TECHNISCHE UNIVERSITÄT MÜNCHEN
Professur für Höchstfrequenztechnik

Systematische Bedatung und Bewertung umfelderfassender Sensormodelle

Alexander Schaermann

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor–Ingenieurs

genehmigten Dissertation.

Vorsitzender:
Prof. Dr. Marc Tornow
Prüfer der Dissertation:
1. Prof. Dr.-Ing. habil. Erwin Biebl
2. Hon.-Prof. Dr. Martin Daumer

Die Dissertation wurde am 12.09.2019 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 02.03.2020 angenommen.

All models are wrong, but some are useful.

George E. P. Box

Kurzfassung

Beim Test und der Absicherung von Fahrerassistenzsystemen ist Simulation eine vielversprechende Technologie. Eine der Herausforderungen ist dabei die Nachbildung des Verhaltens umfelderfassender Sensoren. Bevor jedoch die Sensormodelle zum Testen in der Simulation verwendet werden können, müssen diese validiert werden.

Die meisten heutzutage verwendeten Sensormodelle sind von datengetriebener Natur. Die Messdatenerfassung ist dabei eine Herausforderung. In dieser Arbeit werden deshalb Ansätze zur Reduktion des Parameterraums und Datenanzahl untersucht, ohne dabei die Güte eines Sensormodells maßgeblich zu beeinflussen. Des Weiteren wird eine Methode zur Validierung von phänomenologischen und physikalischen Sensormodellen entwickelt.

Abstract

Simulation is a promising technology for test and validation of advanced driver assistance systems. One of the challenges is the simulation of perception sensors. However, before the sensor models can be used for simulation, they must be validated.

Most sensor models used today are data–driven. Thereby is data acquisition a challenge. Therefore, approaches for reducing the parameter space and the number of necessary data points without significantly influencing the model quality are developed. Furthermore, a method for validation of perception sensor models with focus on phenomenological and physical sensor models is developed.

Danksagung

Die vorliegende Arbeit wurde an der Professur für Höchstfrequenztechnik der Technischen Universität München in Kooperation mit der BMW AG verfasst. Ohne die Unterstützung zahlreicher Personen hätte sie in dieser Form nicht realisiert werden können. Für die vielfältig erfahrene Hilfe möchte ich mich an dieser Stelle sehr herzlich bedanken.

Mein besonderer Dank gilt zunächst meinem Doktorvater Univ.-Prof. Dr.-Ing. Erwin Biebl, der meine Arbeit stets mit viel Verständnis unterstützt hat. Des Weiteren möchte ich Prof. Biebl für die inhaltliche Ausrichtung und die thematische Eingrenzung danken.

Für vielfältige Unterstützung bin ich besonders Herrn Dr. Andreas Rauch verbunden. Er hat mir sehr wertvolle Hilfestellungen gegeben und meine Forschungen entscheidend unterstützt. Durch sein großes Engagement, seine fachlichen Hinweise und sein professionelles Lektorat hat Dr. Rauch wesentlich zum erfolgreichen Abschluss der Arbeit beigetragen.

Für die Beteiligung an den Publikationen gilt mein Dank Herrn Dr. Nils Hirsenkorn und Herrn Timo Hanke, die sich daran fachlich und inhaltlich beteiligt haben. Für die Unterstützung in organisatorischen Belangen möchte ich mich bei Herrn Dr. Ralph Rasshofer bedanken.

Mein besonderer Dank gilt schließlich neben vielen Freunden und Verwandten meinen Eltern, meiner Schwester und meiner Lebensgefährtin, die mich mit großem Engagement unterstützt haben.

Inhaltsverzeichnis

1 Einleitung **1**
1.1 Hintergrund und Motivation 2
1.2 Ziele und Aufbau der Arbeit 3

2 Grundlagen der Sensormodellvalidierung **7**
2.1 Methoden der Sensorsimulation 7
2.1.1 Verwendete Datenformate 8
2.1.2 Ansätze der Sensormodellierung 11
2.2 Methoden der Validierung 14
2.2.1 Begriffe der Modellvalidierung 17
2.2.2 Auswahl relevanter Ansätze der Modellvalidierung 21
2.2.3 Anforderungen an die Modellvalidierung 26

3 Methoden zur Validierung und Bedatung umfelderfassender Sensormodelle **29**
3.1 Methode zur Validierung umfelderfassender Sensormodelle 30
3.2 Architekturvorschlag 34
3.3 Aufbereitung der Messdaten 37
3.3.1 Vorverarbeitung 38
3.3.2 Assoziation der Objekte 38
3.4 Verfahren zur Reduktion der Anzahl von Modellierungsgrößen 40
3.4.1 Ansätze zur Variablenauswahl 41
3.4.2 Umsetzung und Auswertung des Verfahrens 47
3.5 Verfahren zur systematischen Messdatenerfassung 50
3.5.1 Vorstellung des Ansatzes 50
3.5.2 Umsetzung und Auswertung des Verfahrens 53
3.6 Fazit zur Validierung und Bedatung von Sensormodellen 55

4 Methoden zur Bewertung phänomenologischer Sensormodelle **59**
4.1 Messfahrtbasierte Analyse 62
4.1.1 Vorstellung des Ansatzes 62
4.1.2 Umsetzung und Auswertung des Verfahrens 68
4.2 Autokorrelationsanalyse 72
4.2.1 Vorstellung des Ansatzes 73
4.2.2 Umsetzung und Auswertung des Verfahrens 74
4.3 Fazit zur Bewertung phänomenologischer Sensormodelle 77

5 Methoden zur Bewertung physikalischer Sensormodelle 79
5.1 Umgebungsmodellierung 81
5.2 Vorstellung des Ansatzes 85
5.3 Umsetzung und Auswertung des Verfahrens 92
5.4 Fazit zur Bewertung physikalischer Sensormodelle 100

6 Zusammenfassung und Ausblick 103
6.1 Zusammenfassung 103
6.2 Ausblick und abschließende Empfehlung 105

A Anhang 109
A.1 Ausschnitt des Testdatensatzes für Auswahl von Modellierungsgrößen . 109
A.2 Ergebnisse der Auswahl von Modellierungsgrößen 110
A.3 Algorithmus zur Rasterung des Sensorsichtbereichs 112
A.4 Der Dynamic Time Warping (DTW) Algorithmus 113
A.5 Ergebnisse der messfahrtbasierten Analyse 114
A.6 Zwischenschrittergebnisse des Umgebungsmodellierungsprozesses 115
A.7 Eigenschaften des verwendeten Light Detection and Ranging (Lidar)–Sensors 119
A.8 Ergebnisse der physikalischen Analyse eines Lidar–Sensormodells 120

Literaturverzeichnis 121

Eigene Veröffentlichungen 132

Abkürzungsverzeichnis 135

Symbolverzeichnis 137

1 Einleitung

Heutzutage wird das autonome Fahren in diversen Disziplinen, wie der Forschung, Industrie, Politik sowie der Gesellschaft im Allgemeinen, intensiv diskutiert [1–4]. Dafür gibt es zahlreiche Gründe. Die Privatkunden erwarten von der neuen Technologie mehr Komfort und Sicherheit. Kommt es beim Autofahren zu Situationen, die keine Freude machen oder in denen der Fahrer temporär nicht fahrtauglich ist, so implizieren autonome Fahrsysteme neue Chancen. Durch die steigende Sicherheit auf den Straßen sinkt die Zahl der Unfalltoten in Deutschland jährlich [5]. International starben jedoch 2010 laut „World Health Organisation“ 1,24 Millionen Menschen bei Verkehrsunfällen [6]. Dabei sind über 90 Prozent aller Unfälle auf menschliches Versagen zurückzuführen [7]. Schätzungen zufolge sollen durch autonome Fahrzeuge im Zeitraum von Jahr 2035 bis 2050 etwa 585.000 Menschenleben gerettet und rund 234 Milliarden US–Dollar an Kosten eingespart werden [8]. Dies ist Anlass genug, um neue Technologien zu erforschen, die das Potential haben, die Sicherheit auf den Straßen und den Komfort der Passagiere zu erhöhen.

Die Industrie erwartet vom autonomen Fahren neue Transportmöglichkeiten sowie Zugang zu neuen Geschäftsmodellen. Etwa ein Drittel des Gesamtverkehrs ist Wirtschaftsverkehr, wovon wiederum rund ein Drittel auf den Transport von Gütern entfällt [9]. Ein mögliches Szenario in diesem Bereich ist der Transport von Waren nachtsüber und von Menschen während des Tages [4]. Des Weiteren wird erwartet, dass durch den Einsatz von vollautonomen Fahrzeugen pro Jahr in den verkehrsreichsten Städten der Welt etwa 250 Millionen Stunden (28.500 Jahre) aktiver Fahrzeit eingespart werden können. Diese Zeit kann dann für Produkte und Dienstleistungen, wie etwa Wellness oder „Drive–with“– statt „Drive–in“–Angebote, verwendet werden [8].

Die Politik und Städteverwaltungen erwarten von der Nutzung autonomer Fahrzeuge bessere Verkehrssysteme. So können schlecht angebundene Gebiete in der Zukunft flächendeckend erreicht werden. Auch der Straßenraum kann in der Zukunft anders genutzt werden, da kaum öffentliche Parkplätze benötigt werden [10].

Die Möglichkeiten, die durch autonome Fahrzeuge erreicht werden können, bringen jedoch auch große Herausforderungen mit sich. Es wird davon ausgegangen, dass das autonome Fahren den städtischen Verkehr auf unterschiedliche Weisen prägen wird. Als einer der Gründe wird beispielsweise die stärkere Belastung der Infrastruktur genannt. Die Städte können dabei die Akzeptanz von autonomem Fahren sowie die Gestaltung der Straßen maßgeblich mitgestalten, da sie im städtischen Bereich die dafür notwendigen Rahmenbedingung schaffen [10]. Des Weiteren zeigen die letzten Presseberichte, dass die autonomen Fahrfunktionen heutzutage noch nicht das einhalten, was sie versprechen. Dies bestätigen zahlreiche Berichte über Unfälle mit Beteiligung autonomer

Fahrzeuge [11, 12]. Als einer der Gründe für derartige Unfälle werden noch nicht ausgereifte Technologien genannt [13].

1.1 Hintergrund und Motivation

Obwohl die Idee autonomer Fahrzeuge nicht neu ist und erste Berichte bereits aus den 1920er Jahren stammen [14, Seite 595], kann diese Idee erst circa 100 Jahre später zur Realität werden. Dies ist vor allem mit dem technologischen Fortschritt hinsichtlich der Rechenleistung von Prozessoren, die das mooresche Gesetz [15] impliziert, der Fortschritte in der Entwicklung diverser umfelderfassender Sensoren [16, Teil IV] sowie leistungsstarker Algorithmen zu erklären.

Im Bereich des Individualverkehrs wird autonomes Fahren unter dem Oberbegriff Fahrerassistenzsystem (FAS) eingeordnet und von der Bundesanstalt für Straßenwesen (BASt) in fünf Automatisierungsstufen eingeteilt [17]. Mit zunehmender Autonomie steigt auch die Komplexität solcher Systeme. Die Realisierung der Visionen, die einleitend erwähnt wurden, wird nur mit autonomen Systemen der Stufe fünf möglich sein. Da die Komplexität solcher Systeme deutlich höher ist als bei herkömmlichen FAS, sind diese auf zahlreiche umfelderfassende Sensoren angewiesen. Es werden nicht nur quantitativ mehr Sensoren verbaut, sondern auch weitere physikalische Prinzipien verwendet, wie zum Beispiel Radio Detection and Ranging (Radar), Lidar, Kamera, Ultraschall oder Global Navigation Satellite System (GNSS) [18]. Mit der Komplexität eines Systems steigt entsprechend der Entwicklungs– und Testaufwand. In [19] zeigen Wachenfeld und Winner, dass zum Testen eines Autobahn–Piloten, der laut der BASt in die Autonomiestufe vier einzuordnen ist [20], etwa $6{,}6 \cdot 10^9$ Fahrkilometer benötigt werden, um die Sicherheitskriterien entsprechend der Norm ISO 26262 S3 [21] der International Organization for Standardization (ISO) zu erfüllen. Solche Umfänge auf realen Straßen beziehungsweise auf Testfeldern zurückzulegen, ist aus wirtschaftlicher Sicht nicht zielführend, weshalb Alternativen zu realen Tests notwendig sind.

Eine der Möglichkeiten diese Herausforderung zu meistern, ist die Nutzung von Simulationen. Solche Ansätze werden in der Automobilbranche seit Jahrzehnten praktiziert. Abhängig von einem Anwendungsfall und der Entwicklungsphase können Model in the Loop (MIL), Software in the Loop (SIL), Hardware in the Loop (HIL) und weitere Ansätze angewendet werden [22]. Die drei Herangehensweisen werden während der Entwicklung in aufeinander aufbauenden Phasen entsprechend des im Automobilbereich weitverbreiteten V–Modells durchgeführt [22]. Wie im vorherigen Absatz erwähnt, werden autonome Fahrzeuge deutlich stärker auf die umfelderfassende Sensorik angewiesen sein. Heutzutage werden jedoch die Sensortechnologien erst in einer relativ späten Phase der Entwicklung auf HIL–Prüfständen getestet. Dies liegt zum größten Teil daran, dass für die entsprechenden Sensoren in den früheren Entwicklungsphasen keine oder nur rudimentäre Sensormodelle vorliegen [23]. Um also die Testzeiten zu verkürzen, sollen so viele Absicherungsumfänge wie möglich bereits mit MIL– beziehungsweise SIL–Ansätzen durchgeführt werden.

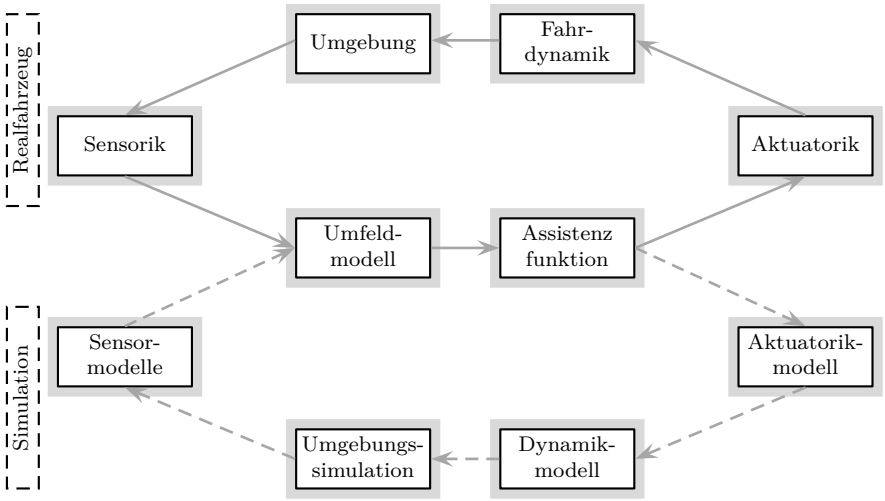


Abbildung 1.1: Schematische Darstellung realer und simulierter Regelkreise moderner Assistenzsysteme.

Einen weiteren Anwendungsfall, neben der Absicherung, stellt die entwicklungsbegleitende Simulation dar. In Abbildung 1.1 sind reale und simulierte Regelkreise einer Assistenzfunktion dargestellt. Anhand dieser Abbildung ist zu erkennen, dass das Umfeldmodell und somit auch die Assistenzfunktion auf Sensordaten angewiesen sind. Das Umfeldmodell wird in dieser Arbeit stellvertretend als Oberbegriff für alle umfelderfassenden Algorithmen, wie beispielsweise Objektfusion, verwendet. Um in einer frühen Phase die Entwicklung dieser Komponenten zu ermöglichen, werden zusätzlich zu der Umgebungssimulation Sensormodelle benötigt.

Die Anwendungsfälle aus den letzten beiden Absätzen zeigen, dass für die umfelderfassenden Sensoren virtuelle Sensormodelle entwickelt werden müssen, die die Eigenschaften realer Sensoren möglichst genau abbilden. Obwohl virtuelle Sensormodelle für die umfelderfassende Sensorik in der Industrie heutzutage kaum Einsatz finden, wurden forschungsseitig dazu bereits zahlreiche Veröffentlichungen gemacht [24–35]. Es gibt bis heute jedoch kaum Publikationen dazu, ob diese Ansätze dabei helfen werden, die Testumfänge zu reduzieren und die Entwicklungszyklen zu beschleunigen, denn die Sensormodelle müssen vor der Nutzung erst validiert werden [19]. Die vorliegende Arbeit legt deshalb den Schwerpunkt auf die Untersuchung neuer Ansätze für die Validierung umfelderfassender Sensormodelle mit dem Fokus auf virtuelles Testen und entwicklungsbegleitende Simulation.

1.2 Ziele und Aufbau der Arbeit

Basierend auf den Erkenntnissen aus dem vorherigen Kapitel, werden in dieser Arbeit vier Aspekte zur Validierung umfelderfassender Sensormodelle behandelt. Diese werden als Forschungsfragen formuliert und bilden gleichzeitig die Struktur der Arbeit:

Forschungsfrage 1 Was ist bei der Validierung umfelderfassender Sensormodelle im Automobilbereich zu beachten und wie können diese systematisch analysiert und bewertet werden?

Das Themengebiet der Sensormodellvalidierung im automobilen Umfeld ist bis jetzt wenig untersucht worden. Im Rahmen dieser Frage gilt es zu recherchieren, welche Modellierungskonzepte für die umfelderfassende Sensoren von Relevanz und welche Validierungsansätze zu ähnlichen Fragestellungen aus anderen Disziplinen bekannt sind. Dazu werden in Kapitel 2 Grundlagen zu Sensorsimulation und Validierungskonzepten vermittelt und darauf basierend in Kapitel 3.1 eine Methode zur Validierung umfelderfassender Sensormodelle abgeleitet.

Forschungsfrage 2 Wie lässt sich die Anzahl der Eingangsdaten von datengetriebenen Sensormodellen quantifizieren und wie können diese reduziert werden?

Eine der Erkenntnisse aus Kapitel 2 wird sein, dass es zahlreiche Ansätze zur Modellierung von Sensoren gibt, die auf der Erhebung realer Daten beruhen. Folglich muss diese Art von Modellen vor der Nutzung mit Eingangsdaten gespeist werden. In Kapitel 3 wird gezeigt, dass die Sammlung von realen Daten sehr zeitaufwändig sein kann und es aus wirtschaftlicher Sicht nicht möglich ist, Messdaten für alle vorstellbaren Szenarien zu erfassen. In Kapitel 3.4 wird deshalb ein Ansatz zur Reduktion der Eingangsparameter erarbeitet und in Kapitel 3.5 eine Methode zur systematischen Erfassung der Daten abgeleitet. Die beiden genannten Ansätze haben das Ziel die Anzahl der zu erfassenden Daten, die als Eingang für die datengetriebenen Sensormodelle dienen, zu reduzieren ohne dabei die Modellgüte merklich zu beeinflussen.

Basierend auf den Ergebnissen aus Kapitel 2 können die Modellierungsansätze in zwei übergreifende Kategorien eingeteilt werden: Phänomenologische und physikalische. Diese werden im Rahmen der folgenden beiden Forschungsfragen untersucht.

Forschungsfrage 3 Wie können phänomenologische beziehungsweise statistische umfelderfassende Sensormodelle objektiv bewertet werden?

Unter phänomenologischen beziehungsweise statistischen Modellen sind Ansätze zu verstehen, die mit Hilfe simulierter Daten die statistischen Eigenschaften realer Sensoren abbilden. Im Fall von umfelderfassenden Sensormodellen im Automobilbereich kann so beispielsweise die Abstandsmessung zwischen zwei Fahrzeugen simuliert werden. Dabei wird die Fehlerverteilung realer Daten auf die perfekten simulierten Daten aufaddiert (siehe Kapitel 2.1). Da im Automobilbereich die Repräsentation der Umwelt oft mittels sogenannter Objektlisten realisiert wird, in denen ähnliche Größen, wie die zuvor beschriebene Abstandsmessung, dargestellt werden, stellt diese Modellierungskategorie einen wichtigen Bestandteil dar. In Kapitel 4 wird deshalb untersucht, wie die statistischen Sensormodelle objektiv bewertet werden können, bevor diese in der Simulation verwendet werden.

Forschungsfrage 4 Wie können physikalische umfelderfassende Sensormodelle objektiv bewertet werden?

Eine weitere Kategorie, neben den phänomenologischen Sensormodellen, ist die physikalische Modellierung. Bei dieser Modellierungsart werden die physikalischen Wirkprinzipien der Sensoren, wie zum Beispiel die Ausbreitung und Reflexion elektromagnetischer Wellen, nachmodelliert. So lassen sich die ungefilterten Messungen der Sensoren bereits in einer frühen Phase der Entwicklung realistischer simulieren. In Kapitel 5 wird deshalb untersucht, wie die physikalischen Sensormodelle objektiv bewertet werden können.

Kapitel 6 schließt die Arbeit mit einer Diskussion über die gewonnen Erkenntnisse und abschließenden Empfehlungen hinsichtlich Bedatung und Validierung von umfelderfassenden Sensormodellen ab.

2 Grundlagen der Sensormodellvalidierung

In diesem Kapitel werden die Grundlagen zu den Themenkomplexen Sensormodellierung und Validierung vermittelt. Diese werden benötigt, um darauf basierend in Kapitel 3 eine Methode zur Validierung umfelderfassender Sensormodelle abzuleiten. Zunächst werden die Aufgaben eines Sensormodells formalisiert und, basierend auf dem Stand der Technik, die aktuellen Entwicklungen im Bereich der Sensormodellierung vorgestellt. Im Anschluss daran werden die Aufgaben der Validierung formalisiert und abschließend die für diese Arbeit relevanten Anforderungen an die Sensormodellvalidierung formuliert.

2.1 Methoden der Sensorsimulation

In diesem Unterkapitel wird zuerst auf die grundlegende Idee der Sensormodellierung eingegangen. Darauf stützend werden für diese Arbeit relevanten Datenformate und aktuelle Entwicklungen aus dem Bereich der Sensormodellierung vorgestellt.

Die grundlegende Aufgabe eines realen umfelderfassenden Sensors ist, dessen Umgebung Ω_r zu erfassen, wobei es sich im Kontext von FAS in der Regel um Verkehrssituationen handelt. Das Ergebnis der Messung ζ_r wird einem Umfeldmodell zur Verfügung gestellt (siehe Abbildung 1.1). Subskript r bezieht sich dabei auf reale Daten, simulierte Größen werden dagegen mit Subskript s gekennzeichnet. Die Umgebungsrepräsentation Ω_r kann beispielsweise folgende Größen enthalten: Position, Geschwindigkeit und Ausdehnung von Objekten, Informationen zu Materialien oder Geometrie, Wetterbedingungen oder Tageszeit, Verkehrszeichen oder deren logische Straßenverläufe [36].

Die Aufgabe eines Sensormodells ist, das Übertragungsverhalten eines realen Sensors realitätsnah nachzubilden. Eine schematische Darstellung dieses Verhaltens wird in Abbildung 2.1 gezeigt. Die virtuelle Umgebung wird meist von einer Simulation in Form einer hochdimensionalen Darstellung Ω_s zur Verfügung gestellt. Eingangsseitig im Modell wird diese zunächst auf einen, abhängig vom Anwendungsfall, relevanten Ausschnitt

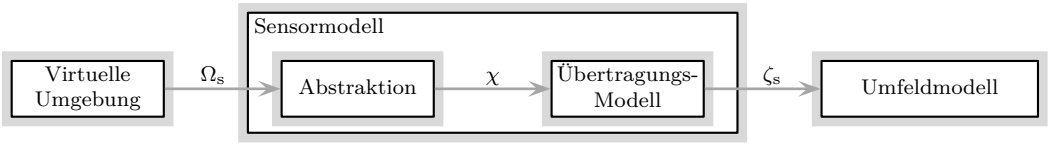


Abbildung 2.1: Grundaufbau eines umfelderfassenden Sensormodells nach [36].

χ reduziert. Dieser Schritt ist verlustbehaftet, weshalb er als kritisch anzusehen ist, da er die Modellgüte negativ beeinflussen kann [36]. Andererseits wird mit zunehmender Anzahl von Parametern in χ die Modellbildung erschwert, da die Mustererkennung in hochdimensionalen Daten aufgrund des Fluchs der Dimensionalität deutlich schwieriger ist [37], [38, Kapitel 1.4]. Dies liegt daran, dass mit jeder zunehmenden Dimension exponentiell mehr Informationen und damit auch Messdaten notwendig sind, um ein Modell vollständig zu beschreiben.

Die genaue Definition der simulierten Umgebung Ω_s, der Abstraktion χ und des Modellergebnisses ζ_s hängt vom Anwendungsfall des zu modellierenden Sensors und des Modellierungsansatzes ab. Im Laufe der Arbeit werden diese Größen an relevanten Stellen konkretisiert.

2.1.1 Verwendete Datenformate

In diesem Abschnitt sollen die verwendeten Ausgangsdatenformate der realen Sensoren ζ_r genauer spezifiziert werden. Da entsprechend Abbildung 1.1 das Umfeldmodell sowohl reale als auch simulierte Daten interpretieren soll, wird an ein Sensormodell die Anforderung gestellt, dass die Schnittstellendefinition dieselbe wie bei einem realen Sensor sein muss. Die nachfolgende Beschreibung gilt also sowohl für ζ_r als auch für ζ_s.

Um die Ausgangsdaten des Sensormodells eindeutig beschreiben zu können, ist die Festlegung von Koordinatensystemen erforderlich. Die drei für diese Arbeit relevanten Koordinatensysteme sind in Abbildung 2.2 dargestellt. Unterschieden wird zwischen Welt– (x^W,y^W,z^W), Fahrzeug– (x^F,y^F,z^F) und Sensorkoordinaten (x^S,y^S,z^S). Die Weltkoordinaten beziehen sich auf einen Punkt in der realen oder virtuellen Welt. In Realität kann der Ursprung sich beispielsweise auf den Startpunkt einer Fahrt beziehen. In der Simulation hingegen wird meist der Nullpunkt der verwendeten Karte als Ursprung genommen. Die Fahrzeugkoordinaten beziehen sich auf die Hinterachse des Fahrzeuges, wobei die Höhe vernachlässigt und die z–Achse auf die Fahrbahn projiziert wird. Die Sensorkoordinaten beziehen sich auf die Verbauposition des jeweiligen Sensors.

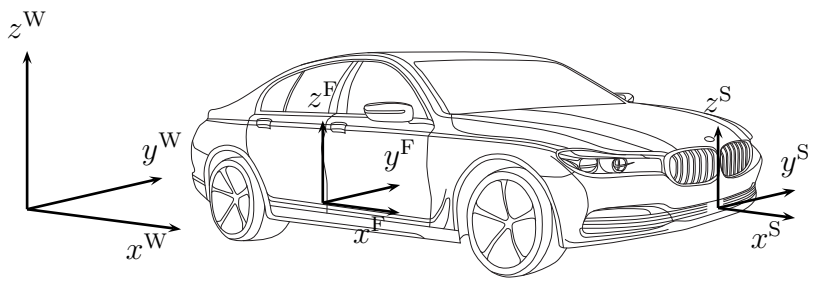


Abbildung 2.2: Verwendete Koordinatensysteme: Weltkoordinaten gekennzeichnet mit Superskript W, Fahrzeugkoordinaten mit F und Sensorkoordinaten mit S. Die Weltkoordinaten sind global und beziehen sich beispielsweise auf den Fahrtbeginn. Fahrzeugkoordinaten beziehen sich auf die Mitte der Fahrzeughinterachse mit Projektion auf die Fahrbahn. Sensorkoordinaten beziehen sich auf die Position des Sensors.

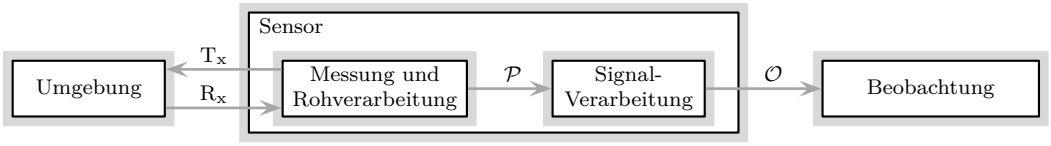


Abbildung 2.3: Überblick des Erfassungsvorgangs eines umfelderfassenden Sensors nach [25].

Der herkömmliche Vorgang einer Messdatenerfassung umfelderfassender Sensoren besteht aus folgenden Schritten: (a) Messung, (b) Rohdatenverarbeitung und (c) Signalverarbeitung. Diese Schritte werden in Abbildung 2.3 visualisiert. Als Erstes tastet ein Sensor seine Umgebung ab T_x und erhält eine Antwort R_x zurück. Diese Daten werden anschließend verarbeitet und in Form von ungefilterten Messungen $\mathcal{P}$ ausgegeben. Es könnte beispielsweise die Punktewolke eines Lidars beziehungsweise Reflexionsliste eines Radars sein. Die ungefilterten Messungen werden anschließend einer Signalverarbeitung unterzogen und liefern die Sensorbeobachtungen in Form einer Objektliste $\mathcal{O}$ zurück. Entsprechend Abbildung 2.3 werden in dieser Arbeit die zwei erwähnten Ausgangsschnittstellen, ungefilterte Messungen und Objektlisten, verwendet und im Folgenden erläutert. Die hier verwendete Beschreibung ist an [36] angelehnt.

Objektlisten ist ein Sensorausgangsformat zur Beschreibung von Fahrsituationen, das in den heutigen umfelderfassenden Sensoren, wie beispielsweise in einem Lidar oder Radar, verwendet wird. Eine Objektliste stellt die Eigenschaften von detektierten Objekten mit wenigen Parametern dar.

Eine Objektliste $\mathcal{O}$ besteht aus einem Zeitstempel t, der die aktuelle Zeit angibt, sowie einer Anzahl N_o von Objekten o_i,

$$\mathcal{O} = \{t, o_i\}_{i=1:N_o} \tag{2.1}$$

wobei o_i die Objektbeschreibung des i–ten Objekts zum Zeitpunkt t ist. Unter Objekten werden dabei nicht nur Fahrzeuge, sondern beliebige Gegenstände, die von dem jeweiligen Sensor erkannt werden, verstanden. Es können unter anderem Fußgänger, Fahrräder, Motorräder, Straßenschilder, Leitplanken und sonstige Ziele sein.

Eine Objektbeschreibung o_i ist eine Datenstruktur, die Informationen zu einem detektierten Objekt enthält. Im Rahmen dieser Arbeit gehören dazu: Identifikationsnummer id_i, Abstand in longitudinaler $d_{x,i}$ und lateraler Richtung $d_{y,i}$, deren geschätzte Standardabweichungen $e_{x,i}$ und $e_{y,i}$, absolute Geschwindigkeit in longitudinaler $v_{x,i}$ und lateraler Richtung $v_{y,i}$, Länge l_i und Breite b_i des Objektes und der Gierwinkel γ_i. Die Identifikationsnummer id_i wird benötigt, um die Objekte über mehrere Zeitschritte verfolgen zu können. Die Objektbeschreibung wird wie folgt dargestellt,

$$o_i = \left\{id_i,\ d_{x,i},\ d_{y,i},\ \sigma_{x,i},\ \sigma_{y,i},\ v_{x,i},\ v_{y,i},\ l_i,\ b_i,\ \gamma_i\right\}. \tag{2.2}$$

Manche Sensoren schätzen zusätzlich den Objekttyp und können die erkannten Objekte als statisch oder dynamisch klassifizieren. Die Beschreibung eines Objekts ist in Abbildung 2.4a schematisch in der Draufsicht visualisiert. Der dabei verwendete Ursprung des

Koordinatensystems ist der Sensor selbst. Um mehrere Sensoren fusionieren zu können, werden diese Informationen meist in das Fahrzeugkoordinatensystem transformiert.

Ungefilterte Messungen ist ein weiteres Datenformat, das aufgrund vielversprechender Ergebnisse in der Fusion immer relevanter wird [39,40]. Die ungefilterten Messungen werden in dieser Arbeit teilweise kurz Rohdaten genannt. Dabei bedeutet es nicht, dass diese Daten keiner Vorverarbeitung unterzogen werden, sondern dass diese im Gegensatz zu den Objektlisten nicht durch Filteralgorithmen, wie beispielsweise dem Kalman–Filter, geglättet werden. Solche Daten können beispielsweise eine Reflexionsliste eines Radars sein, wobei die Reflexionen die Streuzentren mit der lokal höchsten Intensität darstellen [31]. Eine weitere Alternative ist die Punktewolke eines Lidar–Sensors. Diese enthält Informationen zu den Auftreffpunkten der ausgesendeten Laserimpulse. Reflexionslisten und Punktewolken sind zwei ineinander überführbare Formate, wobei meist die Reflexionslisten in zylindrischen und Punktewolken in kartesischen Koordinaten dargestellt werden. Im Rahmen dieser Arbeit wird die Darstellung in kartesischen Koordinaten gewählt. In Abbildung 2.4b ist die Draufsicht einer Punktewolke schematisch dargestellt. Die Auftreffpunkte der Laserimpulse sind mit grauen Punkten markiert. Eine solche Punktewolke kann wie folgt beschrieben werden,

$$\mathcal{P}_{s,n}(x,y,z) = \{d_{\mathrm{x},s,n},\ d_{\mathrm{y},s,n},\ d_{\mathrm{z},s,n}\}_{s=1:S_{\mathcal{P}};n=1:N_{\mathcal{P}}} \tag{2.3}$$

wobei d der Abstand des Auftreffpunkts vom Sensor in x–, y– beziehungsweise z–Richtung ist. Subskript s beschreibt, aus welcher Abtastebene des Laserscanners der

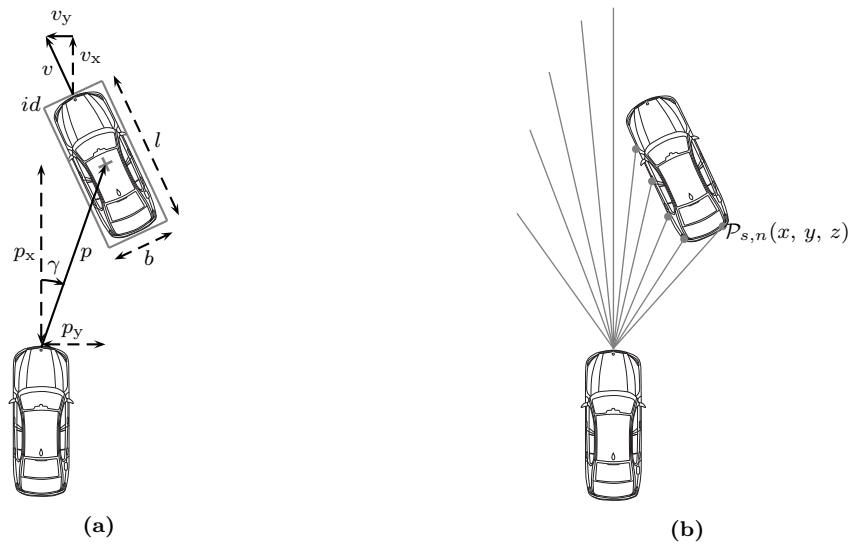


Abbildung 2.4: Visualisierung der verwendeten Datenformate: (a) Objektliste und (b) ungefilterte Messungen in Form einer Punktewolke $\mathcal{P}$. In der Objektliste werden kompakt die Eigenschaften von erkannten Objekten repräsentiert. Eine Punktewolke $\mathcal{P}$ enthält Informationen zu den Auftreffpunkten der ausgesendeten Laserimpulse in kartesischen Koordinaten, wobei Index s die Information zu der Lidar–Abtastebene, aus der der Puls ausgesendet wird, enthält und n die Position des jeweiligen Punktes in der Liste angibt.

jeweilige Impuls gesendet wird und n gibt an, auf welcher Position in s sich dieser befindet. In Abbildung 2.4b sind Strahlen einer Abtastebene schematisch dargestellt. Bei einem realen Lidar–Sensor liegen die Abtastebenen vertikal übereinander.

Referenz ist eine weitere Kategorie an Messungen, die neben den Aufnahmen der umfelderfassenden Sensoren benötigt werden, um die Güte der Sensormodelle zu bewerten. Diese basieren auf GNSS–Technologie mit Differentialkorrektur und Inertialsensorik. Während der Datenaufnahme wird neben den umfelderfassenden Sensoren ein Referenzsystem, sowohl im Eigen– als auch im Zielfahrzeug, verbaut. Zusätzlich zu den Referenzmodulen wird das Eigenfahrzeug mit einem Fusionsmodul ausgestattet, das aus der Differenz der Koordinaten von Eigen– und Zielfahrzeug die Relativabstände zwischen den Fahrzeugen bestimmt. Die Datenübertragung zwischen den Versuchsfahrzeugen erfolgt dabei über drahtloses lokales Netzwerk (WLAN). Die Datenstruktur der Referenzsensorik ist ähnlich zur Objektliste und besitzt den selben Informationsgehalt.

2.1.2 Ansätze der Sensormodellierung

In diesem Abschnitt wird ein Überblick über den aktuellen Stand der Ansätze zur Modellierung von umfelderfassender Sensorik gegeben. In [36] klassifiziert Hirsenkorn diese in drei Kategorien: (a) deterministische Modelle, (b) statistische Modelle sowie (c) Modelle der Feldausbreitung. Zunächst wird in Tabelle 2.1 ein kurzer Überblick mit Vor– und Nachteilen der einzelnen Modellierungskategorien gegeben. Basierend darauf werden im Anschluss die für diese Arbeit relevanten Modellierungsansätze ausgewählt

Tabelle 2.1: Zusammenfassung der Vor– und Nachteile von deterministischen, statistischen und feldausbreitenden Modellierungsansätzen nach [36].

	Vorteile	Nachteile
Deterministisch	+ Reproduzierbarkeit direkt gegeben + Ideales Modell möglich mit genügend Trainingsdaten	– Zufälligkeit realer Messungen nicht abbildbar – Referenz für viele Eingangs–größen notwendig
Statistisch: Parametrisch	+ Anzahl notwendiger Trainings–daten nicht kritisch + Standardisierung leicht umsetzbar	– Hoher Realismus kaum möglich – Vereinfachte Beschreibung der Umgebung
Statistisch: Nicht–Parametrisch	+ Hohe Realismusgrade einzelner Parameter erreichbar + Standardisierung umsetzbar	– Anzahl notwendiger Trainings–daten eher hoch – Vereinfachte Beschreibung der Umgebung
Feldausbreitung: Finite–Elemente und ähnliche	+ Physikalische Eigenschaften gut abbildbar	– Hohe Anforderungen an Speicher und Rechenleistung
Feldausbreitung: Strahlenoptisch	+ Abbildung physikalischer Eigenschaften in Realzeit möglich	– Hohe Anforderungen an die Modellierung der Umgebung

und detailliert beschrieben.

Deterministische Modelle bilden das Verhalten eines Sensors mit Hilfe von Funktionsapproximatoren implizit ab. Dabei wird aus einer Vielzahl von Parametern χ, die Umgebung beschreiben, das Verhalten einer zu modellierenden realen Größe ζ_r gelernt und in der Simulation mit ζ_s abgebildet (siehe Abbildung 2.1). Die Abbildungsfunktion wird dabei aus realen Messaufzeichnungen approximiert. Mit diesem Ansatz kann mit genügend Messdaten das Sensorverhalten ideal abgebildet werden. Eine weitere wichtige Eigenschaft deterministischer Modelle ist, dass sich bei Wiederholung eines Experiments der Ausgang des Modells ζ_s nicht ändert, so lange auch die Werte der Eingangsparameter χ gleich bleiben. Daraus resultiert der Vorteil, dass die Reproduzierbarkeit direkt gegeben ist. Nachteilig ist jedoch, dass dies nicht dem Verhalten realer Sensoren entspricht und die statistischen Eigenschaften nicht abbildbar sind. Dies ist auch der Grund, weshalb die deterministischen Modelle in dieser Arbeit nicht weiter berücksichtigt werden.

Statistische Modelle beschreiben das Sensorverhalten mittels Verteilungsfunktionen. Dabei wird für die zu modellierende Größe ζ_r erst eine Verteilungsfunktion ermittelt und zur Laufzeit der Simulation daraus Realisierungen gezogen, sodass ζ_s der realen Verteilung folgt. Dabei können die Fehlerverteilungen mit parametrischen und nicht–parametrischen Verfahren geschätzt werden. Wie sich die einzelnen Ansätze unterscheiden, ist in Abbildung 2.5 verdeutlicht.

Bei den parametrischen Modellen werden die Realisierungen aus einer parametrischen Verteilung, wie beispielsweise der Normalverteilung, gezogen. Solche Modelle finden zahlreiche Anwendungen im Bereich der Sensormodellierung [25, 31–33, 41–45]. Diese sind mit wenigen Parametern beschreibbar und können daher einfach standardisiert werden. Nachteil der parametrischen Modelle ist, dass sie reale Verteilungen nur bedingt

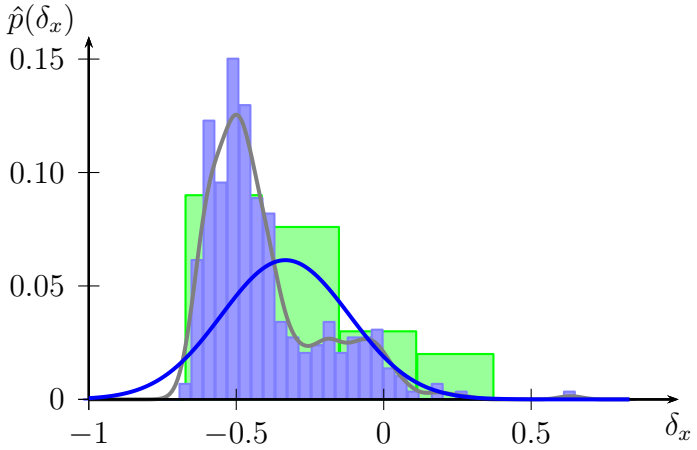


Abbildung 2.5: Vergleich der Ergebnisse verschiedener Schätzer einer Stichprobe nach [36]. Grün: Histogramm mit Klassenbreite $h = 5$. Violett: Histogramm mit $h = 1$. Grau: Kernedichteschätzer mit normal verteiltem Kernel und Bandbreite $b = 1$. Dunkelblau: Normalverteilung.

abbilden können. Es wird in Abbildung 2.5 mit der Visualisierung der Normalverteilung in dunkel blau verdeutlicht.

Die nicht–parametrischen Ansätze stellen eine weitere Möglichkeit zur Schätzung von Verteilungen. Bei solchen Verfahren können Fehlerverteilungen beliebiger Form abgebildet werden. Die einfachste Realisierung der nicht–parametrischen Verteilungsschätzung ist ein Histogramm. In Abbildung 2.5 sind in violett und grün zwei Beispiele dafür gezeigt. Leicht zu erkennen ist, dass sich die Ergebnisse abhängig von der Wahl der Klassenbreite sehr unterscheiden. Bei Histogrammen sind daher die Kenntnisse über die zu schätzenden Datensätze von Vorteil und benötigen Feinjustage, um optimale Ergebnisse der Schätzung zu erzielen. Dies kann beispielsweise mit Hilfe von neuronalen Netzen erreicht werden, wie in [34] gezeigt wird. Hirsenkorn stellt in [27] ein weiteres nicht–parametrisches Verfahren vor, das auf Kerndichteschätzung basiert. Dieses ist im Vergleich zu der Schätzung von Histogrammen mit einem neuronalen Netz generisch und erreicht hohe Realismusgrade [36]. Dieses Verhalten wird anhand der grauen Verteilung in Abbildung 2.5 verdeutlicht.

Im Rahmen dieser Arbeit werden Repräsentanten beider Modellierungskategorien, parametrische und nicht–parametrische, eingesetzt. Eine detaillierte Beschreibung der verwendeten Modelle findet einleitend in Kapitel 4 statt.

Modellierung der Feldausbreitung ermöglicht die Nachbildung des physikalischen Sensorverhaltens. Das Messprinzip der umfelderfassenden Sensoren, wie Radar und Lidar, beruht auf der Ausbreitung elektromagnetischer Wellen im Raum. Dieses Verhalten kann mit den Maxwell–Gleichungen detailliert beschrieben werden. Analytisch sind diese jedoch nur für einfache Geometrien lösbar. Für komplexere Strukturen werden numerische Lösungen benötigt. Dafür eignet sich beispielsweise die Finite–Elemente–Methode [46]. Aufgrund der hohen Anforderungen an die Rechenleistung und Speicher ist damit jedoch aktuell kein Echtzeitverhalten erreichbar. Aus diesem Grund wird diese Methode in der vorliegenden Arbeit nicht berücksichtigt, insbesondere weil zur Nachbildung realer Sensoren Aktualisierungsraten von bis etwa 40 Hz erreicht werden müssen.

Strahlenoptische Verfahren hingegen nutzen Gesetze der geometrischen Optik, um die Maxwell–Gleichungen für hohe Frequenzen anzunähern [47]. In der strahlenoptischen Modellierung sind folgende drei Ansätze am meisten verbreitet: Gerichtete Strahlverfolgung (englisch: ray tracing), ungerichtete Strahlverfolgung (englisch: ray launching) und Strahlenfang (englisch: ray casting). Die Prinzipien dieser Ansätze sind an einem Szenario in Abbildung 2.6 visualisiert.

Bei der gerichteten Strahlverfolgung in Abbildung 2.6a werden die Strahlen von der Sendeantenne ausgesendet und rekursiv reflektiert, bis diese die maximale Rekursionstiefe erreicht haben oder die Empfangsantenne (hier gleich der Sendeantenne) exakt treffen. Dabei können unterschiedliche Geometrien oder Materialien bei der Modellierung berücksichtigt werden, sodass spiegelnde und diffuse Reflexionen abgebildet werden können. Die zwei Reflexionsarten sind in Abbildung 2.6a zu erkennen. Die erste Reflexion ist dabei spiegelnd und die zweite diffus. Dieses Verfahren wird meist zur Be-

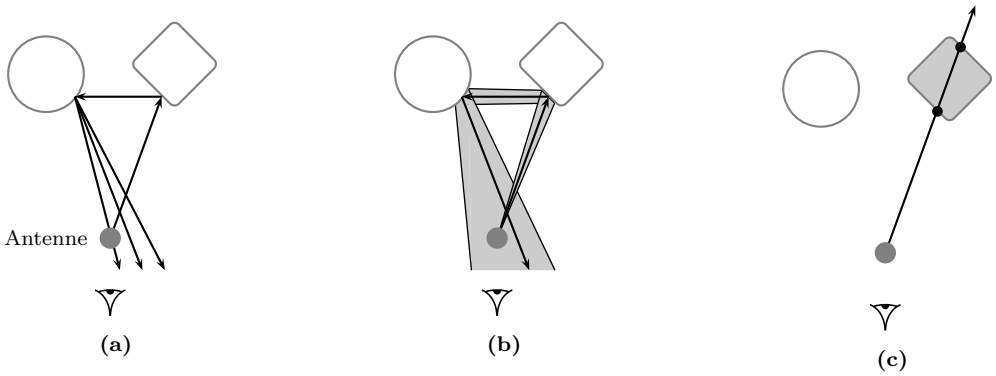


Abbildung 2.6: Visualisierung der strahlenoptischen Modellierungsverfahren: (a) gerichtete Strahlverfolgung (b) ungerichtete Strahlverfolgung (c) Strahlenfang.

rechnung fotorealistischer Bilder verwendet. Beispiele der gerichteten Strahlverfolgung in Verkehrsszenarien werden in [48, 49] gezeigt.

Bei der ungerichteten Strahlverfolgung aus Abbildung 2.6b werden die Strahlen als Volumenkegel interpretiert und gelten als empfangen, sobald diese nah genug an der Empfangsantenne vorbei laufen, beziehungsweise sich diese innerhalb des Volumenkegels des reflektierten Strahls befindet. Ungerichtete Strahlverfolgung ist bei komplexen Szenarien aus theoretischen und technischen Gründen effizienter als gerichtete [36]. Ein Beispiel der Modellierung eines Zwischenfrequenzsignals eines Radars mittels ungerichteter Strahlverfolgung ist in [29] vorgestellt.

Bei der Strahlenfangmethode aus Abbildung 2.6c werden die Strahlen nicht an den Objektoberflächen reflektiert, sondern durchdringen das jeweilige Objekt. Solche Verfahren werden meist für Modellierung von Festkörpern eingesetzt [50] und finden oft im medizinischen Bereich ihre Anwendung [51]. Bei umfelderfassenden Sensoren mit hoher Winkelselektivität hat der Beitrag von Strahlung, die durch Reflexionen höherer Ordnung nach dem ersten Treffer zurück zum Sensor gestreut wird, nur einen marginalen Einfluss auf die resultierenden Messergebnisse. Des Weiteren kann mit der Strahlenfangmethode der Effekt von multiplen Echos eines realen Sensors nachmodelliert werden. In der Praxis treten multiple Echos beispielsweise auf, wenn ein Fahrzeug hinter einer Glasscheibe steht. Der ausgesendete Strahl würde in so einer Situation zwei Echos zurück liefern, eines für die Glasscheibe und ein anderes für das Fahrzeug. Dieses Verfahren kann deshalb für die Modellierung von Punktewolken eines Lidars eingesetzt werden [24].

Im Rahmen dieser Arbeit wird ein physikalisches Modell, das den Strahlenfang–Prinzip folgt, eingesetzt. Eine detaillierte Beschreibung des verwendeten Modells findet einleitend in Kapitel 5 statt.

2.2 Methoden der Validierung

Dieses Unterkapitel behandelt unterschiedliche Aspekte der Modellvalidierungsmethoden. Zuerst wird auf die Problemstellung und Notwendigkeit der Validierung einge-

gangen. Anschließend werden in 2.2.1 grundlegende Begriffe der Modellvalidierung definiert. In 2.2.2 folgt die Beschreibung relevanter Modellvalidierungsmethoden und in 2.2.3 abschließend die Formulierung der Anforderungen an die Validierung von Sensormodellen.

Die Notwendigkeit des virtuellen Testens wurde in Kapitel 1.1 bereits einleitend motiviert. Dieses Kapitel dient daher Konkretisierung der Herausforderungen im Kontext des automatisierten Fahrens. Da die autonome Fahrfunktion als sicherheitsrelevante Software einzustufen ist, kann diese nicht als sicher erachtet werden, so lange sie nicht ISO 26262–konform ist [21]. Koopman identifiziert in [52] fünf Herausforderungen, die für das Testen von autonomen Fahrfunktionen nach dem ISO 26262 V–Modell relevant sind: (a) Fahrer außerhalb der Schleife, (b) komplexe Anforderungen, (c) nicht–deterministische Algorithmen, (d) induktive Lernalgorithmen und (e) ausfallsichere Systeme. Diese werden im Folgenden erläutert.

Zu den offensichtlichsten Herausforderungen bei einem autonomen Fahrzeug zählt die Tatsache, dass der Fahrer eine Passagierrolle einnimmt. Aus der Sicherheitsperspektive kann der Fahrer nicht mehr als eine Rückfallebene in Betracht gezogen werden, da er in einer kritischen Situation nicht mehr die Kontrolle übernehmen kann. Vielmehr muss das Computersystem die Rolle des Ausnahmebehandlers für Fehler, Fehlfunktionen und sonstige Betriebsbedingungen übernehmen. Im Vergleich zu den herkömmlichen Fahrerassistenzsystemen dürfte sich die Komplexität der Ausnahmebehandlung bei autonomen Systemen deshalb deutlich erhöhen [52].

Eine weitere Herausforderung stellen komplexe Anforderungen dar. Ein wesentliches Merkmal des V–Modells ist, dass die rechte Seite des V eine nachvollziehbare Möglichkeit bietet, den Verlauf der linken Seite zu überprüfen. Dies beruht jedoch auf der Annahme, dass die Anforderungen an das System bekannt, richtig, vollständig und eindeutig spezifiziert sind, was im Fall des autonomen Fahrens kaum gegeben sein wird. Der Grund ist die undurchschaubare Anzahl von regulären (Verkehrsszenarien nach der Straßenverkehrsordnung) und unerwünschten (Unwetter, Fehlverhalten anderer Fahrer, beschädigte Schilder und weitere) Ereignissen, die ein autonomer Fuhrpark potenziell sehen kann. Diese sind so vielfältig, dass sie kaum in ein klassisches Lastenheft überführbar sind [52].

Nicht–deterministische und statistische Algorithmen sind eine weitere Herausforderung. Bei autonomen Systemen kommen unter anderem Technologien zum Einsatz, die von Natur aus statistisch sind und daher nicht deterministisch. Die Validierung solcher Systeme stellt insofern eine Herausforderung dar, da diese nicht immer ein eindeutiges Systemverhalten aufweisen und die Testergebnisse daher schwierig zu beurteilen sind. Dies wird aller Voraussicht nach deutlich mehr Tests erfordern, als eine funktionale Validierung, insbesondere wenn das betroffene System sicherheitskritisch ist und eine niedrige Ausfallrate erfordert [52].

Maschinelle Lernsysteme sind eine weitere Technologiekategorie, die in den letzten Jahren an Beliebtheit gewonnen hat [53]. Solche Methoden sind datengetrieben, weshalb die Ansprüche im Gegensatz zum V–Modell nicht in Form eines Satzes funktionaler Anforderungen vorliegen, sondern in Form von Trainingsdaten. Die Validierung ist daher eine offene Frage, die voraussichtlich durch eine Kombination von Datencharakterisie-

rung und Datengenerierung beziehungsweise Datenerfassungsprozessen gelöst werden kann. Eine wesentliche Einschränkung in der Praxis ist jedoch, dass jeder Datenpunkt sehr teuer sein kann. Überwachte Lerntechniken benötigen gelabelte Daten, wobei die Labels meist manuell erstellt werden. Auch unüberwachte Lerntechniken sind möglich, erfordern jedoch ein geschicktes Mapping zur Lösung eines bestimmten Problems. Fraglich dabei ist, dass bei einer Änderung der Lernregeln beziehungsweise Fehler im Lerndatensatz weitere Validierungsdaten gesammelt werden müssen. Des Weiteren sind aufgrund der Komplexität der Anforderungen an ein autonomes System seltene Ereignisse nicht vermeidbar und das Sammeln solcher Daten sehr teuer und schwierig zu skalieren ist [52].

Ausfallsichere Systeme stellen die fünfte Herausforderung im Kontext von autonomen Fahren dar. Wie zuvor angesprochen, fehlt bei den autonomen Systemen die Rückfallebene in Form eines Fahrers. Das bedeutet, dass zumindest ein Teil des Fahrzeugs Fail–Operational und nicht nur Fail–Safe sein muss. Unter Fail–Safe ist dabei ein System zu verstehen, bei dem im Fall von Versagen einer Komponente, diese abgeschaltet wird. Wohingegen bei Fail–Operational–Systemen statt Abschaltung ein Notbetrieb gewährleistet werden muss. Um also ein Fail–Operational–System zu erreichen, sind mindestens drei statt zwei redundante Komponenten erforderlich. Folglich müssen alle Subsysteme ebenfalls entsprechend getestet werden [52].

Zu erkennen ist, dass alle fünf oben aufgeführten Aspekte mit der Validierung und Akkreditierung der autonomen Fahrfunktion zusammenhängen. Das Testen gilt somit als eine der Hauptherausforderungen bei der Entwicklung des autonomen Fahrens. Als mögliche Methoden der Validierung werden beispielsweise Ansätze wie Simulation, formale Beweise oder Fehlerinjektion genannt [52]. Zur Absicherung eines autonomen Systems wird jedoch kaum ein einziger Ansatz ausreichen. Es wird eher eine Kombination unterschiedlicher Methoden notwendig sein. An dieser Stelle seien zwei prominente Beispiele zu erwähnen, die mehrere Ansätze aus der oben genannten Liste nutzen: Waymo und Tesla.

Waymo nennt sein Vorgehen „Safety by Design“, das auf ständiges Lernen des gesamten Systems beruht. Jede Änderung in der Software durchläuft eine Kombination von Tests in der Simulation, auf einem geschlossenem Testgelände und auf der öffentlichen Straße. In der Simulation werden die Änderungen vom Einsatz in der Testflotte geprüft. Dabei werden die herausfordernden Situationen aus der Realität in der Simulation nachgestellt, damit die Software dort üben kann. Im Anschluss wird die neue Software auf einem geschlossenen Gelände durch Testfahrer ausprobiert. Abschließend wird diese sukzessive auf die Flottenfahrzeuge aufgespielt [54]. Waymo legt dabei einen starken Fokus auf die Simulation, wobei bis heute (Mai 2018) circa sechs Milliarden Kilometer virtuell zurückgelegt wurden [55].

Tesla schlägt die sogenannte „Fleet learning“–Teststrategie vor. Das autonome Fahrsystem verfolgt dabei das Verhalten des menschlichen Fahrers, während es inaktiv bleibt und vergleicht das menschliche Verhalten mit dem potentiell eigenen, basierend auf den jeweiligen Sensordaten. Die Zusammenführung dieser Informationen aus allen Fahrzeugen der gesamten Flotte ermöglicht die Identifizierung von Orten und Situationen, in denen das autonome Fahrsystem tendenziell ungenau ist beziehungsweise falsch rea-

giert. Vorteilhaft ist, dass diese Strategie die Erfassung von relevanten Daten vorsieht, auch wenn das System inaktiv ist [56, 57].

Dieser Diskurs zeigt, dass die Sicherheit des autonomen Fahrens in der Gesellschaft intensiv diskutiert wird und viele Herausforderungen für die Validierung und das Testen mit sich bringt. Die beteiligten Meinungsvertreter können dabei grob in zwei Gruppen aufgeteilt werden: Die, die den Fokus auf die technischen Aspekte legen und die, die sich eher mit den gesellschaftlich–politischen Fragestellungen beschäftigen. Wie einleitend in Kapitel 1.2 erwähnt, werden in der vorliegenden Arbeit die technischen Aspekte der Validierung im Kontext des autonomen Fahrens mit dem Fokus auf die Validierung umfelderfassender Sensormodelle behandelt.

2.2.1 Begriffe der Modellvalidierung

Dieser Abschnitt dient der Einführung der grundlegenden Begriffe der Modellvalidierung. In den letzten Jahrzehnten haben es Fortschritte in Rechenleistung, Modellierungsansätzen und numerischen Lösungsalgorithmen ermöglicht, reale Prozesse so weit zu simulieren, dass heutzutage ein Großteil der Experimente ersetzt werden kann. In der Vergangenheit war es notwendig die Systemleistung unter Realbedingungen zu testen. Der aktuelle Ansatz zielt darauf ab, die teuren Realtests durch Simulationsergebnisse zu ersetzen. Dies hat zur Entstehung des Bereichs Verifikation und Validierung (V&V) von Simulation geführt [58]. Es gibt zahlreiche Definitionen dieser Begriffe in der Literatur [59–64], weshalb im Rahmen dieser Arbeit keine neuen vorgeschlagen, sondern die von Oberkampf aus [64] verwendet werden:

Verifizierung ist ein Prozess der Feststellung, dass eine Modellimplementierung die konzeptionelle Beschreibung des Modells und der Lösung des Modells durch den Entwickler genau wiedergibt.

Validierung ist ein Prozess der Bestimmung des Grades, in dem ein Modell eine genaue Darstellung der realen Welt aus der Perspektive der beabsichtigten Verwendung des Modells ist.

Die beiden Prozesse können nach [63] und [65, Kapiteln 2 und 3] entsprechend Abbildung 2.7 in den Modellierungsprozess eingeordnet werden. Diese Beschreibung ist an [66] angelehnt. Es werden drei Typen von Modellen unterschieden: Konzeptionelle, mathematische und computergestützte. Ein **konzeptionelles Modell** spezifiziert dabei: (a) das physikalische System, die Systemumgebung und die interessierenden Phänomene, (b) die Betriebsumgebung des Systems und seinen Anwendungsbereich, (c) die physikalischen Annahmen, die das System und die relevanten Phänomene beschreiben, (d) die interessierenden Ausgangsparameter und (e) die Genauigkeitsanforderungen für die Ausgangsparameter. Das **mathematische Modell** wird aus dem konzeptionellen abgeleitet und umfasst eine Reihe von mathematischen und logischen Beziehungen, die das physikalische System darstellen. Es wird üblicherweise durch eine Reihe von Gleichungen und Rechenvorschriften dargestellt. Ein **computergestütztes Modell**

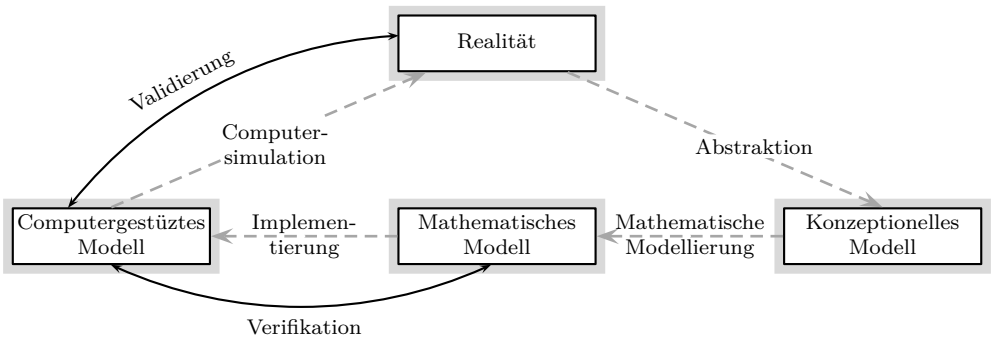


Abbildung 2.7: Modellierungsphasen und die Funktion von V&V in dem Modellierungsprozess nach [65].

ist eine numerische Implementierung des mathematischen Modells, die in Form eines Softwarepakets vorliegt.

Mit Modellvalidierung ist die Validierung des mathematischen Modells gemeint, wobei die Ergebnisse des computergestützten Rechenmodells mit experimentellen Daten verglichen werden. Dabei liegen folgende Annahmen zugrunde: (a) die numerischen Algorithmen sind zuverlässig und genau, (b) das Computerprogramm ist korrekt, (c) es werden keine menschlichen Verfahrensfehler bei der Eingabe oder Ausgabe für die Simulation gemacht und (d) der numerische Lösungsfehler ist vernachlässigbar. Der Nachweis für die Richtigkeit dieser Annahmen muss dabei durch die Verifikation erbracht werden [65].

Des Öfteren kommt es vor, dass nicht immer beide Prozesse (Validierung und Verifizierung) zum Einsatz kommen. Bei der Entwicklung umfangreicher Systeme werden teilweise Modelle eingesetzt, die von Dritten modelliert werden. Diese werden auch „Black–Box“–Modelle genannt, da kein Zugang zum Quellcode gegeben ist [67]. In solchen Situationen sind die Entwickler der Gesamtsysteme daran interessiert, ob die zugrundeliegenden Annahmen solcher Modelle richtig getroffen wurden. Der Fokus liegt dabei auf der Validierung und weniger auf der Verifizierung, da diese vorausgesetzt wird. Eine ähnliche Ausgangslage haben die Automobilhersteller, weil die Aufgabe der Verifikation im Verantwortungsbereich der Lieferanten liegt. Der Schwerpunkt dieser Arbeit liegt deshalb ebenfalls auf der Validierung, weshalb die Sicht der Verifizierung nicht weiter verfolgt wird. Der interessierte Leser, der sich intensiver mit den Fragen der Verifizierung beschäftigen möchte, sei deshalb an dieser Stelle auf die Werke [61, 65, 68] verwiesen.

Die Modellvalidierung kann aus drei Perspektiven betrachtet werden: (a) Entwicklung des mathematischen Modells, (b) Durchführung der Validierungsexperimente und (c) Vergleich der experimentellen Messdaten und der Modellergebnisse [65]. Im Rahmen der Entwicklung des mathematischen Modells werden grundsätzliche Fragen wie die Ziele und die Strategie der Validierung sowie die Möglichkeiten der Bewertung der Modellgüte behandelt. Perspektive (b) beschäftigt sich mit technischen und praktischen Belangen der Durchführung von Validierungsexperimenten und wie sich diese von den traditionellen Experimenten unterscheiden. Beim Vergleich der experimentellen Daten

und der Modellergebnisse (c) wird die Modellgenauigkeit bewertet. Diese Perspektive wird in vielen praktischen Situationen als traditionelle Art der Validierung von Simulationsmodellen gesehen [58, 62, 65, 69], die auch im Fokus dieser Arbeit liegt. Die Perspektiven (a) und (b) werden für (c) zwar vorausgesetzt, werden jedoch in dieser Arbeit nicht weiter vertieft. Der interessierte Leser sei an dieser Stelle auf [65, Kapiteln 10 und 11] verwiesen.

Aufgrund der **Unsicherheiten** in den Messungen und dem Modell ist die Aufgabe der Modellgenauigkeitsbewertung sehr herausfordernd. In den letzten circa 30 Jahren hat die Gemeinschaft der Risikobewertung, vor allem für die Sicherheit von Kernreaktoren, eine praktikable Kategorisierung von Unsicherheiten entwickelt. Es wird zwischen: (a) aleatorischen[1] und (b) epistemischen[2] Unsicherheiten unterschieden. **Aleatorische Unsicherheit** ist begründet durch inhärente Zufälligkeit. Diese wird auch als stochastische Unsicherheit, Variabilität, inhärente Unsicherheit, Zufallsunsicherheit oder Typ–A–Unsicherheit bezeichnet. Die grundlegende Natur der aleatorischen Unsicherheit ist die Zufälligkeit, zum Beispiel aus einem stochastischen Prozess. **Epistemische Unsicherheit** entsteht aufgrund mangelnder Kenntnisse. Diese wird auch als reduzierbare Unsicherheit, Wissensunsicherheit oder subjektive Unsicherheit bezeichnet. Die grundlegende Quelle dieser Unsicherheit ist unvollständiges Wissen jeglicher Art über das zu modellierende System oder seiner Simulation. Bei der epistemischen Unsicherheit wird wiederum zwischen erkannter und blinder Unsicherheit unterschieden (siehe Abbildung 2.8). Bei einer erkannten epistemischen Unsicherheit wird eine bewusste Entscheidung getroffen, diese in irgendeiner Weise bei der Validierung zu behandeln oder sie aus praktischen Gründen zu ignorieren. Bei der blinden epistemischen Unsicherheit wird nicht erkannt, dass das Wissen über das zu modellierende System nicht vollständig ist. Die häufigsten Ursachen dafür sind menschliche Fehler beziehungsweise Fehleinschätzungen. Einige Beispiele sind Programmierfehler in fremder Software oder Fehler in der Aufzeichnung oder Verarbeitung experimenteller Daten, die für die Vali-

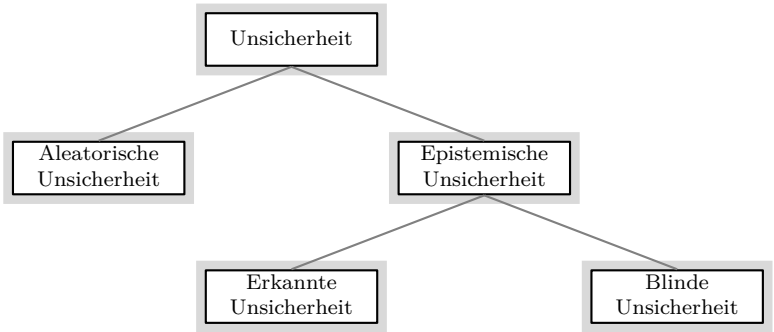


Abbildung 2.8: Klassifizierung von Unsicherheiten nach [65]. Zwei Kategorien von Unsicherheiten werden unterschieden aleatorische[1] und epistemische[2]. Die epistemische wird wiederum in erkannte und blinde Unsicherheiten eingeteilt.

[1] Aleatorisch: Vom Zufall abhängig, auf Zufall beruhend.
[2] Epistemologisch: Wissenschafts–, erkenntnistheoretisch.

dierung verwendet werden. Es gibt keine zuverlässigen Methoden, um das Ausmaß der blinden Unsicherheit und ihre Auswirkung abzuschätzen beziehungsweise zu begrenzen. Um sie jedoch so früh wie möglich zu erkennen, werden diverse Verfahren eingesetzt, die im Grundsatz auf Redundanz basieren. Zum Beispiel redundante Verfahren und Protokolle für Tests oder Analysen [65].

In der Literatur werden die Terme Unsicherheit und **Fehler** teilweise synonym verwendet. Um Fehlinterpretation zu vermeiden, wird in dieser Arbeit unter einem Fehler die Abweichung vom wahren Wert einer Größe verstanden. Die Unsicherheit beschreibt somit die Art einer Quelle und der Fehler eine einzige Realisierung einer unsicheren Einheit [69]. Dabei ist wichtig zwischen den verschiedenen Fehlerquellen zu unterscheiden. Wie bereits erwähnt, handelt es sich bei der Validierung um den Vergleich der experimentellen und simulierten Ergebnisse. Dies kann mathematisch wie folgt ausgedrückt werden,

$$\mathcal{E} = \zeta_\mathrm{s} - \zeta_\mathrm{e} \tag{2.4}$$

wobei $\mathcal{E}$ den Validierungsfehler, ζ_s eine simulierte Größe und ζ_e die experimentellen Validierungsergebnisse bezeichnen. Die Gleichung 2.4 kann wie folgt erweitert werden [65],

$$\mathcal{E} = \underbrace{(\zeta_\mathrm{s} - \zeta_\mathrm{r})}_{\delta_\mathrm{s}} - \underbrace{(\zeta_\mathrm{e} - \zeta_\mathrm{r})}_{\delta_\mathrm{e}} \tag{2.5}$$

wobei ζ_r reale unbekannte Größe darstellt. δ_s und δ_e stehen für den simulierten und experimentellen Fehler und repräsentieren zwei wesentliche Fehlerquellen im Kontext der Validierung. Hier werden aufgrund der zuvor genannten Annahmen bewusst keine Fehlerquellen der Verifizierung berücksichtigt. Diese sind bei Bedarf in [65, Kapitel 10] nachzulesen. δ_e bildet den Messfehler aufgrund der Differenz zwischen dem realen, jedoch unbekanntem Wert ζ_r und der Messung einer physikalischen Größe ζ_e aus den Validierungsexperimenten ab. δ_s stellt den Simulationsfehler dar, der sich aus der Differenz zwischen dem Modell ζ_s und der realen Größe ζ_r ergibt.

Ein weiterer wichtiger Begriff im Kontext der Modellvalidierung ist die **Validierungsmetrik**. Eine Metrik ist ein mathematischer Operator, der ein formales Maß für die Diskrepanz zwischen Experiment– und Modellergebnissen angibt. Eine Metrik ist von zentraler Bedeutung für die Validierung, da diese eine Entscheidungsgrundlage für die Akzeptanz eines Modells bildet. Dabei bedeutet ein niedriger Metrikwert eine gute Übereinstimmung und vice versa. Im Folgenden werden einige wünschenswerte Eigenschaften einer Validierungsmetrik nach [70, 71] vorgestellt. Eine Metrik soll

- intuitiv verständlich,
- auf deterministische und nicht–deterministische Daten anwendbar,
- quantitativ und objektiv sein,
- keine Akzeptanzkriterien beinhalten,
- Unsicherheiten berücksichtigen

– ein Konfidenzinterval in Bezug auf die Anzahl der Messdaten angeben und
– die mathematischen Eigenschaften einer Metrik erfüllen.

Letztere sind wie folgt definiert [65],

$$M(\zeta_\mathrm{e}, \zeta_\mathrm{s}) = M(\zeta_\mathrm{s}, \zeta_\mathrm{e}) \qquad \text{Symmetrie} \tag{2.6}$$

$$M(\zeta_\mathrm{e}, \zeta_{\mathrm{s}_1}) + M(\zeta_{\mathrm{s}_2}, \zeta_\mathrm{e}) \geq M(\zeta_{\mathrm{s}_1}, \zeta_{\mathrm{s}_2}) \qquad \text{Dreiecksungleichung} \tag{2.7}$$

$$M(\zeta_\mathrm{s}, \zeta_\mathrm{e}) = 0 \leftrightarrow \zeta_\mathrm{e} = \zeta_\mathrm{s} \qquad \text{Identität} \tag{2.8}$$

wobei $M()$ eine Metrik ist. Falls $M()$ nicht–symmetrisch ist, die anderen Eigenschaften jedoch erfüllt, wird sie Quasimetrik genannt. Erfüllt $M()$ nur die Dreiecksungleichung nicht, wird sie als Semimetrik bezeichnet. Gilt selbiges nicht für die Identitätsbedingung, bezeichnet man diese als Pseudometrik [72].

2.2.2 Auswahl relevanter Ansätze der Modellvalidierung

In diesem Abschnitt wird ein Überblick über die relevanten Ansätze der Modellvalidierung gegeben. Es werden zwei generische Ansätze und eine angewandte Methode aus dem Automobilbereich vorgestellt.

Modellvalidierung und –verifizierung nach Sargent: In [62] stellt der Autor eine Methode vor, mit der die Validierung und Verifikation mit dem Modellentwicklungsprozess in Verbindung gebracht werden kann. Das Vorgehen ist in Abbildung 2.9 visualisiert. Der Entwicklungsprozess wird in die reale und die simulierte Welt unterteilt.

Zunächst wird ein reales System, dessen Verständnis erwünscht ist, in Form einer Systembeschreibung formalisiert. Darin sollen die Eigenschaften und die kausalen Zusammenhänge des Systems hinterlegt werden. Dazu werden im ersten Schritt die Systemdaten durch Experimente gewonnen. Die Systembeschreibung wird entwickelt, indem das Beobachtete aus dem System abstrahiert und gegen die experimentellen Ergebnisse hypothetisiert wird. Die Systembeschreibung wird durch die Theorievalidierung überprüft. Theoretische Validierung beinhaltet den Vergleich der Systembeschreibung mit den Experimentergebnissen. Die Entwicklung der Systembeschreibung erfordert in der Regel zahlreiche Experimente am realen System.

Sobald die Systembeschreibung vorliegt, kann die Modellentwicklung im Simulationsbereich beginnen. Diese soll nur für eine Reihe an klar definierten Zielen entwickelt werden. Das konzeptionelle Modell ist eine mathematische/logische/graphische Darstellung des Systems. Die Modellspezifikation ist eine schriftliche, detaillierte Beschreibung des Softwareentwurfs, die durch die Spezifikation des konzeptionellen Modells erfolgt. Das Simulationsmodell wird aus der Modellspezifikation implementiert und bildet das konzeptionelle Modell in Form von Computercode ab. Damit werden die Experimente durchgeführt und die Simulationsergebnisse gewonnen.

Die konzeptionelle Modellvalidierung ist definiert als die Feststellung, dass die dem konzeptionellen Modell zugrunde liegenden Annahmen mit denen in der Systembeschreibung übereinstimmen und dass die Modelldarstellung des Systems für den beab-

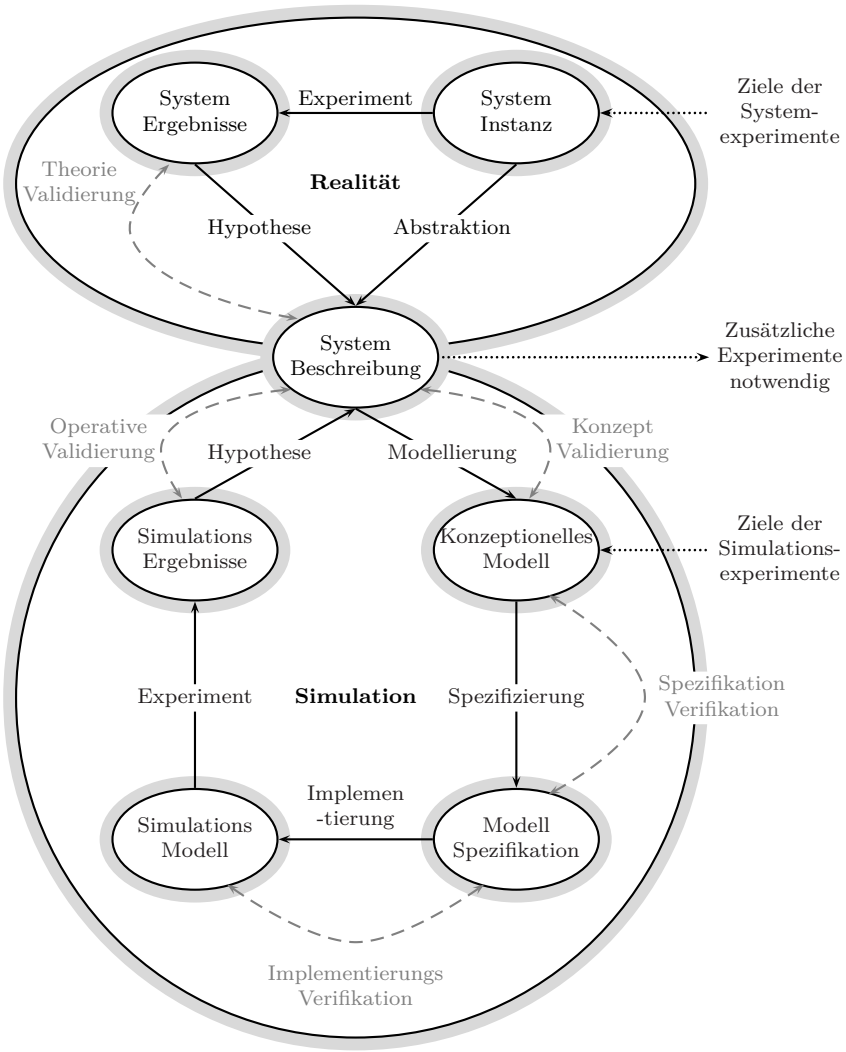


Abbildung 2.9: Modellentwicklungsprozess nach Sargent [62]. Validierung und Verifizierung werden mit dem Modellentwicklungsprozess in Verbindung gebracht. Dieser wird in Realität und Simulation aufgeteilt. Als Erstes werden aus der Realität die Systembeschreibung abgeleitet. Basierend darauf wird im nächsten Schritt das System modelliert.

sichtigten Einsatz angemessen ist. Die Spezifikationsverifikation ist definiert als Sicherstellung, dass der Softwareentwurf und die Spezifikation für die Programmierung und Implementierung des konzeptionellen Modells zufriedenstellend ist. Die Implementierungsverifikation ist definiert als die Sicherstellung, dass das Simulationsmodell gemäß der Spezifikation des Simulationsmodells implementiert wird. Operative Validierung ist definiert als die Feststellung, dass das Aufgabenverhalten des Modells eine zufriedenstellende Genauigkeit für den beabsichtigten Einsatz des Modells über den Bereich der beabsichtigten Anwendbarkeit des Modells aufweist.

Der von Sargent in [62] vorgestellte Ansatz zeigt somit die Beziehungen, die bei der Entwicklung von Systembeschreibung und Simulationsmodellen verwendet werden. Beides wird durch iterative Prozesse erreicht. Dazu wird das System zuerst beobachtet und

daraus die Systembeschreibung abgeleitet. Diese wird auf Korrektheit geprüft, indem die Experimentierergebnisse mit der vorgeschlagenen Systembeschreibung verglichen werden. Dieser Vorgang wird wiederholt, bis ein zufriedenstellender Satz validierter Systembeschreibungen vorliegt. In ähnlicher Art und Weise wird ein Simulationsmodell entwickelt. Dazu wird zuerst aus den validierten Systembeschreibungen das konzeptionelle Modell abgeleitet. Für das konzeptionelle Modell wird eine Simulationsmodell–Spezifikation entwickelt, gegen die ein Simulationsmodell auf dem Rechner implementiert wird. Am Simulationsmodell werden Experimente durchgeführt, um Daten für die operative Validierung zu generieren. In der Regel werden mehrere Versionen eines Modells entwickelt, bevor ein gültiges Simulationsmodell vorliegt.

Modellvalidierung nach Oberkampf und Trucano: Zur Validierung von Simulationsmodellen schlagen die Autoren in [64] ein generisches Vorgehen vor, das drei Aspekte beleuchtet (siehe Abbildung 2.10). Im Gegensatz zu dem zuvor vorgestellten Ansatz von Sargent, bei dem Validierungs– und Verifizierungsaspekte berücksichtigt werden, liegt der Fokus bei diesem Verfahren nur auf der Validierung.

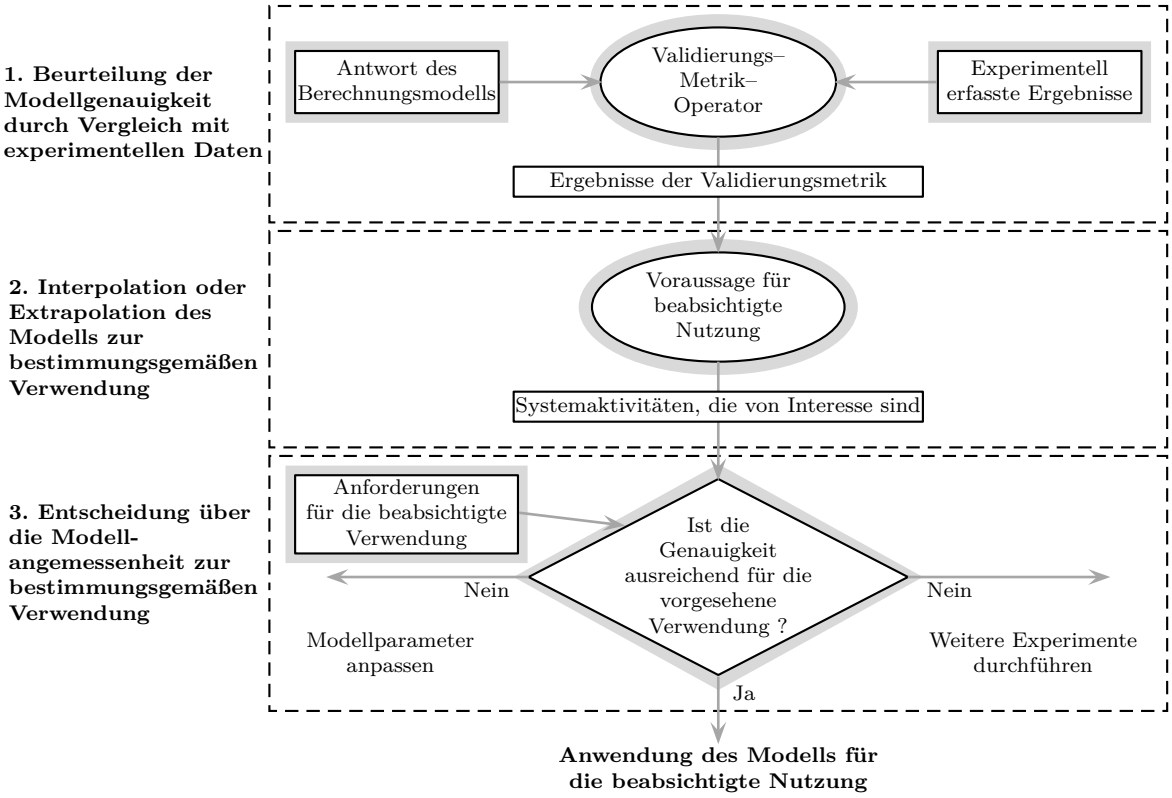


Abbildung 2.10: Generische Validierungsmethode nach Oberkampf [64]. Im ersten Schritt werden die Modellergebnisse mit experimentellen Daten verglichen. Im Zweiten Schritt werden die Modellergebnisse inter– beziehungsweise extrapoliert und schließlich die Modellgenauigkeit unter Berücksichtigung der Anforderungen für beabsichtigte Verwendung bewertet.

Der erste Aspekt befasst sich mit der Beurteilung der Genauigkeit der Ergebnisse aus dem Modell durch Vergleich der Modell– und Experimentdaten. Die Modellgenauigkeit wird mit Hilfe einer Validierungsmetrik quantitativ geschätzt. Im Allgemeinen ist die Metrik ein statistischer Operator, da die experimentellen Ergebnisse oft als Verteilungsfunktionen vorliegen. Häufig wird nur der erste Aspekt als Validierung gesehen, wobei die Autoren der Meinung sind, dass die Aspekte zwei und drei unumgänglich sind.

Beim zweiten Aspekt wird das Modell für die Extra– beziehungsweise Interpolation der Daten für die beabsichtigte Nutzung des Modells verwendet. Die Annahme ist dabei, dass das zu erwartende Ergebnis bekannt ist. Die Prädiktion ist notwendig, um nach der Validierung Aussagen über die Vorhersagefähigkeit eines Modells treffen zu können. Ähnliches ist in dem „Guide for Verification and Validation in Computational Solid Mechanics“ der American Society of Mechanical Engineers [73] definiert.

Der dritte Aspekt befasst sich mit dem Vergleich der geschätzten Genauigkeit des Modells relativ zu den Genauigkeitsanforderungen und der Entscheidung über die Angemessenheit des Modells für den beabsichtigten Einsatz. Mit anderen Worten werden die Ergebnisse aus dem vorherigen Schritt mit dem zu erwartendem Resultat verglichen. Falls die zu erwartenden Anforderungen vom Modell nicht erfüllt werden, muss entweder das Modell neu parametriert oder zusätzliche Experimente durchgeführt werden, um weitere Erkenntnisse zu gewinnen.

Validierung umfelderfassender Sensormodelle nach Roth: In [74] stellt der Autor eine Methode zur Validierung virtueller umfelderfassender Sensormodelle für die Entwicklung von Fahrerassistenzsystemen im Automobilbereich vor (siehe Abbildung 2.11). Im Gegensatz zu den oben beschriebenen generischen Ansätzen, ist diese Methode anwendungsorientiert und hat folgende Zielsetzung: (a) Beschreibung und Implementierung einer Methodik zur halbautomatischen Generierung von synthetischen Sensordaten auf Basis aufgezeichneter Testfahrten, (b) Spezifikation und Implementierung einer

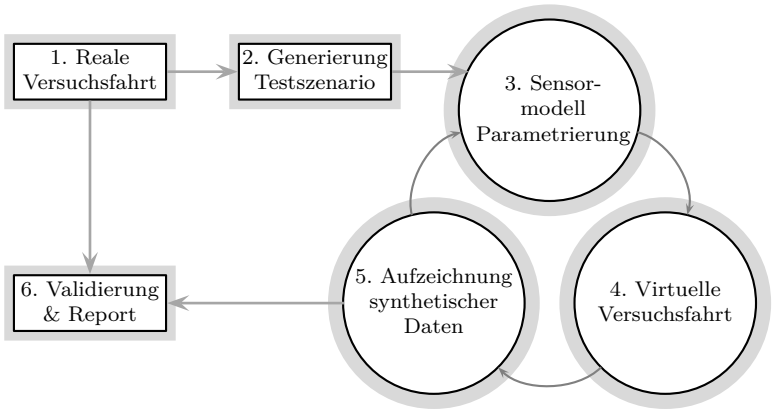


Abbildung 2.11: Methode zur Validierung von Sensormodellen nach Roth [74]. Als Erstes werden reale Daten aufgezeichnet und basierend darauf ein Testszenario für Simulation erstellt. Anschließend wird das Modell parametriert, die virtuelle Fahrt vorgenommen und synthetische Daten aufgezeichnet. Im letzten Schritt werden die realen und synthetischen Daten miteinander verglichen.

Methodik für einen halbautomatischen Vergleich von realen und synthetischen Daten umfelderfassender Sensoren auf Objektlistenebene und (c) Spezifikation von Softwareschnittstellen zur Analyse von realen und synthetischen Sensordaten. Die vorgestellte Methode beinhaltet folgende sechs Schritte:

1. Aufzeichnung realer Daten.
2. Erstellung eines virtuellen Szenarios.
3. Parametrierung der Sensormodelle.
4. Durchführung virtueller Testfahrten.
5. Aufzeichnung synthetischer Sensordaten.
6. Validierung realer und synthetischer Daten.

Im ersten Schritt werden die realen Testfahrten auf Basis eines zuvor definierten Testmanöverkatalogs durchgeführt und die Versuchsparameter protokolliert. Die Aufzeichnung von Referenzpositions– und Umgebungssensordaten während der eigentlichen Testfahrt erfolgt über eine Messeinrichtung namens „RefBox“. Diese ist in allen Fahrzeugen, die an der Testsfahrt teilnehmen, installiert. Die RefBox verwendet ein GNSS–System und dient als Referenz für alle beweglichen Objekte in den Szenarien.

Zweitens wird basierend auf den aufgezeichneten Referenzdaten ein virtuelles Testszenario generiert. Als Simulationsumgebung werden in [74] „Virtual Test Drive“ (VTD) und als Evaluierungsframework Matlab eingesetzt. Die aufgezeichneten Trajektorien der verschiedenen Testläufe werden durch ein Matlab–Framework in ein für VTD lesbares Datenformat umgewandelt. Basierend darauf, ist VTD in der Lage das Testszenario innerhalb der Messgenauigkeit der Referenz wiederzugeben.

Im dritten Schritt werden die Sensormodelle in der Simulation entsprechend den Parametern des realen Versuchsaufbaus und der Sensorausstattung parametriert. Dabei werden idealisierte Sensormodelle verwendet und eine Möglichkeit gegeben den Sensorsichtbereich und die relative Sensorposition zum Fahrzeug zu konfigurieren.

In den Schritten vier und fünf werden die virtuellen Testfahrten auf Basis der zuvor erstellten Testszenarios durchgeführt, synthetische Sensordaten generiert und auf der gleichen Abtastzeitbasis wie die realen Sensordaten aufgezeichnet.

Schließlich werden die erfassten synthetischen und realen Daten evaluiert. Dabei werden zu jedem Simulationszeitschritt die aufgezeichneten realen Daten in Matlab importiert und die Messwerte miteinander qualitativ verglichen.

Nach genauer Betrachtung der vorgestellten Methoden ist festzuhalten, dass diese unterschiedliche Schwerpunkte haben. Die Methode von Sargent betrachtet den allgemeinen Modellentwicklungsprozess und integriert darin die Validierungs– und Verifizierungsaspekte. Die Methode von Oberkampf gibt den Rahmen und die Empfehlungen für einen erfolgreichen Validierungsprozess, unter der Annahme, dass das Modell bereits verifiziert wurde, und integriert einen Prädiktionsschritt des Modells in den Validierungsprozess. Die Methode von Roth beschäftigt sich mit der Frage, wie die benötigten Daten für die Validierung von umfelderfassenden Sensormodellen erfasst werden können und wie die Werkzeugkette für die Evaluierung dieser Daten auszusehen haben.

Alle drei Ansätze basieren auf dem Vergleich von realen und synthetischen Daten, keiner der Autoren schreibt jedoch vor, welche Validierungsmetriken dafür zu verwenden sind. Dies ist bewusst so gehalten, denn die Anzahl möglicher Validierungsmetriken ist groß und die Wahl anwendungsspezifisch, weshalb es keine allgemeingültige Lösung gibt. Die im Rahmen dieser Arbeit verwendeten Validierungsmetriken werden an den jeweiligen Stellen im weiteren Verlauf eingeführt.

2.2.3 Anforderungen an die Modellvalidierung

Basierend auf den Erkenntnissen aus diesem Kapitel, wird hier eine Reihe von Anforderungen an die Validierung von Modellen formuliert, die die Antwort auf den ersten Teil der ersten Forschungsfrage aus Kapitel 1.2 geben.

Wie einleitend erwähnt, beschäftigt sich diese Arbeit mit Validierung umfelderfassender Sensormodelle. Diese entwickeln sich mit dem Entwicklungsprozess von Systemen, in denen diese zum Einsatz kommen, mit. Es ist deshalb wichtig, dass der Modellvalidierungsprozess **durchgehend** in den gesamten Lebenszyklus eines Modells, vom Entwurf bis zum Einsatz, integriert ist. Die Werke [61, 62, 69] bestätigen diese Anforderung. In diesem Zusammenhang ist die **Skalierung** der Modellvalidierung zu erwähnen. Grund hierfür ist die häufige Anwendung einer Validierungsmethode während des Modelllebenszyklus. Eine nicht skalierbare Methode ist deshalb für industrielle Anwendung aus wirtschaftlicher Sicht nicht akzeptabel.

Ein Teil der Methoden von Sargent [62] und Oberkampf [64] ist die **Spezifizierung** der Anforderungen an die Modellvalidierung auf den beabsichtigten Einsatz. Ein Modell kann und muss deshalb nicht vollständig validiert werden, sondern nur zu einem bestimmten Grad innerhalb eines bestimmten Betriebsbereichs unter den Prämissen des Einsatzbereichs. Ähnliches kommt aus den Werken [69, 75] hervor. Aus Systemsicht darf die **Gesamthaftigkeit** jedoch nicht vernachlässigt werden. Denn das erfolgreiche Testen jedes Teilmodells impliziert nicht die Validität des Gesamtsystems. Darüber hinaus garantiert ein akzeptiertes Modell nicht zwangsweise die Akzeptanz der Simulationsergebnisse [61, 64].

In Bezug auf die Sensormodelle wird in [36] eine Liste von Anforderungen formuliert. Der am meisten relevante Punkt für die Modellvalidierung aus dieser Liste ist die **Nachvollziehbarkeit**. Sollten die Ergebnisse eines Modells zu einem Fehler in der nachgelagerten Verarbeitungskette führen, soll die Ursache dieses Verhaltens nachvollziehbar sein. Hierdurch kann beispielsweise identifiziert werden, ob ein Fehler in der Programmierung oder falsche Parametrierung des Modells zu dem Fehler geführt hat. Ähnliches kann aus den Werken [61, 65] abgeleitet werden. Damit hängt auch **Intuitivität** der Validierungsergebnisse zusammen. Die Autoren in den Werken [57, 65, 76] schreiben, dass Validirung für mehrere Interessengruppen von Bedeutung ist. Intuitivität ist deshalb neben Entwicklern auch für andere Entscheidungsträger und die Allgemeinheit relevant, da davon die Akzeptanz einer Validierungsmethode abhängt.

Ein weiterer Aspekt ist die **Objektivität** der Ergebnisse einer Validierungsmethode. Um die Voreingenommenheit des Entwicklers auszuschließen, muss eine Validierungs-

methode unabhängig vom Anwender immer gleiche Ergebnisse bei gleicher Ausgangssituation liefern [61, 65]. Die Planung und **Dokumentation** der Validierungsaktivitäten schafft dafür eine Grundlage. Dadurch steigt die Rückführbarkeit, Glaubwürdigkeit und somit die Akzeptanz einer Validierungsmethode [61, 69].

Bei der Implementierung einer Validierungsmethode ist außerdem auf die **Fehlervermeidung** vom Typ I, II und III zu achten. Ein Fehler vom Typ I liegt vor, wenn die Simulationsergebnisse abgelehnt werden, obwohl sie tatsächlich ausreichend glaubwürdig sind. Ein Typ II–Fehler tritt auf, wenn ungültige Simulationsergebnisse akzeptiert werden, als ob sie ausreichend gültig wären. Ein Fehler vom Typ III tritt auf, wenn das falsche Problem gelöst und bestätigt wird beziehungsweise wenn das formulierte Problem das eigentliche Problem nicht vollständig enthält [61, 65]. Des Weiteren muss eine Validierungsmethode so ausgelegt werden, dass die **Unsicherheiten** aus Kapitel 2.2.1 möglichst klein gehalten werden, um das volle Potential eines Modells auszuschöpfen [62, 65, 69].

Zwei der oben aufgeführten Anforderungen sind als Richtlinien zu verstehen: Durchgängigkeit und Dokumentation. Diese beschreiben eher eine Empfehlung, wie ein Validierungsprozess möglichst wirkungsvoll umgesetzt werden soll und werden ausführlich in den Werken [61, 69] beschrieben. Da diese Punkte keine offenen Fragen hinterlassen, werden sie nicht weiter in dieser Arbeit berücksichtigt. Die acht anderen Aspekte werden beim Entwurf und Umsetzung der Methode zur Validierung umfelderfassender Sensormodelle in Kapiteln 3, 4 und 5 berücksichtigt.

3 Methoden zur Validierung und Bedatung umfelderfassender Sensormodelle

Basierend auf den Erkenntnissen aus Kapitel 1 und 2 werden in diesem Kapitel Methoden zur Validierung und Bedatung von Sensormodellen für umfelderfassende Sensoren im Automobilbereich entwickelt. Wie in der Vorstellung der Modellierungsansansätze in Kapitel 2.1 bereits erläutert, werden für die Modellierung unterschiedlicher umfelderfassender Sensoren in der Praxis unterschiedliche Modellierungsansätze verwendet. Beispielsweise kommen in [28] und [25] statistische (parametrische und nicht–parametrische) Verfahren zum Einsatz. In [27] und [35] werden physikalische (Strahlverfolgung und nicht–parametrische) Modelle eingesetzt. Auch gemischte Verfahren, die beispielsweise parametrische und nicht–parametrische Ansätze verwenden, kommen zum Einsatz [33]. Des Weiteren werden als Ausgangsformate im Automobilbereich meist die in 2.1.1 vorgestellten Objektlisten und Rohdaten verwendet. Diese Erkenntnisse sind in Tabelle 3.1 zusammengefasst.

Die Modellierung der Sensoren kann im Kontext des autonomen Fahrens nicht als in sich geschlossene Aufgabe gesehen werden. Wie anhand von Abbildung 1.1 zu sehen ist, stellen die Sensoren nur eine Teilkomponente des gesamten Systems dar. Des Weiteren handelt es sich bei diesem Anwendungsfall um sehr dynamische Umgebungszustände, auf die die Sensorik mit Aktualisierungsraten von bis zu 100Hz reagieren muss. Alle diese Tatsachen haben einen Einfluss auf die Validierung und müssen berücksichtigt werden. In Kapitel 3.1 wird deshalb eine Methode zur Validierung von Sensormodellen auf Objektlisten– und Rohdatenebene entworfen. Anschließend wird in Kapitel 3.2 da-

Tabelle 3.1: Zusammenfassung der Literaturrecherche zu Modellimplementierungen, sortiert nach Modellierungsansatz und Modellierungsebene [24–35].

		Modellierungsansatz		
		Statistisch		Feldausbreitung
		Parametrisch	Nicht–Parametrisch	Strahlenoptisch
Ebene	Objektliste	[25, 31–33]	[27, 28, 30, 34]	
	Rohdaten		[30, 34]	[24, 29, 33, 35]

für eine Werkzeugkettenarchitektur vorgestellt.

Ein weiterer wichtiger Aspekt für die Validierung von datengetriebenen Sensormodellen ist deren Bedatung. Unter datengetriebenen Modellen im Kontext dieser Arbeit sind solche gemeint, die auf die realen Daten zur Laufzeit beziehungsweise während der Lernphase angewiesen sind. Darunter fallen insbesondere die nicht–parametrischen Methoden und die lernenden Algorithmen, deren Leistungsfähigkeit von der Anzahl der zur Verfügung stehenden realen Szenarien abhängt. Unter Bedatung solcher Modelle ist die Bereitstellung der Eingangsdaten gemeint. Die Güte von datengetriebenen Modellen hängt deshalb unmittelbar von den Eingangsdaten und somit der Bedatung ab. Idealerweise müsste ein Datensatz alle relevanten Szenarien aus der Realität enthalten. Dies ist jedoch aufgrund der Größe des Parameterraums und der damit verbundenen Kombinatorik nicht realisierbar, was mit dem folgenden Beispiel verdeutlicht wird.

Es soll geschätzt werden, wie viele Datenpunkte im Idealfall für die Modellierung eines hypothetischen Sensors auf Objektlistenebene notwendig sind. Dabei werden folgende drei Annahmen getroffen: (a) die Aufzeichnung findet innerhalb des Sichtbereichs statt, (b) der Sensor muss jeden Abschnitt des Sichtbereichs einmal gesehen haben und (c) für dieses Beispiel werden als Parameter nur der Abstand, der Azimutwinkel und die Geschwindigkeit berücksichtigt. Die Anzahl der notwendigen Datenpunkte N wird wie folgt berechnet,

$$N = \frac{D}{\Delta_D} \cdot \frac{\Theta}{\Delta_\Theta} \cdot \frac{V}{\Delta_V} \tag{3.1}$$

wobei $D = 200\text{m}$ die maximale Reichweite, $\Delta_D = 10\text{cm}$ die Abstandsauflösung, $\Theta = 360°$ den Sichtbereich in Azimutebene, $\Delta_\Theta = 0{,}25°$ die Azimutauflösung, $V = 130\frac{\text{km}}{\text{h}}$ die maximale Geschwindigkeit und $\Delta_V = 0{,}25\frac{\text{m}}{\text{s}}$ die Geschwindigkeitsausflösung des Sensors beschreiben. Wie bereits erwähnt, beziehen sich die hier verwendeten Zahlen auf einen hypothetischen Sensor, der realen Sensoren nachempfunden ist. Die Berechnung führt zu $N = 0{,}416 \cdot 10^9$ Datenpunkte. Mit einer Aktualisierungsrate von 25 Hz würde es circa 192,6 Tage kontinuierlicher Messaufnahme erfordern. Dieses Beispiel zeigt, dass die Messdatenaufnahme sehr zeitaufwändig sein kann. Da die Güte datengetriebener Sensormodelle von den Eingangsdaten abhängt und einen unmittelbaren Einfluss auf die Validierung hat, werden in Kapitel 3.4 und Kapitel 3.5 zwei Ansätze für die Datenvorauswahl durch die Reduktion von relevanten Modellierungsparametern und systematischer Datenerfassung entwickelt.

3.1 Methode zur Validierung umfelderfassender Sensormodelle

Unter Beachtung der Erkenntnisse aus Kapitel 1 und 2 sowie den einleitenden Überlegungen in diesem Kapitel, wird hier eine Methode zur Validierung umfelderfassender Sensormodelle formuliert. In Bezug auf die Sensormodellvalidierung lassen sich die acht relevanten Anforderungen aus Kapitel 2.2.3 wie folgt erweitern

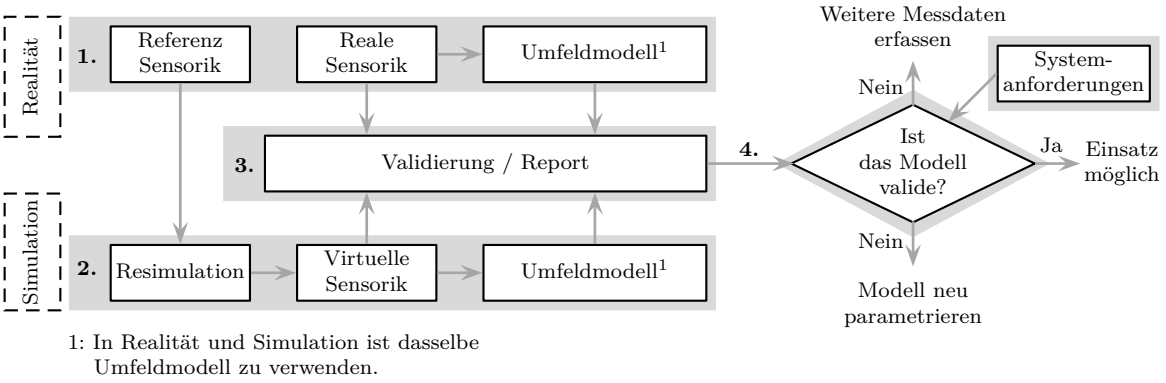


Abbildung 3.1: Visualisierung der Methode zur Validierung umfelderfassender Sensormodelle.

- Die bekannten Modellierungsansätze im Automobilbereich lassen sich in parametrische, nicht–parametrische und strahlenoptische kategorisieren.
- Es werden hauptsächlich zwei Datenformate unterschieden: Objektlisten und Rohdaten.

Die insgesamt zehn aufgeführten Anforderungen werden beim Entwurf der Methode berücksichtigt. Der Vorschlag ist, die Sensormodellvalidierung in vier Schritte zu unterteilen: Als Erstes werden die Referenz– und realen Messungen sowie die Daten des Umfeldmodells aufgezeichnet. Im zweiten Schritt wird, basierend auf den Referenzdaten, das reale Szenario resimuliert und daraus die synthetischen Sensordaten sowie die Umfeldmodellergebnisse generiert. Die erfassten Wertepaare der realen und synthetischen Sensordaten sowie die Ergebnisse des Umfeldmodells werden anschließend im Validierungsschritt bewertet. Schließlich kann die Entscheidung bezüglich der Akzeptanz des Sensormodells, basierend auf dem Validierungsschritt, getroffen werden. Die einzelnen Schritte sind in Abbildung 3.1 visualisiert und werden im Folgenden erläutert.

1. Reale Messdatenerfassung: Grundsätzlich basiert die hier vorgestellte Validierungsmethode auf dem Vergleich realer und synthetischer Daten. Als Erstes sind demzufolge die realen Daten aufzuzeichnen. Im Realfahrzeug läuft die Verarbeitung der Daten folgendermaßen ab (siehe Abbildung 1.1): Die im Fahrzeug verbauten Sensoren (Radar, Lidar, Kamera, Ultraschall und weitere) erfassen die Umgebung und stellen diese Informationen dem Umfeldmodell zur Verfügung. Dort werden im ersten Schritt die Sensordaten fusioniert. Wie in Kapitel 2.1.1 erwähnt, können die Sensordaten als Objektlisten oder Rohdaten vorliegen, weshalb bei der Fusion ebenfalls entsprechend zwischen Objektlistenfusion und Rohdatenfusion unterschieden wird. Das Umfeldmodell stellt dann die fusionierte Repräsentation der Umgebung der autonomen Fahrfunktion zur Verfügung. Diese trifft darauf basierend die Entscheidung, wie sich das Fahrzeug zu verhalten hat und gibt Steuersignale an die Aktuatorik. Dadurch wird die Fahrdynamik beeinflusst und der Vorgang wiederholt sich.

Naheliegend ist, dass die **Messdaten realer zu modellierender Sensoren** zu erfassen sind. Die realen Daten sind für den Nachweis der Validität eines Sensormo-

dells erforderlich, jedoch nicht hinreichend. Wie bereits erwähnt, besteht das autonome System aus zahlreichen Funktionsblöcken, wovon die Sensoren nur eine Komponente darstellen. Die Validität einer Komponente ist erst dann erfüllt, wenn die nachfolgende Komponente ein akzeptables Betriebsverhalten aufweist. Dieser Meinung sind auch die Autoren in [61, 65]. Es wird deshalb vorgeschlagen, die **Daten der nachgelagerten Verarbeitungsstufe** der Sensordaten zu evaluieren und mit in den Validierungsprozess zu integrieren. In dieser Methode wird sie als Umfeldmodell bezeichnet. In dem oben beschriebenen Regelkreis einer autonomen Fahrfunktion handelt es sich konkret um eine Schicht des Umfeldmodells beziehungsweise der Fusion von Sensordaten.

Grundsätzlich unterscheiden sich die Validierungsexperimente von den herkömmlichen [65, Kapitel 10]. Ein Validierungsexperiment dient in erster Linie der Bestimmung der Vorhersagefähigkeit eines mathematischen Modells von physikalischen Prozessen und der Minimierung der Unsicherheiten. Mit anderen Worten, ein Validierungsexperiment wird entworfen, durchgeführt und analysiert, um die Fähigkeit des Modells und seiner Verkörperung in Computercode zur Simulation eines gut charakterisierten physikalischen Prozesses quantitativ zu bestimmen. Die Charakterisierung der Güte wiederum erfolgt meist durch den Vergleich von Fehlerverteilungen. Um all diesen Aspekten Genüge zu tun, wird vorgeschlagen die Validierungsexperimente in den Validierungsprozess mit zu integrieren und die Berechnung der Fehlerverteilungen, basierend auf Referenzdaten, durchzuführen. Dazu werden bei der Datenerfassung für die Validierung zusätzlich die **Messungen der Referenzsensorik** aufgezeichnet. Unter Referenzsensorik werden Sensorsysteme verstanden, die mindestens um etwa eine Größenordnung höhere Genauigkeit, Zuverlässigkeit und Aktualisierungsraten im Vergleich zu dem zu charakterisierenden Sensor aufweisen [77]. Sollten diese Anforderungen erfüllt sein, kann in guter Näherung angenommen werden, dass die Referenzdaten den tatsächlichen realen Daten ζ_r (englisch: Ground Truth (GT)) entsprechen.

2. Simulation: Im nächsten Schritt werden die synthetischen Daten aus der Simulation generiert. Zunächst erfolgt die **Resimulation des real gefahrenen Szenarios**. Unter Resimulation ist das Nachfahren der realen Trajektorien in einer virtuellen Umgebung gemeint. Die Generierung des dynamischen Trajektorienverlaufs erfolgt dabei aus den Referenzdaten. Durch den Einsatz exakt der selben realen Trajektorie in der Simulation wird sichergestellt, dass die Fehler aus Gleichung 2.5 quantitativ erfasst werden können, ohne weitere Unsicherheiten einzuführen.

Die Information aus der Resimulation ist ein Pendant der Realität, oder mit anderen Worten das GT für die virtuelle Sensorik. Das Sensormodell generiert basierend darauf die **realitätsnahen synthetischen Daten**, die als Eingangsdaten für das Umfeldmodell dienen. Wichtig ist dabei, dass es sich um exakt das gleiche Umfeldmodell wie in den Realversuchen handelt. Es dürfen folglich keine Anpassungen daran für die Simulation getätigt werden. Damit wird erreicht, dass sich bis auf die marginalen Fehler in der Referenzsensorik die eingangs– und ausgangsseitigen Bedingungen des Sensormodells im Vergleich zur Realität nicht unterscheiden. Mit anderen Worten, Realität und Simulation sind in den relevanten Aspekten bis auf das Sensormodell gleich. Dies hat

den Vorteil, dass die in Kapitel 2.2.1 angesprochenen Unsicherheiten minimal gehalten werden.

Zusammenfassend werden bei der experimentellen Datenerfassung die Messungen der Referenzsensorik, des zu modellierenden Sensors sowie des Umfeldmodells aufgezeichnet. In der Simulation werden die dazu korrespondierenden synthetischen Wertepaare aus den Sensormodellen und darauf basierende Ergebnisse des Umfeldmodell erfasst, die im darauf folgenden Schritt miteinander verglichen werden.

3. Validierung: Wie bereits mehrmals erwähnt, dient die Validierung der Gütebewertung eines Modells basierend auf dem Abgleich der Realität mit der Simulation und gibt eine Handlungsanweisung, ob ein Modell valide ist. In Tabelle 3.1 ist festgehalten, dass die Daten dabei entweder auf der Objektlisten– oder Rohdatenebene vorliegen, die Validierung jedoch unterschiedliche Modellierungsansätze unterstützen soll.

Der Vergleich kann basierend auf einer qualitativen oder quantitativen Bewertung erfolgen. Die qualitative Bewertung erfolgt meist mittels *Graphen*. Diese stellen die einfachste Möglichkeit der Bewertung, ob die Modellergebnisse das erwartete Verhalten aufweisen. Graphen sind eine sehr einfache und intuitiv verständliche Bewertungsmöglichkeit. Nachteilig ist jedoch der qualitative Charakter, der die Entscheidung über die Validität eines Modells erschwert. Quantitative Bewertung erfolgt mittels mathematischen Operatoren, auch **Metriken** genannt (siehe Kapitel 2.2.1). Für den Automobilbereich relevante Metriken können in folgende Kategorien eingeteilt werden: (a) Hypothesentests, (b) Konfidenzintervalle, (c) Vergleich von Wahrscheinlichkeitsdichtefunktion (WDF) und (d) Korrelationsmessungen [65, 78, 79]. *Hypothesentests* können beispielsweise zum Vergleich von Mittelwerten, Varianzen, Verteilungen und Zeitreihen der Modellierungsparameter, basierend auf einem Akzeptanzniveau, verwendet werden. Diese können jedoch die Frage, ob ein Modell valide ist, nur mit „ja“ oder „nein“ beantworten. Dies ist für viele Anwendungen, bei denen eine kontinuierliche Differenzquantifizierung zwischen Experiment und Modell wünschenswert ist, jedoch nicht ausreichend. *Konfidenzintervalle* können statt „ja“ oder „nein“ einen Intervall für die geschätzte Differenz einer Modellierungsgröße zwischen Sensor und Modell angeben. *Vergleiche von WDF* basieren im Gegensatz zu Hypothesen und Konfidenzintervallen auf dem Vergleich der Verteilungsform und wurden in den letzten Jahren intensiv untersucht [80–82]. Die letzte Kategorie beinhaltet die *Korrelationsmessungen*. Diese dienen der Bewertung der Ähnlichkeit von zwei oder mehreren statistischen Größen. Des Weiteren gibt es in [83, 84] Mischansätze zur Definition anwendungsbezogener Metriken. Im Rahmen der Arbeit werden, abhängig von den zu validierenden Inhalten, unterschiedliche Metriken verwendet und an den jeweiligen Stellen eingeführt.

Ein weiterer Aspekt im Validierungskontext ist die **Auswahl eines Modells** aus mehreren möglichen, das am besten die experimentellen Daten abbildet. Dabei ist wichtig, das Model nicht zu sehr an die aufgezeichneten Daten anzupassen, um die Allgemeingültigkeit nicht zu verlieren. Dieses Verhalten wird als Überanpassung (englisch: Over–Fitting) bezeichnet. Es gibt mehrere Techniken, um die Überanpassung zu

vermeiden. In der Praxis hat sich jedoch das k–fache Kreuzvalidierungsverfahren (englisch: k–Fold Cross–Validation) durchgesetzt. Dazu werden die aufgezeichneten Daten in k gleichgroße Teilmengen unterteilt. Die erste Teilmenge wird zum Testen und die restlichen zum Trainieren des Modells verwendet. Dieser Vorgang wird k–mal mit jeweils einer anderen Teilmenge wiederholt. Am Ende wird die Güte des Modells gemittelt, was eine gute Abschätzung der Modellperformanz ergibt [85]. Das Modell mit dem besten mittleren Ergebnis kann als zielführend erachtet werden.

4. Entscheidungsfindung: Validierung und Entscheidungsfindung sind bewusst voneinander getrennt, denn wie in 2.2.3 erwähnt, müssen die Validierungsergebnisse objektiv und unabhängig vom Anwender und Anwendungsfall sein. Die Akzeptanz eines Modells ist jedoch anwendungsspezifisch, weshalb bei diesem Schritt die Modellanforderungen in Betracht zu ziehen sind. Die Anforderungen können sich dabei sowohl direkt auf das Modell als auch auf das Umfeldmodell beziehen.

Die Frage, ob ein Modell für den beabsichtigten Zweck valide ist, kann mit „ja“ oder „nein“ beantwortet werden. Die „ja“–Entscheidung kann dabei in zwei Richtungen ausfallen: (a) Die gestellten Anforderungen sind erfüllt, weshalb das Modell als valide zu betrachten ist und (b) der Parameterbereich der beabsichtigten Verwendung des Modells ist so einzuschränken, dass die zulässigen Abweichungen nicht überschritten werden. Falls die Entscheidung negativ ausfällt, sind situationsabhängig entweder weitere Experimente durchzuführen, oder das Modell ist neu zu parametrieren. Diese Entscheidung hängt von der Modellierungsart ab.

Die hier vorgestellte Methode gibt die Antwort auf den zweiten Teil der ersten Forschungsfrage aus Kapitel 1.2. Grundsätzlich ist diese Methode zur Validierung von Modellen für Objektlisten und Rohdaten ausgelegt. Aufgrund der Tatsache, dass zur Modellierung dieser Datenformate meist unterschiedliche Modellierungsansätze mit unterschiedlichen Anforderungen verwendet werden (siehe Tabelle 2.1), müssen diese Unterschiede in der Werkzeugkette berücksichtigt werden. Dafür wird im nächsten Unterkapitel ein Vorschlag erarbeitet.

3.2 Architekturvorschlag

Die vorgestellte Validierungsmethode wirft noch einige Umsetzungsfragen der einzelnen Komponenten auf, die in diesem Abschnitt beantwortet werden. Dazu ist in Abbildung 3.2 ein Architekturvorschlag dargestellt. Dieser ist in drei Aufgabenbereiche unterteilt: (a) Datenerfassung, (b) Validierung und (c) Simulation. Der gesamte Vorgang besteht wiederum aus zwei Evaluierungsschleifen: In der ersten Schleife werden die relevanten Modellierungsparameter definiert und die Datenerfassung systematisiert. Dies erfolgt folgendermaßen: (1a) experimentelle Daten werden mit einem Versuchsfahrzeug aufgezeichnet, (1b) reale Daten werden importiert, (1c) importierte Daten werden gereinigt und assoziiert, (1d) die zu modellierenden Parameter priorisiert und die Datenaufzeichnung systematisiert.

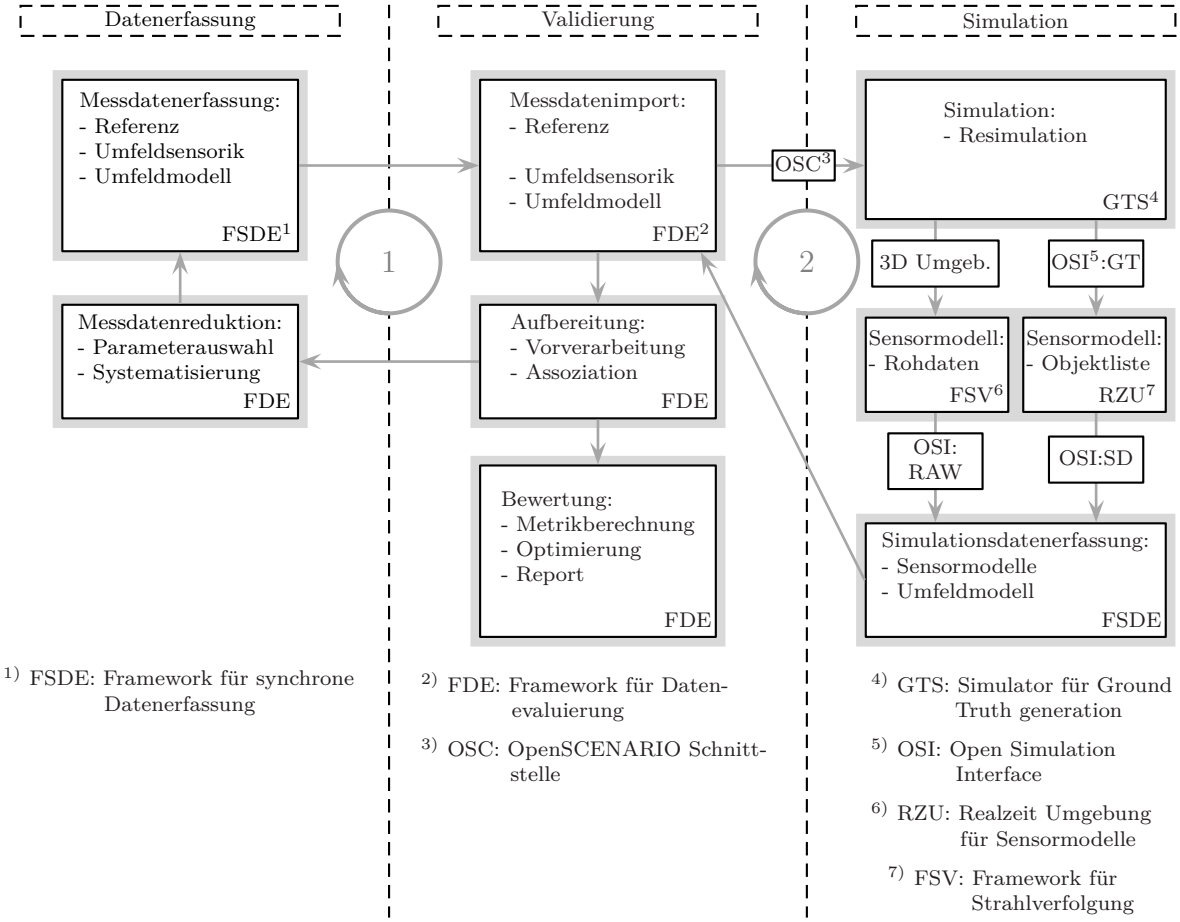


Abbildung 3.2: Schematische Darstellung notwendiger Komponenten zur Validierung umfelderfassender Sensormodelle.

In der zweiten Schleife werden die zuvor gewonnen Informationen verwendet, um die Modelle zu bewerten. Der Vorgang sieht folgendermaßen aus: (2a) experimentelle Daten werden für die reduzierte Anzahl von Parametern mit einem Versuchsfahrzeug aufgezeichnet, (2b) reale Daten importiert, (2c) die realen Szenarien resimuliert und synthetische Daten erzeugt, (2d) simulierte Daten importiert und (2e) identisch zu der ersten Schleife gereinigt und assoziiert, (2f) reale und synthetische Daten ausgewertet. Im Folgenden werden die einzelnen Schritte näher erläutert.

Zur **Erfassung realer Daten** (Abbildung 3.2 links) ist ein Versuchsfahrzeug, ausgestattet mit Umfelderfassungs– und Referenzsensorik, notwendig. Mögliche umfelderfassende Sensoren sind Radar, Lidar oder Kamera. Die Daten dieser Sensoren sollen als Objektlisten und Rohdaten vorliegen (siehe Kapitel 2.1.1). Als Referenzsensorik eignet sich ein Differential Global Navigation Satellite System (DGNSS) in Kombination mit einer inertialen Messeinheit (IMU). Eine Variante ist das Referenzsysten RT3000 von „Oxford Technical Solutions“ [86, 87], das im Rahmen dieser Arbeit zum Einsatz kommt. Die Referenzdaten sollen dabei dieselben Informationen, wie die Objektlisten

der zu validierenden Sensoren zur Verfügung stellen. Des Weiteren sind die Messergebnisse des Umfeldmodells zu erfassen. Im Kontext des autonomen Fahrens repräsentiert das Umfeldmodell (siehe Abbildung 1.1) jenes in der Validierungsmethodik. Wie bereits erwähnt, ist die primäre Aufgabe des Umfeldmodells der Aufbau einer einheitlichen Umgebungsrepräsentation aus den zur Verfügung stehenden Sensordaten. Der erste Schritt eines Umfeldmodells ist meist die Fusion dieser Daten. Da die Sensordaten entweder als Objektlisten oder Rohdaten vorliegen, sind unterschiedliche Fusionsansätze notwendig. Als Objektlistenfusion eignen sich die sogenannten Track–to–Track–Verfahren [18, 88], als Rohdatenfusion zum Beispiel die Belegungsrasterkarten [40,89]. Um die realen Daten zu erfassen, ist ein Framework für synchrone Datenerfassung (FSDE) notwendig. Dafür eignen sich beispielsweise folgende Werkzeuge: „Robot Operating System“ (ROS) [90], „Automotive Data and Time–Triggered Framework“ (ADTF) [91] oder „Vision go embedded“ (ViGEM) [92].

Die erfassten experimentellen Daten müssen im nächsten Schritt vom Versuchsfahrzeug zur weiteren Verarbeitung in ein Framework für Datenevaluierung (FDE) **importiert** werden. Aufgrund der hohen Anzahl der Sensoren und der daraus resultierenden großen Datenmengen, die autonome Fahrzeuge generieren, werden im operativen Betrieb Datencenter benötigt, um diese Daten zu speichern und zu verarbeiten [93, 94]. Entwicklungsbegleitend eignen sich auch Arbeitsstationen. Anwendungsabhängig können die Werkzeuge und Datenformate deshalb sehr unterschiedlich sein. Im Fall von Datencenter könnten beispielsweise Lösungen wie Hadoop [95] oder MapR [96] hergenommen werden. Entwicklungsbegleitend können SciPy [97] oder Matlab [98] ein Mittel der Wahl sein. Alle Lösungen mit deren Erweiterungsmöglichkeiten bieten Voraussetzungen, die erfassten Daten zu evaluieren.

Wie einleitend bereits erwähnt, ist der Evaluierungsprozess in zwei Iterationsschleifen aufgeteilt. Als Erstes wird versucht die Anzahl der Messdaten zu reduzieren. Dieser Schritt ist notwendig, da die Erfassung und Evaluierung hochdimensionaler Daten kaum möglich ist. Diese Thematik wird in den Kapiteln 3.4 und 3.5 näher erläutert. In der zweiten Evaluierungsschleife werden die Daten für den reduzierten Satz an Parametern erneut aufgezeichnet und importiert. Anschließend werden die eingefahrenen Szenarien **resimuliert**. Die Referenzdaten sind dabei in ein Szenariobeschreibungsformat zu überführen. Dafür eignet sich beispielsweise Open Scenario (OSC) [99]. OSC ist ein offenes Dateiformat zur Beschreibung dynamischer Szenarien für virtuelle Fahrversuche. Darauf aufbauend, kann ein Ground Truth Simulator (GTS) synthetische Referenzdaten generieren. VTD [100] und „CarMaker“ [101] sind dafür geeignete Kandidaten.

Für unterschiedliche Sensoren (Radar, Lidar, Kamera) und Modellierungsebenen (Rohdaten und Objektlisten) werden unterschiedliche Sensormodelle eingesetzt. Abhängig von dem zugrundeliegenden Anwendungsfall muss die Simulationsumgebung unterschiedliche Referenzen generieren: (a) 3D Umgebungsinformationen für die strahlenoptische Modellierung und (b) Objektlistenreferenz für die Objektlisten. Dabei ist wichtig, dass die verwendeten Sensormodelle standardisierte Datenformate verwenden [36]. Eine 3D–Umgebungsrepräsentation kann beispielsweise im „OpenSceneGraph“ Format [102]

und die Objekbeschreibung als Offene Simulation Schnittstelle (OSI) [103] vorliegen. OSI enthält eine objektbasierte Umgebungsbeschreibung mit Definition von Referenz OSI:GT, der Sensordaten (SD) OSI:SD und Sensor Rohdaten (RAW) OSI:RAW. Wobei OSI:GT sich auf die Referenzdaten in Weltkoordinaten, OSI:SD auf die Objektlisten in Sensorkoordinaten und OSI:RAW auf die Sensorrohdaten in Sensorkoordinaten beziehen. Für die Berechnung der ray–tracing–Modelle ist ein Framework für Strahlverfolgung (FSV) und für die Modellierung der Objektlisten eine Realzeit–Umgebung (RZU) notwendig. Für FSV existieren bereits Werkzeuge wie OptiX [104] oder Blender [105]. Für die RZU sind keine autarken Lösungen bekannt. Teilweise bieten die GTS–Hersteller integrierte Ansätze mit proprietären Schnittstellen. Der Ansatz mit den standardisierten Formaten hat den Vorteil, dass die Sensormodelle auf einem verteilten Rechnersystem lauffähig sind [25]. Da die Modelle rechenaufwändig sein können, ist es ein nicht unbedeutendes Argument, um die Anforderungen (siehe Kapitel 2.2.3) wie Skalierbarkeit und Gesamthaftigkeit zu erfüllen. Für RZU sind proprietäre Lösungen notwendig, wobei OSI Packaging [106] dafür eine Grundlage bietet. Bei der Entwicklung von RZU ist insbesondere auf das Timing–Konzept zu achten, das einen deterministischen Charakter aufweisen muss. Unter Determinismus ist dabei die Reproduzierbarkeit der Simulationsdurchläufe gemeint, wobei bei der Wahl beziehungsweise Konzeption der Werkzeuge im Rahmen der Validierung im Allgemeinen darauf zu achten ist.

Die simulierten Daten werden schließlich auf dieselbe Weise wie die realen importiert und gegeneinander **validiert**. Die theoretischen Aspekte der Validierung wurden in Kapitel 3.1 ausführlich behandelt und werden im weiteren Verlauf der Arbeit vertieft. In Kapitel 3.3 wird vor allem die Vorverarbeitung der erfassten Daten und in Kapiteln 4 und 5 die Analyse der geeigneten Metriken und darauf basierende Optimierung der verwendeten Sensormodelle näher untersucht. Abschließend ist anzumerken, dass aufgrund der großen Datenmengen der vorgestellte Ansatz von dem Import der Daten bis hin zu deren Evaluierung vollständig zu automatisieren ist.

3.3 Aufbereitung der Messdaten

Die zuvor erwähnten Messdatenreduktions– und Bewertungsschleifen (siehe Abbildung 3.2) haben drei Schritte in der Datenverarbeitungskette gemeinsam: (a) Datenerfassung, (b) Datenimport und (c) Datenaufbereitung. Die ersten beiden Schritte wurden bereits in Kapitel 3.2 erläutert. Nach dem Datenimport und vor der Auswahl der relevanten Parameter beziehungsweise Modellvalidierung müssen die erfassten Messdaten aufbereitet werden. Die einzelnen Schritte werden in diesem Kapitel vorgestellt und ähneln denen aus [36, Kapitel 3.6].

3.3.1 Vorverarbeitung

Als Erstes sind Sensorstörungen und irrelevante Objekte zu eliminieren. Dazu werden die aufgezeichneten Daten aller Sensoren erst in ein einheitliches Objektlistenformat überführt (siehe Kapitel 2.1.1). Anschließend folgen: (a) die Korrektur der Zeitstempelung und (b) Spurenbildung.

Die in den Fahrzeugen verbauten Sensoren verfügen meist nicht über eine zentrale Zeitgebungsinstanz, weshalb die **Zeitstempelung korrigiert** werden muss. Die Zeiten der Messpunktaufnahme werden dabei über den Empfangszeitpunkt am Hauptrechner approximiert. Diese Schätzung ist jedoch fehlerbehaftet. Zum einen nimmt die Signalverarbeitung des Sensors in Abhängigkeit zur Szenenkomplexität unterschiedliche Zeiten in Anspruch. Zum anderen führen unterschiedliche Lastzustände des Übertragungskanals zu Schwankungen der Übertragungszeiten der Datenpakete. Diese Schwankungen werden durch einen Tiefpassfilter geglättet.

Objektlisten $\mathcal{O}^{(t)}$ bilden die Umgebung zu einem bestimmten Zeitpunkt t ab (siehe Kapitel 2.1.1). Für die Evaluierung sind jedoch statt Momentaufnahmen zeitlich verfolgte Daten der gesehenen Objekte relevant, weshalb die Objektlisten $\mathcal{O}^{(t)}$ zunächst in **zeitlich verfolgte Spuren** T_i überführt werden müssen. Die Zuordnung der Objekte in $\mathcal{O}^{(t)}$ findet über eine Identifikationsnummer (ID) statt,

$$T_i = \left\{ o_k \in \mathcal{O}^{(\forall t)} | id_k = \text{const} \right\} \tag{3.2}$$

Eine Spur T_i enthält somit die Daten zu einem Objekt o_k aus den Objektlisten $\mathcal{O}^{(\forall t)}$, bei denen sich die ID id_k im gesamten Zeitraum der Aufzeichnung nicht ändert. Dabei wird angenommen, dass die IDs einzigartig sind und diese bei der Überführung aus der Objektliste in die Spur T_i in sequentieller Form $id_{\forall i}^{(\forall t)} \in \{1 : N_\tau\}$, mit $N_\tau \hat{=}$Anzahl der verfolgten Objekte, abgelegt werden. Für jede Aufnahme ergibt sich somit eine Liste an verfolgten Spuren,

$$\tau = \{T_i\}_{i=1:N_\tau} \tag{3.3}$$

3.3.2 Assoziation der Objekte

In diesem Kapitel wird auf die automatisierte Assoziation der erkannten Objekte eingegangen. Es gibt Situationen, in denen ein wahres Objekt mehrere Detektionen hervorruft. Der Assoziationsvorgang dient dazu, solche Detektionen wieder zu einem Objekt zu verknüpfen. Es gibt mehrere Gründe, weshalb diese Problematik auftreten kann. Ein umfelderfassender Sensor kann aufgrund der Unsicherheit im Objektverfolgungsalgoritmus am Rand des Sensorsichtbereichs zum Zeitpunkt, zu dem das Objekt zum ersten Mal erkannt wird, mehrere Objekte zu einem einzigen korrespondierenden wahren Objekt generieren (siehe Abbildung 3.3a). Eine weitere Ursache ist der Wechsel der Objekt–ID im Laufe der Aufzeichnung (siehe Abbildung 3.3b). Wie im vorherigen Unterkapitel jedoch bereits erwähnt, wird eine eindeutige ID vorausgesetzt. Des Weiteren wird die Fahrzeugumgebung redundant mittels mehrerer Sensoren erfasst. Jeder

korrespondierende Sensor generiert entsprechend eigene Objektlisten (siehe Abbildung 3.3c). Diese müssen ebenfalls zueinander assoziiert werden.

Die Assoziation zweier Objekte basiert auf deren Ähnlichkeit. Um diese ermitteln zu können, ist es vorteilhaft wenn die Daten unterschiedlicher **Sensoren zeitlich synchronisiert** sind. Dies ist jedoch normalerweise aufgrund unterschiedlicher Aktualisierungsraten umfelderfassender Sensoren nicht gegeben. Aus diesem Grund werden die Sensoren mit einer höheren Abtastrate auf die Messzeitpunkte der niederfrequenten Sensoren interpoliert, wodurch im Mittel ein geringerer Interpolierungsfehler zu erwarten ist.

Das **Assoziationsverfahren** setzt voraus, dass die verfolgten Spuren $\tau = \{T_i\}_{i=1:N_\tau}$ bereits vorliegen (siehe Gleichung 3.3) und gibt eine Menge an Assoziationen $\Gamma = \{A_i\}_{i=1:N_\Gamma}$ mit $A_i = \{T_k\}_{k=1:N_{A_i}}$ zurück. Das Verfahren beginnt dann mit dem Einfügen der ersten Spur T_1 in eine noch leere Assoziation A_1. Anschließend wird eine weitere Spur T_k zu der selben Assoziation hinzugefügt und auf Ähnlichkeit mit den bereits in der Menge A_i vorhandenen überprüft. Sofern diese Prüfung für alle in der Assoziation befindlichen Spuren erfolgreich ist, wird der Kandidat belassen, andererseits entfernt. Anzumerken ist, dass beim erfolgreichen Hinzufügen einer Spur T_k zu einer Assoziation A_i alle zuvor abgelehnten Spuren wiederholt überprüft werden müssen. Sobald keine weiteren Spuren T_k zur Assoziation A_i hinzugefügt werden können, wird eine weitere Assoziation A_{i+1} geöffnet und der Vorgang für alle noch verbleibenden Spuren aus τ wiederholt.

Bei dem vorgestellten Ansatz handelt es sich um einen gierigen Algorithmus (englisch: Greedy Algorithm) [107]. Dabei ist das Ergebnis $\Gamma = \{A_i\}_{i=1:N_\Gamma}$ von der Reihenfolge, mit der die Spuren verarbeitet werden, abhängig. Durch eine Vorsortierung

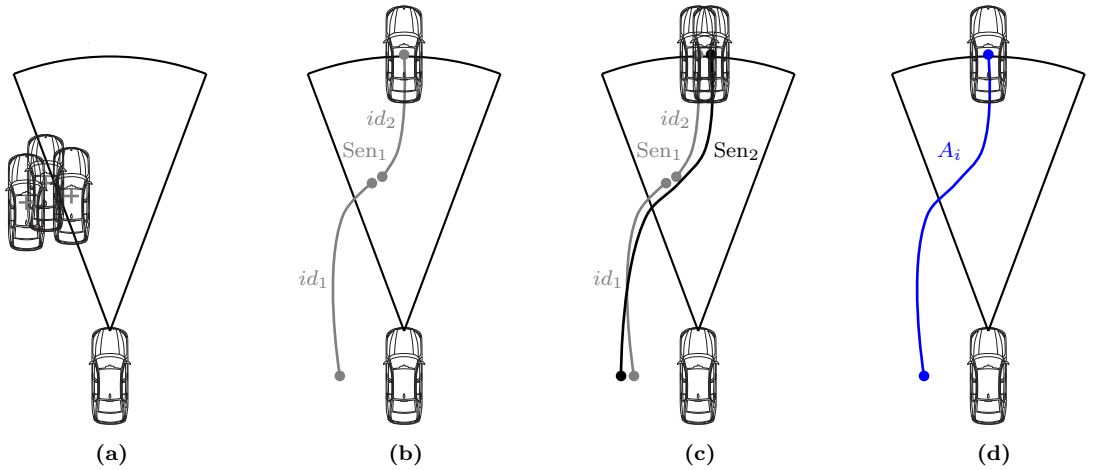


Abbildung 3.3: Visualisierung des Assoziationsproblems: (a) Eigenfahrzeug wird überholt und Sensor erkennt mehrere Objekte am Rand des Sichtbereichs, (b) Eigenfahrzeug wird überholt und ID ändert sich im Laufe der Messaufzeichnung, (c) ein wahres Objekt aus der Sicht mehrerer Sensoren, (d) assoziiertes Objekt.

nach der Länge der Spuren kann dieser Nachteil jedoch größtenteils eliminiert werden. Des Weiteren ist zu beachten, dass die Komplexität des Verfahrens $\mathsf{O}(N_\tau^2)$ entspricht. Es empfiehlt sich daher die Messaufnahmen nicht länger als zwei Minuten zu halten [36].

Die Assoziation mehrerer Sensoren erfolgt durch sequentielles Hinzufügen von Spuren unterschiedlicher Sensoren zu einer Assoziation. Es wird zunächst eine vereinte Menge $\tau = \tau_1 \cup \tau_2$ der Sensoren 1 und 2 ausgewertet. Das Ergebnis ist eine Menge an Assoziationen Γ, aus der eine fusionierte Spurenliste τ_{fus} gebildet wird. Diese enthält deutlich weniger Objekte als die ursprüngliche Liste τ. Der Vorgang wiederholt sich dann für den nächsten Sensor, indem $\tau' = \tau_{\text{fus}} \cup \tau_3$ ausgewertet wird. Eine Alternative zu dem vorgestellten Ansatz wäre die Vereinigung aller Sensoren zu einer Menge $\tau = \tau_1 \cup \tau_2 \cup ...$, wodurch die Rechenzeit jedoch stark ansteigen würde.

Wie zuvor bereits erwähnt, erfolgt die Verknüpfung der Spuren zu Assoziationen basierend auf einer **Ähnlichkeitsprüfung**. Um eine fusionierte Spur V_k' zu erhalten, wird als Erstes in jedem Zeitschritt aus allen in der Assoziation befindlichen Objekten dasjenige mit der größten Grundfläche, berechnet aus Objektlänge und –breite, gewählt. Dabei liegt die Annahme zugrunde, dass mit hoher Wahrscheinlichkeit das Objekt mit der größten Fläche auch die meisten beziehungsweise intensivsten Detektionen hervorruft und somit kein Rauschobjekt ist. Des Weiteren werden zur Bewertung der Ähnlichkeit zwischen sich bereits in einer Assoziation A_i befindlichen Spuren $T_1 ... T_{k-1}$ und des aktuellen Kandidaten T_k folgende Kriterien verwendet: (a) zeitliche Nähe, (b) örtliche Nähe in jedem Zeitschritt und (c) Ähnlichkeit der Geschwindigkeit in jedem Zeitschritt. Die Bewertung erfolgt dabei basierend auf Schwellwerten in Bezug auf Quantile, Minimal–, Mittel– und Maximalwerte. Von Vorteil ist dabei die gesamtheitliche Betrachtung der Messaufzeichnung, die zu deutlich robusteren Ergebnissen als eine Realzeitauswertung führt. Abschließend wird die Fahrbarkeit der abgeleiteten Spur in Bezug auf die physikalischen Eigenschaften, wie zum Beispiel die Beschleunigung, getestet. Sobald alle Überprüfungen erfolgreich waren, wird die Assoziation A_i akzeptiert.

Nachdem alle Assoziierungsschritte durchlaufen sind, liegt zu jedem Objekt von jedem einzelnen Sensor sowie deren Fusion jeweils eine Assoziation A_i vor. Schematisch ist eine solche Assoziation in Abbildung 3.3d gezeigt.

3.4 Verfahren zur Reduktion der Anzahl von Modellierungsgrößen

In Abbildung 3.2 werden wesentliche Schritte zur Validierung umfelderfassender Sensormodelle gezeigt. Das hier diskutierte Verfahren ist ein Teil der ersten Evaluierungsschleife, die folgendermaßen abläuft: (a) Messdatenaufzeichnung, (b) Messdatenimport, (c) Messdatenaufbereitung und (d) Messdatenreduktion. (a) und (b) wurden in Kapitel

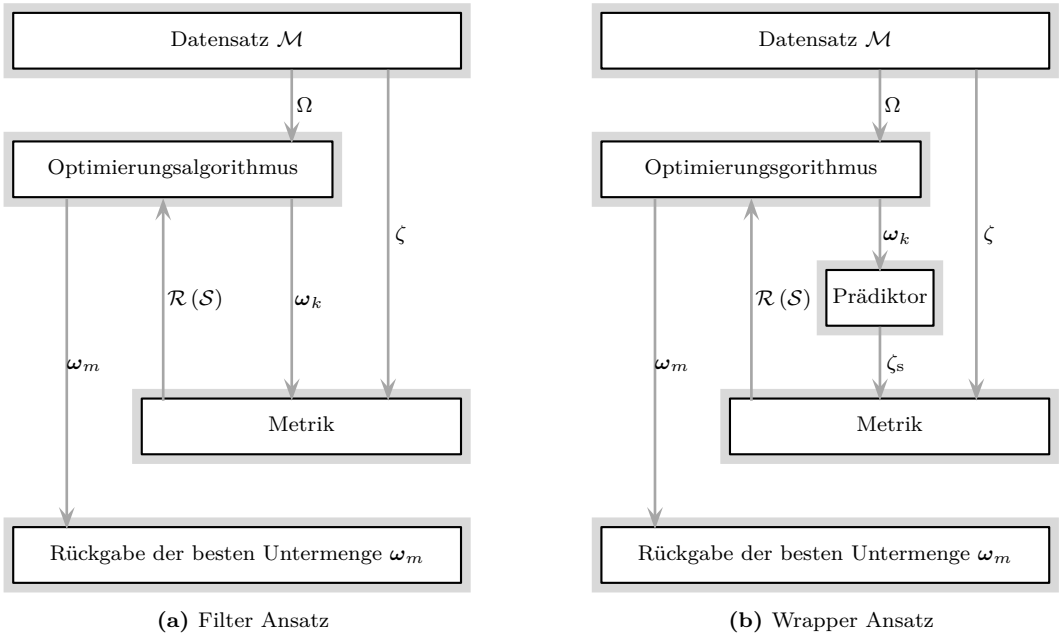
3.2 und (c) in Kapitel 3.3 bereits erläutert. In diesem Unterkapitel wird der Punkt der Parameterauswahl im Rahmen der Messdatenreduktion (d) diskutiert.

Die „richtige“ Wahl von Modellierungsparametern beziehungsweise Modellierungsgrößen Ω und Zielgrößen ζ (siehe Abbildung 2.1) ist eine grundlegende Fragestellung bei der Modellierung physikalischer Vorgänge. Die Zielgrößen ζ werden meist anwendungsspezifisch definiert. Im Fall der umfelderfassenden Sensoren sind es beispielsweise unter anderem die Eingangsgrößen von Fusionsalgorithmen wie Abstand, Geschwindigkeit oder Richtung beziehungsweise deren Fehlerverteilungen. Eine eher offene Frage ist, welche Modellierungsgrößen Ω zu wählen sind, um das Verhalten von ζ möglichst realitätsnah abbilden zu können. Die naive Vorstellung, alle physikalisch messbaren Größen zu berücksichtigen, um ein möglichst genaues Ergebnis zu erzielen, ist in diesem Fall nicht anwendbar. Der Grund hierfür ist der Fluch der Dimensionalität, der in Kapitel 2.1 bereits adressiert wurde. Dieser erschwert nicht nur die Verarbeitung hochdimensionaler Daten, sondern auch deren Aufzeichnung. Mit jeder Dimension steigt die Anzahl der zu erfassenden Datenpunkte exponentiell an und macht die Datenaufzeichnung für die Bedatung von Sensormodellen zu einem „teuren“ Prozess. Des Weiteren erschwert die hohe Anzahl von Modellierungsgrößen die Modellinterpretation und kann zu einer Überanpassung führen. Zusammenfassend ist das Ziel dieses Unterkapitels Verfahren zu identifizieren, welche es ermöglichen, eine Teilmenge an relevanten Modellierungsgrößen eines Sensormodells aus potenziell möglichen zu priorisieren und die Modellgüte zu optimieren. Dabei ist eine Modellierungsgröße relevant, falls es bekannt ist, dass die Zielgröße ζ bedingt von ω_i abhängig ist $p(\zeta|\omega_i) \neq p(\zeta)$ oder, mit anderen Worten, wenn die Modellierungsgröße ω_i die Zielgröße ζ beeinflusst [108].

3.4.1 Ansätze zur Variablenauswahl

Eine Disziplin, die sich mit dieser Fragestellung beschäftigt, heißt Variablen– beziehungsweise Merkmalauswahl (englisch: Variable beziehungsweise Feature Selection) [109]. Dabei bezieht sich in dieser Arbeit die Variablenauswahl auf die Auswahl physikalisch messbarer und die Merkmalauswahl auf die Auswahl konstruierter Größen, die eine Zielgröße ζ beeinflussen. Möchte man zum Beispiel den Fehler δ_{x} in x–Richtung modellieren, der von der relativen Geschwindigkeit v_{x} abhängt, so wäre δ_{x} eine Zielgrösse und v_x eine beeinflussende Variable. Ein Merkmal hingegen ist eine nicht direkt messbare, sondern eine konstruierte Größe, wie zum Beispiel v_{x}^2. Im Folgenden werden die Begriffe Variablen–, Parameter– und Modellierungsgrößenauswahl austauschbar verwendet.

In der Literatur werden drei Ansätze zur Auswahl von Variablen unterschieden: (a) Filter–, (b) Wrapper– und (c) Embedded–Methoden [110,111]. Jeder der erwähnten Ansätze stellt eine Gruppe von Verfahren zur Bewertung der Relevanz von Modellierungsgrößen dar. Filteransätze entscheiden über die Relevanz von Variablen, basierend auf der Korrelation zwischen Modellierungs– und Zielgrößen, ohne dabei einen Prädiktor in die Evaluierungsschleife einzubeziehen. Wrapper–Ansätze beurteilen die Relevanz von Variablen, indem ein Modell als eine Black–Box ausgewertet wird. Dabei werden die Modellierungsgrößen dem Modell bereitgestellt und damit die modellierten Zielgrößen berechnet. Im Gegensatz zu einem Filteransatz basiert die Relevanzevaluierung



(a) Filter Ansatz **(b)** Wrapper Ansatz

Abbildung 3.4: Visualisierung der Filter– und Wrappe–Ansätze zur Auswahl von relevanten Modellierungsgrößen: (a) Der Filteransatz basiert auf dem Vergleich von Modellierungs– und Zielgrößen, (b) Der Wrapper–Ansatz basiert auf dem Vergleich von modellierter und realer Zielgröße unter Nutzung des zugrundeliegenden Modells als Black–Box.

nicht auf dem direktem Vergleich von Einfluss– und Zielgrößen, sondern auf dem Vergleich von modellierter Zielgröße ζ_s und gemessener Zielgröße ζ. Embedded Ansätze evaluieren die Relevanz von Modellierungsgrößen hingegen direkt im Modell. Mit anderen Worten, unterscheiden Embedded–Methoden nicht zwischen der Lern– und der Parameterauswahl–Phase, sodass die Parameter bereits während der Lernphase ausgewählt werden. Die Parameterauswahl bei Embedded–Ansatz ist daher spezifisch für die jeweilige Methode. Die Priorisierung von Parametern in einem neuronalen Netz ist beispielsweise anders als bei Entscheidungsbäumen. In einem neuronalen Netz basiert diese auf einer vordefinierten Aktivierungsfunktion und in einem Entscheidungsbaum auf einem Schwellwert. Aus der Sicht der Validierung eines Sensormodells kommen deshalb nur Methoden aus den ersten beiden Gruppen (Filter und Wrapper) in Frage, denn die zu evaluierenden Modelle liegen nicht immer als White–Boxen vor, was die Embedded–Methoden voraussetzen. Filter– und Wrapper–Ansätze sind in Abbildung 3.4 schematisch dargestellt und werden im Folgenden näher erläutert.

Wie bereits erwähnt, bewerten die **Filter–Methoden** die Relevanz von Variablen direkt aus den zugrundeliegenden Messdaten, ohne dabei ein Modell einzubeziehen. Der Vorgang ist schematisch in Abbildung 3.4a dargestellt. Ein Filter in diesem Kontext ist eine Funktion $\mathcal{F}(\mathcal{M}) \longmapsto \boldsymbol{\omega}_m$, die für ein vorliegenden Datensatz $\mathcal{M}$ einen Subset $\boldsymbol{\omega}_m \in \Omega$ von relevanten Modellierungsgrößen zurück gibt. Die Auswahl der besten Untermenge findet dabei mit Hilfe eines Optimierungsalgorithmus statt. Hierbei wird mit Hilfe einer Metrik der jeweilige Relevanzindex des aktuellen Subsets be-

rechnet. Eine Metrik ist dabei eine Funktion $\mathcal{F}(\boldsymbol{\omega}_k|\mathcal{M}) \longmapsto \mathcal{R}(\boldsymbol{\omega}_k)$, die eine Schätzung $\mathcal{R}(\boldsymbol{\omega}_k)$ der Relevanz für ein Subset $\boldsymbol{\omega}_k \in \Omega$ zurück gibt. Die Annahme ist dabei, dass sich bei der Auswertung die vorliegenden Messdaten nicht ändern und nur $\boldsymbol{\omega}_k$ variiert wird. Dieses Vorgehen wiederholt sich so lange, bis der Optimierungsalgorithmus den bestmöglichen Subset $\boldsymbol{\omega}_m \in \Omega$ an Modellierungsgrößen gefunden hat. Wie die Suche innerhalb des Optimierungsalgorithmus erfolgt, wird im weiteren Verlauf dieses Kapitels erläutert. Die Entscheidung über den besten Subset basiert dabei auf den, von der Metrik zurückgelieferten Relevanzindizes. Diese werden sortiert $\mathcal{R}(\boldsymbol{\omega}_{k_1}) \leq \mathcal{R}(\boldsymbol{\omega}_{k_2}) \cdots \leq \mathcal{R}(\boldsymbol{\omega}_{k_N}) = \mathcal{R}(\boldsymbol{\omega}_m)$ und schließlich die Untermenge $\boldsymbol{\omega}_m$ mit dem höchsten Relevanzindex für die Modellbildung eingesetzt.

Die Bewertung basiert dabei auf der Berechnung von Metriken, wobei sich diese anwendungsspezifisch unterscheiden können. Grob lassen sich diese Metriken in folgende Kategorien einteilen [110, Kapitel 3]: (a) Bayes Statistik, (b) Korrelationsbasierte Metriken, (c) Distanz zwischen Verteilungen, (d) Messung basierend auf der Informations–Theorie und (e) Entscheidungsbäume. Die Daten umfelderfassender Sensoren sind kontinuierlicher Natur, sodass laut [110, Tabelle 3.1 auf S. 112] nur Metriken aus den Kategorien (b)–(d) in Betracht kommen. Im Rahmen dieser Arbeit werden daraus jedoch nur die korrelationsbasierten und auf der Informations–Theorie basierten Metriken verwendet, da sich diese gut beim Messen linearer und nicht–linearer Abhängigkeiten ergänzen (siehe [112] und [110, Kapitel 3]).

Der lineare Korrelationskoeffizient von Pearson ist in der Statistik weit verbreitet und ist wie folgt definiert,

$$M_{\mathrm{P}}(\boldsymbol{\omega}, \boldsymbol{\zeta}) = \frac{\sum_{i=1}^{m}(\omega_i - \bar{\omega})(\zeta_i - \bar{\zeta})}{\sqrt{\sum_{i=1}^{m}(\omega_i - \bar{\omega})^2 \sum_{i=1}^{m}(\zeta_i - \bar{\zeta})^2}} \tag{3.4}$$

wobei M_{P} den Korrelationskoeffizienten nach Pearson, $\boldsymbol{\omega}$ und $\boldsymbol{\zeta}$ die Vektoren mit Messdaten von Modellierungs– beziehungsweise Zielgrößen, m die Größe des Datensatzes, ω_i und ζ_i die Datenpunkte von Modellierungs– beziehungsweise Zielgrößen und $\bar{\omega} = \frac{1}{m}\sum_{i=1}^{m}\omega_i$ den Mittelwert der Modellierungsgröße ω_i (entsprechendes für $\bar{\zeta}$) bezeichnen. $M_{\mathrm{p}}(\boldsymbol{\omega}, \boldsymbol{\zeta})$ ist gleich ± 1 wenn ω und ζ linear abhängig sind und 0, wenn diese absolut unkorreliert sind. Solche Koeffizienten funktionieren gut, sobald die Relation zwischen Modellierungsgrößen und Zielgrößen linear ist.

Um nichtlineare Zusammenhänge zu erfassen, ist die aus der Informations–Theorie stammende Metrik der gegenseitigen Information (englisch: Mutual Information) eine oft verwendete Alternative,

$$M_{\mathrm{I}}(\boldsymbol{\omega}, \boldsymbol{\zeta}) = \sum_{i=1}^{m}\sum_{j=1}^{n} p(\omega_i, \zeta_j) \log \frac{p(\omega_i, \zeta_j)}{p(\omega_i)p(\zeta_j)} \tag{3.5}$$

wobei $p(\omega_l, \zeta_m)$ die Verbundwahrscheinlichkeit (englisch: Joint Probability) von ω_i und ζ_j und $p(\omega_i)$ beziehungsweise $p(\zeta_j)$ die Marginalverteilungen von ω_i und ζ_j sind. Dabei ist zu erwähnen, dass sich diese Definition auf kategorische Größen bezieht. Im Fall von kontinuierlichen Daten, wie es bei umfelderfassenden Sensoren meist der Fall ist, sind

in Gleichung 3.5 die Summen durch Integrale und die Berechnung von Verbundwahrscheinlichkeiten durch WDF zu ersetzen.

Die oben definierten Metriken können auch zur Schätzung des Einflusses einer Gruppe von k Modellierungsgrößen $\boldsymbol{\omega}_k$ auf eine Zielgröße ζ eingesetzt werden. Nach [113, S. 231ff] wächst die Relevanz einer Gruppe von Merkmalen mit steigender Abhängigkeit zwischen der Gruppe und der Zielgröße und sinkt mit wachsender Abhängigkeit zwischen der Gruppe und den Modellierungsgrößen. Sie ist wie folgt definiert,

$$M_{\mathrm{G}}(\boldsymbol{\omega}_k, \zeta) = \frac{\sqrt{k} \cdot M_{\omega\zeta}}{\sqrt{1 + (k-1) \cdot M_{\omega\omega}}} \tag{3.6}$$

wobei k die Anzahl der Modellierungsgrößen in der Gruppe und $M_{\omega\zeta} = \bar{M}(\boldsymbol{\omega}_k, \zeta) = \frac{1}{k}\sum_{i=1}^{k} M_{\mathrm{P}}(\boldsymbol{\omega}_i, \zeta)$ die mittlere Korrelation zwischen einzelnen Modellierungsgrößen und der Zielgröße und $M_{\omega\omega} = \bar{M}(\boldsymbol{\omega}_k, \boldsymbol{\omega}_k) = \frac{1}{k^2}\sum_{i=1}^{k}\sum_{j=1}^{k} M_{\mathrm{P}}(\boldsymbol{\omega}_i, \boldsymbol{\omega}_j)$ die mittlere Korrelation zwischen den einzelnen Modellierungsgrößen bezeichnen. Die Auswahl des Subsets an Modellierungsgrößen kann mit Hilfe unterschiedlicher Suchalgorithmen erfolgen, die ein Teil des Optimierungsalgorithmus (siehe Abbildung 3.4a) sind und im weiteren Verlauf dieses Kapitels vorgestellt werden.

Wrapper–Methoden beziehen im Gegensatz zur Filtermethode auch das zu validierende Modell in die Evaluierungsschleife mit ein (siehe Abbildung 3.4b). Der Ausgangspunkt hier ist wieder der aufgezeichnete Messschrieb, der in Modellierungs– und Zielgrößen getrennt wird. Ähnlich wie bei der Filtermethode, wählt ein Optimierungsalgorithmus ein Subset an Modellierungsgrößen aus und stellt diese dem zugrunde liegenden Black–Box–Modell zur Verfügung. Als nächstes wird der verwendete Prädiktor an den Parameterset angelernt und die Zielgröße ζ_S prädiziert. Da das Modell an jedes Parametersubset angelernt werden muss, liegt die Auswertungszeit der Wrapper–Methoden deutlich über der von Filtern. Des Weiteren tendieren die Wrapper im Gegensatz zu den Filtern die Allgemeingültigkeit der Aussagen zu verlieren, da diese sich an den Datensatz anpassen. Abschließend werden die Prädikationsergebnisse mit der aufgezeichneten Zielgröße ζ verglichen. Die Bewertung der Ähnlichkeit zwischen ζ_S und ζ basiert wiederum auf einer Metrik. Bei kontinuierlichen Größen wird meist die mittlere quadratische Abweichung verwendet. Die Auswahl der besten Untermenge $\boldsymbol{\omega}_m$ findet, ähnlich wie bei der Filtermethode, im Optimierungsalgorithmus statt.

Der **Optimierungsalgorithmus** im Kontext von Filter– und Wrapper–Methoden hat im Wesentlichen zwei Aufgaben: (a) Auswahl der Untermengen an Modellierungsgrößen, die zur Bewertung der Relevanz hergenommen werden und (b) Bestimmung des optimalen Subsets. Dabei ist die Bestimmung des optimalen Subsets trivial, da sie rein auf der Rangreihenfolge basiert. Die Auswahl von Untermengen ist dagegen eine deutlich herausforderndere Aufgabe, die auf unterschiedlichen Suchstrategien basieren kann. Diese werden im Folgenden vorgestellt.

Eine Suchstrategie definiert die Reihenfolge, in welcher die Subsets von Modellierungsgrößen evaluiert werden. Eine Methode, die sicher zur optimalen Lösung führt,

heißt *vollständige Suche* (englisch: Exhaustive Search) beziehungsweise die *Methode der rohen Gewalt* (englisch: Brute–force search). Ein großer Nachteil dieser Methode ist die hohe Anzahl ($2^n - 1$ mit n Gesamtanzahl der Modellierungsgrößen) der möglichen Kombinationen von Subsets, für die der Relevanzindex berechnet werden muss. Leider wird es sogar für eine überschaubare Anzahl an Variablen in praktischen Anwendungen unmöglich sein, alle Subsets zu evaluieren [110].

Aufgrund der Untauglichkeit dieser Suchmethode für praktische Anwendungen wurden andere, „suboptimale“, Suchstrategien entworfen. Eine solche Kategorie stellen die *sequentiellen Suchmethoden* dar. Die erste Methode aus dieser Kategorie ist die sequentielle Rückwärts–Selektion (englisch: Sequential Backward Selection) (SRS), die in [114] vorgestellt wird. Sie startet mit einem Set von Modellierungsgrößen Ω, das alle möglichen Kandidaten beinhaltet. Sequentiell wird jeweils eine Modellierungsgöße aus Ω entfernt und die Relevanz des verbleibenden Subsets mit $\mathcal{R}(\boldsymbol{\omega}_k)$ gemessen. In jeder Iteration wird dabei die Größe endgültig aus dem Subset entfernt, deren Entfernung die Relevanz am wenigsten beeinflusst. Das Vorgehen wird so lange wiederholt, bis die gewünschte Anzahl an Variablen erreicht wird oder die Relevanz zu gering wird. Das gegenteilige Vorgehen wird in der Methode sequentielle Vorwärts–Selektion (englisch: Sequential Forward Selection) (SVS) angewendet [115]. Sie startet mit einem leeren Set und fährt fort mit dem Hinzufügen von Größen in das Subset. Pro Iteration wird diejenige Größe hinzugefügt, die den größten Einfluss auf die Relevanz hat. Das Vorgehen wird so lange wiederholt, bis die gewünschte Anzahl von Größen erreicht wird oder das Ergebnis sich nicht mehr merklich verbessert. Die beschriebenen Suchmethoden fassen die Größen, die einmal in das Subset aufgenommen beziehungsweise daraus entfernt wurden, nicht mehr an. Diese Tatsache wird in der Literatur als Verschachtelungseffekt (englisch: Nesting Effect) bezeichnet. Nachteilig ist, dass Fehlentscheidungen, die zu Beginn der Suche getroffen wurden, später nicht mehr korrigiert werden können, da die aufeinanderfolgenden Variablensätze immer so verschachtelt sind, dass ein Satz eine richtige Untermenge des anderen ist. Um diesem entgegenzuwirken, wird die Rückverfolgung während der Suche eingeführt. Diese Suchmethode heißt sequentielle gleitende Vorwärtselektion (englisch: Sequential Floating Forward Selection) (SFVS). Dazu wird die Suche in zwei verschiedene und abwechselnde Phasen unterteilt: Erst ein Schritt von SVS und anschließend ein Schritt von SRS. Die Modellierungsgrößen werden abwechselnd aus dem Subset entfernt und wieder neue hinzugefügt, so lange der Wert von $\mathcal{R}(\boldsymbol{\omega}_k)$ steigt. Ähnlich ist die Methode sequentielle gleitende Rückwärts–Selektion (englisch: Sequential Floating Backward Selection) (SFRS) aufgebaut, wobei die SVS– und SRS–Phasen umgekehrt ausgeführt werden. Neben den hier vorgestellten Ansätzen gibt es eine Reihe an weiteren Methoden (siehe [110, Kapitel 4.4]), die im Rahmen dieser Arbeit nicht weiter berücksichtigt werden.

Die bis jetzt beschriebenen Suchmethoden sind deterministischer Natur, das heißt unter der Annahme einer gewissen Initialisierungsphase werden diese immer die selben Ergebnisse liefern. Problematisch dabei ist, dass sich das zurückgelieferte Subset an den Datensatz $\mathcal{M}$ anpassen kann. Diese Problematik wird in *stochastischen Suchmethoden* adressiert. In [116] wird eine Methode vorgeschlagen, die an SRS angelehnt ist. Für jede Modellierungsgröße wird dabei eine Relevanz berechnet und zufällig ein

Subset an k irrelevanten Größen entfernt. Sollte sich daraufhin der Wert von $\mathcal{R}(\boldsymbol{\omega}_k)$ deutlich reduzieren, wird angenommen, dass eine oder mehrere als irrelevant angenommene Größen tatsächlich relevant sind und die Eliminierung des Subsets abgebrochen wird. Dies kann so lange wiederholt werden, bis zu viele aufeinanderfolgenden Versuche fehlschlagen, was als Zeichen dafür gewertet wird, dass die verbleibenden Kandidaten für die Vorhersage tatsächlich relevant sind.

Abschließend werden die vorgestellten Methoden und Suchalgorithmen miteinander verglichen und in Tabelle 3.2 zusammengefasst. Die Filter sind meist schnell in Ausführung, da keine Prädiktoren bei jeder Iteration neu gelernt werden müssen. Zudem haben deren Ergebnisse höhere Allgemeingültigkeit, da intrinsische Eigenschaften der Daten ausgewertet werden, die tendenziell für eine Großzahl von Prädiktoren gültig sind. Nachteilig ist hingegen, dass die Subsetgröße meist höher im Vergleich zu anderen Methoden ausfällt, da die Filterfunktionen monoton sind und dazu neigen, alle Modellierungsgrößen als optimale Lösung auszuwählen. Wrapper sind üblicherweise genauer als Filter, da sie auf spezifische Wechselwirkungen zwischen Prädiktor und Datensatz abgestimmt sind. Nachteilig ist die langsame Ausführung, da die Prädiktoren bei jeder Iteration neu gelernt werden müssen und der Mangel an Allgemeingültigkeit, da das finale Subset von Modellierungsgrößen spezifisch für den verwendeten Prädiktor ist [110].

Bei den Suchalgorithmen ist die Vorwärtsselektion SVS schneller als die Rückwärtsselektion SRS, da am Anfang der Suche nur kleine Subsets ausgewertet werden. Dies ist üblicherweise schneller als das volle Subset auszuwerten. Andererseits wertet SVS die Variablen nur im Zusammenhang mit den Variablen aus, die bereits im Subset enthalten sind. So kann es sein, dass es nicht in der Lage ist, eine Variable zu erkennen, die nicht von selbst, sondern nur mit den Informationen, die von einigen anderen Varia-

Tabelle 3.2: Vor– und Nachteile von Filter– und Wrapper–Methoden und der in Frage kommenden Suchalgorithmen im Kontext der Variablenauswahl.

		Vorteile	Nachteile
Methoden	Filter	+ Schnelle Ausführung + Allgemeingültigkeit	– Größere Subsetauswahl
	Wrapper	+ Genauigkeit	– Langsame Ausführung – Mangelnde Allgemeingültigkeit
Suchalgorithmen	SVS	+ Schnelle Ausführung	– Wichtigkeit einzelner Variablen kann übersehen werden – Variablen werden nur einmal angefasst
	SRS	+ Mittelschnelle Ausführungszeit + Üblicherweise bessere Performanz als SVS	– Tendenz zur größeren Subsetauswahl – Variablen werden nur einmal angefasst
	SFVS	+ Kein Schachtelungseffekt + Gute Performanz	– Langsame Ausführung

blen getragen werden, von Vorteil ist. Im Allgemeinen erreicht SRS tendenziell bessere Performanz als SVS, jedoch zum Preis größerer Subsets. Beide Suchalgorithmen, SVS und SRS, leiden zudem unter dem Schachtelungseffekt (englisch: nesting effect), was bedeutet, dass schlechte Entscheidungen, die anfangs getroffen wurden, später nicht mehr korrigiert werden können. Diese Problematik kann zum Preis höherer Laufzeiten mit SFVS gelöst werden.

Abschließend ist zu erwähnen, dass nicht die selben Messdaten für die finale Bewertung des Schätzers herangezogen werden dürfen, die auch während der Suche verwendet wurden, da dies wegen der Überanpassung zu besseren Ergebnissen führen könnte, als es tatsächlich der Fall ist [117].

3.4.2 Umsetzung und Auswertung des Verfahrens

Basierend auf den Erkenntnissen aus dem vorherigen Unterkapitel, soll hier ein Verfahren zur Reduktion von Parametern zur Modellierung umfelderfassender Sensoren abgeleitet werden. Basierend auf den Vor– und Nachteilen der vorgestellten Filter– und Wrapper–Methoden sowie der in Frage kommenden Suchalgorithmen aus Tabelle 3.2, wird ein hybrider Ansatz vorgeschlagen. Unter der Annahme, dass die vorliegenden Daten bereits vorverarbeitet und assoziiert wurden (siehe Kapitel 3.3), sind folgende Schritte durchzuführen: (a) Standardisierung, (b) Vorauswahl der Parameter, (c) Feinanpassung. Die einzelnen Punkte sind in Abbildung 3.5 visualisiert und werden im

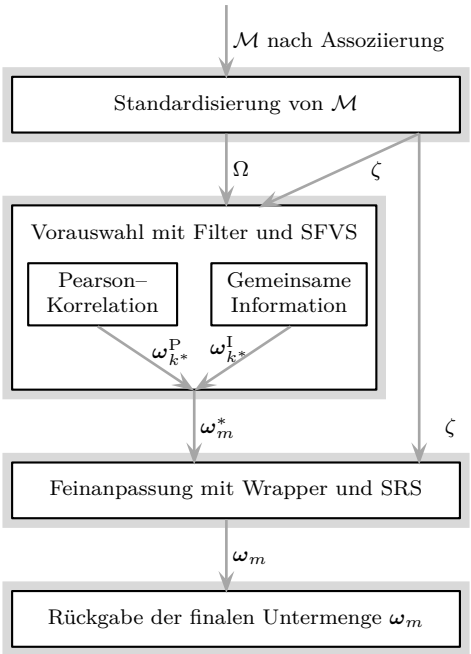


Abbildung 3.5: Visualisierung der Methode zur Auswahl von Modellierungsgrößen basierend auf einem hybriden Vorgehen, bestehend aus einem Filter und einem Wrapper.

Folgenden näher erläutert.

Bevor die Parametersuche beginnen kann, sollten die vorliegenden Daten **standardisiert** werden. Eine Standardisierung der Daten ist notwendig, da die aufgezeichneten Größen unterschiedliche Skalen beziehungsweise Einheiten haben können, obwohl sich diese auf dieselben Objekte beziehen. Klassisch werden die Daten mit $\omega_i' = \frac{(\omega_i - \mu_i)}{\sigma_i}$ standardisiert, wobei μ_i der Mittelwert und σ_i die Standardabweichung des Parameters ω_i sind [118]. Vor der Standardisierung ist zusätzlich darauf zu achten die Ausreißer aus den Daten zu entfernen, da diese die Standardisierungergebnisse beeinflussen können.

Nach der Standardisierung kann die Suche nach relevanten Parametern beginnen. Wie oben erwähnt, haben Filter die Tendenz zur Allgemeingültigkeit, wohingegen die Wrapper den Fokus auf einen bestimmten Prädiktor legen. In den meisten Fällen führt es dazu, dass die Subsets von Filtern größer als die von Wrappern sind. Es wird deshalb vorgeschlagen einen **Filter zur Vorauswahl der Parameter** zu verwenden und im Anschluss einen Wrapper zur weiteren Einschränkung der relevanten Größen einzusetzen (ähnlich zu [119]). Als Suchalgorithmus wird dabei SFVS eingesetzt [120, 121]. Erfahrungsgemäß ist dieser sehr effektiv, ein Subset einer vorgegebenen Größe mit größtem Ranking zu finden [122, 123]. Obwohl SFVS rechenintensiver als SVS und SRS ist, wird dieser als akzeptabel für die Vorauswahl angesehen, da hier nur ein recheneffizienter Filter zum Einsatz kommt. Als Erstes werden aus Ω für alle angefassten $\boldsymbol{\omega}_{k_i}$ mit Subsetlängen $k \in 2 \ldots N$ die mit $\mathcal{R}(\boldsymbol{\omega}_k) = \mathsf{max}(\mathcal{R}(\boldsymbol{\omega}_{k_i}))$ ausgewählt und anschließend daraus das Subset der Länge k^* mit dem größten Relevanzindex gezogen. Als Relevanzmetrik wird dabei die Gruppenkorrelation aus Gleichung 3.6, basierend auf der absoluten Pearson–Korrelation aus Gleichung 3.4 und der gegenseitigen Information aus Gleichung 3.5, verwendet. Die relevanten Subsets $\boldsymbol{\omega}^{\mathrm{P}}_{k^*}$ und $\boldsymbol{\omega}^{\mathrm{I}}_{k^*}$ werden abschließend zu einem $\boldsymbol{\omega}^*_m = \boldsymbol{\omega}^{\mathrm{P}}_{k^*} \cup \boldsymbol{\omega}^{\mathrm{I}}_{k^*}$ vereint und dem Wrapper zur Feinanpassung zur Verfügung gestellt. Der Ablauf ist schematisch in Abbildung 3.5 dargestellt.

Zur weiteren Einschränkung der relevanten Parameter wird ein **Wrapper zur Feinanpassung** vorgeschlagen. Da bei Wrappern der Prädiktor während der Suche bei jeder Iteration neu angelernt werden muss, kommen aufgrund der geringeren Rechenzeit nur SVS und SRS in Frage. Die Rückwärtsselektion erreicht üblicherweise bessere Performanz und wird deshalb trotz höherer Ausführungszeit bevorzugt. Hierbei wird angenommen, dass das Parameterset durch den Filter bereits reduziert wurde und der Rechenaufwand deshalb vertretbar ist. Die aus dem Filter kommenden relevanten Parameter $\boldsymbol{\omega}^*_m$ werden durch den Wrapper weiter verfeinert und die Untermenge $\boldsymbol{\omega}_m$ als finales Resultat zurückgeliefert (siehe Abbildung 3.5).

Die **Auswertung** des Verfahrens erfolgt, basierend auf einem Datensatz, bestehend aus mehreren Testfahrten, wobei bei jeder Fahrt 11 physikalischen Größen erfasst werden. Insgesamt sind für jede Größe circa $5{,}4 \cdot 10^5$ Datenpunkte aufgezeichnet worden. Ein Ausschnitt des Testdatensatzes ist im Anhang A.1 dargestellt.

Idealerweise soll der Prädiktor für den Wrapper identisch zu dem sein, das später als Sensormodell zum Einsatz kommt. Im operativen Betrieb ist es jedoch oft so, dass zum Zeitpunkt der Entscheidungsfindung die Modelle noch nicht fertig implementiert sind. Um dennoch die Performanz potenziell möglicher Modellierungsansätze zur Lösung des zugrunde liegenden Problems abschätzen zu können, eignen sich je nach Wahl des Evaluierungsframeworks unterschiedliche Toolboxen: (a) Die Statistik– und Maschinelles–Lernen–Toolbox für Matlab [124], (b) die „Scikit–Learn"–Toolbox für Python [125] oder (c) die Weka–Toolbox der Waikato University [126]. Unter Zuhilfenahme dieser Toolboxen ist es ohne großen Aufwand möglich, unterschiedliche Modellierungsansätze, wie zum Beispiel Gauß–Prozesse, Entscheidungsbäume oder Stützvektormethoden miteinander zu vergleichen, um die Wahl des finalen Modells zu vereinfachen. Der Testmessdatensatz wird mit Hilfe der Matlab–Toolbox ausgewertet. Der exponentielle gaußsche Regressionsprozess [127] und „Random–Forest" [128] sind laut der Auswertung die geeignetsten Kandidaten für das zugrundeliegende Problem. Die Untersuchung der heutzutage verwendeten Modellierungsansätze für die umfelderfassenden Sensoren am Anfang von Kapitel 3 zeigte, dass der gaußsche Regressionsprozess höhere Relevanz besitzt, weshalb dieser im Folgenden für den Wrapper als Prädiktor eingesetzt wird.

Für die Auswertung werden die Messdaten in Trainings– und Testdatensatz aufgeteilt, wobei 80% zur Parameterauswahl und 20% zum Testen verwendet werden. Diese Aufteilung dient dabei nicht der Validierung während der Suche, sondern dem finalen Test nach der Parameterauswahl. Dies ist notwendig, um Überanpassung auf der Parameterauswahlebene zu vermeiden und brauchbare Schätzungen der erwarteten Endleistung zu erhalten [110]. Zur Vermeidung der Überanpassung auf Modellbildungsebene wird der Trainingssatz während der Suche ebenfalls geteilt. Um die Performanz des Ansatzes messen zu können, werden alle Schritte aus Abbildung 3.5 auf den Trainingsdatensatz angewendet und folgende Fälle miteinander verglichen: Parameterauswahl basierend auf (a) der Pearson–Korrelation $\boldsymbol{\omega}_{k^*}^{\mathrm{P}}$, (b) der gemeinsamen Information $\boldsymbol{\omega}_{k^*}^{\mathrm{I}}$, (c) der vereinten Menge $\boldsymbol{\omega}_{k^*}$, (d) dem hybriden Wrapperansatz $\boldsymbol{\omega}_m$ und (e) dem Wrap-

Tabelle 3.3: Ergebnisse der Parameterauswahl am Beispiel eines **Lidar–Sensors und $\boldsymbol{\delta}_{\mathbf{y}}$ als Zielgröße**, basierend auf der Anzahl der ausgewählten Parameter, der mittleren quadratischen Abweichung und der normierten benötigten Zeit. Zur Auswertung werden Filter– und Wrapper–Methoden einzeln, sowie der hybride Ansatz verwendet.

		Modellierungsgröße	MQF	Zeit
Filter	Pearson–Korrelation (M_{P})	$d_{\mathrm{x}},\ v_{\mathrm{y}},\ v_{\mathrm{x}}^{\mathrm{rel}},\ \gamma,\ \rho$	1,4841	$4{,}8690 \cdot 10^{-5}$
	Gemeinsame Information (M_{I})	$v_{\mathrm{x}},\ v_{\mathrm{y}},\ v_{\mathrm{x}}^{\mathrm{rel}},\ \gamma,\ \rho$	1,5300	$\mathbf{3{,}3710 \cdot 10^{-5}}$
	$M_{\mathrm{P}} \cup M_{\mathrm{I}}$	$d_{\mathrm{x}},\ v_{\mathrm{x}},\ v_{\mathrm{y}},\ v_{\mathrm{x}}^{\mathrm{rel}},\ \gamma,\ \rho$	1,4196	$8{,}6200 \cdot 10^{-5}$
Wrapper	Hybrid	$\boldsymbol{v_{\mathrm{x}},\ v_{\mathrm{x}}^{\mathrm{rel}},\ \gamma,\ \rho}$	**0,8809**	1,0000
	SRS	$d_{\mathrm{x}},\ d_{\mathrm{y}},\ v_{\mathrm{x}},\ v_{\mathrm{x}}^{\mathrm{rel}},\ \gamma,\ \theta,\ \rho$	1,2200	4,3370

per unter Verwendung des kompletten Parametersets ohne Vorauswahl durch einen Filter. In allen fünf Fällen wird ein Gaußscher–Regressionsprozess als Prädiktor zur Bewertung der Parameterauswahl verwendet. In Tabelle 3.3 sind die Ergebnisse für einen Lidar–Sensor und den Sensorfehler in y–Richtung δ_y als Zielgröße vorgestellt. Weitere Ergebnisse für eine Kamera und δ_x als Zielgröße sind im Anhang A.2 zusammengefasst.

Die Zeilen in Tabelle 3.3 zeigen die fünf zuvor aufgelisteten Fälle, die Spalten die entsprechenden Evaluierungsergebnisse für die finale Wahl der Parameter, den mittleren quadratischen Fehler (RMSE) und die für die Parameterauswahl benötigte Zeit normiert auf die Zeit des hybriden Ansatzes. Die Minimalwerte beziehungsweise das kleinste Parametersubset sind in jeder Spalte fett markiert. Ein zu erwartendes Resultat ist, dass die Berechnungszeit für Filter um Größenordnungen kleiner ist als die der Wrapper. Die Zeit für die Auswertung des hybriden Ansatzes ist jedoch um etwa Faktor vier kleiner als der SRS Wrapper ohne Vorauswahl durch einen Filter. Grund hierfür ist die größere Anzahl von Parametern, die SRS im Vergleich zum hybriden Ansatz eingangsseitig bekommt. In Bezug auf RMSE erreicht der hybride Ansatz mit kleinstem Parametersubset die beste Performanz. Die Ursache könnte der Schachtelungseffekt beziehungsweise ein lokales Minimum sein. Dieses Verhalten wird nicht für jeden Sensor und jede Zielgröße beobachtet (siehe Anhang A.2), nimmt man jedoch an, dass die Auswertungszeit für den hybriden Ansatz akzeptabel ist, so lässt sich, basierend auf den Ergebnissen der Auswertung zusammenfassend sagen, dass der hybride Ansatz tendenziell den besten Kompromiss zwischen der Parametersubsetgröße, dem mittleren quadratischen Fehler und der Auswertungszeit erreicht.

3.5 Verfahren zur systematischen Messdatenerfassung

Neben der Entscheidung über die sinnvolle Parameterwahl ist auch die systematische Messdatenerfassung ein wesentlicher Teil der Bedatung datengetriebener Sensormodelle. Wie einleitend erwähnt, ist unter Bedatung die Bereitstellung der Eingangsdaten für die Modellbildung gemeint. Im vorherigen Unterkapitel wurden die Aspekte der Einschränkung des Parameterraums bereits behandelt. Am einleitenden Beispiel in Kapitel 3 wird verdeutlicht, dass selbst für eine kleine Anzahl von Parametern eine vollständige Messdatenerfassung kaum möglich ist. Fokus dieses Kapitels ist deshalb die Reduktion der für die Modellierung notwendigen Messdatenpunkte. Mit anderen Worten soll hier die Frage beantwortet werden, wie die Anzahl der Messdaten, bezogen auf den Sensorsichtbereich, reduziert werden kann, ohne dabei die Modellgüte maßgeblich zu beeinflussen. Bezugnehmend auf den Gesamtüberblick aus Abbildung 3.2 handelt es sich dabei um die Messdatenreduktion der ersten Evaluierungsschleife.

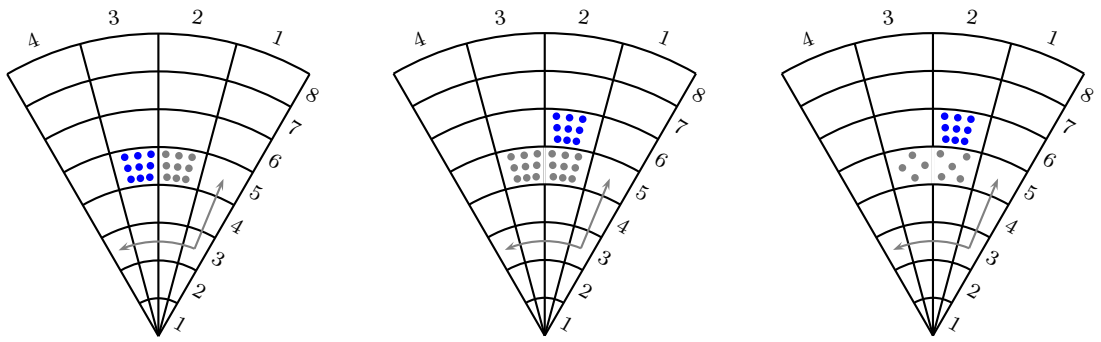
3.5.1 Vorstellung des Ansatzes

Die Literaturrecherche zeigt, dass die Modellierung umfelderfassender Sensoren im Allgemeinen auf Modellierung von Fehlerverteilungen basiert. Eine Möglichkeit, die Datenmenge zu reduzieren, ist also Bereiche mit ähnlichen statistischen Eigenschaften

innerhalb des Sensorsichtbereichs zu finden. Dabei öffnen sich drei weitere Fragen: (a) wie lassen sich zusammenhängende Bereiche mit ähnlichen statistischen Eigenschaften innerhalb des Sensorsichtbereichs finden, (b) welche statistischen Tests eignen sich für diese Aufgabe und (c) wie groß ist die minimale Datenmenge einer Stichprobe, um die statistischen Eigenschaften ableiten zu können? Diese Fragen werden im Folgenden diskutiert.

Um Bereiche mit ähnlichen statistischen Eigenschaften zu finden, wird im Rahmen dieser Arbeit ein Ansatz vorgeschlagen, der erst den gesamten Sensorsichtbereich in Zellen unterteilt und diese im Anschluss, basierend auf den statistischen Eigenschaften der in den benachbarten Zellen befindlichen Datenpunkte, zusammenführt. Das Vorgehen wird in Abbildung 3.6 visualisiert. Die Datenpunkte in den benachbarten Zellen werden gerichtet (siehe graue Pfeile), erst in radiale und anschließend in laterale Richtung, miteinander verglichen. Die dargestellten Punkte beziehen sich dabei auf die Referenzmesspunkte. Es werden jedoch die Messabweichungen zwischen der Referenz und dem unter Experiment stehenden Sensor in den benachbarten Zellen miteinander verglichen. In Abbildung 3.6a sind in grau und blau die Referenzpunkte zweier benachbarter Zellen gezeigt. Diese werden mit Hilfe eines statistischen Tests auf Ähnlichkeit der WDF geprüft. Ist dies der Fall, werden die beiden Zellen zusammengeführt und die nächste benachbarte Zelle mit der vereinten verglichen (siehe Abbildung 3.6b). Dabei wird angenommen, dass nur räumlich zusammenhängende Gebiete innerhalb des Sensorsichtbereichs ähnliche statistische Eigenschaften aufweisen können. Die einzelnen Schritte sind im Anhang A.3 im Algorithmus 1 formalisiert. Nach der Rasterung kann die Anzahl der Messpunkte innerhalb der vereinigten Zellen reduziert werden (siehe Abbildung 3.6c).

Die nächste Frage ist, welche statistischen Tests sich für den Vergleich zweier WDF eignen. Messgrößen umfelderfassender Sensoren sind im Allgemeinen nicht–parametrisch

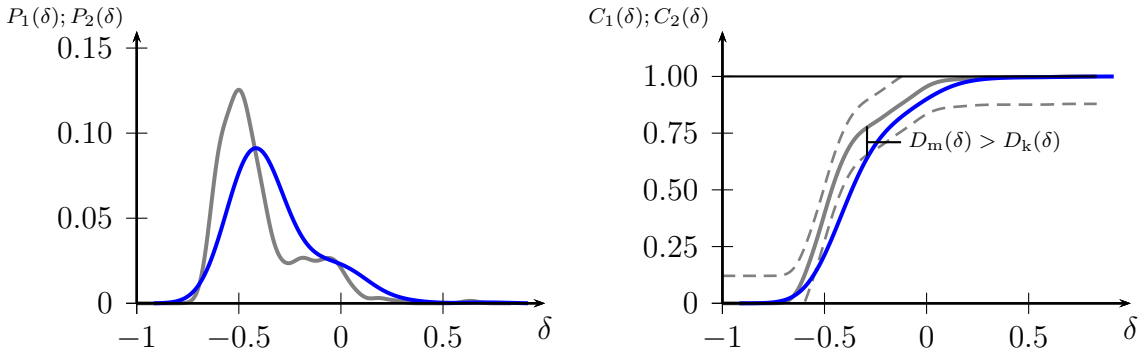


(a) Rasterung initial. **(b)** Vereinigung der Zellen. **(c)** Reduktion der Punkte.

Abbildung 3.6: Visualisierung des Vorgehens zur Rasterung des Sensorsichtbereichs. In (a) ist der Sensorsichtbereich mit initialer Rasterung dargestellt. Die Zellen (5,2) und (5,3) werden auf statistische Ähnlichkeit überprüft. In (b) werden diese zu der Menge $\{(5,2),(5,3)\}$ zusammengeführt und im nächsten Schritt mit der Zelle (6,2) verglichen. Nach der Rasterung kann die Anzahl der Messdatenpunkte in vereinigten Zellen reduziert werden (siehe (c)).

und kontinuierlich [36]. Unter Beachtung dieser Eigenschaften, eignen sich zur Überprüfung der Übereinstimmung zweier WDF die Kolmogorov–Smirnov (KS) und χ^2–Tests [129]. Im Rahmen dieser Arbeit wird vorgeschlagen, den KS–Test zu verwenden, da dieser im Gegensatz zu χ^2 keine Anforderungen an die Gleichheit der Dimensionen von den zu vergleichenden Stichproben stellt [129] und es zu erwarten ist, dass dieser kleinere Abweichungen zwischen den WDF detektieren kann [130].

Der KS–Test ist ein nicht–parametrischer statistischer Hypothesentest auf Übereinstimmung von WDF zweier Stichproben, der kontinuierliche Daten voraussetzt. Es gibt zwei Arten des KS–Tests: Zum Vergleich einer bekannten und einer empirischen Verteilung (Einstichprobenproblem) und zum Vergleich von zwei empirischen Verteilungen (Zweistichprobenproblem). Aufgrund der Problemstellung aus Abbildung 3.6, bei der zwei empirische Stichproben miteinander zu vergleichen sind, werden in dieser Arbeit nur die Aspekte des zweiseitigen Zweistichproben–KS–Tests vorgestellt. Die Grundidee ist dabei, zwei unter Test stehende WDF mit Hilfe deren KVF zu vergleichen. Als Nullhypothese H_0 wird dabei angenommen, dass die Verteilungen, aus denen die Stichproben stammen, gleich sind: $P_1(\delta) = P_2(\delta)$. Die Alternativhypothese H_A sagt aus, dass die Verteilungen verschieden sind: $P_1(\delta) \neq P_2(\delta)$. Das Prinzip des KS–Tests ist in Abbildung 3.7 visualisiert und wird im Folgenden näher erläutert. In Abbildung 3.7a sind zwei WDF $P_1(\delta)$ und $P_2(\delta)$, die auf Gleichheit der zu Grunde liegenden Verteilung getestet werden sollen, dargestellt. In Grau ($P_1(\delta)$) ist die selbe WDF dargestellt, die als Beispiel in Kapitel 2.2.2 gezeigt wurde. Diese soll mit der in Blau ($P_2(\delta)$) dargestellten Verteilungsfunktion mit Hilfe des KS–Tests verglichen werden. Dazu werden beide WDF in KVF $C_1(\delta)$ und $C_2(\delta)$ überführt (siehe Abbildung 3.7b). In grau gestrichelt sind die Konfidenzintervalle von $C_1(\delta)$ gezeigt. Basierend darauf kann entschieden werden, ob die beiden unter Test stehenden Verteilungen ähnliche statistische Eigenschaften aufweisen. Sollte die KVF der Testdaten die Konfidenzintervalle einmal überschreiten, wird als Schlussfolgerung abgeleitet, dass beide Stichproben unterschiedlichen Fehlerverteilungen folgen. Dieser Fall ist in Abbildung 3.7b zu sehen.



(a) WDF. **(b)** Kumulierte Verteilungsfunktion (KVF).

Abbildung 3.7: Visualisierung des Kolmogorov–Smirnov–Tests. Für Stichproben zweier benachbarter Zellen werden aus den WDF $P_1(\delta)$ (grau) und $P_2(\delta)$ (blau) in (a) die KVF $C_1(\delta)$ und $C_2(\delta)$ in (b) berechnet. Basierend auf Konfidenzintervallen wird entschieden, ob beide Datensätze die selbe Verteilung besitzen.

Dort überschreitet die maximale Abweichung $D_\mathrm{m}(\delta)$ zwischen $C_1(\delta)$ und $C_2(\delta)$ den kritischen Wahrscheinlichkeitswert $D_\mathrm{k}(\delta)$, weshalb die Annahme der übereinstimmenden WDF beider Stichproben zu verwerfen ist.

Formal wird beim KS–Test überprüft, ob die maximale Abweichung $D_\mathrm{m} = \mathsf{max}(|C_1(\delta) - C_2(\delta)|)$ den kritischen Wahrscheinlichkeitswert D_k überschreitet,

$$D_\mathrm{m} \overset{?}{>} D_\mathrm{k}. \tag{3.7}$$

Falls ja, wird die Nullhypothese H_0 verworfen. Der kritische Wahrscheinlichkeitswert hängt dabei vom Signifikanzniveau α und den Stichprobengrößen n_1 und n_2 ab. Für kleinere Stichproben kann D_k aus Wertetabellen entnommen werden [130, 131]. In Tabelle 3.4 ist ein Auszug der D_k–Werte für $\alpha = 0{,}05$ beispielhaft dargestellt:

Tabelle 3.4: Auszug der kritischen Werte D_k der maximalen absoluten Differenz zwischen zwei Stichproben für $\alpha = 0{,}05$ nach [132, 133].

$n_2 \backslash n_1$	...	9	10	11	...
9	...	0,67	0,59	0,6	...
10	...	0,59	0,7	0,55	...
11	...	0,6	0,55	0,64	...

Für größere Stichproben kann D_k wie folgt approximiert werden [131, 134, 135],

$$D_\mathrm{k} = \sqrt{\frac{1}{n_1} + \frac{1}{n_2}} \cdot \sqrt{-\frac{1}{2} \ln \frac{\alpha}{2}}. \tag{3.8}$$

D_k gibt dabei die maximale absolute Abweichung zwischen zwei Stichproben der Größe n_1 beziehungsweise n_2 für ein bestimmtes α an. Mit D_k werden auch die in Abbildung 3.7b gezeichneten Konfidenzintervalle definiert.

Die dritte Frage im Kontext der systematischen Messdatenerfassung ist die Frage nach der notwendigen Anzahl von Messdatenpunkten. Wie im vorherigen Absatz erwähnt, wird der kritische Bereich des KS–Tests unter anderem von der Anzahl der Messdatenpunkte bestimmt. Mit steigender Anzahl der Datenpunkte in der Stichprobe sinkt der kritische Wahrscheinlichkeitswert D_k, das Konfidenzintervall wird also „schmäler“. Um beispielsweise 95% sicher zu sein ($\alpha = 0{,}05$), dass der Unterschied zwischen $C_1(\delta)$ und $C_2(\delta)$ den Wert $D_\mathrm{k} = 0{,}5$ nicht überschreitet, sind unter der Annahme, dass beide Stichproben gleich groß sind, $n_1 = n_2 \approxeq 15$ Messpunkte notwendig. Damit $D_\mathrm{k} \leq 0{,}25$ ist, werden $\approxeq 60$ und somit vier mal so viele Datenpunkte benötigt. Wie sehr sich $C_1(\delta)$ und $C_2(\delta)$ unterscheiden dürfen, und wie groß somit die Stichprobengröße sein muss, hängt von der Anwendung ab und basiert auf der Meinung des Experten.

3.5.2 Umsetzung und Auswertung des Verfahrens

Zur Bewertung der gewonnen Erkenntnisse wird eine Reihe an Untersuchungen durchgeführt. Die einzelnen Schritte sind in Abbildung 3.8 visualisiert und sind Teil der

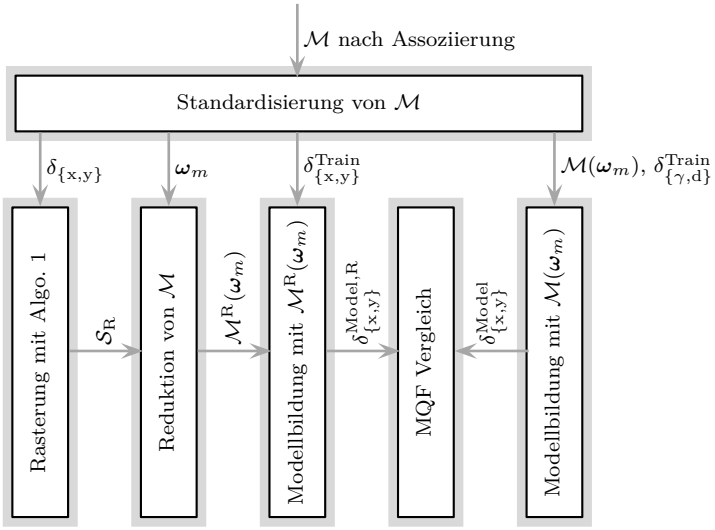


Abbildung 3.8: Visualisierung des Vorgehens zur Rasterung des Sensorsichtbereichs als Teil der systematischen Messdatenreduktion aus Gesamtarchitektur in Abbildung 3.2.

systematischen Messdatenreduktion aus der Gesamtarchitektur in Abbildung 3.2. Der Ausgangspunkt hier ist identisch zu dem Vorgehen aus Kapitel 3.4.2. Es wird derselbe Messdatensatz $\mathcal{M}$ verwendet, dessen Ausschnitt beispielhaft im Anhang A.1 zu sehen ist. Nach der Standardisierung wird dieser mit dem Algorithmus 1 in Gierwinkel γ und Abstand d gerastert. Wie zuvor beschrieben, werden dazu die Messabweichungen δ_{x} und δ_{y} mittels KS–Test verglichen.

Der ermittelte gerasterte Sensorsichtbereich $\mathcal{S}_{\text{R}}$ wird im Anschluss zur Reduktion des ursprünglichen Messdatensatzes $\mathcal{M}$ verwendet. Dabei werden nur die relevanten Modellierungsgrößen $\boldsymbol{\omega}_m$ berücksichtigt. Die Wahl von $\boldsymbol{\omega}_m$ basiert auf den Ergebnissen aus Kapitel 3.4.2. Die Anzahl der Messpunkte in einer vereinigten Rasterzelle wird dabei

Tabelle 3.5: Verwendete Parameter zur Rasterung des Sensorsichtbereichs.

	Parameter	Wert
Rasterung	Abstandsreichweite d	2m . . . 200m
	Abstandsauflösung Δ_d	1m
	Horizontaler Sichtbereich γ	$-50° \cdots +60°$
	Horizontale Auflösung Δ_γ	0,286°
KS–Test	Signifikanzniveau α	0,05
	Minimale Stichprobengröße pro Zelle	15
Modell	Modellierungsgrößen ω_m	Ergebnisse aus Kapitel 3.4.2
	Zielgrößen	$\delta_{\text{x}}, \delta_{\text{y}}$

Tabelle 3.6: Ergebnisse der Rasteranalyse für Lidar– und Kamerasensoren, basierend auf dem Reduktionsfaktor und der mittleren quadratischen Abweichung zwischen prädizierten Größen.

Sensor	Größe $\mathcal{M}(\boldsymbol{\omega}_m)$	Größe $\mathcal{M}^{\mathrm{R}}(\boldsymbol{\omega}_m)$	Reduktionsfaktor	ΔMQF
Lidar für δ_{y}	31.438	21.535	0,315	0,009817
Lidar für δ_{x}	31.438	25.753	0,181	0,021419
Kamera für δ_{y}	49.082	39.781	0,190	0,138950
Kamera für δ_{x}	49.082	44.387	0,096	0,006715

auf den Mittelwert aus den einzelnen initialen Zellen reduziert, indem diese zufällig gezogen werden. Dieser Schritt ist in Abbildung 3.6c visualisiert.

Um die Wirkung des Reduktionsschritts bewerten zu können, werden zwei Gaußsche–Regressionsmodelle (ähnlich zur Parameteranalyse aus Kapitel 3.4.2), basierend auf dem reduzierten Messdatensatz $\mathcal{M}^{\mathrm{R}}(\boldsymbol{\omega}_m)$, und dem vollständigen Messdatensatz $\mathcal{M}(\boldsymbol{\omega}_m)$ zur Modellierung von Messabweichungen in x– und y–Richtung, basierend auf einem Trainingsdatensatz $\delta^{\mathrm{Train}}_{\{\mathrm{x,y}\}}$, gebildet. Die Details der verwendeten Parameter sind in Tabelle 3.5 zusammengefasst. Zu erwähnen ist, dass die Rasterung des Sensorsichtbereichs auf der Annahme basiert, dass eine Zelle nicht größer als die herkömmliche Fläche unter einem Fußgänger ist, die mit $1\mathrm{m}^2$ angenommen wird. Um diese Anforderung in 200m Entfernung zu erfüllen, wird eine Winkelauflösung von 0,286° benötigt.

Im letzten Schritt werden basierend auf einem Testdatensatz und unter Verwendung der generierten Modelle die Zielgrößen $\delta^{\mathrm{Model,R}}_{\{\mathrm{x,y}\}}$ und $\delta^{\mathrm{Model}}_{\{\mathrm{x,y}\}}$ prädiziert und miteinander mittels RMSE verglichen. Die Testergebnisse sind in Tabelle 3.6 zusammengefasst. Zu erkennen ist, dass die Anzahl der Messdatenpunkte um mindestens circa 10% und im Mittel um circa 20% reduziert werden konnte. Gleichzeitig hält sich der ΔRMSE zwischen den RMSE–Werten der zwei Modelle, die aus vollem und reduziertem Messdatensatz abgeleitet werden, in einem niedrigen Bereich, der um Größenordnungen kleiner ist als die RMSE–Werte, die während der Parameteranalyse ermittelt wurden (siehe Tabelle 3.3, A.1, A.2, A.3). Zusammen mit den Ergebnissen aus Kapitel 3.4 trägt der vorgestellte Ansatz seinen Teil zur Beantwortung der zweiten Forschungsfrage bei.

3.6 Fazit zur Validierung und Bedatung von Sensormodellen

In Kapitel 2.2.3 werden acht Anforderungen an die Modellvalidierung formuliert. In diesem Kapitel wird daraufhin in Abbildung 3.1 eine Methode für die Validerung umfelderfassender Sensormodelle im Automobilbereich abgeleitet. Die vorgestellte Methode umfasst vier Schritte: (a) Erfassung realer Daten, (b) Resimulation realer Szenarien und Generierung synthetischer Daten, (c) Validierung realer und synthetischer Wertepaare und (d) Entscheidungsfindung, ob das zugrundeliegende Modell valide ist. Die vorgestellte Methode ist für die Validierung datengetriebener Sensormodelle ausgelegt, wes-

halb die Erfassung realer Messdaten in die Methode integriert ist. Durch Verwendung von Referenzdaten für die Resimulation werden die aleatorischen und epistemischen **Unsicherheiten** minimiert (siehe Kapitel 2.2.1), wodurch die letzte Anforderung aus der Anforderungsliste erfüllt wird.

Die Bewertung der Modellergebnisse basiert dabei neben dem direktem Vergleich realer und synthetischer Sensordaten auch auf den Ergebnissen des Umfeldmodells. Dies verleiht der Methode einen **gesamthaften** Charakter, der dem vierten Punkt aus der Anforderungsliste entspricht. Die Entscheidung bezüglich der Validität eines Modells basiert dabei auf **Spezifizierung** des beabsichtigten Einsatzes des Modells durch das Verwenden des Umfeldmodells. Somit wird die dritte Anforderung an die Modellvalidierung erfüllt.

Die allgemeingültige Sicht der vorgestellten Validierungsmethode wird in Kapitel 3.2 konkretisiert. Es wird ein Architekturvorschlag vorgestellt, um aufzuzeigen, wie weitere Anforderungen an eine Validierungsmethode erfüllt werden können. Der Fokus der Architektur liegt auf Verwendung von offenen beziehungsweise etablierten Schnittstellen, wie beispielsweise OSI, sodass keine Abhängigkeiten von proprietären Tools entstehen. Durch klare Trennung diverser Bestandteile mittels Schnittstellen, wird die Komplexität der Aufgabe reduziert und gleichzeitig die **Nachvollziehbarkeit** erhöht. Des Weiteren werden diverse Schnittstellenformate und parallelisierbare Software–Tools eingeführt, wodurch die **Skalierung** der Methodik ermöglicht wird. Die meisten dieser Formate sind offen, sodass der vorgeschlagene Ansatz keine proprietären Abhängigkeiten hat. Darüber hinaus unterstützen die vorgeschlagenen Formate die, für die Sensormodelle notwendigen, Objektlisten– und Rohdatenbeschreibungen.

In Kapiteln 3.1 und 3.2 wird somit die erste Forschungsfrage beantwortet. Des Weiteren wird gezeigt, wie die oben aufgeführten fünf von acht Anforderungen an einen Validierungsprozess erfüllt werden können. Die restlichen drei Anforderungen *Intuitivität*, *Objektivität* und *Fehlervermeidung* entspringen eher inhaltlicher Natur und sind meist von Metriken abhängig, weshalb diese in den späteren Kapiteln 4 und 5 behandelt werden.

Im weiteren Verlauf dieses Kapitels wurde die zweite Forschungsfrage, bezüglich den Möglichkeiten der Reduktion von Messdaten zur Bedatung eines Sensormodells, beantwortet. Dazu werden Ansätze zur Reduktion der Anzahl an Modellierungsgrößen und zur Rasterung des Sensorsichtbereichs vorgestellt. Ersteres wird in Kapitel 3.4 erläutert. Es wird ein gemischter Filter–Wrapper–Ansatz aus der Variablenauswahl–Disziplin entworfen, wobei beim Design Wert auf die Laufzeit, Robustheit und Allgemeingültigkeit der Ergebnisse gelegt wird. Experimente zeigen, dass in Summe mit diesem Ansatz ein guter Kompromiss zwischen der Größe des Subsets der relevanten Modellierungsgrößen und der Performanz des Sensormodells gefunden werden kann.

Die Rasterung des Sensorsichtbereichs wird in Kapitel 3.5 vorgestellt und zeigt eine weitere Möglichkeit zur Reduktion der für die Bedatung eines Sensormodells notwendiger Daten. Die Basis dieses Ansatzes bilden ein Rasteralgorithmus und der KS–Test zur Überprüfung von zwei KVF auf Ähnlichkeit. Damit konnte mit einer im Mittel um 20% kleineren Anzahl von Messpunkten annähernd gleiche Modellgüte erreicht wer-

den. Basierend auf diesem Ansatz können beispielsweise Tools zur Onlineauswertung während der Messaufzeichnung zur besseren Datennutzung entwickelt werden.

4 Methoden zur Bewertung phänomenologischer Sensormodelle

In diesem Kapitel wird die dritte Forschungsfrage, wie die phänomenologischen Sensormodelle objektiv bewertet werden können, geklärt. Unter phänomenologischen Modellen sind Ansätze zu verstehen, die statistische Eigenschaften realer Sensoren abbilden. Bezugnehmend auf Tabelle 3.1, handelt es von Modellen, die die Sensordaten auf Objektlistenebene unter Verwendung von parametrischen und nicht–parametrischen Modellierungsansätzen nachbilden.

Neben realitätsnaher Abbildung von Fehlerverteilungen muss ein Sensormodell unter anderem diverse weitere Aspekte, wie beispielsweise Netzwerk bedingte Verzögerung beziehungsweise Verlust von Nachrichten, Abbildung des Sensorsichtbereichs, Modellierung von Geisterobjekten oder Klassifizierung von Verkehrsteilnehmern, nachbilden können. Um die Vielfalt der Effekte, die ein Sensormodell berücksichtigen muss, beherrschbar zu machen, wird in [25] ein modulares Sensormodellkonzept vorgestellt, das auch in dieser Arbeit Anwendung findet. Die vorgeschlagene Architektur ist in Abbildung 4.1 dargestellt. Der Ausgangspunkt sind dabei die von der Fahrsimulation zur Verfügung gestellten Referenzdaten $\mathcal{O}_0 = \Omega_\mathrm{s}$. Diese werden dann von den n Sensormo-

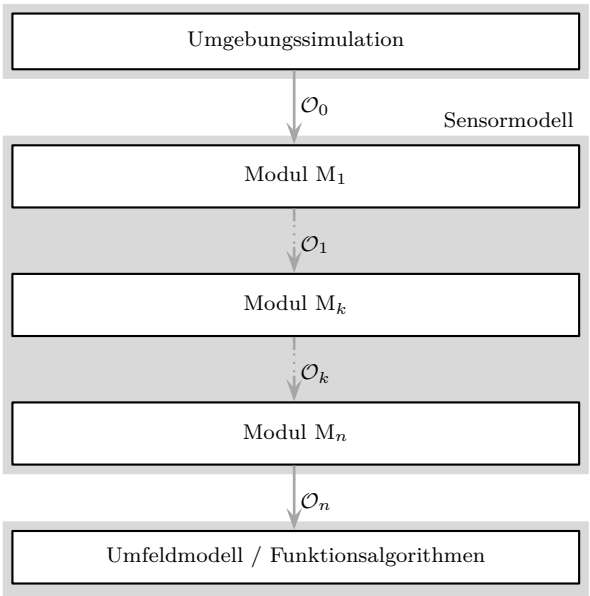


Abbildung 4.1: Modulare Sensormodellarchitektur nach [25].

dellmodulen $\mathrm{M}_k(\mathcal{O}_{k-1})$ nacheinander verarbeitet, die jeweils einen modifizierten Satz von Objektlisten $\mathcal{O}_1 \ldots \mathcal{O}_n$ erzeugen. Die einzelnen Module bilden dabei die unterschiedlichen zuvor erwähnten Sensoreigenschaften ab. Das Modul M_1 kann beispielsweise im ersten Schritt den Sensorsichtbereich abbilden und M_2 im darauf folgenden Schritt das Rauschen hinzufügen. Das Endergebnis ist eine Liste erkannter Objekte $\mathcal{O}_n$, die im operativen Betrieb (siehe Abbildung 1.1) beispielsweise als Eingang für das Umfeldmodell oder einen anderen Fahrerassistenz–Funktionsalgorithmus verwendet wird.

Eines der essentiellen Module im vorgestellten modularen Sensormodellkonzept ist die phänomenologische Modellierung. In dieser Arbeit wird der nicht–parametrische Modellierungsansatz als Repräsentant der phänomenologischen Modelle betrachtet, da dieser aus Sicht der Validierung alle Belange der parametrischen Modellierung, wie beispielsweise Nachbildung bestimmter Fehlerverteilungsformen, mit abdeckt. Somit können die Konzepte der nicht–parametrischen Modellvalidierung auf die Validierung parametrischer Modelle projiziert werden. Als Beispiel für ein nicht–parametrisches Modell wird hier der Ansatz aus [36] vorgestellt. Dieser ist in Abbildung 4.2 visualisiert. Er basiert auf dem Ähnlichkeitsvergleich einer realen und simulierten Szene. Dazu wird erst eine Datenbank von real eingefahrenen Messdaten angelegt und während der Simulation in jedem Zeitschritt ermittelt, welche der in Realität aufgezeichneten Messungen am ähnlichsten zu der aktuell simulierten Szene sind. Basierend darauf, wird eine synthetische Sensormessung ausgegeben, die ähnlich einer realen Messung ist.

In Abbildung 4.2 wird qualitativ gezeigt, wie unter Verwendung dieses Ansatzes eine eindimensionale Größe ζ_s mit einer eindimensionalen Zustandsbeschreibung Ω_s modelliert werden kann. Exemplarisch wird der simulierte Abstand $\zeta_\mathrm{s} = d_\mathrm{s}$ zu einem Vorderfahrzeug, basierend auf dem wahren Abstand $\Omega_\mathrm{s} = d_\mathrm{r}$, gezeigt. Auf der linken

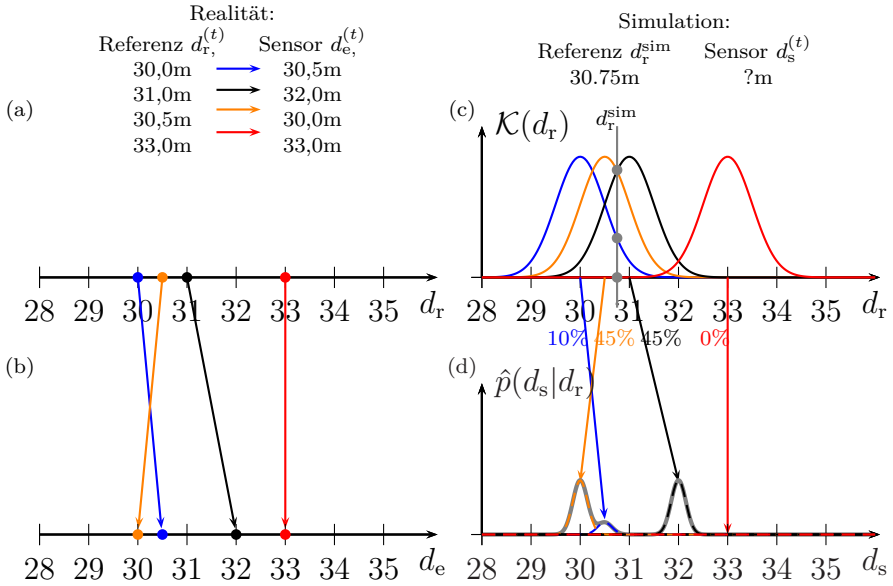


Abbildung 4.2: Prinzip der nicht–parametrischen Modellierung nach [36].

Seite ist die Messdatenerfassung in Realität und auf der rechten die darauf basierenden Simulationsergebnisse dargestellt.

Zunächst werden die realen Messdaten mit einem Versuchsfahrzeug, ausgestattet mit einem Referenz– und dem zu modellierenden Sensor, erfasst. Zu jedem Zeitpunkt t werden Messtupel $\mathbb{M} = \left\{d_\mathrm{r}^{(t)}, d_\mathrm{e}^{(t)}\right\}_{t=1:N_\mathbb{M}}$ von dem Referenzsystem $d_\mathrm{r}^{(t)}$ und dem unter Experiment stehenden Sensor $d_\mathrm{e}^{(t)}$ aufgezeichnet. In Abbildung 4.2 sind die Referenzmessungen unter (a) und die Messungen des nachzubildenden Sensors unter (b) gezeigt. In dem Beispiel beträgt die Anzahl der Messungen $N_\mathbb{M} = 4$, die mit jeweils unterschiedlichen Farben hinterlegt sind. Der Datensatz besteht somit aus Verknüpfungen von Referenz– und Sensormessungen, was mit den farbigen Pfeilen verdeutlicht wird. Die erste Messung umfasst beispielsweise den Referenzwert $d_\mathrm{r}^{(1)} = 30{,}0\mathrm{m}$, wobei der Sensor den Wert $d_\mathrm{e}^{(1)} = 30{,}5\mathrm{m}$ gemessen hat.

Basierend auf dem aufgezeichneten Datensatz soll nun der Abstand zu einem Vorderfahrzeug simuliert werden (Abbildung 4.2 rechts). Es wird angenommen, dass sich das Fahrzeug in der Simulation in $d_\mathrm{r}^\mathrm{sim} = 30{,}75\mathrm{m}$ Entfernung befindet. Um das Verhalten des realen Sensors nachzubilden, wird erst die Relevanz der realen Messungen für die aktuell simulierte Situation mit Hilfe eines Kerns $\mathcal{K}(d_\mathrm{r})$ quantifiziert. Die Gaußschen–Kernfunktionen sind unter (c) zu der jeweiligen realen Messung in der entsprechenden Farbe dargestellt. In dem Beispiel befinden sich die Messungen $d_\mathrm{r}^{(2)} = 31{,}0\mathrm{m}$ und $d_\mathrm{r}^{(3)} = 30{,}5\mathrm{m}$ am nächsten zu dem simulierten Abstand und werden jeweils am höchsten mit $\mathcal{K}\big(d_\mathrm{r}^{(2)}\big) = \mathcal{K}\big(d_\mathrm{r}^{(3)}\big) \approx 45\%$ gewichtet. Die Messungen $d_\mathrm{r}^{(1)} = 30{,}0\mathrm{m}$ und $d_\mathrm{r}^{(4)} = 33{,}0\mathrm{m}$ sind hingegen deutlich weiter weg und werden entsprechend niedriger mit $\mathcal{K}\big(d_\mathrm{r}^{(1)}\big) \approx 10\%$ und $\mathcal{K}\big(d_\mathrm{r}^{(4)}\big) \approx 0\%$ gewichtet.

Im nächsten Schritt werden die ermittelten Gewichtungen verwendet, um den Beitrag der vier Messungen zu der Wahrscheinlichkeitsdichte $\hat{p}(d_\mathrm{s}|d_\mathrm{r})$ zu skalieren. Die gewichtete Summe der einzelnen Wahrscheinlichkeitsdichten bildet dabei die WDF des simulierten Sensors. Diese ist in Abbildung 4.2c in grau gezeigt. Es ist zu erwähnen, dass bei diesem Ansatz zwei unterschiedliche Kerne verwendet werden: Relevanzkern und Beitragskern. Mit dem Relevanzkern werden die Beiträge der einzelnen Messungen ermittelt, die im nächsten Schritt mit den Beitragskernen multipliziert werden. Die beiden Kernfunktionen können, müssen aber nicht, gleich gewählt werden. Die Kernfunktionen werden über die Kernbreiten h_R des Relvanzkernes und h_B des Beitragskernes parametriert. Je größer die Relevanzkernbreite gewählt wird, desto höhere Gewichtung bekommen die weiter weg liegenden Messungen. Die resultierende Verteilugsfunktion wird also „flacher“ und „breiter“.

Im letzten Schritt kann aus der errechneten WDF eine Realisation von d_s gezogen und an die Simulation zur weiteren Verarbeitung ausgegeben werden. Die Untersuchungen in weiterführenden Abschnitten dieses Kapitels basieren auf dem vorgestellten Modellansatz.

In Kapitel 2.1.1 werden die für diese Arbeit relevanten Formate, Objektlisten und ungefilterte Messungen, vorgestellt. Wie einleitend erwähnt, sind für die phänomenologischen Modelle die Objektlisten relevant. Entsprechend Abbildung 2.3 sind Objektlisten

ein Resultat der Signalverarbeitung. Diese unterscheidet sich zwar für unterschiedliche Sensoren, deren Inhalte sind jedoch zumindest für die relevanten Sensorarten wie Radar, Lidar und Kamera ähnlich. Alle gemeinsam haben als Bestandteil der Signalverarbeitung die zeitliche Verfolgung der Objekte [16, Kapiteln 17,18 und 21], [26]. Dies hat den Effekt, dass ein Messpunkt von den vorherigen abhängt. In der Statistik und Signalverarbeitung wird dieser als Autokorrelation bezeichnet. Sie spielt für die Validierung phänomenologischer Sensormodelle eine wichtige Rolle und wird im weiteren Verlauf dieses Kapitels näher erläutert.

Aus Validierungssicht entsteht aus dem zuvor Erwähnten die Notwendigkeit zwei Arten von Analysen durchzuführen. Zum Einen die statistischen Eigenschaften von Sensordaten, basierend auf einer Messfahrt, zu ermitteln und mit den Ergebnissen des Modells zu vergleichen. Diese Art von Auswertung wird als messfahrtbasierte Analyse bezeichnet und in Kapitel 4.1 behandelt. Zum Anderen ist die Autokorrelationsanalyse durchzuführen. Hier geht es um die Bewertung der Abbildung von dynamischen Eigenschaften eines Sensors im Modell. Diese wird in Kapitel 4.2 vorgestellt.

4.1 Messfahrtbasierte Analyse

Die messfahrtbasierte Analyse bezieht sich auf einen Vergleich der Form von Fahrtrajektorien beziehungsweise deren WDF unter Zuhilfenahme diverser Metriken. Dabei werden die statistischen Eigenschaften einzelner Trajektorien erhoben und die Diskrepanz zwischen realen und simulierten Fahrten quantifiziert. Diese Art von Auswertung ist beispielsweise für Trajektorien– und Manöverplanung relevant [83].

4.1.1 Vorstellung des Ansatzes

Grundidee der Validierung ist es, Fehler zwischen Realität und Modell zu detektieren. In dieser Arbeit liegt der Fokus vor allem auf den Fehlern umfelderfassender Sensoren im Automobilbereich. In [26] widmet sich Hanke der Frage, wie diese Sensorfehler klassifiziert werden können. Die Ergebnisse sind in Abbildung 4.3 zusammengefasst. Auf der grundlegendsten Ebene werden diese in zwei Kategorien eingeteilt: (a) die Perzeptionsfehler, stammend aus dem Mess– und Signalverarbeitungsprozess (siehe Abbildung 2.3) und (b) die Paketierungsfehler, die von Enkodierung und Übertragung von Sensorausgangsdaten im Fahrzeug herrühren. Beispiele solcher Fehler für unterschiedliche Sensorarten sind in Tabelle 4.1 zusammengefasst.

Bei Perzeptionsfehlern wird zwischen Erkennung, direkter und indirekter Perzeption unterschieden. Die Erkennung ist ein entscheidender Schritt im Umgang mit Sensordaten auf Objektebene. Dieser Teil des Modells ist für das Vorhandensein oder Nichtvorhandensein von Objekten in der Ausgangsschnittstelle verantwortlich, mit anderen Worten dafür, ob ein Sensor das Objekt sieht oder nicht. Die direkte Perzeption umfasst alle Objekteigenschaften, bei denen einfache Berechnungen aus Messrohdaten zu Objektlistendaten führen. Die direkte Perzeption wird weiter in diejenigen Größen

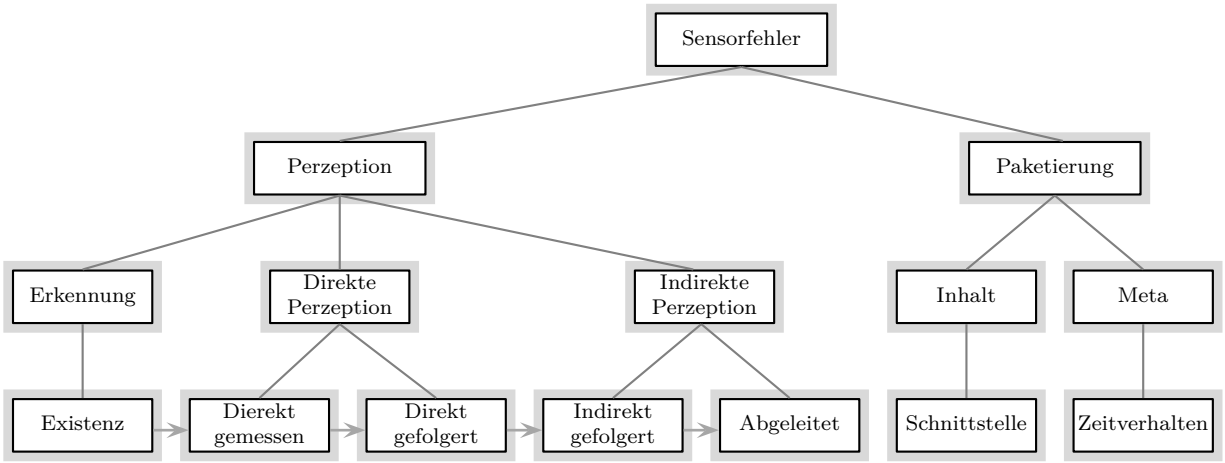


Abbildung 4.3: Klassifizierung von Sensorfehlern nach [26].

eingeteilt, die durch direkte Messung oder direkte Schlussfolgerung zugänglich sind. Ein Beispiel für eine direkte Messung ist die Objektgeschwindigkeit für Sensoren, wie Radar, die den Doppler–Effekt nutzen. Sensoren wie Lidar und Kamera, die ihn nicht verwenden, folgern den aktuellen Wert der Objektgeschwindigkeit aus einer Reihe von Positionsmessungen. Diese Methode wird als direkte Schlussfolgerung bezeichnet. Die indirekte Perzeption wird weiter unterteilt, je nachdem ob die jeweilige Eigenschaft auf das Objekt selbst beschränkt ist oder ein fortgeschrittenes Verständnis der Szene erforderlich ist. Ersteres wird als indirekte Schlussfolgerung bezeichnet. Dies sind Merkmale eines Objekts, die auf der Grundlage probabilistischer Schätzungen abgeleitet werden. Beispiele sind die Klassifizierung eines dynamischen Objekts oder die Schätzung des Bewegungszustands, wie zum Beispiel Bewegung oder Parken. Letzteres erfordert ein umfassenderes Verständnis der Szene über das spezifische Objekt hinaus und wird als

Tabelle 4.1: Fehler Klassifizierung für diverse Messgrößen und Sensorarten nach [26].

Messgröße	Radar	Lidar	Kamera
Erkennung	Existenz	Existenz	Existenz
Position	direkt gemessen	direkt gemessen	direkt gemessen
Geschwindigkeit	direkt gemessen	direkt gefolgert	direkt gefolgert
Beschleunigung	direkt gefolgert	direkt gefolgert	direkt gefolgert
Bounding Box	indirekt gefolgert	direkt gemessen	direkt gemessen
Orientierung	direkt gefolgert	direkt gemessen	direkt gefolgert
Drehrate	direkt gefolgert	direkt gefolgert	direkt gefolgert
Typ / Klasse	indirekt gefolgert	indirekt gefolgert	indirekt gefolgert
Zugehörige Spur	abgeleitet	abgeleitet	abgeleitet
Fahrmanöver	abgeleitet	abgeleitet	abgeleitet
Licht Zustand	—	—	direkt gemessen
Spurmarkierung	—	direkt gefolgert	direkt gefolgert

abgeleitete indirekte Perzeption bezeichnet. Ein Beispiel ist die Zuordnung eines Objekts zu einer Fahrspur, die Informationen über die Objektposition sowie Informationen über Position und Verlauf der Fahrspuren benötigt.

Es gibt eine natürliche Ordnung der Perzeptionsreihenfolge, die durch die Pfeile in Abbildung 4.3 verdeutlicht ist. Bestehen Korrelationen zwischen zwei Effekten, wirken sich die Fehler nur in eine Richtung aus. Eine falsche Positionsmessung kann sich beispielsweise auf die Fahrspurzuordnung auswirken, eine fehlerhafte Zuordnung eines Fahrzeugs zu einer Spur beispielsweise aufgrund von Geisterobjekten hat jedoch keinen Einfluss auf die Positionsmessung.

Bei der Paketierung, das heißt der Enkodierung und Übertragung von Sensorschnittstellendaten, werden zwei Arten von Fehlern unterschieden. Diejenigen, die den Inhalt der Schnittstelle betreffen, aber nicht spezifisch für irgendein Objekt sind, und diejenigen, die die Schnittstelle und die Datenübertragung auf allgemeiner Ebene beeinflussen. Beispiele für erstere sind falsche oder variierende Zeitstempel der Daten, Einschränkungen der Schnittstellenspezifikationen, wie zum Beispiel eine maximale Anzahl von Objekten, die der Sensor aufgrund von Einschränkungen der Nachrichtengröße meldet, oder das Fehlen von Daten, da sich der Sensor zum Beispiel vorübergehend in einem Kalibrierungszustand befindet. Letztere befassen sich hauptsächlich mit den Übertragungszeiten. Unterschiedliche Verarbeitungszeiten der Sensorverarbeitungsfunktionen führen in der Regel zu unregelmäßigen Aktualisierungsintervallen und müssen ebenfalls modelliert werden.

Für die Validierung sind definitionsgemäß nur physikalische Größen relevant. Bezugnehmend auf die vorgestellten Fehlerarten betrifft es die direkt gemessenen und direkt gefolgerten Größen. Auch Paketierungsfehler sind von Relevanz, da diese sich im Allgemeinen auf die Sensormodellierung auswirken. Die direkt gefolgerten und abgeleiteten Perzeptionfehler sind meist kategorischer Natur, wie beispielsweise die Fahrzeugklasse, und nicht Teil dieser Arbeit. Um die relevanten Fehlerarten quantifizieren zu können, werden in diesem Kapitel diverse Metriken aus unterschiedlichen Disziplinen wie Statistik, Robotik, Sprachverarbeitung oder Computer Vision eingeführt. Falls nicht anders erwähnt, bezieht sich der Vergleich bei allen Metriken auf experimentelle $\boldsymbol{\zeta}^{\mathrm{e}}$ und simulierte $\boldsymbol{\zeta}^{\mathrm{s}}$ Messreihen.

Vektornorm ist eine Abbildung, die einem Vektor der Dimension n eine nicht negative reelle Zahl zuordnet. Diese Zahl kann als Vektorlänge interpretiert werden. Mit dieser Eigenschaft kann die Vektornorm als Diskrepanzmaß zweier Vektoren $\boldsymbol{\zeta}^{\mathrm{e}}, \boldsymbol{\zeta}^{\mathrm{s}} \in \mathbb{R}^n$ verwendet werden. Erst wird ein Differenzvektor gebildet und anschließend dessen „Länge“ berechnet. Die formale Definition lautet folgendermaßen,

$$M_{\mathrm{V_p}}(\boldsymbol{\zeta}^{\mathrm{e}}, \boldsymbol{\zeta}^{\mathrm{s}}) = ||\boldsymbol{\zeta}^{\mathrm{e}} - \boldsymbol{\zeta}^{\mathrm{s}}||_p = \left(\sum_{i=1}^{n} |\zeta_i^{\mathrm{e}} - \zeta_i^{\mathrm{s}}|^p \right)^{1/p} \qquad \text{mit } p \in [1, \infty]. \tag{4.1}$$

Aus dieser Definition ergeben sich drei weit verbreitete Distanzmaße: (a) der M_{V_1}–Abstand oder Manhattan–Distanz $M_{\mathrm{V}_1}(\boldsymbol{\zeta}^{\mathrm{e}}, \boldsymbol{\zeta}^{\mathrm{s}}) = \sum_{i=1}^{n} |\zeta_i^{\mathrm{e}} - \zeta_i^{\mathrm{s}}|$, (b) der M_{V_2}–Abstand

oder Euklidische–Distanz $M_{\mathrm{V}_2}(\boldsymbol{\zeta}^{\mathrm{e}}, \boldsymbol{\zeta}^{\mathrm{s}}) = \sqrt{\sum_{i=1}^{n}(\zeta_i^{\mathrm{e}} - \zeta_i^{\mathrm{s}})^2}$ und (c) der M_{V_∞}–Abstand oder Chebyshev–Distanz $M_{\mathrm{V}_\infty}(\boldsymbol{\zeta}^{\mathrm{e}}, \boldsymbol{\zeta}^{\mathrm{s}}) = \max|\zeta_i^{\mathrm{e}} - \zeta_i^{\mathrm{s}}|$. In [84] zeigen die Autoren, wie die Ergebnisse dieser drei Normen zu gegensätzlichen Aussagen und uneindeutiger Modellinterpretation führen können. Der Grund hierfür wird mit der fehlenden Fähigkeit, zwischen Phasen– und Amplituden–Fehler zu unterscheiden, angegeben. Sollten unterschiedliche Normen gegensätzliche Ergebnisse liefern, wird empfohlen, Metriken mit dieser Fähigkeit zur Untersuchung heranzuziehen [84].

Korrelationskoeffizient ist ein Maß zur Beschreibung des linearen Zusammenhangs zweier Vektoren [84]. Dieser wird auch als Pearson–Korrelation bezeichnet und wurde in Gleichung 3.4 im Kontext der Variablenauswahl in Kapitel 3.4.1 bereits eingeführt,

$$M_{\mathrm{P}}(\boldsymbol{\zeta}^{\mathrm{e}}, \boldsymbol{\zeta}^{\mathrm{s}}) = \frac{\sum_{i=1}^{n}\left(\zeta_i^{\mathrm{e}} - \bar{\zeta}^{\mathrm{e}}\right)\left(\zeta_i^{\mathrm{s}} - \bar{\zeta}^{\mathrm{s}}\right)}{\sqrt{\sum_{i=1}^{n}\left(\zeta_i^{\mathrm{e}} - \bar{\zeta}^{\mathrm{e}}\right)^2}\sqrt{\sum_{i=1}^{n}\left(\zeta_i^{\mathrm{s}} - \bar{\zeta}^{\mathrm{s}}\right)^2}}. \tag{4.2}$$

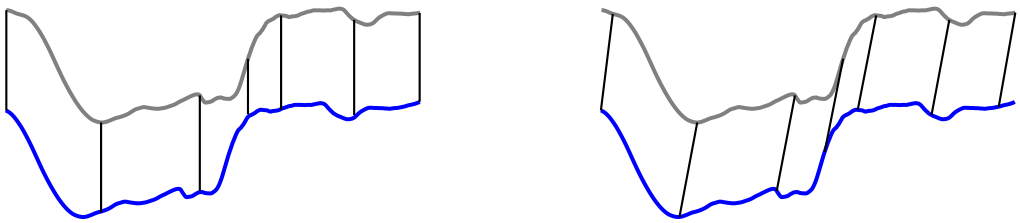
Ein Korrelationskoeffizient kann im Gegensatz zur Vektornorm nur Werte zwischen $+1$ und -1 annehmen, wobei die Maximalwerte vollständigen positiven beziehungsweise negativen linearen Zusammenhang beschreiben. Ähnlich zur Vektornorm reagiert der Korrelationskoeffizient empfindlich auf Phasenunterschiede und kann nicht zwischen Phasen–Fehlern und Amplituden–Fehlern unterscheiden [84].

Kreuzkorrelationskoeffizient ist eine Modifikation des Korrelationskoeffizienten in der Signalverarbeitung. Er kann zur Messung von Phasenverschiebungen zwischen zwei Zeitreihen verwendet werden und ist wie folgt definiert,

$$M_{\mathrm{K}}(\boldsymbol{\zeta}^{\mathrm{e}}, \boldsymbol{\zeta}^{\mathrm{s}}, \tau) = \frac{(n-\tau)\sum_{i=1}^{n-\tau}\zeta_i^{\mathrm{e}}\zeta_{i+\tau}^{\mathrm{s}} - \sum_{i=1}^{n-\tau}\zeta_i^{\mathrm{e}}\sum_{i=1}^{n-\tau}\zeta_{i+\tau}^{\mathrm{s}}}{\sqrt{(n-\tau)\sum_{i=1}^{n-\tau}\zeta_i^{\mathrm{e}2} - \left(\sum_{i=1}^{n-\tau}\zeta_i^{\mathrm{e}}\right)^2}\sqrt{(n-\tau)\sum_{i=1}^{n-\tau}\zeta_i^{\mathrm{s}2} - \left(\sum_{i=1}^{n-\tau}\zeta_i^{\mathrm{s}}\right)^2}} \tag{4.3}$$

wobei n die Länge der Datensätze und $\tau = 0 \ldots n-1$ die Verschiebung beschreiben. Dieser Koeffizient drückt die gemeinsamen Frequenzkomponenten zwischen simulierter und realer Zeitreihen als Funktion der Phasenverschiebung τ aus. Das Maximum liegt bei der zeitlichen Verschiebung τ^* zwischen beiden Messreihen. Korrelationskoeffizienten als Bewertungsmaß legen den Fokus auf die lineare Beziehung zwischen Modell und realen Messwerten. Ein gegen ± 1 strebender Wert weist auf eine hohe positive beziehungsweise negative Korrelation und damit einen linearen Zusammenhang hin.

Dynamic Time Warping (DTW) ist ein Algorithmus zur Messung der Diskrepanz zwischen zwei Zeitreihen, der aus dem Bereich der Spracherkennung kommt [136]. Dieser Algorithmus eignet sich insbesondere, um Zeitreihen unterschiedlicher Längen miteinander zu vergleichen. Die Grundidee ist, durch Zeitverzugstechnik die Spitzen und Täler zweier Signale so anzupassen, dass diese die größte Übereinstimmung ergeben. Dabei werden die Kosten dieser Anpassung berechnet und daraus die Ähnlichkeit zwischen



(a) Euklidisch. **(b)** DTW.

Abbildung 4.4: Visualisierung von (a) Euklidischen Abstand und (b) DTW Berechnung für zwei Zeitreihen.

zwei Zeitreihen abgeleitet. Um den Unterschied zum Euklidischen Abstand zu verdeutlichen, ist ein Beispiel in Abbildung 4.4 dargestellt. Es werden zwei ähnliche, jedoch zueinander verschobene Signale dargestellt, die mittels Euklidischen Abstands und dem DTW–Algorithmus verglichen werden. Im Gegensatz zum Euklidischen Abstand vergleicht der DTW–Algorithmus die Zeitreihen nicht nach Index, sondern berechnet den minimalen Abstand, der durch Verschiebung der beiden Signale gegeneinander finden kann. Der DTW–Algorithmus ist im Anhang A.4 formalisiert.

Der Fokus der bis jetzt aufgeführten Metriken liegt auf dem Vergleich von Absolutwerten der aufgezeichneten Messdaten. Die Sensormodellierung basiert jedoch meist auf Nachbildung von Fehlerverteilungen. Um Diskrepanz zwischen zwei Fehlerverteilungen zu messen, sind beispielsweise Kullback–Leibler–Divergenz (KLD) und die Flächenmetrik geeignet. Der Vergleich bezieht sich dabei auf experimentelle $P^{\mathrm{e}}(\boldsymbol{\zeta}^{\mathrm{e}})$ und simulierte $P^{\mathrm{s}}(\boldsymbol{\zeta}^{\mathrm{s}})$ WDF beziehungsweise experimentelle $C^{\mathrm{e}}(\boldsymbol{\zeta}^{\mathrm{e}})$ und simulierte $C^{\mathrm{s}}(\boldsymbol{\zeta}^{\mathrm{s}})$ KVF.

Kullback–Leibler–Divergenz ist eine ursprünglich aus der Informationstheorie stammende Rechenvorschrift, die der Bestimmung der Diskrepanz zwischen zwei WDF dient. Die KLD ist wie folgt definiert [72],

$$M_{\mathrm{KL}}\big(P^{\mathrm{e}}(\boldsymbol{\zeta}^{\mathrm{e}}), P^{\mathrm{s}}(\boldsymbol{\zeta}^{\mathrm{s}})\big) = \sum_{x \in \{\boldsymbol{\zeta}^{\mathrm{e}} \cap \boldsymbol{\zeta}^{\mathrm{s}}\}} P^{\mathrm{e}}(x) \cdot \Big(\mathsf{log}_2\big(P^{\mathrm{e}}(x)\big) - \mathsf{log}_2\big(P^{\mathrm{s}}(x)\big)\Big) \tag{4.4}$$

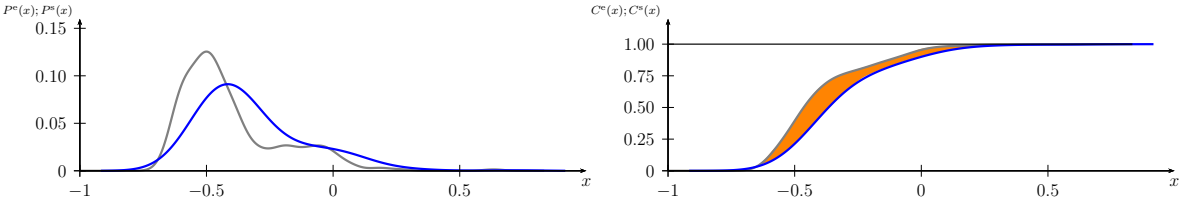
wobei x alle gemeinsamen Werte aus $\boldsymbol{\zeta}^{\mathrm{e}}$ und $\boldsymbol{\zeta}^{\mathrm{s}}$ einnimmt. Aus mathematischer Sicht kann die KLD nicht als Metrik bezeichnet werden, da sie die Symmetrieeigenschaft und die Dreiecksungleichung nicht erfüllt. Trotz alldem ist sie in der Informationstheorie und Mustererkennung sehr weit verbreitet [72]. In der Informationstheorie kann die KLD als die Beschreibung des Informationsverlustes bei Annahme der approximierten Verteilungsfunktion anstatt deren wahrer Korrespondenz interpretiert werden. Der „Mehraufwand“ ist mit den vereinfachenden Annahmen zu begründen, die bei einer simulierten WDF getroffen werden, um die wahre WDF abzubilden. Die, mit log_2 berechnete, KLD hat die Einheit Bit und kann deshalb nicht als einheitslose Prozentangabe interpretiert werden. Es ist deshalb aus Validierungssicht schwierig aus dem KLD–Absolutwert abzuleiten, welche Implikation dies auf ein unter Test stehendes Sensormodell hat. Es

kann jedoch gefolgert werden, dass mit kleiner werdendem KLD–Wert das Verhalten eines realen Sensors mit dem jeweiligen Modell immer besser abgebildet wird.

Flächenmetrik ist eine Metrik zum Vergleich von zwei WDF basierend auf deren KVF [72]. Diese ist wie folgt definiert,

$$M_{\mathrm{F}}\big(C^{\mathrm{e}}(\boldsymbol{\zeta}^{\mathrm{e}}), C^{\mathrm{s}}(\boldsymbol{\zeta}^{\mathrm{s}})\big) = \sum_{x \in \{\boldsymbol{\zeta}^{\mathrm{e}} \cap \boldsymbol{\zeta}^{\mathrm{s}}\}} |C^{\mathrm{e}}(x) - C^{\mathrm{s}}(x)| \tag{4.5}$$

wobei C^{e} und C^{s} entsprechend die experimentellen und simulierten diskreten KVF repräsentieren und x alle gemeinsamen Werte aus $\boldsymbol{\zeta}^{\mathrm{e}}$ und $\boldsymbol{\zeta}^{\mathrm{s}}$ einnimmt. Ausgehend von realen $\boldsymbol{\zeta}^{\mathrm{e}}$ und simulierten $\boldsymbol{\zeta}^{\mathrm{s}}$ Messdaten werden empirische KVF generiert und mittels dieser Metrik bewertet. Dadurch wird die Summe der Differenzen zweier KVF als Maß für deren Übereinstimmung definiert. Das Prinzip dieser Berechnungsvorschrift ist in Abbildung 4.5 visualisiert. Die orange hinterlegte Fläche entspricht der Diskrepanz zweier KVF. Das Ergebnis wird nicht normiert, weshalb beliebig hohe Diskrepanzwerte resultieren können. Ähnlich wie bei der KLD kann der Absolutwert deshalb kaum für Modellbewertung verwendet werden, ein Modell mit dem niedrigsten Metrikwert kann jedoch als bestmögliches angesehen werden. Die Flächenmetrik kann auch zum Vergleich mehrerer Modellierungsgrößen für höhere Dimensionen erweitert werden [81].



(a) WDF. **(b)** KVF.

Abbildung 4.5: Visualisierung der Berechnungsschritte der Flächenmetrik. In (a) werden die experimentelle $P^{\mathrm{e}}(x)$ (grau) und simulierte $P^{\mathrm{s}}(x)$ (blau) WDF und daraus in (b) die KVF $C^{\mathrm{e}}(x)$ und $C^{\mathrm{s}}(x)$ berechnet. Anschließend wird die Fläche zwischen den KVF als Maß für die Modellgüte bestimmt (hier orange hervorgehoben).

Jaccard–Metrik wird als Ähnlichkeitsmaß im Bereich der Mustererkennung verwendet. Diese kann durch die Bestimmung der Schnittfläche zweier Fahrzeug Bounding–Boxen gemittelt über die Datensatzlänge n berechnet werden [137],

$$M_{\mathrm{J}} = \frac{1}{n} \sum_{i=1}^{n} k_{\mathrm{J}_i} \qquad \text{mit } k_{\mathrm{J}} = \begin{cases} 1 - \frac{|A_{\mathrm{e}} \cap A_{\mathrm{s}}|}{|A_{\mathrm{e}} \cup A_{\mathrm{s}}|} \\ 0 \text{ für } |A_{\mathrm{e}} \cup A_{\mathrm{s}}| = 0 \end{cases} \tag{4.6}$$

Damit ergibt sich ein Wert, der das Gesamtverhalten des Sensormodells beurteilt. Dabei fließen in die Bewertung sowohl die Position in x und y Richtung, als auch die Objektdimensionen wie Länge und Breite sowie die Fahrzeugausrichtung mit ein. Abbildung 4.6 visualisiert diese Berechnungsmethode. Zu jedem Zeitpunkt i der aufgezeichneten Trajektorie werden die Schnittflächen der experimentellen Objekte A_{e} und der simulierten

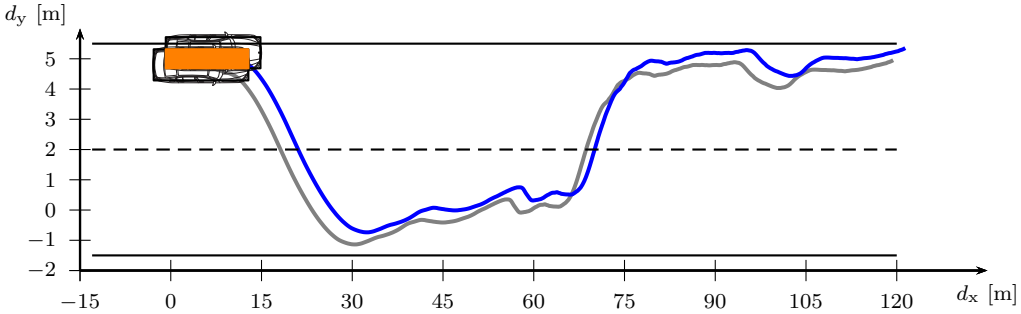


Abbildung 4.6: Visualisierung der Berechnung der Jaccard–Metrik. Grau: reale Trajektorie, Blau: simulierte Trajektorie, Orange: Schnittfläche zwischen Realität und Simulation.

Objekte A_s berechnet und in Gleichung 4.6 eingefügt. Der Wert der Jaccard–Metrik ist um so kleiner, je ähnlicher zwei Trajektorien sind.

Die hier vorgestellten Metriken und Diskrepanzmaße eignen sich für unterschiedliche Aufgaben und werden im folgenden Kapitel zur Eignung der Quantifizierung von phänomenologischen Sensormodellfehlern untersucht.

4.1.2 Umsetzung und Auswertung des Verfahrens

Bezugnehmend auf den Architekturvorschlag aus Abbildung 3.2, sind für die Validierung phänomenologischer Sensormodelle nicht alle Aspekte relevant. Um dies zu verdeutlichen, werden in Abbildung 4.7 die nicht relevanten Anteile ausgegraut. Als Erstes werden Referenz–, Umfeldsensor– und Umfeldmodelldaten erfasst, anschließend daraus die Fahrtrajektorien der relevanten Teilnehmer extrahiert und resimuliert. Als Ergebnis der Resimulation erhält man simulierte Referenztrajektorien, welche als Grundlage für die Simulation des Sensorverhaltens verwendet werden. Das dabei eingesetzte Sensormodell wurde im vorherigen Abschnitt vorgestellt. Um Überanpassung zu vermeiden, werden während der Simulation nur diejenigen Datensätze verwendet, die nicht unter

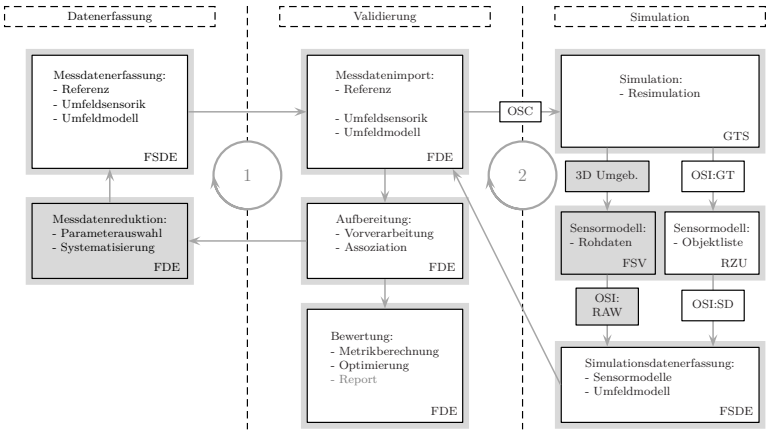


Abbildung 4.7: Relevante Anteile aus der Gesamtarchitektur für die Validierung phänomenologischer Sensormodelle. Irrelevante Anteile sind ausgegraut.

Test stehen. Im Anschluss werden simulierte Sensordaten im Umfeldmodell fusioniert. Reale und synthetische Datensätze werden anschließend in das Evaluierungsframework importiert und assoziiert (siehe Kapitel 3.3.2). Abschließend findet die Auswertung statt. Die beschriebenen Schritte sind für alle Auswertungen in Kapitel 4 gültig.

Grundsätzlich eignen sich alle vorgestellten Metriken, bis auf die Jaccard–Metrik, zur Bewertung beliebiger reeller Größen, weshalb diese auch auf andere reelle Größen, als die in Tabelle 4.1 beschrieben, anwendbar sind. Die Jaccard–Metrik ist dagegen nur bei Abstands– und Orientierungsmodellierung sinnvoll, da diese die Schnittfläche zweier Bounding–Boxen, basierend auf Abstand beziehungsweise Orientierung, misst. Als Modellierungsgrößen werden deshalb hier die Abstände d_{x} in x–Richtung und d_{y} in y–Richtung festgelegt. Um die Eignung der vorgestellten Metriken, Modellierungsfehler zu bestimmen, werden im ersten Schritt drei verschiedene Fehlerarten untersucht:

- Systematische Translationsfehler, die beispielsweise durch eine falsch eingestellte Sensorpositionen entstehen,
- Signalverschiebungen aufgrund von Netzwerklatenzen und
- Falsch modellierte Fehlerverteilung, herbeigeführt beispielsweise durch inkorrekt eingestelltes Signal–Rausch–Verhältnis (SRV).

Damit sollen mögliche Fehlkonfigurationen eines Sensormodells nachgestellt werden. Die verwendeten Parameter sind in Tabelle 4.2 zusammengefasst. Die einzelnen Spalten in Tabelle 4.2 repräsentieren die Sensormodellkonfigurationen, mit denen die aufgelisteten Sensorfehler untersucht und in unterschiedlichen Durchläufen einzeln getestet werden. Zeilenweise ist die Tabelle in modellunabhängige und –abhängige Parameter aufgeteilt. Zu den modellunabhängigen Parametern zählen der Sichtbereich und die unter Test stehenden Modellierungsgrößen, die für alle Konfigurationen gleich bleiben.

Zur Auswertung der aufgelisteten Fehler werden unterschiedliche Datensätze generiert. In der ersten Konfiguration wird die Sensorposition des Sensors im Simulator in Bezug auf die reale Sensorposition um 10cm in positive x– und y–Richtung verschoben. Dazu werden die realen Sensordaten $\boldsymbol{\zeta}^{\mathrm{e}}$ erst in einen separaten Datensatz kopiert, die

Tabelle 4.2: Verwendete Konfigurationen für die Messfahrt basierte Analyse.

	Parameter	Konfig. 1	Konfig. 2	Konfig. 3	Konfig. 4	Konfig. 5
Konfiguration–unabhängig	Abstandsreichweite d in m			$2\ldots200$		
	Horizontaler Sichtbereich γ in °			360		
	Modellierungsgrößen $\boldsymbol{\omega}_m$			$d_{\mathrm{x}}^{\mathrm{r}}, d_{\mathrm{y}}^{\mathrm{r}}$		
	Zielgrößen $\boldsymbol{\zeta}_{\mathrm{s}}$			$d_{\mathrm{x}}, d_{\mathrm{y}}$		
Konfiguration–spezifisch	Verschiebung (x;y) in m	(0,1; 0,1)	(0; 0)	(0; 0)	(0; 0)	(0; 0)
	Latenz in ms	0	200	0	0	0
	Rauschen in dB	0	0	5	15	10

entsprechende Translation in x– und y–Richtung vorgenommen und der Analyse als simulierte Messdaten $\boldsymbol{\zeta}^{\mathrm{s}}$ übergeben.

Anschließend werden mögliche Netzwerkfehler untersucht. Dazu wird im zweiten Durchlauf eine Netzwerklatenz τ von 200ms eingestellt. Da bis auf den DTW–Algorithmus alle anderen Metriken und Diskrepanzmaße Vektoren gleicher Länge am Eingang erwarten, werden aus dem realen Datensatz $\boldsymbol{\zeta}^{\mathrm{e}}$ die letzten und aus dem simulierten Datensatz $\boldsymbol{\zeta}^{\mathrm{s}}$ die ersten $\lfloor \tau \cdot f_{\mathrm{S}} \rfloor$ Werte vernachlässigt, wobei f_{S} die Sensorfrequenz ist. Bei diesem Vorgang werden einige Messwerte verworfen, was jedoch nicht als kritisch zu betrachten ist, da beispielsweise eine 60–sekündige Fahrt eines Lidar–Sensors mit einer Abtastfrequenz von 25Hz etwa 1500 Werte beinhaltet und davon nur 5 Werte entfernt werden.

In den letzten drei Durchläufen wird untersucht, ob die verwendeten Metriken in der Lage sind, ähnliche Fehlerverteilungen zu detektieren. Dazu werden drei synthetische Messfahrten generiert. Dabei wird auf eine reale Referenztrajektorie Rauschen mit einem SRV von 10dB addiert. Anschließend werden drei Simulationsdurchläufe durchgeführt, wobei das Rauschverhalten des Sensormodells entsprechend den Konfigurationen 3–5 variiert wird. Mit anderen Worten wird die Fehlerverteilung, die ein Sensormodell simulieren soll, variiert und untersucht, ob die bestmögliche Realisierung erkannt wird. In Konfiguration 3 wird zu hohes, in Konfiguration 4 zu niedriges und in Konfiguration 5 ein passender Wert für das Rauschen eingestellt.

Die Ergebnisse der Auswertung sind beispielhaft für einen Lidar–Sensor und die Zielgröße d_{x} in Tabelle 4.3 gezeigt. Die entsprechenden Resultate für die y–Richtung sind im Anhang A.5 zusammengefasst. Anhand der ersten Modellkonfiguration, bei der die Sensorposition um 10cm verschoben wird, ist zu erkennen, dass die Vektornorm M_{V_∞} und die Flächenmetrik M_{F} den Absolutwert der Positionsverschiebung einnehmen. Die Korrelationskoeffizienten und die KLD können hingegen solche Offsetfehler nicht erkennen. Basierend auf diesen Erkenntnissen, können Verschiebungsfehler also gut erkannt werden, wenn sich M_{V_∞} und M_{F} zwischen zwei Modellkonfigurationen um einen ähnlichen Wert ändern.

Tabelle 4.3: Ergebnisse für synthetische Datensätze eines phänomenologischen Lidar–Sensormodell in x–Richtung.

Metrik	Konfig. 1	Konfig. 2	Konfig. 3	Konfig. 4	Konfig. 5
M_{V_1}	88,300	585,02	444,36	260,36	318,41
M_{V_2}	2,9715	21,400	18,642	11,041	13,458
M_{V_∞}	0,1000	1,1037	1,9137	1,4276	1,3480
M_{P}	1,0000	0,9999	0,9998	0,9999	0,9999
M_{K}	1,0000	0,9999	0,9999	0,9999	0,9999
M_{DTW}	45,241	4,8183	303,76	183,51	204,57
M_{KL}	0,0000	0,0998	0,0103	0,1219	0,0069
M_{F}	0,0999	0,6657	0,2055	0,1167	0,0175
M_{J}	0,1384	0,3087	0,6213	0,5009	0,4501

Die Ergebnisse der zweiten Modellkonfiguration zeigen, dass in diesem Fall die Netzwerklatenzen gut detektierbar sind, wenn der M_{DTW}–Wert sich nur marginal ändert und gleichzeitig die Flächenmetrik–Werte sich deutlich zwischen den zwei Modellen unterscheiden. Grund für das Erste ist, dass der DTW–Algorithmus speziell für solche Fälle konzipiert wurde, um nicht sensitiv auf Zeitverschiebungen zu reagieren [136]. Flächenmetrik, auf der anderen Seite, reagiert sehr sensitiv auf Mittelwert– und somit auch auf Zeitverschiebungen [72]. Die Korrelationskoeffizienten ändern sich, wie bei der ersten Modellkonfiguration, dagegen kaum. Ähnliches Verhalten ist für alle fünf Tests zu beobachten. Dies liegt daran, dass die realen und simulierten Messwerte sich über den ganzen Messbereich sehr ähneln und somit hohe lineare Korrelation aufweisen. Die Korrelationskoeffizienten eignen sich demzufolge nur, um grobe Fehler zu erkennen und kaum für die Feinanpassung eines Sensormodells.

Die Ergebnisse der letzten drei Simulationsdurchläufe (Modellkonfigurationen 3,4 und 5 aus Tabelle 4.2) zeigen, dass die Güte der Ähnlichkeit von Fehlerverteilungen mit der Kullback–Leibler–Divergenz M_{KL} und der Flächenmetrik M_{F} gut messbar ist. Die Modellkonfiguration 5 wird laut dieser Metriken als die am ähnlichsten zu der vorgegeben Fehlerverteilung mit einem SRV von 10db richtig bewertet. Lässt man die Korrelationskoeffizienten außer Acht, reagieren andere Metriken im Allgemeinen mit höheren Werten auf das Rauschen und sind nur als Richtwert zu betrachten. Mit anderen Worten, wenn beispielsweise der M_{DTW}–Wert hoch ist, heißt es nicht unbedingt, dass die Fehlerverteilungen sich deutlich unterscheiden. Dadurch, dass eine Fahrtrajek-

Tabelle 4.4: Ergebnisse für reale Datensätze für phänomenologische Lidar– und Kamera–Sensormodelle sowie für die Fusion in x–Richtung.

	Metrik	Konfig. 6	Konfig. 7	Konfig. 8
Lidar	M_{V_1}	344,63	133,58	182,95
	M_{V_2}	14,861	6,4261	8,0743
	M_{V_∞}	1,9337	1,1623	1,3070
	M_{DTW}	257,92	56,684	39,124
	M_{KL}	0,6638	1,9766	0,6546
	M_{F}	0,2618	0,0682	0,0279
Kamera	M_{V_1}	1296,3	1223,3	1264,0
	M_{V_2}	42,914	39,419	40,516
	M_{V_∞}	3,0292	2,5563	2,4873
	M_{DTW}	399,88	69,299	107,34
	M_{KL}	0,2285	1,8896	0,2278
	M_{F}	0,3279	0,6229	0,5596
Fusion	M_{V_1}	1650,2	3468,8	2467,8
	M_{V_2}	68,739	156,86	108,14
	M_{V_∞}	5,1311	10,457	7,5722
	M_{DTW}	246,07	523,29	329,07
	M_{KL}	0,0931	0,0915	0,0728
	M_{F}	0,1927	0,0793	0,0321

torie aus einer Referenz und einem Rauschsignal gebildet wird, unterscheiden sich die gezogenen Realisierungen der Fehlerwerte von Durchlauf zu Durchlauf, auch wenn die Fehlerverteilung sich dabei nicht ändert. Als Resultat können sich die Werte solcher Metriken für unterschiedliche Durchläufe deutlich unterscheiden.

Aus dem zuvor Erwähnten lässt sich ableiten, dass, obwohl sich zwei Fehlerverteilungen ähneln, die Absolutwerte zweier Trajektorien sich deutlich unterscheiden können. Dies ist ein Indiz für unkorreliertes Rauschen. Verdeutlicht wird es in Tabelle 4.4. Dort sind Ergebnisse von drei weiteren Sensormodellkonfigurationen zusammengefasst. Es werden Testdurchläufe mit Konfigurationen 6, 7 und 8 durchgeführt, wobei dort die Einstellung der Relevanzkernebreite h_{R} des nicht–parametrischen Sensormodells zwischen $\mathbf{h} = \{0{,}1; 0{,}5; 1\}$ entsprechend variiert und die Ergebnisse mit einer Realfahrt verglichen werden. Die erste Konfiguration mit $h = 1$ führt zu größerer, die zweite mit $h = 0{,}5$ zu mittlerer und die dritte mit $h = 0{,}1$ zu kleiner Streuung der Fehlerwerte. In der Tabelle 4.4 sind die Ergebnisse für Lidar– und Kamera–Sensoren sowie für die Fusion zusammengefasst. Basierend auf den Werten der KLD und der Flächenmetrik wird das Modell mit Konfiguration 8 für alle Sensoren am besten bewertet. Interessant an dieser Stelle ist, dass für die anderen Metriken, vor allem bei der Fusion, die Werte vergleichsweise hoch ausfallen. Dies lässt sich damit begründen, dass der Ausgang von Sensoren und insbesondere der Fusion zeitlich gefiltert wird. Jedoch ist das bei den getesteten Modellkonfigurationen nicht der Fall. So führt es dazu, dass die vergleichsweise „glatten“ Trajektorien der Sensoren und insbesondere der Fusion mit zufällig verrauschten Fehlerwerten aus dem Modell verglichen werden. Diese Tatsache deutet auf fehlende zeitliche Filterung von Messwerten in einem Sensormodell und somit die Notwendigkeit der Autokorrelationsanalyse, die im nächsten Kapitel vorgestellt wird.

Schließlich sind die Ergebnisse der Jaccard–Metrik aus Tabelle 4.3 zu erwähnen. Neben den Korrelationskoeffizienten ist sie die einzige Metrik aus den hier aufgelisteten, die einen absoluten Wertebereich zwischen $0 - 1$ hat. Da die Jaccard–Metrik die x– und y–Position sowie die Fahrzeugbreite, Länge und Ausrichtung in Betracht zieht, kann sie alle hier untersuchten Fehler erkennen, jedoch die Güte eines Modells nur allgemein bewerten. Diese Metrik eignet sich deshalb gut, um einen groben Eindruck über Modellgüte zu erhalten, reicht jedoch nicht alleine, um die einzelnen Fehlerfälle zu detektieren.

4.2 Autokorrelationsanalyse

Im Allgemeinen ist für die Validierung unter anderem interessant, ob ein Modell in der Lage ist, die dynamischen Zeitmuster eines realen Systems zu reproduzieren. So werden beispielsweise bei der Validierung eines Sensormodells die vom Modell erzeugten Fahrzeugtrajektorien mit denen verglichen, die von realen Sensoren erzeugt werden. Die Fahrtrajektorien der beobachteten Objekte werden in der zuvor angesprochenen Signalverarbeitung zeitlich verfolgt. Die Aufgabe der zeitlichen Verfolgung besteht darin, aus Beobachtungen eines Sensors die relevanten Größen zu schätzen. Hierfür existieren unterschiedliche Ansätze, wie beispielsweise der Partikel–Filter oder der Kalman–

Filter [16]. Ein im Automobilbereich sehr oft eingesetztes Verfahren für solche Aufgaben ist der Kalman–Filter [138]. Problemabhängig gibt es unterschiedliche Varianten des Kalman–Filters, wie zum Beispiel der Extended–Kalman–Filter oder Unscented–Kalman–Filter. Im Rahmen dieser Arbeit wird jedoch nur das für die Autokorrelationsanalyse relevante Basiskonzept eingeführt. Ein Kalman–Filter besteht aus zwei alternierenden Schritten. Als Erstes wird eine Größe, zum Beispiel der Abstand zum Vorderfahrzeug gemessen und im nächsten Schritt prädiziert. Die beiden Schritte heißen entsprechend Mess– beziehungsweise Korrekturschritt und Prädiktion. Der Korrekturschritt basiert auf den vom Sensor erfassten Daten, die Prädiktion hingegen auf einem problemspezifischen Modell. Im Fall der Abstandsschätzung wird beispielsweise die Fahrzeugbewegung anhand von Bewegungsgleichungen modelliert. Es wird somit zyklisch ein gewichteter Mittelwert zwischen Messung und Prädiktion berechnet. Wird beispielsweise die Prädiktion stärker gewichtet, so wird der prädizierten Schätzung aus dem letzten Zyklus mehr „vertraut“ als der neuen Messung. Als Ergebnis erhält man in der Regel einen relativ glatten Signalverlauf, da die Messungen und darin enthaltenes Rauschen entsprechend niedriger gewichtet werden [139]. Das heißt, dass die vom Sensor ausgegebenen und in einer Objektliste $\mathcal{O}$ enthalten Größen (siehe Abbildung 2.3) eine zeitliche Korrelation mit sich selbst aufweisen. In Statistik und Signalverarbeitung wird dieses Verhalten als Autokorrelation bezeichnet.

4.2.1 Vorstellung des Ansatzes

Autokorrelation beschreibt die Korrelation eines Signals mit sich selbst zu einem früheren Zeitpunkt. Es wird dabei gemessen, wie ähnlich sich eine von der Zeit t abhängige Zufallsvariable $x^{(t)}$ ist, wenn man sie um die Zeit τ verschiebt. Die Stichproben–Autokorrelationsfunktion (AKF) ist wie folgt definiert,

$$M_{\mathrm{A}}(\tau) = \frac{\mathrm{cov}(x^{(t)}, x^{(t+\tau)})}{\mathrm{cov}(x^{(t)}, x^{(t)})} = \frac{\sum_{t=0}^{n-\tau}(x^{(t)} - \bar{x})(x^{(t+\tau)} - \bar{x})}{\sum_{t=0}^{n}(x^{(t)} - \bar{x})^2} \qquad \text{für } \tau = 0, 1, 2 \cdots < n \tag{4.7}$$

Dabei bezeichnet cov() die Stichprobenkovarianz und n die Stichprobengröße. Der Normierungsfaktor $\mathrm{cov}(x^{(t)}, x^{(t)})$ im Nenner entspricht der Varianz zur Zeitverzögerung $\tau = 0$, sodass die Funktion bei keiner Verschiebung Werte ≥ 1 annehmen kann. Für alle Werte von τ ungleich 0 liegt diese zwischen $-1 \cdots +1$ und geht gegen 0 für $\tau \to \infty$. Für eine zufällige Stichprobe würde $M_{\mathrm{A}}(\tau)$ für $\tau > 1$ sehr schnell auf einen Wert nahe 0 abfallen [85].

In [140] wird vorgeschlagen, die Messung der Autokorrelation zur Quantifizierung des Zeitverhaltens eines Modells zu verwenden und die Entscheidung, ob das Zeitverhalten einer simulierten Fahrt der realen ähnlich ist, basierend auf einem Konfidenzintervall um die Verteilung der experimentellen AKF, zu treffen. Das Konfidenzintervall kann mit Hilfe des Standardfehlers ermittelt werden, der wie folgt definiert ist

$$\sigma_{\mathrm{K}} = \sqrt{\mathrm{var}\big(M_{\mathrm{A_e}}(\tau)\big) + \mathrm{var}\big(M_{\mathrm{A_s}}(\tau)\big)}, \tag{4.8}$$

wobei var() für Varianz steht. $M_{\mathrm{A_e}}(\tau)$ und $M_{\mathrm{A_s}}(\tau)$ bezeichnen die experimentellen und simulierten AKF. Somit müssen zur Bewertung der Autokorrelationsübereinstimmung zwischen Messwerten und Modellwerten die Varianzen von $M_{\mathrm{A_e}}(\tau)$ und $M_{\mathrm{A_s}}(\tau)$ geschätzt werden. In [141] schlägt der Autor einen passenden Ansatz vor, wie die Varianz $\mathrm{var}\big(M_{\mathrm{A}}(\tau)\big)$ für eine Stichprobe der Länge n geschätzt werden kann,

$$\mathrm{var}\big(M_{\mathrm{A}}(\tau)\big) = \frac{1}{n(n+2)} \sum_{i=0}^{n-\tau} (N-i)\Big(M_{\mathrm{A}}(\tau-i) + M_{\mathrm{A}}(\tau+i) - 2 \cdot M_{\mathrm{A}}(\tau) \cdot M_{\mathrm{A}}(i)\Big). \quad (4.9)$$

Unter Verwendung des Varianzschätzers aus Gleichung 4.9 in Gleichung 4.8 kann ein Hypothesentest definiert werden,

$$\begin{aligned} H_0 &: d(1) = M_{\mathrm{A_e}}(1) - M_{\mathrm{A_s}}(1) \approx 0, \ldots, d(k) = M_{\mathrm{A_e}}(\tau) - M_{\mathrm{A_s}}(\tau) \approx 0 \quad \forall\, \tau \\ H_1 &: d(p) = M_{\mathrm{A_e}}(\tau) - M_{\mathrm{A_s}}(\tau) \neq 0 \quad \exists\, \tau. \end{aligned} \quad (4.10)$$

Die Nullhypothese H_0 gilt, wenn sich die Werte der experimentellen AKF $M_{\mathrm{A_e}}$ und der simulierten $M_{\mathrm{A_s}}$ so ähnlich sind, dass sich die Differenzen $d(\tau) = M_{\mathrm{A_e}}(\tau) - M_{\mathrm{A_s}}(\tau)$ für alle Werte von τ nicht außerhalb des Konfidenzintervalls liegen. Entsprechend folgt die Gegenhypothese H_1, falls es wenigstens ein τ gibt, für das sich $d(\tau)$ außerhalb des Konfidenzintervalls befindet. Als letztes muss das Signifikanzniveau des Konfidenzintervalls definiert werden, welches auf dem Standardfehler σ_{K} aus Gleichung 4.8 basiert. Dieses ist anwendungsspezifisch festzulegen und wird für die Autokorrelationsanalyse in dieser Arbeit auf $\alpha = 0{,}05$ eingestellt. Unter der Normalitätsannahme, die für praktische Anwendungen nach [140] plausibel ist, liegt das Signifikanzniveau des Konfidenzintervalls für ein zweiseitigen Test und $\alpha = 0{,}05$ zwischen $-1{,}96 \cdot \sigma_{\mathrm{K}}$ und $+1{,}96 \cdot \sigma_{\mathrm{K}}$ [129].

4.2.2 Umsetzung und Auswertung des Verfahrens

Zur Auswertung der Autokorrelation wird ein autobahnähnliches Szenario auf einem Testgelände untersucht. Es handelt sich um ein Überholmanöver, wobei das Eigenfahrzeug von einem Zielfahrzeug überholt wird. Die relative Fahrtrajektorie des Zielfahrzeugs ist schematisch in Abbildung 4.8 dargestellt. Auf der x–Achse ist der laterale und y–Achse der longitudinale Abstand eingetragen. Das Eigenfahrzeug befindet sich dort auf der Position (0,0) und die relative Trajektorie des Zielfahrzeugs ist in grau dargestellt. Als Erstes folgt das Überholmanöver mit anschließendem Fahrstreifenwechsel. Zum späteren Zeitpunkt wechselt das Zielfahrzeug wieder auf die linke Spur und entfernt sich. Der gesamte Überholvorgang dauert etwa 60s und die Geschwindigkeiten der beiden Fahrzeuge variieren während des Szenarios zwischen circa 65–120$\frac{\mathrm{km}}{\mathrm{h}}$.

Im Rahmen der Auswertung werden Lidar– und Kamera–Sensormodelle verwendet, deren Ergebnisse im Umfeldmodell fusioniert werden. Wie einleitend in Kapitel 4 erwähnt, basiert die Modellierung auf Aufzeichnungen realer Sensoren. Abgesehen von den zugrunde liegenden Datensätzen, sind beide Modelle gleich parametriert: Es werden der longitudinale Abstand d_{x} und der laterale Abstand d_{y} als Modellierungsgrößen

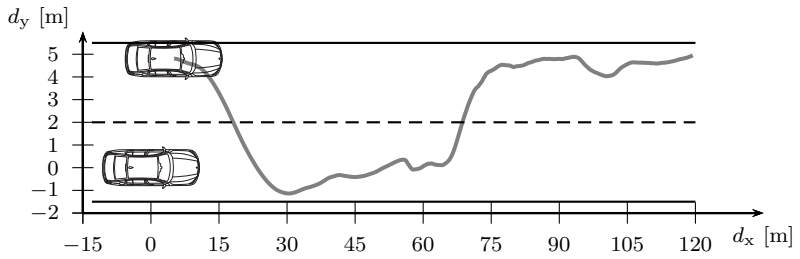


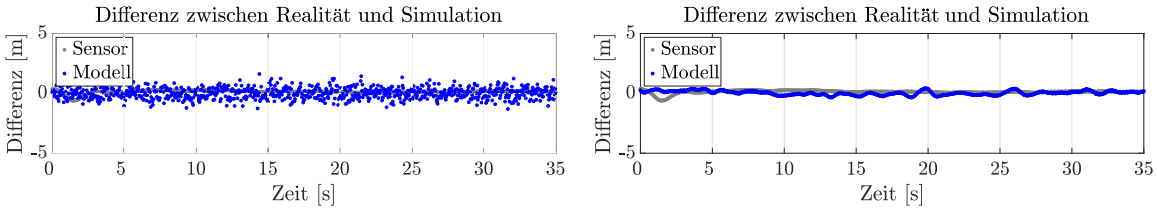
Abbildung 4.8: Visualisierung der für die Analysen verwendeten relativen Fahrtrajektorie des Zielfahrzeugs. Das Eigenfahrzeug befindet sich bei (0,0). Die Trajektorie des Zielfahrzeugs ist in grau dargestellt.

genommen. Die Sensorsichtbereiche sind wie bei einem realen Sensorverbund auf 360° horizontale Abdeckung und 200m Abstandsreichweite festgelegt. Um den Einfluss der zeitlichen Verfolgung zu untersuchen, werden für jeden Sensor (Lidar und Kamera) zwei Modelle untersucht: (a) mit *inaktiver* und (b) mit *aktiver* zeitlicher Verfolgung der Objekte. Die relevanten Parameter sind in Tabelle 4.5 zusammengefasst.

In Abbildung 4.9 ist beispielhaft ein Ausschnitt der aufgezeichneten Messdatendifferenzen für den longitudinalen Abstand des Lidar–Sensors gezeigt. Auf der linken Seite (Abbildung 4.9a) sind die Ergebnisse zu Konfiguration 1 mit *inaktiver* Objektverfolgung und rechts (Abbildung 4.9b) mit *aktiver* Objektverfolgung dargestellt. Zu erkennen ist, dass bei der Konfiguration 1 in blau die Differenz von Referenz einen Rauschcharakter aufweist, wohingegen die Sensorabweichung von Referez in grau eher nach einem zeitlich gefiltertem Signal aussieht. Bei Konfiguration 2 (Abbildung 4.9b) ähneln beide Differenzen einem zeitlich gefilterten Signal. Ob die Modellergebnisse akzeptabel sind und das Umfeldmodell damit die erwünschten Ergebnisse liefern kann, wird im Folgenden untersucht.

Tabelle 4.5: Verwendete Parameter für die Autokorrelationsanalyse.

	Parameter	Konfig. 1 {Lidar, Kamera}	Konfig. 2 {Lidar, Kamera}
Sicht–bereich	Abstandsreichweite d	2m . . . 200m	
	Horizontaler Sichtbereich γ	360°	
AKF–Test	Signifikanzniveau α	0,05	
	Konfidenzintervall	$\pm 1{,}96 \cdot \sigma_{\mathrm{K}}$	
Modell–spezifisch	Modellierungsgrößen $\boldsymbol{\omega}_m$	$d_{\mathrm{x}}^{\mathrm{r}}, d_{\mathrm{y}}^{\mathrm{r}}$	
	Zielgrößen $\boldsymbol{\zeta}_{\mathrm{s}}$	$d_{\mathrm{x}}, d_{\mathrm{y}}$	
	Objektverfolgung	inaktiv	aktiv

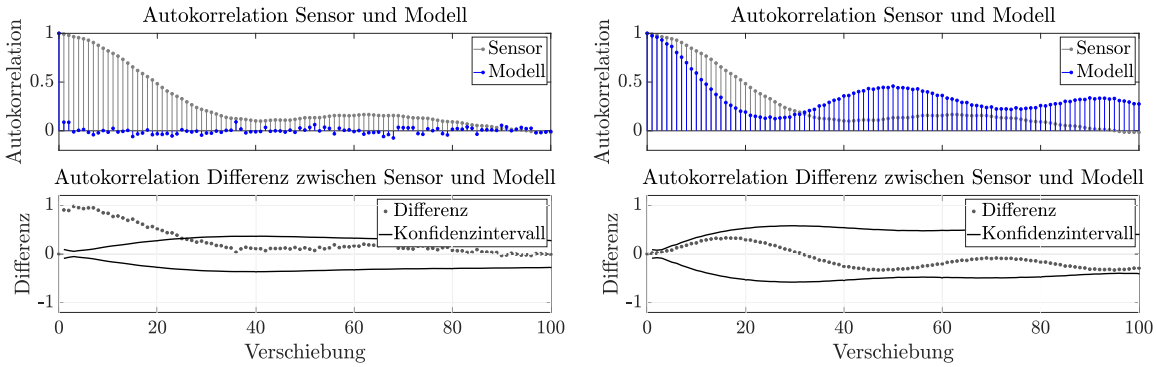


(a) Ergebnisse zu Konfig. 1. **(b)** Ergebnisse zu Konfig 2.

Abbildung 4.9: Aufgezeichnete Differenzen eines Lidar–Sensors in longitudinaler Richtung. In (a) sind Ergebnisse zu Konfiguration 1 mit *inaktiver* zeitlicher Verfolgung und in (b) entsprechende Ergebnisse mit *aktiver* Objektverfolgung zu Konfiguration 2 aufgezeigt.

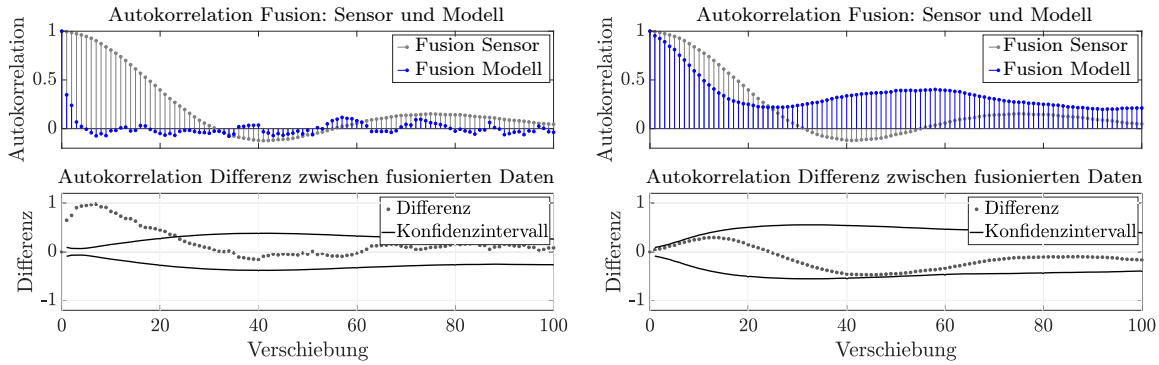
Zunächst werden die Autokorrelationseigenschaften der Sensormodelle und anschließend deren fusionierte Ergebnisse bewertet. Dazu werden für beide Sensoren und entsprechende Modelle {Lidar, Kamera} sowie Modellkonfigurationen {1, 2} die AKF berechnet. Anschließend wird die Differenz zwischen realer und simulierter AKF gebildet und, basierend auf den Konfidenzintervallen aus Gleichung 4.8, die Akzeptanz der Autokorrelationseigenschaften eines Modells bestimmt. Exemplarische Ergebnisse für den longitudinalen Abstand eines Lidar–Sensors sind in Abbildung 4.10 dargestellt. Auf der linken Seite (Abbildung 4.10a) sind wieder die Resultate für Konfiguration 1, auf der rechten Seite (Abbildung 4.10b) für Konfiguration 2 präsentiert. Zu erkennen ist, dass Konfiguration 1 keine Autokorrelation aufweist und deshalb die Differenz zwischen Realität und Simulation außerhalb der Konfidenzintervalle fällt. Bei der Konfiguration 2 mit aktiver zeitlicher Objektverfolgung sind die AKF zwar nicht identisch, die Differenz bewegt sich jedoch innerhalb der Konfidenzintervalle und wäre aus der Sicht der Autokorrelationsanalyse akzeptabel.

Die Frage, die sich als letztes stellt, ist, wie die Fusion auf die modellierten Größen der Sensormodelle mit Konfigurationen 1 und 2 reagiert. Die Ergebnisse für fusionierte



(a) Ergebnisse zu Konfig 1. **(b)** Ergebnisse zu Konfig 2.

Abbildung 4.10: Visualisierung der Autokorrelation eines Lidar–Sensors in longitudinaler Richtung. In (a) sind Ergebnisse zu Konfiguration 1 mit *inaktiver* zeitlicher Verfolgung und in (b) entsprechende Ergebnisse mit *aktiver* Objektverfolgung zu Konfiguration 2 gezeigt. Oben sind die Autokorrelationsfunktionen und unten die jeweiligen Abweichungen zwischen Sensor und Modell dargestellt.



(a) Ergebnisse zu Konfig 1.

(b) Ergebnisse zu Konfig 2.

Abbildung 4.11: Visualisierung der Autokorrelation der fusionierten Sensor– und Modelldaten für Lidar und Kamera in longitudinaler Richtung. In (a) sind Ergebnisse zu Konfiguration 1 mit *inaktiver* zeitlicher Verfolgung und in (b) entsprechende Ergebnisse mit *aktiver* Objektverfolgung zu Konfiguration 2 abgebildet. Oben sind die Autokorrelationsfunktionen und unten die jeweiligen Abweichungen zwischen fusionierten Sensor– und Modelldaten dargestellt.

Daten sind in Abbildung 4.11 gezeigt. Zu erkennen ist, dass das Umfeldmodell die Autokorrelation der nicht zeitlich verfolgten Daten von Konfiguration 1 erhöht (Abbildung 4.11a), dies ist jedoch nicht ausreichend für die Akzeptanz, verglichen mit fusionierten realen Sensordaten, da die Differenz außerhalb des Konfidenzintervalls fällt (Abbildung 4.11a unten). Im Gegensatz dazu, zeigt Abbildung 4.11b, dass die Fusion der Ergebnisse aus Konfiguration 2 das Akzeptanzkriterium erfüllt.

Zusammenfassend lässt sich sagen, dass die zeitliche Objektverfolgung essentiell für die Sensormodellierung ist, da sie ein Teil der realen Sensorsignalverarbeitung ist. Die Analysen mit dem Umfeldmodell bestätigen dies und zeigen, dass Sensormodelle, die die zeitliche Objektverfolgung nicht berücksichtigen, aus Sicht des Gesamtsystems aus Abbildung 1.1 nicht geeignet sind.

4.3 Fazit zur Bewertung phänomenologischer Sensormodelle

Im ersten Teil dieses Kapitels wird eine Methode vorgestellt, mit der phänomenologische Eigenschaften eines Sensormodells analysiert werden können. Die zugrundeliegende Idee ist, zu vergleichen, wie gut ein Sensormodell die Eigenschaften eines realen Sensors auf einer Objektlistenebene abbildet. Dazu werden erst die für umfelderfassende Sensoren typischen Fehlerarten eingeführt. Die für diese Arbeit relevanten Fehler werden auf die Fehler der direkten Perzeption, die bei physikalischen Messungen auftreten, und Paketierungsfehler, wie beispielsweise Netzwerkfehler, eingeschränkt. Daraus wird die Notwendigkeit von zwei Analyseverfahren (messfahrbasierte und Autokorrelationsanalyse) motiviert.

Die messfahrtbasierte Analyse ist ein Verfahren zur Bewertung der Modellgüte, das auf dem Formvergleich von einzelnen Trajektorien sowie deren Fehlerverteilungen basiert. Im Rahmen dieses Verfahrens werden neue potenzielle Metriken und Diskrepanzmaße zur Quantifizierung der Ähnlichkeit zweier Trajektorien eingeführt und basierend auf acht Modellkonfigurationen deren Eignung für die oben formulierte Aufgabe untersucht. Der Korrelationskoeffizient und der Kreuzkorrelationskoeffizient haben sich während der Analyse als nicht sensitiv genug gezeigt, um die Sensormodellfehler zu detektieren. Grund hierfür sind die typischerweise langen Trajektorien, die sich in deren Form zwischen Realität und Simulation kaum unterscheiden und deshalb hohe Korrelation aufweisen.

Im Allgemeinen können einzelne Metriken auf unterschiedliche Fehlerquellen ähnlich reagieren. Es wurden bei der Untersuchung jedoch Muster identifiziert, wie anhand mehrerer Metriken diverse Sensormodellfehler detektiert werden können. Die systematischen Translationsfehler, wie beispielsweise falsch konfigurierte Sensorpositionen, können erkannt werden, wenn sich zwischen zwei Modellkonfigurationen die Vektornorm L_∞ und die Flächenmetrik D_F um einen ähnlichen Betrag ändern. Eine weitere Fehlerart ist die Verschiebung von zwei Signalen gegeneinander, die bei falsch eingestellten Netzwerklatenzen auftritt und zu gegeneinander verschobenen realen und simulierten Messdaten führt. So ein Verhalten hat Auswirkung auf die Ergebnisse aller Metriken außer der DTW–Metrik. Solche Fehler sind deshalb gut erkennbar, wenn sich alle Metriken, außer der DTW merklich ändern. Zusätzlich werden geeignete Metriken für den Vergleich von Fehlerverteilungen zweier Messfahrten identifiziert. KLD und Flächenmetrik sind hierfür geeignete Kandidaten.

Aus der messfahrtbasierten Analyse wird die Notwendigkeit der Autokorrelationsanalyse abgeleitet. Ausschlaggebend dafür waren kontroverse Ergebnisse der formvergleichenden Metriken und Metriken zum Vergleich von Fehlerverteilungen. Die Autokorrelationsanalyse dient der Bewertung der Fähigkeit eines Modells die dynamischen Eigenschaften eines Sensors abzubilden. Mit anderen Worten, hilft die Autokorrelationsanalyse zu entscheiden, ob die dynamischen Eigenschaften eines Sensors in ausreichender Güte in einem Modell abgebildet sind. Diese basiert auf der Messung der Ähnlichkeit eines Signals, verschoben um die Zeit τ und dem ursprünglichen Signal. Damit ist es möglich, die zeitliche Korrelation eines simulierten Signals mit der eines realen Sensors zu vergleichen. Die Ergebnisse der Autokorrelationsanalyse haben gezeigt, dass die zeitliche Filterung der Objekte essentiell für ein Sensormodell ist, insbesondere in Hinblick auf die nachgelagerte Fusion der Sensordaten.

Zusammenfassend liefern die Ergebnisse dieses Kapitels die Möglichkeit, phänomenologische Modelle objektiv zu bewerten und optimale Sensorkonfigurationen, basierend auf Trajektorienform und Fehlerverteilung vergleichenden Metriken sowie der Bewertung der dynamischen Eigenschaften eines Modells, zu finden und liefern somit die Antwort auf die dritte Forschungsfrage.

5 Methoden zur Bewertung physikalischer Sensormodelle

In diesem Kapitel soll die vierte Forschungsfrage, wie physikalische Sensormodelle objektiv bewertet werden können, geklärt werden. Im Kontext dieser Arbeit sind unter physikalischen Sensormodellen Ansätze zu verstehen, die auf strahlenoptischer Modellierung basieren und simulierte Sensordaten als ungefilterte Messungen ausgeben (siehe Kapitel 2.1.2). Die Ergebnisse dieses Kapitels sind zum Teil im Rahmen der Abschlussarbeit [142] entstanden.

Heutzutage finden im Automobilbereich zwei Arten von Lidar–Sensoren Anwendung. Solche, die mechanisch bewegte Komponenten enthalten und bereits in der Serienproduktion eingesetzt werden und die sogenannten Festkörper–Lidar–Sensoren, die sich aktuell in der Entwicklungs– beziehungsweise Industrialisierungsphase befinden [143]. Die Untersuchungen dieses Kapitels basieren auf einem Lidar–Sensor mit mechanischen Komponenten, obwohl sich die hier vorgestellten Konzepte auch auf andere Lidar–Arten anwenden lassen, so lange diese als Ausgangsformat die in Kapitel 2.1.1 definierten ungefilterten Messungen zur Verfügung stellen. Als Beispiel für ein physikalisches Sensormodell wird hier ein Lidar–Modell aus [24] vorgestellt.

Das Messprinzip eines mechanischen Lidars ist in Abbildung 5.1 visualisiert. In 5.1a wird die Entfernungsmessung herkömmlicher Lidar–Sensoren im Automobilbereich dargestellt. Diese basiert auf der Messung der Laufzeit eines Strahls, der vom Sensor ausgesendet und von einer Zieloberfläche reflektiert wird. Die horizontale Winkelauflösung ist in 5.1b dargestellt. Diese wird durch Abtasten des Sensorsichtbereichs erreicht, indem der Spiegel rotiert und mit einer bestimmten Frequenz Laserstrahlen in die Umgebung sendet. Die vertikale Winkelauflösung wird bei diesem Prinzip in der Regel durch

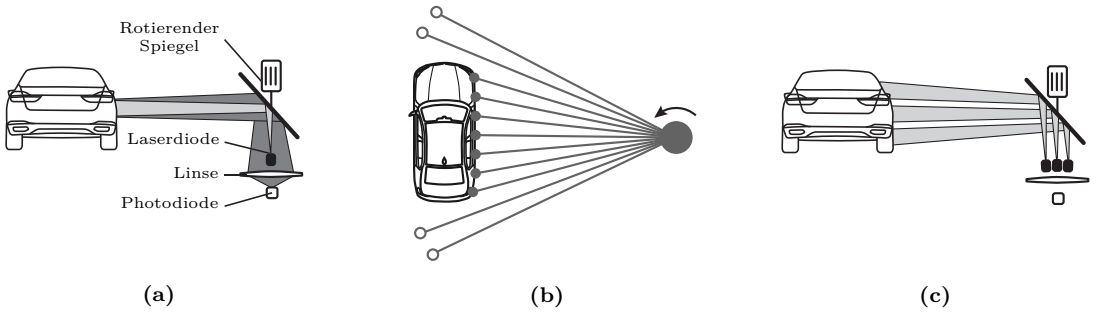


Abbildung 5.1: Visualisierung des Lidar–Messprinzips (a) sowie der Abtastung horizontaler (b) und vertikaler (c) Sichtbereiche.

mehrere nebeneinander fest verbaute Laserdioden erreicht und ist in Abbildung 5.1c dargestellt.

Die Aufgabe des physikalischen Lidar–Modells ist, basierend auf einer dreidimensionalen virtuellen Umgebung (siehe Abbildung 5.3), eine realitätsnahe Lidar–Punktewolke zu generieren. Umgebungsmodellierung ist ein wichtiger Aspekt bei der Validierung physikalischer Sensormodelle und wird in Kapitel 5.1 separat behandelt. Während der Simulation werden, aus der Sicht des virtuellen Sensors, Strahlen in die modellierte Umgebung ausgesendet und so die Position der Objekte ermittelt. Die Anzahl der generierten Strahlen entspricht dabei der Anzahl eines realen Sensors. Aufgrund der hohen Winkelselektivität optischer Empfänger ist der Einfluss von Mehrfachreflexionen bei einem Lidar–Sensor eher gering [24], weshalb die Strahlen mittels Strahlenfang–Prinzip, das in Kapitel 2.1.2 erklärt wird, modelliert werden. Sobald die ausgesendeten Strahlen auf eine Oberfläche in der virtuellen Umgebung treffen, werden, basierend auf einer Materialdatenbank, die Reflexionseigenschaften der getroffenen Texturen ausgewertet und entschieden, ob der jeweilige Treffpunkt in der Realität erkennbar wäre. Die Entscheidung, ob ein Objekt sichtbar ist, beruht dabei auf einem Schwellwert, der basierend auf realen Messungen parametrierbar ist. Neben den Reflexionseigenschaften erlaubt die Materialdatenbank die Definition von Objektarten, wie beispielsweise Fahrbahnoberfläche oder Fahrzeug. Einen weiteren zu modellierenden Effekt stellt das Abstandsrauschen dar. Dieses hängt stark von den Materialeigenschaften der Objekte ab und wird in Form von weißem Rauschen in der Nachverarbeitung hinzugefügt. Die Rauschvarianz hängt dabei von der Objektart, Niederschlagsbedingungen beziehungsweise der allgemeinen Helligkeit ab. Vegetation hat beispielsweise deutlich höhere Varianz als eine glatte Betonwand.

Ein weiterer relevanter Schritt ist die Generierung von Belegungsrasterkarten. Ausgehend von der vorgestellten Validierungsmethode aus Kapitel 3.1, basiert die Validierung von Sensormodellen unter anderem auf dem Vergleich von Fusionsergebnissen. Belegungsrasterkarten stellen eine Möglichkeit dar, ungefilterte Messungen zu fusio-

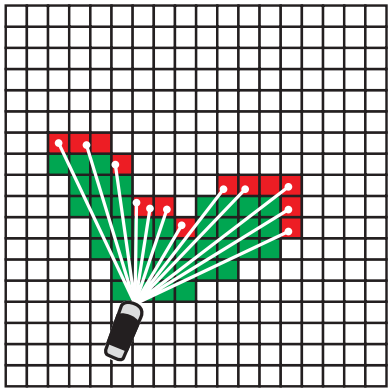


Abbildung 5.2: Visualisierung der Erstellung einer Belegungsrasterkarte auf Basis der Punktewolke eines Lidar–Sensors nach [142].

nieren und sind damit eine Art Pendant zur Kalman–Filterung auf Objektlistenebene. Zumeist werden Rasterkarten zur simultanen Lokalisierung und Kartenerstellung verwendet, um einem Fahrzeug die Navigation in einer statischen Umgebung zu ermöglichen [142]. Dabei wird das Umfeld diskretisiert, indem es in Zellen unterteilt wird. Die Größe der Zellen ist dabei einstellbar [144]. Die Belegungsrasterkarten werden dabei in jedem Zeitschritt mittels probabilistischen Methoden zeitlich gefiltert und basierend darauf belegte und freie Bereiche abgeleitet. Die in dieser Arbeit verwendete Methode basiert dabei auf dem Bayeschen–Belegungs–Filter [145].

Zur Erstellung von Belegungsrasterkarten kommen vorrangig Sensoren zum Einsatz, welche für die Entfernungsmessung geeignet sind. Grewe führt in [146] beispielsweise Lidar– und Radarsensoren sowie Stereo– und Monokameras auf. Dazu werden anhand der durch den Sensor im Raum ermittelten Objekte die probabilistischen Belegungswahrscheinlichkeiten der einzelnen Zellen berechnet. Die Belegungsrasterkarte unterteilt den Raum dabei oftmals in drei Kategorien: (a) belegter Raum, (b) unbelegter Raum und (c) unbekannter Raum. Liegt zu Beginn noch keine Messung eines Sensors vor, wird die Belegungsrasterkarte in ihrer Gesamtheit als unbekannter Raum initialisiert. Bei einer Messung werden Zellen, in welchen ein Objekt lokalisiert wird, als belegt angenommen, während solche, die sich zwischen Sensor und dem Objekt befinden, als Freiraum abgeleitet werden [142]. Die Visualisierung der drei Zellentypen ist in Abbildung 5.2 gezeigt. In der Abbildung stellen die weiß gefärbten Zellen die unbekannten Bereiche dar. In grün sind die freien und in rot die belegten Zellen dargestellt.

5.1 Umgebungsmodellierung

Wie einleitend in diesem Kapitel erwähnt, wird für die Validierung physikalischer Sensormodelle eine dreidimensionale Repräsentation der virtuellen Umgebung benötigt. Diese definiert den Raum, in dem sich virtuelle Objekte während der Simulation bewegen können. Da die Validierung auf dem Vergleich realer und simulierter Daten beruht, muss die virtuelle Umgebung eine Nachbildung eines realen Streckenabschnitts sein, um eine maximale Vergleichbarkeit zwischen der realen und virtuellen Welt zu gewährleisten. Ein Ausschnitt eines solchen Umgebungsmodells ist in Abbildung 5.3 zu sehen.

Abbildung 5.3: Ausschnitt einer dreidimensional nachmodellierten, realen Umgebung [147].

So ein Modell dient als Referenz–Repräsentation der realen Umgebung für ein physikalisches Sensormodell und ist eine weitere notwendige Referenzart neben den DGNSS–Referenzdaten für dynamische Objekte. An so eine Umgebungsrepräsentation müssen deshalb besonders hohe Anforderungen in Bezug auf die Genauigkeit der Geometriemaße des Modells sowie dessen Georeferenzierung gestellt werden. Zusätzlich muss das zu erstellende Umgebungsmodell fotorealistisch sein, um Modelle bildgebender Sensoren, wie beispielsweise Kameras, validieren zu können.

Um unterschiedliche Fahrszenarien auf unterschiedlichen Straßentypen untersuchen zu können, wurde im Rahmen dieser Arbeit ein Areal ausgesucht, bei dem Stadtbereiche, Landstraße und Autobahn möglichst nahe beieinander liegen. Dieses ist in Abbildung 5.4 gezeigt. Im oberen Teil der Abbildung ist ein Landstraßenstück, mittig die Stadt und rechts die Autobahn zu sehen. Der farbig markierte Bereich hat eine Gesamtfläche von circa 240 Hektar. Ziel ist es, dieses Areal zu virtualisieren, um dort unter Verwendung einer Simulationsumgebung virtuelle Fahrten durchführen zu können. Drei Schritte sind notwendig um diese Aufgabe zu erfüllen: (a) Vermessung des gesamten Areals, (b) Fusion der aufgezeichneten Messdaten und (c) Modellierung und Konvertierung der registrierten Daten in ein passendes Format. Die aufgeführten Schritte werden im Folgenden einzeln erläutert.

Wie bereits erwähnt, muss das zu erstellende Umgebungsmodell zwei Anforderungen erfüllen: Genauigkeit und Fotorealismus. Dies erfordert die **Vermessung** sämtlicher realer Objekte im Bereich der in Abbildung 5.4 eingezeichneten Straßen inklusive photorealistischer Texturen sowie aller Objekte im nahen und mittleren Sichtbereich. Dazu werden zwei in der Vermessung weit verbreitete Technologien eingesetzt: Laserscanning

Abbildung 5.4: Darstellung des zu modellierenden Areals mit farbig hinterlegten Aufnahmepunkten der terrestrischen Scanns [147].

und Photogrammetrie. Um hohe Genauigkeiten zu erreichen, eignet sich terrestrisches Laserscanning. Für den Fotorealismus ist Photogrammetrie, abgeleitet aus hochauflösenden Vollformatkameraaufnahmen, geeignet. Da das zu erstellende Umgebungsmodell darüber hinaus georeferenziert sein muss, benötigen die jeweiligen Messgeräte (Laserscanner und Kameras) entsprechende GNSS–Sensoren. Die hohen Genauigkeiten müssen dabei nicht zwangsweise in allen Bereichen des definierten Areals, sondern nur im Sichtbereich der am Testfahrzeug verwendeten Sensoren, erreicht werden. Typische Reichweiten umfelderfassender Sensoren im Automobilbereich liegen bei circa 200m. Damit ist es ausreichend, wenn die relevanten Straßenabschnitte sowie alle Objekte in dieser Entfernung entlang der Straßen mit der höchstmöglichen Genauigkeit vermessen werden. Die als relevant erachteten Straßenabschnitte sind in Abbildung 5.4 mit farbigen Punkten markiert. Andere Bereiche des Areals sind weniger relevant für die Sensormodellvalidierung und können mit entsprechend niedriger Genauigkeit modelliert oder komplett ausgespart werden.

Terrestrisches Laserscanning ist ein Verfahren zur dreidimensionalen Erfassung von Objekten und Landschaften, das zumeist in der Geodäsie seine Anwendung findet [148]. Dabei wird mit Laserscannern die Umgebung mit bis zu 500.000 Messpunkten pro Sekunde abgetastet und daraus hochauflösende Punktewolken generiert. Die einzelnen Aufnahmepunkte für das definierte Areal sind in Abbildung 5.4 farbig markiert. Die einzelnen Scanaufnahmen werden anschließend in eine gesamthafte Punktewolke überführt, wobei die überlappenden Bereiche verwendet werden, um die Einzelscans zueinander auszurichten. Dieser Schritt wird als Registrierung bezeichnet. Ein Ausschnitt der Scanergebnisse ist in Abbildung 5.5a dargestellt.

Die Photogrammetrie beschäftigt sich mit der Gewinnung und Verarbeitung von Informationen über Objekte unter Verwendung von Kamerabildern [149]. Mittels Photogrammetrie können dreidimensionale Objekte fotorealistisch modelliert werden. Im Rahmen der Vermessung des Areals werden zwei unterschiedliche Verfahren eingesetzt: (a) terrestrische und (b) luftgestützte Photogrammetrie. Im Rahmen der terrestrischen Photogrammetrie werden Fotoaufnahmen vom Boden beziehungsweise einer mobilen Bühne erfasst. Hauptsächlich wird damit der Stadtbereich vermessen. Bei der luftgestützten Photogrammetrie wird ein Hubschrauber eingesetzt, um die restlichen Bereiche, vor allem Felder, die Autobahn und Landstraßen zu vermessen. Ein Ausschnitt der

(a) **(b)**

Abbildung 5.5: Visualisierung der Ergebnisse des terrestrischen Scannens und der Photogrammetrie. (a) zeigt eine Punktewolke eines terrestrischen Scans. (b) zeigt eine Photogrammetrie–Aufnahme links und das Modellierungsergebnis rechts [147].

Ergebnisse ist in Abbildung 5.5b gezeigt. Dort sind links die Fotoaufnahme und rechts das dreidimensionale Modell, das aus mehreren Bildern erstellt wird, abgebildet.

Um ein hochgenaues fotorealistisches Umgebungsmodell zu erstellen, müssen die Ergebnisse der Messaufnahmen zusammengeführt werden. Dieser Schritt wird als **Fusion** bezeichnet. Während der Fusion werden die DGNSS–Daten der Referenzstationen, die Punktewolken von Laserscannern und die Photogrammetrie–Aufnahmen aufeinander angepasst. Ziel dieses Schritts ist es, die Geoposition und Modellgüte zu verbessern. Nach der Fusion wird eine Prüfung der erreichten Genauigkeiten durchgeführt. Dazu werden die Fusionsergebnisse mit den Positionen der zuvor vermessenen Kontrollpunkte verglichen. Die relative Genaugkeit des Modells liegt bei < 2cm, die absolute Genauigkeit bei < 5cm. Somit liegt die Genauigkeit der Referenzdaten für die Validierung physikalischer Modelle in einer ähnlichen Größenordnung, wie die der Referenzdaten phänomenologischer Modelle.

Schließlich wird, basierend auf den Fusionsergebnissen, die vermessene Umgebung **modelliert**. Als Erstes werden auf Basis von Scandaten die Grundgeometrien der Objekte rekonstruiert. Diese Geometrien werden dann immer weiter verfeinert, zum Beispiel von Gebäudegeometrien zu Fensterrahmen und Türen, bis das Level of Detail (LoD) 300 nach dem Building–Information–Modeling–Prozess erreicht ist [150]. Anschließend wird nach Möglichkeit die Anzahl der „unnötigen“ Polygone reduziert, um die Echtzeitfähigkeit während der Simulation zu erreichen. Dies erfolgt beispielsweise für große Gebäudeflächen oder Fahrbahnoberflächen. Als nächstes wird das Polygonmodell, basierend auf den Photogrammetrie–Ergebnissen, texturiert. Im letzten Schritt wird zur Qualitätssicherung das fertige Modell mit der korrigierten Punktewolke aus dem Fusionsschritt verglichen. Die Ergebnisse einzelner Zwischenschritte sind im Anhang A.6 dokumentiert und das fertige Modell des gesamten Areals in Abbildung 5.6 gezeigt. Zuletzt wird das fertiggestellte Modell in ein für die Simulationsumgebung geeignetes Format, wie beispielsweise OpenFlight (FLT) oder OpenSceneGraph (IVE),

Abbildung 5.6: Darstellung des fertig modellierten Areals [147].

exportiert.

Ergebnisse dieses Unterkapitels wurden mit freundlicher Unterstützung der Firma ArcTron 3D GmbH [151] erzielt. Insbesondere Vermessung, Prozessierung der Daten und Modellierung wurden von dem Unternehmen durchgeführt.

5.2 Vorstellung des Ansatzes

Fehler, die während der physikalischen Sensormodellierung auftreten, können ähnlich wie für phänomenologische Modelle klassifiziert werden. Ein entscheidender Unterschied ist dabei, dass bei physikalischen Modellen statt Objektlisten und Fehlerverteilungen, Punktewolken und Belegungsrastergitter verglichen werden. Dabei erfolgt die Messung der Ähnlichkeit realer und synthetischer Sensordaten mittels diverser dafür geeigneter Metriken, die als Eingang zwei Matrizen erwarten.

Belegungsrasterkarten sind naturgemäß als Matrizen definiert. Punktewolken auf der anderen Seite sind Tupel, die Eigenschaften eines Punktes im dreidimensionalen Raum beinhalten. Um eine Punktewolke in eine Matrixform zu überführen, wird eine Diskretisierung des Raumes vorgeschlagen. Dies ist ähnlich der Diskretisierung des Raumes durch eine Belegungsrasterkarte, allerdings werden die Daten nicht durch die Anwendung von Filtern oder weiteren Modellierungen verfälscht. Im Folgenden werden zwei Methoden zur Überführung einer Punktewolke in eine Matrixform vorgestellt: Ein allgemeiner und ein sensorspezifischer Ansatz [142].

Der allgemeine Ansatz leitet aus einer dreidimensionalen Punktewolke drei zweidimensionale Punktewolkenprojektionen ab. Die Punkte werden dabei in die drei kartesischen Koordinatenebenen XY, XZ und YZ projiziert und anschließend in quadratische Zellen diskretisiert. Jede Zelle stellt dabei einen Belegungswert dar, welcher 0 oder 1

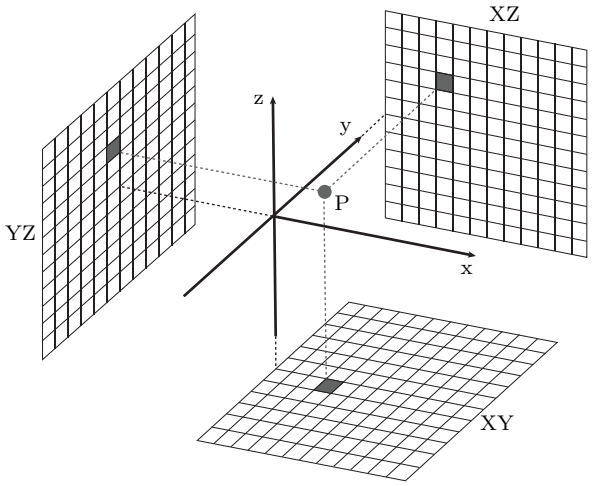


Abbildung 5.7: Visualisierung der Abbildung eines dreidimensionalen Punktes in drei zweidimensionale Projektionsmatrizen nach [142].

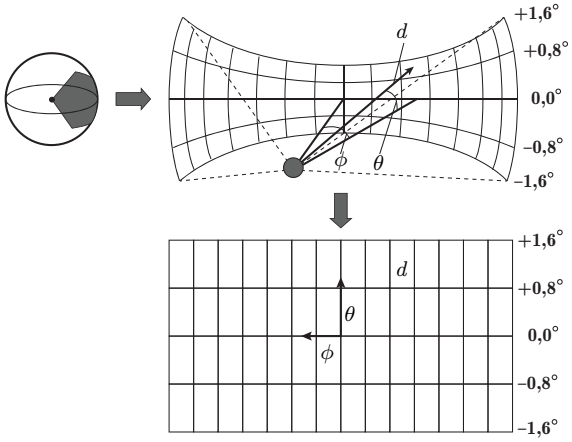


Abbildung 5.8: Projektion eines Reflexionspunktes in die Euklidische–Distanzkarte nach dem sensorspezifischen Ansatz nach [142].

betragen kann. Die Zellen der Projektionsmatrizen werden mit 0 initialisiert. Eine Zelle, in der die Projektion eines Punktes trifft, wird mit dem Wert 1 als belegt angenommen. Das Prinzip ist in Abbildung 5.7 visualisiert. Dieser Ansatz bietet den Vorteil, dass durch ihn eine Aussage für beliebige Punktewolken getroffen werden kann, ohne dabei das Messprinzip beziehungsweise die Geometrie des zur Erhebung verwendeten Sensors zu kennen [142].

Für den sensorspezifischen Ansatz müssen Kenntnisse über das Messprinzip, den Sensorerfassungsbereich und dessen räumliche Auflösung bekannt sein. Im Fall eines Lidars (siehe Abbildung 5.1), bei dem die ausgesendeten Laserstrahlen über die Ablenkung eines Spiegels von einer Diode emittiert werden, findet die Messung deshalb in Kugelkoordinaten statt. Hinter der Überführung der Punktewolke in eine Matrix nach diesem Prinzip, das in Abbildung 5.8 visualisiert ist, stehen folgende Überlegungen. Der im Raum aufgespannte Sensorsichtbereich entspricht räumlich gesehen einem Kugelsegment. Die gedachte Kugeloberfläche wird von den Messstrahlen auf deren vom Augpunkt ausgehenden Weg durchdrungen. Die Oberfläche wird in Zellen von Flächensegmenten eines definierten Winkelbereichs unterteilt. Basiert die zu überführende Punktewolke auf diesem Messprinzip und sind Azimut– und Elevationswinkel eines Strahls konstant, so durchdringt dieser immer dieselbe Zelle unabhängig von seiner zurückgelegten Entfernung. Die Projektionsmatrix stellt dadurch eine Planarisierung des Sensorerfassungsbereichs dar.

Die Matrixgröße des planaren Sensorsichtbereichs kann aus der Sensorgeometrie abgeleitet werden. Dabei entspricht die Spaltenanzahl der Matrix der Anzahl ausgesendeter Laserstrahlen in horizontaler und die Zeilenanzahl in vertikaler Richtung. Ein Punkt aus der Punktewolke wird dabei einer Matrixzelle zugeordnet, der euklidische Abstand d dieses Punkts in die jeweilige Zelle eingetragen und die nicht belegten Zellen mit 0 belegt. Da eine nach diesem Ansatz erstellte Projektionsmatrix die euklidischen Abstände der einzelnen Messpunkte enthält, wird diese als Euklidische–Distanzkarte bezeichnet [142].

Vorteil dieses Ansatzes im Vergleich zu dem allgemeinen liegt darin, dass nur eine Projektionsmatrix statt drei generiert wird und diese üblicherweise kleiner ausfällt, wodurch der Kalkulationsaufwand der Vergleichsmetriken reduziert wird. Bei einem realen Lidar–Sensor, der zum Beispiel einen horizontalen Sensorsichtbereich von $-60°$ bis $+50°$ mit 0,25° Winkelauflösung, einen vertikalen Sichtbereich von $-1{,}6°$ bis $+1{,}6°$ mit 0,8° Winkelauflösung und einer Abstandsreichweite von 200m mit 4cm Auflösung besitzt, resultiert eine Euklidische–Distanzkarte der Größe 440×4. Eine Diskretisierung der Punktewolke in 50cm große Kacheln mittels des allgemeinen Ansatzes ergibt drei Matrizen der Größe 400×655 (XY–Ebene), 22×655 (YZ–Ebene) beziehungsweise 22×400 (XZ–Ebene) [142]. Der sensorspezifische Ansatz wird in dieser Arbeit deshalb gegenüber dem allgemeinen Ansatz bevorzugt.

Neben der Transformation der Punktewolken sind einige Aspekte in Bezug auf Belegungsrasterkarten zu beachten. Eine Belegungsrasterkarte stellt die Grundlage für die weiterführende Fahrentscheidungsalgorithmik dar, denn viele nachstehende Funktionen arbeiten nicht direkt auf der Ebene der ungefilterten Messungen, sondern auf Belegungsrasterkarten. Validierung, die auf Belegungsrasterkarten basiert, kann deshalb auch eine bessere Aussage bezüglich der Auswirkung des Sensormodells auf nachstehende Funktionen treffen. Die Aussage über die Qualität des Sensormodells anhand der aus den Sensormessungen generierten Belegungsrasterkarten ist deshalb als mindestens ebenso wichtig zu erachten, wie die Aussage auf Ebene der ungefilterten Messungen [142].

Die Belegungsrasterkarten müssen in deren Ausrichtung normalisiert werden. Unter Normalisierung ist die Rotation und Verschiebung der ursprünglichen Belegungrasterkarte gemeint, sodass die resultierende Rasterkarte in Fahrzeugrichtung ausgerichtet ist und der Ursprung mit dem Ursprung der Fahrzeugkoordinaten übereinstimmt. Dies ist notwendig, da der überwiegende Teil der Vergleichsmetriken posenvariant ist und für die Vergleichbarkeit von realen und simulierten Daten die Kartenrotation in eine einheitliche Winkelpose überführt werden muss. Dazu sind zwei Schritte notwendig, die in Abbildung 5.9 visualisiert sind. Im oberen Teil der Abbildung ist eine Aufnahme der Frontkamera des Messfahrzeuges zu sehen, um einen besseren Bezug zwischen der Szene und der berechneten Belegungsrasterkarte zu ermöglichen. Im unteren linken Teil der Abbildung 5.9 ist die unbearbeitete Belegungsrasterkarte, sowie diese vom Umfeldmodell ausgegeben wird, zu sehen. Die gewählte Aufnahme entspricht einem Zeitpunkt, bei dem das Fahrzeug aufgrund von Kurvenfahrten nicht mehr seiner Initialpose entspricht. Daher erstreckt sich der aus dem Laserscanner abgeleitete Fahrbahnverlauf diagonal über die Belegungsrasterkarte. Im unteren mittleren Teil der Abbildung 5.9 wird dieselbe Belegungsrasterkarte gezeigt, deren Rotation unter Verwendung des in der Fahrzeugodometrie enthaltenen Fahrzeugrotationswinkels rotiert ist. Nach der Rotation ändert sich neben der Ausrichtung auch die Größe der Belegungskarte. Diese muss im nächsten Schritt auf die ursprüngliche Kartengröße zugeschnitten und in den Fahrzeugursprung verschoben werden. Die resultierende Belegungsrasterkarte ist im unten rechten Teil der Abbildung 5.9 zu sehen. Die in ihr neu enthaltenen Eckbereiche werden mit dem Belegungswahrscheinlichkeitswert „unbekannt“ markiert, da diese

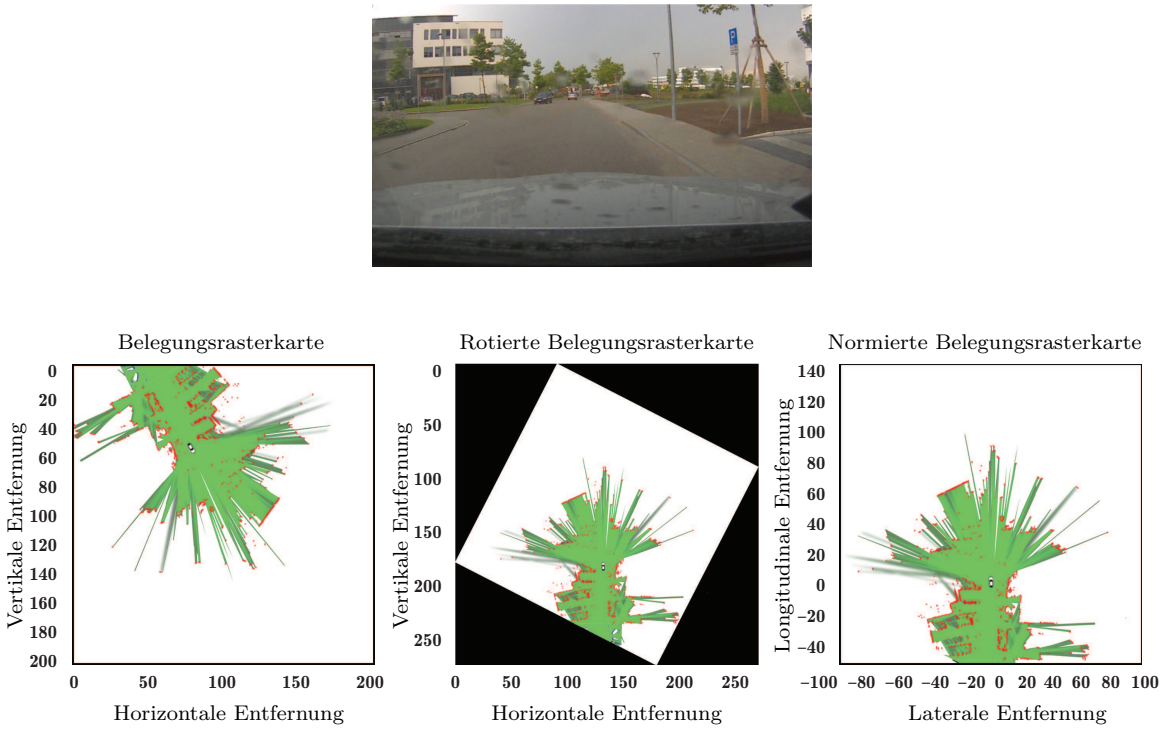


Abbildung 5.9: Verarbeitungsschritte zur Rotationsnormalisierung einer Belegungsrasterkarte nach [142] [152].

Zellen in der ursprünglichen Rasterkarte nicht vorhanden waren und damit außerhalb des Sensorerfassungsbereichs liegen [142].

Die Rotation der gewählten Belegungsrasterkarte wird normalisiert, indem sie entgegengesetzt des Gierwinkels rotiert wird. Die Zellen innerhalb der Belegungsrasterkarte sind ganzzahlig unterteilt, während die errechneten Gierwinkeln reelle Werte haben. Bei der Rotation der Belegungsrasterkartenmatrix werden daher, abgesehen von Vielfachen rechtwinkliger Rotationen, Überdeckungseffekte entstehen. Die Zellwerte der normalisierten Belegungsrasterkartenmatrix müssen deshalb durch Zuordnung zu den ursprünglichen Werten bestimmt werden. Diese Operation ist als kritisch zu erachten, da diese die äquivalenten Zellwerte zwischen der ursprünglichen und der rotierten Belegungsrasterkarte verfälscht. Ohne diesen Schritt ist jedoch keine Rotationsnormalisierung und damit kein Abgleich der Belegungsrasterkarten möglich. Der Zuordnungsschritt ist in Abbildung 5.10 visualisiert. Die quadratische, mit Zellwerten belegte Matrix wird dort um einen Winkel von $\phi = 30°$ um ihren Mittelpunkt rotiert. Durch die Rotation vergrößert sich die neu errechnete Matrix symmetrisch in alle Richtungen. Die Zellwerte der rotierten Matrix werden ermittelt, indem die Zellmittelpunkte der rotierten und der ursprünglichen Matrix verglichen werden. Der Zellwert der ursprünglichen Matrix, dessen Zellmittelpunkt den geringsten Abstand zum Mittelpunkt der zu befüllenden Zelle in der rotierten Matrix besitzt, wird als neuer Wert übernommen. Vereinfacht gesehen entspricht dies auch dem höchsten Überdeckungsgrad der Zellen. Die neu entstandenen Randbereiche werden mit dem Wert 0 belegt. Die so erzeugte

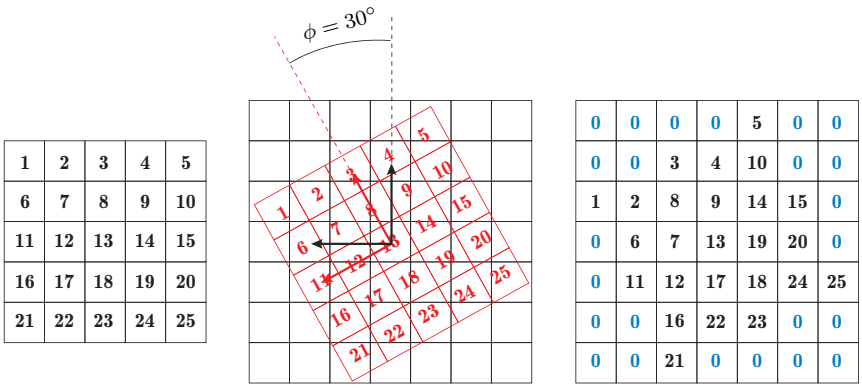


Abbildung 5.10: Zuordnung der Zellenwerte nach der Rotation der Belegungsrasterkarte nach [142].

Belegungsrasterkarte entspricht deshalb in ihren Seitenlängen nicht mehr der ursprünglichen Rasterkarte, sondern ist vergrößert. Für einen zellenweisen Vergleich müssen die Dimensionen von den zu vergleichenden Belegungsrasterkarten jedoch übereinstimmen. Nach der Rotation ist dies nicht mehr gewährleistet, weshalb die realen und simulierten Rasterkarten auf die Größe ihres ursprünglichen Äquivalents zugeschnitten werden müssen. Da die Belegungsrasterkarte quadratische Ausdehnungsmaße besitzt, ist die resultierende Matrixvergrößerung symmetrisch um den Rasterkartenmittelpunkt. Eine einfache symmetrische Zuschneidung der vergrößerten Belegungsrasterkartenmatrix wirkt sich dadurch nicht auf die Translation des Fahrzeugs beziehungsweise des Sensors innerhalb der Karte aus [142]. Mit anderen Worten befindet sich nach symmetrischer Zuschneidung der Belegungskarte das Eigenfahrzeug in beiden Rasterkarten auf der selben Position.

Im Folgenden werden Metriken vorgestellt, die für Bewertung der Ähnlichkeit zweier Matrizen geeignet sind. Der zellenweise Vergleich wird dabei als zielführend erachtet, da die Bewertung vollkommen unabhängig von anwendungsspezifischen Merkmalextraktionen und weiterführenden Datenverarbeitungen erfolgt. Falls nicht anders erwähnt, bezieht sich der Vergleich bei allen Metriken auf experimentelle $\boldsymbol{\zeta}^{\mathrm{e}}$ und simulierte $\boldsymbol{\zeta}^{\mathrm{s}}$ Daten, repräsentiert als Matrizen der Größe $m \times n$. Da die Metriken zellenweisen Vergleich nutzen, wird vorausgesetzt, dass die zu vergleichenden Karten die gleiche Umgebung darstellen und Position sowie Orientierung der Rasterkarten identisch sind. Deshalb wird davon ausgegangen, dass eine Zelle ζ^{e}_{ij} einer Rasterkarte $\boldsymbol{\zeta}^{\mathrm{e}}$ und eine Zelle ζ^{s}_{ij} der zu vergleichenden Rasterkarte $\boldsymbol{\zeta}^{\mathrm{s}}$ für alle ij denselben physikalischen Raum abbilden.

Matrixnorm ist eine auf der Vektornorm (siehe Gleichung 4.1) basierende Metrik, die als Diskrepanzmaß zweier Matrizen $\boldsymbol{\zeta}^{\mathrm{e}}, \boldsymbol{\zeta}^{\mathrm{s}} \in \mathbb{R}^{n \times m}$ verwendet werden kann. Die allgemeine Definition der Matrixnorm lautet,

$$M_{\mathrm{M_p}}(\boldsymbol{\zeta}^{\mathrm{e}}, \boldsymbol{\zeta}^{\mathrm{s}}) = \left(\sum_{i=1}^{n} \sum_{j=1}^{m} |\zeta^{\mathrm{e}}_{ij} - \zeta^{\mathrm{s}}_{ij}|^p \right)^{1/p} \qquad \text{mit } p \in [1, \infty]. \tag{5.1}$$

Mit $p = 1$ ergibt sich aus dieser Definition ein in [153] verwendetes Distanzmaß, das dort als „Overall–Error“ bezeichnet wird (siehe Gleichung 5.2). Bei der Overall–Error Metrik handelt es sich um eine vergleichsweise einfache Korrelationsmessung. Sie summiert die, über einen zellenweisen Vergleich der Rasterkarten, gebildeten Beträge der Residuen

$$M_{\mathrm{OE}}(\boldsymbol{\zeta}^{\mathrm{e}}, \boldsymbol{\zeta}^{\mathrm{s}}) = \sum_{i=1}^{n} \sum_{j=1}^{m} |\zeta_{ij}^{\mathrm{e}} - \zeta_{ij}^{\mathrm{s}}|. \tag{5.2}$$

Gerade aufgrund ihrer Einfachheit wird die Metrik als sehr stark im Aufzeigen der Unterschiede zwischen Rasterkarten eingeschätzt. Einige Stärken sind dabei besonders hervorzuheben: Die Metrik lässt sich unter anderem zum Vergleich jeglicher Matrizen anwenden, da sie keine probabilistischen Wahrscheinlichkeitswerte voraussetzt. Außerdem sind Ergebnisse sehr einfach interpretierbar, da das Ergebnis die absolute Gesamtabweichung zwischen den Datensätzen ist. So liegen die aus der Metrik hervorgehenden Residuen im gleichen Datenformat und der gleichen Einheit wie die Eingangswerte vor [142].

Map–Score ist eine von Martin und Moravec vorgestellte Metrik, die in den Zellen vorhandene probabilistische Belegungswerte nutzt, um zwei Rasterkarten auf deren Ähnlichkeit zu überprüfen [154]. Es wird davon ausgegangen, dass jede Zelle die Wahrscheinlichkeit ihrer Belegtheit unter Verwendung von drei Werten (unbelegt 0, belegt 1 und unbekannt 0,5), enthält. Die Wahrscheinlichkeit, dass beide Zellen belegt sind, beträgt $\zeta_{ij}^{\mathrm{e}} \cdot \zeta_{ij}^{\mathrm{s}}$. Im Umkehrschluss beträgt die Wahrscheinlichkeit der Unbelegtheit beider Zellen $\bar{\zeta}_{ij}^{\mathrm{e}} \cdot \bar{\zeta}_{ij}^{\mathrm{s}}$. Der probabilistische Wert der Unbelegtheit einer Zelle berechnet sich dabei aus $\bar{\zeta}_{ij}^{\mathrm{e}} = 1 - \zeta_{ij}^{\mathrm{e}}$ und analog für $\bar{\zeta}_{ij}^{\mathrm{s}}$. Die Wahrscheinlichkeit, dass beide Zellen identisch sind, ist demnach $\zeta_{ij}^{\mathrm{e}} \cdot \zeta_{ij}^{\mathrm{s}} + \bar{\zeta}_{ij}^{\mathrm{e}} \cdot \bar{\zeta}_{ij}^{\mathrm{s}}$. Die Wahrscheinlichkeit, dass beide Rasterkarten identisch sind ergibt sich somit durch $\prod_{i \in m} \prod_{j \in n} \left(\zeta_{ij}^{\mathrm{e}} \cdot \zeta_{ij}^{\mathrm{s}} + \bar{\zeta}_{ij}^{\mathrm{e}} \cdot \bar{\zeta}_{ij}^{\mathrm{s}}\right)$. Da die Multiplikation einer solchen Vielzahl an Belegungswahrscheinlichkeiten zu einer extrem niedrigen Zahl führt, wird eine auf dem Logarithmus basierende Korrelationsgröße eingeführt,

$$M_{\mathrm{MS}}(\boldsymbol{\zeta}^{\mathrm{e}}, \boldsymbol{\zeta}^{\mathrm{s}}) = \sum_{i \in m} \sum_{j \in n} \left[1 + \mathsf{abs}\Big(\mathsf{log}_2\big(\zeta_{ij}^{\mathrm{e}} \cdot \zeta_{ij}^{\mathrm{s}} + \bar{\zeta}_{ij}^{\mathrm{e}} \cdot \bar{\zeta}_{ij}^{\mathrm{s}}\big)\Big)\right]. \tag{5.3}$$

Wird diese Metrik zum Vergleich einer rekonstruierten Rasterkarte und ihrem idealen Gegenstück verwendet, spricht man von „Map–Score“. Der Map–Score ist demzufolge eine Zahl, die umso mehr gegen die Anzahl der verglichenen Zellen konvergiert, je größer deren Identität ist [155]. Damit wird erreicht, dass der Metrikwert umso größer ist, je unterschiedlicher die Rasterkarten sind und der Anzahl von Zellen entspricht, falls beide Rasterkarten identisch sind.

Zu erwähnen ist, dass diese Metrik instabil in Bezug auf komplementär belegte Zellen ist. Ist beispielsweise Zelle ζ_i^{e} mit einem Wert von 0 und die zu vergleichende Zelle ζ_i^{s} mit einem Wert von 1 belegt, so ergibt sich ein zu logarithmierender Wert von 0. Da

der Logarithmus für 0 nicht definiert ist beziehungsweise dem $\lim_{x \to 0} x = -\infty$ entspricht, müssen solche Fälle zum Beispiel durch Zuordnung eines Maximalwertes abgefangen werden.

Quadratsumme der Residuen ist eine von Collins in [156] verwendete Metrik zur Berechnung der Summe quadrierter Differenzen über alle Zellen von $\boldsymbol{\zeta}^{\mathrm{e}}$ und $\boldsymbol{\zeta}^{\mathrm{s}}$. Diese Metrik ähnelt der Residuenquadratsumme für Matrizen und ist wie folgt definiert,

$$M_{\mathrm{RQS}}(\boldsymbol{\zeta}^{\mathrm{e}}, \boldsymbol{\zeta}^{\mathrm{s}}) = \sum_{i=1}^{n} \sum_{j=1}^{m} \left(\zeta_{ij}^{\mathrm{e}} - \zeta_{ij}^{\mathrm{s}}\right)^2. \tag{5.4}$$

Der von der Metrik ermittelte Wert kann beliebig groß werden und ist umso kleiner, je ähnlicher zwei zu vergleichende Rasterkarten sind, sodass zwei identische Karten einen Wert von 0 ergeben.

Picture–Distance–Function ist eine Metrik, die zwei Abbildungen, basierend auf dem Abgleich der enthaltenen Farbwerte f vergleicht [157]. Durch die hohe Ähnlichkeit zwischen einer Rasterkartenmatrix und einem Bild kann diese Metrik auch für Belegungsrasterkarten angewendet werden. Die in den Zellen vorhandenen Werte müssen dabei innerhalb eines festen Wertebereichs F liegen. Bei Belegungswahrscheinlichkeiten des Datentyps uint8 können die Wahrscheinlichkeitswerte beispielsweise auf $F = [0, 255]$ projiziert werden. Die Picture–Distance–Function ist wie folgt definiert,

$$\begin{aligned} M_{\mathrm{PD}}(\boldsymbol{\zeta}^{\mathrm{e}}, \boldsymbol{\zeta}^{\mathrm{s}}) &= \sum_{f \in F} d(\boldsymbol{\zeta}^{\mathrm{e}}, \boldsymbol{\zeta}^{\mathrm{s}}, f) + d(\boldsymbol{\zeta}^{\mathrm{s}}, \boldsymbol{\zeta}^{\mathrm{e}}, f) \\ \text{mit } d(\boldsymbol{\zeta}^{\mathrm{e}}, \boldsymbol{\zeta}^{\mathrm{s}}, f) &= \frac{\sum_{\zeta_{ij}^{\mathrm{e}} = f} \mathsf{min}\left\{\mathsf{md}(\zeta_{ij}^{\mathrm{e}}, \zeta_{ij}^{\mathrm{s}}) | \zeta_{ij}^{\mathrm{s}} = f)\right\}}{n_f(\boldsymbol{\zeta}^{\mathrm{e}})}, \end{aligned} \tag{5.5}$$

wobei $\zeta_{ij}^{\mathrm{e}} = f$ eine Zelle in der Rasterkarte $\boldsymbol{\zeta}^{\mathrm{e}}$ auf der Position ij der Farbe f beschreibt. $\mathsf{md}(\zeta_{ij}^{\mathrm{e}}, \zeta_{ij}^{\mathrm{s}}) = |i_{\zeta_{ij}^{\mathrm{e}}} - i_{\zeta_{ij}^{\mathrm{s}}}| + |j_{\zeta_{ij}^{\mathrm{e}}} - j_{\zeta_{ij}^{\mathrm{s}}}|$ ist die Manhattan–Distanz zum nächstgelegenen Pixel mit dem gleichen Farbwert in der jeweils anderen Karte, wobei nur horizontale und vertikale Verbindungen möglich sind. $n_f(\boldsymbol{\zeta}^{\mathrm{e}})$ ist die Anzahl der Zellen der Farbe f in $\boldsymbol{\zeta}^{\mathrm{e}}$. Die Metrik berechnet also die Summe der durchschnittlichen Manhattan–Distanzen aller Zellen der Farbe f in der Rasterkarte $\boldsymbol{\zeta}^{\mathrm{e}}$ im Vergleich zur nächstgelegenen Zelle der Farbe f in der Rasterkarte $\boldsymbol{\zeta}^{\mathrm{s}}$ und umgekehrt [142].

Für eine Berechnung der Manhattan–Distanzen kommen sogenannte Distanzkarten zum Einsatz, welche eine einfache Softwareimplementierung der Metrik ermöglichen. Für jeden im Farbintervall enthaltenen Farbwert f wird pro Belegungsrasterkarte eine Distanzkarte benötigt. Die Distanzkarte enthält die minimalen Manhattan–Distanzen zum nächsten Pixel des Farbwertes innerhalb einer Belegungsrasterkarte. Mit den erstellten Distanzkarten können die minimalen Manhattan–Distanzen zwischen den Rasterkarten durch ein einfaches Abfragen der Pixelposition ermittelt werden. Die Berechnung so einer Distanzkarte für einen bestimmten Farbwert f ist in Abbildung 5.11 dargestellt [142].

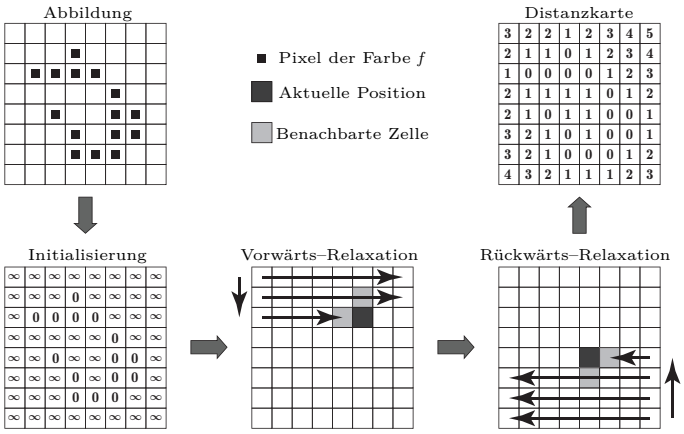


Abbildung 5.11: Visualisierung der Erstellung einer Distanzkarte für einen bestimmten Farbwert zur Berechnung der Picture–Distance–Function nach [157].

Zu Beginn der Erstellung ist eine Abbildung zu sehen, die eine bestimmte Anzahl an Pixeln der Farbe f enthält. Zur Erstellung der zugehörigen Distanzkarte wird eine Matrix derselben Größe initialisiert. Pixelzellen, welche im Bild mit dem entsprechenden Farbwert belegt sind, erhalten den Initialisierungswert 0. Alle anderen Pixel werden mit dem Wert ∞ belegt. Die Distanzkarte wird ausgehend von dieser Ausgangslage mittels zwei Relaxationsschritten berechnet. Die Distanzwerte werden zuerst in einem vorwärtsgewandten und anschließend in einem rückwärtsgewandten Schritt bestimmt. Der neue Zellwert der aktuell besuchten Zelle ist das Minimum der Werte von aktueller und den benachbarten Zellen. Die so erstellte Distanzkarte d_f enthält dann in jeder Pixelposition die minimale Manhattan–Distanz zum nächsten Pixel des Farbwertes f. Die Distanzkarten werden für die in einer Rasterkarte vorhandenen Farben generiert und bieten die Möglichkeit die Distanzen basierend auf Zellenposition für die jeweilige Farbe abzurufen [157]. Bei Belegungsrasterkarten werden die Belegungswahrscheinlichkeiten den Farbwerten $F = [0, 255]$ zugeordnet.

Die hier vorgestellten Metriken eignen sich für unterschiedliche Aufgaben und werden im folgenden Kapitel zur Eignung der Quantifizierung von physikalischen Sensormodellfehlern untersucht.

5.3 Umsetzung und Auswertung des Verfahrens

Bezugnehmend auf den Architekturvorschlag aus Abbildung 3.2 sind für die Validierung physikalischer Sensormodelle nicht alle Aspekte relevant. Um dies zu verdeutlichen, wurden in Abbildung 5.12 die nicht relevanten Anteile ausgegraut. Als erstes werden Referenz–, Umfeldsensor– und Umfeldmodelldaten erfasst, anschließend daraus die Fahrtrajektorien der relevanten Teilnehmer extrahiert und resimuliert. Als Ergebnis der Resimulation erhält man simulierte Referenztrajektorien, welche als Grundlage der Simulation des Sensorverhaltens verwendet werden. Das dabei eingesetzte Sensormo-

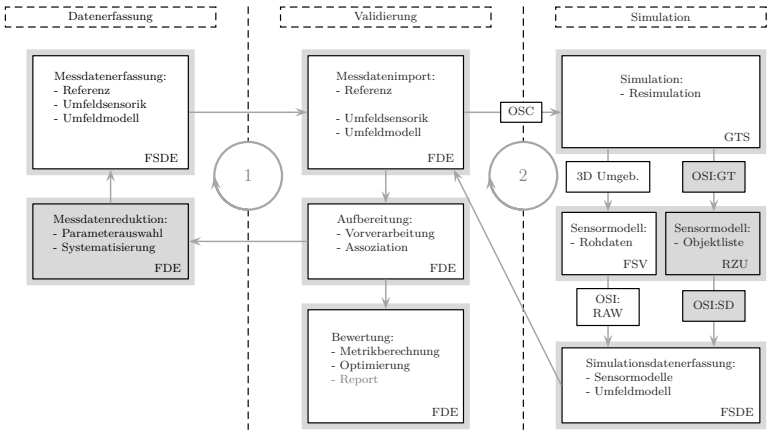


Abbildung 5.12: Relevante Anteile aus der Gesamtarchitektur für die Validierung physikalischer Sensormodelle. Nicht relevante Anteile sind ausgegraut.

dell wurde im vorherigen Abschnitt bereits vorgestellt. Im Anschluss werden simulierte Sensordaten im Umfeldmodell fusioniert sowie abschließend reale und synthetische Datensätze in das Evaluierungsframework importiert und bewertet.

Ähnlich zum modularen Prinzip des phänomenologischen Modells, kann auch das physikalische Sensormodell mehrere Effekte des Messprozesses abbilden, die zu– beziehungsweise abschaltbar sind. Im Rahmen dieser Arbeit werden folgende mögliche Modellfehler untersucht:

- Fehler in der modellierten Sensorgeometrie,
- systematische Translationsfehler, die beispielsweise durch eine falsch eingestellte Sensorposition entstehen,
- systematische Rotationsfehler in Roll–, Gier– und Nickwinkel,
- Objektverluste in Realität, die in der Simulation nicht abgebildet werden.

Damit sollen diverse Fehlkonfigurationen eines physikalischen Sensormodells auf ihre Auswirkung analysiert werden. Die verwendeten Modellkonfigurationen sind in Tabelle 5.1 zusammengefasst. Als erstes werden in den Konfigurationen 1 und 2 die geometrischen Eigenschaften des Modells untersucht. Anschließend werden eine falsch konfigurierte Sensorposition in den Konfigurationen 3 und 4, anschließend diverse Variationen in Roll–, Nick– und Gierwinkel in den Konfigurationen 5–10, sowie abschließend die Erkennung der Objektverluste in den Konfigurationen 11–15 analysiert.

Der erste Schritt zur Validierung des Sensormodells ist die Überprüfung der korrekten Abbildung des geometrischen Modellteils. Darunter wird die Implementierung des Messprinzips des Sensors verstanden oder, mit anderen Worten, ob die Strahlen eines Sensors in der Simulation aus geometrischer Sicht richtig ausgesendet werden. Sollte bereits dieser Teil des Sensormodells nicht der Realität entsprechen, ist es nicht sinnvoll, detailliertere Untersuchungen vorzunehmen. In dieser Auswertung soll untersucht werden, ob eine falsch implementierte Sensorgeometrie detektierbar ist. Folgende Schritte sind dafür erforderlich: (a) Generierung von Referenzstrahlen, die aus der Sensorgeo-

Tabelle 5.1: Verwendete Konfigurationen für die Analyse physikalischer Sensormodelle.

	Parameter	Konfig. 1,2	Konfig. 3,4	Konfig. 5,6	Konfig. 7,8	Konfig. 9,10	Konfig. 11–15
Konfiguration–unabhängig	Abstandsreichweite d in m			$2 \ldots 200$			
	Horizontaler Sichtbereich γ in °			130			
	Modellierungsgrößen $\mathbf{\Omega}_\mathrm{S}$			$d_\mathrm{x}^\mathrm{r}, d_\mathrm{y}^\mathrm{r}$, Umgebungsmodell			
	Zielgrößen $\boldsymbol{\zeta}_\mathrm{s}$			$\mathcal{P}_\mathrm{s}$			
Konfigurationsspezifisch	Abtastmaske	kartesisch/ sphärisch	sphärisch	sphärisch	sphärisch	sphärisch	sphärisch
	Verschiebung $(x;y)$ in m	$(0;0)$	$(0{,}1;0{,}1)$ $(0{,}5;0{,}5)$	$(0{,}0;0{,}0)$	$(0{,}0;0{,}0)$	$(0{,}0;0{,}0)$	$(0{,}0;0{,}0)$
	Sensorrotation Rollwinkel in °	0,0	0,0	1,0 3,0	0,0	0,0	0,0
	Sensorrotation Nickwinkel in °	0,0	0,0	0,0	1,0 3,0	0,0	0,0
	Sensorrotation Gierwinkel in °	0,0	0,0	0,0	0,0	1,0 3,0	0,0
	Objektverlust bei Abstand in m	inaktiv	inaktiv	inaktiv	inaktiv	inaktiv	{2,10,20,30,40}

metrie abgeleitet werden und (b) Erfassung und Vergleich der realen und resimulierten Punktewolken.

Wie einleitend in diesem Kapitel erläutert, senden Laserscanner die zur Entfernungsmessung genutzten Lichtstrahlen bei jedem Messzyklus in gleiche definierte Raumwinkel ihres Koordinatensystems aus. Diese Strahlen werden als Referenz zum Vergleich der realen und synthetischen Punktewolken herangezogen. Die Referenzstrahlen werden dabei aus den Eigenschaften des realen Laserscanners berechnet und sind in Abbildung 5.13 gezeigt. Dort sind die Referenzstrahlen der vier Abtastebenen in den Farben gelb, grün, blau und rot dargestellt. Die Sensoreigenschaften, aus denen diese resultieren, sind in Anhang A.7 zusammengefasst. Die Gesamtheit aller Referenzstrahlen ergibt den Erfassungsbereich des Sensors. Die erfassten Punktewolken müssen dabei nicht zwingend für jeden Referenzstrahl einen Messpunkt beinhalten. Sind jedoch zurückgeworfene Lichtstrahlen durch die Empfangselektronik registriert worden, so muss das verursachende Reflexionsobjekt, abgesehen von den hier vernachlässigten Mehrfachreflexionen, auf dem zugehörigen Referenzstrahl liegen. Die dabei vom Sensor erfassten Daten werden im ersten Schritt mittels Koordinatentransformation von kartesischen in Kugelkoordinaten umgerechnet. Dadurch erhält man für jeden Reflexionspunkt einen zugehörigen Azimut– und Elevationswinkel, wodurch sich der Punkt eindeutig einem Referenzstrahl zuordnen lässt. Diese Annahme wurde mittels Vergleich der Referenzstrahlen aus Abbildung 5.13 mit Daten eines realen Sensors verifiziert. Die detektierten Reflexionspunkte der aufgezeichneten Punktewolken stimmen dabei erwartungsgemäß exakt mit den Referenzstrahlen überein [142].

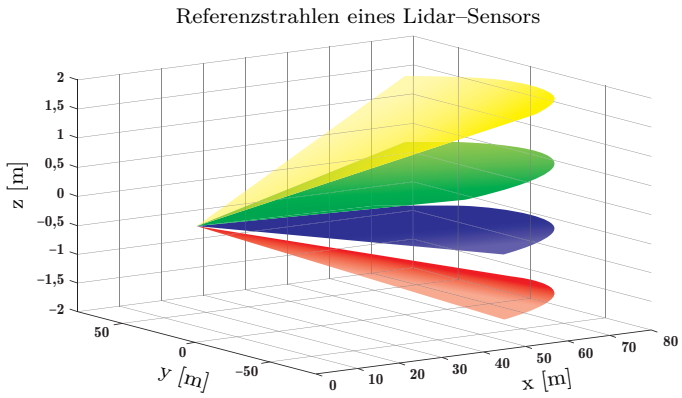


Abbildung 5.13: Visualisierung der Referenzstrahlen eines realen Sensors mit vier Abtastebenen nach [142].

Die Sensorgeometrie im Sensormodell wird mittels einer von der Strahlverfolgungsumgebung verwendeten Abtastmaske zur Erzeugung von Strahlen in der Simulation nachgestellt. Die Form dieser Maske muss dabei der des realen Sensors entsprechen. Für diese Untersuchung wurden zwei unterschiedliche Abtastmasken eingesetzt. Diese sind in Abbildung 5.14 dargestellt. Als erstes wird die im Strahlverfolgungsbereich meistverwendete planare Maske (siehe Abbildung 5.14a) eingesetzt, daraus eine synthetische Punktewolke generiert und mit den Referenzstrahlen aus Abbildung 5.13 verglichen. Die Ergebnisse für die planare Maske sind in Abbildung 5.15 zusammengefasst und mit M_1 gekennzeichnet. Dort sind die Abweichungen von den Referenzstrahlen für die vier Abtastebenen in Grad dargestellt. Zu erkennen ist, dass die in M_1 ausgesendeten Strahlen von den Referenzstrahlen umso mehr abweichen, je weiter diese von der Mitte entfernt sind. Der Grund hierfür ist die für die Strahlgenerierung falsch gewählte Abtastmaske. Die planare Maske entspricht nicht der Geometrie eines Lidar–Sensors, weshalb die Strahlen in falsche Richtungen gesendet werden. Im zweiten Durchlauf wird eine sphärische Maske (siehe Abbildung 5.14b) eingesetzt. Ähnlich wie zuvor, wird die synthetische Punktewolke mit den Referenzstrahlen verglichen. Die Ergebnisse für die sphärische Maske sind in Abbildung 5.15 mit M_2 gekennzeichnet. Hier ist zu erkennen, dass keiner der ausgesendeten Strahlen eine merkliche Abweichung von den Referenzstrahlen aufweist, was auf eine richtig gewählte Abtastmaske hindeutet. Mit diesem

(a) **(b)**

Abbildung 5.14: Visualisierung der verwendeten Abtastmasken für die Untersuchung der Nachbildung der Sensorgeometrie. (a) planare Maske, (b) sphärische Maske.

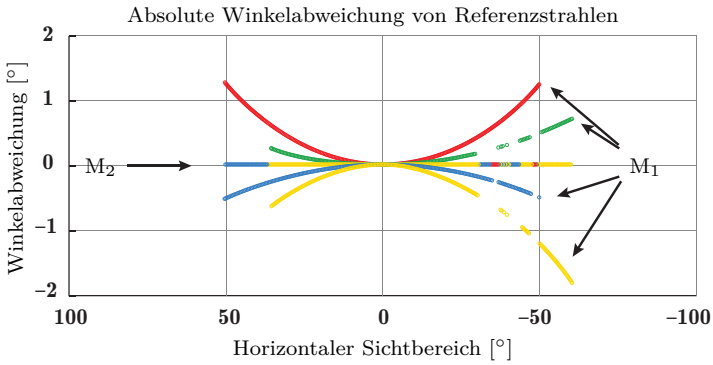


Abbildung 5.15: Visualisierung der Ergebnisse der Sensorgeometrieuntersuchung.

Vorgehen können nicht nur falsch gewählte, sondern auch falsch parametrierte Masken detektiert werden, bei denen beispielsweise die horizontalen oder vertikalen Winkel falsch eingestellt sind.

Im zweiten Schritt werden systematische Translations– und Rotationsfehler anhand der Konfigurationen 3–10 aus Tabelle 5.1 untersucht. Dafür wird ein landstraßenähnliches Szenario verwendet. Eine Momentaufnahme daraus ist in Abbildung 5.16 gezeigt. Dort sind auf der linken Seite eine Realkameraaufnahme sowie die entsprechende Belegungsrasterkarte abgebildet. Auf der rechten Seite sind korrespondierende simulierte Momentaufnahmen gezeigt. Es ist zu erkennen, dass in den Rasterkarten die Straßenkonturen, abgeleitet aus den Leitplanken, und das Fahrzeug qualitativ gute Übereinstimmung zwischen Realität und Simulation zeigen. Die Bereiche mit Vegetation

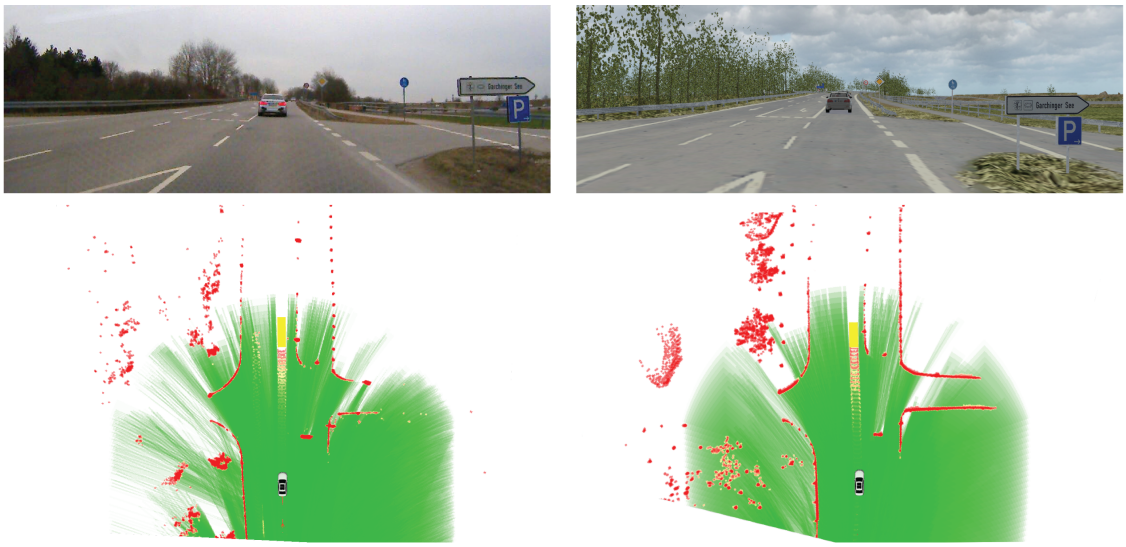


Abbildung 5.16: Momentaufnahme einer realen Fahrt links und einer simulierten Fahrt rechts. Oben sind die Kameraaufnahme und das gerenderte Bild, unten die jeweiligen Ergebnisse der Belegungsrasterkarten dargestellt [152].

(unterer linker Quadrant auf den Rasterkarten) weisen dagegen Unterschiede auf. Dies liegt daran, dass die Vegetationsobjekte aus zeitlichen und wirtschaftlichen Gründen nicht einzeln nachmodelliert werden können. Selbst wenn dies der Fall wäre, könnte in der Simulation deren Wachstum und jahreszeitbedingte Veränderungen kaum nachgestellt werden. Die Unterschiede in den Vegetationsbereichen sind deshalb in Kauf zu nehmen. Bis auf wenige Ausnahmen, wie beispielsweise landmarkenbasierte Lokalisierung, ist diese Einschränkung jedoch vertretbar, da diese sich in den meisten Fällen außerhalb der Straßenbereiche befinden.

Die Einzelaufnahmen der Euklidischen–Distanzkarten (Punktewolken–Ebene) und der Belegungsrasterkarten werden mittels der oben vorgestellten Metriken verglichen und die Einzelwerte für das gesamte Szenario gemittelt. Für die Picture–Distance–Function werden die Abstandswerte der Euklidischen–Distanzkarte und die Wahrscheinlichkeitswerte der Belegungsrasterkarte als Farbwerte interpretiert und dazu in `uint8` Werte überführt. Zu den Ergebnissen der Map–Score Metrik ist zu sagen, dass diese auf der Punktewolken–Ebene gegen den Wert 2.080 (520×4 Zellen) und auf der Rasterkarten–Ebene gegen den Wert 1.048.576 (1024×1024 Zellen) konvergieren, je ähnlicher die zu vergleichenden Matrizen sind. Die Ergebnisse der Auswertung sind in Tabelle 5.2 und im Anhang A.8 zusammengefasst. In Tabelle 5.2 sind dabei als Vergleichsbasis Metrikwerte eines ideales Modells sowie die Ergebnisse der Konfigurationen $3, 5, 7, 9$ und somit die jeweils kleineren Ausprägungen der untersuchten Fehler festgehalten. Im Anhang A.8 sind Resultate der Konfigurationen $4, 6, 8, 10$ und somit die jeweils größeren Ausprägungen zusammengefasst. Im Allgemeinen ist festzuhalten, dass die Absolutwerte der Metriken auf der Rasterkarten–Ebene um mindestens eine Größenordnung höher sind als die auf der Punktewolken–Ebene. Dies liegt an der deutlich höheren Anzahl der Zellen in einer Rasterkarte verglichen mit einer Euklidischen–Distanzkarte. Des Weiteren ist zu erkennen, dass sich die Map–Score Metrik bei allen getesteten Szenarien in ähnlicher Größenordnung befindet. Dies liegt daran, dass die Mehrzahl der Zellen in beiden unter Vergleich stehenden Karten (unabhängig ob

Tabelle 5.2: Ergebnisse der Analyse für das physikalischen Lidar–Sensormodell.

	Metrik	Ideales Modell	Konfig. 3	Konfig. 5	Konfig. 7	Konfig. 9
Punktewolke	M_{OE}	0,0	$7{,}594 \cdot 10^3$	$11{,}080 \cdot 10^3$	$14{,}383 \cdot 10^3$	$12{,}113 \cdot 10^3$
	M_{MS}	$2{,}080 \cdot 10^3$	$2{,}496 \cdot 10^3$	$2{,}502 \cdot 10^3$	$2{,}491 \cdot 10^3$	$2{,}496 \cdot 10^3$
	M_{RQS}	0,0	7,831	10,055	12,932	11,896
	M_{PD}	0,0	$0{,}944 \cdot 10^3$	$1{,}771 \cdot 10^3$	$1{,}684 \cdot 10^3$	$1{,}209 \cdot 10^3$
Rasterkarte	M_{OE}	0,0	$10{,}187 \cdot 10^6$	$3{,}223 \cdot 10^6$	$3{,}867 \cdot 10^6$	$16{,}372 \cdot 10^6$
	M_{MS}	$1{,}049 \cdot 10^6$	$1{,}237 \cdot 10^6$	$1{,}240 \cdot 10^6$	$1{,}242 \cdot 10^6$	$1{,}189 \cdot 10^6$
	M_{RQS}	0,0	$13{,}654 \cdot 10^3$	$3{,}134 \cdot 10^3$	$3{,}774 \cdot 10^3$	$18{,}733 \cdot 10^3$
	M_{PD}	0,0	$1{,}262 \cdot 10^4$	$1{,}247 \cdot 10^4$	$1{,}461 \cdot 10^4$	$5{,}883 \cdot 10^4$

Punktewolke– oder Rasterkarten–Ebene) unbekannt sind und deshalb die Sensitivität der Map–Score Metrik in Richtung der unbekannten Zellen verschoben ist. Bei Euklidischen–Distanzkarten ist hier die Mehrzahl der Zellen aufgrund des verwendeten Landstraßenszenarios mit viel Freifläche unbelegt (siehe Abbildung 5.16). Bei Belegungsrasterkarten ist die Mehrzahl der Zellen unbekannt, da die Laserstrahlen entweder keine Objekte treffen beziehungsweise die getroffenen Objekte meist sich in der Nähe des Eigenfahrzeugs befinden und die dahinter liegenden Zellen definitionsgemäß unbekannt sind (siehe Abbildung 5.9). Aufgrund der hohen Anzahl unbekannter Zellen, werden die belegten Zellen vernachlässigt. Zu einer ähnlichen Schlussfolgerung kommt [156], weshalb vorgeschlagen wird, nur die belegten Zellen bei der Berechnung der Map–Score in Betracht zu ziehen.

Auffallend ist, dass konfigurations– und ebenenunabhängig (Punktewolke beziehungsweise Belegungsrasterkarte) sich für die einzelnen Konfiguration alle Metrikergebnisse in einer ähnlichen Größenordnung liegen und gleiche Reihenfolge haben (M_{OE} ist beispielsweise am größten). Mit anderen Worten führen alle Metrikergebnisse zu gleicher Schlussfolgerung. Anders als bei der phänomenologischen Analyse ist es deshalb schwierig anhand gemittelter Ergebnisse unterschiedlicher Metriken Fehlerfälle zu identifizieren. Im Folgenden werden deshalb die Einzelwerte der Metriken anhand des Szenarioverlaufs begutachtet. Beispielhaft sind in Abbildung 5.17 die Ergebnisse der Picture–Distance–Function für Translation und Verkippung (Konfigurationen 3 und 7) jeweils auf der Punktewolken– und Rasterkarten–Ebene dargestellt. In Rot sind die Werte für die Euklidischen–Distanzkarten und in Blau für die Rasterkarten aufgetragen. Zu erkennen ist, dass die Abweichungsverläufe durchgehend relativ konstant sind und ähnliche Absolutwerte in beiden Fällen aufweisen. Ähnliches Verhalten ist für alle systematischen Testfälle (Konfigurationen 3–10) und Metriken zu beobachten. Daraus lässt sich schließen, dass systematische Positions– und Rotationsfehler anhand der vorgestellten Metriken und Ebenen zwar gut detektierbar, jedoch kaum voneinander unterscheidbar sind. Für die Analyse solcher Art von Fehlern wird deshalb die Verwendung eines

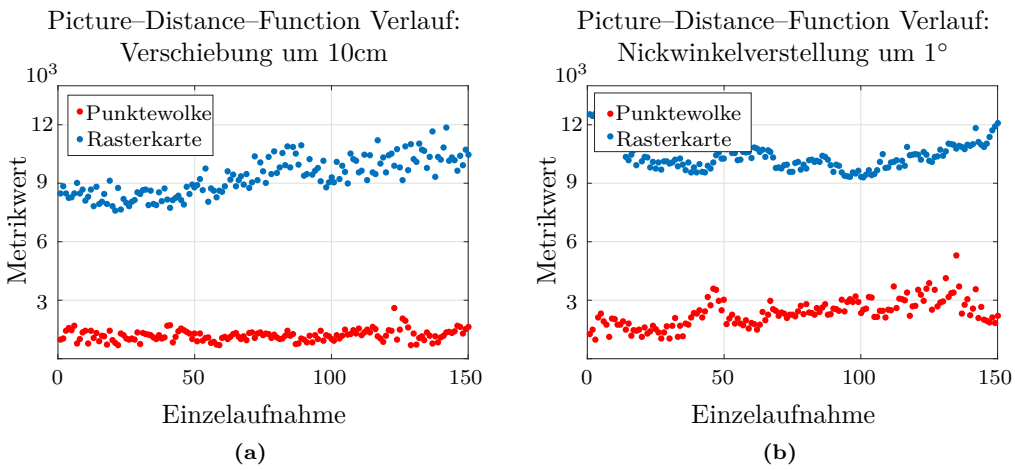


Abbildung 5.17: Verlauf der Picture–Distance–Function Werte für das Testszenario aus Konfigurationen 3 und 7. Links 10cm Translationsfehler und rechts 1° Nickwinkelfehler.

statischen Szenarios mit Referenzprüfkörpern empfohlen. Die reale Referenzszene wird dabei in der Simulation nachgestellt und im Anschluss gegen die Ergebnisse der realen Aufzeichnung basierend auf der Abstandsmessung verglichen [158].

Als nächstes wird untersucht, ob Objektverluste detektierbar sind. Dazu wird bei einer statischen Szene während der Simulation für etwa 100ms das Zielfahrzeug ausgeblendet. Zu erwarten ist, dass die Fähigkeit Objektverluste zu erkennen mit größer werdender Entfernung zwischen dem Eigen– und Zielfahrzeug abnimmt. Hier wird deshalb zusätzlich die Sensitivität der Metriken für unterschiedliche Abstände untersucht. Wie zuvor, werden für diese Analyse Metrikverläufe statt Mittelwerte hergenommen. In Abbildung 5.18a ist beispielhaft der Mittelwert bereinigte Verlauf der Picture–Distance–Function für das Szenario mit 2m Abstand zwischen Eigen– und Zielfahrzeug (Modellkonfiguration 11 aus Tabelle 5.1) gezeigt. Zu erkennen ist, dass die Objektverluste sowohl für die Euklidische–Distanzkarte als auch für die Belegungsrasterkarte gut detektierbar sind. Dabei liegen die absoluten Abweichungen auf der Rasterkarten–Ebene, wie auch bei den Untersuchungen zuvor, deutlich höher als auf der Punktewolken–Ebene. Der Grund hierfür ist die kleinere Anzahl an Zellen, die sich zwischen Realität und Simulation unterscheiden. Im Fall von Euklidischen–Distanzkarten sind es bei einem Objekt meist nur wenige Zellen, die der Anzahl von Lidar–Strahlen, die das betroffene Objekt treffen, entsprechen. Bei Rasterkarten ist diese Anzahl hingegen meist deutlich höher. Dabei hängt die Anzahl der betroffenen Zellen vom Abstand zwischen dem Eigenfahrzeug beziehungsweise dem Sensor und dem Zielobjekt. Je näher ein Objekt ist, desto mehr und je weiter entfernt desto weniger Strahlen treffen dieses Objekt.

Als nächstes soll deshalb untersucht werden, wie die Entfernung zwischen dem Eigenfahrzeug und einem Zielobjekt die Erkennung eines Objektverlustes auf Punktewolken– und Rasterkarten–Ebene beeinflusst. Dazu werden fünf Szenarien mit Abständen $2\text{m}, 10\text{m}, 20\text{m}, 30\text{m}$, und $40\text{m}\}$ (siehe Modellkonfigurationen 11–15 aus Tabelle

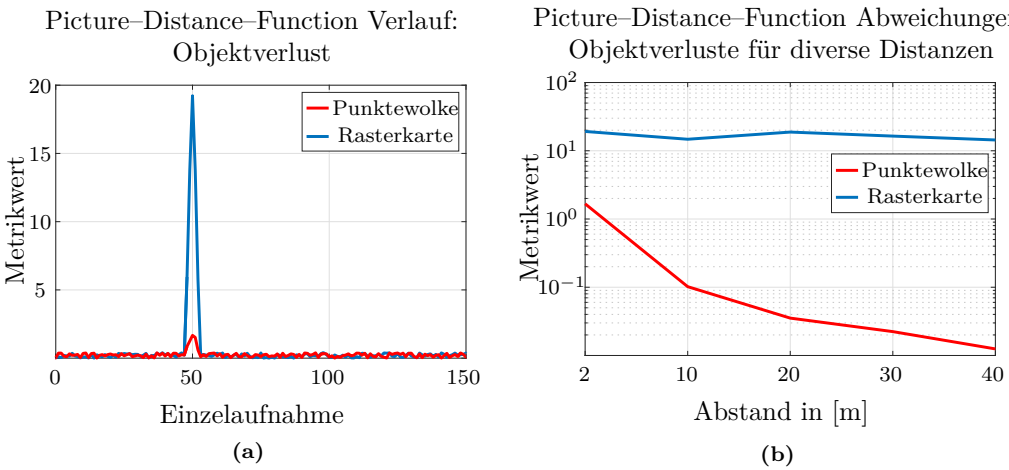


Abbildung 5.18: Verlauf der Picture–Distance–Function Werte für: (a) das Testszenario mit einem Objektverlust und (b) Metrikverlauf für Detektion mehrere Objektverluste bei unterschiedlichen Abständen zwischen Eigen– und Zielfahrzeug.

5.1) zwischen dem Eigen– und Zielfahrzeug untersucht. Die Ergebnisse sind in Abbildung 5.18b zusammengefasst. Auf logarithmischer Skala sind dort die Abweichungswerte der Picture–Distance–Function für Objektverluste in den zuvor erwähnten Entfernungen aufgetragen. In Rot sind die Abweichungen für die Euklidischen–Distanzkarten (Punktewolken–Ebene) und in Blau für die Rasterkarten dargestellt. Zu erkennen ist, dass für weit entfernte Objekte die Fähigkeit Objektverluste auf der Punktewolken–Ebene zu detektieren stark abnimmt, wohingegen auf der Rasterkarten–Ebene diese weitestgehend konstant bleibt. Ähnliche Ergebnisse zeigen andere Metriken.

Basierend auf den Ergebnissen aus diesem Kapitel lässt sich zusammenfassen, dass die Modellfehler systematischer Natur mittels der hier vorgestellten Metriken sich zwar gut detektieren, jedoch kaum von einander unterscheiden lassen. Empfehlenswert ist deshalb zur Erkennung von Fehlern, die aufgrund von Sensorverschiebung oder Rotation entstehen können, statische Szenarien in Kombination mit Prüfkörpern zu verwenden. Eine weitere Erkenntnis, ist die Notwendigkeit der ungefilterten Messungen für die Sensormodellvalidierung, da ansonsten nicht systematische Fehler wie beispielsweise Objektverluste übersehen werden können. Diese stellen aus Sicht des Gesamtsystems aus Abbildung 1.1 jedoch kritische Szenarien dar und müssen deshalb in Sensormodellen ebenfalls abgebildet werden.

5.4 Fazit zur Bewertung physikalischer Sensormodelle

Im ersten Teil dieses Kapitels wird eine Methode vorgestellt, mit der physikalische Eigenschaften eines Sensormodells analysiert werden können. Die zugrundeliegende Idee basiert auf dem Vergleich der Eigenschaften eines realen Sensors, der ungefilterter Sensormessungen ausgibt und eines physikalischen Sensormodells, das auf einem Strahlverfolgungsprinzip die ungefilterten Sensormessungen nachbildet. Die dabei verwendete Definition von Fehlerarten ähnelt der, die für phänomenologische Modelle verwendet wurde. Die im Rahmen dieser Arbeit ausgewerteten Fälle werden auf Sensorgeometrie, Sensorverschiebung, Sensorrotation sowie Objektverluste eingeschränkt.

Die Performanz physikalischer Sensormodelle weist eine starke Abhängigkeit von der Güte der Modellierung der virtuellen Umgebung auf. Um die notwendige Genauigkeit eines solchen Umgebungsmodells zu erreichen, wird ein Ansatz gewählt, der auf zwei Vermessungstechnologien basiert: Laserscanning und Photogrammetrie. Laserscanning liefert als Ergebnis eine real abgetastete Umgebung in Form einer Punktewolke zurück. Aus der Photogrammetrie resultiert ein fotorealistisches Umgebungsmodell, das sich aus mehreren Einzelbildaufnahmen ergibt. Durch die Kombination beider Verfahren wird ein fotorealistisches Umgebungsmodell erreicht, das einen absoluten Fehler < 5cm, verglichen mit zuvor vermessenen Kontrollpunkten, aufweist.

Das erstellte Umgebungsmodell wird im Anschluss verwendet, um typische Sensormodellfehler zu untersuchen. Die Sensorgeometrie wird basierend auf den aus dem Daten-

blatt eines realen Lidar–Sensors abgeleiteten Referenzstrahlen validiert. Damit können Fehler in der Sensorgeometrie detektiert und korrigiert werden. Diese Art von Tests ist essentiell, da eine falsch gewählte Sensorgeometrie alle folgenden Bewertungen nutzlos macht. Des Weiteren wurden falsch konfigurierte Sensorpositionen und –rotationen untersucht. Solche Fehler sind systematischer Natur, die sich basierend auf den vorgestellten Metriken zwar gut detektieren, jedoch kaum voneinander unterscheiden lassen. Für die Analyse solcher Fehler ist es deshalb empfehlenswert Geometrie basierte Tests unter Verwendung von Referenzkörpern einzusetzen.

Abschließend wurde untersucht, wie gut sich die nicht systematischen Ereignisse, wie Objektverluste, detektieren lassen. Als Resultat lässt sich schlussfolgern, dass die Analyse der Metrikverläufe über die Zeit dazu gut geeignet ist. Mittels Picture–Distance–Function können daraus nicht systematische Ereignisse basierend auf den Ausreißern im Verlauf der Metriken detektiert und zeitlich eingeschränkt werden. Des Weiteren sind zusätzlich zur Analyse auf Punktewolken–Ebene die daraus resultierenden Rasterkarten zu verwenden, da diese deutlich sensitiver auf Abweichungen zwischen Realität und Simulation reagieren. Insbesondere weit entfernte Zielobjekten können auf Punktewolken–Ebene übersehen werden.

Zusammenfassend liefern die Ergebnisse dieses Kapitels die Möglichkeit, physikalische Sensormodelle objektiv zu bewerten und optimale Sensorkonfigurationen basierend auf Sensorgeometrie und bildvergleichenden Metriken zu finden, was die Antwort auf die vierte und letzte Forschungsfrage ist.

6 Zusammenfassung und Ausblick

In diesem Kapitel wird die vorliegende Arbeit rekapituliert. Zunächst werden die gewonnenen Erkenntnisse dem bisherigen Stand der Technik entgegengesetzt. Im Anschluss daran werden die neu entwickelten Validierungsansätze zusammengefasst und bewertet. Abschließend folgt ein Ausblick der noch offenen Themen.

6.1 Zusammenfassung

Die vorliegende Arbeit beschäftigt sich mit der Validierung umfelderfassender Sensormodelle mit dem Schwerpunkt auf Automobilanwendungen. Der Umfang der Arbeit wird anhand von vier Forschungsfragen definiert. Die Antworten darauf werden im Folgenden zusammengefasst.

Als erstes wird die Frage, was bei der Validierung umfelderfassender Sensormodelle zu beachten ist und wie diese systematisch analysiert werden können, beantwortet. Dazu wird, basierend auf dem aktuellen Stand der Technik, eine generische Methode zur Sensormodellvalidierung abgeleitet, die ihren Fokus auf die Validierung umfelderfassender Sensormodelle legt. Der entscheidende Unterschied zum aktuellen Stand der Technik ist das zweischrittige Vorgehen, bei dem erst die Sensormodelle selbst untersucht werden und anschließend die darauf basierenden Ergebnisse des Umfeldmodells, die für reale und simulierte Sensordaten berechnet werden. Das Umfeldmodell ist im Rahmen dieser Arbeit ein verallgemeinernder Begriff für Fusionsalgorithmen. Der Vergleich anhand von Ergebnissen des Umfeldmodells bietet den Vorteil, dass sich die Validierung des Sensormodells mehr auf den Bereich der korrekten Abbildung der aus den Sensormessungen abgeleiteten Objekthypothesen konzentriert. Die Ergebnisse aus den Kapiteln 4 und 5 zeigen, dass mit dem vorgeschlagenen Vorgehen relevante Sensormodellfehler wie beispielsweise falsch konfigurierte Autokorrelationseigenschaften der phänomenologischen Sensormodelle oder Objektverluste bei physikalischen Sensormodellen deutlich besser identifizierbar sind, wohingegen auf der Sensormodellebene solche Fehler übersehen werden können.

In Kapitel 3 wird anschließend untersucht, wie sich die Anzahl der notwendigen Parameter und die Datenmenge für datengetriebene Sensormodelle reduzieren lassen. Die Datenerfassung und damit die Bedatung ist einer der teuersten Prozesse im Rahmen der Validierung und Bereitstellung datengetriebener Sensormodelle. Nichtsdestotrotz sind datengetriebene Modelle sehr vielversprechend und bieten viele Vorteile gegenüber den parametrischen. Im dritten Kapitel werden deshalb zwei Ansätze entwickelt, wie

die notwendige Menge an Daten für die Sensormodellbedatung reduziert werden kann. Die erste Möglichkeit ist die Eingrenzung und Reduzierung der für die Modellbildung notwendigen Eingangsparameter. Dazu wird ein aus der Variablen– beziehungsweise Merkmalauswahl motiviertes Vorgehen entwickelt. Das Verfahren basiert auf einem hybriden Filter–Wrapper–Parameterauswahlverfahren. Die Größe des Parametersets ist von besonderer Bedeutung, weil mit jedem Parameter, der für die Modellierung als relevant eingestuft wird, die Anzahl der zu erfassenden Daten exponentiell steigt. Die Auswertungen haben gezeigt, dass das entwickelte Verfahren tendenziell einen guten Kompromiss zwischen Parametersetgröße, Modellperformanz und Auswertungszeit erreicht, sodass in Einzelfällen sogar eine bessere Modellperformanz mit kleinerer Anzahl an Modellparametern und in kürzerer Auswertungszeit gefunden werden kann.

Des Weiteren kann zur Reduktion der zur Bedatung notwendigen Daten der Erfassungsprozess systematisiert werden, sodass nur die notwendige Menge an Daten erfasst wird und nicht mehr. Um einem Sensormodell in ausreichender Menge Messdaten zur Verfügung zu stellen, sollen idealerweise für den gesamten Sichtbereich für jede Parameterkombination in der jeweiligen Sensorauflösung Messpunkte vorliegen. Diese Anforderung ist jedoch kaum zu erfüllen. Es wird deshalb ein Ansatz entwickelt, der innerhalb des Sensorsichtbereichs Bereiche mit ähnlichen statistischen Eigenschaften findet, sodass ein reduzierter Datensatz ausreicht, um sie zu charakterisieren. Dazu wird der Sensorsichtbereich erst in kleine Kacheln gerastert und anschließend, basierend auf dem Vergleich der kumulativen Verteilungsfunktion benachbarter Zellen, entschieden, ob diese zusammengelegt werden können. Diese Methode bietet somit die Möglichkeit, den Datenerfassungsprozess gezielt zu steuern und dabei die notwendige Datenmenge zu reduzieren. Die Testergebnisse zeigen, dass die für die Modellbedatung erforderliche Datenmenge um mindestens circa 10% und im Mittel um etwa 20% reduziert werden konnte, ohne die Modellgüte maßgeblich negativ zu beeinflussen.

Im weiteren Verlauf der Arbeit wurden die Forschungsfragen zur objektiven Bewertung phänomenologischer und physikalischer Sensormodelle adressiert. Dazu wird zunächst in Kapitel 4 ein Ansatz zur Validierung phänomenologischer Sensormodelle entwickelt. Phänomenologische Modelle bilden Sensoreigenschaften auf statistischer Ebene mittels parametrischer und nicht–parametrischer Ansätze ab. Der vorgestellte Ansatz basiert dabei auf der hier entwickelten Validierungsmethode und zeigt dessen Praxistauglichkeit. Erst werden mögliche Sensorfehler klassifiziert und geeignete Metriken zur Quantifizierung der für die Validierung relevanten Sensorfehler vorgestellt. Im Rahmen der messfahrtbasierten Untersuchung von typischen Sensorfehlern, wie Netzwerklatenzen, Sensorpositionsverschiebungen und Abweichung von Fehlerverteilungen, werden, basierend auf diversen Testszenarien, die jeweils geeigneten Metriken identifiziert. Im Rahmen der Untersuchung wird auf Basis einer Autokorrelationsanalyse gezeigt, dass die zeitliche Filterung der Signale in die phänomenologischen Sensormodelle integriert werden muss. Mittels iterativer Anwendung der beiden Schritte der phänomenologischen Validierung (messfahrtbasierte Analyse und Autokorrelationsanalyse) kann ein phänomenologisches Sensormodell sukzessive verbessert werden, bis der gewünschte

Gütegrad erreicht ist.

Der letzte, in Kapitel 5 untersuchte Ansatz konzentriert sich auf die Validierung physikalischer Sensormodelle. Dabei handelt es um Ansätze, die die Sensoreigenschaften auf Ebene der Feld– beziehungsweise Strahlausbreitung mittels geometrischer Optik nachbilden. Als Werkzeug werden für die Implementierung solcher Modelle meist Strahlverfolgungsumgebungen, wie beispielsweise Nvidia®OptiX™, eingesetzt. Essentiell ist dabei die Umgebungsmodellierung, in der die Szenarien simuliert werden, denn die meisten Modellierungsverfahren leiten die Sensoreigenschaften ähnlich wie in Realität, basierend auf den Reflexionen der ausgesendeten Strahlen an den Oberflächen des zur Verfügung gestellten Umgebungsmodells, ab. Dazu wird ein Areal als Testumgebung, bestehend aus unterschiedlichen Straßentypen wie Stadt, Landstraße und Autobahn festgelegt, hochgenau vermessen und modelliert. Anschließend wird anhand eines Lidar–Modells untersucht, wie typische Sensormodellfehler aufgedeckt werden können. Dazu werden Fehlerfälle wie falsche Nachbildung der Sensorgeometrie, Sensorpositionsverschiebung und –rotation sowie Objektverluste definiert. Im Rahmen der Untersuchung wird gezeigt, wie auf Basis von Referenzstrahlen, abgeleitet aus der Sensorgeometrie, Fehler der in der Strahlverfolgung verwendeten Abtastmasken erkannt werden können. Weitere Ergebnisse zeigen, dass Translations– und Rotationsfehler, wie zum Beispiel eine Sensorverschiebung oder –verkippung, auf der Punktewolken– beziehungsweise Belegungsrasterkarten–Ebene zwar detektierbar jedoch kaum voneinander unterscheidbar sind, weshalb ein alternativer, auf Referenzkörper basierter, Ansatz vorgeschlagen wird. Die nicht systematischen Fehler, wie beispielsweise Objektverluste, können hingegen basierend auf den Metrikverläufen gut detektiert und eingeschränkt werden.

In Summe liefert die Sensormodellvalidierung Hilfestellungen um die Entscheidung bezüglich Validität eines Modells zu erleichtern und das Potenzial den Aufwand der Absicherung von Fahrerassistenzsystemen basierend auf virtueller Absichrung zu reduzieren. Obwohl in dieser Arbeit einige Methoden und Ansätze zur Validierung umfelderfassender Sensormodelle untersucht und entwickelt wurden, ist im Allgemeinen festzuhalten, dass die Ergebnisse der Sensormodellvalidierung sehr anwendungsspezifisch sind und es schwierig bis unmöglich ist alle potenziellen Unstimmigkeiten zwischen Realität und Simulation aufzudecken.

6.2 Ausblick und abschließende Empfehlung

Die vorliegende Arbeit soll als Anregung dienen das Thema Sensormodellvalidierung noch detaillierter zu untersuchen, um die virtuelle Absicherung im Allgemeinen einen Schritt nach vorne zu bringen. Zum jetzigen Zeitpunkt bleibt die Frage offen, ob es zielführend ist erst die Simulationsansätze selbst zu validieren und basierend darauf die eigentlichen Fahrerassistenzsysteme abzusichern. Im Folgenden wird deshalb eine Reihe an offenen Fragen adressiert, welche in Zukunft noch zu untersuchen sind.

Für die Validierung sind im Allgemeinen die Realdaten bestehend aus Referenz– und Sensormessung zwingend erforderlich. In dieser Arbeit wird vorgeschlagen, die Referenzdaten basierend auf der DGNSS Technologie zu generieren. Dabei müssen alle an der Testfahrt teilnehmenden Fahrzeuge mit zusätzlichen Sensoren ausgestattet werden. Aufgrund dessen, dass die Fahrsituationen sehr komplex sein können und nach Möglichkeit alle Verkehrsteilnehmer und statische Objekte referenziert werden sollen, stellt das in dieser Arbeit eingesetztes Referenzverfahren einen limitierenden Faktor dar. Der nächste erforderliche Schritt ist deshalb zu untersuchen, wie die Referenz für alle Objekte in einem Szenario generiert werden kann, ohne dabei alle Verkehrsteilnehmer mit zusätzlichen Referenzsensoren auszustatten.

Bei den strahlenoptischen Modellierungsansätzen ist auf lange Sicht die höchste Realitätstreue zu erwarten. Diese sind jedoch auch mit anderen Herausforderungen konfrontiert. Es wurde zwar bereits gezeigt, dass solche Modelle echtzeitfähig sein können, sehr aufwändig bei dieser Art von Modellen ist jedoch die Erstellung des Umgebungsmodells. Je detaillierter so ein Umgebungsmodell ist, desto realitätsnäher werden die Ergebnisse des Sensormodells sein. Es wird gezeigt, wie eine reale Umgebung nachmodelliert werden kann, diese Lösung ist jedoch für eine flächendeckende Modellierung, von zum Beispiel ganz Deutschland, aus wirtschaftlicher Sicht nicht geeignet. Diese Tatsache stellt somit einen Engpass sowohl für die Modellierung als auch für die Validierung dar. Als nächstes soll deshalb untersucht werden, ob entweder alternative, skalierbare Ansätze für die Umgebungsmodellerstellung oder andere Verfahren für die Generierung von ungefilterten Messungen entwickelt werden können. Es sollte beispielsweise untersucht werden, ob solche Umgebungsmodelle automatisiert erstellt werden können und wie sich die damit eventuell einhergehende Reduktion der Datenqualität auf die Performanz der Sensormodelle auswirkt.

Die Ansätze in dieser Arbeit beschränken sich auf die Validierung einzelner Sensoren beziehungsweise der fusionierten Ergebnisse von zwei einzelnen Sensoren. In heutigen Fahrerassistenzsystemen wird der Einsatz einzelner Sensoren kaum reichen. Es werden in der Regel Sensorverbunde, die auf Redundanz unterschiedlicher Messverfahren und Sensoreinheiten basieren, eingesetzt. Der nächste logische Schritt ist deshalb, sich die Validierung eines Sensorverbundes näher anzuschauen.

Auf Parameterebene sollte der Umfang der Auswertung ebenfalls erweitert werden. Die hier vorgestellten Verfahren berücksichtigen Verläufe und Eigenschaften einzelner Ausgangsparameter wie beispielsweise Abstand in x– oder y–Richtung, lassen die Korrelation und mehrdimensionale Fehlerverteilungen mehrere Modellierungsparameter jedoch außer Acht. Dies ist ebenfalls ein Aspekt, der eine detailliertere Auswertung erfordert.

Als Grundlage der hier vorgestellten Methode wird die gesamthafte Betrachtung des Sensordatenflusses gelegt. Demzufolge werden nicht nur der Sensorausgang, sondern auch die Ergebnisse des Umfeldmodells berücksichtigt. Aus Sicht eines Fahrerassistenz-

systems ist dies jedoch nicht ausreichend, um eine Validität des Gesamtsystems zu gewährleisten. Der nächste erforderliche Schritt ist deshalb die hier angefangene Betrachtungsweise auf das Gesamtsystem auszuweiten.

In eine ähnliche Richtung geht die Ausweitung der Validierung der hier betrachteten physikalischen Aspekte auf diverse andere Eigenschaften, die heutige Sensoren zur Verfügung stellen. Definitionsgemäß handelt Validierung von physikalischen Effekten, was auch die Grundlage dieser Arbeit darstellt. Die Modellierung rein physikalischer Effekte wird jedoch nicht ausreichen, um das Verhalten und alle Eigenschaften heutiger Sensoren vollumfänglich abzubilden. Neben den physikalischen Eigenschaften stellt die Sensorsignalverarbeitung zahlreiche abgeleitete Eigenschaften, wie beispielsweise die Objektklassifizierung, zur Verfügung. Solche Eigenschaften werden meistens ebenfalls mit statistischen Modellen abgebildet und erfordern daher auch einen Validierungsschritt.

A Anhang

A.1 Ausschnitt des Testdatensatzes für Auswahl von Modellierungsgrößen

Ausschnitt des Testdatensatzes, der während der Auswahl der Modellierungsgrößen verwendet wurde. Die Bedeutungen der einzelnen Parameter sind im Symbolverzeichnis aufgelistet.

$$
\begin{array}{c}
 \\ t_1 \\ t_2 \\ \vdots \\ t_n
\end{array}
\begin{array}{c}
\begin{array}{ccccccccccc}
d_\mathrm{x} & d_\mathrm{y} & v_\mathrm{x} & v_\mathrm{y} & v_{,\mathrm{x}}^{\mathrm{rel}} & v_{,\mathrm{y}}^{\mathrm{rel}} & a_\mathrm{x} & a_\mathrm{y} & \gamma & \theta & \rho
\end{array} \\
\begin{bmatrix}
5{,}5 & 4{,}5 & 37{,}2 & -0{,}8 & 7{,}7 & -0{,}1 & -0{,}2 & 0{,}4 & -0{,}1 & 0{,}7 & 7{,}1 \\
5{,}9 & 4{,}6 & 37{,}3 & -0{,}9 & 7{,}6 & -0{,}1 & -0{,}6 & 0{,}5 & -0{,}0 & 0{,}6 & 7{,}4 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
51{,}5 & -0{,}4 & 23{,}5 & -0{,}4 & -3{,}6 & -0{,}4 & -1{,}9 & -0{,}1 & -0{,}1 & -0{,}1 & 51{,}0
\end{bmatrix}
\end{array}
$$

Abbildung A.1: Visualisierung der Daten für die Evaluierung der Parameterrelevanz.

A.2 Ergebnisse der Auswahl von Modellierungsgrößen

In diesem Kapitel werden die Ergebnisse der Modellierungsgrößenauswahl für einen Lidar–Sensor und eine Kamera in Bezug auf δ_x und δ_y als Zielgrößen zusammengefasst.

Tabelle A.1: Ergebnisse der Parameterauswahl am Beispiel eines **Lidar–Sensors und $\boldsymbol{\delta_\mathrm{x}}$ als Zielgröße** basierend auf Anzahl der ausgewählten Parameter, mittleren quadratischen Abweichung und normierten benötigten Zeit. Zur Auswertung wurden Filter– und Wraper–Methoden einzeln, sowie der hybride Ansatz verwendet.

		Modellierungsgröße	MQF	Zeit
Filter	Pearson–Korrelation (M_P)	$\boldsymbol{d_\mathrm{x}}$, $\boldsymbol{v_\mathrm{x}^\mathrm{rel}}$	1,6096	$4{,}417 \cdot 10^{-4}$
	Gemeinsame Information (GI)	d_x, d_y, v_x, v_y, $v_\mathrm{x}^\mathrm{rel}$	**1,5874**	$\mathbf{0{,}6920 \cdot 10^{-4}}$
	PK ∩ GI	d_x, d_y, v_x, v_y, $v_\mathrm{x}^\mathrm{rel}$	1,5888	$5{,}1100 \cdot 10^{-4}$
Wrapper	Hybrid	d_x, d_y, v_x, v_y, $v_\mathrm{x}^\mathrm{rel}$	1,5895	1,0000
	SRS	d_x, d_y, v_x, $v_\mathrm{x}^\mathrm{rel}$, γ, θ, ρ	1,5902	6,9050

Tabelle A.2: Ergebnisse der Parameterauswahl am Beispiel einer **Kamera und $\boldsymbol{\delta_\mathrm{y}}$ als Zielgröße** basierend auf Anzahl der ausgewählten Parameter, mittleren quadratischen Abweichung und normierten benötigten Zeit. Zur Auswertung wurden Filter– und Wraper–Methoden einzeln, sowie der hybride Ansatz verwendet.

		Modellierungsgröße	MQF	Zeit
Filter	Pearson–Korrelation (M_P)	v_x, v_y, $v_\mathrm{x}^\mathrm{rel}$, γ, ρ	4,0529	$6{,}0430 \cdot 10^{-5}$
	Gemeinsame Information (M_I)	$\boldsymbol{d_\mathrm{y}}$, $\boldsymbol{v_\mathrm{x}}$, $\boldsymbol{\rho}$	3,4140	$\mathbf{4{,}4980 \cdot 10^{-5}}$
	PK ∩ GI	d_y, v_x, v_y, $v_\mathrm{x}^\mathrm{rel}$, γ, ρ	3,8113	$10{,}540 \cdot 10^{-5}$
Wrapper	Hybrid	d_y, v_x, $v_\mathrm{x}^\mathrm{rel}$, γ, ρ	3,5956	1,0000
	SRS	d_x, d_y, v_x, $v_\mathrm{x}^\mathrm{rel}$, θ, ρ	**3,3753**	4,0008

Tabelle A.3: Ergebnisse der Parameterauswahl am Beispiel einer **Kamera und $\boldsymbol{\delta_x}$ als Zielgröße** basierend auf Anzahl der ausgewählten Parameter, mittleren quadratischen Abweichung und normierten benötigten Zeit. Zur Auswertung wurden Filter– und Wraper–Methoden einzeln, sowie der hybride Ansatz verwendet.

		Modellierungsgröße	MQF	Zeit
Filter	Pearson–Korrelation (M_P)	$\boldsymbol{d_\mathrm{y}}$, $\boldsymbol{v_\mathrm{x}^\mathrm{rel}}$, $\boldsymbol{\rho}$	20,297	$5{,}5740 \cdot 10^{-5}$
	Gemeinsame Information (M_I)	d_x, $v_\mathrm{x}^\mathrm{rel}$, $v_\mathrm{y}^\mathrm{rel}$, a_x, ρ	20,288	$\mathbf{3{,}8850 \cdot 10^{-5}}$
	PK $\cap$ GI	d_x, d_x, $v_\mathrm{x}^\mathrm{rel}$, $v_\mathrm{y}^\mathrm{rel}$, a_x, ρ	**20,267**	$9{,}4590 \cdot 10^{-5}$
Wrapper	Hybrid	d_x, d_y, $v_\mathrm{x}^\mathrm{rel}$, ρ	20,386	1,0000
	SRS	d_x, d_y, v_x, $v_\mathrm{x}^\mathrm{rel}$, γ, θ	20,297	2,7560

A.3 Algorithmus zur Rasterung des Sensorsichtbereichs

In diesem Kapitel wird das Vorgehen zur Rasterung des Sensorsichtbereichs im Algorithmus 1 formalisiert. Visualisierung des Algorithmus ist in Abbildung 3.6 dargestellt und die einzelnen Schritte in Kommentaren erläutert.

Algorithm 1 Rasterung des Sensorsichtbereichs

Initialisierung: $\mathrm{r}_{\min}, \mathrm{r}_{\max}, \mathrm{r}_{\epsilon}, \alpha_{\min}, \alpha_{\max}, \alpha_{\epsilon}, \mathrm{n}_{\min}$ ▷ Für Sensorsichtbereich spezifische Parameter
Angabe: $\mathcal{S}_{\mathrm{I}}$ ▷ Initial gerasterte Matrix mit Datenpunkten
Ergebnis: $\mathcal{S}_{\mathrm{R}}$ ▷ Final gerasterte Matrix mit Datenpunkten

```
 1: S_R = zeros(size(S_I))                                  ▷ Initialisiere S_R mit Nullen
 2: raster = true                                           ▷ Initialisiere raster Flag mit true
 3: while raster do                                         ▷ Suche solange raster Flag auf true
 4:     for i_r = r_min : r_ε : r_max − r_ε do              ▷ Iteriere über Zellen in Abstandrichtung
 5:         for i_α = α_min : α_ε : α_max − α_ε do          ▷ Iteriere über Zellen in Winkelrichtung
 6:             if length(S_I(i_r, i_α)) < n_min then       ▷ Überprüfe min. Datenanzahl der aktuellen Zelle
 7:                 continue                                ▷ Gehe zu der nächsten Zelle
 8:             else
 9:                 if S_R(i_r, i_α) == 0 then              ▷ Überprüfe ob der Zelle noch keine Klasse zugeordnet ist
10:                     S_R(i_r, i_α) == max(S_R) + 1       ▷ Setze die Klasse der aktuellen Zelle
11:                 end if
12:
13:                 if S_I(i_r, i_α + 1) > n_min then       ▷ Überprüfe min. Datenanzahl der nächsten Winkelzelle
14:                     accept = test(S_I(i_r, i_α), S_I(i_r, i_α + 1))   ▷ Teste benachbarte Winkelzellen auf Ähnlichkeit
15:                     if accept then                      ▷ Überprüfe ob Zellen ähnlich sind
16:                         S_R(i_r, i_α + 1) = S_R(i_r, i_α)   ▷ Setze Klasse der nächsten Winkelzelle
17:                     end if
18:                 end if
19:
20:                 if S_I(i_r + 1, i_α) > n_min then       ▷ Überprüfe min. Datenanzahl der nächsten Abstandszelle
21:                     accept = test(S_I(i_r, i_α), S_I(i_r + 1, i_α))   ▷ Teste benachbarte Abstandszellen auf Ähnlichkeit
22:                     if accept then                      ▷ Überprüfe ob Zellen ähnlich sind
23:                         S_R(i_r + 1, i_α) = S_R(i_r, i_α)   ▷ Setze Klasse der nächsten Abstandszelle
24:                     end if
25:                 end if
26:
27:             end if
28:         end for
29:     end for
30:
31:     if not change(S_R) then                            ▷ Überprüfe ob S_R sich geändert hat
32:         raster = false                                 ▷ Setze raster Flag auf false
33:     end if
34: end while
35: return S_R
```

A.4 Der DTW Algorithmus

In diesem Kapitel wird der DTW Algorithmus formalisiert. Visualisierung des Algorithmus ist in Abbildung 4.4 dargestellt und die einzelnen Schritte in Kommentaren erläutert.

Algorithm 2 Standardform des DTW Algorithmus

Angabe: v_1, v_2 ▷ Zeitreihen v_1 und v_2 der Längen m und n die zu vergleichen sind.
Ergebnis: $M_{\mathrm{D}}(m, n)$ ▷ Ähnlichkeitsmatrix der Größe $m \times n$ mit „Abständen“ zwischen v_1 und v_2.

```
M_D := [1 : m, 1 : n]                          ▷ Definition der Ähnlichkeitsmatrix.
M_D(1, 1) = 0                                  ▷ Initialisierung des ersten Eintrags der Ähnlichkeitsmatrix mit „Abstand“ 0.
for i = 2 to m do                              ▷ Initialisierung der restlichen Einträge der Ähnlichkeitsmatrix mit „Abstand“ ∞.
    M_D(1, i) = ∞
end for
for j = 2 to n do
    M_D(j, 1) = ∞
end for
for i = 2 to m do                              ▷ Inkrementelles Füllen der Ähnlichkeitsmatrix
    for j = 2 to n do                          ▷ mit „Abständen“ zwischen Vektoren v_1 und v_2.
        c = cost(v_1(i), v_2(j))               ▷ Methode zur Messung des „Abstandes“ zwischen v_1 und "v_2.
        M_D(i, j) = c + min(                   ▷ Fülle die Ähnlichkeitsmatrix mit minimalem „Abstand“ aus
                M_D(i − 1, j),                 ▷ vorwärts Schritt,
                M_D(i, j − 1),                 ▷ rückwärts Schritt und
                M_D(i − 1, j − 1)              ▷ Match.
                )
    end for
end for
return M_D
```

A.5 Ergebnisse der messfahrtbasierten Analyse

In diesem Kapitel werden die Ergebnisse der messfahrtbasierten Analyse für einen Lidar–Sensor, Kamera und Fusion aus Kapitel 4.1 zusammengefasst.

Tabelle A.4: Ergebnisse für synthetische Datensätze eines phänomenologischen Lidar–Sensormodell in y–Richtung.

Metrik	Konfig. 1	Konfig. 2	Konfig. 3	Konfig. 4	Konfig. 5
M_{V_1}	88,300	88,740	441,59	258,38	303,32
M_{V_2}	2,9715	4,1462	18,858	10,982	12,803
M_{V_∞}	0,1000	0,4217	2,1269	1,2618	1,4424
M_{P}	1,0000	0,9977	0,9554	0,9843	0,9788
M_{K}	0,9999	0,9992	0,9848	0,9948	0,9929
M_{DTW}	26,973	0,2403	267,54	163,92	174,32
M_{KL}	0,0014	0,0141	0,0236	0,0037	0,0289
M_{F}	0,0999	0,0519	0,1673	0,1007	0,0288
M_{J}	0,1384	0,3087	0,6213	0,4501	0,5009

Tabelle A.5: Ergebnisse für reale Datensätze für phänomenologische Lidar– und Kamera–Sensormodelle sowie für die Fusion in y–Richtung.

	Metrik	Konfig. 6	Konfig. 7	Konfig. 8
Lidar	M_{V_1}	472,98	428,05	429,14
	M_{V_2}	18,976	15,651	15,749
	M_{V_∞}	1,8548	0,9517	0,8769
	M_{DTW}	269,85	112,83	103,09
	M_{KL}	0,0923	1,7874	0,0801
	M_{F}	0,1307	0,1689	0,1241
Kamera	M_{V_1}	669,74	600,90	601,95
	M_{V_2}	22,566	18,380	18,588
	M_{V_∞}	2,0381	1,0513	1,1907
	M_{DTW}	392,42	152,14	155,79
	M_{KL}	0,0793	1,7898	0,0775
	M_{F}	0,1805	0,1078	0,0635
Fusion	M_{V_1}	532,08	557,33	554,39
	M_{V_2}	23,164	34,882	27,598
	M_{V_∞}	2,9228	4,5471	3,2564
	M_{DTW}	197,41	238,92	155,80
	M_{KL}	0,1434	0,1204	0,2008
	M_{F}	0,0640	0,0705	0,0653

A.6 Zwischenschrittergebnisse des Umgebungsmodellierungsprozesses

In diesem Abschnitt werden die Zwischenergebnisse der Modellierung zu Dokumentationszwecken vorgestellt. Die Unterschriften der einzelnen Abbildungen beschreiben chronologisch die jeweiligen Modellierungsschritte.

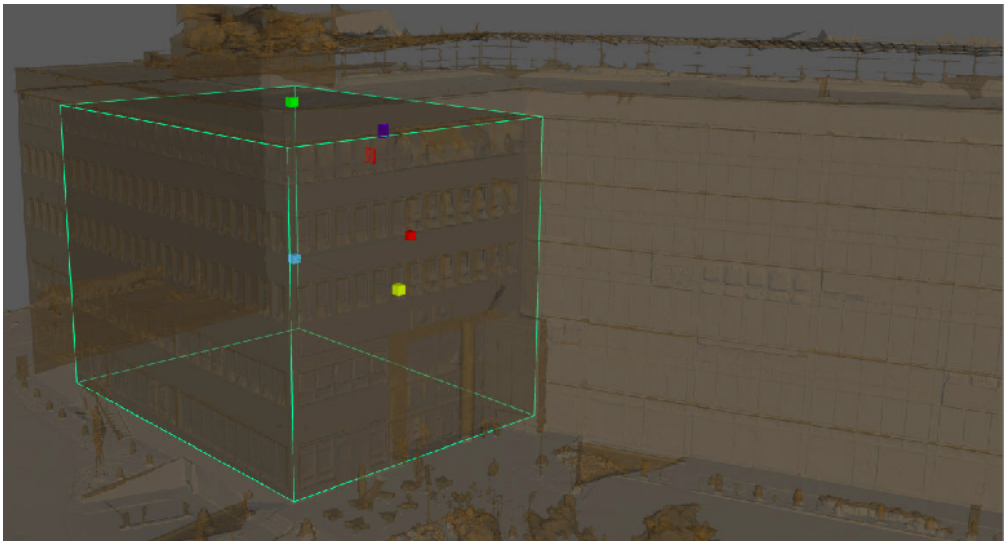


Abbildung A.2: Beispiel des groben Modellierungsschrittes [147].

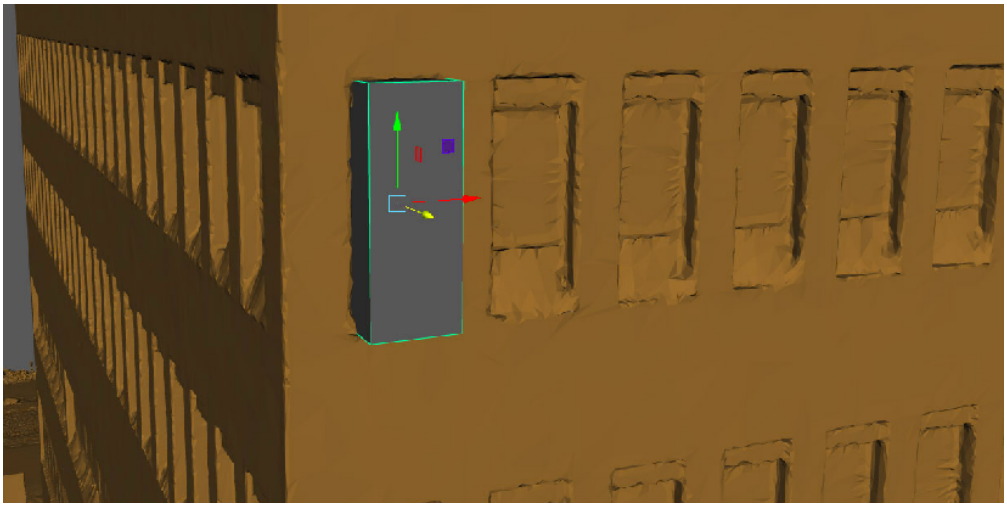


Abbildung A.3: Beispiel des Verfeinerungsschrittes [147].

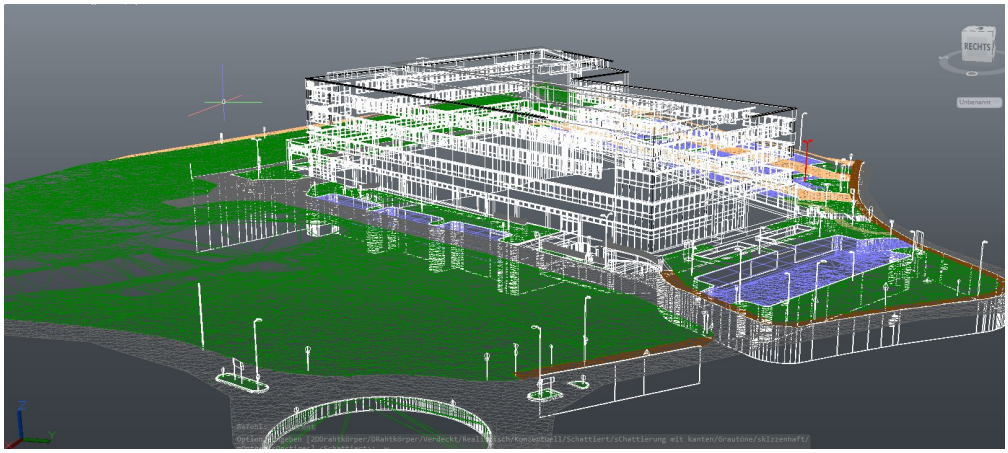


Abbildung A.4: Beispiel eines fertig modellierten Gebäudes auf Polygonebene [147].

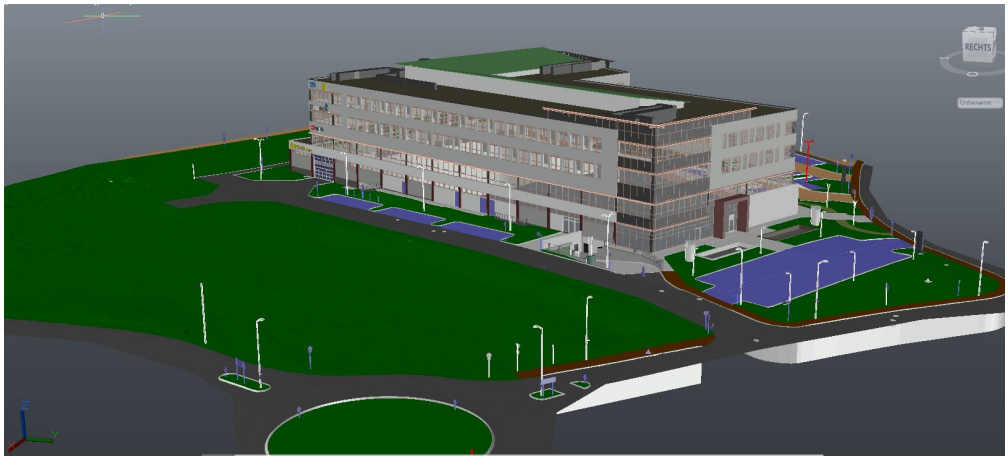


Abbildung A.5: Beispiel eines fertig modellierten Gebäudes auf Texturebene [147].

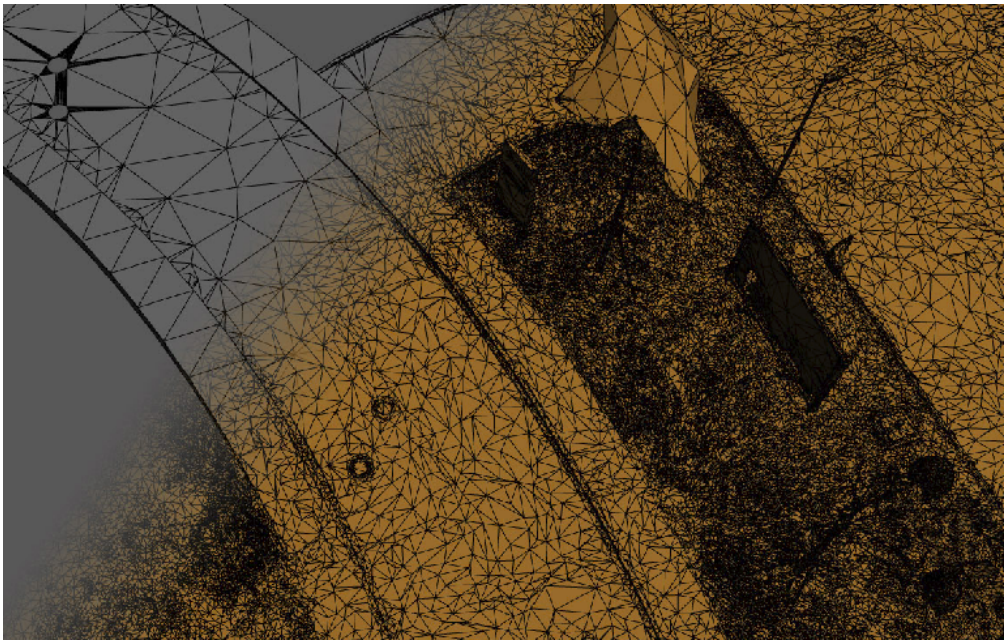


Abbildung A.6: Beispiel des Polygon–Optimierungsschrittes [147].

Abbildung A.7: Beispiel der Texturoptimierung aus Photogrammetrie [147].

Abbildung A.8: Visualisierung der Qualitätssicherung. Modell und ursprüngliche Punktewolke sind übereinander gelegt [147].

A.7 Eigenschaften des verwendeten Lidar–Sensors

Tabelle A.6: Eigenschaften des verwendeten Lidar–Sensors.

Eigenschaft	Wert
Horizontaler Sichtbereich	130 °
Horizontale Auflösung	0,25 °
Vertikaler Sichtbereich	3,2 °
Vertikale Auflösung	0,8 °
Abtastebenen	4 (parallel)

A.8 Ergebnisse der physikalischen Analyse eines Lidar–Sensormodells

In diesem Abschnitt werden die Ergebnisse der physikalischen Analyse eines Lidar–Sensormodells zusammengefasst. Die hier vorgestellten Ergebnisse basieren auf den Modellkonfigurationen 4,6,8, und 10 aus Tabelle 5.1.

Tabelle A.7: Ergebnisse der Analyse für das physikalischen Lidar–Sensormodell.

	Metrik	Ideales Modell	Konfig. 4	Konfig. 6	Konfig. 8	Konfig. 10
Punktewolke	M_{OE}	0,0	$11{,}828 \cdot 10^3$	$17{,}431 \cdot 10^3$	$17{,}710 \cdot 10^3$	$16{,}300 \cdot 10^3$
	M_{MS}	$2{,}080 \cdot 10^3$	$2{,}495 \cdot 10^3$	$2{,}519 \cdot 10^3$	$2{,}476 \cdot 10^3$	$2{,}501 \cdot 10^3$
	M_{RQS}	0,0	11,182	15,143	15,395	14,919
	M_{PD}	0,0	$1{,}537 \cdot 10^3$	$4{,}280 \cdot 10^3$	$2{,}627 \cdot 10^3$	$1{,}919 \cdot 10^3$
Rasterkarte	M_{OE}	0,0	$6{,}075 \cdot 10^6$	$4{,}917 \cdot 10^6$	$16{,}104 \cdot 10^6$	$18{,}277 \cdot 10^6$
	M_{MS}	$1{,}049 \cdot 10^6$	$1{,}240 \cdot 10^6$	$1{,}241 \cdot 10^6$	$1{,}241 \cdot 10^6$	$1{,}138 \cdot 10^6$
	M_{RQS}	0,0	$8{,}737 \cdot 10^3$	$5{,}041 \cdot 10^3$	$21{,}168 \cdot 10^3$	$22{,}264 \cdot 10^3$
	M_{PD}	0,0	$1{,}536 \cdot 10^4$	$1{,}883 \cdot 10^4$	$2{,}780 \cdot 10^4$	$5{,}277 \cdot 10^4$

Literaturverzeichnis

[1] F.A.Z. Autonomes Fahren: Die Debatte. http://www.faz.net/aktuell/wissen/live-debatte-auf-faz-net-wann-lenkt-der-computer-das-auto-15012006.html, 2017. [Online; aufgerufen am 15. Juni 2020].

[2] Asendorpf D. Das Auto-Auto und die Ethiker. https://www.zeit.de/2017/37/autonomes-fahren-regeln-kommission-ethik, 2017. [Online; aufgerufen am 15. Juni 2020].

[3] Wenzel F. Die Schuldfrage ist nicht geklärt. http://www.fr.de/politik/autonomes-fahren-die-schuldfrage-ist-nicht-geklaert-a-1471051, 2018. [Online; aufgerufen am 15. Juni 2020].

[4] Kemme M. Autonomes Fahren entlastet den Lieferverkehr. https://www.welt.de/wirtschaft/bilanz/article173817082/Autonomes-Fahren-als-Loesung-unserer-Mobilitaetsprobleme.html, 2018. [Online; aufgerufen am 15. Juni 2020].

[5] Maurer M. Einleitung. In *Autonomes Fahren: Technische, rechtliche und gesellschaftliche Aspekte: Einleitung*, chapter 1. Springer Vieweg, 2015.

[6] Sminkey L. Few countries have road safety laws addressing all five key risk factors. http://www.who.int/mediacentre/news/releases/2013/road_safety_20130314/en/, 2013. [Online; aufgerufen am 15. Juni 2020].

[7] DEKRA Automobil GmbH. DEKRA Verkehrssicherheitsreport 2017: Schritte zur Realisierung der Vision Zero. https://www.dvr.de/download/vision-zero-dekra-vsr-2017.pdf, 2017. [Online; aufgerufen am 15. Juni 2020].

[8] Lanctot R. Accelerating the Future: The Economic Impact of the Emerging Passenger Economy. https://newsroom.intel.com/newsroom/wp-content/uploads/sites/11/2017/05/passenger-economy.pdf, 2017. [Online; aufgerufen am 15. Juni 2020].

[9] Flämig H. Autonome Fahrzeuge und autonomes Fahren im Bereich des Gütertransportes. In *Autonomes Fahren: Technische, rechtliche und gesellschaftliche Aspekte*, chapter 18. Springer Vieweg, 2015.

[10] Dornier Consulting International. Autonomes Fahren Erwartungen an die Mobilität der Zukunf. https://www.dornier-consulting.com/wp-content/uploads/2017/04/DCI_Autonomes_Fahren.pdf, 2017. [Online; aufgerufen am 15. Juni 2020].

[11] Vetter P. Autonomes Fahren tötet. Warum? Weshalb? Egal! https://www.welt.de/wirtschaft/article174745972/Uber-Unfall-mit-selbstfahrendem-Fahrzeug-koennte-Technik-zurueckwerfen.html, 2018. [Online; aufgerufen am 15. Juni 2020].

[12] Klovert H. Tesla-Fahrer verunglückte mit eingeschaltetem Autopilot. http://www.spiegel.de/auto/aktuell/tesla-autopilot-war-bei-toedlichem-unfall-eingeschaltet-a-1200740.html, 2018. [Online; aufgerufen am 15. Juni 2020].

[13] ingenieur.de/lis. Immer wieder Unfälle mit autonomen Autos. https://www.ingenieur.de/technik/fachbereiche/fahrzeugbau/unfaelle-mit-autonomen-autos/, 2018. [Online; aufgerufen am 15. Juni 2020].

[14] Radio News. Radio Used As Fire Fighter. http://www.americanradiohistory.com/Archive-Radio-News/20s/Radio-News-1925-11-R.pdf, 2018. [Online; aufgerufen am 15. Juni 2020].

[15] Moore G. Cramming more components onto integrated circuits. *Electronics Magazine*, 1965.

[16] Winner H., Hakuli S., Lotz F., and C. Singer. *Handbuch Fahrerassistenzsysteme*. Springer Vieweg, 2015.

[17] Gasser T., Seeck A., and Smith B. Grundlagen, Komponenten und Systeme für aktive Sicherheit und Komfort. In *Handbuch Fahrerassistenzsysteme*, chapter 3. Springer Vieweg, 2015.

[18] Aeberhard M., Rauch S., Bahram M., Tanzmeister G., Thomas J., Pilat Y., Homm F., Huber W., and Kaempchen N. Experience, Results and Lessons Learned from Automated Driving on Germany's Highways. *IEEE Intelligent Transportation Systems Magazine*, pages 42–57, 2015.

[19] Wachenfeld W. and Winner H. The New Role of Road Testing for the Safety Validation of Automated Vehicles. In *Automated driving: Safer and more efficient future driving*, chapter 17. Springer International Publishing, 2017.

[20] Watzenig D. and Horn M. Introduction to Automated Driving. In *Automated driving: Safer and more efficient future driving*, chapter 1. Springer International Publishing, 2017.

[21] ISO. 26262: Road vehicles-Functional safety. Technical report, International Organization for Standardization, 2011.

[22] Paulweber M. Validation of Highly Automated Safe and Secure Systems. In *Automated driving: Safer and more efficient future driving*, chapter 18. Springer International Publishing, 2017.

[23] Ulbrich S., Schuldt F., Homeier K., Steinhoff M., Krause T., Menzel J., and M. Maurer. Validation of Highly Automated Safe and Secure Systems. *Automated driving: Safer and more efficient future driving*, 2017.

[24] Schaermann A., Hanke T., Geiger M., Weiler K., Hirsenkorn N., Rauch A., Schneider A., and Biebl E. Generation and Validation of Virtual Point Cloud Data for Automated Driving Systems. *2017 IEEE 20th International Conference on Intelligent Transportation Systems*, pages 762–767, 2017.

[25] Hanke T., Hirsenkorn N., Dehlink B., Rauch A., Rasshofer R., and Biebl E. Generic architecture for simulation of ADAS sensors. *International Radar Symposium*, pages 125–130, 2015.

[26] Hanke T., Hirsenkorn N., Dehlink B., Rauch A., Rasshofer R., and Biebl E. Classification of sensor errors for the statistical simulation of environmental perception in automated driving systems. *IEEE International Conference on Intelligent Transportation Systems*, pages 643–648, 2016.

[27] Hirsenkorn N., Hanke T., Rauch A., Dehlink B., Rasshofer R., and Biebl E. Virtual sensor models for real-time applications. *Advances in Radio Science*, pages 31–37, 2016.

[28] Hirsenkorn N., Hanke T., Rauch A., Dehlink B., Rasshofer R., and Biebl E. A non-parametric approach for modeling sensor behavior. *International Radar Symposium*, pages 131–136, 2015.

[29] Hirsenkorn N., Subkowski P., Hanke T., Schaermann A., Rauch A., Rasshofer R., and Biebl E. A Ray Launching Approach for Modeling an FMCW Radar System. *International Radar Symposium*, 2017.

[30] Hirsenkorn N., Kolsi H., Selmi M., Schaermann A., Hanke T., Rauch A., Rasshofer R., and Biebl E. Learning Sensor Models for Virtual Test and Development. *Workshop Fahrerassistenzsysteme und automatisiertes Fahren*, pages 115–124, 2017.

[31] Rasshofer R., Rank J., and Zhang G. Generalized modeling of radar sensors for next-generation virtual driver assistance function prototyping. *12th World Congress on Intelligent Transport Systems*, 2005.

[32] Buhren M. and Yang B. Simulation of Automotive Radar Target Lists using a Novel Approach of Object Representation. *2006 IEEE Intelligent Vehicles Symposium*, pages 314–319, 2006.

[33] Cao P., Wachenfeld W., and Winner H. Perception sensor modeling for virtual validation of automated driving. *Information Technology*, page 243–251, 2015.

[34] Wheeler T., Holder M., Winner H., and Kochenderfer M. Deep stochastic radar models. *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 47–53, 2017.

[35] Schuler K., Becker D., and Wiesbeck W. Extraction of Virtual Scattering Centers of Vehicles by Ray-Tracing Simulations. *IEEE Transactions on Antennas and Propagation*, pages 3543–3551, 2008.

[36] Hirsenkorn N. *Modellierung und Simulation der Fahrzeugunfeldsensorik*. PhD thesis, Technische Universität München, 12/2017.

[37] Bellman R. *Adaptive Control Processes: A Guided Tour*. Princeton University Press, 1961.

[38] Bishop C. *Pattern recognition and machine learning*. Springer Verlag, 2006.

[39] Thrun S., Burgard W., and Fox D. *Probabilistic robotics*. MIT Press, 2010.

[40] Tanzmeister G. and Wollherr D. Evidential Grid-Based Tracking and Mapping. *IEEE Transactions on Intelligent Transportation Systems*, 2016.

[41] Robin S. and Norman M. *Fahrerassistenzsysteme und Effiziente Antriebe*. Springer Verlag, 2015.

[42] Bernsteiner S., Zoltan Ferenc M., Lindvai-Soos D., and Eichberger A. Phänomenologisches Radarsensormodell zur Simulation längsdynamisch regelnder Fahrerassistenzsysteme. *16. Internationaler Kongress Elektronik im Fahrzeug*, 2013.

[43] Bernsteiner S., Magosi Z., Lindvai-Soos D., and Eichberger A. Radar sensor model for the virtual development process. *ATZelektronik worldwide*, pages 46–52, 2015.

[44] Kloeden H., Schwarz D., Biebl E., and R. Rasshofer. Effectiveness study of cooperative sensor systems for VRU-safety. *IEEE Intelligent Vehicles Symposium*, pages 879–884, 2012.

[45] van Dam J., Krose B., and Groen F. Adaptive sensor models. *Multisensor Fusion and Integration for Intelligent Systems*, pages 705–712, 1996.

[46] Jin J. *The finite element method in electromagnetics*. John Wiley & Sons, 2015.

[47] Balanis C. *Advanced engineering electromagnetics*. John Wiley & Sons, 1999.

[48] Maurer J. *Strahlenoptisches Kanalmodell für die Fahrzeug-Fahrzeug-Funkkommunikation*. PhD thesis, Universität Karlsruhe, 2005.

[49] Knörzer S. *Funkkanalmodellierung für OFDM-Kommunikationssysteme bei Hochgeschwindigkeitszügen*. PhD thesis, Karlsruher Institut für Technologie, 2009.

[50] Roth S. Ray casting for modeling solids. *Computer graphics and image processing*, pages 109–144, 1982.

[51] Borzęcki M., Skurski A., Kamiński M., Napieralski A., Kasprzak J., and Lipiec P. Applications of ray-casting in medical imaging. *Information Technologies in Biomedicine, Volume 3*, pages 3–14, 2014.

[52] Koopman P. and Wagner M. Challenges in Autonomous Vehicle Testing and Validation. *SAE International Journal of Transportation Safety*, pages 15–24, 2016.

[53] Wachenfeld W. and Winner H. Lernen autonome Fahrzeuge? In *Autonomes Fahren: Technische, rechtliche und gesellschaftliche Aspekte*, chapter 22. Springer Vieweg, 2015.

[54] Waymo. Waymo Safety Report: On the Road to Fully Self-Driving. `https://storage.googleapis.com/sdc-prod/v1/safety-report/waymo-safety-report-2017.pdf`, 2017. [Online; aufgerufen am 15. Juni 2020].

[55] O'Kane S. How Tesla and Waymo are tackling a major problem for self-driving cars: Data. `https://www.theverge.com/transportation/2018/4/19/17204044/tesla-waymo-self-driving-car-data-simulation`, 2018. [Online; aufgerufen am 15. Juni 2020].

[56] Oremus W. How Tesla Fixed a Deadly Flaw in Its Autopilot. `http://www.slate.com/articles/technology/future_tense/2016/09/how_tesla_s_software_update_fixed_a_deadly_flaw_in_autopilot.html`, 2017. [Online; aufgerufen am 15. Juni 2020].

[57] Hicks D. The Safety of Autonomous Vehicles: Lessons from Philosophy of Science. *IEEE Technology and Society Magazine*, pages 62–69, 2018.

[58] Coleman H. and Steele G. *Experimentation, validation, and uncertainty analysis for engineers*. John Wiley & Sons, 2009.

[59] NASA. Standard for Models and Simulations: An Implementation Guide for NASA-STD-7009. Technical report, National Aeronautics and Space Administration, 2013.

[60] DOD. Dodd 5000.59: Modeling and simulation (m&s) management. Technical report, U.S. Department of Defense Directive, 2007.

[61] Balci O. Verification, Validation, and Testing. In *Handbook of simulation: Principles, methodology, advances, applications, and practice*, chapter 19. Wiley and Co-published by Engineering & Management Press, 1998.

[62] Sargent R. Verification and validation of simulation models. *Journal of Simulation*, pages 12–24, 2012.

[63] Schwer L. Guide for verification and validation. *Computational Solid Mechanics*, 2009.

[64] Oberkampf W. and Trucano T. Verification and validation benchmarks. *Nuclear Engineering and Design*, pages 716–743, 2008.

[65] Oberkampf W. and Roy C. *Verification and validation in scientific computing.* Cambridge Univ. Press, 2010.

[66] Schlesinger S. Terminology for model credibility. *Simulation*, pages 103–104, 1979.

[67] Sommerville I. *Software engineering.* Pearson, 2011.

[68] William P. *Effective methods for software testing.* Wiley, 2006.

[69] Carlsson M. *Methods for early model validation: Applied on simulation models of aircraft vehicle systems.* Department of Management and Engineering, Linköping University, 2013.

[70] Liu Y. Toward a Better Understanding of Model Validation Metrics. *Journal of Mechanical Design*, July 2011.

[71] Oberkampf W. and Barone M. Measures of agreement between computation and experiment: Validation metrics. *Journal of Computational Physics*, pages 5–36, 2006.

[72] Ferson S., Oberkampf W., and Ginzburg L. Model validation and predictive capability for the thermal challenge problem. *Computer Methods in Applied Mechanics and Engineering*, pages 2408–2430, 2008.

[73] ASME. Guide for verification and validation in computational solid mechanics: Asme v&v 10-2006. Technical report, The American Society of Mechanical Engineers, 2006.

[74] Roth E., Dirndorfer T., Knoll A., Neuman-Cosel K., Ganslmeier T., Kern A., and Fischer M. Analysis and Validation of Perception Sensor Models in an Integrated Vehicle and Environment Simulation. *Enhanced Safety of Vehicle Conference*, 2011.

[75] Box G. Robustness in the strategy of scientific model building. *Robustness in statistics*, pages 201–236, 1979.

[76] Kalra N. and Groves D. *The Enemy of Good: Estimating the Cost of Waiting for Nearly Perfect Automated Vehicles.* RAND Corporation, 2017.

[77] Brahmi M., Siedersberger K., Siegel A., and Maurer M. Reference Systems for Environmental Perception: Requirements, Validation and Metric-based Evaluation. *6. Tagung Fahrerassistenzsysteme*, 2013.

[78] Sargent R. Verifying and Validating Simulation Models. *Winter Simulation Conference*, pages 118–131, 2014.

[79] Keimel C. *Design of Video Quality Metrics with Multi-Way Data Analysis*. Springer, 2016.

[80] Ferson S. and Oberkampf W. Validation of imprecise probability models. *International Journal of Reliability and Safety*, pages 3–22, 2009.

[81] Li W., Chen W., Jiang Z., Lu Z., and Liu Y. New validation metrics for models with multiple correlated responses. *Reliability Engineering & System Safety*, pages 1–11, 2014.

[82] L., Zhao, Z., Lu, W., Yun, and W., Wang. Validation metric based on Mahalanobis distance for models with multiple correlated responses. *Reliability Engineering & System Safety*, pages 80–89, 2017.

[83] Quehl J., Hu H., Tas O., Rehder E., and Lauer M. How Good is My Prediction? Finding a Similarity Measure for Trajectory Prediction Evaluation. *2017 IEEE 20th International Conference on Intelligent Transportation Systems*, pages 120–125, 2017.

[84] Sarin H., Kokkolaras M., Hulbert G., Papalambros P., and Yang S., Barbatand R. Comparing time histories for validation of simulation models: error measures and metrics. *Journal of dynamic systems, measurement, and control*, 2010.

[85] Shalizi C. *Advanced data analysis from an elementary point of view*. Cambridge University Press Cambridge, 2013.

[86] OXTS. RT3000: Hochleistungs-GNSS/INS für dynamische Anwendungen. `https://www.oxts.com/products/rt3000/?lang=de`, 2018. [Online; aufgerufen am 15. Juni 2020].

[87] Klandt J., Radimirsch M., Kirchner A., and Philipps H. Referenzierung von Fahrerassistenzsystemen. *VDI-Berichte 2166: Fahrerassistenz und Integrierte Sicherheit*, pages 215–225, 2012.

[88] Aeberhard M., Rauch A., Rabiega M., Kaempchen N., and Bertram T. Track-to-track fusion with asynchronous sensors and out-of-sequence tracks using information matrix fusion for advanced driver assistance systems. *IEEE Intelligent Vehicles Symposium*, 2012.

[89] Steyer S., Tanzmeister G., and Wollherr D. Grid-Based Environment Estimation Using Evidential Mapping and Particle Tracking. *IEEE Transactions on Intelligent Vehicles*, 2018.

[90] ROS. Robot Operating System. http://www.ros.org/, 2018. [Online; aufgerufen am 15. Juni 2020].

[91] Elektrobit. Automotive Data and Time-Triggered Framework. https://www.elektrobit.com/products/eb-assist/adtf/, 2018. [Online; aufgerufen am 15. Juni 2020].

[92] ViGEM. Vision go embedded. https://www.vigem.de/de/startseite, 2018. [Online; aufgerufen am 15. Juni 2020].

[93] Accenture. Autonomous Vehicles: The Race is on. https://www.accenture.com/t20180309T092359Z__w__/us-en/_acnmedia/PDF-73/Accenture-Autonomous-Vehicles-The-Race-Is-On.pdf, 2018. [Online; aufgerufen am 15. Juni 2020].

[94] Uwe Voelzke G., Rudolph. Three Sensor Types Drive Autonomous Vehicles. https://www.sensorsmag.com/components/three-sensor-types-drive-autonomous-vehicles, 2017. [Online; aufgerufen am 15. Juni 2020].

[95] Apache. Hadoop. http://hadoop.apache.org/, 2018. [Online; aufgerufen am 15. Juni 2020].

[96] MapR. MapR. https://mapr.com/, 2018. [Online; aufgerufen am 15. Juni 2020].

[97] Scipy.org. SciPy. https://www.scipy.org/, 2019. [Online; aufgerufen am 15. Juni 2020].

[98] Matlab. Matlab. https://de.mathworks.com/products/matlab.html, 2018. [Online; aufgerufen am 15. Juni 2020].

[99] VIRES Simulationstechnologie GmbH. OpenSCENARIO. http://www.openscenario.org/, 2018. [Online; aufgerufen am 15. Juni 2020].

[100] VIRES Simulationstechnologie GmbH. Virtual Test Drive. https://vires.com/vtd-vires-virtual-test-drive/, 2018. [Online; aufgerufen am 15. Juni 2020].

[101] IPG Automotive. CarMaker. https://ipg-automotive.com/de/produkte-services/simulation-software/carmaker/, 2018. [Online; aufgerufen am 15. Juni 2020].

[102] OpenSceneGraph. Open Scene Graph. http://www.openscenegraph.com/, 2018. [Online; aufgerufen am 15. Juni 2020].

[103] Hanke T., Hirsenkorn N., Garcia-Ramos P., Schiementz M., and Schneider S. Open Simulation Interface. `http://www.hot.ei.tum.de/forschung/automotive-veroeffentlichungen/`, 2017. [Online; aufgerufen am 15. Juni 2020].

[104] Nvidia. OptiX. `https://developer.nvidia.com/optix`, 2018. [Online; aufgerufen am 15. Juni 2020].

[105] Blender. Cycles. `https://www.blender.org/features/rendering/`, 2018. [Online; aufgerufen am 15. Juni 2020].

[106] OSI-Community. OSI-Packaging. `https://github.com/OpenSimulationInterface/osi-sensor-model-packaging`, 2018. [Online; aufgerufen am 15. Juni 2020].

[107] Cormen T., Leiserson C., Rivest R., and Stein C. *Introduction to algorithms*. The MIT Press, 2009.

[108] Kohavi R. and John G. Wrappers for feature subset selection. *Artificial intelligence*, pages 273–324, 1997.

[109] Guyon I. and Elisseeff A. An introduction to variable and feature selection. *Journal of machine learning research*, pages 1157–1182, 2003.

[110] Guyon I., Gunn S., M. Nikravesh, and L. Zadeh. *Feature Extraction: Foundations and Applications*. Springer, 2006.

[111] Mladenić D. Feature selection for dimensionality reduction. *Subspace, latent structure and feature selection*, pages 84–102, 2006.

[112] Frénay B., Doquire G., and Verleysen M. Is mutual information adequate for feature selection in regression? *Neural networks : the official journal of the International Neural Network Society*, 2013.

[113] Jungermann H. *Decision Making and Change in Human Affairs*. Springer Netherlands, 1977.

[114] Marill T. and Green D. On the effectiveness of receptors in recognition systems. *IEEE Transactions on Information Theory*, pages 11–17, 1963.

[115] Whitney A. A Direct Method of Nonparametric Measurement Selection. *IEEE Transactions on Computers*, pages 1100–1103, 1971.

[116] Stracuzzi D. and UtgoffE P. Randomized variable elimination. *Journal of Machine Learning Research*, pages 1331–1362, 2004.

[117] Reunanen J. A Pitfall in Determining the Optimal Feature Subset Size. *Pattern Recognition in Information Systems*, page 176, 2004.

[118] Jajuga K. and Walesiak M. Standardisation of Data Set under Different Measurement Scales. *Classification and Information Processing at the Turn of the Millennium*, pages 105–112, 2000.

[119] Hsu H., Hsieh C., and Lu M. Hybrid feature selection by combining filters and wrappers. *Expert Systems with Applications*, pages 8144–8150, 2011.

[120] Theodoridis S., Pikrakis A., Koutroumbas K., and Cavouras D. *Introduction to pattern recognition: a matlab approach*. Academic Press, 2010.

[121] Beckel C., Sadamori L., Staake T., and Santini C. Revealing household characteristics from smart meter data. *Energy*, pages 397–410, 2014.

[122] Kudo M. and Sklansky J. Comparison of algorithms that select features for pattern classifiers. *Pattern Recognition*, pages 25 – 41, 2000.

[123] Jain A. and Zongker D. Feature selection: evaluation, application, and small sample performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 153–158, 1997.

[124] Matlab. Statistics and Machine Learning Toolbox. `https://de.mathworks.com/products/statistics.html`, 2018. [Online; aufgerufen am 15. Juni 2020].

[125] INRIA. Scikit-learn: Machine Learning in Python. `https://scikit-learn.org/stable/index.html`, 2018. [Online; aufgerufen am 15. Juni 2020].

[126] Waikato University. Weka 3: Data Mining Software in Java. `https://www.cs.waikato.ac.nz/ml/weka/index.html`, 2018. [Online; aufgerufen am 15. Juni 2020].

[127] Rasmussen C. Gaussian processes in machine learning. *Advanced lectures on machine learning*, pages 63–71, 2004.

[128] Breiman L. Random Forests. *Machine Learning*, pages 5–32, 2001.

[129] Kanji G. *100 Statistical Tests*. Sage Publications Ltd, 2006.

[130] Massey F. The Kolmogorov-Smirnov test for goodness of fit. *Journal of the American statistical Association*, pages 68–78, 1951.

[131] Miller L. Table of Percentage Points of Kolmogorov Statistics. *Journal of the American Statistical Association*, pages 111–121, 1956.

[132] Gail M. and Green S. Critical Values for the One-Sided Two-Sample Kolmogorov-Smirnov Statistic. *Journal of the American Statistical Association*, pages 757–760, 1976.

[133] Siegel S. *Nonparametric statistics for the behavioral sciences*. McGraw-Hill and Kogakusha, 1956.

[134] Knuth D. *The art of computer programming: sorting and searching.* Pearson Education, 1997.

[135] Fasano G. and Franceschini A. A multidimensional version of the Kolmogorov–Smirnov test. *Monthly Notices of the Royal Astronomical Society.*

[136] Sakoe H. and Chiba S. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, pages 43–49, 1978.

[137] Sven Kosub. A note on the triangle inequality for the jaccard distance. *Pattern Recognition Letters*, 2019.

[138] Kalman R. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, pages 35–45, 1960.

[139] Markus Buehren. *Simulation und Verarbeitung von Radarziellisten im Automobil.* PhD thesis, Uniersität Stuttgart, 2008.

[140] Yaman Barlas. An autocorrelation function test for out put validation. *Simulation*, pages 7–16, 1990.

[141] Anderson O. Sampled serial correlations from ARIMA processes. *Applied time series analysis (North-Holland, Amsterdam)*, pages 5–14, 1982.

[142] K. Weiler. Validierung eines LiDAR-Sensormodells anhand von Punktewolken und Belegungsrasterkarten. Masterarbeit, Hochschule Kempten, 2017.

[143] Eberhardt D. Die Vorteile von Lidar-Sensoren im autonomen Fahrzeug. `https://www.next-mobility.news/die-vorteile-von-lidar-sensoren-im-autonomen-fahrzeug-a-710222/`, 2018. [Online; aufgerufen am 15. Juni 2020].

[144] Dietmayer K., Nuß D., and Reuter S. *Handbuch Fahrerassistenzsysteme: Repräsentation fusionierter Umfelddaten.* Springer Vieweg, 2015.

[145] G. Tanzmeister, J. Thomas, D. Wollherr, and M. Buss. Grid-based mapping and tracking in dynamic environments using a uniform evidential environment representation. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6090–6095, 2014.

[146] Grewe R. *Optimierung der ReprÄsentation von Occupancy-Grids fÜr Fahrerassistenzsysteme.* VDI-Verlag, 2014.

[147] Mit freundlicher Unterstützung von © BMW AG & ArcTron 3D GmbH, 2019.

[148] Paffenholz J. *Terrestrisches Laserscanning 2017.* DVW – Gesellschaft für Geodäsie, 2017.

[149] Schwidefsky K. and Ackermann F. *Photogrammetrie: Grundlagen, Verfahren, Anwendungen. 7 Auflage.* Teubner, 2013.

[150] Borrmann A., König M., Koch C., and Beetz J. *Building Information Modeling: Technologische Grundlagen und industrielle Praxis.* Springer-Verlag, 2015.

[151] Arctron 3D. Ingenieurbüro für 3D-Vermessung & Softwareentwicklung. `https://www.arctron.de/`, 2019. [Online; aufgerufen am 15. Juni 2020].

[152] Mit freundlicher Unterstützung von © BMW AG, 2019.

[153] Carlson J., Murphy R., Christopher S., and Casper J. Conflict metric as a measure of sensing quality. *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 2032–2039, 2005.

[154] Martin C. and Moravec P. Robot Evidence Grids. *Carnegie Mellon University*, 1996.

[155] Balaguer B., Balakirsky S., Carpin S., and Visser A. Evaluating maps produced by urban search and rescue robots: lessons learned from RoboCup. *Autonomous Robots*, 2009.

[156] Collins J., Eaton M., Mansfield M., Haskett D., and O'Sullivan S. Developing a benchmarking framework for map building paradigms. *Proceedings of the 9th International Symposium on Artificial Life and Robotics*, pages 614–617, 2004.

[157] Birk A. Learning Geometric Concepts with an Evolutionary Algorithm. *In Proceedings of The Fifth Annual Conference on Evolutionary Programming*, 1996.

[158] Tallavajhula A. *Lidar Simulation for Robotic Application Development: Modeling and Evaluation.* PhD thesis, Carnegie Mellon University, 2018.

Eigene Veröffentlichungen

[159] N. Hirsenkorn, P. Subkowski, T. Hanke, A. Schaermann, A. Rauch, R. Rasshofer, and E. Biebl. A ray launching approach for modeling an fmcw radar system. In *International Radar Symposium (IRS)*, 2017.

[160] N. Hirsenkorn, H. Kolsi, M. Selmi, A. Schaermann, T. Hanke, A. Rauch, R. Rasshofer, and E. Biebl. Learning Sensor Models for Virtual Test and Development. In *Workshop Fahrerassistenzsysteme und automatisiertes Fahren*, pages 115–124, 2017.

[161] A. Schaermann, A. Rauch, N. Hirsenkorn, T. Hanke, R. Rasshofer, and E. Biebl. Validation of vehicle environment sensor models. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 405–411, 2017.

[162] A. Schaermann, T. Hanke, M. Geiger, K. Weiler, N. Hirsenkorn, A. Rauch, A. Schneider, and E. Biebl. Generation and Validation of Virtual Point Cloud Data for Automated Driving Systems. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 762–767, 2017.

[163] A. Schaermann and T. Hanke. Generation and Validation of Sensor Models for Automated Driving Systems Using VIRES VTD. *Engineering Reality Magazine*, page 14–17, Jun 2019.

Abkürzungsverzeichnis

ADTF „Automotive Data and Time–Triggered Framework“

AKF Autokorrelationsfunktion

ASIL Automotive Sicherheitsanforderungsstufe (englisch: Automotive Safety Integrity Level)

BASt Bundesanstalt für Straßenwesen

KVF Kumulierte Verteilungsfunktion (englisch: Cumulative distribution function)

DGNSS Differential Global Navigation Satellite System

DTW Dynamic Time Warping

FAS Fahrerassistenzsystem

FDE Framework für Datenevaluierung

FLT OpenFlight

FSDE Framework für synchrone Datenerfassung

FSV Framework für Strahlverfolgung

GNSS Global Navigation Satellite System

GT Ground Truth

GTS Ground Truth Simulator

HIL Hardware in the Loop

ID Identifikationsnummer

IMU inertialen Messeinheit (englisch: Inertial Measurement Unit)

ISO International Organization for Standardization

IVE OpenSceneGraph

KLD Kullback–Leibler–Divergenz

KS Kolmogorov–Smirnov

LoD Level of Detail

Lidar Light Detection and Ranging

MIL Model in the Loop

RMSE mittleren quadratischen Fehler (englisch: Root–Mean–Square Error)

OSI Offene Simulation Schnittstelle (englisch: Open Simulation Interface)

OSC Open Scenario

Radar Radio Detection and Ranging

RAW Rohdaten (englisch: Raw data)

ROS „Robot Operating System“

RZU Realzeit–Umgebung

SD Sensordaten (englisch: Sensor Data)

SFRS sequentielle gleitende Rückwärts–Selektion (englisch: Sequential Floating Backward Selection)

SFVS sequentielle gleitende Vorwärtselektion (englisch: Sequential Floating Forward Selection)

SIL Software in the Loop

SRS sequentielle Rückwärts–Selektion (englisch: Sequential Backward Selection)

SRV Signal–Rausch–Verhältnis

SVS sequentielle Vorwärts–Selektion (englisch: Sequential Forward Selection)

V&V Verifikation und Validierung (englisch: Verification and Validation)

ViGEM „Vision go embedded“

VTD „Virtual Test Drive“

WDF Wahrscheinlichkeitsdichtefunktion

WLAN drahtloses lokales Netzwerk (englisch: Wireless Local Area Network)

Symbolverzeichnis

$\square^{\mathrm{r}}, \square_{\mathrm{r}}$	Reale, unbekannte Größe beziehungsweise Referenzgröße eines Referenzsensors. Eng.: Ground Truth (situationsabhängig Super–, oder Subskript)
$\square^{\mathrm{e}}, \square_{\mathrm{e}}$	Experimentelle, gemessene Größe (situationsabhängig Super–, oder Subskript)
$\square^{\mathrm{s}}, \square_{\mathrm{s}}$	Simulierte Größe (situationsabhängig Super–, oder Subskript)
$\square^{\mathrm{W}}$	Größe in Weltkoordinaten
$\square^{\mathrm{F}}$	Größe in Fahrzeugkoordinaten
$\square^{\mathrm{S}}$	Größe in Sensorkoordinaten
$\square^{(t)}$	Größe zum Zeitpunkt t
$\square^{\mathrm{rel}}$	relative Größe, z.B. $v_{\mathrm{x}}^{\mathrm{rel}}$: relative Geschwindigkeit in x–Richtung zwischen zwei Fahrzeugen
$\mathcal{M}$	Messdaten
Ω	Beschreibung der Umgebung. Ω_{r}: real, Ω_{s}: simuliert. Im Kontext von phänomenologischen Modellen, dynamische und statische Objekte eines Szenarios. Bei strahlenoptischen Modellen die Beschreibung der statischen Umgebung in Form von Oberflächentexturen mit hinterlegten Materialeigenschaften
χ	Zur Modellierung verwendete reduzierte Repräsentation der Umgebungsbeschreibung Ω
ζ	Beschreibung des Sensor- beziehungsweise Modellausgangsdatenformates. ζ_{r}: real (Ground Truth), ζ_{e}: experimentell, ζ_{s}: simuliert
O	Objektliste
o	Objektbeschreibung
t	Zeitstempel
id	Identifikationsnummer eines Objektes

$d_\square$	Abstand zum erkannten Ziel in longitudinaler x–, lateraler y– beziehungsweise z–Richtung bei einer Objektliste beziehungsweise einer Punktewolke
$\delta_\square$	Fehler in longitudinaler x–Richtung beziehungsweise lateraler y–Richtung
$\sigma_\square$	Geschätzte Standardabweichung der Position in longitudinaler x– beziehungsweise lateraler y–Richtung
$v_\square$	Absolute Geschwindigkeit eines erkannten Ziels in longitudinaler x– beziehungsweise lateraler y–Richtung
$a_\square$	Absolute Beschleunigung eines erkannten Ziels in longitudinaler x– beziehungsweise lateraler y–Richtung
l	Länge eines erkannten Objektes
b	Breite eines erkannten Objektes
γ	Gierwinkel
θ	Nickwinkel
ρ	Rollwinkel
$\mathcal{P}$	Punktewolke
$\mathcal{E}$	Fehlerwert aus der Validierung
T	Zeitlich verfolgtes Objekt
τ	Alle zeitlich verfolgten Objekte V einer Messaufzeichnung
A	Assoziation: Menge an zeitlich verfolgten Objekten V, die durch dieselbe Ursache hervorgerufen wurden
Γ	Menge an Assoziationen A
$M_\square$	Metrik z.B $M_\mathrm{P} \hat{=}$ Korrelationskoeffizient nach Pearson
ω	Modellierungsgröße
$P(\square)$	Wahrscheinlichkeitsdichte Funktion WDF
$C(\square)$	Kumulative Wahrscheinlichkeitsdichte Funktion KVF
$D_\square$	Diskrepanz zwischen zwei WDF beziehungsweise KVFen D_m: maximal, D_k: kritisch
α	Signifikanzniveau eines statistischen Tests
$\Delta_\square$	Auflösung einer Größe