten Szenarien als valide klassifiziert wird. In allen drei Szenarien wird die Mindestanzahl $n^r_{min} = 2$ (siehe Abb. 6.9, Abb. 6.11 und Abb. 6.13) erreicht.

Szenario		T1	T2	T3	T4	$V_{\omega,j}$
A-1	p_{T_i}	n/a	0.4377	0.8758	0.4700	1
A-3		0.9546	0.7237	0.8752	0.6553	1
B-1		1	1	0.6847	0.8875	1

Abbildung 6.18: Ergebnisse der Validierung

6.4.2 Diskussion

6.4.2.1 Einordnung der Ergebnisse

Alle drei betrachteten konkreten Szenarien werden als valide eingestuft, wobei für die Szenarien A-3 und B-1 alle p-Werte größer als 0.65 sind. Die beiden niedrigsten p-Werte treten bei A-1 für die Testergebnisse $T2$ und $T4$ auf, wobei dies weiterhin einen für eine erfolgreiche Validierung ausreichend hohen Wert

[5] https://docs.scipy.org/doc/scipy/index.html

darstellt. Begründen lässt sich dies einerseits mit der Existenz von 58 MC-Samples mit $T4$ und andererseits mit der höheren Anzahl der R-Samples im Vergleich zu den Szenarien A-3 und B-1. Die Anzahl der R-Samples bei B-1 führt dazu, dass für den p-Wert bei $T1$ und $T2$ der Wert 1 erreicht wird.

6.4.2.2 Bewertung der Methode

Ziel der Validierung ist der statistische Abgleich der Testergebnis-Verteilungen zwischen XiL und Realtest mithilfe des Barnard-Tests. Der p-Wert ist ein Maß dafür, wie wahrscheinlich eine noch größere Abweichung der Verteilungen von XiL und Realtest ist. Ziel für eine erfolgreiche Validierung ist ein möglichst hoher p-Wert. Der verwendete Grenzwert von 0.1 wurde exemplarisch gewählt als Kompromiss zwischen einer möglichst hohen Wahrscheinlichkeit für eine Annahme eines validen Szenarios (niedriger Alphafehler) und der ausreichend hohen Mindestanzahl an R-Samples. Es besteht weiteres Forschungspotential für einen systematischen und objektiven Ansatz zur Ermittlung eines Grenzwerts für p.

Der hier angewandte Ansatz bezieht sich ausschließlich auf die Statistik der Testergebnisse und ist somit von der Wahl der PFCs abhängig. Eine Übertragbarkeit auf neue SUTs und Szenarien ist gegeben, wenn PFCs entsprechend gewählt werden. Darüber hinaus ist eine statistische Berücksichtigung der plausiblen MC-Samples oder der Szenariodistanzmaße als Erweiterung der bestehenden Ansätze möglich.

6.5 Diskussion der Glaubwürdigkeitsbewertung

Dieses Kapitel zeigt die grundsätzliche Anwendbarkeit der Glaubwürdigkeitsbewertung in einem Entwicklungsprojekt. Die Ergebnisse der Prädiktion zeigen für alle sieben Szenarien ein zu erwartendes Systemverhalten. Für die drei detaillierter betrachteten Szenarien sind die Schritte Plausibilisierung und Validierung erfolgreich.

Referenz für die Glaubwürdigkeitsbewertung ist stets der Realtest auf dem Prüfgelände. Für die Übertragbarkeit der Ergebnisse auf Feldtests ist daher zu bewerten, welche weiteren Einflüsse dies auf die Glaubwürdigkeit hat.

Potentielle Unsicherheiten existieren unter anderem durch Prüfmittel (Soft Targets) oder dem Verhalten des Fahrers oder anderer Verkehrsteilnehmer [313]. Diese Aspekte werden in dieser Dissertation nicht spezifisch betrachtet, bieten jedoch weiteres Forschungspotential.

Eine Herausforderung bei der Glaubwürdigkeitsbewertung ist die begrenzte Verfügbarkeit von Realdaten. Sowohl Szenariovarianz als auch die im Nutzfahrzeug im Speziellen auftretende Fahrzeugvarianz führen zu ggf. hohen Realtestaufwänden. Aus diesem Grund werden die in Abschnitt 5.2 referenzierten Konfidenzstufen eingeführt, sodass bereits eine Abstufung der erforderlichen Realtestumfänge vorgenommen werden kann. Neben dem in dieser Dissertation vorgestellten Vorgehen in Anlehnung an die ISO 26262 ist es möglich, Konfidenzstufen auf Basis von Expertenwissen oder mit einem anderen objektiven und datengetriebenen Verfahren festzulegen. Unter den in Abschnitt 5.5 getroffenen Annahmen kann eine Validierung mit maximal vier R-Samples vorgenommen werden, was eine realisierbare Anzahl darstellt.

Von Bedeutung für ein Entwicklungsprojekt ist der Umgang mit unterschiedlichen Reifegraden von SUT, Sensoren, Steuergeräten oder Aktorik. Für die Plausibilisierung und Validierung muss sichergestellt sein, dass XiL und Realtest konsistente Software- und Hardwarestände verwenden, da anderenfalls kein Abgleich von XiL und Realtest möglich ist. Im Laufe eines Entwicklungsprojekts müssen daher Simulationsmodelle inklusive Unsicherheitsmodelle kontinuierlich aktualisiert werden. Bei der Prädiktion ist dies nicht zwingend notwendig, da hier kein direkter Abgleich erfolgt. Hingegen ist es möglich, bewusst Varianz hinsichtlich Software bzw. Hardware-Modellen zu modellieren, um deren Auswirkung auf die Testergebnis-Verteilungen zu evaluieren.

Die a-priori Prädiktion der Konfidenz wäre ein ergänzender bzw. alternativer Ansatz im Vergleich zur vorgestellten a-posteriori Glaubwürdigkeitsbewertung. Eine Folge wäre eine Anpassung des Prozesses aus Abschnitt 4.3. Möglich wäre dann die Verwendung eines kontinuierlichen Maßes für die Konfidenz im Gegensatz zur den hier verwendeten diskreten Konfidenzstufen.

Die Glaubwürdigkeitsbewertung kann über den Bereich der Fahrautomatisierung hinaus auch in anderen Feldern angewendet werden. Voraussetzung ist, dass es möglich ist, szenariobasiert zu testen und dass geeignete PFCs sowie Szenariodistanzmaße existieren. Letztere erfordern entsprechendes Expertenwissen in der jeweiligen Domäne.

7 Zusammenfassung und Ausblick

7.1 Zusammenfassung

7.1.1 Überblick

Mit steigender Anzahl von Fahrerassistenzsystemen sowie höheren Automatisierungslevels stoßen bestehende Test- und Absicherungsmethoden an ihre Grenzen. Lag bislang der Schwerpunkt auf anforderungsbasierten Realtests auf dem Prüfgelände oder der Erprobung im Feld, so muss dieses Vorgehen mit neuen effizienteren Methoden ergänzt werden. Eine besondere Rolle spielt dabei der szenariobasierte Ansatz, mithilfe dessen Testfälle effizient definiert und skaliert werden können, was potentiell redundante Testaktivitäten minimiert. Auf der Seite der Testumgebungen bieten X-in-the-Loop-Umgebungen weiteres Potential zur Steigerung der Effizienz. XiL-Umgebungen stimulieren das System Under Test virtuell mithilfe von Simulationsmodellen und kommen bereits heute in der Serienentwicklung zum Einsatz. Aufgrund der Möglichkeit, Testfälle deutlich effizienter zu skalieren als im Realtest, gewinnen XiL-Umgebungen weiter an Bedeutung. Gerade die Kombination von szenariobasiertem Testen und XiL stellt einen vielversprechenden Weg dar, um den Herausforderungen beim Testen von Fahrautomatisierung zu begegnen und stellt den Aufsatzpunkt dieser Dissertation dar. Darauf basierend werden in Kapitel 1 vier Forschungsfragen definiert, die in diesem Kapitel final diskutiert werden.

Forschungsfrage 1 evaluiert, wie der szenariobasierte XiL-Test in bestehende Entwicklungsprozesse integriert werden kann, um die Vorteile des Ansatzes zur Geltung zu bringen. Hierfür werden zunächst fünf Anforderungen definiert. Anschließend werden basierend auf einem in den Grundlagen eingeführten Entwicklungsprozess für Fahrerassistenzsysteme drei Einsatzmöglichkeiten des szenariobasierten XiL-Test vorgestellt und anhand eines Beispielprozes-

ses beschrieben. Darüber hinaus werden jeweils die mögliche Repräsentation des SUT sowie mögliche XiL-Umgebungen je Einsatzmöglichkeit genannt. Durch Abgleich mit den zuvor definierten Anforderungen werden die Einsatzmöglichkeiten diskutiert und bewertet. Da die Einsatzmöglichkeit Software-Qualifizierungstest potentiell die höchste Anforderungserfüllung bietet, wird diese als Referenz für die folgenden Teile der Dissertation herangezogen und weiter detailliert.

In **Forschungsfrage 2** geht es um die effiziente Zuordnung konkreter Szenarien auf unterschiedliche Testumgebungen. Hierfür wird der für die Einsatzmöglichkeit Software-Qualifizierungstest definierte Beispielprozess weiter detailliert. Der Prozessentwurf sieht vor, dass konkrete Szenarien zunächst in der effizientesten verfügbaren XiL-Umgebung getestet werden. Anschließend erfolgt ein Clustering, um ähnliche Szenarien zu aggregieren und jeweils ein repräsentatives konkretes Szenario je Cluster zu identifizieren. Anschließend wird die Glaubwürdigkeit der XiL-Umgebung für jedes Cluster bewertet. Ergebnisse glaubwürdiger Cluster werden angenommen, während bei nicht glaubwürdigen Clustern ein erneuter Test in der effizientesten verbleibenden Testumgebung erfolgt.

Die im Prozessentwurf genannte Glaubwürdigkeitsbewertung ist Kern von **Forschungsfrage 3**. Als Maß für Glaubwürdigkeit werden in Einklang mit den Definitionen in 3.3.1 sogenannte diskrete Konfidenzstufen eingeführt. In einem Vorgehen ähnlich zur Bestimmung der ASILs in der ISO 26262 wird die für das jeweilige konkrete Szenario und die XiL-Umgebung erforderliche Konfidenzstufe bestimmt. Anschließend wird in einem dreistufigen Prozess die tatsächlich erreichte Konfidenzstufe ermittelt.

Bei der Prädiktion wird mithilfe von Unsicherheitsmodellen in der XiL-Umgebung und eines Monte-Carlo-Ansatzes die Unsicherheit zwischen XiL und Realtest sowie innerhalb des Realtests modelliert. Dies nimmt Bezug auf **Forschungsfrage 4**. Mögliche Unsicherheitsmodelle für Fahrzeug, Fahrer und Sensoren werden eingeführt. Der Einfluss der Unsicherheit wird anschließend anhand der Testergebnis-Verteilungen der Monte-Carlo(MC)-Samples sowie entsprechender Konfidenzintervalle auf Basis einer Multinomialverteilung quantifiziert. Für die Prädiktion wird eine Mindestanzahl von 510 MC-Samples abgeschätzt. Wird lediglich die Prädiktion durchgeführt, so ist die Konfidenzstufe K1 erreicht.

Als zweiter Schritt schließt die Plausibilisierung an. Hier wird zusätzlich mindestens ein Realtest(R)-Samples aus dem Realtest benötigt. Es erfolgt ein Abgleich jedes vorhandenen R-Samples mit jedem MC-Sample. Mithilfe von Pass/Fail-Kriterien und Szenariodistanzmaßen wird hierbei auf Äquivalenz geprüft. Existiert für jedes R-Sample mindestens ein äquivalentes MC-Sample, ist die Plausibilisierung insgesamt erfolgreich. Bei erfolgreicher Plausibilisierung wird die Konfidenzstufe K2 erreicht, im Fall einer nicht erfolgreichen Plausibilisierung K0.

Der dritte mögliche Schritt ist die Validierung. Voraussetzung hierfür ist das Vorhandensein einer Mindestanzahl von R-Samples. Mithilfe des exakten Tests nach Barnard findet ein statistischer Abgleich der Testergebnis-Verteilungen aus XiL-Test und Realtest statt. Die Validierung ist erfolgreich, wenn über den Barnard-Test kein statistisch signifikanter Einfluss der Testumgebung auf das Testergebnis nachgewiesen werden kann. Bei 510 MC-Samples sind im „Worst Case“ vier R-Samples für die Validierung erforderlich. Eine erfolgreiche Validierung führt zu einer Konfidenzstufe K3, während eine nicht erfolgreiche Validierung ebenfalls Konfidenzstufe K0 zur Folge hat.

Zur Evaluation der Glaubwürdigkeitsbewertung wird diese im Umfeld eines Serienentwicklungsprojekts für Nutzfahrzeuge angewendet. Zum Einsatz kommt hierbei eine SiL-Umgebung zum Test eines Steuergeräts für Fahrerassistenzsysteme als SUT. Insgesamt werden sieben Szenarien berücksichtigt, wobei bei drei die Plausibilisierung und Validierung durchgeführt wird. Während in zwei dieser drei Szenarien das SUT das erwünschte Systemverhalten (Bremseingriff und/oder keine Kollision) in der überwiegenden Mehrheit der MC-Samples aufweist, führt der Einfluss von Regen in einem Szenario zu einer Mehrzahl an MC-Samples mit einer unerwünschten Kollision. In allen drei Fällen deckt sich das Verhalten mit dem Realtest und die XiL-Umgebung wird für die jeweiligen konkreten Szenarien als valide klassifiziert. Die Anwendbarkeit der Glaubwürdigkeitsbewertung in einem Entwicklungsprojekt kann somit demonstriert werden. Für die Nutzung der Ansätze mit weiteren SUTs oder Szenarien sind Pass/Fail-Kriterien und Szenariodistanzmaße ggf. anzupassen.

7.1.2 Wissenschaftlicher Beitrag

Aufbauend auf dem Stand der Wissenschaft und Technik werden in dieser Dissertation Prozesse und Methoden vorgestellt, um den szenariobasierten X-in-the-Loop-Test möglichst wirkungsvoll bei der Entwicklung von Fahrerassistenzsystemen einzusetzen. Elementarer Bestandteil ist die Glaubwürdigkeitsbewertung einer XiL-Umgebung.

Kapitel 4 zeigt auf, wie der szenariobasierte XiL-Test in einen bestehenden Entwicklungsprozess integriert werden kann. Hierfür wird eine systematische und anforderungsbasierte Bewertungsmethodik für mögliche Anwendungsfälle eingeführt. In einem Prozessentwurf für die Zuordnung von konkreten Szenarien auf Testumgebungen ist erstmals die Kombination eines Clustering-Ansatzes mit einer anschließenden Glaubwürdigkeitsbewertung vorgesehen.

Für die Glaubwürdigkeitsbewertung (siehe Kapitel 5) wird der in der wissenschaftlichen Diskussion im Kontext XiL für Fahrautomatisierung häufig verwendete Begriff der Validierung abgegrenzt und als Prozessschritt definiert. Neu eingeführt werden die Prozessschritte Prädiktion und Plausibilisierung als Vorstufen bzw. Voraussetzungen zur Validierung. Diskrete Konfidenzstufen werden als Maß für die Glaubwürdigkeit festgelegt und ermöglichen eine szenarioabhängige Anforderungsdefinition für Glaubwürdigkeit.

In der Prädiktion erlaubt die Kombination und Integration von Unsicherheitsmodellen eine Berücksichtigung von Einflüssen verschiedener Unsicherheitsquellen einer XiL-Umgebung in aggregierter Form. Die anschließende Analyse auf Basis einer Multinomialverteilung berücksichtigt die statistische Unsicherheit der Testergebnis-Verteilungen.

Die Plausibilisierung erweitert bestehende Methoden des Vergleichs von XiL und Realtest, insbesondere durch die systematische Verwendung von Pass/Fail-Kriterien und Szenariodistanzmaßen.

Für die Validierung werden statistische Effekte auf das Testergebnis aufgrund von Unsicherheiten in den Testumgebungen mithilfe des Barnard-Tests berücksichtigt.

7.2 Ausblick

In Abschnitt 4.2 werden drei Einsatzmöglichkeiten für den szenariobasierten XiL-Test vorgestellt. Analog zum Prozessentwurf für den Software-Qualifikationstest ist auch eine Detaillierung der verbleibenden Einsatzmöglichkeiten sinnvoll. Darüber hinaus bietet der szenariobasierte XiL-Test weitere potentielle Einsatzmöglichkeiten. Hierbei stellt sich insbesondere die Frage, welche Teile des Systems als SUT in welcher Testumgebung getestet werden. Für den Test des Subsystems Sense ist zu bewerten, inwiefern synthetische Daten aus Sensormodellen nutzbar sind. Ein geschlossener Regelkreis ist hingegen nicht zwingend erforderlich in diesem Fall. Eine weitere zu lösende Herausforderung ist, eine Durchgängigkeit in der Szenario- und Testfallbeschreibung zwischen verschiedenen Testumgebungen sicherzustellen. Dies erfordert eine konsistente Verwendung von Schnittstellen und Modellen.

Der Prozessentwurf aus Abschnitt 4.3 stellt einen Startpunkt dar, erfordert jedoch die Detaillierung einzelner Schritte. Fokus dieser Dissertation liegt auf der Glaubwürdigkeitsbewertung. Hingegen ist das Vorgehen bei der Auswahl logischer Szenarien, der Generierung konkreter Szenarien, dem Clustering und der Ergebnisaggregation nicht vollständig spezifiziert. Während insbesondere für die beiden erstgenannten Schritte Forschungsaktivitäten und -ansätze existieren, bietet der Schritt Clustering hierfür besonderes Potential. Vor allem die Frage der geeigneten Szenariodistanzmaße sollte weiter untersucht werden. Bei der Ergebnisaggregation ist es essentiell, geeignete Metriken für die Bewertung des Systems sowohl lokal für einzelne logische Szenarien als auch auf globaler Ebene zu definieren. Bestehende Metriken und Pass/Fail-Kriterien stellen hierfür einen Startpunkt dar. Für eine aussagekräftige Analyse besteht jedoch weiterhin Forschungsbedarf im Bereich Pass/Fail-Kriterien, Metriken und deren Aggregation.

Grundlage der in dieser Dissertation vorgestellten Glaubwürdigkeitsbewertung sind aufgezeichnete Daten aus XiL und Realtest auf dem Prüfgelände (siehe Abschnitt 6.5). Für die bessere Übertragbarkeit auf Feldtests wäre die Betrachtung des Einflusses von Unsicherheiten im Prüfgelände (z.B. durch Prüfmittel oder Fahrerverhalten) ein möglicher weiterer Schritt. Ebenfalls von Relevanz ist die a-priori Prädiktion der Konfidenz als Ergänzung oder Alternative zum vorgestellten a-posteriori Ansatz. Eine Anwendung der Glaubwürdigkeitsbe-

wertung auf weitere Systeme, höhere Automatisierungslevels oder alternative Szenarien ist ebenfalls möglich.

Für die Prädiktion haben die verwendeten Unsicherheitsmodelle entscheidenden Einfluss auf den Erfolg der Methode (siehe 6.2.3.2). Hier besteht weiterhin Forschungsbedarf, insbesondere bei der Übertragbarkeit der Parametrierung von Unsicherheitsmodellen auf neue Szenarien. Ansätze basierend auf Maschinellem Lernen sind vielversprechend, erfordern aber einen größeren Modellierungsaufwand [325]. Darüber hinaus sind Unsicherheitsmodelle für weitere Komponenten bzw. Subsysteme von Bedeutung, da in dieser Dissertation aufgrund der Umsetzbarkeit lediglich drei Subsysteme betrachtet werden.

Die Plausibilisierung verwendet Pass/Fail-Kriterien und Szenariodistanzmaße (siehe 6.3.3.2). Forschungsbedarf besteht bei der standardisierten und objektiven Auswahl von Pass/Fail-Kriterien und Szenariodistanzmaßen. Auch die Validierung von Szenariodistanzmaßen [326] und die Definition der jeweiligen Grenzwerte sind noch offene Forschungsfragen. Darüber hinaus wäre eine quantitative Berücksichtigung der Anzahl plausibler MC-Samples eine mögliche Erweiterung der Plausibilisierung. Die Auswahl von Pass/Fail-Kriterien, Szenariodistanzmaßen und deren Grenzwerte ließe sich auch in einer Norm bzw. einem Standard berücksichtigen.

Bei der Validierung werden Testergebnis-Verteilungen basierend auf dem Barnard-Test evaluiert (siehe 6.4.2.2). Für den dabei verwendeten Grenzwert des p-Werts besteht weiterer Forschungsbedarf hinsichtlich einer systematischen und objektiven Methode. Zudem ließe sich die Validierung mit weiteren Kriterien bzw. statistischen Größen wie z.B. der Anzahl plausibler MC-Samples oder Szenariodistanzmaßen ergänzen.

Grundsätzlich ist die Anwendung des szenariobasierten XiL-Tests auch über den Bereich der Fahrautomatisierung hinaus denkbar. Beispiele sind der Test von Antriebsstrang-Systemen oder Komfort-/Body-Systeme im Fahrzeug. In diesen Fällen ist zu prüfen, welche Anpassungen bei den Unsicherheitsmodellen und Szenariodistanzmaßen zu treffen sind.

A Anhang

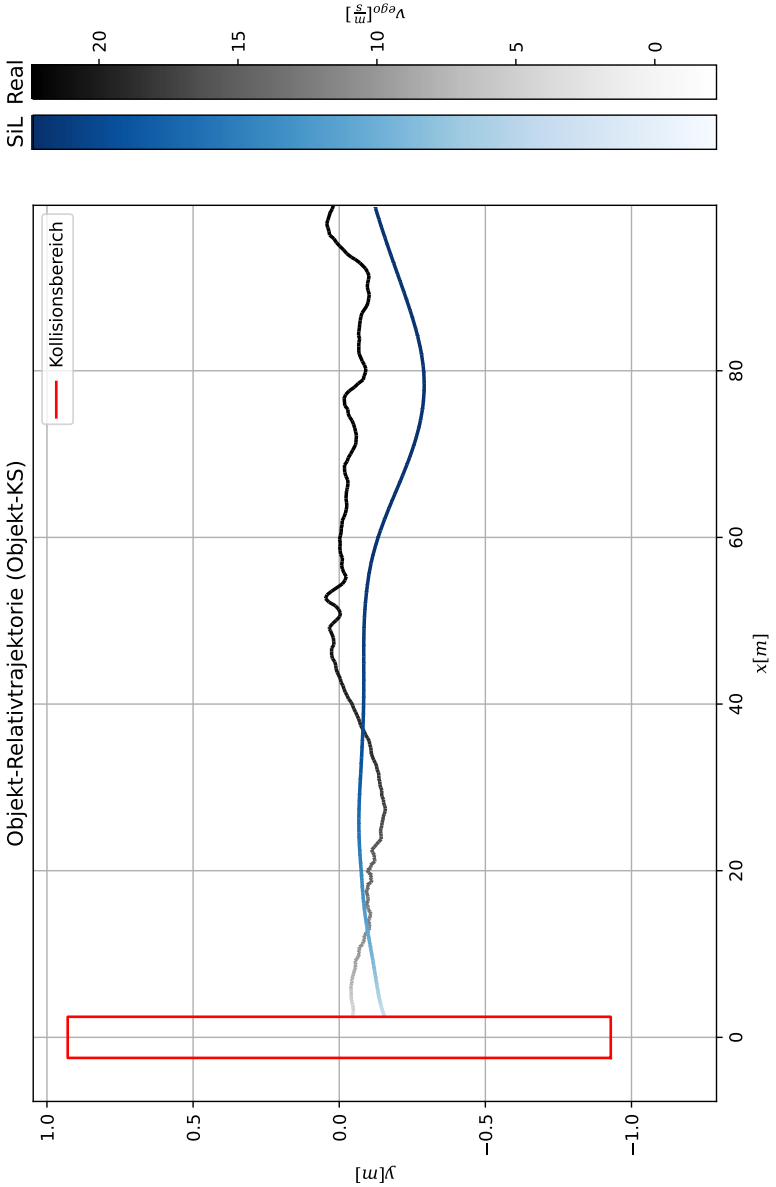

Abbildung A.1: Trajektoriendaten einer plausiblen Sample-Kombination (A-1)

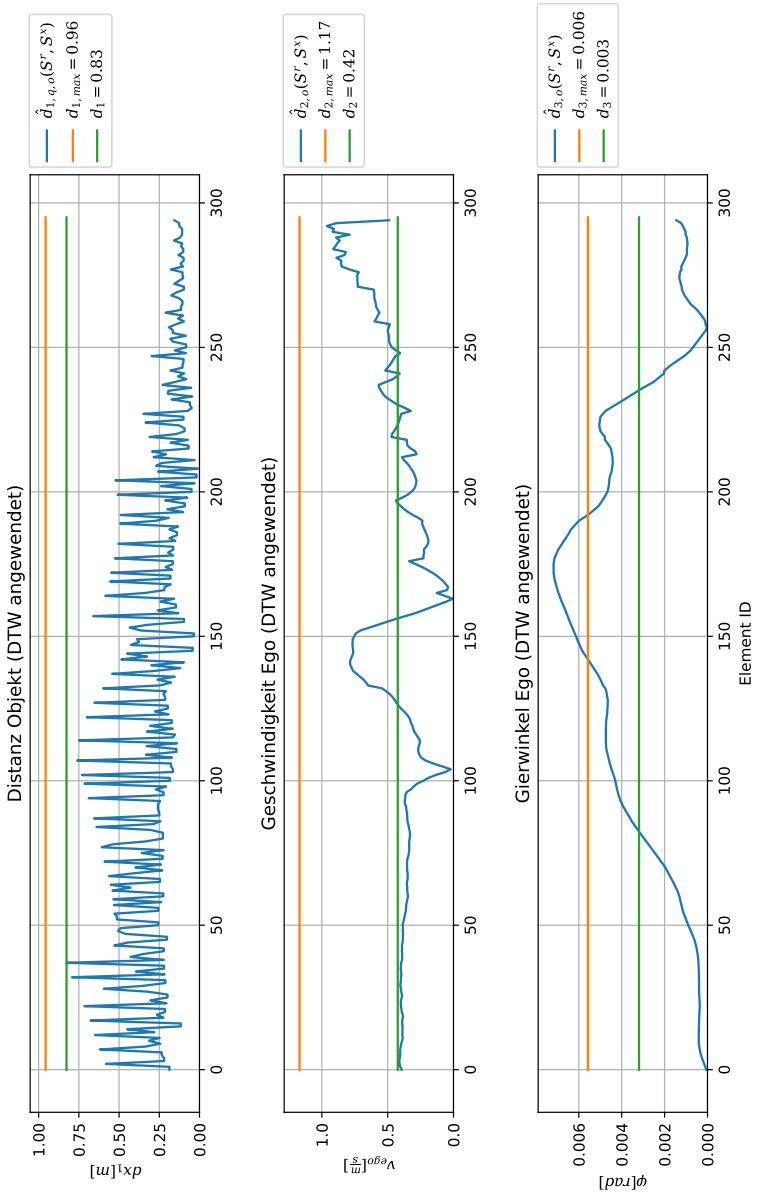

Abbildung A.2: Szenariodistanzmaße einer plausiblen Sample-Kombination (A-1)

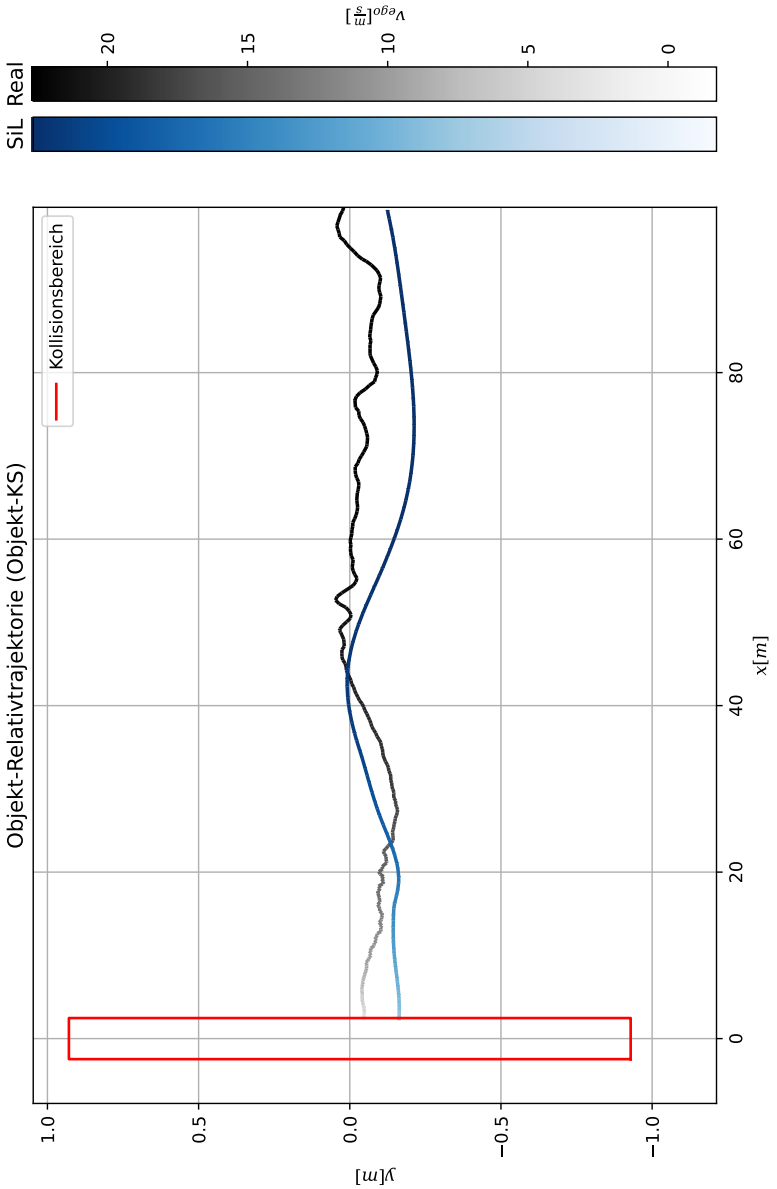

Abbildung A.3: Trajektoriendaten einer nicht plausiblen Sample-Kombination (A-1)

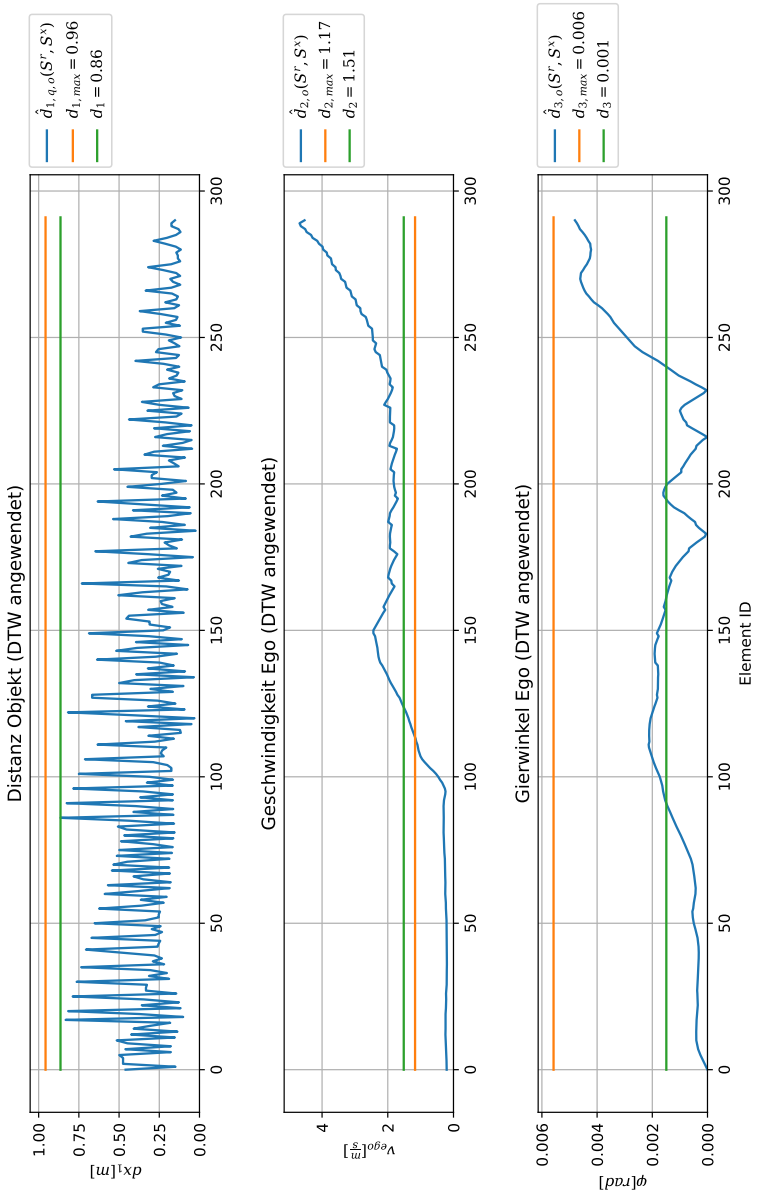

Abbildung A.4: Szenariodistanzmaße einer nicht plausiblen Sample-Kombination (A-1)

B Abkürzungen und Symbole

Abkürzungen

ABA	Active Brake Assist
ACC	Adaptive Cruise Control
ADS	Automated Driving System
AEBS	Advanced Emergency Braking System
AIS	Adaptive Importance Sampling
ASGA	Active Sideguard Assist
ASIL	Automotive Safety Integrity Level
BSIS	Blind Spot Information System
BTN	Brake Threat Number
CAN	Controller Area Network
CDF	Cumulative Distribution Function
CSV	Comma-separated values
DAS	Driving Automation System
DCE	Distance-of-Closest-Encounter
DiL	Driver-in-the-Loop
DDT	Dynamic Driving Task

DTW	Dynamic Time Warping
E/E	Elektrik/Elektronik
ECU	Electronic Control Unit
ESP	Elektronisches Stabilitätsprogramm
FAS	Fahrerassistenzsystem
FBM	Feature-based Metric
FMI	Functional Mock-up Interface
FMU	Functional Mock-up Unit
FTFI	Failure-Triggering Fault Interaction
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
GSR	General Safety Regulation
GUI	Graphical User Interface
HD	High-Definition
HiL	Hardware-in-the-Loop
HW	Hardware
ISPUD	Importance Sampling using Design Points
ITIV	Institut für Technik der Informationsverarbeitung
KIT	Karlsruher Institut für Technologie
KPI	Key Performance Indicator
LDWS	Lane Departure Warning System
LKW	Lastkraftwagen

MDF	Measurement Data File
MiL	Model-in-the-Loop
ODD	Operational Design Domain
OEDR	Object and Event Detection and Response
OEM	Original Equipment Manufacturer
OSI	Open Simulation Interface
PAM	Process Assessment Model
PDF	Probability Density Function
PET	Post-Encroachment-Time
PFC	Pass/Fail-Criterion
PiL	Processor-in-the-Loop
PKW	Personenkraftwagen
PRM	Process Reference Model
QM	Quality Management
RCS	Radar Cross Section
RTE	Run-Time Environment
SAE	Society of Automotive Engineers
SiL	Software-in-the-Loop
SLAM	Simultaneous Localization and Mapping
SOTIF	Safety of the Intended Functionality
SRQ	System Response Quantity
SUT	System Under Test

SW	Software
TDM	Time-domain Metric
TTB	Time-to-Brake
TTC	Time-to-Collision
TTCE	Time-to-Closest-Encounter
TTEC	Time-to-Edge-Crossing
TTK	Time-to-Kickdown
TTLC	Time-to-Lane-Crossing
TTR	Time-to-React
TTS	Time-to-Steer
V2X	Vehicle-to-X
ViL	Vehicle-in-the-Loop
VDA	Verband der Automobilindustrie
WTTC	Worst-Time-to-Collision
XiL	X-in-the-Loop
XML	Extensible Markup Language

Glossar

Äquivalenz
Resultat des Vergleichs zweier Szenarien und Voraussetzung für eine erfolgreiche Plausibilisierung.

Feature-based Metric (FBM)
Aus SRQs und TDMs abgeleitete Metrik, die keine zeitliche Abhängigkeit besitzt.

Key Performance Indicator (KPI)
Schlüsselkennzahl, die SRQ, TDM, FBM oder PFC sein kann.

Konfidenzstufe
Diskretes Maß (K0 bis K3) für die Konfidenz einer XiL-Umgebung.

MC-Sample
Ausführung eines identischen konkreten Szenarios in einer XiL-Umgebung basierend auf Monte-Carlo-Sampling.

Pass/Fail-Criterion (PFC)
Binäre Größe, die eine qualitative Aussage über den Ausgang eines Tests besitzt.

Plausibilisierung
Abgleich eines R-Samples mit einem MC-Samples zur Prüfung von Äquivalenz.

Prädiktion
Abschätzung von Multinomial-Konfidenzintervallen möglicher Testergebnisse, die über ein Monte-Carlo-Sampling in einer XiL-Umgebung generiert wurden.

R-Sample
Ausführung eines identischen konkreten Szenarios im Realtest.

System Response Quantity (SRQ)
Größe, die in einem Test direkt messbar ist.

Szenariodistanzmaß
Distanzmaß zum quantitativen datenbasierten Vergleich zweier Szenarien.

Testbreite
Anzahl logischer Testfälle pro System.

Testergebnis
Zusammenfassung aller PFCs in einem Vektor.

Testtiefe
Anzahl nicht redundanter konkreter Testfälle pro logischem Testfall.

Time-domain Metric (TDM)
Aus SRQs abgeleitete Metrik, die eine zeitliche Abhängigkeit besitzt.

Validierung
Statistischer Abgleich der Testergebnis-Verteilungen aller R-Samples und MC-Samples mithilfe des Barnard-Tests.

Formelzeichen

a_a	Anforderungsabdeckung
a_b	Anteil bekannter logischer Szenarien
$abaFB$	PFC für ABA Vollbremsung
$abaPB$	PFC für ABA Teilbremsung
$abaW$	PFC für ABA Warnung
$asgaB$	PFC für ASGA Bremsung
$asgaI$	PFC für ASGA Information
$asgaW$	PFC für ASGA Warnung
$a_{err,lon}(t)$	Beschleunigungsfehler longitudinal im Ego-Koordinatensystem
$a_{lon}(t)$	Longitudinale Beschleunigung nach Manipulation im Ego-Koordinatensystem
a_{max}	Maximal mögliche Beschleunigung
a_{thMax}	Schwellwert für Beschleunigung
$a_{pre,lon}(t)$	Longitudinale Beschleunigung vor Manipulation im Ego-Koordinatensystem
a_q	Betrag der absoluten Objektbeschleunigung
a_r	Realtest-Anteil
a_{req}	Benötigte Beschleunigung zur Kollisionsvermeidung
a_t	Testabdeckung
a_u	Zuordnungspfad DTW
$aThTimeTh$	PFC für Dauer der Überschreitung von a_{thMax}
A	DTW-Sequenz

$A_i(w)$	Amplitude der periodischen Unsicherheit i
$A_{err,lon}(t,s)$	Unsicherheit der Beschleunigung longitudinal im Ego-Koordinatensystem
B	DTW-Sequenz
b_v	Zuordnungspfad DTW
BTN	Brake Threat Number
c	Kostenfunktion
C	Coverage Toleranzintervall
$C(t)$	Kovarianzmatrix Gauß-Prozess
$C_{f,\omega}$	Fixe Kosten pro XiL-Umgebung
$C_{t,\omega}$	Effektive Kosten pro Testfall
$C_{v,\omega}$	Variable Kosten pro Testdurchführung
$CI_{i,\omega,j}$	Konfidenzintervall für Testergebnis i, XiL-Umgebung ω und Cluster j
d_{ges}	Gefahrene Distanz
d	Szenariodistanzmaß
d_{min}	Minimales Szenariodistanzmaß zwischen zwei Clustern
d_k	Szenariodistanzmaß k
$d_{k,max}$	Grenzwert für Szenariodistanzmaß k
d_q	Relativdistanz zu Objekt q
E	Äquivalenz Samples gesamt
E_1	Äquivalenz Samples Kriterium 1
E_2	Äquivalenz Samples Kriterium 2
$f_o(i_1,...,i_n)$	Funktion mit Inputs i_1 bis i_n und Output o

FBM	Feature-based Metric
$G(T_i)$	Kontingenztafel
H_0	Nullhypothese
H_1	Alternativhypothese
$I(t)$	Einheitsmatrix
$k(t,t')$	Kovarianzfunktion Gauß-Prozess
k_{par}	Maximale Ausprägungen pro Parameter
$K_{\omega,j}$	Konfidenzstufe der XiL-Umgebung ω für Cluster j
K_{ω}	Konfidenzstufe der XiL-Umgebung ω
K_{min}	Minimal erforderliche Konfidenzstufe
KPI	Key Performance Indicator
m_i	Größe nach Manipulation Unsicherheitsmodell 1 (Sensoren)
$m_{i,pre}$	Größe vor Manipulation Unsicherheitsmodell 1 (Sensoren)
n_a	Anzahl durch ausgeführte Testfälle abgedeckte Anforderungen
$n_{a,ges}$	Anzahl Anforderungen gesamt
n_c	Anzahl konkreter Testfälle
n_c^r	Anzahl real durchgeführter konkreter Testfälle
$n_{c,\omega}^x$	Anzahl durchgeführter konkreter Testfälle in XiL-Umgebung ω
$n_{c,t}$	Anzahl bekannter getesteter nicht redundanter konkreter Testfälle
$n_{c,u}$	Anzahl bekannter ungetesteter nicht redundanter konkreter Testfälle
n_e	Anzahl Events
n_l	Anzahl logische Testfälle

$n_{l,b}$	Anzahl bekannter logischer Szenarien
$n_{l,u}$	Anzahl unbekannter logischer Szenarien
n^r_{min}	Minimale Anzahl R-Samples
n_{par}	Anzahl Parameter gesamt
n^s	Anzahl Systeme
n^t	Anzahl ausgeführte Testfälle
$n^{t,ges}$	Anzahl mögliche Testfälle
n^r	Anzahl R-Samples
n^r_i	Anzahl R-Samples mit Testergebnis i
n^r_{min}	Minimale Anzahl R-Samples
n^x	Anzahl MC-Samples
n^x_i	Anzahl MC-Samples mit Testergebnis i
n^x_{min}	Minimale Anzahl MC-Samples
$noColl$	PFC für Kollisionsvermeidung
$N_{c,t}$	Menge aller zuvor getesteten Szenarien
$o_{err,i}$	Offsetfehler i
$O_{err,i}(w)$	Offsetunsicherheit i
$p(S_{i,j})$	Wahrscheinlichkeit der Parameterkombination des konkreten Szenarios i
$p_j(T_i)$	Wahrscheinlichkeit von Testergebnis i für das repräsentative konkrete Szenario j
$p_{j,min}(T_i)$	Untere Schranke des Konfidenzintervalls für die Wahrscheinlichkeit von Testergebnis i für das repräsentative konkrete Szenario j

$p_{j,max}(T_i)$	Obere Schranke des Konfidenzintervalls für die Wahrscheinlichkeit von Testergebnis i für das repräsentative konkrete Szenario j
p_{T_i}	p-Wert für Testergebnis i
$p_{err,i}(t)$	Periodischer Fehler i
$P_{err,i}(t,w)$	Periodische Unsicherheit i
$P_{\omega,j}$	Plausibilität der XiL-Umgebung ω für Cluster j
PFC	Pass/Fail Criterion
q	Objekt
r_{ego}	Ego-Trajektorie
$r_{err,i}(t)$	Rauschfehler i
$R_{err,i}(t,w)$	Rauschunsicherheit i
R_i	Relevanz des logischen Szenarios i
r_q	Relativtrajektorie von Objekt q zu Ego-Fahrzeug
s	Inputs für Unsicherheitsmodell 2 (Fahrzeug)
$s_{err,i}$	Skalierungsfehler i
$S_{err,i}(w)$	Skalierungsunsicherheit i
S_i	Konkretes Szenario i
S_j	Repräsentatives konkretes Szenario für Cluster j
S^l	Logisches Szenario
S^r, S^r_m	Konkretes Realtest-Sample
S^x, S^x_n	Konkretes Monte-Carlo-Sample
$SRQ(t)$	System Response Quantity
t	Zeit bzw. Zeitschritt

t	Anzahl kombinierter Parameter bei t-weiser Abdeckung
t_b	Testbreite
t_{min}	Minimales t bei t-weiser Abdeckung
$t_I(C_j)$	Mittleres Zeitintervall zwischen zweifachem Eintreten eines beliebigen konkreten Szenarios des Clusters j
$t_I(S^l)$	Mittleres Zeitintervall zwischen zweifachem Eintreten des logischen Szenarios l
t_t	Testtiefe
$t_{thATimeMax}$	Schwellwert für Dauer der Überschreitung von a_{thMax}
T	Testergebnis
T_{ABA}	Testergebnis für ABA
T_{ACC}	Testergebnis für ACC
T_{ASGA}	Testergebnis für ASGA
T_i^r	Realtest-Testergebnis
T_i^x	XiL-Testergebnis
T_m^r	Testergebnis eines R-Samples
T_n^x	Testergebnis eines MC-Samples
$TDM(t)$	Time-domain Metric
TTC	Time-to-Collision
TTC_{min}	Minimale Time-to-Collision
$ttcTh$	PFC für Nichtunterschreiten von TTC_{min}
u	DTW-Zuordnungspfad
U_a	Aleatorische Unsicherheit
U_e	Epistemische Unsicherheit

U^{ges}	Gesamte Unsicherheit zwischen XiL und Realtest
U^{x}	Unsicherheit XiL
U^{r}	Unsicherheit Realtest
v	DTW-Zuordnungspfad
$v_{acc,set}$	ACC-Setzgeschwindigkeit
$v_{egoCollMax}$	Schwellwert für Geschwindigkeit des Ego-Fahrzeugs bei Kollision
$v_{err,lon}(t)$	Geschwindigkeitsfehler longitudinal im Ego-Koordinatensystem
$v_{err,lat}(t)$	Geschwindigkeitsfehler lateral im Ego-Koordinatensystem
$v_{err,x}(t)$	Geschwindigkeitsfehler x im Inertialkoordinatensystem
$v_{err,y}(t)$	Geschwindigkeitsfehler y im Inertialkoordinatensystem
v_{lon}	Geschwindigkeit x im longitudinal im Ego-Koordinatensystem
$v_{pre,x}(t)$	Geschwindigkeit x im Inertialkoordinatensystem vor Manipulation
$v_{pre,y}(t)$	Geschwindigkeit y im Inertialkoordinatensystem vor Manipulation
v_q	Betrag der absoluten Objektgeschwindigkeit
v_{soll}	Sollgeschwindigkeit des Fahrerreglers nach Manipulation
$v_{soll,pre}$	Sollgeschwindigkeit des Fahrerreglers vor Manipulation
$v_x(t)$	Geschwindigkeit x im Inertialkoordinatensystem nach Manipulation
$v_y(t)$	Geschwindigkeit y im Inertialkoordinatensystem nach Manipulation
$vEgoCollTh$	PFC für Überschreiten von $v_{egoCollMax}$
$dv_{lane,y}$	Relative Spurgeschwindigkeit relativ lateral zu Ego-Fahrzeug

$dv_{q,x}$	Relative Objektgeschwindigkeit longitudinal zu Ego-Fahrzeug
$dv_{q,y}$	Relative Objektgeschwindigkeit lateral zu Ego-Fahrzeug
$V(T_i)$	Validität von Testergebnis i
$V_{\omega,j}$	Validität der XiL-Umgebung ω für Cluster j
w	Zeitabhängige Eingänge Unsicherheitsmodell 1 (Sensoren)
$x, x(t)$	Position x im Inertialkoordinatensystem vor Manipulation
$x_{ego}(t)$	Einlenkposition nach Manipulation
$x_{ego,of}(t)$	Offset Einlenkposition
$x_{ego,pre}(t)$	Einlenkposition vor Manipulation
$x_{err}(t)$	Positionsfehler x im Inertialkoordinatensystem
$x_{pre}(t)$	Position x im Inertialkoordinatensystem nach Manipulation
dx_q	Relative Objektposition longitudinal zu Ego-Fahrzeug
x_{fov}	Länge des longitudinalen Sichtfelds für Objekte
X	Längsachse
$y, y(t)$	Position y im Inertialkoordinatensystem nach Manipulation
$y_{err}(t)$	Positionsfehler y im Inertialkoordinatensystem
y_{fov}	Breite des lateralen Sichtfelds für Objekte
$y_{pre}(t)$	Position y im Inertialkoordinatensystem vor Manipulation
dy_{lane}	Relative Spurdistanz lateral zu Ego-Fahrzeug
dy_q	Relative Objektposition lateral zu Ego-Fahrzeug
Y	Querachse
z	z-Wert Standardnormalverteilung
$z_{a,lon}(t)$	Hilfsvariable Gauß-Prozess Unsicherheit Beschleunigung longitudinal

$z_{\psi}(t)$	Hilfsvariable Gauß-Prozess Unsicherheit Gierrate
Z	Vertikalachse
α	Signifikanzniveau
γ	Tatsächlicher Wert (gegenüber Fehler)
γ^{r}	Wert Realtest
γ^{ref}	Referenzwert
γ^{x}	Wert XiL
δ	Kronecker-Delta
ε	Fehler
ε^{ges}	Gesamtfehler zwischen XiL und Realtest
ε^{r}	Fehler Realtest
ε^{x}	Fehler XiL
θ	Nickwinkel
κ_{lane}	Spurkrümmung
$\dot{\kappa}_{lane}$	Spurkrümmungsänderung
λ	Fehlerrate
μ_{a}	Mittelwert aleatorische Unsicherheit
μ_{e}	Mittelwert epistemische Unsicherheit
$\mu_{a,lon}$	Mittelwert der Unsicherheit des Beschleunigungsfehlers longitudinal
μ^{ges}	Mittelwert der gesamten Unsicherheit zwischen XiL und Realtest
$\mu_{\dot{\psi}}$	Mittwelwert der Unsicherheit des Gierratenfehlers
χ	Sensitivität gegenüber XiL-Unsicherheit

σ_a	Sigma aleatorische Unsicherheit
σ_e	Sigma epistemische Unsicherheit
$\sigma_{a,lon}$	Sigma der Unsicherheit des Beschleunigungsfehlers longitudinal
σ^{ges}	Sigma der gesamten Unsicherheit zwischen XiL und Realtest
$\sigma_{\dot{\psi}}$	Sigma der Unsicherheit des Gierratenfehlers
$\Sigma_{i,r}(z)^2$	Kovarianzmatrix der Rauschunsicherheit i
φ	Wankwinkel
$\psi_(t)$	Gierwinkel im Inertialkoordinatensystem nach Manipulation
$\psi_{err}(t)$	Gierwinkelfehler im Inertialkoordinatensystem
ψ_{lane}	Spurgierwinkel
$\psi_{pre}(t)$	Gierwinkel im Inertialkoordinatensystem vor der Manipulation
ψ_q	Verdrehung Objekte im Ego-Koordinatensystem
$\dot{\psi}(t)$	Gierrate im Inertialkoordinatensystem nach Manipulation
$\dot{\psi}_{err}(t)$	Gierratenfehler im Inertialkoordinatensystem
$\dot{\Psi}_{err}(t,s)$	Unsicherheit der Gierrate
ω	XiL-Umgebung
ψ	Gierwinkel
Ω	Menge aller XiL-Umgebungen
$\Omega_i(z)$	Kreisfrequenz der periodischen Unsicherheit i

Literaturverzeichnis

[1] Mercedes-Benz Group AG. (2023). “CASE – Intuitive Mobility.”, Adresse: https://group.mercedes-benz.com/innovation/case-2.html (besucht am 05. 05. 2023).

[2] W. Bernhart und S. Riederle, *Challenges abound, Ongoing crises call for a proactive approach*, Roland Berger, 2020.

[3] Daimler Truck AG, *Ziele für Daimler Truck als künftig eigenständiges Unternehmen*, Daimler Truck AG, Hrsg., Stuttgart, 20. Mai 2021.

[4] F. Meissner, K. Shirokinskiy und M. Alexander, *Computer on wheels, Disruption in automotive electronics and semiconductors*, Roland Berger, 2020.

[5] SAE, *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*, Warrendale, PA: SAE On-Road Automated Driving (ORAD) committee, 2021.

[6] J. Deichmann, E. Ebel, K. Heineke, R. Heuss, M. Kellner und F. Steiner, *Autonomous driving's future: Convenient and connected*, McKinsey & Company, 2023.

[7] Bundesministerium für Verkehr und digitale Infrastruktur, *Straßenverkehrs-Zulassungs-Ordnung*, StVZO, 2012.

[8] Daimler Truck AG, *Predictive Powertrain Control (PPC) - 10 questions and answers about the predictive cruise control from Mercedes-Benz Trucks*, Daimler Truck AG, Hrsg., Stuttgart, 19. Juni 2020.

[9] ——, (2021). “Der Actros: Hohe Sicherheit - Mercedes-Benz Trucks - Trucks you can trust”, Adresse: https://www.mercedes-benz-trucks.com/de_DE/models/new-actros/greater-safety.html (besucht am 11. 06. 2021).

[10] Europäisches Parlament, *Verordnung (EG) Nr. 661/2009 des Europäischen Parlaments und des Rates*, EG 661/2009, 2009.

[11] ——, *Verordnung (EU) 2019/2144 des Europäischen Parlaments und des Rates*, EG 2019/2144, 2019.

[12] R. Zarif, C. Starks, A. Sussman, A. Kukreja und S. Abidi, *Autonomous trucks lead the way, Many companies are shifting focus from R&D to making driverless models work at scale*, Deloitte Development LLC., 2021.

[13] D. A. Weitzel, *Absicherungsstrategien für Fahrerassistenzsysteme mit Umfeldwahrnehmung, Bericht zum Forschungsprojekt FE 82.0546/2012*, ger, Ser. Berichte der Bundesanstalt für Straßenwesen Fahrzeugtechnik. Bremen, 2014, Bd. 98.

[14] VDA QMC Working Group 13 / Automotive SIG, *Automotive SPICE, Process Reference Model Process Assessment Model*, 2017.

[15] ISO, Hrsg., *ISO 26262:2018*, Geneva: International Organization for Standardization, 2018.

[16] T. Gordon, D. LeBlanc, J. Sayer, C. Winkler, M. Hagan, S. Bogard und J. Devonshire, "Field Operational Tests-Evaluating Driver-Assistance Systems Under Real World Conditions", *SAE Technical Paper Series*, 2006.

[17] Tesla, *Tesla Autonomy Day*, Video, 2019. Adresse: `https://www.youtube.com/watch?v=Ucp0TTmvqOE` (besucht am 05. 05. 2023).

[18] S. Ulbrich, T. Menzel, A. Reschka, F. Schuldt und M. Maurer, "Defining and Substantiating the Terms Scene, Situation, and Scenario for Automated Driving", in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2015, S. 982–988. DOI: `10.1109/itsc.2015.164`.

[19] G. Bagschik, T. Menzel, A. Reschka und M. Maurer, "Szenarien für Entwicklung, Absicherung und Test von automatisierten Fahrzeugen", in *Workshop Fahrerassistenzsysteme und automatisiertes Fahren*, UniDAS e. V., Hrsg., 2017.

[20] C. King, L. Ries, J. Langner und E. Sax, "A Taxonomy and Survey on Validation Approaches for Automated Driving Systems", in *International Symposium on Systems Engineering (ISSE)*, IEEE, 2020, S. 1–8. DOI: `10.1109/ISSE49799.2020.9272219`.

[21] F. Hauer, T. Schmidt, B. Holzmuller und A. Pretschner, "Did We Test All Scenarios for Automated and Autonomous Driving Systems?", in *Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2019, S. 2950–2955. DOI: 10.1109/ITSC.2019.8917326.

[22] S. Moten, F. Celiberti, M. Grottoli, A. van der Heide und Y. Lemmens, "X-in-the-loop advanced driving simulation platform for the design, development, testing and validation of ADAS", in *Intelligent Vehicles Symposium (IV)*, IEEE, 2018, S. 1–6. DOI: 10.1109/IVS.2018.8500409.

[23] K. v. Neumann-Cosel, "Virtual Test Drive", Dissertation, TU München, 2014.

[24] S. Jesenski, J. E. Stellet, W. Branz und J. M. Zollner, "Simulation-Based Methods for Validation of Automated Driving: A Model-Based Analysis and an Overview about Methods for Implementation", in *Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2019, S. 1914–1921. DOI: 10.1109/ITSC.2019.8917072.

[25] F. Schuldt, "Ein Beitrag für den methodischen Test von automatisierten Fahrfunktionen mit Hilfe von virtuellen Umgebungen", Dissertation, TU Braunschweig, 2017.

[26] M. Steimle, N. Weber und M. Maurer, "Toward Generating Sufficiently Valid Test Case Results: A Method for Systematically Assigning Test Cases to Test Bench Configurations in a Scenario-Based Test Approach for Automated Vehicles", *IEEE Access*, Jg. 10, S. 6260–6285, 2022, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2022.3141198.

[27] P. Rosenberger, J. T. Wendler, M. F. Holder, C. Linnhoff, M. Berghöfer, H. Winner und M. Maurer, "Towards a Generally Accepted Validation Methodology for Sensor Models - Challenges, Metrics, and First Results", in *Graz Symposium Virtual Vehicle (GSVF)*, 2019.

[28] M. Viehof, "Objektive Qualitätsbewertung von Fahrdynamiksimulationen durch statistische Validierung", Dissertation, TU Darmstadt, 2018.

[29] S. Riedmaier, J. Nesensohn, C. Gutenkunst, T. Düser, B. Schick und H. Abdellatif, "Validation of X-in-the-Loop Approaches for Virtual Homologation of Automated Driving Functions", in *11th Graz Symposium Virtual Vehicle*, 2018.

[30] K. Groh, S. Wagner, T. Kuehbeck und A. Knoll, "Simulation and Its Contribution to Evaluate Highly Automated Driving Functions", *SAE Int. J. Advances & Curr. Prac. in Mobility*, Nr. 1, S. 539–549, 2019. DOI: 10.4271/2019-01-0140.

[31] D. Notz, M. Sigl, T. Kuhbeck, S. Wagner, K. Groh, C. Schutz und D. Watzenig, "Methods for Improving the Accuracy of the Virtual Assessment of Autonomous Driving", in *8th International Conference on Connected Vehicles and Expo (ICCVE)*, IEEE, 2019. DOI: 10.1109/ICCVE45908.2019.8965040.

[32] Daimler Truck AG, *Rundum sicher, Der neue Mercedes-Benz Actros*, 2019.

[33] T. M. Gasser, *Rechtsfolgen zunehmender Fahrzeugautomatisierung, Gemeinsamer Schlussbericht der Projektgruppe ; Bericht zum Forschungsprojekt F 1100.5409013.01*, Ser. Berichte der Bundesanstalt für Strassenwesen F, Fahrzeugtechnik. Bremerhaven, 2012, Bd. 83.

[34] NHTSA, *Preliminary statement of policy concerning automated vehicles*, Washington, D.C.: National Highway Traffic Safety Administration, 2013.

[35] ACEA, Hrsg., *Code of Practice for the Design and Evaluation of ADAS*, Brussels: European Automobile Manufacturers' Association (ACEA), 2009.

[36] T. M. Gasser, A. Seeck und B. W. Smith, "Rahmenbedingungen für die Fahrerassistenzentwicklung", in *Handbuch Fahrerassistenzsysteme*, H. Winner, S. Hakuli, F. Lotz und C. Singer, Hrsg., Wiesbaden: Springer Fachmedien, 2015, S. 27–54. DOI: 10.1007/978-3-658-05734-3_3.

[37] M. Wood, P. Robbel, M. Maass, R. D. Tebbens, M. Meijs, M. Harb, J. Reach, K. Robinson, D. Wittmann, T. Srivastava, M. Essayed, S. Liu, Y. Wang, C. Knobel, D. Boymanns, M. Löhning, B. Dehlink, D. Kaule, R. Krüger, J. Frtunikj, M. Gruber, J. Steck, J. Meija-Hernandez, S. Syguda, P. Blüher, K. Klonecki, P. Schnarz, T. Wiltschko, S. Pukallus, K. Sedlaczek, N. Garbacik, D. Smerza, D. Li, A. Timmons, M. Bellotti, M. O'Brien, M. Schöllhorn, U. Dannebaum, J. Weast, A. Tatourian, B. Dornieden, P. Schnetter, T. Weidner und P. Schlicht, *Safety First for Automated Driving*, 2019.

[38] O. S. Tas, F. Kuhnt, J. M. Zollner und C. Stiller, “Functional system architectures towards fully automated driving”, in *Intelligent Vehicles Symposium (IV)*, IEEE, 2016, S. 304–309. DOI: 10.1109/IVS.2016.7535402.

[39] B. Abendroth und R. Bruder, “Die Leistungsfähigkeit des Menschen für die Fahrzeugführung”, in *Handbuch Fahrerassistenzsysteme*, H. Winner, S. Hakuli, F. Lotz und C. Singer, Hrsg., Wiesbaden: Springer Fachmedien, 2015, S. 3–15. DOI: 10.1007/978-3-658-05734-3_1.

[40] E. Donges, “Fahrerverhaltensmodelle”, in *Handbuch Fahrerassistenzsysteme*, H. Winner, S. Hakuli, F. Lotz und C. Singer, Hrsg., Bd. 68, Wiesbaden: Springer Fachmedien, 2015, S. 17–26. DOI: 10.1007/978-3-658-05734-3_2.

[41] R. Matthaei und M. Maurer, “Autonomous driving – a top-down-approach”, *at - Automatisierungstechnik*, Jg. 63, Nr. 3, 2015, ISSN: 0178-2312. DOI: 10.1515/auto-2014-1136.

[42] J. Bach, S. Otten und E. Sax, “A Taxonomy and Systematic Approach for Automotive System Architectures - From Functional Chains to Functional Networks”, in *Proceedings of the 3rd International Conference on Vehicle Technology and Intelligent Transport Systems*, Scitepress, 2017, S. 90–101. DOI: 10.5220/0006307600900101.

[43] H.-R. Tränkler, “Einführung in die Sensortechnik”, in *Sensortechnik, Handbuch für Praxis und Wissenschaft*, Ser. VDI-Buch, H.-R. Tränkler und L. M. Reindl, Hrsg., 2. Aufl., Berlin: Springer Vieweg, 2014, S. 3–20. DOI: 10.1007/978-3-642-29942-1_1.

[44] J. K. Baruah, R. Bera und S. Dhar, “Ranking of Sensors for ADAS—An MCDM-Based Approach”, in *Advances in Communication, Devices and Networking*, Ser. Lecture Notes in Electrical Engineering 462, R. Bera, S. K. Sarkar und S. Chakraborty, Hrsg., Bd. 462, Springer Singapore, 2018, S. 563–571. DOI: 10.1007/978-981-10-7901-6_61.

[45] H. Winner, “Radarsensorik”, in *Handbuch Fahrerassistenzsysteme*, H. Winner, S. Hakuli, F. Lotz und C. Singer, Hrsg., Wiesbaden: Springer Fachmedien, 2015, S. 259–316. DOI: 10.1007/978-3-658-05734-3_17.

[46] K. Reif, Hrsg., *Sensoren im Kraftfahrzeug*, 3. Aufl., Bosch Fachinformation Automobil, Wiesbaden: Springer Vieweg, 2016, 209 S.

[47] D. Belgiovane und C.-C. Chen, "Micro-Doppler characteristics of pedestrians and bicycles for automotive radar sensors at 77 GHz", in *11th European Conference on Antennas and Propagation (EUCAP), 19-24 March 2017*, IEEE, 2017, S. 2912–2916. DOI: 10.23919/EuCAP.2017.7928457.

[48] M. Punke, S. Menzel, B. Werthessen, N. Stache und M. Höpfl, "Kamera-Hardware", in *Handbuch Fahrerassistenzsysteme*, H. Winner, S. Hakuli, F. Lotz und C. Singer, Hrsg., Wiesbaden: Springer Fachmedien, 2015, S. 347–368. DOI: 10.1007/978-3-658-05734-3_20.

[49] H. Gotzig und G. O. Geduld, "LIDAR-Sensorik", in *Handbuch Fahrerassistenzsysteme*, H. Winner, S. Hakuli, F. Lotz und C. Singer, Hrsg., Wiesbaden: Springer Fachmedien, 2015, S. 317–334. DOI: 10.1007/978-3-658-05734-3_18.

[50] J. Kocic, N. Jovicic und V. Drndarevic, "Sensors and Sensor Fusion in Autonomous Vehicles", in *Telecommunications Forum (TELFOR)*, IEEE, 2018, S. 420–425. DOI: 10.1109/TELFOR.2018.8612054.

[51] R. H. Rasshofer und K. Gresser, "Automotive Radar and Lidar Systems for Next Generation Driver Assistance Functions", *Advances in Radio Science*, Jg. 3, S. 205–209, 2005.

[52] M. Noll und P. Rapps, "Ultraschallsensorik", in *Handbuch Fahrerassistenzsysteme*, H. Winner, S. Hakuli, F. Lotz und C. Singer, Hrsg., Wiesbaden: Springer Fachmedien, 2015, S. 243–258. DOI: 10.1007/978-3-658-05734-3_16.

[53] A. Ziebinski, R. Cupek, H. Erdogan und S. Waechter, "A Survey of ADAS Technologies for the Future Perspective of Sensor Fusion", in *International Conference on Computational Collective Intelligence (ICCCI)*, Bd. 9876, Springer, 2016, S. 135–146. DOI: 10.1007/978-3-319-45246-3_13.

[54] S. Campbell, N. O'Mahony, L. Krpalcova, D. Riordan, J. Walsh, A. Murphy und C. Ryan, "Sensor Technology in Autonomous Vehicles : A review", in *Irish Signals and Systems Conference (ISSC)*, IEEE, 2018, S. 1–4. DOI: 10.1109/ISSC.2018.8585340.

[55] H. G. Seif und X. Hu, "Autonomous Driving in the iCity—HD Maps as a Key Challenge of the Automotive Industry", *Engineering*, Jg. 2, Nr. 2, S. 159–162, 2016, ISSN: 20958099. DOI: 10.1016/J.ENG.2016.02.010.

[56] F. Jomrich, J. Schmid, S. Knapp, A. Hoss, R. Steinmetz und B. Schuller, "Analysing communication requirements for crowd sourced backend generation of HD Maps used in automated driving", in *IEEE Vehicular Networking Conference (VNC)*, IEEE, 2018, S. 1–8. DOI: 10.1109/VNC.2018.8628335.

[57] S. Thrun, "Simultaneous Localization and Mapping", in *Robotics and Cognitive Approaches to Spatial Mapping*, Ser. Springer Tracts in Advanced Robotics, M. E. Jefferies und W.-K. Yeap, Hrsg., Bd. 38, Berlin, Heidelberg: Springer, 2008, S. 13–41. DOI: 10.1007/978-3-540-75388-9_3.

[58] H. Fuchs, F. Hofmann, H. Löhr und G. Schaaf, "Car-2-X", in *Handbuch Fahrerassistenzsysteme*, H. Winner, S. Hakuli, F. Lotz und C. Singer, Hrsg., Wiesbaden: Springer Fachmedien, 2015, S. 525–540. DOI: 10.1007/978-3-658-05734-3_28.

[59] M. Darms, "Fusion umfelderfassender Sensoren", in *Handbuch Fahrerassistenzsysteme*, H. Winner, S. Hakuli, F. Lotz und C. Singer, Hrsg., Wiesbaden: Springer Fachmedien, 2015, S. 439–451. DOI: 10.1007/978-3-658-05734-3_24.

[60] B. Paden, M. Cap, S. Z. Yong, D. Yershov und E. Frazzoli, "A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles", *IEEE Transactions on Intelligent Vehicles*, Jg. 1, Nr. 1, S. 33–55, 2016, ISSN: 2379-8858. DOI: 10.1109/TIV.2016.2578706.

[61] D. Gonzalez, J. Perez, V. Milanes und F. Nashashibi, "A Review of Motion Planning Techniques for Automated Vehicles", *IEEE Transactions on Intelligent Transportation Systems*, Jg. 17, Nr. 4, S. 1135–1145, 2016, ISSN: 1524-9050. DOI: 10.1109/TITS.2015.2498841.

[62] R. Matthaei, A. Reschka, J. Rieken, F. Dierkes, S. Ulbrich, T. Winkle und M. Maurer, "Autonomes Fahren", in *Handbuch Fahrerassistenzsysteme*, H. Winner, S. Hakuli, F. Lotz und C. Singer, Hrsg., Bd. 11, Wiesbaden: Springer Fachmedien, 2015, S. 1139–1165. DOI: 10.1007/978-3-658-05734-3_61.

[63] H. Czichos, “Aktorik”, in *Mechatronik, Grundlagen und Anwendungen technischer Systeme*, H. Czichos, Hrsg., 3. Aufl., Wiesbaden: Springer Vieweg, 2015, S. 157–185. DOI: 10.1007/978-3-658-09950-3_6.

[64] L. Eckstein, *Längsdynamik von Kraftfahrzeugen: Vorlesungsumdruck Fahrzeugtechnik I*. Forschungsgesellschaft Kraftfahrwesen mbH, 2015.

[65] K. Reif, *Automobilelektronik*. Wiesbaden: Springer Fachmedien, 2014. DOI: 10.1007/978-3-658-05048-1.

[66] J. Remfrey, S. Gruber und N. Ocvirk, “Hydraulische Pkw-Bremssysteme”, in *Handbuch Fahrerassistenzsysteme*, H. Winner, S. Hakuli, F. Lotz und C. Singer, Hrsg., Wiesbaden: Springer Fachmedien, 2015, S. 555–577. DOI: 10.1007/978-3-658-05734-3_30.

[67] K. Dörner, W. Schwertberger und E. Hipp, “Bahnführungsassistenz für Nutzfahrzeuge”, in *Handbuch Fahrerassistenzsysteme*, H. Winner, S. Hakuli, F. Lotz und C. Singer, Hrsg., Wiesbaden: Springer Fachmedien, 2015, S. 1009–1027. DOI: 10.1007/978-3-658-05734-3_53.

[68] B. Bayer, A. Büse, P. Linhoff, B. Piller, P. E. Rieth, S. Schmitt, B. Schmittner und J. Völkel, “Elektromechanische Bremssysteme”, in *Handbuch Fahrerassistenzsysteme*, H. Winner, S. Hakuli, F. Lotz und C. Singer, Hrsg., Wiesbaden: Springer Fachmedien, 2015, S. 579–589. DOI: 10.1007/978-3-658-05734-3_31.

[69] L. Eckstein, *Vertikal- und Querdynamik von Kraftfahrzeugen: Vorlesungsumdruck Fahrzeugtechnik II*. Forschungsgesellschaft Kraftfahrwesen mbH, 2014.

[70] G. Reimann, P. Brenner und H. Büring, “Lenkstellsysteme”, in *Handbuch Fahrerassistenzsysteme*, H. Winner, S. Hakuli, F. Lotz und C. Singer, Hrsg., Wiesbaden: Springer Fachmedien, 2015, S. 591–617. DOI: 10.1007/978-3-658-05734-3_32.

[71] M. Bäumer, H. Hautzinger, M. Pfeiffer, W. Stock, B. Lenz, T. Kuhnimhof und K. Köhler, *Fahrleistungserhebung 2014 - Inländerfahrleistung*, Ser. Berichte der Bundesanstalt für Strassenwesen Verkehrstechnik. Bremen: Fachverlag NW, 2017.

[72] H. Winner und M. Schopper, “Adaptive Cruise Control”, in *Handbuch Fahrerassistenzsysteme*, H. Winner, S. Hakuli, F. Lotz und C. Singer, Hrsg., Wiesbaden: Springer Fachmedien, 2015, S. 851–891. DOI: 10.1007/978-3-658-05734-3_46.

[73] Daimler Truck AG, *Mercedes-Benz Trucks präsentiert zwei Weltneuheiten im Lkw für mehr Sicherheit auf der Straße*, Daimler Truck AG, Hrsg., Stuttgart, 23. Sep. 2020.

[74] B. Fildes, M. Keall, N. Bos, A. Lie, Y. Page, C. Pastor, L. Pennisi, M. Rizzi, P. Thomas und C. Tingvall, "Effectiveness of low speed autonomous emergency braking in real-world rear-end crashes", *Accident Analysis and Prevention*, Jg. 81, S. 24–29, 2015. DOI: 10.1016/j.aap.2015.03.029.

[75] ADAC, *Information zu neuen Fahrzeugsystemen zur Erhöhung der Verkehrssicherheit*, 2019.

[76] UNECE, *Einheitliche Bedingungen für die Genehmigung der Kraftfahrzeuge hinsichtlich ihres Spurhaltewarnsystems*, R 130, 2013.

[77] ——, *Einheitliche Bedingungen für die Genehmigung von Kraftfahrzeugen hinsichtlich des Notbremsassistenzsystems (AEBS)*, R 131, 2014.

[78] ——, *Einheitliche Bedingungen für die Genehmigung von Kraftfahrzeugen hinsichtlich des Totwinkel-Assistenten zur Erkennung von Fahrrädern*, R 151, 2020.

[79] M. von der Beeck, "Development of logical and technical architectures for automotive systems", *Software & Systems Modeling*, Jg. 6, Nr. 2, S. 205–219, 2007, ISSN: 1619-1366. DOI: 10.1007/s10270-006-0022-z.

[80] ISO/IEC/IEEE, Hrsg., *ISO/IEC/IEEE 24765*, Geneva: International Organization for Standardization, 2017.

[81] DIN, Hrsg., *DIN IEC 60050-351*, Berlin: Deutsches Institut für Normung e. V., 2014.

[82] AUTOSAR, *Glossary*, 2020.

[83] B. W. Boehm, "Verifying and Validating Software Requirements and Design Specifications", *IEEE Software*, Jg. 1, Nr. 1, S. 75–88, 1984, ISSN: 0740-7459. DOI: 10.1109/MS.1984.233702.

[84] G. Hab und R. Wagner, *Projektmanagement in der Automobilindustrie*. Wiesbaden: Springer Fachmedien, 2017. DOI: 10.1007/978-3-658-10472-6.

[85] R. G. Cooper, "Perspective: The Stage-Gate ® Idea-to-Launch Process—Update, What's New, and NexGen Systems", *Journal of Product Innovation Management*, Jg. 25, Nr. 3, S. 213–232, 2008, ISSN: 0737-6782. DOI: 10.1111/j.1540-5885.2008.00296.x.

[86] E. Sax, Hrsg., *Automatisiertes Testen Eingebetteter Systeme in der Automobilindustrie*, 1. Aufl., München: Hanser Verlag, 2008.

[87] S. Balaji und M. S. Murugaiyan, "Waterfall vs. V-Model vs. Agile: A comparative study on SDLC", *International Journal of Information Technology and Business Management*, Jg. 2, Nr. 1, S. 26–30, 2012.

[88] P. Rook, "Controlling software projects", *Software Engineering Journal*, Jg. 1, Nr. 1, S. 7, 1986, ISSN: 02686961. DOI: 10.1049/sej.1986.0003.

[89] W. Dröschel und M. Wiemers, Hrsg., *Das V-Modell 97, Der Standard für die Entwicklung von IT-Systemen mit Anleitung für den Praxiseinsatz*, München: De Gruyter Oldenbourg, 1999. DOI: 10.1515/9783486800265.

[90] P. Ammann und J. Offutt, *Introduction to Software Testing*. New York: Cambridge University Press, 2008.

[91] R. Kneuper, "Die geschichtliche Entwicklung des V-Modells", *IUBH Discussion Papers*, Jg. 1, Nr. 1, 2018.

[92] D. Angermeier, C. Bartelt, O. Bauer, B. Gerd, K. Bergner, U. Birowicz, T. Bliß, C. Breitenstrom, N. Cordes, D. Cruz, P. Dohrmann, J. Friedrich, M. Gnatz, U. Hammerschall, I. Hidvegi-Barstorfer, H. Hummel, D. Israel, T. Klingenberg, K. Klugseder, I. Küffer, M. Kuhrmann, M. Kranz, W. Kranz, H.-J. Meinhardt, M. Meisinger, S. Mittrach, H.-J. Neußer, D. Niebuhr, K. Plögert, D. Rauh, A. Rausch, T. Rittel, W. Rösch, E. Saas, J. Schramm, M. Sihling, T. Ternité, S. Vogel, B. Weber und M. Wittmann, *V-Modell XT, Das deutsche Referenzmodell für Systementwicklungsprojekte*, Verein zur Weiterentwicklung des V-Modell XT e.V, 2006.

[93] K. Schwaber und J. V. Sutherland, *Software in 30 days, How Agile managers beat the odds, delight their customers, and leave competitors in the dust*. Hoboken, NJ, USA: John Wiley & Sons Inc, 2012.

[94] T. Dyba und T. Dingsoyr, "What Do We Know about Agile Software Development?", *IEEE Software*, Jg. 26, Nr. 5, S. 6–9, 2009, ISSN: 0740-7459. DOI: 10.1109/MS.2009.145.

[95] Digital.ai, *14th Annual State of Agile Report*, 2020.

[96] K. Schwaber und J. Sutherland, *The Scrum Guide*, 2020. Adresse: https://www.scrum.org/resources/scrum-guide (besucht am 02. 05. 2021).

[97] ISO, Hrsg., *ISO 21448*, Geneva: International Organization for Standardization, 2022.

[98] ISO, Hrsg., *ISO/TR 4804*, Geneva: International Organization for Standardization, 2020.

[99] A. Spillner und T. Linz, *Basiswissen Softwaretest, Aus- und Weiterbildung zum Certified Tester : Foundation Level nach ISTQB-Standard*, 6. Aufl. 2019.

[100] ISO, Hrsg., *ISO/IEC/IEEE 29119-2*, Geneva: International Organization for Standardization, 2013.

[101] P. Liggesmeyer, *Software-Qualität, Testen, Analysieren und Verifizieren von Software*, 2. Aufl. Spektrum Akademischer Verlag, 2009.

[102] ISO, Hrsg., *ISO/IEC/IEEE 29119-4*, Geneva: International Organization for Standardization, 2015.

[103] D. W. Hoffmann, *Software-Qualität*. Berlin, Heidelberg: Springer, 2013. DOI: 10.1007/978-3-642-35700-8.

[104] M. Grindal, J. Offutt und S. F. Andler, "Combination testing strategies: a survey", *Software Testing, Verification and Reliability*, Jg. 15, Nr. 3, S. 167–199, 2005, ISSN: 0960-0833. DOI: 10.1002/stvr.319.

[105] W. Wachenfeld, "How Stochastic can Help to Introduce Automated Driving", Dissertation, TU Darmstadt, 2017.

[106] ISO, Hrsg., *ISO 15622:2018*, Geneva: International Organization for Standardization, 2018.

[107] L. Eckstein und A. Zlocki, "Safety Potential of ADAS – Combined Methods for an Effective Evaluation", in *International Technical Conference on the Enhanced Safety of Vehicles (ESV)*, 2013.

[108] DLR, *PEGASUS Abschlussbericht Gesamtprojekt*, 2020.

[109] M. Reichenbach. (2018). “Fahrerassistenz auf dem Weg zum autonomen Fahren”. Springer Professional, Hrsg., Adresse: https://www.springerprofessional.de/fahrerassistenz/automatisiertes-fahren/fahrerassistenz-auf-dem-weg-zum-autonomen-fahren/15690872 (besucht am 06. 11. 2020).

[110] Waymo LLC, *Waymo Safety Report*, Waymo LLC, 2018.

[111] H.-P. Schöner, W. Hurich, J. Luther und R. G. Herrtwich, “Koordiniertes automatisiertes Fahren für die Erprobung von Assistenzsystemen”, *ATZ - Automobiltechnische Zeitschrift*, Jg. 113, Nr. 1, S. 40–45, 2011, ISSN: 0001-2785. DOI: 10.1365/s35148-011-0010-7.

[112] S. Nord, M. Lindgren und J. Spetz, “HiFi Visual Target—Methods for Measuring Optical and Geometrical Characteristics of Soft Car Targets for ADAS and AD”, in *Advanced Microsystems for Automotive Applications 2017*, Ser. Lecture Notes in Mobility, C. Zachäus, B. Müller und G. Meyer, Hrsg., Cham: Springer International Publishing, 2018, S. 201–209. DOI: 10.1007/978-3-319-66972-4_17.

[113] V. Sandner, “Development of a test target for AEB systems”, in *International Technical Conference on the Enhanced Safety of Vehicles (ESV)*, 2013.

[114] P. Lemmen, J. Stoll, U. Bergelt, P. Seiniger, M. Wisch, O. Bartels, E. Schubert, M. Kunert, I. Knight, D. Brookes u. a., “Evaluation of pedestrian targets used in AEB testing: A report from Harmonistion Platform 2 dealing with test equipment”, in *International Technical Conference on the Enhanced Safety of Vehicles (ESV)*, 2013.

[115] A. Zlocki, C. Rösener, S. Klaudt und L. Eckstein, “Ganzheitliche Werkzeugkette für die Entwicklung und Bewertung des automatisierten Fahrens”, *ATZextra*, Jg. 23, Nr. S5, S. 16–21, 2018, ISSN: 2195-1454. DOI: 10.1007/s35778-018-0046-3.

[116] T. Machida und K. Shitaoka, “Test Coverage Index for ADAS/ADS Assessment Based on Various Real-World Information Points”, *IEEE Transactions on Intelligent Transportation Systems*, S. 1–13, 2020, ISSN: 1524-9050. DOI: 10.1109/TITS.2020.3026679.

[117] M. Benmimoun und L. Eckstein, “Detection of Critical Driving Situations for Naturalistic Driving Studies by Means of an Automated Process”, *Journal of Intelligent Transportation and Urban Planning*, S. 11–21, 2014, ISSN: 23730757. DOI: 10.18005/ITUP0201002.

[118] M. Benmimoun, M. Ljung Aust, F. Faber, G. Saint Pierre und A. Zlocki, "Safety analysis method for assessing the impacts of advanced driver assistance systems within the European large scale field test euroFOT", in *ITS European Congress*, 2011.

[119] D. Hibberd, T. Louw, E. Aittoniemi, R. Brouwer, M. Dotzauer, F. Fahrenkrog, S. Innamaa, S. Kuisma, N. Merat, B. Metz, N. Neila, M. Penttinen, P. Puente Guillen, C. Rösener, A. Silla, T. Streubel, F. Tango, B. van den Boom, H. Weber, J. Woerle und A. Zerbe, *L3Pilot: From Research Questions to Logging Requirements, Deliverable D3.1*, L3Pilot Consortium, 2018.

[120] J. Bach, "Methoden und Ansätze für die Entwicklung und den Test prädiktiver Fahrzeugregelungsfunktionen", Dissertation, Karlsruher Institut für Technologie, 2018.

[121] U. Baake, K. Wüst, M. Maurer und A. Lutz, "Versuchs- und simulationsbasierte Absicherung von ESP-Systemen für Transporter", *ATZ - Automobiltechnische Zeitschrift*, Jg. 116, Nr. 2, S. 46–51, 2014, ISSN: 0001-2785. DOI: 10.1007/s35148-014-0043-9.

[122] G. Chance, A. Ghobrial, K. McAreavey, S. Lemaignan, T. Pipe und K. Eder, "On Determinism of Game Engines Used for Simulation-Based Autonomous Vehicle Verification", *IEEE Transactions on Intelligent Transportation Systems*, Jg. 23, Nr. 11, S. 20 538–20 552, 2022, ISSN: 1524-9050. DOI: 10.1109/TITS.2022.3177887.

[123] The MathWorks. (2020). "Simulink - Simulation und Model-Based Design", Adresse: https://de.mathworks.com/products/simulink.html (besucht am 28. 11. 2020).

[124] dSpace GmbH. (2020). "TargetLink Product Information", Adresse: https://www.dspace.com/shared/data/pdf/2020/dSPACE-TargetLink_Product-information_2020_02_EN.pdf (besucht am 28. 11. 2020).

[125] AUTOSAR, *Specification of RTE Software*, 2019.

[126] dSpace GmbH. (2020). "Software-in-the-Loop-Testing", Adresse: https://www.dspace.com/shared/data/pdf/2020/Software-in-the-Loop-testing_TwoPager_06_201006.pdf (besucht am 28. 11. 2020).

[127] Synopsis Inc., *Silver Datasheet*, 2020.

[128] H. Wallentowitz und K. Reif, *Handbuch Kraftfahrzeugelektronik*. Wiesbaden: Vieweg, 2006. DOI: 10.1007/978-3-8348-9121-1.

[129] H.-H. Braess, T. Breitling, J. Weissinger, N. Grawunder, U. Hackenberg, V. Liskowsky und U. Widmann, "Produktentstehungsprozess", in *Vieweg Handbuch Kraftfahrzeugtechnik*, S. Pischinger und U. Seiffert, Hrsg., Wiesbaden: Springer Fachmedien Wiesbaden, 2016, S. 1257–1369. DOI: 10.1007/978-3-658-09528-4_11.

[130] DIN, Hrsg., *DIN ISO 8855:2013-11*, Berlin: Deutsches Institut für Normung e. V., 2013.

[131] M. Ayoubi, A. Eilemann, H. Mankau, E. Pantow, C. Repmann, U. Seiffert, M. Wawzyniak und A. Wiebelt, "Fahrzeugphysik", in *Vieweg Handbuch Kraftfahrzeugtechnik*, S. Pischinger und U. Seiffert, Hrsg., Wiesbaden: Springer Fachmedien Wiesbaden, 2016, S. 57–130. DOI: 10.1007/978-3-658-09528-4_3.

[132] K. Abdelgawad, M. Abdelkarim, B. Hassan, M. Grafe und I. Grassler, "A modular architecture of a PC-based driving simulator for advanced driver assistance systems development", in *International Workshop on Research and Education in Mechatronics (REM)*, IEEE, 2014, S. 1–8. DOI: 10.1109/REM.2014.6920237.

[133] J. Mauss, "Virtuelle Steuergeräte für die Antriebsentwicklung", in *VPC - Simulation und Test 2015*, Wiesbaden: Springer Vieweg, 2018, S. 311–321. DOI: 10.1007/978-3-658-20736-6_21.

[134] V. Jaikamal und T. Zurawka, "Advanced Techniques for Simulating ECU C-code on the PC", in *SAE Technical Paper Series*, Ser. SAE Technical Paper Series, SAE International, 2010. DOI: 10.4271/2010-01-0431.

[135] M. Krammer, H. Martin, Z. Radmilovic, S. Erker und M. Karner, "Standard Compliant Co-simulation Models for Verification of Automotive Embedded Systems", in *Languages, Design Methods, and Tools for Electronic System Design*, Ser. Lecture Notes in Electrical Engineering 385, R. Drechsler und R. Wille, Hrsg., Bd. 385, Springer, 2016, S. 29–47. DOI: 10.1007/978-3-319-31723-6_2.

[136] F. Christen und Q. Huang, *Das Fahrermodell im Verkehrsflusssimulationsprogramm PELOPS : Modellierung und Applikationsmöglichkeiten*, 2008.

[137] S. Krauß, “Microscopic modeling of traffic flow: Investigation of collision free vehicle dynamics”, Deutsches Zentrum für Luft– und Raumfahrt, Forschungsbericht, Universität zu Köln, Cologne, 1998.

[138] P. G. Gipps, “A behavioural car-following model for computer simulation”, *Transportation Research Part B: Methodological*, Jg. 15, Nr. 2, S. 105–111, 1981. DOI: 10.1016/0191-2615(81)90037-0.

[139] M. Treiber und D. Helbing, “Realistische Mikrosimulation von Straßenverkehr mit einem einfachen Modell”, in *Simulationstechnik*, Ser. Frontiers in Simulation, Bd. 16, Society for Computer Simulation International, 2003, S. 514–520.

[140] R. Wiedemann, “Simulation des Straßenverkehrsflusses”, *Schriftenreihe des Instituts für Verkehrswesen der Universität Karlsruhe*, Jg. 8, 1974.

[141] H. Fritzsche, “A Model for Traffic Simulation”, *Traffic Engineering & Control*, Jg. 35, Nr. 317-321, 1994.

[142] M. Semrau, *Untersuchung zur Modellierung von chinesischem Fahrverhalten auf Autobahnen für den Test pilotierter Fahrfunktionen*. Wiesbaden: Springer Fachmedien, 2018. DOI: 10.1007/978-3-658-23761-5.

[143] J. Dallmeyer, *Simulation des Straßenverkehrs in der Großstadt*. Wiesbaden: Springer Fachmedien, 2014. DOI: 10.1007/978-3-658-05207-2.

[144] MSC.Software GmbH, *Virtual Test Drive Brochure*, 2017.

[145] IPG Automotive GmbH. (2020). “CarMaker”, Adresse: https://ipg-automotive.com/de/produkte-services/simulation-software/carmaker/ (besucht am 13. 11. 2020).

[146] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez und V. Koltun, “CARLA: An Open Urban Driving Simulator”, in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, S. 1–16.

[147] IPG Automotive GmbH. (2020). “TruckMaker”, Adresse: https://ipg-automotive.com/de/produkte-services/simulation-software/truckmaker/ (besucht am 13. 11. 2020).

[148] Siemens Digital Industries Software. (21.09.2020). “Simcenter Prescan”, Adresse: https://www.plm.automation.siemens.com/global/en/products/simcenter/prescan.html (besucht am 13. 11. 2020).

[149] Applied Intuition. (13.11.2020). “Simulation Platform for Testing Your Autonomy Stack | Applied Intuition”, Adresse: https://www.appliedintuition.com/simulation (besucht am 13. 11. 2020).

[150] D. Salomon, “Rendering”, in *The computer graphics manual*, Ser. Texts in Computer Science, D. Salomon, Hrsg., London: Springer, 2011, S. 851–889. DOI: 10.1007/978-0-85729-886-7_17.

[151] F. Schmidt, “Funktionale Absicherung kamerabasierter Aktiver Fahrerassistenzsysteme durch Hardware-in the-Loop-Tests”, Dissertation, Universität Kaiserslautern, 2012.

[152] E. Roth, T. Dirndorfer, K. v. Neumann-Cosel, M.-O. Fischer, T. Ganslmeier, A. Kern und A. Knoll, “Analysis and validation of perception sensor models in an integrated vehicle and environment simulation”, in *Proceedings of the 22nd Enhanced Safety of Vehicles Conference*, 2011.

[153] D. Tellmann, “Hardware-in-the-loop-gestützte Entwicklungsplattform für Fahrerassistenzsysteme, Modelle der Umfeldsensorik und angepasste Fahrermodelle”, Dissertation, Universität Kassel, 2012.

[154] T. Ponn, F. Muller und F. Diermeyer, “Systematic Analysis of the Sensor Coverage of Automated Vehicles Using Phenomenological Sensor Models”, in *Intelligent Vehicles Symposium (IV)*, IEEE, 2019, S. 1000–1006. DOI: 10.1109/IVS.2019.8813794.

[155] R. Schubert, N. Mattern und R. Bours, “Simulation of Sensor Models for the Evaluation of Advanced Driver Assistance Systems”, *ATZelektronik worldwide*, Jg. 9, Nr. 3, S. 26–29, 2014. DOI: 10.1365/s38314-014-0247-5.

[156] M. Feilhauer, “Simulationsgestützte Absicherung von Fahrerassistenzsystemen”, Dissertation, 2018.

[157] J. E. Stellet, M. R. Zofka, J. Schumacher, T. Schamm, F. Niewels und J. M. Zollner, "Testing of Advanced Driver Assistance Towards Automated Driving: A Survey and Taxonomy on Existing Approaches and Open Questions", in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2015, S. 1455–1462. DOI: 10.1109/ITSC.2015.236.

[158] P. Cao, W. Wachenfeld und H. Winner, "Perception sensor modeling for virtual validation of automated driving", *it - Information Technology*, Jg. 57, Nr. 4, 2015, ISSN: 1611-2776. DOI: 10.1515/itit-2015-0006.

[159] S. Bernsteiner, Z. Magosi, D. Lindvai-Soos und A. Eichberger, "Radarsensormodell für den virtuellen Entwicklungsprozess", *ATZelektronik*, Jg. 10, Nr. 2, S. 72–79, 2015, ISSN: 1862-1791. DOI: 10.1007/s35658-015-0508-y.

[160] S. Muckenhuber, H. Holzer, J. Rubsam und G. Stettinger, "Object-based sensor model for virtual testing of ADAS/AD functions", in *International Conference on Connected Vehicles and Expo (ICCVE)*, IEEE, S. 1–6. DOI: 10.1109/ICCVE45908.2019.8965071.

[161] R. Molenaar, A. van Bilsen, R. van der Made und R. de Vries, "Full spectrum camera simulation for reliable virtual development and validation of ADAS and automated driving applications", in *Intelligent Vehicles Symposium (IV)*, IEEE, 2015, S. 47–52. DOI: 10.1109/IVS.2015.7225661.

[162] M. Holder, P. Rosenberger, H. Winner, T. Dhondt, V. P. Makkapati, M. Maier, H. Schreiber, Z. Magosi, Z. Slavik, O. Bringmann und W. Rosenstiel, "Measurements revealing Challenges in Radar Sensor Modeling for Virtual Validation of Autonomous Driving", in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2018, S. 2616–2622. DOI: 10.1109/ITSC.2018.8569423.

[163] M. Elgharbawy, A. Schwarzhaupt, M. Frey und F. Gauterin, "A real-time multisensor fusion verification framework for advanced driver assistance systems", *Transportation Research Part F: Traffic Psychology and Behaviour*, Jg. 61, S. 259–267, 2019. DOI: 10.1016/j.trf.2016.12.002.

[164] Modelica Association. (2022). “Functional Mock-up Interface Specification, Version 3.0”, Adresse: https://fmi-standard.org/docs/3.0/ (besucht am 28. 01. 2023).

[165] DLR. (2020). “Open Simulation Interface - pegasus”, Adresse: https://www.pegasusprojekt.de/de/72/open-simulation-interface (besucht am 07. 12. 2020).

[166] ASAM. (2020). “Welcome to Open Simulation Interface’s documentation! — Open Simulation Interface v3.1.2 documentation”, Adresse: https://opensimulationinterface.github.io/osi-documentation/ (besucht am 13. 11. 2020).

[167] Google LLC. (2019). “Protocol Buffers | Google Developers”, (besucht am 13. 11. 2020).

[168] N. Kalra und S. Paddock, *Driving to Safety: How Many Miles of Driving Would It Take to Demonstrate Autonomous Vehicle Reliability?* RAND Corporation, 2016. DOI: 10.7249/RR1478.

[169] J. Hedderich und L. Sachs, “Zufallsvariablen, Verteilungen”, in *Angewandte Statistik, Methodensammlung mit R*, J. Hedderich und L. Sachs, Hrsg., 16. Aufl., Berlin, Heidelberg: Springer, 2018, S. 202–324. DOI: 10.1007/978-3-662-56657-2_5.

[170] D. Åsljung, J. Nilsson und J. Fredriksson, “Comparing Collision Threat Measures for Verification of Autonomous Vehicles using Extreme Value Theory”, *IFAC-PapersOnLine*, Jg. 49, Nr. 15, S. 57–62, 2016, ISSN: 24058963. DOI: 10.1016/j.ifacol.2016.07.709.

[171] ——, “Using Extreme Value Theory for Vehicle Level Safety Validation and Implications for Autonomous Vehicles”, *IEEE Transactions on Intelligent Vehicles*, Jg. 2, Nr. 4, S. 288–297, 2017, ISSN: 2379-8858. DOI: 10.1109/TIV.2017.2768219.

[172] Daimler Truck AG, *Daimler Trucks and Torc Robotics celebrate one year of successful collaboration – adding testing center in New Mexico*, Daimler Truck AG, Hrsg., Blacksburg / Stuttgart / Portland, 3. Sep. 2020.

[173] M. Feilhauer, J. Haering und S. Wyatt, “Current Approaches in HiL-Based ADAS Testing”, *SAE International Journal of Commercial Vehicles*, Jg. 9, Nr. 2, S. 63–69, 2016, ISSN: 1946-3928. DOI: 10.4271/2016-01-8013.

[174] J. Langner, S. Otten, E. Sax, C. Esselborn, M. Holzäpfel, M. Eckert und J. Bach, "Framework for using real driving data in automotive feature development and validation", in *8. Tagung Fahrerassistenz*, 2017.

[175] V. Entin, T. Ganslmeier und Zawicki Krystian, "Formale und formatunabhängige Fahrszenarienbeschreibung für automatisierte Testvorgange im Bereich der Entwicklung von Fahrer-Assistenzsystemen", in *Beiträge der 39. Jahrestagung der Gesellschaft für Informatik e.V.*, 2009.

[176] M. R. Zofka, F. Kuhnt, R. Kohlhaas, C. Rist, T. Schamm und J. M. Zöllner, "Data-driven simulation and parametrization of traffic scenarios for the development of advanced driver assistance systems", in *International Conference on Information Fusion*, IEEE, 2015, S. 1422–1428.

[177] J. Löchner, J. Wagner, M. Wolter und A. Fernandez, "Validating advanced driver assistance systems (ADAS) using comprehensive, loss-free in-vehicle measurements", in *16. Internationales Stuttgarter Symposium, Automobil- und Motorentechnik*, Ser. Proceedings, Wiesbaden: Springer Vieweg, 2016, S. 1143–1154. DOI: 10.1007/978-3-658-13255-2_85.

[178] R. Pfeffer, E. Sax und S. Schmidt, "Realdatenbasierte simulationsgestützte Absicherung hochautomatisierter Fahrfunktionen", *ATZelektronik*, Jg. 14, Nr. 11, S. 24–29, 2019, ISSN: 1862-1791. DOI: 10.1007/s35658-019-0116-3.

[179] I. Passchier, G. van Vugt und M. Tideman, "An Integral Approach to Autonomous and Cooperative Vehicles Development and Testing", in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2015, S. 348–352. DOI: 10.1109/ITSC.2015.66.

[180] P. Junietz, W. Wachenfeld, V. Schönemann, K. Domhardt, W. Tribelhorn und H. Winner, "Gaining Knowledge on Automated Driving's Safety—The Risk-Free VAAFO Tool", in *Control Strategies for Advanced Driver Assistance Systems and Autonomous Driving Functions*, Ser. Lecture Notes in Control and Information Sciences, H. Waschl, I. Kolmanovsky und F. Willems, Hrsg., Bd. 476, Cham: Springer International Publishing, 2019, S. 47–65. DOI: 10.1007/978-3-319-91569-2_3.

[181] A. Koenig, M. Gutbrod, S. Hohmann und J. Ludwig, "Bridging the Gap between Open Loop Tests and Statistical Validation for Highly Automated Driving", *SAE International Journal of Transportation Safety*, Jg. 5, Nr. 1, S. 81–87, 2017, ISSN: 2327-5634. DOI: 10.4271/2017-01-1403.

[182] J. Bach, M. Holzäpfel, S. Otten und E. Sax, "Reactive-Replay Approach for Verification and Validation of Closed-Loop Control Systems in Early Development", in *SAE Technical Paper Series*, Ser. SAE Technical Paper Series, SAE International, 2017. DOI: 10.4271/2017-01-1671.

[183] S. Shalev-Shwartz, S. Shammah und A. Shashua, *On a Formal Model of Safe and Scalable Self-driving Cars*, Mobileye, 2017.

[184] S. Mitsch, K. Ghorbal und A. Platzer, "On Provably Safe Obstacle Avoidance for Autonomous Robotic Ground Vehicles", in *Robotics: Science and Systems IX*, Robotics: Science and Systems Foundation, 2013. DOI: 10.15607/RSS.2013.IX.014.

[185] A. Leitner, D. Watzenig und J. Ibanez-Guzman, Hrsg., *Validation and verification of automated systems, Results of the ENABLE-S3 Project*, Cham: Springer, 2020. DOI: 10.1007/978-3-030-14628-3.

[186] EICT GmbH. (2023). "VVM Projekt", Adresse: www.vvm-projekt.de/projekt (besucht am 08. 05. 2023).

[187] DLR. (2021). "SET Level - Projekt", Adresse: www.setlevel.de/projekt (besucht am 08. 05. 2023).

[188] H. Weber, J. Bock, J. Klimke, C. Roesener, J. Hiller, R. Krajewski, A. Zlocki und L. Eckstein, "A framework for definition of logical scenarios for safety assurance of automated driving", *Traffic Injury Prevention*, Jg. 20, Nr. S1, S65–S70, 2019. DOI: 10.1080/15389588.2019.1630827.

[189] T. Helmer, L. Wang, K. Kompass und R. Kates, "Safety Performance Assessment of Assisted and Automated Driving by Virtual Experiments: Stochastic Microscopic Traffic Simulation as Knowledge Synthesis", in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2015, S. 2019–2023. DOI: 10.1109/ITSC.2015.327.

[190] 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*, Jg. 8, Nr. 3, S. 183–189, 2014. DOI: 10.1049/iet-its.2012.0188.

[191] J. Bach, S. Otten und E. Sax, "Model based scenario specification for development and test of automated driving functions", in *Intelligent Vehicles Symposium (IV)*, IEEE, 2016, S. 1149–1155. DOI: 10.1109/ivs.2016.7535534.

[192] F. Schuldt, F. Saust, B. Lichte, M. Maurer und S. Scholz, *Effiziente systematische Testgenerierung für Fahrerassistenzsysteme in virtuellen Umgebungen*, TU Braunschweig.

[193] G. Bagschik, T. Menzel und M. Maurer, "Ontology based Scene Creation for the Development of Automated Vehicles", in *Intelligent Vehicles Symposium (IV)*, IEEE, 2018, S. 1813–1820. DOI: 10.1109/IVS.2018.8500632.

[194] J. Bock, R. Krajewski, L. Eckstein, J. Klimke, J. Sauerbier und A. Zlocki, "Data Basis for Scenario-Based Validation of HAD on Highways", in *Aachen Colloquium Automobile and Engine Technology*, 2018.

[195] A. Zlocki, D. Raudszus, J. Bock und L. Eckstein, "Methoden zur Absicherung von Komponenten und Funktionen", *ATZextra*, Jg. 25, Nr. 1, S. 28–33, 2020, ISSN: 2195-1454. DOI: 10.1007/s35778-020-0111-6.

[196] M. Elgharbawy, A. Schwarzhaupt, M. Frey und F. Gauterin, "Ontology-based adaptive testing for automated driving functions using data mining techniques", *Transportation Research Part F: Traffic Psychology and Behaviour*, Jg. 66, S. 234–251, 2019. DOI: 10.1016/j.trf.2019.07.021.

[197] M. Dupuis, E. Hekele und A. Biehn, *OpenDRIVE Format Specification, Rev. 1.5*, 2019.

[198] ASAM, *ASAM OpenDRIVE 1.7.0*, 2021.

[199] ——, *OpenSCENARIO User Guide, Version 1.2.0*, 2022.

[200] ——, *ASAM OpenSCENARIO 2.0.0*, 2022.

[201] A. Pütz, A. Zlocki, J. Bock und L. Eckstein, "System validation of highly automated vehicles with a database of relevant traffic scenarios", in *ITS European Congress*, 2017.

[202] 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", in *Workshop Fahrerassistenzsysteme und automatisiertes Fahren*, Uni-DAS e. V., Hrsg., 2018.

[203] G. Bagschik, T. Menzel, C. Körner und M. Maurer, "Wissensbasierte Szenariengenerierung für Betriebsszenarien auf deutschen Autobahnen", in *Workshop Fahrerassistenzsysteme und automatisiertes Fahren*, Uni-DAS e. V., Hrsg., 2018.

[204] P. Elspas, J. Langner, M. Aydinbas, J. Bach und E. Sax, "Leveraging Regular Expressions for Flexible Scenario Detection in Recorded Driving Data", in *International Symposium on Systems Engineering (ISSE)*, IEEE, 2020, S. 1–8. DOI: 10.1109/ISSE49799.2020.9272025.

[205] D. Perdomo Lopez, R. Waldmann, C. Joerdens und R. Rojas, "Scenario Interpretation based on Primary Situations for Automatic Turning at Urban Intersections", in *Proceedings of the 3rd International Conference on Vehicle Technology and Intelligent Transport Systems*, Scitepress, 2017, S. 15–23. DOI: 10.5220/0006150300150023.

[206] M. Benmimoun, F. Fahrenkrog, A. Zlocki und L. Eckstein, "Incident Detection Based on Vehicle CAN-Data Within the Large Scale Field Operational Test "euroFOT"", in *International Technical Conference on the Enhanced Safety of Vehicles (ESV)*, 2011.

[207] R. Gruner, P. Henzler, G. Hinz, C. Eckstein und A. Knoll, "Spatio-temporal representation of driving scenarios and classification using neural networks", in *Intelligent Vehicles Symposium (IV)*, IEEE, 2017, S. 1782–1788. DOI: 10.1109/ivs.2017.7995965.

[208] H. Beglerovic, T. Schloemicher, S. Metzner und M. Horn, "Deep Learning Applied to Scenario Classification for Lane-Keep-Assist Systems", *Applied Sciences*, Jg. 8, Nr. 12, S. 2590, 2018. DOI: 10.3390/app8122590.

[209] J. Firl, "Probabilistic Maneuver Recognition in Traffic Scenarios", Dissertation, Karlsruher Institut für Technologie, 2014.

[210] B. Higgs und M. Abbas, “Segmentation and Clustering of Car-Following Behavior: Recognition of Driving Patterns”, *IEEE Transactions on Intelligent Transportation Systems*, Jg. 16, Nr. 1, S. 81–90, 2015, ISSN: 1524-9050. DOI: 10.1109/tits.2014.2326082.

[211] X. Ma und I. Andreasson, “Behavior Measurement, Analysis, and Regime Classification in Car Following”, *IEEE Transactions on Intelligent Transportation Systems*, Jg. 8, Nr. 1, S. 144–156, 2007, ISSN: 1524-9050. DOI: 10.1109/tits.2006.883111.

[212] M. Herrmann, “Methodology for the generation and execution of scenarios for the virtual driving test with automated driving functions”, in *Automatisiertes Fahren 2019*, Ser. Proceedings, Wiesbaden: Springer Vieweg, 2019, S. 145–154. DOI: 10.1007/978-3-658-27990-5_13.

[213] C. Amersbach und H. Winner, “Funktionale Dekomposition-Ein Beitrag zur Überwindung der Parameterraumexplosion bei der Validation von höher automatisiertem Fahren”, in *Workshop Fahrerassistenzsysteme und automatisiertes Fahren*, Uni-DAS e. V., Hrsg., 2018.

[214] ——, “Defining Required and Feasible Test Coverage for Scenario-Based Validation of Highly Automated Vehicles”, in *Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2019, S. 425–430. DOI: 10.1109/ITSC.2019.8917534.

[215] M. Rasch, P. Ubben, T. Most, V. Bayer und R. Niemeier, “Safety Assessment and Uncertainty Quantification of Automated Driver Assistance Systems using Stochastic Analysis Methods”, in *NAFEMS World Congress*, 2019.

[216] M. O’Kelly, A. Sinha, H. Namkoong, J. Duchi und R. Tedrake, “Scalable End-to-End Autonomous Vehicle Testing via Rare-event Simulation”, in *Advances in Neural Information Processing Systems 31*, 2018.

[217] E. de Gelder und J.-P. Paardekooper, “Assessment of Automated Driving Systems using real-life scenarios”, in *Intelligent Vehicles Symposium (IV)*, IEEE, 2017, S. 589–594. DOI: 10.1109/IVS.2017.7995782.

[218] D. Zhao, “Accelerated Evaluation of Automated Vehicles in Car-Following Maneuvers”, Dissertation, 2016.

[219] N. Weber, D. Frerichs und U. Eberle, "A simulation-based, statistical approach for the derivation of concrete scenarios for the release of highly automated driving functions", in *Automotive meets Electronics*, 2020, S. 1–6.

[220] J. Bach, J. Langner, S. Otten, E. Sax und M. Holzapfel, "Test scenario selection for system-level verification and validation of geolocation-dependent automotive control systems", in *International Conference on Engineering, Technology and Innovation (ICE/ITMC)*, IEEE, 2017, S. 203–210. DOI: 10.1109/ICE.2017.8279890.

[221] J. Langner, J. Bach, L. Ries, S. Otten, M. Holzapfel und E. Sax, "Estimating the Uniqueness of Test Scenarios derived from Recorded Real-World-Driving-Data using Autoencoders", in *Intelligent Vehicles Symposium (IV)*, IEEE, 2018, S. 1860–1866. DOI: 10.1109/IVS.2018.8500464.

[222] F. Schuldt, T. Menzel und M. Maurer, "Eine Methode für die Zuordnung von Testfällen für automatisierte Fahrfunktionen auf X-in-the-Loop Verfahren im modularen virtuellen Testbaukasten", in *Workshop Fahrerassistenzsysteme und automatisiertes Fahren*, Uni-DAS e. V., Hrsg., 2015, S. 171–182.

[223] E. Böde, M. Büker, U. Eberle, M. Fränzle, S. Gerwinn und B. Kramer, "Efficient Splitting of Test and Simulation Cases for the Verification of Highly Automated Driving Functions", in *Computer Safety, Reliability, and Security*, B. Gallina, A. Skavhaug und F. Bitsch, Hrsg., Cham: Springer International Publishing, 2018, S. 139–153. DOI: 10.1007/978-3-319-99130-6_10.

[224] R. Krajewski, J. Bock, L. Kloeker und L. Eckstein, "The highD Dataset: A Drone Dataset of Naturalistic Vehicle Trajectories on German Highways for Validation of Highly Automated Driving Systems", in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2018, S. 2118–2125. DOI: 10.1109/ITSC.2018.8569552.

[225] R. Pfeffer, K. Bredow und E. Sax, "Trade-off analysis using synthetic training data for neural networks in the automotive development process", in *Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2019, S. 4115–4120. DOI: 10.1109/ITSC.2019.8917249.

[226] C. Neurohr, L. Westhofen, T. Henning, T. de Graaff, E. Mohlmann und E. Bode, "Fundamental Considerations around Scenario-Based Testing for Automated Driving", in *Intelligent Vehicles Symposium (IV)*, IEEE, 2020, S. 121–127. DOI: 10.1109/IV47402.2020.9304823.

[227] C. King, "Bewertung von Fahrerassistenzsystemen im Umfeld des szenariobasierten Testens", Dissertation, Karlsruher Institut für Technologie, 2023.

[228] W. L. Oberkampf und M. F. Barone, "Measures of agreement between computation and experiment: Validation metrics", *Journal of Computational Physics*, Jg. 217, Nr. 1, S. 5–36, 2006, ISSN: 00219991. DOI: 10.1016/j.jcp.2006.03.037.

[229] D. Ao, Z. Hu und S. Mahadevan, "Dynamics Model Validation Using Time-Domain Metrics", *Journal of Verification, Validation and Uncertainty Quantification*, Jg. 2, Nr. 1, 2017, ISSN: 2377-2158. DOI: 10.1115/1.4036182.

[230] C. Esselborn, M. Eckert, M. Holzäpfel, E. Wahl und E. Sax, "Method for a scenario-based and weighted assessment of map-based advanced driving functions", in *20. Internationales Stuttgarter Symposium*, Ser. ATZ live, Springer Vieweg, 2020, S. 193–207. DOI: 10.1007/978-3-658-30995-4_22.

[231] H. Winner, S. Geyer und M. Sefati, "Maße für den Sicherheitsgewinn von Fahrerassistenzsystemen", in *6. Darmstädter Kolloquium Mensch + Fahrzeug*, Stuttgart: Ergonomia Verlag, 2013, S. 63–82.

[232] P. Junietz, J. Schneider und H. Winner, "Metrik zur Bewertung der Kritikalität von Verkehrssituationen und -szenarien", in *Workshop Fahrerassistenzsysteme und automatisiertes Fahren*, Uni-DAS e. V., Hrsg., 2017, S. 149–160.

[233] J. C. Hayward, "Near miss determination through use of a scale of danger", 1972.

[234] J. Hillenbrand, K. Kroschel und V. Schmid, "Situation assessment algorithm for a collision prevention assistant", in *Intelligent Vehicles Symposium (IV)*, IEEE, 2005, S. 459–465. DOI: 10.1109/IVS.2005.1505146.

[235] J. Hillenbrand, A. M. Spieker und K. Kroschel, "A Multilevel Collision Mitigation Approach—Its Situation Assessment, Decision Making, and Performance Tradeoffs", *IEEE Transactions on Intelligent Transportation Systems*, Jg. 7, Nr. 4, S. 528–540, 2006, ISSN: 1524-9050. DOI: 10.1109/TITS.2006.883115.

[236] W. Wachenfeld, P. Junietz, R. Wenzel und H. Winner, "The worst-time-to-collision metric for situation identification", in *Intelligent Vehicles Symposium (IV)*, IEEE, 2016, S. 729–734. DOI: 10.1109/IVS.2016.7535468.

[237] R. Karlsson, J. Jansson und F. Gustafsson, "Model-based statistical tracking and decision making for collision avoidance application", in *American Control Conference (ACC)*, American Automatic Control Council, 2004, S. 3435–3440. DOI: 10.23919/ACC.2004.1384441.

[238] M. Brannstrom, J. Sjoberg und E. Coelingh, "A situation and threat assessment algorithm for a rear-end collision avoidance system", in *Intelligent Vehicles Symposium (IV)*, IEEE, 2008, S. 102–107. DOI: 10.1109/IVS.2008.4621250.

[239] J. Eggert, "Predictive risk estimation for intelligent ADAS functions", in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2014, S. 711–718. DOI: 10.1109/ITSC.2014.6957773.

[240] T. J. Gordon, L. P. Kostyniuk, P. E. Green, M. A. Barnes, D. F. Blower, S. E. Bogard, A. D. Blankespoor, D. J. LeBlanc, B. R. Cannon und S. B. McLaughlin, *A Multivariate Analysis of Crash and Naturalistic Driving Data in Relation to Highway Factors*. Washington, D.C.: Transportation Research Board, 2013. DOI: 10.17226/22849.

[241] J. Eggert und T. Puphal, "Continuous Risk Measures for Driving Support", *International Journal of Automotive Engineering*, Jg. 9, Nr. 3, S. 130–137, 2018, ISSN: 2185-0984. DOI: 10.20485/jsaeijae.9.3_130.

[242] M. Schreier, "Bayesian Environment Representation, Prediction, and Criticality Assessment for Driver Assistance Systems", Dissertation, TU Darmstadt, 2015.

[243] A. Eidehall und L. Petersson, "Statistical Threat Assessment for General Road Scenes Using Monte Carlo Sampling", *IEEE Transactions on Intelligent Transportation Systems*, Jg. 9, Nr. 1, S. 137–147, 2008, ISSN: 1524-9050. DOI: 10.1109/TITS.2007.909241.

[244] P. Junietz, F. Bonakdar, B. Klamann und H. Winner, "Criticality Metric for the Safety Validation of Automated Driving using Model Predictive Trajectory Optimization", in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2018, S. 60–65. DOI: 10.1109/ITSC.2018.8569326.

[245] P. Songchitruksa und A. P. Tarko, "The extreme value theory approach to safety estimation", *Accident Analysis and Prevention*, Jg. 38, Nr. 4, S. 811–822, 2006. DOI: 10.1016/j.aap.2006.02.003.

[246] L. Fahrmeir, *Multivariate statistische Verfahren*, 2. Aufl. De Gruyter, 1996. DOI: 10.1515/9783110816020.

[247] P. Elspas, Y. Klose, S. Isele, J. Bach und E. Sax, "Time Series Segmentation for Driving Scenario Detection with Fully Convolutional Networks", in *International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS)*, SCITEPRESS, 2021, S. 56–64. DOI: 10.5220/0010404700560064.

[248] J. Kerber, S. Wagner, K. Groh, D. Notz, T. Kühbeck, D. Watzenig und A. Knoll, "Clustering of the Scenario Space for the Assessment of Automated Driving", in *2020 IEEE Intelligent Vehicles Symposium (IV)*, 2020, S. 578–583. DOI: 10.1109/IV47402.2020.9304646.

[249] H. Watanabe, "Methodik zur Determinierung repräsentativer und relevanter Testszenarien für prädiktive Sicherheitsfunktionen", Dissertation, TU Dresden, 2022.

[250] C. King, T. Braun, C. Braess, J. Langner und E. Sax, "Capturing the Variety of Urban Logical Scenarios from Bird-view Trajectories", in *International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS)*, SCITEPRESS, 2021, S. 471–480. DOI: 10.5220/0010441204710480.

[251] T. Braun, L. Ries, M. Hesche, S. Otten und E. Sax, "Maneuver-based Visualization of Similarities between Recorded Traffic Scenarios", in *International Conference on Data Science, Technology and Applications*, SCITEPRESS, 2022, S. 236–244. DOI: 10.5220/0011140600003269.

[252] S. Hartmann, "The World as a Process", in *Modelling and Simulation in the Social Sciences from the Philosophy of Science Point of View*, W. Leinfellner, G. Eberlein und R. Hegselmann, Hrsg., Dordrecht: Springer Netherlands, 2010, S. 77–100. DOI: 10.1007/978-94-015-8686-3_5.

[253] J. M. Durán, "What is a Simulation Model?", *Minds and Machines*, Jg. 30, Nr. 3, S. 301–323, 2020, ISSN: 0924-6495. DOI: 10.1007/s11023-020-09520-z.

[254] S. Schlesinger, R. Crosbie, R. Gagne, G. Innis, C. Lalwani, J. Loch, R. Sylvester, R. Wright, N. Kheir und D. Batros, "Terminology for model credibility", *SIMULATION*, Jg. 32, Nr. 3, S. 103–104, 1979, ISSN: 0037-5497. DOI: 10.1177/003754977903200304.

[255] S. Riedmaier, B. Danquah, B. Schick und F. Diermeyer, "Unified Framework and Survey for Model Verification, Validation and Uncertainty Quantification", *Archives of Computational Methods in Engineering*, 2020, ISSN: 1134-3060. DOI: 10.1007/s11831-020-09473-7.

[256] S. I. Gass und L. S. Joel, "Concepts of model confidence", *Computers & Operations Research*, Jg. 8, Nr. 4, S. 341–346, 1981, ISSN: 03050548. DOI: 10.1016/0305-0548(81)90019-8.

[257] R. G. Sargent, "Verification and validation of simulation models", in *Winter Simulation Conference (WSC)*, IEEE, 2010, S. 166–183. DOI: 10.1109/WSC.2010.5679166.

[258] L. W. Schruben, "Establishing the credibility of simulations", *SIMULATION*, Jg. 34, Nr. 3, S. 101–105, 1980, ISSN: 0037-5497. DOI: 10.1177/003754978003400310.

[259] W. L. Oberkampf, S. M. DeLand, B. M. Rutherford, K. V. Diegert und K. F. Alvin, "Error and uncertainty in modeling and simulation", *Reliability Engineering & System Safety*, Jg. 75, Nr. 3, S. 333–357, 2002, ISSN: 09518320. DOI: 10.1016/S0951-8320(01)00120-X.

[260] C. J. Roy und W. L. Oberkampf, "A comprehensive framework for verification, validation, and uncertainty quantification in scientific computing", *Computer Methods in Applied Mechanics and Engineering*, Jg. 200, Nr. 25-28, S. 2131–2144, 2011, ISSN: 00457825. DOI: 10.1016/j.cma.2011.03.016.

[261] N. Hirsenkorn, H. Kolsi, M. Selmi, A. Schaermann, T. Hanke, A. Rauch, R. Rasshofer und E. Biebl, "Learning sensor models for virtual test and development", in *Workshop Fahrerassistenzsysteme und automatisiertes Fahren*, Uni-DAS e. V., Hrsg., 2017, S. 115–124.

[262] T. Eder, A. Prinz, L. Brabetz und E. Biebl, "Szenarienbasierte Validierung eines hybriden Radarmodells für Test und Absicherung automatisierter Fahrfunktionen", in *Automobil-Sensorik 3*, T. Tille, Hrsg., Berlin, Heidelberg: Springer, 2020, S. 21–43. DOI: 10.1007/978-3-662-61260-6_1.

[263] P. Rosenberger, M. Holder, S. Huch, H. Winner, T. Fleck, M. R. Zofka, J. M. Zollner, T. D'hondt und B. Wassermann, "Benchmarking and Functional Decomposition of Automotive Lidar Sensor Models", in *Intelligent Vehicles Symposium (IV)*, IEEE, 2019, S. 632–639. DOI: 10.1109/IVS.2019.8814081.

[264] P. Rosenberger, "Metrics for Specification, Validation, and Uncertainty Prediction for Credibility in Simulation of Active Perception Sensor Systems", Dissertation, TU Darmstadt, 2023.

[265] A. Schaermann, A. Rauch, N. Hirsenkorn, T. Hanke, R. Rasshofer und E. Biebl, "Validation of vehicle environment sensor models", in *Intelligent Vehicles Symposium (IV)*, IEEE, 2017, S. 405–411. DOI: 10.1109/IVS.2017.7995752.

[266] M.-O. Shin, G.-M. Oh, S.-W. Kim und S.-W. Seo, "Real-Time and Accurate Segmentation of 3-D Point Clouds Based on Gaussian Process Regression", *IEEE Transactions on Intelligent Transportation Systems*, Jg. 18, Nr. 12, S. 3363–3377, 2017, ISSN: 1524-9050. DOI: 10.1109/TITS.2017.2685523.

[267] M. Viehof und H. Winner, "Stand der Technik und der Wissenschaft: Modellvalidierung im Anwendungsbereich der Fahrdynamiksimulation", Forschungsbericht, TU Darmstadt, 2017.

[268] R. Wade-Allen, J. P. Chrstos, G. Howe, D. H. Klyde und T. J. Rosenthal, "Validation of a non-linear vehicle dynamics simulation for limit handling", *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, Jg. 216, Nr. 4, S. 319–327, 2002, ISSN: 0954-4070. DOI: 10.1243/0954407021529147.

[269] P. J. Mcnaull, D. A. Guenther, G. J. Heydinger, P. A. Grygier und M. K. Salaani, "Validation and Enhancement of a Heavy Truck Simulation Model with an Electronic Stability Control Model", in *SAE Technical Paper Series*, Ser. SAE Technical Paper Series, SAE International, 2010. DOI: 10.4271/2010-01-0104.

[270] G. J. Heydinger, C. Schwarz, M. K. Salaani und P. A. Grygier, "Model Validation of the 2006 BMW 330i for the National Advanced Driving Simulator", in *SAE Technical Paper Series*, Ser. SAE Technical Paper Series, SAE International, 2007. DOI: 10.4271/2007-01-0817.

[271] E. Kutluay, "Development and Demonstration of a Validation Methodology for Vehicle Lateral Dynamics Simulation Models", Dissertation, TU Darmstadt, 2012.

[272] J. Klemmer, J. Lauer, V. Formanski, R. Fontaine, P. Kilian, S. Sinsel, A. Erbes und J. Zäpf, "Definition and Application of a Standard Verification and Validation Process for Dynamic Vehicle Simulation Models", *SAE International Journal of Materials and Manufacturing*, Jg. 4, Nr. 1, S. 743–758, 2011, ISSN: 1946-3987. DOI: 10.4271/2011-01-0519.

[273] ISO, Hrsg., *ISO 19365:2016*, Geneva: International Organization for Standardization, 2016.

[274] H. Sarin, M. Kokkolaras, G. Hulbert, P. Papalambros, S. Barbat und R.-J. Yang, "A Comprehensive Metric for Comparing Time Histories in Validation of Simulation Models With Emphasis on Vehicle Safety Applications", in *International Design Engineering Conferences and Computers and Information in Engineering Conference*, ASME, 2008, S. 1275–1286. DOI: 10.1115/DETC2008-49669.

[275] M. Muresan und D. Pitica, "Software in the Loop environment reliability for testing embedded code", in *International Symposium for Design and Technology of Electronic Packaging (SIITME)*, IEEE, 2012, S. 325–328. DOI: 10.1109/SIITME.2012.6384402.

[276] G. Tibba, C. Malz, C. Stoermer, N. Nagarajan, L. Zhang und S. Chakraborty, "Testing automotive embedded systems under X-in-the-loop setups", in *Proceedings of the 35th International Conference on Computer-Aided Design*, ACM, 2016, S. 1–8. DOI: 10.1145/2966986.2980076.

[277] R. Linssen, F. Uphaus und J. Mauss, “Software-in-the-Loop at the junction of software development and drivability calibration”, in *16. Internationales Stuttgarter Symposium, Automobil- und Motorentechnik*, Ser. Proceedings, Wiesbaden: Springer Vieweg, 2016, S. 451–465. DOI: 10.1007/978-3-658-13255-2_32.

[278] I. Raghupatruni, T. Goeppel, M. Atak, J. Bou und T. Huber, “Empirical Testing of Automotive Cyber-Physical Systems with Credible Software-in-the-Loop Environments”, in *International Conference on Connected Vehicles and Expo (ICCVE)*, IEEE, S. 1–6. DOI: 10.1109/ICCVE45908.2019.8965169.

[279] K. Davis und J. G. Hayes, “Analysis of electric vehicle powertrain simulators for fuel consumption calculations”, in *International Conference on Electrical Systems for Aircraft, Railway, Ship Propulsion and Road Vehicles & International Transportation Electrification Conference (ESARS-ITEC)*, IEEE, 2016, S. 1–6. DOI: 10.1109/ESARS-ITEC.2016.7841414.

[280] B. Danquah, S. Riedmaier, J. Rühm, S. Kalt und M. Lienkamp, “Statistical Model Verification and Validation Concept in Automotive Vehicle Design”, *Procedia CIRP*, Jg. 91, S. 261–270, 2020, ISSN: 22128271. DOI: 10.1016/j.procir.2020.02.175.

[281] M. I. Kuang, M. Fodor und D. Hrovat, “Hydraulic brake system modeling and control for active control of vehicle dynamics”, in *American Control Conference (ACC)*, IEEE, 1999, S. 4538–4542. DOI: 10.1109/ACC.1999.786447.

[282] M. Chou, X. Xia und C. Kayser, “Modelling and model validation of heavy-haul trains equipped with electronically controlled pneumatic brake systems”, *Control Engineering Practice*, Jg. 15, Nr. 4, S. 501–509, 2007, ISSN: 09670661. DOI: 10.1016/j.conengprac.2006.09.006.

[283] R. Nippold und P. Wagner, “Calibration of car-following models with single- and multi-step approaches”, in *Winter Simulation Conference (WSC)*, IEEE, 2012, S. 1–11. DOI: 10.1109/wsc.2012.6465070.

[284] S. Wagner, K. Groh, T. Kuhbeck und A. Knoll, “Towards Cross-Verification and Use of Simulation in the Assessment of Automated Driving”, in *Intelligent Vehicles Symposium (IV)*, IEEE, 2019. DOI: 10.1109/IVS.2019.8814268.

[285] O. Balci, “Principles and techniques of simulation validation, verification, and testing”, in *Winter Simulation Conference (WSC)*, IEEE, 1995, S. 147–154. DOI: 10.1109/WSC.1995.478717.

[286] C. Thule, C. Gomes, J. Deantoni, P. G. Larsen, J. Brauer und H. Vangheluwe, “Towards the Verification of Hybrid Co-simulation Algorithms”, in *Software Technologies: Applications and Foundations*, Ser. Lecture Notes in Computer Science, M. Mazzara, I. Ober und G. Salaün, Hrsg., Cham: Springer, 2018, S. 5–20. DOI: 10.1007/978-3-030-04771-9_1.

[287] G. Stevens und S. Atamturktur, “Mitigating Error and Uncertainty in Partitioned Analysis: A Review of Verification, Calibration and Validation Methods for Coupled Simulations”, *Archives of Computational Methods in Engineering*, Jg. 24, Nr. 3, S. 557–571, 2017, ISSN: 1134-3060. DOI: 10.1007/s11831-016-9177-0.

[288] R. Pfeffer, “Szenariobasierte simulationsgestützte funktionale Absicherung hochautomatisierter Fahrfunktionen durch Nutzung von Realdaten”, Dissertation, Karlsruher Institut für Technologie, 2020.

[289] W. Wachenfeld und H. Winner, “Die Freigabe des autonomen Fahrens”, in *Autonomes Fahren*, M. Maurer, J. C. Gerdes, B. Lenz und H. Winner, Hrsg., Berlin, Heidelberg: Springer, 2015, S. 439–464. DOI: 10.1007/978-3-662-45854-9_21.

[290] S. Riedmaier, T. Ponn, D. Ludwig, B. Schick und F. Diermeyer, “Survey on Scenario-Based Safety Assessment of Automated Vehicles”, *IEEE Access*, Jg. 8, 2020, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.2993730.

[291] R. Y. Rubinstein und D. P. Kroese, *Simulation and the Monte Carlo Method*, eng, 2. Aufl., Ser. Wiley series in probability and mathematical statistics. Hoboken, NJ, USA: John Wiley & Sons, Inc, 2007, 345 S. DOI: 10.1002/9780470230381.

[292] J. Hedderich und L. Sachs, “Schätzen”, in *Angewandte Statistik, Methodensammlung mit R*, J. Hedderich und L. Sachs, Hrsg., 16. Aufl., Berlin, Heidelberg: Springer, 2018, S. 325–444. DOI: 10.1007/978-3-662-56657-2_6.

[293] L. A. Goodman, “On Simultaneous Confidence Intervals for Multinomial Proportions”, *Technometrics*, Jg. 7, Nr. 2, S. 247, 1965, ISSN: 00401706. DOI: 10.2307/1266673.

[294] S. K. Thompson, "Sample Size for Estimating Multinomial Proportions", *The American Statistician*, Jg. 41, Nr. 1, S. 42, 1987, ISSN: 00031305. DOI: 10.2307/2684318.

[295] H. Sakoe und S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition", *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Jg. 26, Nr. 1, S. 43–49, 1978, ISSN: 0096-3518. DOI: 10.1109/TASSP.1978.1163055.

[296] M. Müller, Hrsg., *Information Retrieval for Music and Motion*, Berlin, Heidelberg: Springer, 2007. DOI: 10.1007/978-3-540-74048-3.

[297] D. S. Young, "tolerance : An R Package for Estimating Tolerance Intervals", *Journal of Statistical Software*, Jg. 36, Nr. 5, 2010. DOI: 10.18637/jss.v036.i05.

[298] G. A. Barnard, "Significance tests for 2 X 2 tables", *Biometrika*, Jg. 34, Nr. 1-2, S. 123–138, 1947, ISSN: 0006-3444. DOI: 10.1093/biomet/34.1-2.123.

[299] J. Janssen und W. Laatz, Hrsg., *Statistische Datenanalyse mit SPSS, Eine anwendungsorientierte Einführung in das Basissystem und das Modul Exakte Tests*, 9. Aufl., Springer eBook Collection, Berlin, Heidelberg: Springer Gabler, 2017. DOI: 10.1007/978-3-662-53477-9.

[300] R. A. Fisher, *Statistical Methods for Research Workers*, 5. Aufl. Edinburgh: Oliver and Boyd Ltd., 1934.

[301] J. Hedderich und L. Sachs, "Hypothesentest", in *Angewandte Statistik, Methodensammlung mit R*, J. Hedderich und L. Sachs, Hrsg., 16. Aufl., Berlin, Heidelberg: Springer, 2018, S. 445–795. DOI: 10.1007/978-3-662-56657-2_7.

[302] A. S. Mato und A. M. Andrés, "Simplifying the calculation of the P-value for Barnard's test and its derivatives", *Statistics and Computing*, Jg. 7, Nr. 2, S. 137–143, 1997, ISSN: 09603174. DOI: 10.1023/A:1018573716156.

[303] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman,

I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa und P. van Mulbregt, “SciPy 1.0: fundamental algorithms for scientific computing in Python”, *Nature methods*, Jg. 17, Nr. 3, S. 261–272, 2020. DOI: 10.1038/s41592-019-0686-2.

[304] ISO, Hrsg., *ISO 22733-1:2022*, Geneva: International Organization for Standardization, 2022.

[305] J. Stoll, A. Schneider, M. Wisch, P. Seiniger und T. Schaller, *PROactive Safety for PEdestrians and CyclisTs Deliverable D3.1*, 2016.

[306] Vector Informatik GmbH. (2023). “Effizientes Speichern von Messdaten mit dem MDF-Format”, Adresse: https://www.vector.com/de/de/produkte/anwendungsgebiete/steuergeraete-kalibrierung/messen/mdf/ (besucht am 20. 03. 2023).

[307] Digitalwerk. (2023). “ADTF3 Guides”, Adresse: https://support.digitalwerk.net/adtf/v3/guides/index.html (besucht am 20. 03. 2023).

[308] S. Seabold und J. Perktold, “statsmodels: Econometric and statistical modeling with python”, in *9th Python in Science Conference*, 2010.

[309] T. Giorgino, “Computing and Visualizing Dynamic Time Warping Alignments in R : The dtw Package”, *Journal of Statistical Software*, Jg. 31, Nr. 7, 2009. DOI: 10.18637/jss.v031.i07.

Eigene wissenschaftliche Beiträge

[310] F. Reisgys, M. Elgharbawy, A. Schwarzhaupt und E. Sax, “Argumentation on ADAS Simulation Validity using Aleatory and Epistemic Uncertainty Estimation”, in *Proceedings of the Driving Simulation Conference Europe VR*, Driving Simulation Association, 2021, S. 25–32.

[311] F. Reisgys, J. Plaum, A. Schwarzhaupt und E. Sax, “Scenario-based X-in-the-Loop Test for Development of Driving Automation”, in *Workshop Fahrerassistenzsysteme und automatisiertes Fahren*, Uni-DAS e. V., Hrsg., 2022.

[312] ——, “Validation and Plausibilization of X-in-the-Loop Tests for Driving Automation”, in *International Technical Conference on the Enhanced Safety of Vehicles (ESV)*, 2023.

[313] F. Reisgys, C. Steinhauser, A. Schwarzhaupt und E. Sax, “How Uncertainty Affects Test Results for Driving Automation”, in *International Automated Vehicle Validation Conference (IAVVC)*, IEEE, 2023.

[314] T. Braun, J. Fuchs, F. Reisgys, L. Ries, J. Plaum, B. Schütt und E. Sax, “A Review of Scenario Similarity Measures for Validation of Highly Automated Driving”, in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2023.

[315] C. Ballarin, F. Reisgys, I. Scherhaufer und C. Tresp, “Fahrerassistenzsysteme im Nutzfahrzeug”, in *Handbuch Assistiertes und Automatisiertes Fahren*, H. Winner, K. C. J. Dietmayer, L. Eckstein, M. Jipp, M. Maurer und C. Stiller, Hrsg., Wiesbaden: Springer Fachmedien, 2024 (angekündigt).

Eigene Patente

[316] F. Reisgys, “Verfahren zum effizienten szenariobasierten Testen eines automatisierten Fahrsystems: Offenlegungsschrift”, 10 2023 102 523.6, 2.02.2023.

Betreute Abschlussarbeiten

[317] R. Brenner, “Konzeption und Implementierung einer modularisierbaren Hardware-in-the-Loop-Umgebung für Fahrerassistenzsysteme”, Bachelorarbeit, Hochschule Aalen, 2022.

[318] P. Freimann, “Modellierung von Unsicherheiten des Ego-Fahrverhaltens in einer X-in-the-Loop Umgebung für Fahrerassistenzsysteme”, Masterarbeit, Technische Universität Braunschweig, 2023.

[319] F. Janßen, “Entwicklung eines Konzepts zur Modellierung von Unsicherheit in X-in-the-Loop-Tests für Fahrerassistenzsysteme”, Bachelorarbeit, Hochschule Konstanz, 2022.

[320] J. Keller, “Konzeption und Implementierung eines Sampling-Algorithmus für den Szenariobasierten XiL-Test für Fahrerassistenzsysteme”, Bachelorarbeit, Hochschule Esslingen, 2022.

[321] J. Li, “Definition und Cluster konkreter Szenarien für den XiL-Test von Fahrerassistenzsystemen”, Masterarbeit, Universität Stuttgart, 2022.

[322] T. Müller, “Szenariobasierte Bewertung der System Under Test-Einbindung in X-in-the-Loop-Testumgebungen für Fahrerassistenzsysteme”, Bachelorarbeit, Karlsruher Institut für Technologie, 2021.

[323] T. Münster, “Konzeption eines Frameworks zur Prototypen-Implementierung eines Moving-off-Information-System auf Basis der VRDU3-Autorar-Architektur konform der General-Safety-Regulation für Nutzfahrzeuge”, Masterarbeit, Hochschule Esslingen, 2021.

[324] T. Reichle, “Entwicklung eines metrikenbasierten Sampling-Algorithmus für den szenariobasierten XiL-Test von Fahrerassistenzsystemen”, Bachelorarbeit, Hochschule Esslingen, 2022.

[325] M. Rohrmann, “Fehlerprädiktion einer X-in-the-Loop Simulationsumgebung mittels künstlicher Intelligenz”, Masterarbeit, Universität Stuttgart, 2022.

[326] L. Sommer, “Entwicklung und Evaluation eines Szenariodistanzmaßes für den Test automatisierter Fahrzeuge”, Masterarbeit, Karlsruher Institut für Technologie, 2022.