Hochschule für Technik und Wirtschaft Dresden
Fakultät Maschinenbau

HTWD Hochschule für Technik und Wirtschaft Dresden University of Applied Sciences

Aufgabenblatt für die Diplomarbeit

im Studiengang / Studienrichtung: **Fahrzeugtechnik / Kraftfahrzeugtechnik**

Name des Diplomanden / Matrikel-Nr.: **Tom Irrasch / 49114**

Thema: **Entwicklung eines Funktions-Prototypen für die erweiterte Abfahrkontrolle autonomer Fahrzeuge laut AFGBV**

Beschreibung:

Die „Verordnung zur Genehmigung und zum Betrieb von Kraftfahrzeugen mit autonomer Fahrfunktion in festgelegten Betriebsbereichen (Autonome-Fahrzeuge-Genehmigungs-und-Betriebs-Verordnung - AFGBV)“ regelt die Randbedingungen für die Typzulassung und den Betrieb autonomer Fahrzeuge in Deutschland. Von besonderer Bedeutung sind dabei die konkreten Anforderungen für die täglich notwendige erweiterte Abfahrkontrolle. In der Diplomarbeit soll daher an einem Funktions-Prototyp auf Basis eines BMW i3 verschiedene neue Prüfabläufe und Testverfahren entwickelt und bewertet werden.

Folgende Arbeitspunkte sollen bearbeitet werden:

- Einarbeitung in die Grundlagen autonomer Fahrfunktionen
- Konzeption und Inbetriebnahme einer erweiterbaren automatisierten Längs- und Querführung für einen BMW i3 auf Basis von Matlab-Simulink
- Konzeption und Umsetzung verschiedener Prüfszenarien für eine sicherheitsrelevante autonome Fahrfunktion (automatische Notbremse)
- Durchführung und Bewertung von ausgewählten Testabläufen
- Erstellung einer generischen Anforderungsmethodik für weitere Prüfabläufe

Die Ergebnisse der Arbeit sind in einem Poster zu dokumentieren.

Betreuer HTW: Prof. Dr. rer. nat. Toralf Trautmann
Dipl.-Ing. (FH) Dirk Engert

Ausgehändigt am: 03.06.2024 Einzureichen bis: 04.11.2024

.....
Prof. Dr. rer. nat. Toralf Trautmann
Verantwortlicher Hochschullehrer

.....
Prof. Dr.-Ing. Thomas Himmer
Prüfungsausschussvorsitzender

Hinweise zum Erstellen der Diplomarbeit unter
https://www.htw-dresden.de/hochschule/fakultaeten/maschinenbau/studium/diplomsemester
Die Aufgabenstellung kann nach Absprache und dem Vorliegen von Teilergebnissen erweitert bzw. eingeengt werden.

Hochschule für Technik und
Wirtschaft Dresden
University of Applied Sciences

Diplomarbeit

Entwicklung eines Funktions-Prototypen für die erweiterte Abfahrkontrolle autonomer Fahrzeuge laut AFGBV

vorgelegt von:	Tom Irrasch
Ort:	HTW Dresden
Fakultät:	Maschinenbau
Studiengang:	Fahrzeugtechnik
Betreuer:	Prof. Dr. rer. nat. Toralf Trautmann Dipl.-Ing. (FH) Dirk Engert
Abgabedatum:	04.11.2024

Inhaltsverzeichnis

Abkürzungsverzeichnis

AFGBV	Verordnung zur Genehmigung und zum Betrieb von Kraftfahrzeugen mit autonomer Fahrfunktion in festgelegten Betriebsbereichen (Autonome-Fahrzeuge-Genehmigungs-und-Betriebs-Verordnung)
BUS	Binary Unit System
CAN	Controller Area Network
DOF	Degrees Of Freedom
EURO NCAP	The European New Car Assessment Programme
GPS	Global Positioning System
HTWD	Hochschule für Technik und Wirtschaft Dresden
ISO	International Organization for Standardization
LiDAR	Light Detection and Ranging
LIN	Local Interconnect Network
MEMS	Micro-Electro-Mechanical System
MOST	Media Oriented Systems Transport
Radar	Radio Detection and Ranging
SAE	Society of Automotive Engineers
StVO	Straßenverkehrs-Ordnung
StVZO	Straßenverkehrs-Zulassungs-Ordnung
UE	Unreal Engine
UN-ECE	United Nations Economic Commission for Europe

Verzeichnis der Formelzeichen und Symbole

Symbol	Einheit	Beschreibung
c	m/s, km/h	Ausbreitungsgeschwindigkeit
f	Hz, kHz, GHz	Frequenz
L	m	Radstand
l_d	m	Vorschaudistanz
M	[m,m]	Mittelpunkt
MRZ	[m,m]	momentanes Rotationszentrum
P	[m,m]	Beliebiger Punkt
R, r	m	Radius
t	s, ms	Zeit
v	m/s, km/h	Geschwindigkeit
x, y	m	Koordinaten
ZP	[m,m]	Zielpunkt
α, γ, ϕ	°, rad	Winkel
β	°	Schwimmwinkel
δ	°	Radlenkwinkel
ϑ	°, rad	Nickwinkel
λ	m, cm, mm	Wellenlänge
φ	°, rad	Wankwinkel
ψ	°, rad	Gierwinkel
$\dot{\psi}$	°/s, rad/s	Gierrate

Abbildungsverzeichnis

Tabellenverzeichnis

1. Einleitung

1.1. Motivation

Autonomes Fahren: Ein Thema, das immer mehr an Relevanz gewinnt. Nicht nur, dass immer mehr Berichte in den Medien auftauchen, sondern auch beim deutschen Patent- und Markenamt macht sich dies bemerkbar. Die veröffentlichten Infografiken zu den Jahren 2008–2017 auf der Website zeigen, dass bereits vor einigen Jahren die Patentanmeldungen in diesem oder eng dazugehörigen Bereichen stark angestiegen sind [1]. Viele große und auch kleine Unternehmen forschen in Deutschland und der ganzen Welt an autonomen Fahrzeugen oder an den dafür benötigten Technologien [2].

Auch die Forschungsabteilung „Mechlab“ der Hochschule für Technik und Wirtschaft Dresden (HTWD) plant, ein autonomes Transferfahrzeug zu entwickeln. Das Ziel ist es, die Hochschule mit dem nahegelegenen Hauptbahnhof zu verbinden und Personen sicher zu befördern. Zur Umsetzung dieses Projekts sind viele Teilschritte notwendig. Sowohl die technischen als auch die rechtlichen Anforderungen sind sehr komplex. Es gibt viele Vorschriften zu beachten, unter anderem die erst am 01.07.2022 in Kraft getretene Verordnung zur Genehmigung und zum Betrieb von Kraftfahrzeugen mit autonomer Fahrfunktion in festgelegten Betriebsbereichen (Autonome-Fahrzeuge-Genehmigungs-und-Betriebs-Verordnung) (AFGBV). Sie regelt beispielsweise, wo und unter welchen Voraussetzungen die Fahrzeuge genehmigt und betrieben werden dürfen. Eine dieser Voraussetzungen ist, dass jedes im Straßenverkehr zugelassene, autonome Fahrzeug täglich vor Fahrtantritt eine Probefahrt und eine anschließende Durchsicht zu absolvieren hat [3]. Um den Prozess zu beschleunigen und die Effizienz zu erhöhen, soll dieser Teil weitgehend automatisiert werden.

1.2. Zielstellung

Ziel dieser Diplomarbeit ist es, erste Grundlagen für die Umsetzung dieses Prozesses zu schaffen. Dafür stellt die Hochschule ein Fahrzeug zur Verfügung, das bereits für das fahrerlose Fahren umgerüstet ist, jedoch keine Software-Modelle zum Absolvieren der Probefahrt besitzt. Im ersten Schritt soll das Fahrzeug mit einer erweiterbaren automatisierten Längs- und Querführung ausgestattet werden. Weiterhin soll, als erste autonome Fahrfunktion, eine automatische Notbremse entwickelt und realisiert werden. Für die Überprüfung dieser oder zukünftiger autonomer Fahrfunktionen soll abschließend ein allgemein gültiges Prüfkonzept erarbeitet und Tests durchgeführt werden.

2. Theoretische Grundlagen

Dieses Kapitel soll einen grundlegenden Überblick über das Thema autonomes Fahren verschaffen. Dabei werden wichtige Gesetze angesprochen, grundlegende theoretische Ansätze der Fahrzeugbewegung erläutert und auf ausgewählte Regelverfahren eingegangen, die für PKWs allgemein oder speziell für deren autonome Fahrfunktionen wichtig sind.

2.1. Überblick autonomes Fahren

Um autonomes Fahren und die damit verbundenen Fahrfunktionen zu ermöglichen und sicher zu gestalten, sind klare Vorschriften und Gesetze sowie verschiedene Technologien erforderlich.

In Deutschland gelten für autonome Fahrzeuge nicht nur die bekannte Straßenverkehrs-Ordnung (StVO) und die Straßenverkehrs-Zulassungs-Ordnung (StVZO), sondern auch die AFGBV. Sie legt die rechtlichen Grundlagen für autonomes Fahren fest. Auf europäischer Ebene sind zusätzliche rechtliche Anforderungen zu beachten, darunter die EU-Verordnung 2019/2144, die eng mit der AFGBV verknüpft ist und sich hauptsächlich auf die Verbesserung der Verkehrssicherheit sowie die Reduzierung von Umweltbelastungen durch Fahrzeuge konzentriert. Weltweit gibt es jedoch kaum einheitliche Vorschriften oder Gesetze zum autonomen Fahren. Verschiedene Organisationen, darunter die United Nations Economic Commission for Europe (UN-ECE), die International Organization for Standardization (ISO) und die Society of Automotive Engineers (SAE) International, bemühen sich um die Einführung harmonisierter Standards und Richtlinien [3],[4],[5],[6].

Bereits heute sind zahlreiche Fahrzeuge mit autonomen Fahrsystemen ausgestattet. Diese Systeme sind bereits zum Teil Pflicht und unterstützen den Fahrer. Sie können in kritischen Situationen eigenständig Entscheidungen treffen und das Fahrzeug kurzfristig steuern. Beispiele hierfür sind der Notbremsassistent, der Spurhalteassistent und der Ausweichassistent. Der Notbremsassistent bremst automatisch, sobald eine Gefahr erkannt wird und der Fahrer nicht reagiert. Der Spurhalteassistent hält das Fahrzeug durch gezielte Gegenbewegungen in der Spur, wenn der Fahrer unaufmerksam ist und beispielsweise über den Mittelstreifen fährt. Der Ausweichassistent verlässt kurzzeitig die Spur, um einer Gefahr auszuweichen, die vom Fahrer nicht wahrgenommen wird. Wichtig ist, dass der Fahrer jederzeit die Kontrolle über das Fahrzeug behalten und diese Systeme überstimmen kann, da er für das Verhalten des Fahrzeugs verantwortlich ist. Derzeit dienen diese Systeme primär der Verbesserung der Sicherheit im Straßenverkehr. Zukünftig

sollen sie so weiterentwickelt und verknüpft werden, dass kein Fahrer mehr benötigt und die Kontrolle dauerhaft vollständig an das Fahrzeug abgegeben wird [3],[4].

Zur Orientierung auf diesem Weg hat die SAE die Norm J3016 veröffentlicht. Sie beschreibt die Einteilung von Fahrzeugen nach ihrem Automatisierungsgrad in sechs Hauptbereiche. Diese Automatisierungsstufen sind in Abbildung 2.1 dargestellt.

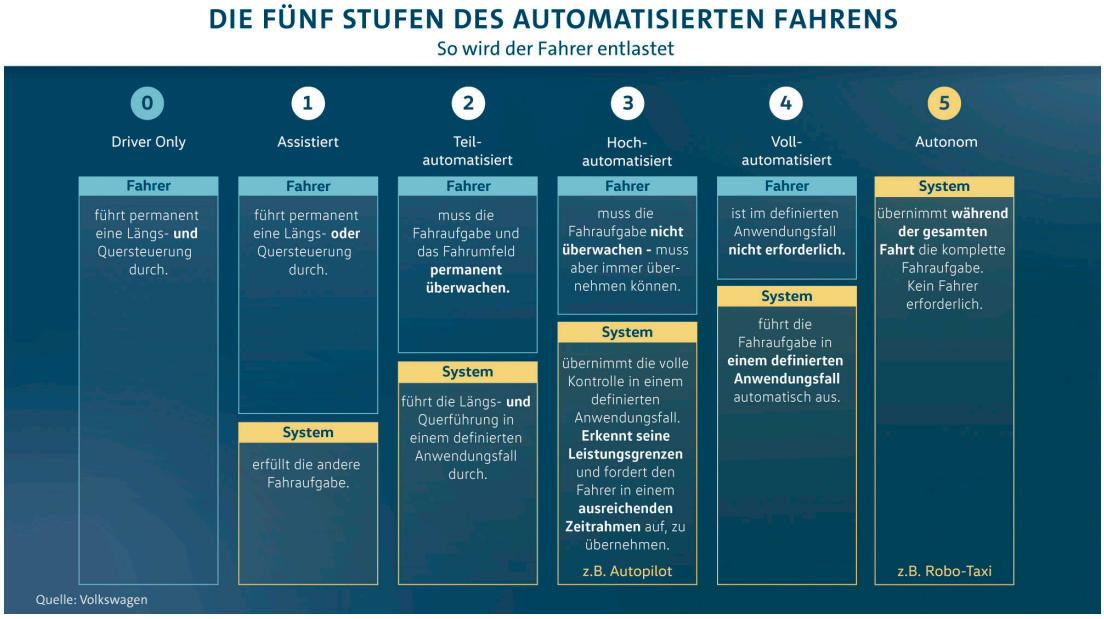


Abbildung 2.1.: Stufen des automatisierten Fahrens nach SAE J3016 [7]

Darin wird deutlich, dass mit zunehmender Automatisierungsstufe die Verantwortung und Kontrolle des Fahrers abnimmt, während die des Fahrzeugs zunimmt. Die heutigen Systeme, wie der Notbremsassistent, Spurhalteassistent und Ausweichassistent, entsprechen dabei typischerweise den Anforderungen der Level 2 oder niedriger. Diese Systeme unterstützen den Fahrer, übernehmen jedoch nicht die vollständige Kontrolle des Fahrzeugs. Fahrzeuge, die die Anforderungen für Level 3 oder höher erfüllen, sind derzeit noch selten. Im Optimalfall sollen die Systeme in Zukunft so weiterentwickelt werden, dass schließlich Level 5 erreicht wird, bei dem vollumfänglich kein Fahrer mehr benötigt wird [8].

Zusammenfassend versucht die Kombination aus der Einhaltung gesetzlicher Vorschriften und technischen Fortschritten, autonome Fahrzeuge zu entwickeln und zusätzlich die Sicherheit im Straßenverkehr zu erhöhen.

2.2. Einführung in die Fahrzeugfahrdynamik

Im Folgenden werden grundlegende Theorien erläutert, die dazu dienen, die komplexe Fahrdynamik eines Fahrzeugs zu vereinfachen und sie in Algorithmen zu implementieren, die es ermöglichen, effizient realistische Systeme nachzubilden.

2.2.1. Grundbewegungsarten eines PKW

Ein Fahrzeug kann sich vereinfacht entlang drei Achsen bewegen und drehen. Die Bewegungs- und Drehrichtungen werden in Abbildung 2.2 veranschaulicht.

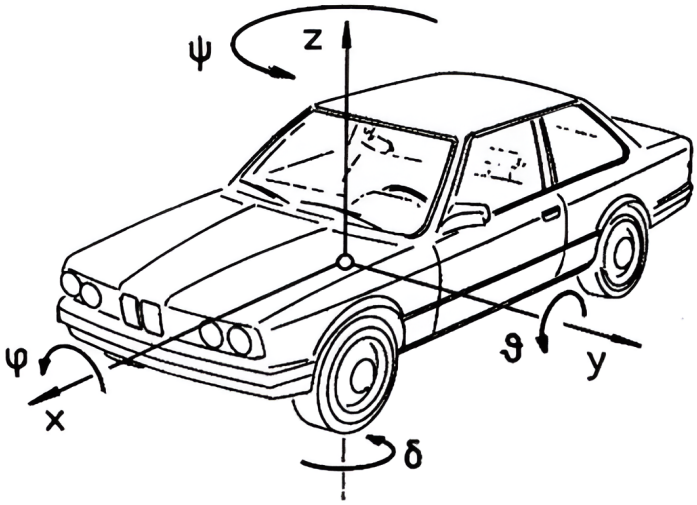


Abbildung 2.2.: Bewegungsrichtungen eines PKW [9]

Bewegungen sind vorwärts oder rückwärts in x-Richtung, links oder rechts in y-Richtung sowie auf oder ab in z-Richtung möglich. Die Bewegung in z-Richtung wird auch als Huben bezeichnet. Die Drehung des Fahrzeugs während einer Kurvenfahrt, wenn es sich nach links oder rechts neigt, wird als Wanken (φ) bezeichnet. Nicken (ϑ) beschreibt die Drehung um die y-Achse, die beim Beschleunigen oder Bremsen auftritt. Ändert das Fahrzeug beim Lenken die Richtung, tritt Gieren (ψ) auf. Jede Bewegung oder Drehung entlang einer Achse entspricht einem Freiheitsgrad. Ein PKW besitzt dementsprechend sechs Freiheitsgrade (engl.: Degrees Of Freedom (DOF)) [9].

2.2.2. Einspurmodell

Die Bewegungen eines Fahrzeugs werden oft durch Modelle beschrieben, die das Verhalten des realen Fahrzeugs ausreichend genau wiedergeben. Eines dieser Modelle ist das sogenannte Einspurmodell. Es beschränkt sich auf drei DOF und vereinfacht die Bewegung des Fahrzeugs erheblich, wie in Abbildung 2.3 gezeigt.

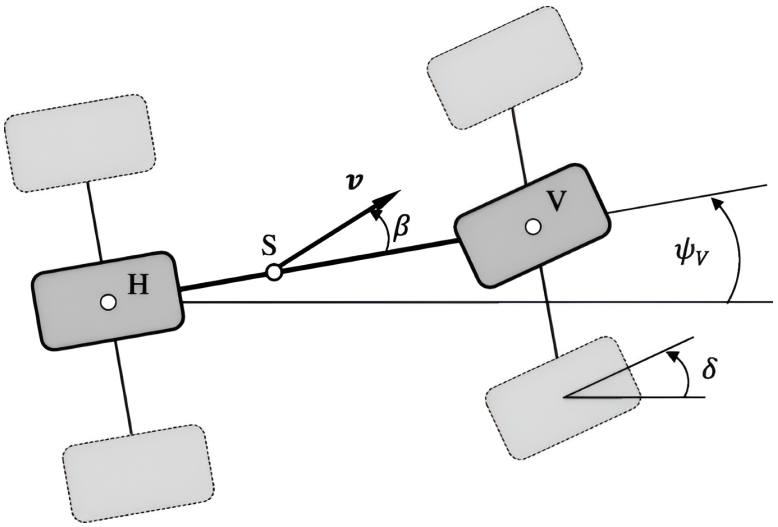


Abbildung 2.3.: Einspurmodell eines PKW [10]

Für dieses Modell werden folgende Annahmen getroffen:

- Die Räder an der Vorder- und Hinterachse werden jeweils zu einem Rad in der Achsmitte zusammengefasst. Dort entstehen die gedachten Radaufstandspunkte V und H.
- Die Radlenkwinkel δ der Vorderräder werden gemittelt und zu einem zusammengefasst.
- Im Massenmittelpunkt S wird die Fahrzeugmasse zusammengefasst.
- Sämtliche Wank-, Hub- und Nickbewegungen werden vernachlässigt.
- Entlang seiner Bahnkurve wird die Geschwindigkeit des Fahrzeugschwerpunkts als konstant betrachtet.
- Die Rückstellmomente und die Reifennachläufe werden vernachlässigt.
- Zwischen der Vorder- und Hinterachse bleibt die Radlastverteilung konstant.
- Die Reifenumfangskräfte werden vernachlässigt [10].

ψ_v beschreibt den Gierwinkel, der Auskunft über die Gierrate $\dot{\psi}$ gibt. Der Schwimmwinkel, also die Abweichung der Schwerpunktgeschwindigkeit v von der Fahrzeuglängsachse, wird als β bezeichnet [10].

2.3. Koppelnavigation

Eine Methode zur Bestimmung neuer Positionen in der Robotik ist die Koppelnavigation. Diese Methode ermöglicht die Positionsbestimmung durch die Nutzung der zuletzt bekannten Position in Kombination mit den aktuellen Eigenbewegungsdaten. Sie basiert auf der Anwendung der Kreisformel in Parameterform unter Berücksichtigung der Anfangsbedingungen. Zur Veranschaulichung dieser Methode dient die Abbildung 2.4.

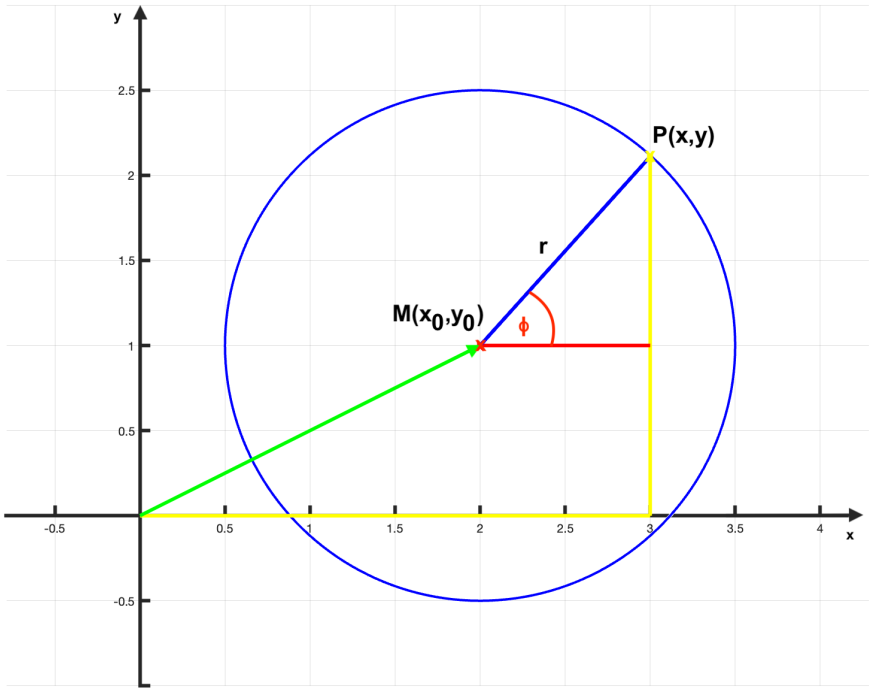


Abbildung 2.4.: Grafische Darstellung der Kreisformel [11]

Die Kreisformel in Parameterform, unter Berücksichtigung der Anfangsbedingungen, beschreibt einen Kreis mit dem Radius r, dessen Mittelpunkt M bei den Koordinaten (x_0, y_0) liegt. Diese Koordinaten repräsentieren die Anfangsbedingungen. Befindet sich der Mittelpunkt des Kreises im Koordinatenursprung, gelten $x_0 = 0$ und $y_0 = 0$, wodurch die Anfangsbedingungen entfallen. Im Allgemeinen ermöglichen diese Koordinaten jedoch die Beschreibung der aktuellen Lage des Kreises im Koordinatensystem. Der Punkt P mit den Koordinaten (x, y) liegt auf dem Rand des Kreises. Der Winkel ϕ beschreibt die Position des Punktes P relativ zur positiven x-Achse in einem imaginären, auf den Kreismittelpunkt M verschobenen Koordinatensystem. Unter Verwendung des Radius r und des Winkels ϕ können mithilfe der Winkelfunktionen im rechtwinkligen Dreieck die Gleichungen (2.1) und (2.2) zur Bestimmung der neuen Position (P) aufgestellt werden [11],[12].

$$x = x_0 + r \cdot cos(\phi) \tag{2.1}$$

$$y = y_0 + r \cdot sin(\phi) \tag{2.2}$$

2.4. Einführung in Regelungen

Regelungen sind ein zentraler Bestandteil selbstfahrender Fahrzeuge und deren Funktionsweise. Um deren Prinzipien zu verstehen, ist es wichtig, zunächst den Unterschied zur Steuerung eines Systems zu klären.

Bei einer Steuerung wird eine festgelegte Abfolge von Anweisungen ausgeführt, ohne dass das Ergebnis kontinuierlich überwacht oder angepasst wird. Dies kann dazu führen, dass unbemerkte Fehler oder Abweichungen auftreten. Im Gegensatz dazu kontrollieren Regelungen aktiv das Ergebnis eines Systems. Sie vergleichen kontinuierlich den Ist-Wert mit dem Soll-Wert und passen die Steuerung entsprechend an, um Abweichungen oder Störungen zu minimieren. Das Ziel ist, die Abweichungen so gering wie möglich zu halten und eine präzise und stabile Systemleistung zu gewährleisten. Abbildung 2.5 zeigt den grundlegenden Aufbau einer Regelstrecke.

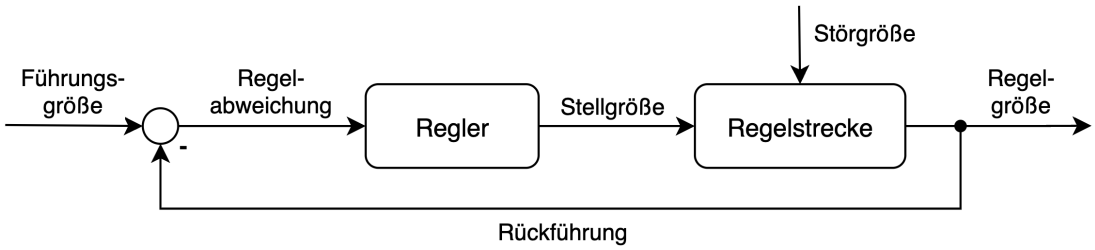


Abbildung 2.5.: Grundlegender Aufbau einer Regelstrecke [13]

In einem Regelungssystem bezeichnet die Führungsgröße den Soll-Wert, den das System erreichen soll. Dieser Soll-Wert wird kontinuierlich mit der zurückgeführten Regelgröße, dem Ist-Wert, verglichen. Auf diese Weise wird sichergestellt, dass die Regelung über die genaue Abweichung vom gewünschten Zustand informiert ist. Die Regelabweichung beschreibt, wie stark das System vom gewünschten Zustand abweicht.

Der Regler erfasst diese Abweichung und erzeugt ein Steuerungssignal, die sogenannte Stellgröße, um die Abweichung zu kompensieren. Wenn der Soll-Wert größer ist als der Ist-Wert, muss das System zusätzliche Ressourcen aufbringen, um die Differenz auszugleichen. Umgekehrt muss das System Ressourcen reduzieren, wenn der Ist-Wert den Soll-Wert übersteigt. Das erzeugte Signal wird dann in das zu regelnde System eingespeist, um die Abweichung zu korrigieren.

Da das System möglicherweise nicht wie beabsichtigt reagiert oder von externen Störungen beeinflusst wird, können zusätzliche Abweichungen auftreten. Daher ist es entscheidend, die Regelgröße kontinuierlich zurückzuführen, um sicherzustellen, dass das reale Ergebnis dem gewünschten Ergebnis so nahe wie möglich

kommt. Andernfalls kann es zu einer zunehmenden Abweichung zwischen Soll-Wert und Ist-Wert kommen, was im Extremfall zum Systemausfall führen kann [14].

2.4.1. PID Regler

Der PID-Regler (Proportional-Integral-Differential-Regler) gehört zu den grundlegendsten Regelungsstrategien. Er setzt sich aus drei Komponenten zusammen: dem proportionalen (P), dem integralen (I) und dem differentiellen (D) Anteil. Der proportionale Anteil wirkt als Multiplikator und reagiert direkt auf die Regelabweichung. Der integrale Anteil integriert die Regelabweichung und versucht so, langfristige Abweichungen zu minimieren. Der differentielle Anteil hilft den zukünftigen Verlauf zu schätzen und zu glätten, indem er die Änderungsrate der Abweichung bestimmt. Die Regelungsgleichung 2.3 beschreibt die Kombination und das Zusammenspiel dieser drei Anteile zur Steuerung des Regelprozesses.

$$u(t) = K_p \cdot e(t) + K_i \cdot \int_0^t e(\tau)\, d\tau + K_d \cdot \frac{d}{dt} e(t) \tag{2.3}$$

Der PID-Regler ermöglicht die Bildung verschiedener Regelkombinationen, indem unterschiedliche Anteile des Reglers variiert oder kombiniert werden. Diese Kombinationen beeinflussen das Verhalten des Reglers auf unterschiedliche Weise, was in Abbildung 2.6 veranschaulicht wird. Die Abbildung zeigt die grundlegenden Auswirkungen der verschiedenen Reglerkonfigurationen auf das Regelverhalten, einschließlich der Stabilität, Reaktionsgeschwindigkeit und Genauigkeit des Regelkreises.

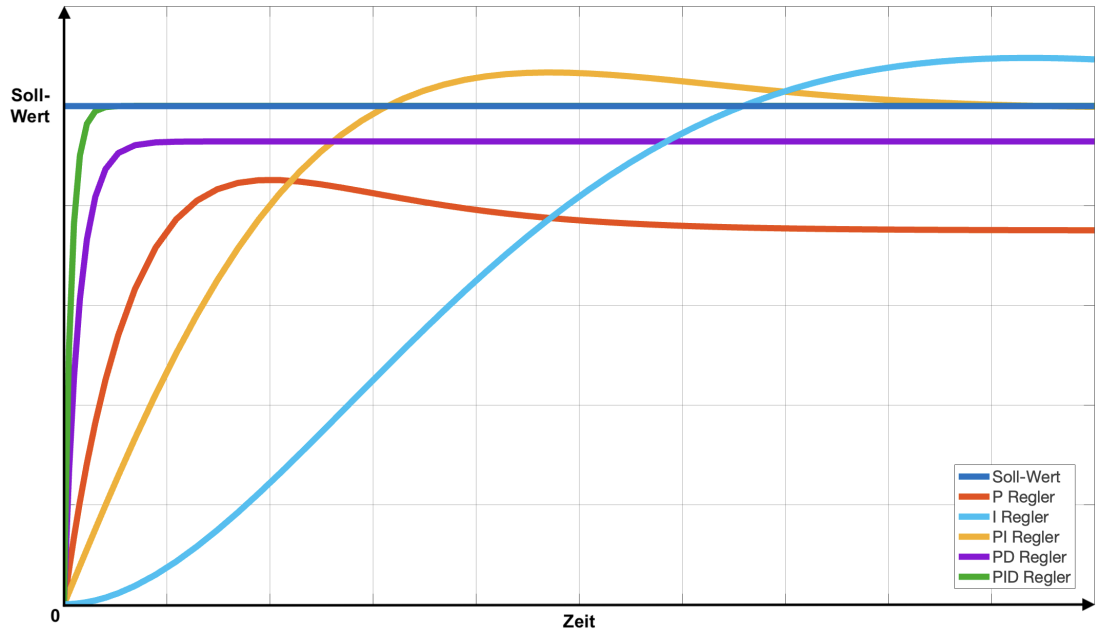


Abbildung 2.6.: PID-Regler Kombinationen [15]

Durch die Kombination der Anteile und die Anpassung der Reglerparameter kann die Leistung des PID-Reglers optimiert und an spezifische Anforderungen angepasst werden. Beispielsweise kann der Regler auf eine schnelle Reaktion eingestellt werden, um eine hohe Dynamik zu erreichen, oder auf hohe Genauigkeit, um den Soll-Wert präzise zu treffen. Es ist jedoch zu beachten, dass zu hohe Reglerparameter zu einem Überschwingen führen können, während zu niedrige Werte eine unzureichende Reaktionsgeschwindigkeit oder das Nicht-Erreichen des Soll-Werts zur Folge haben können. Diese Eigenschaften verdeutlichen die Einschränkung, dass in der Regel nicht alle Anforderungen gleichzeitig erfüllt werden können, wie in Abbildung 2.7 veranschaulicht wird.

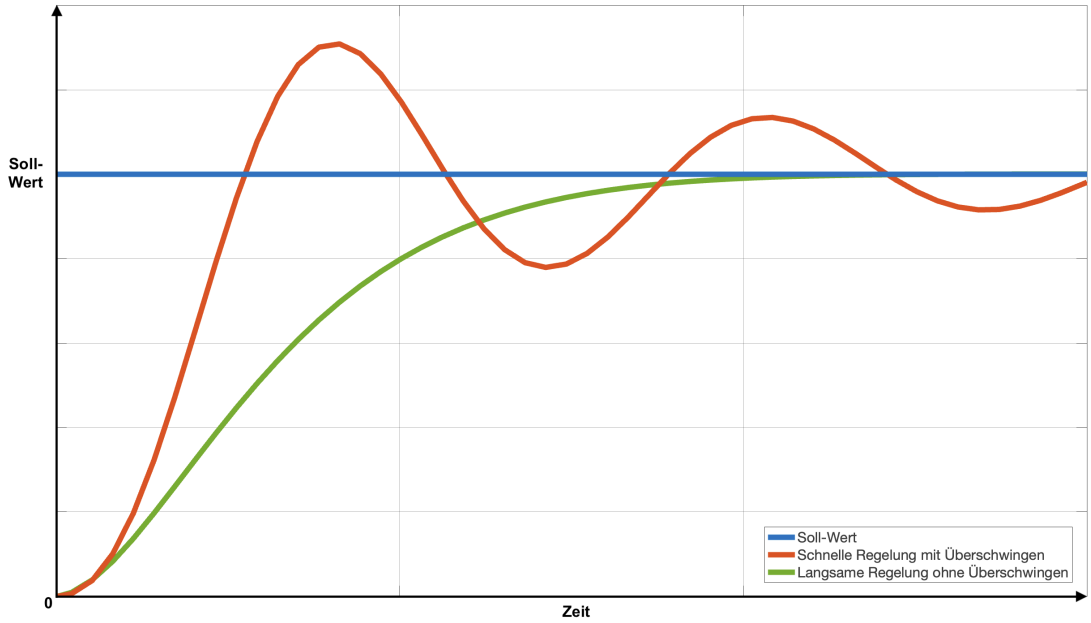


Abbildung 2.7.: Einschwingverhalten einer Reglung [16]

Zur Verbesserung der Ergebnisse des PID-Reglers können Erweiterungen wie beispielsweise der Smith-Prädiktor eingesetzt werden. Dieser zielt darauf ab, die Systemleistung zu optimieren, indem er das zukünftige Verhalten eines Systems schätzt. Dies erfolgt durch Modellierung und Vorhersage der Systemdynamik, wodurch Verzögerungen und andere Effekte kompensiert werden, die die Regelung beeinträchtigen könnten [16],[17].

2.4.2. Pure Pursuit Regler

Im Gegensatz zum PID-Regler, der auf der Korrektur von Fehlern basiert und sehr vielseitig einsetzbar ist, ist der Pure Pursuit-Regler (engl. für reine Verfolgung) ein auf Geometrie und Pfadverfolgung optimierter Regler. Er wird oft in der Robotik eingesetzt und nutzt das Einspurmodell. Dabei folgt er dem einfachen Prinzip des Vorausschauens. In Abbildung 2.8 ist ein Fahrzeug zu sehen, das versucht, dem orangefarbenen Punkt auf der Wunschtrajektorie zu folgen. Der orange Punkt ist

der als Nächstes anzusteuernde Punkt auf dem vorgegebenen Pfad. Die Entfernung zwischen diesem Punkt und dem Hinterrad des Fahrzeugs wird Vorausschaudistanz (engl. lookahead distance) genannt.

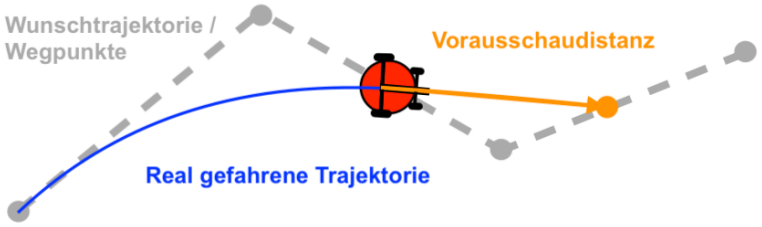


Abbildung 2.8.: Schematische Visualisierung des Pure Pursuit Reglers [18]

Damit das Fahrzeug den Punkt ordnungsgemäß anfahren kann, berechnet der Regler den benötigten Radlenkwinkel δ. Zur Veranschaulichung der nachfolgenden Berechnung wird Abbildung 2.9 herangezogen.

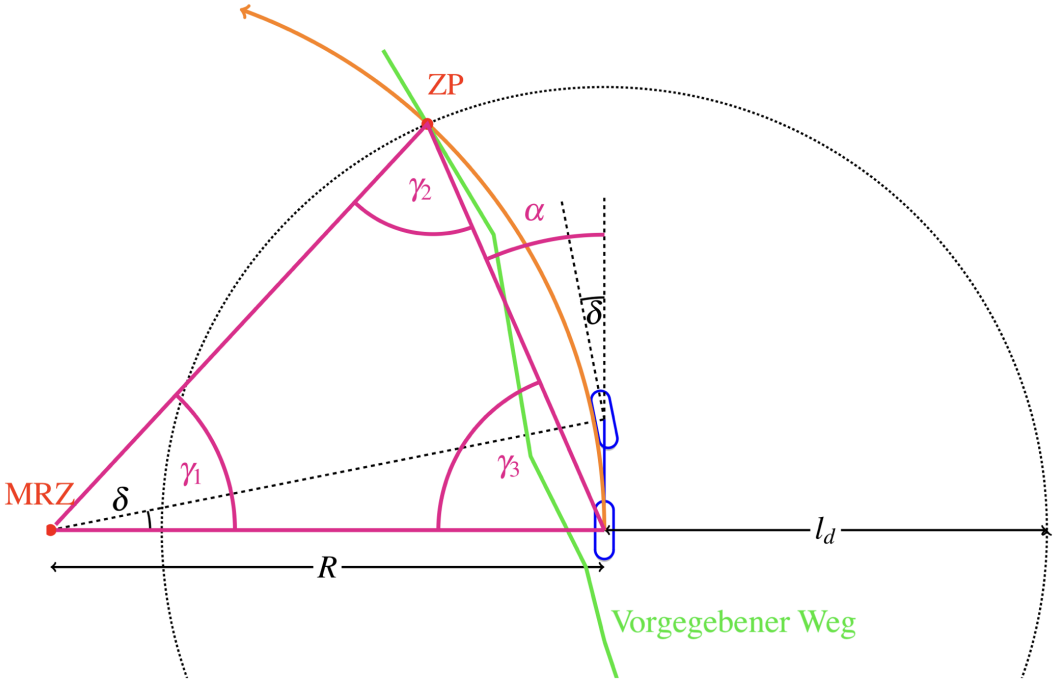


Abbildung 2.9.: Übersicht der Pure Pursuit Regler Gleichungsvariablen [19]

Die Abbildung zeigt den vorgegebenen Pfad (grün) und das Einspurmodell des Fahrzeugs (blau). Der gepunktete Kreis besitzt einen Radius, welcher der Vorausschaudistanz l_d entspricht. Dieser Kreis schneidet den vorgegebenen Pfad an einem Punkt. An dieser Stelle befindet sich der Zielpunkt ZP, den das Fahrzeug anzusteuern versucht. Um diesen Punkt zu erreichen, muss es den Winkel δ an den Rädern einstellen. Dieser Winkel führt dazu, dass das Fahrzeug dem orangen Kreisbogen folgen wird. Um den Winkel δ zu berechnen, wird mit der Mitte des hinteren Rades des Einspurmodells, dem Zielpunkt ZP und dem momentanen Rotationszentrum MRZ das magentafarbene Dreieck aufgespannt. Das momentane Rotationszentrum stellt den Punkt dar, um den das Fahrzeug mit dem aktuell

eingestellten Radlenkwinkel δ fahren würde. Aus dieser Kreisfahrt ergibt sich der zweiter Radius R.

Da die Innenwinkel des gleichschenkligen Dreieckes zusammen 180° ergeben müssen, kann folgende Annahme getroffen werden:

$$180° = \gamma_1 + \gamma_2 + \gamma_3 = \gamma_1 + (90° - \alpha) + (90° - \alpha) \tag{2.4}$$

Mithilfe des Sinussatzes lässt sich die Formel (2.4) aufstellen.

$$\frac{l_d}{\sin(\gamma_1)} = \frac{R}{\sin(\gamma_2)} \tag{2.5}$$

Durch das Einsetzten und Umstellen der Gleichung (2.5) erhält man:

$$\frac{l_d}{\sin(2\alpha)} = \frac{R}{\sin(90° - \alpha)} \tag{2.6}$$

Setzt man nun andere Identitäten für die Nenner ein, erhält man:

$$\frac{l_d}{2\sin(\alpha)\cos(\alpha)} = \frac{R}{\cos(\alpha)} \tag{2.7}$$

Da das Fahrzeug mit seinem Radstand L und dem MRZ ebenso ein kleines Dreieck aufspannt, gilt Folgendes:

$$\tan(\delta) = \frac{L}{R} \tag{2.8}$$

Durch die Verwendung und das Umstellen der Gleichungen 2.7 und 2.8 ergibt sich die finale Formel für δ:

$$\delta = \arctan\left(\frac{2L\sin(\alpha)}{l_d}\right) \tag{2.9}$$

Aufgrund der Definition des Reglers berücksichtigt dieser stets eine festgelegte Vorausschaudistanz. Daher werden Punkte, die innerhalb des Kreises mit dem Radius l_d liegen, nicht beachtet. Ist diese Distanz zu groß, hat das zur Folge, dass insbesondere bei scharfen Kurven der Ausgangspunkt der Kurve bereits angesteuert wird. Infolgedessen kann es vorkommen, dass Kurven stark geschnitten oder abgekürzt werden, was ebenfalls in Abbildung 2.8 veranschaulicht ist. Im Gegensatz dazu kann eine zu geringe Vorausschaudistanz dazu führen, dass der Regler instabil wird und zwischen den Punkten stark oszilliert [18],[19],[20].

3. Technische Grundlagen

Autonome Fahrzeuge müssen in der Lage sein, komplexe Situationen sicher wahrzunehmen, zu verstehen und eigenständig darauf zu reagieren. In diesem Kapitel werden die wesentlichen Technologien zur Umfelderfassung, zur Aufnahme von Fahrzeugdaten und zur Fahrzeugkommunikation behandelt. Sie sind entscheidend, um den Fahrfunktionen grundlegende Daten bereitzustellen und die erforderlichen Anweisungen an die Aktoren zu übermitteln. Dabei werden die grundlegenden Funktionsprinzipien und Strukturen dieser Komponenten erläutert sowie deren Anwendungsbereiche beschrieben.

Die Abbildung 3.1 veranschaulicht die grundlegende Struktur einer autonomen Fahrfunktion und vermittelt vorab ein grundlegendes Verständnis für den Zusammenhang der Systembausteine.

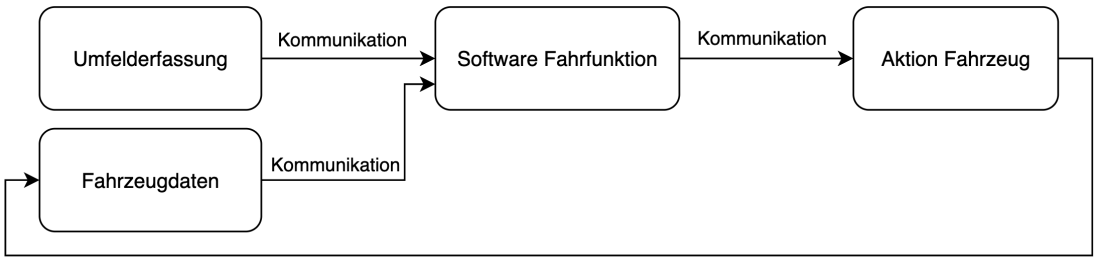


Abbildung 3.1.: Grundlegende Funktionsweise autonomer Fahrfunktionen

Eine autonome Fahrfunktion benötigt Daten, die Informationen über den relevanten Zustand des Fahrzeugs sowie dessen Umwelt liefern. Sobald alle erforderlichen Daten vorliegen und kommuniziert wurden, kann die Fahrfunktion diese verarbeiten und die entsprechenden Handlungen an das Fahrzeug übermitteln. Das Fahrzeug führt dann die vorgegebenen Handlungen aus, um den angestrebten Zustand zu erreichen. Dieser Vorgang wird kontinuierlich wiederholt.

3.1. Sensoren für die Umfelderfassung

Ein wesentlicher Bestandteil autonomer Fahrzeuge ist die Umfelderfassung. Auf öffentlichen Straßen ändern sich die Umgebungsbedingungen sehr schnell und es können zahlreiche kritische Situationen auftreten. Daher ist es von großer Bedeutung, eine Kombination von Sensoren zu verwenden, die in der Lage sind, eine Vielzahl möglicher Situationen selbst unter den schwierigsten Bedingungen optimal zu erfassen. Abbildung 3.2 zeigt typische Umfeldsensoren eines PKW sowie deren Messverfahren, die im Folgenden näher erläutert werden.

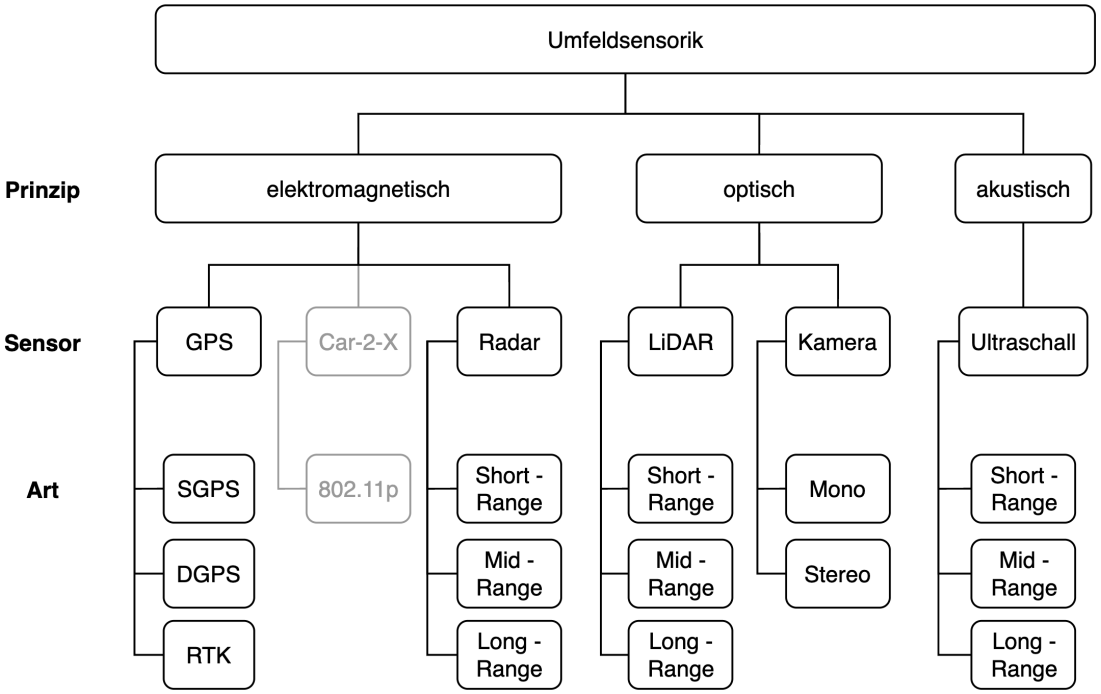


Abbildung 3.2.: Übersicht der Umfeldsensorik im PKW [21]

3.1.1. GPS

Um sich sicher in einer Umgebung bewegen zu können, benötigt ein Fahrzeug seinen exakten Standort. Dies verhindert Kollisionen und ermöglicht eine präzise Navigation. Die Ortung erfolgt in der Regel unter Nutzung des Global Positioning System (GPS). Dieses System basiert auf der Trilateration und ermöglicht die Bestimmung der Fahrzeugposition, indem es die Abstände zu mindestens drei bekannten Punkten berechnet und vergleicht. Der GPS-Empfänger empfängt dazu Radiosignale von mindestens drei Satelliten und erfasst die Zeit des Eintreffens dieser Signale. Die von den Satelliten gesendeten Signale enthalten sowohl die Position des Senders als auch die Zeit des Aussendens. Aus den Zeitdifferenzen und unter Berücksichtigung der Lichtgeschwindigkeit werden die Entfernungen zu den Satelliten berechnet, wodurch die Position des Fahrzeugs ermittelt wird. Da die Ortung durch das Standard-GPS unter Umständen abbrechen kann und häufig mit Abweichungen im Bereich von wenigen Metern zu ungenau für autonome Fahrzeuge ist, sind genauere Systeme wie Differential-GPS (D-GPS) oder Real-Time Kinematic-GPS (RTK-GPS) erforderlich. Solche präziseren Systeme sind jedoch oft mit mehr Aufwand und höheren Kosten verbunden [22].

3.1.2. Car-2-X

Eine der neuesten Technologien im Bereich der Umfelderfassung ist Car-2-X. Sie ermöglicht die Kommunikation des Fahrzeugs mit anderen Verkehrsobjekten wie beispielsweise anderen Fahrzeugen, Baustellen, Personen oder Infrastrukturele-

menten. Jedes Verkehrsobjekt stellt Informationen bereit, die über WLANp, einen drahtlosen Kommunikationsweg ohne Mobilfunknetz, von nahegelegenen Verkehrsteilnehmern empfangen und verarbeitet werden können. WLAN 802.11p, das für diese Funktion verwendet wird, hat typischerweise eine Reichweite von bis zu 800m. Weil Car-2-X auf WLAN basiert, kann diese Technologie nur bedingt der Umfeldsensorik zugeordnet werden, da es zwar Informationen über das Umfeld liefert, jedoch eher eine kommunikationsbasierte Funktion als einen klassischen Sensor darstellt [23].

3.1.3. Radar, LiDAR und Ultraschall

Nicht nur die Positionierung im Raum ist entscheidend, sondern auch das Erkennen und Verstehen der Umgebung. In autonomen Fahrzeugen werden daher vor allem Radio Detection and Ranging (Radar)-, Light Detection and Ranging (LiDAR)- und Ultraschall-Sensoren eingesetzt. Obwohl diese drei Sensortypen in Abbildung 3.2 unterschiedlichen Messverfahren zugeordnet sind, ist ihre Funktionsweise grundsätzlich ähnlich. Alle Sensoren arbeiten nach dem in Abbildung 3.3 dargestellten Time-of-Flight-Prinzip.

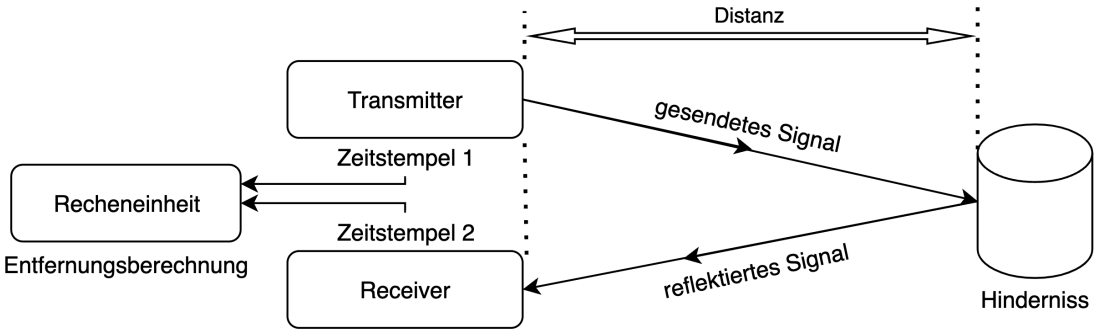


Abbildung 3.3.: Skizze Time-of-Flight Prinzip [24]

Dabei sendet ein Transmitter Wellen aus und erzeugt einen Sende-Zeitstempel. Trifft die Welle auf ein Hindernis, wird sie je nach Oberfläche und Material unterschiedlich reflektiert, was die Genauigkeit und Intensität des Signals beeinflusst. Die reflektierten Wellen werden anschließend von einem Receiver empfangen, der einen Empfangs-Zeitstempel erzeugt. Abschließend wird in der Recheneinheit aus der Zeitdifferenz Δt zwischen dem Sende- und Empfangs-Zeitstempel die Entfernung d des gemessenen Punktes ermittelt, wie in Formel 3.1 dargestellt. Dabei ist die Ausbreitungsgeschwindigkeit c des Übertragungsmediums erforderlich. Es muss berücksichtigt werden, dass Δt die gesamte Zeit für den Hin- und Rückweg umfasst. Für die Berechnung der Entfernung d wird jedoch nur die Hälfte des insgesamt zurückgelegten Weges benötigt.

$$d = \frac{c \cdot \Delta t}{2} \tag{3.1}$$

Da die Sensoren für unterschiedliche Messentfernungen optimiert werden können, existieren für jeden dieser Sensortypen sogenannte Short-, Mid- und Long-Range-Varianten [24],[25].

Um die Unterschiede deutlich zu machen, werden die im Automobilbereich verwendeten Sensoren in der nachfolgenden Tabelle 3.1 gegenübergestellt.

	Radar	**Ultraschall**	**LiDAR**
Wellenart	Radiowellen	Schallwellen	Lichtwellen
Ausbreitungs-geschwindigkeit	299.792.458m/s	343,5m/s	299.792.458m/s
Frequenzbereich	24-81GHz	50-60kHz	193-331THz
Wellenlängenbereich	12,5-3,7mm	6,9-5,7mm	1550-905nm
Ausbreitungsform	keulenartig	keulenartig	strahlenartig
Vorteile	kann gezielt Materialien durchdringen	Oberflächeneigenschaften des Objekts haben keinen Einfluss	sehr präzise Abbildung der Umgebung
Nachteile	nur grobe Objekterkennung möglich	geringe Reichweite	erkennt schwarze und transparente Oberflächen schlecht
Anwendung	Abstandswarnung	Einparkhilfe	3D-Kartierung

Tabelle 3.1.: Übersicht Radar, Lidar, Ultraschall [24],[26],[27],[28],[29],[30]

Der Zusammenhang zwischen der Ausbreitungsgeschwindigkeit c, der Frequenz f und der Wellenlänge λ kann mit der Formel 3.2 beschrieben werden [31].

$$\lambda = \frac{c}{f} \tag{3.2}$$

Da der LiDAR-Sensor in dieser Arbeit von besonderer Bedeutung ist, wird im Folgenden detaillierter auf ihn eingegangen.

LiDAR-Sensoren finden nicht nur mobile Anwendung, wie etwa in autonomen Fahrzeugen zur Visualisierung der Umgebung, sondern auch auf Wasser zur Vermessung des Meeresbodens oder in Flugzeugen zur Kartierung der Erdoberfläche. Auch stationär werden LiDAR-Sensoren eingesetzt, zum Beispiel zur Vermessung von Gebäuden. Im Automobilbereich gibt es verschiedene Typen von LiDAR-Sensoren, die, wie in Abbildung 3.4 dargestellt, klassifiziert werden können [24].

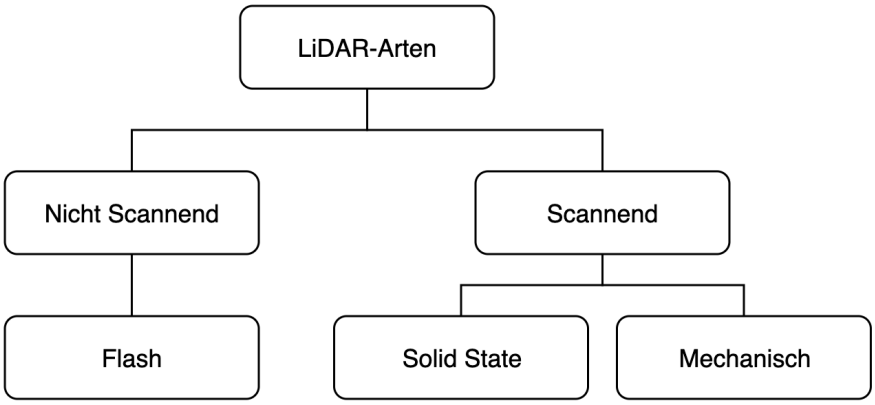
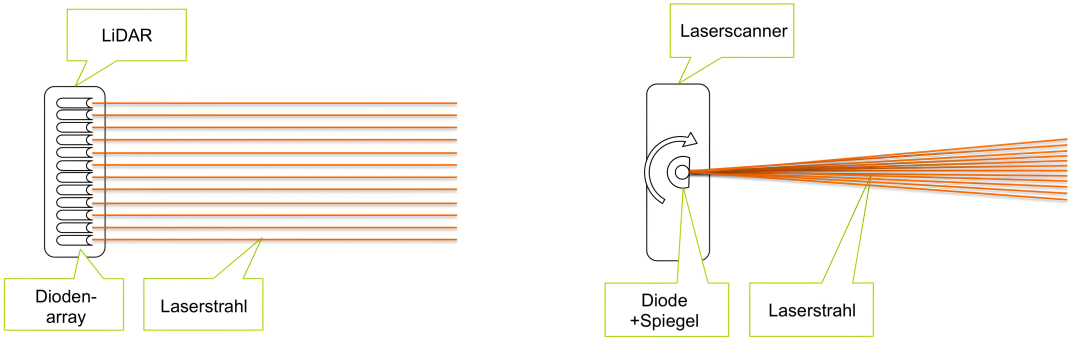


Abbildung 3.4.: LiDAR-Arten [24],[29]

Als scannende LiDAR-Sensoren werden solche bezeichnet, die über bewegliche Teile verfügen, welche systematisch bewegt werden. Im Gegensatz dazu sind nicht scannende LiDAR-Sensoren solche, die keine beweglichen Teile besitzen. Abbildung 3.5 zeigt jeweils ein Beispiel für einen nicht scannenden und einen scannenden LiDAR-Sensor.



(a) Flash-LiDAR (b) Laserscanner

Abbildung 3.5.: Schematische Visualisierung des Flash-LiDAR- und des Laserscanner-Prinzips [24]

Der Flash-LiDAR zeichnet sich dadurch aus, dass er über ein Dioden-Array verfügt und somit die gesamte Umgebung mit nur einer Auslösung erfassen kann. Dies macht ihn besonders robust gegenüber schnellen Bewegungen. Im Gegensatz dazu ist der Laserscanner, als mechanischer LiDAR, mit einem sich drehenden Spiegel und einer oder mehreren Dioden ausgestattet. Er muss mehrfach ausgelöst werden, da er jeweils nur einen bestimmten Bereich der Umgebung abtastet. Dies dauert etwas länger, liefert jedoch präzisere Messungen. Der Solid-State-LiDAR ist ebenfalls scannend, verwendet aber meist mehrere Dioden oder andere elektronische Mittel, um die Lichtstrahlen zu lenken. Dies ermöglicht eine schnellere Funktionsweise im Vergleich zu mechanischen Systemen. Alle drei Scanner erzeugen aus den generierten Daten eine dreidimensionale Punktwolke, aus der umfangreiche Informationen über die Umwelt und deren Objekte gewonnen werden können.

Je nach Typ und Anwendungsbereich können sie Entfernungen von wenigen Metern bis zu mehreren Kilometern messen und erreichen dabei eine Genauigkeit im Bereich von wenigen Zentimetern bis hin zu Millimetern [24],[29],[32],[33].

3.1.4. Kamera

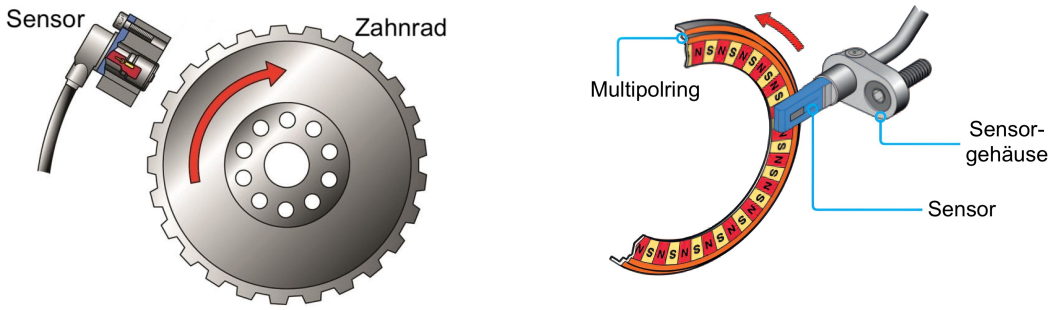
Kameras sind ein wesentlicher Bestandteil zur Erfassung der Umwelt. Sie ermöglichen beispielsweise die zweidimensionale und farbige Darstellung der Umgebung. Sie können sowohl Bilder als auch Videos liefern. Abhängig von der Kamerahardware und deren Einsatz kann sie für verschiedene Zwecke optimiert werden. Beispiele sind die Optimierung für Nachtaufnahmen oder zur Erfassung der räumlichen Tiefe. Zur Erzeugung der Tiefenwahrnehmung wird ein Stereo-Aufbau verwendet. Im Gegensatz zum Mono-Aufbau, bei dem nur ein Bildsensor verwendet wird, werden beim Stereo-Aufbau zwei Bildsensoren in einem festen Abstand installiert, um die räumliche Tiefe zu berechnen. Die Bildsensoren bestehen aus Millionen von Fotodioden, die die Lichtintensitäten erfassen, die durch Reflexion der Lichtstrahlen von der Umgebung auf sie fallen. Um nicht nur Schwarz-Weiß-Bilder zu erhalten, werden häufig Farbfilter vor den Fotodioden installiert. Diese Filter lassen nur Licht einer bestimmten Wellenlänge hindurch. Dadurch können Intensitäten spezifischer Wellenlängen, also Farben, erfasst werden. Die so gewonnenen Daten werden anschließend vom Bildprozessor zu einem vollständigen Bild zusammengesetzt [34].

3.2. Sensoren zur Erfassung der Eigenbewegungsdaten

Da autonome Fahrzeuge nicht nur ihre Umwelt korrekt wahrnehmen und verstehen, sondern auch ihren eigenen Zustand überwachen müssen, sind zusätzliche Sensoren erforderlich. Eine Vielzahl der Fahrzeugsysteme greift auf diese Sensoren zu, um beispielsweise die Geschwindigkeit oder den Lenkeinschlag zu erfassen. Im Folgenden werden die für diese Arbeit wichtigsten Sensoren näher betrachtet.

3.2.1. Raddrehzahlsensoren

Um die Fahrzeuggeschwindigkeit zu bestimmen oder die Anti-Schlupf-Regelung präzise zu steuern, werden im PKW sogenannte Raddrehzahlsensoren verbaut. Diese Sensoren messen die Drehzahlen der Räder und lassen sich nach ihrer Funktionsweise in zwei Gruppen einteilen: passive und aktive Sensoren. Die Funktionsprinzipien dieser Sensoren sind in Abbildung 3.6 dargestellt.



(a) Passiver Raddrehzahlsensor (b) Aktiver Raddrehzahlsensor

Abbildung 3.6.: Schematische Abbildungen des passiven und aktiven Raddrehzahlsensors [35]

Passive Sensoren arbeiten nach dem Prinzip der elektromagnetischen Induktion. Sie bestehen aus einer Spule und einem Dauermagneten und sind über einem sich drehenden Zahnrad platziert. Die abwechselnde Anordnung von Zahnlücken und Zähnen verändert das Magnetfeld. Diese Veränderung induziert eine Wechselspannung in der Spule, die als Ausgangssignal dient. Passive Sensoren benötigen keine externe Stromversorgung, da das erzeugte Signal direkt durch die induzierte Spannungsänderung entsteht.

Aktive Sensoren hingegen benötigen eine Spannungsversorgung und enthalten Elektronik, die mit der angelegten Spannung betrieben wird. Im Gegensatz zu passiven Sensoren, welche mit einem Zahnrad arbeiten, nutzen aktive Sensoren ein Impulsrad. Dieses kann entweder ein Multipolrad mit wechselnden Magnetfeldern oder ein Stahlrad sein. Bei der Verwendung eines Stahlrads ist zusätzlich ein Dauermagnet erforderlich, der ein konstantes Magnetfeld erzeugt. Dreht sich das Rad, erkennt der Sensor bei beiden Varianten eine Veränderung des Magnetfelds, die er dann in ein digitales Signal umwandelt. Die Genauigkeit des Sensors liegt dabei zwischen 0,5 und 1,5% des Messbereiches. Aufgrund ihrer geringeren Störanfälligkeit und der Fähigkeit, selbst kleine Bewegungen präzise zu registrieren, sind aktive Sensoren die bevorzugte Wahl in der Automobilindustrie [35],[36].

3.2.2. Beschleunigungssensor

Beschleunigungssensoren werden häufig mit sogenannten Micro-Electro-Mechanical System (MEMS) realisiert. Diese Bauteile, die im Mikro- bis Millimeterbereich realisiert werden, erzeugen elektrische Signale durch mechanische Bewegung. Abbildung 3.7 zeigt den stark vereinfachten Aufbau eines MEMS.

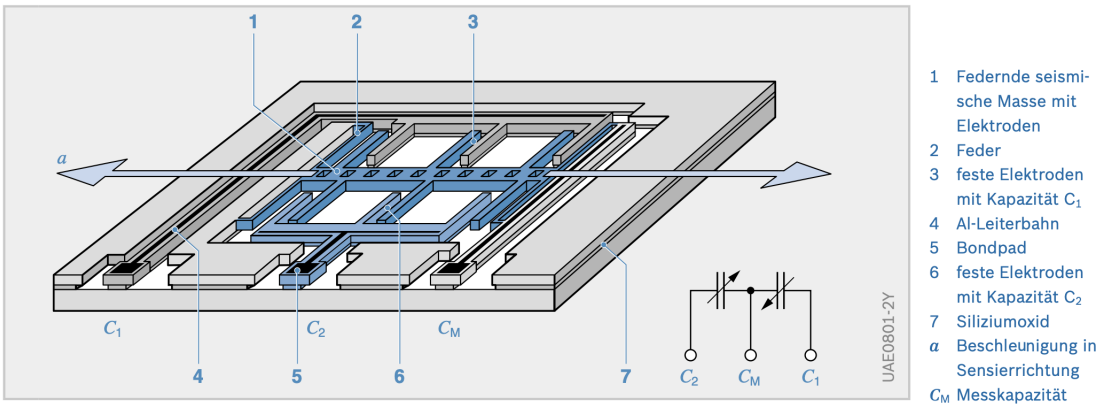


Abbildung 3.7.: Grundlegender Aufbau eines MEMS [37]

Die MEMS bestehen aus kammähnlichen Strukturen. Dabei sind die beiden äußeren Kämme fest, während sich zwischen ihnen eine Masse befindet, die über zwei Federn, jeweils links und rechts, in der Mitte gehalten wird. Wenn der Sensor beginnt, sich in Richtung der Federn zu bewegen, kann die Masse schwingen. Diese Schwingung führt zu einer Kapazitätsänderung, die als Signal ausgegeben wird. Auf diese Weise kann der Sensor Bewegungen entlang einer Achse erfassen. Es gibt auch MEMS, die mehrere Bewegungsachsen erfassen können, ihr Aufbau ist komplexer, doch das Funktionsprinzip ist gleich [37],[38],[39].

3.2.3. Drehratensensor

Der Drehratensensor misst die Winkelgeschwindigkeit um eine Achse und besteht in der einfachsten Konfiguration (MM1) aus zwei Beschleunigungssensoren, die über ein Doppelkreuz mechanisch gekoppelt sind. Dabei werden die beiden Sensoren gegensinnig ausgelenkt. Derzeitige Fahrzeugsensoren basieren auf der Kategorie MM3 und funktionieren mit einem ähnlichen Prinzip. Sie nutzen ebenfalls die Coriolis-Kraft, jedoch werden hier statt zweier Sensoren die zwei Enden einer im Mikrometerbereich großen Stimmgabel nach außen ausgelenkt. Durch die Kombination mehrerer solcher Sensoren ist es möglich, die Drehrate um drei Achsen zu messen und den in Abbildung 3.8 gezeigten Drehratensensor zu realisieren.

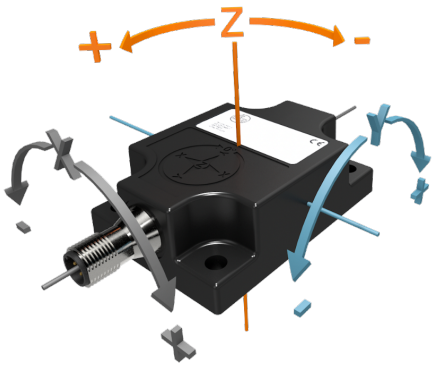


Abbildung 3.8.: Beispiel mehrachsiger Drehratensensor [40]

Damit der Sensor ordnungsgemäß funktioniert und aussagekräftige Werte liefert, muss er möglichst im Fahrzeugschwerpunkt verbaut werden. Andernfalls ist eine Korrektur der Sensordaten erforderlich. Andere mögliche Fehlerquellen, die zu beachten sind, umfassen die Empfindlichkeit gegenüber Drehungen außerhalb der Hauptmessachse, sowie das Rauschen und den Drift des Sensors [37],[40],[41],[42].

3.2.4. Lenkwinkelsensor

Für das Auslesen der Lenkradstellung ist der Lenkwinkelsensor zuständig. Es gibt optische und magnetische Sensoren, wobei meist die magnetischen im Automobil bevorzugt werden. Diese werden an der Lenksäule befestigt und messen den Winkel, um den sich die Säule dreht. Dies geschieht mithilfe eines Zahnrades, das zwei mit Magneten bestückte Messzahnräder antreibt. Durch die Drehung ändert sich die Magnetfeldrichtung, was zu einer Änderung des elektrischen Widerstands im Sensor führt. Das entstehende analoge Signal wird dann in ein digitales umgewandelt und im Fahrzeug bereitgestellt [43].

3.3. Kommunikationsnetze im Fahrzeug

Ein Großteil der Kommunikation im Fahrzeug funktioniert über sogenannte Binary Unit System (BUS). Diese dienen dazu, Daten zwischen den Teilnehmern eines Netzwerkes auszutauschen. Da vor allem in modernen Fahrzeugen sehr viele Daten anfallen, gibt es verschiedene BUS. Aufgrund der unterschiedlichen Anforderungen haben sich im PKW-Bereich besonders folgende fünf durchgesetzt:

- Local Interconnect Network (LIN)
- Controller Area Network (CAN)
- FlexRay
- Media Oriented Systems Transport (MOST)
- Ethernet

Ein Vernetzungsbeispiel der Fahrzeugkommunikation in einem Mittelklasse-Pkw ist in Abbildung 3.9 zu sehen. Die Abbildung 3.10 gibt einen Überblick über die BUS [44].

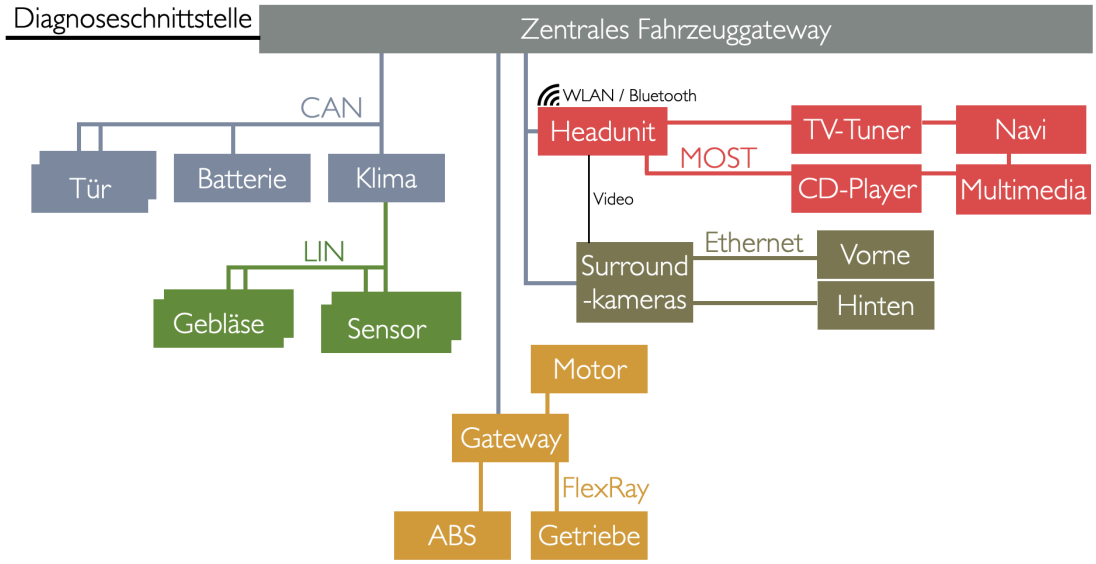


Abbildung 3.9.: Exemplarische Vernetzung der BUS im Fahrzeug [44],[45]

OSI-Schicht	CAN	LIN	FlexRay	MOST	Ethernet
Application Layer (7)	Die Bedeutung der Daten ist anwendungsabhängig – Sensordaten, Stellgrößen – Diagnosekommunikation – Audio-/Videodaten – Internetdienste				
Presentation Layer (6)					
Session Layer (5)					
Transport Layer (4)	Spezielles Transportprotokoll wird zur Übermittlung großer Datenblöcke benötigt. Z. B. ISO-TP für Diagnose-Kommunikation			Synchrones Streaming, MHP, Ethernet	TCP, UDP, AVP-TP
Network Layer (3)	Adressierung 11 Bit oder 29 Bit CAN-Identifier, 6 Bit LIN-Identifier		FlexRay Frames mit PDUs	MOST Network Services	IPv6, IPv4, AVB
Data Link Layer (2)	Busarbitrierung, Kollisionsvermeidung (CSMA/CA)	Zeitgesteuertes Übertragungssystem festgelegte Zeitslots für jedes Steuergerät		Synchrone Übertragung von Frames mit 44,1 oder 48 kHz im MOST-Ring	MAC-Adressen, Ethernet-Switches, Priorisierung, VLAN
Physical Layer (1)	Differentielle Spannungssignale über verdrillte Zweidrahtleitung	Spannungsignale gegenüber Masse auf einzelner Datenleitung	Differentielle Spannungssignale über verdrillte Zweidrahtleitung	Optische oder elektrische Übertragung, Ringtopologie	Differentielle Spannungssignale über verdrillte Zweidrahtleitung

Abbildung 3.10.: Übersicht BUS im Fahrzeug [44]

LIN wird für Karosserieelektronik wie Spiegelversteller oder Fensterheber genutzt. MOST hat sich besonders im Multimediabereich durchgesetzt, und FlexRay wird oft bei Hochleistungs-Antriebselektronik eingesetzt. Auf die anderen beiden BUS, CAN und Ethernet, wird im Folgenden aufgrund ihrer Bedeutung für diese Arbeit näher eingegangen [44].

3.3.1. CAN

Das CAN ist eines der bekanntesten Fahrzeugnetzwerke. Ein Großteil der Sensoren und Aktuatoren eines Fahrzeugs ist damit verbunden. Es zeichnet sich nicht nur durch hohe Robustheit aus, sondern auch durch einen vergleichsweise einfachen Aufbau und eine unkomplizierte Funktionsweise. In Abbildung 3.11 ist eine Skizze des Aufbaus zu sehen.

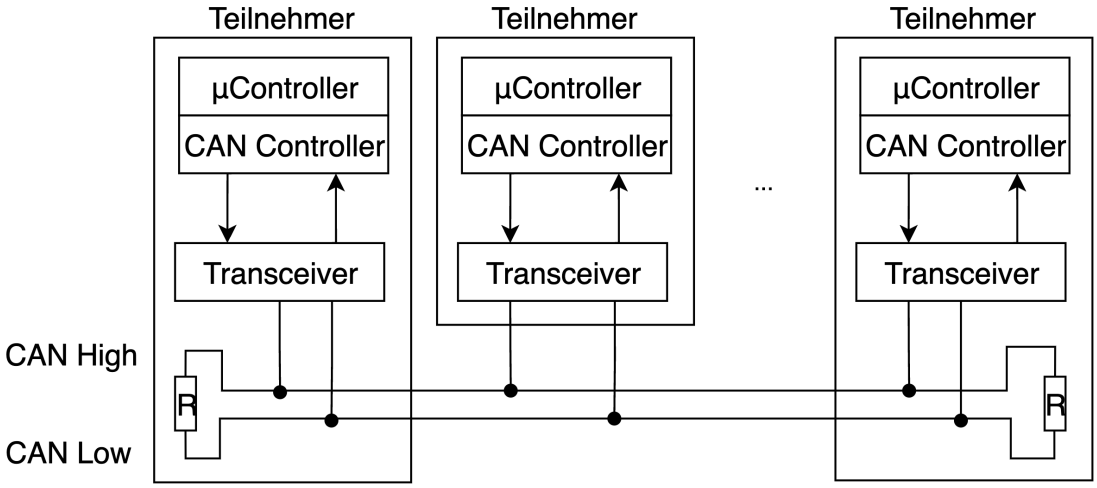


Abbildung 3.11.: Grundlegender Aufbau eines CAN-Netzwerkes [44]

Das CAN besteht aus mehreren Teilnehmern, die auf einen gemeinsamen Übertragungsweg zugreifen. Dieser muss an den Enden mit 120-Ohm-Abschlusswiderständen versehen werden, um Signalreflexionen zu minimieren. Zusätzlich besteht die CAN-Leitung aus zwei verdrillten Drähten, um elektromagnetische Störungen zu reduzieren. Alle Teilnehmer können sowohl Nachrichten senden als auch empfangen. Um die Nachrichten ordnungsgemäß zu übertragen, verwendet das CAN-Protokoll das CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance)-Verfahren. Dabei kann ein Teilnehmer erst Daten senden, wenn der Bus frei ist. Versuchen zwei Teilnehmer gleichzeitig zu senden, setzt sich der durch, dessen CAN-Nachricht die höchste Priorität hat. Bei ungünstigen Prioritätseinstellungen kann es vorkommen, dass keine Nachricht gesendet wird. Die Übertragungsgeschwindigkeiten und -längen variieren je nach Typ. Es gibt das Low-Speed-CAN mit Datenraten von bis zu 125 Kilobit pro Sekunde und Reichweiten bis zu 1 km, sowie das High-Speed-CAN mit Datenraten von bis zu 1 Megabit pro Sekunde und Reichweiten bis zu 40 m. Jeder Teilnehmer kann jedoch alle verfügbaren Nachrichten jederzeit mitlesen [44],[46].

3.3.2. Ethernet

Für größere Datenmengen ist Ethernet gut geeignet. Es bietet Übertragungsraten von bis zu 10 Gigabit pro Sekunde. Allerdings können diese hohen Datenraten nur über kurze Leitungslängen von bis zu 15 Metern erreicht werden. Die Leitungen sind häufig verdrillt, um elektromagnetische Störungen zu reduzieren, und Abschlusswiderstände sind in der Regel nicht erforderlich. Die gesendeten Daten werden direkt zugestellt. Um die Pakete zwischen mehreren Teilnehmern korrekt weiterzuleiten, kommen sogenannte Switches zum Einsatz. Diese verteilen die ankommenden Pakete und helfen zusätzlich, Kollisionen zu vermeiden. Das Verfahren zur Kollisionsvermeidung heißt CSMA/CD (Carrier Sense Multiple Access/Collision Detection). Bei diesem Verfahren sendet ein Teilnehmer seine Daten, während alle anderen warten. Falls zwei Teilnehmer gleichzeitig senden möchten, warten beide eine zufällige Zeit, bevor sie einen erneuten Versuch starten. Switches bieten zusätzlich die Möglichkeit, Daten priorisiert zuzustellen. Wenn jedoch zu viele Teilnehmer in einem Netzwerk vorhanden sind, steigt das Risiko von gleichzeitigen Sendeversuchen, was das Netzwerk überlasten kann. Dies kann zu Verzögerungen bei der Zustellung, Paketverlusten oder einer verringerten Datenrate führen [44].

3.4. Sensorfusion

Ein Fahrzeug stellt eine große Menge an Daten von unterschiedlichen Sensoren bereit. Abbildung 3.12 veranschaulicht ein Fahrzeug mit den verschiedenen Sichtbereichen der verbauten Sensoren.

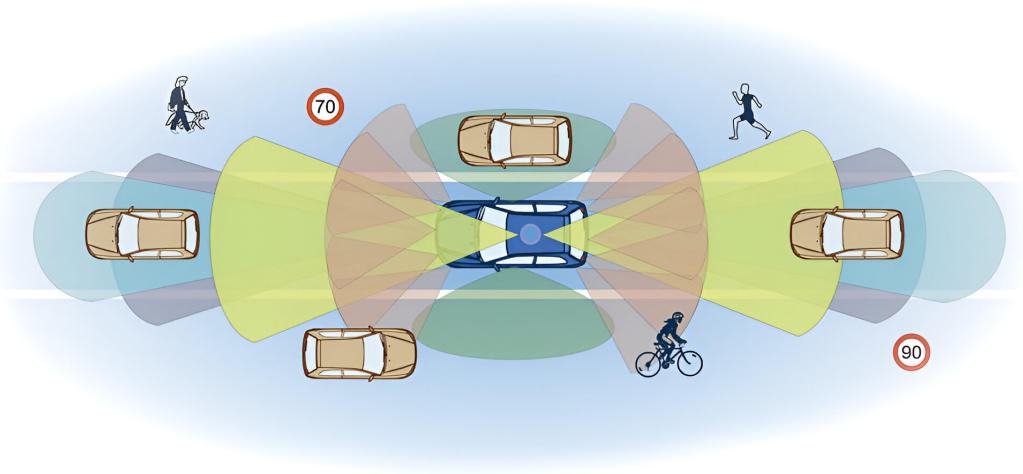


Abbildung 3.12.: Sichtbereiche verschiedener Umfeldsensoren im PKW [47]

Darin ist zu sehen, dass Fahrzeuge sowohl Sensoren für den Nahbereich als auch für den Fernbereich besitzen. Damit die autonomen Fahrfunktionen die Vorteile

der Sensoren optimal nutzen und ihre Nachteile ausgleichen können, müssen die gesammelten Daten zusammengeführt werden. Dieser Prozess wird als Sensorfusion bezeichnet. Dabei übermitteln alle Sensoren ihre Messungen der Umgebung an das Fahrzeug. Eine Software entscheidet dann, welche Daten logisch zusammenpassen, welche verworfen werden müssen und wie relevant sie für die jeweilige Fahrfunktion sind. Als Beispiel veranschaulicht Abbildung 3.13 die Sensorfusion anhand der Positionsbestimmung [48].

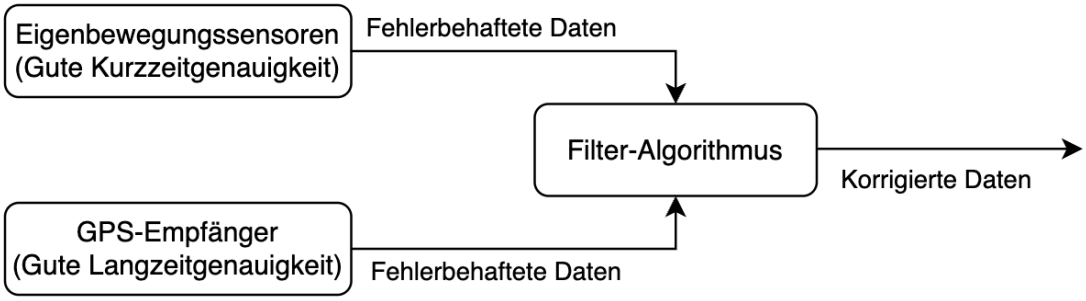


Abbildung 3.13.: Sensorfusion am Beispiel der Positionsbestimmung [49]

Für eine korrekte Positionsbestimmung und zur Kompensation von Fehlern benötigt das System sowohl Daten vom GPS-Empfänger als auch von den Eigenbewegungssensoren. Die Fusion der Daten ermöglicht es, diese zu validieren und zu verbessern. Das GPS bietet dabei über längere Zeiträume hinweg eine grobe Positionsbestimmung, die durch die Eigenbewegungsdaten, die insbesondere für kurze Distanzen präzise sind, weiter verfeinert werden kann. Sollte das GPS-Signal vorübergehend ausfallen, bietet die Sensorfusion eine mögliche Rückfallebene. Dadurch kann die Positionsbestimmung vorübergehend ausschließlich auf Basis der Eigenbewegungsdaten fortgesetzt werden. Da beide Systeme jedoch Fehler aufweisen können, ist eine zusätzliche Filterung sinnvoll, um unrealistische Signalsprünge herauszufiltern und die Genauigkeit weiter zu erhöhen [49].

4. Forschungsrahmen

In diesem Kapitel wird das Ausgangsproblem geschildert und auf die relevanten Rahmenbedingungen eingegangen. Weiterhin werden die grundlegenden Anforderungen für die prototypische Umsetzung des Projekts zusammengetragen und die entworfenen Konzepte vorgestellt.

4.1. Problemstellung

Die AFGBV regelt die erweiterte Abfahrkontrolle in §13 Abs. 1 (2), Abs. 6 und Abs. 7. Zur Veranschaulichung wurde diese in Abbildung 4.1 dargestellt.

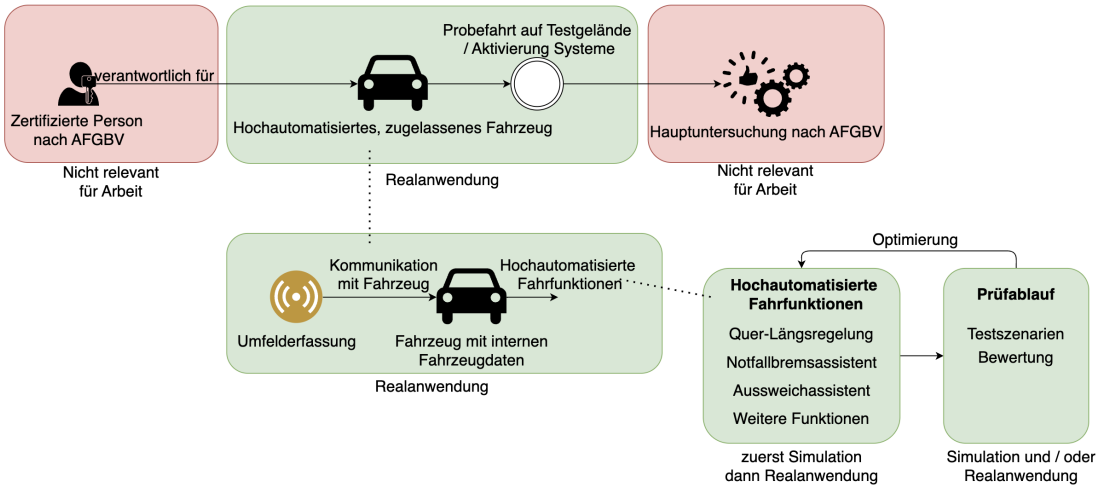


Abbildung 4.1.: Visualisierung der AFGBV §13 Abs. 1(2), 6, 7

Die AFGBV fordert in §13 Abs. 1 (2), Abs. 6 und Abs. 7, dass jedes autonome Fahrzeug, das am öffentlichen Straßenverkehr teilnehmen möchte, vor Fahrtantritt täglich durch eine zertifizierte Person auf seine Funktionstüchtigkeit geprüft werden muss. Diese Überprüfung umfasst eine Probefahrt und die Inspektion der Brems-, Lenk- und Lichtanlagen sowie der Räder und des Fahrwerks. Ebenso müssen die elektronischen Systeme und Sensoren sowie alle mechanischen Systeme für die aktive (z.B. Bremskraftverstärkung) und passive Sicherheit (z.B. Gurte) überprüft werden. In dieser Arbeit wird nicht auf die Anforderungen an die zertifizierte Person oder die Kontrolle nach der Fahrt eingegangen. Der Fokus liegt auf dem autonomen Fahrzeug, seinen Fahrfunktionen und der Durchführung der Probefahrt.

Um die Grundlagen für die Automatisierung dieser Abfahrkontrolle zu schaffen, sind ein Betriebs- bzw. Testgelände und ein Fahrzeug mit autonomen Fahrfunktionen erforderlich. Da das für diesen Zweck vorgesehene Versuchsfahrzeug weder über eine automatisierte Längs- und Querführung noch über die erforderlichen autonomen Fahrfunktionen verfügt, müssen diese Technologien entwickelt

werden. Als eine solche autonome Fahrfunktion wird eine automatische LiDAR-Notbremsfunktion implementiert. Diese Fahrfunktionen müssen getestet, optimiert und verifiziert werden. Hierzu soll zunächst die Entwicklung einer Simulationsumgebung für das Simulieren der Fahrfunktion erfolgen, bevor reale Tests durchgeführt werden. Die Zulassung und Prüfung solcher Systeme sind ebenfalls in der AFGBV geregelt. Um diesen Anforderungen zu entsprechen, soll zusätzlich ein generisches Prüfkonzept entwickelt werden.

4.2. Rahmenbedingungen

4.2.1. Betriebsbereich

Als Betriebsbereich wird das in Abbildung 4.2 dargestellte Gelände genutzt. Es gehört zur HTWD und ist ein Privatgelände, das für das Testen von Prototypen ausgelegt ist. Das Gelände verfügt über ein Testfeld, das mit genormten Fahrbahnmarkierungen und Absperrungen ausgestattet ist. Es weist einen unebenen Boden auf. Die zulässige Höchstgeschwindigkeit auf diesem Gelände beträgt 30 km/h. Das Gelände wurde bereits mittels LiDAR kartiert.

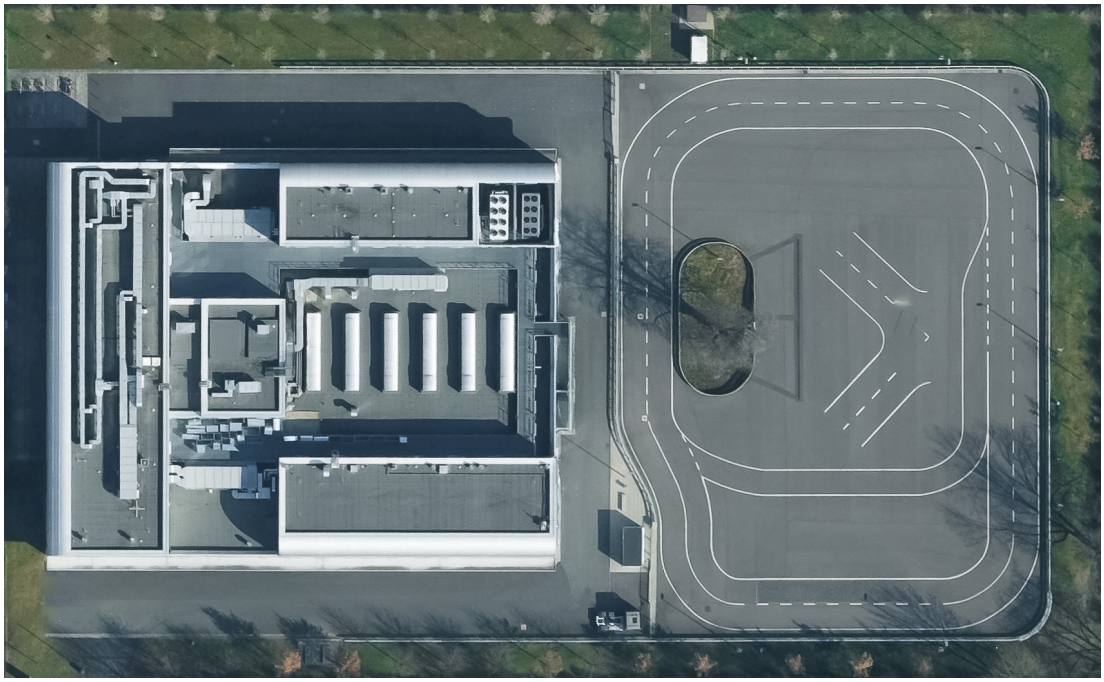


Abbildung 4.2.: Festgelegter Betriebsbereich

4.2.2. Versuchsfahrzeug

Als Versuchsträger steht ein BMW i3 zur Verfügung, der bereits von Mitarbeitern und anderen Studierenden für das automatisierte Fahren umgebaut und vorbereitet wurde. Die vorgenommenen Änderungen am Fahrzeug umfassen den Einbau eines Notausschalters, den Umbau der Lenk- und Fahrpedalanlage, die Implementierung einer CAN-Schnittstelle sowie die Integration eines Computers und eines Displays. Der Computer läuft mit Windows 11 als Betriebssystem. MATLAB und Simulink sind bereits installiert. Bilder sowie Dokumentationen der Umbauten sind in Anlage A und Anlage G.1 zu finden. Zusätzlich wurde ein Ouster OS1-64 LiDAR sowie eine Befestigungsvorrichtung zur Montage auf dem Dach des Fahrzeugs bereitgestellt. Abbildung 4.3 zeigt das Fahrzeug mit dem installierten LiDAR-Sensor.

Abbildung 4.3.: BMW i3 mit montiertem LiDAR-Sensor

Die Abbildung 4.4 zeigt eine Skizze der vorhandenen Modifikationen.

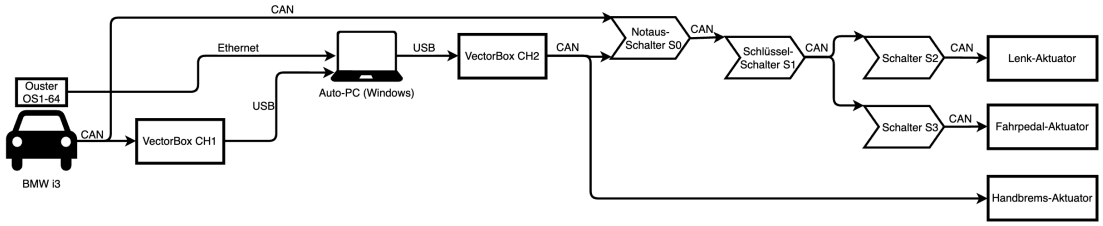


Abbildung 4.4.: Vorhandene Modifikationen im BMW i3

In Abbildung 4.4 ist zu erkennen, dass das Fahrzeug die Möglichkeit bietet, den Lenk- und den Fahrpedal-Aktuator getrennt in Betrieb zu nehmen. Zudem besteht die Option, zwischen dem manipulierten, aus der Vector-Box Channel zwei kommenden, und dem vom Fahrzeug kommenden CAN-Signal zu wechseln. Ein zusätzliches Schloss gewährleistet, dass die Manipulation des Lenk- und Fahrpedal-Aktuators nur funktioniert, wenn sowohl dieses als auch der Notausschalter aktiviert sind. Im Notfall schaltet der Notausschalter alle zusätzlich installierten Sys-

teme ab und setzt das Fahrzeug in den Normalbetrieb zurück. Im Kofferraum ist zusätzlich eine erweiterte Stromversorgung integriert, die es ermöglicht, den Computer und andere benötigte Geräte zu betreiben.

Bei dem verwendeten LiDAR-Sensor handelt es sich um einen OS1 von Ouster. Dieser ist in Abbildung 4.5 dargestellt. Dieser Mid-Range LiDAR-Sensor bietet eine Reichweite von bis zu 200 Metern, wobei die Mindestentfernung bei 0,8 Metern liegt. Er kann bis zu 64 vertikale und 2048 horizontale Ebenen auflösen. Die Abfragerate beträgt bis zu 20 Hertz, wobei der Sensor maximal etwa 1,3 Millionen Punkte pro Sekunde generieren kann. Dies führt dazu, dass nur bestimmte Kombinationen von Einstellungen möglich sind, beispielsweise ist die Konfiguration mit 64 vertikalen Ebenen, 1024 horizontalen Ebenen und einer Abfragerate von 20 Hertz möglich. Das Sichtfeld des Sensors erstreckt sich über 360° horizontal und 45° vertikal. Die Daten werden über Ethernet übertragen. Der Ethernet-Anschluss des LiDAR-Sensors erfolgt entweder direkt oder über den im Kofferraum installierten Switch, der mit dem Computer verbunden ist. Zusätzlich ist der Sensor mit einem Beschleunigungs- und Drehratensensor ausgestattet [33],[50].

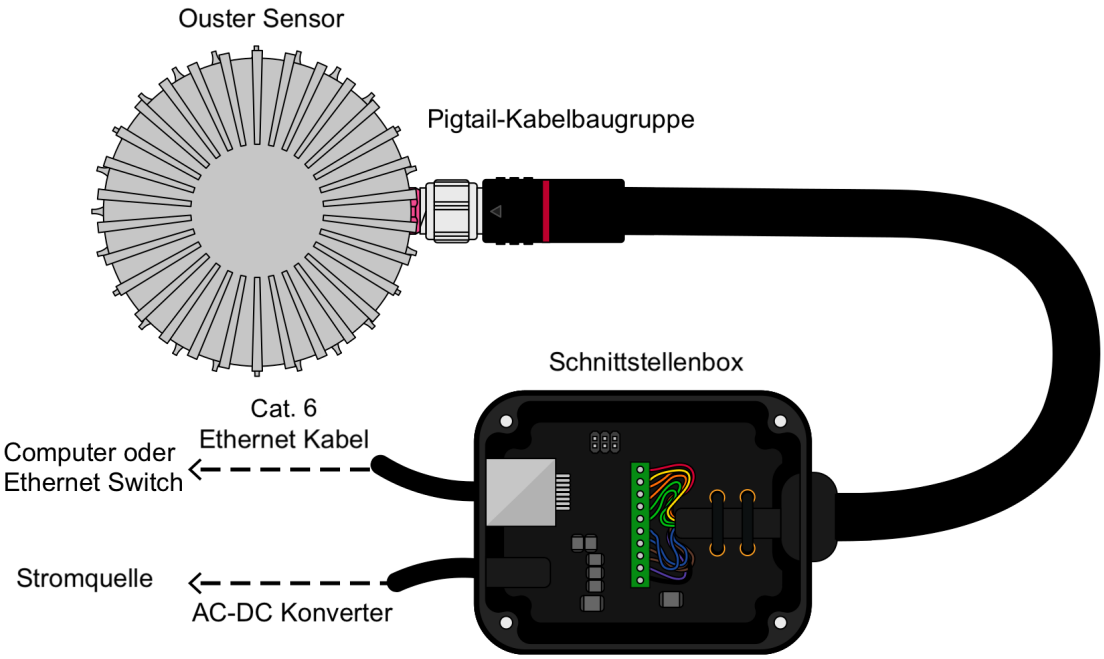


Abbildung 4.5.: LiDAR-Sensor Ouster OS1 [51]

4.3. Anforderungen

4.3.1. Anforderungen an die Längs- und Querführung

Sowohl seitens der Aufgabenstellung als auch durch geltende gesetzliche Vorschriften werden spezifische Anforderungen an die Längs- und Querführung des Fahrzeugs gestellt. Im Folgenden werden die wichtigsten Punkte zusammengefasst. Eine vollständige Übersicht der Anforderungen findet sich in der Anlage B.1. Der BMW i3 der HTWD soll als Versuchsfahrzeug verwendet werden, und das hauseigene Prüffeld dient als Betriebsbereich. Seitens der Aufgabenstellung ist zudem festgelegt, dass MATLAB-Simulink als Softwareumgebung verwendet werden soll und das darin erstellte Modell erweiterbar sein muss. Im Allgemeinen muss die Längs- und Querführung des Fahrzeugs die gültigen Regeln der StVO einhalten. Darüber hinaus sind gemäß der AFGBV sicherheitsrelevante Aspekte zu berücksichtigen, um sicherzustellen, dass das System den Verkehr nicht behindert. Andernfalls wäre eine Aktivierung des Systems nicht zulässig. Da es sich um die Entwicklung eines Prototyps handelt, der vorerst nicht auf öffentlichen Straßen fahren soll, sollen die Anforderungen auf das Wesentliche beschränkt werden. Das primäre Ziel für die Umsetzung besteht darin, sicherzustellen, dass das Fahrzeug keine Schäden verursacht und die Probefahrt zuverlässig durchgeführt wird.

4.3.2. Anforderungen an die autonome Fahrfunktion

Als autonome Fahrfunktion soll eine automatische Notbremse, ausschließlich basierend auf LiDAR, umgesetzt werden. Die Anforderungen, welche seitens der Aufgabenstellung an diese Funktion gestellt werden, entsprechen denen der erweiterbaren Längs- und Querführung. Die AFGBV gibt hierzu nur begrenzte Vorgaben, da hauptsächlich auf die EU-Verordnung 2019/2144 verwiesen wird. Diese Verordnung legt unter anderem fest, dass die Funktion ohne die Verwendung biometrischer Daten auskommen muss und die Möglichkeit bestehen muss, Warnsignale zu unterdrücken. Die Verordnung sieht vor, dass die Notbremsfunktion verpflichtend verbaut ist und in der Lage ist, sowohl auf stehende als auch auf sich bewegende Fahrzeuge vor dem Kraftfahrzeug automatisch und selbstständig zu bremsen. Ebenso muss die Erkennung von Fußgängern und Radfahrern sicher erfolgen. Unter den Automobilherstellern hat sich durchgesetzt, dass das System Warntöne und optische Signale ausgeben muss. Die Funktion wird zudem in zwei Schritten ausgeführt: einer Anbremsung und einer Notbremsung. Darüber hinaus sind diese Systeme überwiegend für den Stadtverkehr bis zu einer Geschwindigkeit von 60 km/h konzipiert. Für die vorliegende Arbeit wird diese Fahrfunktion genauer betrachtet, wobei möglichst auf Vereinfachungen verzichtet werden soll. Eine umfassende Übersicht der Anforderungen findet sich in der Anlage B.2.

4.3.3. Anforderungen an die Prüfung

Für die Prüfung verlangt die Aufgabenstellung, dass das Konzept generisch ist. Die AFGBV legt dazu fest, dass die Prüfungen auf einem vom Fahrzeughersteller erstellten Prüfkatalog basieren müssen, der möglichst viele Testszenarien abbildet. Während einige Tests in Realversuchen durchgeführt werden müssen, kann in anderen Fällen eine reine Simulation ausreichen. Dabei ist es entscheidend, dass die Simulation möglichst realitätsnah ist. Zusätzlich muss eine umfassende Funktionsbeschreibung Teil des Prüfkatalogs sein. Für die Testszenarien sind insbesondere die Vorgaben der The European New Car Assessment Programme (EURO NCAP) von Bedeutung, da sie bereits Testszenarien für bestimmte Fahrfunktionen beschreiben und durchführen. Eine vollständige Übersicht der Anforderungen ist in der Anlage B.3 zu finden. Aufgrund der prototypischen Umsetzung besteht das Ziel hauptsächlich darin, einen grundlegenden Aufbau dieses Prüfkatalogs zu erstellen, sodass erste Tests der Fahrfunktionen damit durchgeführt werden können.

4.4. Konzeption

In diesem Abschnitt wird das entwickelte Konzept für die Arbeit vorgestellt und erläutert. Zur besseren Verständlichkeit wurde es in vier Unterbausteine gegliedert.

4.4.1. Aufbau einer Simulationsumgebung

Da die Anforderungen sowohl Real- als auch Simulationstests erfordern, soll ein ausreichend genauer digitaler Zwilling des Testfeldes und des Fahrzeugs für Simulationen erstellt werden. Dieser Schritt ermöglicht es, Änderungen an bestehenden Systemen, neue Funktionen oder Tests neuer Prüfabläufe zunächst ohne ein reales Fahrzeug durchzuführen. Dies minimiert sowohl die Risiken, die bei der Implementierung am realen Fahrzeug auftreten können, als auch die Kosten.

4.4.2. Konzept für die erweiterbare automatisierte Längs- und Querführung

Um das autonome Fahrzeug sicher zu bewegen, benötigt es eine zuverlässige Längs- und Querführung. Diese soll über die Zeit erweiterbar und mit den Anforderungen aus Tabelle B.1 vereinbar sein. Aus diesen Gegebenheiten wurde das Konzept in Abbildung 4.6 abgeleitet.

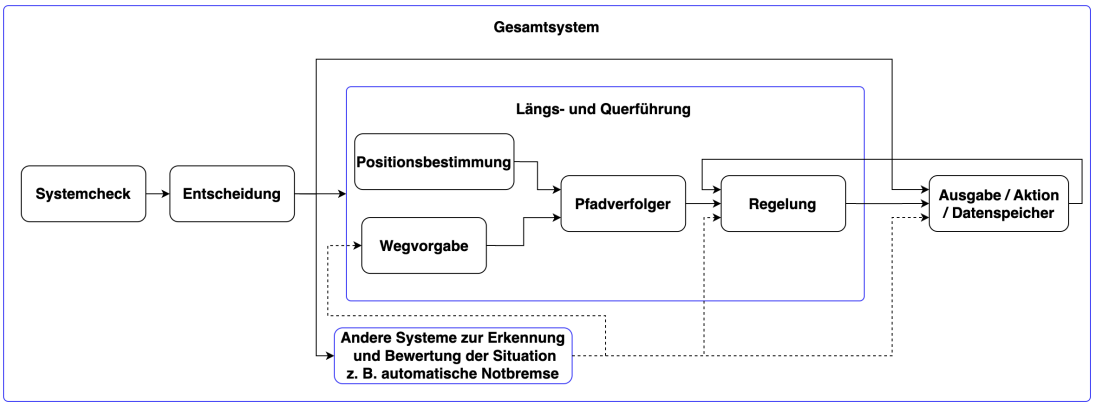


Abbildung 4.6.: Konzeptskizze der Längs- und Querführung

Das Konzept zur autonomen Längs- und Querführung beschreibt den grundsätzlichen Ablauf der Fahrzeugsteuerung mit autonomen Fahrfunktionen, welche den Anforderungen aus Tabelle B.1 genügt. Das System wurde in modularen Bausteinen entworfen, sodass Erweiterungen problemlos hinzugefügt oder einzelne Funktionen leicht ausgetauscht werden können. Diese modulare Struktur ermöglicht auch die Auslagerung der einzelnen Funktionen auf verschiedene Steuergeräte. Im Folgenden wird das Konzept näher erläutert:

Systemcheck
Im ersten Schritt des Prozesses wird ein umfassender Systemcheck des Gesamtsystems durchgeführt. Dieser dient dazu, die Funktionsfähigkeit und Betriebsbereitschaft aller Systemkomponenten zu überprüfen, einschließlich der Hard- und Softwareelemente.

Entscheidung
Nach dem Systemcheck erfolgt die Entscheidung über die Einsatzbereitschaft der Systeme. Diese Entscheidung kann positiv, neutral oder negativ ausfallen und bestimmt das weitere Verhalten der nachfolgenden Systeme. Im Fall eines positiven Feedbacks setzen die Systeme ihren Betrieb wie vorgesehen fort. Bei einem neutralen Ergebnis, bei dem das System nicht einsatzbereit ist, wird, sofern vorhanden, auf eine Notfallfunktion umgeschaltet. Bei einem negativen Ergebnis, in dem das System als nicht betriebsbereit gilt, wird es vollständig deaktiviert. Für jede der drei möglichen Ausgänge wird eine entsprechende Hinweisausgabe bereitgestellt.

Positionsbestimmung
Befindet sich das System der Längs- und Querführung im neutralen oder positiven Modus, wird als nächster Schritt die aktuelle Position des Fahrzeugs ermittelt. Dies kann durch verschiedene Techniken realisiert werden, darunter die GPS-Ortung, die Ortung durch Sensorfusion oder andere Lokalisierungsmethoden.

Wegvorgabe
Nachdem die Positionsbestimmung erfolgt ist, benötigt das System Informationen über die Strecke, die das Fahrzeug zurücklegen soll. Hierbei können verschiedene Ansätze verfolgt werden, wie zum Beispiel die Vorgabe des Weges durch Wegpunkte oder die dynamische Analyse der Route mittels Pfadfindungsalgorithmen.

Pfadverfolger
Da das System nun sowohl die Ist- als auch die Soll-Position des Fahrzeugs kennt, können im nächsten Schritt Abweichungen ermittelt und die erforderlichen Stellgrößen berechnet werden. Dies ermöglicht eine kontinuierliche Überprüfung, ob die Bewegungen des Fahrzeugs dem vorgegebenen Pfad entsprechen.

Regelung
Nachdem die Stellgrößen des Pfadverfolgers ausgegeben wurden, muss nun versucht werden, die entsprechenden Aktuatoren auf diese Größen in Echtzeit zu regeln.

Andere Systeme zur Erkennung und Bewertung der Situation
Da der reine Pfadverfolger nicht alleinig für das Bestimmen des Fahrzeugverhaltens zuständig ist, sind parallel weitere Systeme erforderlich, die die Situation und Umgebung analysieren und bewerten. Auf Basis dieser Informationen wird erneut beurteilt, ob das Fahrzeug seinen geplanten Weg fortsetzen kann oder ob andere Maßnahmen erforderlich sind. Insbesondere gefährliche Situationen können das Verhalten des Systems beeinflussen. So kann beispielsweise die Wegvorgabe durch einen Ausweichassistenten angepasst, die Regelung durch einen Notbremsassistenten modifiziert oder zusätzliche Warnhinweise ausgegeben werden.

Ausgabe / Aktion / Datenspeicher
Im letzten Schritt werden die erforderlichen Aktionen umgesetzt, die zugehörigen Hinweise ausgegeben und alle relevanten Informationen sowie Daten im digitalen Datenspeicher gesichert. Beispielsweise könnte eine Aktion das Einleiten einer Notbremsung sein. Als Hinweis könnte das Abspielen eines Warntons vorgesehen werden. Dieser sicherheitskritische Fall muss ebenfalls im Datenspeicher dokumentiert werden, um eine spätere Datenanalyse zu ermöglichen.

Da das System kontinuierlich seine Funktionalität sicherstellen muss, wird der gesamte Prozess während der gesamten Nutzungsdauer wiederholt.

4.4.3. Konzept für eine sicherheitsrelevante autonome Fahrfunktion

Der Baustein „Andere Systeme zur Erkennung und Bewertung der Situation“ in Abbildung 4.6 umfasst auch die gemäß AFGBV erforderlichen autonomen Fahrfunktionen, wie beispielsweise den Notbremsassistenten und den Ausweichassistenten. In dieser Arbeit wird der autonome Notbremsassistent als sicherheitsrelevante Fahrfunktion betrachtet. Das entwickelte Konzept wird in Abbildung 4.7 dargestellt.

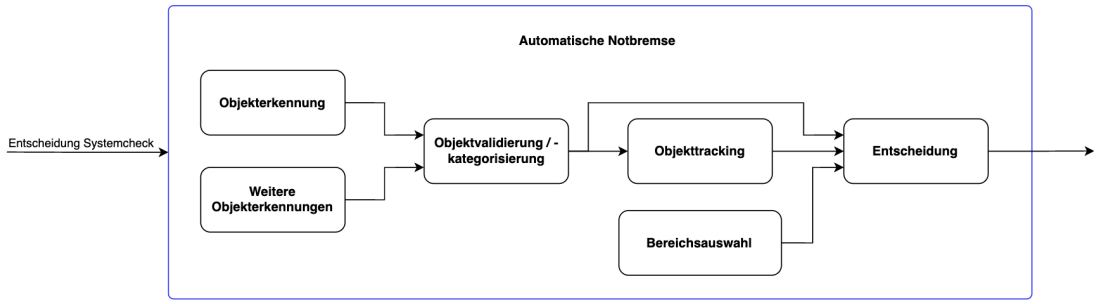


Abbildung 4.7.: Konzeptentwurf der automatischen Notbremse

Das Konzept des Notbremsassistenten wurde auf Grundlage der Anforderungen in Anlage B.2 entwickelt. Um eine bessere Erweiterbarkeit und Austauschbarkeit zu gewährleisten, wurde das System in einzelne Bausteine unterteilt. Diese Modularität ermöglicht auch die Auslagerung einzelner Funktionen auf verschiedene Steuergeräte, um den Rechenaufwand effizient zu verteilen. Im Folgenden werden die einzelnen Bausteine des Notbremsassistenten vorgestellt und erläutert:

Entscheidung Systemcheck
Im ersten Schritt überprüft das System alle relevanten Komponenten. Dazu greift es auf die Ergebnisse des Systemchecks zurück. War dieser Check erfolgreich, setzt das System seinen Betrieb wie vorgesehen fort. Bei einem negativen Ergebnis wird das System deaktiviert. Die entsprechenden Ausgaben und Aktionen werden entsprechend der Entscheidung des Systemchecks ausgelöst und im Datenspeicher dokumentiert. Im Falle eines Defekts könnten beispielsweise ein Warnton oder eine Warnleuchte aktiviert werden.

Objekterkennung
Die Objekterkennung ist dafür zuständig, das Umfeld des Fahrzeugs zu erfassen und zu analysieren. Dieser Baustein soll alle relevanten Objekte identifizieren und dem System bereitstellen. Für diesen Zweck sollten im Optimalfall mehrere unabhängige Systeme herangezogen werden. Eine effektive Lösung könnte beispielsweise die Kombination von Kamera-, LiDAR- und Radar-Sensoren sein, um eine umfassende und präzise Objekterkennung zu gewährleisten.

Objektvalidierung
Nachdem alle Objekte erkannt wurden, erfolgt in diesem Baustein die Sensorfusion. Dabei werden die erfassten Objekte validiert und kategorisiert. Das bedeutet, dass die Objekte auf ihre Richtigkeit überprüft und in relevante Gruppen eingeteilt werden. Zu den relevanten Gruppen gehören beispielsweise Autos, Fußgänger und Radfahrer.

Objekttracking
Alle validierten Objekte sollen anschließend, sofern möglich, verfolgt werden. Das bedeutet, dass die Objekte kontinuierlich erfasst werden, um ihre Positionen auch bei kurzzeitigem Sichtverlust zu schätzen und dadurch ihre Bewegungen zuverlässig überwachen zu können.

Bereichsauswahl
In diesem Baustein wird der Bereich festgelegt, der für die Notbremsung von Bedeutung ist. Es kann sinnvoll sein, diesen Bereich dynamisch zu gestallten oder in kleinere Unterbereiche zu gliedern, um präzisere Entscheidungen treffen zu können.

Entscheidung
Sowohl die validierten als auch die getrackten Objekte fließen in den Entscheidungsprozess ein. In diesem Schritt wird mithilfe des festgelegten relevanten Bereichs entschieden, ob ein Objekt im kritischen Bereich erkannt wurde. Die jeweilige Entscheidung wird dann umgesetzt und beeinflusst das Verhalten des Fahrzeugs. Falls keine kritische Situation erkannt wird, sind keine weiteren Aktionen erforderlich. Bei Erkennung einer kritischen Situation, wie etwa einer unmittelbaren Gefahr, muss jedoch eine Notbremsung eingeleitet werden. Dies erfordert unter anderem die Weitergabe der Daten an das System zur Längs- und Querführung. Zudem müssen optische und akustische Warnsignale ausgegeben sowie ein Eintrag im Datenspeicher vorgenommen werden.

Auch hier muss das System kontinuierlich überwacht werden, um die Umwelt fortlaufend wahrzunehmen und zu bewerten. Dies erfordert eine ständige Ausführung des Systems, um eine kontinuierliche und zuverlässige Funktionalität sicherzustellen.

4.4.4. Generisches Konzept für Prüfabläufe

Laut den Anforderungen der AFGBV muss ein Prüfszenarienkatalog erstellt werden. Auf Basis der Anforderungen aus Anlage B.3 wurde das in Abbildung 4.8 dargestellte Konzept entwickelt.

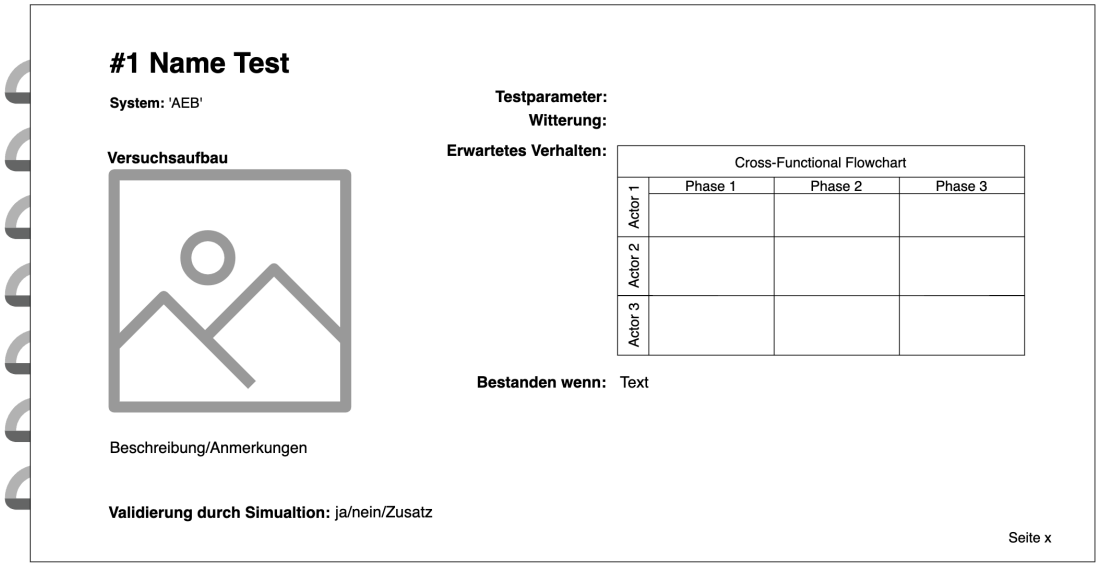


Abbildung 4.8.: Konzeptentwurf des Prüfkataloges nach AFGBV

Dieses Konzept sieht einen Prüfkatalog vor, der über die Zeit erweiterbar ist. Der Katalog soll zunächst eine kurze funktionale Beschreibung jedes Systems enthalten und anschließend die jeweiligen Testszenarien detailliert darstellen und erläutern. Dabei ist es wichtig, dass jedes Szenario eindeutig definiert und durch eine Skizze visualisiert wird. Die AFGBV legt beispielsweise fest, dass mindestens drei Parameterkonstellationen getestet werden müssen. Auch wird festgelegt, wann eine Simulation ausreicht oder durch einen Realversuch ergänzt werden sollte. Zudem muss das erwartete Verhalten der Systeme beschrieben werden.

5. Implementierung

Dieses Kapitel befasst sich mit der Umsetzung des entwickelten Konzepts. Dazu wurden die in Anlage C beschriebenen Programme verwendet.

Vor Beginn des eigentlichen Projekts wurde die Datenorganisation strukturiert. Dadurch konnte eine geordnete Struktur geschaffen werden, die alle bereits gewonnenen oder zukünftig erlangten Erkenntnisse über den Versuchsträger und die dazugehörigen Projekte festhält. Dieser Schritt gewährleistet, dass jeder Mitarbeiter Zugriff auf das Projekt hat, wodurch es fortgeführt und erweitert werden kann. Es wurde insbesondere Material, welches den BMW i3 und dessen Umbauten betrifft, zusammengetragen. In der Forschungsabteilung existierten bereits verschiedene Verzeichnisse mit dazugehörigen Informationen, welche ebenfalls betrachtet, sortiert und integriert werden konnten. Um die Verwaltung im gesamten Team zu gewährleisten, wurde die Informationssammlung in Englisch verfasst. In Absprache mit dem Team wurden Übergruppen definiert. Abbildung 5.1 zeigt das in GitLab angelegte Projekt.

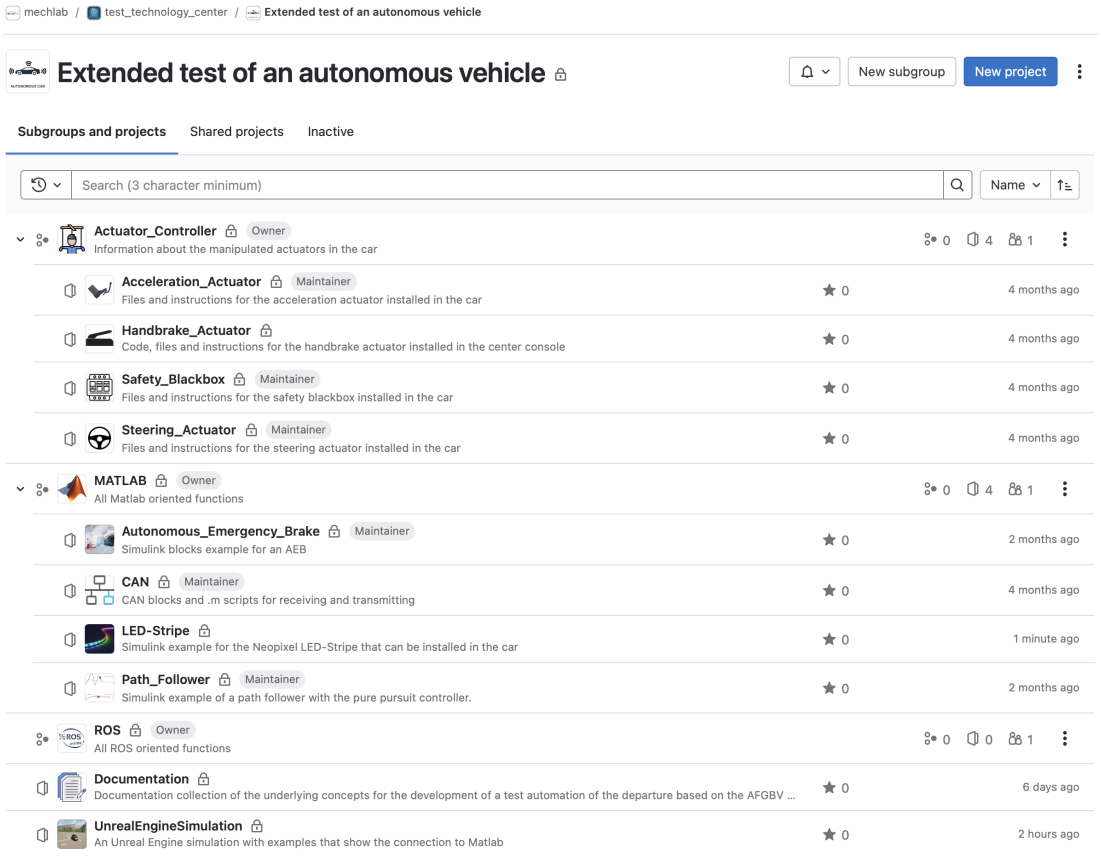


Abbildung 5.1.: Projektorganisation im internen Mechlab-Bereich von GitLab

Darin sind die Übergruppen zu sehen, welche die im Fahrzeug zusätzlich verbauten Aktuatoren, MATLAB, ROS, die Dokumentation und die Simulation umfassen. Innerhalb dieser Übergruppen sind entweder direkt dazugehörige Dateien und Be-

schreibungen bereitgestellt, oder es werden die einzelnen Bausteine aufgelistet. Diese enthalten dann wiederum Informationen oder Fortschritte, wie beispielsweise Softwarelösungen für verschiedene Sensoren, die zu den jeweiligen Bausteinen gehören. Die vorher sortierten Daten wurden den einzelnen Unterprojekten zugeordnet und entsprechend aufbereitet. Zur zusätzlichen Übersichtlichkeit erhielten die Gruppen Vorschaubilder und Kurzbeschreibungen. Im Laufe der Arbeit mussten regelmäßig Daten hochgeladen und aktualisiert werden. Der Link zum GitLab-Speicher ist in Anlage G.1 zu finden.

5.1. Umsetzung der Simualtionsumgebung

Um die Simulationsumgebung zu erstellen, wurde zunächst die Unreal Engine (UE) von der Herstellerseite heruntergeladen und die Version 5.1 installiert. Da MATLAB für die Anbindung zur UE bestimmte Toolboxen benötigt, mussten diese ebenfalls heruntergeladen und installiert werden. Anschließend erfolgte mithilfe der Tutorials auf der offiziellen MATLAB-Seite [52],[53] die Anbindung der Engine.

Danach konnte das Testfeld digitalisiert werden. Dafür musste zunächst die Grundplatte in die UE eingefügt werden, auf der der digitale Zwilling des Betriebsgeländes aufgebaut ist. Zur Erstellung des Geländes konnte die bereitgestellte LiDAR-Karte des Betriebsgeländes verwendet werden, welche im Rahmen der Diplomarbeit von Dipl.-Ing. Tristan Weiland angefertigt wurde.

Diese Karte wurde in Blender importiert und anschließend iterativ modifiziert, sodass daraus ein dreidimensionales Netz des Betriebsgeländes entstand. Abbildung 5.2 zeigt das in Blender erzeugte Flächenmodell, welches mithilfe der sogenannten Modifier erstellt werden konnte. Diese Modifier sind konfigurierbar und konvertieren die Punkte zunächst in Dichtefelder. Das bedeutet, dass um die einzelnen Punkte Volumenelemente erzeugt werden. Anschließend werden diese durch viele kleine Dreiecke verbunden. Je nach den Einstellungen der beiden Modifier werden verschiedene Netze aufgespannt und erzeugt [54].

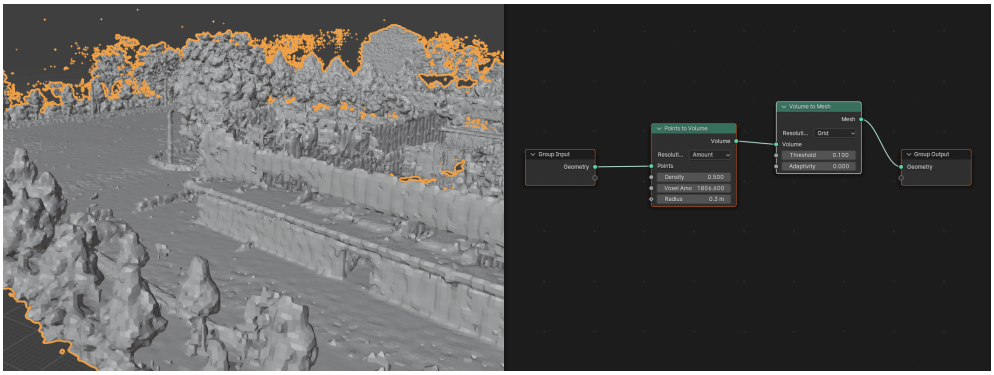


Abbildung 5.2.: Konvertierung der Punktwolke zum Flächenmodell

Da durch die Konvertierung fehlerhafte Punkte in der Punktwolke verstärkt wurden, musste die generierte Oberfläche noch geglättet und störende Bereiche entfernt werden. Anschließend musste das Gelände im Dateiformat $.obj$ exportiert werden, damit die UE die Daten nutzen kann.

Nachdem das Netz in die UE importiert wurde, musste eine flache Bodenplatte eingefügt und an das Netz angepasst werden, um die Höhenunterschiede des Testfeldbodens im Modell zu berücksichtigen. Dies konnte mit den eigenen Bearbeitungswerkzeugen der UE durchgeführt werden.

Die Fahrspuren wurden mithilfe einer generativen Textur grob nachgebildet. Dazu wurde eine Textur aus dem in UE implementierten, kostenlosen 3D-Ressourcenshop namens Quixel heruntergeladen, nach Quelle [55] modifiziert und eingebunden.

Als Nächstes konnte mithilfe der MATLAB-Anleitung [56],[57] das in Abbildung 5.3 gezeigte dreidimensionale Modell des BMW i3, welches im Mechlab bereits vorhanden war, in Blender so modifiziert werden, dass es von MATLAB in der UE gesteuert werden kann. Dabei mussten insbesondere bestimmte Namensgebungen und Ausrichtungen beachtet werden. Dieses Modell wurde anschließend als $.fbx$-Datei exportiert und als Fahrzeugvorlage in die UE eingefügt.

Abbildung 5.3.: Vorbereitetes 3D-Modell des BMW i3 für die Unreal Engine

Abbildung 5.4 zeigt den fertiggestellten digitalen Zwilling, einschließlich des integrierten Fahrzeugs.

Abbildung 5.4.: Ausschnitt aus der Unreal Engine Simulationsumgebung

Um unabhängig von der ressourcenintensiven Engine zu sein, wurde die Simulationsumgebung abschließend in eine auf Windows ausführbare Datei verpackt [58]. Diese Datei wurde auf GitLab hochgeladen und kann an die Teammitglieder verteilt werden. Dadurch ist gewährleistet, dass alle Mitglieder ihre eigenen Simulationen durchführen können. Das fertige Unreal Engine Projekt der Simulationsumgebung und zusätzliche Dateien sind in Anlage G.2 (Simulationsumgebung) zu finden. Die verteilbare und mit MATLAB ansteuerbare $.exe$-Datei, sowie zugehörige Dateien sind in Anlage G.2 (Simulationsumgebung/Windows) abgelegt.

5.2. Umsetzung der Längs- und Querführung

Nachdem die Simulationsumgebung umgesetzt war, richtete sich der Fokus auf die Längs- und Querführung des Fahrzeugs. Dabei mussten zunächst zwei wesentliche Fragen geklärt werden: Erstens, welche Daten werden vom Fahrzeug geliefert und zweitens, welche Strecke soll das Fahrzeug zurücklegen? Daraufhin wurde festgelegt, dass die Daten über das bereits im Fahrzeug integrierte CAN-Case abgegriffen werden sollen. Für die Testfahrt erfolgte mit dem betreuenden Professor die Entscheidung, eine zuvor aufgezeichnete Route als Strecke zu verwenden.

Die anfängliche Datenaufbereitung beinhaltete einen CAN-Empfangs- und Sende-Block für Simulink. Nach Rücksprache mit dem Betreuer, der die entsprechende CAN-Datenbank für den BMW i3 bereitstellte, konnte die Positionsbestimmung begonnen werden.

5.2.1. Positionsbestimmung

Für die Simulation war die Positionsbestimmung vergleichsweise einfach, da MATLAB über die UE-Anbindung bereits die Position des Fahrzeugs in x- und y-Koordinaten in Metern sowie den Gierwinkel in Radiant ausgibt. Schwieriger gestaltete sich jedoch die Positionsbestimmung des realen Fahrzeugs, da dieses über kein ausreichend genaues Ortungssystem verfügt. Daher musste ein System zur Ortung entwickelt werden.

Nach Rücksprache mit dem betreuenden Professor sollten für diesen Prototyp weder hohe Kosten entstehen noch umfangreiche Umbauten stattfinden. Daher wurde entschieden, die Ortung über die Eigenbewegungsdaten des Fahrzeugs zu realisieren. Dazu wurden die Geschwindigkeit, die Gierrate und die hinteren Raddrehzahlen aus dem CAN-Empfangs-Block, gemäß Abbildung 5.5, abgegriffen und aufgezeichnet [59].

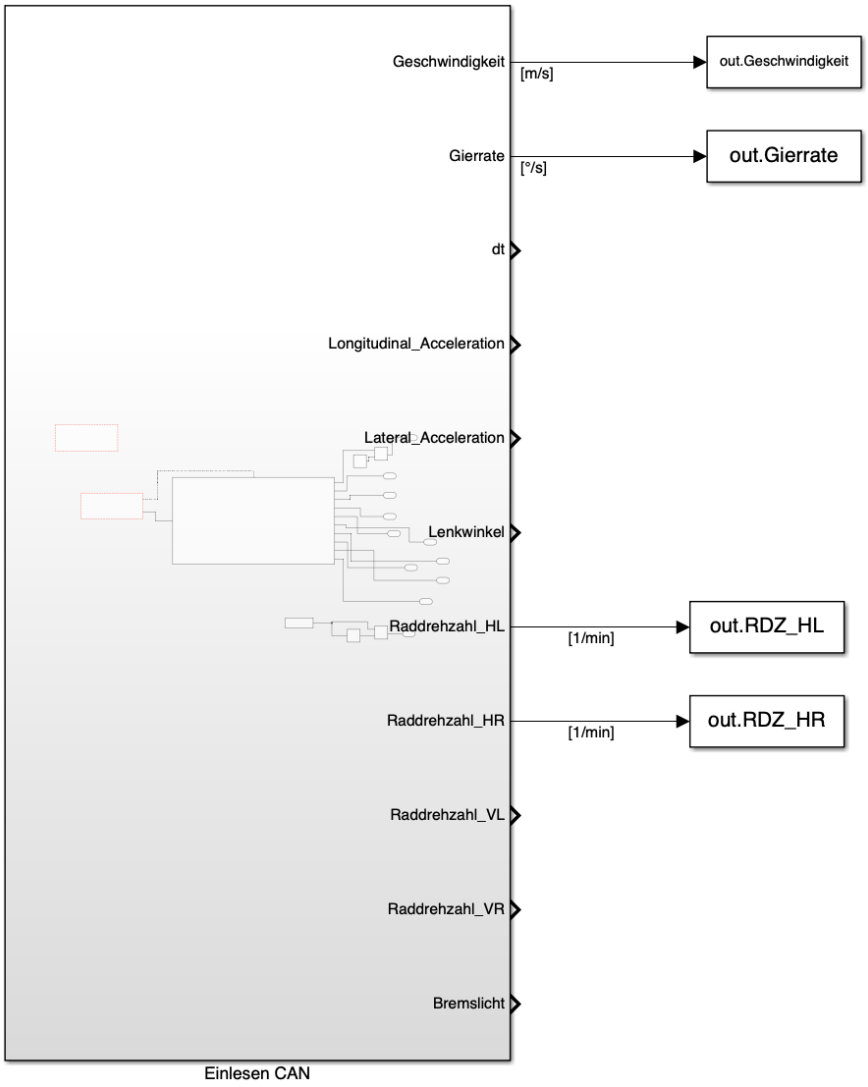


Abbildung 5.5.: Modell zur Messung der Gierrate, der Geschwindigkeit und der Raddrehzalen

Anschließend konnte ein Algorithmus entwickelt werden, der auf den Gleichungen 2.1 und 2.2 basiert. Mithilfe der aufgenommenen Geschwindigkeit und Zeit konnte die zurückgelegte Strecke bestimmt werden. Für die Berechnung des Winkels ϕ wurden zwei Ansätze umgesetzt und verglichen. Der erste Ansatz beruht auf der Berechnung mithilfe der ausgelesenen Gierrate vom Gierratensensor, während der zweite Ansatz die Berechnung durch Gleichung 5.1 mithilfe der ausgelesenen Raddrehzahlen von den Raddrehzahlsensoren vornimmt.

$$\phi = \frac{\Delta Raddrehzahlen * Abrollumfang}{Spurbreite} \cdot \frac{180}{\pi} \tag{5.1}$$

Aufgrund der Ergebnisse aus Auswertung 6.1.1 wurde sich dazu entschieden mit der Berechnung basierend auf der Gierrate fortzufahren.

5.2.2. Aufzeichnung einer Testfahrt

Für die Aufzeichnung des Referenzpfades wurde das Modell aus Abbildung 5.5 mit dem im Unterabschnitt 5.2.1 ausgeführten Algorithmus erweitert. Das erweiterte Modell ist in Abbildung 5.6 zu sehen. Alle zugehörigen Dateien sind in Anlage G.2 (Pfadverfolger/Testfahrt) zu finden.

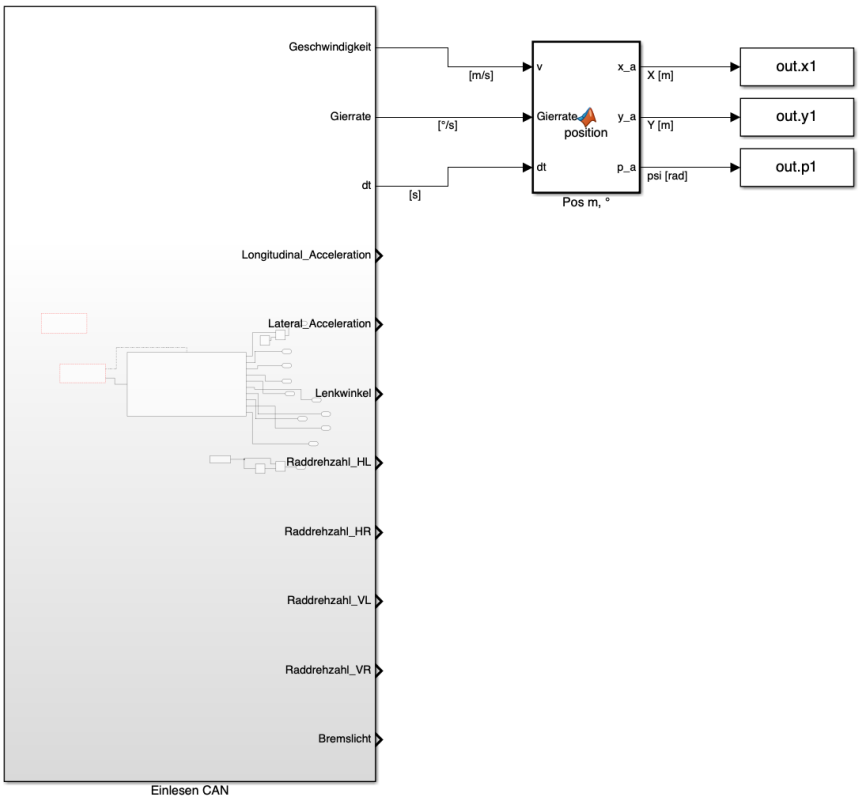


Abbildung 5.6.: Modell zur Aufzeichnung des Referenzpfades

Als nächstes musste ein Referenzpfad festgelegt und eingefahren werden. Die gewählte Strecke umfasste die Ausfahrt vorwärts aus der Garage des BMW i3, eine Runde über das Testfeld und die Rückfahrt in die Garage. Dabei fährt das Fahrzeug

auf dem Rückweg vorwärts in die Garage, sodass es bei der theoretisch folgenden Durchsicht in der richtigen Ausgangsposition, zum Beispiel für einen bereits aufgebauten Test der Scheinwerfer-Einstellungen, steht. Um für spätere Versuche eine konsistente Ausgangsposition zu gewährleisten, wurde die Startposition des Fahrzeugs markiert. Diese Markierung ist in Abbildung 5.7 zu sehen.

Abbildung 5.7.: Startposition des BMW i3

Dafür wurde die Mitte des vorderen und hinteren linken Rades mit Klebeband auf dem Boden markiert. Zusätzlich wurde ein lokales Koordinatensystem festgelegt, welches dem der Simulationsumgebung gleicht und für die realen Tests sowie zukünftige Versuche gelten soll. Der Nullpunkt des Koordinatensystems befindet sich dabei am hinteren linken Rad. Die y-Achse verläuft positiv in Fahrtrichtung, und die positive x-Achse zeigt in das Rad hinein. Zudem wurde der Berechnung der Position ein Startoffset von 90 Grad hinzugefügt. Dies soll sicherstellen, dass das festgelegte Koordinatensystem für das Testfeld und die Startpose des Fahrzeugs übereinstimmen. Um die Startposition in der Simulation zu berücksichtigen, musste die Simulationsumgebung entsprechend angepasst werden. Das Koordinatensystem und der aufgenommene Pfad sind in 5.8 zu sehen.

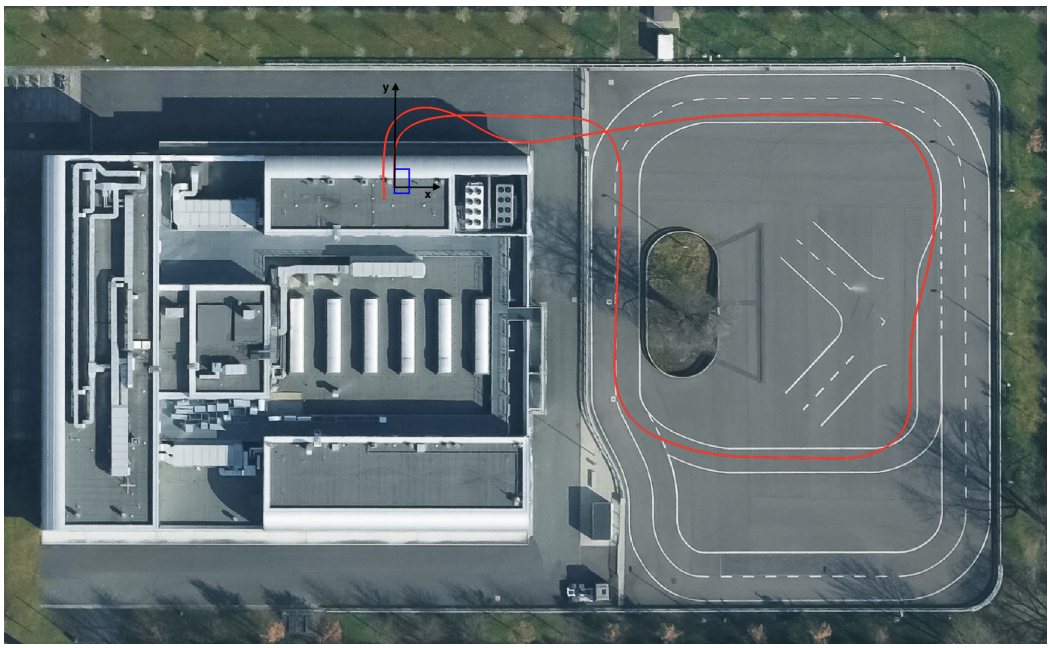


Abbildung 5.8.: Koordinatensystem-Orientierung und aufgezeichnete Testfahrt

Das blaue Rechteck stellt das Fahrzeug in der Startposition dar, das den rot dargestellten Pfad abgefahren ist. Obwohl versucht wurde, sich an die Fahrbahnmarkierungen zu halten, ist die in der Auswertung 6.1.1 angesprochene Abweichung der Positionsbestimmung zu erkennen. Dadurch ist das eingezeichnete Ende der Fahrt zur linken statt zur rechten Seite versetzt. Die reale Endposition des hinteren Linken Rades lag bei $x = 1,69m$, $y = 2,23$. Die Pfadverfolgung und somit das Abfahren des Pfades sollte aufgrund der gleichen Berechnungsgrundlage jedoch nicht davon betroffen sein.

5.2.3. Erweiterbares Längs- und Querführungs Simulink Modell

Damit die aufgenommene Testfahrt mithilfe eines Simulink-Modells abgefahren werden konnte, musste zunächst eine Entscheidung über den grundlegenden Regleransatz getroffen werden. Nach eingehender Recherche fiel die Wahl auf den im Unterabschnitt 2.4.2 erläuterten Pure Pursuit Regler. Dieser Regler ist in der von MATLAB bereitgestellten Navigation Toolbox bereits als vorgefertigter Simulink-Block verfügbar. Obwohl der Pure Pursuit Regler aufgrund seiner Eigenschaften in erster Linie für geringe Geschwindigkeiten ausgelegt ist, konnte er für den prototypischen Einsatz implementiert werden.

Das Modell für die Positionsbestimmung, wie in Abbildung 5.6 dargestellt, wurde entsprechend erweitert. Das erweiterte Modell, welches in Anlage E.1 enthalten ist und auf einer MATLAB-Vorlage [60] basiert, wird folgend detaillierter erläutert. Zur Unterstützung dient der in Anlage E.2 abgebildete Ablaufplan. Alle Dateien, die zum Pfadverfolger gehören, sind in Anlage G.2 (Pfadverfolger) zu finden.

Simulation

Damit das Modell simuliert werden kann, benötigt es entweder ein reales oder ein virtuelles Fahrzeug. Für Zweiteres konnte das in Simulink enthaltene 3-DOF-Modell verwendet werden, welches in der Vehicle Dynamics Blockset Toolbox zu finden ist. Das Modell wurde auf den Modus „Single Track“ umgestellt, um die Berechnungen auf dem im Unterabschnitt 2.2.2 erläuterten Einspurmodell basieren zu lassen. Zudem mussten die Einstellungen zur Spurbreite und zum Gewicht des Fahrzeugs gemäß den in den sortierten Unterlagen enthaltenen Werten angepasst werden [61].

Zur besseren Visualisierung wurde das Fahrzeug mithilfe der Animation 3D Toolbox in die Simulationsumgebung integriert. Diese Einbindung erfolgte über die in der Animation 3D Toolbox bereitgestellten Blöcke. Die Simulationskonfiguration legt die Quelle der Simulation fest, in diesem Fall das komprimierte UE-Programm. Anschließend wurde das Fahrzeugmodell durch einen Fahrzeug-Block eingefügt, wobei der BMW i3 als Modell festgelegt wurde. Eine Kamera wurde hinzugefügt, um das Fahrzeug kontinuierlich zu verfolgen [56].

Für die Einstellung der Aktuatoren des Fahrzeugs mussten der Radlenkwinkel, die Fahrpedalstellung und die Handbremsstellung definiert werden. Der Radlenkwinkel wurde in Radiant umgerechnet, um ihn an das 3-DOF-Modell anzupassen. Die Handbrems- und Fahrpedalstellungen wurden so angepasst, dass sie gegenseitig Einfluss aufeinander haben und das Fahrzeug verzögern. Dabei kann die Verzögerung an die Beschaffenheit des Bodens angepasst werden. Als Ausgänge des Modell-Blocks wurden der Radlenkwinkel, die Geschwindigkeit und die Fahrzeugpose, also die x-, y-Koordinate sowie der Gierwinkel, festgelegt.

Reales Fahrzeug

Für das reale Fahrzeug wurden die gleichen Ein- und Ausgänge wie für die Simulation verwendet, um ein nahtloses Umschalten zwischen beiden Modi zu ermöglichen. Im Gegensatz zum virtuellen Fahrzeug bezieht das reale Fahrzeug seine Daten wie Gierrate, Lenkwinkel und Geschwindigkeit aus dem CAN-Empfangsblock. Die Position wird aus der Gierrate und der Geschwindigkeit berechnet, analog zur Testfahrt im Unterabschnitt 5.2.2. Zusätzlich wird der Lenkwinkel des Lenkrads (±470°) ausgelesen.

Um das Verhalten des Fahrzeugs zu ändern, muss der CAN-Sendeblock verwendet werden. Dieser Block gibt die berechneten Stellgrößen über das am Computer angebundene CAN-Case an die Aktuatoren des Fahrzeugs weiter. Dabei ist zu beachten, dass der vom Modell berechnete Radlenkwinkel, der in den Fahrzeugblock

geht, noch in den tatsächlichen Lenkwinkel umgerechnet wird. Da der Radlenkwinkel des realen Fahrzeugs aber aufgrund des Umbaus nur über ein Stellmoment verändert werden kann, ist eine Umrechnung erforderlich, bevor die Werte an den Aktuator weitergegeben werden. Das maximal anzulegende Moment wurde in den Unterlagen mit +1 und -1 angegeben und im Modell berücksichtigt.

Strecke einlesen
Das Modell muss wissen, welche Strecke es abfahren soll. Dafür wurde die Möglichkeit geschaffen, den aufgenommenen Pfad als x- und y-Koordinaten in Metern einzulesen. Dies erfolgt in Tabellenform und die Abtastrate kann im Block angepasst werden.

Abfrage Endposition
Da der Regler stets vorausschaut, auch wenn die Endposition bereits erreicht ist, muss eine Abbruchbedingung implementiert werden. Diese Funktion wird im Block „endReached“ realisiert. Hier wird überprüft, wie weit die aktuelle Position von der letzten Position des Referenzpfades entfernt ist. Abhängig von dieser Distanz wird die Geschwindigkeit angepasst. Beispielsweise wird die Geschwindigkeit bei der Ein- und Ausfahrt in und aus der Garage auf 3 km/h und auf dem Prüffeld auf 5 km/h begrenzt.

Sobald die Endposition auf etwa 3 Meter genau erreicht ist, wird das Fahrzeug zum Stillstand gebracht. Da die Bremsen des Fahrzeugs nicht ohne neue, zusätzliche Aktuatoren angesteuert werden können, muss das Fahrzeug ausrollen. Das bedeutet, dass die Geschwindigkeit auf Null gesetzt und die Handbremse verzögert angezogen werden muss. Nach Abschluss dieser Maßnahmen wird das Modell gestoppt. Das Stoppen erfolgt verzögert, um sicherzustellen, dass die letzten Aktionen ausgeführt und das Fahrzeug ordnungsgemäß abgestellt wird. Vor Erreichen der Endposition sollen zudem die Räder auf Geradeausfahrt ausgerichtet werden.

Pfadverfolgung
Wie bereits beschrieben, ist der Pure Pursuit Regler eine wesentliche Komponente des Modells. Er erfordert den Referenzpfad mit den x- und y-Koordinaten in Metern, eine vorgegebene lineare Geschwindigkeit in Metern pro Sekunde, eine Vorausschaudistanz in Metern sowie die aktuelle Position des Fahrzeugs mit den x- und y-Koordinaten in Metern. Der Regler berechnet die erforderliche Richtungsänderung, um dem Referenzpfad zu folgen. Die Ausgabe umfasst sowohl den Radlenkwinkel in Radiant als auch die Geschwindigkeit.

Berechnung des Radlenkwinkels

Da das Fahrzeug seine Richtung ausschließlich durch Verstellen des Radlenkwinkels ändern kann, muss dieser zunächst bestimmt werden. Hierfür wird die berechnete Richtungsänderung des Pure Pursuit Reglers mithilfe der Formel 2.9 in den Radlenkwinkel umgerechnet. Es ist zu beachten, dass der Arcustangens-Funktionsblock seine Werte in Radiant ausgibt, welche anschließend in Grad umgerechnet werden müssen. Zur Sicherheit wurde eine Begrenzung des Radlenkwinkels auf +36 ° und -36 ° eingefügt, da dies in etwa dem maximalen und minimalen Radlenkwinkel des BMW i3 entspricht. Diese Berechnung wurde in ein Untersystem verlagert, um die Übersichtlichkeit zu verbessern.

Berechnung und Ausgabe der Stellgrößen

Nachdem die benötigte Geschwindigkeit und der erforderliche Radlenkwinkel ermittelt wurden, müssen die Zielgrößen entsprechend eingestellt werden. Zu diesem Zweck wurden zwei Subsysteme erstellt. Diese Subsysteme benötigen als Eingaben die Ist- und Soll-Werte, wobei die Ist-Werte vom Fahrzeug stammen und die Soll-Werte vom Regler. Die Ausgänge geben die Stellgrößen zurück.

Die Subsysteme enthalten jeweils einen PI-Regler, der aus Simulink-Blöcken zusammengesetzt ist, sowie eine Begrenzung der Ausgabewerte. Die Parameter des Reglers und der Begrenzung sind anpassbar. Die Parameter wurden zunächst auf Basis von Praktikumsunterlagen des Betreuers grob festgelegt und anschließend iterativ angepasst. Die Begrenzung für das Fahrpedal wurde auf minimal 0 und maximal 100 Prozent festgelegt, während für den Lenkwinkel eine Begrenzung von ±470 Grad definiert wurde.

Zur Optimierung der Übersichtlichkeit, Funktionalität und Auswertbarkeit wurden weitere Anpassungen vorgenommen. Dazu zählen Beschriftungen, Schalter zum Wechseln zwischen dem Simulations- und Realmodus sowie die Möglichkeit zur Pfadaufzeichnung. Einige Konstanten, wie beispielsweise die Abtastrate der CAN-Schnittstelle, konnten in den Model-Workspace verlegt werden. Auch das Laden der austauschbaren Referenzstrecke $Mechlab\_Runde.mat$ erfolgt über einen Callback.

Simualtionstests

Zur Absicherung und um erste Informationen über das erstellte System zu erhalten, wurden zuerst Simulationstests durchgeführt. Dabei wurde festgestellt, dass das Fahrzeug bei engen Stellen häufig sehr knapp vorbeifährt und die Tore leicht streift. Diese Beobachtung zeigt, dass solche Stellen bei den realen Versuchen besondere Aufmerksamkeit erfordern.

Realtests

Nachdem die Simualtionstests abgeschlossen waren und das System im Allgemeinen zuverlässig funktionierte, wurde es auf das reale Fahrzeug übertragen. Während der ersten Fahrversuche traten jedoch zwei wesentliche Probleme auf.

Erstens zeigte sich, dass der eingestellte Pure Pursuit Regler im realen Fahrzeug übermäßig stark reagierte. Dies führte dazu, dass das Fahrzeug zunächst Schwierigkeiten hatte, die Garage zu verlassen und zudem die Gefahr bestand, dass es beim Ausfahren das Garagentor streifte. Zweitens versuchte das Fahrzeug übermäßig stark einzulenken, was zur automatischen Abschaltung der Lenkvorrichtung führte. Die Ursache des Problems war zunächst schwer zu identifizieren. Die Vermutung ist, dass der Stellmotor möglicherweise bei zu hohen Momenten aus Sicherheitsgründen abgeschaltet wird.

Um beide Probleme zu beheben, wurden die Parameter des PI-Reglers der Lenkvorrichtung und die Vorausschaudistanz des Reglers iterativ optimiert, bis die Fahrt ohne übermäßiges Aufschwingen und Fehler erfolgreich durchgeführt werden konnte. Nachdem diese Anpassungen vorgenommen wurden, konnte das System für die weiteren Tests in der Auswertung 6.1.2 genutzt werden.

5.3. Umsetzung der automatischen Notbremse

Nachdem die Längs- und Querführung umgesetzt war, konnte die erste Erweiterung entwickelt werden. Da der BMW i3 kein LiDAR-basiertes und in MATLAB umgesetztes Notbremssystem besitzt, wurde hierfür zunächst das Konzept aus dem Unterabschnitt 4.4.3 umgesetzt, in der Simulation getestet und später am realen Fahrzeug implementiert. Um die Funktionsweise des Systems zu erklären, soll Abbildung 5.9 herangezogen werden. Alle Dateien die zur automatischen Notbremse gehören sind in Anlage G.2 (AEB) zu finden.

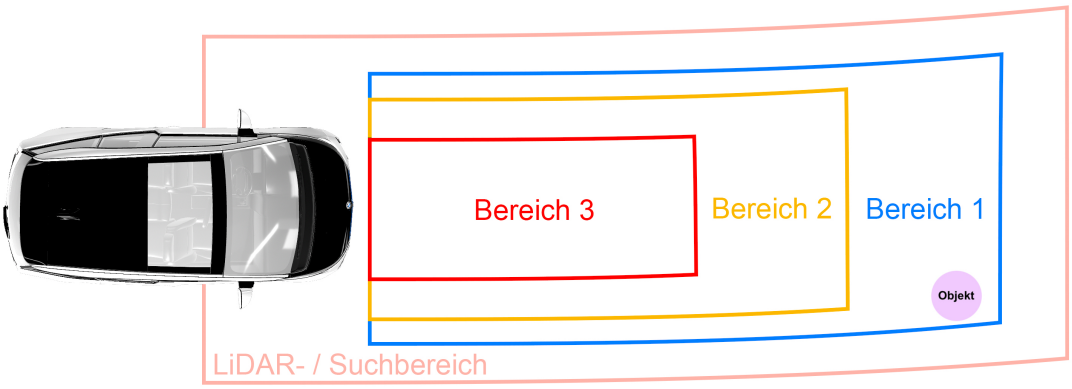


Abbildung 5.9.: Funktionsskizze der automatischen Notbremse bei Kurvenfahrt

Die Punktwolke des auf dem Fahrzeug angebrachten LiDAR-Sensors wird so zurechtgeschnitten, dass sie den Bereich in Fahrtrichtung abbildet. Je nachdem, wie schnell gefahren und wie stark gelenkt wird, passt sich dieser Bereich dynamisch an. Das Gleiche gilt für die drei farbigen Bereiche. Diese stellen die Zonen dar, in denen das System versucht, relevante Objekte zu erkennen. Der blaue Bereich ist unkritisch und gibt nur einen Warnhinweis aus. Erst im gelben Bereich wird eine kurze Aufmerksamkeitsbremsung durchgeführt. Befindet sich ein Objekt im roten Bereich, wird eine Vollbremsung über die Handbremse ausgelöst.

Das fertige Modell, sowie der zugehörige Ablaufplan sind in den Anlagen E.3, E.4 und E.5 zu finden. Auch dieser Aufbau soll nachfolgend näher erklärt werden.

MATLAB-Systemblöcke

Da das Modell auf viele Funktionen der MATLAB-Toolboxen zugreift, die in Simulink nicht vorhanden sind, mussten eigene Simulink-Blöcke mittels MATLAB-Systemblöcken erstellt werden. Sie ermöglichen das Abrufen der MATLAB bekannten Funktionen. Der Quellcode eines Blocks der Notbremse ist in Anlage E.6 zu finden. Er soll als Beispiel und zum Verständnis des Aufbaus dieser Blöcke dienen.

Grundsätzlich können Ein- und Ausgänge festgelegt und Berechnungen durchgeführt werden. Dabei ist jedoch darauf zu achten, dass der Block genau weiß, um welche Art von Signalen es sich handelt. Bei Eingängen muss darauf geachtet werden, ob die Datenmenge variabel oder fest ist. Sollte sie variabel sein, muss ein entsprechender Code-Block eingefügt werden. Bei Ausgängen ist darauf zu achten, dass nur bestimmte Datentypen unterstützt werden und geklärt sein muss, wie groß die Datenmenge ist, ob es sich um ein variables Signal handelt und ob das Signal komplex ist. Variablen Signalen muss eine feste Größe zugeordnet werden, die sie als eine Art Puffer nutzen. Auch die Ausgabe von Simulink-BUS ist möglich. Für die Notbremse wurde bei Objekterkennungen auf den MATLAB-eigenen objectDetection-BUS und bei der Objektverfolgung auf den objectTrack-BUS zurückgegegriffen [62],[63]. Zusätzlich zu diesen Funktionen kann eine Benutzeroberfläche für einstellbare Parameter für den Block festgelegt werden, die für die Berechnung genutzt werden können. Dies ermöglicht das Einstellen der Abfragerate sowie des Ausführmodus des Blocks und auch das Anlegen von Tabs in dieser Oberfläche ist möglich [64],[65]. Systemblöcke sollten möglichst wenig Code enthalten, da das Modell sonst immer auf den langsamsten Block wartet und somit starke Performance-Einbrüche auftreten können [66]. Diese können durch den Simulink-Profiler identifiziert werden. Er ermöglicht es, die Ausführungszeiten der einzelnen Blöcke zu analysieren, um Optimierungsmöglichkeiten zu erkennen und die Effizienz des Modells zu verbessern [67].

Simulation

Um dieses System umzusetzen, wurden Fahrzeug- und LiDAR-Daten benötigt. Hierfür konnte das gleiche Untersystem verwendet werden, das bereits in der Längs- und Querführung implementiert wurde. Der einzige Unterschied bestand darin, dass das darin enthaltene Fahrzeug zusätzlich mit einem LiDAR-Sensor ausgestattet wurde. Dieser Block ist ebenfalls Bestandteil der 3D Animation Toolbox. Die Parameter des Blocks mussten so angepasst werden, dass sie mit den Informationen zum verbauten Ouster übereinstimmen. Die dadurch generierte Punktwolke erzeugt einen neuen Ausgang. Zusätzlich wurde der Ausgang der Fahrzeugposition durch den Lenkwinkel ersetzt, der vorher noch errechnet werden musste.

Reales Fahrzeug

Das reale Fahrzeug wurde in diesem Fall nur durch den CAN-Empfangsblock und den LiDAR-Block dargestellt. Aus dem Empfangsblock wurden nur die Geschwindigkeit und der Lenkwinkel ausgelesen. Der LiDAR-Block stellt den Ouster dar und ist so konzipiert, dass durch kleine Anpassungen möglichst viele Ouster-Sensoren unterstützt werden. Hier wurde er auf den vorhandenen Ouster OS1-64 konfiguriert. Die erzeugte Punktwolke kann zudem gedreht und verschoben werden, um sie für die nachfolgenden Blöcke verwendbar zu machen. Die Punktwolke muss so eingestellt werden, dass sie in Fahrtrichtung zeigt und der Versatz zum Boden berücksichtigt wird.

Wurden alle Daten von der Simulation oder dem realen Fahrzeug zur Verfügung gestellt, können diese in das Hauptuntersystem eingeleitet werden. Dieses enthält die nachfolgenden Blöcke.

Suchbereichauswahl

Zuerst wird die Punktwolke so gedreht und verschoben, dass sie mit der Fahrtrichtung übereinstimmt. Danach wird anhand der Geschwindigkeit und Lenkung ein grober Suchbereich berechnet. Dieser Suchbereich dient dazu, die Punktwolke anschließend zu beschneiden, um zum einen die Leistung für zukünftige Schritte zu verbessern und zum anderen Fehlerkennungen oder Erkennungen außerhalb des betreffenden Bereichs zu vermeiden. Auch der Boden wird aus den zuvor genannten Gründen mithilfe einer MATLAB-Funktion entfernt.

Objekterkennung

Danach wird die beschnittene Punktwolke an die LiDAR-Objekterkennung übergeben. Diese verwendet einen von MATLAB bereitgestellten, clusterbasierten Algorithmus, der darauf abzielt, die Punktwolke in Cluster von Punkten zu unterteilen, die bestimmten Vorgaben entsprechen. Diese Vorgaben, wie die Punktdichte, wer-

den von außen an den Block übergeben und sind entsprechend einstellbar. Die erkannten Cluster werden anschließend über einen Simulink-BUS aus dem Block geführt. Dieser ermöglicht später eine Sensorfusion, da er auch von anderen Sensoren erkannte Objekte empfangen und weiterverarbeiten kann.

Objektauswahl
Die erkannten Objekte werden anschließend in der Objektauswahl anhand vorgegebener Parameter wie Größe und Position aussortiert und kategorisiert. Die verbleibenden Objekte werden über den Simulink-BUS weitergeleitet. Auch die Erkennung mehrerer Objekte ist möglich, indem der Block dupliziert wird, man die $ObjectClassID$ erhöht und neue $DetectParams$ einführt. Dadurch entsteht die Möglichkeit, später eine Fusion mehrerer Objektkategorien durchzuführen.

Objekttracker
Die verbleibenden Objekte, die über den BUS vom Objekttracker empfangen werden, werden von einem vorgefertigten MATLAB-Trackerblock [68] ausgewertet. Erkennt der Algorithmus ein Objekt mehrfach hintereinander, versucht dieser, die zukünftige Position vorherzusagen.

Trajektorieschätzung
Um zu entscheiden, ob die Objekte eine Gefahr darstellen, wird zunächst die Trajektorie des Fahrzeugs basierend auf der Geschwindigkeit und dem Lenkwinkel ermittelt.

Bereichserstellung
Mithilfe dieser Trajektorie werden drei immer kleiner werdende Bereiche definiert. Diese passen sich dynamisch an die Geschwindigkeit und den Lenkwinkel an und lassen sich durch extern vorgegebene Parameter beeinflussen.

Entscheidung
Anhand der zukünftigen Trajektorie und der erkannten sowie getrackten Objekte wird bestimmt, in welchem Bereich sich die Objekte befinden und welche Maßnahmen ergriffen werden müssen. Die resultierenden Entscheidungen, dargestellt durch die Werte 0 oder 1, werden dann zur weiteren Verarbeitung aus dem Untersystem ausgegeben.

Visualisierung
Zur Visualisierung beim Testen oder für Demonstrationszwecke wurde ein Block entwickelt, der die Punktwolke, die erkannten und getrackten Objekte sowie die Bereiche anzeigt. Dieser kann zur Performance-Verbesserung deaktiviert werden.

Nach der Ausführung des Untersystems stehen Werte für nachfolgende Schritte zur Verfügung. Drei Aktionen wurden implementiert und werden nur ausgeführt, wenn der Reset nicht aktiv ist und das entsprechende Signal zur Aktivierung der Aktion vorliegt.

Reset, Überprüfung und Farbausgabe
In diesem Abschnitt wird überprüft, ob der Reset aktiviert ist. Ist dies nicht der Fall, erfolgt eine weitere Prüfung, um festzustellen, ob bereits eine Bremsung stattgefunden hat und ob es sich dabei um eine Notbremsung oder eine Aufmerksamkeitsbremsung handelte. Je nach Entscheidung wird im nächsten Schritt unterschiedlich verfahren. Ist der Reset jedoch aktiv, werden weder eine Tonausgabe noch eine Bremsausgabe durchgeführt, die entsprechenden Funktionen werden deaktiviert und zurückgesetzt. Für jede der Varianten wird eine spezifische Farbe festgelegt, die im nächsten Schritt ausgegeben wird.

Aktionen und Ausgaben
Die Aktion des ersten und größten Bereichs gibt beim Erkennen eines Objekts ein Warnsignal aus. Der zweite Bereich, etwas kleiner als der erste, führt zusätzlich eine kurze Aufmerksamkeitsbremsung durch. Der dritte und kleinste Bereich löst sowohl eine Warnung als auch eine Notbremsung bis zum Stillstand des Fahrzeugs aus. Damit diese Aktionen durchgeführt werden können, wurden drei Untersysteme entworfen.

Im ersten Untersystem wird das Handbremssignal über einen CAN-Sendeblock an das Fahrzeug gesendet, falls es sich um das reale Fahrzeug handelt. Andernfalls wird das Signal an das virtuelle Fahrzeug gesendet. Das zweite Untersystem ist für die Tonausgabe zuständig, die über den entsprechenden Audioausgang des Rechners erfolgt. Das dritte Untersystem steuert einen von Herrn Wiesenberg entwickelten Leuchtstreifen an, der, falls installiert, eine visuelle Ausgabe ermöglicht. Da der Leuchtstreifen derzeit nicht im BMW i3 verbaut ist, wurden zwei Simulink-Lampen implementiert. Diese leuchten grün, wenn das System aktiv ist, und grau, wenn es inaktiv ist. Wird im ersten Bereich ein Objekt erkannt, leuchtet das Warndreieck lila. Im zweiten Bereich leuchtet das Dreieck gelb und der Haken einmal rot. Im dritten Bereich leuchten beide Symbole durchgängig rot, bis das System zurückgesetzt wird. Das Zurücksetzen des Systems könnte zukünftig beispielsweise von der ausgewählten Fahrstufe abhängig sein.

Datenspeicher

Damit die Auslösungen nachvollzogen werden können, wurde erstmals ein Datenspeicher implementiert. Da der BMW i3 über keinen eigenen Datenspeicher verfügt, schreibt das Modell die Daten mit und speichert sie direkt in einer $.mat$-Datei ab. Dies garantiert, dass die Daten auch bei einem vorzeitigen Abbruch des Modells vorhanden sind. Wird das Modell gestoppt, wird diese Datei automatisch geöffnet und in einem Diagramm dargestellt.

Da das einfache Umschalten zwischen der Simulation und dem realen Fahrzeug sowie die automatische Abfrage nach vorhandenen Systemen hier schwierig ist, müssen die nicht genutzten Blöcke manuell auskommentiert werden. In Anlage E.7 befindet sich die vollständig ergänzte Modifikationsstruktur des Fahrzeugs.

5.4. Prüfkatalog

Für den Prüfkatalog wurde eine prototypische und erweiterbare Vorlage erstellt. Für diese musste eine Auswahl an Prüfszenarien festgesetzt werden. Da vorerst nur der Notbremsassistent vorhanden ist, welcher zudem hauptsächlich auf Fußgänger ausgelegt wurde, konnten die Tests darauf beschränkt werden. Für den Test und zur Überprüfung des Assistenten wurde auf EURO NCAP-Tests zurückgegriffen. Aus den auf der Website zu findenden Tests, welche in Abbildung 5.10 dargestellt sind, wurden die rot markierten ausgewählt.

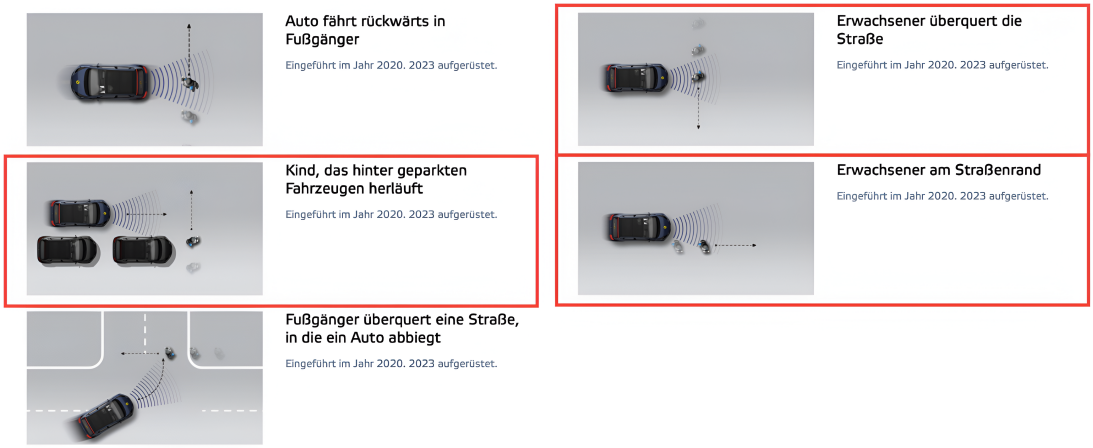


Abbildung 5.10.: Mögliche und davon ausgewählte EURO-NCAP Notbrems-Testszenarien [69]

Aus diesen Beispielen wurden die im Prüfkatalog in Anlage F gezeigten Prüfszenarien prototypisch abgeleitet. Eines dieser Szenarien wurde in der Auswertung 6.2 umgesetzt.

6. Auswertung

In diesem Kapitel erfolgt die Auswertung des Pfadverfolgers und der automatischen Notbremse.

6.1. Längs- und Querführung

6.1.1. Positionsbestimmmung

Um die Genauigkeit der Positionsbestimmung durch die Berechnung mithilfe der hinteren Raddrehzahlsensoren und dem Gierratensensor zu vergleichen, wurde auf dem Testfeld der in Abbildung 6.1 skizzierte Versuch durchgeführt.

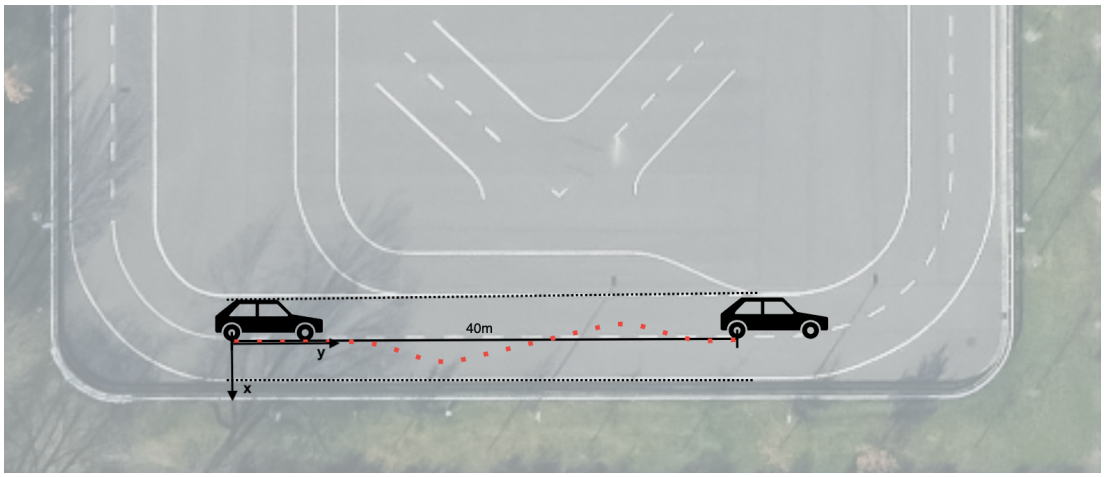


Abbildung 6.1.: Skizze zum Versuch der Positionsauswertung

Dabei soll das Fahrzeug auf dem hinteren Teil des Testfelds eine gerade Strecke zwischen zwei Punkten zurücklegen. Der Endpunkt liegt 40 m vom Startpunkt entfernt. Das Fahrzeug steht mit der Hinterachse am Ausgangspunkt und startet aus dem Stand. Es soll möglichst nah am Endpunkt zum Stehen kommen. Die gefahrene Strecke soll dabei sowohl in der Geschwindigkeit als auch vom Lenkwinkel variieren, um eine Vielzahl von möglichen Wegen zum gleichen Endpunkt abzubilden. Das Fahrzeug wird vorwärts bis zu einer Geschwindigkeit von 10 km/h und nur zwischen den zwei äußeren Markierungen bewegt. Für jeden Durchgang muss die Abweichung der realen zu der gewünschten Endposition festgehalten und in der Auswertung berücksichtigt werden. Die Aufzeichnung der Daten erfolgt mithilfe des in Abbildung 5.5 gezeigten MATLAB Simulink Modells. Bilder der Durchführung, die Messungen zu den Versuchen und die Skripte für die Auswertung sind in Anlage G.2 (Auswertung/Positionierung) enthalten.

Um die Wiederholbarkeit zu sichern wurden folgende Randbedingungen festgehalten:

- Asphalt: trocken
- Wetter: sonnig, kaum Wolken
- Außentemperatur: 28℃
- Reifenabrollumfang Hinterräder: 2,115m
- Spurbreite: 1,556m (aus Datenblatt)
- Bereifung: Bridgestone Ecopia EP500
- Reifendimensionen: vorn: 155/70R19 84Q, hinten: 175/60R19 84Q
- Luftdruck: vorn: 2,3bar, hinten: 2,8bar
- Reifendruck im Fahrzeugmenü zurückgesetzt
- Beladung: 1 Person (Fahrer, circa 66kg)
- Simulink Modell Simulationstempo: An (1)
- Abfragerate CAN-Block: 0,01s
- PI-Reglerparameter Lenkung: P: 0,02; I: 0,00025
- PI-Reglerparameter Fahrpedal: P: 0,06; I: 0,00015
- Lookahead Distance: 3m

Folgende Ergebnisse wurden erzielt:

	$\Delta x_{Endposition}$ [m]	$\Delta y_{Endposition}$ [m]
Messung 1	0,005	0,025
Messung 2	0,005	0,030
Messung 3	0,020	0,020
Messung 4	0,005	0,030
Messung 5	0,005	0,000
Messung 6	0,010	0,005

Tabelle 6.1.: Gemessene Abweichung zum Referenzpunkt Versuch 1

	$\Delta x_{Gierratensensor}$ [m]	$\Delta x_{Gierratenberechnung}$ [m]
Messung 1	-0,448	-0,814
Messung 2	0,063	-0,382
Messung 3	-0,183	-0,464
Messung 4	-0,204	-0,525
Messung 5	-0,712	-1,234
Messung 6	-0,406	-1,142
Δx [m] gemittelt	-0,315	-0,760

Tabelle 6.2.: Berechnete Positionsabweichung in x-Richtung Versuch 1

	$\Delta y_{Gierratensensor}$ [m]	$\Delta y_{Gierratenberechnung}$ [m]
Messung 1	2,122	2,063
Messung 2	2,326	2,274
Messung 3	2,275	2,231
Messung 4	1,919	1,915
Messung 5	1,948	1,904
Messung 6	2,067	2,018
Δy [m] gemittelt	2,110	2,068

Tabelle 6.3.: Berechnete Positionsabweichung in y-Richtung Versuch 1

Wie in Tabelle 6.1 dargestellt, konnte der Endpunkt bei jeder Messung mit hoher Genauigkeit angefahren werden. Die Abweichungen lagen im niedrigen, einstelligen Zentimeterbereich, was von Vorteil ist, da mögliche Fehler durch ungenaues Anfahren vernachlässigbar klein sind.

Tabelle 6.2 zeigt die berechneten Abweichungen der beiden Ansätze zum Referenzpunkt in x-Richtung, unter Berücksichtigung der realen Endposition. Die gemittelten Abweichungen sowie die Extremwerte verdeutlichen, dass die Positionierung in x-Richtung durch den Gierratensensor signifikant präziser ist als durch die Raddrehzahlsensoren. Alle Messungen, die mit dem Gierratensensor durchgeführt wurden, wiesen eine höhere Genauigkeit auf.

Abbildung 6.2 zeigt beispielhaft eine der, zur Auswertung, generierten Übersichten. Diese wurde vom MATLAB Skript $Vergleich\_Positionen.m$ erzeugt und zeigt die Messung 6.

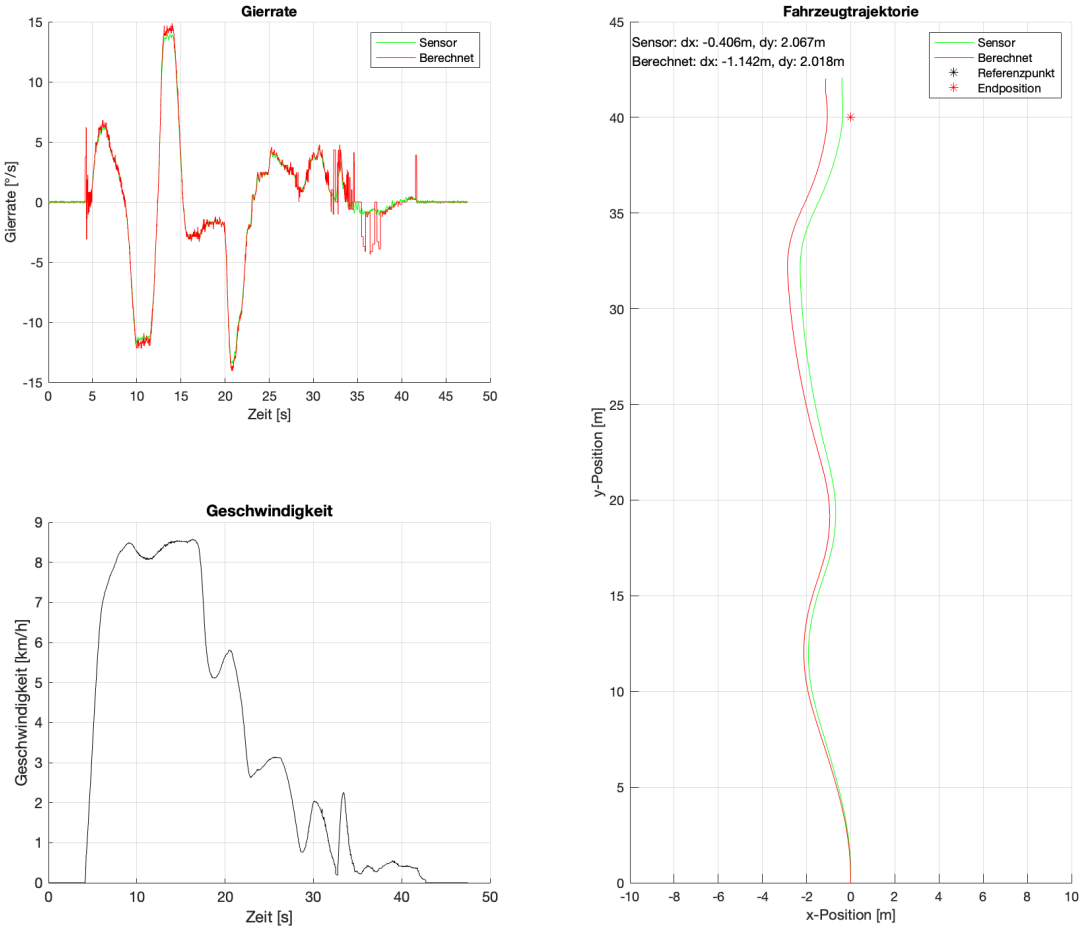


Abbildung 6.2.: Übersicht der Fahrtauswertung von Messung 6

In Abbildung 6.2 ist deutlich zu erkennen, dass die berechnete Gierrate anhand der Raddrehzahlen insbesondere bei niedrigen Geschwindigkeiten in Kombination mit schnellen Geschwindigkeitsänderungen erhebliche Sprünge aufweist. Zusätzlich tritt an den Wendepunkten der Kurvenfahrt ein Rauschen auf, das in allen Messungen zu beobachten ist. Im Gegensatz dazu zeigt die Gierrate des Gierratensensors bei sämtlichen Messungen und Fahrzuständen keine derartigen Sprünge, sondern lediglich ein konstant geringes Rauschen. Diese Unterschiede spiegeln sich auch in der Fahrzeugtrajektorie wider. Während das konstante Rauschen des Gierratensensors anscheinend gut kompensiert wird, führen die Sprünge bei der Berechnung durch die Raddrehzahlen zu einer signifikanten Abweichung vom Kurs.

In Abbildung 6.3 wird ein Ausschnitt der Raddrehzahlen aus Messung 6 mithilfe des Skripts $Vergleich\_Raddrehzahlen.m$ detaillierter betrachtet, um die beobachteten Sprünge besser nachvollziehen zu können.

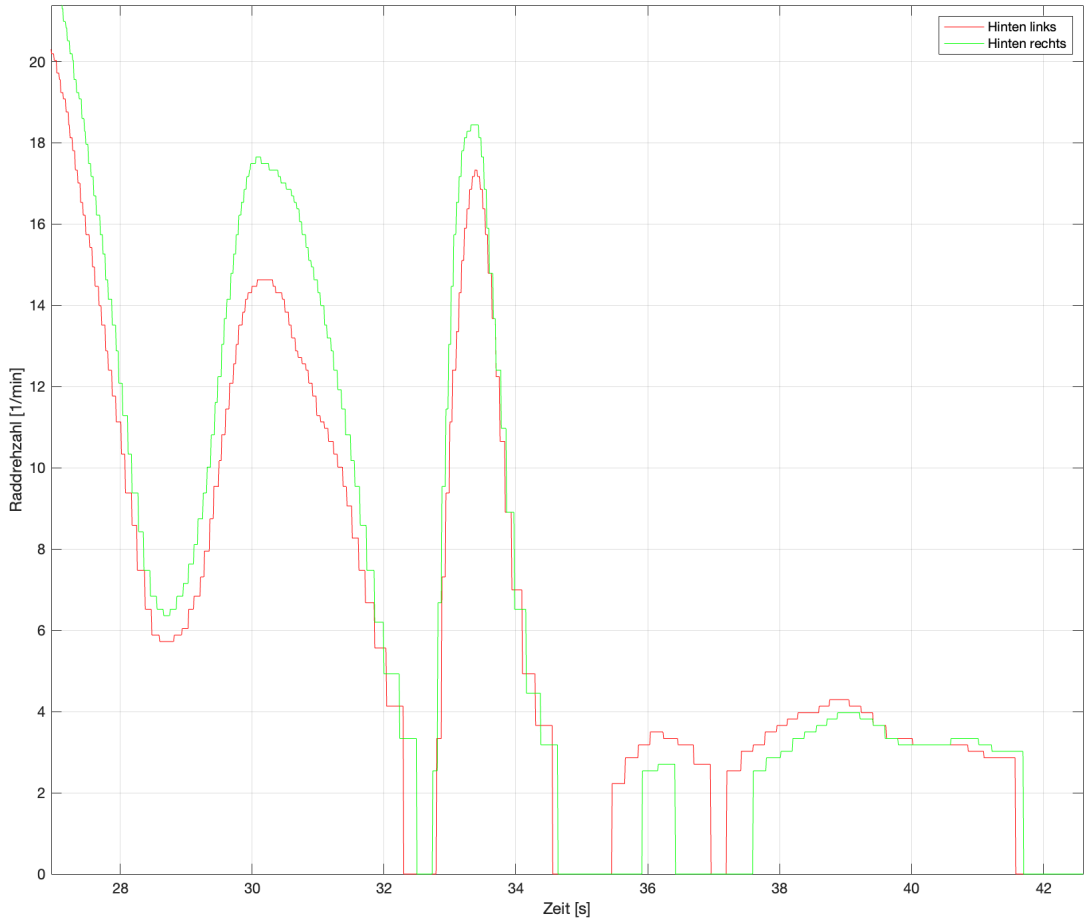


Abbildung 6.3.: Auszug untere Raddrehzahlen Messung 6

Dabei lässt sich beobachten, dass diese Sprünge auf die Differenzbildung bei der Berechnung der Gierrate, Gleichung 5.1, zurückzuführen sind, da die Raddrehzahlen einen gewissen Versatz aufweisen. Dieser Versatz kann bei schneller Beschleunigung während der Geradeausfahrt auftreten, wenn beispielsweise ein Rad mehr Schlupf hat als ein anderes oder sich die Räder bei Kurvenfahrt unterschiedlich schnell bewegen. Beide Szenarien entsprechen den beobachteten Sprüngen und dem Rauschen an den Wendepunkten. Zudem scheint die Abtastrate des Signals mit zunehmender Raddrehzahl zu steigen, da die des Simulink-Modells konstant ist. Dies kann aufgrund des Messverfahrens des Sensors normal sein. Bei Raddrehzahlen unter etwa 5 Umdrehungen pro Minute sind die Sprünge wesentlich ausgeprägter als bei Drehzahlen über etwa 10 Umdrehungen pro Minute. Dies deutet darauf hin, dass die Werte länger benötigen, um aktualisiert zu werden. Dies wird auch in Abbildung 6.4 deutlich, die den oberen Bereich der Raddrehzahlen aus Messung 6 zum Vergleich zeigt.

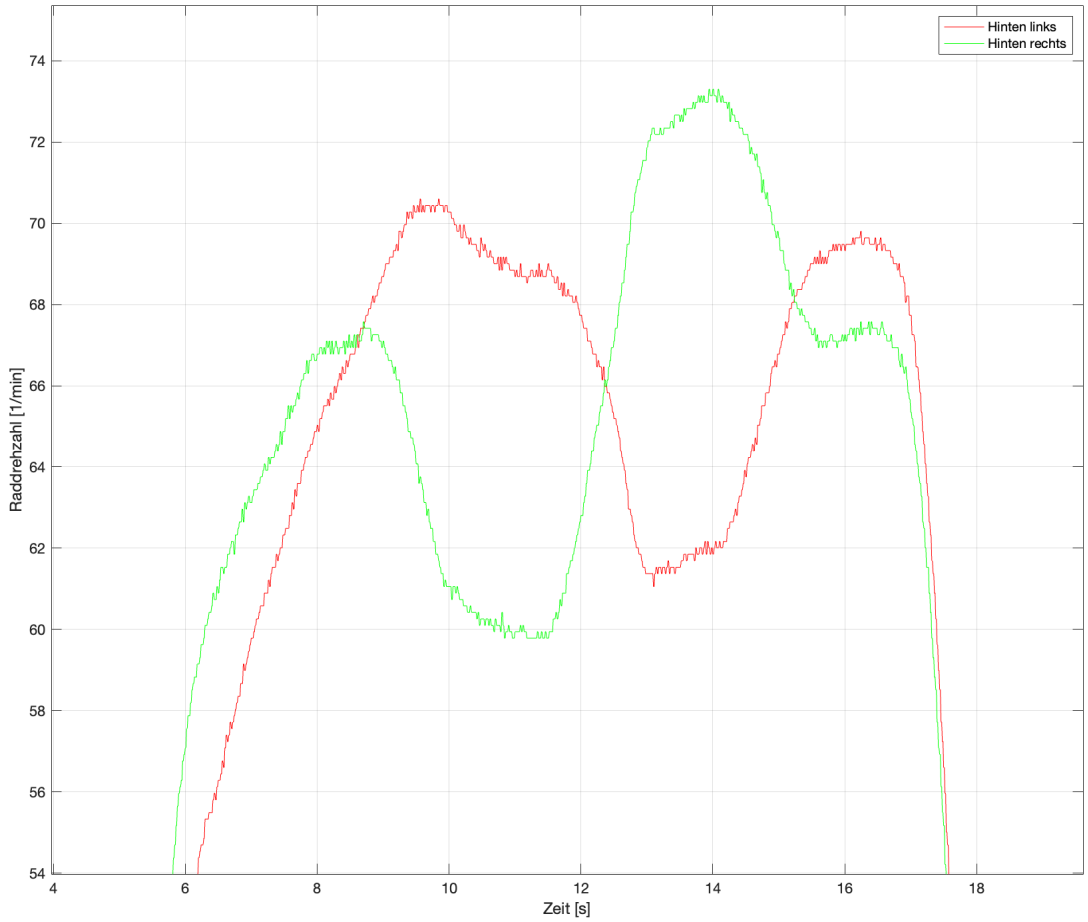


Abbildung 6.4.: Auszug obere Raddrehzahlen Messung 6

Gut zu erkennen ist, dass das Signal, im Gegensatz zu den niedrigen Drehzahlen in Abbildung 6.3, bei höheren Raddrehzahlen schmalere Stufen aufweist und somit häufiger aktualisiert wird. Dies führt zu einem geringeren Rauschen. Im unteren Raddrehzahlbereich ist dieses Verhalten nicht gegeben. Um derartige Fehler zu minimieren und die Sprünge auszugleichen, könnte eine Filterung des Signals in Betracht gezogen werden. Dies könnte auch für das Signal des Gierratensensors sinnvoll sein, da es ein leichtes Rauschen aufweist. Da für den Referenzpunkt kein vollständig exakter Wert ermittelt werden konnte – aufgrund fehlender Messreferenzen und der leichten Neigung des Testfelds – kann nicht abschließend geklärt werden, ob die geringe Abweichung in x-Richtung auf einen Messfehler zurückzuführen ist. Da der Gierratensensor jedoch ebenfalls durch die Unebenheiten des Testfelds beeinflusst werden könnte, wird der Großteil der Abweichung als Messfehler angesehen.

Tabelle 6.3 zeigt die berechneten Abweichungen zum Referenzpunkt unter Berücksichtigung der realen Endposition in y-Richtung. Dabei wird deutlich, dass, im Gegensatz zur x-Richtung, jede der Berechnungen mithilfe der Raddrehzahlen ein genaueres Ergebnis erzielt. Dies spiegelt sich auch in den Mittelwerten wider. Zwar ist die Berechnung auf Grundlage der Raddrehzahlen hier besser, jedoch

ist der Unterschied nicht so signifikant wie bei den Abweichungen in x-Richtung. Die Werte weichen stark von der real angefahrenen Position ab. Während sich die Abweichungen in x-Richtung gut erklären lassen, deuten gerade die großen Abweichungen in y-Richtung auf eine fehlerhaft ausgelesene Geschwindigkeit hin.

Nach mehrfacher Überprüfung der MATLAB-Skripte und Rücksprache mit dem Betreuer wurde beschlossen, den Faktor zur Berechnung der Geschwindigkeit in der CAN-Datenbank iterativ anzupassen. Diese Anpassung war notwendig, da die CAN-Datenbank von Hochschulmitarbeitern erstellt wurde und nicht direkt vom Fahrzeughersteller stammte, wodurch Fehler nicht auszuschließen waren. Dieser Wert wurde von 0,0156 auf 0,0153 reduziert. Anschließend konnte der Versuch unter identischen Randbedingungen erneut durchgeführt. Dabei wurde die Versuchsreihe in zwei Teile unterteilt, um genauere Informationen zu erhalten. Die Messungen 11 bis 13 umfassen Geradeausfahrten, während die Messungen 14 bis 19 beliebige Kurvenfahrten beinhalten, jeweils drei nach links und drei nach rechts startend.
Folgende Ergebnisse wurden erzielt:

	$\Delta x_{Endposition}$ [m]	$\Delta y_{Endposition}$ [m]
Messung 11	0,010	0,015
Messung 12	0,010	-0,010
Messung 13	0,020	0,010
Messung 14	0,010	0,025
Messung 15	0,010	0,000
Messung 16	0,005	0,005
Messung 17	0,020	0,000
Messung 18	0,000	0,000
Messung 19	0,000	0,025

Tabelle 6.4.: Gemessene Abweichung zum Referenzpunkt Versuch 2

	$\Delta x_{Gierratensensor}$ [m]	$\Delta x_{Gierratenberechnung}$ [m]
Messung 11	-0,269	-0,631
Messung 12	-0,366	-0,457
Messung 13	-0,345	-0,753
Δx [m] gemittelt	-0,327	-0,614

Tabelle 6.5.: Berechnete Positionsabweichung in x-Richtung Versuch 2

	$\Delta y_{Gierratensensor}$ [m]	$\Delta y_{Gierratenberechnung}$ [m]
Messung 11	-0,031	-0,035
Messung 12	0,048	0,046
Messung 13	-0,004	-0,009
Δx [m] gemittelt	0,004	0,001

Tabelle 6.6.: Berechnete Positionsabweichung in y-Richtung Versuch 2

	$\Delta x_{Gierratensensor}$ [m]	$\Delta x_{Gierratenberechnung}$ [m]
Messung 14	-0,187	-0,118
Messung 15	-0,358	-0,536
Messung 16	-0,083	-0,579
Messung 17	-0,421	-0,262
Messung 18	-0,069	-0,059
Messung 19	-0,250	-0,476
Δx [m] gemittelt	-0,228	-0,338

Tabelle 6.7.: Berechnete Positionsabweichung in x-Richtung Versuch 2

	$\Delta y_{Gierratensensor}$ [m]	$\Delta y_{Gierratenberechnung}$ [m]
Messung 14	0,335	0,270
Messung 15	0,289	0,237
Messung 16	0,129	0,078
Messung 17	0,372	0,308
Messung 18	0,120	0,083
Messung 19	0,270	0,213
Δx [m] gemittelt	0,253	0,198

Tabelle 6.8.: Berechnete Positionsabweichung in y-Richtung Versuch 2

Auch bei der zweiten Durchführung des Versuchs erkennt man in Tabelle 6.4, dass die Referenzposition sehr genau angefahren wurde. Somit treten keine signifikanten Unterschiede im Vergleich zum ersten Versuch auf, die die Messungen beeinflussen könnten.

Bei den Messungen der Geradeausfahrt, wie in Tabelle 6.5 dargestellt, sind zunächst nur geringe Verbesserungen der Abweichungen in x-Richtung erkennbar. Anders verhält es sich bei der Kurvenfahrt. In Tabelle 6.7 zeigen beide Berechnungsmethoden deutliche Verbesserungen, wobei die Abweichungen bei der Berechnung mithilfe der Raddrehzahlsensoren stark reduziert wurden. Unter Berücksichtigung, dass bei der ersten Durchführung sowohl Fahrten mit unterschiedlichen

Kurvenradien durchgeführt wurden, kann der Mittelwert aller Messungen aus der zweiten Versuchsreihe gebildet werden. Daraus ergibt sich in x-Richtung für die Gierrate vom Gierratensensor eine durchschnittliche Abweichung von -0,278m und für die berechnete Gierrate aus den Raddrehzahlen eine durchschnittliche Abweichung von -0,476m. Vergleicht man diese Werte mit denen aus der ersten Durchführung, so erkennt man, dass beide Berechnungsmethoden verbessert wurden. Insbesondere die Berechnung mithilfe der Raddrehzahlsensoren zeigt eine signifikante Verbesserung.

Bei den Abweichungen beider Berechnungen in y-Richtung zeigen sich sowohl in Tabelle 6.6 als auch in Tabelle 6.8 im Vergleich zu Tabelle 6.3 signifikante Verbesserungen. Auch die gemittelten Werte über alle Messungen der zweiten Versuchsdurchführung sind besser. Für die Berechnung der y-Richtung mit der Gierrate vom Gierratensensor ergibt sich eine durchschnittliche Abweichung von 0,129m, und für die berechnete Gierrate aus den Raddrehzahlen beträgt die durchschnittliche Abweichung 0,100m. Durch die Anpassung der CAN-Datenbank haben sich alle Werte deutlich verbessert und befinden sich in einem akzeptablen Bereich. Es ist jedoch zu erkennen, dass die Werte für die Geradeausfahrt wesentlich genauer sind als die bei Kurvenfahrt. Dies könnte darauf hindeuten, dass die Geschwindigkeit des Fahrzeugs über die Raddrehzahlsensoren berechnet wird und von deren Fehlern, beispielsweise den Sprüngen, beeinflusst wird.

Durch die Anpassung der CAN-Datenbank und den damit erzielten Ergebnissen wäre denkbar, dass auch die Abweichungen in x-Richtung noch weiter reduziert werden könnten, indem auch die Faktoren der jeweiligen CAN-Botschaften für die Raddrehzahlsensoren und den Gierratensensor weiter optimiert werden. Eine genauere Betrachtung dieser Optimierungen soll im Rahmen dieser Arbeit nicht erfolgen, da die Werte aufgrund der Sensoreigenfehler plausibel erscheinen und kleine Abweichungen auch auf die Testfeldeigenschaften zurückzuführen sind.

Abschließend wird in diesem Fall jedoch die Verwendung der Gierrate vom Gierratensensor bevorzugt. Abbildung 6.2 zeigt deutlich, dass die Abweichung in x-Richtung die entscheidende Rolle bei der Bewertung der aktuellen Position und Orientierung an der Endposition spielt. Die Positionierung in y-Richtung ist in diesem Fall weniger relevant, sollte jedoch zumindest einmal genauer betrachtet worden sein, da diese Arbeit auf eine prototypische Umsetzung hinarbeitet, die im späteren Verlauf einen Pfadverfolger implementiert, der auf der Vergleichbarkeit und nicht der Genauigkeit der Route beruht. Zusätzlich ist die Berechnung der Gierrate mit den Raddrehzahlsensoren aufgrund des sich durch äußere Umstände ändernden Abrollumfangs sehr fehleranfällig.

6.1.2. Zuverlässigkeitstest Pfadverfolger

Um die Zuverlässigkeit des Pfadverfolgers zu bewerten, wurde vor Versuchsbeginn eine reale Referenzfahrt aufgenommen. Die Aufzeichnung erfolgte nach dem gleichen Prinzip wie die Testfahrt in Unterabschnitt 5.2.2. Die Randbedingungen waren identisch zu denen im Unterabschnitt 6.1.1 beschrieben. Als Messpunkte wurden die Durchfahrten durch die Tore der Garage und des Testfeldes festgelegt. Zwischen den Pfosten der Tore wurde eine gerade Linie mit Markierschaum gezogen. Vor Versuchsbeginn wurden zuerst der Lenk- und der Fahrpedal-Aktuator aktiviert, dann das Modell gestartet und zum Schluss der Notaus gezogen. Nach der Fahrt wurde bei den Toreinfahrten und Torausfahrten die Entfernung der Außenkante des Reifenabdrucks des hinteren linken Rades, der durch die Durchfahrt durch den Markierschaum entstand, parallel zum jeweils in Fahrtrichtung linken Pfosten der Tore gemessen. Die aufgenommene Referenzfahrt ist in Abbildung 6.5 dargestellt. Bilder der Durchführung, die Messungen zu den Versuchen und die Skripte für die Auswertung sind in Anlage G.2 (Auswertung/Pfadverfolger) enthalten.

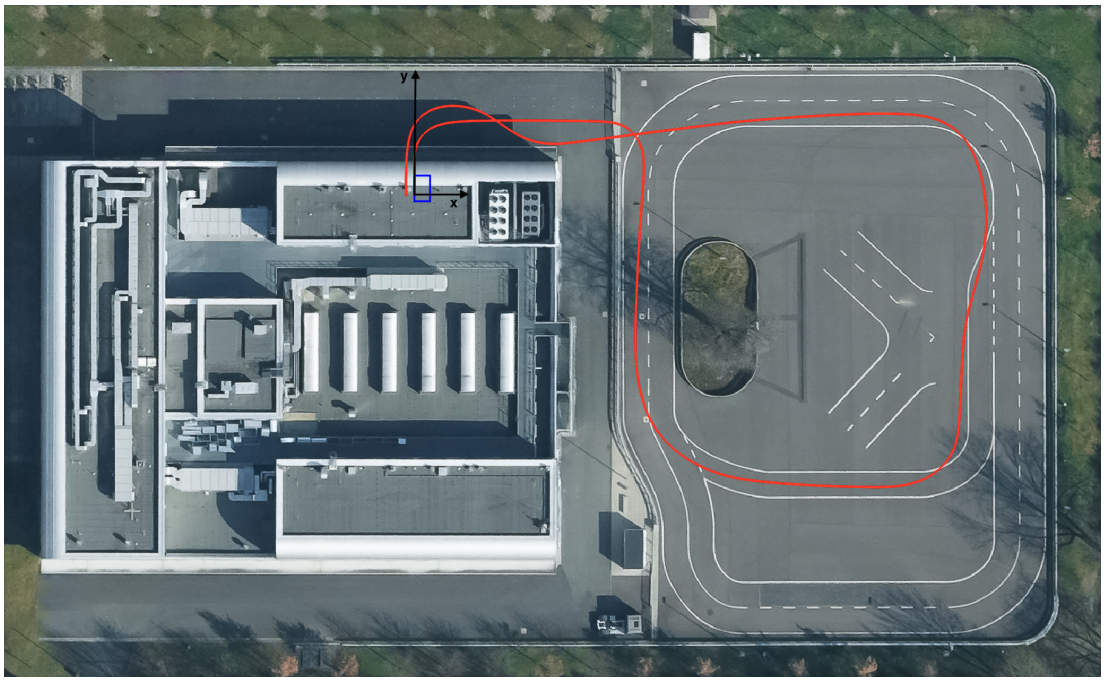


Abbildung 6.5.: Aufgezeichnete Referenzfahrt

Die erzielten Ergebnisse werden im Folgenden dargestellt.

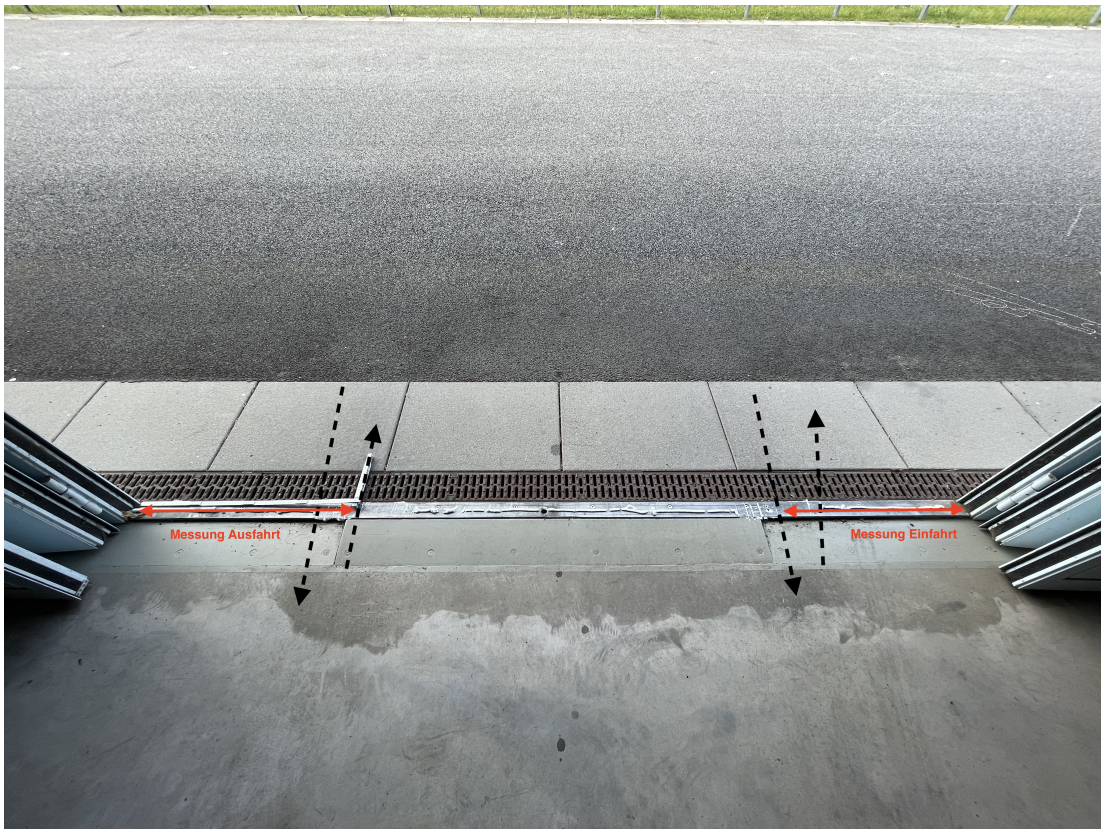


Abbildung 6.6.: Messung der Abstände am Garagentor

	Abstand Ausfahrt [m]	Abstand Einfahrt [m]
Referenzfahrt	0,81	0,67
Messung 1	0,95	0,58
Messung 2	0,97	0,63
Messung 3	0,97	0,60
Messung 4	0,96	0,84
Messung 5	0,96	0,60
Messungen gemittelt	0,96	0,65
Δ zu Referenz	0,15	0,02
Δ Messungen zu Referenz gemittelt	0,15	0,09

Tabelle 6.9.: Gemessene Abstände am Garagentor

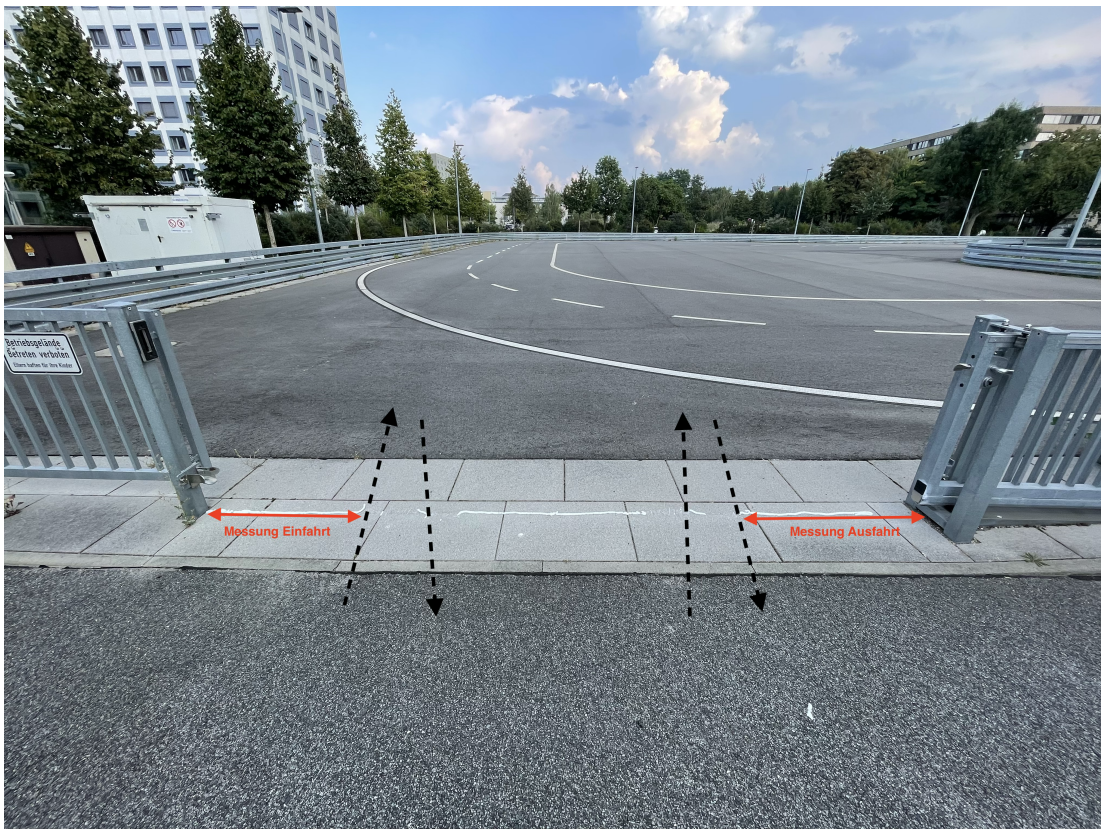


Abbildung 6.7.: Messung der Abstände am Testfeldtor

	Abstand Einfahrt [m]	Abstand Ausfahrt [m]
Referenzfahrt	0,92	1,02
Messung 1	1,25	1,24
Messung 2	1,13	1,31
Messung 3	1,25	1,27
Messung 4	1,10	0,94
Messung 5	1,24	1,26
Messungen gemittelt	1,19	1,20
Δ zu Referenz	0,27	0,18
Δ Messungen zu Referenz gemittelt	0,27	0,22

Tabelle 6.10.: Gemessene Abstände am Testfeldtor

	$x_{Endposition}$ [m]	$y_{Endposition}$ [m]
Referenzfahrt	1,66	2,88
Messung 1	1,75	3,01
Messung 2	1,67	3,06
Messung 3	1,72	3,58
Messung 4	1,56	2,43
Messung 5	1,82	2,74
Messungen gemittelt	1,70	2,96
Δ zu Referenz	0,04	0,08
Δ Messungen zu Referenz gemittelt	0,08	0,32

Tabelle 6.11.: Gemessene Endpositionen

Betrachtet man die gemittelten Abweichungen der Messfahrten zur Referenzfahrt aus Tabelle 6.9, so wird deutlich, dass die Ausfahrt aus dem Garagentor wesentlich größere Abweichungen aufweist als die Einfahrt in die Garage. Ein ähnliches Verhalten lässt sich bei der Durchfahrt des Testfeldtores beobachten, wie in Tabelle 6.10 dargestellt. Auf den ersten Blick scheint es, als würden die Abweichungen gegen Ende der Fahrt geringer werden. Betrachtet man jedoch Abbildung 6.8, wird ersichtlich, dass dieses Verhalten auf den Pure Pursuit Regler zurückzuführen ist.

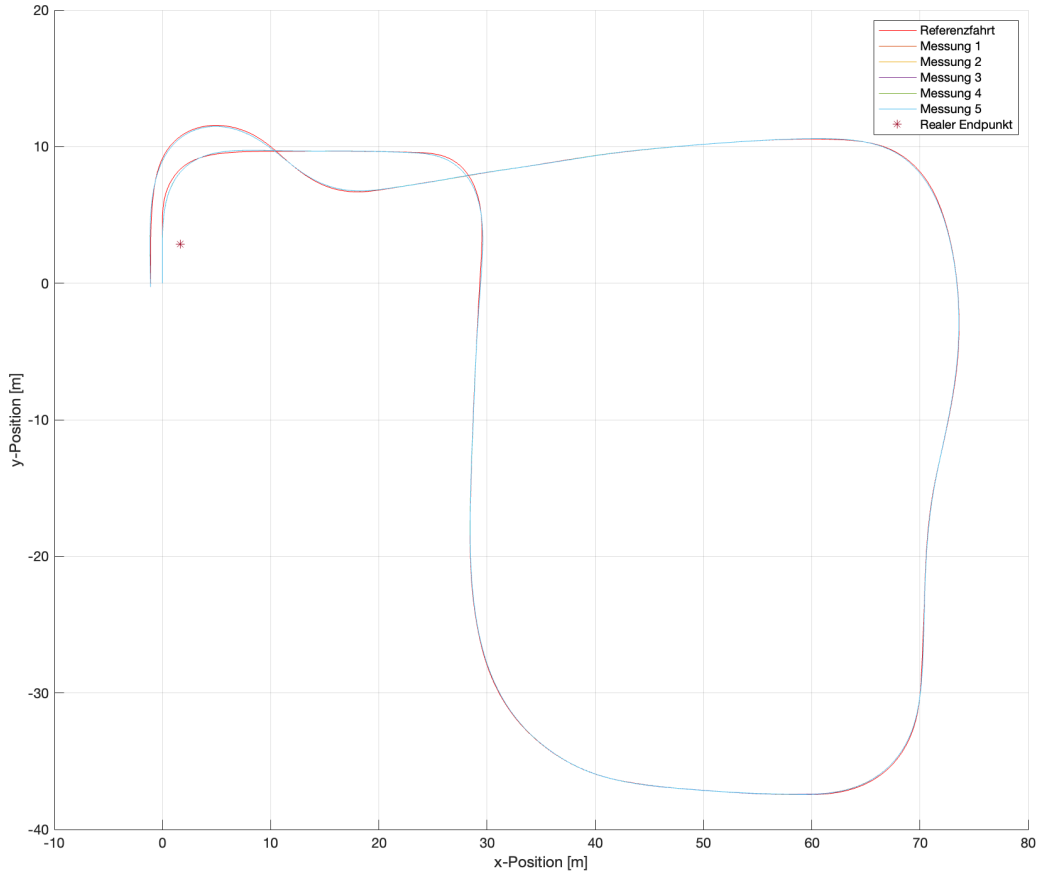


Abbildung 6.8.: Aufgezeichnete (annähernd deckungsgleiche) Messfahrten mit dem Pfadverfolger

Die geringer werdenden Abweichungen können darauf zurückgeführt werden, dass bei der Ausfahrt aus dem Garagentor sowie der Einfahrt in das Testfeldtor die Messpunkte vor Kurven liegen, die durch den Regler eher angefahren werden. Dies führt dazu, dass das hintere Rad an diesen Messpunkten einen größeren Versatz aufweist, was in Abbildung 6.9 deutlich wird.

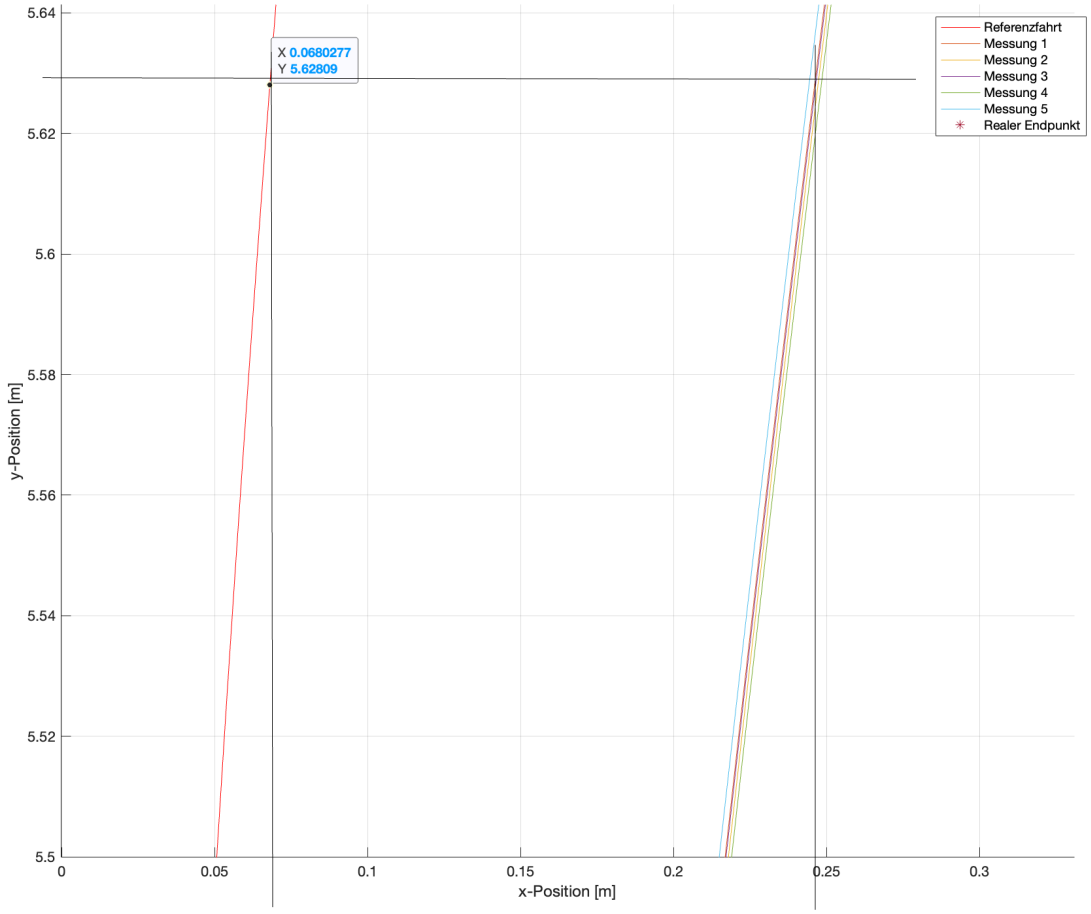


Abbildung 6.9.: Gemessene Torabstände

Zu erkennen ist, dass bei der Torausfahrt, die sich bei $y = 5,63m$ befindet, die Referenzfahrt eine Abweichung von etwa $x = 0,07m$ aufweist. Die Positionen der Messfahrten liegen hingegen bei etwa $x = 0,24m$. Dies entspricht einer Differenz von 17cm, was ungefähr der gemittelten Differenz zur Referenzfahrt aus Tabelle 6.9 bei der Garagenausfahrt entspricht. Somit bleibt nach Abzug des typischen Verhaltens des Reglers lediglich eine Abweichung von 2cm bestehen.

Für die Einfahrt auf das Testfeld liegt diese Abweichung bereits bei 11cm. Bei der Ausfahrt aus dem Testfeld fährt das Fahrzeug schräg, aber gerade durch das Tor, was zur Folge hat, dass die Abweichungen durch das Reglerverhalten sehr gering sind. Trotzdem erhöhte sich die Abweichung auf 21cm. Bei der Einfahrt in die Garage wurde die Messung erst Richtung Kurvenausgang durchgeführt, was zur Folge hatte, dass der Messpunkt kurz nach dem Kreuzen der Kurven der Referenzfahrt und der Messfahrt lag. Dies begünstigte, dass die Abweichung auf 2cm sank.

Weiterhin ist zu beobachten, dass die Messungen in Abbildung 6.9 untereinander nur geringe Abweichungen aufweisen. Dies lässt darauf schließen, dass der Regler zuverlässig funktioniert. Betrachtet man Abbildung 6.10 fällt auf, dass die realen Messungen an den Messpunkten jedoch größere Schwankungen aufweisen.

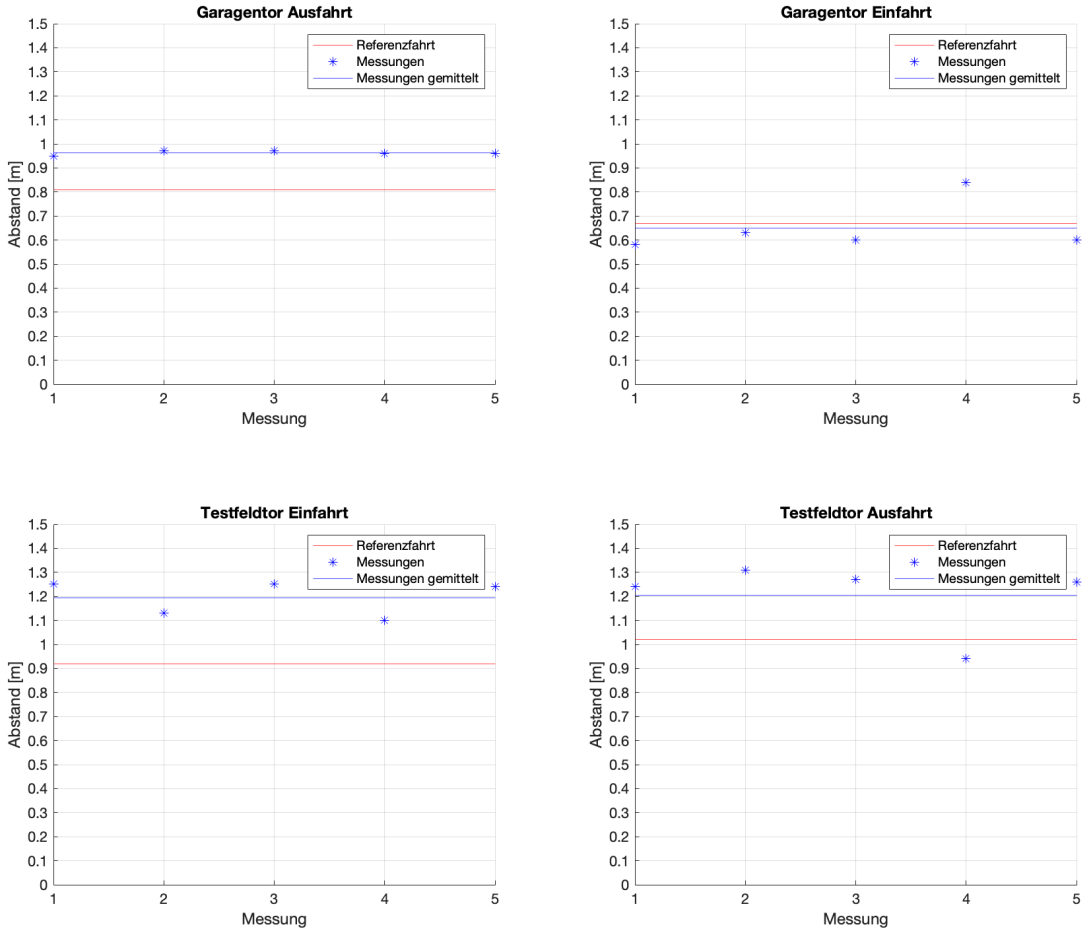


Abbildung 6.10.: Gemessene Torabstände

Obwohl die Mittelwerte der Messungen sich insgesamt sehr nah an der Referenzfahrt orientieren, erscheinen sie etwas widersprüchlich zu den Ergebnissen in Abbildung 6.8.

Dieser Widerspruch kann mit Abbildung 6.11 begründet werden. Darin ist zu erkennen, dass die Geschwindigkeit und die Gierrate bei allen Messfahrten nahezu identisch verlaufen, jedoch an wenigen Stellen größere Unterschiede aufweisen, die die Messungen beeinflussen.

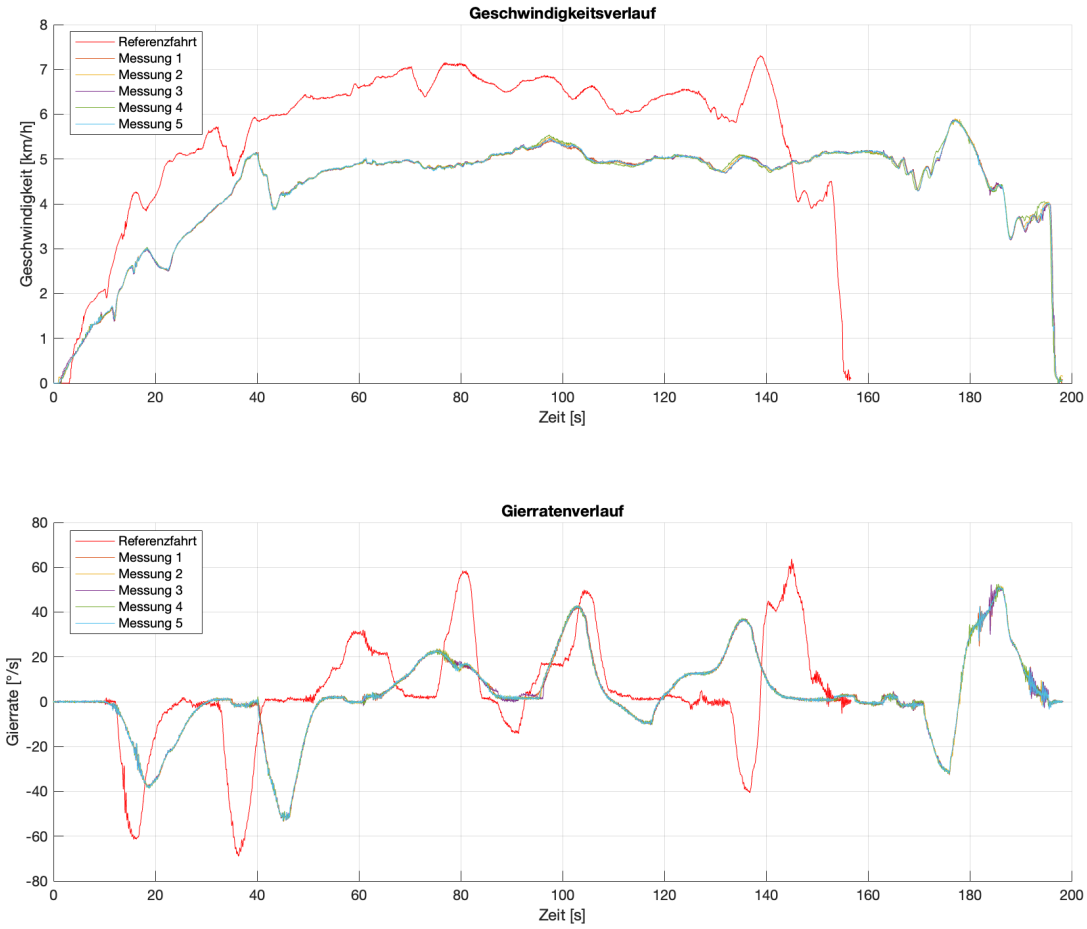


Abbildung 6.11.: Aufgezeichnete Gierraten des Pfadverfolger

Auffällig, aber begründbar, ist auch der zeitliche Versatz der Endpunkte sowie die geringere Geschwindigkeit und Gierrate. Die verringerte Geschwindigkeit und der zeitliche Versatz lassen sich darauf zurückzuführen, dass das Modell auf 5 km/h begrenzt wurde. Diese Begrenzung führt zusätzlich zu länger anhaltenden, aber niedrigeren Gierraten, da das Fahrzeug mehr Zeit hat, seinen Kurs anzupassen. Kurz vor Ende der Messung steigt die Geschwindigkeit erneut an, was auf die unebene Fahrbahn vor der Garageneinfahrt zurückzuführen ist. Da das Fahrzeug nicht bremsen kann, rollt es die Schräge hinab, bevor es nach der Kurve wieder die Steigung zur Garageneinfahrt überwinden muss.

Nicht nur in den Abstandsmessungen zeigt sich die Abweichung von den aufgezeichneten zu den realen Messfahrten, sondern auch in Tabelle 6.11. Abbildung 6.12 visualisiert diese Tabelle und zeigt die Abweichungen der einzelnen Messungen der Endposition des hinteren linken Rades zu dessen realer Endposition. Um die Abweichungen zu verdeutlichen, wurde der Koordinatenursprung auf die Referenzendposition gelegt.

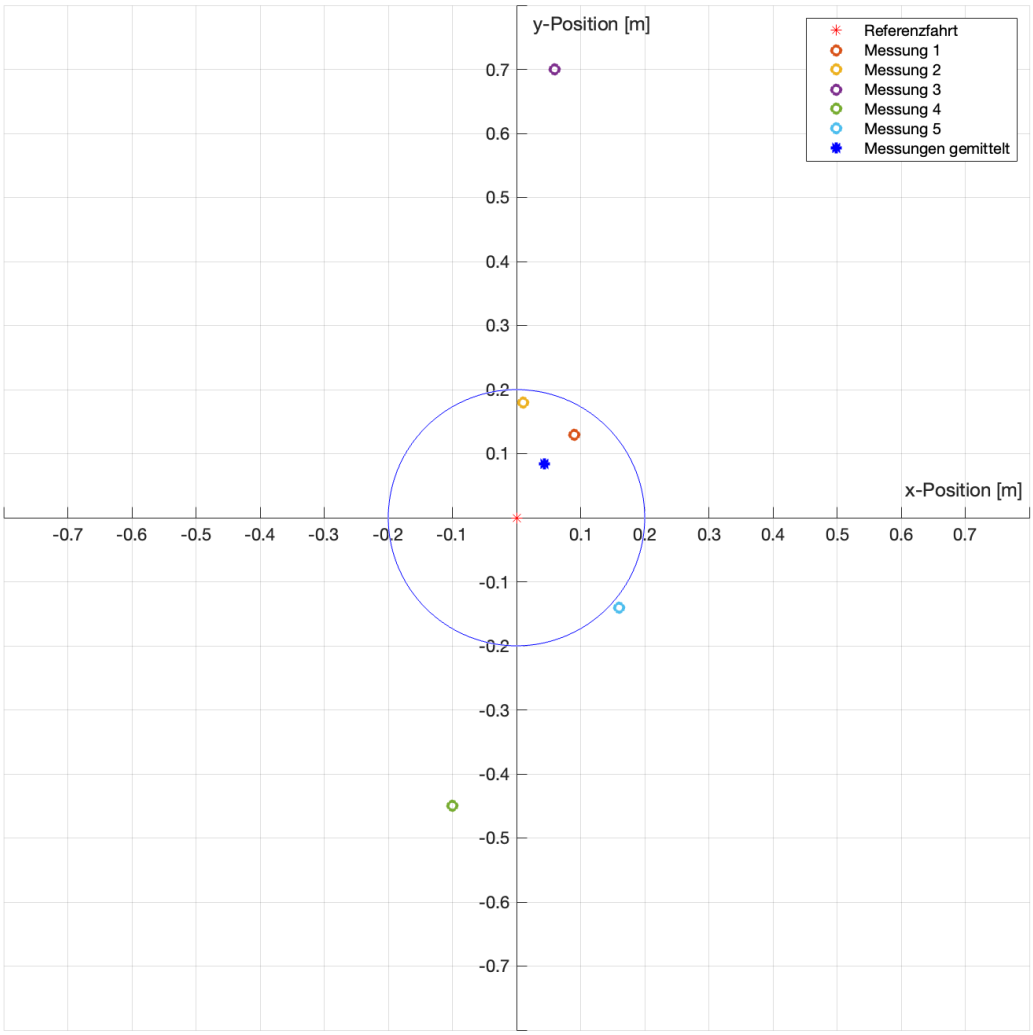


Abbildung 6.12.: Abweichungen der Endpositionen nach der Pfadverfolgung

Zu sehen ist, dass die Messungen im Mittel sehr nah an der Referenzendposition liegen, jedoch die Abweichungen teils erheblich variieren. Die starken Abweichungen in der y-Position lassen sich durch den Algorithmus des Pfadverfolgers erklären, der das Fahrzeug zunächst etwas ausrollen lässt, bevor das Handbremssignal gesendet wird. Die Abweichungen in x-Richtung entsprechen den zuvor Betrachteten und sind somit nachvollziehbar. Hervorzuheben ist, dass das Fahrzeug trotz der engen Tordurchfahrten jede Messung bis zum Stillstand in der Garage absolvieren konnte und die Endpositionen überwiegend innerhalb oder sehr knapp am Rand des blauen Kreises mit einem Radius von 20 cm lagen.

Da das Pfadverfolgermodell die Referenzstrecke sehr präzise nachverfolgen kann, deutet dies darauf hin, dass der Großteil der entstandenen Fehler von den Sensoren stammt, wie bereits im Unterabschnitt 5.2.1 thematisiert wurde. Insgesamt arbeitet das Modell sehr präzise und ist in Kombination mit der Sensorik in der Lage, das Fahrzeug bei diesen Bedingungen sicher über das Testfeld zu navigieren und in der Garage abzustellen.

6.2. Notbremstest nach Prüfkatalog

Zur Auswertung der automatischen Notbremse wird der Versuch eins aus dem in Anlage F gezeigten Prüfkatalog umgesetzt. Danach wird der gleiche Versuchsaufbau in der Simulation wiederholt, um die Ergebnisse zu vergleichen. Versuch eins bietet mehrere Vorteile: Er ermöglicht die Validierung der Funktionalität der Notbremsfunktion in einer kritischen Situation, die Beurteilung der Bremsleistung der Handbremse und lässt sich zudem einfach in der Simulation nachbilden. Für den Versuch wurde eine erwachsene Schaufensterpuppe als Dummy verwendet.

Für den Versuch wurde das Fahrzeug mit der Front bündig zum oberen Tor des Testfelds platziert. Der Dummy musste an einer in der Simulation genau bestimmbaren Stelle positioniert werden, um eine möglichst exakte Nachbildung zu ermöglichen. Zu diesem Zweck wurde der Dummy auf dem Testfeld in einem Abstand von 40 Metern zum Tor und frontal in Fahrtrichtung des Fahrzeugs auf der Fahrbahnmarkierung positioniert. Bilder des Aufbaus und die Messergebnisse sind in Anlage G.2 (Auswertung/AEB) enthalten.

6.2.1. Realversuch

Für den realen Versuch wurden folgende Randbedingungen festgehalten:

- Asphalt: trocken
- Wetter: Sonnig/Bewölkt
- Außentemperatur: 19℃
- Bereifung: Bridgestone Ecopia EP500
- Reifendimensionen: vorn: 155/70R19 84Q, hinten: 175/60R19 84Q
- Luftdruck: vorn: 2,3bar, hinten: 2,8bar
- Beladung: 1 Person (Fahrer, circa 66kg)
- Simulink Modell Simulationstempo: An (1.7)
- Abfragerate CAN-Block: 0,01s
- Blockabfrageraten: 0,2s

Im realen Versuch wurde der geringste Abstand zwischen der Front des stehenden Fahrzeugs nach der Notbremsung und der Mitte des Dummys bestimmt. Dabei wurden folgenden Ergebnisse erzielt:

	15 km/h	20 km/h	30 km/h
Abstand Messung 1 [m]	4,75	5,60	5,32
Abstand Messung 2 [m]	4,51	5,23	3,44
Abstand Messung 3 [m]	5,23	5,09	1,87
Δ Messungen gemittelt [m]	4,83	5,31	3,54

Tabelle 6.12.: Real gemessene Abstände des Notbremsversuchs

In Tabelle 6.12 ist zu erkennen, dass das Fahrzeug bei jeder Messung trotz der geringen Abfragerate von 5 LiDAR-Bildern pro Sekunde und nur mithilfe der elektronisch gesteuerten Handbremse rechtzeitig zum Stehen gekommen ist. Die einzelnen Messungen bei 15 km/h und 20 km/h weichen untereinander nur geringfügig voneinander ab. Auffällig ist jedoch, dass bei der Messreihe mit 30 km/h die Messungen sehr stark voneinander abweichen. Eine Fehlerquelle dafür könnte sein, dass die Geschwindigkeit durch den Fahrer nicht exakt konstant gehalten werden konnte und beim Anfahrvorgang jeder Messung unterschiedlich stark beschleunigt wurde. Dieser Fehler ist jedoch bei jeder der Messungen enthalten. Die aufgezeichneten Daten aus dem Datenspeicher-Block zeigen zusätzliche Auffälligkeiten. Eine Übersicht ausgewählter Aufzeichnungen zu jeder Geschwindigkeit wurde in Abbildung 6.13 dargestellt.

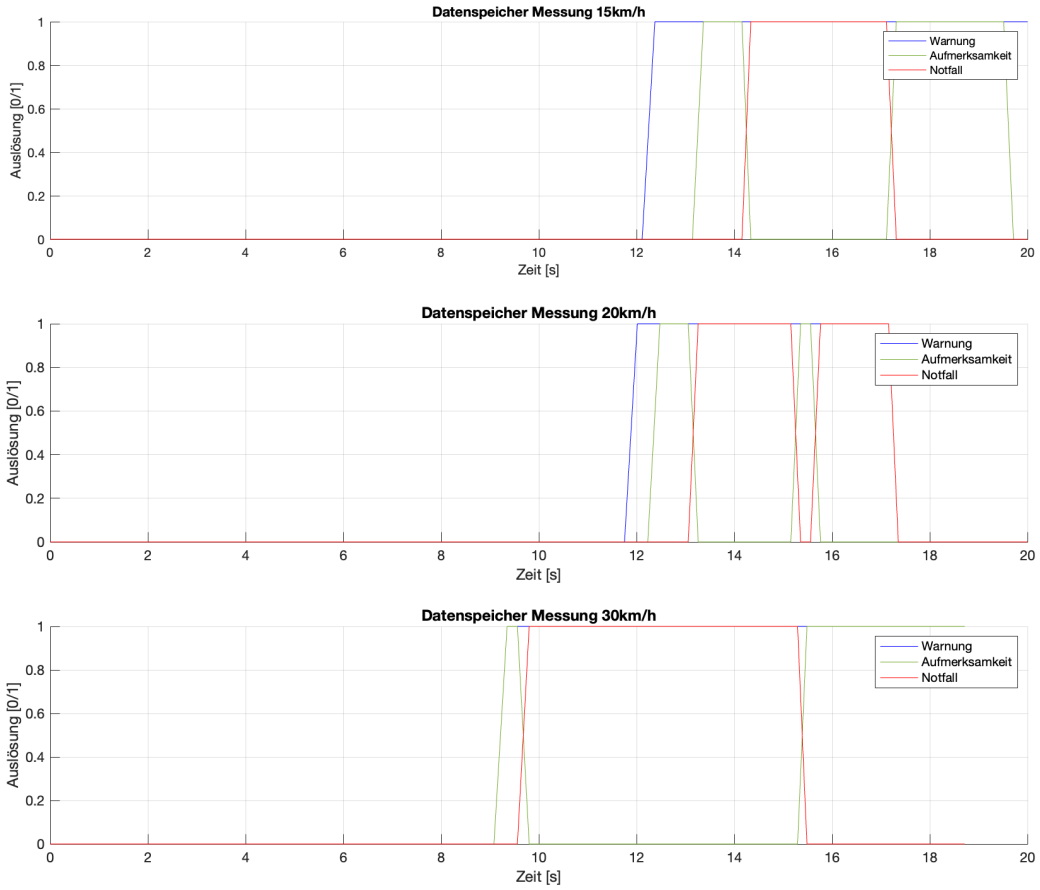


Abbildung 6.13.: Datenspeichersätze-Auswahl der realen Notbremsmessungen

Es fällt auf, dass der reine Warnbereich mit zunehmender Geschwindigkeit später beziehungsweise gar nicht ausgelöst wird. Dies lässt darauf schließen, dass dieser Bereich bei zu hohen Geschwindigkeiten entweder zu kurz ist, um wahrgenommen zu werden oder dass die Auffächerung der Punkte des LiDAR-Sensors mit zunehmender Distanz größer wird. Dies hat wiederum Einfluss auf die Clustererkennung des Assistenten, was zur Folge hat, dass das Hindernis erst ab einer bestimmten Entfernung richtig erkannt wird. Beide Fehlerquellen haben gemeinsam, dass der Assistent möglicherweise das Objekt einige Bilder später erst erkennt.

Unter Berücksichtigung der Geschwindigkeit von 30 km/h beziehungsweise 8,33 m/s und der Abfragerate der Blöcke von 0,2 Sekunden ergibt sich eine zurückgelegte Distanz von 1,67 m pro Bild. Nimmt man an, dass der Mittelwert dieser Geschwindigkeit aus Tabelle 6.12 von 3,54 m der korrekte Wert ist, so entspricht ein Abstand von 5,21 m einer Erkennung eher und ein Abstand von 1,87 m einer Erkennung später. Diese Werte decken sich sehr gut mit den in der Tabelle ermittelten Werten, wodurch anzunehmen ist, dass die Abweichungen daher rühren.

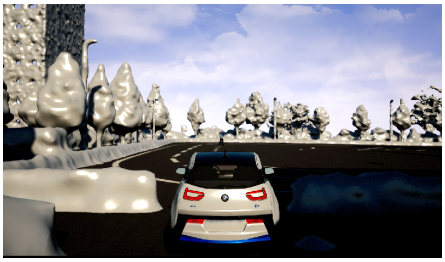
6.2.2. Simualtionsversuch

Die Randbedingungen aus dem realen Versuch konnten nur bedingt in der Simulation umgesetzt werden. Folgende Einstellungen wurden in der Simulation berücksichtigt:

- Asphalt: trocken (Verzögerung $8m/s^2$)
- Außentemperatur: 19℃
- Fahrzeuggewicht: 1416kg
- Simulink Modell Simulationstempo: An (1.7)
- Blockabfrageraten: 0,2s

Der Aufbau entspricht dem des realen Versuchs:

(a) Start reale Umgebung



(b) Start simulierte Umgebung

Abbildung 6.14.: Startposition des realen und simulierten Notbremsversuchs

In der Simulation wird die Messung der Distanzen durch das Auslesen und die Differenzbildung zwischen der y-Koordinate des Fahrzeugs, wie in Abbildung 6.15 dargestellt, und der Position des Dummys im Simulink-Modell, wie in Abbildung 6.16 dargestellt, durchgeführt. Da die Koordinate des Fahrzeugs den Mittelpunkt des BMW i3 widerspiegelt, muss zusätzlich die Distanz vom Fahrzeugmittelpunkt zur Fahrzeugfront addiert werden. Dieses Offset wurde durch Ausmessen in Blender bestimmt und beträgt 2,12m.

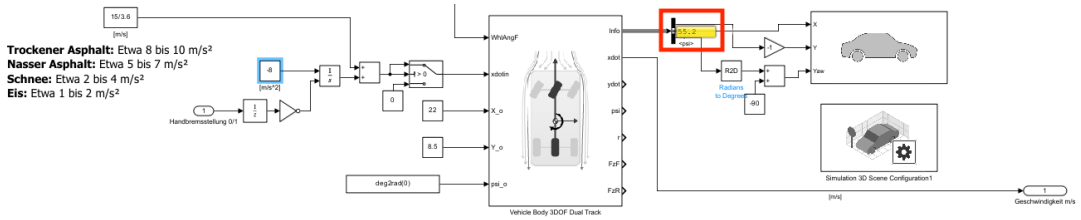


Abbildung 6.15.: Messpunkt des Fahrzeugs in Simulink

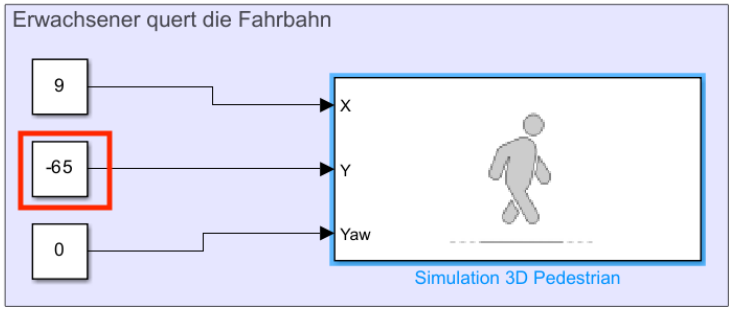


Abbildung 6.16.: Messpunkt des Dummys in Simulink

Für die erste Messung in der Simulationsumgebung bei einer Geschwindigkeit von 15 km/h und einer Verzögerung von $8m/s^2$ ergab sich ein Abstand von 7,68m. Da dieser Abstand nicht mit dem Mittelwert der Messung aus Tabelle 6.12 übereinstimmte, wurde der Verzögerungswert iterativ auf $2,5m/s^2$ gesenkt. Dieser Wert ist für trockenen Asphalt zwar sehr gering, jedoch aufgrund der elektronischen Handbremse, die nicht die volle Verzögerung leisten kann, angemessen. Mithilfe dieser Annahme wurden dann folgende Ergebnisse erzielt:

	15 km/h	20 km/h	30 km/h
Abstand Messung 1 [m]	5,28	5,88	2,78

Tabelle 6.13.: Simulativ gemessene Abstände des Notbremsversuchs

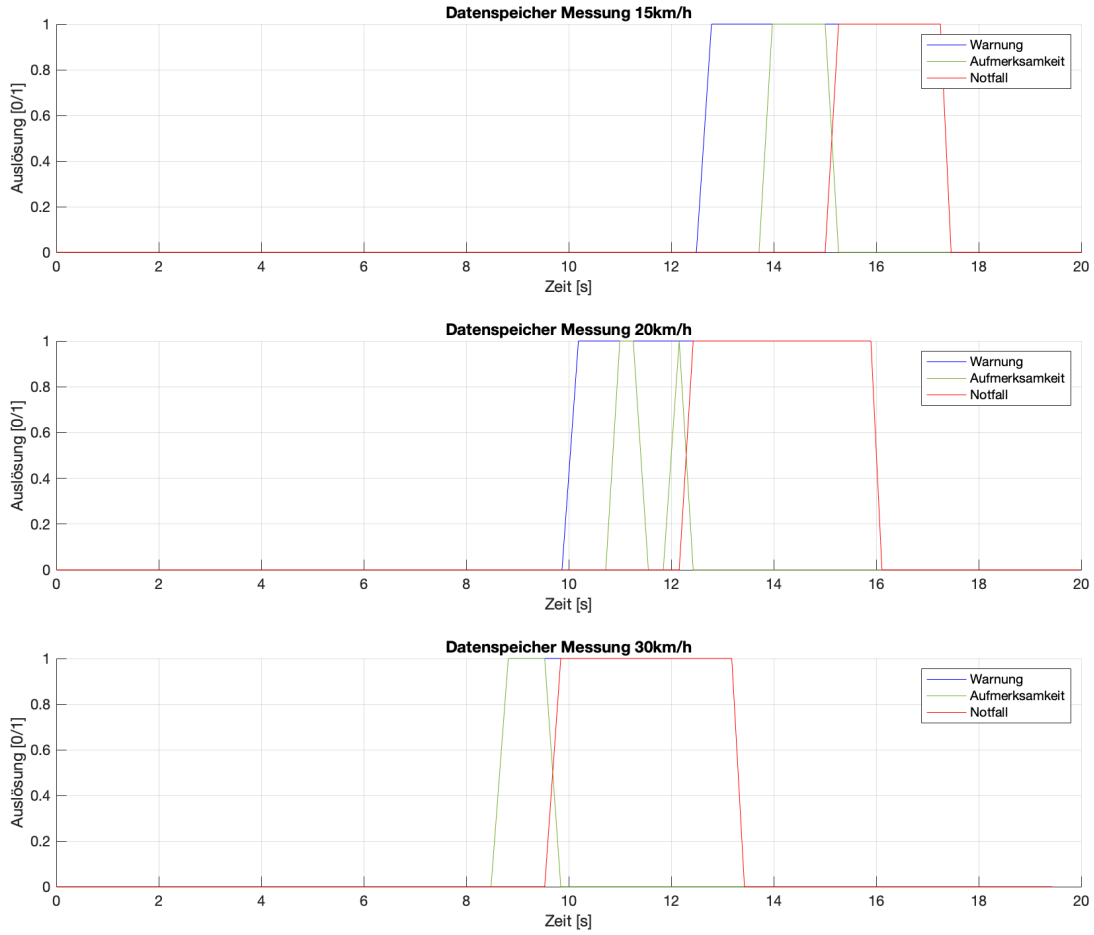


Abbildung 6.17.: Datenspeichersätze der simulierten Notbremsmessungen

Zu erkennen ist, dass sowohl die Werte aus Tabelle 6.12 und Tabelle 6.13 annähernd übereinstimmen, als auch die Abbildungen 6.13 und 6.17 die gleichen Eigenschaften aufweisen. Die Abstände weisen zwar eine Differenz von bis zu circa 1m auf, jedoch zeigt diese Auswertung, dass sowohl bei der Entwicklung als auch beim Testen der Fahrfunktionen eine realistische Nachbildung in der Simulationsumgebung möglich ist. Zudem wurde eine prototypische Umsetzung durchgeführt, sodass die Ergebnisse durch weitere Optimierungen, beispielsweise die Erhöhung der Abfragerate der Blöcke von 0,2 Sekunden auf 0,1 Sekunden, und Tests verbessert werden können. Zusätzlich wäre die Anpassung der Bereiche vorteilhaft, damit der Fahrer mehr Zeit zum Reagieren hat.

Auch die Umsetzung des Prüfszenarios durch den Prüfkatalog war erfolgreich, da in allen Fällen ein Warnton ausgegeben und bis zum Stillstand gebremst wurde. Lediglich die Annahme der Verzögerung von $8m/s^2$ musste auf $2,5m/s^2$ angepasst werden.

7. Ausblick

Diese Arbeit hat den Grundstein für die prototypische Umsetzung der Abfahrkontrolle gemäß AFGBV gelegt. Es wurden viele positive Erkenntnisse gewonnen und solide Grundlagen für weitere Forschungen geschaffen. Für eine zulassungstaugliche Reife sind jedoch noch zahlreiche Verbesserungen erforderlich. Zukünftig müssen die einzelnen Bausteine daher nochmals gesondert untersucht und nachfolgende mögliche Ideen umgesetzt werden.

Ein zukünftiger Ansatz wäre, die Simulationsblöcke aus den Modellen auszulagern und die Simulationsumgebung zu verfeinern sowie deren Übereinstimmung mit der Realität weiter zu testen. Auch die Implementierung überarbeiteter und neuer Fahrfunktionen am Fahrzeug ist notwendig, um alle rechtlichen Anforderungen zu erfüllen. Beispielsweise müssten mehr Recheneinheiten eingesetzt werden, um aus Sicherheits- und Performancegründen die einzelnen Funktionen auf mehrere Steuergeräte zu verteilen. Darüber hinaus muss das Fahrzeug seine Umgebung besser erfassen können, was bedeutet, dass die Umfelderkennung durch Sensorfusion erweitert und verbessert werden muss. Dies würde auch der Entwicklung eines notwendigen Ausweichassistenten zugutekommen. Auch die Wegvorgabe, etwa durch eine Navigationsroute, sollte genauer betrachtet werden, damit das Fahrzeug in Zukunft seinen Weg selbst berechnen kann und keine manuell aufgezeichnete Route benötigt. In diesem Zusammenhang wäre es auch notwendig, den Pfadverfolger durch die Einbindung der zurückgegebenen Werte der automatischen Notbremse oder zukünftiger Systeme zu beeinflussen. Weiterhin könnte die Implementierung neuer Pfadverfolgungsalgorithmen getestet werden, um das Fahrzeug schneller und effizienter zu bewegen. Mit der Erweiterung der Funktionalität des Fahrzeugs müsste zusätzlich der Prüfkatalog erweitert, verbessert und angepasst werden. Dabei könnten die realen Tests durch eine konstante Steuerung des Fahrzeugs verbessert werden und die Automatisierung der Auswertung der Prüfszenarien durch Drittsoftware erfolgen.

Trotz der vielen Anforderungen und der Tatsache, dass die HTWD noch einige Aufgaben vor sich hat, bis der Shuttle vollständig einsatzbereit ist, ist es möglich, das Projekt langfristig durch kontinuierliche Verbesserungen und Erweiterungen zu realisieren. So könnte für den Personentransport in Zukunft eine Lösung entstehen, bei der die Abfahrkontrolle gemäß AFGBV Anwendung findet.

Literatur- und Quellenverzeichnis

[1] Deutsches Patent- und Markenamt. Autonomes Fahren, Teil 3: Zahlen und Fakten. https://www.dpma.de/dpma/veroeffentlichungen/hintergrund/autonomesfahren/autonomesfahren-technikteil1/autonomesfahren-zahlenteil3/index.html [Aufgerufen am: 25.06.2024]. 2024.

[2] Ben Impey. Anzahl der angemeldeten Patente im Bereich autonomes Fahren weltweit in den Jahren 2009 bis 2019*. https://de.statista.com/statistik/daten/studie/1062558/umfrage/anzahl-der-patente-im-bereich-autonomen-fahren-weltweit/ [Aufgerufen am: 25.06.2024]. 2024.

[3] Bundesministerium der Justiz. Verordnung zur Genehmigung und zum Betrieb von Kraftfahrzeugen mit autonomer Fahrfunktion in festgelegten Betriebsbereichen (Autonome-Fahrzeuge-Genehmigungs-und-Betriebs-Verordnung-AFGBV). https://www.gesetze-im-internet.de/afgbv/AFGBV.pdf [Aufgerufen am: 18.06.2024]. 2022.

[4] Europäisches Parlament und der Rat der Europäischen Union. VERORDNUNG (EU) 2019/2144 DES EUROPÄISCHEN PARLAMENTS UND DES RATES. https://eur-lex.europa.eu/legal-content/DE/TXT/?uri=CELEX:32019R2144 [Aufgerufen am: 26.06.2024]. 2019.

[5] Bundesministeriums der Justiz. Straßenverkehrs-Ordnung (StVO). https://www.gesetze-im-internet.de/stvo_2013/StVO.pdf [Aufgerufen am: 25.06.2024]. 2013.

[6] Bundesministeriums der Justiz. Straßenverkehrs-Zulassungs-Ordnung (StVZO). https://www.gesetze-im-internet.de/stvzo_2012/StVZO.pdf [Aufgerufen am: 25.06.2024]. 2012.

[7] Volkswagen AG. Volkswagen will Autonomes Fahren zur Marktreife bringen. https://www.volkswagen-newsroom.com/de/bilder/detail/volkswagen-will-autonomes-fahren-zur-marktreife-bringen-30974 [Aufgerufen am: 03.07.2024]. 2019.

[8] SAE International. Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles. https://www.sae.org/standards/content/j3016_202104/preview/ [Aufgerufen am: 18.06.2024]. 2021.

[9] Wolfgang Matschinsky. „Radführungen der Straßenfahrzeuge“. In: *Radführungen der Straßenfahrzeuge*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, S. 5–6. ISBN: 978-3-540-71197-1. DOI: 10.1007/978-3-540-71197-1. URL: https://doi.org/10.1007/978-3-540-71197-1.

[10] Dieter Schramm, Manfred Hiller und Roberto Bardini. In: *Modellbildung und Simulation der Dynamik von Kraftfahrzeugen*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2018, S. 225–254. ISBN: 978-3-662-54481-5. DOI: 10.1007/978-3-662-54481-5_10. URL: https://doi.org/10.1007/978-3-662-54481-5_10.

[11] Mathepedia. Posberehcnung. https://mathepedia.de/Kreis.html [Aufgerufen am: 25.06.2024]. o.J.

[12] Prof. Dr. O. Bittel. Lokalisierung. https://www-home.htwg-konstanz.de/~bittel/ain_robo/Vorlesung/05_Lokalisierung.pdf [Aufgerufen am: 17.06.2024]. 2020.

[13] enArgus. Regelkreis. https://www.enargus.de/pub/bscw.cgi/d9362-2/*/*/Regelkreis?search=Regelkreis&op=Wiki.getwiki [Aufgerufen am: 17.06.2024]. o.J.

[14] Horst Czichos. „Regelung und Steuerung“. In: *Mechatronik: Grundlagen und Anwendungen technischer Systeme*. Wiesbaden: Springer Fachmedien Wiesbaden, 2019, S. 63–74. ISBN: 978-3-658-26294-5. DOI: 10.1007/978-3-658-26294-5_4. URL: https://doi.org/10.1007/978-3-658-26294-5_4.

[15] Temperatur Profis. Temperaturregler Typen: PID-Regler. https://temperatur-profis.de/temperaturregler/reglertypen-pid [Aufgerufen am: 17.06.2024]. o.J.

[16] Technik und Wissen: Das Fachmagazin für die Industrie. Die Krux mit der Totzeit. https://www.technik-und-wissen.ch/krux-mit-totzeit-regelungstechnik-mit-smith-praediktor.html [Aufgerufen am: 17.06.2024]. o.J.

[17] Elotech Industrielektronik GmbH. Grundlagen Regelungstechnik. https://www.elotech.de/fileadmin/user_upload/PID-Regelungsgrundlagen.pdf [Aufgerufen am: 27.06.2024]. 2018.

[18] MathWorks. Pure Pursuit Controller. https://de.mathworks.com/help/nav/ug/pure-pursuit-controller.html [Aufgerufen am: 25.06.2024]. o.J.

[19] Mario Theers und Mankaran Singh. Pure Pursuit. https://thomasfermi.github.io/Algorithms-for-Automated-Driving/Control/PurePursuit.html [Aufgerufen am: 25.06.2024]. o.J.

[20] Yan Ding. Three Methods of Vehicle Lateral Control: Pure Pursuit, Stanley and MPC. https://www.shuffleai.blog/blog/Three_Methods_of_Vehicle_Lateral_Control.html [Aufgerufen am: 18.06.2024]. 2021.

[21] Maria-Despina Makri u. a. „Auswirkungen infrastruktureller Maßnahmen autonomer/hochautomatisierter Buslinien auf die Wirtschaftlichkeit“. In: *Towards the New Normal in Mobility: Technische und betriebswirtschaftliche Aspekte*. Hrsg. von Heike Proff. Wiesbaden: Springer Fachmedien Wiesbaden, 2023, S. 1047–1062. ISBN: 978-3-658-39438-7. DOI: 10.1007/978-3-658-39438-7_59. URL: https://doi.org/10.1007/978-3-658-39438-7_59.

[22] MagicMaps. GNSS-Wissen. https://www.magicmaps.de/gnss-wissen/ [Aufgerufen am: 06.07.2024]. 2020.

[23] Volkswagen AG. Car2X im neuen Golf: Ein „technischer Meilenstein“. https://www.volkswagen-newsroom.com/de/storys/car2x-im-neuen-golf-ein-technischer-meilenstein-5919 [Aufgerufen am: 06.07.2024]. 2020.

[24] Dr. Martin Large. Lidar: Was es ist, wie es funktioniert und was es kann. https://www.all-electronics.de/automotive-transportation/lidar-sensoren-automotive-575.html [Aufgerufen am: 25.06.2024]. 2017.

[25] Alza.cz a.s. Time of Flight Sensor - ein Durchbruch beim 3D-Scannen? https://www.alza.de/time-of-flight-sensor [Aufgerufen am: 07.08.2024]. 2020.

[26] Global GPS Systems. 905nm VS 1550nm: was ist besser für Kfz-LiDAR? https://www.lslidar.com/de/905nm-vs-1550nm-which-is-better-for-automotive-lidar?/ [Aufgerufen am: 07.08.2024]. 2022.

[27] InnoSenT. Techniktrend Radar. https://www.innosent.de/radar/ [Aufgerufen am: 27.06.2024]. o.J.

[28] cetecom advanced. Frequenzbereiche für die Automotive Radar-Technologie. https://cetecomadvanced.com/de/news/frequenzbereiche-fuer-die-automotive-radar-technologie/ [Aufgerufen am: 05.08.2024]. 2019.

[29] Paul Balzer. Fahrzeugumfeldsensorik: LiDAR, Radar, Infrarot, Ultraschall und Video im Vergleich – Funktionsweise, Vor- und Nachteile, Sensordatenfusion. https://www.zukunft-mobilitaet.net/79615/strassenverkehr/fahrzeugumfeldsensorik-funktionsweise-lidar-radar-infrarotsensor-ultraschall-videosensor/ [Aufgerufen am: 07.08.2024]. 2014.

[30] ifm electronic gmbh. Vorteile von Ultraschallsensoren. https://www.ifm.com/de/de/shared/technologien/ultraschallsensoren/vorteile [Aufgerufen am: 07.08.2024]. o.J.

[31] Joachim Herz Stiftung. Mechanische Wellen. https://www.leifiphysik.de/mechanik/mechanische-wellen/grundwissen/groessen-zur-beschreibung-einer-welle [Aufgerufen am: 07.08.2024]. o.J.

[32] Global GPS Systems. Explore the Different Types of Lidar Technology: From Mechanical to Solid-State Lidar. https://globalgpssystems.com/lidar/explore-the-different-types-of-lidar-technology-from-mechanical-to-solid-state-lidar/ [Aufgerufen am: 07.08.2024]. o.J.

[33] Ouster Inc. OS1 Mid-range digital lidar sensor. https://ouster.com/products/hardware/os1-lidar-sensor [Aufgerufen am: 18.06.2024]. o.J.

[34] Jeff Meyer und Alex Summersby. Bildsensoren erklärt. https://www.canon.de/pro/infobank/image-sensors-explained/ [Aufgerufen am: 07.08.2024]. o.J.

[35] HELLA GmbH & Co. KGaA. ABS SENSOR PRÜFEN UND WECHSELN. https://www.hella.com/techworld/de/Technik/Sensoren-und-Aktuatoren/ABS-Sensor-pruefen-und-wechseln-4074/ [Aufgerufen am: 07.08.2024]. o.J.

[36] Bosch Engineering GmbH. Speed Sensor Hall-Effect HA-Di. https://www.bosch-motorsport.com/content/downloads/Raceparts/Resources/pdf/Data%20Sheet_69979403_Speed_Sensor_Hall-Effect_HA-Di.pdf [Aufgerufen am: 07.08.2024]. 2024.

[37] Erich Zabler u. a. „Sensoren“. In: *Sensoren im Kraftfahrzeug*. Hrsg. von Konrad Reif. Wiesbaden: Springer Fachmedien Wiesbaden, 2016, S. 112–171. ISBN: 978-3-658-11211-0. DOI: 10.1007/978-3-658-11211-0_3. URL: https://doi.org/10.1007/978-3-658-11211-0_3.

[38] Robert Bosch GmbH. Peripherer Beschleunigungssensor. https://www.bosch-mobility.com/de/loesungen/sensoren/peripherer-beschleunigungssensor/ [Aufgerufen am: 06.09.2024]. o.J.

[39] Conrad Electronic SE. MEMS » Mikro-Elektronisch-Mechanische-Systeme einfach erklärt. https://www.conrad.de/de/ratgeber/industrie-40/mems.html [Aufgerufen am: 06.09.2024]. 2022.

[40] ifm electronic gmbh. Neigungssensoren Technologie. https://www.ifm.com/de/de/shared/produkte/neigungssensoren/technologie [Aufgerufen am: 06.09.2024]. o.J.

[41] iBosch Mobility. DE | Bosch Funktionsprinzip eines Drehratensensors für ESP®. https://www.youtube.com/watch?v=6_yhOORMpc8 [Aufgerufen am: 06.09.2024]. 2014.

[42] Bonnie Baker. Anwendung der Sensorfusion auf Beschleunigungsmesser und Gyroskope. https://www.digikey.de/de/articles/apply-sensor-fusion-to-accelerometers-and-gyroscopes [Aufgerufen am: 06.09.2024]. 2018.

[43] Robert Bosch GmbH. Lenkwinkelsensor Ermittlung des Lenkwinkels und der Lenkwinkelgeschwindigkeit. https://www.bosch-mobility.com/de/loesungen/sensoren/lenkwinkelsensor/ [Aufgerufen am: 06.09.2024]. o.J.

[44] Simon Fürst u. a. „Digitalisierung /Elektrik/Elektronik/Software“. In: *Vieweg Handbuch Kraftfahrzeugtechnik*. Hrsg. von Stefan Pischinger und Ulrich Seiffert. Wiesbaden: Springer Fachmedien Wiesbaden, 2021, S. 926–936. ISBN: 978-3-658-25557-2. DOI: 10.1007/978-3-658-25557-2_7. URL: https://doi.org/10.1007/978-3-658-25557-2_7.

[45] Alan A. Varghese. Automobil- und Automobil-Ethernet-Hersteller Multi-Gig. https://www.redeweb.com/de/Artikel/Autohersteller-und-Multi-Gig-Automotive-Ethernet/ [Aufgerufen am: 06.09.2024]. 2021.

[46] Ansgar Meroth und Petre Sora. „CAN Bus“. In: *Sensornetzwerke in Theorie und Praxis: Embedded Systems-Projekte erfolgreich realisieren*. Wiesbaden: Springer Fachmedien Wiesbaden, 2021, S. 263–266. ISBN: 978-3-658-31709-6. DOI: 10.1007/978-3-658-31709-6_10. URL: https://doi.org/10.1007/978-3-658-31709-6_10.

[47] Nicole Poettrich. fusion-systems-360-grad-umgebungserfassung-mittels-multi-sensor-datenfusion-vertikal. https://www.autoland-sachsen.com/schluesseltechnologien-fuer-das-automatisierte-fahren/fusion-systems-360-grad-umgebungserfassung-mittels-multi-sensor-datenfusion-vertikal/ [Aufgerufen am: 07.08.2024]. 2018.

[48] Semiconductor Components Industries. ADAS. https://www.onsemi.com/solutions/automotive/adas [Aufgerufen am: 07.08.2024]. o.J.

[49] Mariam Elazab. Integrated Cooperative Localization in VANETs for GPS Denied Environments. https://www.researchgate.net/figure/KF-overview-for-GPS-RISS-integration_fig4_282359540 [Aufgerufen am: 07.08.2024]. 2015.

[50] Ouster Inc. OS1 Mid-Range High-Resolution Imaging Lidar. https://data.ouster.io/downloads/datasheets/datasheet-rev7-v3p1-os1.pdf [Aufgerufen am: 18.06.2024]. o.J.

[51] Ouster Inc. 7.2.1 Interface Box 24V Compatibility (Standard). https://static.ouster.dev/sensor-docs/hw_user_manual_OS2/hw_common_sections_OS2/Accessories.html [Aufgerufen am: 18.06.2024]. 2022.

[52] The MathWorks Inc. Using Unreal Engine with Simulink. https://de.mathworks.com/videos/series/using-unreal-engine-with-simulink.html [Aufgerufen am: 05.07.2024]. o.J.

[53] The MathWorks Inc. Customize 3D Scenes for Vehicle Dynamics Simulations. https://de.mathworks.com/help/vdynblks/ug/customize-3d-scenes-for-vehicle-dynamics-simulations.html [Aufgerufen am: 05.07.2024]. o.J.

[54] Harry McKenzie. How can I convert a complex point cloud to mesh? https://blender.stackexchange.com/questions/301218/how-can-i-convert-a-complex-point-cloud-to-mesh [Aufgerufen am: 05.07.2024]. 2024.

[55] Mykel Terrell. Unreal Engine Spline Decal Blueprint Breakdown. https://youtu.be/ZEFzR4PQv-c [Aufgerufen am: 05.07.2024]. 2022.

[56] The MathWorks Inc. Preparing A Custom Vehicle Mesh for the Unreal Editor. https://www.youtube.com/watch?v=A5P1aMTrq_s [Aufgerufen am: 05.07.2024]. 2022.

[57] The MathWorks Inc. Prepare Custom Vehicle Mesh for the Unreal Editor. https://de.mathworks.com/help/driving/ug/prepare-custom-vehicle-mesh-for-the-unreal-editor.html?s_eid=PSM_15028 [Aufgerufen am: 05.07.2024]. o.J.

[58] The MathWorks Inc. Compiling Custom Scenes | Using Unreal Engine with Simulink, Part 4. https://de.mathworks.com/videos/using-unreal-engine-with-simulink-part-4-compiling-custom-scenes-1634213550519.html [Aufgerufen am: 05.07.2024]. 2021.

[59] Toralf Trautmann u. a. Design and Experiments of Autonomous Path Tracking Based on Dead Reckoning. https://www.mdpi.com/2076-3417/13/1/317 [Aufgerufen am: 04.08.2024]. 2022.

[60] The MathWorks Inc. Vehicle Path Tracking Using Pure Pursuit Controller. https://www.youtube.com/watch?v=zMdoLO4kRKg&t=481s [Aufgerufen am: 05.07.2024]. 2020.

[61] The MathWorks Inc. Vehicle Body 3DOF. https://de.mathworks.com/help/vdynblks/ref/vehiclebody3dof.html [Aufgerufen am: 05.07.2024]. o.J.

[62] The MathWorks Inc. objectDetection. https://de.mathworks.com/help/fusion/ref/objectdetection.html [Aufgerufen am: 05.07.2024]. o.J.

[63] The MathWorks Inc. objectTrack. https://de.mathworks.com/help/fusion/ref/objecttrack.html [Aufgerufen am: 05.07.2024]. o.J.

[64] The MathWorks Inc. Create Custom Blocks Using MATLAB System Block and System objects. https://de.mathworks.com/help/simulink/ug/what-is-matlab-system-block.html [Aufgerufen am: 05.07.2024]. o.J.

[65] The MathWorks Inc. Implementierung von Blöcken mit System objects. https://de.mathworks.com/help/simulink/matlab-system-block.html [Aufgerufen am: 05.07.2024]. o.J.

[66] The MathWorks Inc. Techniques to Improve Performance. https://de.mathworks.com/help/matlab/matlab_prog/techniques-for-improving-performance.html [Aufgerufen am: 05.07.2024]. o.J.

[67] The MathWorks Inc. Analyze Simulation Execution Using Simulink Profiler. https://de.mathworks.com/help/simulink/slref/introduction-to-profiling-models.html [Aufgerufen am: 05.07.2024]. o.J.

[68] The MathWorks Inc. Multi-Object Tracker. https://de.mathworks.com/help/driving/ref/multiobjecttracker.html [Aufgerufen am: 05.07.2024]. o.J.

[69] Euro NCAP. AEB Pedestrian. https://www.euroncap.com/en/car-safety/the-ratings-explained/vulnerable-road-user-vru-protection/aeb-pedestrian/ [Aufgerufen am: 05.07.2024]. o.J.

[70] Marlis Reisenauer. Bremsassistent: So macht er das Autofahren sicherer. https://www.da-direkt.de/autoversicherung/ratgeber/bremsassistent#wann-greift-bremsassistent-ein [Aufgerufen am: 25.06.2024]. 2024.

[71] GitLab B.V. 10 reasons why enterprises choose GitLab. https://about.gitlab.com/why-gitlab/#open-core [Aufgerufen am: 04.07.2024]. o.J.

[72] Viktor Peters. GitLab: Was ist das und was kann das? https://www.mittwald.de/blog/mittwald/tools/gitlab [Aufgerufen am: 04.07.2024]. 2019.

[73] JGraph Ltd. Our range of draw.io branded integrations. https://www.drawio.com [Aufgerufen am: 04.07.2024]. o.J.

[74] o.A. The easiest way for Confluence teams to collaborate using diagrams. https://drawio-app.com [Aufgerufen am: 04.07.2024]. o.J.

[75] The MathWorks Inc. MATLAB. https://de.mathworks.com/products/matlab.html [Aufgerufen am: 04.07.2024]. o.J.

[76] The MathWorks Inc. Simulink. https://de.mathworks.com/products/simulink.html [Aufgerufen am: 04.07.2024]. o.J.

[77] The MathWorks Inc. Produkte & Dienstleistungen. https://de.mathworks.com/products.html [Aufgerufen am: 04.07.2024]. o.J.

[78] The Blender Foundation. Blender 4.1. https://www.blender.org [Aufgerufen am: 05.07.2024]. o.J.

[79] Inc. Epic Games. Wir liefern die Engine. Sie machen sie Unreal. https://www.unrealengine.com/de [Aufgerufen am: 04.07.2024]. o.J.

[80] Inc. Epic Games. Erzeugen Sie Simulationen, die sich (fast) wie die Realität anfühlen. https://www.unrealengine.com/de/uses/simulation [Aufgerufen am: 04.07.2024]. o.J.

[81] The MathWorks Inc. Unreal Engine Simulation Environment Requirements and Limitations. https://de.mathworks.com/help/vdynblks/ug/unreal-engine-simulation-environment-requirements-and-limitations.html [Aufgerufen am: 04.07.2024]. o.J.

[82] CARLA Team. CARLA Open-source simulator for autonomous driving research. https://carla.org [Aufgerufen am: 04.07.2024]. o.J.

[83] The MathWorks Inc. Audio Toolbox Entwickeln und Analysieren von Sprach-, Akustik- und Audioverarbeitungssystemen. https://de.mathworks.com/products/audio.html [Aufgerufen am: 07.07.2024]. o.J.

[84] The MathWorks Inc. Automated Driving Toolbox Entwicklung, Simulation und Testen von ADAS und Systemen für autonomes Fahren. https://de.mathworks.com/products/automated-driving.html [Aufgerufen am: 07.07.2024]. o.J.

[85] MathWorks Automated Driving Toolbox Team. Automated Driving Toolbox Interface for Unreal Engine Projects. https://de.mathworks.com/matlabcentral/fileexchange/74555-automated-driving-toolbox-interface-for-unreal-engine-projects [Aufgerufen am: 04.07.2024]. 2024.

[86] The MathWorks Inc. Computer Vision Toolbox Entwurf und Test von Computer Vision-Systemen. https://de.mathworks.com/products/computer-vision.html [Aufgerufen am: 07.07.2024]. o.J.

[87] The MathWorks Inc. DSP System Toolbox Entwicklung und Simulation von Streaming-Signalverarbeitungssystemen. https://de.mathworks.com/products/dsp-system.html [Aufgerufen am: 07.07.2024]. o.J.

[88] The MathWorks Inc. Image Processing Toolbox Durchführung der Bildverarbeitung, Visualisierung und Analyse. https://de.mathworks.com/products/image.html [Aufgerufen am: 04.07.2024]. o.J.

[89] The MathWorks Inc. Lidar Toolbox Entwurf, Analyse und Testen von LiDAR-Verarbeitungssystemen. https://de.mathworks.com/products/lidar.html [Aufgerufen am: 07.07.2024]. o.J.

[90] The MathWorks Inc. Lidar Toolbox Support Package for Ouster Lidar Sensors. https://de.mathworks.com/matlabcentral/fileexchange/106705-lidar-toolbox-support-package-for-ouster-lidar-sensors [Aufgerufen am: 07.07.2024]. o.J.

[91] The MathWorks Inc. Navigation Toolbox Entwerfen, Simulieren und Bereitstellen von Algorithmen zur autonomen Navigation. https://de.mathworks.com/products/navigation.html [Aufgerufen am: 07.07.2024]. o.J.

[92] The MathWorks Inc. Sensor Fusion and Tracking Toolbox Entwurf, Simulation und Test von Multisensor-Tracking- und Navigationssystemen. https://de.mathworks.com/products/sensor-fusion-and-tracking.html [Aufgerufen am: 07.07.2024]. o.J.

[93] The MathWorks Inc. Signal Processing Toolbox Signalverarbeitung und -analyse. https://de.mathworks.com/products/signal.html [Aufgerufen am: 07.07.2024]. o.J.

[94] The MathWorks Inc. Simulink 3D Animation Simulation und Visualisierung dynamischer Systeme in einer 3D-Umgebung. https://de.mathworks.com/products/3d-animation.html [Aufgerufen am: 04.07.2024]. o.J.

[95] The MathWorks Inc. Vehicle Dynamics Blockset Modellieren und Simulation der Fahrzeugdynamik in einer virtuellen 3D-Umgebung. https://de.mathworks.com/products/vehicle-dynamics.html [Aufgerufen am: 04.07.2024]. o.J.

[96] MathWorks Automotive Team. Vehicle Dynamics Blockset Interface for Unreal Engine Projects. https://de.mathworks.com/matlabcentral/fileexchange/65966-vehicle-dynamics-blockset-interface-for-unreal-engine-projects [Aufgerufen am: 04.07.2024]. 2024.

[97] The MathWorks Inc. Vehicle Network Toolbox Communicate with in-vehicle networks using CAN, J1939, and XCP protocols. https://de.mathworks.com/products/vehicle-network.html [Aufgerufen am: 04.07.2024]. o.J.

Eidesstattliche Erklärung

Das vorliegende Dokument wurde an der Hochschule für Technik und Wirtschaft Dresden unter der Leitung von Prof. Dr. rer. nat. Toralf Trautmann und Dipl.-Ing. (FH) Dirk Engert angefertigt. Hiermit erkläre ich, dass ich die vorliegende Arbeit zum Thema

„Entwicklung eines Funktions-Prototypen für die erweiterte Abfahrkontrolle autonomer Fahrzeuge laut AFGBV“

selbstständig und ohne Benutzung anderer Quellen und Hilfsmittel als angegeben angefertigt habe. Insbesondere versichere ich, dass ich alle wörtlichen und sinngemäßen Übernahmen aus Werken als solche kenntlich gemacht habe. Ferner gestatte ich der Hochschule für Technik und Wirtschaft Dresden, die vorliegende Diplomarbeit unter Beachtung insbesondere urheber-, datenschutz-, und wettbewerbsrechtlicher Vorschriften für Lehre und Forschung zu nutzen.

Meißen, 23.10.2024

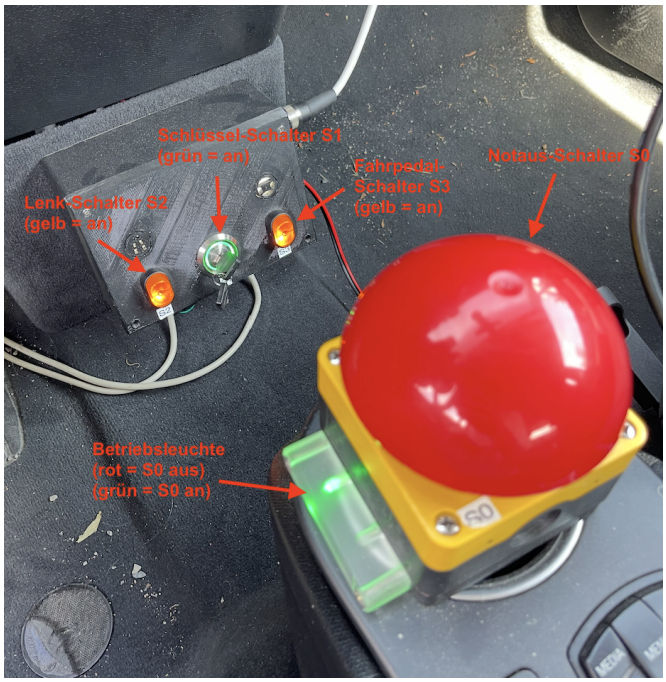
Ort, Datum

Tom Irrasch

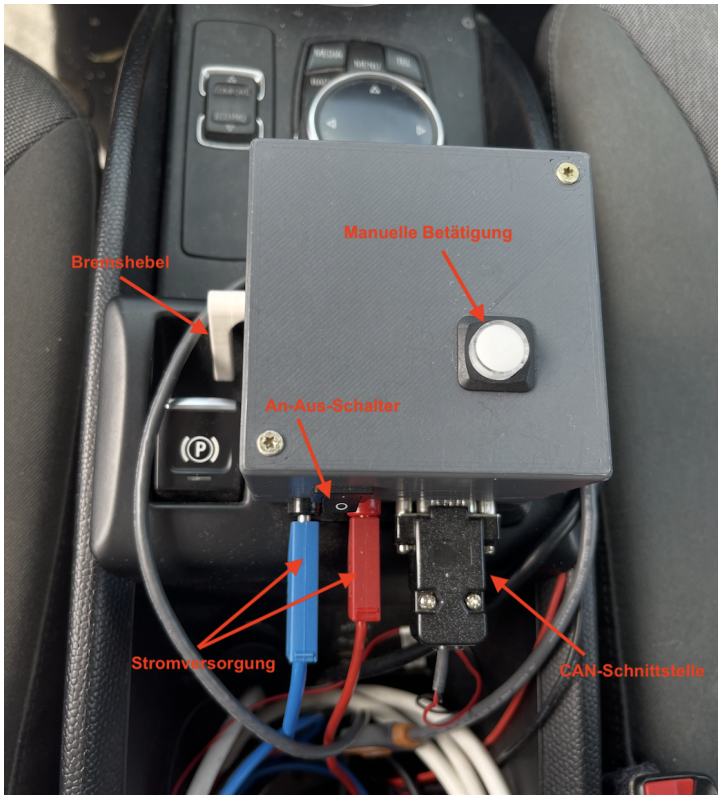
Anlagen

A. Umbauten BMW i3

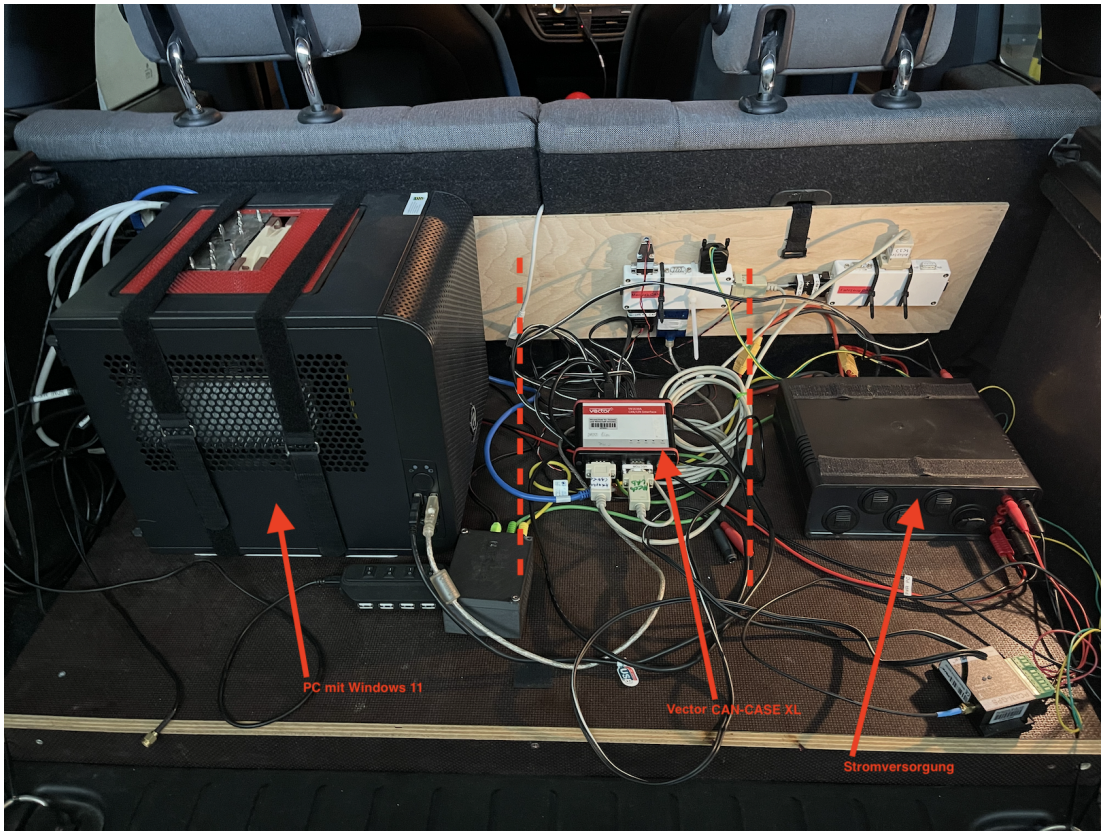
A.1. Notaus und Zuschaltung der Aktuatoren



A.2. Handbrems-Aktuator



A.3. Verbaute Technik im BMW i3 Kofferraum



B. Anforderungen

B.1. Anforderungen an die Quer- und Längsführung

Anforderung	**Vorgabe durch [3],[5],[6]**
Kompatibilität mit dem Versuchsträger (BMW i3)	Aufgabenstellung
Umsetzung in MATLAB Simulink	Aufgabenstellung
Alle StVO/StVZO-Vorschriften beachten	StVO/StVZO
Sicherheit und Leichtigkeit des Straßenverkehrs bewahren, sowie Gefährdung von Leib oder Leben von Personen ausschließen	AFGBV § 4 Abs.1 4.
Deaktivierung oder Freigabe von Fahrmanövern während Betrieb	AFGBV § 10 Abs.1 3.
Geeignete Wahl von Fahrpfad und Geschwindigkeit selbstständig und stetig anpassend	AFGBV A1 T1 1.
Vorausfahrende Verkehrsteilnehmende auf der Fahrbahn werden erkannt	AFGBV A1 T1 1. 1.2 a)
Immer angemessener Sicherheitsabstand einzuhalten (bestimmt durch § 4 der StVO)	AFGBV A1 T1 1. 1.2 a)
Bei Defekt muss in Notfahrfunktion übergegangen werden (Schrittgeschwindigkeit und Warnblinker)	AFGBV A1 T1 3.
Fahrstreifenwechsel anderer Fahrzeuge benachbarter Fahrstreifen müssen erkannt + berücksichtigt werden	AFGBV A1 T1 1. 1.2 b)
Einsatzfahrzeuge oder Fahrstreifenwechsel bedingte Manöver erkennen und durchführen	AFGBV A1 T1 1. 1.2 d)
Mehr als drei Sekunden bezüglich des Vorfahrtsberechtigten einzuhalten sonst Begründung	AFGBV A1 T1 1. 1.3 d)
Geschwindigkeit anpassen auch an besondere Fahrtechnische Hindernisse erkennen und anpassen (Kurven, Schule, Engstellen)	AFGBV A1 T1 1. 1.3 a) / b)
Veränderte Fahrbahneigenschaften (z.B. Verschleiß, Änderungen etc.) erkennen und handeln	AFGBV A1 T1 1. 1.3 f)

Fortsetzung auf nächster Seite

Anforderung	**Vorgabe durch [3]**
Digitaler, nicht flüchtiger Speicher benötigt (Aufzeichnung z.B. ab Abfahrt) Die gespeicherten Daten müssen im stromlosen Zustand erhalten bleibe	AFGBV A1 T3 13. 13.2
Systeme zur Erkennung und Bewertung der Situation (Sachschäden, Personen, Tiere) -> geeignetes Fahrmanöver ausführen (insbesondere Notbrems- / Ausweichassisstent)	AFGBV §3 Abs.8
Einsatzkräfte müssen autonome Fahrfunktion sicher und erkennbar abschalten können, um Fahrzeug festzusetzen und nur manuell bewegen können	AFGBV A1 T1 7. 7.2.2 c)
Erreicht die autonome Fahrt die Grenzen des festgelegten Betriebsbereichs, muss das Kraftfahrzeug durch die autonome Funktion in den risikominimalen Zustand versetzt werden (zunehmende Warnung z.B. durch Warntöne)	AFGBV A1 T4 14. 14.2
Alterung / Abnutzung des Systems bedenken	AFGBV §3 Abs.2 1. a)
Verlassen des Fahrstreifens in folgenden Fällen: „Fahrstreifenwechsel“, für Manövrieren im niedrigen Geschwindigkeitsbereich (z.B. Einparken, im Bereich enger Kreuzungen), Ausweichen bei Hindernissen und bei entgegenkommenden Fahrzeugen, zur Kollisionsvermeidung sowie um Einsatzfahrzeugen auszuweichen.	AFGBV A1 T2 10. 10.2.2
Wetter-, Umwelt- und Straßeninfrastrukturbedingungen werden im Geschwindigkeits- und Fahrtverlauf berücksichtigt	AFGBV A1 T1 1. 1.4

B.2. Anforderungen an die autonome Fahrfunktion

Anforderung	**Vorgabe durch [3],[4],[70]**
Kompatibilität mit dem Versuchsträger (BMW i3)	Aufgabenstellung
Umsetzung in MATLAB Simulink	Aufgabenstellung
Optisches / Akustisches Warnsignal	Allgemein
Erkennung fester und beweglicher Hindernisse	Allgemein
Sicher Radfahrer und Fußgänger identifizieren	Allgemein
Vorerst Geschwindigkeit reduzieren danach Bremsung (Teil, Voll)	Allgemein
Kombination mit anderen Funktionen z.B. Gurtstraffer oder Ausweichen	Allgemein
Oft City Notbremse bis 60 km/h vereinzelt Autobahn 130 km/h (selten Abbiegeerweiterung)	Allgemein
Erkennung hauptsächlich vorn aber auch teilweise seitlich und hinten	Allgemein
autonome Fahrfunktion darf nicht aktivierbar sein solange defekt besteht	AFGBV A1 T1 5. 5.2
Selbstständige, zeitlich begrenzte Deaktivierung bei (äußeren) Mängeln	EU-V 2019/2144 (15)
Darf nur nacheinander durch Abfolge vom Fahrer durchzuführenden Handlungen abgeschaltet werden	EU-V 2019/2144 K2 A7 (4) a)
Muss ohne Verwendung biometrischer Daten von Fahrern oder Fahrgästen funktionieren	EU-V 2019/2144 (10)
Gewährleistung, dass das System während des gesamten Lebenszyklus des Fahrzeugs sicher betrieben werden kann	EU-V 2019/2144 (10)
Bei jeder Aktivierung des Hauptkontrollschalters des Fahrzeugs im Normalbetrieb befinden	EU-V 2019/2144 K2 A7 (4) b)
Möglichkeit akustische Warnsignale zu unterdrücken; zugleich jedoch keine anderen Funktionen außer akustischen Warnsignalen unterdrückt werden	EU-V 2019/2144 K2 A7 (4) c)
Info Fahrer Deaktivierung zeitlich begrenzt und nur so lange wie das System nicht vollständig einsatzfähig	EU-V 2019/2144 (15)

B.3. Anforderungen an die Testszenarien

Anforderung	**Vorgabe durch [3],[69]**
Beachtung des resultierenden Schadensausmaßes	NCAP
Auftretenswahrscheinlichkeit einer bestimmten Verkehrssituation	NCAP
Nach Notwendigkeit durchzuführen + begründen	AFGBV A1 T2 10.
Ausreichende Testabdeckung für alle Szenarien, Testparameter und Umwelteinflüsse bieten	AFGBV A1 T2 10.
Empirische nicht personenbezogenen Daten	AFGBV A1 T2 10.
Geeigneter Nachzuweisen, dass Maß an Sicherheit des Kraftfahrzeugs mit autonomer Fahrfunktion höher ist als Maß an Sicherheit bei Fahrzeugen, die von Personen geführt werden	AFGBV A1 T2 10.
Müssen geeignet sein, eine hinreichende Robustheit der technischen Ausrüstung zur Umgebungswahrnehmung gegen die Störung von Eingabe-/Sensordaten und ungünstige Umweltbedingungen nachzuweisen	AFGBV A1 T2 10.
Künstliche Fehler dürfen verursacht werden oder anderer Fahrbereich als der genehmigte	AFGBV A1 T2 10. 10.1 a)
Entsprechend des vorgesehenen Betriebsbereichs wählt das Kraftfahrt-Bundesamt Testszenarien im Rahmen der Prüfung aus (auf der Basis des Katalogs von Testszenarien des Herstellers)	AFGBV A1 T2 10. 10.2
Im Rahmen der Erteilung der Betriebserlaubnis Fahrtests im realen Straßenverkehr durchgeführt werden. Die Prüfung wird durch Simulationen und Durchführungen von Fahrmanövern auf einem Testgelände ergänzt	AFGBV A1 T2 10. 10.2
Bestehenskriterien richten sich nach Auswahl Abweichungen müssen begründet und dokumentiert werden	AFGBV A1 T2 10. 10.2.
Neben realen Fahrzeugen auch dem Stand der Technik entsprechende Testwerkzeuge eingesetzt werden	AFGBV A1 T2 11.

Fortsetzung auf nächster Seite

Anforderung	Vorgabe durch [3]
Am Versuch beteiligten Personen nicht gefährdet werden. Die jeweiligen Vorgaben des Arbeitsschutzes sind zu berücksichtigen	AFGBV A1 T2 11.
Für eine Leistungsbewertung der Sensorik relevanten Eigenschaften, realen Fahrzeugen und anderen am Verkehr Teilnehmenden entsprechen	AFGBV A1 T2 11.
Simulationswerkzeuge erlaubt aber validieren.(mittels Abgleichs zu einer repräsentativen Auswahl von realen Versuchen erfolgen; kein signifikanter Unterschied zwischen Kennwerten aus Simulation und Fahrversuch)	AFGBV A1 T2 11.
Das Leistungsvermögen der Sensorik in Bezug auf Erkennung und Klassifizierung von Objekten in Abhängigkeit von unterschiedlichen Entfernungen und Umweltbedingungen ist für die Simulation in realen Tests zu ermitteln. Jede Simulationsreihe ist, falls dies vom technischen Dienst als notwendig erachtet wird, durch reale Tests zu ergänzen	AFGBV A1 T2 11.
Jede in Verordnung beschriebene Anforderung, die im vorgesehenen Betriebsbereich für den autonomen Fahrbetrieb entsprechend der beantragten Betriebserlaubnis relevant ist und jedes nach Nummer 7.2 identifizierte gefährliche Szenario sind zumindest durch Simulation zu prüfen. zu testende Kraftfahrzeug im autonomen Fahrbetrieb durch geeignete Wahl der Verkehrsumgebung in die entsprechende Situation zu bringen. Es ist mindestens zu prüfen, wie sich das Kraftfahrzeug mit autonomer Fahrfunktion in den in Nummer 7.2 als gefährlich identifizierten Szenarien verhält. Für diese Prüfung sind mindestens drei Parameterkonstellationen zu wählenp	AFGBV A1 T2 11.
Für Prüfungen im Rahmen der Genehmigungserteilung nach § 3 kann der für die Genehmigung vorgesehene festgelegte Betriebsbereich selbst genutzt werden, sofern dort Tests gefahrlos für andere Verkehrsteilnehmende und unbeteiligte Dritte erfolgen können. Tests sind unter verschiedenen Umweltbedingungen durchzuführen	AFGBV A1 T2 12.

C. Benutzte Programme

GitLab als Verwaltungswerkzeug

GitLab ist eine der führenden Webanwendungen für die Zusammenarbeit, Verwaltung und Versionierung von Projekten. Sie ermöglicht es Einzelpersonen oder Teams, ihre Arbeit online organisiert zu sichern. Ein wesentlicher Vorteil von GitLab ist, dass nur die neuesten Änderungen hochgeladen werden. Dies ermöglicht eine einfache Versionskontrolle. Diese Versionierung erlaubt es, ältere Arbeitsstände wiederherzustellen und Projekte zu überarbeiten, ohne das Hauptprojekt zu gefährden. Diese Funktionen sind insbesondere für größere Teams von großer Bedeutung, weshalb auch die HTWD GitLab für ihre Projektverwaltung nutzt [71],[72].

DrawIO für die grafische Aufbereitung

Bei größeren Projekten kann die Übersichtlichkeit schnell verloren gehen, weshalb eine geeignete Software zur Visualisierung von Strukturen, Abläufen und Zusammenhängen notwendig ist. DrawIO, ein plattformunabhängiges Programm, bietet genau diese Möglichkeit. Mit DrawIO können Grafiken erstellt werden, um Abläufe und Strukturen klar und anschaulich darzustellen. Ein wesentlicher Vorteil dieses Werkzeugs ist seine Kompatibilität mit GitLab, was es besonders gut für den Einsatz in Teams geeignet macht. Auch das Hochschulteam nutzt bereits dieses Programm für ihre Projektarbeit [73],[74].

MATLAB als Programmierumgebung

Die Softwarekombination MATLAB und MATLAB Simulink, Version 2024a, von MathWorks spielt eine zentrale Rolle im Forschungsbereich der HTWD. MATLAB ermöglicht umfassende Berechnungen und Auswertungen, während Simulink zur Erstellung und Durchführung umfangreicher Simulationen eingesetzt wird. Ein Vorteil von Simulink ist das einfache, erweiterbare und austauschbare Blocksystem. Dieses modulare Design ermöglicht eine flexible Anpassung und Erweiterung von Simulationsmodellen. Ergänzend dazu bietet MathWorks eine Vielzahl von Zusatzmodulen, sogenannten Toolboxen, an. Diese Toolboxen werden entweder direkt von MathWorks oder von anderen Firmen entwickelt und bereitgestellt. Eine Übersicht der verwendeten Toolboxen ist in der Anlage D.1 zu finden [75],[76],[77].

Blender für die Bearbeitung von 3D-Modellen

Blender ist eine der führenden Open-Source-Softwarelösungen für die Erstellung komplexer, dreidimensionaler Animationen und Modelle. Die Software findet sowohl im Bereich der Videoproduktion als auch bei der Entwicklung von Spielen Einsatz. Blender ermöglicht zudem die Erstellung und Anwendung von Texturen und Materialien auf dreidimensionale Objekte, was eine detaillierte und realistische Gestaltung von 3D-Modellen ermöglicht [78].

Unreal Engine als Simulationsumgebung

Die Unreal Engine von Epic Games ist primär für die Entwicklung von Videospielen konzipiert. Aufgrund ihrer physikalischen Simulationen und umfangreichen Einstellmöglichkeiten bietet sie jedoch auch eine hervorragende Grundlage für Simulationsumgebungen. Neben MATLAB, das eine Verknüpfung der Version 5.1 der Engine mit Simulink-Modellen ermöglicht, wird die Unreal Engine auch in anderen Entwicklerwerkzeugen verwendet. Ein Beispiel ist CARLA, ein führender Open-Source-Simulator für autonomes Fahren, der ebenfalls auf dieser Software basiert [79],[80],[81],[82].

D. Verwendete MATLAB Simulink Toolboxen

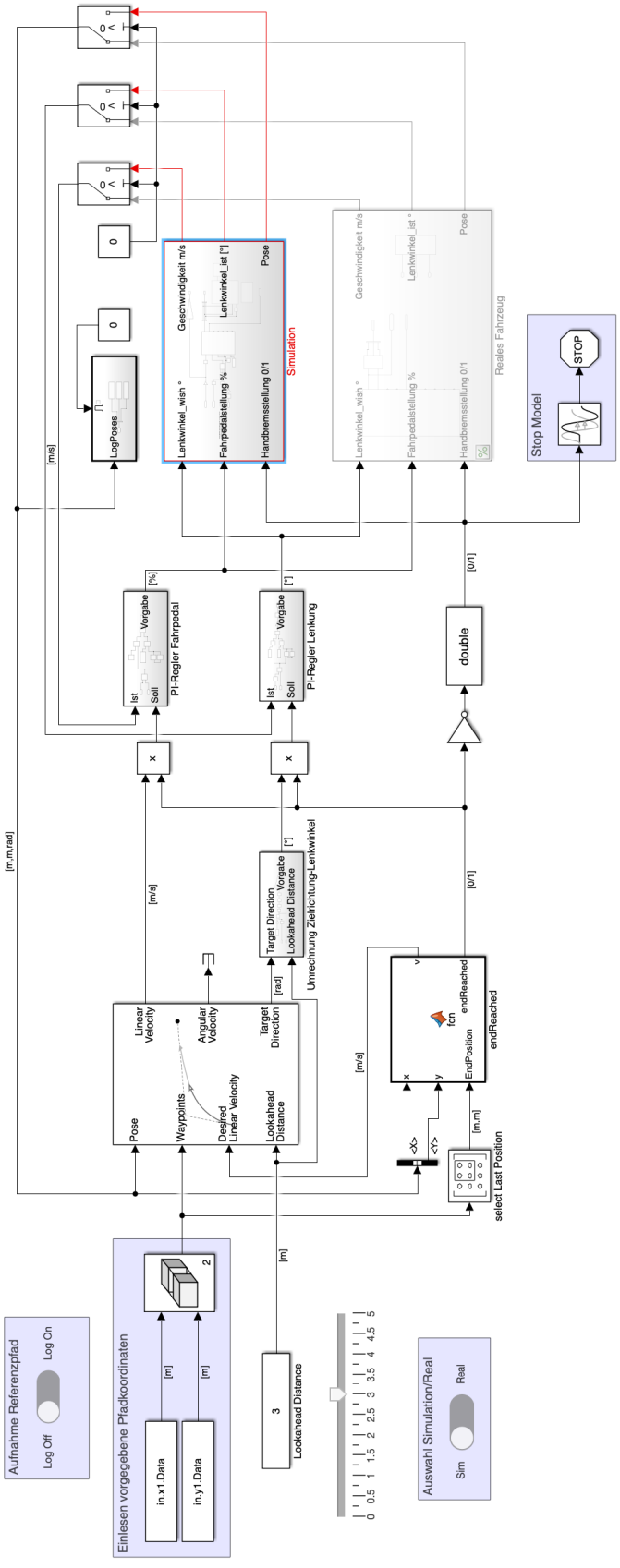
Toolbox	Beschreibung	Einsatzgebiet
Audio Toolbox	Biete Werkzeuge für die Verarbeitung und Analyse von Audiosignalen [83].	Verwendet im Modell der automatischen Notbremse für die Ausgabe von Hinweistönen.
Automated Driving Toolbox (& Interface for Unreal Engine Projects)	Stellt Werkzeuge und Algorithmen bereit die für die Entwicklung, Simulation und Tests von Anwendungen im Bereich autonomes Fahren relevant sind [84],[85].	Verwendet für die Simulationsbausteine.
Computer Vision Toolbox	Enthält Algorithmen und Apps die für den Entwurf und zum Testen von Computer-Vision-Systemen relevant sind. Beispielsweise sind Objekterkennungs- und Objektverfolgungs-Algorithmen Bestandteil [86].	Benutzt in den Simulationsbausteinen für die Videoausgabe der Szene.
DSP System Toolbox	Bietet für den Bereich Signalverarbeitung verschiedene Entwicklungs-, Simulations- und Analysewerkzeuge [87].	Benötigt für den Betrieb der Audio Toolbox.
Image Processing Toolbox	Bietet Algorithmen und Anwendungen zur Verarbeitung, Analyse und Visualisierung von Bildmaterial [88].	Benötigt für den Betrieb der Computer Vision Toolbox.
Lidar Toolbox	Beinhaltet Algorithmen und Anwendungen für den Bereich LiDAR. Beispielsweise für die Entwicklung und zur Analyse [89].	Benutz im Modell der automatischen Notbremse zur Darstellung und Verarbeitung der LiDAR-Daten
Lidar Toolbox Support Package for Ouster Lidar Sensors	Ermöglicht die Einbindung von Ouster Sensoren in MATLAB [90].	Benutzt im Modell der automatischen Notbremse zur Anbindung des Ouster Sensors.

Fortsetzung auf nächster Seite

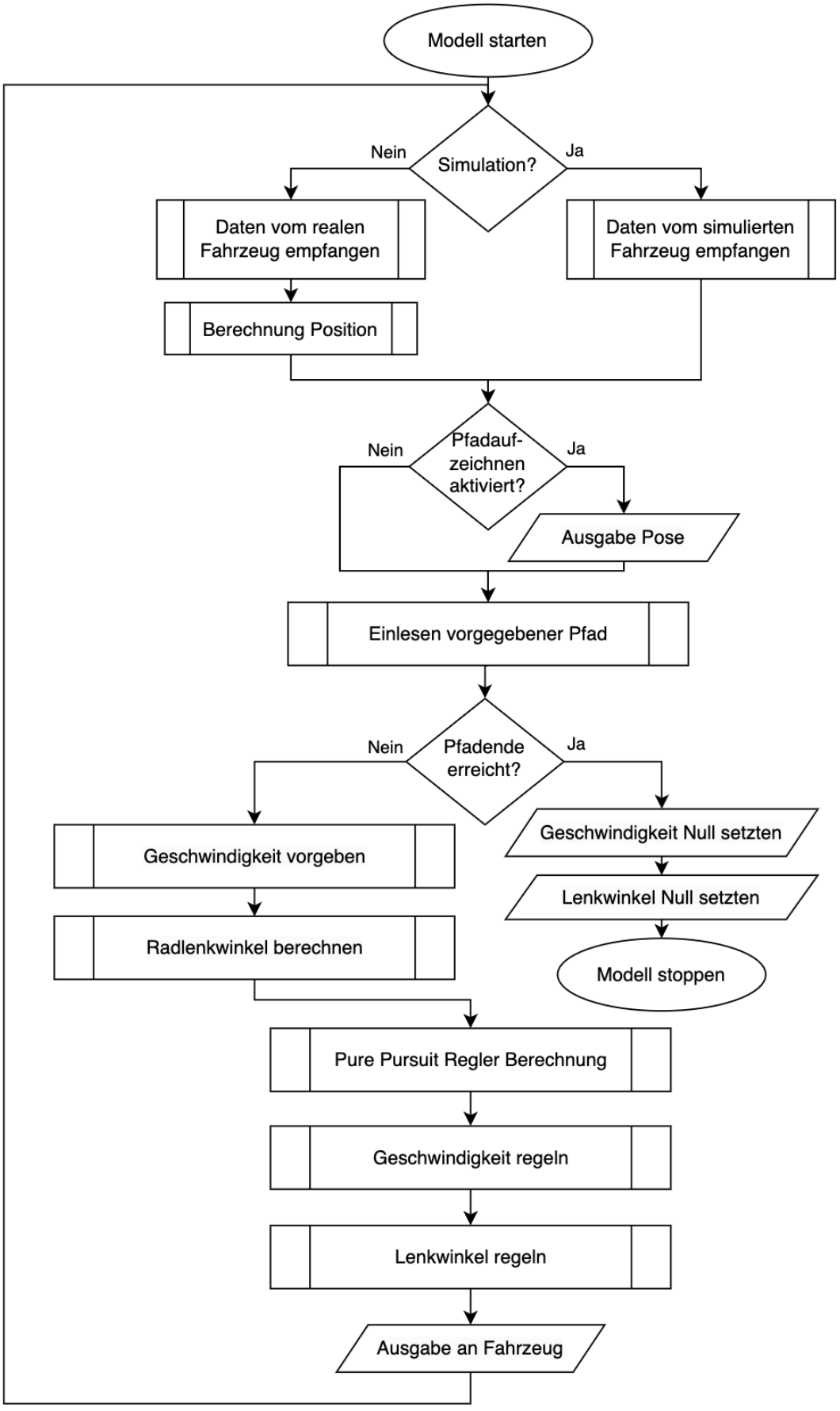
Toolbox	Beschreibung	Einsatzgebiet
Navigation Toolbox	Besitzt Algorithmen und Werkzeuge für die Bewegungsplanung, Kartierung und Positionsbestimmung. Ebenso sind Pfadplanungs- und Validierungsfunktionen Bestandteil [91].	Im Modell des Pfadverfolgers, zur Verfolgung des Pfades mit Hilfe des Pure Pursuit Reglers, benutzt.
Sensor Fusion and Tracking Toolbox	Umfasst Algorithmen und Werkzeuge für die Kombination multipler Sensordateneingänge [92].	Benutzt für die Sensor Fusion im Modell der automatischen Notbremse.
Signal Processing Toolbox	Beinhaltet Funktionen und Apps für Signale. Beispielsweise für die Verwaltung und Analyse oder auch zur Visualisierung [93].	Benutzt für Grundlagenfunktionen in beiden Simulink-Modellen
Simulink 3D Animation	Ermöglicht die Anbindung an die UE und bietet Funktionen zur Erstellung individueller virtueller Szenen [94].	Benutzt zur Umsetzung der Simulationsblöcke, speziell für die Anbindung an die UE, sowie zur Simulation von Passanten und des LiDAR-Sensors im Notbrems-Modell.
Vehicle Dynamics Blockset (& Interface for Unreal Engine Projects)	Bietet Funktionen und Anwendungen zur Simulation von Kraftfahrzeugen und deren Komponenten [95],[96].	Verwendet für die physikalische Simulation des virtuellen Fahrzeugs.
Vehicle Network Toolbox	Enthält Funktionen und Anwendungen für die Kommunikation mit Fahrzeugnetzwerken. Beispielsweise das Empfangen,(De-)Kodieren und Senden von CAN-Nachrichten [97].	Verwendet für das Empfangen und Senden der CAN-Botschaften vom und zum realen Fahrzeug.

E. Modelle

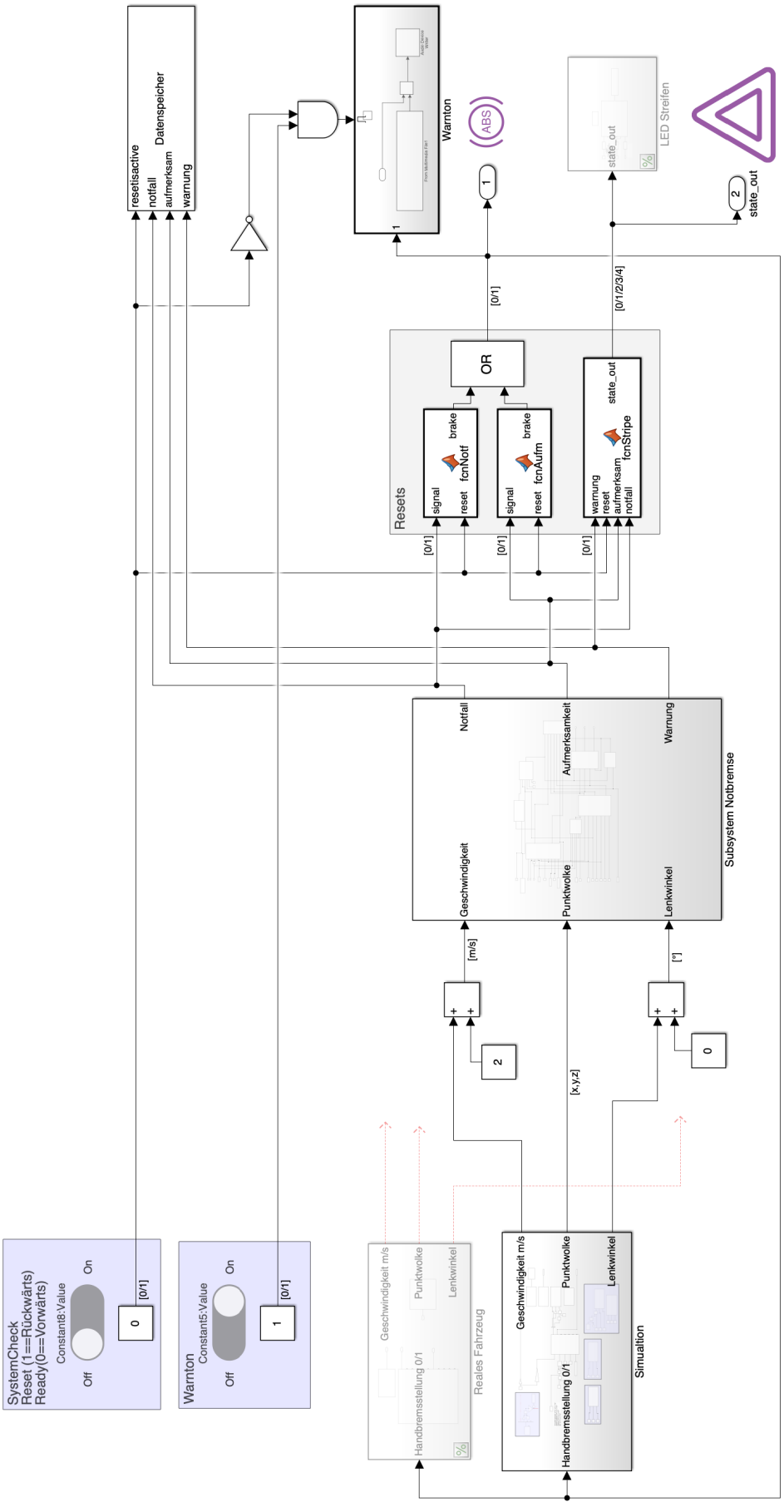
E.1. Simulink Modell: Pfadverfolger



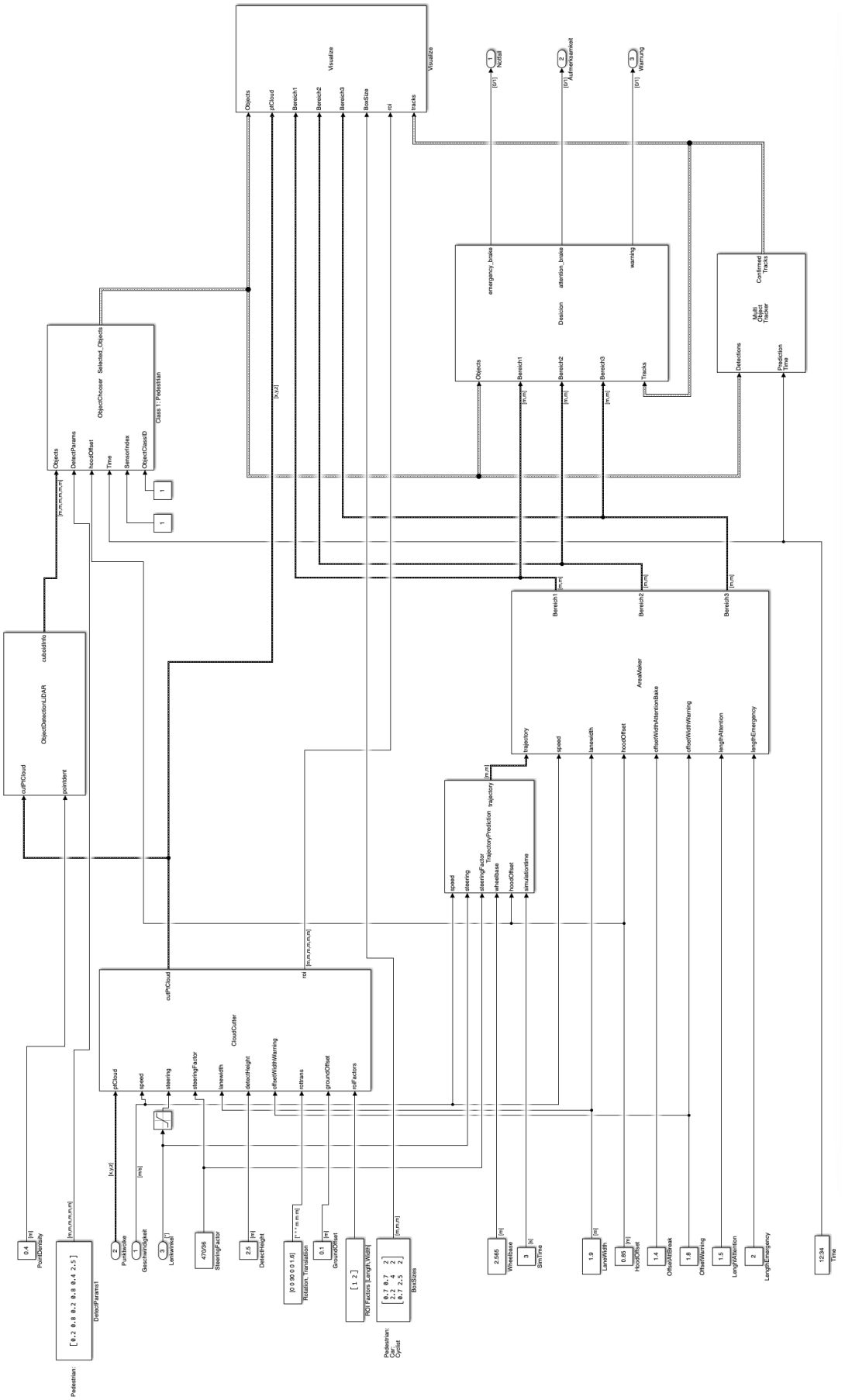
E.2. Ablaufplan: Pfadverfolger



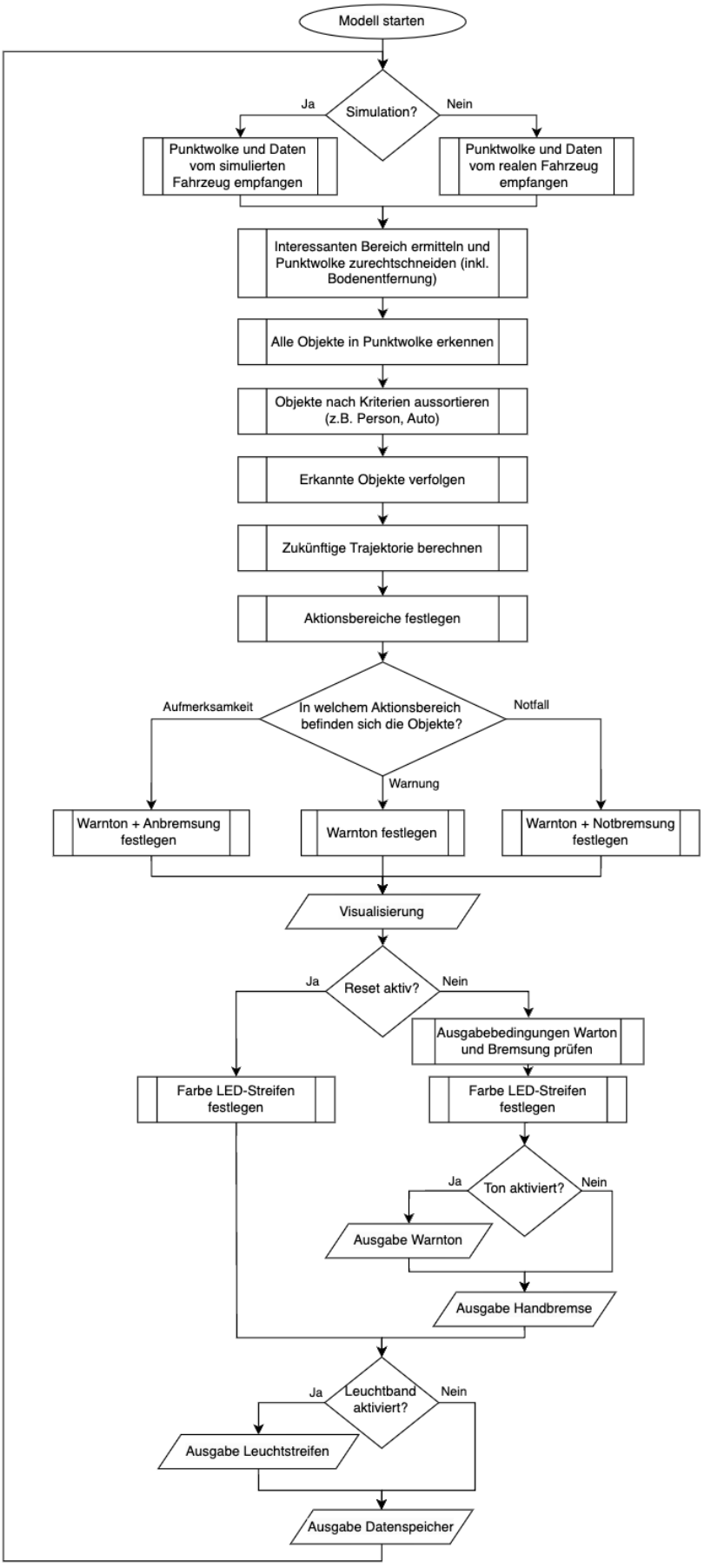
E.3. Simulink Modell: Automatische Notbremse



E.4. Hauptuntersystem der automatischen Notbremse



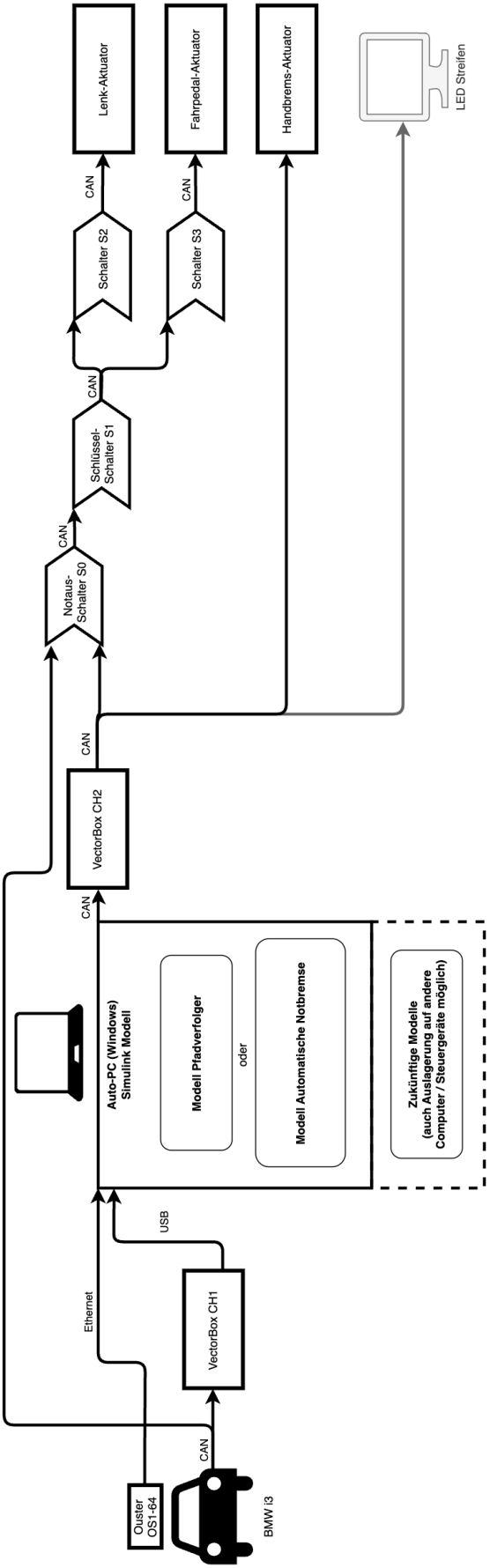
E.5. Ablaufplan: Automatische Notbremse



E.6. Quellcode: MATLAB System Block ObjectChooser.m

```
classdef ObjectChooser < matlab.System

    %Eigenschaften die nicht veränderbar sind während der Laufzeit
    properties (Nontunable)
        SampleTime = 0.1;
    end

    %Eigenschaften setzten
    methods
        function obj = DesicionBus(varargin)
            setProperties(obj,nargin,varargin{:})
        end
    end

    %Hauptprogramm
    methods (Access = protected)

        function [Selected_Objects] = stepImpl(~, Objects, DetectParams,...
                hoodOffset, Time, SensorIndex, ObjectClassID)

            obj1 = [0 0 0];

            %Objekte auswählen die vorgegebenen Parametern entsprechen
            for i = 2:height(Objects)
                if (Objects(i,4)>DetectParams(1,1) && Objects(i,5)>DetectParams(3,1) && Objects(i,6)>DetectParams(5,1) && ...
                        Objects(i,4)<DetectParams(2,1) && Objects(i,5)<DetectParams(4,1) && Objects(i,6)<DetectParams(6,1) && ...
                        Objects(i,2)>hoodOffset)
                    obj1 = [obj1; Objects(i,1:3)];
                end
            end

            %Objekte auf BUS schreiben
             num=0;
             for i=1:20
                detectionElement.Time = Time;
                try
                    detectionElement.Measurement = obj1(i+1,1:3)'; %Geschwindigkeit von Objekt kann hier zusätzlich gesetzt werden
                    num=num+1;
                catch
                    detectionElement.Measurement = [0 0 0]';
                end
                detectionElement.MeasurementNoise = eye(3);  %Beispiel, Anpassung möglich
                detectionElement.ObjectClassID = ObjectClassID;  %Beispiel, Anpassung möglich
                detectionElement.SensorIndex = uint32(SensorIndex);%Beispiel, Anpassung möglich

                %detectionElement zu detectionStruct schreiben
                detectionStruct.Detections(i) = detectionElement;
             end

             %Ausgänge beschreiben
             detectionStruct.NumDetections = uint32(num);
             Selected_Objects = detectionStruct;
        end

    end

    %Einstellungen
    methods(Static, Access = protected)

        %Statische Eigenschaften
        function simMode = getSimulateUsingImpl
            simMode = 'Interpreted execution';
        end

        function sts = getSampleTimeImpl(obj)
            sts = createSampleTime(obj, 'Type', 'Discrete', 'SampleTime', obj.SampleTime);
        end

        %Variable Eingänge erlauben
        function flag = isInputSizeMutableImpl(~, ~)
            flag = true;
        end

        function inType = getInputDataTypeImpl(~,~)
            inType = 'Bus: detectionBus';
        end

        %Ausgänge festlegen
        function [o1] = getOutputSizeImpl(~)
            o1 = [1 1];
        end

        function [o1] = getOutputDataTypeImpl(~)
            o1 = 'Bus: detectionBus';
        end

        function [o1] = isOutputComplexImpl(~)
            o1 = false;
        end

        function [o1] = isOutputFixedSizeImpl(~)
            o1 = true;
        end
    end
end
```

E.7. Erweiterte Modifikationsstruktur des BMW i3



F. Prüfkatalog

Prüfkatalog für autonome Fahrfunktionen

Fahrzeug: BMW i3

Betriebsbereich: Tests auf dem Testfeld der HTW Dresden

Inhaltsverzeichnis

Abkürzungsverzeichnis

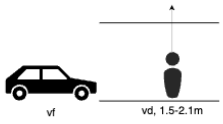
ad = Abstand Dummy
vd = Geschwindigkeit Dummy
vf = Geschwindigkeit Fahrzeug

II

Automatische Notbremse (LiDAR)

#1 Erwachsener quert die Fahrbahn

Versuchsaufbau:



Dieser Test soll das Notbremsverhalten der automatischen Notbremse validieren.

Validierung durch Simualtion:

*Der Grundlegende Versuch ist realisierbar durch Simulationen in Unreal Engine.

Verzögerung: 8 m/s^2

Temperatur: 10-40°C

Witterung: Trocken

Erwartetes Verhalten:

	Parameterkonstellation		
	vf=15 km/h	vf=20 km/h	vf=30 km/h
vd=0 km/h	Warnen + Bremsen (0 km/h)	Warnen + Bremsen (0 km/h)	Warnen + Bremsen (0 km/h)
⋮	...	...	...

Wiederholungen: Jeweils 3

Bestanden wenn: Bei allen Tests erst gewarnt und dann bis zum Stillstand gebremst wurde.
Kein Dummy vom Fahrzeug erfasst wurde.

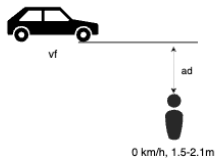
*Annahme, muss in Auswertung noch bestätigt werden.

Seite 1

Automatische Notbremse (LiDAR)

#2 Erwachsener neben der Fahrbahn

Versuchsaufbau:



Dieser Test soll das Verhalten des Aufmerksamkeitsbereiches der automatischen Notbremse validieren.

Validierung durch Simualtion:

*Der Grundlegende Versuch ist realisierbar durch Simulationen in Unreal Engine.

Verzögerung: 8 m/s^2

Temperatur: 10-40°C

Witterung: Trocken

Erwartetes Verhalten:

Parameterkonstellation			
	vf=15 km/h	vf=20 km/h	vf=30 km/h
ad=1 m	Warnen + 1x Anbremsen	Warnen + 1x Anbremsen	Warnen + 1x Anbremsen
...	...	...	...

Wiederholungen: Jeweils 3

Bestanden wenn: Bei jedem Test gewarnt und 1x angebremst wurde.

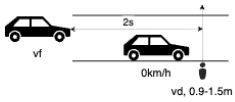
*Annahme, muss in Auswertung noch bestätigt werden.

Seite 2

Automatische Notbremse (LiDAR)

#3 Kind rennt hinter parkendem Fahrzeug vor

Versuchsaufbau:



Dieser Test soll das Notbremserhalten der automatischen Notbremse validieren.

Validierung durch Simualtion:

*Der Grundlegende Versuch ist realisierbar durch Simulationen in Unreal Engine.

Verzögerung: 8 m/s^2

Temperatur: 10-40°C

Witterung: Trocken

Erwartetes Verhalten:

Parameterkonstellation			
	vf=15 km/h	vf=20 km/h	vf=30 km/h
vd=15 km/h	Warnen + Bremsen (0 km/h)	Warnen + Bremsen (0 km/h)	Warnen + Bremsen (0 km/h)
...	...	...	...

Wiederholungen: Jeweils 3

Bestanden wenn: Bei allen Tests erst gewarnt und dann bis zum Stillstand gebremst wurde.
Kein Dummy vom Fahrzeug erfasst wurde.

*Annahme, muss in Auswertung noch bestätigt werden.

Seite 3

G. Datenträger

G.1. GitLab-Speicher

`https://gitlab.com/htw_nives/test_technology_center/extended-test-of-an-autonomous-vehicle`

Der Zugang zum GitLab-Verzeichnis kann bei den Betreuern angefordert werden.

G.2. SD-Karte