

HERMANN STEFFAN, BSc

---

**Implementierung eines Systems zur autonomen  
Trajektorienfahrt von Fahrzeugen anhand von  
DGPS Lokalisierung**

---

MASTERARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

Masterstudium Telematik

eingereicht an der

TECHNISCHEN UNIVERSITÄT GRAZ

**BETREUER:**

Univ.-Prof. DI Dr. techn. Martin Horn, MSc

Institut für Regelungs- und Automatisierungstechnik

25. Januar 2016

## **Zusammenfassung**

Aktive Sicherheit spielt im Automobilbereich eine immer wichtigere Rolle. Fahrzeughersteller entwickeln seit Jahren neue Assistenzsysteme, die den Lenker unterstützen. Durch die Entwicklung von selbständigen Bremsassistenten konnte die Gefahr von Auffahr- und Fußgängerunfällen deutlich reduziert werden. Um eine internationale Standardisierung der Testszenarien dieser Systeme zu erreichen, haben Firmen wie die Dr. Steffan Datentechnik GmbH Roboter entwickelt, die gefährliche Situationen nachstellen und beliebig oft wiederholen können. Bei den aktuellen Systemen fehlt jedoch bisher eine standardisierte Bewegung der Testfahrzeuge. Um eine Standardisierung gewährleisten zu können, müssen die Testfahrzeuge autonom gesteuert werden können. Damit ein sicheres und zuverlässiges autonomes Fahren verwirklicht werden kann, müssen Regelalgorithmen gefunden werden, die sich für die Lenkwinkelregelung eignen. Dazu sind diese sowohl auf ihre positiven, als auch negativen Eigenschaften zu analysieren. Um eine richtige Evaluierung der Algorithmen durchführen zu können wurde eine Testumgebung entwickelt. Anhand der dadurch gewonnenen Daten wird ein System ausgewählt, das in ein autonomes aktives Sicherheits-Testsystem integriert werden kann.

## **Abstract**

Active Safety plays a particular important role in automotive industry. Different Advanced Driver Assistance Systems (ADAS) have been established in the past, in order to support the drivers. The development of collision avoidance systems helps to prevent dangers from rear-end collisions and pedestrian accidents. In order to achieve international standardization of the test scenarios of these systems, companies such as Dr. Steffan Datentechnik GmbH developed robots for the arbitrary repeatable simulation of dangerous situations. These systems currently lack a way of standardized movements of the test vehicles. To provide this possibility, the steering of the test vehicles needs to be implemented completely autonomously. For realization of a safe and reliable autonomous driving, control algorithms have to be found, that are suitable for the steering angle control. Thus, an evaluation of the positive and negative properties of the different control algorithms is necessary. For this analysis, a test environment was developed. Based on the collected data, a system was chosen that is valuable for the integration in an autonomous active-safety test system.

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>7</b>
1.1	Motivation . . . . .	7
1.2	Aktive Sicherheit . . . . .	8
1.3	Testen von akt. Systemen . . . . .	8
1.4	Aufgabenstellung . . . . .	11
<b>2</b>	<b>Stand der Technik</b>	<b>12</b>
2.1	Bestehende Komponenten . . . . .	12
2.1.1	Ultraflat overrunable Robot (UFO) . . . . .	12
2.1.2	Regelungsbezogene Hardware . . . . .	14
2.1.3	Hardware zur Positionsbestimmung . . . . .	15
2.2	PC-Crash . . . . .	16
2.3	PCAN-USB . . . . .	17
<b>3</b>	<b>Methoden</b>	<b>19</b>
3.1	Anforderungen an die Regelung . . . . .	19
3.2	Datenübertragung durch Controller Area Network (CAN)-BUS . . . . .	19
3.2.1	Grundlagen des CAN-BUS . . . . .	20
3.3	Entwicklungsumgebung . . . . .	21
3.3.1	Simulationsumgebungen . . . . .	22
3.3.2	Regelungsumgebung . . . . .	23
3.3.3	PC-Crash Realtime-Simulation-CAN-BUS-Interface . . . . .	25
3.4	LMPC . . . . .	28
3.4.1	Eigenschaften, MP-Regelung . . . . .	30
3.4.2	Theoretische Grundlagen . . . . .	30
3.4.3	Modellbildung . . . . .	39
3.4.4	Implementierung . . . . .	41
3.5	PID Regelung . . . . .	43
3.5.1	Pure Pursuit . . . . .	43

3.5.2	Lookahead Steering . . . . .	46
3.6	Fuzzy-Regelung . . . . .	48
3.6.1	Theoretische Grundlagen . . . . .	48
3.6.2	Mathematische Grundlagen . . . . .	50
3.6.3	Implementierung . . . . .	53
<b>4</b>	<b>Ergebnisse</b>	<b>57</b>
4.1	Testumgebung . . . . .	57
4.2	Testszenarien . . . . .	61
4.3	Bewertungskriterien . . . . .	63
4.4	Simulationsergebnisse . . . . .	65
4.5	MPC . . . . .	66
4.6	Pure Pursuit . . . . .	76
4.7	Lookahead Steering . . . . .	86
4.8	Fuzzy-Regelung . . . . .	98
<b>5</b>	<b>Diskussion und Ausblick</b>	<b>108</b>
5.1	Vergleich . . . . .	108
5.2	Zusammenfassung . . . . .	114
5.3	Ausblick . . . . .	115
	<b>Literatur</b>	<b>116</b>

## EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Graz, am .....  
.....  
(Unterschrift)

## STATUTORY DECLARATION

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

.....  
date  
.....  
(signature)

# Abkürzungsverzeichnis

**UFO** Ultraflat overrunable Robot

**DSD** Dr. Steffan Datentechnik GmbH

**QP** Quadratisches Programm

**dGPS** Differential Global Positioning System

**MPC** modellbasierte prädiktive Regelung

**CAN** Controller Area Network

**SPS** speicherprogrammierbare Steuerung

**ARM** Acorn RISC Machines

**PC** Personal Computer

**CSMA/CR** Carrier Sense Multiple Access / Collision Resolution

**PID** Proportional-Integral-Derivative

# 1 Einleitung

## 1.1 Motivation

Im Jahr 2014 ereigneten sich insgesamt 37.957 Verkehrsunfälle in Österreich. Dazu zählen sowohl LKW- und PKW-Unfälle, als auch Unfälle mit Beteiligung von Motorrädern, Straßenbahnen, Zügen etc. Dabei wurden 47.670 Personen verletzt und 430 getötet. Unter den Verletzten waren 4.007 Fußgänger. Im Vergleich zum Vorjahr wurden um 5,5% weniger Verkehrstote registriert [1].

Die sinkende Tendenz kann zum einen durch eine verbesserte medizinische Versorgung und zum anderen durch aktive und passive Sicherheitssysteme erklärt werden. Wobei Passive Sicherheit die konstruktiven Maßnahmen an einem Fahrzeug beschreibt, die den Innenraum und damit die Insassen schützen sollen [2] und der Begriff Aktive Sicherheit, Systeme beschreibt, die aktiv in das Fahrverhalten eines Fahrzeuges eingreifen, um Unfälle zu vermeiden [3]. Wobei sich, aufgrund der allmählich begrenzten Möglichkeiten in der passiven Sicherheit, zunehmend der Bereich der aktiven Sicherheit als vielversprechend herausstellt [4].

Deshalb ist es auch notwendig, standardisierte Testmethoden für die Industrie zu definieren und weiter zu entwickeln, um die Effektivität dieser Sicherheitssysteme optimieren zu können und die Zuverlässigkeit zu erhöhen.

Aufgrund der Komplexität aktiver Systeme, ist die Überprüfung ebenfalls nur mit aufwendigen Testmethoden zu bewerkstelligen.

Insbesondere die Interaktion der verschiedenen Verkehrsteilnehmer (Fußgänger, Fahrradfahrer, Autofahrer, .. usw.) stellt eine große Herausforderung für die verwendeten Algorithmen dar und sollte deshalb in den Testszenarien berücksichtigt werden.

## 1.2 Aktive Sicherheit

Als aktive Sicherheit in der Fahrzeugtechnik versteht man Hilfssysteme, die aktiv in das Fahrverhalten eingreifen, um so Unfälle vermeiden zu können oder die Unfallfolgen zu reduzieren und den Fahrer vor gefährlichen Situationen zu schützen [3].

Speziell in den letzten Jahren wurden eine Reihe von Assistenzsystemen entwickelt, die den Fahrzeuglenker durch die Bereitstellung von Informationen und Handlungsanweisungen, oder gar durch Eingreifen in die ausgeführte Handlung, in gefährlichen Situationen unterstützen sollen [5].

Derartige Systeme sind beispielsweise das Elektronische Stabilitätsprogramm (ESP), das Antiblockiersystem (ABS), der Bremsassistent, der Abstandsregeltempomat usw.

Die wichtigsten Bereiche der aktiven Sicherheit werden in folgende Kategorien aufgeteilt:

- Fahrstabilität (ESP, ABS, ...)
- Konditionssicherheit (Adaptives Fahrwerk, ...)
- Wahrnehmungssicherheit (Nachtsichtkameras, ...)
- Bedienungssicherheit (Sprachsteuerung, ...)
- Kollisionsvermeidung (Not-/Bremsassistent, Fußgängererkennung, ...)

Aufgrund der zunehmenden Entwicklung im Bereich des autonomen Fahrens, gewinnt die aktive Sicherheit weiter an Bedeutung und wird in zukünftigen Fahrzeugen unverzichtbar sein. Dadurch nimmt die Anzahl der Systeme im Fahrzeug zu und eine Überprüfung der korrekten Funktion aller Systeme wird zu einer zunehmenden Herausforderung.

## 1.3 Testen von aktiven Sicherheitssystemen

Um aktive Sicherheitssysteme zu testen, ist es notwendig verschiedenste Straßenverkehrsszenarien nachzustellen. Die Firma Dr. Steffan Datentechnik GmbH (DSD) ([www.dsd.at](http://www.dsd.at)) hat dazu das sogenannte Ultraflat overrunable Robot (UFO) entwickelt. Das UFO ist eine mobile, nicht-holonome Roboterplattform, die als Versuchsträger für Dummyobjekte anderer Verkehrsteilnehmer im Testszenario dient. Robotertypen die als nicht-holonom bezeichnet werden, sind jene, die für ihre Bewegung nicht alle Freiheitsgrade



zur Verfügung haben. Diese Dummyobjekte sollen aus Sicht des Versuchsfahrzeuges die jeweiligen Verkehrsteilnehmer in ihrem Erscheinungsbild möglichst detailliert nachbilden.

Beispiele für derartige Objekte sind:




Beschreibung	Bild
Aufblasbare Fußgängerpuppen	
Schaumstoff-Fahrzeugmodelle	
Fahrräder	

Tabelle 1.1: Auflistung der Dummyobjekttypen

Da es im Testbetrieb auch zu Kollisionen kommen kann, ist es notwendig, dass diese Objekte einerseits die Testfahrzeuge möglichst nicht beschädigen und andererseits selbst kostengünstig zu reparieren bzw. zu ersetzen sind. Dies trifft auch für die UFO-Plattform selbst zu, daher wurde bei der Entwicklung dieses Roboters besonders darauf geachtet, dass eine vorgegebene Bauhöhe nicht überschritten wird. Dies ermöglicht ein problemloses Überrollen durch das Versuchsfahrzeug.

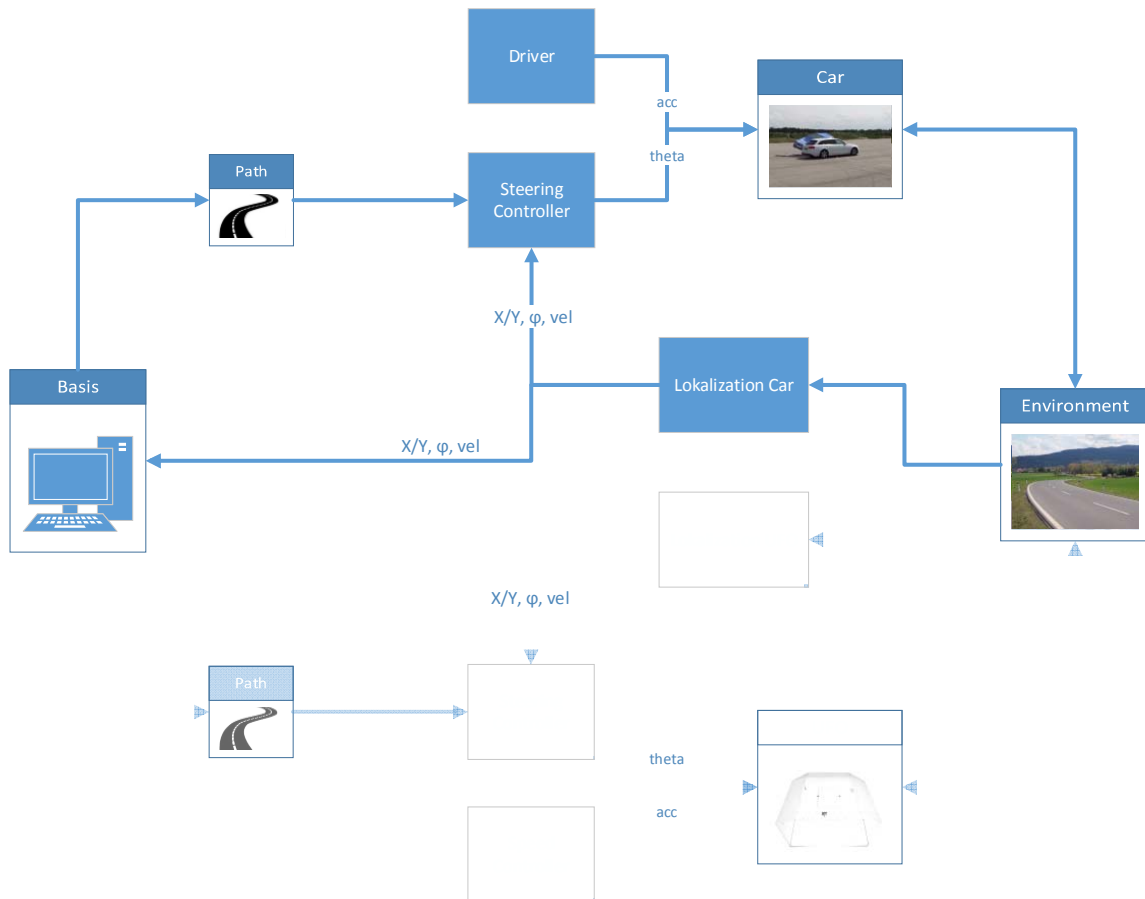


Abbildung 1.1: Aufbau des Gesamtsystems mit UFO

In Abbildung 1.1 ist der Aufbau des realisierten Projektes grafisch dargestellt. Die Funktionalitäten der aufgehellten Blöcke werden durch bestehende Module und die dazugehörigen Schnittstellen abgedeckt. Die restlichen Systemteile stellen die in dieser Masterarbeit realisierten Komponenten dar.

Die Basis des Systems ist ein Computer, bei dem alle Informationen der am Test-szenario beteiligten Objekte (UFOs und Fahrzeuge) zusammentreffen. Eine Verteilung der Daten über eine zentrale Einheit ist notwendig, um die Fahrzeuge beispielsweise in

Geschwindigkeit und Lenkwinkel synchronisieren zu können. Ebenfalls verbessert eine zentrale Bedieneinheit auch die Übersichtlichkeit des gesamten Systems.

Die Basiseinheit verteilt die Pfadpunkte an die Regelungssysteme der am Testszenario teilnehmenden Objekte. Über diese Pfadpunkte kann die Basis auch online das Verhalten der Testteilnehmer beeinflussen.

Wie in Abbildung 1.1 ersichtlich, ist in der ersten Phase des Projektes für den PKW eine manuelle Geschwindigkeitsvorgabe vorgesehen. Beim UFO wird die Geschwindigkeit über einen Algorithmus geregelt, der einerseits gewährleistet, dass das vorgegebene Geschwindigkeitsprofil umgesetzt wird und andererseits ermöglicht er eine Synchronisation der Bewegung des UFOs an die Fahrzeugbewegung. Das ist von besonderer Wichtigkeit, da gewährleistet sein muss, dass alle Testteilnehmer einen synchronisierten Bewegungsablauf durchführen, damit zu jedem Zeitpunkt die vordefinierten relativen Abstände eingehalten werden können.

Über eine Lokalisierungseinheit werden die Positionsdaten der Fahrzeuge und Objekte ermittelt. Beim UFO wird zur Lokalisierung ein D-GPS System verwendet. Dieses System ist auch für die Positionsermittlung des PKWs vorgesehen.

Die Positionsdaten werden sowohl an die Regelungseinheiten als auch an die Basisstation gesendet. Über die Basisstation kann eine Synchronisation und Beeinflussung aller Testteilnehmer vorgenommen werden.

## 1.4 Aufgabenstellung

Vorrangiges Ziel dieser Arbeit ist die Entwicklung eines robusten Systems, das es erlaubt, fahrende Objekte auf einem vorgegebenen Pfad autonom zu steuern. Voraussetzung für dieses System ist, dass es mit nur geringem Aufwand für unterschiedliche Fahrzeugtypen adaptiert werden kann. Dieses autonom fahrende Fahrzeug soll sich mit dem bereits entwickelten UFO synchronisieren. Dadurch können reale und synthetische Testszenarien kreiert und durchgeführt werden. Des Weiteren soll dies in ein schon bestehendes aktives Sicherheits-Testsystem der Firma DSD integriert werden.

Um ein geeignetes autonomes Lenkverhalten eines Fahrzeuges erreichen zu können, werden mehrere Regelungsarten gegenübergestellt und auf verschiedene Kriterien, unter anderem auf ihre Verwendbarkeit, Stabilität und Komplexität, untersucht.

## 2 Stand der Technik

### 2.1 Bestehende Komponenten

Da die Entwicklungen in das bestehende System der Firma DSD integriert werden sollen, wurde eine Studie der aktuellen und funktionsbereiten Komponenten durchgeführt [6]. Für die Hardwarekomponenten der Steuerung soll weitestgehend auf bereits bestehende Baugruppen des UFOs zurückgegriffen werden.

In den folgenden Unterkapiteln werden die für dieses Projekt angedachten Hardwarekomponenten genauer beschrieben.

**Anmerkung:** In dieser Arbeit wurden die Hardwarekomponenten durch Software-Interfaces zur Simulation ersetzt.

#### 2.1.1 Ultraflat overrunable Robot (UFO)

Das UFO ist eine mobile, nicht holonome Roboterplattform, deren Hauptaufgabe darin liegt, Dummyobjekte anderer Verkehrsteilnehmer in Testszenarien zu transportieren. Diese Objekte sollen aus Sicht des Versuchsfahrzeuges die jeweiligen Verkehrsteilnehmer in ihrem Erscheinungsbild möglichst detailliert nachbilden.



Abbildung 2.1: Top-Ansicht des UFO-Prototypen

Abbildung 2.1 zeigt den Aufbau eines UFOs. In der Mitte ist die Klappe erkennbar, die zum Batteriefach führt. Die Antenne dient der kabellosen Verbindung mit der Basisstation und die auf der Seite angehängten Rampen ermöglichen ein leichteres Überrollen des Roboters.

Aus dem Handbuch gehen folgende Kenndaten hervor [7]:

- 98mm Höhe
- Größe mit Rampen: 1800 x 1800mm
- 45km/h oder 70km/h max. Geschw.
- Max. Beschl.  $3m/s^2$  bzw.  $4m/s^2$
- Max. Verzögerung  $6m/s^2$  bzw.  $7.5m/s^2$
- IMU + dGPS
- Synchronisiertes Fahren

- Max. Zuladung 120kg
- Grafisches User-Interface mit der Möglichkeit der Aufzeichnung und Anzeige von Positionsdaten

Die geringe Höhe von nur 98 mm erschwert die Unterbringung der Motoren und der zur Steuerung benötigten Hardware enorm. Daher wurden nur geometrisch sehr kompakte Systeme verbaut. Dies ist jedoch mit höheren Kosten für die Hardware verbunden.

### 2.1.2 Regelungsbezogene Hardware

Es ist notwendig die wichtigen zur Regelung des Roboters benötigten Komponenten im Detail zu beschreiben. Dazu zählen der Steuerungscomputer und die Positionsbestimmungs-Hardware

#### OLIMEX STM32-E407

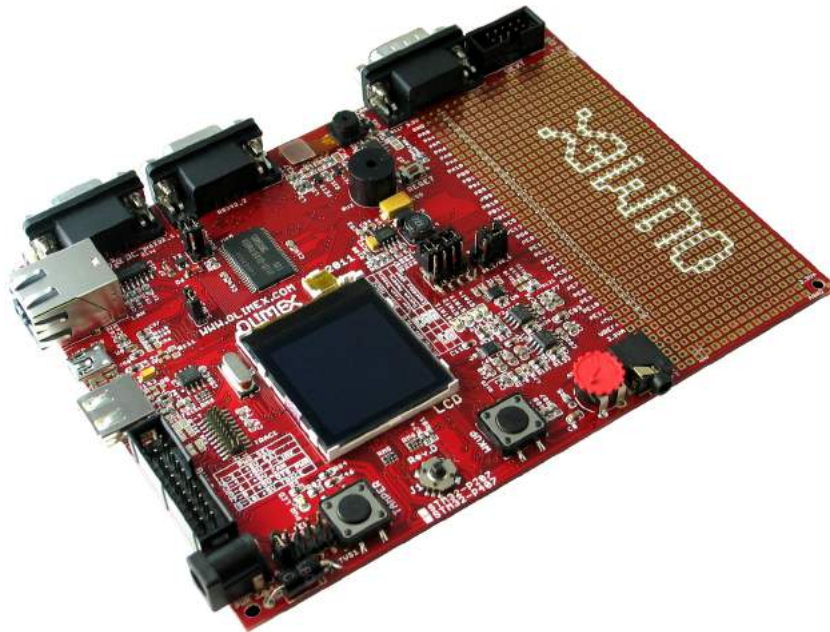


Abbildung 2.2: Foto einer Olimex Entwicklungsplattform

Die speicherprogrammierbare Steuerung (SPS) wird durch einen Acorn RISC Machines (ARM) Controller realisiert. Die OLIMEX STM32-E407 Entwicklungsplattform bietet laut Hersteller [8] folgende in Tabelle 2.1 ersichtliche Kenngrößen:

<b>Kenngröße</b>	<b>Wert</b>
Prozessor	Cortex-M4 210DMIP (ARM)
Speicher	1MB Flash
RAM	196KB
USB	USB On-The-Go HS
Ethernet	1
CAN Schnittstelle	2

Tabelle 2.1: Technische Daten zur Olimex-Plattform

### 2.1.3 Hardware zur Positionsbestimmung

#### OXFORD dGPS Plattform



Abbildung 2.3: Die gesamte Oxford-Plattform mit Koffer

Die diversen Differential Global Positioning System (dGPS) Module von Oxford bieten eine große Genauigkeit. Diese enthalten neben einem Differential Global Positio-

ning System (dGPS)-Empfänger auch eine Inertial Measurement Unit (IMU), mit der die Beschleunigungen in xyz-Richtung gemessen werden können. Des Weiteren ist ein Kalman-Filter implementiert, der eine zusätzliche Genauigkeit verspricht. Der Kalman-Filter ermöglicht, dass Messungen verschiedener Sensoren in Einklang gebracht werden können (Sensor Fusion). Er beinhaltet ein Modell des Sensorensystems und kann auf diese Weise eine zukünftige Position schätzen (Prädiktion). Erhält der Filter dann durch die Sensoren neue Daten, so kann er die Schätzungen mit Hilfe der tatsächlichen Daten verbessern (Korrektur) [9]. Über eine CAN- und Ethernet-Schnittstelle werden die Daten dieser Plattform an den Mikrocontroller übermittelt [10].

## 2.2 PC-Crash

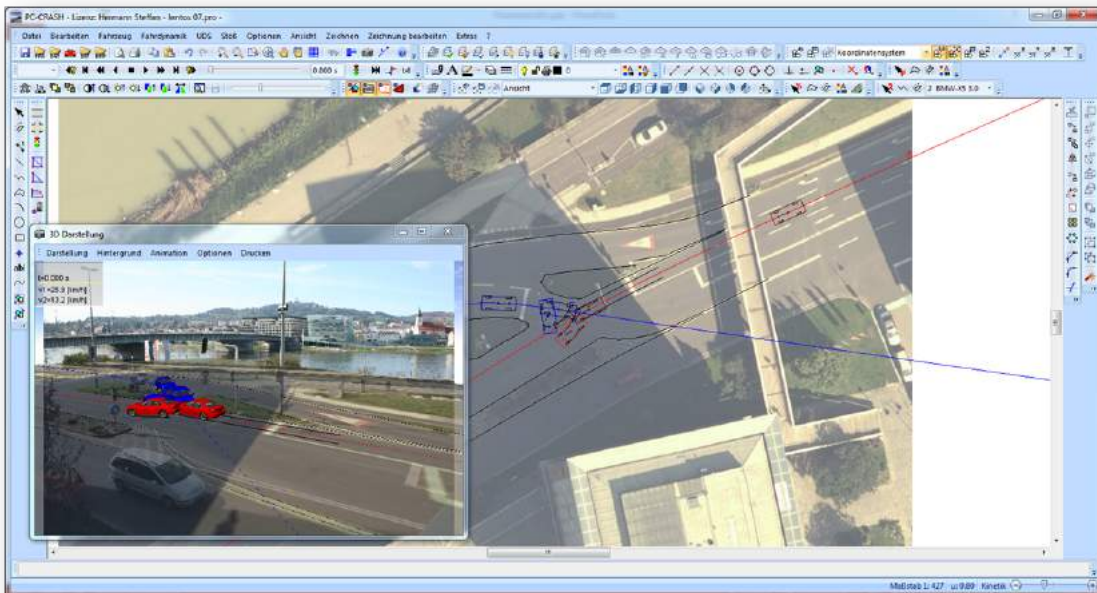


Abbildung 2.4: Screenshot aus PC-Crash

Zur Simulation wurde das Softwarepaket PC-Crash der DSD verwendet.

Eine der wichtigsten Vorteile in der Verwendung von PC-Crash liegt in der

».... Verwendung unterschiedlicher Berechnungsmodelle. Hierbei kann, ausgehend von einer einfachen kinematischen Simulation, jederzeit auf ein komplexeres Modell übergegangen werden. Alle vorher erfolgten Definitionen wer-



den in diesem Fall automatisch übernommen. ... es werden in PC-CRASH sämtliche Ergebnisse sofort grafisch und zahlenmäßig angezeigt. «[11]

PC-Crash ist mit mehr als 5000 Installationen weltweit [12] Marktführer im Bereich der Software für Verkehrsunfall-Simulationen.

Für diese Arbeit bietet sich PC-Crash besonders dahingehend an, als die oftmals validierten Simulationsmodelle in einem hohen Grad der Realität entsprechen [13]. So kann angenommen werden, dass Simulationsergebnisse ohne aufwendige Anpassungen in eine reale Umgebung transferiert werden können. Jedoch müssen auch die in der Realität vorhandenen Störeinflüsse untersucht werden. Einflüsse wie Ungenauigkeit von Daten in der Positionsbestimmung, Ausfall der Verbindung zum D-GPS, Latenzen in der Datenübertragung, unterschiedliche Fahrbahnverhältnisse etc. Des Weiteren bietet PC-Crash die Möglichkeit einer 3D-Visualisierung. Damit können die Ergebnisse und der dynamische Ablauf sowohl als Einzelbild als auch als Animation dargestellt werden.

## 2.3 PCAN-USB



Abbildung 2.5: PCAN-USB Adapter

PCAN-USB ist ein USB zu CAN Konverter. Das Gerät wird von der Firma PEAK-System Technik© die im Jahr 1999 gegründet wurde [14], hergestellt. Der Adapter

ermöglicht eine unkomplizierte Verbindung eines PCs mit diversen Geräten mit Hilfe einer CAN-Schnittstelle.

Aus der Produktbeschreibung gehen folgende technische Eigenschaften hervor [14]:

- Adapter für den USB-Anschluss (USB 1.1, kompatibel mit USB 2.0 und USB 3.0)
- Spannungsversorgung über USB-Übertragungsraten bis zu 1 Mbit/s
- Timestamp-Auflösung ca. 42  $\mu$ s
- Erfüllt die CAN-Spezifikationen 2.0A (11-Bit-ID) und 2.0B (29-Bit-ID)
- Anschluss an CAN-Bus über D-Sub, 9-polig
- NXP CAN-Controller SJA1000 mit 16 MHz Taktfrequenz
- NXP CAN-Transceiver PCA82C251
- 5-Volt-Versorgung am CAN-Anschluss durch Lötjumper zuschaltbar, z. B. für externe Buskonverter
- Erweiterter Betriebstemperaturbereich von -40 bis 85 °C

# 3 Methoden

## 3.1 Anforderungen an die Regelung

Eine Anforderung an die Regelung ist die Möglichkeit einer einfachen und schnellen Adaption der fahreugspezifischen Gegebenheiten. Des Weiteren ist es wichtig, ein Fahrzeug auch bei höheren Geschwindigkeiten möglichst exakt auf der Spur halten zu können, und ein Überschwingen bei schnellen Kursänderungen zu vermeiden.

Die Komplexität und der Rechenaufwand sind ebenfalls wichtige Kriterien bei der Beurteilung der untersuchten Algorithmen.

## 3.2 Datenübertragung durch CAN-BUS

Aufgrund des teilweise hohen Rechenaufwandes der Regelungsalgorithmen wird die Berechnung auf einem externen leistungsstarken Personal Computer (PC) durchgeführt. Daraus resultierend folgt das Problem der Datenübertragung zwischen PC und Steuerungseinheit (UFO Lenkaktuator, Lenkroboter in einem PKW, direkter Zugriff auf Servolenkung in PKW).

Die Steuerungseinheit in Verbindung mit einer Positionseinheit gibt die empfangenen Stellgrößen an die Aktuatoren weiter und liefert dem Regelalgorithmus seine Positionsdaten zurück.

Eine stabile Datenübertragung ist notwendig, um Datenverlust möglichst zu vermeiden und eine deterministische Datenübertragung zu gewährleisten. Erreicht ein Datenpaket den Regelungsalgorithmus mit zu großer Verzögerung, kann es für die Ermittlung der Stellgröße unter Umständen nicht mehr verwendet werden. Die daraus resultierende Ungenauigkeit könnte theoretisch durch eine Implementierung in Echtzeit eliminiert werden.

In der Praxis ist es jedoch eine Herausforderung, "harte Echtzeit" zu garantieren. Aus diesem Grund ergibt sich der Ansatz, zumindest die Kommunikationszeit soweit zu verringern, dass eine Verzögerung reduziert werden kann. In der Automobilbranche hat

sich als Übertragungskonzept der CAN-Bus bewährt. Aus diesem Grund wurde in dieser Arbeit auch auf dieses Konzept zurückgegriffen. Die Verwendung der CAN-Schnittstelle bietet bei der Steuerung eines PKW die Möglichkeit einer direkten Verbindung zum fahrzeugeigenen Netzwerk. Damit kann durch Vorgabe der entsprechenden CAN-Nachrichten die Lenkung direkt ohne weitere Hardwarkomponenten beeinflusst werden.

### 3.2.1 Grundlagen des CAN-BUS

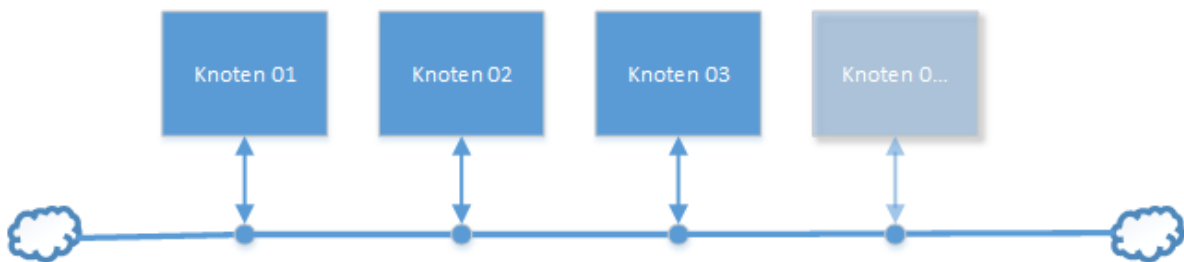


Abbildung 3.1: Aufbau eines einfachen Controller Area Netzwerkes

Der CAN-Bus wurde von BOSCH und INTEL entwickelt, um ein Kommunikations-Netzwerk mit serieller Datenübertragung zwischen einzelnen Steuergeräten in Fahrzeugen zu ermöglichen. Im Jahr 1986 wurde die Technik beim SAE-Kongress in Detroit vorgestellt [15]. Es wurde anfangs für die Automobilindustrie entwickelt, ist aber mittlerweile auch in der Automatisierungstechnik gängiger Standard.

Das CAN-Bussystem ist ein 2-Draht-Netzwerk, mit differentieller (CAN-High und CAN-Low) serieller Datenübertragung in definierten Nachrichtenformaten. Die Impedanz der Leitung beträgt  $120 \Omega$ .

In Abbildung 3.1 ist der schematische Aufbau eines Netzwerkes mit CAN-Knoten zu sehen. Jeder Knoten greift auf beide Leitungen zu. Ein Knoten besteht aus einem Zugangscontroller auf den BUS und einem Steuergerät, welches die empfangenen Daten an einen Aktuator weitergibt, oder Daten eines Sensors wieder an das Netzwerk sendet. Die Knoten empfangen immer gleichzeitig alle Nachrichten am BUS und können über den Identifikator der Nachricht, die Nachrichten filtern, die für den jeweiligen Knoten bestimmt sind.

Complete CAN Frame																																																						
Arbitration Field											Control			Data				CRC Field						End of Frame																														
Start of frame	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0	Request Remote	ID Ext. Bit	Reserved	DL3	DL2	DL1	DL0	DE7	DE6	DE5	DE4	DE3	DE2	DE1	DE0	CRC14	CRC13	CRC12	CRC11	CRC10	CRC9	CRC8	CRC7	CRC6	CRC5	CRC4	CRC3	CRC2	CRC1	CRC0	CRC Delimiter	Address Slot Bit	Address Delimiter	EOF8	EOF5	EOF4	EOF3	EOF2	EOF1	EOF0	FS2	FS1	FS0
	11											4			8				15																																			

Abbildung 3.2: Bitverteilung in CAN-Frame

Um gleichzeitiges Senden auf den BUS und somit Kollisionen bei der Nachrichtenübertragung zu vermeiden, wird die Technik der BUS-Arbitrierung (Carrier Sense Multiple Access / Collision Resolution (CSMA/CR)) verwendet. Arbitrierung ermöglicht die Vergabe von Prioritäten durch dominante Bits im Identifier einer Nachricht. In Abbildung 3.2 kann man erkennen, dass in einem CAN-Frame (bei CAN-Datentelegramm im Base Frame Format) 11 Bit für den Identifier der Nachricht reserviert sind, 8 Byte (64 Bit) sind für die Daten vorgesehen, und der 15 Bit große CRC-Frame dient als Prüfsummenfeld. Jeder BUS-Teilnehmer beobachtet während des Schreibens auf den BUS das Netzwerk. Kommt es zu einer Überschneidung mit einem dominanten Bit einer Nachricht eines anderen Teilnehmers, wird das Senden eingestellt. Sind die Identifier bei zwei Teilnehmern gleich, werden die weiteren Bits der Nachricht betrachtet. Es ist jedoch sinnvoll Nachrichten, soweit möglich, mit einzigartiger Identifizierung zu versehen [16].

### 3.3 Entwicklungsumgebung

In dieser Arbeit wurde ein vollständiges System zum Testen aktiver Sicherheitssysteme entwickelt. Dazu war es notwendig eine Entwicklungsumgebung zu entwerfen, die auf einfache Weise von einer Simulation in ein reales System transferiert werden kann.

In Abbildung 3.3 ist ein Blockdiagramm skizziert, das den Aufbau der in dieser Arbeit entwickelten Umgebung zeigen soll.

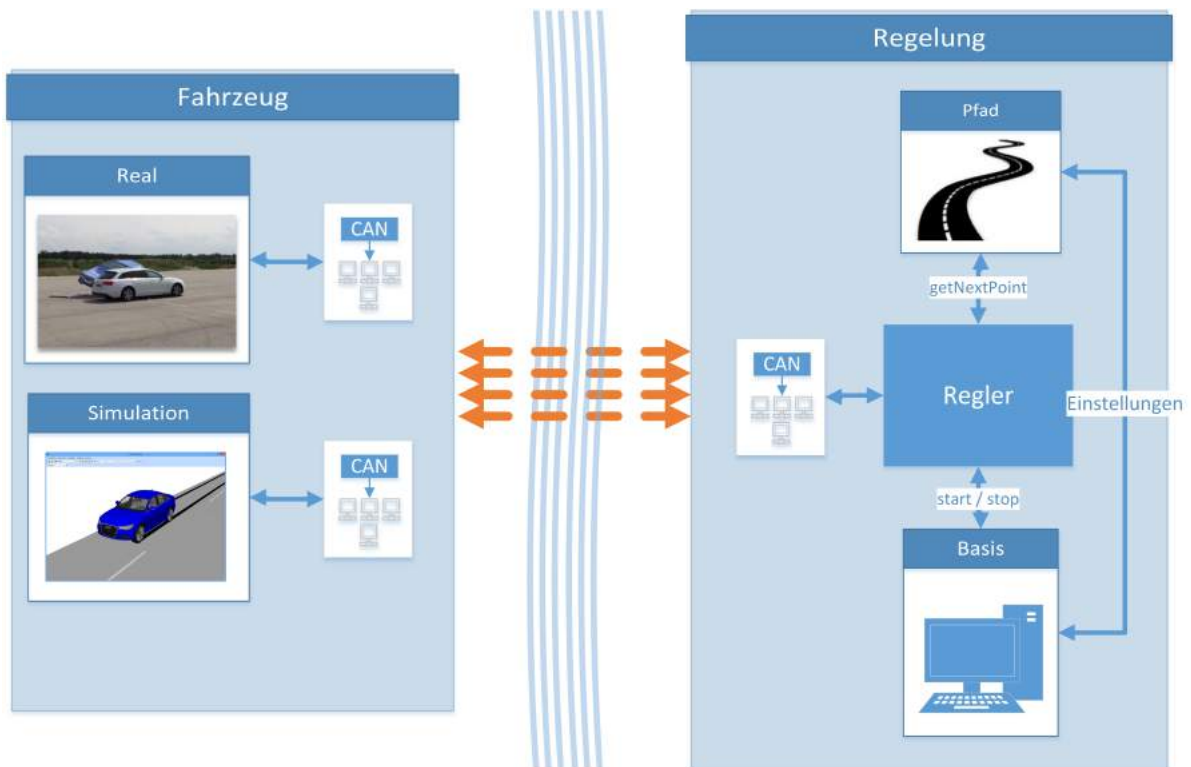


Abbildung 3.3: Aufbau der Entwicklungsumgebung

Es ist zu erkennen, dass die Schnittstelle zu realen Fahrzeugen (CAN-Schnittstelle) in gleicher Weise, wie die Simulationsschnittstelle vorgesehen wurde. So ist es in dieser Arbeit möglich, die Simulation mit der realen Welt gleich zu setzen.

Die gesamte Entwicklungsumgebung teilt sich in Simulationsumgebung, Regelungsumgebung und deren Verbindung durch CAN-BUS auf.

Aus Sicht der Regelung ergibt sich daher kein Unterschied zwischen Simulation und einem realen Fahrzeug.

Im Folgenden wird der fahrzeugseitige Ablauf der Regelung in der Simulationsumgebung PC-Crash dargestellt. Ohne Einschränkung der Allgemeinheit können diese Schritte auch auf andere Systeme, beispielsweise ein reales Fahrzeug, übertragen werden.

### 3.3.1 Simulationsumgebungen

Den wichtigsten Teil der Entwicklungsumgebung stellt die Simulationsumgebung dar. Hierbei wurde das zuvor im Kapitel "Stand der Technik" beschriebene Programm PC-Crash verwendet. PC-Crash kann zusammengefasst betrachtet, die realen Umweltbedingungen sehr gut darstellen. Es wurde in zahlreichen internationalen Publikationen

validiert und ist durch seine aufwendige 3D-Darstellung für die Vorstellungskraft des Entwicklers während des Entwicklungsprozesses sehr hilfreich.

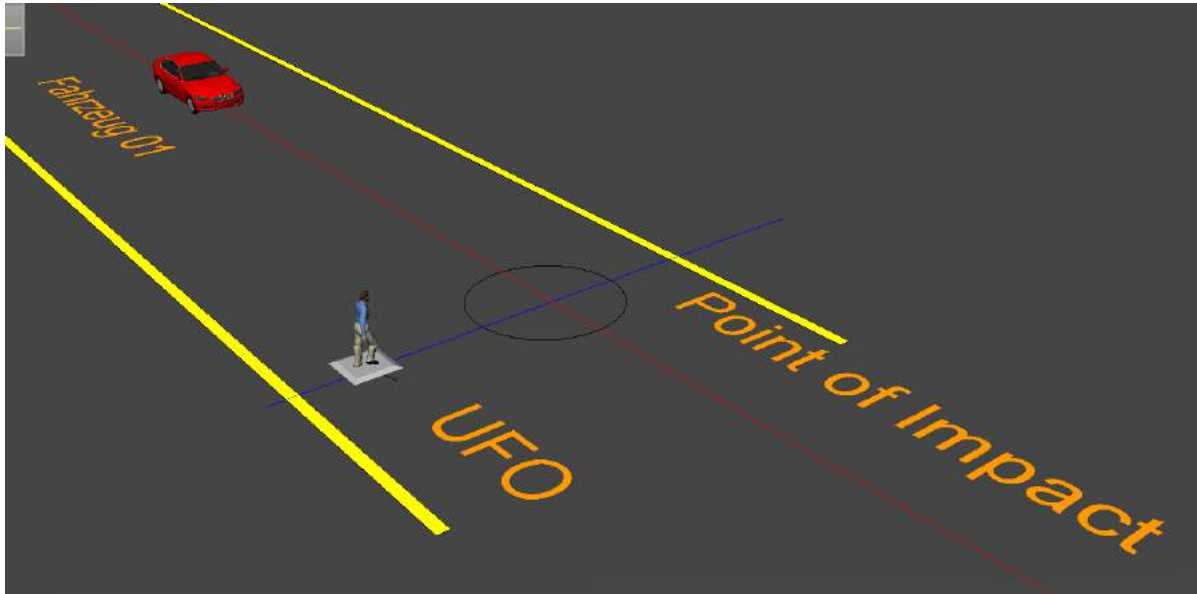


Abbildung 3.4: Szenario eines vom Gehsteig auf die Straße laufenden Fußgängers

Abbildung 3.4 zeigt eine 3D-Darstellung eines typischen Testes für aktive Sicherheitssysteme.

Durch diese Bilder ist ein Fehlverhalten des entwickelten Regelungssystems sehr schnell erkennbar.

Um nun die verschiedenen Regelungssysteme in einfacher und gleichbleibender Methodik entwickeln zu können, wurde eine Regelungsumgebung in C++ implementiert.

### 3.3.2 Regelungsumgebung

Um verschiedene Reglertypen umsetzen zu können, wurde ein System entwickelt, das es ermöglicht, den Regler ohne viel Aufwand im Software-Layer auszutauschen.

Die Regelungsumgebung wurde deshalb softwaretechnisch extrahiert und der Datenaustausch der einzelnen Reglersysteme standardisiert.

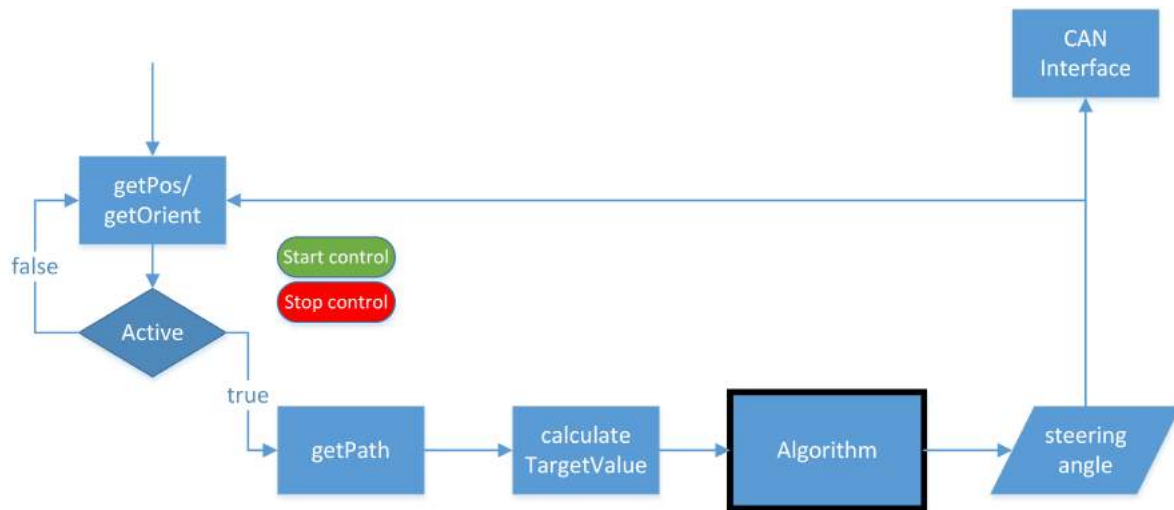


Abbildung 3.5: Blockdiagramm der Reglerimplementierungen

In Abbildung 3.5 kann man den extrahierten Reglerteil und den standardisierten Ablauf der Regelung innerhalb des Programmes erkennen.

Das Programm greift mit Hilfe der Funktionen **getPos** bzw. **getOrient** auf die momentanen Positions- und Richtungsgrößen des Fahrzeuges zu. Sobald die Funktion **startControl** ausgeführt wird, wird die Variable **Active** auf **true** gesetzt und der Regelalgorithmus gestartet.

Der Algorithmus fordert über **getPath** den Pfad an. Die Methode **calculateTargetValue** berechnet nun anhand der Position des Fahrzeuges und des Pfades die Soll-Werte für den Regelalgorithmus. Dieser berechnet die Stellgröße in Form des Lenkwinkels und speichert diesen in der Variablen **steeringAngle**. Mit Hilfe der Funktionalität des CAN-Übertragungsteils des Programms wird nun die Stellgröße an die Simulation übermittelt. Dieser Regelungsablauf wird nun solange wiederholt, bis die Funktion **stopControl** ausgeführt wird und so durch das Invertieren der Variable **Active**, der Algorithmus unterbrochen wird.

Diese Regelungsumgebung wurde in C++ realisiert. Abbildung 3.6 zeigt ein Klassendiagramm des objektorientierten Aufbaus dieser Software.



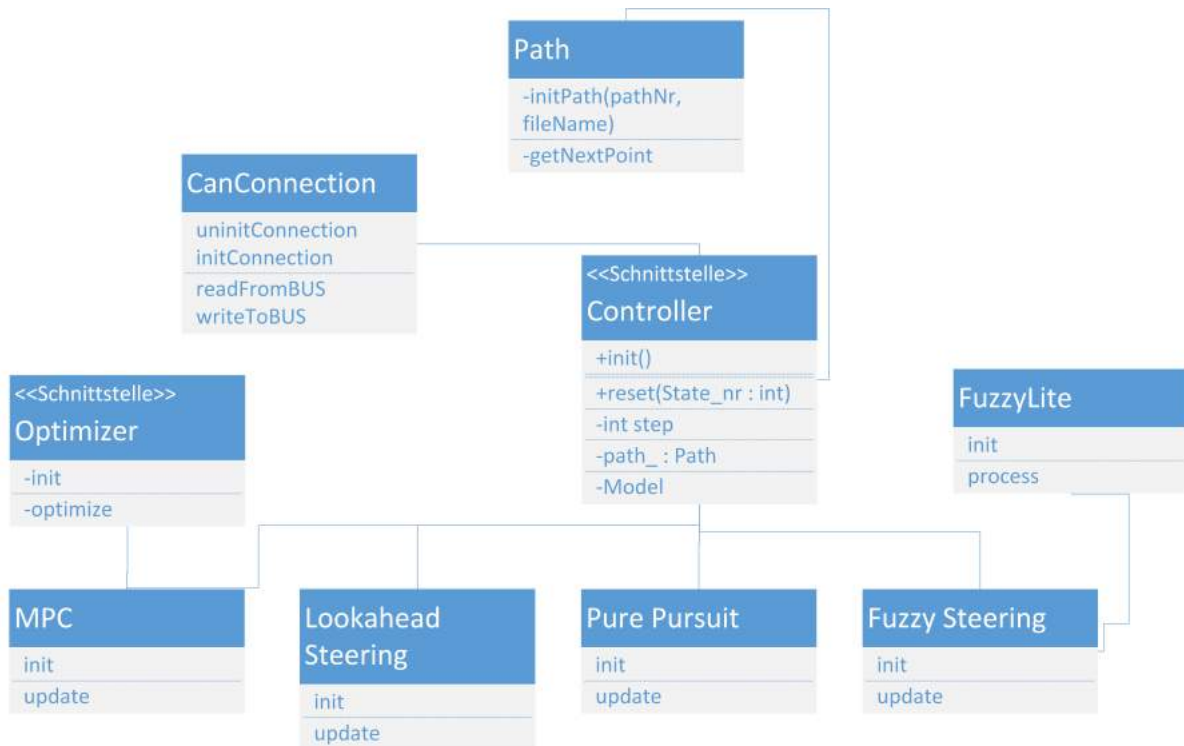


Abbildung 3.6: Klassendiagramm der C++ Regelungsumgebung

Die Klasse **Controller** bildet die Basisklasse für alle implementierten Reglertypen. Von dieser Basisklasse werden die **LookaheadSteering**, **PurePursuit**, **FuzzySteering** und **MPC** abgeleitet. Diese Klassen beinhalten die Grundfunktionalität jedes Reglers. Die Funktionen **init** und **update** werden von der Basisklasse vererbt.

### 3.3.3 PC-Crash Realtime-Simulation-CAN-BUS-Interface

Das PC-Crash Realtime-Simulation -CAN-BUS-Interface ist eine Erweiterung der Dynamik-Schnittstelle von PC-Crash, die einen Zugriff auf programminterne Funktionen zulässt. Das Softwarepaket rund um PC-Crash bietet viele Vorteile, die in Kapitel PC-Crash genauer beschrieben wurden.

Um diese Vorteile auch für die Evaluierung und Visualisierung von Regelalgorithmen zu nutzen, war eine physische Schnittstelle zu den berechneten Parametern notwendig. Zu diesem Zweck wurde eine CAN-Schnittstelle implementiert, die den Datenaustausch via CAN-BUS ermöglicht. Um auch die tatsächliche Performance des BUS-Systems evaluieren zu können, war es notwendig die Simulation in Echtzeit durchzuführen. Diese Funktion wurde ebenfalls in dieser Software-Hardware Schnittstelle umgesetzt.

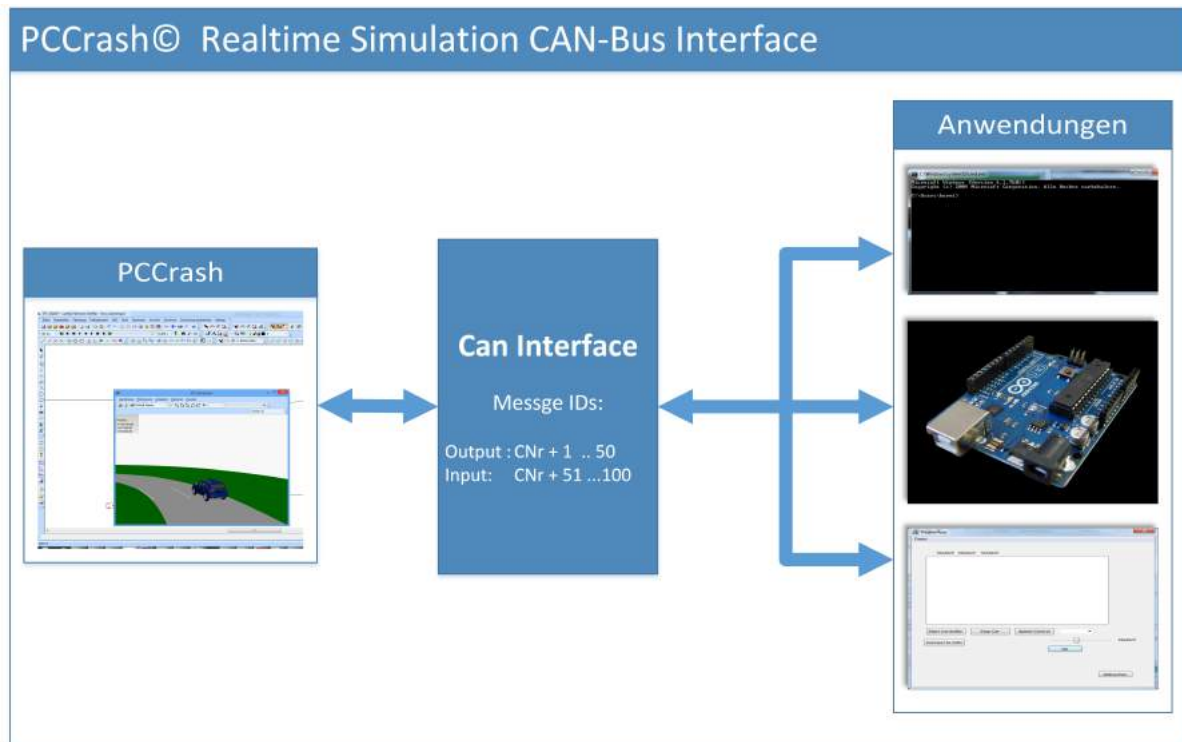


Abbildung 3.7: Schematischer Aufbau des PC-Crash Realtime-Simulation CAN-Bus Interface

Abbildung 3.7 zeigt nun den genauen Aufbau des Interfaces. In jedem Simulationsschritt werden die definierten Daten aus PC-Crash an den BUS gesendet. Eine Anwendungssoftware, das kann ein Regelalgorithmus oder ein Überwachungssystem sein, empfängt die Daten vom BUS, verarbeitet diese und sendet im Fall einer Regelung Stellgrößen wiederum zurück auf den BUS, damit PC-Crash diese verarbeiten kann.

Die Zugehörigkeit der Nachrichten wird über die ID-Vergabe der CAN-Nachrichten geregelt. In Abbildung 3.8 wird der Mechanismus der ID-Vergabe dargestellt.

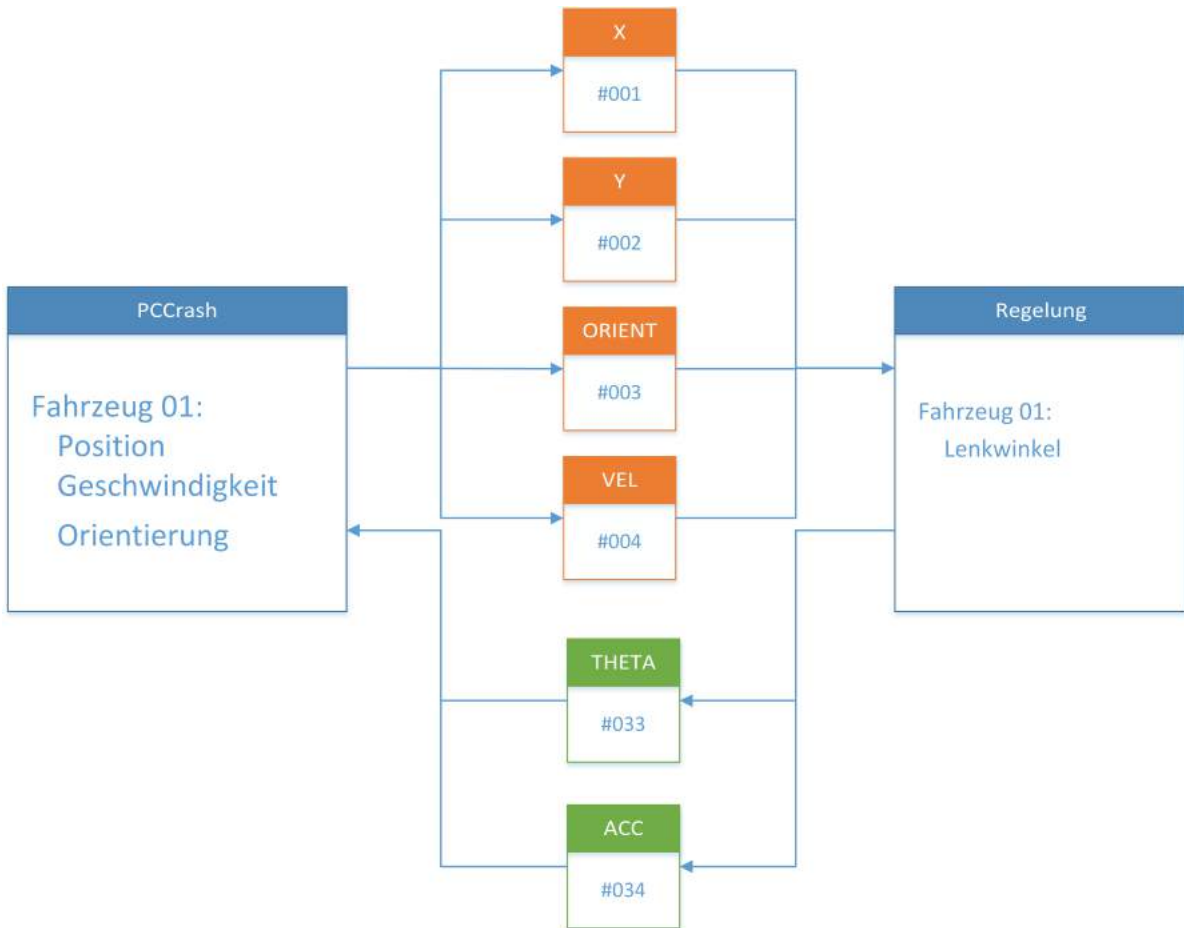


Abbildung 3.8: Verbindung der Simulations- und Regelungseinheit

Da mehrere Fahrzeuge beeinflusst bzw. die fahrzeugbezogenen Daten abgefragt werden können, wurde ein Fahrzeug-spezifischer ID-Offset definiert. Dazu wird die Möglichkeit der Masken- und Filterimplementierung von CAN-Steuergeräten ausgenutzt.

Jedes Steuergerät im Netzwerk kann durch eine spezifische Maske, Nachrichten durch White Listing verwerfen oder bearbeiten [17]. Das Konzept von Masken funktioniert folgendermaßen:

Jeder Knoten im Netzwerk besitzt eine binäre Maske. Der Identifikator jeder Nachricht durchläuft nun diese Maske. Jedes Bit, welches in der Maskendefinition des Steuergerätes 1 ist, ist für den Filter relevant.

Ein Beispiel:

Eine Nachricht hat die ID 110 und die Maske eines bestimmten Steuergerätes ist 111,

das bedeutet, dass für einen möglichen Filter im Steuergerät alle ersten 3 Bits in der ID jeder Nachricht zu berücksichtigen sind[17].

Da auf nicht mehr als 25 Werte aus der Simulation zugegriffen werden muss, ist es sinnvoll, die Maske auf 6 Bits zu beschränken. Deshalb wird ein dezimaler Offset von 64 für jedes Fahrzeug verwendet.

$$offset = n_{Fahrzeug} * 64 \quad (3.1)$$

Ein Beispiel wäre:

Will den Lenkwinkel des Fahrzeuges mit der Nummer 01 regeln, muss man die Position und Orientierung des Fahrzeuges in der Regelung berücksichtigen. Die Positionsdaten (X/Y) dieses Fahrzeuges in der Simulation werden anhand zweier Nachrichten mit den IDs 01 und 02 an den Bus übermittelt.

Anhand der Offset-Berechnung muss nun der Controller der Regelungseinheit, die ebenfalls am CAN-Bus hängt, die Nachrichten mit den IDs  $01 \cdot 64 + 01 = 65$  und  $01 \cdot 64 + 02 = 66$  abfangen und als Positionsdaten verarbeiten.

Umgekehrt verhält es sich in exakt gleicher Weise. Alle in die Simulationssoftware PC-Crash eingehenden Nachrichten beginnen bei ID 32. Will nun die Regelungssoftware den Lenkwinkel des Fahrzeuges 01 in der Simulation beeinflussen, muss diese die Nachrichten, die den Wert des Lenkwinkels beinhalten, mit der ID  $01 \cdot 64 + 33 = 97$  markieren. So kann nun PC-Crash den Lenkwinkel von Fahrzeug 01 auf den übertragenen Wert umstellen.

Da nun die Entwicklungsumgebung alle Funktionalitäten besitzt, die ein Testen verschiedener Reglertypen auf getrennten Systemen zulassen, wird in den folgenden Kapiteln auf die Regelungarten eingegangen.

## 3.4 Lineare Modellbasierte Prädiktive Regelung

Modellbasierte Prädiktive Regelungen werden häufig in der Industrie eingesetzt. Sie eignen sich besonders zur Regelung verfahrenstechnischer Prozesse. Die modellbasierte prädiktive Regelung (MPC) zählt zu den nichtlinearen Regelungsverfahren, sie kann jedoch sowohl auf lineare als auch auf nichtlineare Regelstrecken angewandt werden.

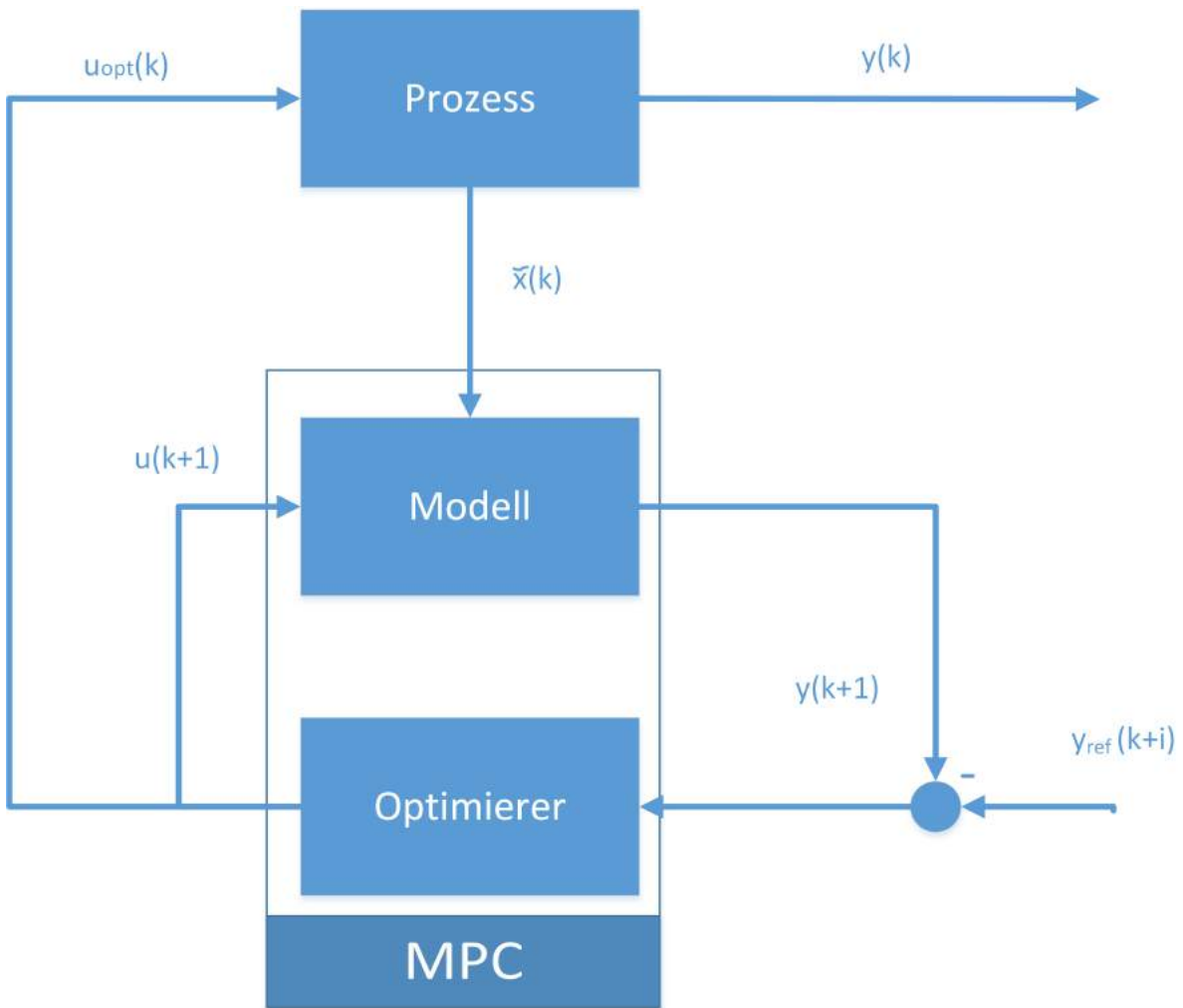


Abbildung 3.9: Aufbau des Regelkreises eines modellbasierten prädiktiven Reglers

Abbildung 3.9 zeigt den Aufbau des Regelkreises einer MPC wie sie in dieser Arbeit umgesetzt wurde. Der Regelkreis setzt sich aus einem Prozess und der MPC zusammen. Das Modell des Reglers wird mit den aktuellen Zustandsgrößen  $\tilde{\mathbf{x}}$  des realen Prozesses versorgt. Anhand derer errechnet das Modell den Ausgangsvektor  $\mathbf{y}(k+i)$  des zukünftigen Verhaltens. Anhand dieser Ausgangswerte und den Referenz- bzw. Sollwerten wird nun mit Hilfe eines Optimierungsverfahrens eine Stellgröße  $\mathbf{u}(k+i)$  für den Prozess und das Referenzmodell ermittelt [18].

### 3.4.1 Eigenschaften, MP-Regelung

#### EIGENSCHAFTEN

Die MPC ist eine sich in der Regelungstechnik schnell verbreitende, komplexe aber auch genaue Regelungsmethode. Meist wird sie in verfahrenstechnischen Bereichen eingesetzt, kann aber auch auf jedes beliebige Regelungsproblem übertragen werden.

MPC zählt zu den nichtlinearen Regelungsverfahren, kann jedoch sowohl auf lineare Regelstrecken mit Beschränkung, als auch auf nichtlineare angewandt werden.

Die Idee der Modellbildung zur Regelung dient zum einen der Verbesserung des Verständnisses für die Prozesse und Funktionsweisen der zu regelnden Strecken. Zum anderen ist es möglich eine Aussage über zukünftige Zustände des Systems zu treffen und diese in die Regelung miteinzubeziehen. [18]

### 3.4.2 Theoretische Grundlagen

Die Grundidee eines modellbasierten prädiktiven Reglers besteht darin, anhand eines Referenzmodells des zu regelnden Systems, Zustandsgrößen in die Zukunft zu berechnen und dadurch die Stellgröße zu optimieren. Vor allem ein Überschwingen soll dadurch minimiert werden.

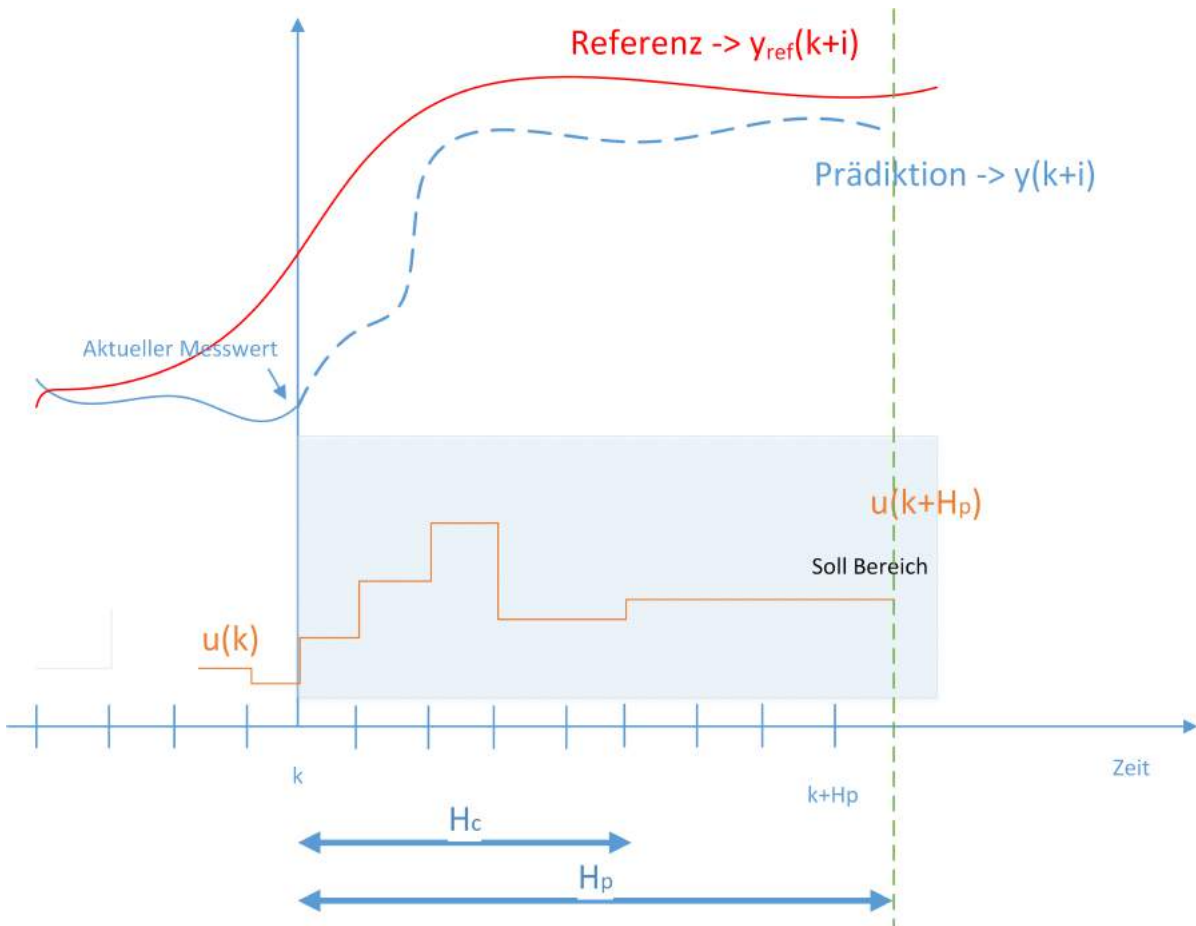


Abbildung 3.10: Ablauf der Stellgrößenberechnung bei MPC

Der Ablauf der MPC gliedert sich in folgende Abschnitte, die für jeden Regelschritt online ausgeführt werden [19] .

- Prädiktion
- Optimierung
- Gleitender Horizont

### Prädiktion

Bei den Vorhersagen wird mit Hilfe eines dynamischen Prozessmodells der zukünftige Verlauf der Ausgangsgröße  $y$  und die Stellgrößenfolge  $\mathbf{u}$  ermittelt. In Abbildung 3.10 ist ersichtlich, dass zu einem Zeitpunkt  $k$  ein Wert von  $\mathbf{u}(k-1)$  bekannt sein muss. Die Stellgröße  $u$  wird bis zu einem Stellhorizont  $H_c$  bestimmt. Die Länge der Vorhersage der

Ausgangsgröße  $y$  wird durch den Prädiktionshorizont  $H_p$  beschränkt. Es gilt allgemein  $H_p \geq H_c$ .

Es kann sowohl ein lineares als auch ein nichtlineares Prozessmodell zur Prädiktion hinzugezogen werden. Für diese Arbeit wurde ein lineares zeitdiskretes System gewählt.

Allgemein betrachtet kann die Berechnung eines Folgezustandes in einem diskreten linearen System wie folgt angeschrieben werden:

$$\begin{aligned}\mathbf{x}(k+1) &= \mathbf{A} \cdot \mathbf{x}(k) + \mathbf{B} \cdot \mathbf{u}(k), \\ \mathbf{y}(k) &= \mathbf{C} \cdot \mathbf{x}(k)\end{aligned}\tag{3.2}$$

,wobei  $u$  die diskreten Eingangsgrößen,  $\mathbf{y}$  den diskreten Ausgangsvektor,  $\mathbf{x}$  den Zustandsvektor,  $\mathbf{A}$  die Systemmatrix und  $\mathbf{B}$  den Eingangsvektor abbildet. Die Größen sind wie folgt definiert.  $\mathbf{x} \in \mathbb{R}^n$ ,  $\mathbf{u} \in \mathbb{R}^m$  und  $\mathbf{y} \in \mathbb{R}^r$

Um die Herleitung der Berechnung aller zukünftigen Ausgangsgrößen zu erleichtern werden hier noch keine Beschränkungen berücksichtigt.

Will man nun den momentanen Stellgrößenvektor  $\mathbf{u}(k)$  ermitteln, braucht man die „,vergangenen“ Eingangswerte  $\mathbf{u}(k-1)$  und die Stellgrößenveränderung  $\Delta\mathbf{u}(k)$ . Daraus ergibt sich:

$$\mathbf{u}(k) = \mathbf{u}(k-1) + \Delta\mathbf{u}(k)\tag{3.3}$$

Setzt man nun  $\mathbf{u}(k)$  in die Gleichung 3.25 ein, kann diese folgendermaßen angeschrieben:

$$\begin{aligned}\mathbf{x}(k+1) &= \mathbf{A} \cdot \mathbf{x}(k) + \mathbf{B} \cdot (\mathbf{u}(k-1) + \Delta\mathbf{u}(k)) \\ \mathbf{x}(k+2) &= \mathbf{A}^2 \cdot \mathbf{x}(k) + (\mathbf{A} + \mathbf{I}) \cdot \mathbf{B} \cdot \mathbf{u}(k-1) + (\mathbf{A} + \mathbf{I}) \cdot \mathbf{B} \cdot \Delta\mathbf{u}(k) \\ &\quad + \mathbf{B} \cdot \mathbf{u}(k+1) \\ &\quad \vdots \\ \mathbf{x}(k+i) &= \mathbf{A}^i \cdot \mathbf{x}(k) + (\mathbf{A}^{i-1} + \mathbf{A}^{i-2} + \dots + \mathbf{I}) \cdot \mathbf{B} \cdot \mathbf{u}(k-1) \\ &\quad + \sum_{j=1}^i (\mathbf{A}^{i-1} + \mathbf{A}^{i-2} + \dots + \mathbf{I}) \cdot \mathbf{B} \cdot \Delta\mathbf{u}(k+j-1)\end{aligned}\tag{3.4}$$

Löst man nun die Zustandsgrößen bis zum Stellhorizont  $H_c$  ergibt sich folgende Sum-



mengleichung:

$$\mathbf{x}(k+H_c) = \mathbf{A}^{H_c} \cdot \mathbf{x}(k) + (\mathbf{A}^{H_c-1} + \mathbf{A}^{H_c-2} + \dots + \mathbf{A} + \mathbf{E}) + \sum_{j=1}^{H_c} (\mathbf{A}^{H_c-j} + \dots + \mathbf{A} + \mathbf{E}) + \mathbf{B} \cdot \Delta \mathbf{u}(k+j-1) \quad (3.5)$$

Da jedoch der Prädiktionshorizont  $H_p$  als  $H_p \geq H_c$  definiert ist, muss auch die Berechnung der Zustandsgrößen nach dem Stellhorizont betrachtet werden. In diesen Berechnungsschritten wird für die Eingangsgrößen  $\mathbf{u}$  der letzte ermittelte Wert innerhalb des Stellhorizonts eingesetzt daher kann man  $\Delta \mathbf{u}(k+i) = 0$  definieren. Daraus ergibt sich folgende Definition zur Berechnung der Zustandsgrößen:

$$\begin{aligned} \mathbf{x}(k+H_c+1) &= \mathbf{A}^{H_c+1} \cdot \mathbf{x}(k) + (\mathbf{A}^{H_c} + \mathbf{A}^{H_c-1} + \dots + \mathbf{E}) \cdot \mathbf{B} \cdot (\mathbf{u}(k-1)) \\ &\quad + \sum_{j=1}^{H_c} (\mathbf{A}^{H_c+1-j} + \dots + \mathbf{A} + \mathbf{E}) + \mathbf{B} \cdot \Delta \mathbf{u}(k+j-1) \\ &\quad \vdots \\ \mathbf{x}(k+H_p) &= \mathbf{A}^{H_p} \cdot \mathbf{x}(k) + (\mathbf{A}^{H_p-1} + \mathbf{A}^{H_p-2} + \dots + \mathbf{E}) \cdot \mathbf{B} \cdot (\mathbf{u}(k-1)) \\ &\quad + \sum_{j=1}^{H_c} (\mathbf{A}^{H_p-j} + \dots + \mathbf{A} + \mathbf{E}) + \mathbf{B} \cdot \Delta \mathbf{u}(k+j-1) \end{aligned} \quad (3.6)$$

So werden alle zukünftigen Zustands- und Eingangsgrößen definiert. Durch Multiplikation der Zustandsgrößen mit dem Ausgangsvektor können nun die Ausgangsgrößen ermittelt werden:

$$\mathbf{y}(k+H_p) = \mathbf{C} \cdot \mathbf{x}(k+H_p) \quad (3.7)$$

Fasst man nun alle die für alle Schritte des Prädiktionshorizonts bestimmten Gleichungen zusammen, kann man Gleichung 3.25 folgendermaßen anschreiben:

$$\bar{\mathbf{y}} = \begin{bmatrix} \mathbf{y}(k+1) \\ \mathbf{y}(k+2) \\ \vdots \\ \mathbf{y}(k+H_p) \end{bmatrix} = \mathbf{C} \cdot \mathbf{x}(k+H_p) = \mathbf{C} \cdot \begin{bmatrix} \mathbf{x}(k+1) \\ \mathbf{x}(k+2) \\ \vdots \\ \mathbf{x}(k+H_p) \end{bmatrix} \quad (3.8)$$

Die Eingangsgrößen, die für jeden Schritt des Kontrollhorizonts definiert sind, werden

wie folgt zusammengefasst:

$$\Delta \bar{\mathbf{u}}(k) = \begin{bmatrix} \Delta u(k+1) \\ \Delta \mathbf{u}(k+2) \\ \vdots \\ \Delta \mathbf{u}(k+H_c) \end{bmatrix} \quad (3.9)$$

Abschließend kann nun die Gleichung für die Ausgangsgrößen für jeden Schritt innerhalb des Prädiktionshorizonts wie folgt angeschrieben werden:

$$\bar{\mathbf{y}}(k+1) = \mathbf{F} \cdot \mathbf{x}(k) + \mathbf{G} \cdot \mathbf{u}(k-1) + \mathbf{H} \cdot \Delta \bar{\mathbf{u}}(k) \quad (3.10)$$

,wobei  $\mathbf{F}$  definiert ist als:

$$\mathbf{F} = \begin{bmatrix} \mathbf{C} \cdot \mathbf{A} \\ \mathbf{C} \cdot \mathbf{A}^2 \\ \vdots \\ \mathbf{C} \cdot \mathbf{A}^{H_p} \end{bmatrix} \quad (3.11)$$

Die Matrix  $\mathbf{G}$  setzt sich wie folgt zusammen:

$$\mathbf{F} = \begin{bmatrix} \mathbf{C} \cdot \mathbf{B} \\ \mathbf{C} \cdot (\mathbf{A} + \mathbf{E}) \cdot \mathbf{B} \\ \vdots \\ \mathbf{C} \cdot (\mathbf{A}^{H_p} + \mathbf{E}) \cdot \mathbf{A}^{H_p} \end{bmatrix} \quad (3.12)$$

Und die Matrix  $\mathbf{H}$  ist definiert als:

$$\mathbf{H} = \begin{bmatrix} \mathbf{C} \cdot \mathbf{B} & 0 & \dots & 0 \\ \mathbf{C} \cdot (\mathbf{A} + \mathbf{E}) \cdot \mathbf{B} & \mathbf{C} \cdot \mathbf{B} & \dots & 0 \\ \vdots & \vdots & \dots & 0 \\ \mathbf{C}(\mathbf{A}^{H_c} + \dots + \mathbf{E}) \cdot \mathbf{B} & \mathbf{C}(\mathbf{A}^{H_c-1} + \dots + \mathbf{E}) \cdot \mathbf{B} & \dots & \mathbf{C} \cdot (\mathbf{A} + \mathbf{E}) \cdot \mathbf{B} \\ \mathbf{C}(\mathbf{A}^{H_p-1} + \dots + \mathbf{E}) \cdot \mathbf{B} & \mathbf{C}(\mathbf{A}^{H_p-2} + \dots + \mathbf{E}) \cdot \mathbf{B} & \dots & \mathbf{C}(\mathbf{A}^{H_p-H_p} + \dots + \mathbf{E}) \cdot \mathbf{B} \end{bmatrix} \quad (3.13)$$

Im Schritt der Prädiktion wurden nun alle Größen definiert, die durch das Modell gegeben sind. Die Matrizen  $\mathbf{F}$ ,  $\mathbf{G}$ ,  $\mathbf{H}$  und die Vektoren  $\bar{\mathbf{y}}(k+1)$ ,  $\mathbf{x}(k)$  und  $\mathbf{u}(k-1)$  können durch die Modellparameter bestimmt werden. Der Vektor  $\bar{\Delta \mathbf{u}}(k)$  muss jedoch durch Optimierung ermittelt werden.

### Optimierung:

Die zu minimierende Kostenfunktion wird meist über das quadratische Gütemaß beschrieben. Dieses setzt sich nach Berücksichtigung des Prädiktions- und Stellhorizonts wie folgt zusammen:

$$J = \sum_{j=1}^{H_p} (\mathbf{y}(k+j) - \mathbf{r}(k+j))^T \cdot \mathbf{Q} \cdot (\mathbf{y}(k+j) - \mathbf{r}(k+j)) + \sum_{j=1}^{H_c} \Delta \mathbf{u}^T(k+j-1) \cdot \mathbf{R} \cdot \Delta \mathbf{u}(k+j-1) \quad (3.14)$$

,wobei  $\mathbf{Q}$  und  $\mathbf{R}$  die Gewichtungen abbilden.  $H_p$  und  $H_c$  sind jeweils Prädiktions- und Stellhorizont.  $\mathbf{r}(k)$  definiert einen Referenzwert.

Die Aufgabe des Optimierens ist nun, durch Minimierung des Gütemaßes die optimale Stellgrößenänderung  $\Delta \mathbf{u}$  zu finden. Dazu müssen die oben hergeleiteten Gleichungen in das quadratische Gütemaß eingesetzt werden. Um diesen Vorgang zu vereinfachen wird in Gleichung 3.10 der vordere Teil durch  $\bar{\mathbf{g}}(k)$  ersetzt. Gleichung 3.10 kann nun folgendermaßen angeschrieben werden:

$$\bar{\mathbf{y}}(k+1) = \bar{\mathbf{g}}(k) + \mathbf{H} \cdot \bar{\Delta \mathbf{u}} \quad (3.15)$$

Eine weitere Vereinfachung wird durch die Bedingung

$$\bar{\mathbf{e}}(k) = \bar{\mathbf{g}}(k) - \bar{\mathbf{r}}(k+1) \quad (3.16)$$

getroffen.

Der nächste Schritt ist das Einsetzen in das quadratische Gütemaß. Setzt man nun Gleichung 3.10 in Gleichung 3.14 unter den vorher beschriebenen Bedingungen ein, ergibt sich folgende Gleichung:

$$J = \bar{\Delta \mathbf{u}}^T(k) (\mathbf{H}^T \cdot \mathbf{Q} \cdot \mathbf{H} + \mathbf{R}) \Delta \bar{\mathbf{u}}(k) + \bar{\Delta \mathbf{u}}^T(k) (\mathbf{H}^T \cdot \mathbf{Q} \cdot \mathbf{H} \cdot \bar{\mathbf{e}}(k) \cdot 2) + \bar{\mathbf{e}}(k)^T \cdot \mathbf{Q} \cdot \bar{\mathbf{e}}(k) \quad (3.17)$$

Dieses Optimierungsproblem kann mit einem Problem vom Typ ‘‘Quadratisches Programm (QP)’’ verglichen werden. Ein QP-Problem ist folgendermaßen definiert [20]:

$$f(x) = \frac{1}{2} \cdot \mathbf{x}^T \cdot \mathbf{Q} \cdot \mathbf{x} + \mathbf{c}^T \cdot \mathbf{x}$$

Grenzen (Constraints):

$$\mathbf{A} \cdot \mathbf{x} \leq \mathbf{b}$$

$$\mathbf{E} \cdot \mathbf{x} = \mathbf{d}$$

### Stellgrößenbeschränkung mit Constraints

Ein Vorteil des Umwandeln in ein QP-Problem, ist neben der Vielzahl an schon umgesetzten Lösungsbibliotheken, die einfache Behandlung von Beschränkungen. Um die Stellgrößen zu beschränken müssen Stellgrößenbeschränkungen in Abhängigkeit von  $\Delta \bar{\mathbf{u}}(k)$  formuliert werden.

$$\begin{aligned}
\mathbf{u}_{min} &\leq \mathbf{u}(k) = \mathbf{u}(k-1) + \Delta\mathbf{u}(k) && \leq \mathbf{u}_{max} \\
\mathbf{u}_{min} &\leq \mathbf{u}(k+1) = \mathbf{u}(k-1) + \Delta\mathbf{u}(k) + \Delta\mathbf{u}(k+1) && \leq \mathbf{u}_{max} \\
\mathbf{u}_{min} &\leq \mathbf{u}(k+1) = \mathbf{u}(k-1) + \Delta\mathbf{u}(k) + \Delta\mathbf{u}(k+1) + \Delta\mathbf{u}(k+2) && \leq \mathbf{u}_{max} \\
&\vdots \\
\mathbf{u}_{min} &\leq \mathbf{u}(k+H_c-1) = \mathbf{u}(k-1) + \Delta\mathbf{u}(k) + \dots + \Delta\mathbf{u}(k+H_c-1) && \leq \mathbf{u}_{max}
\end{aligned} \tag{3.18}$$

Die max/min Werte lassen sich nun in Vektorschreibweise zusammenfassen. Zum einen der Vektor  $\bar{\mathbf{u}}_{min} = [\mathbf{u}_{min}^T \dots \mathbf{u}_{min}^T, \mathbf{u}_{min}^T]^T$  zum anderen  $\bar{\mathbf{u}}_{max} = [\mathbf{u}_{max}^T \dots \mathbf{u}_{max}^T, \mathbf{u}_{max}^T]^T$ . Die Vektoren haben die Länge  $m \cdot H_c$ , wobei  $m$  die Anzahl der Stellgrößen definiert und  $H_c$  den Stellhorizont. Mit Hilfe dieser Vektoren lässt sich nun Gleichung 3.18 in Matrixschreibweise anschreiben:

$$\bar{\mathbf{u}}_{min} \leq \mathbf{E} \cdot \mathbf{u}(k-1) + \mathbf{D} \cdot \Delta\mathbf{u}(k) \leq \bar{\mathbf{u}}_{max} \tag{3.19}$$

,wobei  $\mathbf{E}$  als eine Matrix aus Einheitsmatritzen mit den Dimensionen  $m \cdot H_c \times m$  definiert ist und  $\mathbf{D}$  sich aus Einheitsmatritzen in Dreiecksform zusammensetzt. Matrix  $\mathbf{D}$  mit der Dimension  $m \cdot H_c \times m \cdot H_c$  sieht wie folgt aus:

$$\mathbf{D} = \begin{bmatrix} \mathbf{I} & 0 & 0 & \dots & 0 \\ \mathbf{I} & \mathbf{I} & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{I} & \mathbf{I} & \mathbf{I} & \dots & \mathbf{I} \end{bmatrix} \tag{3.20}$$

Neben den Beschränkungen der Stellgrößen gibt es auch noch Beschränkung der Änderungsrate der Stellgrößen und Beschränkungen der Zustands bzw. Ausgangsgrößen [18].

### Optimierungssoftware/CVXGEN

Als numerische Lösungsbibliothek wurde CVXGEN verwendet. Dieses Software-Tool erlaubt die Formulierung von konvexen Optimierungsproblemen. Daraus kann ein C-Code generiert werden, der einen Hochgeschwindigkeits-Lösungsalgorithmus erzeugt [20].

Ein Beispiel für die Formulierung eines MPC-Regelproblems ist folgender, der Webseite der Entwickler entnommener Code [21].

Listing 3.1: CVXGEN Formulierung eines MPC Problems

```

dimensions
  m = 2 # inputs.
  n = 5 # states.
  T = 10 # horizon.
end

parameters
  A (n,n) # dynamics matrix.
  B (n,m) # transfer matrix.
  Q (n,n) psd # state cost.
  Q_final (n,n) psd # final state cost.
  R (m,m) psd # input cost.
  x[0] (n) # initial state.
  u_max nonnegative # amplitude limit.
  S nonnegative # slew rate limit.
end

variables
  x[t] (n), t=1..T+1 # state.
  u[t] (m), t=0..T # input.
end

minimize
  sum[t=0..T](quad(x[t], Q) + quad(u[t], R))
  + quad(x[T+1], Q_final)
subject to
  x[t+1] == A*x[t] + B*u[t], t=0..T # dynamics constraints.
  abs(u[t]) <= u_max, t=0..T # maximum input box constraint.
  norminf(u[t+1] - u[t]) <= S, t=0..T-1 # slew rate constraint.
end

```

Diese Problemformulierung wird dann in kompilierbaren C-Code umgewandelt. Ein großer Nachteil dieser Methode ist, dass der Prädiktionshorizont aufgrund der vorallo-

kierten Matrix-Größen, auf einen Maximalwert vor der Kompillierung begrenzt ist.

### **Gleitender Horizont:**

Wird die Optimierung in jedem Schritt neu gestartet und das Optimierungsintervall weitergeschoben, spricht man von einem gleitenden Horizont (receding horizon) [19].

### **3.4.3 Modellbildung**

Bei der Modellbildung für MPC ist vor allem darauf zu achten, dass sowohl Genauigkeit als auch Rechenzeit für die Lösung berücksichtigt werden.

Je genauer das System ist, desto besser sind die Ergebnisse. Bei einem komplexeren Modell wird jedoch auch der Rechenaufwand erhöht.

Im Folgenden wird ein kinematisches Modell eines nicht holonomen Fahrzeuges verwendet.

### **Kinematisches Modell**

Die einfachste Möglichkeit das Fahrverhalten eines Fahrzeuges zu beschreiben ist das Einspurmodell nach Riekert und Schunck [22]. Um den Rechenaufwand für die Regelung zu vermindern, wurde das dynamische Verhalten vernachlässigt und nur die kinematischen Beziehungen berücksichtigt. Das Weglassen der dynamischen Einflüsse ist für langsame Kurvenfahrten auch zulässig [23].

Bei langsamer Kurvenfahrt, z. B. beim Einparken, ist die Fliehkraft praktisch Null, und damit auch die Summe der Seitenkräfte am Fahrzeug ... [23]

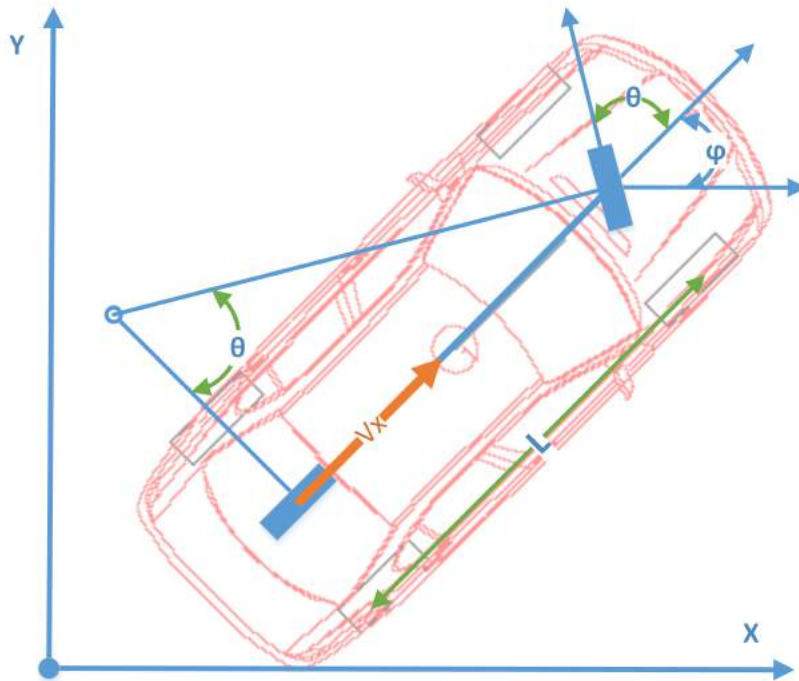


Abbildung 3.11: Beziehungen eines einfachen kinematischen Modells

Die Bewegung des Fahrzeuges wird nun mit folgenden Bewegungsgleichungen beschrieben. Die Gleichungen 3.21 und 3.22 berechnen Bewegung in X- und Y-Richtung. Wobei  $v_x$  die Geschwindigkeit in Fahrzeuggaengerichtung ist und  $\varphi$  die Orientierung des Fahrzeuges zur X-Achse beschreibt.

$$\frac{dX}{dt} = v_x \cdot \cos(\varphi) \quad (3.21)$$

$$\frac{dY}{dt} = v_x \cdot \sin(\varphi) \quad (3.22)$$

Die Winkeländerung  $\dot{\varphi}$  errechnet sich wie in Gleichung 3.23 beschrieben mit Hilfe der Ackermannbedingung aus dem Lenkwinkel  $\theta$ . Der Lenkwinkel wird bei Modellen, die sich nach der Ackermannbedingung verhalten, auch Ackermannwinkel genannt [24].

$$\frac{d\varphi}{dt} = \frac{v_x \cdot \tan(\theta)}{L} \quad (3.23)$$

$$\frac{dv_x}{dt} = a \quad (3.24)$$



### 3.4.4 Implementierung

Um das im vorherigen Kapitel hergeleitete Modell zur modellbasierten prädiktiven Regelung verwenden zu können, wird dieses nun als lineares zeitinvariantes System in Zustandsraumdarstellung angeschrieben.

$$\begin{aligned}\mathbf{x}(k+1) &= \mathbf{A} \cdot \mathbf{x}(k) + \mathbf{B} \cdot \mathbf{u}(k) \\ \mathbf{y}(k) &= \mathbf{C} \cdot \mathbf{x}(k)\end{aligned}\quad (3.25)$$

,wobei  $u$  die diskrete Eingangsgröße und  $y$  die diskrete Ausgangsgröße darstellt.

$A$ , in Gleichung 3.27 definiert, bildet die Systemmatrix und  $B$  den linearisierten diskreten Eingangsvektor ab.

$$\mathbf{x} = \begin{pmatrix} X \\ Y \\ \varphi \\ v_x \end{pmatrix}\quad (3.26)$$

Der Zustandsvektor  $\mathbf{x}$  ist für das obige kinematische Modell wie folgt definiert:

Setzt man nun die im vorherigen Abschnitt 3.4.3 hergeleiteten Modellgleichungen in das Zustandsraummodell 3.25 ein, erhält man für die Systemmatrix  $\mathbf{A}$  und den Eingangsvektor  $\mathbf{B}$  folgende Gleichungen [25]:

$$\mathbf{A}(k) = \begin{bmatrix} 1 & 0 & -v_x^n(k) \cdot \sin(\varphi^n(k)) \cdot T_d & \cos(\varphi^n(k)) \cdot T_d \\ 0 & 1 & v_x^n(k) \cdot \cos(\varphi^n(k)) \cdot T_d & \sin(\varphi^n(k)) \cdot T_d \\ 0 & 0 & 1 & T_d \cdot \frac{\tan(\theta^n(k))}{L} \\ 0 & 0 & 0 & 1 \end{bmatrix}\quad (3.27)$$

$$\mathbf{B}(k) = \begin{bmatrix} 0 \\ 0 \\ \frac{v_x^n(k) \cdot (1 + \tan^2(\theta^n(k)))}{L} \cdot T_d \\ 0 \end{bmatrix}\quad (3.28)$$

Die Nominalwerte  $\varphi^n(k)$ ,  $v_x^n(k)$  und  $\theta^n(k)$  errechnen sich aus dem vorgegebenen Pfad

und dienen zur Linearisierung des Systems im Arbeitspunkt. Diese Linearisierung basiert auf der Idee, dass solange sich das Fahrzeug annähernd auf einem gewissen erreichbaren Pfad befindet, sein Verhalten als linear anzunehmen sei [26].

Die oben angeführten Nominalwerte errechnen sich im Detail wie folgt:

$P_{Pfad}(k)$  sei der Punkt auf dem Pfad auf den das Fahrzeug im Schritt  $k$  zusteuern soll.

$P_{Pfad}(k)$  setzt sich folgendermaßen zusammen:

$$P_{Pfad}(k) = [X_{Pfad}(k), Y_{Pfad}(k)] \quad (3.29)$$

Die nominalen Gierwinkel  $\varphi^n(k)$  bis  $\varphi^n(H_p)$  werden nun aus den Koordinaten des momentanen Pfadpunktes und aus den zukünftigen Punkten berechnet.

$$\varphi^n(k) = \arctan \left( \frac{Y_{Pfad}(k) - Y_{Pfad}(k-1)}{X_{Pfad}(k) - X_{Pfad}(k-1)} \right) \quad (3.30)$$

Die Soll-Werte der vorwärtsgerichteten Fahrzeuggeschwindigkeiten  $v_x^n(k)$  werden mit folgender Gleichung bestimmt:

$$v_x^n(k) = \frac{\sqrt{(X_{Pfad}(k) - X_{Pfad}(k-1))^2 + (Y_{Pfad}(k) - Y_{Pfad}(k-1))^2}}{T_d} \quad (3.31)$$

Die letzten nominalen Größen sind die Lenkwinkel  $\theta^n(k)$ , die sich wie folgt aus dem vorgegebenen Pfad berechnen:

$$\theta^n(k) = \arctan \left( \frac{L \cdot (\varphi^n(k) - \varphi^n(k-1))}{v_x^n(k)} \right) \quad (3.32)$$

## 3.5 Proportional-Integral-Derivative Regelung

Um die Ergebnisse von Regelungsarten evaluieren zu können, braucht man Referenzergebnisse. Dazu wurden zusätzliche Reglerarten implementiert.

Nach dem oben ausgeführten Regelsystem, das zu den “Optimalen Regelungsmethoden” zählt, folgt in diesem Abschnitt die Herangehensweise an das Problem mittels Proportional-Integral-Derivative (PID) Regelung. Um eine PID-Regelung umsetzen zu können, ist es notwendig neben der Ist-Größe auch eine Soll-Größe einzuführen. Für die Ermittlung der Soll-Größe wurden 2 verschiedene Ansätze untersucht:

- Pure Pursuit
- Lookahead Steering

### 3.5.1 Pure Pursuit

Die Grundidee des Pure Pursuit Algorithmus ist, eine Kurve zu berechnen, die ein Fahrzeug zu einem gewünschten Zielpunkt auf einem vordefinierten Pfad zurückbringt. Der Algorithmus iteriert kontinuierlich mit einer fortlaufenden Zielposition am Pfad. Der Abstand zu diesem Zielpunkt ist durch die sogenannte Lookahead Distanz definiert [27].

## Theoretische Grundlagen

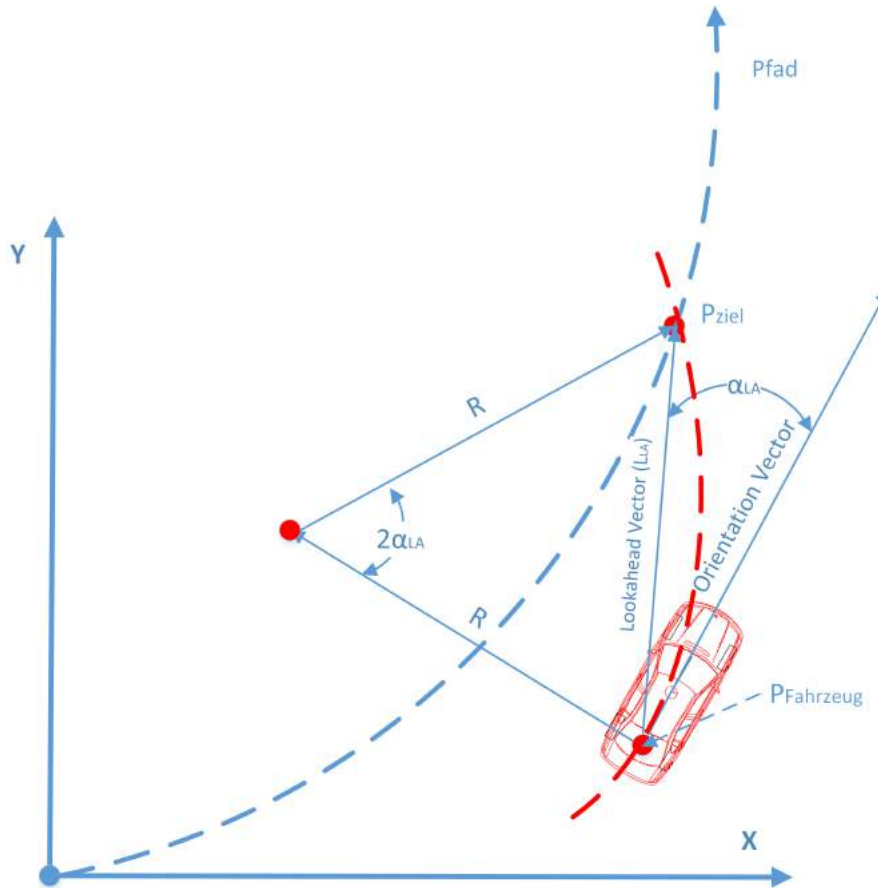


Abbildung 3.12: Beziehungen eines einfachen kinematischen Modells

In Abbildung 3.12 lässt sich erkennen, auf welche Art die Winkel zum gewünschten Zielpunkt am Pfad bestimmt werden können.

Der Zielpunkt auf dem Pfad  $P_{Ziel}$  wird durch eine Lookahead-Distanz und die Position des Fahrzeuges bestimmt.

## Mathematische Grundlagen

Die Position des Fahrzeuges wird durch den Punkt  $P_{fahrzeug} = [X_{fahrzeug}, Y_{fahrzeug}]$  beschrieben.

Der Zielpunkt auf dem Pfad wird als  $P_{ziel} = [X_{ziel}, Y_{ziel}]$  definiert.

In Abbildung 3.12 ist zu erkennen, dass der Lookahead-Vektor  $\vec{r}_{LA}$  entscheidend für die Bestimmung der Soll-Richtung des Fahrzeuges ist. Dieser ergibt sich aus der Fahr-

zeugposition und der Position des Zielpunktes auf dem Pfad.

$$\vec{r}_{LA} = \begin{pmatrix} X_{ziel} - X_{fahrzeug} \\ Y_{ziel} - Y_{fahrzeug} \end{pmatrix} \quad (3.33)$$

Durch das Vektorprodukt des in Formel 3.33 berechneten Lookahead-Vektors  $\vec{r}_{LA}$  und dem Richtungsvektor des Fahrzeuges  $\vec{r}_{Fahrzeug}$ , der sich aus der Positionsänderung bestimmen lässt, kann nun mit Formel 3.34 der Lookahead Winkel  $\alpha_{LA}$  berechnet werden.

$$\alpha_{LA} = \arccos\left(\frac{\vec{r}_{LA} \cdot \vec{r}_{Fahrzeug}}{|\vec{r}_{LA}| \cdot |\vec{r}_{Fahrzeug}|}\right) \quad (3.34)$$

Um den für die Annäherung des Fahrzeuges an den Pfad notwendigen Kreisbogen zu berechnen, stellt man nun den Lookahead-Winkel  $\alpha_{LA}$  und die Lookahead-Distanz  $L_{LA}$  mit dem durch den Radius  $R$  definierten Kreis (siehe Abbildung 3.12), in Zusammenhang:

$$\frac{L_{LA}}{\sin(2 \cdot \alpha_{LA})} = \frac{R}{\sin(\frac{\pi}{2} - \alpha_{LA})} \quad (3.35)$$

daraus folgt:

$$\frac{L_{LA}}{\sin(\alpha_{LA})} = 2 \cdot R \quad (3.36)$$

Da die Krümmung des Kreises als  $k = \frac{1}{R}$  definiert ist, kann die Gleichung 3.36 nun folgendermaßen angeschrieben werden:

$$k = \frac{2 \cdot \sin(\alpha_{LA})}{L_{LA}} \quad (3.37)$$

Aus der Gleichung zur Berechnung des Ackermannwinkels [24] kann jetzt der kinematische Lenkwinkel bestimmt werden:

$$\theta = \tan^{-1}\left(\frac{2 \cdot \sin(\alpha_{LA})}{L_{LA}}\right) \quad (3.38)$$

Betrachtet man Gleichung 3.38 genauer und formt man diese folgendermaßen um:

$$\sin(\alpha_{LA}) = \frac{e_{ld}}{L_{LA}} \rightarrow k = \frac{2}{L_{LA}} \cdot e_{ld} \quad (3.39)$$

wobei  $e_{ld}$  als Abweichung des Fahrzeuges vom Pfad definiert wird, so kann man erkennen, dass es sich beim Pure Pursuit-Algorithmus um einen Proportionalitätsregler

handelt.

$$u(t) = K_p \cdot e(t) \quad (3.40)$$

Die Parameter dieses PID-Reglers werden über die Lookahead-Länge  $L_{LA}$  definiert. Der Proportionalitätsfaktor  $K_p$  ergibt sich, wie aus Gleichung 3.40 ersichtlich, als folgender Term:

$$K_p = \frac{2}{L_{LA}} \quad (3.41)$$

### 3.5.2 Lookahead Steering

#### Theoretische Grundlagen

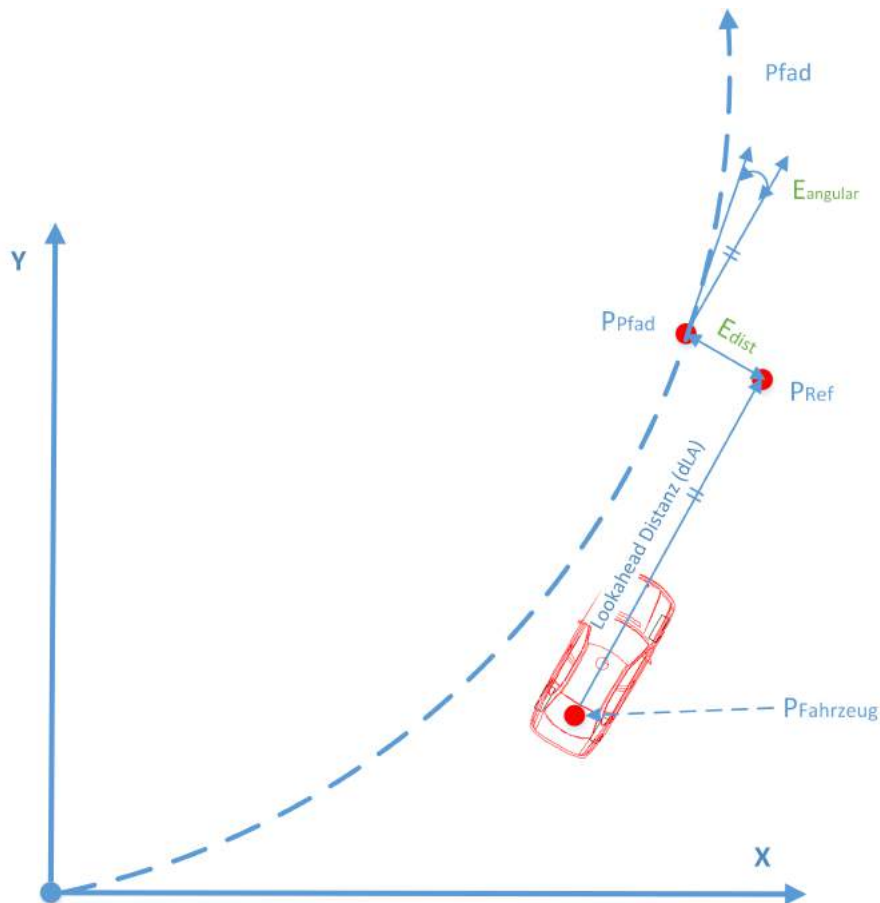


Abbildung 3.13: Bestimmung der Abweichungen mit dem Lookahead Steering-Verfahren

Mit Hilfe des Lookahead Steering-Verfahrens werden zwei verschiedene Abweichungen des Fahrzeuges zum Pfad ermittelt. Sowohl die Abweichung in der Distanz des Fahrzeuges zum Pfad  $E_{dist}$  als auch die Abweichung der Fahrtrichtung des Fahrzeuges zum Verlauf des Pfades  $E_{angular}$ .

In Abbildung 3.13 kann man erkennen, dass ein Referenzpunkt  $P_{ref}$  für das Fahrzeug definiert ist. Dieser ist durch die Zeit definiert, die „vorausgesehen“ werden soll. Daher stammt auch der Name dieser Methode: voraussehen  $\rightarrow$  eng. „lookahead“.

### Mathematische Grundlagen

Um das Lookahead Steering-Verfahren zur Bestimmung der Abweichung des Fahrzeuges zum vorgegebenen Pfad anwenden zu können, muss ein Referenzpunkt bestimmt werden. Dieser Punkt wird durch die Lookahead-Distanz  $d_{LA}$  definiert. Diese Distanz ist wiederum abhängig von der Geschwindigkeit des Fahrzeuges und wird in Gleichung 3.42 definiert.

$$d_{LA} = t_{LA} \cdot v_{Fahrzeug} \quad (3.42)$$

Anhand des Referenzpunktes, der in Gleichung 3.43 berechnet wird, wird nun mit Hilfe des nächstgelegenen Punktes am Pfad  $P_{Pfad}$  der Abstand zum Pfad bestimmt.

$$P_{ref} = P_{Fahrzeug} + d_{LA} \cdot \vec{d}_{Fahrzeug} \quad (3.43)$$

$$E_{dist} = (P_{ref} - P_{Pfad}) \cdot \vec{n}_{Pfad} \quad (3.44)$$

Der in Gleichung 3.44 beschriebene Abstandsfehler wird über die Differenz des Pfadpunktes und des Referenzpunktes, multipliziert mit dem Normalvektor (Gleichung 3.45) am Pfad, berechnet.

$$\vec{n}_{Pfad} = \begin{pmatrix} -\sin(\varphi_{Pfad}) \\ \cos(\varphi_{Pfad}) \end{pmatrix} \quad (3.45)$$

Um ein Überschwingen des Fahrzeuges zu vermindern, wird eine zweite Abweichung vom Pfad in der Regelung berücksichtigt. Der Winkelfehler  $E_{angular}$  beschreibt die Abweichung der Richtung des Fahrzeuges zum Verlauf des Pfades im Pfadpunkt  $P_{Pfad}$ .

$$E_{angular} = \varphi_{Fahrzeug} - \varphi_{Pfad} \quad (3.46)$$

Für den Regelalgorithmus wird die Stellgröße nach folgender Gleichung berechnet:

$$\theta = E_{angular} + \left( K_p \cdot E_{dist} + K_d \cdot \dot{E}_{dist} + K_i \cdot \int E_{dist} dt \right) \quad (3.47)$$

## 3.6 Fuzzy-Regelung

Als weitere Regelungsmethode wurde die Fuzzy-Regelung implementiert. Sie entwickelte sich aus der im Jahr 1965 von Lotfi A. Zadeh eingeführten Fuzzy Logik [28].

Die Fuzzy Logik kann durch einen Abgleich mit einer Wahrheitstabelle und Zugehörigkeitsfunktionen, im Gegensatz zu binärer Entscheidungsfindung, auch unscharfe Wahrheits- oder Falschheitsentscheidungen verarbeiten.

»... Fuzzy Logik kann Aussagen verarbeiten, die nur zu einem gewissen Grad wahr oder falsch sind .... «[29]

### 3.6.1 Theoretische Grundlagen

Der Vorgang der Fuzzy-Regelung wird in 3 Verarbeitungsschritte aufgeteilt:

- Fuzzifizierung
- Inferenz
- Defuzzifizierung

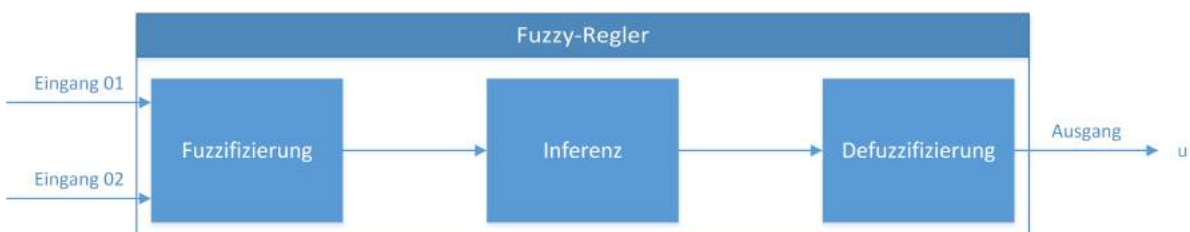


Abbildung 3.14: Die drei Schritte der Fuzzy-Regelung

### Reglerdesign

In der Entwurfsphase eines Fuzzy-Reglers müssen für alle Eingangs- und Ausgangsgrößen Fuzzy-Sets definiert werden. Diese teilen die jeweiligen Wertebereiche in Klassen auf.

Ein numerisches Beispiel aus dieser Arbeit ist:

Winkelabweichung zum Pfad:  $E_{angular} = 30^\circ$



Bezeichnung	Wert
Stark Positiv	100 .. 200 °
Mittel Positiv	0 .. 150 °
Zero	-50 .. 50 °
Mittel Negativ	-150 .. 0 °
Stark Negativ	-200 .. -100 °

Tabelle 3.1: Einteilung des Winkelfehlers in Fuzzy-Sets

Nach Einteilung in Fuzzy-Sets ist diese Abweichung mit der linguistischen Bezeichnung **Mittel Positiv** und **Zero** Abweichung zu beurteilen.

### Fuzzifizierung

Die Fuzzifizierung beschreibt die Umwandlung von diskreten Eingangsgrößen in Zugehörigkeitsgrade zu den Fuzzy-Sets.

### Inferenz

Die Inferenz beschreibt den Vorgang der numerischen Auswertung von Regelbedingungen. Das bedeutet, dass das Reglerverhalten bei verschiedenen Eingangsgrößen definiert werden muss. Dazu werden die Formulierungen „WENN Vorbedingung DANN Folgerung“ eingesetzt. Regeln können durch UND bzw. ODER verknüpft werden.

Ein Beispiel einer Regel für die in dieser Arbeit realisierte Lenkregelung mit Distanz- ( $E_{Distanz}$ ) und Winkelfehler ( $E_{angular}$ )-Berechnung, ist folgende Formulierung:

WENN  $E_{angular} =$  „Mittel Positiv“ ODER  $E_{Distanz} =$  „Mittel Negativ“ DANN  
Lenkwinkel = „Mittel Positiv“

Die Gesamtheit aller Regeln ergibt die Zugehörigkeitsgrade der Ausgangsgröße (Stellgröße) zu den dazugehörigen Fuzzy-Sets.

### Defuzzifizierung

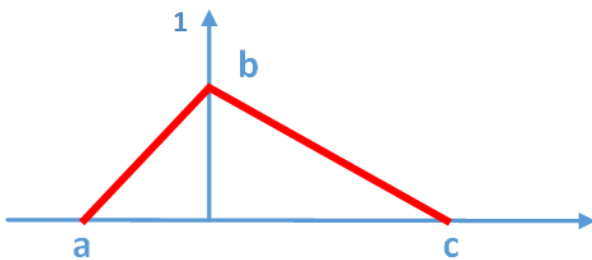
Im Schritt der Defuzzifizierung wird nun aus den in der Inferenz bestimmten Zugehörigkeitsgraden der Ausgangsgrößen, ein diskreter Wert als Stellgröße für die Regelung ermittelt.

### 3.6.2 Mathematische Grundlagen

Bei der Fuzzifizierung soll ein diskreter Wert in unscharfen Fuzzy-Sets abgebildet werden [30].

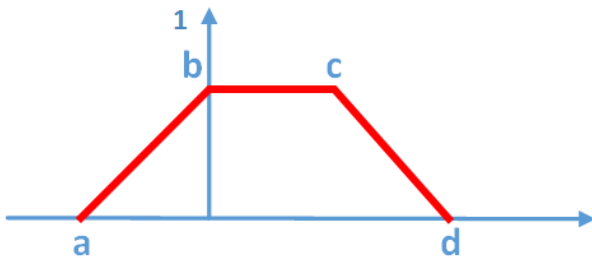
Diese Einteilung basiert auf einer sogenannten Zugehörigkeitsfunktion. Ein diskreter Eingangswert  $x$  wird anhand einer Funktion in die zugehörigen Fuzzy-Sets aufgeteilt.  $\mu(x)$  gibt den Grad der Zugehörigkeit zu einem Set an. Beispiele für solche Zugehörigkeitsfunktionen sind:

#### Dreieck



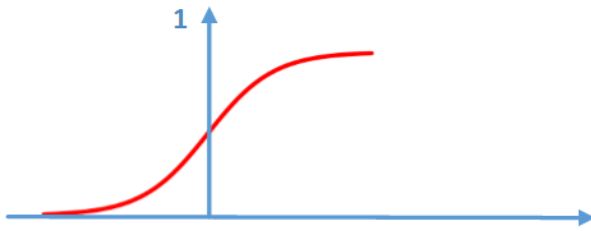
$$\mu(x) = \begin{cases} 0 & \text{für } x \leq b \\ \frac{x-a}{b-d} & \text{für } a \leq x \leq b \\ \frac{c-x}{c-b} & \text{für } b \leq x \leq c \\ 0 & \text{für } x \geq c \end{cases} \quad (3.48)$$

#### Trapez



$$\mu(x) = \begin{cases} 0 & \text{für } x \leq d \\ \frac{x-a}{b-d} & \text{für } a \leq x \leq b \\ 1 & \text{für } b \leq x \leq c \\ \frac{c-x}{c-d} & \text{für } c \leq x \leq d \\ 0 & \text{für } x \geq d \end{cases} \quad (3.49)$$

Abbildung 3.15: Trapez

**Sigmoid**

$$\mu(x) = \frac{1}{1 + e^{-x}} \quad (3.50)$$

Abbildung 3.16: Sigmoid

Ist nun die Zugehörigkeit eines diskreten Wertes zu einem Fuzzy-Set bestimmt, muss die Entscheidungslogik umgesetzt werden. Hierbei wird der Zusammenhang zwischen Eingabe- und Ausgabewert in linguistischen Regeln ausgedrückt. Nehmen wir hierzu das Beispiel aus dem vorherigen Abschnitt.

WENN  $E_{angular}$  = „Mittel Positiv“ ODER  $E_{Distanz}$  = „Mittel Negativ“ DANN  
Lenkwinkel = „Mittel Positiv“

Der Vorgang der Auswertung dieser Bedingungen teilt sich in zwei Schritte, die Implikation und die Aggregation.

**Implikation**

Mit Hilfe der Implikation werden die Vorbedingungen verknüpft. Das bedeutet, die Implikation behandelt das Problem der Verknüpfung von Regeln durch UND bzw. ODER.

Bei der Fuzzy-Logik bedient man sich der Verknüpfungsmethoden der klassischen Logik.

Es gibt Vereinigung-, Schnitt- und Komplementoperatoren, um Mengen zu verknüpfen. Um die Verknüpfungsmethoden UND bzw. ODER mathematisch auszudrücken, gibt es zwei Ansätze, einerseits durch min/max-Operatoren und andererseits durch Produkt und der algebraischen Summe.

**UND-Operator**

Der UND-Operator wird durch die min-Methode bzw. das Produkt dargestellt.

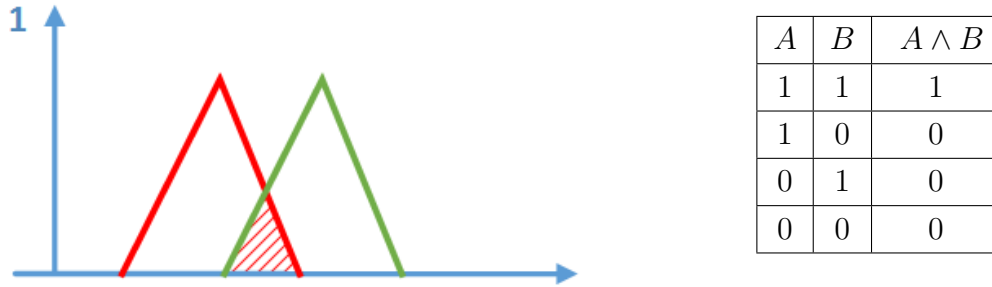


Abbildung 3.17: Konjunktion

Nimmt man die min-Methode, ergibt sich für die Kombination Lenkwinkel = Mittel Positiv folgende Berechnung:

$$\mu_{\text{Lenkwinkel, MittelPositiv}} = \min(\mu_{E_{\text{angular, MittelPositiv}}}, \mu_{E_{\text{Distanz, MittelNegativ}}}) \quad (3.51)$$

Bei der Produkt-Methode werden  $\mu_{E_{\text{angular, MittelPositiv}}}$  und  $\mu_{E_{\text{Distanz, MittelNegativ}}}$  multipliziert.

$$\mu_{\text{Lenkwinkel, MittelPositiv}} = \mu_{E_{\text{angular, MittelPositiv}}} \cdot \mu_{E_{\text{Distanz, MittelNegativ}}} \quad (3.52)$$

### ODER-Operator

ODER-Verknüpfungen werden mathematisch durch den max-Operator bzw. ein probabilistisches ODER (algebraische Summe) ersetzt.

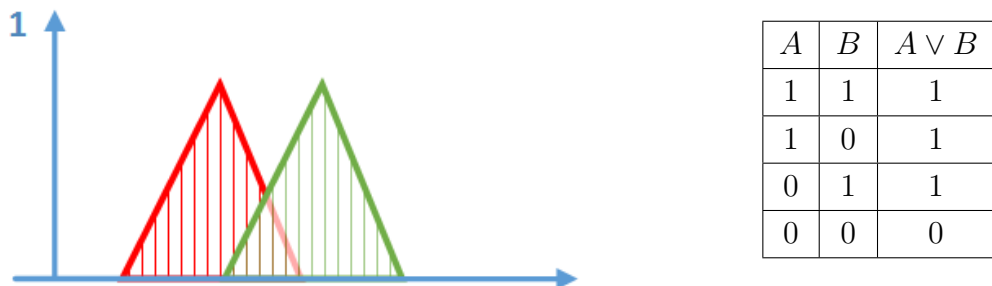


Abbildung 3.18: Disjunktion

$$\mu_{\text{Lenkwinkel, MittelPositiv}} = \max(\mu_{E_{\text{angular, MittelPositiv}}}, \mu_{E_{\text{Distanz, MittelNegativ}}}) \quad (3.53)$$

In einer zweiten Methode werden  $\mu_{E_{angular},MittelPositiv}$  und  $\mu_{E_{Distanz},MittelNegativ}$  mittels des probabilistischen ODER-Operators verknüpft.

$$\begin{aligned} \mu_{Lenkwinkel,MittelPositiv} = & \mu_{E_{angular},MittelPositiv} + \mu_{E_{Distanz},MittelNegativ} \\ & - \mu_{E_{angular},MittelPositiv} \cdot \mu_{E_{Distanz},MittelNegativ} \end{aligned} \quad (3.54)$$

### Aggregation

Im Schritt der Aggregation werden die Konsequenzen ermittelt. Das bedeutet, es wird das „DANN“ der linguistischen Regel ermittelt.

Diese Ermittlung der Schlussfolgerung kann mit einer Verknüpfung aller Unterentscheidungen durch einen ODER-Operator verglichen werden. Deshalb kann hier der max-Operator bzw. ein probabilistisches ODER (Algebraische Summe) verwendet werden.

Im letzten Schritt, der Defuzzifizierung wird nun aus den einzelnen Zugehörigkeitsgraden zu den Fuzzy-Sets ein diskreter Wert ermittelt. Dafür gibt es mehrere Methoden, in dieser Arbeit wird jedoch die Schwerpunktmethode verwendet:

$$u_s = \frac{\int_{u_{min}}^{u_{max}} u \cdot \mu_r(u) du}{\int_{u_{min}}^{u_{max}} \mu_r(u) du} \quad (3.55)$$

$u_s$  : diskrete Ausgangsgröße.

$\mu_r$  : Zugehörigkeitsgrad für u

### 3.6.3 Implementierung

Bei der Implementierung eines Fuzzy-Reglers ist es wichtig, die Eingangsgrößen in sinnvolle Fuzzy-Sets einzuteilen. Meist ist es von Vorteil die Sets nach sprachlichen Einteilungen zu wählen.

In dieser Arbeit wurden der Winkelfehler und Distanzfehler, die im Kapitel Lookahead-Steering hergeleitet wurden, in fünf Gruppen eingeteilt.

#### Einteilung Winkelfehler

Die Einteilung und der gewählte Wertebereich für den Winkelfehler sind in Tabelle 3.2 angeführt.

Bezeichnung	Wert
Stark Positiv	100 .. 200 °
Mittel Positiv	0 .. 150 °
Zero	-50 .. 50 °
Mittel Negativ	-150 .. 0 °
Stark Negativ	-200 .. -100 °

Tabelle 3.2: Einteilung des Winkelfehlers in Fuzzy-Sets

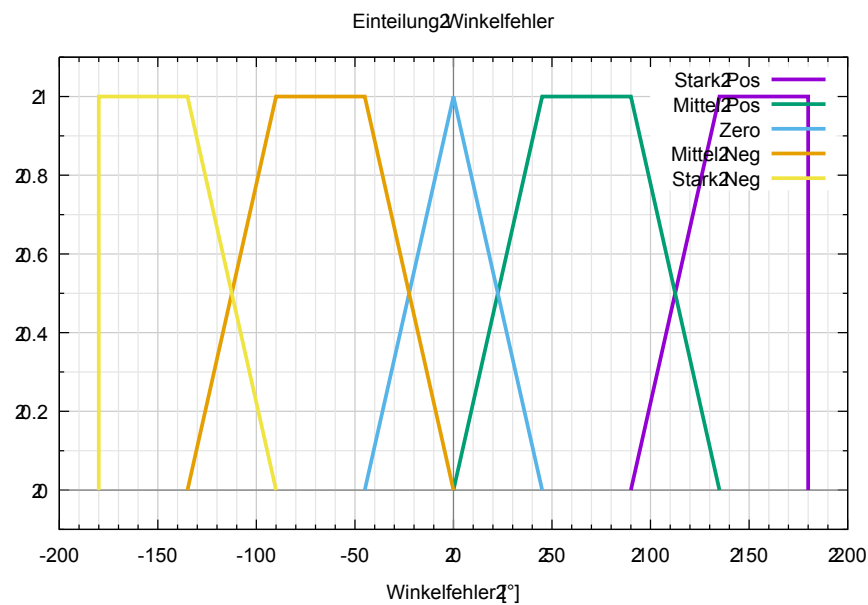


Abbildung 3.19: Einteilung des berechneten Winkelfehlers

### Einteilung Distanzfehler

Die Einteilung und der gewählte Wertebereich für den Distanzfehler sind in Tabelle 3.4 angeführt.

Bezeichnung	Wert
Stark Positiv	3 ... 10 m
Mittel Positiv	0 ... 5 m
Zero	-5 ... 5 m
Mittel Negativ	-5 ... 0 m
Stark Negativ	-10 ... -3 m

Tabelle 3.3: Einteilung des Distanzfehlers in Fuzzy-Sets

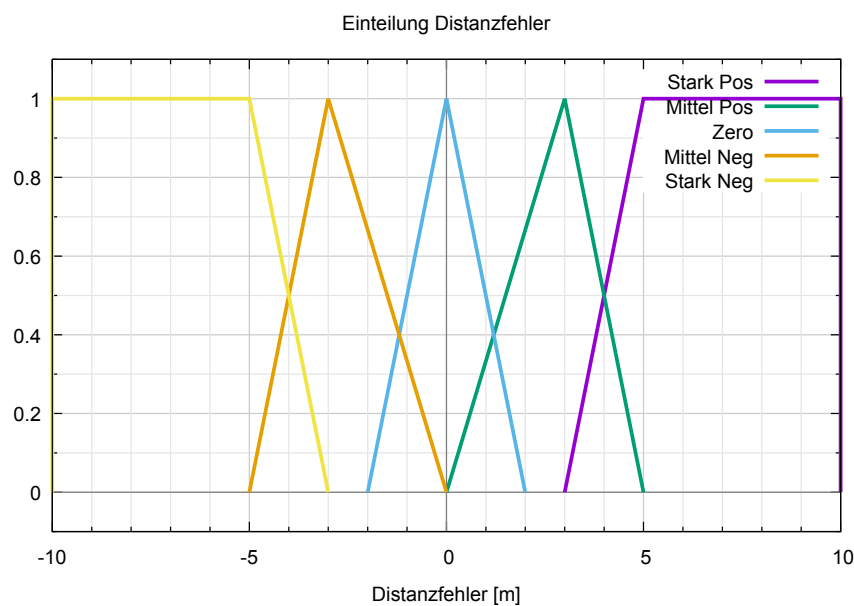


Abbildung 3.20: Einteilung des berechneten Distanzfehlers

### Einteilung des Lenkwinkels

Bezeichnung	Wert
Stark Positiv	30 ... 50 °
Mittel Positiv	15 ... 25 °
Zero	-5 ... 5 °
Mittel Negativ	-25 ... -15 °
Stark Negativ	-50 ... -30 °

Tabelle 3.4: Einteilung des Lenkwinkels in Fuzzy-Sets

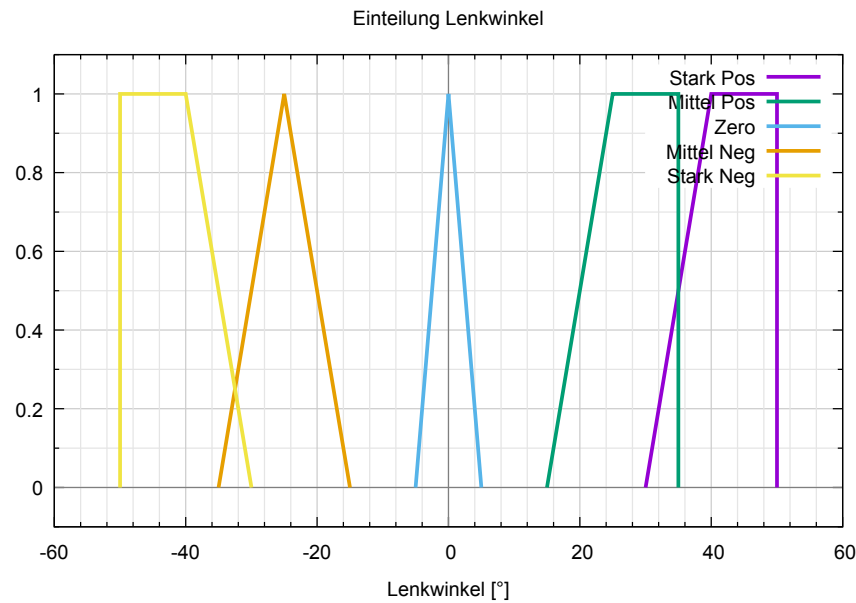


Abbildung 3.21: Einteilung des Lenkwinkels

## Fuzzylite



Für die programmiertechnische Umsetzung von Fuzzy Logik gibt es eine Vielzahl an Bibliotheken. Die für C++ am besten dokumentierte und zugleich unter Open-Source-Lizenz stehende Bibliothek ist “fuzzylite”. Die Entwickler beschreiben “fuzzylite” als freie und Open-Source-Bibliothek, die es einem Programmierer in einfacher Weise ermöglicht, in C++ objektorientiert Fuzzy-Logik-Regler zu implementieren, ohne zusätzliche Bibliotheken verwenden zu müssen [31].



# 4 Ergebnisse

## 4.1 Testumgebung

Um die oben implementierten Regelungssysteme ausreichend bzw. unter annähernd realen Bedingungen testen zu können, ist die Herstellung eines funktionalen Testaufbaus notwendig.

### **Testaufbau**

Hierzu wurde ein System, bestehend aus den in Abbildung 4.1 dargestellten Komponenten, aufgebaut. Dabei handelt es sich einerseits um einen Desktop-PC sowie einen Laptop und andererseits um eine physische Verbindung beider Komponenten über eine CAN-BUS-Schnittstelle.

Die Kenndaten der verwendeten Teilsysteme werden in Tabelle 4.1 genauer definiert.

<b>Bezeichnung</b>	<b>Laptop</b>	<b>Desktop PC</b>
Betriebssystemname	Microsoft Windows 7 Professional	Microsoft Windows 7 Professional
Version	6.1.7601 Service Pack 1 Build 7601	6.1.7601 Service Pack 1 Build 7601
Betriebssystemhersteller	Microsoft Corporation	Microsoft Corporation
Systemhersteller	Sony Corporation	MEDIONPC
Systemmodell	SVS13A1Y9ES	MS-7667
Systemtyp	x64-basierter PC	x64-basierter PC
Prozessor	Intel(R) Core(TM) i7-3520M @ CPU 2.90GHz, 2901MHz, 2 Kern(e)	Intel(R) Core(TM) i7-2600K CPU @ 3.40GHz, 3401 MHz, 4 Kern(e), 8 logische(r) Prozessor(en)
BIOS-Version/-Datum	Insyde Corp. R0140C5, 30.03.2012	American Megatrends Inc. E7667MLN.20C, 31.03.2011
SMBIOS-Version	2.7	2.6
Installierter physikalischer Speicher	12 GB	8 GB
Gesamter realer Speicher	1.9 GB	7.98 GB
Gesamter virtueller Speicher	23.8 GB	16 GB
Windows Leistungsindex	5.7	7.6

Tabelle 4.1: Leistungskenndaten der verwendeten Testcomputer

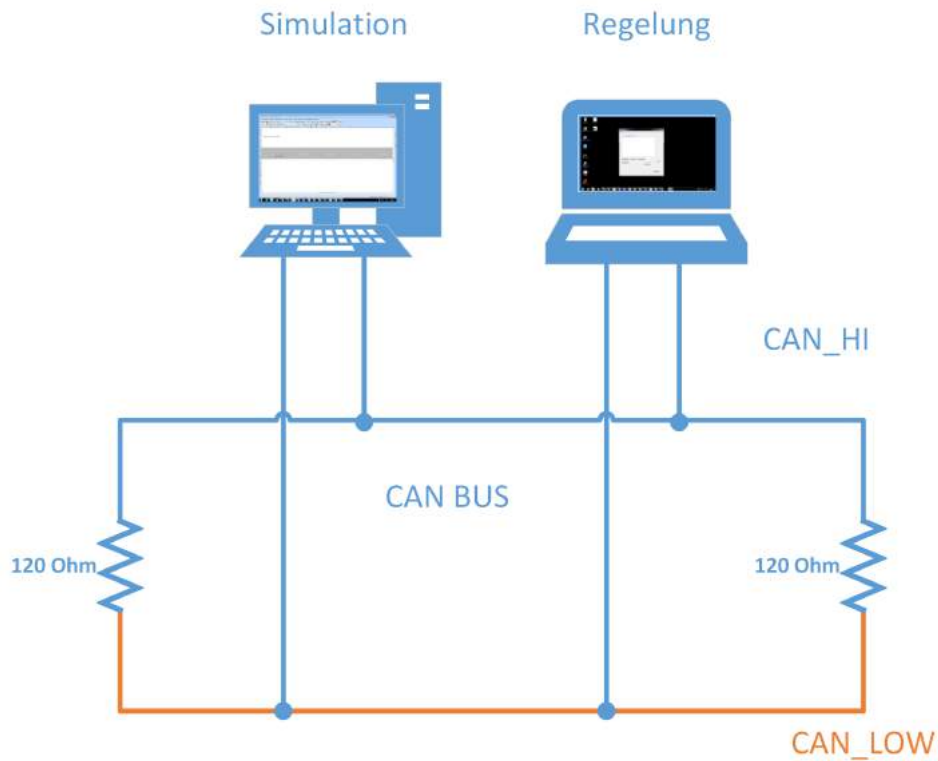


Abbildung 4.1: Aufbau und Verbindung der Testgeräte

Das Simulationsprogramm wurde auf dem Desktop-PC ausgeführt und die Regelungsalgorithmen auf einem mobilen Laptop-System angewandt.

Für die Verbindung der PC-Systeme zum CAN-BUS wurden USB zu CAN-Adapter der Firma PEAK-System Technik verwendet.

**Simulationsfahrzeug**

Abbildung 4.2: 3D Darstellung des Simulationsfahrzeugs

Als Simulationsfahrzeug wurde ein BMW 316i Modelljahr 2010 aus der PC-Crash Datenbank gewählt. Der PKW besitzt laut Datenbank DSD2015 folgende Kenndaten:

<b>Bezeichnung</b>	<b>Wert</b>
Hersteller:	BMW
Verkehrsbezeichnung:	316i
gefertigt von:	40179
gefertigt bis:	40513
Hubraum [ccm]:	1599
Leistung [kW]:	90
Motorart:	Benzinmotor
Karosserieform:	Limousine
Leergewicht [kg]:	1350
Länge [m]:	4.530
Breite [m]:	1.815
Höhe [m]:	1.5342
Spurweite [m]:	1.505
Wendekreis [m]:	11
Antriebsart:	RWD, DB
Anzahl der Achsen:	2
Fahrzeugart:	10
Gangzahl:	6
Achsübersetzung:	23437
Getriebeübersetzungen:	4.32 2.46 1.66 1.23 1 0.85
Drehzahl bei Maximalleistg:	6000
max. Moment [Nm]:	160
Drehzahl bei max. Moment:	4250
Radstand [m]:	2.77
Reifendimension 1:	205/55 R 16

Tabelle 4.2: Fahrzeugspezifische Werte des Simulationsfahrzeuges einem BMW 316i Modelljahr 2010

## 4.2 Testszenarios

Um ein standardisiertes Testen der Reglertypen gewährleisten zu können, ist es notwendig einheitliche Testbedingungen zu schaffen. Hierzu wurden in dieser Arbeit zwei typische Streckenszenarien ausgewählt, die sich zum Testen von fahrzeugtechnischen

Regelungssystemen bewährt haben.

Hierbei handelt es sich sowohl um einen Spurwechsel mit hoher bzw. geringer Geschwindigkeit, als auch um einen Fahrstrecken-Sprung bei mittlerer Geschwindigkeit.

### Spurwechsel

Als erstes Fahrmanöver wurde ein Spurwechsel gewählt, dessen Verlauf auf Abbildung 4.3 dargestellt ist.

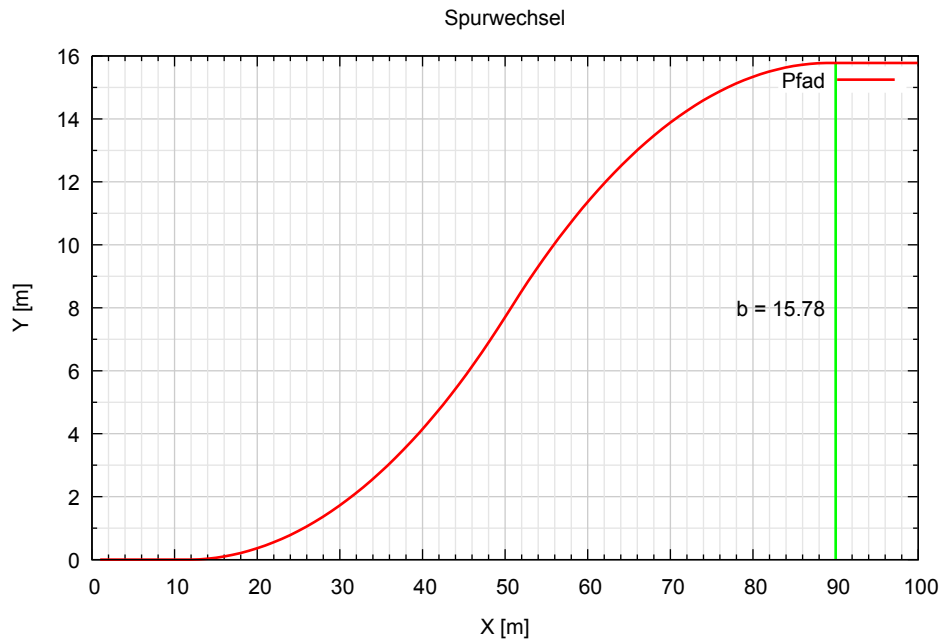


Abbildung 4.3: Der Verlauf der Pfadlinie des Spurwechselversuches

Der Spurwechsel wurde bewusst mit  $b = 15.78m$  etwas breiter definiert, um so einen längeren regelten Zeitabschnitt betrachten zu können.

Die Testgeschwindigkeiten wurden wie folgt gewählt:

Testname	Bezeichnung	Geschwindigkeit	Reibungskoeffizient $\mu$
Test01	langsam	$10m/s \rightarrow ca.36km/h$	trocken 0.8
Test02	schnell	$20m/s \rightarrow ca.72km/h$	trocken 0.8

Tabelle 4.3: Auflistung der Testgeschwindigkeiten

## Sprung

Ebenfalls wurde eine Sprungsituation als Testszenario ausgewählt, um das Verhalten der Regelungsalgorithmen bei schnellem Sollwertwechsel zu untersuchen. Bei diesem Versuch wird vor allem das Überschwingverhalten der implementierten Regler untersucht.

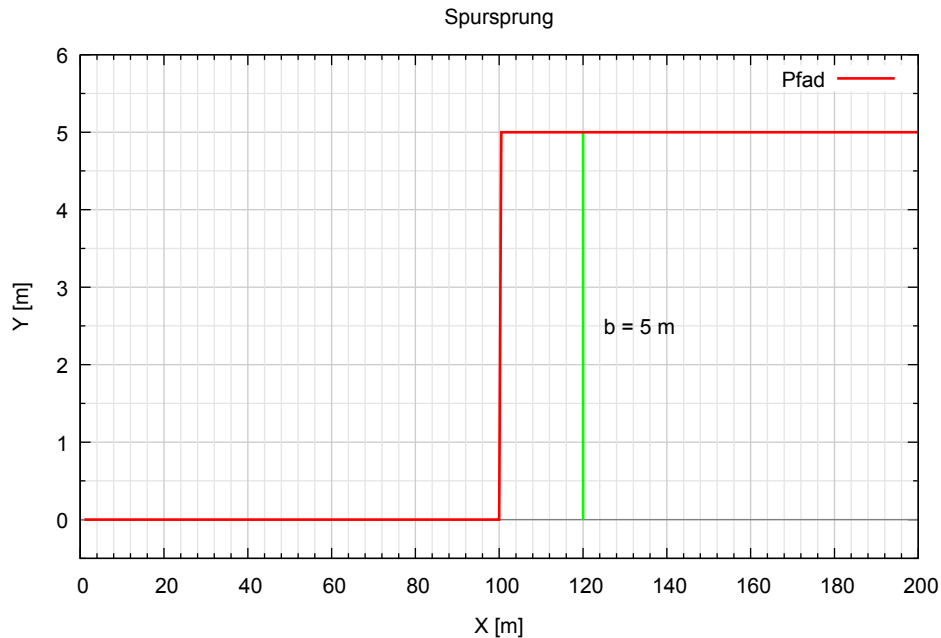


Abbildung 4.4: Der Verlauf der Pfadlinie des Spursprungversuches

Die Sprungweite beträgt  $b = 5\text{ m}$ . Für die Versuche mit diesem Pfad wurde nur eine Sollgeschwindigkeit für das Fahrzeug gewählt, da dieses Testszenario bei geringen Geschwindigkeiten kaum zu einem Überschwingen führt.

Testname	Bezeichnung	Geschwindigkeit	Reibungskoeffizient $\mu$
Test03	mittel	$15\text{ m/s} \rightarrow \text{ca. } 54\text{ km/h}$	trocken 0.8

Tabelle 4.4: Auflistung der Testgeschwindigkeiten

## 4.3 Bewertungskriterien

Um die entwickelten Regelungsalgorithmen bewerten zu können, ist es notwendig diese auf bestimmte Kriterien zu untersuchen. Folgende Kriterien wurden dazu ausgewählt:

- Gütekriterien

- Geschwindigkeit
- Positionsstreuungsverhalten
- Ausfallsverhalten
- Komplexität

### Gütekriterien

Die Zuverlässigkeit bzw. Güte eines Regelkreises kann anhand der Sprungantwort bestimmt werden. Dazu werden folgende Werte des zeitlichen Verlaufes der Sprungantwort abgelesen [32].

- bleibende Regeldifferenz
- Anstiegszeit  $T_r$  (rise time)
- Ausregelzeit  $T_s$  (settling time)
- Überschwingweite  $\ddot{u}$

**Anmerkung:** In dieser Arbeit werden die Sprungantworten nicht im Zeitbereich analysiert, sondern die Strecke in Fahrzeuginnenrichtung wird als Maß verwendet. Daher werden im weiteren Verlauf der Arbeit die Bezeichnungen  $D_r$  für die Anstiegsdistanz und  $D_s$  als die Ausregeldistanz eingeführt.

### Geschwindigkeit

Mit der Geschwindigkeit wird die Berechnungsdauer für die Ausgangsgröße zu jedem Abtastzeitpunkt definiert. Ist hoher Berechnungsaufwand für die Bestimmung der Regelgröße notwendig, sinkt die Geschwindigkeit.

### Positionsstreuungsverhalten

Das Positionsstreuungsverhalten präzisiert die Reaktion des Regelungsalgorithmus auf Abweichungen in der Positionsbestimmung. Das bedeutet, je weiter die an den Regler übergebene Position (Ist-Größe) von der tatsächlichen Position abweicht, desto höher ist die Streuung.



**Ausfallsverhalten**

Mit diesem Kriterium wird untersucht, wie sich ein Regler auf den Ausfall der Positions- bzw. Sollwertübergabe verhält.

**Komplexität**

Das Komplexitätskriterium soll die Bewertung des Aufwands der Implementierung und Umsetzung der jeweiligen Regelungsart beschreiben. Je mehr Zeit in die Umsetzung investiert wird, desto teurer würde dann eine tatsächliche Entwicklung werden. Dieser Aspekt muss ebenso wie die typischen Untersuchungskriterien in der Regelungstechnik berücksichtigt werden.

**4.4 Simulationsergebnisse**

Nach der Definition der Kriterien werden in diesem Abschnitt der Arbeit die Ergebnisse der Simulationen betrachtet.

Beginnend mit der MPC werden alle gefahrenen Testszenarien aller implementierten Regelalgorithmen numerisch ausgewertet und nach den oben genannten Kriterien bewertet.

Folgende Eigenschaften und Werte werden von jedem Regelalgorithmus bei jedem Testszenario ausgewertet.

<b>Bezeichnung</b>
Soll/Ist Position
Lenkwinkel
Quereschleunigungen
Abweichung

Tabelle 4.5: Auflistung der Auswertungs-Kenngrößen

Um die Testauswertung übersichtlicher zu halten, werden in Tabelle 4.6 noch einmal alle Testszenarien mit gefahrenem Pfad und eingestellter Sollgeschwindigkeit aufgelistet.

Bezeichnung	Pfad	Geschwindigkeit
Test01	Spurwechsel	10m/s
Test02	Spurwechsel	20m/s
Test03	Sprung	15m/s

Tabelle 4.6: Simulierte Testszenarien mit deren Spurfaden und Sollgeschwindigkeiten

Des Weiteren werden die quantitativen Werte der Gütekriterien für den Spursprungversuch jedes Regelungstypes angegeben.

## 4.5 Simulationsergebnisse - MPC

Zur Durchführung der Simulationen mit dem modellbasierten prädiktiven Regelungsalgorithmus wurde folgende Reglerparametrierung definiert:

Parameter	Wert
Abtastzeit $T_d$	50ms
$Q$	[10, 100, 100, 0.5]
$R$	15
$\theta_{min}$	-0.4rad
$\theta_{max}$	0.4rad
$d\theta$	$0.15 \cdot T_d$
$H_p = H_c$	20

Tabelle 4.7: Auflistung der Reglerparameter für Test01

**Test01 - Spurwechsel**  $v = 10m/s$

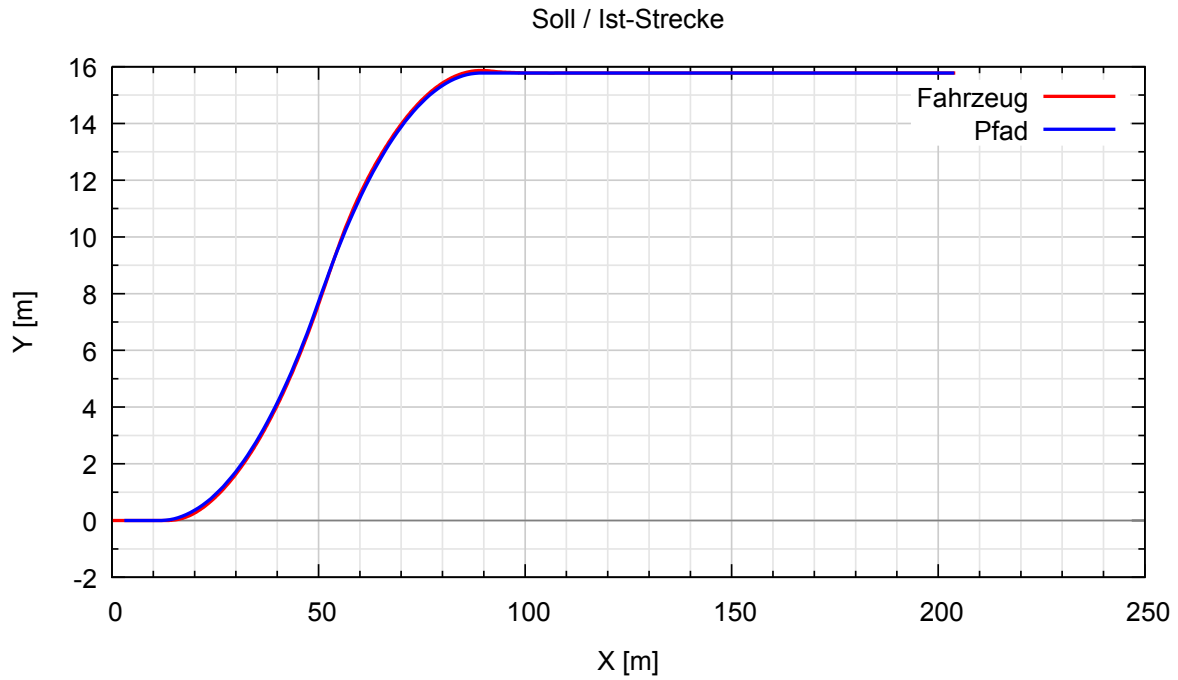


Abbildung 4.5: Soll/Ist-Weg

Anhand der Abbildung des Soll/Ist Weges ist ersichtlich, dass der Referenzpfad zufriedenstellend nachgefahren wird. Das kann auf die niedrige Geschwindigkeit des Fahrzeuges zurückgeführt werden. Bei niedrigen Geschwindigkeiten ist das Fahrzeugverhalten nur gering dynamisch und die Beschreibung durch das kinematische Fahrzeugmodell ist ausreichend.

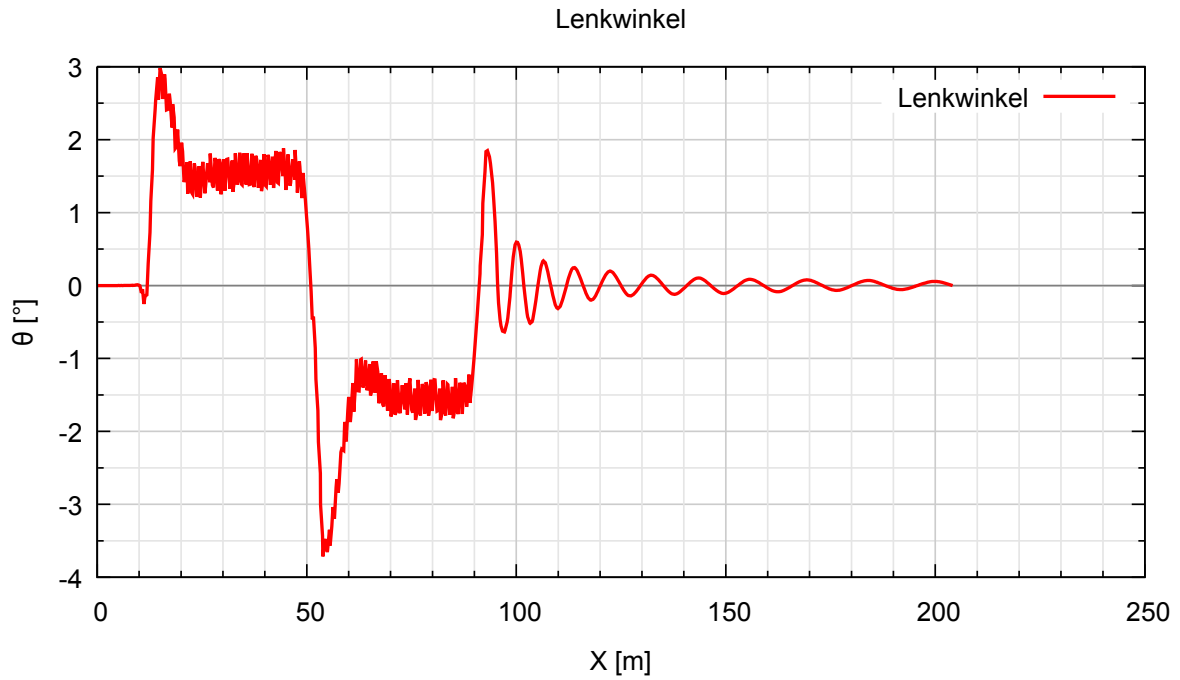


Abbildung 4.6: Lenkwinkel über die Simulationsstrecke

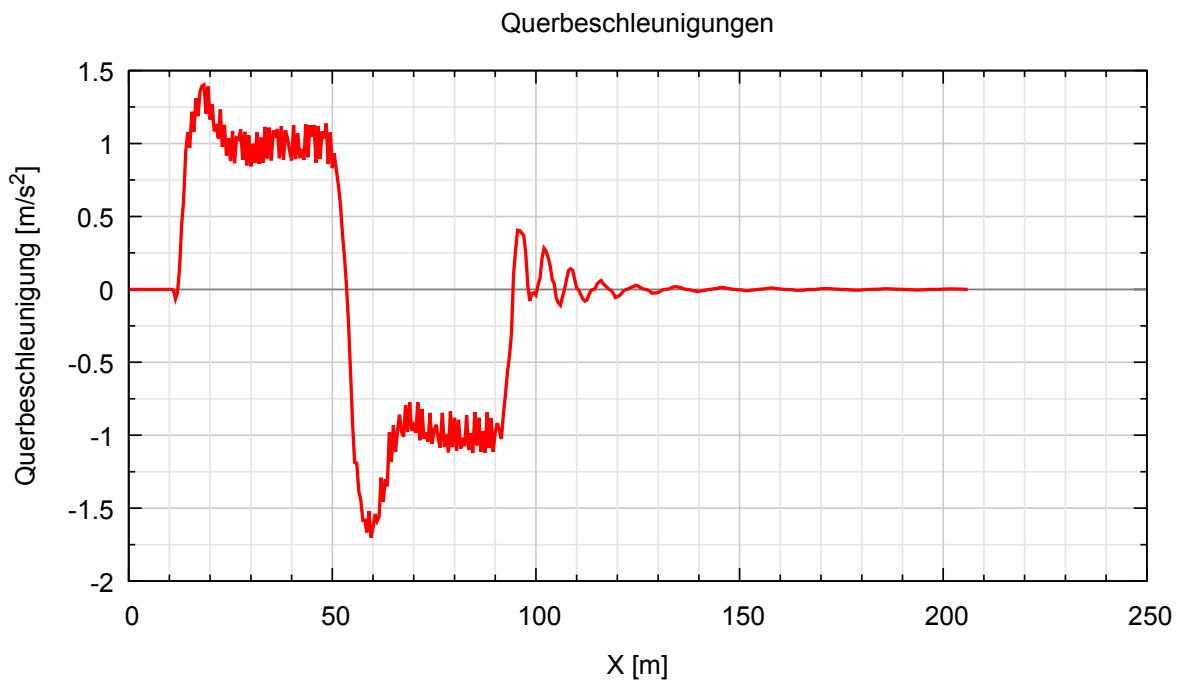


Abbildung 4.7: Querbeschleunigungen des Fahrzeuges während der Kurvenfahrt

Wie erwartet fallen bei der niedrigen Fahrzeuggeschwindigkeit von 10m/s auch die Querbeschleunigungen des Fahrzeuges sehr gering aus. Beim Zurücklenken wird ein maximum von  $1.7m/s^2$  erreicht.

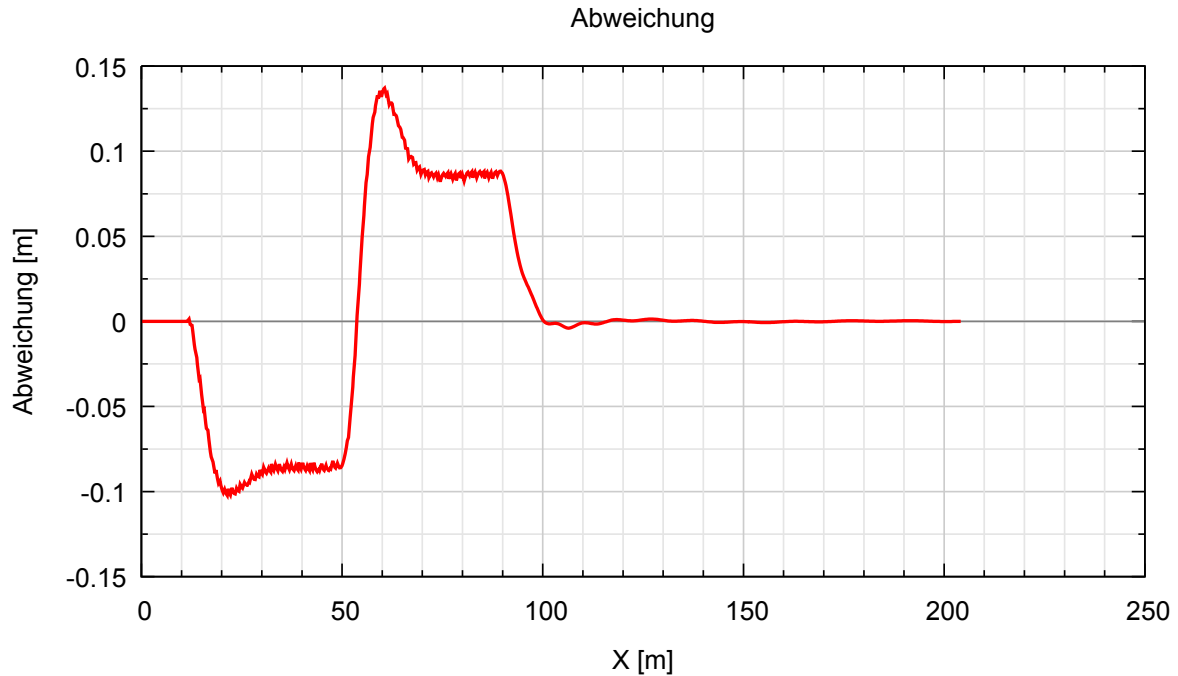


Abbildung 4.8: Regelfehler über die Strecke

In der Darstellung des Regelfehlers ist ersichtlich, dass die Abweichung sehr gering ist. Das kann auf die geringe Geschwindigkeit zurückgeführt werden. Bei geringen Geschwindigkeiten reicht das in dieser Arbeit entwickelte Prozessmodell aus, um das Fahrzeugverhalten derart zu beschreiben, dass die MPC ausreichend gute Stellgrößen liefern kann.

Parameter	Wert
Abtastzeit $T_d$	50ms
$Q$	[10, 100, 100, 0.5]
$R$	15
$\theta_{min}$	-0.4rad
$\theta_{max}$	0.4rad
$d\theta$	0.15rad
$H_p = H_c$	20

Tabelle 4.8: Auflistung der Reglerparameter für Test02 und Test03

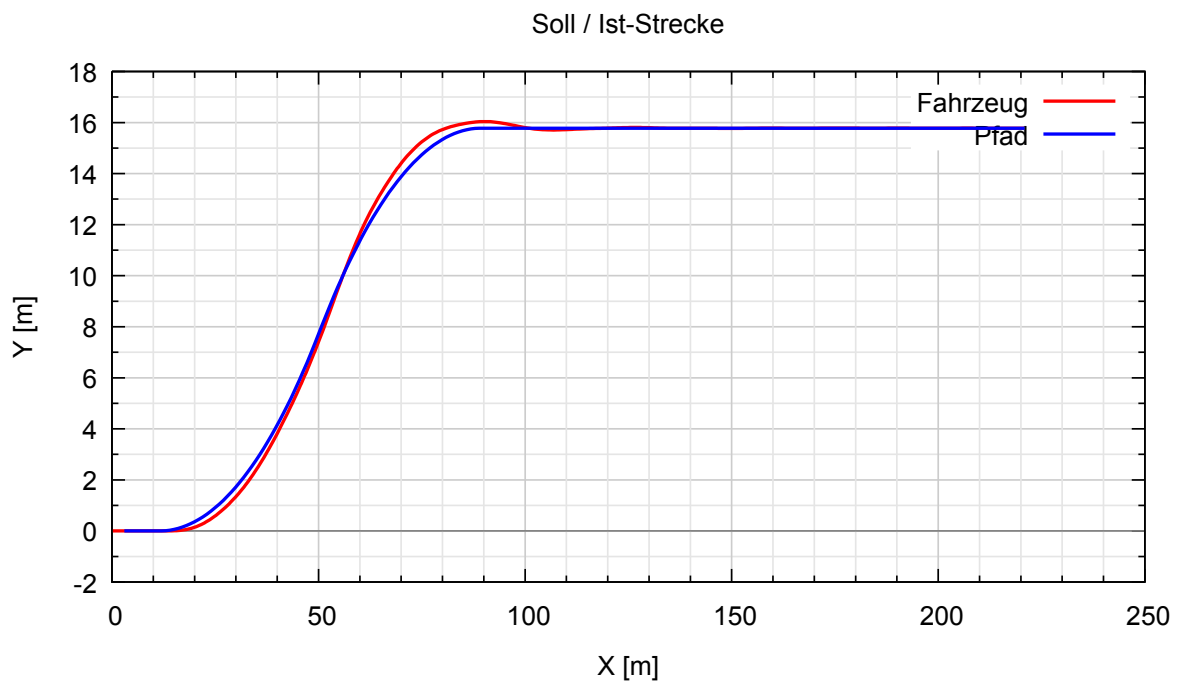
**Test02 - Spurwechsel  $v = 20m/s$** 

Abbildung 4.9: Soll/Ist-Weg

Auf Abbildung 4.9 ist zu erkennen, dass der MPC bei etwa 90m Fahrt einlenkt, um ein Ausbrechen des Fahrzeuges zu vermeiden. Das wird zum Einen durch die Beschränkung des Lenkwinkels allgemein und zum Anderen durch die Beschränkung des maximalen Anstiegs des Lenkwinkels verursacht. Diese Beschränkungen verhindern zu schnelle Lenkbewegungen. Die Wahl dieser Grenzen kommt durch die Vorgaben der gängigen Lenkroboter-Hersteller zustande.

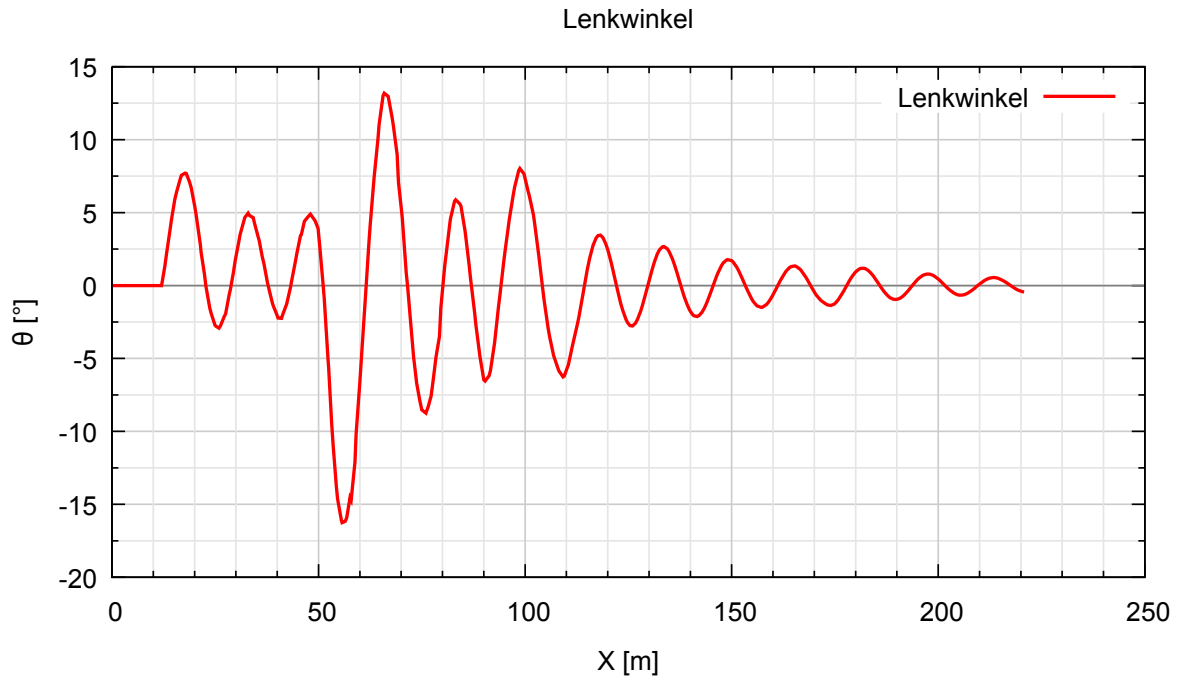


Abbildung 4.10: Lenkwinkel über die Strecke

Die oszillierende Lenkbewegung, die auf Abbildung 4.10 zu erkennen ist, kommt durch die Gewichtungsmatrizen des MPC zustande. Die Gewichtung wurde bei diesem Versuch sehr stark auf die Y-Position und die Orientierung des Fahrzeuges gelegt. Das verursacht jedoch einen andauernden Fehler in der Gütefunktion, welche das Optimierungsverfahren durch Sprünge der Stellgröße zu verbessern versucht.

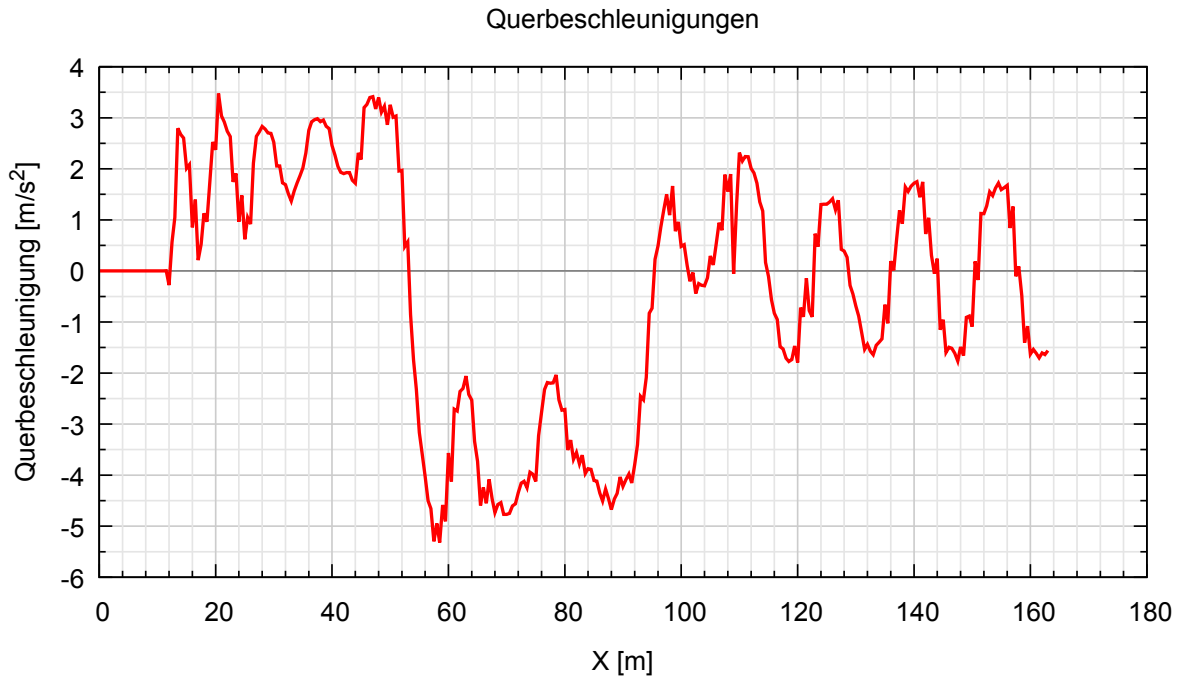


Abbildung 4.11: Querbeschleunigungen des Fahrzeuges bei  $v = 20 \text{ m/s}$

Die Querbeschleunigungen betragen beim Zurücklenken des Fahrzeuges nach dem Spurwechsel  $5 \text{ m/s}^2$ . Dieser Wert ist hoch, liegt jedoch innerhalb der Grenzen für eine stabile Fahrt eines Fahrzeuges.



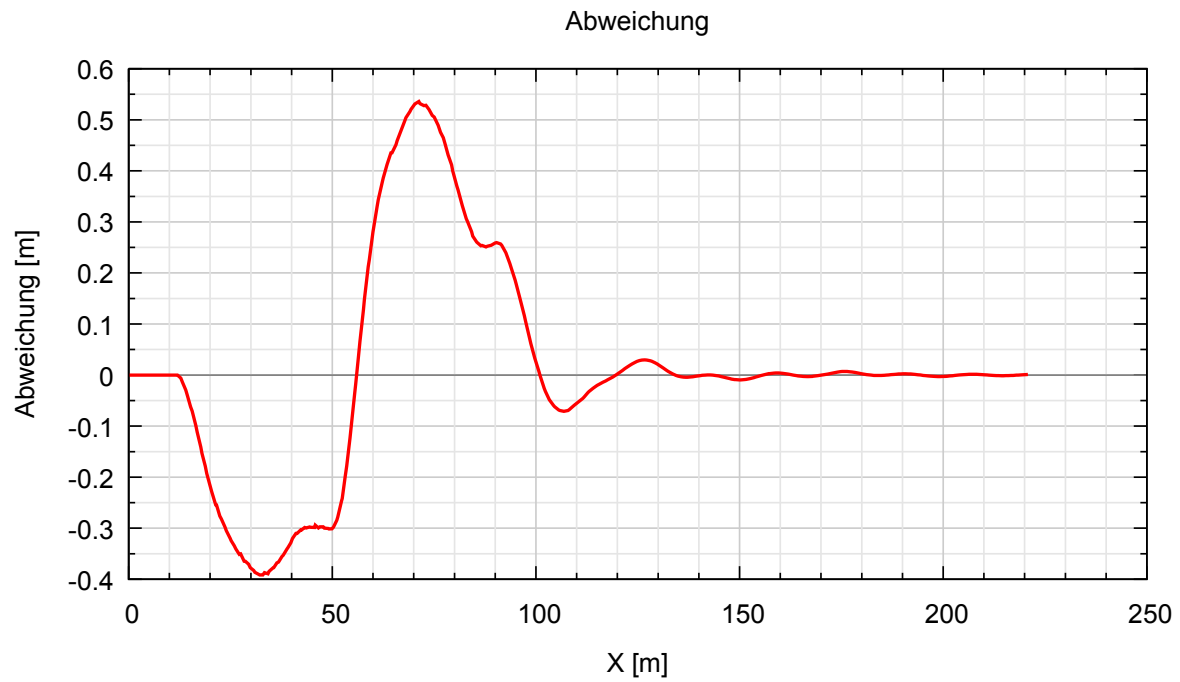


Abbildung 4.12: Regelfehler über die Strecke

Die Abweichung beträgt im Kurveneingang bzw. -ausgang ca  $40\text{cm}$ . Das Fahrzeug kommt jedoch schnell wieder auf den Sollweg zurück und die Abweichung oszilliert gering um die Abszisse.

**Test03 - Spursprung**  $v = 15\text{m/s}$

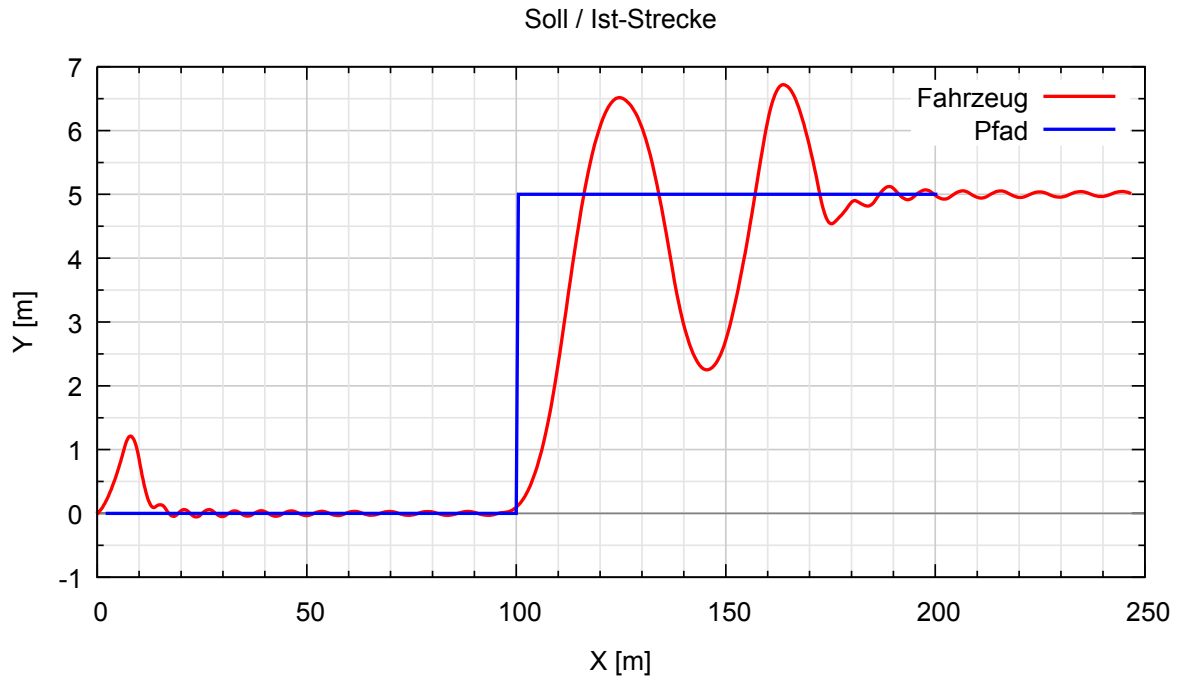


Abbildung 4.13: Soll/Ist-Weg

In Abbildung 4.13 ist der Verlauf der Fahrtstrecke des Fahrzeuges während des Spursprunges zu erkennen. Aufgrund der Geschwindigkeitsregelung ist eine kleine Pendelbewegung erkennbar. Diese lässt sich durch den Fehler, der durch den schnellen Geschwindigkeitsanstieg verursacht wird, erklären. Durch die die starken Querbewegungen beim Zurücklenken des Fahrzeuges nach der Sprungbewegung, zu sehen auf Abbildung 4.15, kommt es zu einer starken Schwingung ab etwa 120m. Diese kommt durch das starke dynamische Verhalten des Fahrzeuges zustande, welches in der Modellierung des Fahrzeuges nicht ausreichend berücksichtigt wird.

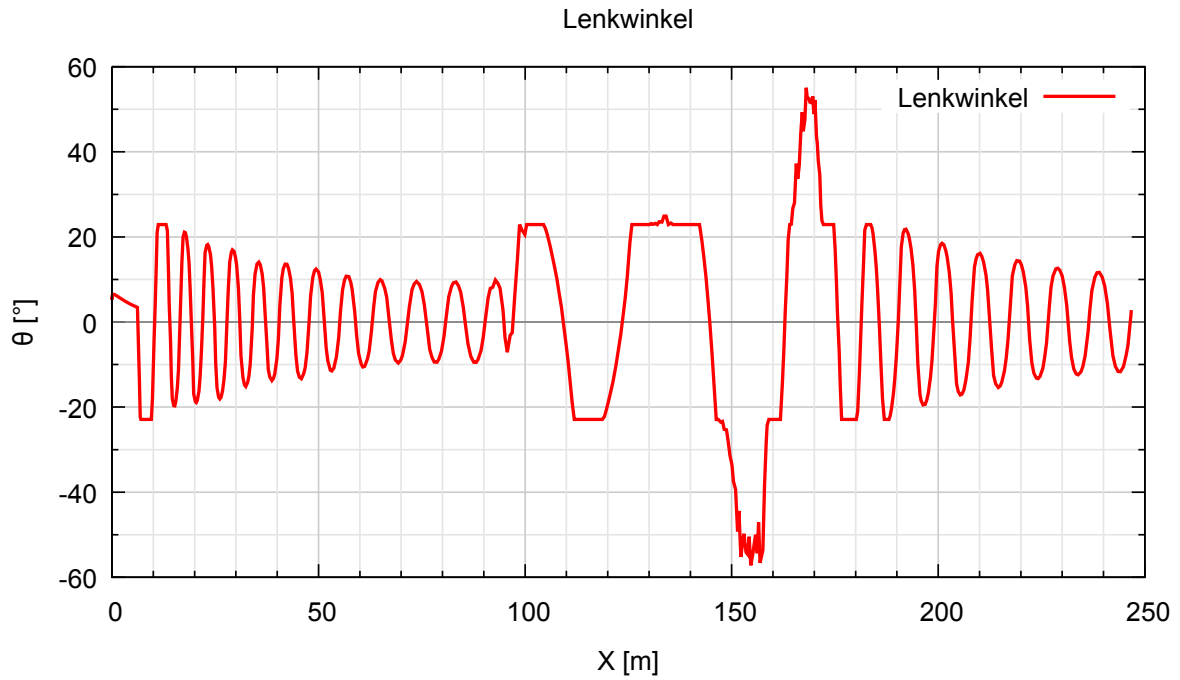
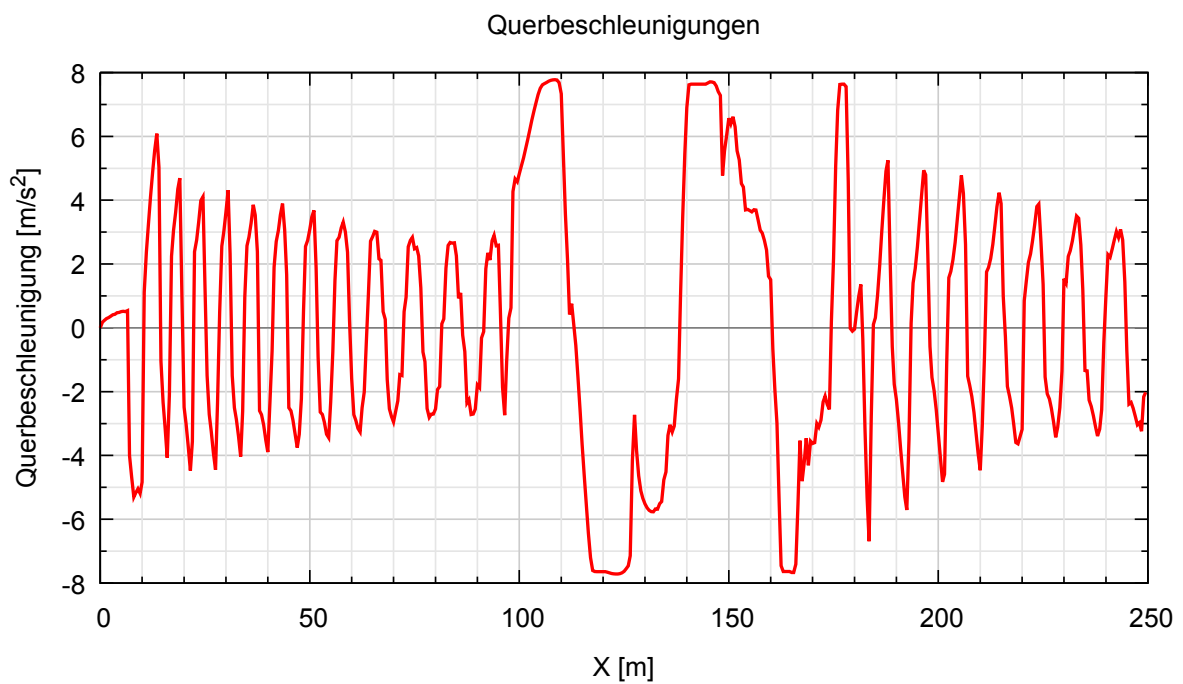


Abbildung 4.14: Lenkwinkel über die Simulationsstrecke

Abbildung 4.15: Querbeschleunigungen des Fahrzeuges bei Spursprung und  $v = 15\text{m/s}$

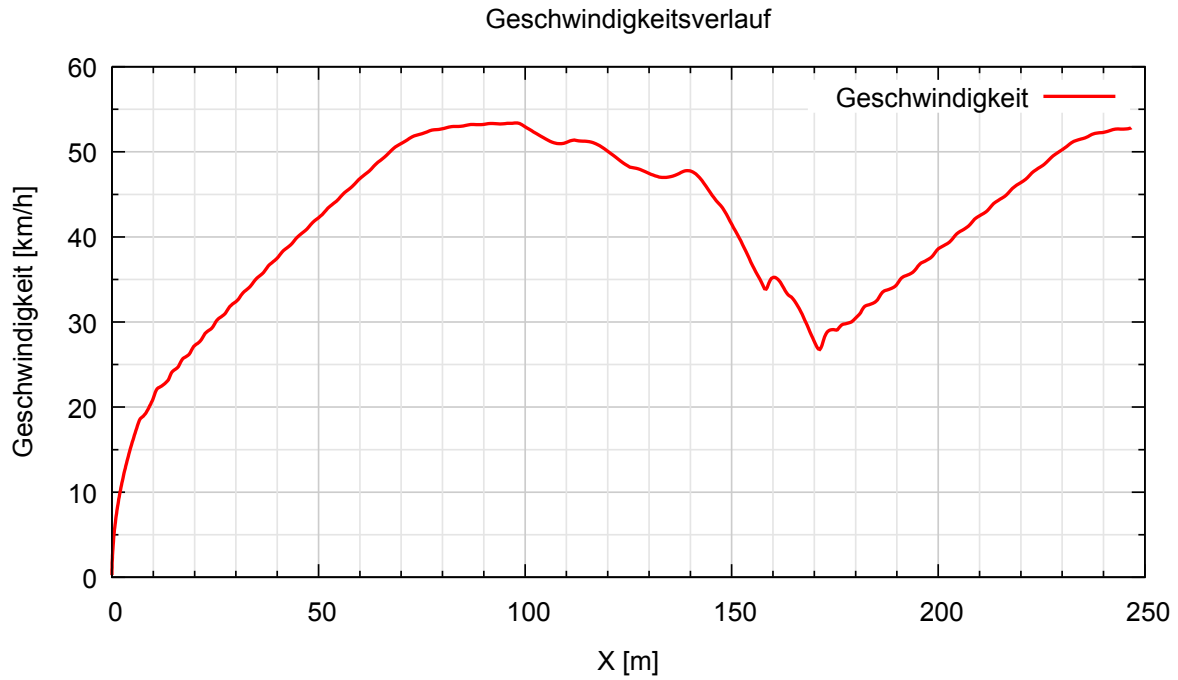


Abbildung 4.16: Geschwindigkeitsverlauf über die Simulationsstrecke

### Gütekriterien

Kriterium	Wert
Bleibende Regeldifferenz	0.026m
Anstiegsdistanz $D_r$	15.55m
Ausregeldistanz $D_s$	16.09m
Überschwingen	8.59%

Tabelle 4.9: Numerische Werte der Gütekriterien

## 4.6 Simulationsergebnisse - Pure Pursuit Steering

Test01 - Spurwechsel  $v = 10m/s$

Parameter	Wert
Abtastzeit	50ms
$\theta_{min}$	-0.4rad
$\theta_{max}$	0.4rad
$t_{LA}$	0.5s
$v_{start}$	10ms

Tabelle 4.10: Auflistung der Reglerparameter für Test01 und Test02

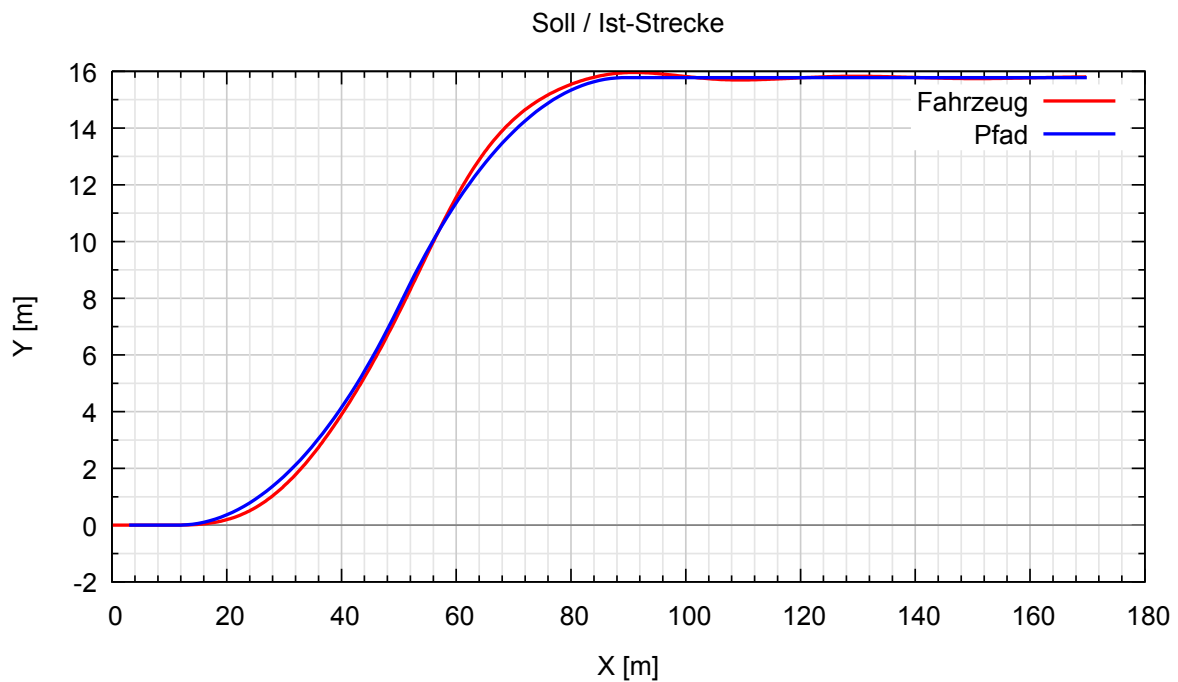


Abbildung 4.17: Soll/Ist-Weg

Auf Abbildung 4.17 ist zu erkennen, dass die Pure Pursuit Regelung bei geringen Geschwindigkeiten sehr gut verwendbar ist. Das ist darauf zurückzuführen, dass die geschwindigkeitsabhängige Verstärkung des in diesem Regelalgorithmus definierten P-Reglers noch gering genug ist, um ein Überschwingen zu verursachen.

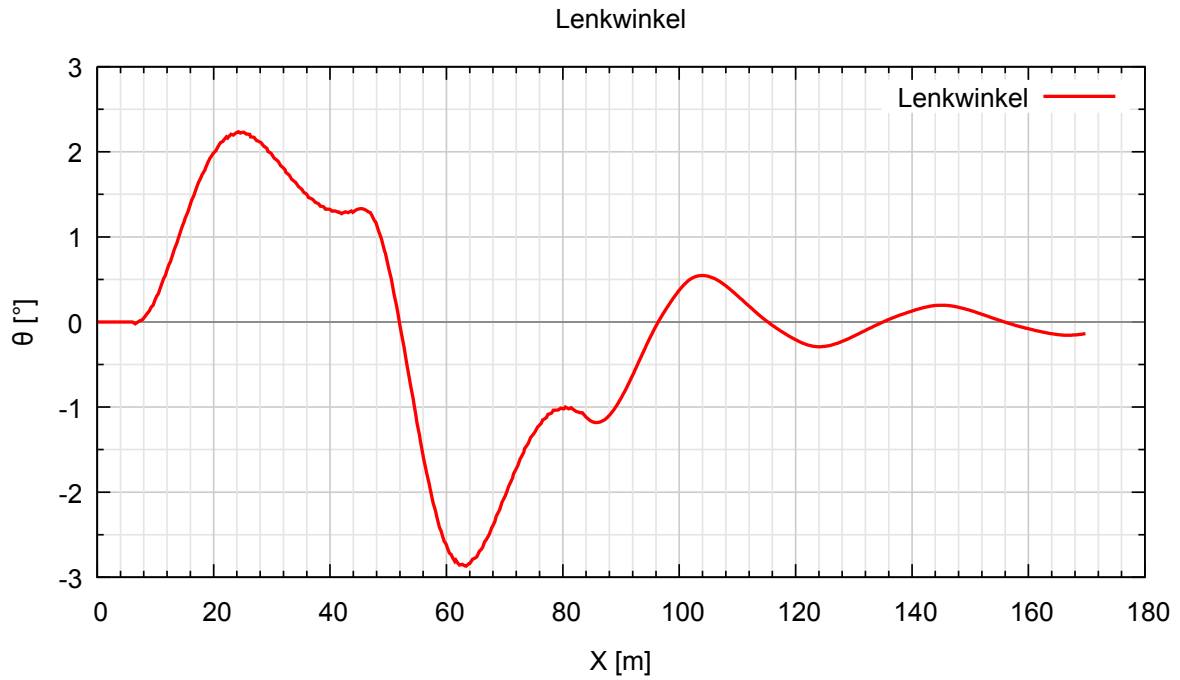
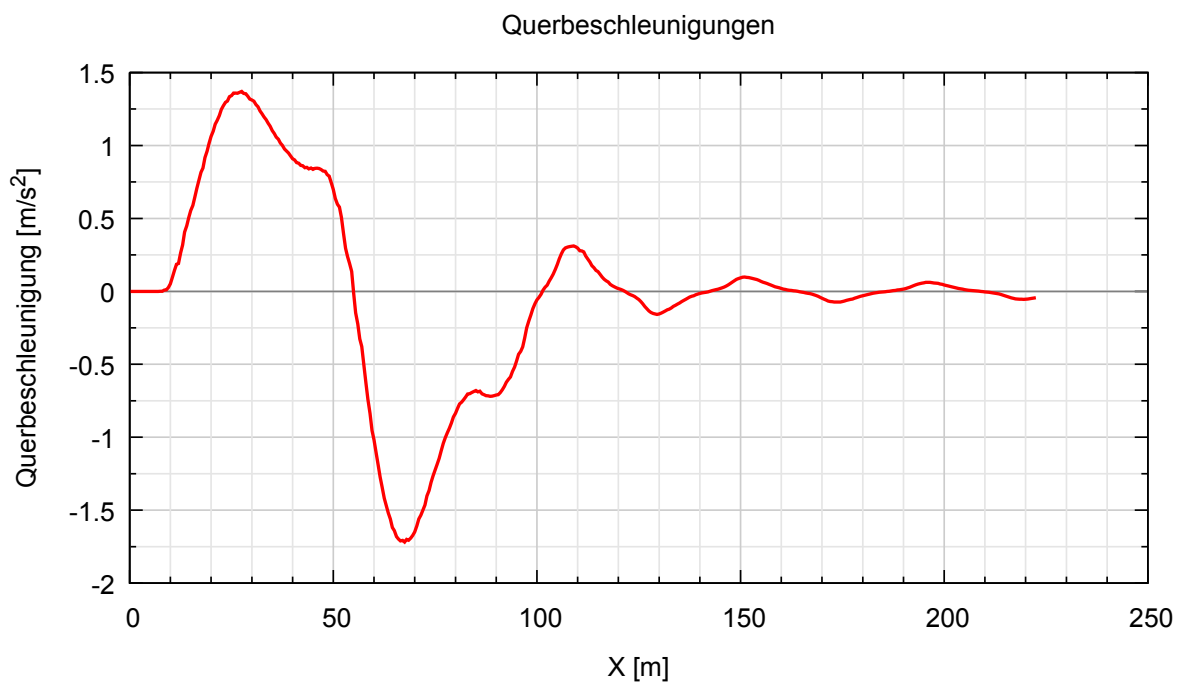


Abbildung 4.18: Lenkwinkel über die Simulationsstrecke

Abbildung 4.19: Querbeschleunigungen des Fahrzeuges bei Spurwechsel und  $v = 10\text{m/s}$

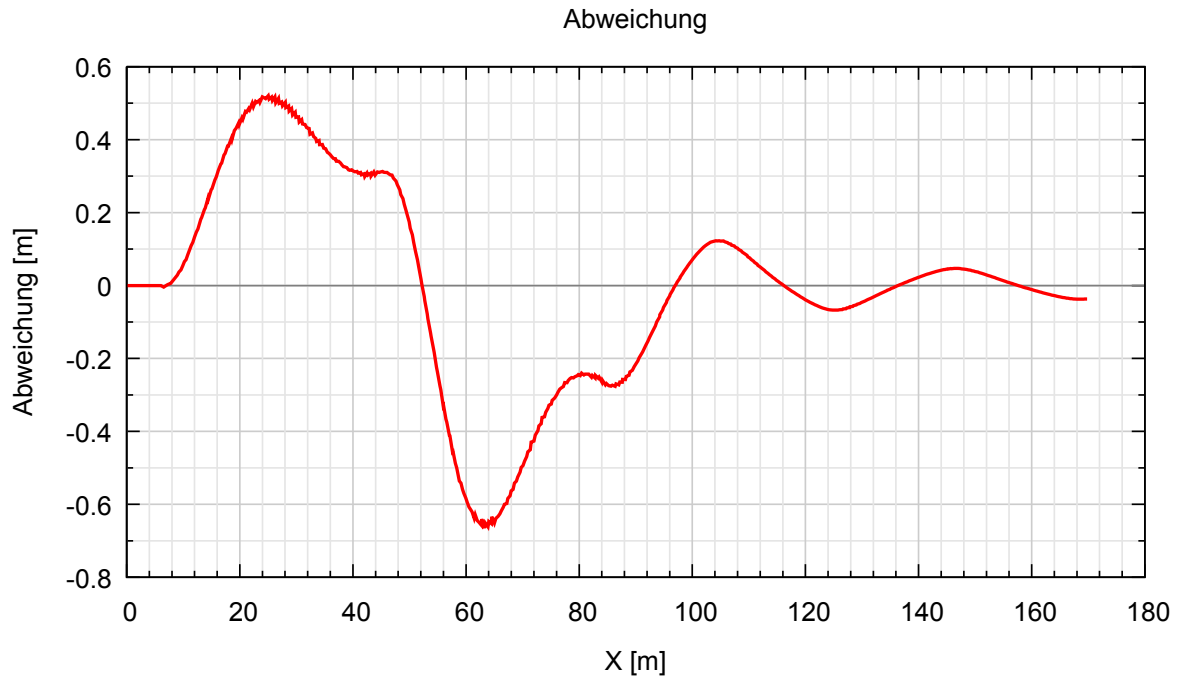


Abbildung 4.20: Regelfehler über die Strecke

Die in Abbildung 4.20 erkennbare Abweichung des Pure Pursuit Algorithmus ist mit maximal 60 cm noch vertretbar. Jedoch ist die Abweichung bei dieser Geschwindigkeit die höchste aller verglichenen Reglertypen.

**Test02 - Spurwechsel**  $v = 20\text{m/s}$

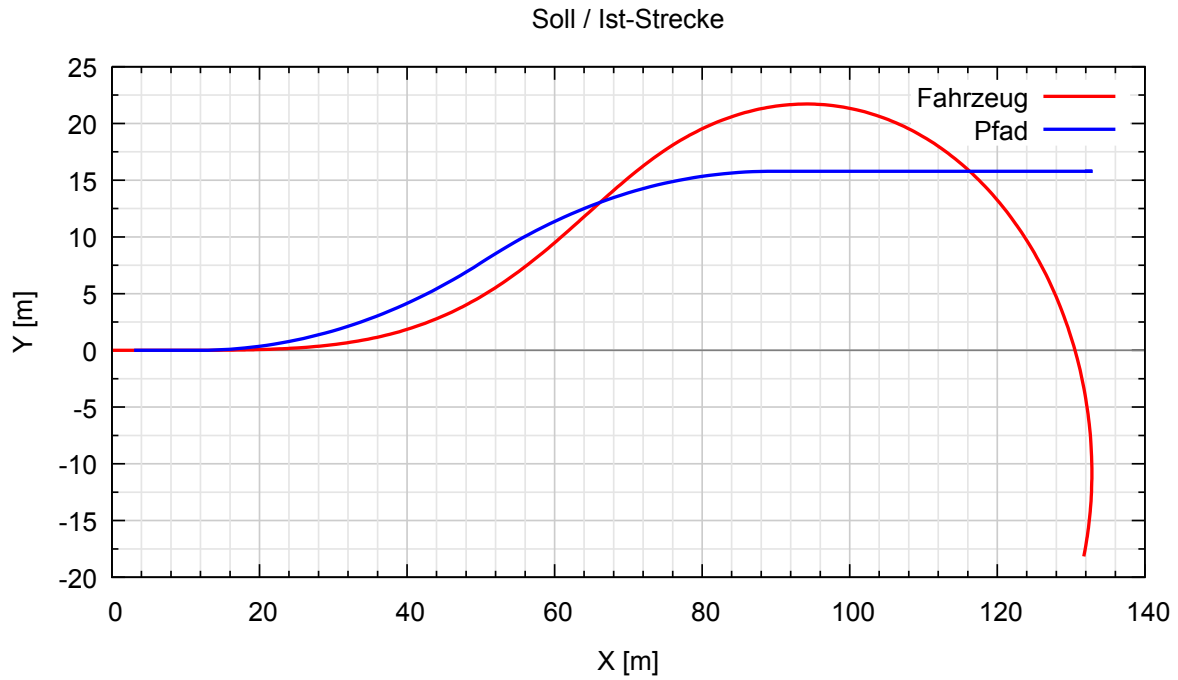


Abbildung 4.21: Soll/Ist-Weg

Betrachtet man den in Abbildung 4.21 dargestellten Verlauf der Fahrzeugbewegung, so ist zu erkennen, dass das Fahrzeug beim Zurücklenken die Kontrolle verloren hat. Dieses Verhalten ist daraus zu erklären, dass die durch den Lenkwinkel abhängige Verstärkung des P-Reglers des Pure Pursuit Reglers zu groß für ein stabiles Zurücklenken wird. Durch zusätzliche Dämpfungsglieder könnte dieses Verhalten möglicherweise verbessert werden.



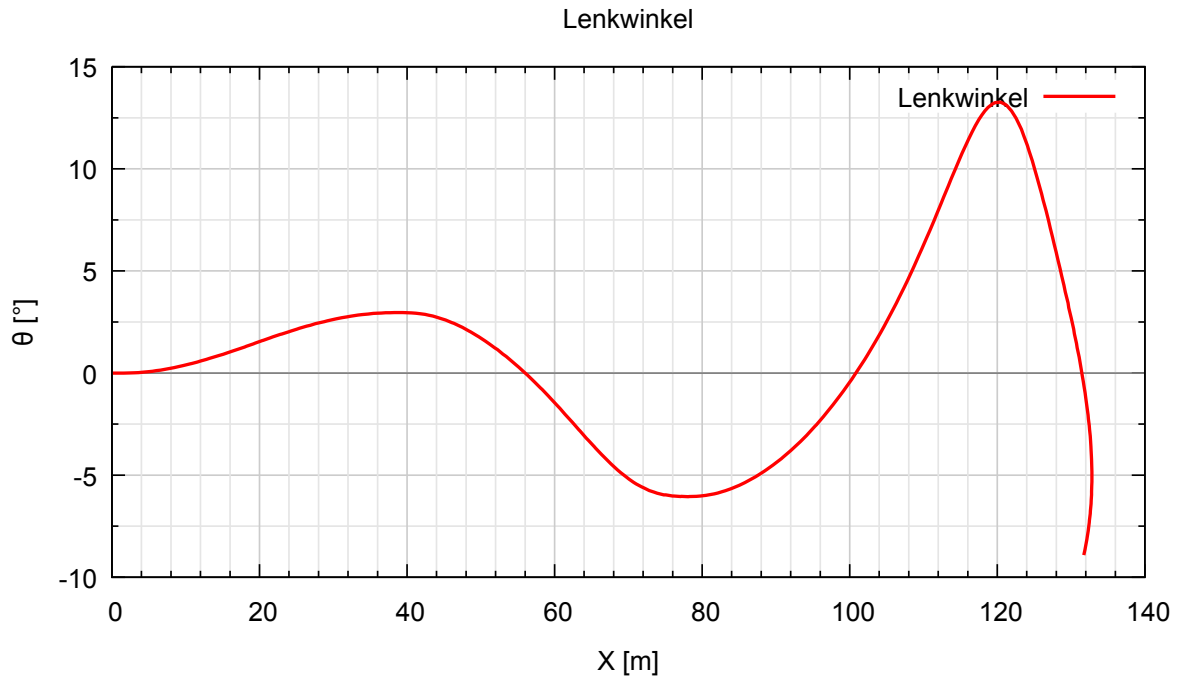
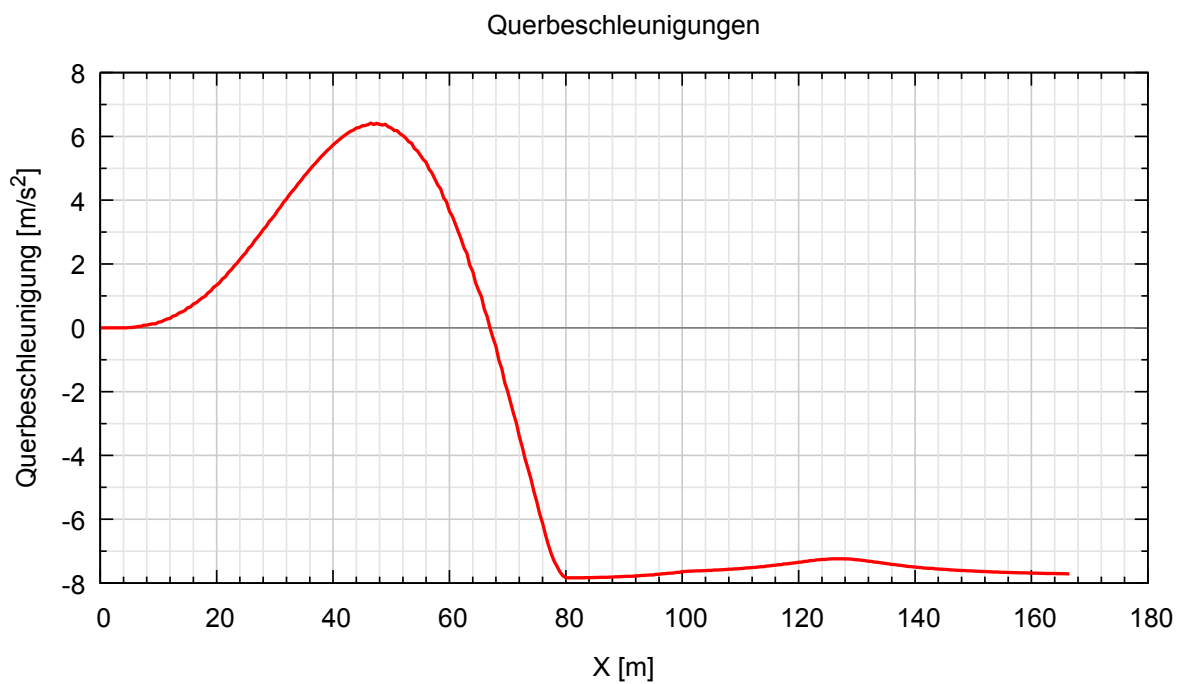


Abbildung 4.22: Lenkwinkel über die Simulationsstrecke

Abbildung 4.23: Querbeschleunigungen des Fahrzeuges bei Spurwechsel und  $v = 20\text{m/s}$

Der Verlauf der Querbeschleunigungen zeigt deutlich, dass das Fahrzeug, beim Versuch wieder auf Spur zu kommen, ausbricht. Bei einer absoluten Querbeschleunigung von über  $6m/s^2$  ist ein stabiles Fahren selbst bei einem Fahrbahnreibungskoeffizienten von  $\mu = 0.8$ , der eine trockene Fahrbahn beschreibt, nicht mehr möglich.

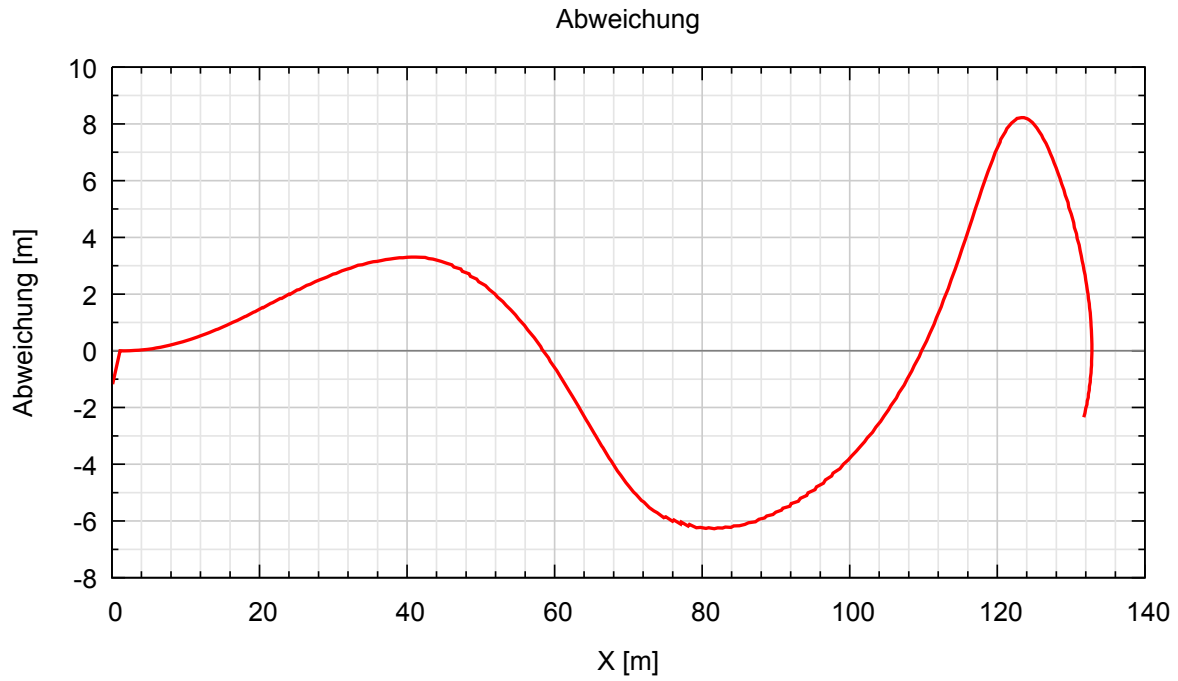


Abbildung 4.24: Regelfehler über die Strecke

**Test03 - Spursprung**  $v = 15m/s$

Parameter	Wert
Abtastzeit	$50ms$
$\theta_{min}$	$-0.4rad$
$\theta_{max}$	$0.4rad$
$t_{LA}$	$1s$

Tabelle 4.11: Auflistung der Reglerparameter für Test03

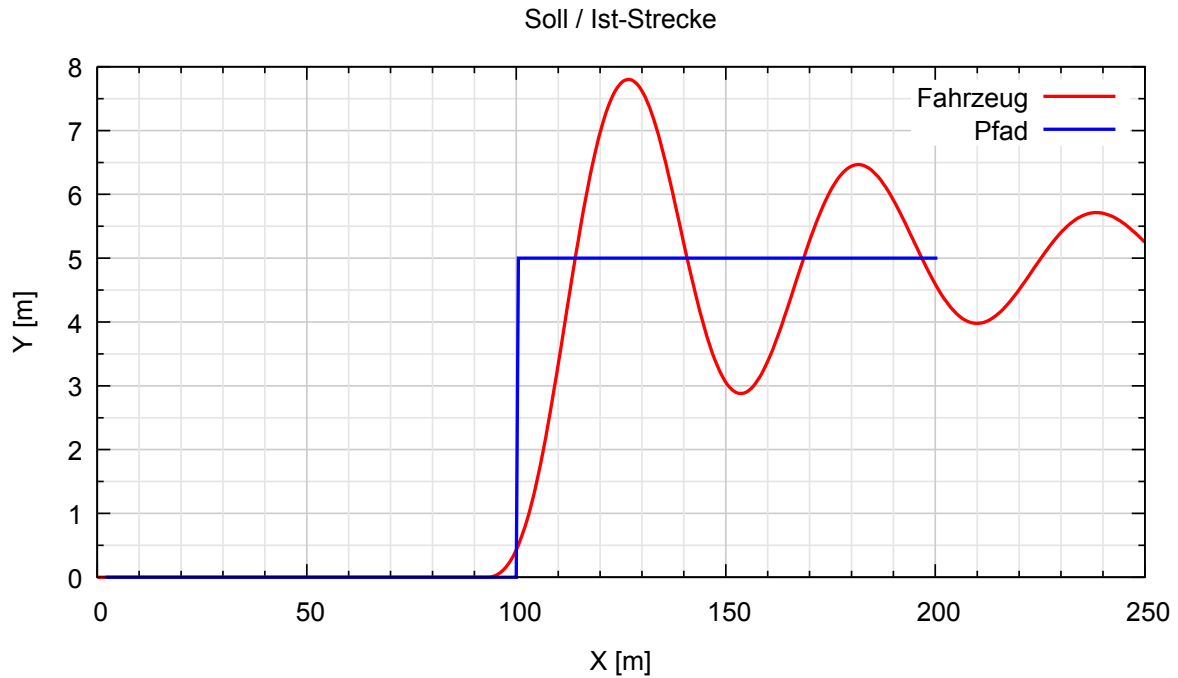


Abbildung 4.25: Soll/Ist-Weg

Die Bewegung des Fahrzeuges während des Sprungversuches kann auf Abbildung 4.25 betrachtet werden. Es ist deutlich ein sehr großes Überschwingen nach dem Spursprung zu erkennen. Aufgrund der weiter oben beschriebenen abweichungsabhängigen Verstärkung kann dieser Reglertyp Sprünge schlecht ausgleichen. Eine Beschränkung der Steigung der Stellgröße würde hier eine Verbesserung bringen. Jedoch bleibt nach der Definition des Reglertyps die hohe Verstärkung des P-Reglers zurück.

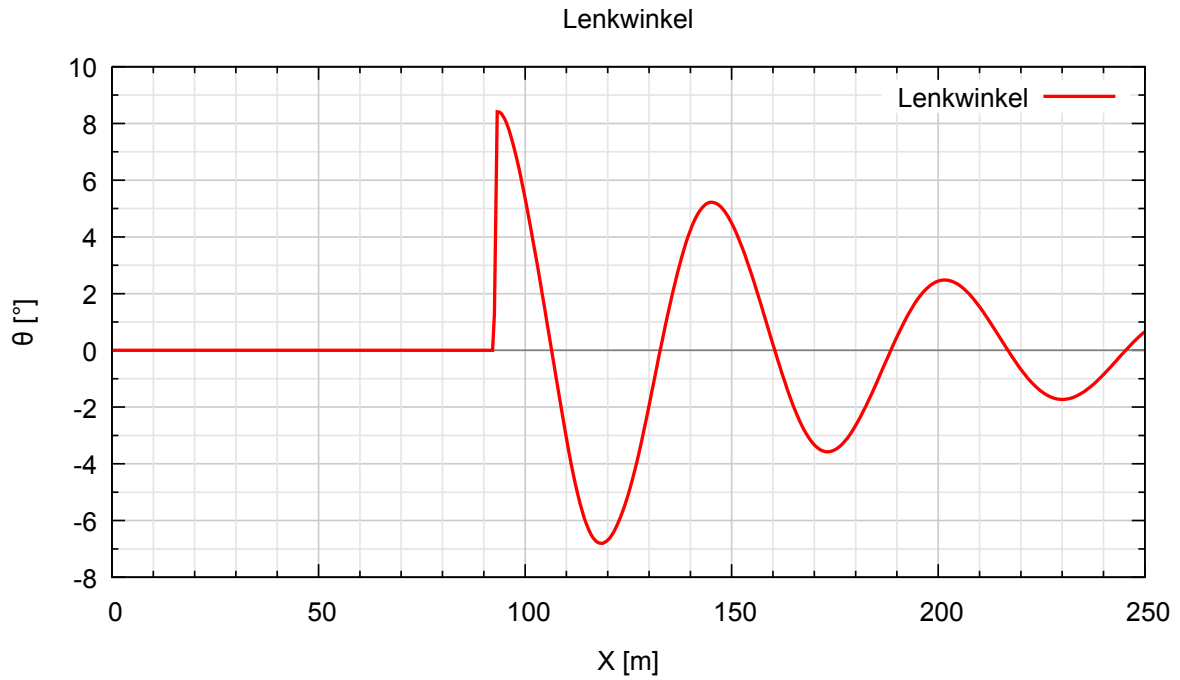
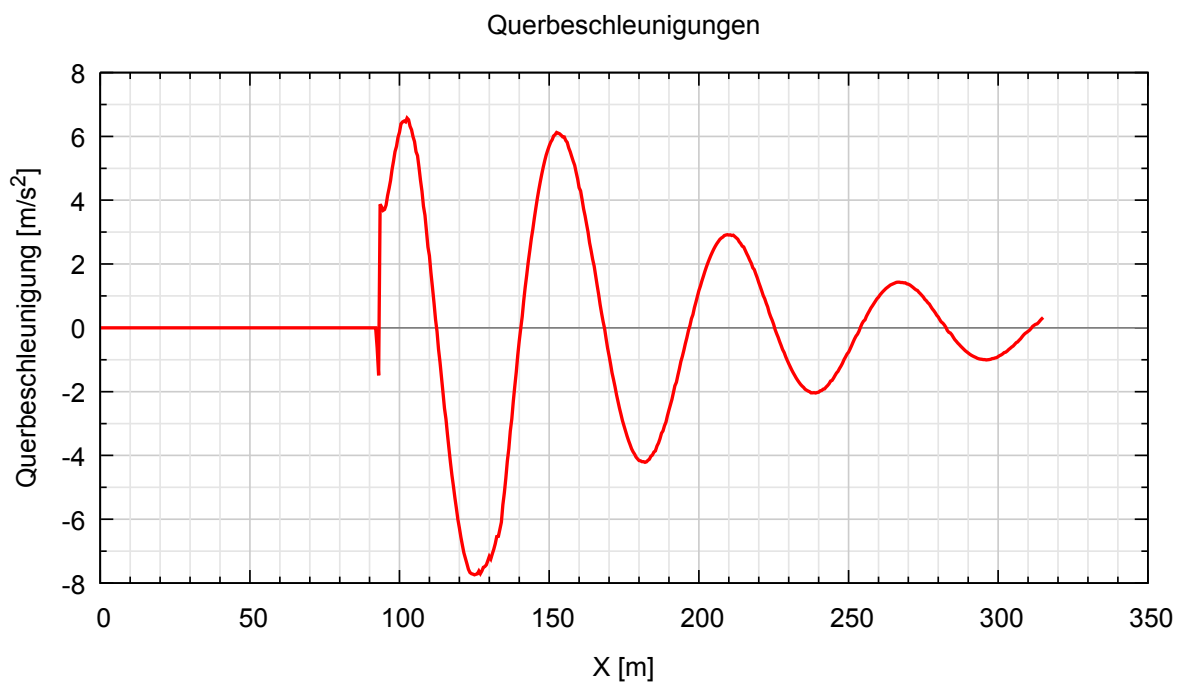


Abbildung 4.26: Lenkwinkel über die Simulationsstrecke

Abbildung 4.27: Querbeschleunigungen des Fahrzeuges bei Spursprung und  $v = 15\text{m/s}$

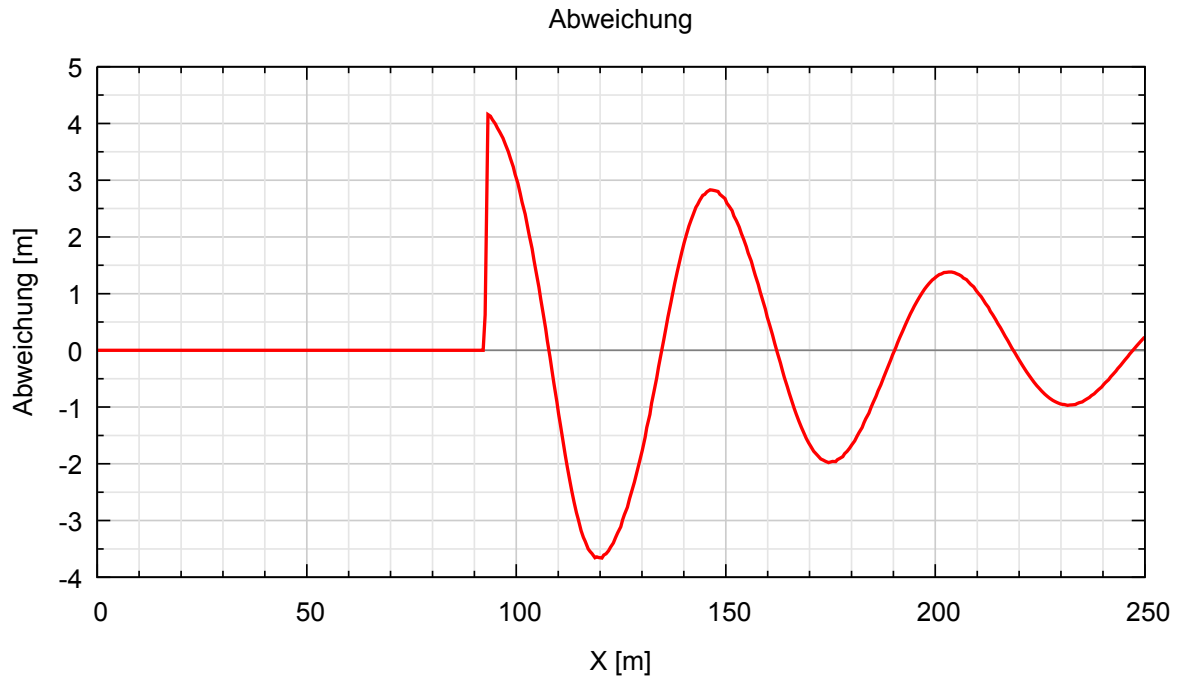


Abbildung 4.28: Regelabweichung des Fahrzeuges

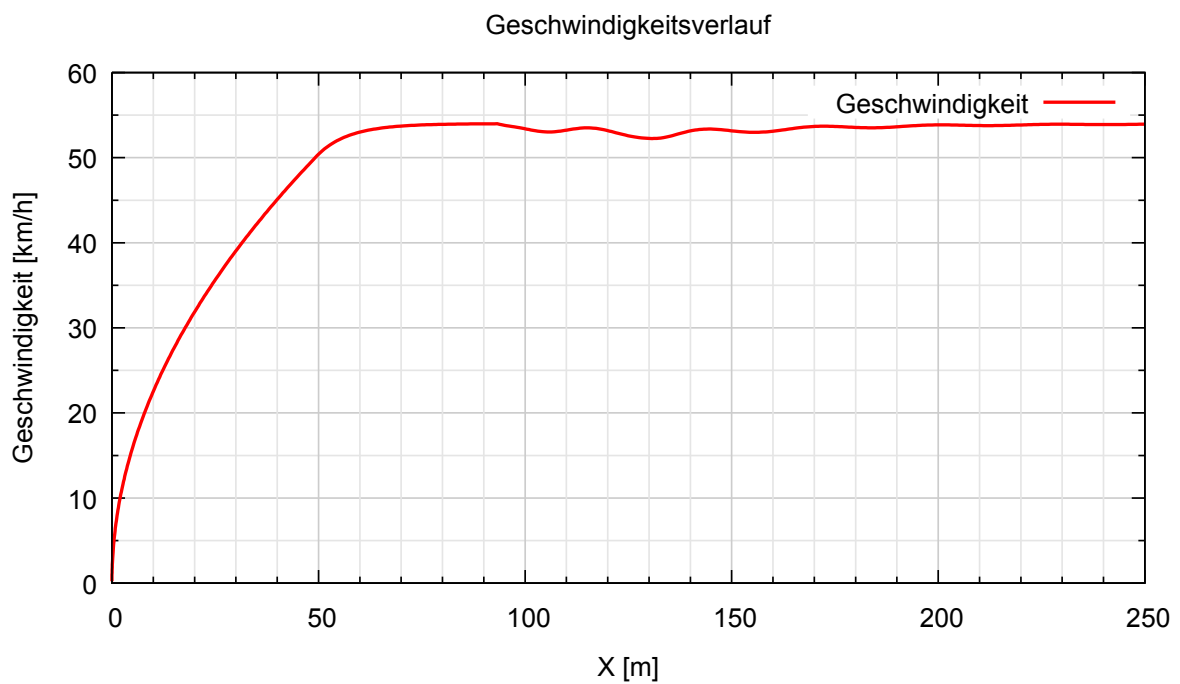


Abbildung 4.29: Geschwindigkeitsverlauf über die Simulationsstrecke

In Abbildung 4.17 kann man erkennen, dass der Pure Pursuit-Regelalgorithmus genügend Stabilität aufweist. Auch ist nur ein geringes Überschwingen ablesbar. In Abbildung 4.18 lässt sich erkennen, dass der Regler, durch die Anpassung des Lenkwinkels, ein leichtes Nachschwingen vermindern will. Die Abweichung liegt im Bereich von  $\pm 0.6m$

Erhöht man nun die Geschwindigkeit des Fahrzeuges, so wird in Abbildung 4.21 deutlich, dass eine stabile Spurverfolgung nicht mehr möglich ist. Abbildung 4.23 zeigt, dass bei hohen Geschwindigkeiten die Querbeschleunigungen zu hoch für eine stabile Kurvenfahrt sind.

Bei der Simulation des Spursprunges mit  $v = 15m/s$  ist in Abbildung 4.25 deutlich erkennbar, dass das Fahrzeug nach einem sehr großen Überschwingen, die Oszillation um den Sollwert, auf dem in der für die Simulation vorgesehen Pfadabschnitt, nicht mehr vermindern kann.

### Gütekriterien

Kriterium	Wert
Bleibende Regeldifferenz	$0.45m$
Anstiegsdistanz $D_r$	$13.87m$
Ausregeldistanz $D_s$	$14.5m$
Überschwingen	$14.04\%$

Tabelle 4.12: Numerische Werte der Gütekriterien

## 4.7 Simulationsergebnisse - Lookahead Steering

Zur Durchführung der Simulationen mit dem Lookahead Steering-Regelungsalgorithmus wurde folgende Reglerparametrierung definiert:

Parameter	Wert
Abtastzeit	$50ms$
$K_p$	$0.7$
$\theta_{min}$	$-0.4rad$
$\theta_{max}$	$0.4rad$
$t_{LA}$	$0.5s$

Tabelle 4.13: Auflistung der Reglerparameter für Test01

Test01  $v = 10\text{m/s}$

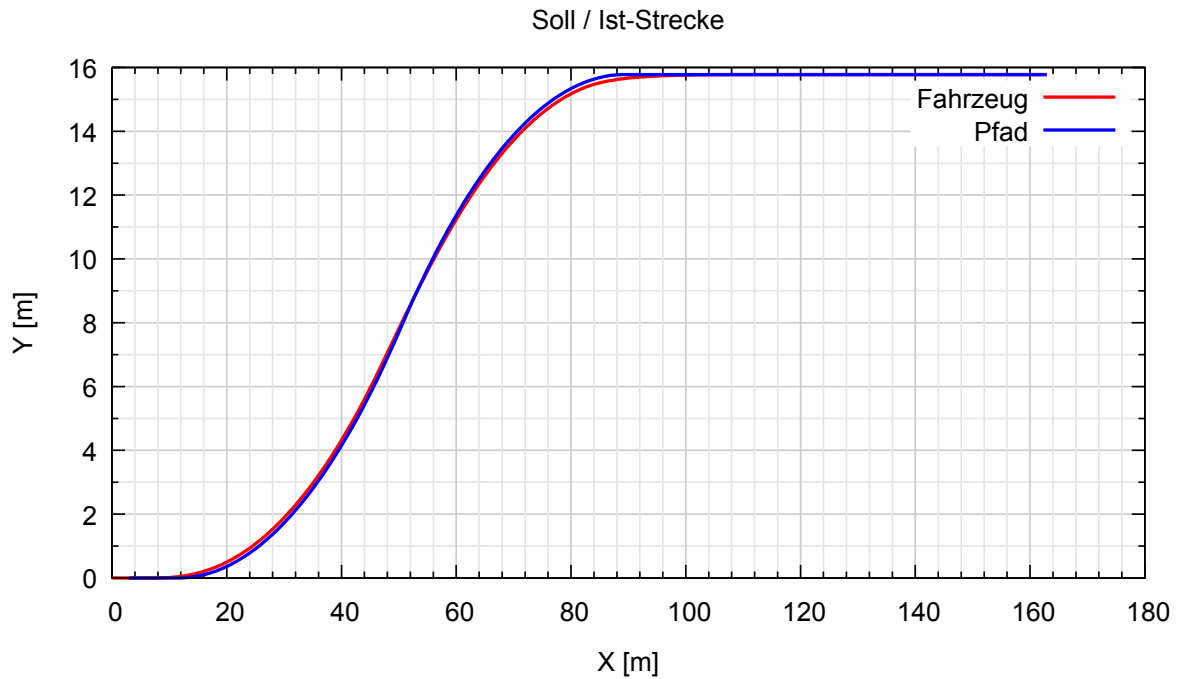


Abbildung 4.30: Soll/Ist-Weg

In Abbildung 4.30 lässt sich der zurückgelegte Pfad im Vergleich zum Soll-Pfad erkennen. Wie auch bei den vorherigen Auswertungen sind die beiden Pfade gering abweichend. Die kleinen Unterschiede, vor allem das „Kurvenschneiden“, kommen durch den 0.5s verschobenen Vorrasschaupunkt zustande.

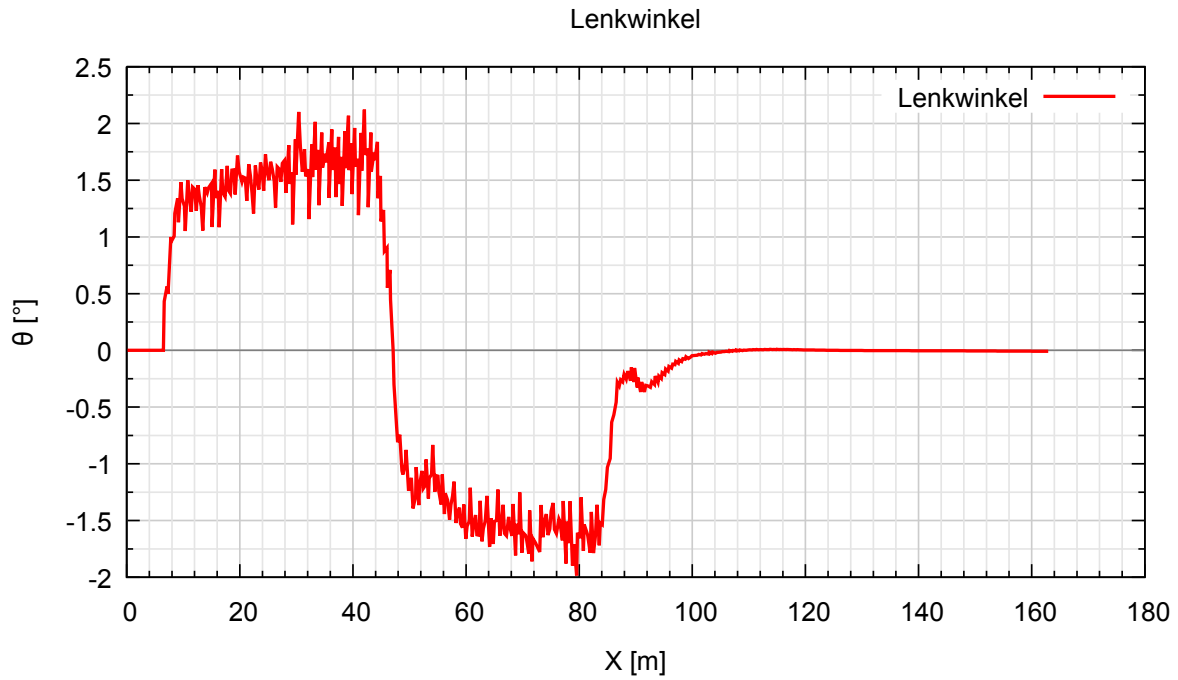
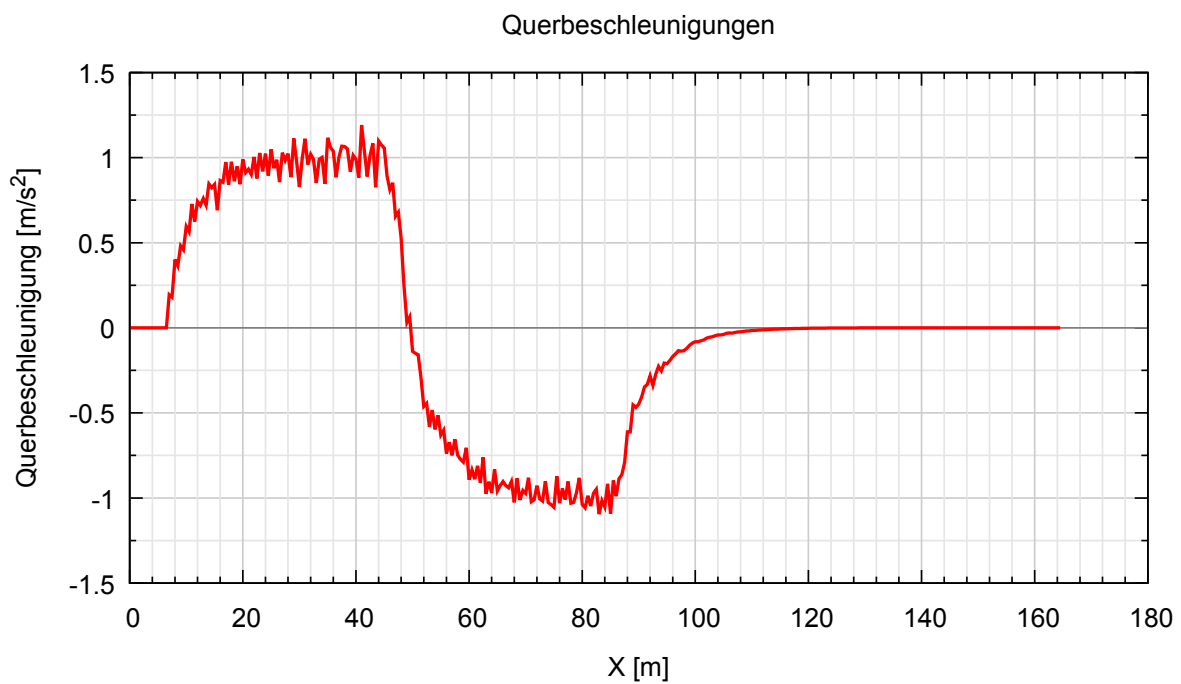


Abbildung 4.31: Lenkwinkel über die Simulationsstrecke

Abbildung 4.32: Querbeschleunigungen des Fahrzeuges bei Spurwechsel und  $v = 10\text{m/s}$



Die Querbewegung des Fahrzeuges ist sehr gering, was auf die kleinen Lenkbewegungen, wie in Abbildung 4.31 zu sehen ist, zurückzuführen ist.

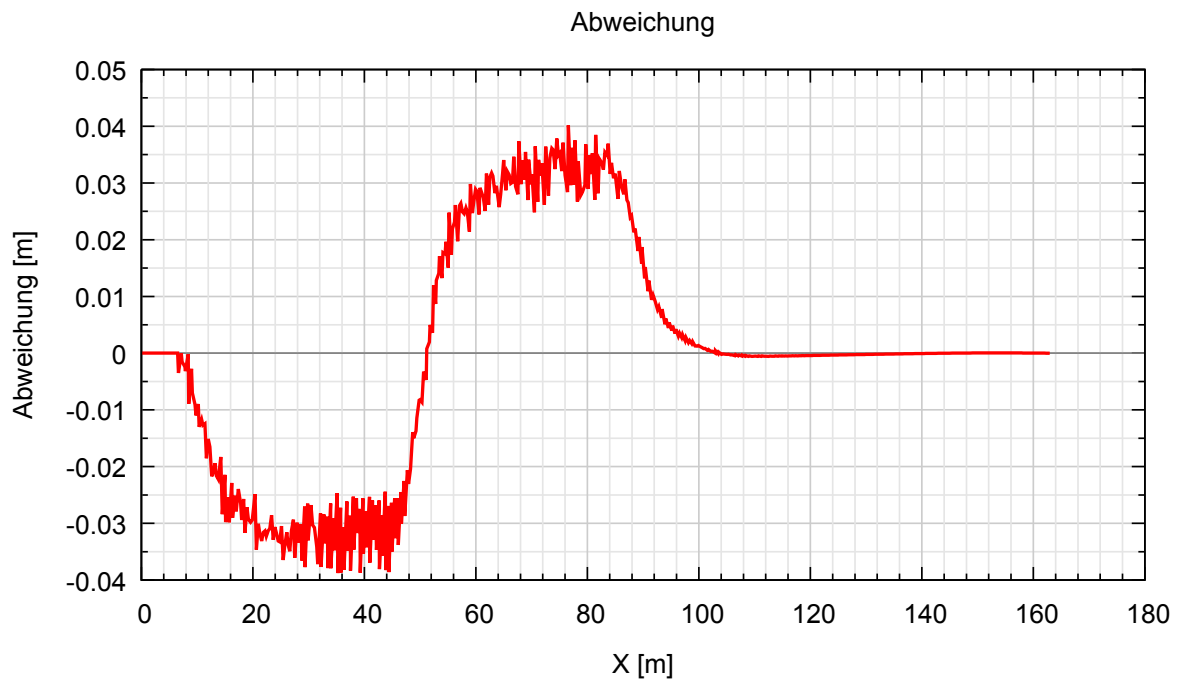


Abbildung 4.33: Regelfehler über die Strecke

Die Abweichung von der Sollgröße ist mit unter 4 mm als sehr gut zu bewerten. die Abweichungsausschläge sind jeweils beim Einfahren und Ausfahren aus der Kurve zu erkennen.

**Test02**  $v = 20m/s$

Parameter	Wert
Abtastzeit	50ms
$K_p$	0.5
$K_d$	0.1
$\theta_{min}$	-0.4rad
$\theta_{max}$	0.4rad
$t_{LA}$	0.5s

Tabelle 4.14: Auflistung der Reglerparameter für Test02

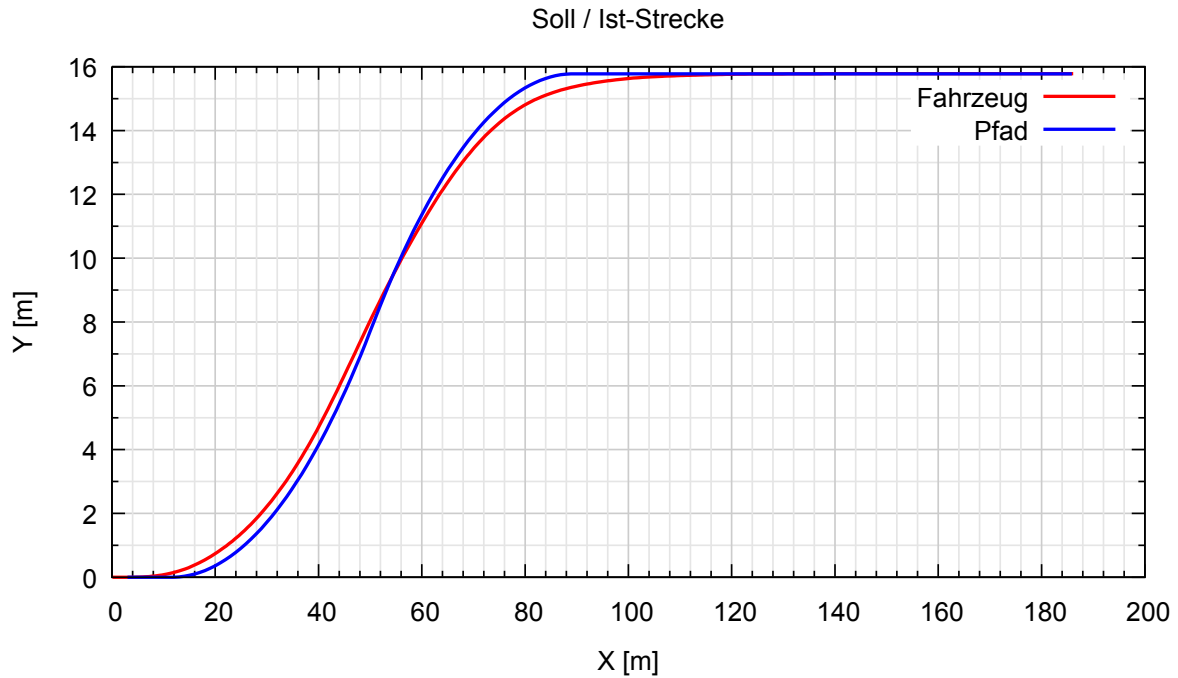


Abbildung 4.34: Soll/Ist-Weg

In Abbildung 4.34 kann man erkennen, dass das „Kurevnenschneiden“ im Vergleich zum vorherigen Versuch, ersichtlich in Abbildung 4.30, deutlich stärker ist. Das ergibt sich daraus, dass die Lookahead Distanz geschwindigkeitsabhängig angepasst wird. Daher ist bei einer höheren Geschwindigkeit auch der Referenzpunkt weit vor dem Fahrzeug.

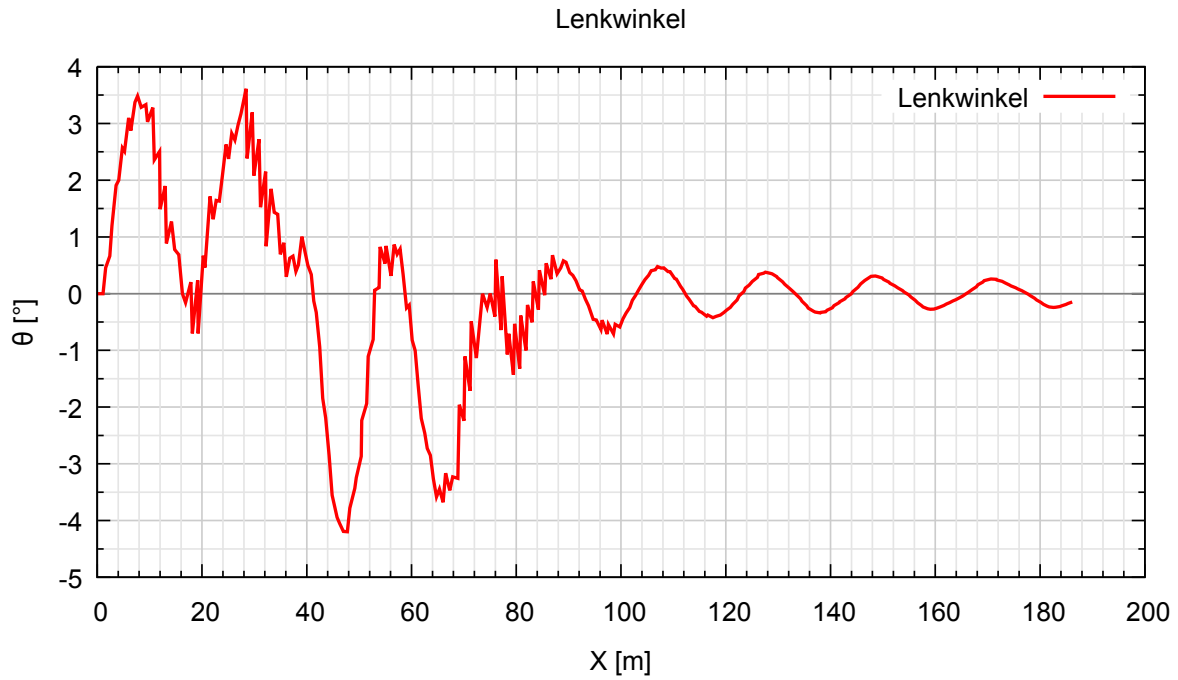


Abbildung 4.35: Lenkwinkel über die Simulationsstrecke

Bei der Betrachtung des Verlaufes des Lenkwinkels, kann man erkennen, dass sich nach ca 100m ein niederfrequentes Schwingen abzeichnet. Dieses Schwingen resultiert aus der hohen Abtastzeit. Aufgrund einer Amplitude von  $0.3^\circ$  kann diese jedoch vernachlässigt werden.

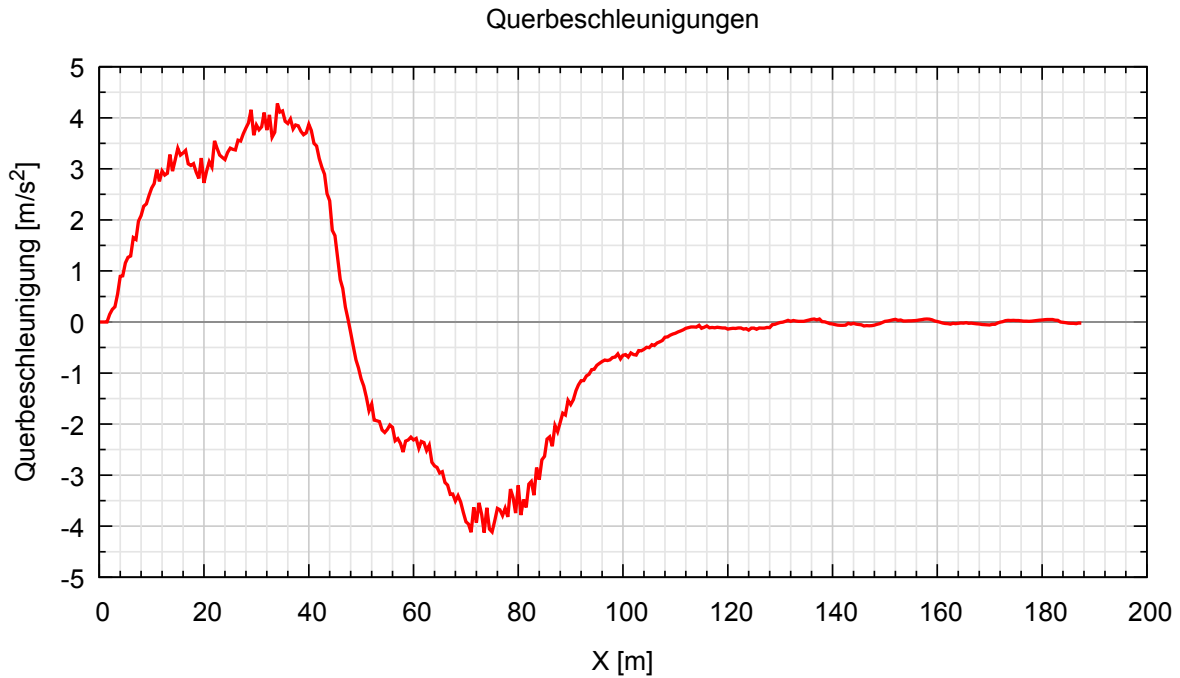


Abbildung 4.36: Querbeschleunigungen des Fahrzeuges bei Spurwechsel und  $v = 20\text{m/s}$

Die Querbeschleunigungen beim Einlenken und Zurücklenken in den Spurwechsel, sind bei  $v = 20\text{m/s}$  ungefähr viermal höher als bei Test01 mit  $v = 10\text{m/s}$ . Jedoch sind  $4\text{m/s}^2$  bei trockener Fahrbahn noch in der Toleranz für ein stabiles Fahrverhalten eines Fahrzeuges mit durchschnittlich hohem Schwerpunkt.

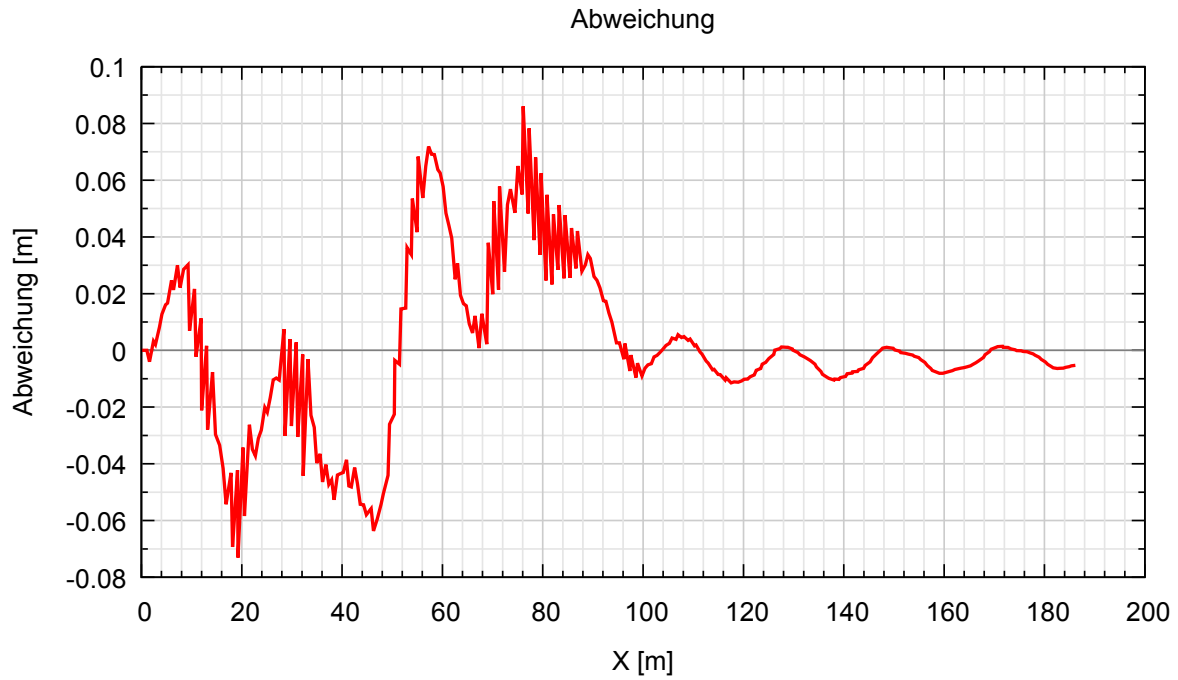


Abbildung 4.37: Regelfehler über die Strecke

Die Abweichung beträgt maximal 8 cm und ist damit für einen Fahrversuch mit hoher Geschwindigkeit als sehr gut zu bewerten.

**Test03**  $v = 15\text{m/s}$

Parameter	Wert
Abtastzeit	$50\text{ms}$
$K_p$	0.5
$K_d$	0
$\theta_{min}$	$-0.4\text{rad}$
$\theta_{max}$	$0.4\text{rad}$
$t_{LA}$	1s

Tabelle 4.15: Auflistung der Reglerparameter für Test03

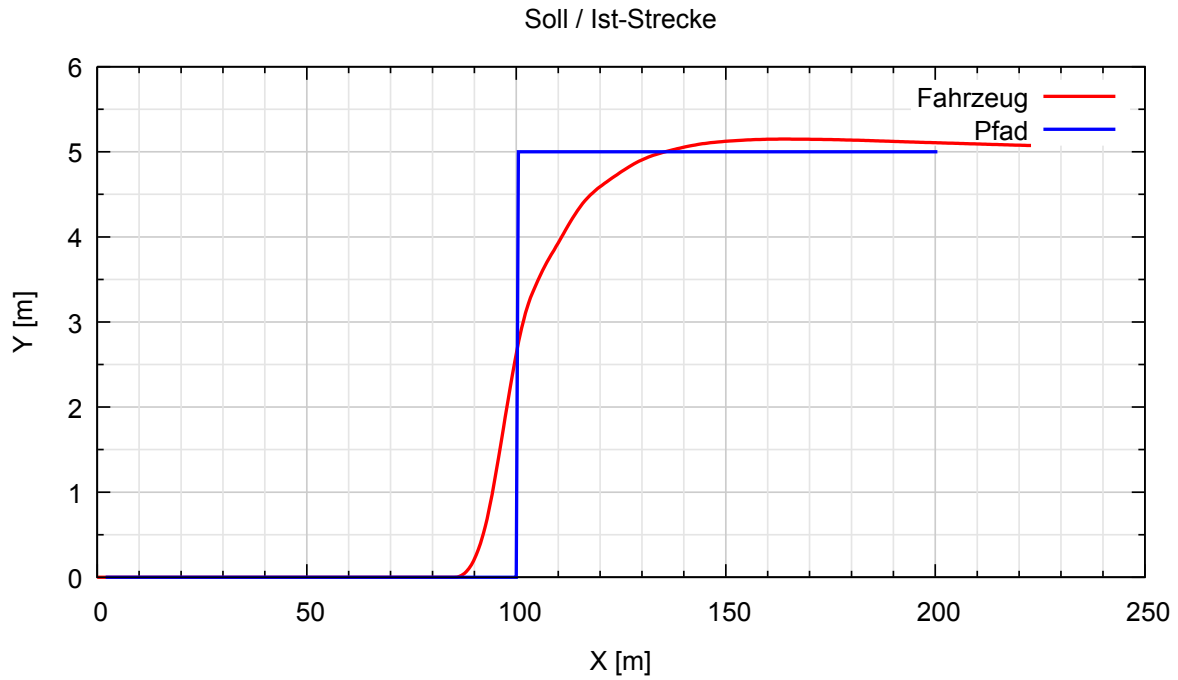


Abbildung 4.38: Soll/Ist-Weg

Anhand des Fahrzeugverlaufes in Abbildung 4.38 ist gut zu erkennen, dass durch das vorausschauende Regeln des Lookahead Steering Verfahrens das Fahrzeug den Spursprung durch früheres Einlenken ausgleicht. Dadurch kommt es zu einem geringen Überschwingen, jedoch auch zu einer hohen Abweichung vom Pfad, wie in Abbildung 4.41 zu sehen.

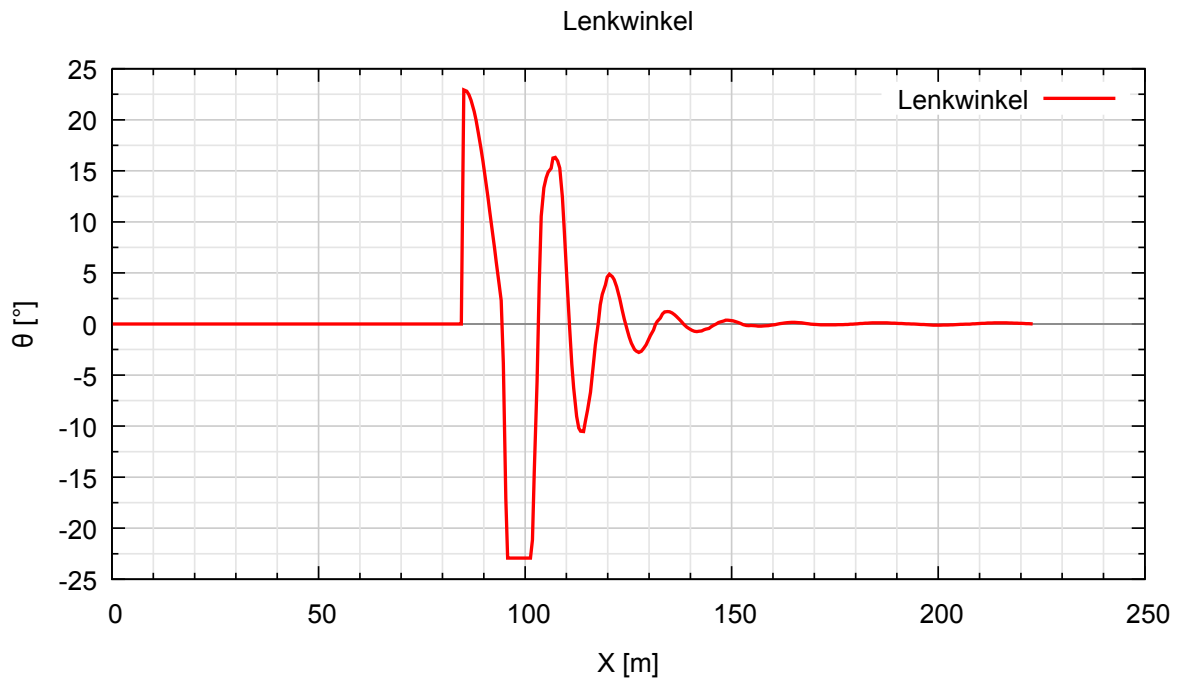


Abbildung 4.39: Lenkwinkel über die Simulationsstrecke

In Abbildung 4.41 ist deutlich zu erkennen, dass beim Zurücklenken des Fahrzeuges der Regelalgorithmus die Stellgröße bis zur Begrenzung ansetzt.

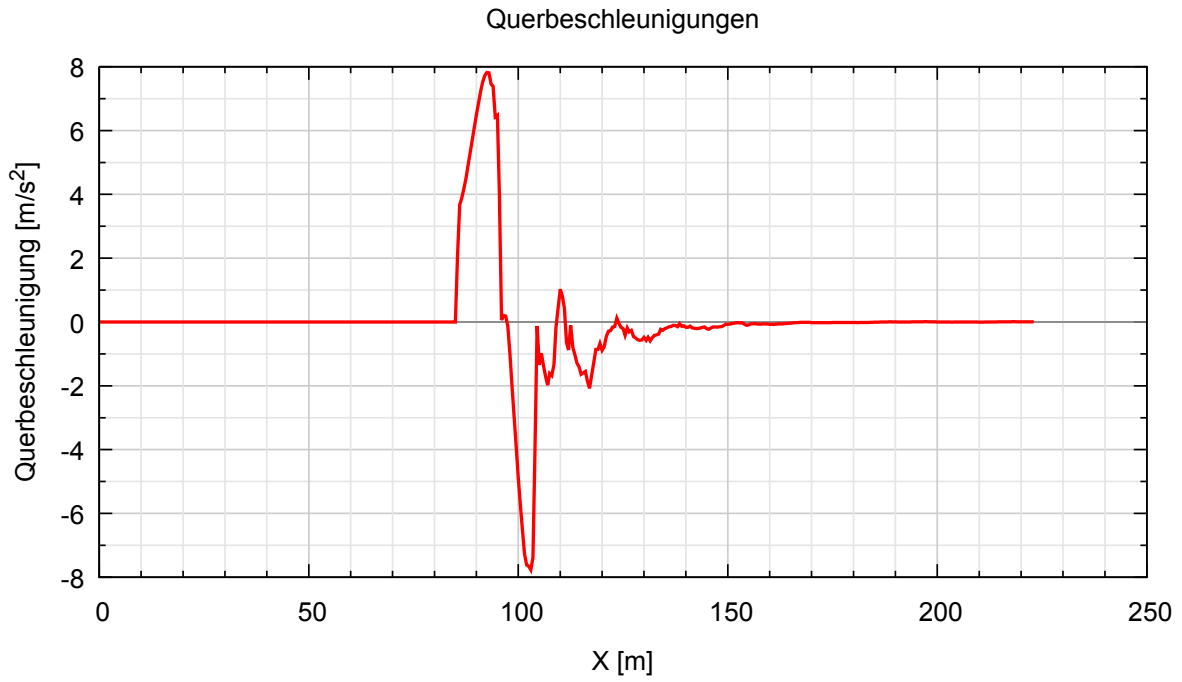
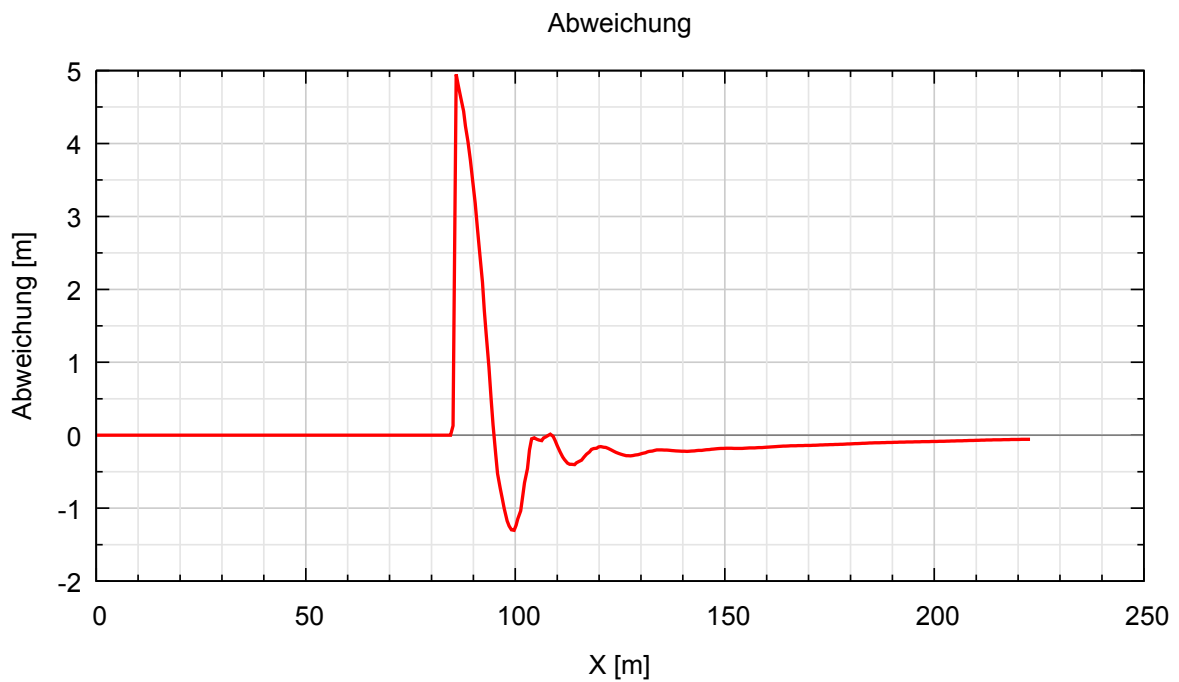
Abbildung 4.40: Querbeschleunigungen des Fahrzeuges bei Spursprung und  $v = 15\text{m/s}$ 

Abbildung 4.41: Regelfehler über die Strecke



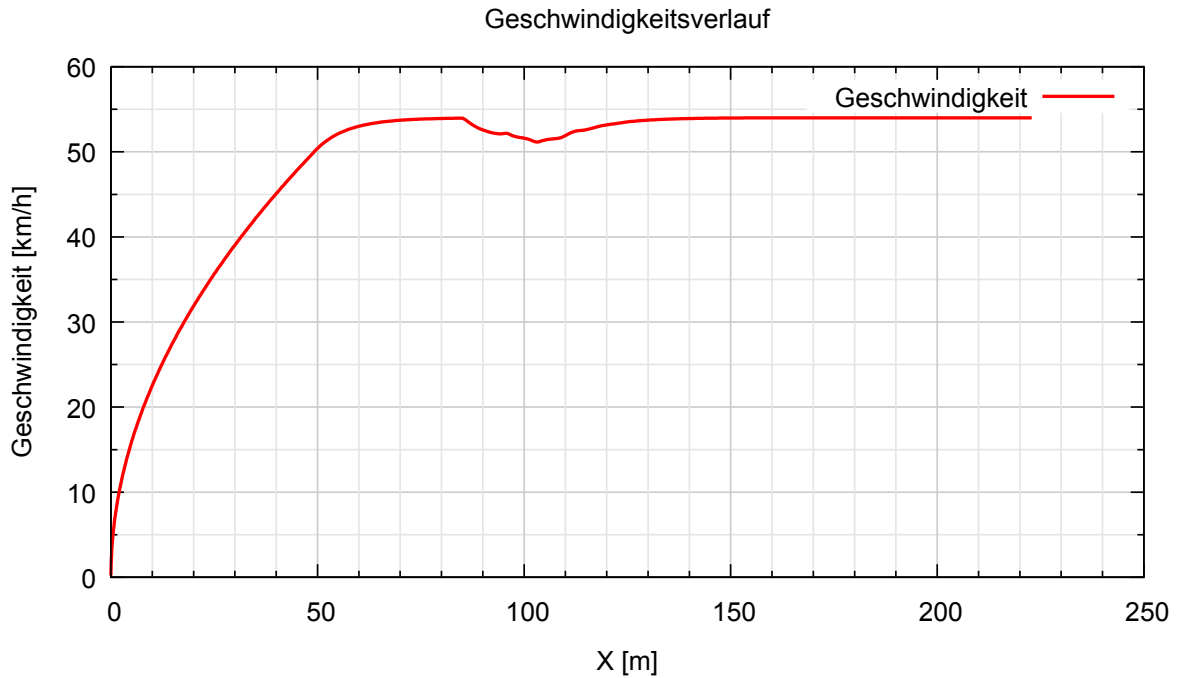


Abbildung 4.42: Geschwindigkeitsverlauf über die Simulationsstrecke

Bei der Lookahead Steering-Regelung ist zu erkennen, dass sich sowohl bei den schnellen, als auch langsamen Simulationsfahrten, das Überschwingen in Grenzen hält.

Auch die Abweichung des Fahrzeuges zum Pfad ist mit einem Maximalwert von 0.04m beim langsamen Spurwechsel bzw. 0.08m bei 20 m/s sehr gering. Beim Spursprungversuch kann man erkennen, dass das Fahrzeug nach kurzem Pendeln zügig wieder auf den Pfad kommt. Ebenfalls kann auf Abbildung 4.39 das Erreichen der Begrenzungen des Lenkwinkels abgelesen werden. Es wird sowohl die negative, als auch die die positive Lenkwinkelgrenze erreicht.

### Gütekriterien

Kriterium	Wert
Bleibende Regeldifferenz	0.077m
Anstiegsdistanz $D_r$	24.757m
Ausregeldistanz $D_s$	25.2m
Überschwingen	0.0747%

Tabelle 4.16: Numerische Werte der Gütekriterien

## 4.8 Simulationsergebnisse - Fuzzy-Regelung

Test01  $v = 10m/s$

Parameter	Wert
Abtastzeit	$50ms$
$\theta_{min}$	$-0.4rad$
$\theta_{max}$	$0.4rad$
$t_{LA}$	$0.5s$

Tabelle 4.17: Auflistung der Reglerparameter für Test01 und Test02

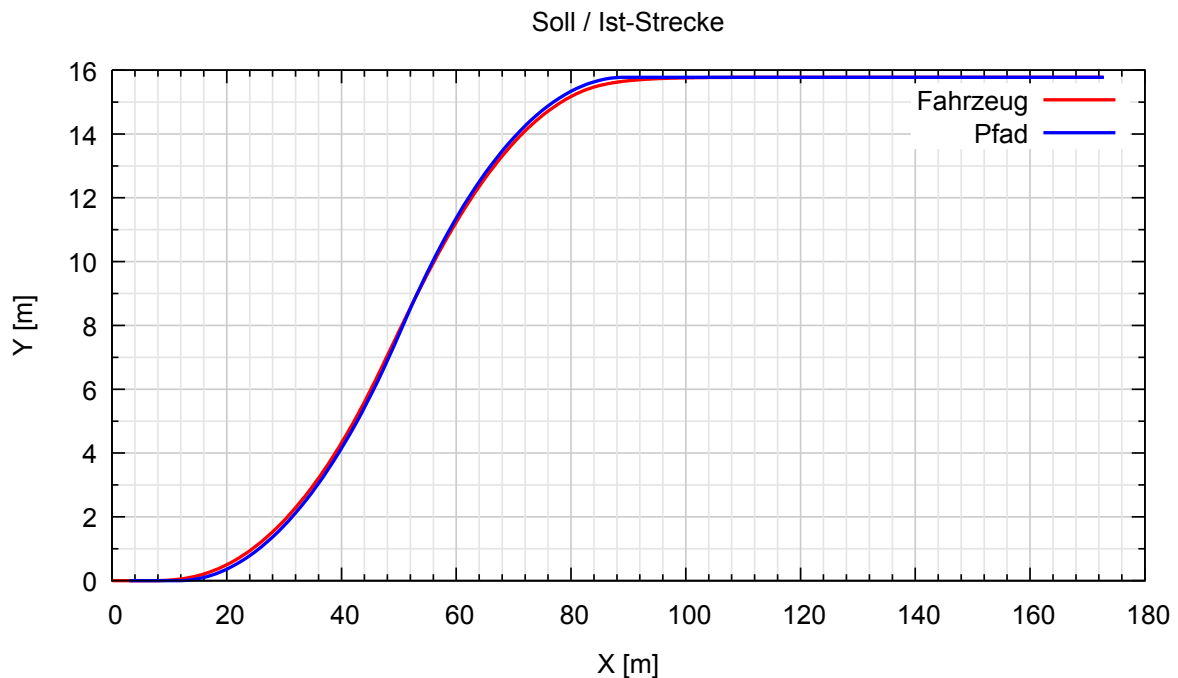


Abbildung 4.43: Soll/Ist-Weg

Da die Bestimmung des Referenzpunktes für die Regelung beim Fuzzy Steering und beim Lookahead Steering identisch definiert ist, ist auch das zuvor erwähnte „Abkürzen“ bei der Betrachtung von Abbildung 4.43 erkennbar. Um auf diesen Effekt Einfluss nehmen zu können, muss die Lookahead-Distanz angepasst werden. Das kann jedoch Auswirkung auf die Stabilität des Reglers haben.

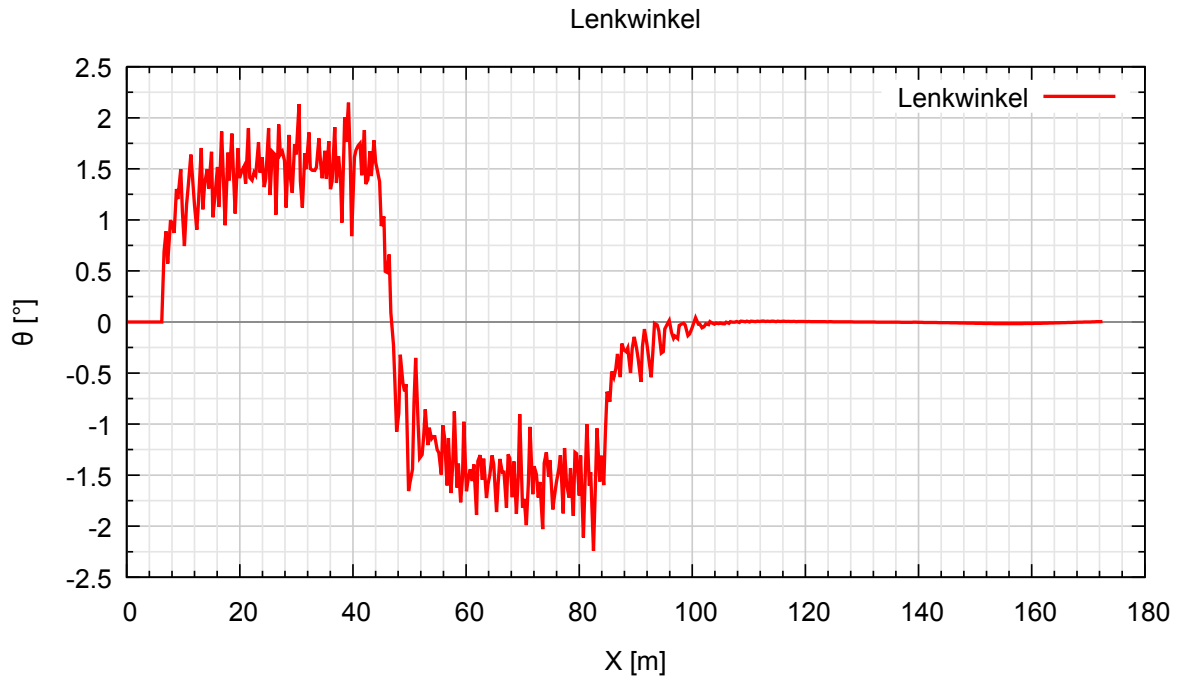
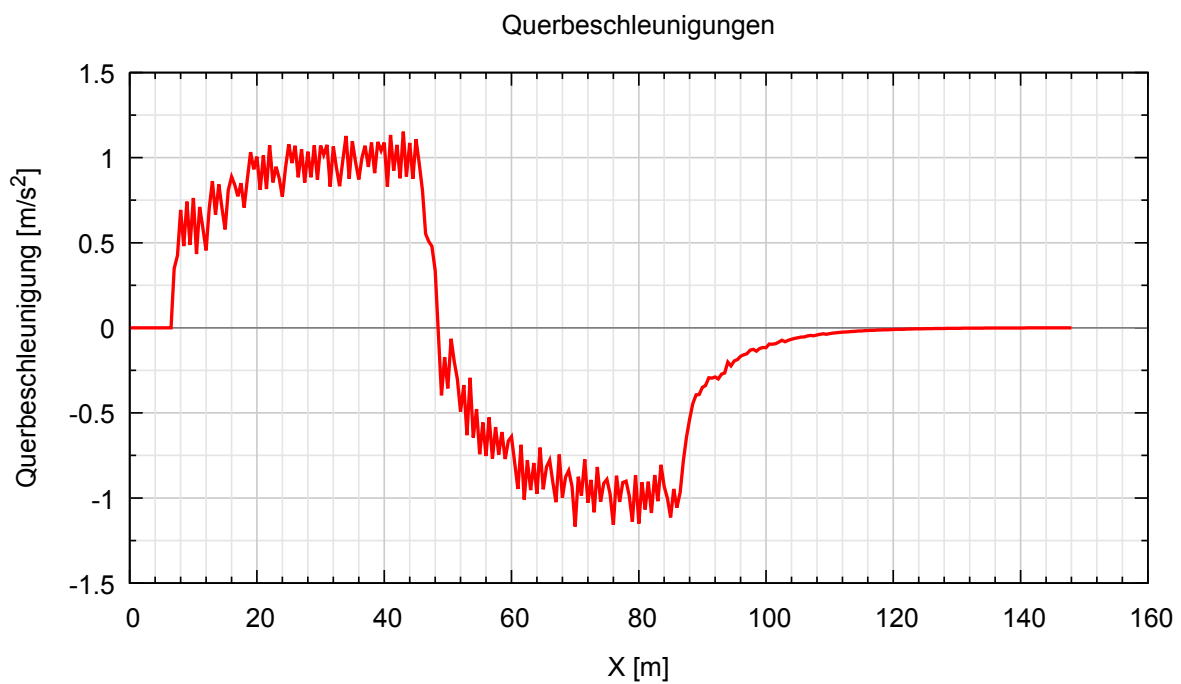


Abbildung 4.44: Lenkwinkel über die Simulationsstrecke

Abbildung 4.45: Querbeschleunigungen des Fahrzeuges bei Spurwechsel und  $v = 10\text{m/s}$

Auf Abbildung 4.45 kann man die Querbewegungen des Spurwechselltests mit  $v = 10\text{m/s}$  erkennen. Es lässt sich daraus ableiten, dass der Fuzzy Regler das Fahrzeug bei dieser Geschwindigkeit gut unter Kontrolle hält. In Abbildung 4.44 lassen sich die minimalen Lenkbewegungen, bzw Stellwertänderungen erkennen.

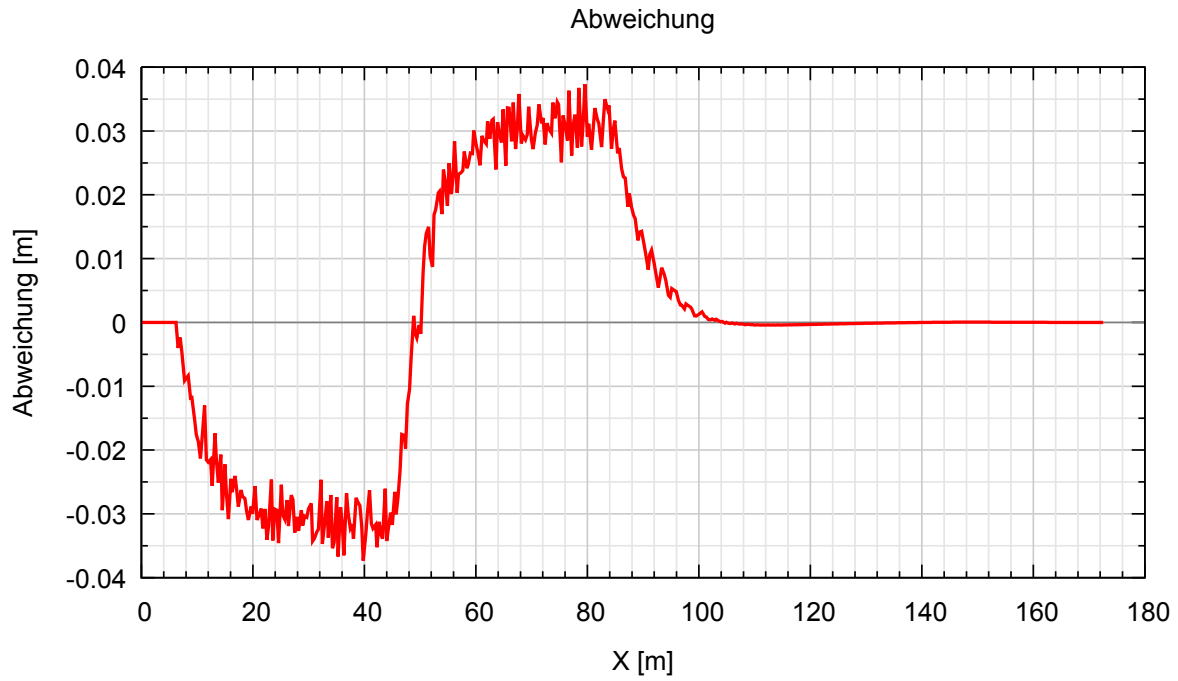


Abbildung 4.46: Regelfehler über die Strecke

**Test02**  $v = 20\text{m/s}$

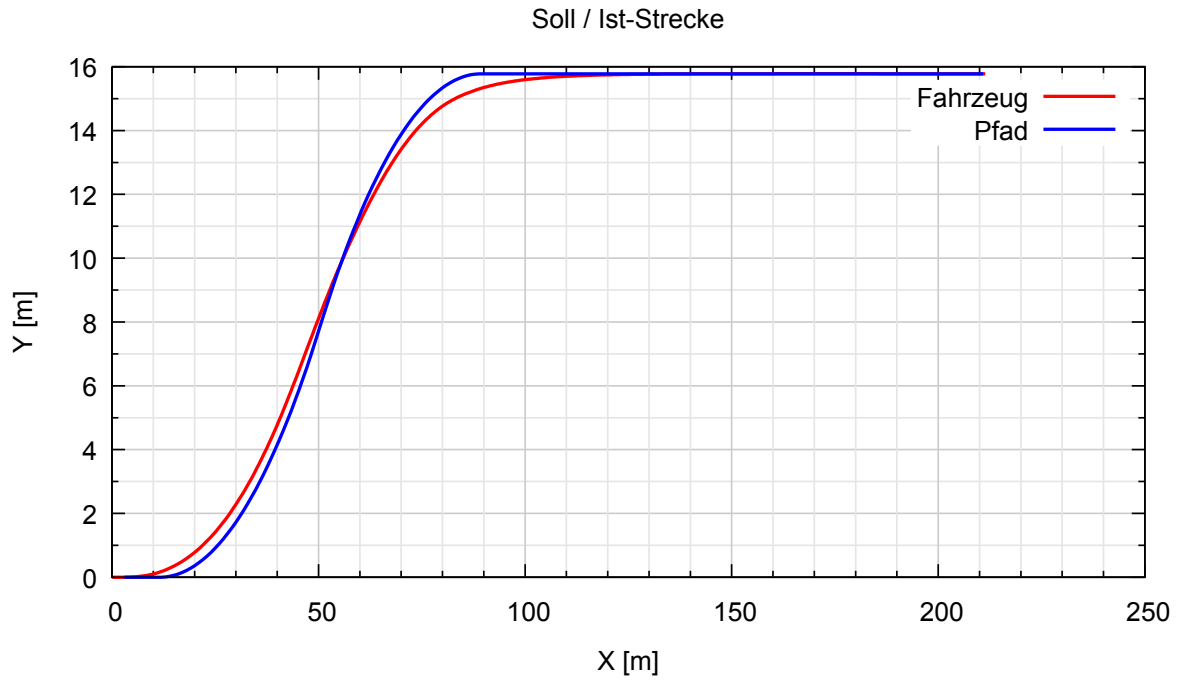


Abbildung 4.47: Soll/Ist-Weg

Der Fuzzy-Regler verhält sich auch bei schnelleren Geschwindigkeiten ähnlich dem Lookahead Regler. Jedoch ist das Nachschwingen deutlich geringer. Das kommt durch die Wahl der Fuzzy-Sets zustande. Da diese Sets bei diesem Regler sehr überlappend in Bezug auf die Einteilung der Abweichungen zum Pfad waren, ist die dämpfende Wirkung höher.

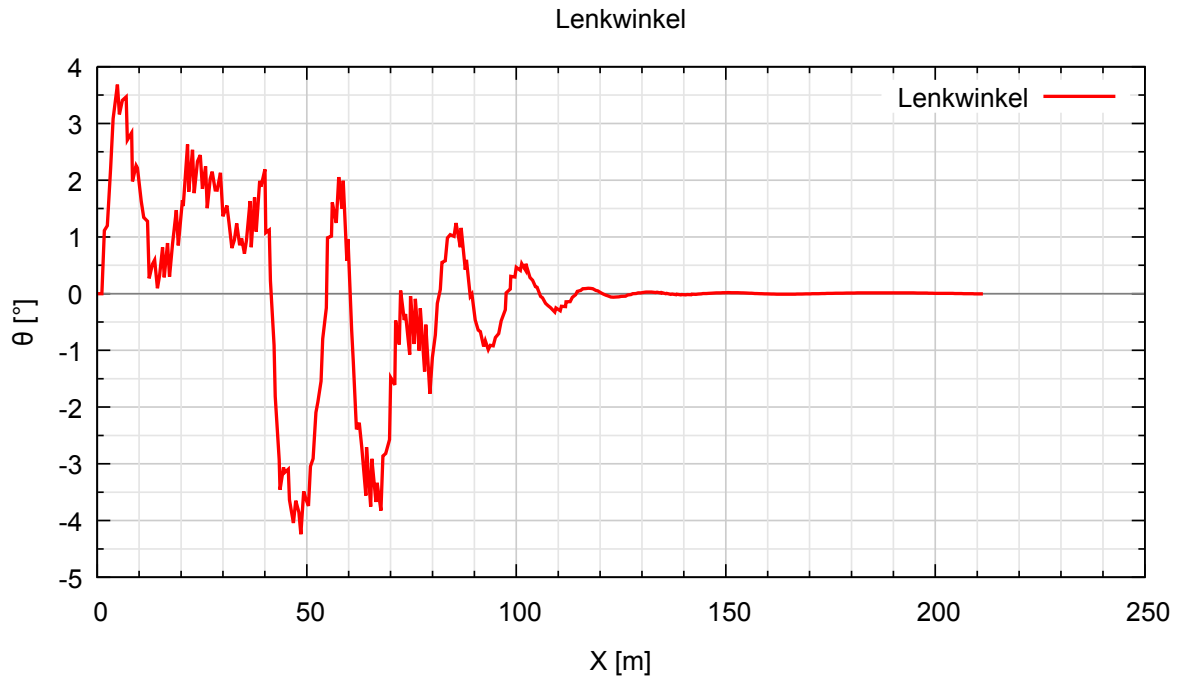
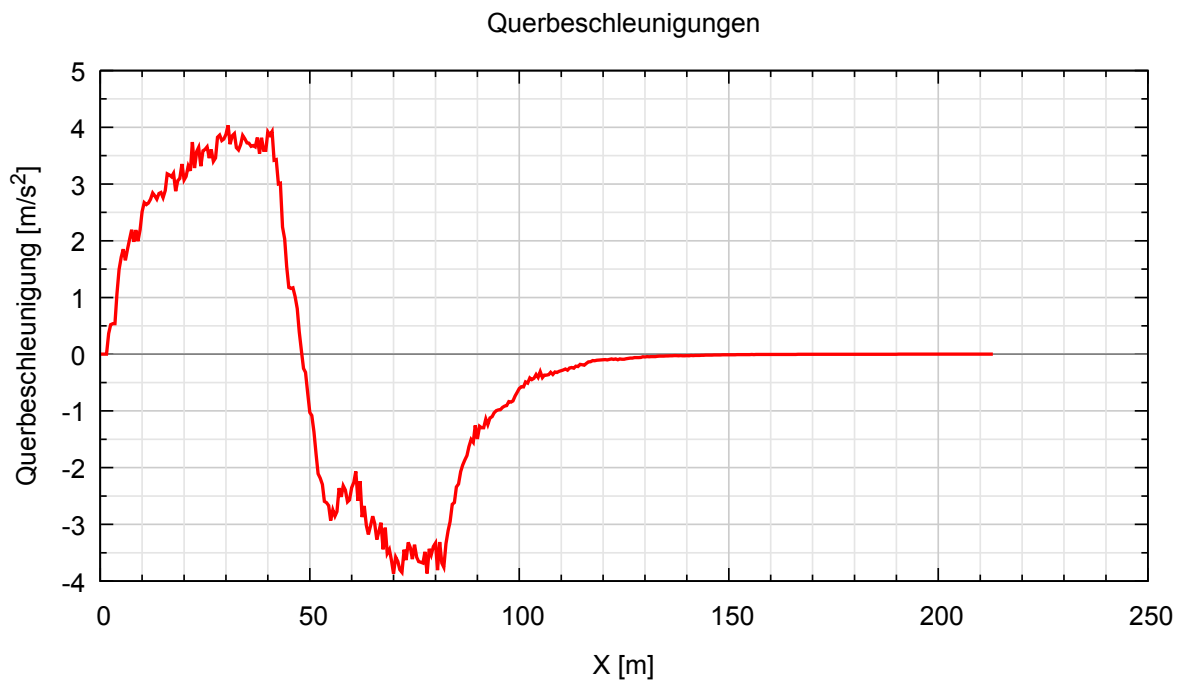


Abbildung 4.48: Lenkwinkel

Abbildung 4.49: Querbeschleunigungen des Fahrzeuges bei Spurwechsel und  $v = 20\text{m/s}$

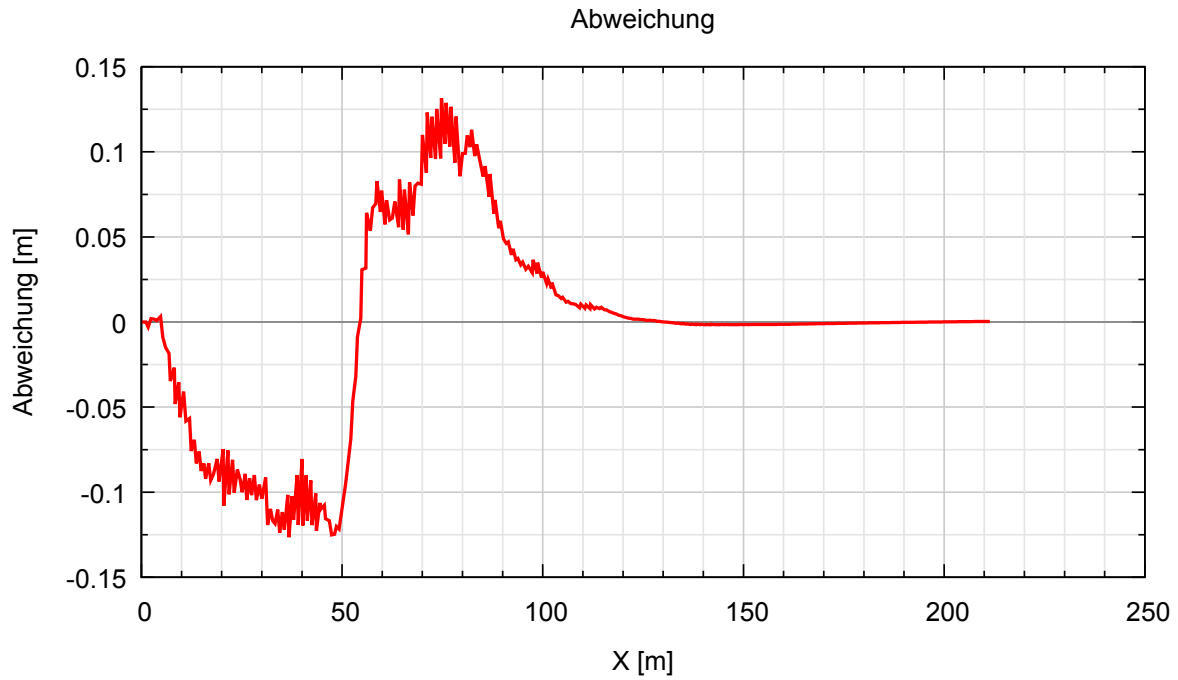


Abbildung 4.50: Regelfehler über die Strecke

**Test03**  $v = 15m/s$

Parameter	Wert
Abtastzeit	$50ms$
$\theta_{min}$	$-0.4rad$
$\theta_{max}$	$0.4rad$
$t_{LA}$	$1s$

Tabelle 4.18: Auflistung der Reglerparameter

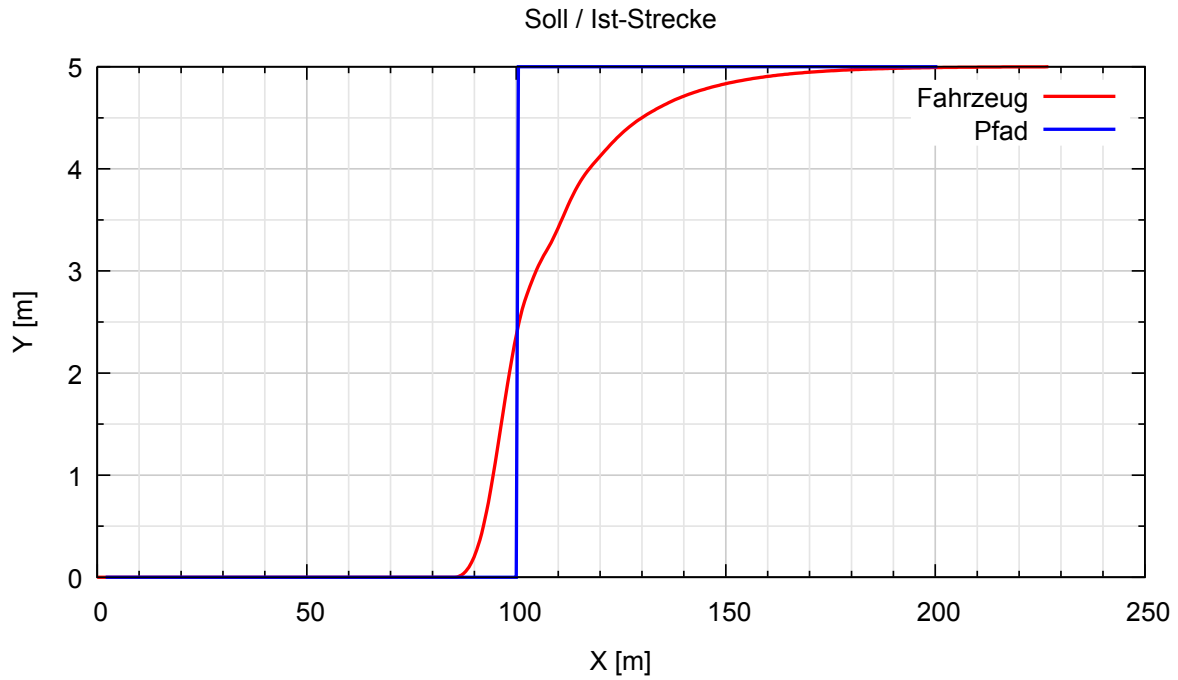


Abbildung 4.51: Soll/Ist-Weg

In Abbildung 4.51 ist zu erkennen, dass das Fahrzeugverhalten dem des Fahrzeuges, das mit dem Lookahead Steering Algorithmus geregelt wurde, sehr ähnlich ist. Auch hier kommt es aufgrund des „vorausschauenden“ Fahrens zu einem früheren Einlenken des Fahrzeuges und somit zu einem geringen Überschwingen. Das Überschwingen ist mit 0.0034% kleiner als beim Lookahead Steering bei gleicher Testfahrt.



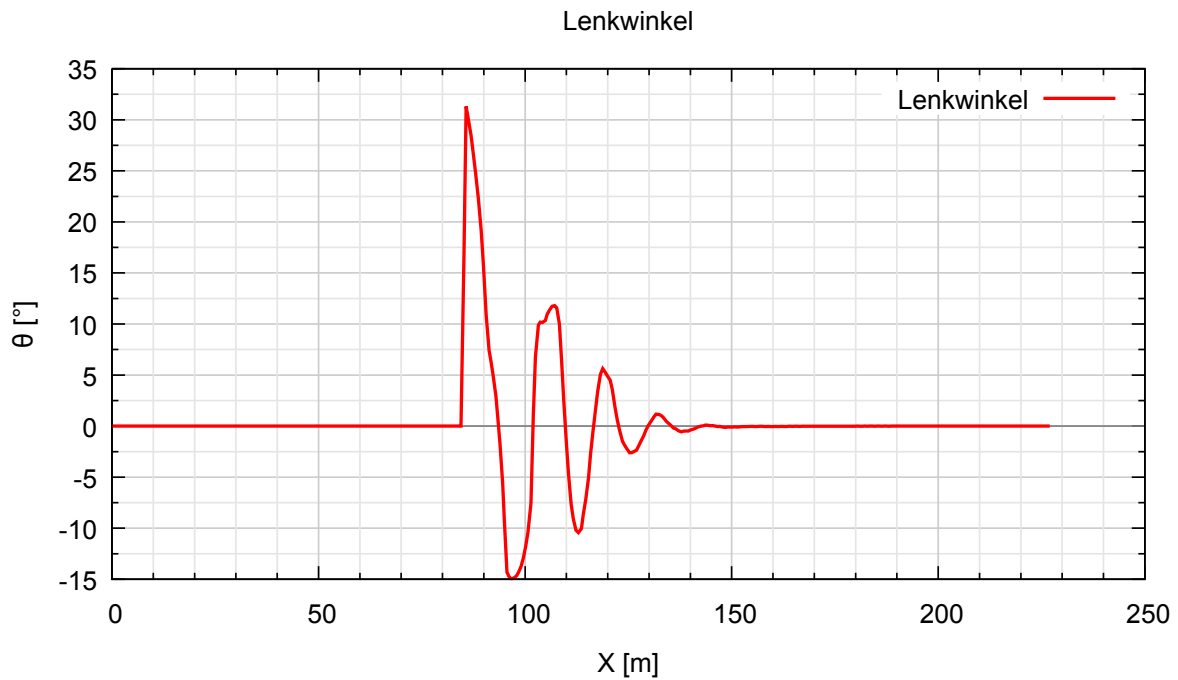
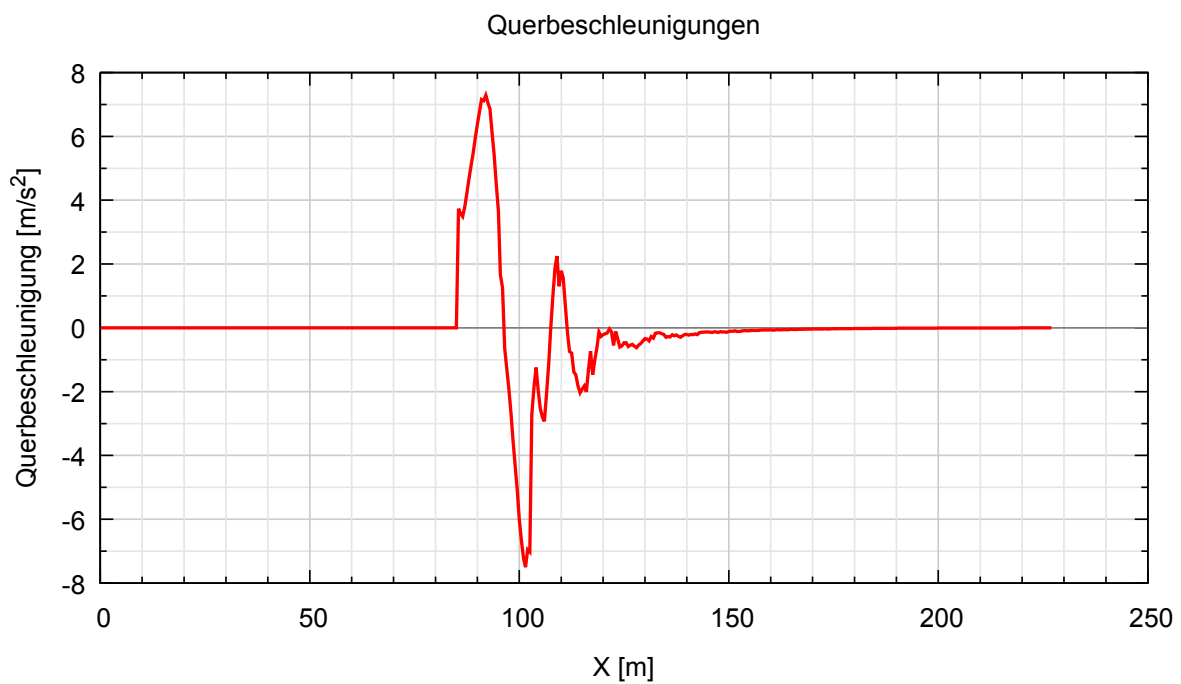


Abbildung 4.52: Lenkwinkel über die Simulationsstrecke

Abbildung 4.53: Querbeschleunigungen des Fahrzeuges bei Spursprung und  $v = 15\text{m/s}$

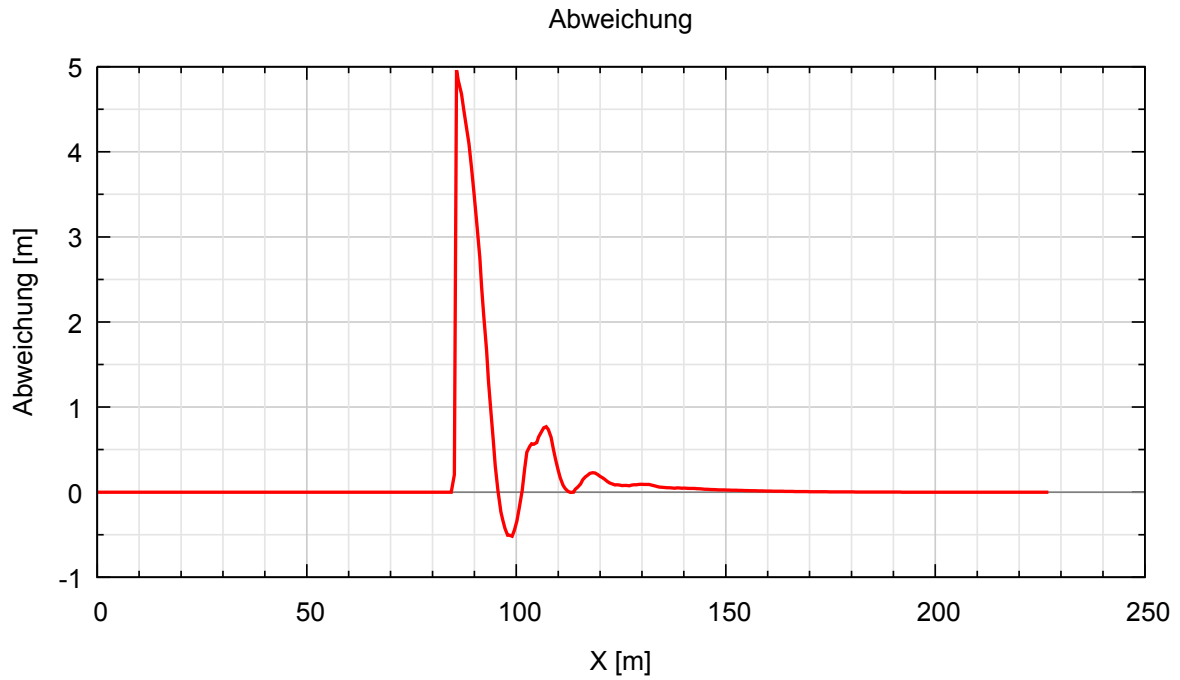


Abbildung 4.54: Regelfehler über die Strecke

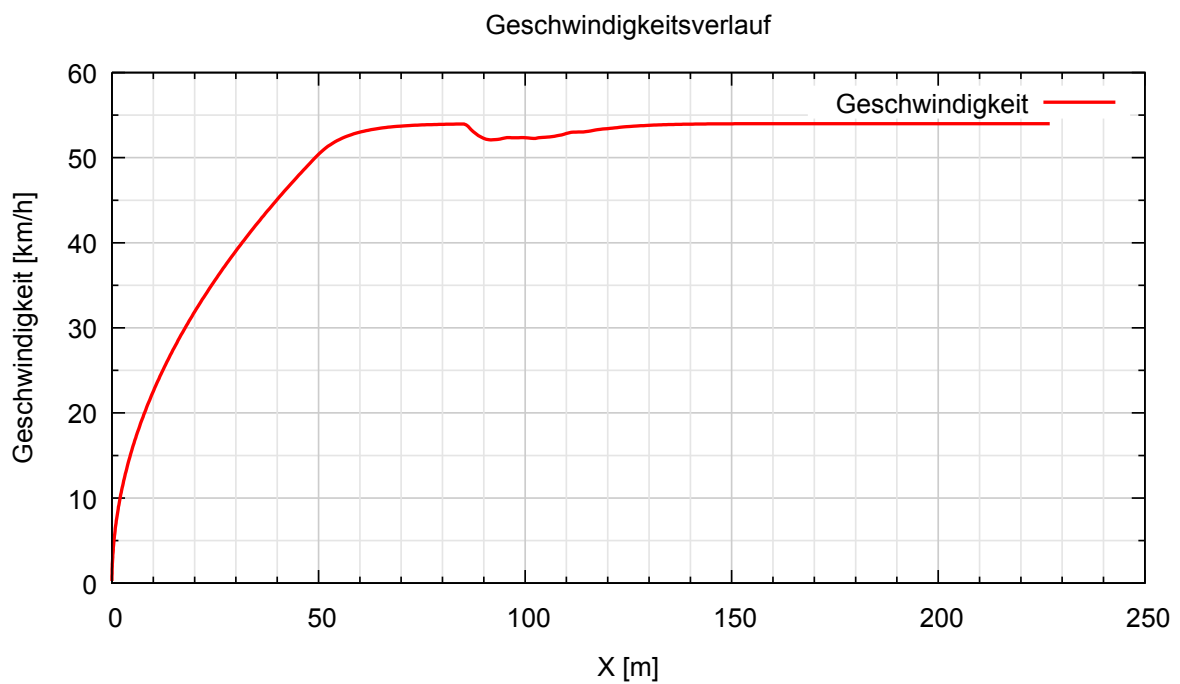


Abbildung 4.55: Geschwindigkeitsverlauf über die Simulationsstrecke

In Abbildung 4.43 ist zu sehen, dass der Fuzzy-Regler, wie andere vorausschauende Regelungsmethoden, kaum zum Überschwingen neigt. Auch bei schnelleren Geschwindigkeiten, siehe Abbildung 4.47, bleibt das Fahrzeug auf dem Pfad.

Die Abweichungen zum Sollpfad betragen durchschnittlich 10 cm in Test01 und Test02 (siehe Abbildungen 4.46 und 4.50).

In Abbildung 4.52 ist ein deutliches Nachschwingen des Lenkwinkels im Bereich von 80 bis 150 Metern Fahrtstrecke zu erkennen.

### Gütekriterien

<b>Kriterium</b>	<b>Wert</b>
Bleibende Regeldifferenz	0.00106m
Anstiegsdistanz $D_r$	43.21m
Ausregeldistanz $D_s$	0m
Überschwingen	0.0034%

Tabelle 4.19: Numerische Werte der Gütekriterien

# 5 Diskussion und Ausblick

## 5.1 Vergleich der Reglertypen anhand der Bewertungskriterien

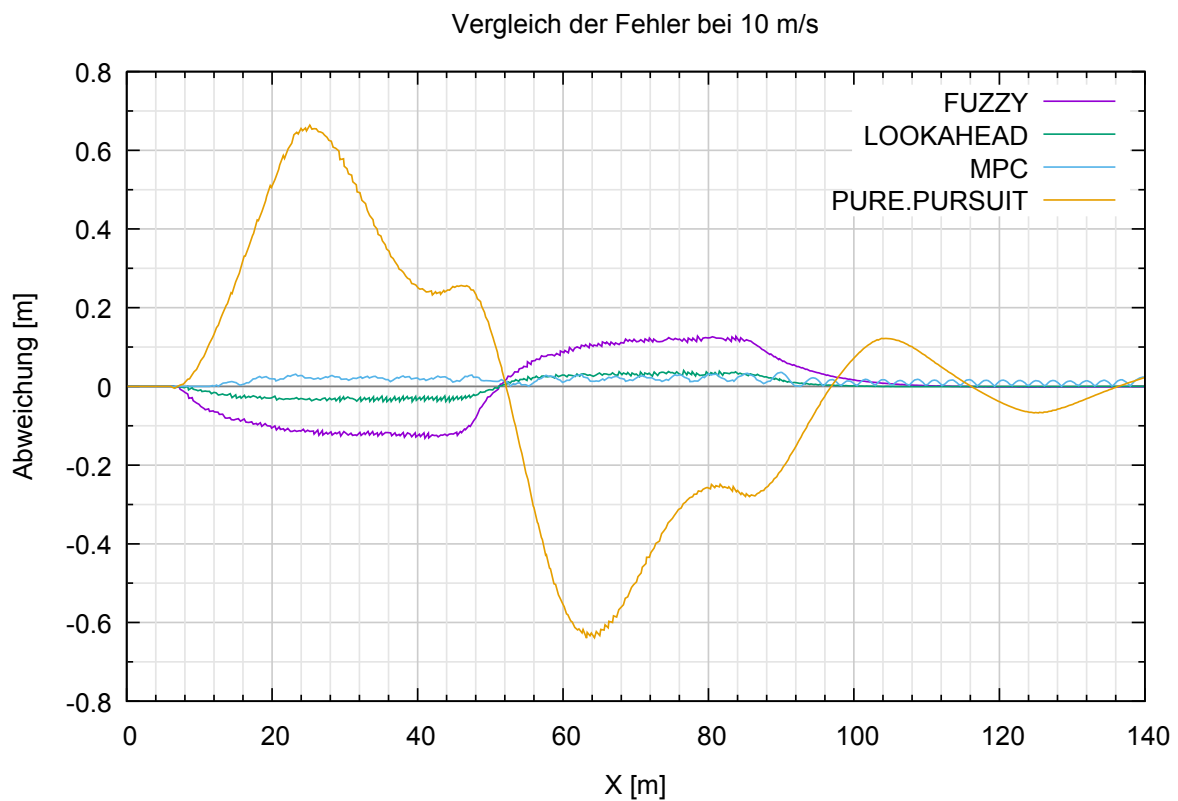


Abbildung 5.1: Vergleich der Regelabweichungen bei Test01

In Abbildung 5.1 sind die Regelabweichungen aller Regelungsalgorithmen dargestellt. Man kann deutlich erkennen, dass der Pure Pursuit-Algorithmus den höchsten Fehler aufweist. Das ist damit zu erklären, dass die mathematische Definition des Algorithmus nur fixe fahrzeuggeschwindigkeitsabhängige Verstärkungen für das P-Glied zulässt.

Würde man hier einen zusätzlichen Dämpfungsfaktor einrechnen, würden sich die Ergebnisse für die langsamen Fahrtgeschwindigkeiten verbessern.

Die Regelabweichung des Lookahead Steering-Reglers ist die geringste und liegt im Durchschnitt unter  $0.02m$ .

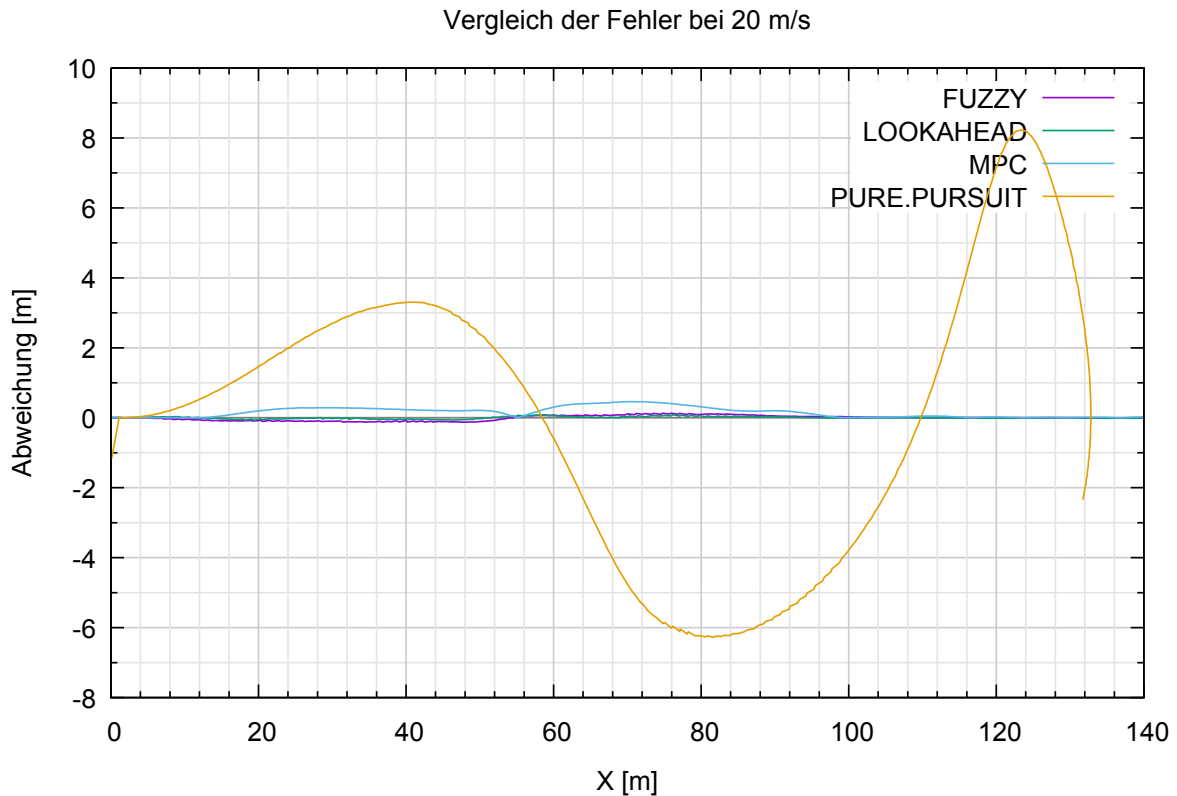


Abbildung 5.2: Vergleich der Regelabweichungen bei Test02

Abbildung 5.2 zeigt die Regelabweichungen aller Algorithmen bei schnellerem Spurwechselversuch mit  $v = 20m/s$ . Es ist sofort erkennbar, dass wiederum die Simulation des Fahrzeuges mit dem Pure Pursuit-Algorithmus eine sehr hohe Abweichung vom Sollwert zeigt. Das resultiert daraus, dass die vom Regler ermittelten Stellgrößen für den Lenkwinkel zu groß für eine stabile Kurvenfahrt waren. Abbildung 4.23 zeigt einen kontinuierlichen Anstieg der Querbeschleunigung des Fahrzeuges auf über  $8m/s^2$ . Diese Querbeschleunigung ist zu hoch für eine stabile Weiterfahrt. Die anderen Regelungstypen bleiben bei diesem Versuch bei unter  $4m/s^2$ .

Da die Abweichung des Pure Pursuit-Algorithmus durch das Schleudern am Ende des Spurwechsels sehr hoch ist, wird in Abbildung 5.3 dieser weggelassen. Dadurch erhöht sich der Detailgrad der anderen Algorithmen.

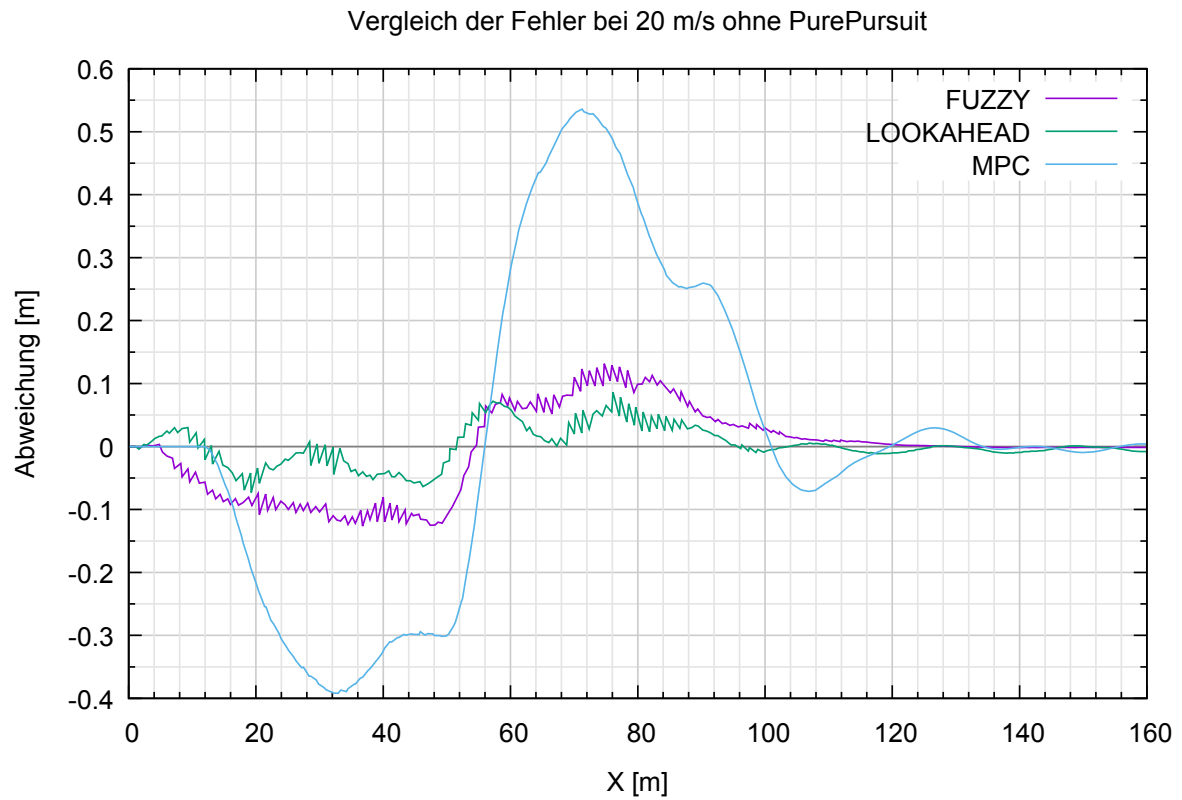


Abbildung 5.3: Vergleich der Regelabweichungen bei Test02 ohne Pure Pursuit Regelalgorithmus

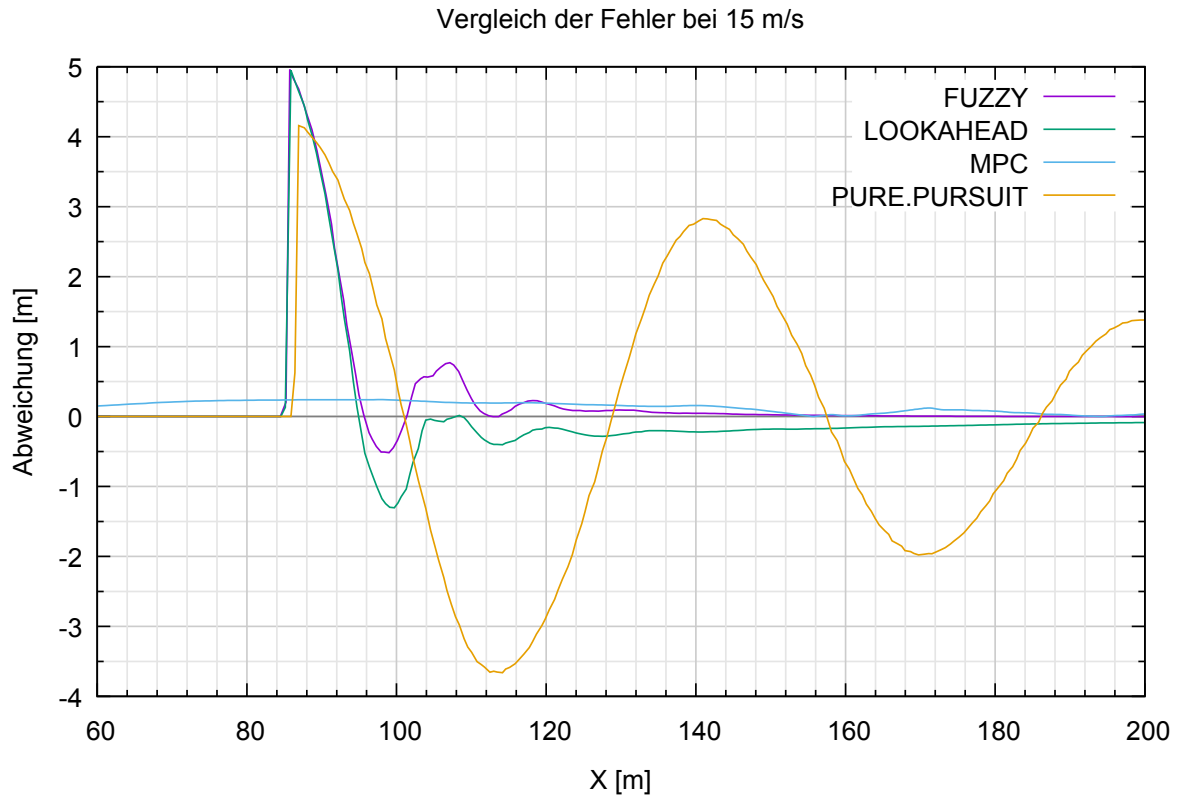


Abbildung 5.4: Vergleich der Regelabweichungen bei Test03

Der Spursprung mit  $v = 15\text{m/s}$  stellt für alle implementierten Regelungstypen eine Herausforderung dar. Der mit 5 m sehr hohe Sprung erzeugt bei Lookahead- und Fuzzy-Regler ein kurzes Überschwingen. Beim Pure Pursuit-Regelungsalgorithmus ist jedoch zu beobachten, dass die Rückkehr in die Ruhelage deutlich länger dauert. Das ergibt sich aus der hohen Verstärkung und dem nicht vorhandenen Dämpfungsglied.

### Mit Rauschen bei Positionsbestimmung

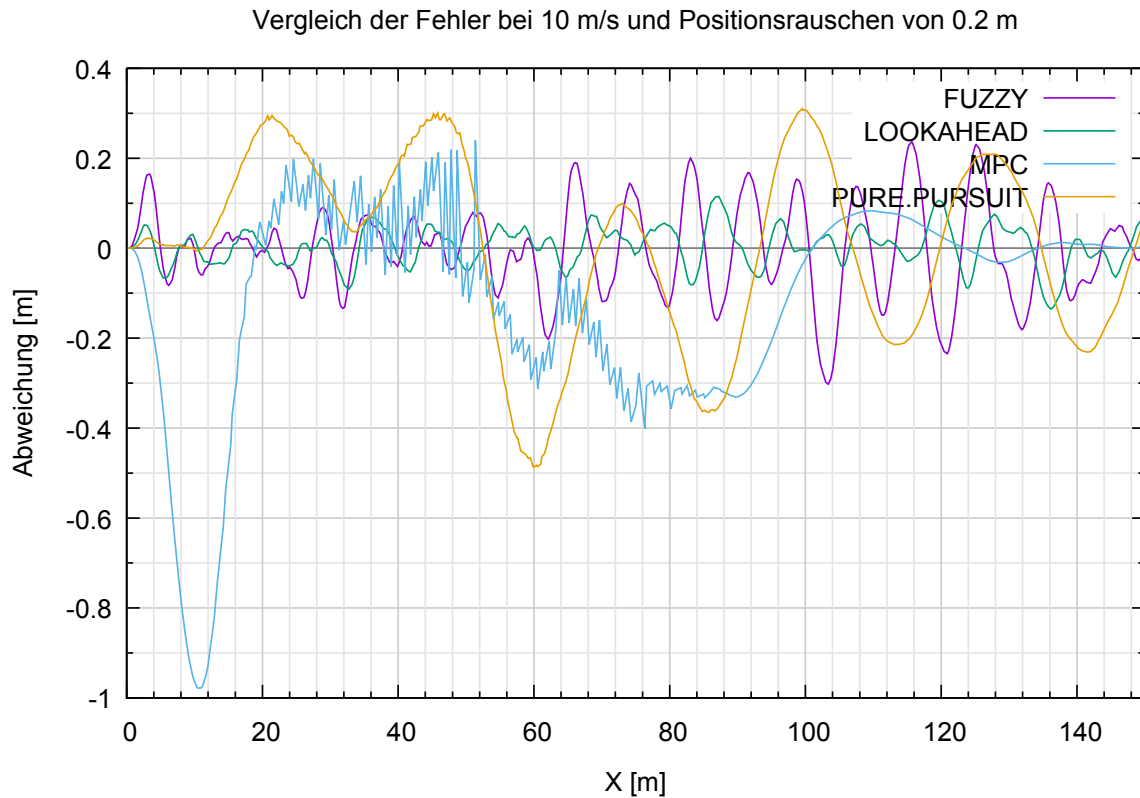


Abbildung 5.5: Vergleich der Regelabweichungen bei Test01 und einem Rauschen der übergebenen Positionen von  $\pm 0.2m$

In Abbildung 5.5 ist die Regelabweichung bei verrauschtem Eingangssignal zu erkennen. Man sieht, dass die Regelabweichungen verglichen mit den Tests, die ein unverraushtes Positionssignal erhalten, höher sind.

Der MPC Algorithmus verhält sich mit den verrauschten Positionsdaten wie erwartet schlechter. Das resultiert daraus, da er versucht auf vorgegebene Sollgrößen zu optimieren. Passiert es jedoch, wie am Anfang des Tests, dass durch das verrauschte Signal die vom Regler angenommene Fahrzeugposition vor der Sollposition liegt, versucht der MPC das Fahrzeug zu wenden. Dadurch kommt die höhere Abweichung beim MPC zustande.

Um einen Überblick über die Eigenschaften der vier umgesetzten Regelungsalgorithmen zu bekommen, sind diese in der folgenden Tabelle noch einmal gegenübergestellt.



Kriterium	MPC	Pure Pursuit	Lookahead Steering	Fuzzy
<b>Güte</b>				
$D_r$	15.55m	13.87m	24.757m	43.21m
$D_s$	16.09m	14.5m	25.2m	24.3m
$\ddot{u}$	8.59%	70.04%	10.747%	10.034%
stationäre Genauigkeit	0.026m	0.45m	0.077m	0.00106m
<b>Rechenzeit</b>	2 – 6.8ms	0.4 – 0.6ms	0.7 – 1ms	1 – 1.2ms
<b>Positionsrauschverhalten</b>	sehr schlecht	gut	schlecht	mittel
<b>Komplexität</b>	sehr hoch	hoch	mittel	einfach

Tabelle 5.1: Vergleich der Regelungssysteme aus den vorherigen Kapiteln (MPC, Pure Pursuit, Lookahead Steering und Fuzzy Regelung)

Betrachtet man die vorherige Tabelle, kommt man zu dem Ergebnis, dass sich drei Regelsysteme hervorheben. MPC, Lookahead Steering und die Fuzzy-Regelung. Die MPC überzeugt durch eine schnelle Stelldistanz. Hat jedoch mit 8.6% ein hohes Überschwingen. Die stationäre Genauigkeit ist bei allen Reglern sehr hoch.

Zusammenfassend betrachtet zeigt sich, dass sich der Lookahead Steering-Algorithmus trotz schlechtem Rauschverhalten sehr gut an alle Szenarien anpassen lässt. Auch ist die Adaptierung und Implementierung mit nur sehr wenig Aufwand verbunden.

## 5.2 Zusammenfassung

Der Entwurf von Systemen zur Weiterentwicklung von Aktiven Sicherheitssystemen bringt häufig Probleme mit sich. Zum einen muss die Genauigkeit und Zuverlässigkeit gewährleistet werden, zum anderen muss verhindert werden, dass ein Steuerungsalgorithmus ein gewisses ökonomisch vertretbares Maß in der Entwicklung und Implementierung überschreitet.

Deshalb ist es auch notwendig, die verschiedenen Regelungsalgorithmen auf ihre Verwendbarkeit in autonom fahrenden Fahrzeugen zu überprüfen.

Im Rahmen dieser Arbeit wurde nun versucht, diese Möglichkeiten der Regelung in autonom lenkenden Fahrzeugen zu untersuchen und vergleichbar zu machen. Dabei ist deutlich geworden, dass eine Vielzahl von Algorithmen existieren, die sich zur Lenkung autonomer Fahrzeuge eignen.

In dieser Arbeit wurden 4 Regelungskonzepte ausgewählt, implementiert und in verschiedenen Szenarien simuliert.

Bei der nähergehenden Untersuchung dieser Algorithmen ist aufgefallen, dass sie sich vor allem in Bezug auf die Komplexität stark unterscheiden. Ist bei einem gewöhnlichen PID-Regler das Verständnis und die Implementierung schnell durchgeführt, ist jedoch bei einem Modellbasierten Prädiktiven Regelungssystem mehr Aufwand in der Recherche und Umsetzung notwendig. Höherer Zeitaufwand bedeutet höhere Kosten.

Auch ist die Komplexität des Modells in der MPC sehr kritisch zu betrachten. Es ist theoretisch wie praktisch gesehen ein Kompromiss zwischen der Größe des Detailgrades des Modells und der Schnelligkeit in der Berechnung der Stellgröße zu finden. Mit dem in dieser Arbeit verwendeten kinematischen Modell ist ein stabiles Verhalten bei langsamen Fahrten und wenig Positionsrauschen zu erwarten. Werden die Querbeschleunigungen jedoch zu groß und weichen zu stark vom kinematischen Modellverhalten ab, ist das Ergebnis erheblich schlechter als bei gewöhnlichen stetigen Regelungsalgorithmen, wie

dem PID-Regler.

Allgemein ist zu sagen, dass der Ansatz des Lookahead Steerings eine hohe Genauigkeit und Zuverlässigkeit bietet. Deshalb ist es wohl auch eine “sichere” Möglichkeit diesen Algorithmus auf reale Fahrzeuge zu übertragen.

## 5.3 Ausblick

Weitere Verwendungen mit einigen in dieser Arbeit entwickelten Erkenntnissen und Softwarelösungen sind auch in Zukunft geplant.

Die evaluierten Regelungsalgorithmen sollen in einem Lenkroboter integriert werden, der momentan von der Firma DSD entwickelt wird.

Das als extrahierte Komponente entwickelte PC-Crash Realtime-Simulation-CAN-BUS-Interface eignet sich ebenfalls dafür, die sehr gut evaluierte Simulationssoftware PC-Crash in Hardware in the Loop Systeme integrierbar zu machen.

# Literatur

- [1] Karel A. Brookhuis, Dick de Waard und Wiel H. Janssen.  
„Behavioural impacts of Advanced Driver Assistance Systems—an overview“.  
In: *EJTIR* 1, no. 3 (Nov. 2001), S. 245–253.
- [2] Volkswagen. *Passive Sicherheit*.  
URL: [http://www.volkswagen.de/de/technologie/techniklexikon/passive\\_sicherheit.html](http://www.volkswagen.de/de/technologie/techniklexikon/passive_sicherheit.html) (besucht am 05.12.2015).
- [3] Volkswagen. *Aktive Sicherheit*.  
URL: [http://www.volkswagen.de/de/technologie/techniklexikon/aktive\\_sicherheit.html](http://www.volkswagen.de/de/technologie/techniklexikon/aktive_sicherheit.html) (besucht am 01.10.2015).
- [4] *Analyse des Fahrerverhaltens vor dem Unfall eine Methode für eine verbesserte Fehleranalyse bei der Untersuchung realer Verkehrsunfälle*. Mai 2006.
- [5] Nicola Fricke Charlotte Glaser und Monica de Phillipis.  
„Passive und Aktive Sicherheitsmaßnahmen im Kraftfahrzeug“. In: (2006).
- [6] Hermann Steffan. *Implementierung eines Systems zur autonomenen Trajektorienfahrt von Fahrzeugen anhand von DGPS Lokalisierung*. Techn. Ber. TU Graz, 2014.
- [7] Dr. Steffan Datentechnik. *Operating manual UFO*. Linz, 2014.
- [8] STM32-E407 development board USERS MANUAL. *Olimex Ltd.* BULGARIA, 2015.
- [9] R. E. Kalman. „A New Approach to Linear Filtering And Prediction Problems“. In: *ASME Journal of Basic Engineering* (1960).
- [10] Oxford Technical Solutions Limited. *RT Inertial and GPS Measurement Systems*. Upper Heyford Oxfordshire, 2011.
- [11] Moser A und Steffan H an Steffan G. *PC-CRASH Ein Simulationsprogramm für Verkehrsunfälle Bedienungs- und technisches Handbuch*. Linz, 2015.

- [12] Dr. Steffan Datentechnik GmbH. *PC-Crash*.  
URL: [http://dsd.at/index.php?option=com\\_content&view=article&id=323:pc-crash-deutsch&catid=53&Itemid=175&lang=de](http://dsd.at/index.php?option=com_content&view=article&id=323:pc-crash-deutsch&catid=53&Itemid=175&lang=de) (besucht am 11.11.2015).
- [13] W. Cliff und D. Montgomery. „Validation of PC-Crash - A Momentum-Based Accident Reconstruction Program“. In: *SAE Technical Paper 960885* (1996).  
DOI: 10.4271/960885.
- [14] Peak Systems. *PCAN-USB*.  
URL: <http://www.peak-system.com/PCAN-USB.199.0.html> (besucht am 20.10.2015).
- [15] H. Wallentowitz und K. Reif. *Handbuch Kraftfahrzeugelektronik: Grundlagen - Komponenten - Systeme - Anwendungen*. ATZ/MTZ-Fachbuch. Vieweg+Teubner Verlag, 2010. ISBN: 9783834807007.  
URL: <https://books.google.at/books?id=yMjcMIdGceoC>.
- [16] Wikipedia. *Controller Area Network*.  
URL: [https://de.wikipedia.org/wiki/Controller\\_Area\\_Network](https://de.wikipedia.org/wiki/Controller_Area_Network) (besucht am 09.11.2015).
- [17] Projekt daedalus TU München. *Funktionsweise des CAN-Bus*.  
URL: <http://www.daedalus.ei.tum.de/index.php/de/gemischtes/kommstruktur/funktionsweise-des-can-bus> (besucht am 21.10.2015).
- [18] Jürgen Adamy. *Nichtlineare Regelungen*. Springer, 2009.  
URL: <http://tubiblio.ulb.tu-darmstadt.de/43820/>.
- [19] Dittmar und Rainer.  
*Modellbasierte prädiktive Regelung : eine Einführung für Ingenieure*.  
München: Oldenbourg, 2004. ISBN: 3486275232.
- [20] Jacob Mattingley und Stephen Boyd.  
„CVXGEN: a code generator for embedded convex optimization“. In: (2011).
- [21] CVXGEN. *Example: Model Predictive Control (MPC)*.  
URL: <http://cvxgen.com/docs/mpc.html> (besucht am 28.10.2015).
- [22] Manfred Mitschke. „Das Einspurmodell von Riekert-Schunck“. German.  
In: *ATZ - Automobiltechnische Zeitschrift* 107.11 (2005), S. 1030–1031.  
ISSN: 0001-2785. DOI: 10.1007/BF03223515.  
URL: <http://dx.doi.org/10.1007/BF03223515>.

- [23] Manfred Mitschke und Henning Wallentowitz.  
*Dynamik der Kraftfahrzeuge*. German. 5;5., überarb. u. erg. Aufl. 2014;  
Wiesbaden: Springer Vieweg, 2014. ISBN: 3658050675;9783658050672;
- [24] Karl-Ludwig Haken. *Grundlagen der Kraftfahrzeugtechnik*.  
Hanser Fachbuchverlag, 2011. ISBN: 3446426043.
- [25] Ida Petersson und Johanna Risö.  
„Automotive Path Following using Model Predictive Control“.  
Magisterarb. Sweden: CHALMERS UNIVERSITY OF TECHNOLOGY, 2014.
- [26] Felipe Kühne und Joao Manoel Gomes da Silva Jr.  
*Model Predictive Control of a Mobile Robot Using Linearization*. Techn. Ber.  
Federal University of Rio Grande do Sul, 2004.
- [27] J. Giesbrecht, D. Mackay, J. Collier und S. Verret.  
„Path Tracking for Unmanned Ground Vehicle Navigation Implementation and  
Adaptation of the Pure Pursuit Algorithm“. In: *Technical Memorandum* (2005).
- [28] L.A. Zadeh. „Fuzzy Sets“. In: *Information Control* 8 (1965), S. 338–353.
- [29] Heinz Unbehauen. *Klassische Verfahren zur Analyse und Synthese linearer  
kontinuierlicher Regelsysteme, Fuzzy-Regelsysteme ; mit 25 Tabellen*.  
Braunschweig u.a.: Vieweg, 2005. ISBN: 3528213329.
- [30] Universitaet Muenster. *Einführung in Fuzzy Systeme Der Fuzzy-Controller*.  
URL: [http://cs.uni-muenster.de/Professoren/Lippe/lehre/skripte/  
wwwFuzzyScript/fscon.html](http://cs.uni-muenster.de/Professoren/Lippe/lehre/skripte/wwwFuzzyScript/fscon.html) (besucht am 09.11.2015).
- [31] Juan Rada-Vilela. *fuzzylite: a fuzzy logic control library*. 2014.  
URL: <http://www.fuzzylite.com>.
- [32] S.Z. Manfred Reuter und S. Zacher. *Regelungstechnik für Ingenieure: Analyse,  
Simulation und Entwurf von Regelkreisen*. Viewegs Fachbücher der Technik.  
Vieweg+Teubner Verlag, 2004. ISBN: 9783528050047.  
URL: <https://books.google.at/books?id=YTU0QCvCcecC>.