Patrick Weißensteiner, BSc

Fahrdynamische Auslegung eines allradgetriebenen Elektrofahrzeugs mit omnidirektionaler Lenkung

Masterarbeit

zur Erlangung des akademischen Grades

Dipl.-Ing.

Masterstudium: Wirtschaftsingenieurwesen-Maschinenbau

eingereicht an der

Technische Universität Graz

in Zusammenarbeit mit dem

Kompetenzzentrum - Das virtuelle Fahrzeug Forschungsgesellschaft mbH

Betreuer

Dipl.-Ing. (FH) Andreas Kerschbaumer

Institut für Regelungs- und Automatisierungstechnik
Institutsleiter: Univ.-Prof. Dipl-Ing. Dr.techn. Martin Horn

Graz, Oktober 2017

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Das in TUGRAZonline hochgeladene Textdokument ist mit der vorliegenden Masterarbeit identisch.

Datum

Unterschrift

Vorwort

Die vorliegende Arbeit entstand während meiner Tätigkeit als Masterand am Kompetenzzentrum - Das virtuelle Fahrzeug (ViF) im Zeitraum von Februar bis August 2017.

Großer Dank gilt hier vor allem Andreas Kerschbaumer, welcher mir die Chance gab meine Masterarbeit über ein Themengebiet zu verfassen, welches von größtem Interesse für mich ist. Auch für den reibungslosen Ablauf während der Durchführung möchte ich mich herzlich bedanken. Ebenso großer Dank gebührt Martin Rudigier, welcher die Arbeit seitens des ViF mitbetreute und stets mit fachlichem Rat zur Seite stand. Weiters möchte ich mich auch bei all jenen Personen am ViF bedanken, die durch ihre Unterstützung zum Gelingen dieser Masterarbeit beigetragen haben.

Ein besonderer Dank gilt auch Herrn Prof. Dr. techn. Martin Horn. Zum einen für die Übernahme der Betreuung dieser Arbeit und zum anderen für die kompetente Beantwortung der aufgetretenen Fragen während der Durchführung.

Weiters danke ich meinen Eltern, die mir während meines Studiums jederzeit unterstützend zur Seite standen. Zu guter Letzt danke ich dir, Bianca, für die moralische Unterstützung während der Durchführung und vor allem während des Verfassens dieser Arbeit.

Patrick Weißensteiner

Graz, 11. September 2017

Abstract

Mobile robots are increasingly sharing their work envelope with humans. Compared to robots with locomotor systems, which are based on animals, the movement with standard wheels has an advantage and is currently the most used chassis type. Robots with wheel-based chassis can be differentiated through the amount of available degrees of freedom, relating to the instantaneous center of rotation. In the course of the work the suspension kinematics of a mobile robot with four independent steerable and drivable wheels is looked upon.
The task of the robot is to follow a given trajectory and in addition reach a given orientation of the robots body in relation to it. Based on given points, which also include a desired robot velocity and orientation, a set of auxiliary points is automatically generated. With these additional points a continuous trajectory is reached.
For the robot to be able to follow a given path, a velocity vector as connection between the robots center of gravity and a point of the path, ahead of the robot, is defined. The angle of that vector is proportional controlled as a function of the deviation between the robot and the path. To determine the deviation, the current position of the robot is measured and geometric considerations are applied. The orientation of the robot is regulated using a first-order sliding-mode controller, as actuating value the angular velocity of the robots body is used. The knowledge of the needed angular velocity of the robots body, as well as the velocity vector of the robot, allows the use of inverse kinematics to determine the steering angles and the wheel speeds. For the regulation of the steering angles a sliding-mode controller with super-twisting algorithm is used, the wheel speeds are regulated using a PI-controller with anti-windup. For both cases a luenberger-observer, using a simple model of a non-deformable wheel, was implemented to determine the relevant restoring torque. The measured variables, which are needed for the implemented controllers, are the current steering angles and wheel speeds of all wheels. The actuating values are the steering- and drive torques for the wheels.
The controller design was made using a Co-simulation between MATLAB/Simulink and ADAMS/Car. The used vehicle model in ADAMS/Car was developed with a series of driving maneuvers. From this preliminary investigation arose that no additional springs or dampers, with the exception of the tires, are needed. This was gained from the fact that driving on a rough road with the robots maximum velocity of 50 $\frac{km}{h}$ did not lead to a total loss of vertical tire forces.
In the given concept, one must already consider during planning of the desired robot trajectory, that the instantaneous center of rotation is far enough away from any wheel center. If that is not the case, high steering velocities are needed and if they cannot be achieved, a safe robot motion is not guaranteed. In addition, too high lateral accelerations have to be avoided as well, with the upper bound found during preliminary investigations. Under these conditions it can be seen that the robot is able to follow a given trajectory in satisfying accuracy.

Kurzfassung

Mobile Roboter teilen sich ihre Arbeitsräume zunehmend mit Menschen. Gegenüber Roboter mit Bewegungsapparaten, welche an die Tierwelt angelehnt sind, ist die Fortbewegung mittels Standardrädern derzeit noch im Vorteil und der meist verwendete Fahrwerkstyp. Roboter mit radbasierten Fahrwerken unterscheiden sich durch die Anzahl an verfügbaren Freiheitsgraden, bezogen auf den Momentanpol der Bewegung. Im Verlauf dieser Arbeit wurde die Fahrwerkskinematik eines Roboters mit vier gelenkten und angetriebenen Standardrädern betrachtet.
Die Aufgabe des Roboters besteht darin, einer vorgegebenen Trajektorie zu folgen und einen vorgegebenen Winkel seines Aufbaus bezüglich dieser zu erreichen. Es werden Punkte vorgegeben, welche die Geschwindigkeit, als auch den Winkel des Roboteraufbaus zur Trajektorie festlegen. Mittels automatisch generierter Zwischenpunkte werden diese zu einer durchgängigen Trajektorie verbunden.
Damit der Roboter dem Pfad folgen kann, wird ein Geschwindigkeitsvektor als Verbindung zwischen dem Robotermittelpunkt und einem Punkt entlang des Pfades definiert. Die Richtung des Vektors wird dabei proportional der Abweichung des Roboters vom Pfad geregelt, als Messgröße wird die momentane Roboterposition verwendet. Der Aufbauwinkel des Roboters wird mittels Sliding-Mode Regelung bestimmt, als Stellgröße dient die Winkelgeschwindigkeit des Aufbaus. Mittels inverser Kinematik können aus dieser Winkelgeschwindigkeit, sowie dem Geschwindigkeitsvektor, die einzelnen Lenkwinkel und Raddrehzahlen bestimmt werden. Die Lenkwinkel werden durch einen Sliding-Mode-Regler mit Super-Twisting Algorithmus, die Raddrehzahlen mittels PI-Regelung mit Anti-Windup bestimmt, zusätzlich wurde für beide Fälle ein Luenberger-Beobachter für das jeweilige Rückstellmoment entworfen. Die Messgrößen sind die jeweiligen Lenkwinkel und Antriebsdrehzahlen der einzelnen Räder, als Stellgrößen dienen die Lenk- und Antriebsmomente.
Für die Auslegung der verwendeten Regler wurde eine Co-Simulation zwischen MATLAB/Simulink und ADAMS/Car aufgebaut. Das in ADAMS/Car verwendete Fahrzeugmodell wurde durch eine Voruntersuchung mittels fahrdynamischer Manöver erhalten. Die Untersuchung ergab, dass keine zusätzliche Federung, sowie Dämpfung, abgesehen von den Reifen, notwendig ist, da kein Abheben von einer unebener Straße beim Befahren mit der Roboter-Höchstgeschwindigkeit von 50 $\frac{km}{h}$ auftritt.
Bei dem vorliegenden Konzept muss bereits im Zuge der Planung der Trajektorie darauf geachtet werden, dass der Momentanpol immer einen ausreichend großen Abstand zu den Radmittelpunkten besitzt. Ist dies nicht der Fall, treten womöglich nicht bewältigbare Lenkwinkelgeschwindigkeiten auf und ein sicheres Bewegen des Fahrzeuges ist nicht möglich. Ebenso müssen zu hohe Querbeschleunigungen vermieden werden. Unter diesen Voraussetzungen kann gezeigt werden, dass es dem Roboter möglich ist vorgegebe Pfade mit zufriedenstellender Genauigkeit zu absolvieren.
Auf ein energieoptimiertes Befahren der Trajektorie wurde verzichtet, da dies nur mittels eines dynamischen Modells möglich ist.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

PKW	Personenkraftwagen
FTS	führerloses Transportsystem
MP	Momentanpol
MKS	Mehrkörpersimulation
VDI	Verein Deutscher Ingenieure
ROS	Robot Operating System

1. Einführung

1.1. Motivation und Ziel der Arbeit

Wir leben in einer Zeit der kontinuierlich voranschreitenden industriellen Automatisierung. Im Zuge dessen gewannen auch Roboter in den letzten Jahrzehnten immer mehr an Bedeutung in der industriellen Produktion. Hier werden sie meist stationär in Zellen betrieben und treten kaum mit Menschen in Kontakt. Die Entwicklung zukünftiger Robotersysteme geht jedoch in eine Richtung, in welche die Systeme immer stärker mit Menschen kooperieren und auch in die Lebenräume dieser eingreifen (Connette, 2013). Die Roboter verlassen also zunehmend ihre Montagezellen und werden Teil des täglichen Lebens (Schraft et al., 2004). Eine Bestätigung liefert die Studie World Robotics 2009 (Hägele, 2009), welche den Servicerobotern, entgegen dem Trend bei Industrierobotern, ein weiteres Wachstum prognostiziert.

Wenn sich also in Zukunft Roboter verstärkt ihren Arbeitsraum unmittelbar mit Menschen teilen, ändern sich die Anforderungen an solche Systeme gravierend. Die Roboter müssen sich menschlichen Gewohnheiten anpassen und ein hohes Maß an Autonomie, Robustheit und Flexibiliät aufweisen (Graf et al., 2004). Unter diesem Blickwinkel erscheinen gewöhnliche auf Differential- oder Ackermann-Kinematik basierende Roboterfahrwerke, ein zu geringes Maß an Flexibilität und Mobiliät aufzuweisen (Connette, 2013).

In Zukunft ist wohl damit zu rechnen, dass Roboter, welche aus der Tierwelt entlehnte Bewegungsapparate aufweisen, ihre momentan noch vorhandenen Nachteile hinsichtlich Energieeffizienz, Traglast, Robustheit, sowie technischen Aufwand, überwinden und den Robotern mit radbasierten Fahrwerken überlegen sein werden (Connette, 2013). Einstweilig sind jedoch Roboter mit Fahrwerken basierend auf Standardrädern, ein guter Kompromiss, die an den modernen Roboter gestellten Anforderungen zu erfüllen. Deshalb befasst sich die vorliegende Arbeit auch mit einem Roboter mit ebensolcher Fahrwerkskinematik.

Das Ziel dieser Arbeit besteht darin, ausgehend von dem Konzept eines Roboters mit vier Rädern und radbasierter Fahrwerkskinematik, die zu Grunde liegende Kinematik zu untersuchen und den Roboter so zu steuern, dass er einer vorgegebenen Trajektorie folgt. Der Einsatz von Simulationsmethoden ermöglicht es, in einem frühem Stadium der Entwicklung erste Erkenntnisse hinsichtlich des potentiellen Verhaltens des realen Systems zu erhalten, welche wiederrum in den Entwicklungsprozess einfließen.

1.2. Einordnung und Abgrenzung der Arbeit

Die vorliegende Arbeit beschäftigt sich mit der fahrdynamischen Auslegung eines vierrädrigen Roboters, welcher allradgetrieben ist und alle vier Räder unabhängig voneinander lenken kann. Der Fokus liegt also auf einem bestimmten Roboter, bei dessen Entwicklung die gewonnenen Erkenntnisse dieser Arbeit einfließen. Zu Beginn der Arbeit werden andere potentielle Ausführungen von radbasierten Roboterfahrwerken diskutiert und miteinander verglichen. Das letztlich umgesetzte Konzept wird hinsichtlich seiner Vor- und Nachteile mit anderen möglichen Konzepten verglichen und die verwendete Methode zur Steuerung und Regelung des Roboters wird von anderen Strategien abgegrenzt.

1.3. Aufbau der Arbeit

Der Aufbau der vorliegenden Arbeit gliedert sich im wesentlichen in drei Abschnitte, siehe Abbildung 1.1. Zuerst werden in Kapitel 2 die notwendigen Grundlagen über Roboter mit radbasierten Fahrwerkskinematiken erarbeitet. Hier wird der Fokus vor allem auf die kinematischen Grundlagen gelegt, da diese für die Gesamtsimulation in Kapitel 4 von Bedeutung sind und in das erstellte Konzept einflossen. Als Grundlage für die Gesamtsimulation dient eine fahrdynamische Voruntersuchung, welche in Kapitel 3 behandelt wird und das erstellte Mehrkörpersimulationsmodell näher erläutert. Im Hauptteil, der Gesamtsimulation, wird eine Co-Simulation betrieben und die Roboterbewegung entlang einer vorgegebenen Trajektorie bestimmt.

1.4. Verwendete Konventionen

Im Folgenden werden die wichtigsten verwendeten Konventionen bezüglich der Darstellung von Variablen erläutert.

Mit $\underline{\underline{\square}}$ wird eine Matrix bezeichnet, $\overrightarrow{\square}$ beschreibt einen Vektor. Hierbei versteht man die physikalische, gerichtete Größe (Betrag und Richtung), eine Ausnahme bildet der in Kapitel 2.2 eingeführte Twist des Roboters, da in diesem die Einheiten nicht konsistent sind. Mittels $\square_{ab}^{c}$ werden Variablen relativ zwischen den Koordinatensystemen b und a im Koordinatensystem c ausgedrückt. Eine Transformationsmatrix von einem Koordinatensystem b nach Koordinatensystem a wird mittels $\underline{\underline{\square}}_{b}^{a}$ dargestellt. $\square_{s,i}$ bezeichnet eine Variable für das Lenken des i-ten Rades des Roboters, während eine Variable $\square_{d,i}$ für die Beschreibung des Antriebes des i-ten Rades des Roboters verwendet wird.

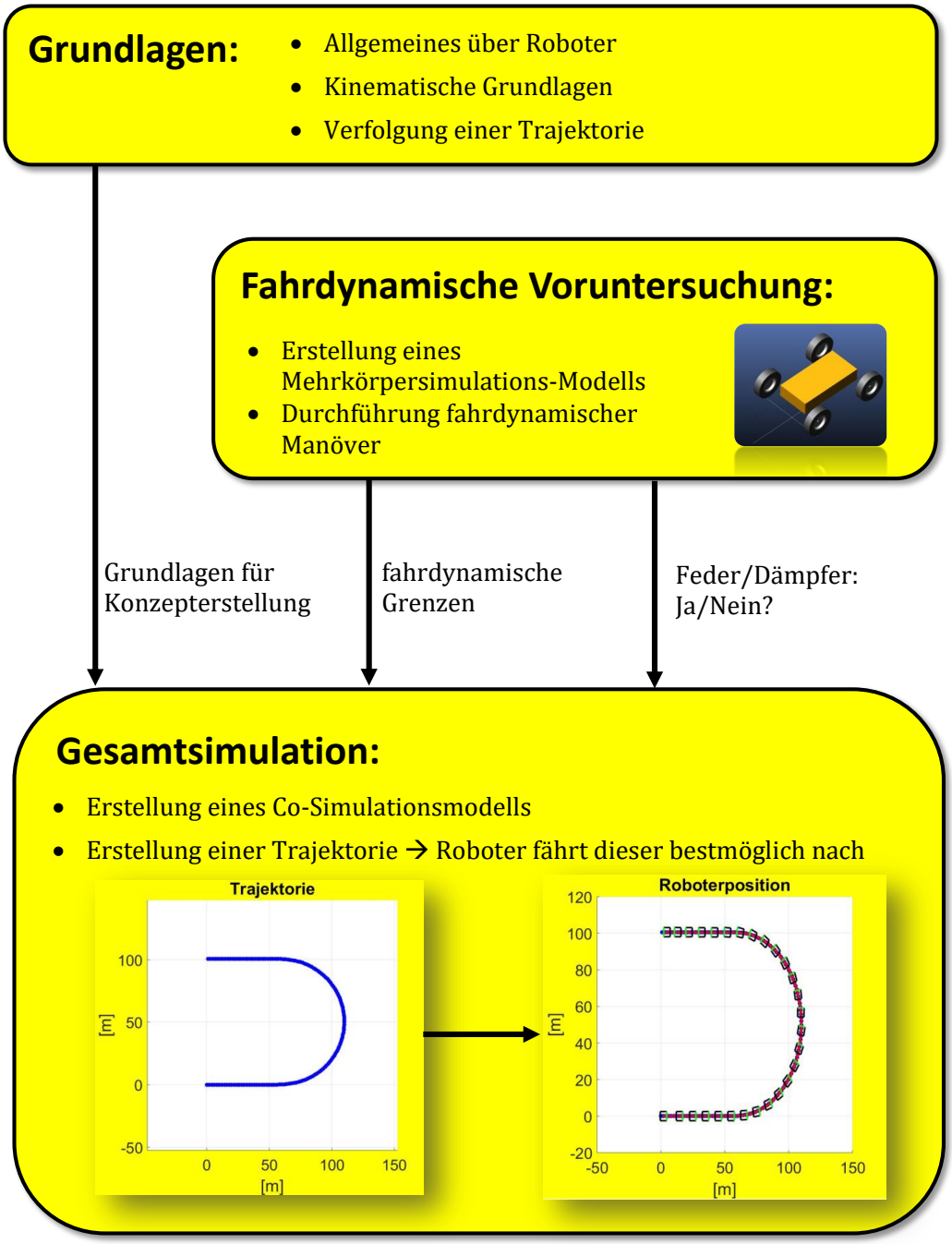


Abbildung 1.1.: Grundsätzlicher Aufbau der vorliegenden Arbeit

2. Grundlagen

In diesem Kapitel werden die Grundlagen von radbasierten Fahrwerken vorgestellt. Im ersten Abschnitt 2.1 wird zuerst ein grober Überblick über Roboter im Allgemeinen geboten, um sich dann im Unterkapitel 2.1.1 auf die möglichen Arten von Rädern, welche für die Anwendung bei Robotern in Frage kommen, zu konzentrieren. Ausgehend von der Fokussierung auf Roboter mit radbasierten Fahrwerken, werden in Kapitel 2.2 die für den weiteren Verlauf der vorliegenden Arbeit notwendigen kinematischen Grundlagen vorgestellt. Dazu zählt eine genaue Betrachtung der verfügbaren kinematischen Freiheitsgrade verschiedener Arten von Robotern, sowie deren mögliche Bestimmung. Ebenfalls genauere Betrachtung findet der Momentanpol, dieser wird unter dem Aspekt einer möglichen Eignung für die Steuerung des Fahrzeuges im Unterkapitel 2.2.1.3 behandelt. Das Unterkapitel 2.2.1.4 befasst sich mit der Thematik der inversen Kinematik und stellt eine notwendige Grundlage für das in Kapitel 4 umgesetzte Konzept dar. Abschluss der kinematischen Grundlagen bildet eine Erläuterung der auftretenden Singularitäten in Unterkapitel 2.2.2. Der Abschluss dieses einleitenden Kapitels findet durch die Betrachtung möglicher Strategien eines Roboters zur Verfolgung einer Trajektorie in Unterkapitel 2.3 statt.

2.1. Allgemeines über Roboter

Roboter sind seit einigen Jahren nicht mehr aus unserem Alltag wegzudenken. In verschiedensten Ausführungsformen vereinfachen sie das Leben der Menschen um ein Vielfaches. Dabei ist das hinter einem Roboter stehende Forschungsgebiet eine höchst interdisziplinäre Wissenschaft, welche Beziehungen zu den unterschiedlichsten Disziplinen aufweisen kann. Dies reicht von offensichtlichen Vertretern wie Informatik, Elektrotechnik, Maschinenbau bis hin zu Bereichen wie Psychologie, sowie auch der Biologie, um Roboter noch ähnlicher ihrem Vorbild, dem Menschen, zu gestalten. Daraus ergeben sich sehr unterschiedliche Einsatzgebiete. Vom Forschungsroboter zur Erkundung fremder Planeten, über den Einsatz beim Militär, beispielsweise als Drohne zur Aufklärung, bis hin zum Industrieroboter zur Erledigung von Fließbandarbeit ist vieles Denkbar und bereits im Einsatz. (Oubbati, 2009)

Der Verein Deutscher Ingenieure (VDI) definiert in seiner Richtlinie 2860 Industrieroboter folgendermaßen:

> *Industrieroboter sind universell einsetzbare Bewegungsautomaten mit mehreren Achsen, deren Bewegungen hinsichtlich Bewegungsfolge und Wegen bzw. Winkel frei programmierbar sind.*

Durch den frei programmierbaren Bewegungsablauf sind sie für verschiedenste Aufgaben einsetzbar.

Ein weiterer wichtiger Begriff ist *autonom*. Darunter versteht man laut Oubbati (2009) einen Roboter der ohne externe Unterstützung in seiner jeweiligen Umgebung seinen angedachten Funktionsumfang erfüllen kann. Je nach Einsatzumgebung erfordert dies verschiedene Grade der Autonomie. Ein Industrieroboter muss demnach einen geringeren Grad an Autonomie aufweisen als etwa ein Roboter welcher als persönlicher Assistent eines Menschen fungieren soll.

Die Bewegungsfreiheit des Roboters ist eine weitere wichtige Unterscheidung. Ist dieser an einen festen Punkt gebunden, spricht man von einem stationärem Roboter. Ein mobiler Roboter hingegen kann sich in seiner jeweiligen Umgebung (Land, Wasser, Luft) frei bewegen. (Oubbati, 2009)

Unabhängig welcher Typ von Roboter, die aller notwendigsten Grundbestandteile müssen bei jedem von ihnen vorhanden sein. Wie in Abbildung 2.1 zu sehen, sind dies mehrere Sensoren und Aktoren, sowie eine Steuereinheit. Dabei stehen die Sensoren an erster Stelle. Sie empfangen Umgebungseinflüsse, also physikalische Signale und reagieren darauf mit einem elektrischen Signal, welches als Information an die Steuereinheit weitergegeben wird. Diese Steuereinheit stellt dann das Gehirn des Roboters dar. Die von den Sensoren erhaltenen Informationen werden verarbeitet, im Falle eines mobilen Roboters wird die gewünschte Bewegung berechnet und die Aktoren (Elektromotor oder ähnliches) angesteuert. (Oubbati, 2009)

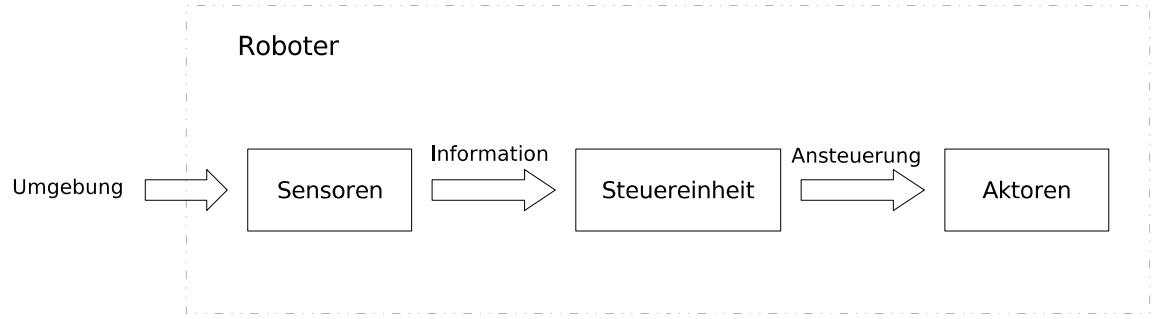


Abbildung 2.1.: Grundbestandteile eines Roboters (Oubbati, 2009, S.11)

Im weiteren Verlauf dieser Arbeit richtet sich der Fokus auf mobile Roboter mit radbasierten Fahrwerken. Dies bedeutet der Antrieb geschieht mittels Räder, wodurch Roboter besonders auf festen Untergründen eine sehr hohe Effizienz der Fortbewegung erreichen. Im Vergleich zu anderen denkbaren Antriebsmethoden, wie etwa Kettenfahrwerke oder auch biomimetischen Systemen (kriechende, raupenartige oder auch schlangenartige Fortbewegung) weisen Radfahrwerke einige praktische Vorteile auf, wie etwa der mechanisch weit weniger komplexe Aufbau. Besonders bei Anwendungen in der Nähe von Menschen geschaffenen Strukturen können radbasierte Fahrwerke ihre zu Grunde liegenden Vorteile vollends ausspielen. (Connette, 2013)

2.1.1. Arten von Rädern

Da sich für Roboter mit radbasierten Fahrwerken eine Vielzahl von verschiedenen Einsatzmöglichkeiten ergibt, hat sich auch eine dementsprechend große Anzahl an Fahrwerksvarianten entwickelt, welche alle im Wesentlichen auf einer Kombination von 4 verschiedenen Radtypen basieren.(Connette, 2013)

In Abbildung 2.2 ist ein Standardrad, lenkbar sowie nicht-lenkbar, zu sehen. Dabei beschreibt die Achse a jeweils die Antriebsachse und b die Lenkachse. Als drittes Rad ist das sogenannte Castor-Rad (auch unter Lenkrolle bekannt) zu sehen. Bei diesem Rad sind gegenüber dem gelenkten Standardrad die beiden Achsen a und b um einen horizontalen Abstand verschoben. Solche Räder werden meist als passive (nicht angetriebene und nicht gelenkte) Räder verwendet. Ein bekanntes Beispiel dafür sind die Räder von Einkaufswägen. In Abbildung 2.3 ist links ein Mecanum-Rad zu sehen. Sie bestehen aus einem Set aus tonnenförmigen, nicht angetriebenen Rollen, welche um einen Winkel, meist um die 45 Grad, um eine zentrale Trägerrolle angeordnet sind. In dieser Rolle befindet sich auch der Antrieb bei solchen Radausführungen. Ein Nachteil bei diesen Rädern ist die beschränkte Funktion bei rauhem Untergrund. Rechts im Bild ist das vierte und letzte der betrachteten Radtypen zu sehen, nämlich das Kugelrad. Es kann sich in allen Richtungen der Ebene frei bewegen, was praktisch jedoch schwierig zu realisieren ist. (Gfrerrer, 2. Ausgabe 2014)

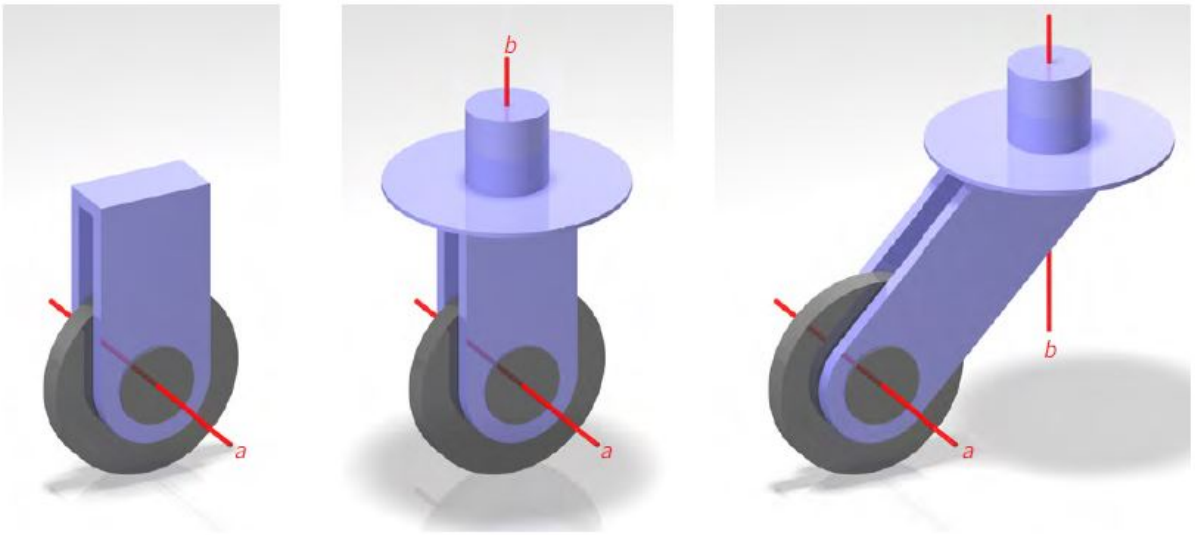


Abbildung 2.2.: Typen von Rädern von links nach rechts: Standardrad (nicht lenkbar), Standardrad (lenkbar) Castor-Rad (Lenkrolle) (Gfrerrer, 2. Ausgabe 2014, S.43)

Prinzipiell ließen sich diese vorgestellten Radtypen beliebig kombinieren, jedoch wird nach Campion et al. (1996) meist eine Einteilung in 5 Kategorien vorgenommen. In der Tabelle 2.1 ist diese Einteilung ersichtlich. Unterscheidungskriterien stellen dabei der Grad der Lenkbarkeit δ_s und der Grad der Mobilität δ_m dar. Diese werden im Kapitel 2.2.1.2 erläutert.

Als erste Spalte in Abbildung 2.1 ist die aus dem PKW (Personenkraftwagen) bekannte Ackermann-Kinematik ersichtlich. Sie setzt sich aus einer Kombination aus gelenkten

	Ackermann-Kinematik	Differential-Kinematik	nicht-holonom, omnidirektional Typ 1	nicht-holonom, omnidirektional Typ 2	holonom, omnidirektional
Radtypen	fixe Räder & gelenkte Räder	fixe Räder	gelenkte Räder	gelenktes Rad & Mecanum-, Castor-Räder, Kugelräder	Mecanum-, Castor-Räder, Kugelräder
Einsatz	Indoor, Outdoor, Offroad, (FTS, PKW, ...)	Indoor (kleine Roboter)	Indoor (Serviceroboter) Offroad (Expl.-Roboter)	Indoor, bei ebenen Böden	Indoor, bei ebenen Böder (Schwerlast-FTS)
Manövrier-barkeit	–	o	+	+	+
Schnelle Fahrt	+	o	o	–	–
Weicher Untergrund	+	o	+	–	–
Unebener Untergrund	+	o	+	–	–
Kosten (Motoren)	+	+	–	o/–	o/–
Hohe Last	+	o	o	o/–	o/–
δ_m	1	2	1	2	3
δ_s	1	0	2	1	0

Tabelle 2.1.: Fahrwerkstypen und deren Vorteile (+), Nachteile (-) und Einsatzbereiche (Connette, 2013, S.32)

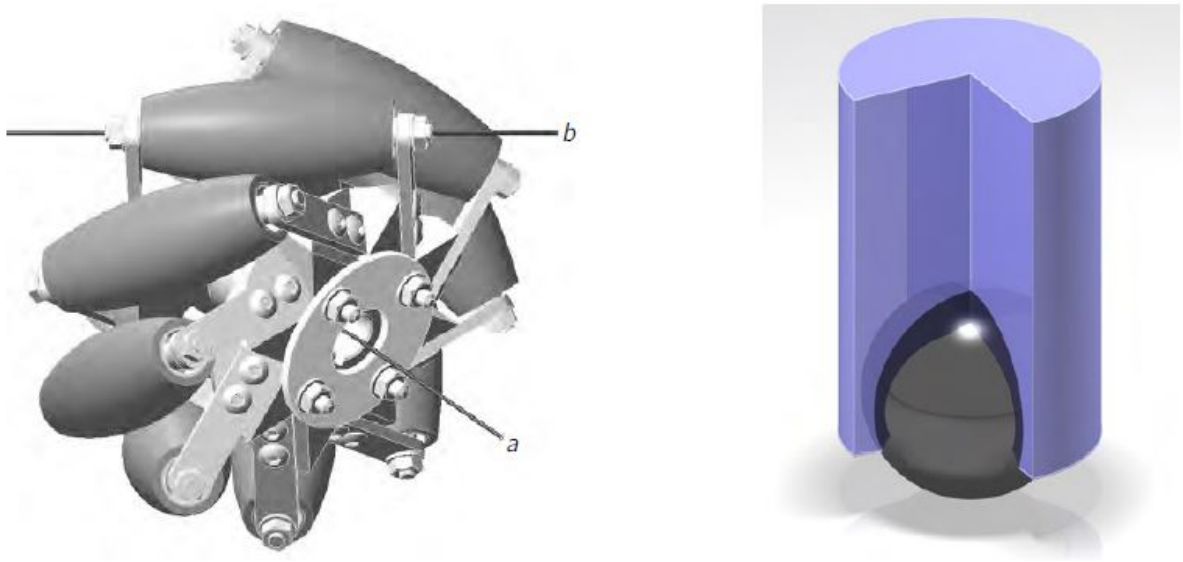


Abbildung 2.3.: Verschiedene Typen von Rädern: links das Mecanum-Rad und rechts das Kugelrad (Gfrerrer, 2. Ausgabe 2014, S.44)

und fixen Standardrädern zusammen (meist zwei gelenkte Räder an der Vorder- und zwei fixe Räder an der Hinterachse) und stellt eine sehr ausgeglichene Lösung dar. Einziger Nachteil ist die eingeschränkte Manövrierbarkeit, siehe hierzu auch Abbildung 2.4. In der zweiten Spalte ist die Differential-Kinematik angeführt. Diese Kinematik entspricht dem Standard bei einfachen Anwendungen in der mobilen Robotik und besteht nur aus fixen Standardrädern. Gelenkt wird dieser Typ von Roboter über eine Drehzahldifferenz zwischen den beiden Seiten des Roboters. Der nächste Vertreter ist die Kinematik "nicht holonom, omnidirektional Typ 1". Eine Erläuterung bezüglich dem Term *holonom* ist in Kapitel 2.2 zu finden. Unter omnidirektional wird *jede Richtung* verstanden. Diese Roboter besitzen ein erhöhtes Maß an Manövrierbarkeit und sind trotz alle dem auch bei weichem und unebenem Untergrund noch einsatzfähig. Auf diesen Typ Roboter wird sich der Fokus dieser vorliegenden Arbeit legen. Die letzen zwei in der Abbildung 2.1 angeführten Roboterkinematiken zeichnen sich durch die höchste Manövrierbarkeit im Vergleich zu den anderen Typen aus. Jedoch sind diese, durch den Einsatz von Mecanum- oder auch Kugelrädern teuer und empfindlich bei weichem und unebenem Untergrund. Eine Anwendung von holonomen, omnidirektionalen Fahrwerken besteht bei führerlosen Transportsystemen (FTS) mit schwerer Last.

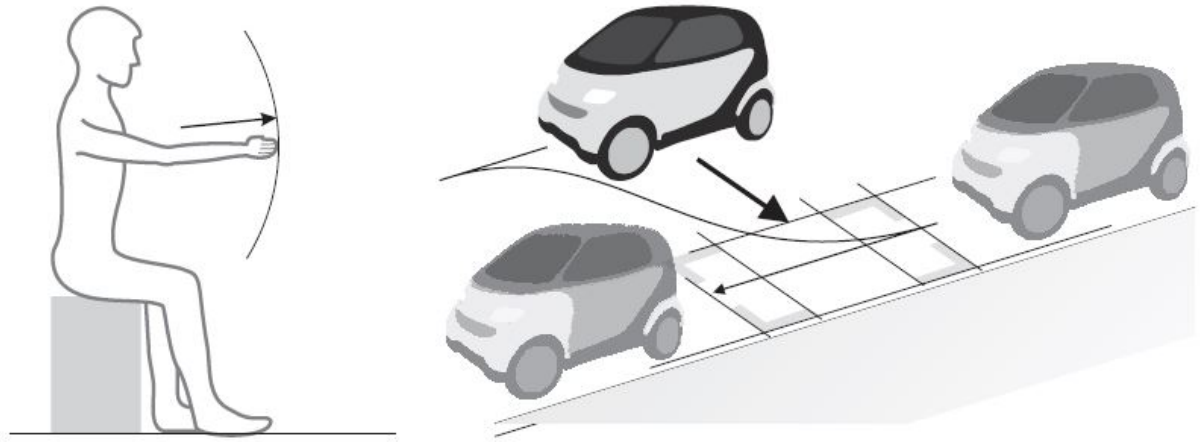
2.2. Kinematische Grundlagen

In diesem Kapitel werden nun die notwendigen kinematischen Grundlagen vorgestellt. Zuerst müssen für den weiteren Verlauf essentielle Bedingungen definiert werden. Nämlich jene der holonomen und nicht-holonomen.

Eine nicht-holonome Bedingung lässt sich als nicht integrable kinematische Beschränkung eines Körpers beschreiben.Darunter versteht man Beschränkungen von möglichen Be-

wegungsrichtungen oder auch Geschwindigkeiten von Körpern, welche sich nicht auf die Position des jeweiligen Körpers zurückführen lassen. Ein praktisches Beispiel für eine holonome Bindung ist im linken Bereich der Abbildung 2.4 zu sehen. Je nach Position des menschlichen Körpers ist die Reichweite seiner ausgestreckten Arme jeweils der für ihn erreichbare Raum. Ein anschauliches Beispiel für eine nicht-holonome Bedingung ist ein PKW, ausgestattet mit der bereits erwähnten Ackermann-Kinematik. Grundsätzlich kann ein PKW alle Punkte in der Ebene erreichen. Jedoch ist beispielsweise ein seitliches Einparken nur durch wiederholtes Rangieren möglich, falls die Größe der Lücke überhaupt ausreichend ist. (Connette, 2013)

Eine genaue Betrachtung einer essentiellen nicht-holonomen Bindung, nämlich jener bei dem Rad-Straße-Kontakt, geschieht in Kapitel 2.2.1.1.



(a) Holonome Beschränkung - Armlänge (b) Nicht-holonome Beschränkung - seitliches Einparken eines PKW

Abbildung 2.4.: Vergleich zwischen holonomer und nicht-holonomer Beschränkung (Connette, 2013, S.39)

Um im weiteren Verlauf die notwendigen kinematischen Überlegungen darzulegen, muss zuerst der Roboter selbst ausreichend im Raum definiert werden. Wie in Abbildung 2.5 zu erkennen ist, gibt es ein Weltkoordinatensystem (im weiteren Verlauf auch als globales Koordinatensystem bezeichnet) mit dem Ursprung O_w welches mit w indiziert ist. In diesem globalen System befindet sich der Roboter mit der Position (x_r, y_r). Das Roboterkoordinatensystem wird mit r indiziert und hat seinen Ursprung O_r im Schwerpunkt des Roboters. Mit φ_r wird der Winkel des Roboters bezüglich dem festen Weltkoordinatensystem bezeichnet. Da mobile Roboter meist in Bereichen eingesetzt werden in denen eine, zumindest in erster Näherung, ebene Fläche befahren wird, kann auf eine erweiterte Darstellung (schiefe Ebene, zusätzliche Winkel bezüglich der z- und y-Achse) verzichtet werden. Die relative Lage des Roboters zwischen Roboter- und Weltkoordinatensystem ist somit beschrieben durch (x_r, y_r, φ_r) . Wenn man nun noch die Koordinaten $(\overrightarrow{\varphi_s})$ als Vektor aller Lenkwinkel der lenkbaren Roboterräder, sowie $(\overrightarrow{\varphi_d})$ für die Rotation der Räder einführt, erhält mit die sogenannte Konfiguration des Roboters. (Connette, 2013)

Die Bewegung des Roboters im Weltkoordinatensystem wird meist als Twist oder Drehwinder zusammengefasst, siehe Gleichung 2.1. Unter $v_{x,w}$ und $v_{y,w}$ versteht man die kartesischen Geschwindigkeiten des Roboters in Weltkoordinaten, φ_{wr}^{w} bezeichnet

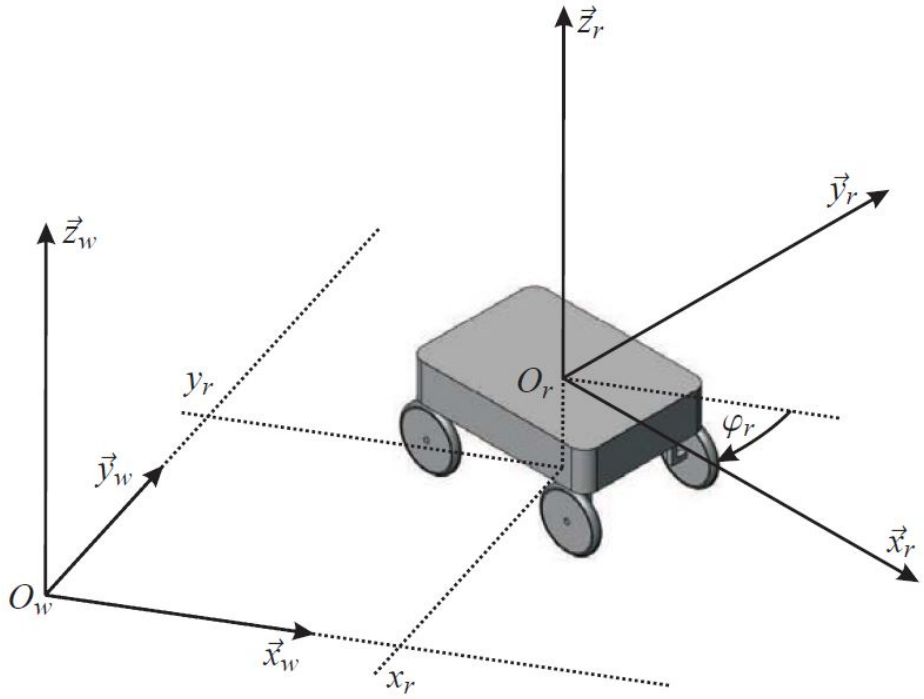


Abbildung 2.5.: Darstellung der Beziehung zwischen Roboter- und Weltkoordinatensystem (Connette, 2013, S.34)

die Orientierung des Roboters, also den Winkel zwischen den x-Achsen von Welt- und Roboterkoordinatensystem, in der Abbildung 2.5 wird dieser Winkel als φ_r bezeichnet. Für den Twist wird der Winkel abgeleitet um mit $\dot{\varphi}^w_{wr}$ die Winkelgeschwindigkeit zu erhalten. Um den Twist im Koordinatensystem des Roboters zu bekommen, muss der auf das Weltkoordintatensystem bezogene Twist mit der Matrix aus Gleichung 2.2 multipliziert werden. Dann erhält man den in Gleichung 2.3 ersichtlichen Twist bezogen auf das Roboterkoordinatensystem. (in Anlehnung an Connette, 2013)

$$\overrightarrow{t^w_{wr}} = \begin{pmatrix} v_{x,w} \\ v_{y,w} \\ \dot{\varphi}^w_{wr} \end{pmatrix} \tag{2.1}$$

$$\underline{\underline{R}}^r_w(\varphi_r) = \begin{pmatrix} \cos(\varphi_r) & -\sin(\varphi_r) & 0 \\ \sin(\varphi_r) & \cos(\varphi_r) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.2}$$

$$\overrightarrow{t^r_{wr}} = \underline{\underline{R}}^r_w(\varphi_r) \cdot \overrightarrow{t^w_{wr}} \tag{2.3}$$

2.2.1. Kinematische Freiheitsgrade

In diesem Abschnitt folgt nun eine genauere Betrachtung der vorhandenen kinematischen Freiheitsgrade, ebenso wie die vorhandenen Einschränkungen. Wie bereits die Tabelle 2.1 versuchte aufzuzeigen, unterscheiden sich die möglichen 5 Typen von

Roboter-Fahrwerkskinematiken in ihren vorhandenen kinematischen Freiheitsgraden. Diese entscheiden über die Manövrierbarkeit und geben so nicht nur ein wichtiges Entscheidungskriterium für eine bestimmte Art von Fahrwerkskinematik vor, sondern ermöglichen diese übersichtliche Unterscheidung der einzelnen Typen. Im weiteren wird nun versucht auf den Ursprung dieser Freiheitsgrade hinzuführen.

2.2.1.1. Zwangsbedingungen

Den meisten mechanischen Systemen sind in verschiedenster Art und Weise Zwangsbedingungen auferlegt. Die Ursachen dieser auferlegten Zwänge können innere Kräfte (Abstand der Massenpunkte eines festen Körpers bleibt, eine Idealvorstellung vorausgesetzt, immer gleich), oder auch äußere Kräfte (Gas wird in einem Behälter mit konstantem Volumen eingeschlossen) sein. (Straumann, 2015)

Eine, besonders im Bereich der mobilen Roboter mit radbasierten Fahrwerken, sehr wichtige Zwangsbedingung ist jene des abrollenden Rades auf einer Ebene (Rad-Straße-Kontakt). In Abbildung 2.6 ist dies vereinfacht mit einer rollenden Scheibe dargestellt. Die Scheibe besitzt den Radius a, sie soll ohne Schlupf auf der (x, y) - Ebene abrollen und ihre Rotationsachse ist dabei stets parallel zu der Abroll-Ebene. Zur Beschreibung der Bewegung wird ein Punkt P auf der Scheibe gewählt. Der Winkel ϑ beschreibt somit den Winkel zwischen diesem Punkt P und dem Kontaktpunkt der Scheibe mit der Ebene (Punkt Q). Die Ableitung des Winkels ϑ entspricht der Winkelgeschwindigkeit der Scheibe. Der Winkel φ wird zwischen Tangente an die Scheibe im Punkt Q und der x-Achse des Koordinatensystem gemessen. Somit bestimmen die Größen $(x, y, \vartheta, \varphi)$ vollständig die Positon der Scheibe. (Straumann, 2015)

Die Bedingung, dass die Scheibe ohne Schlupf auf der Ebene abrollen soll bedingt, dass die Geschwindigkeit im Punkt Q immer null ist. Die Geschwindigkeit setzt sich zusammen aus der Mittelpunktsgeschwindigkeit der Scheibe, sowie der Geschwindigkeit auf Grund der Rotation mit der Winkelgeschwindigkeit $\dot{\vartheta}$, wie in der Gleichung 2.4 ersichtlich. Auf Grund mangelnder Kenntnisse über den Verlauf von ϑ kann diese nicht integriert werden und ist somit eine nicht holonome Bindung.

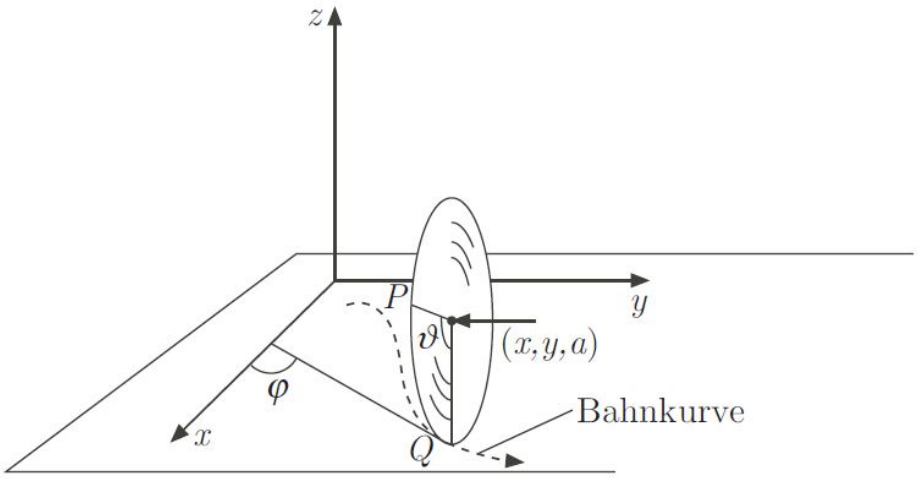


Abbildung 2.6.: Zwangsbedingung für eine rollende Scheibe (Straumann, 2015, S.96)

$$\begin{aligned} \dot{x} + a \cdot \dot{v} \cdot \cos(\varphi) &= 0 \\ \dot{y} + a \cdot \dot{v} \cdot \sin(\varphi) &= 0 \end{aligned} \tag{2.4}$$

Mit dem eben vorgestellten Beispiel der rollenden Scheibe wurde versucht eine einfache Erklärung für die nicht-holonome Bindung bei einem schlupffrei abrollenden Rad zu liefern. Nun werden die beim gelenkten Standardrad auftretenden Bedingungen genauer betrachtet. Namentlich geht es hierbei um die bereits bei der rollenden Scheibe behandelte Rollbedingung, sowie die sogenannte non-slipping Bedingung. Hierbei wird vorausgesetzt, dass die Geschwindigkeit senkrecht zur Fahrtrichtung des Rades zu null wird (kein Rutschen des Rades). Die folgende Herleitung dieser Bedingungen folgt in Anlehnung an Connette (2013).

Prinzipiell wird der Geschwindigkeitsvektor längs der x-Achse des betrachteten Rades, sowie der Geschwindigkeitsvektor senkrecht dazu, gebildet. Zu sehen ist dies in Abbildung 2.7. Beide Geschwindigkeiten müssen im Kontaktpunkt mit der Ebene null werden (entspricht dem Punkt Q, siehe Abbildung 2.6).

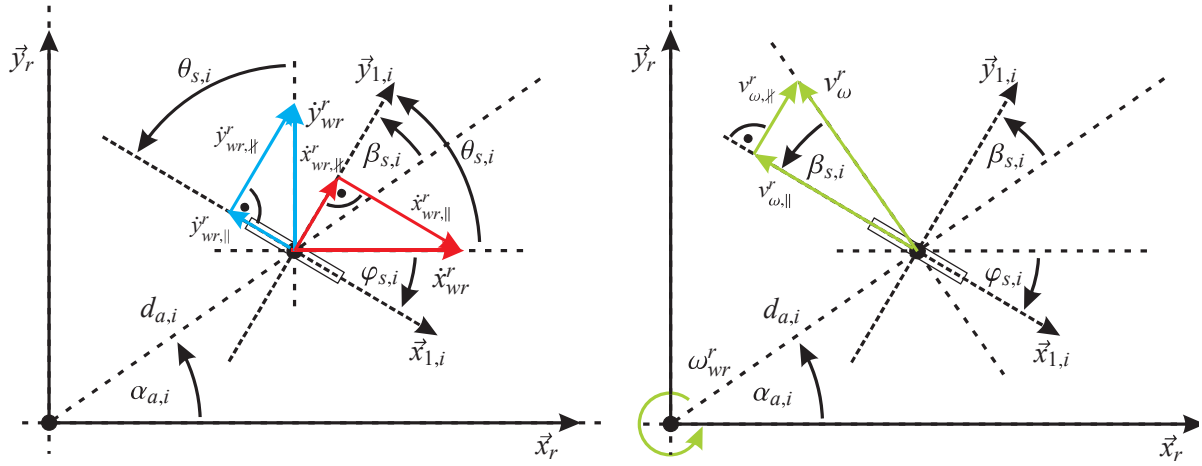


Abbildung 2.7.: Resultierende Geschwindigkeitsvektoren bei einem Standardrad (Connette, 2013, S.192)

Wie in Gleichung 2.5 ergeben die Anteile in x-Richtung des Rades der Geschwindigkeitsvektoren im Roboterkoordinatensystem, zusammen mit dem parallelen Anteil des Vektors v^r_ω, welcher eine zusätzliche Drehung des Roboters berücksichtigt, den resultierenden Vektor parallel zur x-Achse des Rades. Ident dazu, nur in y-Richtung des Rades, ergibt sich der resultierende Geschwindigkeitsvektor in Gleichung 2.8. Beide Gleichungen wurden so modifiziert, dass die Darstellung mittels dem bereits erwähnten Twist, bezogen auf das Roboterkoordinatensystem, möglich wird. Dabei wurden die Winkel $\beta_{s,i}$, sowie $\theta_{s,i}$ eingeführt, siehe Gleichung 2.6, beziehungsweise Gleichung 2.7.

$$\begin{aligned} v_{res,\parallel} &= \dot{x}^r_{wr,\parallel} + \dot{y}^r_{wr,\parallel} + v^r_{\omega,\parallel} \\ &= \left(\sin(\theta_{s,i}), -\cos(\theta_{s,i}), -d_{a,i} \cdot \cos(\beta_{s,i})\right) \cdot \overrightarrow{t^r_{wr}} \end{aligned} \tag{2.5}$$

$$\beta_{s,i} = \varphi_{s,i} - \alpha_{a,i} + \frac{\pi}{2} \tag{2.6}$$

$$\begin{aligned} \theta_{s,i} &= \beta_{s,i} + \alpha_{a,i} \\ &= \varphi_{s,i} + \frac{\pi}{2} \end{aligned} \tag{2.7}$$

$$\begin{aligned} v_{res,\nparallel} &= \dot{x}^r_{wr,\nparallel} + \dot{y}^r_{wr,\nparallel} + v^r_{\omega,\nparallel} \\ &= \left(\cos\left(\theta_{s,i}\right), \sin\left(\theta_{s,i}\right), d_{a,i} \cdot \sin\left(\beta_{s,i}\right)\right) \cdot \overrightarrow{t^r_{wr}} \end{aligned} \tag{2.8}$$

In Gleichung 2.9 ist nun die eigentliche Formulierung der Rollbedingung zu sehen. Dabei wurden die verwendeten Winkelbeziehungen wieder rück-substituiert und dann mit dem Geschwindigkeitsvektor $r \cdot \dot{\varphi}_{d,i}$ welcher im Kontaktpunkt des Rades mit der Ebene beim Abrollen auftritt, gleichgesetzt. Umgeformt ergibt sich, ident zu der formulierten Bedingung bei der rollenden Scheibe, dass die Geschwindigkeit im Kontaktpunkt null sein muss. Die non-slipping Bedingung ergibt sich direkt aus der Forderung, dass der in Gleichung 2.8 erhaltene resultierende Geschwindigkeitsvektor null sein muss. In beiden Bedingungen wird der Twist in Bezug zum Weltkoordinatensystem verwendet und muss deshalb zusätzlich mit $\underline{\underline{R^r_w}}$ multipliziert werden.

$$\left(-\cos\left(\varphi_{s,i}\right), -\sin\left(\varphi_{s,i}\right), -d_{a,i} \cdot \sin\left(\varphi_{s,i} - \alpha_{a,i}\right)\right) \cdot \underline{\underline{R^r_w}}\left(\varphi^w_{wr}\right) \cdot \overrightarrow{t^w_{wr}} + r \cdot \dot{\varphi}_{d,i} = 0 \tag{2.9}$$

$$\left(-\sin\left(\varphi_{s,i}\right), \cos\left(\varphi_{s,i}\right), d_{a,i} \cdot \cos\left(\varphi_{s,i} - \alpha_{a,i}\right)\right) \cdot \underline{\underline{R^r_w}}\left(\varphi^w_{wr}\right) \cdot \overrightarrow{t^w_{wr}} = 0 \tag{2.10}$$

$$\underline{\underline{R^r_w}} = \begin{bmatrix} \cos\left(\varphi^w_{wr}\right) & \sin\left(\varphi^w_{wr}\right) & 0 \\ -\sin\left(\varphi^w_{wr}\right) & \cos\left(\varphi^w_{wr}\right) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.11}$$

2.2.1.2. Bestimmung der vorhandenen Freiheitsgrade

Im Kapitel 2.2.1.1 wurden, exemplarisch für ein Standardrad, die vorhandenen kinematischen Zwangsbedingungen abgeleitet. Es wird der Versuch einer allgemeineren Darstellung unternommen, wofür nach Siciliano & Khatib (2008) die Gleichung 2.9 allgemeiner dargestellt wird, indem die Terme vor der Transformationsmatrix $\underline{\underline{R^r_w}}$ zusammengefasst werden. Dies wird für alle Räder bei dem jeweilig betrachteten Roboter durchgeführt. Somit entsteht die in Gleichung 2.12 ersichtliche Matrix $\underline{\underline{C}}_1\left(\varphi_{s,S}\right)$, diese ist von allen vorhandenen Lenkwinkeln $\varphi_{s,S}$ abhängig. Jede Zeile der Matrix ist eine Funktion eines Lenkwinkels, die Matrix hat somit die Größe $(N \times 3)$, wobei N der Anzahl an Rädern entspricht. Diese Matrix lässt sich aufspalten, siehe Gleichung 2.13, in einen Teil mit ungelenkten und einen Teil mit gelenkten Rädern. $\underline{\underline{C}}_{1f}$ steht

für die ungelenkten Räder und hat die Größe $\left(N_f \times 3\right)$, N_f steht für die Anzahl der ungelenkten Räder.

$$\underline{\underline{C}}_1\left(\varphi_{s,S}\right) \cdot \underline{\underline{R}}_w^r \cdot \overrightarrow{t_{wr}^w} = 0 \tag{2.12}$$

$$\underline{\underline{C}}_1\left(\varphi_{s,S}\right) = \begin{pmatrix} \underline{\underline{C}}_{1f} \\ \underline{\underline{C}}_{1s} \end{pmatrix} \tag{2.13}$$

Mit Hilfe dem Rang der Matrix $\underline{\underline{C}}_1\left(\varphi_{s,S}\right)$, äquivalent mit der Anzahl an linear von einander abhängigen Zeilen, kann nun die in Tabelle 2.1 getroffene Einteilung erläutert werden. In Gleichung 2.14 wird der Grad der Mobilität definiert, es gilt $1 \leq \delta_m \leq 3$. Der Rang der Matrix kann zwischen null und drei liegen, da die Spaltenanzahl immer drei beträgt. Um einen bewegungsfähigen Roboter zu erhalten muss $\delta_m > 0$ sichergestellt sein. (Siciliano & Khatib, 2008)

$$\delta_m = 3 - rang[\underline{\underline{C}}_1\left(\varphi_{s,S}\right)] \tag{2.14}$$

Der Grad der Lenkbarkeit entspricht direkt dem Rang der Matrix $\underline{\underline{C}}_1\left(\varphi_{s,S}\right)$, siehe Gleichung 2.15. Es gilt $0 \leq \delta_s \leq 2$. Die obere Grenze wird nur für Roboter ohne ungelenkte Räder erreicht. Der untere Wert wird nur für Roboter ohne lenkbare Räder erreicht. Ein Roboter der mehr gelenkte Räder als δ_s besitzt, muss deren Bewegung auf einander abstimmen. (Siciliano & Khatib, 2008)

$$\delta_s = rang[\underline{\underline{C}}_1\left(\varphi_{s,S}\right)] \tag{2.15}$$

Die Addition beider Freiheitsgrade ergibt den Grad der Manövrierbarkeit, siehe Gleichung 2.16. Dieser liegt zwischen zwei und drei, zwei Freiheitsgrade sind notwendig sind für eine Bewegung in der Ebene. Das Maximum für die ebene Bewegung wiederum liegt bei drei. Durch die Kombination dieser Freiheitsgrade können nun die bereits erwähnten 5 Typen von Fahrwerkskinematiken erhalten werden.

$$\delta_M = \delta_m + \delta_s \tag{2.16}$$

2.2.1.3. Kinematische Beschreibung über den Momentanpol

Eine weitere wichtige Erkenntnis liefert die geometrische Interpretation der non-slipping Bedingung. Sie liefert eine anschauliche Erklärung der Existenz des Momentanpols (MP). Wenn wir von einem festen Körper ausgehen, welcher sich auf einer allgemeinen Kreisbahn bewegt, dann muss der Geschwindigkeitsvektor in jedem Punkt dieses Körpers orthogonal zu der Verbindung des Punktes am Körper mit dem Momentanpol sein. Dies gilt auch für jedes Rad, insbesondere den betrachteten Kontaktpunkt zwischen Rad und der jeweiligen Ebene. Eine kinematisch einwandfreie Bewegung ist somit nur unter Einhaltung der non-slipping Bedingung möglich und entspricht dem Prinzip der Festkörperbewegung nach Descartés. (Connette, 2013)

Es stellt sich nun die Frage ob es möglich ist, den Roboter unter Vorgabe eines Momentanpols zu steuern. Hierfür wird, nach Dietrich et al. (2011), der Momentanpol auf das Roboterkoordinatensystem bezogen und in Polarkoordinaten dargestellt. Zur besseren Veranschaulichung siehe Abbildung 2.8, wobei unter *icc* der Momentanpol und unter α_c der Winkel φ^r_{MP} verstanden wird. Der bezeichnete Abstand des Momentanpols zum Robotermittelpunkt r_c wird im weiteren als r^r_{MP} bezeichnet. Unter $\dot{\alpha_c}$ wird die Winkelgeschwindigkeit verstanden mit der sich der als Festkörper betrachtete Roboter um den Momentanpol bewegt.

Wie bereits in Kapitel 2.2 beschrieben, kann ein Roboter eindeutig durch die Angabe von Position und Winkel global definiert werden. Hier offenbart sich jedoch ein Problem der Darstellung über den Momentanpol. Wird sowohl die Geschwindigkeit v_c als auch die Winkelgeschwindigkeit $|\dot{\alpha_c}|$ verdoppelt, ergibt sich der selbe Radius r^r_{MP}, der Roboter vollführt jedoch nicht dieselbe Bewegung. Der Übergang von der Beschreibung der Roboterposition im Raum auf die Darstellung über den Momentanpol, ist somit nicht eindeutig und nur eingeschränkt für eine Steuerung des Roboters verwendbar. (Dietrich et al., 2011) Präzise formuliert ergeben sich für jeden Momentanpol, bei Rädern ohne Lenkwinkelbeschränknungen, 2^N mögliche Varianten, wobei mit N die Anzahl der lenkbaren Rädern gemeint ist. (Connette, 2013)

$$\begin{pmatrix} x_{MP} \\ y_{MP} \end{pmatrix} = r^r_{MP} \cdot \begin{pmatrix} \cos\varphi_{MP} \\ \sin\varphi_{MP} \end{pmatrix} \tag{2.17}$$

$$r^r_{MP} = \frac{v_c}{|\dot{\varphi^r_{MP}}|} \quad , \quad v_c = \sqrt{v^2_{x,w} + v^2_{y,w}} \tag{2.18}$$

Ein weiteres Problem bei der Darstellung über den Momentanpol stellt die Geradeausfahrt dar. Stehen alle Räder parallel liegt der Momentanpol im Unendlichen $\left(r^r_{MP} \to \infty\right)$. Abhilfe schafft hier die Substitution des Radius mit der Krümmung $\left(\rho = \frac{1}{r^r_{MP}}\right)$. (Todoran & Bader, 2016)

Die vorangegangenen Ausführungen bezüglich dem Momentanpol legen nahe, dass dieser als Definition der Bewegung des Roboters im Raum unzureichend ist. Ein Blick auf die Abbildung 2.9 soll dies untermauern. Auf der linken Seite ist ein Roboter zu sehen welcher einen Viertelkreis fährt. Dabei dreht sich der Aufbau des Roboters mit, der Winkel zwischen Roboteraufbau und dem zu fahrenden Pfad bleibt dabei idealisiert null. Mit anderen Worten die Rotation des Roboters (ω_b) entspricht der Drehung des Körpers um den Momentanpol (hier dargestellt als Schnittpunkt der orthogonal auf die Geschwindigkeitsvektoren am Rad stehenden Linien). Der rechte Teil der Abbildung 2.9 offenbart jedoch ein Problem, denn hier durchfährt der Roboter ebenfalls einen Viertelkreis, jedoch bleibt der Winkel des Roboteraufbaus konstant. Dies lässt sich durch eine parallele Stellung aller Räder realisieren. Es kann kein Momentanpol durch den Schnittpunkt der orthogonal auf den Geschwindigkeitsvektoren der Räder stehenden Linien gefunden werden, da dieser im Unendlichen liegen würde. Der Roboter als Festkörper macht jedoch eine Rotationsbewegung um einen virtuellen Momentanpol, welcher sich durch den Schnittpunkt der orthogonal auf den Geschwindigkeitsvektor des Robotermittelpunktes liegenden Linien ergibt. Die Änderung der Richtung des Geschwindigkeitsvektor bezüglich dem Roboter wird als

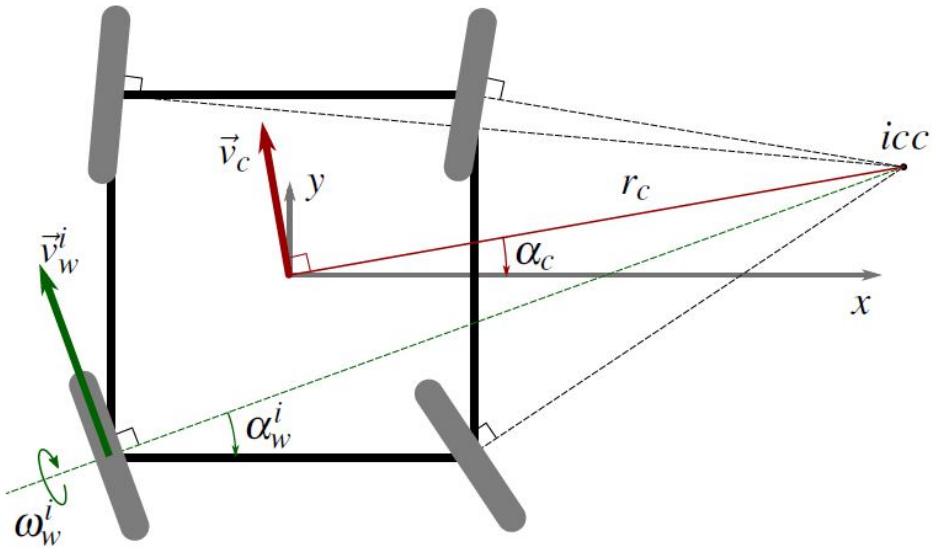


Abbildung 2.8.: Berechnungen der Radvektoren über Kenntnis des Momentanpols (Todoran & Bader, 2016)

ω_c bezeichnet. Wenn der Roboter links mit der gleichen Geschwindigkeit unterwegs ist wie rechts, gilt $\omega_b = \omega_c$. Allgemeiner lässt sich formulieren, dass die gesamte Rotationsbewegung des Roboters, als Festkörper betrachtet, einer Addition der beiden Terme ω_b und ω_c entspricht, wie in Gleichung 2.19 ersichtlich. Für eine vollständige Beschreibung des Roboters ist es notwendig den Verlauf des Geschwindigkeitsvektors des Robotermittelpunktes, als auch die Orientierung des Roboters anzugeben, um Eindeutigkeit zu erreichen. (Todoran & Bader, 2016)

$$\omega_{traj} = \omega_b + \omega_c \tag{2.19}$$

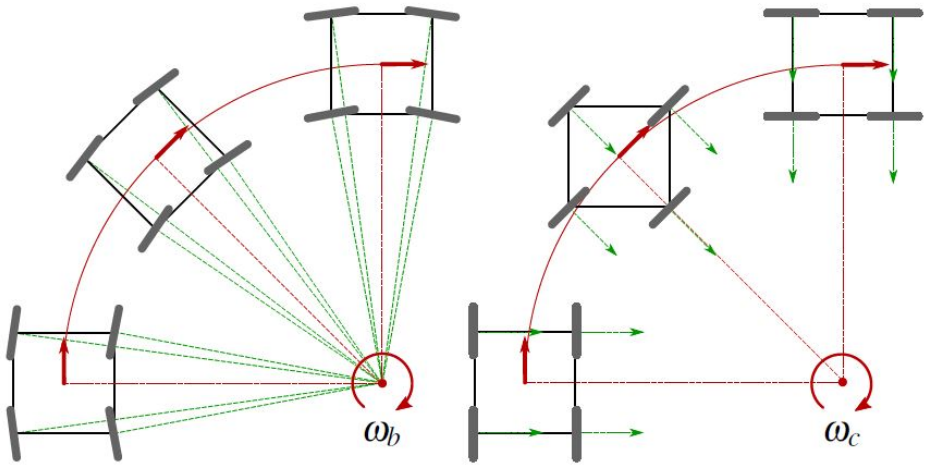


Abbildung 2.9.: Momentanpol über Schnittpunkt der Radvektoren sowie virtueller Momentanpol (Schnittpunkt der Geschwindigkeitsvektoren des Roboters) (Todoran & Bader, 2016)

Die Bewegung des Roboters durch die Lage des Momentanpols zu beschreiben, führt jedoch zu einer anschaulichen Darstellung der bereits besprochenen kinematischen

Freiheitsgrade, was in Gleichung 2.20 ersichtlich ist. Der Roboter wird durch seine Schwerpunktsgeschwindigkeit in x- und y-Richtung $(\dot{x}^r_{wr}, \dot{y}^r_{wr})$, seine Orientierung im Roboterkoordinatensystem φ^r_{MP}, sowie der Position des Momentanpols in Polarkoordinaten beschrieben. Die Freiheitsgrade werden dann durch den Vektor auf der rechten Seite der Gleichung definiert. Dabei beschreibt ρ_{MP} den Betrag und das Vorzeichen der Rotationsbewegung um den Momentanpol.

Die Freiheitsgrade u_φ und u_r entsprechen der Position des Momentanpols. Man erhält somit insgesamt drei Freiheitsgrade, was mit der Anzahl an Freiheitsgraden eines Roboters mit gelenkten Standardrädern übereinstimmt. Dabei kann nun ρ_{MP} dem Grad der Mobilität zugeordnet werden. Mit ihm kann direkt die Rotationsgeschwindigkeit des Roboters um den Momentanpol beeinflusst werden. Die restlichen zwei Freiheitsgrade können dem Grad der Lenkbarkeit zugeordnet werden, sie beinflussen die Position des Momentanpols (in diesem dargestellten Fall als Polarkoordinaten). (Connette, 2013)

$$\begin{pmatrix} \dot{x}^r_{wr} \\ \dot{y}^r_{wr} \\ \dot{\varphi}^r_{wr} \\ \varphi^r_{MP} \\ r^r_{MP} \end{pmatrix} = \begin{pmatrix} \sin\left(\varphi^r_{MP}\right) & 0 & 0 \\ -\cos\left(\varphi^r_{MP}\right) & 0 & 0 \\ \frac{1}{r^r_{MP}} & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \rho_{MP} \\ u_\varphi \\ u_r \end{pmatrix} \tag{2.20}$$

Durch Lenken der Standardräder kann der Momentanpol somit in jede Position in der Ebene gebracht werden. Allerdings ist dabei zu beachten, dass dies nur auf kontinuierlichen Pfaden passieren kann, siehe Abbildung 2.10. Ein sprunghaftes Verändern des Momentanpols ist nur für Roboterfahrwerke mit maximaler Mobilität möglich, zum Beispiel ein Roboter mit vier Kugelrädern, wie er in Abbildung 2.11 ersichtlich ist.

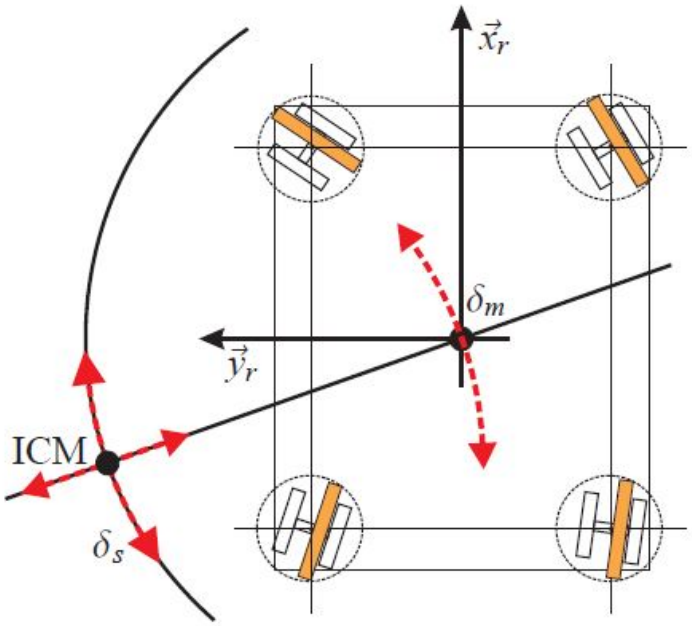


Abbildung 2.10.: Freiheitsgrade des Momentanpols in Polarkoordinaten (Connette, 2013, S.46)

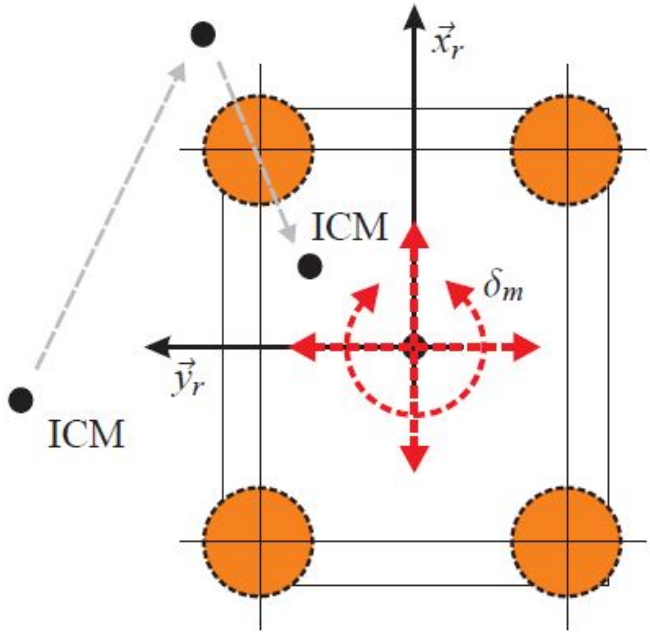


Abbildung 2.11.: Momentanpol bei omnidirektionaler Kinematik (Connette, 2013, S.204)

2.2.1.4. Inverse Kinematik

Mit Kenntnis des Geschwindigkeitvektors eines Roboters, sowie seiner Drehung im Raum, lässt sich direkt auf die Geschwindigkeitsvektoren am Rad, unter Einhaltung der bereits besprochenen Zwangsbedingungen, schließen. Die Ausgangssituation ist in Abbildung 2.12 zu sehen. Es wird eine kinematische Kette, bestehend aus einer Ebene Σ_0, auf welcher das Rad abrollt, dem Roboteraufbau Σ_1, der Radaufhängung Σ_2, sowie dem Rad Σ_3, beschrieben. Wir beziehen uns auf ein gelenktes Standardrad, es gelten also die in Gleichung 2.21 zu sehenden Einschränkungen. (Gfrerrer, 2. Ausgabe 2014)

$$\left\{ \begin{array}{c} nicht\ lenkbares\ Standardrad \\ lenkbares\ Standardrad \\ Lenkrolle\ (Castor-Rad) \end{array} \right\} = \left\{ \begin{array}{c} p = 0\ ,\ \psi = const. \\ p = 0\ ,\ \psi = \psi(t) \\ p \neq 0\ ,\ \psi = \psi(t) \end{array} \right\} \tag{2.21}$$

Folgende Ableitungen geschehen in Anlehnung an Gfrerrer (2. Ausgabe 2014). Zunächst wird in Gleichung 2.22 die bereits bekannte non-slipping Bedingung forumliert. Die Indizes der Vektoren geben an zwischen welchen Elementen der kinematischen Kette sie wirken.

$$\overrightarrow{v_{P,03}} = \overrightarrow{v_{P,01}} + \overrightarrow{v_{P,23}} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{2.22}$$

In Gleichung 2.23 wird der Positionsvektor des Punktes P aufgestellt.

$$\begin{bmatrix} p_x \\ p_y \end{bmatrix} = \begin{bmatrix} c \cdot \cos\gamma + p \cdot \cos(\gamma + \psi) \\ c \cdot \sin\gamma + p \cdot \sin(\gamma + \psi) \end{bmatrix} \tag{2.23}$$

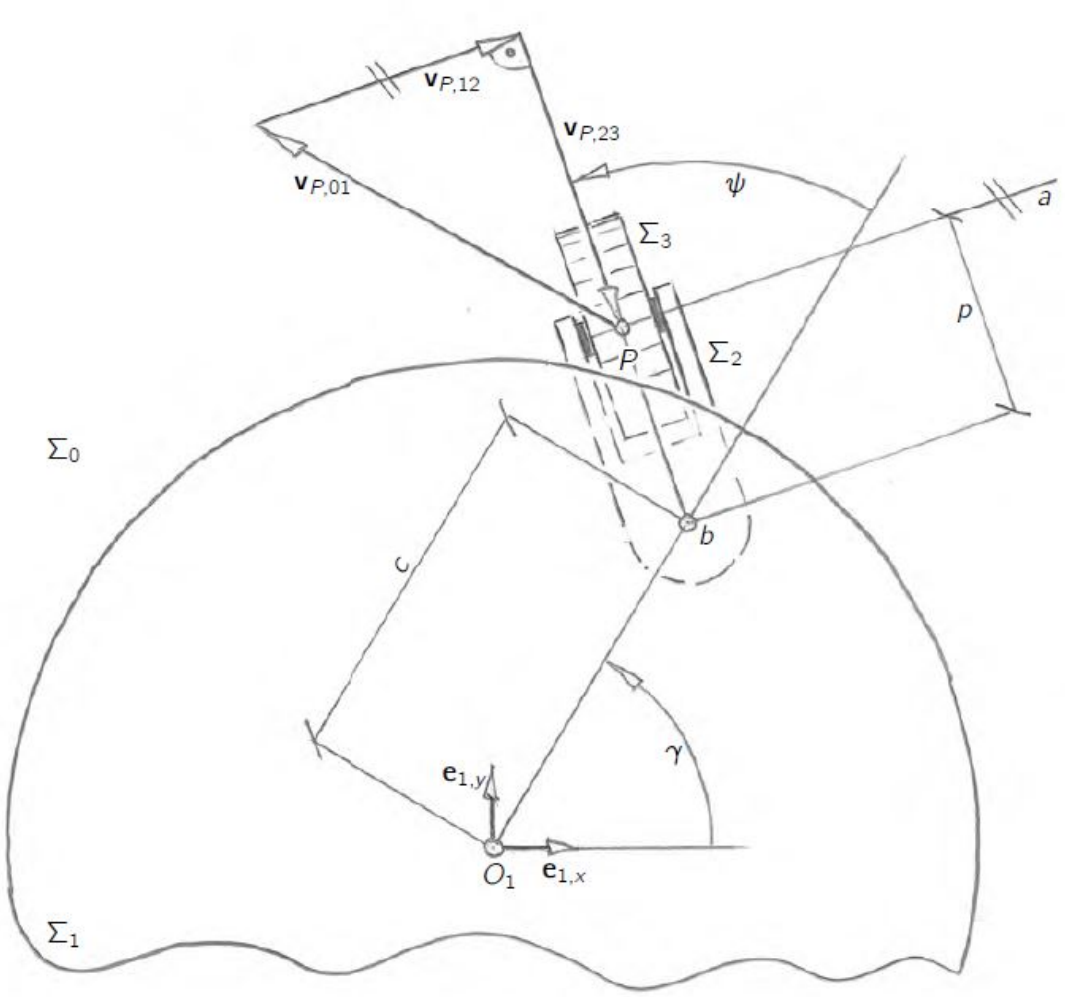


Abbildung 2.12.: Kinematische Beschreibung eines Standard- bzw. Castor-Rades (Gfrerrer, 2. Ausgabe 2014, S.45)

Da sich der Punkt P in einem bewegten System befindet ergibt sich der Geschwindigkeitsvektor zu:

$$\overrightarrow{v_{P,01}} = \begin{bmatrix} v_x - \omega \cdot p_y \\ v_y + \omega \cdot p_x \end{bmatrix} = \begin{bmatrix} v_x - \omega \cdot (c \cdot \sin\gamma + p \cdot \sin(\gamma + \psi)) \\ v_y + \omega \cdot (c \cdot \cos\gamma + p \cdot \cos(\gamma + \psi)) \end{bmatrix} \tag{2.24}$$

Dabei beschreiben v_x und v_y die Geschwindigkeit im Robotermittelpunkt O_1 und ω die Winkelgeschwindigkeit des Roboters. Der Geschwindigkeitsvektor der in Gleichung 2.25 formuliert ist, tritt für den Fall des gelenkten Standardrades nicht auf, da sich der Kontaktpunkt entlang der Lenkachse befindet.

$$\overrightarrow{v_{P,12}} = p \cdot \dot{\psi} \cdot \begin{bmatrix} -\sin(\gamma + \psi) \\ \cos(\gamma + \psi) \end{bmatrix} \tag{2.25}$$

Der Geschwindigkeitsvektor in Gleichung 2.26 ensteht durch die Rotation des Rades, ω_1 entspricht der Raddrehzahl und r dem Radius des Rades.

$$\overrightarrow{v_{P,23}} = r \cdot \omega_1 \cdot \begin{bmatrix} \cos(\gamma + \psi) \\ \sin(\gamma + \psi) \end{bmatrix} \tag{2.26}$$

Durch Einsetzen und Umformen der erhaltenen Beziehungen für die Geschwindigkeitsvektoren erhält man Gleichung 2.27.

$$r \cdot \omega_1 \cdot \cos(\gamma + \psi) - p \cdot (\omega + \dot{\psi}) \cdot \sin(\gamma + \psi) = -v_x + \omega \cdot c \cdot \sin\gamma \tag{2.27}$$

Die Aufgabe der inversen Kinematik besteht darin, bei gegebener Drehgeschwindigkeit und bei Kenntnis des Geschwindigkeitsvektors des Roboters, die Raddrehzahlen und die Lenkgeschwindigkeit zu bestimmen. Dazu wird zuerst Gleichung 2.27 für ein gelenktes Standardrad vereinfacht und man erhält Gleichung 2.28. Dies entspricht dem x- bzw y-Term des Geschwindigkeitsvektors des Radmittelpunktes.

$$r \cdot \omega_1 \cdot \sin(\gamma + \psi) + p \cdot (\omega + \dot{\psi}) \cdot \cos(\gamma + \psi) = -v_y - \omega \cdot c \cdot \cos\gamma \tag{2.28}$$

Werden nun die Gleichungen 2.29 und 2.30 quadriert und addiert erhält man Gleichung 2.31. Daraus lässt sich nun bereits die gewünschte Raddrehzahl herausformen, siehe Gleichung 2.32.

$$r \cdot \omega_1 \cdot \cos(\gamma + \psi) = -v_x + \omega \cdot c \cdot \sin\gamma =: -\dot{x}_1 \tag{2.29}$$

$$r \cdot \omega_1 \cdot \sin(\gamma + \psi) = -v_y - \omega \cdot c \cdot \cos\gamma =: -\dot{y}_1 \tag{2.30}$$

$$r^2 \cdot {\omega_1}^2 = \dot{x}_1^2 + \dot{y}_1^2 \tag{2.31}$$

$$\omega_1 = \frac{\pm\sqrt{\dot{x}_1^2 + \dot{y}_1^2}}{r} \tag{2.32}$$

Kombiniert man Gleichung 2.31 mit 2.29 erhält man die Gleichung 2.33. Nun wird die Gleichung 2.29 durch die Gleichung 2.30 dividiert und man erhält Gleichung 2.34. Abgeleitet ergibt sich Gleichung 2.35. Setzt man Gleichung 2.33 in Gleichung 2.35 ein, erhält man die gewünschte Lenkgeschwindigkeit in Gleichung 2.36.

$$\cos^2(\gamma + \psi) = \frac{\dot{x}_1^2}{r^2 \cdot {\omega_1}^2} = \frac{\dot{x}_1^2}{\dot{x}_1^2 + \dot{y}_1^2} \tag{2.33}$$

$$\tan(\gamma + \psi) = \frac{\dot{y}_1}{\dot{x}_1} \tag{2.34}$$

$$\dot{\psi} \cdot \frac{1}{\cos^2(\gamma + \psi)} = \frac{\dot{x}_1 \cdot \ddot{y}_1 - \ddot{x}_1 \cdot \dot{y}_1}{\dot{x}_1^2} \tag{2.35}$$

$$\dot{\psi} = \frac{\dot{x}_1 \cdot \ddot{y}_1 - \ddot{x}_1 \cdot \dot{y}_1}{\dot{x}_1^2 + \dot{y}_1^2} \tag{2.36}$$

2.2.2. Singularitäten

Eine wichtige Eigenheit von Robotern des Typs $(\delta_m = 1, \delta_s = 2)$ sind die auftretenden Singularitäten. Dies sind besondere Situationen in denen die Freiheitsgrade des Roboters eingeschränkt sind, was sich auf die Zunahme der linearen Abhängigkeit der non-slipping Bedingungen zurückführen lässt und zu einer Abnahme im Rang der in Kapitel 2.2.1.1 vorgestellten C-Matrix führt. Gruber & Hofbaur (2016) stellen für diese Art von Roboter, falls er mehr als 3 gelenkte Räder besitzt, zwei Singularitäten vor.

Die erste Singularität ist in Abbildung 2.14 dargestellt. Der Roboter besteht aus drei gelenkten Standardrädern deren Antriebsachsen sich nicht entlang einer Linie befinden, den nur zwei der Räder besitzen dieselbe Achse. Der Momentanpol liegt in diesem Fall entlang der Verbindungslinie von zwei Antriebsachsen. Da jedoch noch ein drittes Rad vorhanden ist, welches die Bewegung einschränkt, sinkt der Rang der C-Matrix nicht (und somit auch keine Veränderung der Freiheitsgrade). Anders ist das beim Fall in Abbildung 2.13 bei dem der Roboter nur 2 Räder besitzt und der Momentanpol entlang der Verbindungslinie der Räder liegt. Hier sinkt der Rang der C-Matrix und der Roboter wird zu dem Typ $(\delta_m = 2, \delta_s = 0)$, was laut Tabelle 2.1 einem Roboter mit Differentialkinematik (fixe Räder) entspricht.

Die zweite von Gruber & Hofbaur (2016) vorgestellte Singularität für einen $(\delta_m = 1, \delta_s = 2)$ - Roboter tritt ein wenn der Momentanpol in der Nähe eines Radmittelpunktes zu liegen kommt, siehe Abbildung 2.15. Hier ist der Idealfall dargestellt, bei dem der Momentanpol genau auf einem Radmittelpunkt liegt. Werden die Lenkwinkel über die Lage des Momentanpols bestimmt ergeben sich hier Probleme, da der Lenkwinkel des Rades auf dem sich der Momentanpol befindet nicht bestimmt werden kann, da dieser

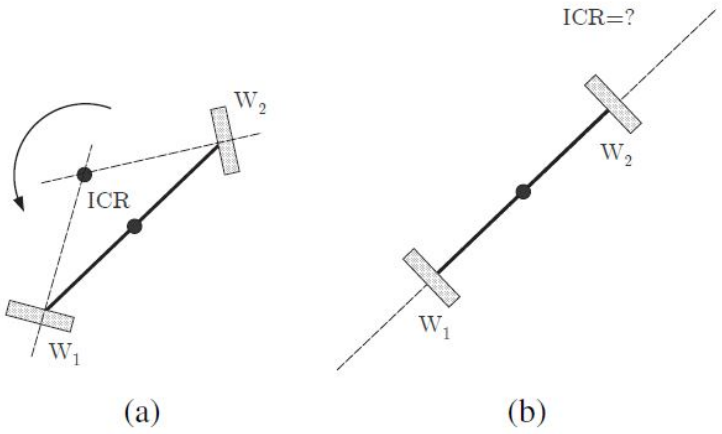


Abbildung 2.13.: Momentanpol bei 2 Standardrädern (Giordano et al., 2009)

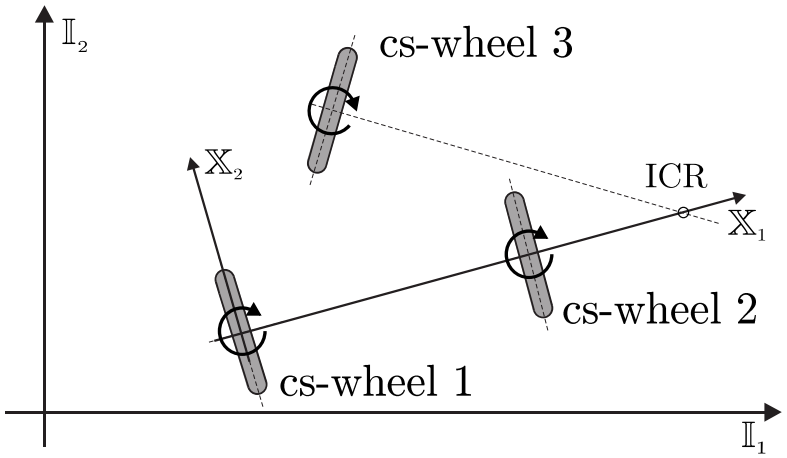


Abbildung 2.14.: Singularität bei drei Standardrädern (Gruber & Hofbaur, 2016, S. 101)

singulär ist (Gruber & Hofbaur, 2016). Das im realen Betrieb größere Problem stellt die Situation dar, wenn der Momentanpol sehr nahe an einem Radmittelpunkt liegt. Dies bedingt hohe geforderte Lenkwinkelgeschwindigkeiten, was große Lenkmomente ergibt, die unter Umständen nicht aufgebracht werden können. Dies führt zu kinemtisch unsauberen Bewegungen (rutschen der Räder) kann aber auch zum Problem für das gesamte System werden. (Giordano et al., 2009)

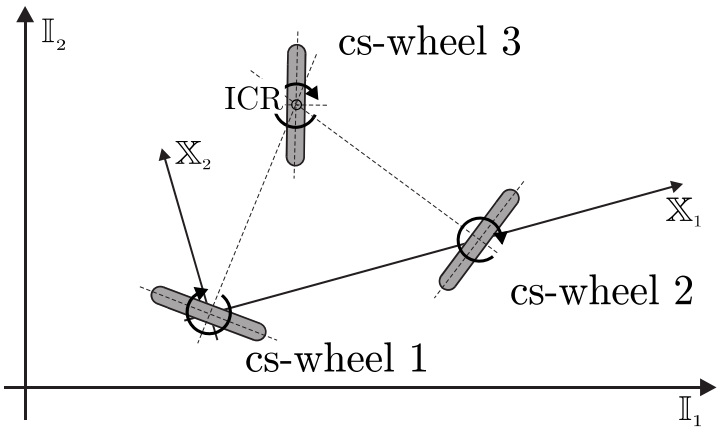


Abbildung 2.15.: Singularität bei drei Standardrädern (Gruber & Hofbaur, 2016, S. 102)

Besonders für Indoor-Roboter stellt diese Singularität eine Herausforderung dar. Dietrich et al. (2011) stellen eine Methode vor, die auf Potentialfeldern basiert, die ‚ähnlich der Methode der Pfadfindung mit Potentialfeldern, die Radmittelpunkte und deren Umgebung als Hindernisse charaktersiert und den Roboter seinen Pfad (auf den Momentanpol bezogen) darum planen lässt. Wie von Giordano et al. (2009) und auch Connette (2013) erwähnt, ergibt sich die einfachste Vermeidung dieser Singularität durch eine Berücksichtigung in der Planung des Pfades des Roboters.

2.2.3. Separate Betrachtung von Rekonfiguration und ebener Bewegung

Unter Rekonfiguration des Fahrwerkes eines Roboters wird im Allgemeinen das Lenken verstanden. Es stellt sich nun die Frage ob die Regelaufgaben, welche zum Einen das Lenken des Roboters und zum Anderen der Antrieb des Roboters sind, getrennt voneinander betrachtet werden können. Es kann gezeigt werden (Connette, 2013), dass eine getrennte Betrachtung in erster Näherung möglich ist. Ohne genauer auf die Herleitung einzugehen, wird zuerst Gleichung 2.37 betrachtet. Hier wird die Energie des Roboters während seiner Bewegung beschrieben. Der Ursprung des Bezugsystems fällt dabei in den Roboterschwerpunkt. Mit I wird das Trägheitsmoment um die z-Achse beschrieben, ω ist die Winkelgeschwindigkeit, m ist die Masse und v die Geschwindigkeit des Roboters. Die in der Rotation der Räder des Roboters gespeicherte Energie wird dabei vernachlässigt. Es kann nun durch Umformungen gezeigt werden, dass diese kinetische Energie einzig und allein von einer Variable abhängt, welche die ebene Bewegung charakterisiert (siehe Kapitel 2.2.1.3).

Connette (2013) liefert auch eine physikalische Interpretation diesbezüglich. Unter Einhaltung der nicht-holonomen Bindungen im Rad-Straße-Kontakt wirkt die durch

das Lenken aufgebrachte Kraft immer orthogonal zur Fahrrichtung und bewirkt somit eine Richtungsänderung der Geschwindigkeit. Das Lenken des Fahrzeuges benötigt idealisiert betrachtet keine Energie, es wird keine Arbeit verrichtet (da es idealisiert betrachtet keine Verluste im Rad-Straße-Kontakt gibt). Bei Robotern mit harten Vollgummirädern, welche sich ausreichend weit vom Grenzbereich (immer größer werdende Walkarbeit des Reifens) entfernt befinden, gilt diese Annahme noch als ausreichend erfüllt.

$$E_{kin,b} = \frac{1}{2} \cdot m \cdot v^2 + \frac{1}{2} \cdot I \cdot \omega^2 \tag{2.37}$$

Auf Ebene der gelenkten Räder kann ebenfalls eine energetische Betrachtung erfolgen, siehe Gleichung 2.38. Dabei stehen $\underline{\underline{I}}_d$ und $\underline{\underline{I}}_s$ für die Trägheitsmomente um die Antriebs- bzw. um die Lenkachse. $\dot{\varphi}_d$ und $\dot{\varphi}_s$ stehen für die Drehzahl des Rades bzw. für die Lenkwinkelgeschwindigkeit. Es kann nun, ebenfalls nach Connette (2013) gezeigt werden, dass diese Energie nur von den Freiheitsgraden für die Rekonfiguration des Fahrwerkes (Lenken) abhängig ist (siehe Kapitel 2.2.1.3). Die gesamte Energie setzt sich nun aus beiden Teilen zusammen, siehe Gleichung 2.39. Somit kann eine getrennte Betrachtung bezüglich der Regelung von Antrieb und Lenken des Roboters erfolgen.

$$E_{kin,q} = \frac{1}{2} \cdot \overrightarrow{\dot{\varphi}_d^T} \cdot \underline{\underline{I}}_d \cdot \overrightarrow{\dot{\varphi}_d} + \frac{1}{2} \cdot \overrightarrow{\dot{\varphi}_s^T} \cdot \underline{\underline{I}}_s \cdot \overrightarrow{\dot{\varphi}_s} \tag{2.38}$$

$$E = E_{kin,b} + E_{kin,q} \tag{2.39}$$

2.3. Verfolgung einer Trajektorie (Path-Tracking)

Eine der zentralen Aufgaben von mobilen Robotern besteht meist in dem Befahren einer vorgegebenen Trajektorie, siehe Abbildung 2.16. Aufgabe des Roboters ist es, von einem Startpunkt ausgehend, einen bestimmten Weg bis zum Zielpunkt zu nehmen. Soll dieser vorgebene Pfad zusätzlich innerhalb einer bestimmten Zeit erfolgen spricht man von einer Trajektorie (Corke, 2011). Ein Roboter kann in der Lage sein einem Pfad zu folgen, jedoch auf Grund unzureichender Motorisierung ist es ihm nicht möglich der vorgegeneben Trajektorie zu folgen. Im Umkehrschluss bedeutet dies, dass bereits bei der Definition einer Trajektorie die Möglichkeiten des Roboters hinsichtlich Beschleunigungsverhalten etc. bekannt sein müssen.

Eine zentrale Aufgabe bei dem Folgen einer Trajektorie besteht in der Strategie wie eine auftretende Abweichung versucht wird zu korrigieren. Grundsätzlich kann eine Unterscheidung zwischen geometrische, kinematische und dynamische Methoden getroffen werden. Der einfachste geometrische Algorithmus ist der Follow-the-carrot nach Lundgren (2003). Hier wird, ausgegangen von einem Fahrzeug mit Ackermann-Kinematik an der Vorderachse, welches auf ein Einspurmodell reduziert wurde, der Winkel der Lenkstellung als lineare Funktion zwischen Fahrzeugausrichtung und Ziel berechnet, siehe Gleichung 2.40 und Abbildung 2.17. Der gewählte Punkt entlang der Sollspur entscheidet zum Einen über die Genauigkeit zum Anderen über die Stabilität und hängt vor allem von der Geschwindigkeit des Roboters ab.

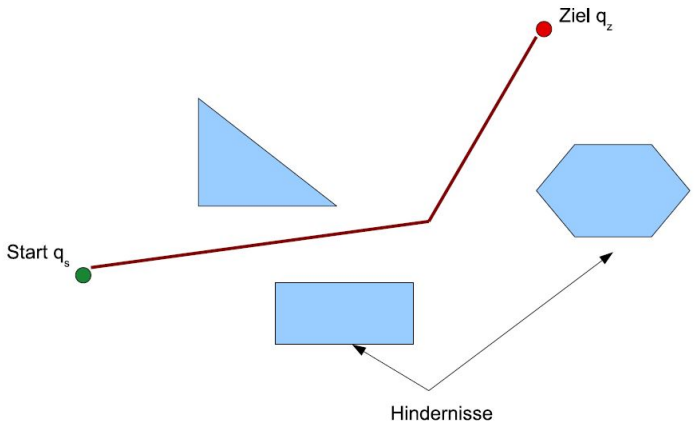


Abbildung 2.16.: Möglicher Verlauf einer Trajektorie für einen mobilen Roboter (Oubbati, 2009, S.52)

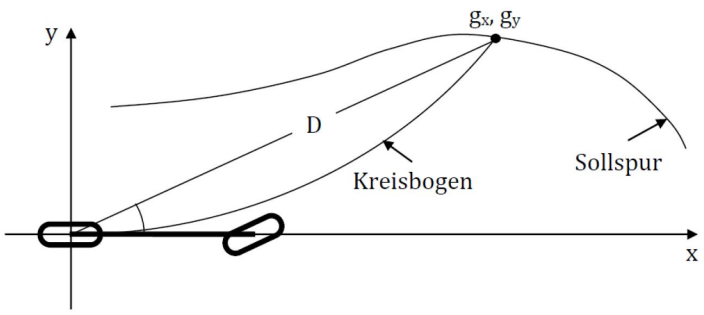


Abbildung 2.17.: Geometrische Überlegungen bezüglich Path-Tracking (Nikolov, 2011, S.11)

$$y_{ftc} = K_p \cdot \arctan\left(\frac{g_y}{g_x}\right) \tag{2.40}$$

Der Pure-Pursuit Algorithmus von Coulter (1992) verbindet das Fahrzeug und den Zielpunkt am Sollpfad mit einem Kreisbogen. Aus der errechneten Krümmung mittels Gleichung 2.41 lässt sich der zu realisierende Lenkwinkel ableiten.

$$\gamma = \frac{2 \cdot g_y}{D^2} \tag{2.41}$$

Beide Methoden haben, als rein geometrische Methoden, den Nachteil den Verlauf der Trajektorie zwischen dem Fahrzeug und dem Zielpunkt (g_x, g_y) nicht zu berücksichtigen. Dies führt unter anderem zu einem Schneiden von Kurven. Kinematische Methoden befassen sich hingegen mit den Fahrzeugbewegungen ohne Ursachenbetrachtung der auftretenden Kräfte. Die sogenannte Stanley-Methode nach Hoffmann et al. (2007) beschreibt eine solche Variante. Hier wird gefordert, dass die Lenkwinkelstellung immer ident mit der Richtung des Sollpfades ist. Zusätzlich wird noch ein zweiter Term addiert, welcher abhängig von der Geschwindigkeit ist. Durch den Fokus auf die Stellung der Vorderräder, sowie der Einbeziehung der Geschwindigkeit, ist diese Methode wesentlich genauer, kann jedoch zu Überschwingen führen, da keine Ziele vor dem Fahrzeug berücksichtigt werden (Snider, 2009). Bei dynamischen Methoden werden die auf das Fahrzeug wirkenden Kräfte betrachtet.

In Hoffmann et al. (2007) ist auch eine dynamische Variante der Stanley-Methode beschrieben. Dynamische Methoden kommen dann zum Einsatz wenn sich das betrachtete Fahrzeug in einem nicht-linearen Bereich bewegt und Effekte wie Reifenschlupf etc. nicht mehr vernachlässigbar sind. Das betrachtete Fahrzeug in Hoffmann et al. (2007) bewegte sich fast ausschließlich in unwegsamen Gelände und das Verwenden eines dynamischen Modells unumgänglich.

Bei jeder der aufgezählten Methoden muss jedoch, von einer meist stetig definierten Trajektorie ausgehend, eine lineare Segmentierung dieser erfolgen, siehe Abbildung 2.18, da man sich bei Anwendung der Path-Tracking Methoden immer auf einen bestimmten Punkt entlang dieser Trajektorie bezieht.

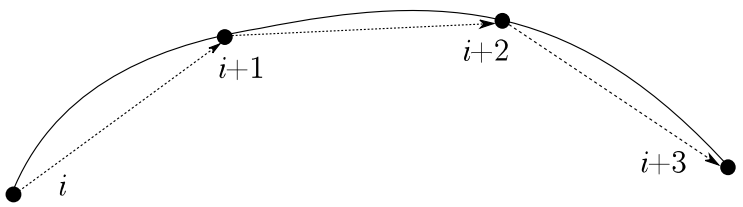


Abbildung 2.18.: Lineare Segmentierung einer Trajektorie (Andersen et al., 2016)

3. Fahrdynamische Voruntersuchung

In diesem Kapitel wird die durchgeführte fahrdynamische Voruntersuchung des Roboters behandelt. Begonnen wird im Unterkapitel 3.1 mit der Erklärung und dem Aufbau des verwendeten Mehrkörpersimulationsmodells für die fahrdynamische Voruntersuchung. Mit Hilfe dieses Modells, werden im Abschnitt 3.2 ausgewählte Fahrmanöver durchgeführt, deren Ergebnisse als Grundlage für die weitere Betrachtung dienen und im Unterkapitel 3.3 diskutiert werden. Als Abschluss dieses Kapitels und als Überleitung für das Kapitel 4, in welchem die Gesamtsimulation vorgestellt wird, dient das Unterkapitel 3.4, welches die umgesetzte Co-Simulation behandelt.

3.1. Erklärung und Aufbau des verwendeten Modells

In diesem Kapitel wird das verwendete Mehrkörpersimulations-Modell (MKS-Modell) des Roboters genauer erklärt. Grundsätzlich handelt es sich bei Mehrkörpersystemen um ein System aus jeweils starren Teilkörpern, welche untereinander und mit der Umgebung durch masselose Kraftelemente und Gelenke miteinander verbunden sind. Es erfolgt eine physikalische Diskretiserung durch Trennung der materiellen Eigenschaften Dichte, Elastizität und Viskosität. Dies erlaubt eine recheneffiziente Darstellung der Längs- und Querdynamik von Fahrzeugen in der Gesamtsimulation, sowie auch von einzelnen Teilsystemen, wie etwa der Radaufhängung. Für die Modellierung eines Gesamtfahrzeugsystems müssen drei Teilbereich abgedeckt sein. Die erste Komponente ist das Chassis, hier werden die Reaktionskräfte aus der Fahrbahn übertragen und es werden die Gewichts- Trägheits- und Antriebs- bzw. Bremskräfte aufgenommen. Die zweite unverzichtbare Komponente stellt das Antriebs- und Bremssystem dar. Das dritte und letzte Teilsystem ist das Führungsystem, darunter wird die Methode verstanden mit der die Kurshaltung des Fahrzeuges erfolgt. In lateraler Richtung erfolgt dies durch die Lenkung. (Hirschberg & Waser, 2016)

Das zu untersuchende Fahrzeug wird in diesem Fall mit dem MKS-Programmpaket ADAMS/Car modelliert, welches auf die MKS-Simulation von Fahrzeugen aller Art spezialisiert ist. In Abbildung 3.1 ist ein Überblick über die Datenhierarchie in ADAMS dargestellt. Mit *Template designs* lassen sich Vorlagen erstellen, beispielsweise für eine spezielle Art von Rädaufhängung. Mit sogenannten Subsystemen (*Subsystem data*) erfolgt dann die gewünschte Parametrierung dieser Vorlage. So kann beispielsweise ein und diesselbe Vorlage für unterschiedlich große Fahrzeuge verwendetet werden. Es müssen nur die Parameter geändert werden. Im Gesamtmodell (*Vehicle model*) werden dann die einzelnen Subsysteme kinematisch korrekt miteinander verbunden.

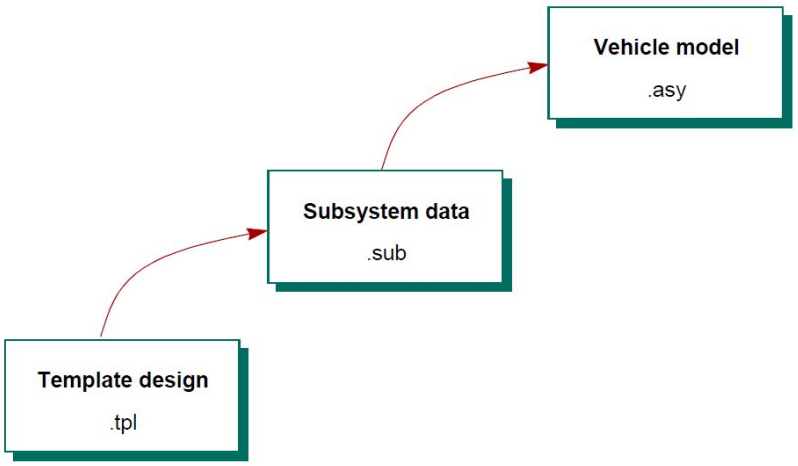


Abbildung 3.1.: Datenhierarchie in ADAMS/Car® (ADAMS/Car®, 2001, S.30)

Es wurden zwei unterschiedliche Modelle des Roboters erstellt. Das für die fahrdynamische Voruntersuchung verwendete Modell besitzt nicht lenkbare Hinterräder und eine Ackermann-Kinematik der Vorderräder und entspricht somit einem typischen PKW. Währen der Kurvenfahrt befindet sich der Momentanpol somit immer auf Höhe der Hinterachse. Dieses Modell, welches augenscheinlich nicht der gewünschten Kinematik des Roboters entspricht, eignet sich jedoch gut um gewisse Manöver zu testen. Mittels ADAMS/Car lassen sich diese Manöver beispielsweise unter der Vorgabe eines gewünschten Lenkwinkels (würde dem Lenkwinkel eines Lenkrades bei einem PKW entsprechen) ausführen.

In Abbildung 3.2 ist das MKS-Modell zu sehen, Abbildung 3.3 zeigt die verwendeten Subsysteme. Wie bereits zuvor erwähnt, müssen alle für ein Fahrzeug notwendigen Funktionen abgebildet werden. Das erste betrachtete Subsystem *active_front_axle* beinhaltet dabei die beiden Einzelradaufhängungen an der Vorderachse. Für ein besseres Verständnis sei auf Abbildung 3.4 verwiesen.

Die nach ihrem Erfinder benannte Mc-Pherson Radaufhängung besteht grundsätzlich aus 5 kinematisch relevanten Komponenten: Dem Radträger und Dreieckslenker, der Spurstange, der Zahn- bzw. Lenkstange und dem Federbein, in der Abbildung als Dämpferrohr und Dämpferstange dargestellt.

Kinematisch betrachtet besitzt die Aufhängung zwei Freiheitsgrade, einen zum Lenken und einen zum Einfedern (dessen Weg ist begrenzt). Ein dritter Freiheitsgrad tritt bei Ausführung der Spurstange mit zwei Kugelgelenken auf. Hier ist eine Drehung der Stange möglich, ohne andere Körper zu beeinflussen, man spricht von einem isolierten Freiheitsgrad (Schramm et al., 2013). Wie in der Einleitung des Kapitels erwähnt, ist ein Ziel dieser Voruntersuchung die Beantwortung der Frage ob für das zu untersuchende Fahrzeug eine Federung und eine Dämpfung notwendig ist. Für diesen Zweck wird die vorgestellte Radaufhängung um den Freiheitsgrad des Einfederns beraubt. Dies bedeutet im Umkehrschluss, dass die auftretenden Kräfte am Rad nun direkt in den Aufbau eingeleitet werden, mit der Ausnahme einer geringen Feder- und Dämpferwirkung der Reifen.

Das Subsystem für die Hinterachse (*active_rear_axle*) ist eine vereinfachte Version der

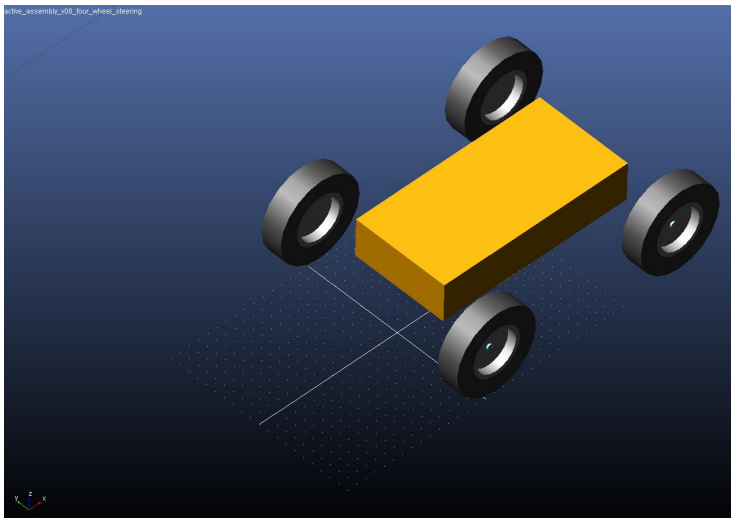


Abbildung 3.2.: ADAMS/Car Modell des Roboters

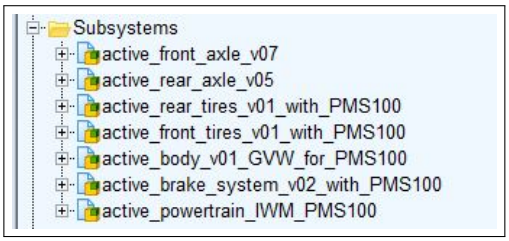


Abbildung 3.3.: Strukturbaum des ADAMS/Car Modell

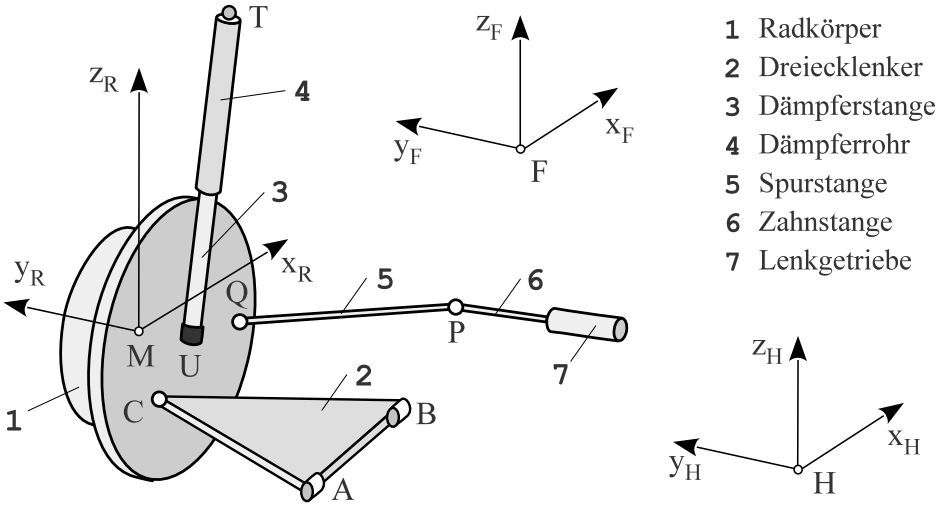


Abbildung 3.4.: Modellkörper einer McPherson-Radaufhängung (Rill & Schaeffer, 2014)

Vorderachse, ein Lenken ist nicht möglich, die Anzahl der Freiheitsgrade verringert sich somit auf null, da auch an der Hinterachse keine Federung und Dämpfung vorgesehen ist. Die beiden Subsystems *active_rear_tires* und *active_front_tires* beinhalten die verwendeten Reifenmodelle (vorne und hinten ident). Es wurde ein empirisch-mathematisches Modell von Pacejka & Besselink (1997) verwendet, welches bewusst auf eine physikalische Modellierung verzichtet und mathematische Approximationen trifft, welche für eine Simulation sehr effizient handhabbar sind (Hirschberg & Waser, 2016). Das in ADAMS implementierte Reifenmodell ist ein herunterskaliertes Modell eines 17 Zoll Reifens auf die gewünschte Größe von 10 Zoll. Dies ist zwar sehr einfach durchführbar, man verliert jedoch an Genauigkeit, da bei einem mathematischen Reifenmodell a-priori Kenntnisse des betrachteten Reifentyps vorliegen müssen (Hirschberg & Waser, 2016).

Das Chassis wird als einfacher starrer Körper modelliert und entspricht dem Subsystem *active_body*. Das Subsystem *active_brake_system* entspricht einem hydraulischen Bremssystem mit Scheibenbremsen. Das letzte Subsystem *active_powertrain* enthält die verwendeten vier Synchronmotoren, welche als Radnabenmotoren ausgeführt sind und sich somit innerhalb der Felge jedes Rades befinden. Der Drehmomentverlauf dieser Motoren ist in Abbildung 3.6 zu sehen, dabei wird zusätzlich noch ein Planetengetriebe verwendet, um das Drehmoment am Rad um den Faktor 7 zu erhöhen. Das Gewicht eines Motors beträgt 7,2 kg, befindet sich direkt im Rad und erhöht somit dessen Gesamtmasse deutlich. Besonders im Hinblick auf den gewünschten Verzicht einer Federung, sowie einer Dämpfung stellt dies einen negativen Aspekt dar. In Abbildung 3.5 sind einige Daten wie Gesamtmasse und die Trägheitsmomente des Roboters zu sehen. Abbildung 3.7 zeigt nochmals schematisch den Aufbau und die groben Abmessungen des zu untersuchenden Fahrzeuges.

```
Modell welches mit Masse beaufschlagt wird:  .active_assembly_v08_four_wheel_steering
Bezogen auf das globale Koordinatensystem:
  Masse               : 296.09625239 kg
  Schwerpunkt         :
      Position        : 502.0974240411, 0.0, 253.5854157139 (mm, mm, mm)
      Orientierung    : 90.0, 7.086129942E-002, 270.0 (deg)
  Trägheitstensor     :
      IXX             : 1.34020142E+008 kg-mm**2
      IYY             : 3.0891953695E+008 kg-mm**2
      IZZ             : 3.0266363517E+008 kg-mm**2
      IXY             : 0.0 kg-mm**2
      IZX             : 3.7560531002E+007 kg-mm**2
      IYZ             : 0.0 kg-mm**2
```

Abbildung 3.5.: Robotermodell in ADAMS/Car mit Masse beaufschlagt

3.2. Ergebnisse der fahrdynamischen Voruntersuchung

Das zuvor beschriebe ADAMS-Modell des Roboters mit Ackermann-Kinematik an der Vorderachse, wird nun einigen fahrdynamischen Manövern unterworfen, mit dem Ziel die Grenzen des Fahrzeuges in verschiedenen Bereichen auszuloten und wertvolle Informationen für den weiteren Entwicklungsprozess zu erlangen. Es folgt jeweils eine Erklärung des Manövers sowie deren Ergebnisse, am Ende erfolgt eine zusammenfassende Diskussion.

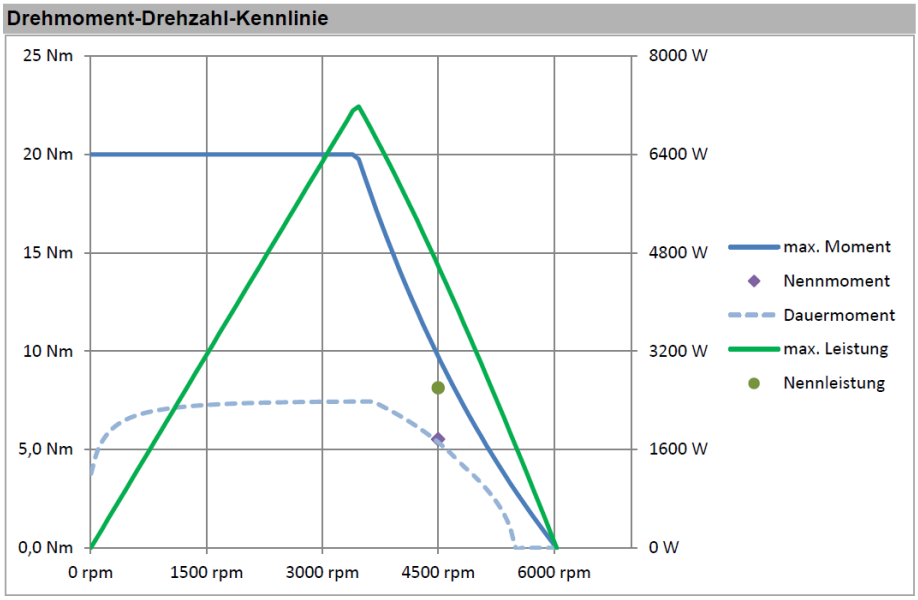


Abbildung 3.6.: Drehmomentenverlauf des gewählten Radmotors für den Antrieb (Heinzmann, 2017)

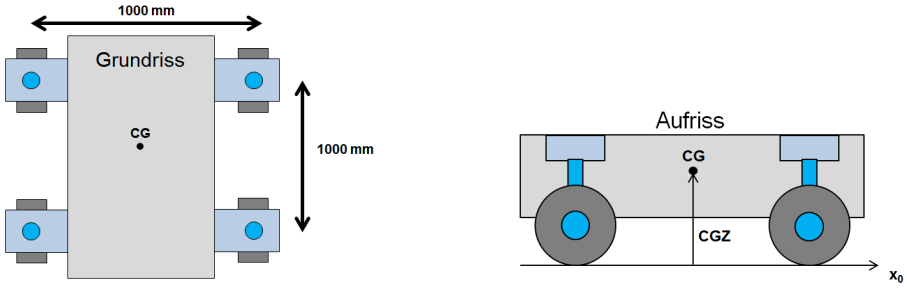


Abbildung 3.7.: Abmessungen des Roboters

3.2.1. Beschleunigung auf Höchstgeschwindigkeit

In diesem Manöver wird das Fahrzeug mit größtmöglicher Beschleunigung bis zur angestrebten Höchstgeschwindigkeit angetrieben. Wie in Abbildung 3.8 zu sehen ist, kann mit maximalem Moment aller Radmotoren (Abbildung 3.9 zeigt den Verlauf des Drehmomentes eines Radmotors) eine Beschleunigung von knapp 0,85 g erreicht werden, siehe Abbildung 3.10. Der Schlupf (prozentuelle Angabe wie weit sich die Drehzahl des angetriebene Rades von einem frei rollenden Rad unterscheidet) zeigt für die Räder an der Vorderachse (Abbildung 3.11) etwa 10 Prozent, diese entspricht in etwa dem Bereich in dem die maximale Antriebskraft übertragen werden kann (Hirschberg & Waser, 2016). Durch die Gewichtsverlagerung bei der Beschleunigung ist der auftretende Schlupf an den Hinterrädern (Abbildung 3.12) deutlich geringer.

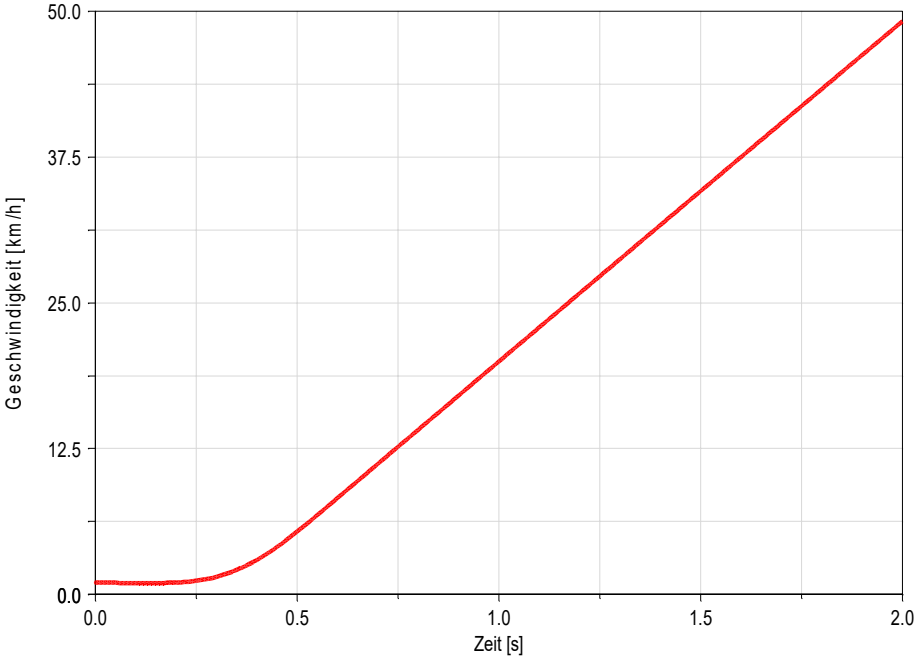


Abbildung 3.8.: Geschwindigkeitsverlauf bei maximaler Beschleunigung bis v_{max}

3.2.2. Befahren einer unebenen Straße mit Höchstgeschwindigkeit

Dieses Manöver lässt das Fahrzeug über eine mittelmäßig unebene Straße fahren. Diese Straße ist als C-Straße klassifiziert laut ISO 8608:2016. Dabei werden Straßen mit einem Messfahrzeug vermessen und die auftretende Vertikalbeschleunigung gemessen. Aus dieser Information wird der Versuch unternommen Straßen zu klassifizieren und mathematisch zu beschreiben, da die Fahrbahnbeschaffenheit grundsätzlich zufälliger Natur ist, insbesondere ihre Unebenheit (Hirschberg & Waser, 2016). Der Roboter befährt diese Straße mit seiner ausgewiesenen Höchstgeschwindigkeit von 50 $\frac{km}{h}$. Es wurden zwei Varianten untersucht, zum Einen mit der auch im weiteren verwendeten Standardmasse, von etwa 300 kg (siehe Abbildung 3.5) und zum Anderen mit der Hälfte (150 kg) des ursprünglichen Gewichtes. In Abbildung 3.13 ist zu sehen, dass mit der Standardmasse ein Abheben des Rades (und somit ein zu null werden der

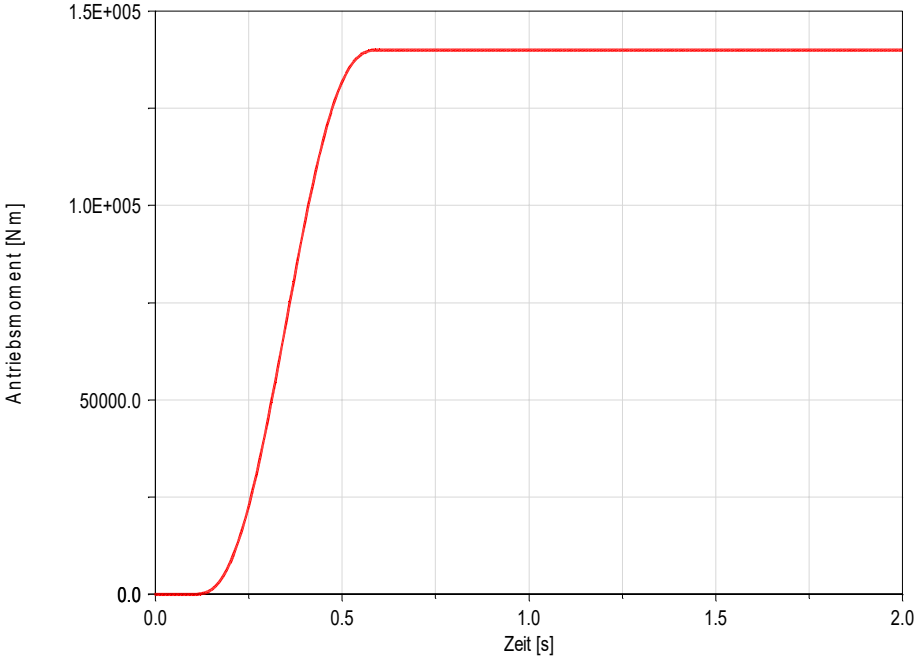


Abbildung 3.9.: Verlauf des Antriebsdrehmomentes für den gesamten Roboter

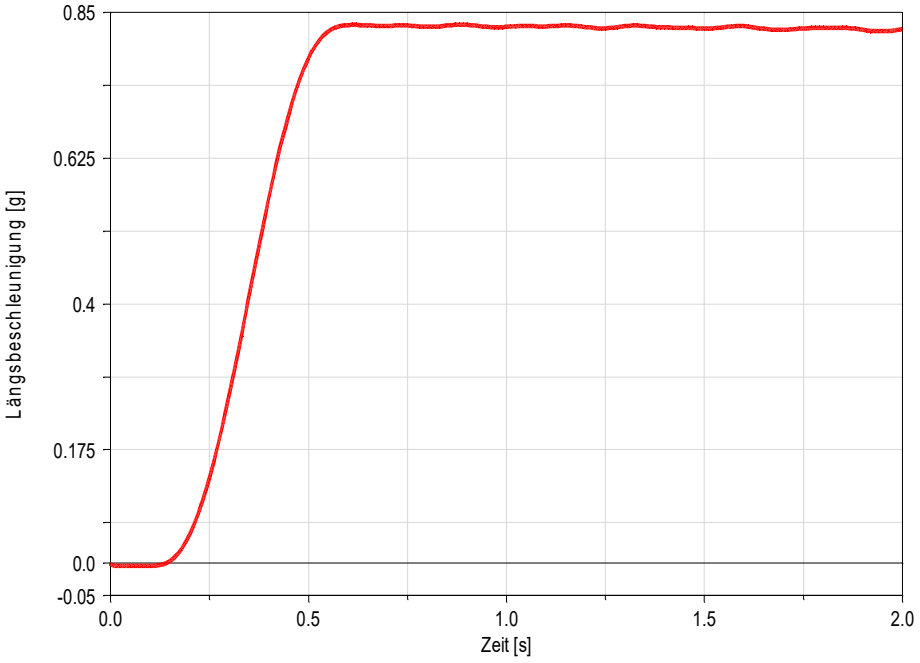


Abbildung 3.10.: Verlauf der Längsbeschleunigung während der Beschleunigung

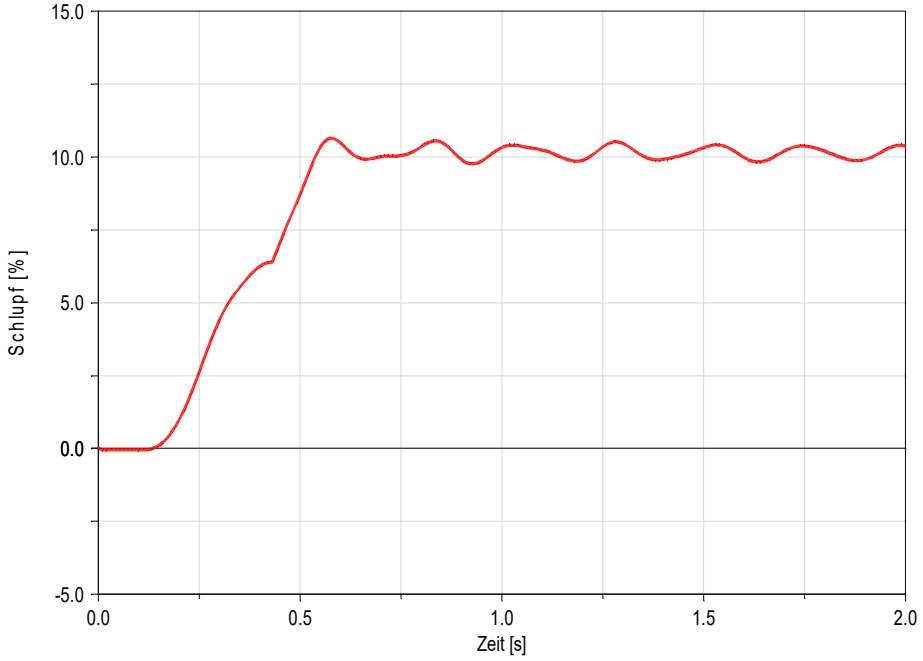


Abbildung 3.11.: Verlauf des Schlupfes an den beiden Vorderrädern während der Beschleunigung

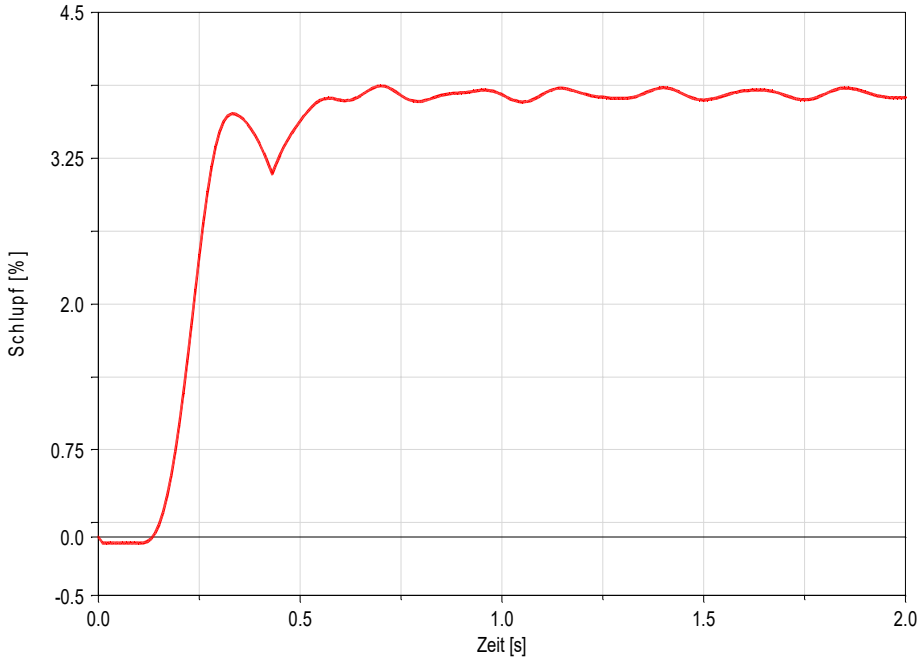


Abbildung 3.12.: Verlauf des Schlupfes an den beiden Hinterrädern während der Beschleunigung

Radaufstandskraft) vermieden werden kann und eine ausreichende Kraftreserve am Rad vorherrscht um für etwaige Kurvenfahrten Seitenkräfte aufzunehmen. Bei der zweiten Variante, Abbildung 3.14 , sieht man jedoch ein kurzes Abheben des Rades vom Untergrund. Hier ist ein Limit erreicht, es können keine Seitenkräfte mehr aufgenommen werden. Die Aufstandskräfte der restlichen Räder sind in Anhang A zu finden.

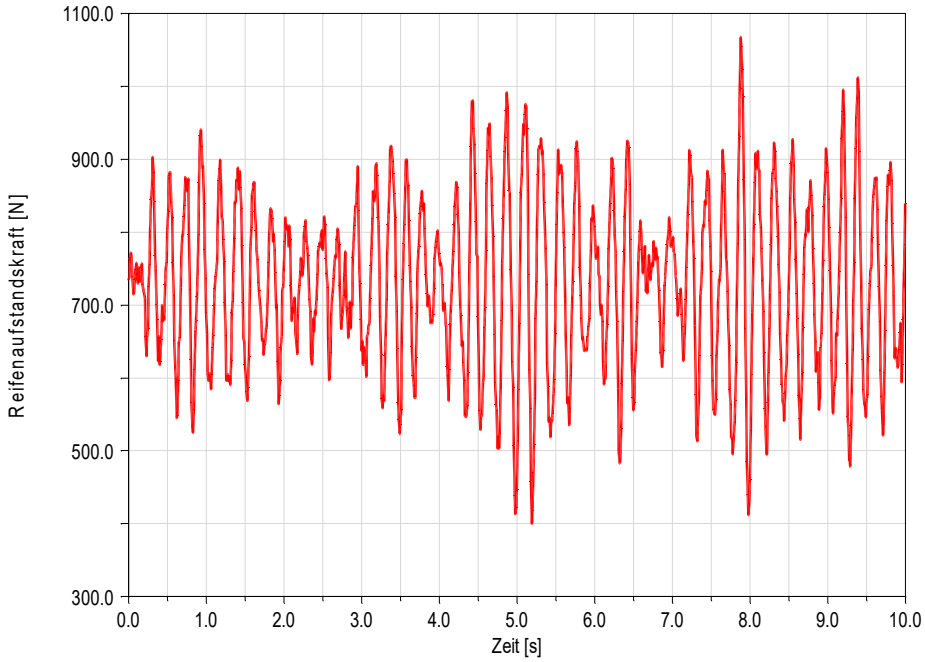


Abbildung 3.13.: Verlauf der Radaufstandskraft vorne links

3.2.3. Kurvenfahrt mit konstantem Lenkwinkel

Bei diesem Manöver wird das Fahrzeug mit der Höchstgeschwindigkeit von 50 $\frac{km}{h}$ mit einem konstanten Lenkwinkel von 2 Grad bewegt. Bei idealer Betrachtung sollte sich eine Kreisbahn ergeben, Abbildung 3.19 zeigt eine doch deutliche Abweichung davon, dies deutet auf einen recht großen Schräglaufwinkel an den Rädern hin. In Abbildung 3.15 ist die auftretende Querbeschleunigung zu sehen, diese schwingt sich langsam auf, es scheint ein sicherer Betriebsbereich verlassen worden zu sein. Weitere Auswertungen mit konstantem Lenkwinkel bei geringerer Geschwindigkeit ergeben eine Grenze von 0,5 g welche zu keinem Aufschwingen führt. Das Aufschwingen ist auf den internen Regler in ADAMS zurückzuführen, welcher die Vorgabe eines konstanten Lenkwinkel, sowie einer konstanten Geschwindigkeit weiter versucht einzuhalten obwohl die Grenzen des Fahrzeuges erreicht sind. Es ist jedoch noch kein Abheben der kurveninneren Räder zu befürchten, siehe Abbildung 3.18. Den Verlauf der Geschwindigkeit, sowie des befahrenen Radius zeigen Abbildung 3.16 und 3.20.

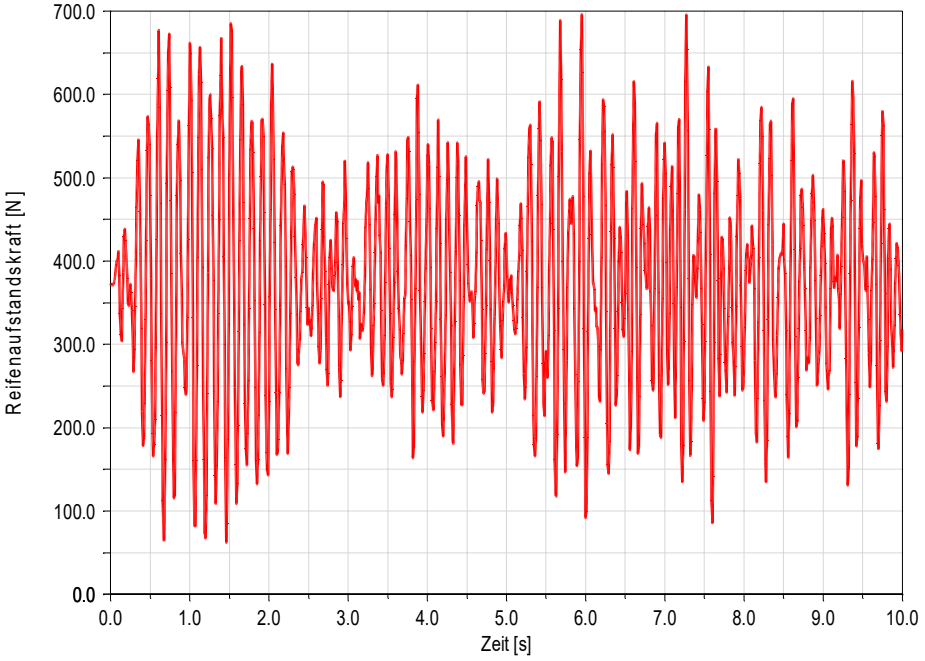


Abbildung 3.14.: Verlauf der Radaufstandskraft hinten rechts bei halbem Gewicht

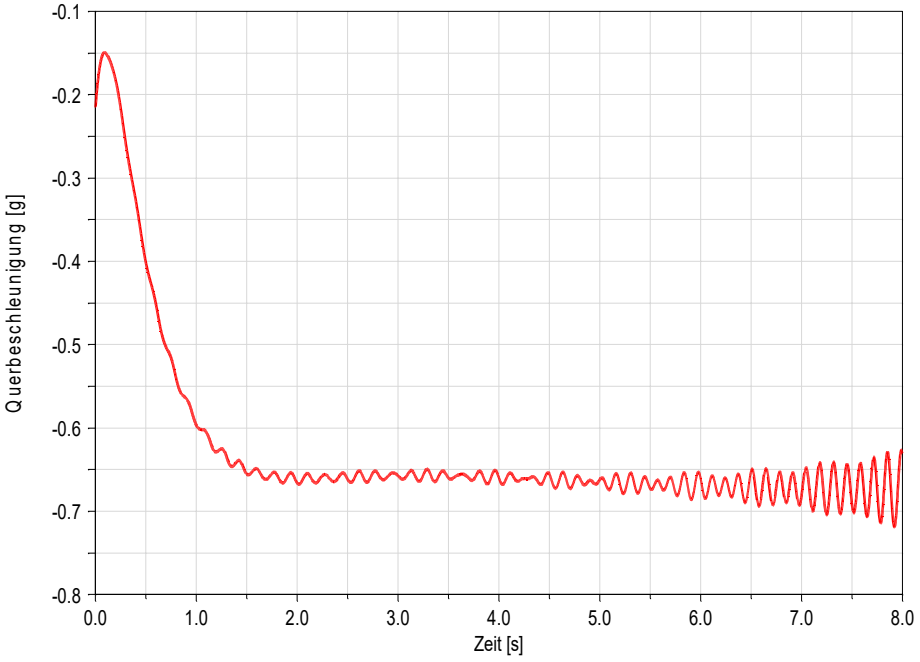


Abbildung 3.15.: Verlauf der Querbeschleunigung während der Kurvenfahrt

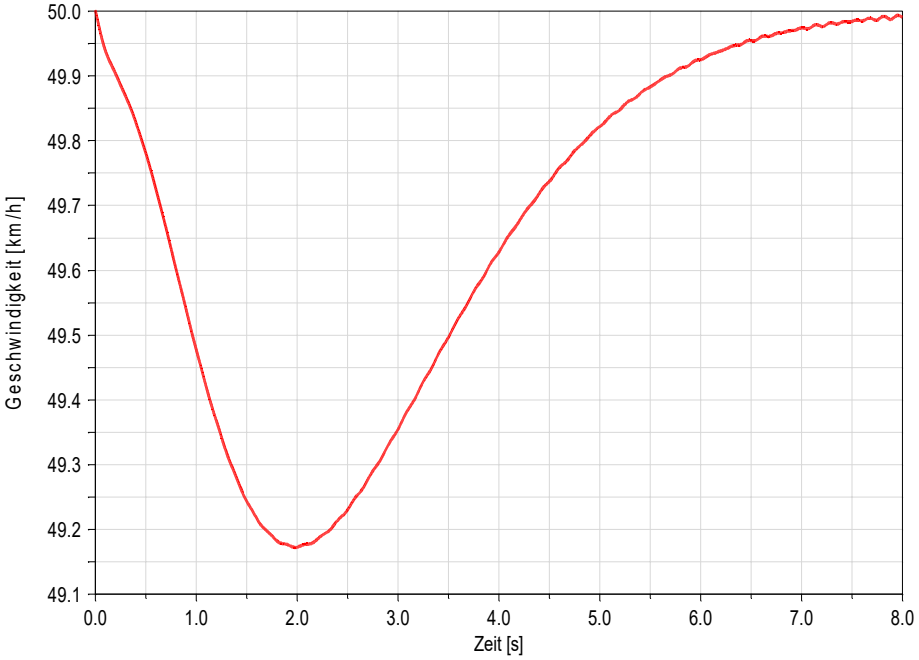


Abbildung 3.16.: Verlauf der Geschwindigkeit während der Kurvenfahrt

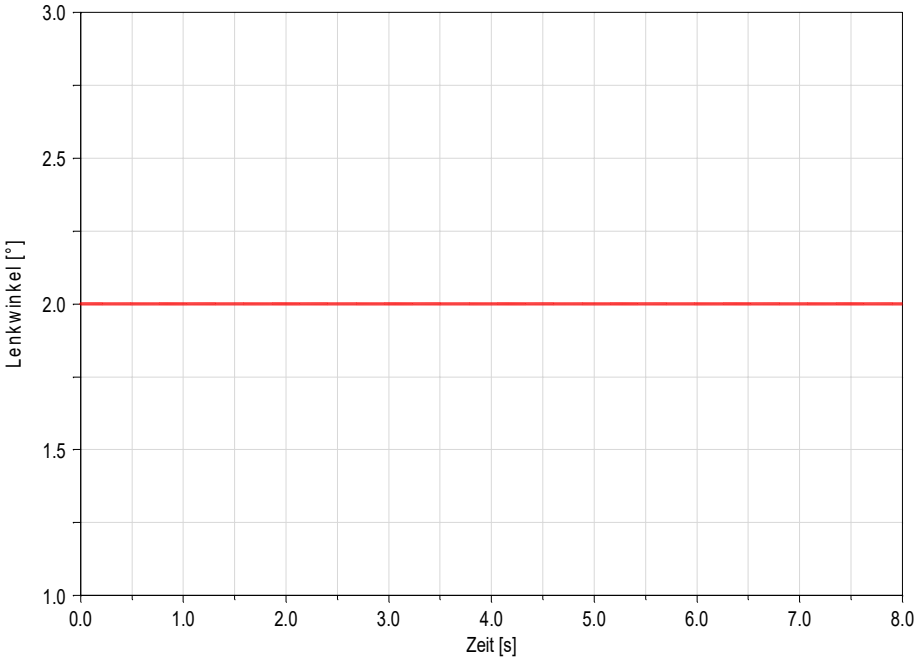


Abbildung 3.17.: Verlauf des Lenkwinkels während der Kurvenfahrt

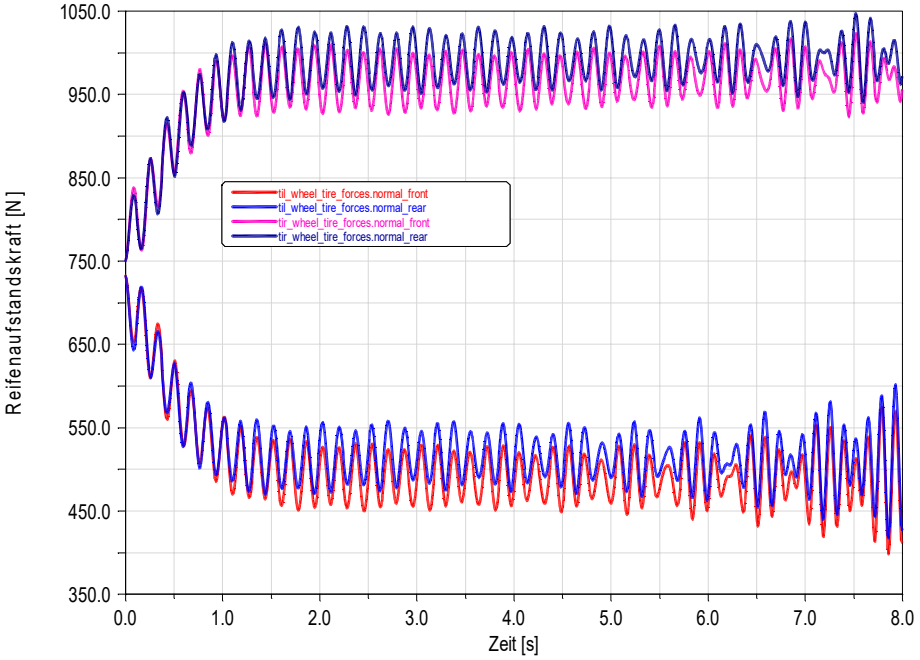


Abbildung 3.18.: Verlauf der Radaufstandskräfte während der Kurvenfahrt

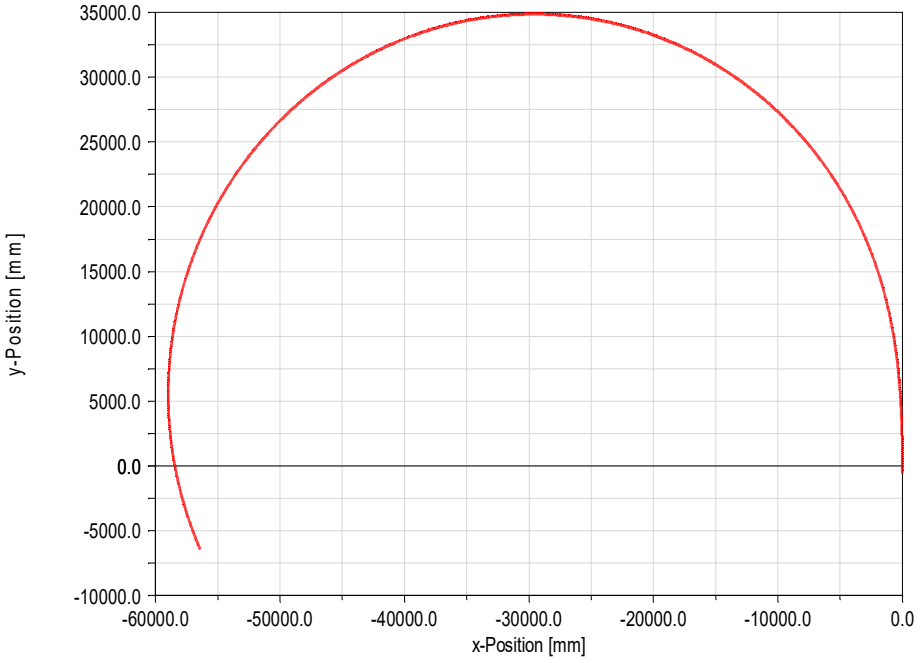


Abbildung 3.19.: Verlauf der Roboterposition während der Kurvenfahrt

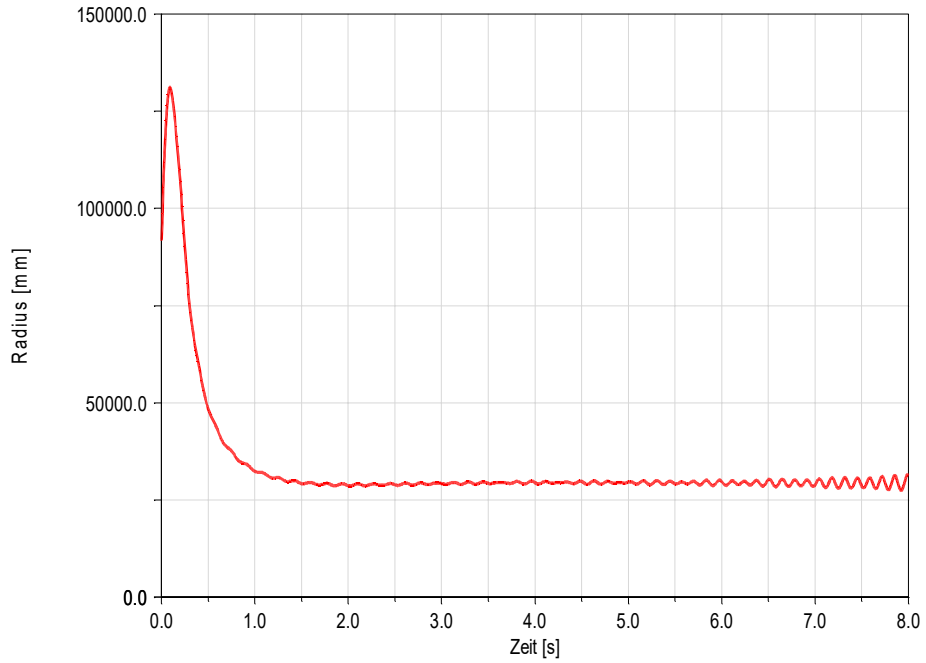


Abbildung 3.20.: Verlauf des gefahrenen Radius während der Kurvenfahrt

3.2.4. Slalomfahrt bei niedriger Geschwindigkeit

Das letzte betrachtete Manöver ist eine Slalomfahrt mit sehr niedriger Geschwindigkeit. Es wird ein Sinusförmiger-Verlauf des Lenkwinkels vorgegeben, mit einer Periodendauer von 6 Sekunden und einer Amplitude von 25 Grad Lenkwinkel. Die Geschwindigkeit soll 1 $\frac{km}{h}$ betragen. Siehe dazu Abbildung 3.23 und Abbildung 3.24. Das daraus resultierende Lenkmoment zeigt Abbildung 3.21. Es werden recht hohe Werte für das Lenkmoment gefordert, was aus den großen Änderungsgeschwindigkeiten des Lenkwinkels resultiert. Der Verlauf der Roboterposition der sich aus diesem Manöver ergibt ist in Abbildung 3.22 ersichtlich.

3.3. Diskussion der fahrdynamischen Voruntersuchung

Auf Grund der durchgeführten Manöver an dem zu untersuchenden Fahrzeug, lassen sich folgende Schlüsse ziehen: Das befahren von unebenen Straße hat gezeigt, dass, zumindest mit dem Standardgewicht, ein sicheres Befahren auch ohne zusätzliche Federung, sowie Dämpfung, noch möglich ist. Da keine Personen als Fahrgäste zu erwarten sind, können die auftretenden Komforteinbußen vernachlässig werden. In der Kurvenfahrt ist, solange man unter einer Querbeschleunigung von 0,5 g bleibt, keine Instabiliät zu erwarten. Durch den niedrigen Schwerpunkt des Roboters, verlieren bei wesentlichem Überschreiten dieser Grenze zuerst die Reifen die Haftung, eine Umkippen des Roboters kann somit ausgeschlossen werden. An dieser Stelle sei auch erwähnt, dass gerade im Falle eines Fehlens von Federung und Dämpfung, der Luftdruck eine noch größere Rolle spielt. Die durchgeführten Simulation erfolgten alle mit einem Luftdruck von 2 bar, ein verringern dieses Wertes würde die Dämpferwirkung

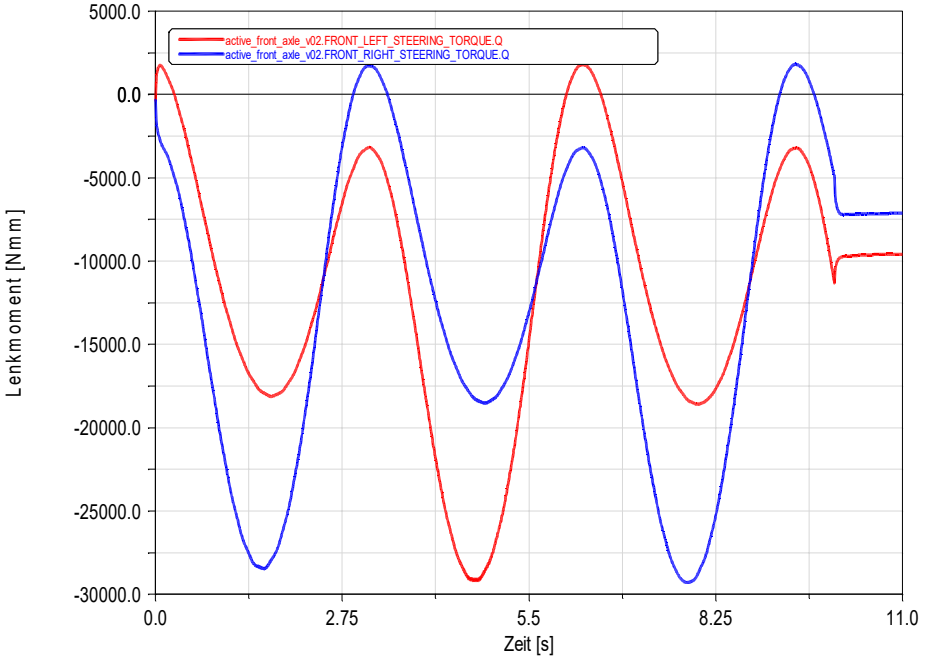


Abbildung 3.21.: Verlauf des Lenkmomentes bei Slalomfahrt

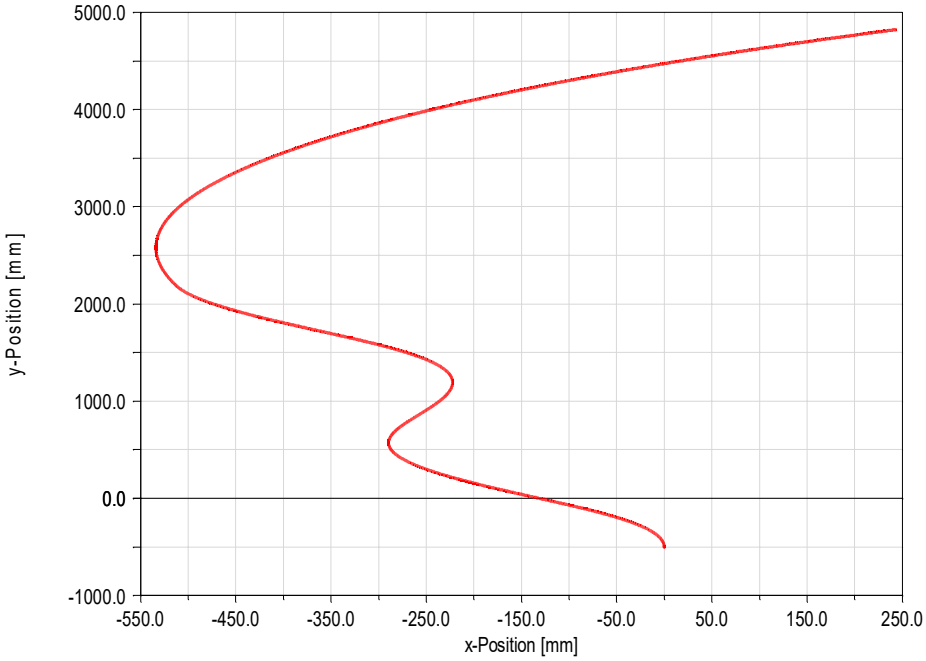


Abbildung 3.22.: Verlauf der Roboterposition bei Slalomfahrt

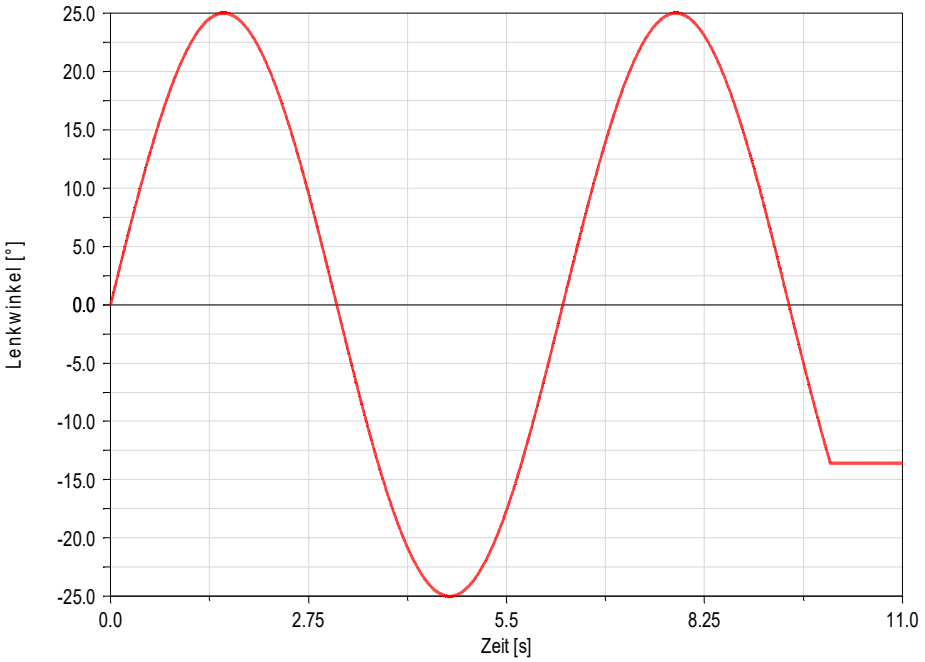


Abbildung 3.23.: Verlauf des Lenkwinkels bei Slalomfahrt

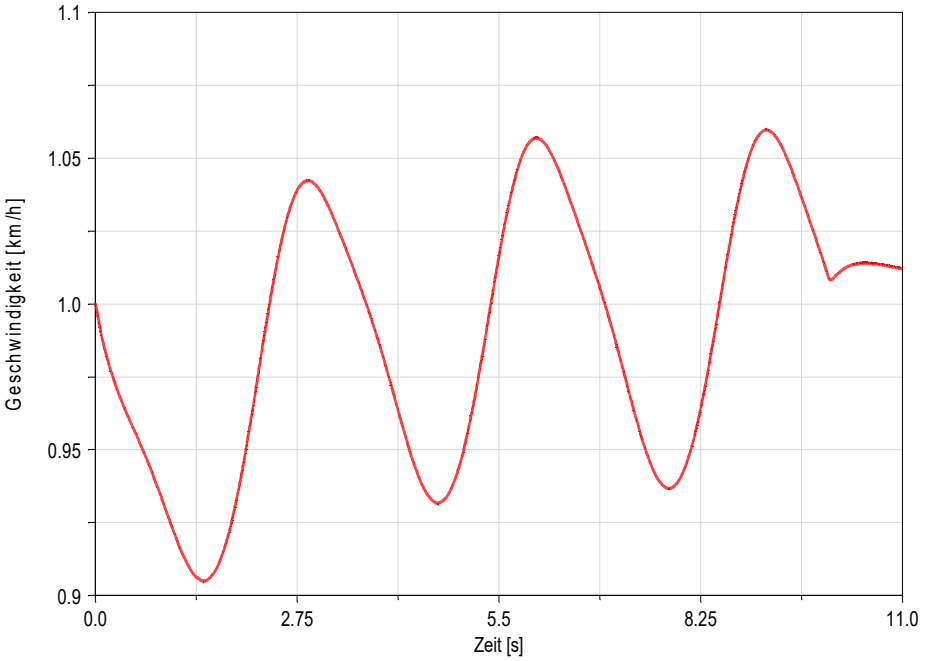


Abbildung 3.24.: Verlauf der Geschwindigkeit bei Slalomfahrt

des Reifens erhöhen. Dies führt allerdings auch zu einem erhöhten Rollwiderstand und somit zu mehr Energieverbrauch. Eine weitere Erkenntnis ist die Größe des Lenkmomentes, hier wurde ein extremer Fall der sehr langsamen Slalomfahrt untersucht und die auftretenden Lenkmomente dienen als Input für die Auswahl von entsprechenden Motoren.

3.4. Co-Simulation

Für die umgesetzte Co-Simulation zwischen ADAMS/Car und MATLAB wurde ein zweites, leicht abgeändertes MKS-Modell in ADAMS erstellt. Bei diesem Modell sind, zum Unterschied zum verwendeten Modell für die fahrdynamische Voruntersuchung, alle vier Räder einzeln und unabhängig voneinander lenkbar. Dies erfordert eine Änderung der Vorlage für die Einzelrädaufhängung, welche für die fahrdynamische Voruntersuchung verwendet wurde. Jedes einzelne Rad kann nun unabhängig voneinander mit einem externen Moment um die Hochachse gedreht, sprich gelenkt, werden. Für eine erfolgreiche Co-Simulation müssen nun geeignete Inputs für das MKS-Modell in ADAMS definiert werden. Die Wahl fiel auf die Antriebsmomente für die Antriebsmotoren und vier Lenkmomente für das Lenken der Räder. Als Output des MKS-Modells sind mehrere Variablen erforderlich, siehe Abschnitt 4.2. Sind die Variablen erfolgreich definiert, kommunizieren die beiden Programmpakte jeden Berechnungsschritt miteinander. Die Dauer des Berechnungsschrittes in MATLAB muss mit der einstellbaren Dauer, in welcher der ADAMS-Block innerhalb MATLAB Informationen aufnimmt, bzw. liefert, zusammenstimmen.

4. Gesamtsimulation

In diesem Kapitel wird die erstellte Gesamtsimulation erläutert. Ausgehend von den Erkenntnissen in Kapitel 3, wird nun die vorliegende Problemstellung in Unterkapitel 4.1 formuliert. Es wird das entworfene Konzept für die Fahrdynamikregelung in Unterkapitel 4.2 vorgestellt und im nächsten Schritt die einzelnen Module der Gesamtsimulation im Abschnitt 4.3 beschrieben. Die Ergebnisse der Gesamtsimulation werden im Kapitel 4.4 präsentiert. Es werden verschiedenste Szenarien simuliert um einen möglichst großen Anwendungsbereich abzudecken. Zum Abschluss dieses Kapitels werden die Ergebnisse der durchgeführten fahrdynamischen Manöver ausführlich diskutiert. Die im vorliegenden Kapitel erwähnten und im Simulationsmodell verwendeten Variablen werden, den entsprechenden Modulen zugeordnet, im Anhang B aufgezählt.

4.1. Problemstellung

Mittels der aufgebauten Co-Simulation ist es nun möglich, in einer frühen Phase der Entwicklung, ein realistisches Fahrzeugverhalten zu simulieren und somit die, für das möglichst genaue Befahren einer Trajektorie, notwendigen Regler des Fahrzeuges innerhalb dieser Gesamtsimulation zu entwerfen und zu testen. Das Ziel dabei ist, neben einer möglichst geringen Positionsabweichung, auch eine möglichst genaue Erfüllung der geforderten Geschwindigkeit, sowie des Aufbauwinkels des Roboters zu erreichen. Das Konzept für die Erreichung dieser Ziele ist in Kapitel 4.2 beschrieben.

4.2. Konzept für die Fahrdynamikregelung

Um die in Abschnitt 4.1 dargestellte Problemstellung bestmöglich zu lösen, wurde ein Konzept, welches in Abbildung 4.1 ersichtlich ist, entwickelt. Es besteht im wesentlichen aus vier getrennten Blöcken und wird im weiteren Verlauf genauer erläutert. Die Aufgabe des ersten betrachteten Blocks ganz links besteht in der Bestimmung der Position bzw. der Abweichung des Roboters entlang der Trajektorie. Dazu bekommt dieser Block als Input die Matrix M, sowie die globalen Roboterkoordinaten. Der genaue Inhalt der Matrix M wird in den Kapiteln 4.3.1, 4.3.2, sowie 4.3.3, erläutert. Im Wesentlichen enthält die Matrix alle Koordinaten der vorgegebenen Punkte und die Sollbewegung des Roboters. Mit welchen Methoden nun die Position und die Abweichung entlang der Trajektorie bestimmt werden, zeigen die Abschnitte 4.3.2 und 4.3.3.

Die erhaltene Abweichung S_e, die Matrix M, sowie die Position entlang der Trajektorie (als Index der Matrix M) dient nun, zusammen mit den globalen Positionskoordinaten, dem Gierwinkel, sowie der Robotergeschwindigkeit, als Input für den nächsten Block. Dieser soll den Geschwindigkeitsvektor, sowie die Winkelgeschwindigkeit bestimmen, zusammengefasst also den bereits vorgestellten Twist (siehe Kapitel 2.2, sowie Gleichung 2.1 des Roboters). Kapitel 4.3.4 erläutert die genaue vorgehensweise innerhalb dieses Blocks.

Die Aufgabe des Blocks *Motion Controller* besteht darin, aus dem Twist des Roboters, sowie den aktuellen Raddrehzahlen und Lenkwinkeln, die für die gewünschte Bewegung des Roboters erforderlichen Lenk-, sowie Antriebsmomente zu ermitteln, welche als Input für das MKS-Modell in ADAMS/Car dienen. Das MKS-Modell soll den realen Roboter abbilden und wird auf einer 2D-Ebene betrieben. Als Rückkopplung des MKS-Modells für die vorhergehenden Blöcke werden nur Werte verwendet, welche auch im realen Fahrzeug durch Messung verfügbar sind, dazu gehören die globalen Roboterkoordinaten, die Lenk-, sowie der Gierwinkel und die Raddrehzahlen.

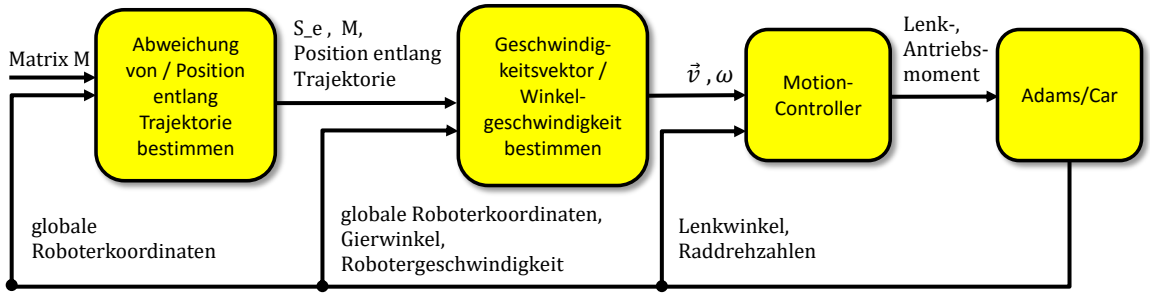


Abbildung 4.1.: Grundsätzlicher Aufbau der Gesamtsimulation

Eine essentielle Vorgabe war dabei die übergegebenen Variablen an den Block *Motion Controller*. Hier wurde gefordert möglichst standardisierte Vektoren zu übergeben, welche auch im Robot Operating System (ROS) vorhanden sind, da dieser Block inklusive den verwendeten Reglern im Block *Geschwindigkeitsvektor / Winkelgeschwindigkeit bestimmen* in C-Code kompiliert wird und anstatt eines MKS-Modells in ADAMS/Car, ein Roboter in ROS betrieben wird. Ein weitere, sich direkt dadurch ableitbare Vorgabe besteht in dem möglichst modularem Aufbau des gesamten Modells, um einzelne Blöcke gezielt kompilieren und in ROS verwenden zu können.

4.3. Module der Gesamtsimulation

Bevor nun auf die einzelnen Blöcke näher eingegangen wird, erfolgt ein Versuch das erstellte Konzept aus der Sicht eines potentiellen Anwenders darzustellen. In Abbildung 4.3 sind die drei wichtigsten Schritte dargstellt. Zuerst definiert ein Anwender (im weiteren auch als Benutzer oder Nutzer bezeichnet) beliebige Punkte in einer 2D-Ebene mit x- und y-Koordinaten. Zusammen mit einer, ebenfalls vom

Nutzer definierten Geschwindigkeit, ergibt sich eine zu fahrende Trajektorie des Roboters, welche gewisse Grenzen nicht überschreiten darf. Eine dieser Grenzen ist der Maximalwert der Querbeschleunigung und ein Ergebniss der fahrdynamischen Voruntersuchung (siehe Abschnitt 3). Um die Grenzen auszuloten bietet sich an, das Fahrzeug vereinfacht als Punktmasse zu betrachten, siehe Abbildung 4.2. Damit ist die Berechnung der Querbeschleunigung mittels Gleichung 4.1 bereits in dieser Phase in erster Näherung möglich und es wird verhindert, dass eine nicht fahrbare Trajektorie definiert wird. Dabei versteht man unter $v_{x,S}$ die Geschwindigkeit tangential an die Trajektorie (Bahngeschwindigkeit) und R als den Radius der momentan befahrenen Bahn. Die Beschleunigung a_y beschreibt die Änderung der Bewegungrichtung des betrachteten Körpers.

$$a_y = \frac{v_{x,S}^2}{R} \tag{4.1}$$

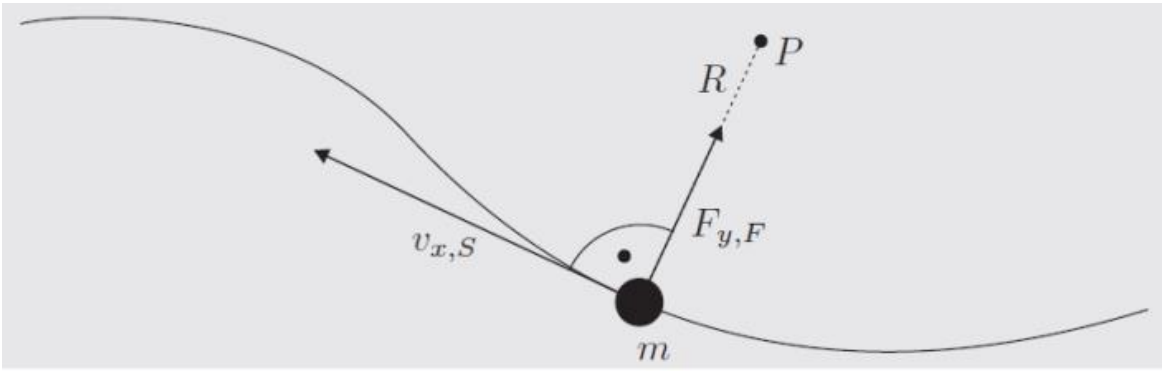


Abbildung 4.2.: Vereinfachte Betrachtung des Roboters als Punktmasse (Moseberg & Roppenecker, 2014, S.219)

Weiters wird noch der Soll-Winkel des Roboteraufbaus vom Anwender definiert. Hier gibt es ebenfalls Grenzen zu beachten, damit beispielsweise eine Überschreitung des für den Roboter zulässigen Lenkwinkels nicht möglich ist. Eine mögliche Strategie bestünde darin, den Roboter ohne Drehung seines Aufbaus (entspricht dem in Kapitel 4.4 definierten Robotermodus) die Trajektorie in der Simulation durchfahren zu lassen. Am einfachsten lässt sich dies dadurch realiseren, den Winkel des Geschwindigkeitsvektors bezogen auf den Aufbauwinkel des Roboters, welcher konstant bleibt, auszuwerten. Überschreitet dieser Winkel die vorgegebene Lenkwinkelbeschränknung nicht, bzw. ist weit genug davon entfernt, ist auch die Überschreitung im realen Betrieb unwahrscheinlich.

Im zweiten Schritt werden Hilfspunkte (im weiteren auch als Zwischenpunkte bezeichnet) generiert, welche einen konstanten Zeitabstand besitzen und für die geforderte Genauigkeit der zu befahrenen Trajektorie notwendig sind. Durch diese Vorab-Berechnung der Zwischenpunkte, gibt es eine Einschränkung für die Distanz zwischen den vom Nutzer definierten Punkten. Als dritten und letzten Schritt kann der Roboter, der nun ausreichend definierten Trajektorie, nachfahren, Abweichungen werden durch die implementierten Regler ausgeglichen.

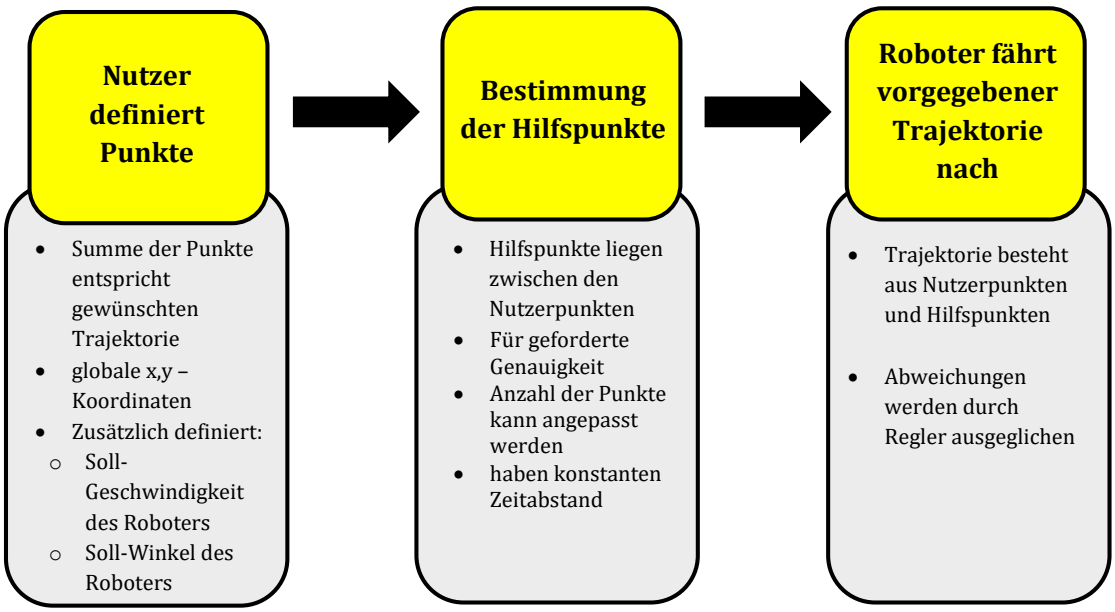


Abbildung 4.3.: Verfolgtes Konzept aus der Sicht eines potentiellen Benutzers

4.3.1. Definition von Benutzereingaben

Wie bereits in Abschnitt 4.3 erwähnt, muss ein potentieller Benutzer einige Eingaben definieren, um ein gewünschtes Befahren einer Trajektorie zu erreichen. Im diesem Kapitel werden nun die ersten Elemente, der bereits erwähnten Matrix M erläutert, der Rest folgt in Kapitel 4.3.2. Der vollständige Inhalt der Matrix M ist in Abbildung 4.4 ersichtlich, dabei soll zuerst ein Blick auf die gelb markierten Spalten eins, zwei, vier und fünf geworfen werden, denn diese Spalten enthalten Eingaben welche direkt vom Benutzer kommen. Dazu zählen die Position der Punkte, in einem globalen Koordinatensystem, welche vom Roboter befahren werden sollen, sowie die Geschwindigkeit an diesen Punkten (die Geschwindigkeit zwischen den Punkten wird somit vereinfacht als linear angenommen) und der Winkel des Roboters zu der Verbindung von zwei Punkten entlang der Trajektorie.

Die restlichen Elemente der Matrix werden anschließend vorab der eigentlichen Simulation der Fahrzeugbewegung generiert und dienen, wie bereits Abbildung 4.1 vorwegnahm, als Input für die Bestimmung der Abweichung von der gegebenen Trajektorie. Nach diesem Schritt enthält die Matrix M nun Eingaben in den gelb markierten Spalten, die Anzahl an Zeilen entspricht der Anzahl an gewünschten Nutzerpunkten.

4.3.2. Berechnung der Soll-Trajektorie

In diesem Abschnitt wird nun die vollständige, für das Befahren des Roboters notwendige, Trajektorie mitsamt allen Informationen in der Matrix M definiert.

Um die Eingabe der Koordinaten der vom Nutzer gewünschten Punkte zu erleichtern, wurde die Darstellung in Polarkoordinaten vorgenommen, siehe Abbildung 4.5.

	1	2	3	4	5	6	7	8	9	10	11	12	13
Einheit	[m]	[m]	[m]	Grad [°]	[m/s]	[s]	[m]	[m]	[m]	[m]	[m]	[m/s^2]	[-]

Erklärung der Spalteneinträge:

1) Globale x- Koordinate des Punktes
2) Globale y- Koordinate des Punktes
3) Weg s (jeweils gesamte Strecke bis zu dem Punkt der Zeile)
4) Winkel Beta (Winkel von Roboter-Aufbau zur Trajektorie)
5) Soll-Geschwindigkeit des Roboters
6) Zeit t (jeweils gesamte Zeit bis zu dem Punkt der Zeile)
7) x- Komponente des Tangentenvektors der Geschwindigkeit
8) y- Komponente des Tangentenvektors der Geschwindigkeit
9) Krümmung der Kurve
10) x- Koordinate des Kurvenmittelpunktes
11) y- Koordinate des Kurvenmittelpunktes
12) Beschleunigung a
13) Variable welche anzeigt ob ein Eintrag (Zeile) ein Nutzer- oder Zwischenpunkt ist
    1...Nutzerpunkt
    0...Zwischenpunkt

Abbildung 4.4.: Erläuterungen bezüglich der Matrix M

Es wird ein gewünschter Startpunkt definiert, alle darauffolgenden Punkte werden nun mittels dem Winkel zwischen zwei Punkten, welcher zur globalen x-Achse hin gemessen wird, und dem Abstand zwischen den Punkten, definiert. Eine exemplarische Berechnung des Punktes P_{i+1} ausgehend von Punkt P_i erfolgt in Gleichung 4.2. Dabei ist $r_{i,i+1}$ der Abstand zwischen den Punkten, und $\varphi_{i,i+1}$ der Winkel zur globalen x-Achse.

$$P_{i+1} = P_i + r_{i,i+1} \cdot \begin{bmatrix} \cos(\varphi_{i,i+1}) \\ \sin(\varphi_{i,i+1}) \end{bmatrix} \tag{4.2}$$

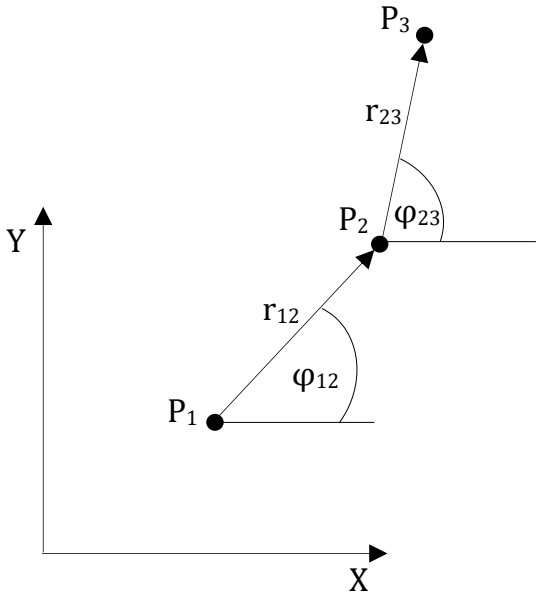


Abbildung 4.5.: Punkte definiert in Polarkoordinaten

Nun wird damit begonnen aus den erhaltenen Eingaben des Nutzers, die entsprechenden Elemente der Matrix M abzuleiten. Als allererstes ist dabei der Weg s (dritte Spalte der Matrix) zu betrachten. Hier ist der Vergleich der Winkel φ entscheidend: Ist der Winkel des Punktes P_{i+1} gleich wie der des Punktes P_i dann wird angenommen, dass die gewünschte Trajektorie zwischen diesen Punkten einer Geraden entsprechen soll. Ist der Winkel nicht ident, entspricht diese einer Krümmung der Trajektorie, der Roboter soll also eine Kurve befahren. Bei dem Vergleich von zwei aufeinanderfolgenden Winkeln φ werden insgesamt also drei verschiedene Punkte betrachtet, siehe auch Abbildung 4.5. Mit Hilfe dieser drei Punkte kann nun ein Kreisradius berechnet werden, es wird also vereinfacht eine konstante Krümmung zwischen den definierten Punkten des Benutzers angenommen. Mittels dieser Berechnung, welche in Anhang B.1 zu finden ist, erhält man den Radius r des Kreisbogens, als auch die Mittelpunktskoordinaten des Mittelpunktes C, siehe Abbildung 4.6. Nun kann die Bogenlänge b des Abschnittes zwischen zwei Punkten berechnet werden, siehe Gleichung 4.3 und 4.4. Die Bogenlänge b wird dabei als Weg für die Spalte drei der Matrix M verwendet.

Die Koordinaten des Krümmungsmittelpunktes werden in die Spalten 10 und 11 der Matrix M eingetragen, die Krümmung wird in die Spalte 9 eingetragen. Nun ist es

auch möglich den Tangentenvektor an einen Nutzerpunkt mit gekrümmter Trajektorie zu berechnen, dazu wird der Verbindungsvektor zwischen Mittelpunkt der Krümmung und dem Nutzerpunkt um 90 Grad gedreht. Mit ihm werden die Spalten 7 und 8 befüllt, er wird im weiteren Verlauf jedoch nicht weiter benötigt.

$$\alpha = 2 \cdot \arcsin\left(\frac{s}{2 \cdot r}\right) \tag{4.3}$$

$$b = r \cdot \alpha \tag{4.4}$$

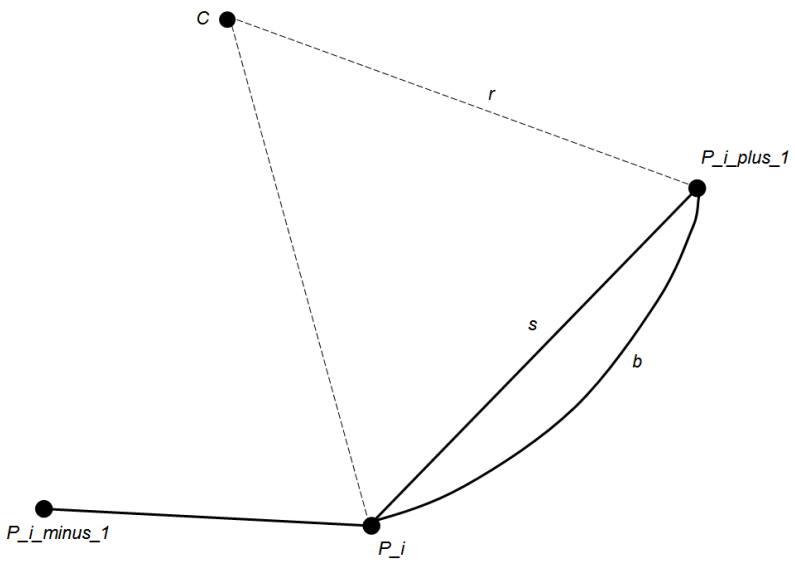


Abbildung 4.6.: Berechnung der Bogenlänge der Trajektorie

Nachdem nun auch die dritte Spalte der Matrix M für jeden Punkt ausgefüllt werden konnte, ist es nun möglich auch die nächste leere Spalte, nämlich Spalte 6 (gesamte Zeit bis zum jeweilig betrachteten Punkt) zu bestimmen. Mit Hilfe der vom Benutzer vorgegebenen Geschwindigkeit lässt sich das Geschwindigkeit-Zeit-Diagramm benutzen, in welchem der Weg s als Fläche erscheint. Diese Fläche lässt sich, da immer ein linearer Geschwindigkeitsverlauf angenommen wurde, als rechtwinklige Trapezfläche betrachten, siehe Gleichung 4.5. Dabei beschreibt Δs den Weg zwischen zwei Punkten, die Zeit Δt die Zeit zwischen zwei Punkten. Mit v_{i+1} sowie v_i werden die Geschwindigkeiten an den Punkten P_{i+1} sowie P_i beschrieben. Die Zeit Δt aufgetrennt und die Gleichung 4.5 umgeformt ergibt die notwendige Gleichung 4.6 um die Zeit für den jeweiligen Punkt P_{i+1} bestimmen zu können.

$$\Delta s = \frac{(v_{i+1} + v_i) \cdot \Delta t}{2} \tag{4.5}$$

$$t_{i+1} = t_i + \frac{2 \cdot \Delta s}{(v_{i+1} + v_i)} \tag{4.6}$$

Die Beschleunigung im Punkt P_{i+1} kann nun ebenso bestimmt werden, siehe Gleichung 4.7. Sie füllt die Spalte 12 der Matrix M und wird aus der Differenz von Geschwindigkeit und Zeit zwischen den Punkten P_{i+1} und P_i bestimmt.

$$a_{i+1} = \frac{\Delta v}{\Delta t} \tag{4.7}$$

Wie bereits die Abbildung 4.3 zu erklären versuchte, reichen die vom Benutzer definierten Punkte nicht aus um eine Trajektorie zu definieren welche ein gewünschtes Befahren möglich machen würde. Es müssen nun Hilfspunkte zwischen diesen Punkten generiert werden, die dem Roboter dabei helfen, den Wunsch des Benutzers bestmöglich erfüllen zu können. Zuallererst muss definiert werden wie viele Zwischenpunkte notwendig sind. Mehr Punkte erhöhen die Genauigkeit, der Abstand zwischen den Punkten wird jedoch immer geringer, letzlich muss noch eine Verarbeitbarkeit gegeben sein. Hier wurde nun folgender Weg gewählt: Anstatt zwischen den Punkten einen Abstand zu definieren, wurde ein konstanter Zeitabstand gewählt. Dies hat gegenüber der bloßen Abstandsdefinition den Vorteil, dass man einen an die Geschwindigkeit angepassten Abstand erhält und hier somit keine weiteren Anpassungen notwendig sind.

Mit der gewählten Zeitdifferenz Δt_{step}, welche gleich groß wie das Berechnungsintervall zwischen MATLAB und ADAMS/Car gewählt wurde (eine kleinere Auflösung würde keinen Genauigkeitsgewinn bringen, größere Auflösungen haben einen Verlust der Genauigkeit zur Folge), ergeben sich nun nach Gleichung 4.8 direkt die Anzahl der Zwischenpunkte. Punkte mit dem Index i werden dabei als zuletzt befahrenen Nutzerpunkt, Punkte mit Index $i+1$ als den nächsten anzufahrenden Nutzerpunkt bezeichnet. Die Zwischenpunkte werden mit dem Index $i+1 \rightarrow i,n$ bezeichnet und entsprechen dem n-ten Punkt zwischen den Punkten P_i und P_{i+1}. Weiters wird die Matrix M, welche bis zu dem jetzigen Schritt nur mit Nutzerpunkten befüllt ist, erweitert um so Platz für die Zwischenpunkte zu haben. Dabei wird die Zeilenanzahl der Matrix M jeweils um die Anzahl an benötigten Zwischenpunkten, wie in Gleichung 4.8 ersichtlich ist, erweitert. Um später noch zwischen den Nutzer- und Zwischenpunkten unterscheiden zu können, was vor allem für die Kalkulation der Vorschaudistanz wichtig ist, siehe Kapitel 4.3.4.2, wird in Spalte 13 der Matrix M eine Variable eingeführt die dies kennzeichnet, siehe Abbildung 4.4.

$$\textit{Anzahl an Zwischenpunkte } n = \frac{t_{i+1} - t_i}{\Delta t_{step}} \tag{4.8}$$

Den Weg zwischen den einzelnen Punkten kann man nun mittels der Gleichung 4.9 berechnen. Da ein linearer Geschwindigkeitsverlauf zwischen den Vorgaben an den Nutzerpunkten angenommen wird, ist die Beschleunigung $a_{i+1\rightarrow i}$ zwischen zwei Nutzerpunkten jeweils konstant, somit liegt eine gleichförmig beschleunigte Bewegung vor. Die Integrationskonstanten sind die Geschwindigkeit und der Weg, zu Beginn des betrachteten Abschnittes, v_i und s_i. Die verwendete Zeit $t_{i+1\rightarrow i,n}$ entspricht zwischen dem momentan betrachteten n-ten Zwischenpunkt und dem Nutzerpunkt am Beginn des betrachteten Abschnittes.

$$s_{i+1\to i,n} = \frac{a_{i+1\to i} \cdot t_{i+1\to i,n}^2}{2} + v_i \cdot t_{i+1\to i,n} + s_i \tag{4.9}$$

Im Fall eines geraden Trajektorienstücks können die Zwischenpunkte nun einfach berechnet werden. Wie in Gleichung 4.10 ersichtlich, wird, vom Punkt P_i ausgehend, ein Richtungsvektor $\overrightarrow{n}_{i+1\to i,n}$ welcher zwischen den Punkten P_{i+1} und P_i aufgestellt wurde, mit der Streckendifferenz multipliziert und zu dem Punkt P_i addiert. Unter der Streckendifferenz $(s_{i+1\to i,n} - s_i)$ wird dabei der Weg zwischen dem letzten Nutzerpunkt P_i und dem momentan betrachteten Zwischenpunkt $P_{i+1\to i,n}$ verstanden.

$$P_{i+1\to i,n} = P_i + (s_{i+1\to i,n} - s_i) \cdot \overrightarrow{n}_{i+1\to i,n} \tag{4.10}$$

Die Geschwindigkeit bei einem Zwischenpunkt kann nun ebenfalls, eine gleichförmig beschleunigte Bewegung angenommen, berechnet werden. Dazu wird in Gleichung 4.11 die Geschwindigkeit v_i am Punkt P_i vermehrt um einen Geschwindigkeitterm, welcher durch die Einwirkung der Beschleunigung $a_{i+1\to i}$ für den Zeitraum $t_{i+1\to i,n}$ entsteht.

$$v_{i+1\to i,n} = v_i + a_{i+1\to i} \cdot t_{i+1\to i,n} \tag{4.11}$$

Nun geht es darum die Zwischenpunkte auch für gekrümmte Trajektorienabschnitte zu bestimmen. Dazu wird zuallererst, ausgehend vom Punkt P_i, der Öffnungswinkel zwischen den Punkten P_i und $P_{i+1\to i,n}$ berechnet. In Abbildung 4.14 entspricht dies dem Winkel zwischen $\overrightarrow{P_i - C}$ und $\overrightarrow{P_{i\_plus\_1} - C}$. Dieser Winkel wird laut Gleichung 4.12 berechnet. Die Strecke $(s_{i+1\to i,n} - s_i)$ entspricht dabei der Bogenlänge b.

$$\alpha_{i+1\to i,n} = (s_{i+1\to i,n} - s_i) \cdot \rho_{i+1\to i,n} \tag{4.12}$$

Der gewünschte Zwischenpunkt $P_{i+1\to i,n}$ wird nun folgendermaßen berechnet: Ausgegangen wird vom Krümmungsmittelpunkt (da im Abschnitt zwischen 2 Nutzerpunkten immer eine konstante Krümmung herrscht, auch Kreismittelpunkt), welcher für den i-ten Nutzerpunkt P_i in der i-ten Zeile der Matrix M_u (Matrix M ohne Zwischenpunkte) und in der Spalte 10 und 11 zu finden ist. Nun wird ein Vektor zwischem dem Krümmungsmittelpunkt (auch als C_i bezeichnet) und dem zuletzt betrachteten Nutzerpunkt errichtet und dieser um den, in Gleichung 4.12 berechneten, Winkel $\alpha_{i+1\to i,n}$ mit Hilfe der Transformationsmatrix T, siehe Gleichung 4.13, gedreht. Zusammen ergibt dies den nächsten Zwischenpunkt $P_{i+1\to i,n}$, welcher in Gleichung 4.14 zu sehen ist.

$$\underline{\underline{T}} = \begin{bmatrix} \cos\alpha_{i+1\to i,n} & \sin\alpha_{i+1\to i,n} \\ -\sin\alpha_{i+1\to i,n} & \cos\alpha_{i+1\to i,n} \end{bmatrix} \tag{4.13}$$

$$P_{i+1\to i,n} = \begin{bmatrix} \underline{\underline{M}}_{u(i,10)} \\ \underline{\underline{M}}_{u(i,11)} \end{bmatrix} + \underline{\underline{T}} \cdot \left(\overrightarrow{P_i - C_i}\right) \tag{4.14}$$

Die aufgezählten Schritte werden für jeden Abschnitt durchgeführt, so wird die Trajektorie durchgängig mit Zwischenpunkten aufgefüllt. In Abbildung 4.7 ist ein kurzer Abschnitt einer solchen Trajektorie zu erkennen. Dabei stehen die blauen Punkte für Nutzerpunkte, die schwarzen Punkte stellen Zwischenpunkte dar.

Nachdem nun die Matrix M und somit alle notwendigen Inputdaten vollständig definert sind, kann nun die eigentliche Simulation beginnen. Es werden nun im weiteren die einzelnen Blöcke, welche bereits in Abbildung 4.1 gezeigt wurden, einzeln im Detail vorgestellt.

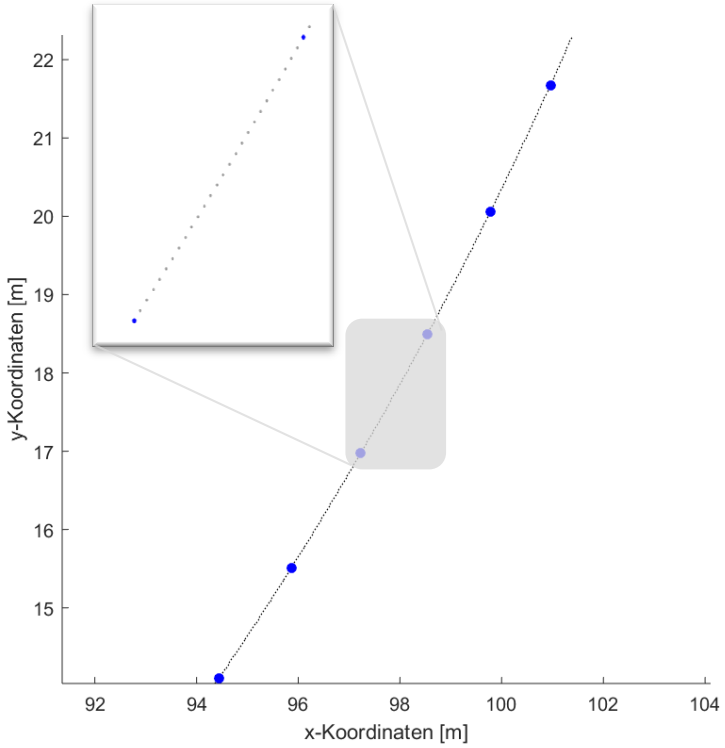


Abbildung 4.7.: Nutzerpunkte und Zwischenpunkte

4.3.3. Abweichung zur Soll-Trajektorie bestimmen

In diesem Block soll die Abweichung von der Trajektorie, sowie auch die Position entlang dieser, berechnet werden. Als Input dienen dabei die bereits besprochenen Informationen aus der Matrix M, sowie die globalen Roboterkoordintaten, welche während jedes Zeitschrittes der Simulation aktualisiert aus ADAMS/Car zur Verfügung stehen.

Die Bestimmung der Position entlang der Trajektorie bedeutet, den Bereich zwischen zwei Punkten der Trajektorie zu finden, in dem sich der Roboter gerade befindet. In der Abbildung 4.8 ist dieser Vorgang exemplarisch für einen Ausschnitt einer Trajektorie mit vier Punkten dargestellt. Die momentane Roboterposition, in der Abbildung 4.8 als $P_{current}$ bezeichnet, befindet sich nun zwischen jenen beiden Punkten der Trajektorie, bei welchen die beiden Gleichungen 4.15, sowie 4.16, beide erfüllt sind. Hier wird mittels Skalarprodukt zwischen den Verbindungen der einzelnen Punkte überprüft ob

ein spitzer, bzw. stumpfer Winkel vorhanden ist. Für eine erfolgreiche Überprüfung sind folgende zwei Eigenschaften notwendig: Es müssen jeweils zwei Punkte vor der momentanen Roboterposition liegen und zwei dahinter. Das bedeutet konkret, dass für den Beginn, aber vor allem für das Ende einer Trajektorie, jeweils ein zusätzlicher Hilfspunkt notwendig ist.

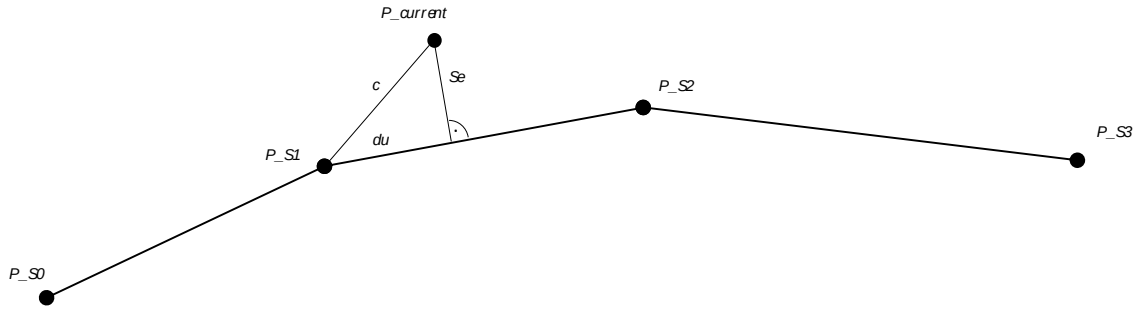


Abbildung 4.8.: Abweichung zur Trajektorie geometrisch bestimmen (in Anlehung an Rauh & Mössner-Beigel, 2008)

$$\left(\overrightarrow{P_{current} - P_{S1}}\right) \cdot \left(\overrightarrow{P_{S2} - P_{S0}}\right) \geq 0 \tag{4.15}$$

$$\left(\overrightarrow{P_{current} - P_{S2}}\right) \cdot \left(\overrightarrow{P_{S3} - P_{S1}}\right) < 0 \tag{4.16}$$

Die Bestimmung des Abstandes des Roboters auf die Trajektorie lässt sich vereinfachen zu dem Problem den Abstand eines Punktes zu einer Geraden zu bestimmen, siehe Abbildung 4.9. Die Fläche $A_{\triangle}$ die durch die beiden Vektoren $\overrightarrow{AP}$ und $\overrightarrow{u}$ aufgespannt wird, lässt sich auf zweierlei Arten berechnen. Zum Einen beschreibt die Fläche ein Dreieck, welches mit der Gleichung 4.17 definiert ist, g entspricht dabei dem Betrag des Vektors $\overrightarrow{u}$ und h den Abstand zwischen der Gerade g und dem Punkt P. Zum Anderen kann die Fläche aber auch über den Betrag des Kreuzproduktes der Vektoren $\overrightarrow{AP}$ und $\overrightarrow{u}$ bestimmt werden, siehe Gleichung 4.17.

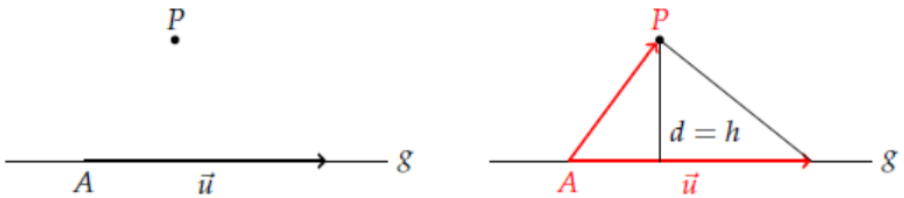


Abbildung 4.9.: Abstand eines Punktes zu einer Geraden (de Brabandt, 2015)

$$A_{\triangle} = \frac{1}{2} \cdot |\overrightarrow{AP} \times \overrightarrow{u}| \tag{4.17}$$

$$A_{\triangle} = \frac{1}{2} \cdot g \cdot h \tag{4.18}$$

Diese zwei Arten der Flächenbestimmung werden in Gleichung 4.19 gleichgesetzt und nach dem Umformen kann der Abstand d eines Punktes zu einer Geraden bestimmt werden, wenn die Gerade durch einen Punkt und einen Richtungsvektor gegeben ist, siehe Gleichung 4.20.

$$\begin{aligned} \frac{1}{2} \cdot g \cdot h &= \frac{1}{2} \cdot |\overrightarrow{AP} \times \overrightarrow{u}| \\ g = |\overrightarrow{u}| \ , \ h = d \ , \ \overrightarrow{AP} &= \overrightarrow{p} - \overrightarrow{a} \end{aligned} \tag{4.19}$$

$$d = \frac{|(\overrightarrow{p} - \overrightarrow{a}) \times \overrightarrow{u}|}{|\overrightarrow{u}|} \tag{4.20}$$

Nach dem nun die Position entlang der Trajektorie, sowie die Abweichung von der Trajektorie in jedem Zeitschritt bestimmt werden kann, wird als nächstes der Block bearbeitet, welcher den Twist des Roboters berechnet.

4.3.4. Twist des Roboters bestimmen

In diesem Abschnitt soll nun der Twist des Roboters bestimmt werden. Als Input erhält dieser Block die bereits besprochene momentane Abweichung, normal zu dem momentanen Trajektorienstück zwischen 2 Punkten, welches auch den Index, die zweite Übergabevariable, festlegt. Das simulierte Fahrzeug liefert dabei die momentane Position, die Geschwindigkeit, sowie den Winkel des Roboters bezüglich dem globalen Koordinatensystem.

Die oberste Ebene des erstellten MATLAB-Simulink Modells ist dabei in Abbildung 4.10 zu sehen. Der grundsätzliche Ablauf in diesem Block lässt dabei folgendermaßen beschreiben: Der gesuchte Twist enthält den Geschwindigkeitsvektor, somit muss sowohl Betrag als auch die Richtung dieses Vektor, ausgehend vom Roboterschwerpunkt, bestimmt werden. Der Betrag des Geschwindigkeitsvektors wird im Block *Velocity_Control*, siehe Kapitel 4.3.4.1, bestimmt. Die Richtung des Geschwindigkeitsvektors wird dabei, ausgehend von einem anvisierten Punkt entlang der Trajektorie, über eine proportionale Regelung bestimmt. Besonderes Augenmerk wird auf die Position des Roboters bezüglich der Trajektorie gelegt, siehe Kapitel 4.3.4.3, in dem der Block *P-Control for Position* erläutert wird. Das zweite Element des Roboter-Twists ist die Winkelgeschwindigkeit des Roboters. Mit Hilfe dieser Größe wird der Aufbauwinkel des Roboters mittels eines Sliding-Mode-Regler geregelt, siehe Kapitel 4.3.4.4. Im MATLAB-Simulink Modell entspricht dies dem Block *calculating_omega_b*.

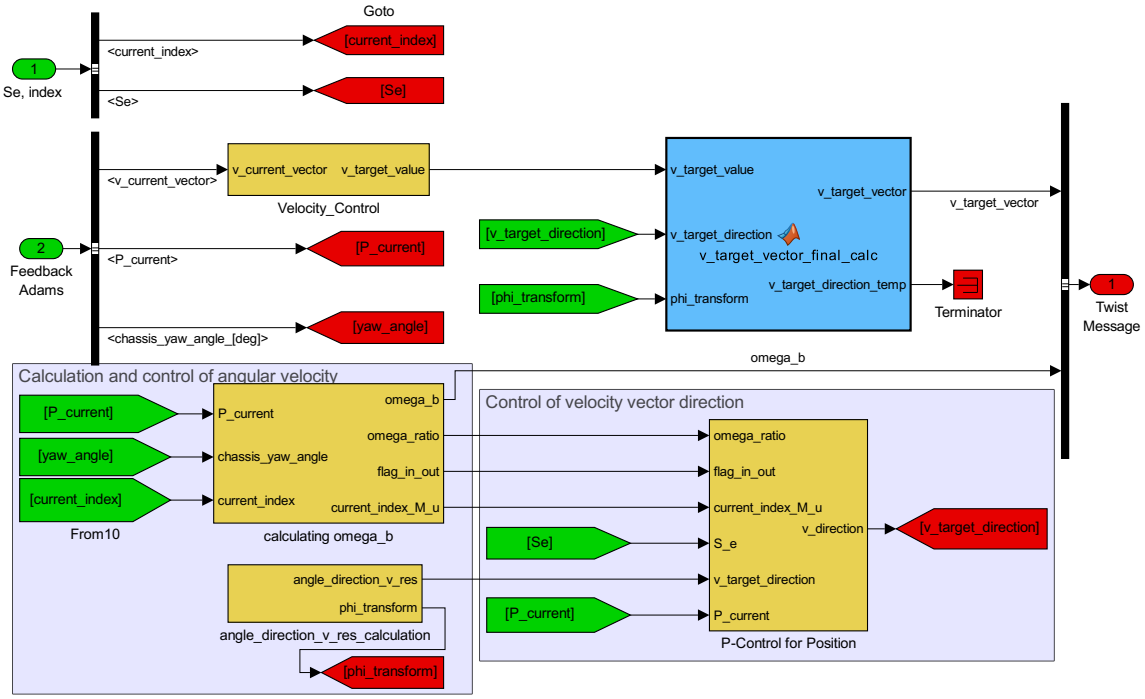


Abbildung 4.10.: Ablauf für die Bestimmung des Roboter-Twists

4.3.4.1. Geschwindigkeitsregelung

In diesem Kapitel erfolgt nun die Erkärung der Regelung für die Robotergeschwindigkeit. Ausgegangen wird von den berechneten erforderlichen Geschwindigkeiten, welche in Zeitabhängigkeit angegebenen werden können. Möglich macht dies die Tatsache, dass eine Trajektorie stets auch eine Funktion der Zeit ist. Hier bestünde auch die Möglichkeit zu fordern, die Trajektorie in der vorgegebenen Zeit zu absolvieren, jedoch unter Umständen nicht unter Einhaltung der vorgegbenen Geschwindigkeiten. Auf diese Möglichkeit wurde verzichtet. Somit kann in Gleichung 4.21 die benötigte Geschwindigkeit berechnet werden um die vorgegebenen Anforderungen zu erfüllen. Dabei entspricht $v(t)$ der geforderten Geschwindigkeit, $v_{current}$ der aktuellen Robotergeschwindigkeit und p_v einem gewähltem Parameter.

Wie in Abschnitt 4.4 zu sehen ist, würde die Robotergeschwindigkeit stets langsamer als die geforderte Geschwindigkeit sein, da der undeformierte Reifenradius für die Berechnung der Winkelgeschwindigkeit ω in Abschnitt 4.3.5 verwendet wird. Der reale Radius vom Radmittelpunkt zum Momentanpol des Rades ist stets geringer (Hirschberg & Waser, 2016), deswegen ist auch die berechnete Raddrehzahl zu gering. Die proportionale Regelung versucht diesen Umstand auszugleichen, sowohl $v(t)$ als auch $v_{current}$ wurden, um zu große Differenzen zwischen zwei Zeitschritte zu vermeiden, mit zwei PT1-Gliedern verzögert. Sollte der Betrag der vorherrschenden Differenz $(v(t) - v_{current})$ einen vorgegebenen Grenzwert überschreiten, erfolgt die Meldung eines Fehlers.

$$v_{target\ value} = v(t) - p\_v \cdot (v(t) - v_{current}) \tag{4.21}$$

Für das Stehenbleiben des Roboters kann bei Bedarf auf eine eigenständige Regelung zurückgegriffen werden. Ist der Roboter nur mehr eine zuvor definierte Distanz vom Zielpunkt entfernt wird diese Regelung aktiviert und die Geschwindigkeit in Abhängigkeit von der Entfernung zum Zielpunkt berechnet, siehe Gleichung 4.22. Die Variable *distance from goal* steht dabei für die Entfernung des Roboters bis zum Ziel (entlang der Trajektorie). Der Parameter k_v wurde in Anlehnung an Corke (2011) gewählt. Es wurde zusätzlich eine obere Grenze eingerichtet, sollte die Geschwindigkeit nach dem Umschalten auf die in Gleichung 4.22 ausgeführte Regelung höher sein als die vorgegebene Geschwindigkeit $v(t)$.

$$v_{target\ value} = k\_v \cdot distance\ from\ goal \tag{4.22}$$

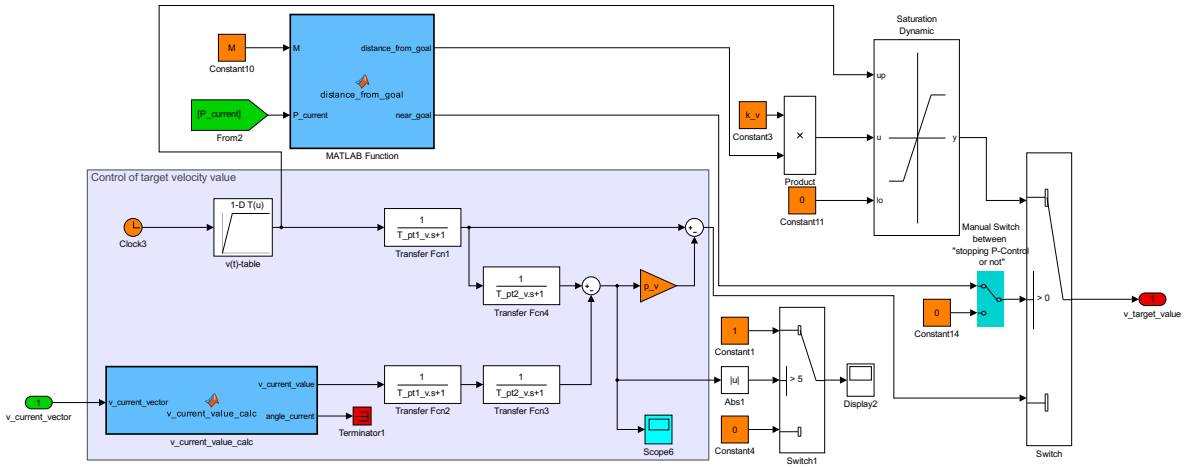


Abbildung 4.11.: Ablauf für die Bestimmung der Robotergeschwindigkeit

4.3.4.2. Berechnung des Vorschaupunktes

Die Richtung des Geschwindigkeitsvektor wird bestimmt, in dem ein Vektor aufgestellt wird zwischen dem Robotermittelpunkt und einem, eine gewisse Anzahl an Punkten vor dem Roboter liegendem, Punkt auf der zu befahrenden Trajektorie. Das MATLAB-Simulink Modell für die Bestimmung des Vorschaupunktes ist in Abbildung B.1 zu sehen. Die wählbare Variable *look_ahead_points* entspricht der Anzahl an Zeilen in der Matrix M, die nach vorne gezählt werden um zu dem gewünschten Vorschaupunkt zu gelangen. Eine große Vorschaudistanz liefert dabei ein stabileres Fahrverhalten, jedoch werden vermehrt Segmente wie scharfe Kurven und Ecken abgekürzt. Kürzere

Vorschaudistanzen liefern ein genaueres Abfahren der Trajektorie, können jedoch ein instabiles Verhalten aufweisen. Eine zusätzliche Vorgabe ist, dass nicht mehr als durch die Variable *number_of_user_points_ahead* definierte Anzahl an Nutzerpunkte nach vorne geschaut werden darf.

In den definierten MATLAB-Funktionen erfolgt eine Abfrage die überprüft, wie viel in den, in der Matrix M übersprungenen Zeilen, Nutzerpunkte vorhanden sind. Dies geschieht mit der bereits erwähnten Spalte 13 der Matrix M. Ist die Vorschaudistanz zu groß gewählt, verringert sie sich durch diese Überprüfung auf den maximal zulässigen Wert. Exakt betrachtet schwankt dieser maximale Wert durch das Befahren von Nutzerpunkten ständig. Wird ein Nutzerpunkt passiert, steigt die maximal zulässige Vorschaudistanz sprunghaft an und verringert sich wieder bis zum Erreichen des nächsten Vorschaupunktes (bei Annahme konstanter Abstände der Nutzerpunkte). Eine mögliche Abhilfe ist, die Distanz von der momentanen Position entlang der Trajektorie bis zum nächstfolgenden Nutzerpunkt zu der Variable *look_ahead_points* zu addieren.

Wurde nun eine Vorschaudistanz, ausgedrückt in Punkten entlang der Trajektorie und durch die Variable *look_ahead*, berechnet, kann im nächsten Schritt der Winkel des Geschwindigkeitsvektors bestimmt werden. Dabei wird, siehe Gleichung 4.23, der berechnete Vektor zwischen Vorschaupunkt und der Roboterposition $\overrightarrow{v_{res}}$ normiert und durch die Transformationsmatrix T gedreht, um vom globalen Koordinatensystem zum Roboterkoordinatensystem zu gelangen. Die Matrix T, siehe Gleichung 4.13, ist in diesem Fall eine Funktion des Roboteraufbauwinkels, bezeichnet mit der Variable *yaw_angle* und wird vom simulierten Fahrzeug geliefert.

$$\overrightarrow{v}_{target\ direction} = \underline{\underline{T}} \cdot \frac{\overrightarrow{v_{res}}}{|\overrightarrow{v_{res}}|} \tag{4.23}$$

Vom dem erhaltenen Richtungsvektor $\overrightarrow{v}_{target\ direction}$ wird nun der Winkel, bezeichnet mit *angle_direction_v_res* bezogen auf die y-Achse, bezeichnet mit Y, des Roboters berechnet, siehe Gleichung 4.24. In der Abbildung 4.21 entspricht dies dem Winkel α zwischen der y-Achse des Roboters und dem Geschwindigkeitsvektor. Um nun noch zu bestimmen ob der Geschwindigkeitsvektor links oder rechts der y-Achse des Roboters liegt, wird das Kreuzprodukt zwischen dem Geschwindigkeitsvektor und der y-Achse des Roboters gebildet, siehe Gleichung 4.25. Die einzige Komponente welche nicht null ergibt (da sich die beiden Vektoren in der Ebene befinden), ist die z-Komponente. Mit Hilfe des Vorzeichens dieser Komponente kann bestimmt werden, auf welcher Seite der Geschwindigkeitsvektor liegt.

$$angle\_direction\_v\_res = \arccos\left(\frac{\overrightarrow{v}_{target\ direction} \cdot \overrightarrow{Y}}{|\overrightarrow{v}_{target\ direction}| \cdot |\overrightarrow{Y}|}\right) \tag{4.24}$$

$$\overrightarrow{c} = \overrightarrow{v}_{target\ direction} \times \overrightarrow{Y} \tag{4.25}$$

Durch das Ausdrücken der Richtung des Geschwindigkeitsvektors als Winkel, bezogen auf die y-Achse des Roboterkoordinatensystem kann der Winkel im weiteren (Kapitel

4.3.4.3) einer proportionalen Regelung, abhängig von der Abweichung zur Trajektorie, unterworfen werden.

4.3.4.3. Berechnung der Richtung des Geschwindigkeitvektors

In diesem Abschnitt wird nun eine proportionale Regelung für den Winkel des Geschwindigkeitsvektors des Roboters vorgestellt. Gleichung 4.26 zeigt dabei die zugrunde liegende mathematische Formulierung. Der in Kapitel 4.3.4.2 berechnete Winkel des Geschwindigkeitsvektors, hier mit $v_{target\ direction}$ bezeichnet, wird vermehrt oder verringert, um den Korrekturterm $v_{direction\ correction}$ multipliziert mit der Variablen $ratio$, welche angibt wie stark der Roboteraufbau sich während einer Kurvenfahrt mitbewegt. Dies wird in Anhang B.3.1 näher erläutert. Der Term $v_{direction\ correction}$ ist abhängig von der Größe der Abweichung des Roboters von der Trajektorie S_e, siehe Gleichung 4.27. Der Parameter p wurde dabei empirisch bestimmt und ist ebenfalls von S_e abhängig. In der Abbildung 4.13 ist dieser Umstand ersichtlich. Das gesamte MATLAB-Simulink Modell für die proportionale Regelung des Geschwindigkeitsvektors ist in Abbildung 4.12 zu sehen.

$$v_{direction} = v_{target\ direction} \pm (ratio \cdot v_{direction\ correction}) \tag{4.26}$$

$$v_{direction\ correction} = S\_e \cdot p \tag{4.27}$$

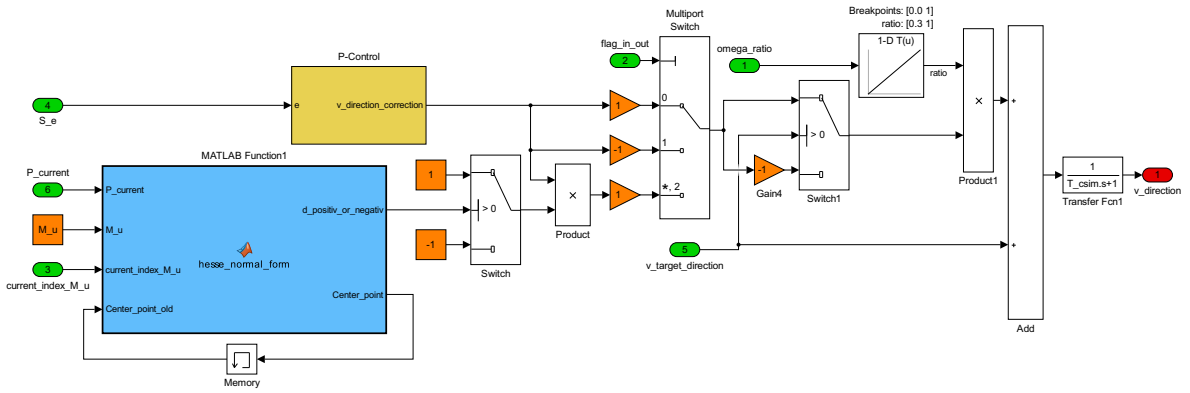


Abbildung 4.12.: Ablauf für die Berechnung der Richtung des Geschwindigkeitsvektors

Das zu bestimmende Vorzeichen in der Gleichung 4.26 ist davon abhängig, ob sich der Roboter außerhalb oder innerhalb der Trajektorie befindet. Dies ist notwendig, da die Abweichung der Trajektorie als Absolutwert S_e vorliegt, es jedoch von Interesse ist, in welche Richtung sich der Roboter nun verstärkt (proportionale Regelung) bewegen soll, um die vorliegende Abweichung S_e bestmöglich auszugleichen.

Prinzipiell werden zwei Fälle unterschieden:

- Bewegung entlang eines gekrümmten Trajektorienstücks

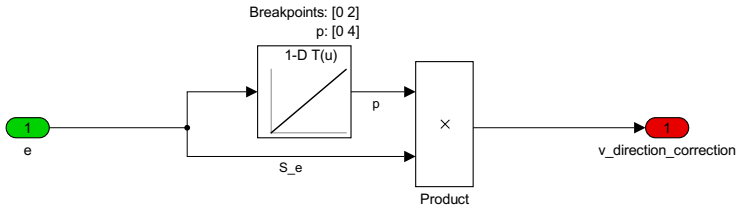


Abbildung 4.13.: P-Regelung abhängig von der Abweichung der Trajektorie

  - Befindet sich die Roboterposition entlang eines Stückes der Trajektorie, welche eine Krümmung aufweist, wird, siehe Abbildung 4.14, zuerst der Mittelpunkt dieser Krümmung, vereinfacht über lineare Geradestücke, berechnet. Der Roboter befinde sich in Abbildung 4.14 zwischen den Punkten P_2 und P_3, es werden somit zwei Punkte vor und zwei Punkte hinter der Roboterposition benötigt. Nach der Bildung von zwei Normalvektoren, dessen Schnittpunkt den Mittelpunkt der Krümmung darstellen, wird nun überprüft wie weit der Roboter von dem Punkt C entfernt ist: Ist die Entfernung größer, als die Entfernung der Punkte P_2 und P_3 vom Punkt C, wird angenommen, dass sich die Roboterposition außerhalb der Trajektorie befindet. Die Variable $flag\_in\_out$ legt dabei fest, ob sich der Roboter außen, innen, oder auf einem Geradenstück befindet.
- Bewegung entlang eines geraden Trajektorienstücks
  - Bewegt sich der Roboter nun entlang einer Geraden, muss eine andere Strategie gefunden werden, um zu bestimmen, ob der Roboter sich außen oder innen befindet. Mit Hilfe der Darstellung eines betrachteten Trajektorienstücks als Gerade, bestimmt über zwei Punkte, in Hessescher Normalform, siehe Anhang B.6, kann diese Bestimmung erfolgen. Bei dieser Darstellung kann der Abstand eines Punktes, in diesem Fall die momentane Roboterposition, zu einer Geraden bestimmt werden. Da dies vorzeichenbehaftet erfolgt, kann eine Aussage über die relative Position entlang der Trajektorie, bezogen auf ein Referenzkoordinatensystem getroffen werden. Das globale Koordinatensystem kann hierfür nicht verwendet werden, wie in der Abbildung 4.15 deutlich wird. Für das erste befahrene Geradenstück ist die Frage, was nun als innen oder als außen bezeichnet wird, abhängig vom weiteren Verlauf der Trajektorie. Es wird nun also für das blaue Geradenstück das Koordinatensystem (x_1, y_1) gewählt, für das rote Geradenstück am Ende der Trajektorie wird dagegen das Koordinatensystem (x_2, y_2) gewählt. Die Ursprünge der Koordinatensysteme entsprechen dabei den Mittelpunkten der jeweiligen Krümmung der Trajektorie. Mit der Variable $d\_positiv\_or\_negativ$ wird nun festgelegt, ob sich der Roboter innen oder außen befindet.

4.3.4.4. Regelung des Roboter-Aufbauwinkels

Das zweite Element des zu bestimmenden Roboter-Twists ist die Winkelgeschwindigkeit des Roboteraufbaus. Mit dem einher geht die Regelung des Aufbauwinkels des

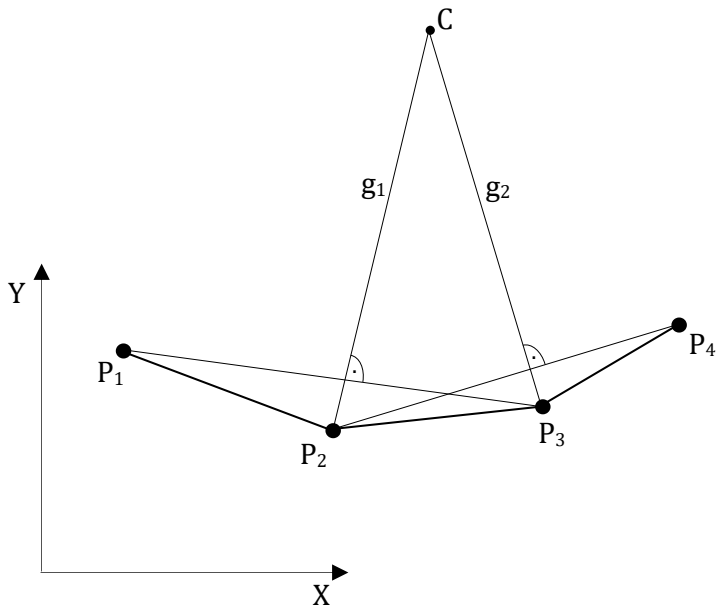


Abbildung 4.14.: Mittelpunktsbestimmung über vier Punkte

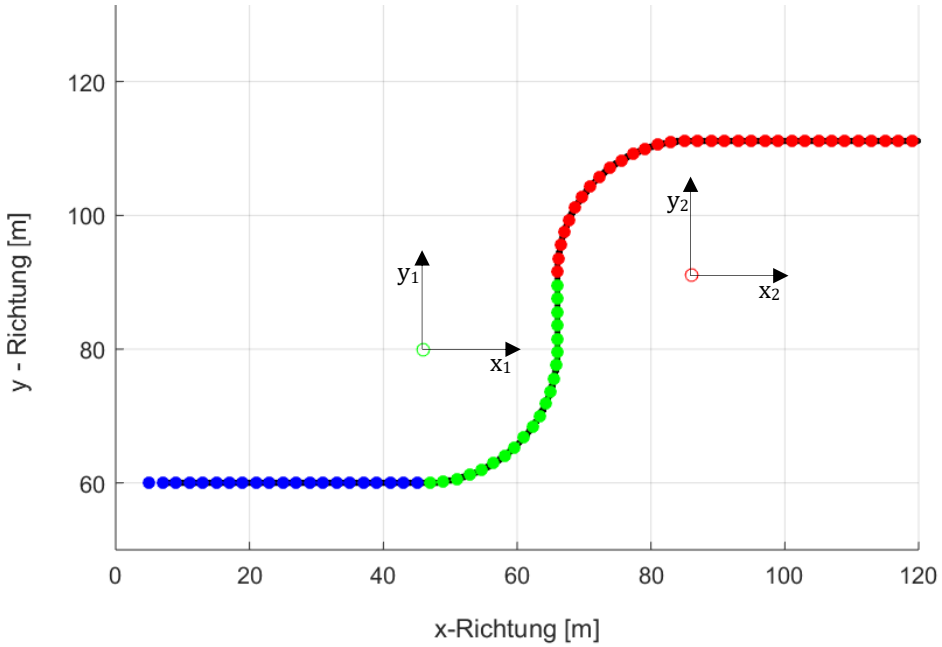


Abbildung 4.15.: Trajektorie mit s-Verlauf

Roboters, um die vorgegebenen Werte bestmöglich einzuhalten. Einen Überblick über die vorherrschenden Winkel gibt die Abbildung 4.16. Der Winkel ψ wird definiert als der Winkel zwischen einem Trajektorienstück und der globalen x-Achse. Des weiteren wird der Winkel β vom Nutzer selbst definiert, er beschreibt den Winkel zwischen Trajektorie und y-Achse des Roboters. Der Winkel befindet sich in der Spalte 4 der Matrix M und ist für jeden vom Nutzer definierten Punkt angegeben. Der Winkel φ lässt sich dann direkt aus den zwei angeführten Winkeln ableiten und beschreibt den Winkel zwischen y-Achse des Roboters und der x-Achse des globalen Koordinatensystems.

Diesen Winkel φ gilt es jetzt für jeden Zeitschritt zu bestimmen. Dafür wurden zwei Möglichkeiten in Betracht gezogen:

- Berechnung über geometrische Zusammenhänge
  - In dieser Variante wird ein Vektor $\left(\overrightarrow{P_{n+1} - P_n}\right)$ zwischen dem zuletzt befahrenen und dem nächsten Punkt (gleichgültig ob Nutzer- oder Zwischenpunkt) aufgestellt , da die momentane Position entlang der Trajektorie bekannt ist. Nun wird in der Gleichung 4.28 der Winkel zwischen eben diesem Vektor und der globalen x-Achse des Roboters aufgestellt. Ist dieser Winkel berechnet, muss er noch dem richtigen Quadranten zugeordnet werden, da die Umkehrfunktion der cos-Funktion nur einen Wertebereich von $-\frac{\pi}{2}$ bis $\frac{\pi}{2}$ aufweist. Zu guter Letzt kann nun in der Gleichung 4.29 der Winkel φ für den betrachteten Punkt berechnet werden. Der Winkel β wird dabei, da er nur für die Nutzerpunkte definiert ist, linear interpoliert, um auch Werte für die Zwischenpunkte zu erhalten.

$$\psi = \arccos\left(\frac{\overrightarrow{P_{n+1} - P_n} \cdot \overrightarrow{X}}{|\overrightarrow{P_{n+1} - P_n}| \cdot |\overrightarrow{X}|}\right) \tag{4.28}$$

$$\varphi = \psi - \beta \tag{4.29}$$

- Berechnung über Winkelgeschwindigkeiten
  - Bei der zweiten Variante erfolgt eine Betrachtung der vorherrschenden Winkelgeschwindigkeiten. Wie bereits in Kapitel 2.2.1.3 in Gleichung 2.19 ausgeführt, lässt sich die rotatorische Bewegung des Roboters aus zwei verschiedenen Komponenten der Winkelgeschwindigkeit zusammensetzen. Dies wird nun dazu benützt die gewünschte Bewegung des Roboters zu definieren. Mittels der Gleichung 4.30 lässt sich die Winkelgeschwindigkeit ω_c als Winkeländerung $\Delta\beta$ des Roboters bezüglich der Trajektorie, während der Zeitdifferenz Δt beschreiben. Die beiden Differenzen sind dabei zwischen dem nächsten und dem zuletzt befahrenen Nutzerpunkt definiert. Die Winkelgeschwindigkeit einer Punktmasse entlang der vorgegebenen Trajektorie ω_{traj} lässt sich, wie in Gleichung 4.31 zu sehen ist, beschreiben als die Geschwindigkeit an einem Zwischenpunkt $v_{i+1\to i,n}$, multipliziert mit der Krümmung des gerade befahrenen Trajektoriestückes, zwischen zwei Nutzerpunkten P_{i+1} und P_i. Da sowohl ω_{traj} als auch ω_c bestimmt sind, lässt sich durch Umstellen von Gleichung 2.19 die Winkelgeschwindigkeit des Roboters ω_b bestimmen. Diese kann nun, unter Berücksichtigung einer geeigneten Anfangsbedingung (Lage des Roboters zu Beginn), integriert werden um φ zu erhalten, was aus Gleichung 4.32 ersichtlich wird.

$$\omega_c = \frac{\Delta\beta}{\Delta t}\ ,\ \Delta\beta = \beta_{i+1} - \beta_i\ ,\ \Delta t = t_{i+1} - t_i \tag{4.30}$$

$$\omega_{traj} = v_{i+1\to i,n} \cdot \rho_{i+1\to i} \tag{4.31}$$

$$\varphi = \left(\int_{t_{i+1\to i,n}}^{t_{i+1\to i,n+1}} \omega_b \right) + \varphi_0 \tag{4.32}$$

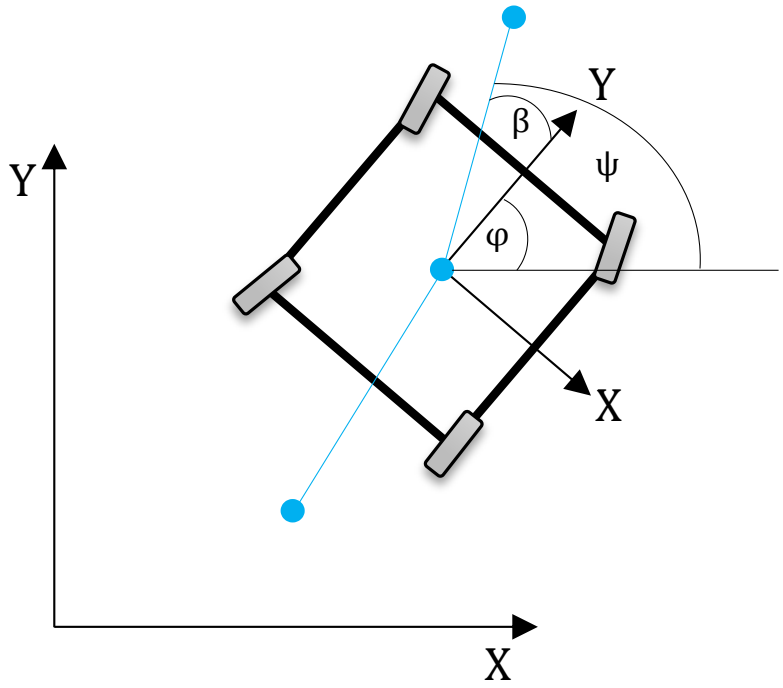


Abbildung 4.16.: Winkelbeziehungen des Roboters zum globalen Koordinatensystem und zur Trajektorie

Die Vor- und Nachteile beider Varianten werden in Kapitel 4.4 näher erläutert. Im weiteren Verlauf diese Kapitels wird nun die erste, der beiden Varianten, näher betrachtet. Das MATLAB-Simulink Modell diesbezüglich ist in der Abbildung 4.17 zu sehen. Die beiden Winkel ψ und φ (im Modell als *psi* und *phi_target* bezeichnet) werden abgeleitet und für die in Anhang B.3.1 erläuterte Berechnung von *omega_ratio* verwendet. Die Variable *phi_target* wird, zusammen mit dem vom simulierten Fahrzeug erhaltenen momentanen Winkel des Roboteraufbaus *phi_current*, als Input für den implementierten Sliding-Mode Regler verwendet.

Die Abbildung 4.18 zeigt den Aufbau des in Abbildung 4.17 zu sehenden Blockes *Sliding_Mode_Controller*. Die beiden Variablen *phi_target* und *phi_current* werden mit einem PT1-Glied, dessen Zeitkonstante T_{csim} dem Berechnungsintervall Δt_{step} entspricht, verzögert und voneinander abgezogen, siehe Gleichung 4.33. Dabei entspricht die Variable $e(t)$ dem Regelfehler, die Variable *phi_target* steht für die Führungsgröße und *phi_current* entspricht der Regelgröße. Diese muss messbar sein, was sowohl bei dem simulierten Fahrzeug, als auch bei der späteren realen Umsetzung der Fall ist. Übersteigt der Betrag des Regelfehlers $e(t)$ einen festgelegten Wert (die Variable *max_phi_error*),

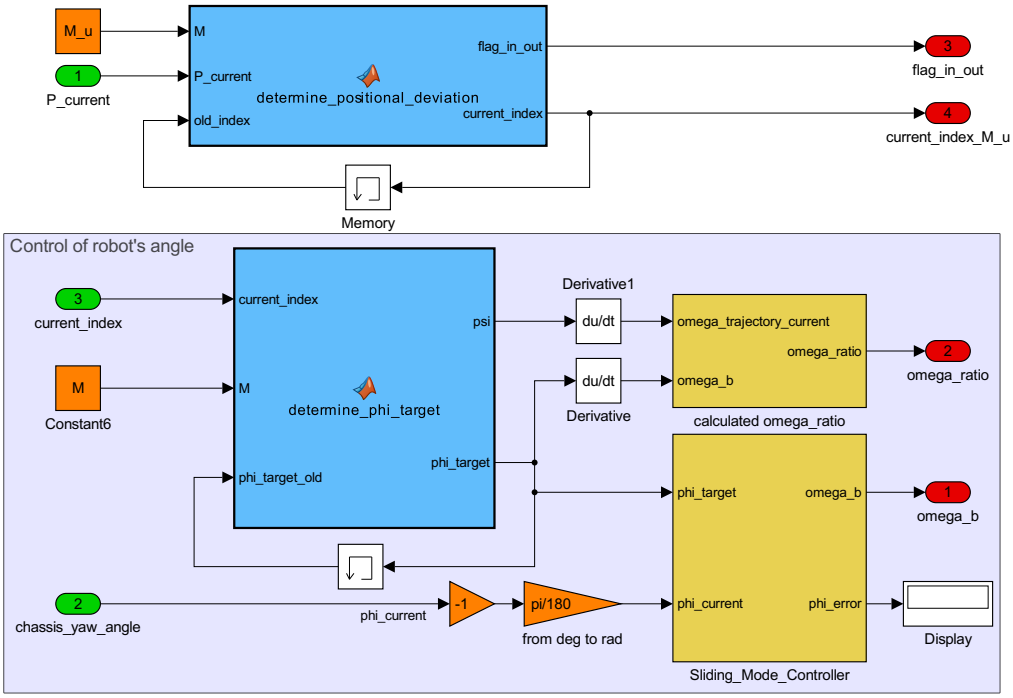


Abbildung 4.17.: Ablauf für die Bestimmung des Robotersaufbauwinkels

wird ein Fehler gemeldet, da im späteren realen Betrieb ein technisches Gebrechen oder ähnliches vorliegen könnte.

$$e(t) = phi\_target - phi\_current \tag{4.33}$$

Die Sliding-Mode Regelung ist nun in Abbildung 4.19 dargestellt und entspricht dem Block *Sliding_Mode_Control* in Abbildung 4.18. Der relative Grad des Systems ergibt sich zu eins, da der Ausgang des Sliding-Mode Reglers (Winkelgeschwindigkeit des Roboteraufbaus) einmal differenziert werden muss, um zu der Regelgröße (Winkel des Roboteraufbaus) zu gelangen. Es wird also ein Sliding-Mode Regler erster Ordnung verwendet, dessen grundsätzliche mathematische Beschreibung in Gleichung 4.34 zu finden ist. Die konstante U wurde durch das Befahren verschiedener Szenarien, siehe Kapitel 4.4, bestimmt und muss positiv sein.

$$omega\_b = U \cdot sgn(e) \tag{4.34}$$

Da die Verwendung der Vorzeichenfunktion zu einem starken *chattering* (Rattern des Reglerausganges mit hoher, endlicher Freqenz)führt, ist das Ausgangssignal *omega_b* für die weitere Verwendung unbrauchbar. Abhilfe schafft die Annäherung des Vorzeichenwechsels von e mit Hilfe von Gleichung 4.35. Die Variable *epsilon* bestimmt dabei die Genauigkeit des Reglers. Umso größer diese gewählt wird, umso geringer das *chattering*-Phänomen, jedoch steigt damit auch die Ungenauigkeit des Reglers. Bei der Wahl des Parameters *epsilon* wurde vor allem auf die Qualität des Ausgangssignals *omega_b* geachtet, siehe Kapitel 4.4.

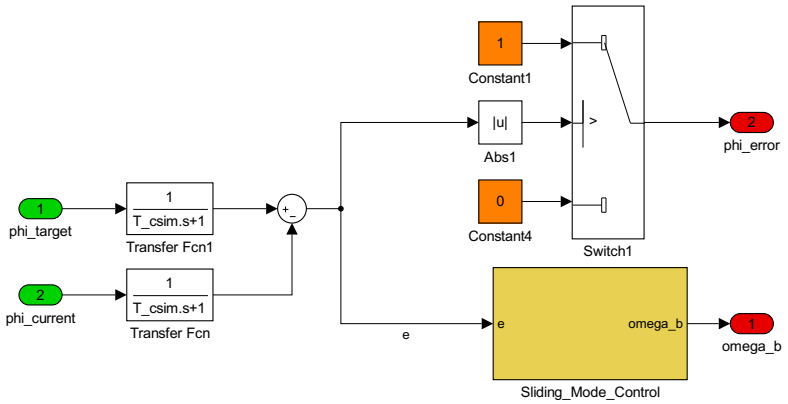


Abbildung 4.18.: Sliding Mode Controller für die Regelung des Roboteraufbauwinkels und Fehlerüberprüfung für den Aufbauwinkel

$$omega\_b = U \cdot \frac{e}{|e| + epsilon} \tag{4.35}$$

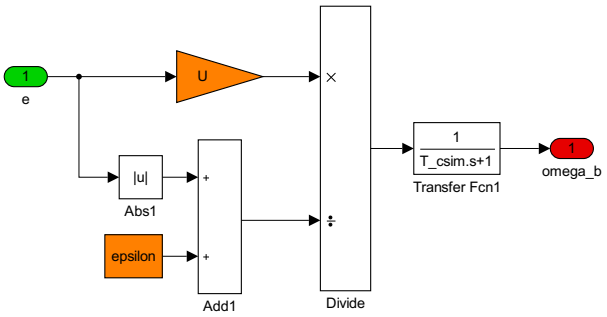


Abbildung 4.19.: Sliding Mode Controller für die Regelung des Roboteraufbauwinkels

4.3.5. Modul Motion-Controller

Die Aufgabe des Blockes *Motion-Controller* ist es, aus dem berechneten Twist des Roboters, die erforderlichen Lenk- und Antriebsmomente abzuleiten, um die gewünschte Bewegung auszuführen. Als weiteren Input kann auf die aktuellen Lenkwinkeln und Raddrehzahlen aller Räder zurückgegriffen werden. Der prinzipielle Aufbau des Blockes ist in Abbildug 4.20 zu sehen. Aus dem Robotertwist und den aktuellen Lenkwinkeln und Raddrehzahlen werden, die für die gewünschte Bewegungen notwendigen Lenkwinkel und Raddrehzahlen berechnet und daraus die erforderlichen Lenk-, sowie Antriebsmomente, bestimmt.

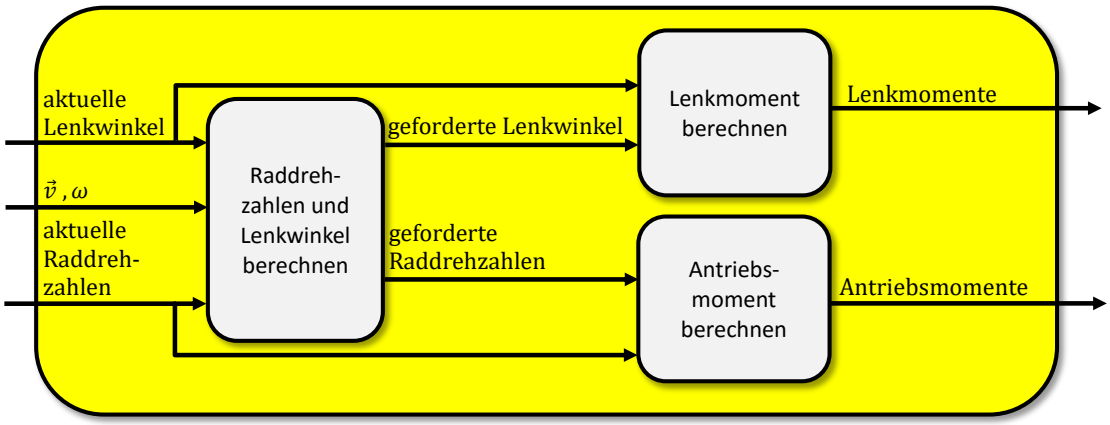


Abbildung 4.20.: Aufbau des Blockes *Motion-Controller*

4.3.5.1. Bestimmung der erforderlichen Lenkwinkel und Raddrehzahlen

Die Aufgabe aus einem gegebenen Twist auf die Lenkwinkel und Raddrehzahlen zu schließen entspricht der Bestimmung der inversen Kinematik aus Kapitel 2.2.1.4.

Ausgegangen wird von der Situation in Abbildung 4.21, in der ein Geschwindigkeitvektor $\overrightarrow{v}$ gegeben, sowie das Roboterkoordinatensystem (x, y) und das Radkoordinatensystem (x_{wheel}, y_{wheel}) zu sehen sind. Zusätzlich wird in Abbildung 4.22 eine Winkelgeschwindigkeit $\overrightarrow{\omega}$ als zweite Komponente des Twists betrachtet. Diese erzeugt am betrachteten Rad den Vektor $\overrightarrow{\omega \cdot r}$, der Abstand r versteht sich als die Distanz zwischen Roboterkoordinatensystem-Ursprung und dem Radmittelpunkt. Mittels Gleichung 4.36 kann eine Vektoraddition erfolgen und der resultierende Geschwindigkeitsvektor am Rad $\overrightarrow{v_{res}}$ bestimmt werden.

$$\overrightarrow{v_{res}} = \overrightarrow{v} + \overrightarrow{\omega \cdot r} \tag{4.36}$$

Der Lenkwinkel δ liegt zwischen $\overrightarrow{v_{res}}$ und der y-Achse des Radkoordinatensystems. Die Raddrehzahl ω kann, ident zu der Vorgehensweise in Gleichung 2.32, mittels Gleichung 4.37 aus Kenntnis des resultierenden Geschwindigkeitsvektors $\overrightarrow{v_{res}}$ am Rad und dem Radradius r_{wheel} (vereinfacht wurde der undeformierte Radius verwendet) berechnet werden.

$$\omega = \frac{\overrightarrow{v_{res}}}{r_{wheel}} \tag{4.37}$$

Im Anhang ist in der Abbildung B.4 das MATLAB-Simulink Modell für die Bestimmung der erforderlichen Lenkwinkel und Raddrehzahlen zu sehen. Es erfolgt die eben ausgeführte Berechnung, allerdings mit einigen zusätzlichen Überprüfungen. Mittels der Variablen *trigger2* wird ständig überprüft, ob der Twist des Roboters null sein soll, beispielsweise weil ein Stehenbleiben das Roboters gefordert ist. Dieser Fall führt zu einem Problem, da der Lenkwinkel nicht mehr bestimmt werden kann. Wird

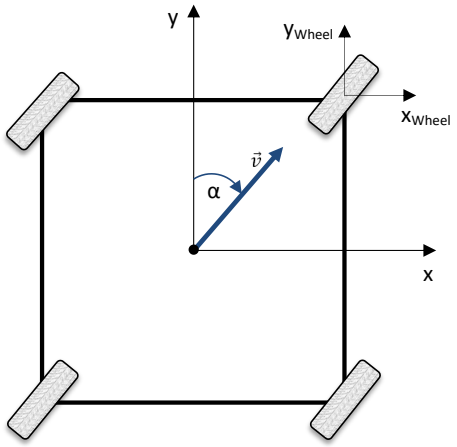


Abbildung 4.21.: Roboterbewegung ohne Rotation des Aufbaus

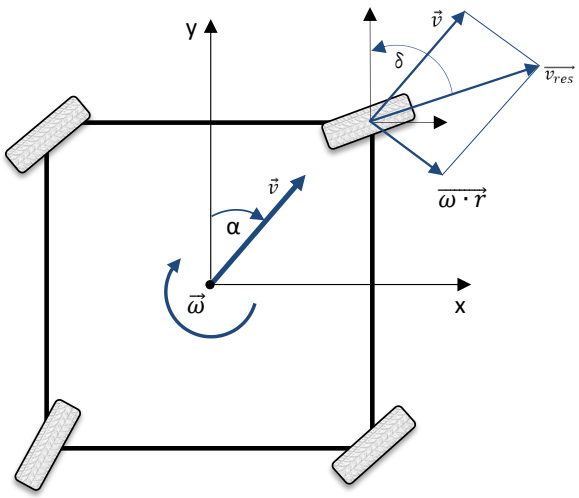


Abbildung 4.22.: Berechnung des Geschwindigkeitsvektors am Rad

nun also *trigger2* zu null, wird auf einen vorhergegangenen Wert für den Lenkwinkel zurückgegriffen, der einige Zeitschritte zurück liegt und bei dem die Geschwindigkeit noch nicht zu null gefordert wurde. Somit wird auch ein Stehenbleiben durch eine Vorgabe der Sollgeschwindigkeit von null ermöglicht, ohne auf die zusätzliche implementierte proprotionale Regelung für das Stehenbleiben zurückzugreifen. Für die Raddrehzahl ergeben sich dadurch keine Einschränkungen.

Der gewünschte Lenkwinkel wird umgerechnet und auf die y-Achse des Radkoordinatensystems bezogen, sodass eine Geradeausfahrt einem Lenkwinkel von null Grad entspricht. Dem Roboter steht ein eingeschränkter Lenkwinkelbereich zur Verfügung. Wird dieser zu Überschreiten versucht, gibt die Simulation einen Fehler aus. Um auf Lenkwinkelsprünge nicht mit zu starken Lenkbewegungen zu reagieren, wird der Lenkwinkel noch mittels einem PT1-Glied mit der Zeitkonstante *T_steering_angle* verzögert.

Für jeden vorgegebenen Lenkwinkel gibt es, Lenkwinkelbeschränkungen außen vor gelassen, zwei mögliche Stellungen des Rades, siehe Abbildung 4.23. Die Punkte A und B sollen diesen Umstand symbolisieren. Anders formuliert gibt es immer einen kurzen und einen langen (bezogen auf die Lenkwinkeldifferenz) Weg, damit das Rad eine gewünschte Lenkwinkelstellung erreicht. Es wird nun überprüft, wie groß die Differenz zwischen dem momentanen und dem geforderten Lenkwinkel ist. In der Abbildung 4.23 entspricht dies dem Winkel φ_2. Der momentane Lenkwinkel entspricht hierbei, in diesem Fall bezogen auf die Achse x_{wheel}, dem Wert φ_1. Ist nun der Differenzwinkel größer als $\frac{\pi}{2}$, wird der geforderte Winkel umgerechnet, sodass das Rad um den Winkel φ_3 vom Punkt A zu A' gedreht wird. Somit können auch große Richtungsänderungen des Geschwindigkeitsvektors, beispielsweise durch Änderung von vorwärts zu rückwärts fahren, bearbeitet werden, welche auf Grund der herrschenden Lenkwinkelbeschränkung nicht möglich wären.

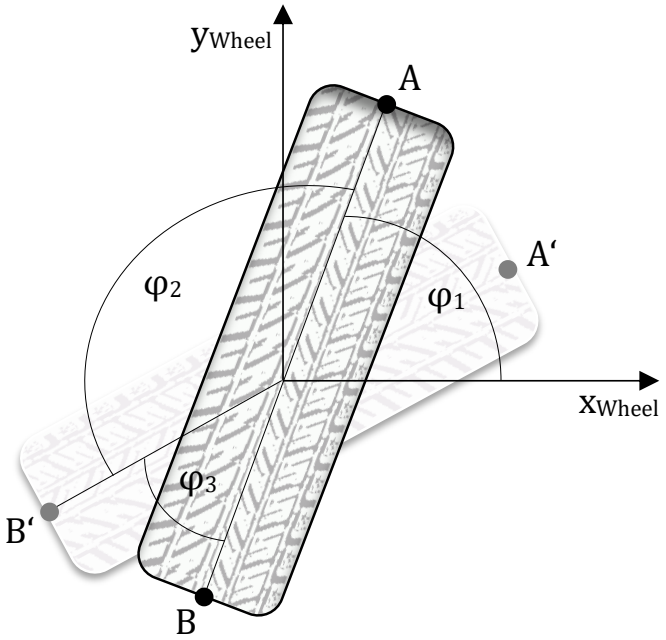


Abbildung 4.23.: Überlegungen bezüglich möglicher Radstellungen

4.3.5.2. Regelung des Lenkwinkels

Für die Regelung des Lenkwinkels wird auf eine einfache Modellbildung des Rades zurückgegriffen. Es wird als undeformierbar angenommen und lässt sich mittels Drallsatz in Gleichung 4.38 beschreiben. J_δ entspricht dem Trägheitsmoment um die z-Achse des Rades, $M_{\delta i}$ ist das gesuchte Lenkmoment und $M_{r\delta i}$ das Rückstellmoment. Das resultierende Moment $J_\delta \cdot \ddot{\delta}_i$ wird mittels Sliding-Mode Regler ermittelt, das nicht durch Messung zur Verfügung stehende Rückstellmoment $M_{r\delta i}$ wird über einen Luenberger-Beobachter bestimmt, somit lässt sich das gesuchte Lenkmoment $M_{\delta i}$ berechnen. Das erstellte Beobachter-Modell für das Rückstellmoment ist in Anhang B.4.2 beschrieben.

$$\ddot{\delta}_i = \frac{1}{J_\delta} \cdot (M_{\delta i} - M_{r\delta i}) \tag{4.38}$$

Wird die Regelabweichung zwischen aktuellem und geforderten Lenkwinkel zu groß, wird ein Fehler ausgegeben, ebenso wenn das Lenkmoment außerhalb der realisierbaren Grenzen liegt. In beiden Fällen kann beispielsweise ein Lenkmotor blockiert sein und eine sichere Bewegung ist nicht mehr möglich.

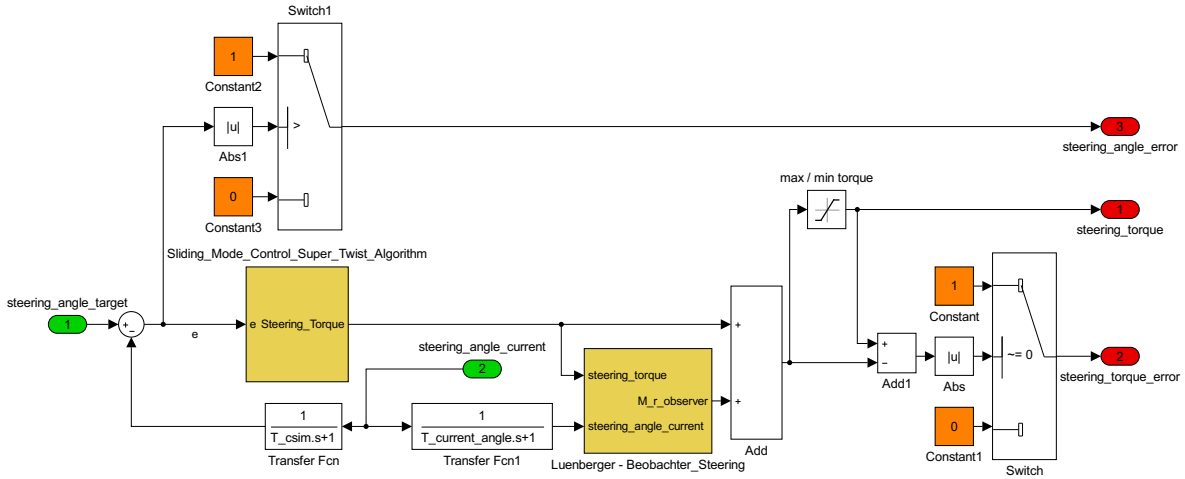


Abbildung 4.24.: MATLAB-Simulink Block zur Berechnung des Lenkmomentes

Die Sliding-Mode Regelung wird, in Anlehnung an Bartolini et al. (1999), mit einem Super-Twisting Algorithmus ausgeführt und ist in Abbildung 4.25 zu sehen. Die Variable *sigma* wird dabei nach Gleichung 4.40 bestimmt, es liegt ein System mit relativem Grad zwei vor, somit wurde eine Sliding-Mode Regelung zweiter Ordnung verwendet. Der Regelfehler e wird dabei durch die Gleichung 4.39 beschrieben, *steering_angle_current* steht für den momentanen Lenkwinkel, *steering_angle_target* für den geforderten. Die bestimmende Gleichung für das gesuchte resultierende Lenkmoment, in der Abbildung 4.25 als *Steering_Torque* bezeichnet, findet sich in 4.41. Die

Variable *I_zz_wheel* bezeichnet dabei das Trägheitsmoment um die z-Achse, die Parameter *lambda_s*, *W_s* sowie der Parameter p von Gleichung 4.40, können frei gewählt werden.

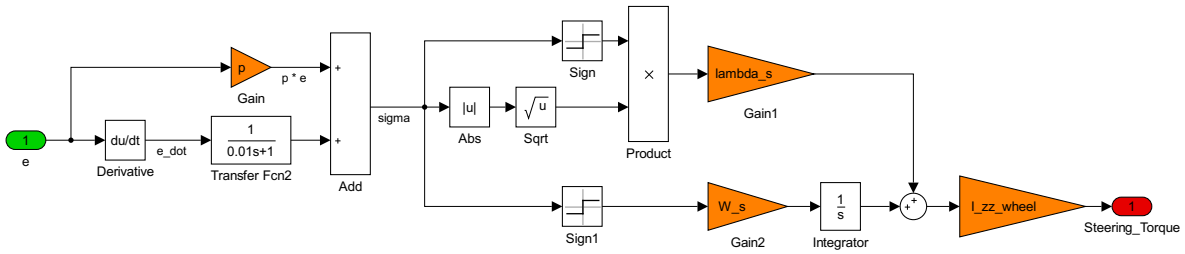


Abbildung 4.25.: Sliding-Mode Regler mit Super Twist Algorithm

$$e = steering\_angle\_target - steering\_angle\_current \tag{4.39}$$

$$sigma = \sigma = p \cdot e + \dot{e} \tag{4.40}$$

$$Steering\_Torque = I\_zz\_wheel \cdot \left(lambda\_s \cdot sgn(\sigma) \cdot \sqrt{|\sigma|} + \int_0^t W\_s \cdot sgn(\sigma) d\tau \right) \tag{4.41}$$

4.3.5.3. Regelung der Antriebsdrehzahl

Die Regelung der Antriebsdrehzahl erfolgt mit dem gleichen Modell wie bei dem Lenkwinkel. Die Gleichung 4.42 beschreibt den Drallsatz bezüglich der Antriebsachse des Rades. Es gilt das Antriebsmoment $M_{\omega i}$ zu bestimmen, dabei wird das resultierende Moment $J_\omega \cdot \dot{\omega}_i$ mittels PI-Regelung mit Anti-Windup bestimmt, das Rückstellmoment $M_{r\omega i}$ liefert ein Luenberger-Beobachter, siehe Anhang B.4.3.

$$\dot{\omega}_i = \frac{1}{J_\omega} \cdot (M_{\omega i} - M_{r\omega i}) \tag{4.42}$$

Das MATLAB-Simulink Modell für die Berechnung des Antriebsmoments ist in Abbildung 4.26 zu sehen. Wie bereits beim Lenkmoment, wird auch hier ein Fehler ausgegeben, wenn die Differenz in den Raddrehzahlen zu groß wird, sowie wenn das Antriebsmoment die vorgegebenen Grenzen überschreitet, jedoch wird im Falle des

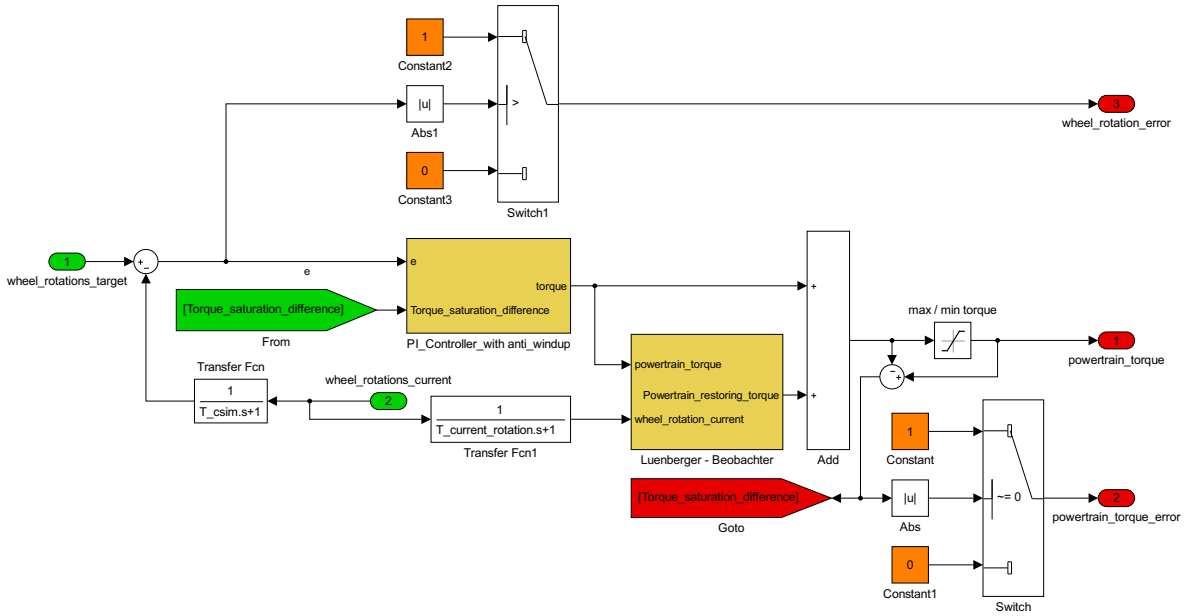


Abbildung 4.26.: MATLAB-Simulink Modell für die Berechnung des Antriebsmoments

Antriebsmoments die Simulation nicht gestoppt, es könnte auch eine für den Roboter nicht bewältigbare Beschleunigung gefordert werden.

Die für die Bestimmung des resultierenden Moments notwendige PI-Regelung, ist in Abbildung 4.27 ersichtlich. Der Regelfehler ist in Gleichung 4.43 zu sehen, die Variablen *wheel_rotations_target* und *wheel_rotations_current* stehen für die geforderte, sowie die momentane Raddrehzahl. Das resultierende Lenkmoment, in diesem Fall als *torque* bezeichnet, wird durch die Gleichung 4.44 beschrieben. Die Parameter für den proportionalen und integrierenden Anteil, *P_motion* und *I_motion*, sind frei wählbar. Die Variable *I_yy_wheel* steht für das Trägheitsmoment um die Antriebsachse des Rades. Da die Stellgröße, das Antriebsmoment $M_{\omega i}$, in Abbildung 4.26 als *powertrain_torque* bezeichnet, beschränkt ist, besteht die Gefahr eines Aufwickelns des integrierenden Anteils der Regelung, wenn die Stellgrößenbeschränkung wirksam wird. Nach Horn (2015) wird der integrierende Anteil der PI-Regelung um einen Anteil erweitert, siehe Gleichung 4.45, der dem Aufwickeln bei aktiver Stellgrößenbeschränkung entgegen wirkt. Der zusätzliche Parameter *a_w_motion* kann ebenfalls frei gewählt werden, der Parameter *Torque_saturation_difference* ist die Differenz aus der Stellgrößenbeschränkung und dem Antriebsmoment.

$$e = wheel\_rotations\_target - wheel\_rotations\_current \tag{4.43}$$

$$torque = \left(P\_motion \cdot e + I\_motion \cdot \int_0^t e(\tau) d\tau \right) \cdot I\_zz\_wheel \tag{4.44}$$

$$u_i = I\_motion \cdot \int_0^t e(\tau)d\tau \tag{4.45}$$

$$Anti - Windup : u_i = I\_motion \cdot \int_0^t e(\tau)d\tau + a\_w\_motion \cdot Torque\_saturation\_difference$$

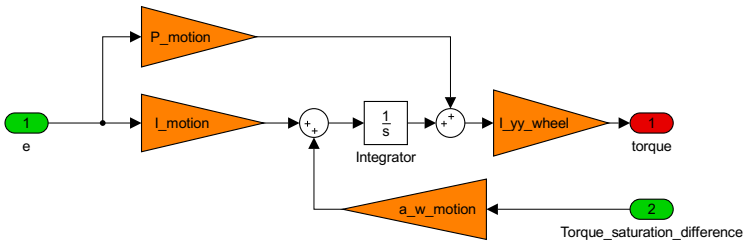


Abbildung 4.27.: PI-Regelung mit Anti-Windup

4.3.6. Modul Adams/Car

Die im Block *Motion-Controller* bestimmten Lenk- und Antriebsmomente dienen nun als Input für das MKS-Modell des Roboters im Block *ADAMS/Car*, siehe Abbildung 4.28. Das Fahrzeug wird durch die wirkenden Momente bewegt und als Output von ADAMS erhält man die Position des Fahrzeuges im ADAMS-Koordinatensystem. Durch Umrechnung auf das definierte globale Roboterkoordinatensystem lässt sich die Roboterposition bestimmen und für den nächsten Berechnungsschritt verwenden. Die Geschwindigkeit in Bewegungsrichtung, sowie normal dazu, erhält man ebenso aus dem MKS-Modell, beide Informationen lassen sich zu einem Geschwindigkeitsvektor verknüpfen. Die für die Regelung des Roboters benötigten Messgrößen Gierwinkel, Lenkwinkel und Raddrehzahlen sind ebenfalls Output des erstellten MKS-Modells und stehen für den nächsten Berechnungsschritt zur Verfügung. Das gesamte MATLAB-Simulink Modell ist in Abbildung 4.29 zu sehen.

4.4. Ergebnisse der Gesamtsimulation

In diesem Abschnitt werden die Ergebnisse der simulierten Szenarien präsentiert. Bei der Wahl dieser Szenarien wurde dabei ein besonderes Augenmerk darauf gelegt, einen möglichst breiten Anwendungsbereich abzudecken. Von einem Szenario mit geringer Geschwindigkeit, bis hin zu Situationen in der Nähe des in Kapitel 3 ausgewiesenen Grenzbereichs und darüber hinaus, ist ein breites Spektrum an Ergebnissen vorhanden.

Bei den Plots der Roboterposition wurde dabei folgende Einteilung getroffen: Eine rote Line besteht aus den einzelnen Punkten der Roboterposition je Zeitschritt. Die

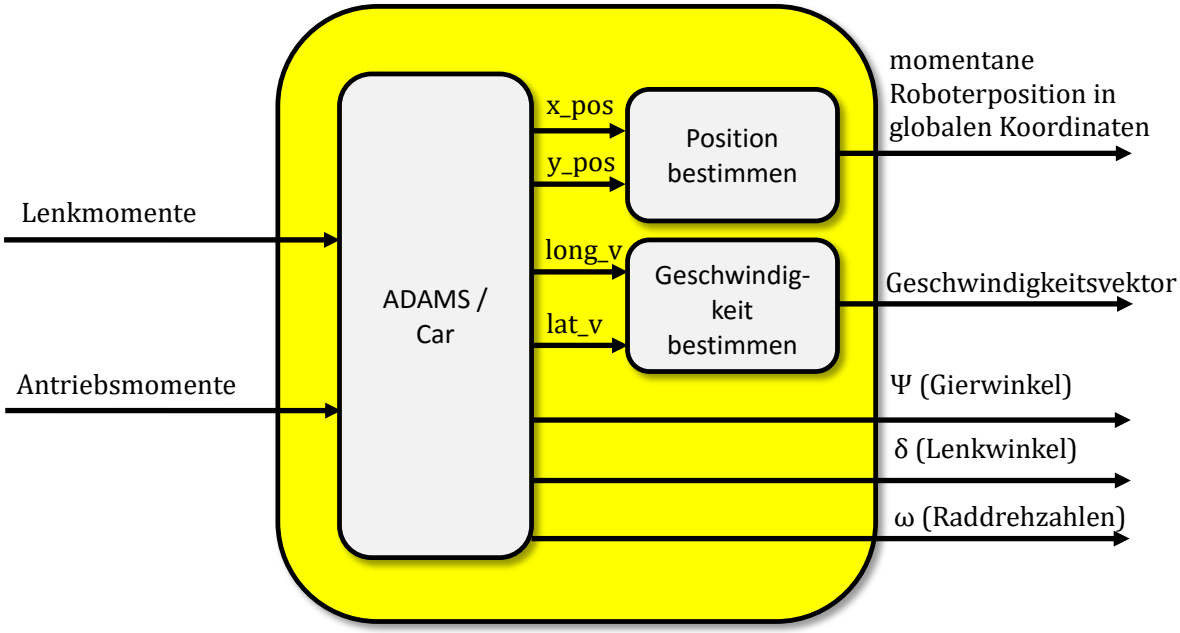


Abbildung 4.28.: Aufbau des Blockes *ADAMS/Car*

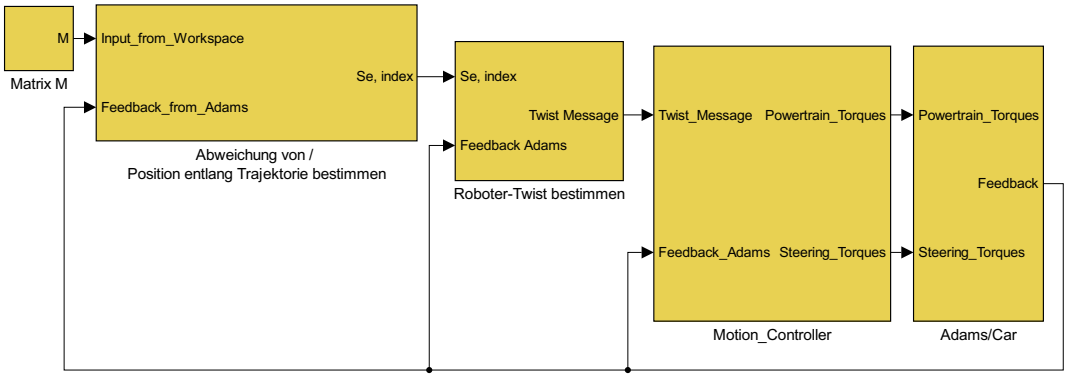


Abbildung 4.29.: Gesamtes MATLAB-Simulink Modell

blauen Punkte sind die vom Nutzer definierten Punkte und dienen zur generellen Definition der Trajektorie, die schwarzen Punkte zwischen den Nutzerpunkten sind die automatisch generierten Zwischenpunkte. Durch die Überdeckung mit der Roboterposition (rote Linie) sind diese meist schlecht zu erkennen, bedeutet im Umkehrschluss jedoch eine hohe Spurtreue des Roboters. Der Roboter selbst wird als Viereck mit einer definierten Orientierung (die Vorderkante ist grün gekennzeichnet) dargestellt, um eine Rotation des Roboteraufbaus erkennen zu können. In manchen Plots ist die Größe des Roboters überproportional dargstellt, um diesen in ausreichendem Maße erkennen zu können.

Für die Unterscheidung der ausgeführten Manöver werden zwei Begriffe eingeführt:

- *PKW-Modus*
  - In diesem Modus bewegt sich der Roboter ähnlich wie ein PKW, es findet jedoch keine Bewegung nach der in Kapitel 2 erwähnten Ackermann-Kinematik statt. Dieser Modus ist lediglich durch folgende Eigenschaft gekennzeichnet: Der Winkel β (Spalte 4 der in Kapitel 4.3.1 behandelten Matrix M), welcher in Abbildung 4.16 zu erkennen ist, wird immer als null festgelegt. Die y-Achse der Roboters (entspricht der Längsachse des Roboters während der Bewegung) versucht sich somit ständig genau entlang der Trajektorie auszurichten und es entsteht eine PKW-ähnliche Bewegung.
- *Roboter-Modus*
  - In diesem Modus findet keine Drehung des Roboteraufbaus statt. Der Winkel φ bleibt somit unverändert zu dem Winkel bei Beginn der Bewegung. Wird der Roboter in diesem Modus betrieben ergeben sich für alle vier Räder immer die gleichen Lenkwinkel, wie auch in Abbildung 2.9 ersichtlich ist.

Diese Aufteilung wurde nur auf Grund der leichteren Unterscheidung getroffen. Das umgesetzte Konzept bietet bezüglich der Bewegungen, solange keine Stellgrößenbeschränkungen verletzt werden (diese stellen jedoch eine Eigenheit des jeweiligen Systems im realen Betrieb dar und sind nicht konzeptbedingt), keine Einschränkungen, bezogen auf die ausgeführten Definitionen in Kapitel 2. Die Kapitel 4.4.5 behandelten Szenarien stellen auch Bewegungen des Roboters vor, welche keinem der beiden Modi zuweisbar sind.

4.4.1. Regelung des Lenkwinkels

Für die Auslegung des in Kapitel 4.3.5.2 vorgestellten Reglers, für den Lenkwinkel eines Rades des Roboters, wird ein Lenkwinkelsprung auf das System aufgebracht, dieser gilt für alle vier Räder zugleich, somit wird der Roboter im *Roboter-Modus* betrieben. Der Roboter wird zunächst auf eine Soll-Geschwindigkeit beschleunigt, siehe Abbildung 4.31, zum Zeitpunkt $t = 5s$ wirkt der Sprung des Lenkwinkels. In Abbildung 4.30 ist dieser Lenkwinkelsprung zu erkennen, ebenso wie die Antwort der Räder darauf. Bei der Auslegung wurde darauf Acht gelegt, möglichst kein Überschwingen und somit instabiles Fahrverhalten zu erzeugen. Das Rückstellmoment von dem entworfenen Luenberger-Beobachter (siehe Anhang B), ist in Abbildung 4.33 erkennbar und bietet einen zufriedenstellenden Verlauf.

Das resultierende Lenkmoment, Ausgang des in Kapitel 4.3.5.2 vorgestellten Sliding-Mode Reglers, ist in Abbildung 4.32 zu sehen.

Hier ist ein leichtes Rauschen erkennbar, welches durch die Ausführungen von Bartolini et al. (1999) erklärbar ist. Bei der Anwendung von Sliding-Mode Reglern mit Super-Twisting Algorithmen auf Systeme mit relativem Grad von zwei, ist ein gänzliches Verschwinden der bekannten Sliding-Mode Regler Problematik (Rattern - *chattering* - der Stellgröße) nicht garantiert. Es wurde dem bereits insofern entgegengewirkt, als das auch in diesem Sliding-Mode Regler die Vorzeichenfunktionen durch eine Näherung (vergleiche Gleichung 4.35, selber Parameter *epsilon*) ersetzt wurde. Auf einen besseren Ersatz der Vorzeichenfunktion, welche keinen Genauigkeitsverlust zur Folge hat, wie von Efimov et al. (2015) beschrieben, wird verzichtet.

Das sich durch Kombination aus resultierendem Moment und Rückstellmoment ergebende Lenkmoment, ist in Abbildung 4.34 zu sehen und dient als Input für das Fahrzeugmodell.

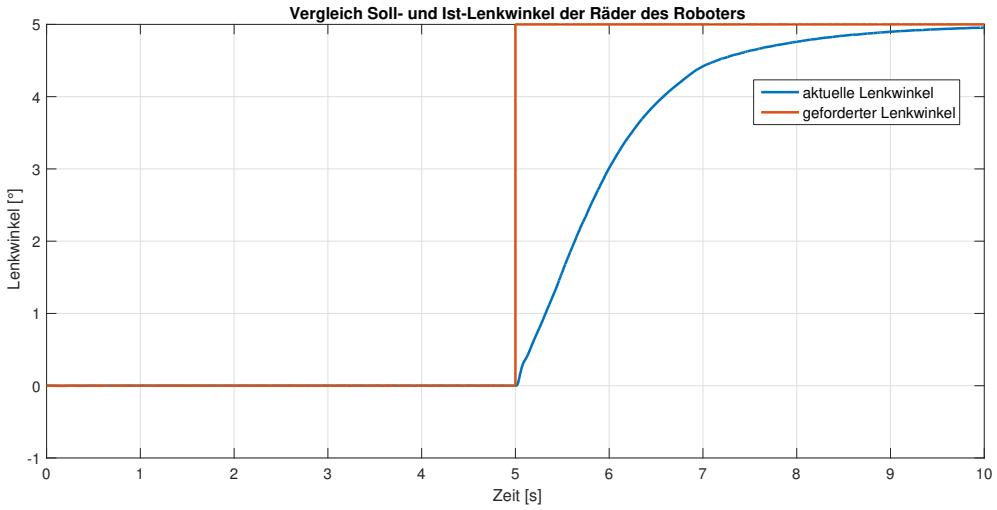


Abbildung 4.30.: Verlauf des Roboterlenkwinkels bei einem Lenkwinkelsprung

4.4.2. Regelung der Antriebsdrehzahl

Für die Auslegung der in Kapitel 4.3.5.3 vorgestellten Regelung der Antriebsdrehzahl, wurde zuerst ein verzögerter Drehzahlsprung auf das System aufgebracht. In der Abbildung 4.35 ist die Antwort der Raddrehzahlen des Roboters zu erkennen. Es werden alle vier Räder gleichzeitig auf die geforderte Drehzahl beschleunigt, die auftretende Gewichtsverlagerung, durch die wirkende Längsbeschleunigung, hat noch keinerlei Einfluss auf die Regelung der Raddrehzahlen. Ein leichtes Überschwingen der Ist- über die Soll-Raddrehzahl wird toleriert, da es zum Einen die Stablität der Bewegung nicht gefährdet und zum Anderen, wie bereits in Kapitel 4.3.4.1 erwähnt, die Soll-Drehzahl, aus der vorgegebenen Geschwindigkeit zu niedrig berechnet wird.

Der Verlauf des Rückstellmomentes, erhalten aus dem Luenberger-Beobachter (Anhang B.4.3), ist in Abbildung 4.36 ersichtlich. Das letzlich für den Antrieb des Roboters

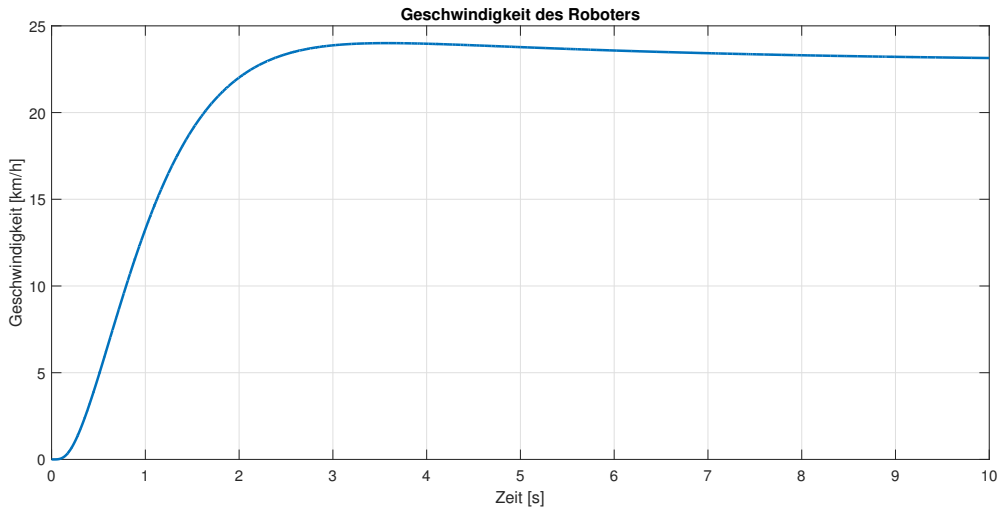


Abbildung 4.31.: Geschwindigkeit des Roboters bei einem Lenkwinkelsprung

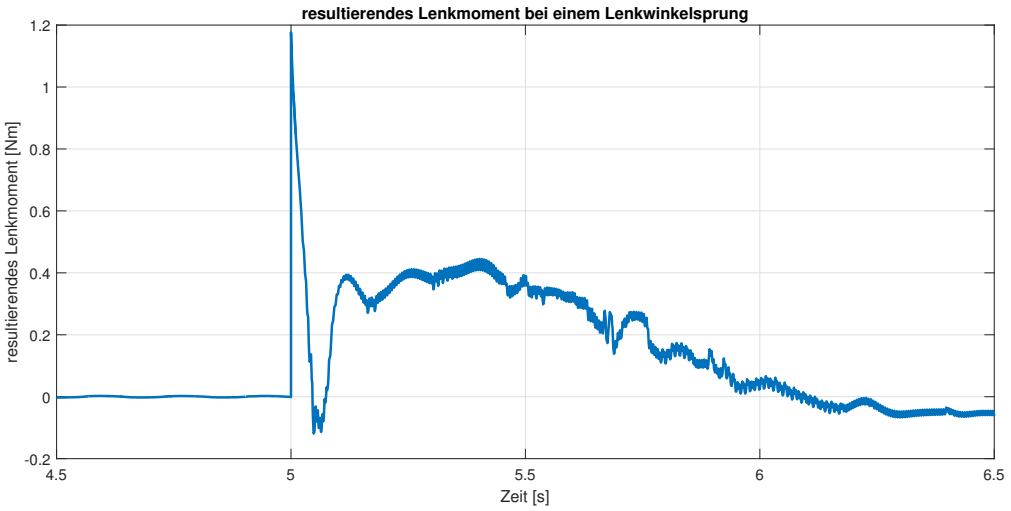


Abbildung 4.32.: Resultierendes Lenkmoment am Rad bei einem Lenkwinkelsprung

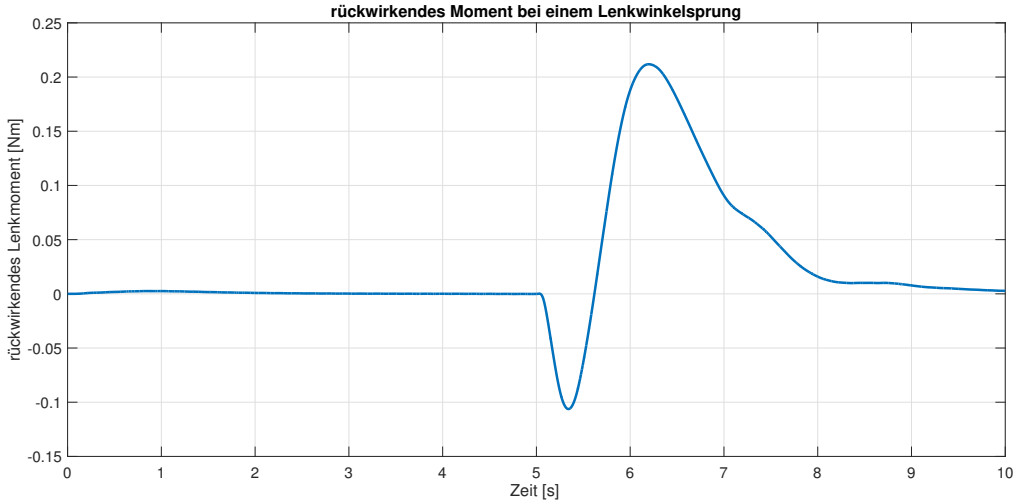


Abbildung 4.33.: Rückstellmoment am Rad bei einem Lenkwinkelsprung

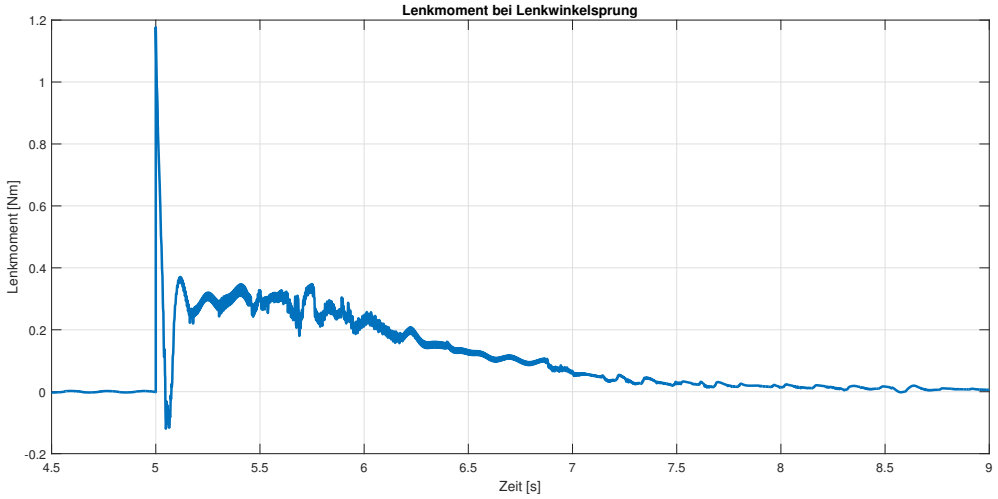


Abbildung 4.34.: Lenkmoment am Rad bei einem Lenkwinkelsprung

verantwortliche Antriebsmoment ist in Abbildung 4.37 zu sehen. Dabei wurde das Drehmoment des Antriebmotors, Abbildung 3.6, bereits mit der Übersetzung des zusätzlich vorhandenen Planetengetriebes multipliziert.

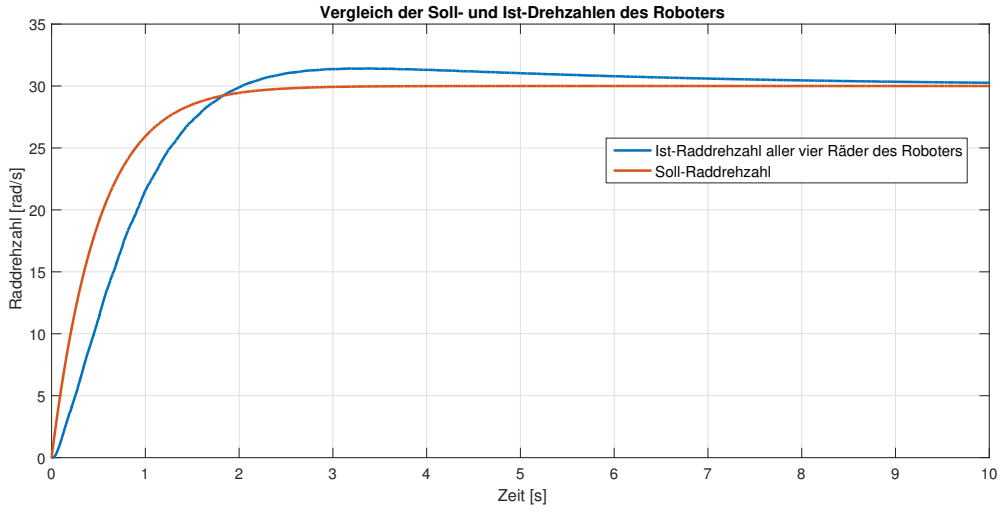


Abbildung 4.35.: Verlauf der Raddrehzahlen der Räder des Roboters

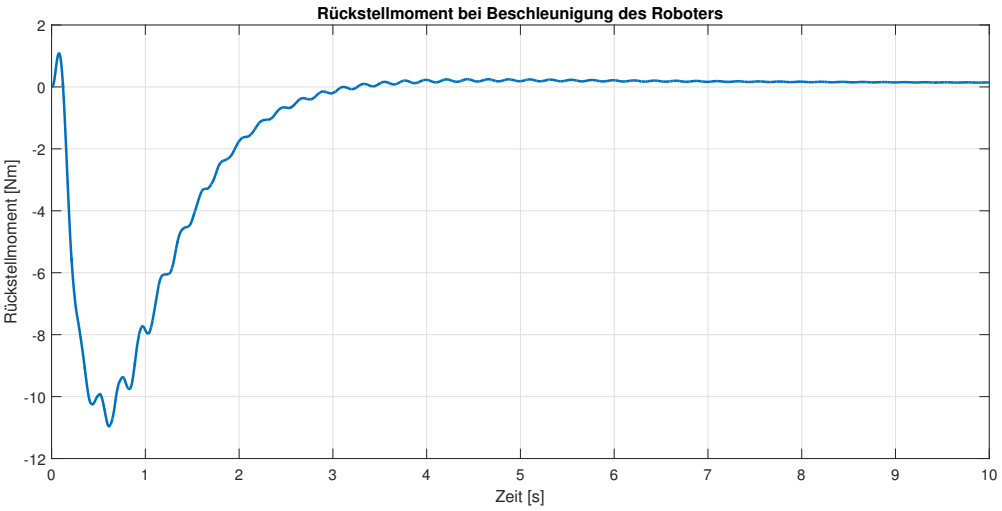


Abbildung 4.36.: Verlauf des Rückstellmomentes eines Rad bei einem Drehzahlsprung

4.4.2.1. Folgen einer zu starken Beschleunigung

Wird ein unverzögerter Drehzahlsprung auf das System aufgebracht, siehe Abbildung 4.38, ist die auftretende Längsbeschleunigung des Fahrzeuges dermaßen stark, dass es zu einer zu großen Gewichtsverlagerung kommt, der Roboter verliert, während des Beschleunigungsvorganges, mehrmals den Bodenkontakt an der Vorderachse. Ersichtlich wird dies durch einen starken Drehzahlanstieg, hauptsächlich an der Vorderachse, in Momenten des Abhebens der Vorderräder. Zu sehen ist dieser Umstand in Abbildung

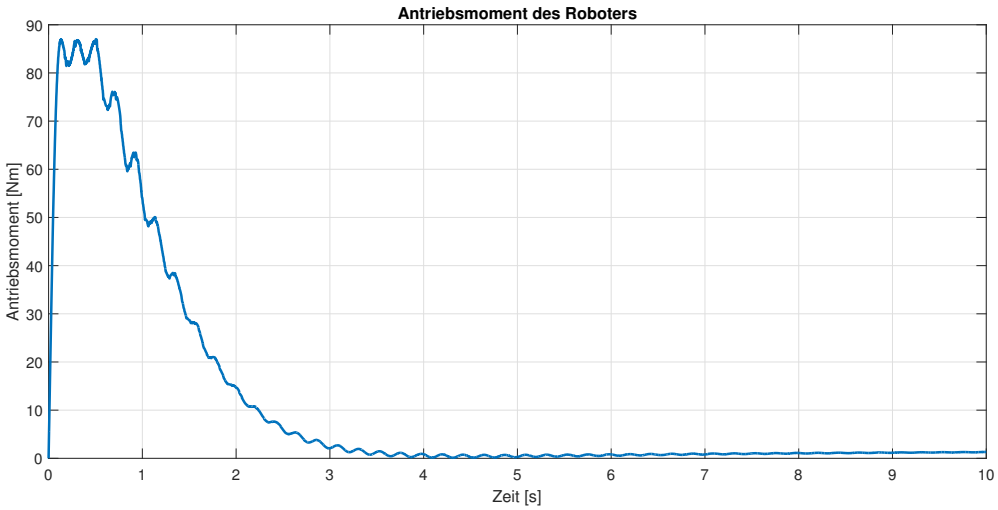


Abbildung 4.37.: Verlauf des Antriebsmoment eines Rad bei einem Drehzahlsprung

4.41, in Momenten, in denen die horizontale Reifenaufstandskraft zu null wird. In diesen Momenten bricht auf Grund des großen Drehzahlanstieges das Antriebsmoment ein, siehe Abbildung 4.39. Der Verlauf der Längsbeschleunigung in Abbildung 4.40, zeigt deutlich, wie sich die Beschleunigungsverläufe bei einem Drehzahlsprung und bei einem verzögerten Drehzahlsprung verhalten. Der unverzögerte Drehzahlsprung führt zu einem sehr instabilen Verhalten, wohingegen der verzögerte Drehzahlsprung einen zufriedenstellenden Verlauf der Längsbeschleunigung liefert.

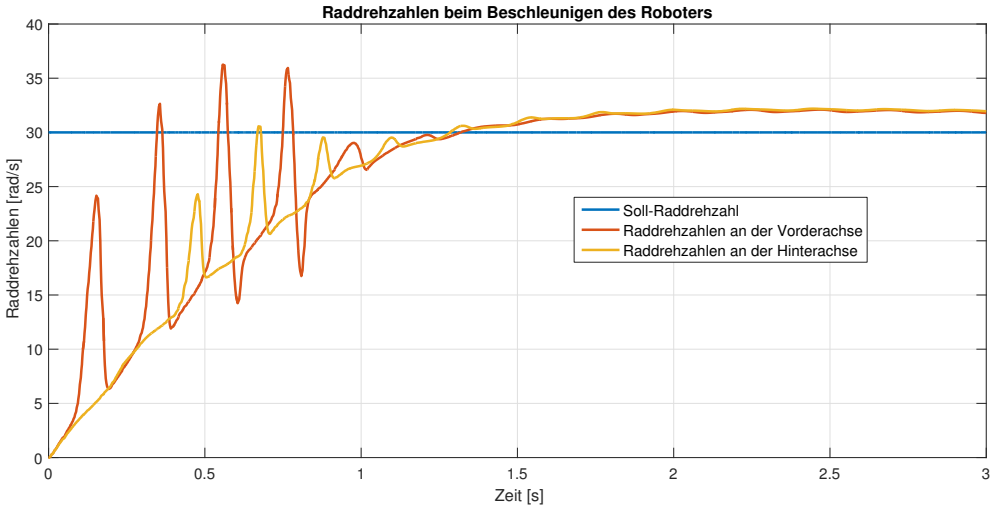


Abbildung 4.38.: Verlauf der Raddrehzahlen am Roboter bei zu starker Beschleunigung

4.4.3. Fahrmanöver: Ecke

In diesem Fahrmanöver soll der Roboter eine Trajektorie mit 90 Grad Winkel (*Ecke*) befahren. Dazu gibt es mehrere Möglichkeiten, zwei davon werden in den folgenden

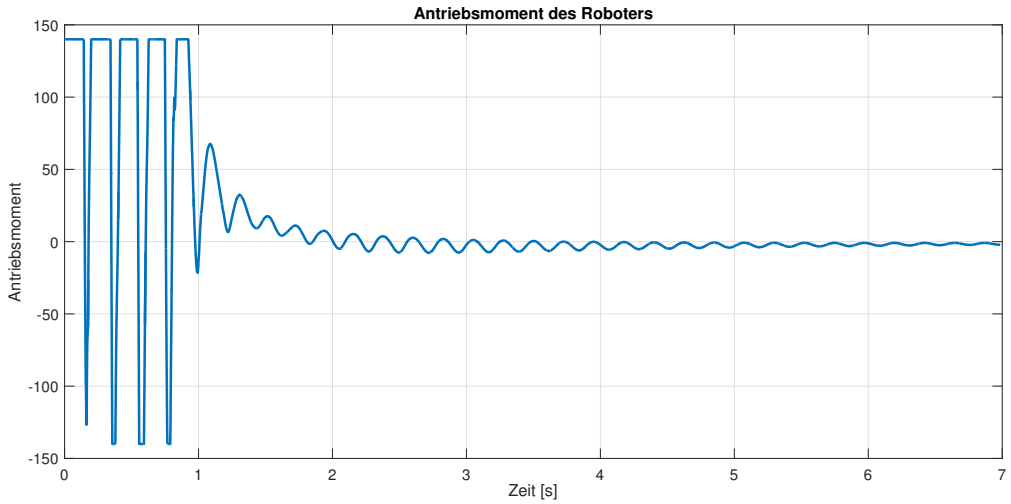


Abbildung 4.39.: Verlauf der Antriebsmomentes an einem Rad an der Vorderachse des Roboters bei zu starker Beschleunigung

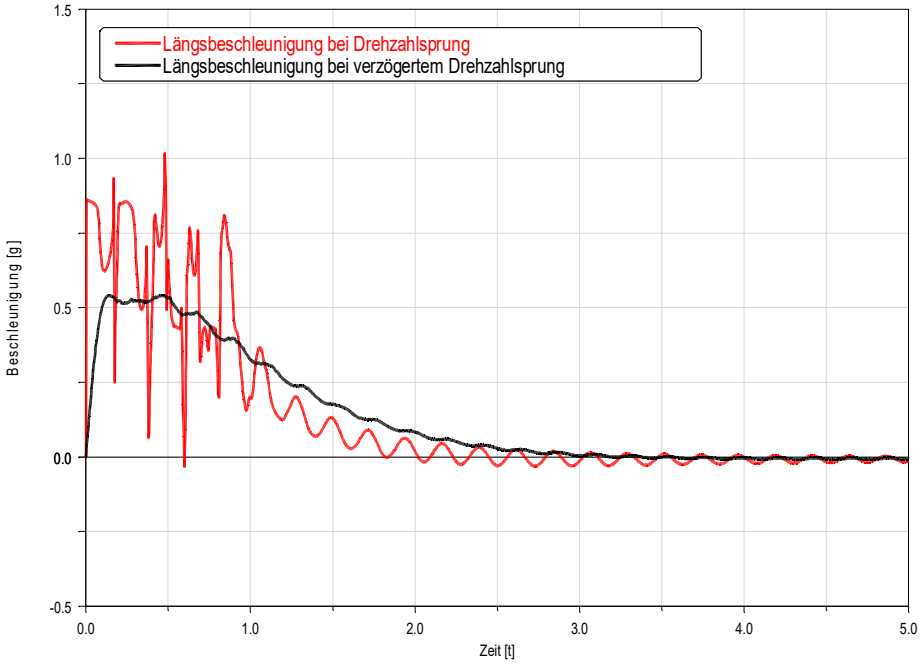


Abbildung 4.40.: Verlauf der Längsbeschleunigung des Roboters bei zu starker und bei normaler Beschleunigung

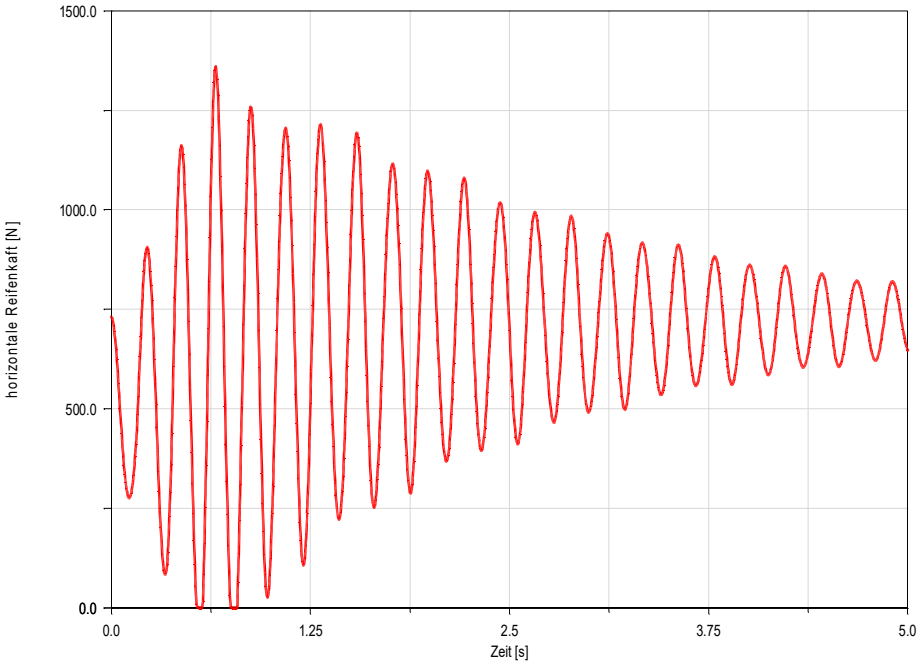


Abbildung 4.41.: Verlauf der Reifenaufstandskraft an einem Rad der Vorderachse bei zu starker Beschleunigung

Kapiteln 4.4.3.1, sowie 4.4.3.2 behandelt. Eine einfache, jedoch nicht weiter behandelte Methode zum Befahren dieser Trajektorie besteht darin, am Eckpunkt stehen zu bleiben und den Roboteraufbau nachzudrehen (*PKW-Modus*), bzw. die Räder neu ausrichten (*Roboter-Modus*). Ein flüssiges Befahren der Trajektorie, aus einem Blickwinkel der benötigten Zeit für das Absolvieren des Weges zwischen Start und Ziel betrachtet, ist diese Methodik jedoch nicht besonderes förderlich.

4.4.3.1. Befahren im PKW-Modus

Eine Trajektorie, welche sich als Ecke beschreiben lässt, ist für das Befahren im PKW-Modus ohne Anpassung ungeeignet. Eine natürliche Bewegung ergibt sich, wenn die Ecke "geschnitten"wird, also als Viertelkreis befahren wird. Wird keine Anpassung an der Trajektorie vorgenommen, setzt die Drehung des Roboters (im Fall der Ecke 90 Grad) erst ein, wenn der Eckpunkt befahren bzw. überfahren wird, siehe dazu auch Abbildung 4.42. Der Eckpunkt in dieser Abbildung befindet sich bei $(x = 15, y = 0)$. Durch die erst bei dem Eckpunkt eingeleitete Drehung würde eine große Abweichung von der Trajektorie stattfinden (ausgenommen der Roboter bleibt am Eckpunkt stehen).

Eine mögliche Abhilfe besteht darin, die Ecke mittels zwei linearen Geradenstücken anzunähern. Auch das Generieren eines Viertelkreises um die Ecke ist möglich. Im Falle der Abbildung 4.42 würde sich der Mittelpunkt bei $(x = 10, y = -5)$ befinden und der Radius 5 Meter betragen. In diesem Fall wird die Ecke mittels linearen Geradenstücken angenähert. Desweiteren wurde, wie in Kapitel 4.3.4.4 ausgeführt, die Berechnung des Winkels φ über die Winkelgeschwindigkeiten durchgeführt, da die Berechnung

über geometrische Zusammenhänge Sprünge im Soll-Winkelverlauf zur Folge hätte (Übergang von einem Geradenstück zum Nächsten).

Die Position des Roboters entlang dieser Ersatztrajektorie ist in Abbildung 4.42 ersichtlich. Bezogen auf die Ersatztrajektorie ergeben sich keine allzu großen Abweichungen (Abbildung 4.43), zur ursprünglichen Trajektorie ist der Abstand dementsprechend größer. Das gesamte Manöver wurde nach der Beschleunigungsphase mit konstanter niedriger Geschwindigkeit ausgeführt, siehe Abbildung 4.44. Durch die Ausführung der Ersatztrajektorie und die erwähnte Bestimmung des Winkels φ, ergibt sich ein linearer Verlauf des Roboterwinkels, um die gewünschte Drehung des Roboters zu erreichen, zu sehen in Abbildung 4.45. Den Vergleich der Winkelgeschwindigkeit des Roboteraufbaus mit der Soll-Winkelgeschwindigkeit zeigt die Abbildung 4.46.

Die Vorschaudistanz, der Abstand zwischem dem Punkt welcher vom Geschwindigkeitsvektor anvisiert wird und dem Roboterschwerpunkt, zu sehen in Abbildung 4.47, wurde so gewählt, dass sich eine möglichst geringe Abweichung von der Trajektorie ergibt und trotzdem noch ein stabiles Fahrverhalten ermöglicht wird.

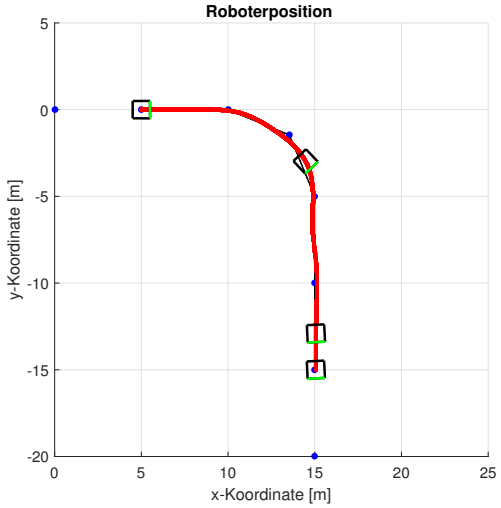


Abbildung 4.42.: Fahrmanöver *Ecke*: Roboterposition während des Befahrens der Ersatztrajektorie im PKW-Modus

4.4.3.2. Befahren im Robotermodus

Ist der Winkel des Roboteraufbaus am Ende der Trajektorie nicht von Bedeutung, bietet sich die Möglichkeit an, den 90 Grad Knick auch im Roboter-Modus zu befahren. Der Aufbauwinkel ändert sich somit nicht und es muss auch keine Ersatztrajektorie gebildet werden. Es wird einzig und allein die Vorschaudistanz etwas vergrößert, siehe Abbildung 4.49, um den Roboter frühzeigt zum Abbiegen zu bewegen. Kurzfristig ist die Abweichung zur vorgegebenen Trajektorie somit etwas größer, wie die Abbildung 4.48 zeigt.

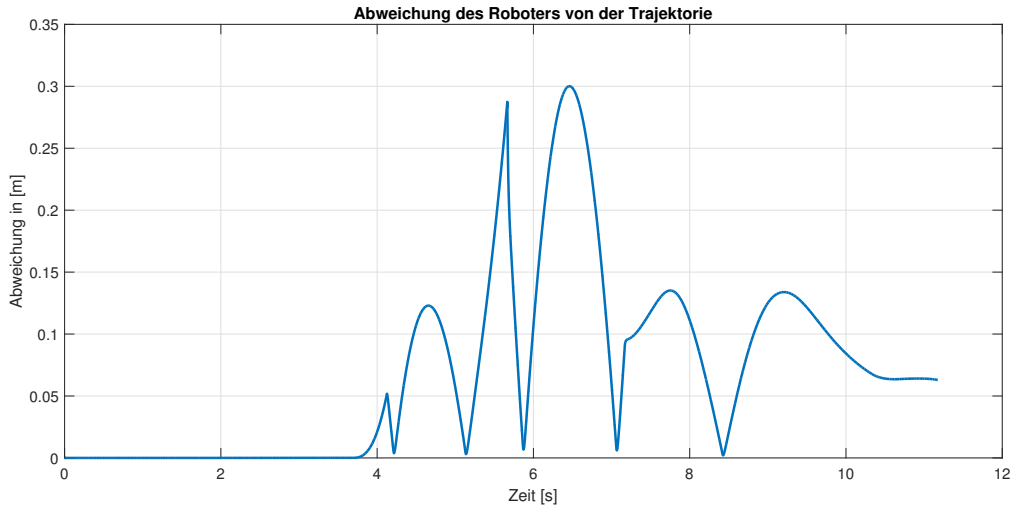


Abbildung 4.43.: Fahrmanöver *Ecke*: Abweichung des Roboters von der Ersatztrajektorie beim Befahren im PKW-Modus

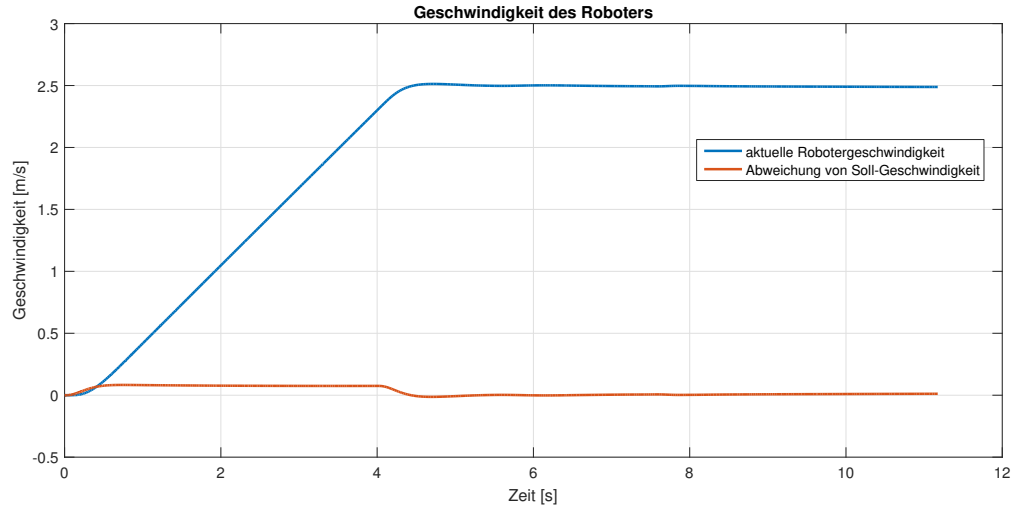


Abbildung 4.44.: Fahrmanöver *Ecke*: Geschwindigkeitsverlauf des Roboters beim Befahren der Trajektorie im PKW-Modus

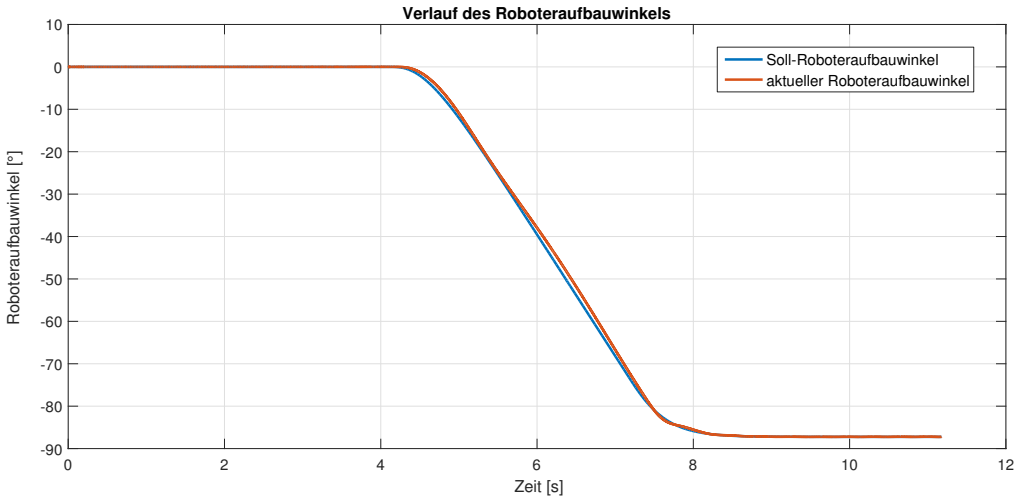


Abbildung 4.45.: Fahrmanöver *Ecke*: Verlauf des Roboteraufbauwinkels beim Befahren der Ersatztrajektorie im PKW-Modus

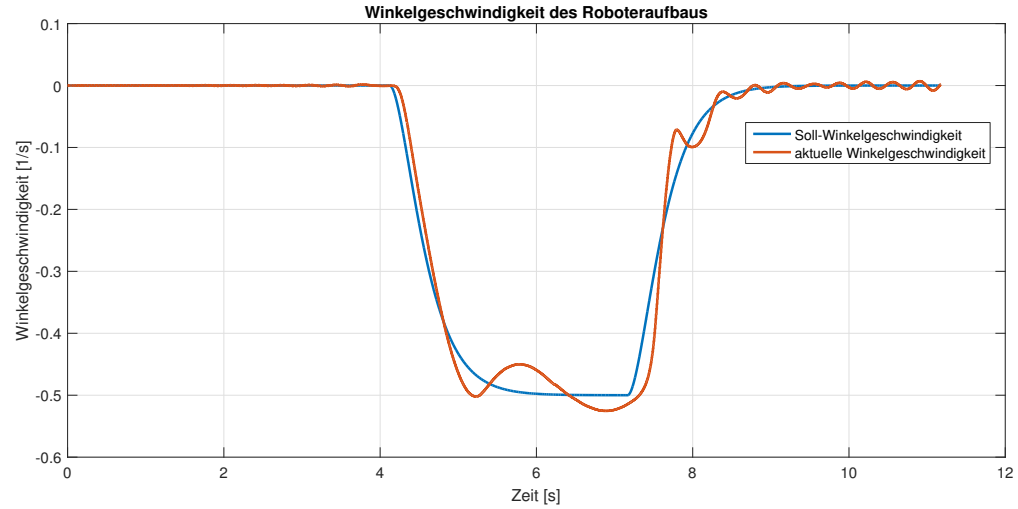


Abbildung 4.46.: Fahrmanöver *Ecke*: Verlauf der Winkelgeschwindigkeit des Roboteraufbaus beim Befahren der Ersatztrajektorie im PKW-Modus

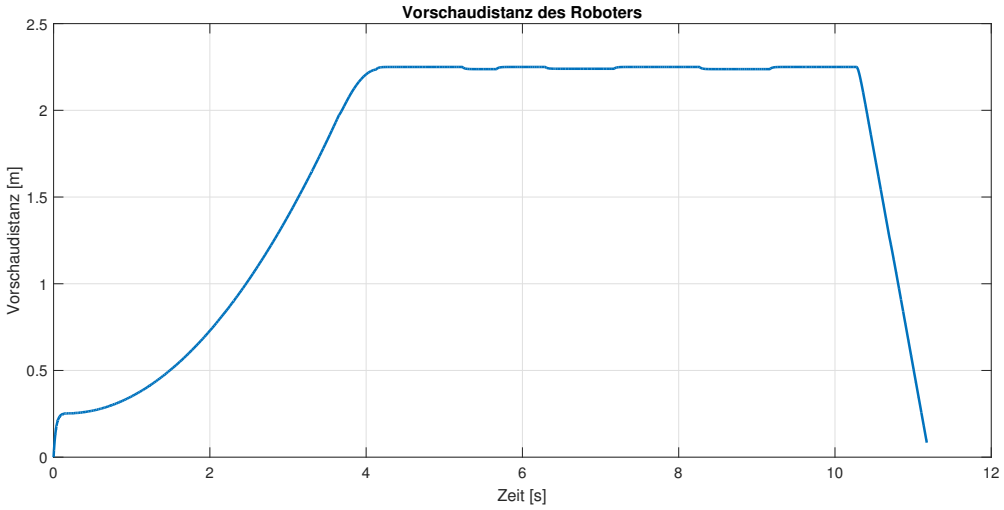


Abbildung 4.47.: Fahrmanöver *Ecke*: Vorschaudistanz des Roboters beim Befahren der Ersatztrajektorie im PKW-Modus

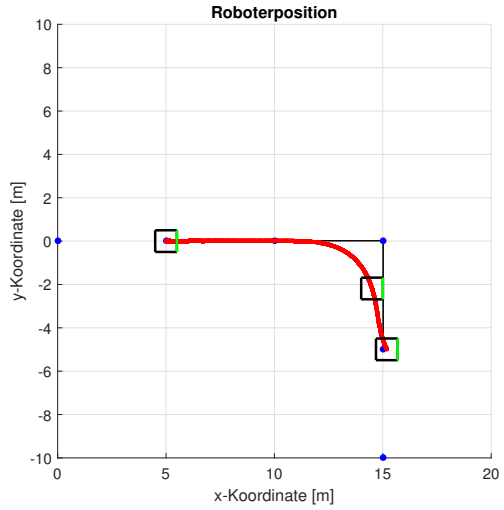


Abbildung 4.48.: Fahrmanöver *Ecke*: Roboterposition beim Befahren der Trajektorie im Robotermodus

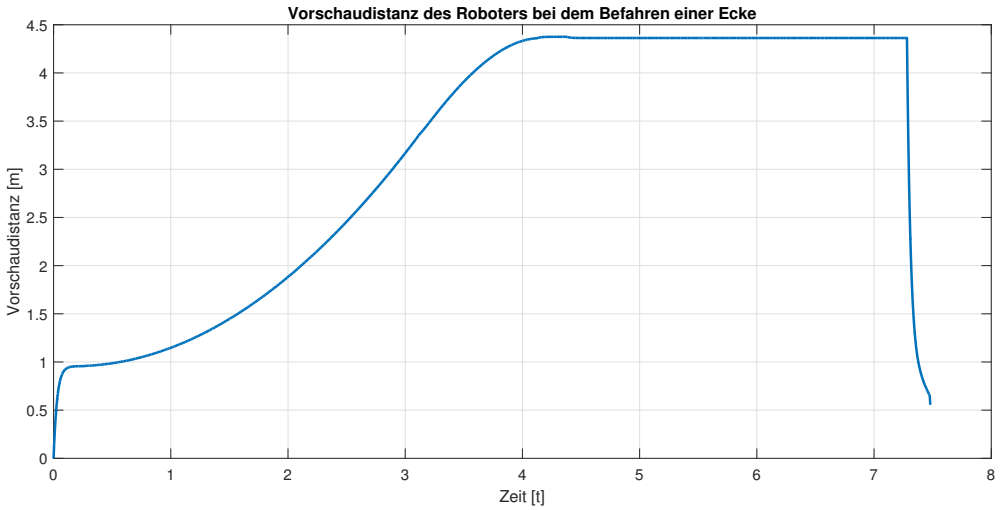


Abbildung 4.49.: Fahrmanöver *Ecke*: Vorschaudistanz des Roboters beim Befahren der Trajektorie im Robotermodus

4.4.4. Fahrmanöver: µ-Split-Bremsung

In diesem Manöver wird der Roboter bewusst in eine Grenzsituation gebracht, um zu sehen wie stabil dieser in solchen Situationen reagiert. Der Roboter wird auf die gewünschte Geschwindigkeit, knapp unter 30 $\frac{km}{h}$, beschleunigt, siehe Abbildung 4.52, um dann mit etwa 0,7 g auf einer ebenen Straße verzögert zu werden, welche auf der rechten Seite (betrifft die Räder rechts vorne und hinten) nur einen Reibkoeffizienten von 0,5 aufweist. Die linke Seite der Fahrbahn besitzt einen Reibkoeffizienten von 1, die Situation lässt sich mit einer Straße vergleichen, auf welcher eine Seite der Fahrbahn nass und die andere trocken ist.

Durch die Verzögerung beginnt sich, auf Grund der unterschiedlich großen Längskräfte an der Rädern, siehe Abbildung 4.54, der Roboteraufbau zu drehen und erreicht Abweichungen von der Nulllage im zweistelligen Bereich, wie in Abbildung 4.55 zu sehen ist. Der Seitversatz des Fahrzeug hält sich im Rahmen, wie die Abbildung 4.51 zeigt, obwohl eine merkbare Querbeschleunigung auf das Fahrzeug wirkt, zu sehen in Abbildung 4.53. Trotz dieser Grenzsituation ist es möglich den Roboter zum Stillstand zu bringen und seine Ausrichtung wieder auf die Ausgangslage zurück zubewegen.

4.4.5. Fahrmanöver: Klothoide

In diesem Manöver befährt das Fahrzeug einen Halbkreis, allerdings nicht mit konstanter Krümmung, da der Übergang zwischen den Geradestücken und dem eigentlichen Kreisradius (konstante Krümmung) mittels einer Klothoide realisiert wurde, siehe Abbildung 4.56. Eine Klothoide bezeichnet dabei eine geometrische Funktion deren Krümmung proportional zur Länge der Funktion (des Bogens) ist und liefert somit einen besseren Übergang von einer Geraden zu einem Kreis, da ein Sprung in der Krümmung verhindert wird, siehe Abbildung 4.57, diese zeigt den gesammten Krümmungsverlauf der Trajektorie.

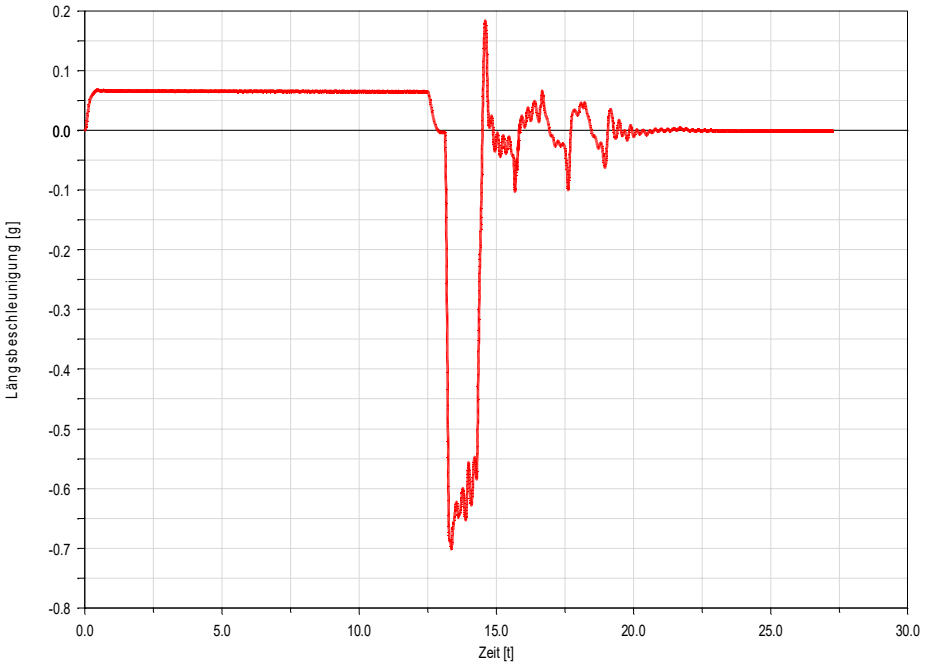


Abbildung 4.50.: Fahrmanöver *µ-Split-Bremsung*: Längsbeschleunigung des Roboters

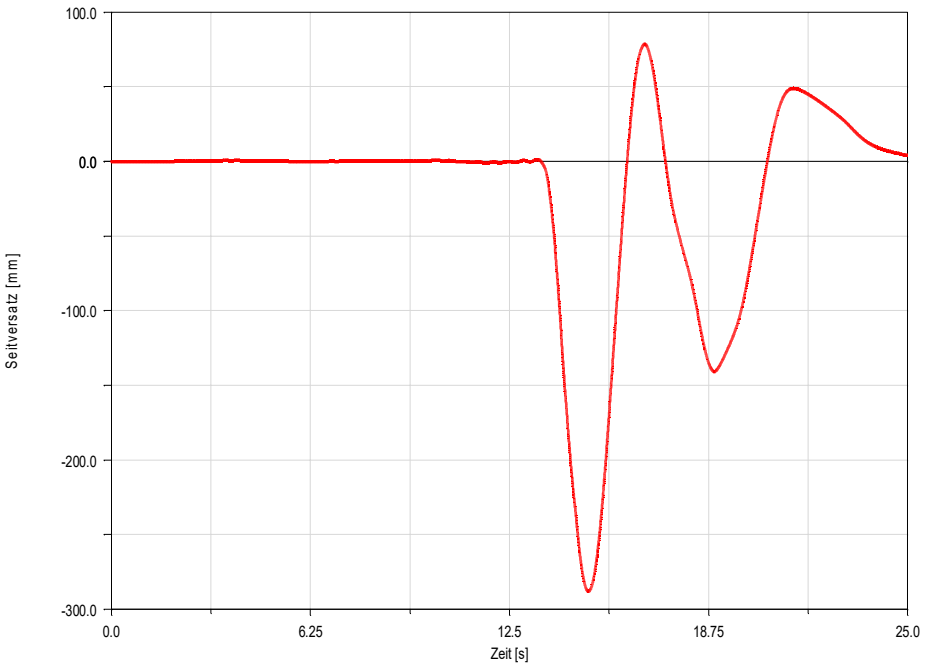


Abbildung 4.51.: Fahrmanöver *µ-Split-Bremsung*: Seitversatz des Roboters

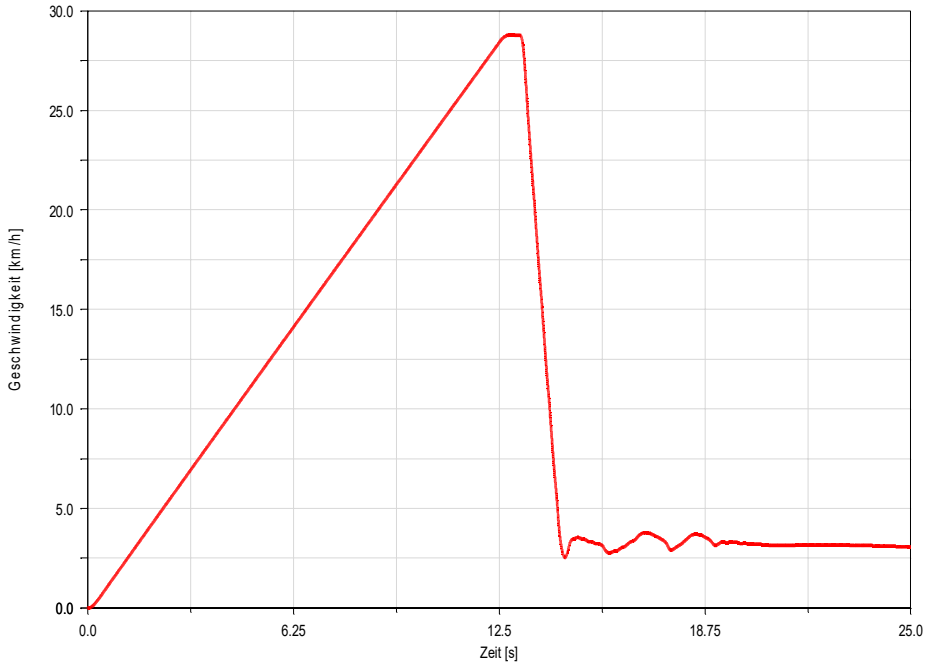


Abbildung 4.52.: Fahrmanöver *µ-Split-Bremsung*: Geschwindigkeit des Roboters

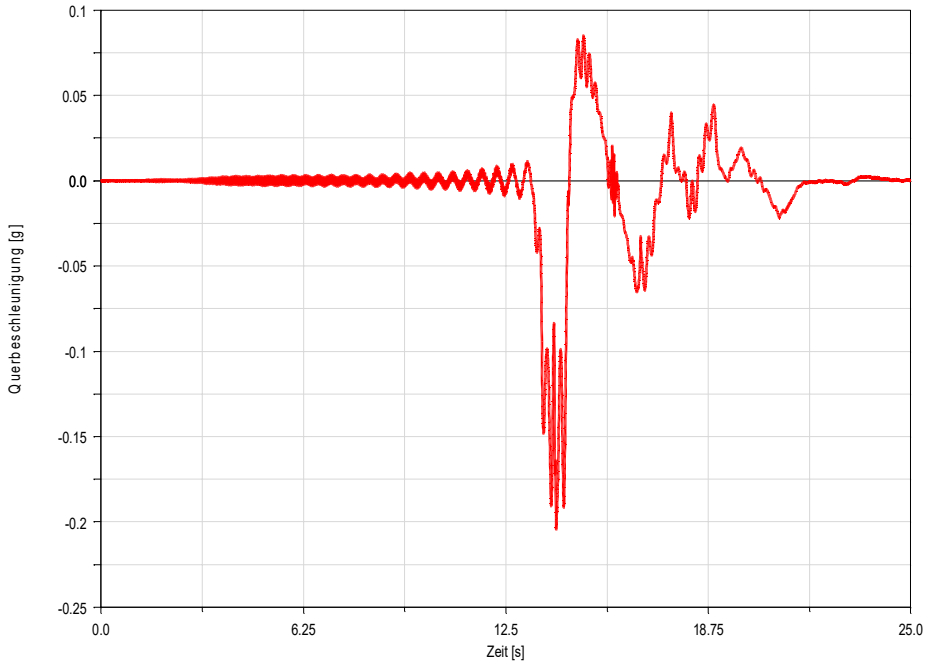


Abbildung 4.53.: Fahrmanöver *µ-Split-Bremsung*: Querbeschleunigung des Roboters

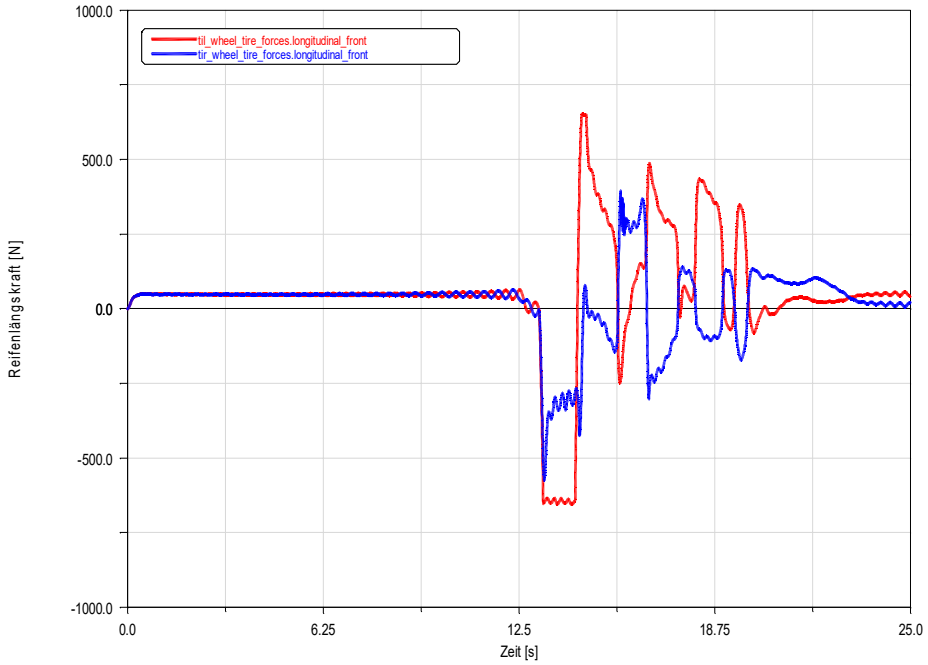


Abbildung 4.54.: Fahrmanöver *µ-Split-Bremsung*: Reifenlängskraft des Roboters vorne links (rote Kurve) und vorne rechts (blaue Kurve)

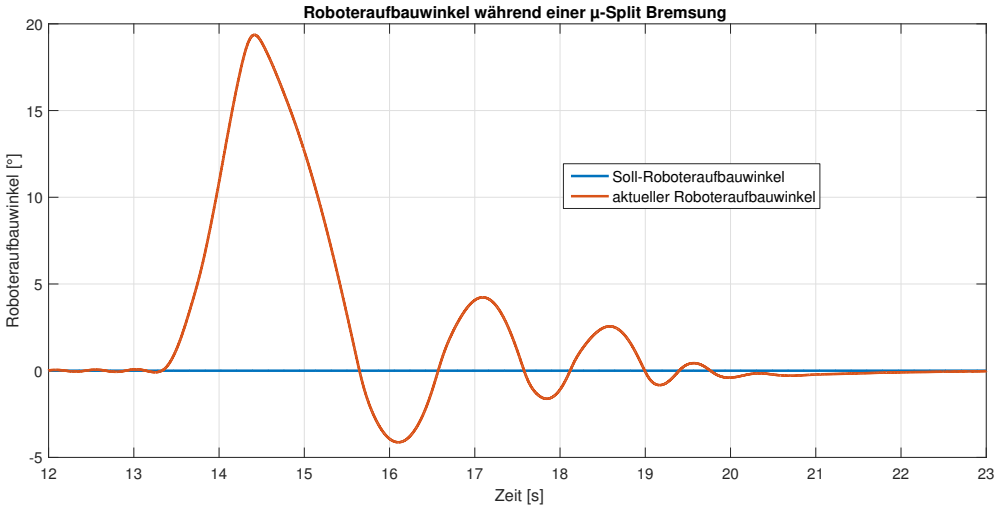


Abbildung 4.55.: Fahrmanöver *µ-Split-Bremsung*: Aufbauwinkel des Roboters

Bewegt sich der Roboter auf dieser vorgegebenen Trajektorie, vom Startpunkt $(x = 0, y = 0)$ in Richtung Ziel, ergibt sich der in Abbildung 4.58 zu sehende Verlauf. Der Roboter wird bewusst mit hoher Geschwindigkeit (50 $\frac{km}{h}$) betrieben, es ergibt sich eine Querbeschleunigung, welche in Kapitel 3 als grenzwertig bezeichnet wurde. Dementsprechend groß ist auch der Schräglaufwinkel an den Rädern, in der Abbildung 4.64 beispielhaft für das Rad links vorne dargestellt. Die Abweichung von der Trajektorie, siehe Abbildung 4.59, bietet einen nachvollziehbaren Verlauf. Die beiden zu sehenden Spitzen ergeben sich durch Kurvenein- bzw. -ausfahrt, da stabilitätsbedingt eine Vorschaudistanz von etwa 8 Meter (bei Höchstgeschwindigkeit) gewählt wurde. Am Ende wird der Roboter durch die proportionale Geschwindigkeitsregelung wieder bis zum Stillstand verzogert und erreicht nahezu exakt den Zielpunkt. Der Soll-Aufbauwinkel φ konnte hier nach der geometrischen Methode bestimmt werden, der Roboteraufbau kann dem gut folgen, nur die Kombination aus Kurvenausgang und Verzögerung führt zu kleinen Abweichungen, bis zum Zielpunkt wird jedoch die gewünschte Roboterstellung ohne Probleme erreicht, siehe Abbildung 4.60.

Der Verlauf des Lenkwinkels für das linke vordere Rad, zu sehen in der Abbildung 4.61, zeigt einen schwankenden Verlauf während des Befahrens des Halbkreises, dies liegt daran, dass das obere Limit der zulässigen Vorschaudistanz gewählt wurde und die in Kapitel 4.3.4.2 angeführte Lösung hier nicht verwendet wird. Die zulässige Vorschaudistanz wird niemals überschritten, dafür ist die Bewegung des Fahrzeuges innerhalb der Kurve weniger flüssig. Der angeführte Verlauf der Raddrehzahl für das linke vordere Rad zeigt keine Besonderheiten, dem vorgegebene Sollwert kann zufriedenstellend gefolgt werden, siehe Abbildung 4.62. Die Abbildung 4.65 zeigt die auftretende Längsbeschleunigung während des Befahrens der Trajektorie, vor allem die Bremsung fällt relativ stark aus.

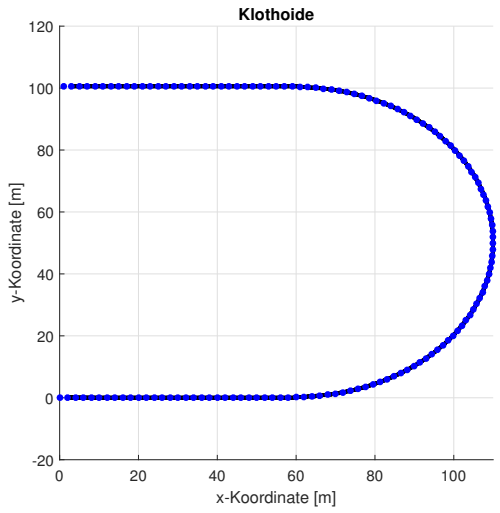


Abbildung 4.56.: Fahrmanöver *Klothoide*: Verwendete Trajektorie

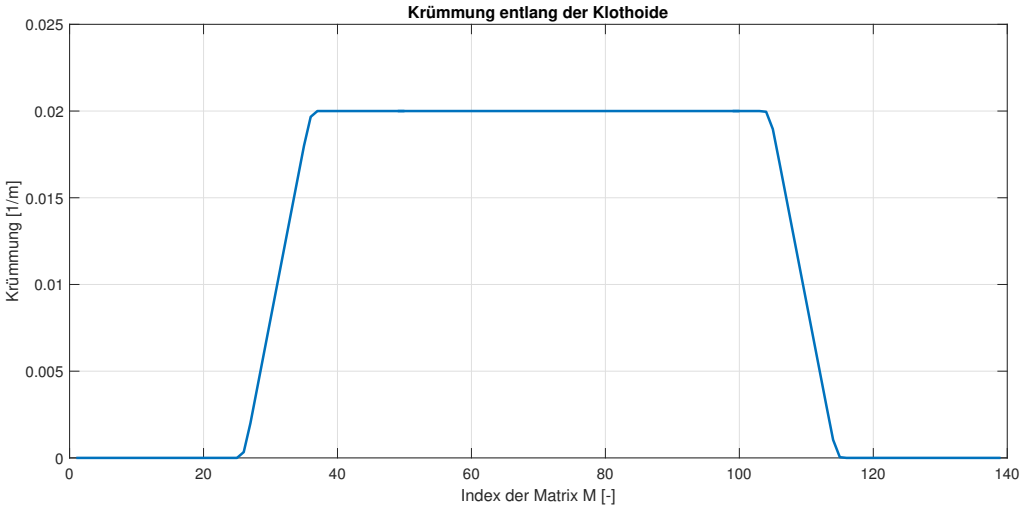


Abbildung 4.57.: Fahrmanöver *Klothoide*: Krümmung der verwendeten Trajektorie

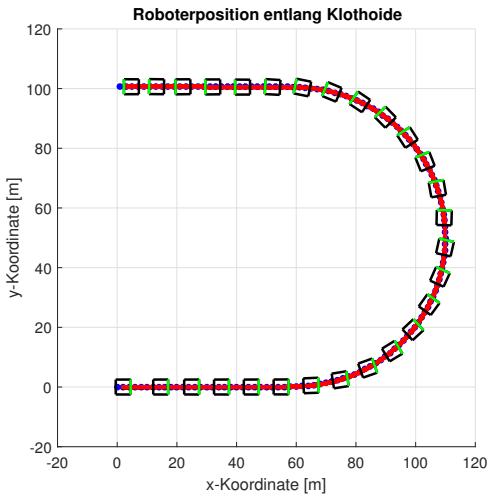


Abbildung 4.58.: Fahrmanöver *Klothoide*: Roboterposition während des Befahrens der Trajektorie im PKW-Modus

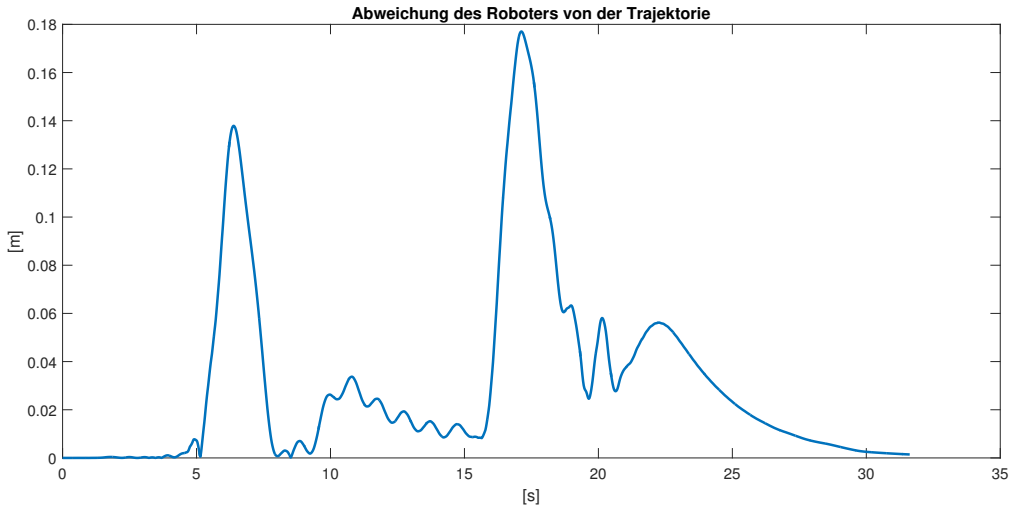


Abbildung 4.59.: Fahrmanöver *Klothoide*: Abweichung des Roboters von der Trajektorie beim Befahren im PKW-Modus

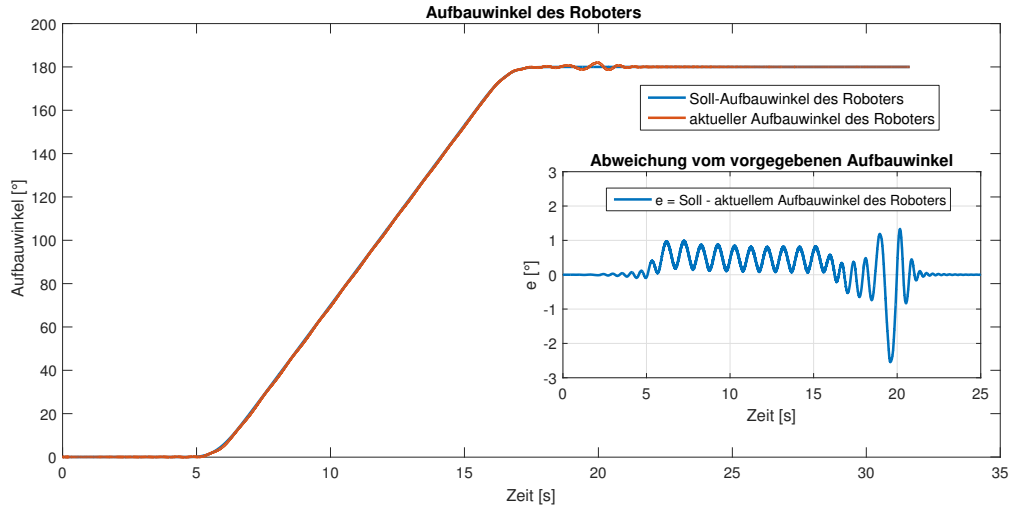


Abbildung 4.60.: Fahrmanöver *Klothoide*: Aufbauwinkel des Roboters beim Befahren der Trajektorie im PKW-Modus

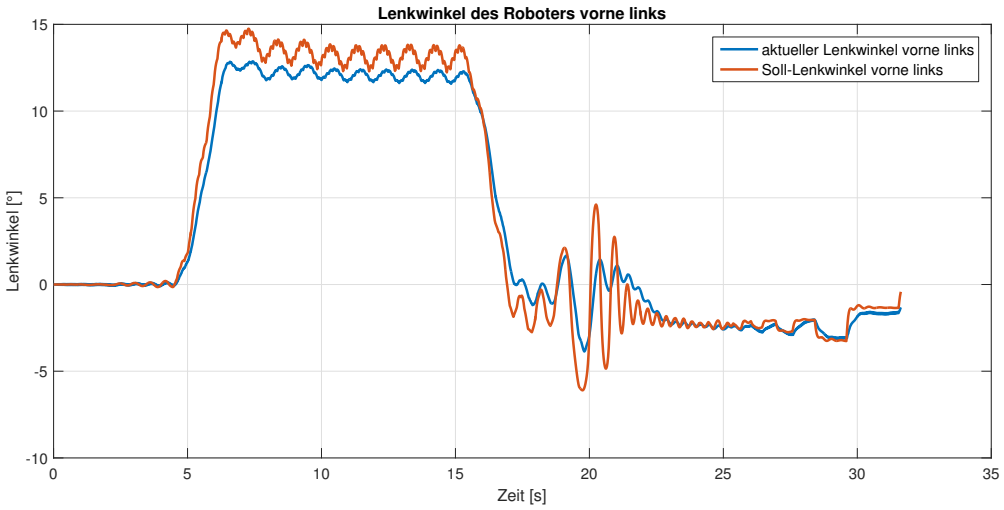


Abbildung 4.61.: Fahrmanöver *Klothoide*: Lenkwinkel des Roboters beim Befahren der Trajektorie im PKW-Modus

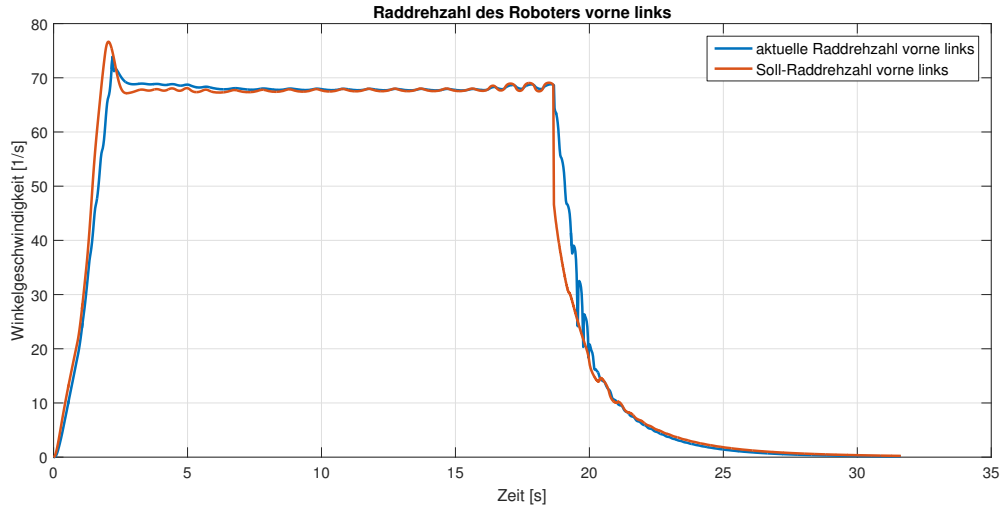


Abbildung 4.62.: Fahrmanöver *Klothoide*: Raddrehzahl des Roboters beim Befahren der Trajektorie im PKW-Modus

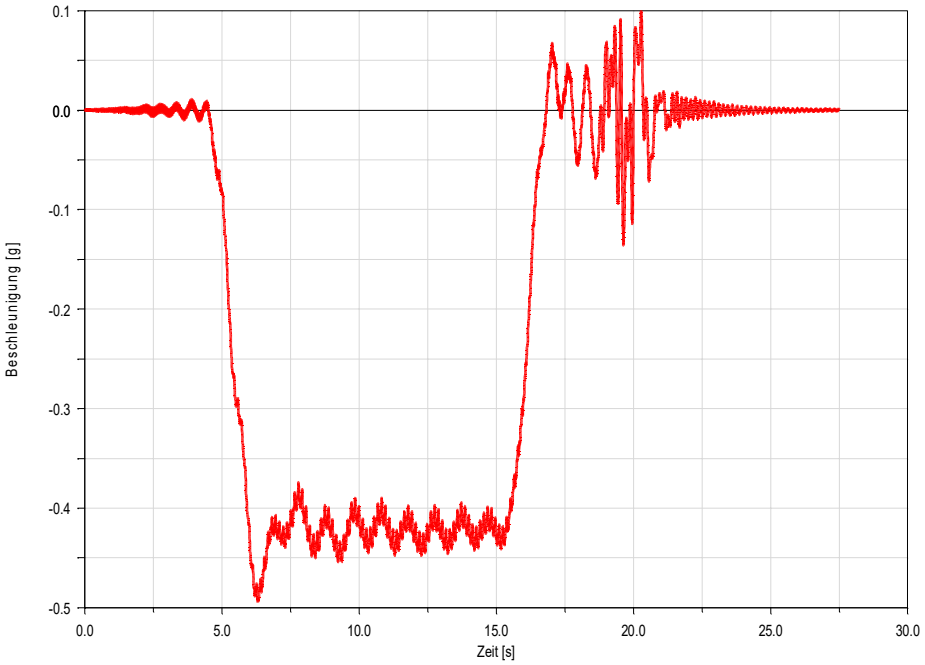


Abbildung 4.63.: Fahrmanöver *Klothoide*: Querbeschleunigung des Roboters während des Befahrens der Trajektorie im PKW-Modus

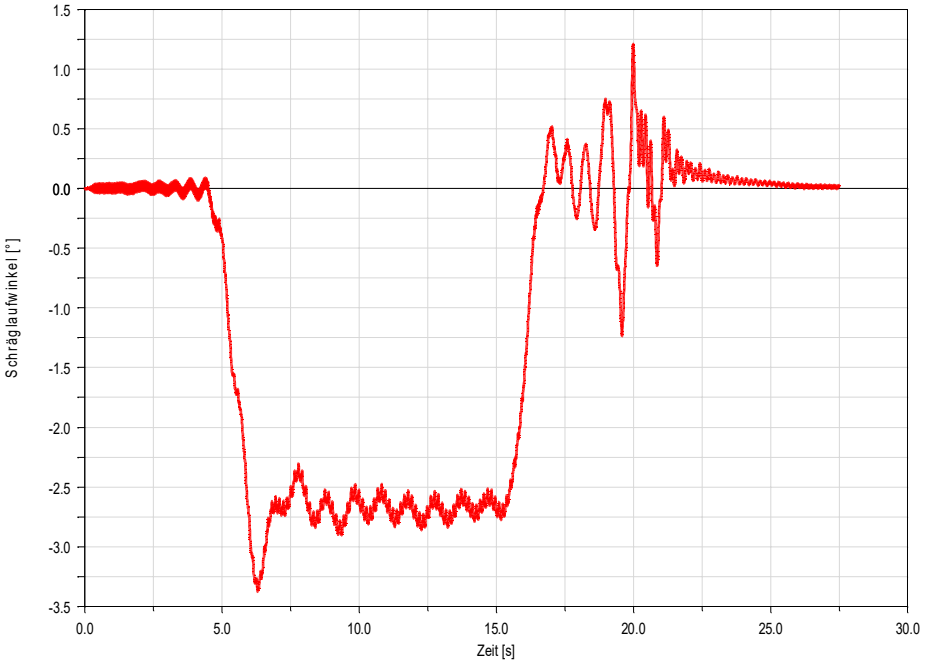


Abbildung 4.64.: Schräglaufwinkel des Roboters während des Befahrens der Klothoide im PKW-Modus

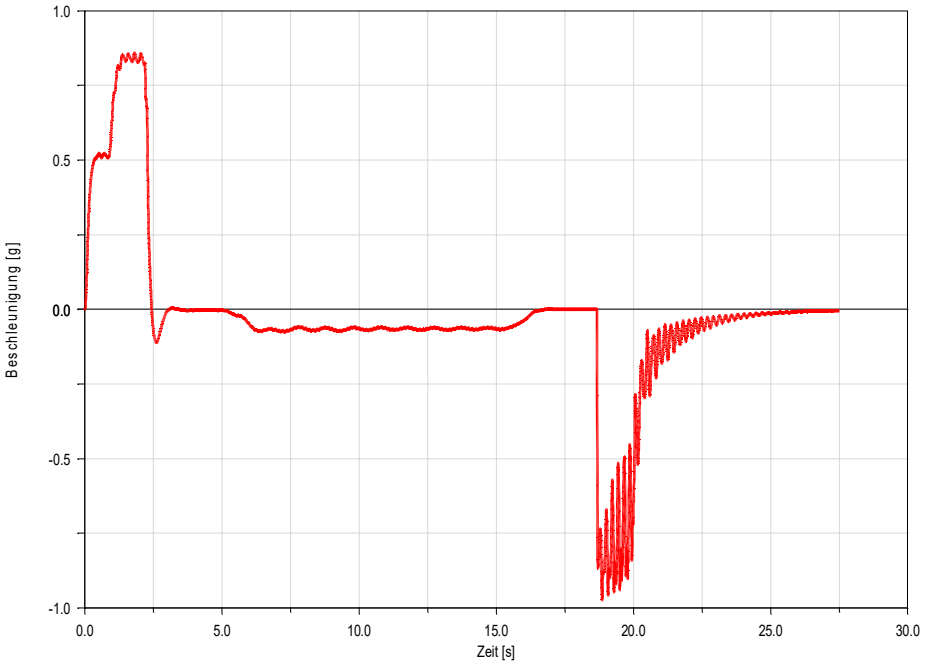


Abbildung 4.65.: Fahrmanöver *Klothoide*: Längsbeschleunigung des Roboters während des Befahrens der Trajektorie im PKW-Modus

4.4.5.1. Klothoide mit Bremsen und Beschleunigen

Um die eben besprochene Trajektorie nun etwas zu erschweren, wird entweder eine Beschleunigung oder Verzögerung während des Befahrens des Halbkreises auf den Roboter aufgebracht. Der resultierende Geschwindigkeitsverlauf für eine Beschleunigungsphase inmitten der Trajektorie, ist in Abbildung 4.67 ersichtlich. Es ergibt sich zwar eine erhöhte Abweichung von der Trajektorie, wie in Abbildung 4.66 zu sehen, dies bleibt jedoch die einzige erkennbare Auswirkung.

Für die eingebrachte Verzögerungsphase ist der Geschwindigkeitsverlauf des Roboters in Abbildung 4.69 ersichtlich. Gegenüber der aufgebrachten Beschleunigung ist die aufgebrachte Verzögerung für eine größere Abweichung von der Trajektorie verantwortlich, wie in Abbildung 4.68 zu sehen ist.

4.4.5.2. Klothoide mit spezieller Drehung

Selbstverständlich kann die betrachtete Trajektorie nicht nur im PKW-Modus befahren werden, auch das Absolvieren im Roboter-Modus ist einwandfrei möglich. Hier werden jedoch zwei andere Varianten gezeigt: Zum Einen, die in Abbildung 4.70 ersichtliche Situation, hier vollführt der Roboter eine ganze Drehung um seine eigene Achse, zum Anderen ist in Abbildung 4.71 eine dreiviertel Drehung zu sehen. Die Abweichung von der vorgegebenen Trajektorie wird durch diese Drehungen nur unwesentlich beeinflusst.

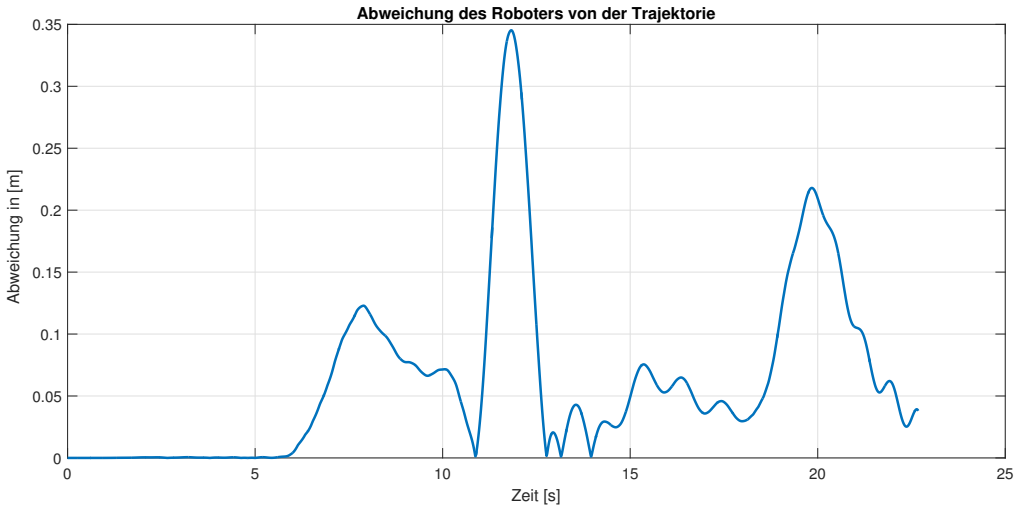


Abbildung 4.66.: Fahrmanöver *Klothoide - Beschleunigen*: Abweichung des Roboters von der Trajektorie beim Befahren im PKW-Modus und einer Beschleunigungsphase

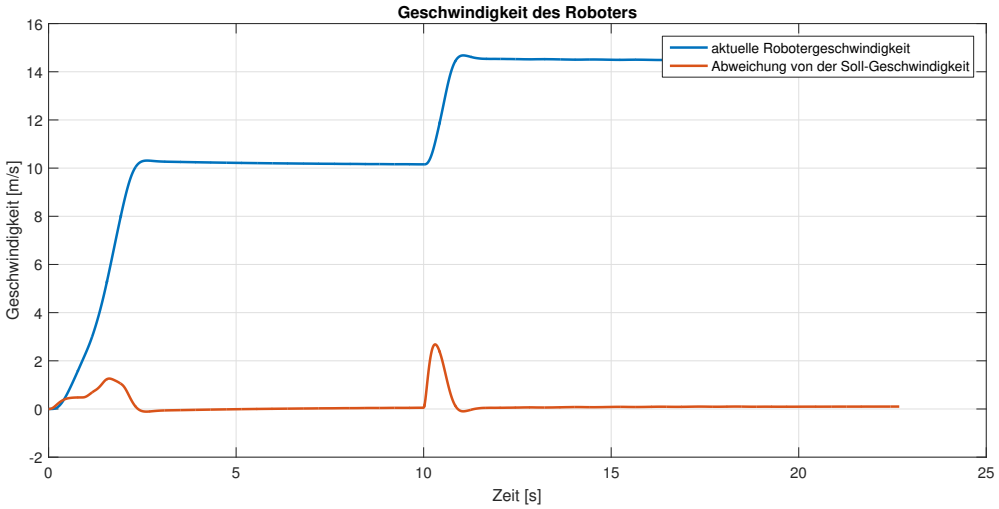


Abbildung 4.67.: Fahrmanöver *Klothoide - Beschleunigen*: Geschwindigkeitsverlauf des Roboters beim Befahren der Trajektorie im PKW-Modus und einer Beschleunigungsphase

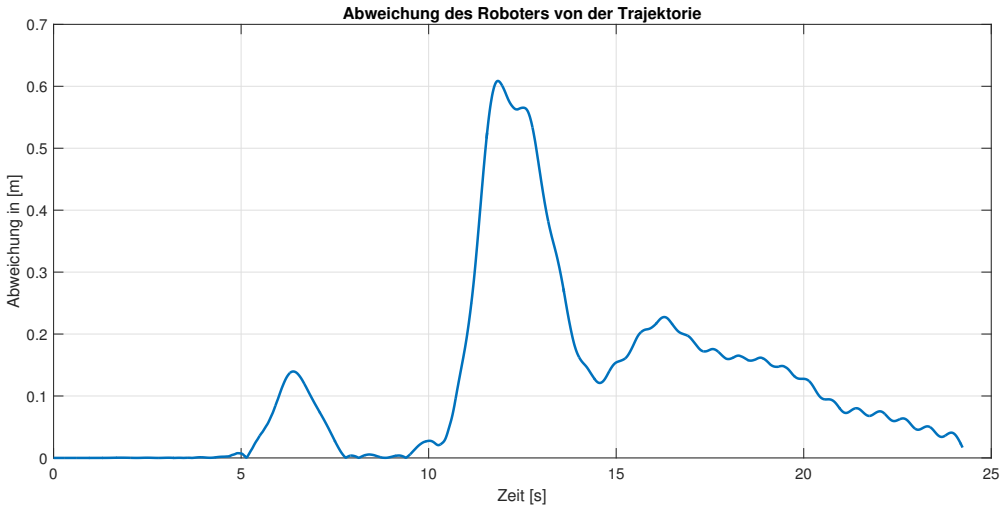


Abbildung 4.68.: Fahrmanöver *Klothoide - Bremsen*: Abweichung des Roboters von der Trajektorie beim Befahren im PKW-Modus und einer Verzögerungsphase

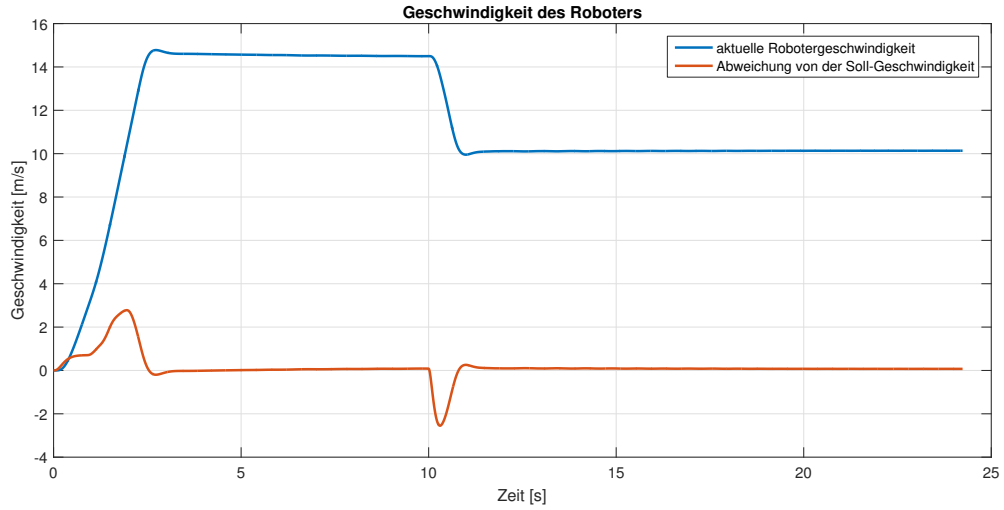


Abbildung 4.69.: Fahrmanöver *Klothoide - Bremsen*: Geschwindigkeitsverlauf des Roboters beim Befahren der Trajektorie im PKW-Modus und einer Verzögerungsphase

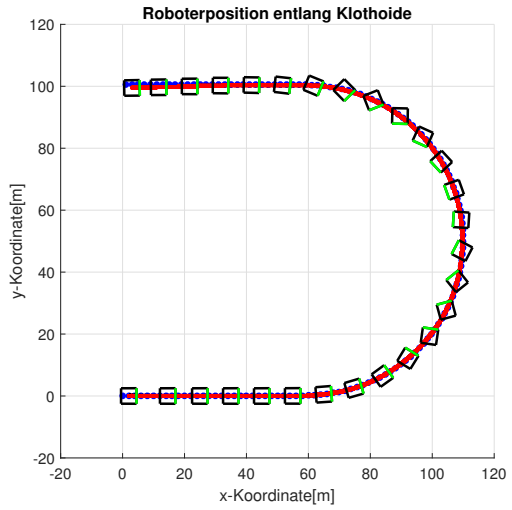


Abbildung 4.70.: Fahrmanöver *Klothoide*: Roboterposition während dem Befahren der Trajektorie mit ganzer Drehung um die eigene Achse

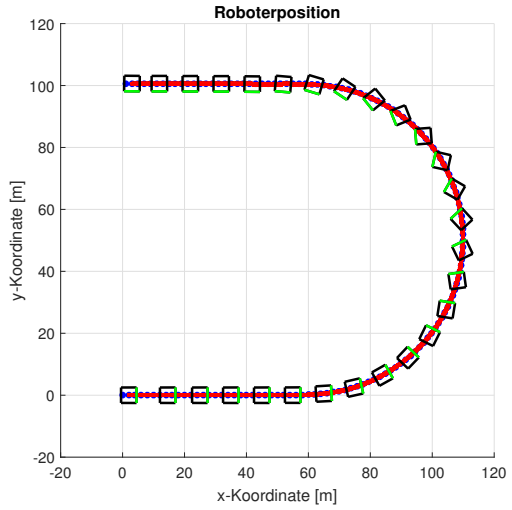


Abbildung 4.71.: Fahrmanöver *Klothoide*: Roboterposition während dem Befahren der Trajektorie mit halber Drehung um die eigene Achse

4.4.6. Fahrmanöver: S-Kurve

Bei diesem Manöver wird eine S-förmige Trajektorie befahren. Im Gegensatz zu dem Krümmungsverlauf der vorgestellten Klothoide in Kapitel 4.4.5, besitzt diese Trajektorie deutliche Sprünge in der Krümmung, wie in Abbildung 4.72 ersichtlich wird und stellt den Roboter somit vor größeren Herausforderungen, als das Manöver der Klothoide in Kapitel 4.4.5. Weiters stellt diese Art der Trajektorie eine besondere Herausforderung an die Regelung des Geschwindigkeitvektors, welche in Kapitel 4.3.4.3 bereits erläutert wurde.

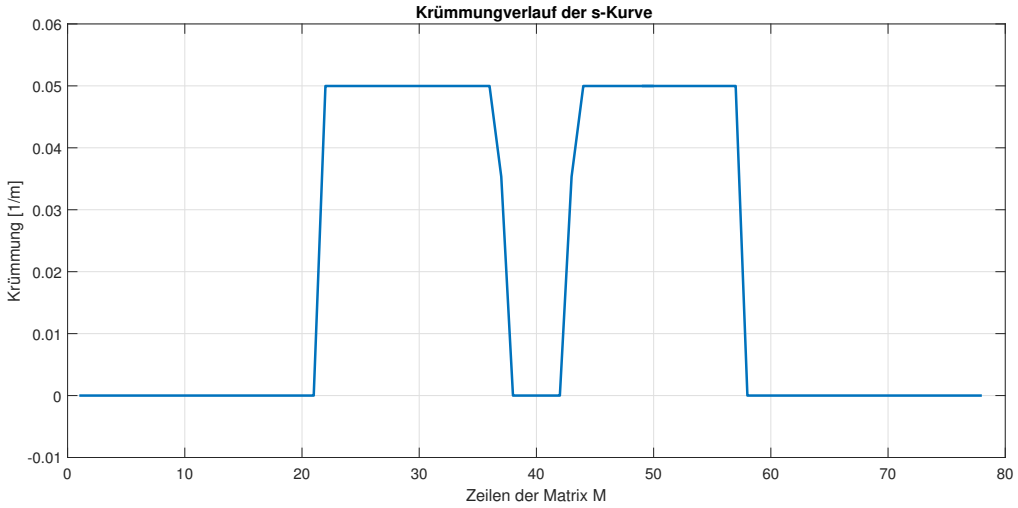


Abbildung 4.72.: Fahrmanöver *S-Kurve, PKW-Modus*: Verlauf der Krümmung der Trajektorie

4.4.6.1. S-Kurve im PKW-Modus

Die Position des Roboters beim Befahren der Trajektorie im PKW-Modus ist in Abbildung 4.73 zu sehen. In der Abbildung 4.74, welche die Abweichung des Roboters von der Trajektorie zeigt, sind zwei größere Ausschläge zu beobachten. Die erste, kleinere Abweichung, ergibt sich bei Kurveneinfahrt, bedingt durch den Vorschaupunkt, findet immer ein verfrühtes Einlenken statt. Die zweite und größte Abweichung ergibt sich durch die große wirkende Querbeschleunigung (Abbildung 4.78), verbunden mit dem sprunghaften Anstieg der Krümmung entlang der Trajektorie. Trotz der hohen Querbeschleunigung in beiden Richtungen, kann die Abweichung von der Trajektorie am Zielpunkt nahezu auf Null verringert werden.

Der Verlauf der Raddrehzahlen, beispielhaft für das linke vordere Rad des Roboters in Abbildung 4.75 gezeigt, lässt keine Auffälligkeiten erkennen, weder die in Längsrichtung aufgebrachte Beschleunigung noch die Verzögerung sind zu stark gewählt. Der Aufbauwinkel des Roboters ist in Abbildung 4.76 erkennbar, die größten Abweichungen treten beim Kurvenein- und -ausgang auf, hier herrscht naturgemäß die größte Änderung der Winkelgeschwindigkeit, ersichtlich in Abbildung 4.77. In dieser Abbildung ist auch ein leichtes Schwingen am Kurvenausgang zu sehen, dies hängt

zum Einen mit dem Kurvenausgang zusammen, zum Anderen mit der Reduzierung der Geschwindigkeit, wie in Abbildung 4.79 erkennbar ist.

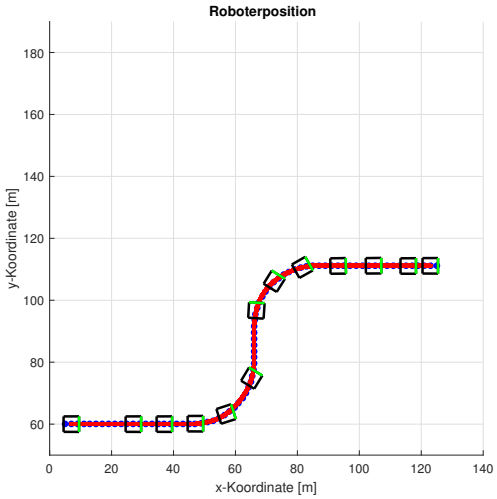


Abbildung 4.73.: Fahrmanöver *S-Kurve, PKW-Modus*: Position des Roboters entlang der Trajektorie

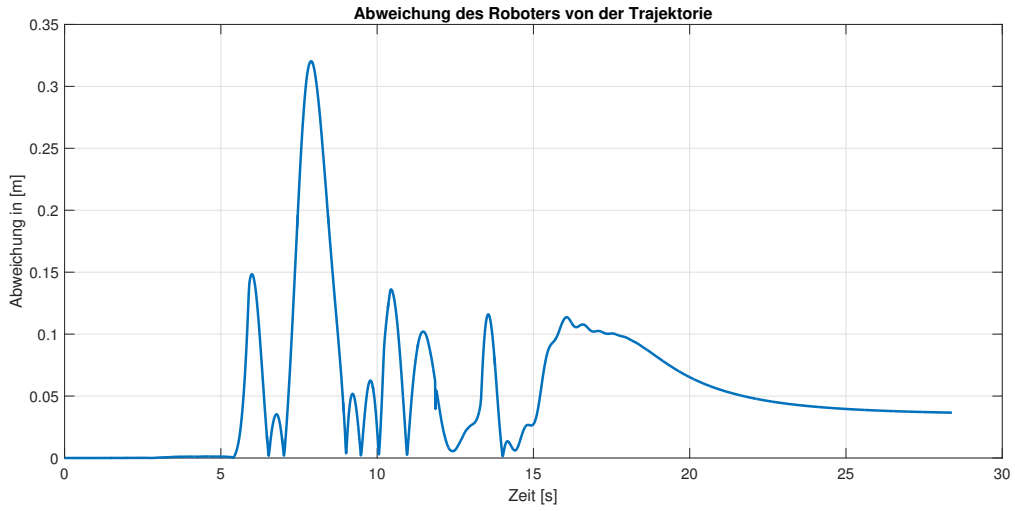


Abbildung 4.74.: Fahrmanöver *S-Kurve, PKW-Modus*: Abweichung des Roboters von der Trajektorie

4.4.6.2. S-Kurve in Robotermodus

Selbstverständlich kann die vorgegebene Trajektorie auch im Robotermodus befahren werden, wie in Abbildung 4.80 ersichtlich ist. Die Richtung des Geschwindigkeitsvektors entlang der Trajektorie ist in Abbildung 4.82 zu sehen, der leicht gewellte Verlauf zeigt an, dass ständig das Maximum der zulässigen Vorschaudistanz ausgenutzt wird. Dies ist notwendig, da durch die Verwendung des Robotermodus die Lenkwinkel sehr hoch sind, es ergeben sich wie in Abbildung 4.83 ersichtlich, etwas größere Abweichungen.

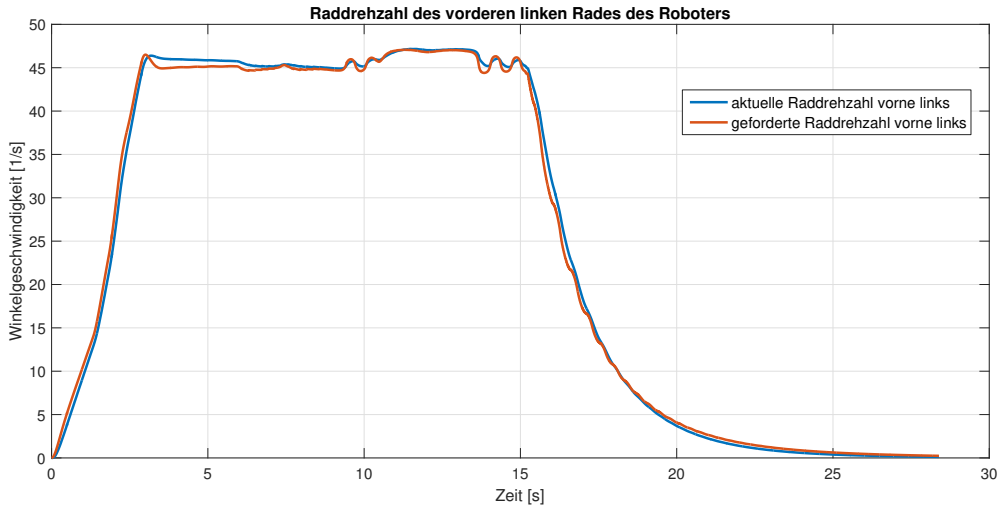


Abbildung 4.75.: Fahrmanöver *S-Kurve, PKW-Modus*: Winkelgeschwindigkeit eines Rades des Roboters während dem Befahrens der Trajektorie

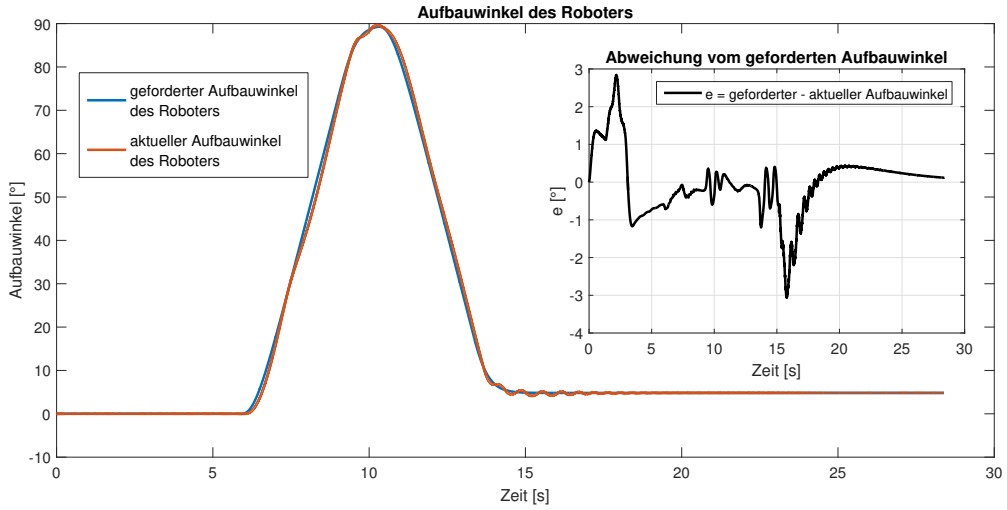


Abbildung 4.76.: Fahrmanöver *S-Kurve, PKW-Modus*: Aufbauwinkel des Roboters während dem Befahrens der Trajektorie

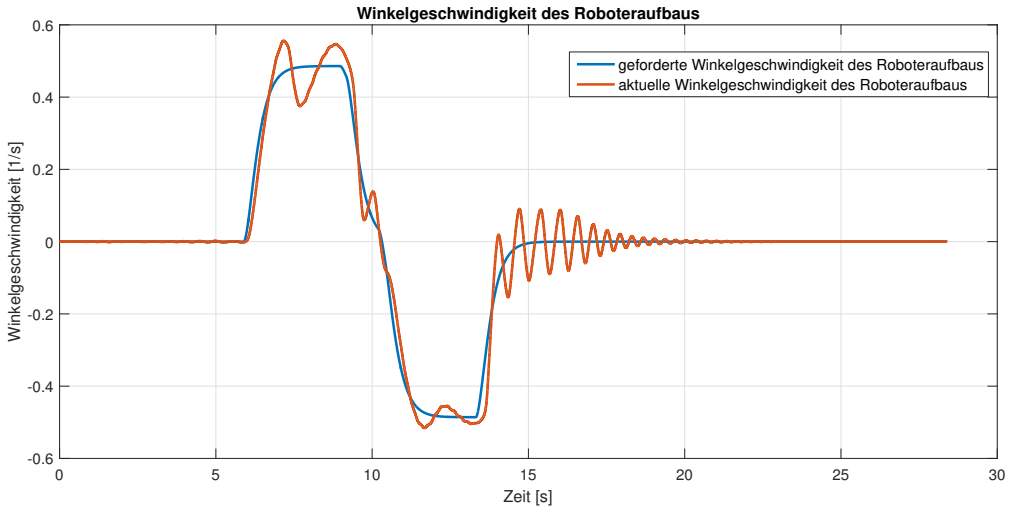


Abbildung 4.77.: Fahrmanöver *S-Kurve, PKW-Modus*: Aufbauwinkelgeschwindigkeit des Roboters während dem Befahrens der Trajektorie

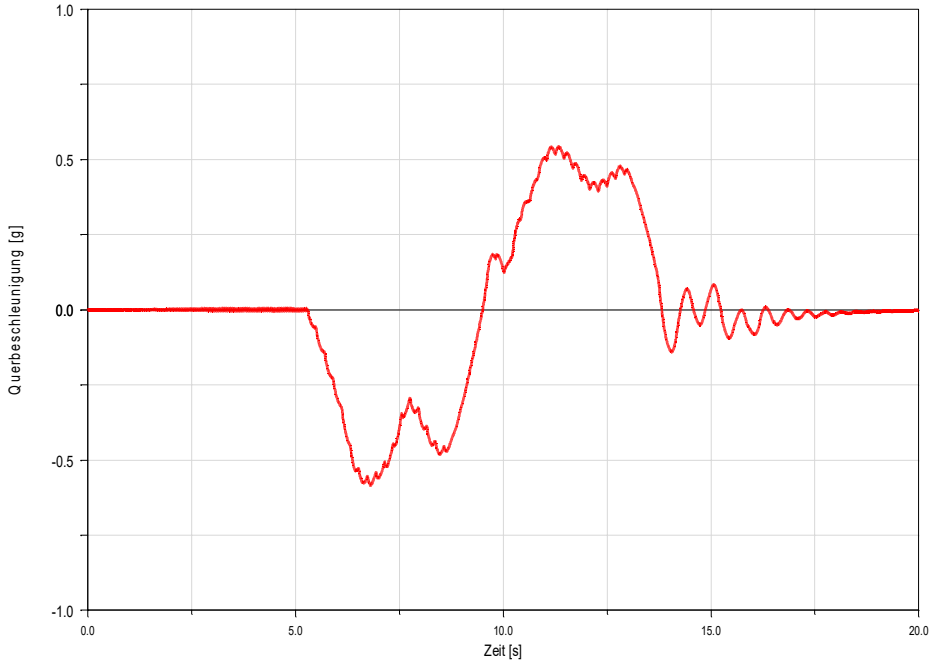


Abbildung 4.78.: Fahrmanöver *S-Kurve, PKW-Modus*: Querbeschleunigung des Roboters während dem Befahrens der Trajektorie

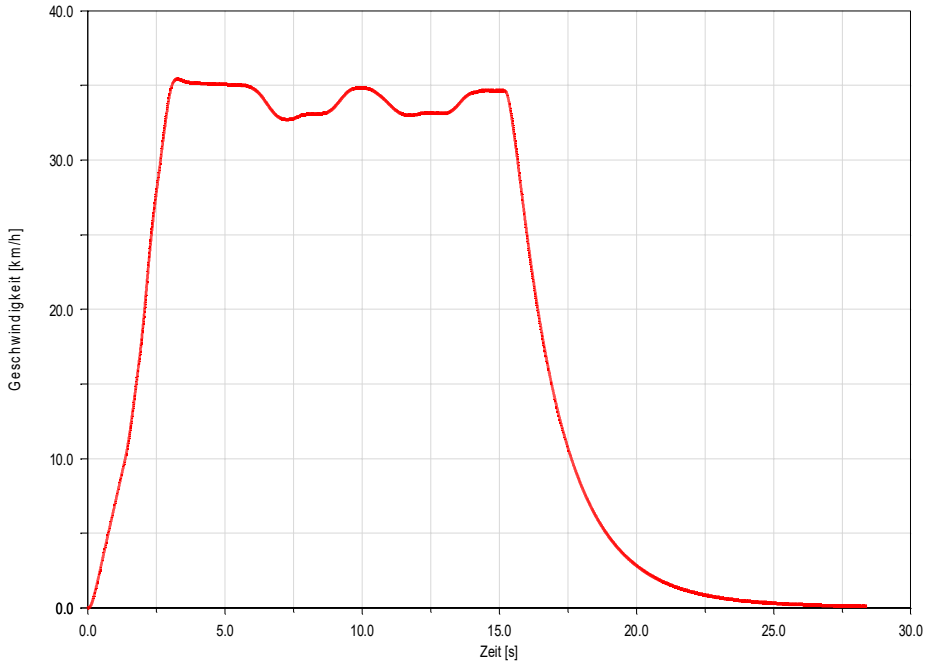


Abbildung 4.79.: Fahrmanöver *S-Kurve, PKW-Modus*: Geschwindigkeit des Roboters während dem Befahrens der Trajektorie

Die Abweichungen bezüglich dem Lenkwinkel schlagen sich direkt in der Abweichung zur Trajektorie nieder, Abbildung 4.81 zeigt dies im Vergleich mit dem PKW-Modus. Für beide Modi wurde die gleiche Vorschaudistanz gewählt, erkennbar durch die zeitgleich auftretende erste Abweichung von der Trajektorie. Der PKW-Modus ist jedoch in der Lage, durch die geringeren Lenkwinkeländerungen, eine etwas kleinere Abweichung zu erreichen, besonders im Bereich des zweiten Kurvenausganges ($t \geq 12s$), ist die unterschiedlich große entstehende Abweichung gut ersichtlich.

4.4.7. Fahrmanöver: doppelter Spurwechsel

Im folgenden wird ein Manöver nach der Norm ISO 3888-2:2011 befahren, nämlich der doppelte Spurwechsel. Dem Roboter bieten sich hier Vorteile, da er mit einem Meter Breite schmaler ist als konventionelle Fahrzeuge. Der doppelte Spurwechsel soll das Ausweichen eines Hindernisses nachstellen, dabei besteht die Aufgabe darin, den Kurs mit möglichst hoher Geschwindigkeit zu absolvieren, innerhalb der Begrenzungen zu bleiben und keine Kontrolle über das Fahrzeug zu verlieren.

In Abbildung 4.84 ist der in ADAMS/Car nachgebildete Spurwechsel zu sehen, die Abbildung 4.85 zeigt die simulierte Bewegung des Roboters entlang der Trajektorie. Die färbigen Linien in der Abbildung deuten in diesem Fall die Grenzen an, welche nicht überschritten werden dürfen und in einem realen Versuch mittels Pylonen gekennzeichnet sind. Da es für dieses Ausweichmanöver von Vorteil ist, den Roboteraufbau nicht noch zusätzlich zu drehen, wird das Manöver im Robotermodus ausgeführt. Die Abbildung 4.86 zeigt die auftretende Abweichung, welche zwar etwas größer als bei den anderen bereits vorgestellten Szenarien ausfällt, allerdings ist das Ausweichsegment auch nur mittels Geradenstücken modelliert worden, mit der

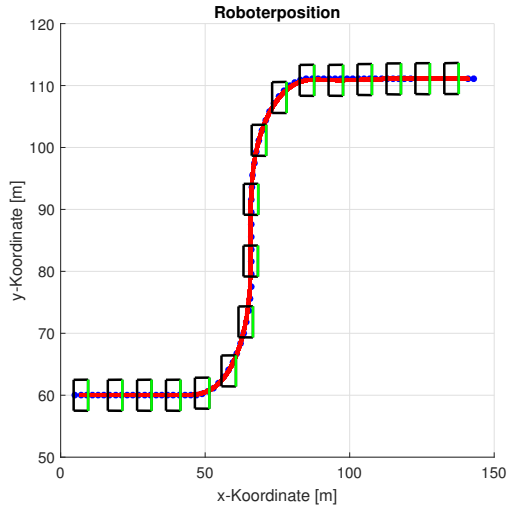


Abbildung 4.80.: Fahrmanöver *S-Kurve, Robotermodus*: Roboterposition beim Befahren der Trajektorie

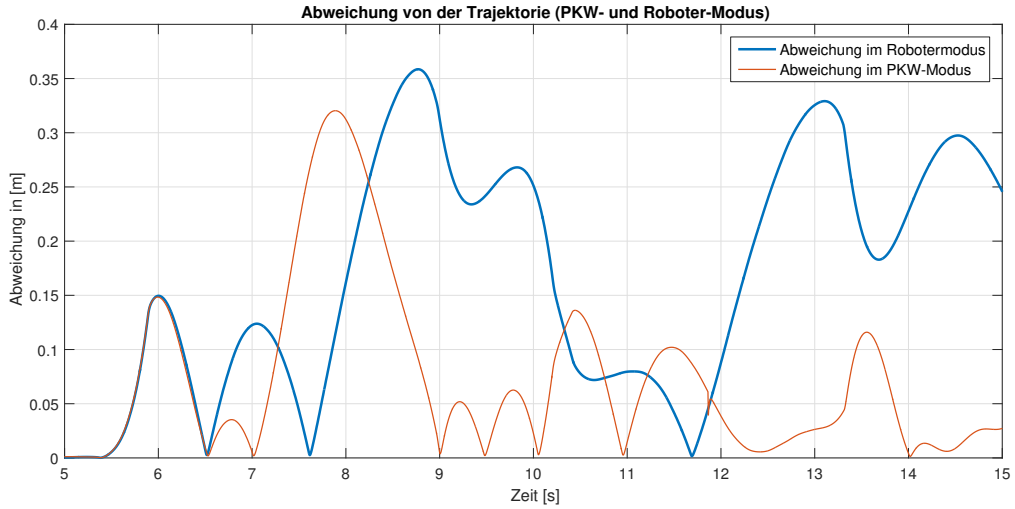


Abbildung 4.81.: Fahrmanöver *S-Kurve*: Vergleich zwischen Roboter- und PKW-Modus beim Befahren der Trajektorie

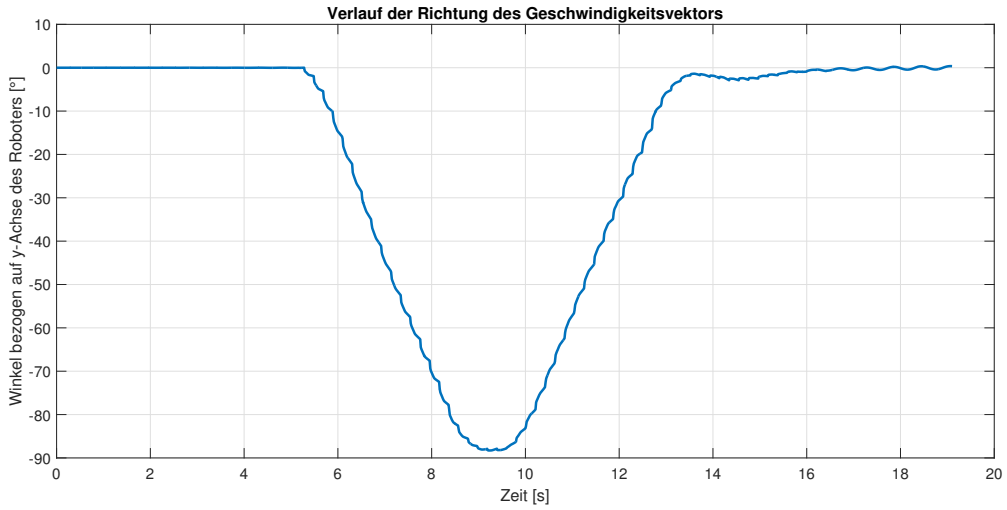


Abbildung 4.82.: Fahrmanöver *S-Kurve, Robotermodus*: Geschwindigkeitsvektor des Roboters bei dem Befahren der Trajektorie

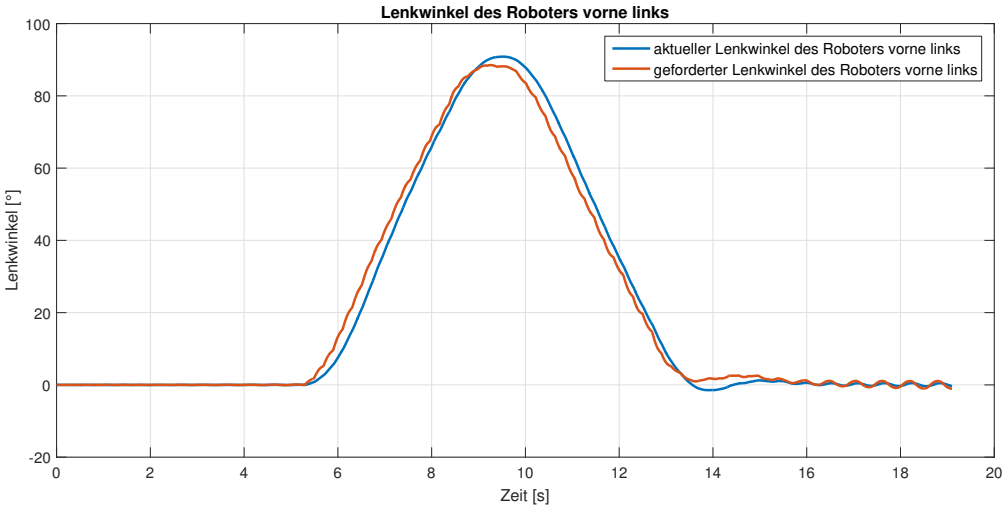


Abbildung 4.83.: Fahrmanöver *S-Kurve, Robotermodus*: Lenkwinkel der Räder des Roboters bei dem Befahren der Trajektorie

vorhandenen Vorschaudistanz, siehe Abbildung 4.88, ergibt sich somit eine prinzipbedingte Abweichung. Der Roboter war in der Lage den doppelten Spurwechsel mit einer Geschwindigkeit von etwa 30 $\frac{km}{h}$ zu durchfahren, die dabei auf das Fahrzeug wirkenden Querbeschleunigungen sind in Abbildung 4.89 zu sehen.

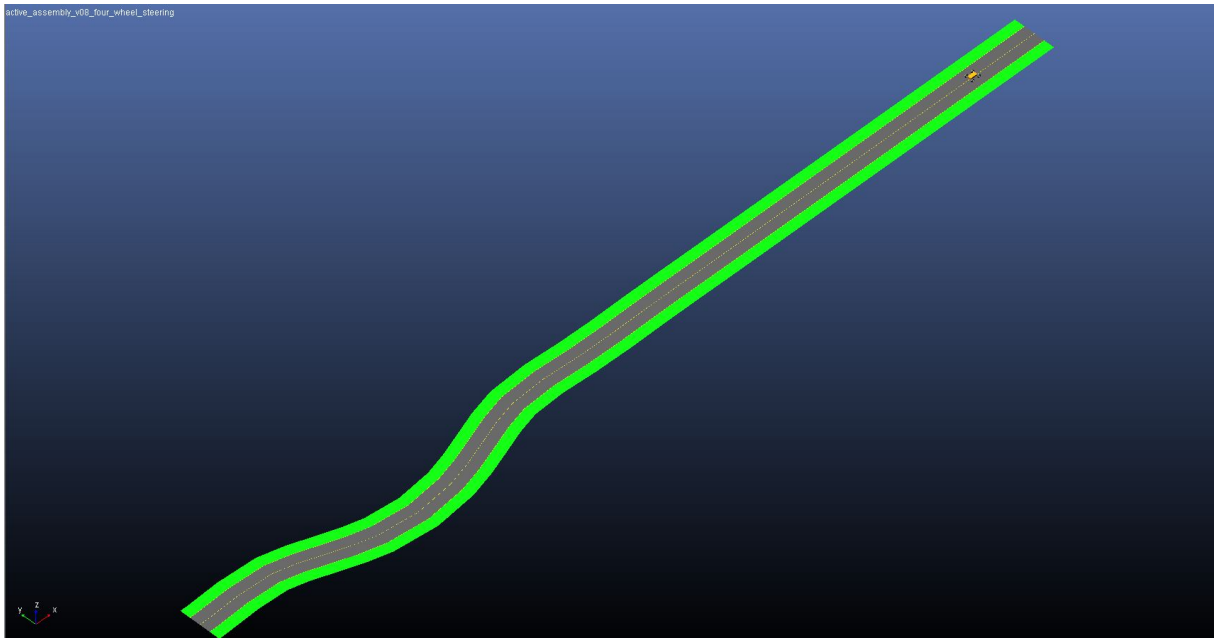


Abbildung 4.84.: Fahrmanöver *doppelter Spurwechsel*: Vorgegebene Trajektorie

4.4.8. Fahrmanöver: Beschleunigen und Bremsen

In dem letzten behandelten Fahrmanöver, wird der Roboter mit den maximal möglichen Antriebsmomenten beschleunigt, anschließend verzögert und dabei auf einer geraden Trajektorie bewegt. Die dabei auftretende Längsbeschleunigung ist in Abbildung 4.90 ersichtlich, sowohl in der Beschleunigungsphase, als auch in der Verzögerungsphase wird eine Beschleunigung von einem g überschritten.

Beim Beschleunigungsvorgang verlieren die Räder an der Vorderachse mehrmals den Bodenkontakt, siehe Abbildung 4.91, welche die Momente an einem Rad der Vorderachse zeigen, sowie Abbildung 4.93, hier sind die Reifenaufständskräfte für ein Rad an der Vorderachse gezeigt. Beim Verzögerungsvorgang tritt der umgekehrte Fall ein, zu sehen in den Abbildungen 4.92 für das Antriebsmoment und Abbildung 4.94 für die Reifenaufstandskraft für ein Rad der Hinterachse. Der Effekt ist jedoch nicht so stark wie beim Beschleunigen, da beim Verzögern die Geschwindigkeit kontinuierlich verringert wird, weshalb kein Verlust des Bodenkontaktes auftritt, weder an der Vorder- noch an der Hinterachse. Den Verlauf der Geschwindigkeit des Fahrzeuges während dieses Manövers ist in Abbildung 4.95 zu sehen.

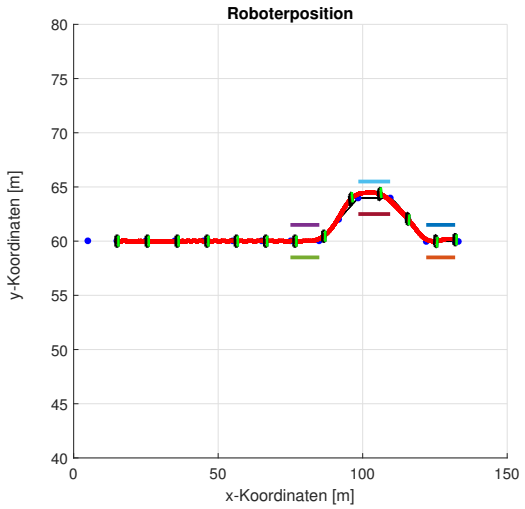


Abbildung 4.85.: Fahrmanöver *doppelter Spurwechsel*: Roboterposition während des Befahrens der Trajektorie im Robotermodus

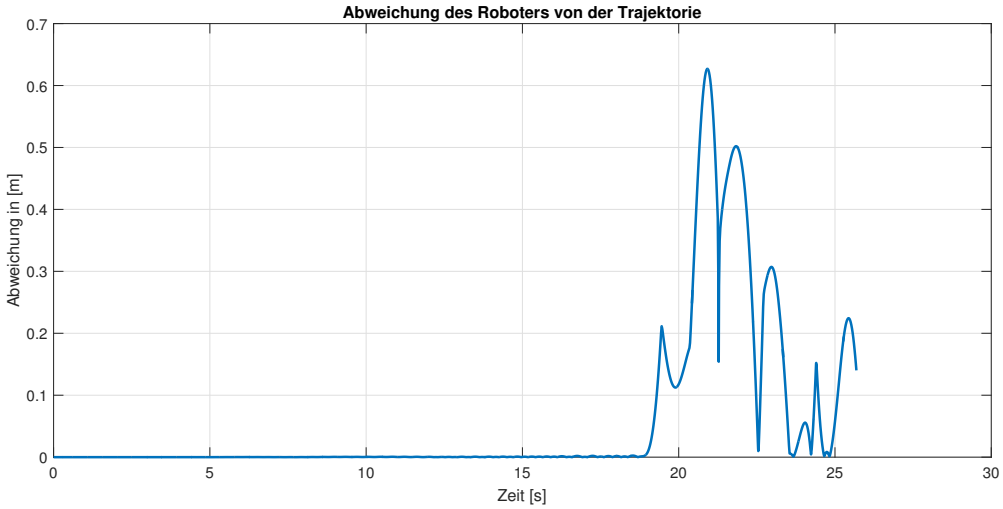


Abbildung 4.86.: Fahrmanöver *doppelter Spurwechsel*: Abweichung des Roboters während des Befahrens der Trajektorie im Robotermodus

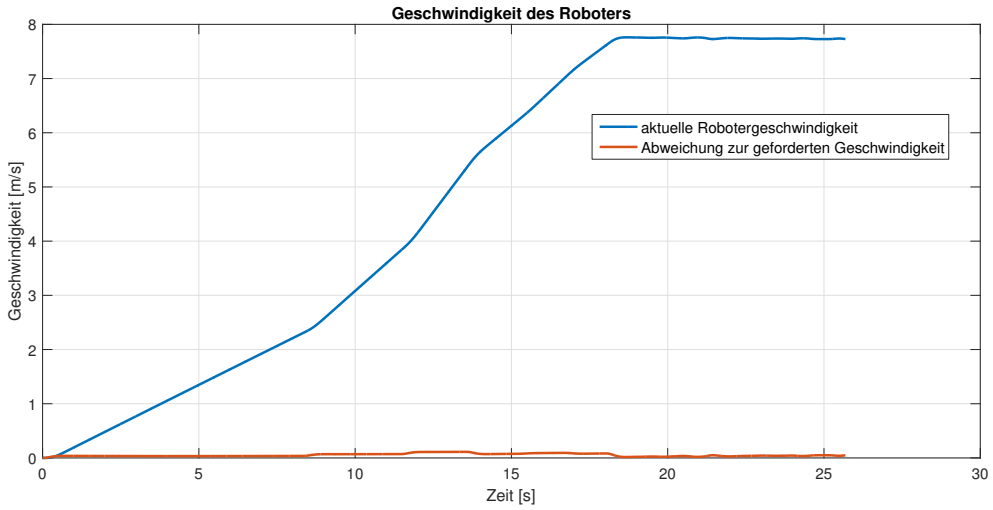


Abbildung 4.87.: Fahrmanöver *doppelter Spurwechsel*: Geschwindigkeit des Roboters während des Befahrens der Trajektorie im Robotermodus

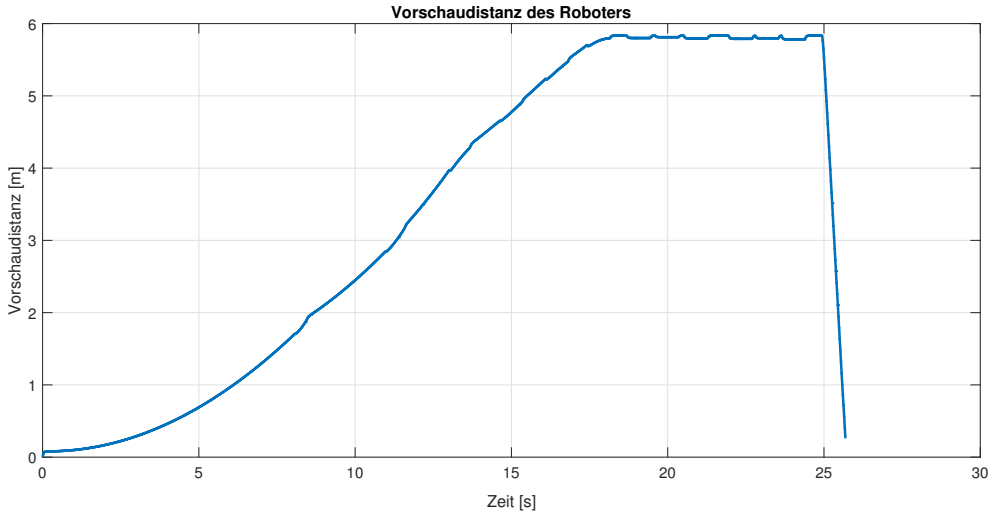


Abbildung 4.88.: Fahrmanöver *doppelter Spurwechsel*: Vorschaudistanz des Roboters während des Befahrens der Trajektorie im Robotermodus

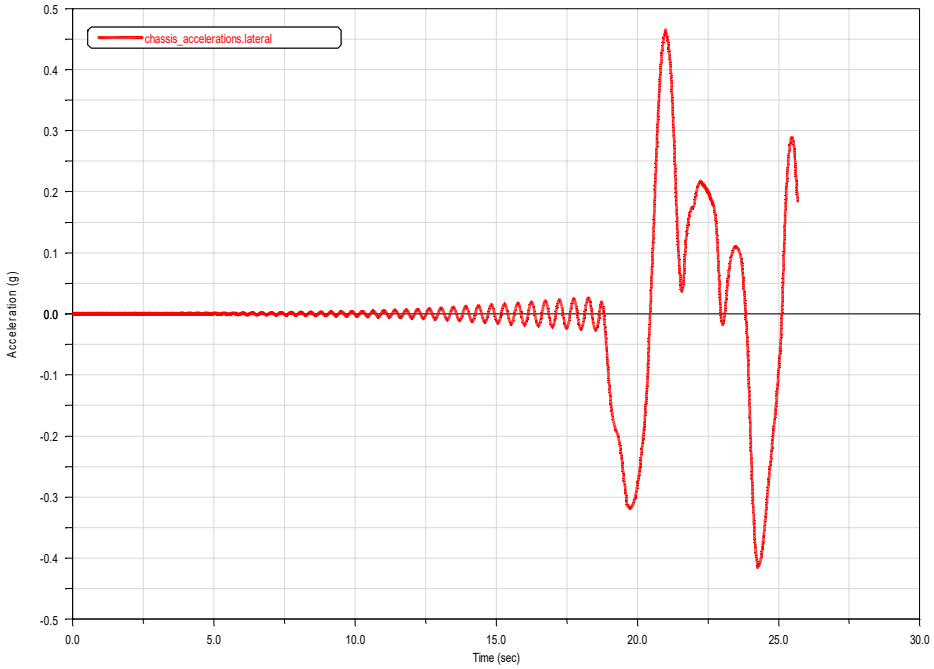


Abbildung 4.89.: Fahrmanöver *doppelter Spurwechsel*: Querbeschleunigung des Roboters während des Befahrens der Trajektorie im Robotermodus

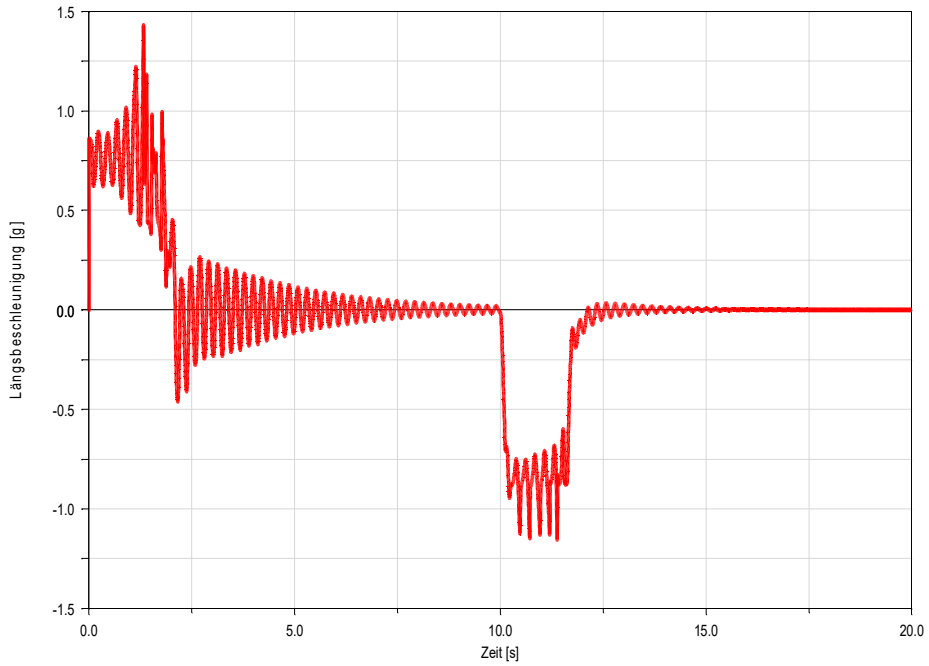


Abbildung 4.90.: Fahrmanöver *Beschleunigen-Bremsen*: Verlauf der Längsbeschleunigung

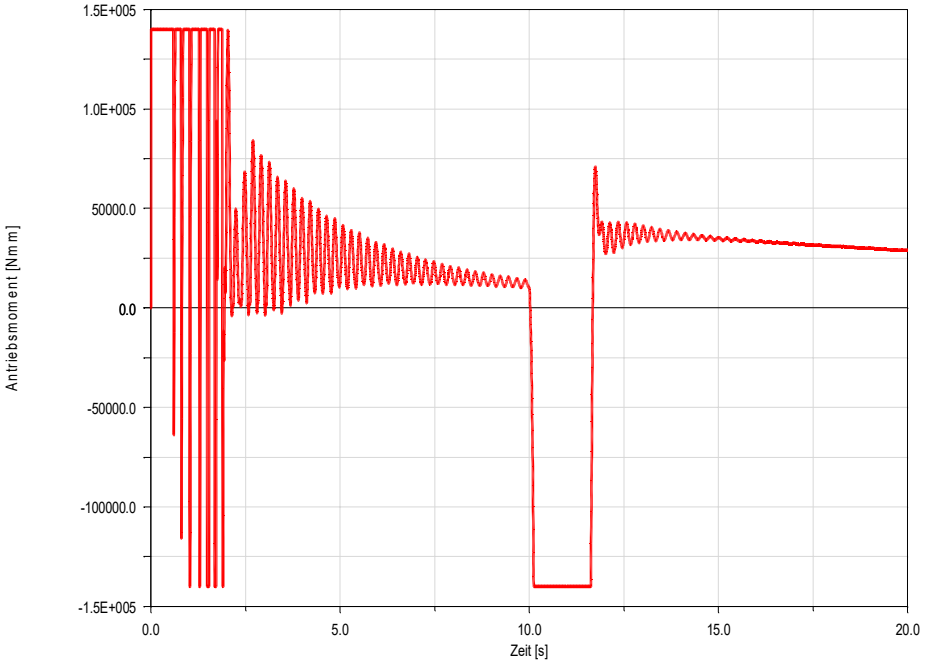


Abbildung 4.91.: Fahrmanöver *Beschleunigen-Bremsen*: Verlauf des Antriebsmomentes an einem Rad der Vorderachse

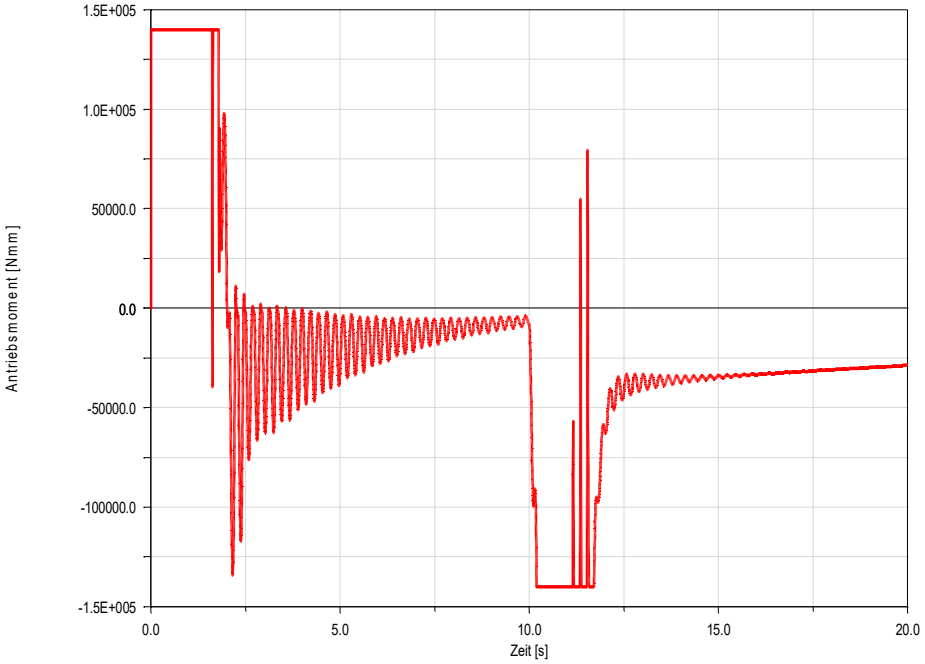


Abbildung 4.92.: Fahrmanöver *Beschleunigen-Bremsen*: Verlauf des Antriebsmomentes an einem Rad der Hinterachse

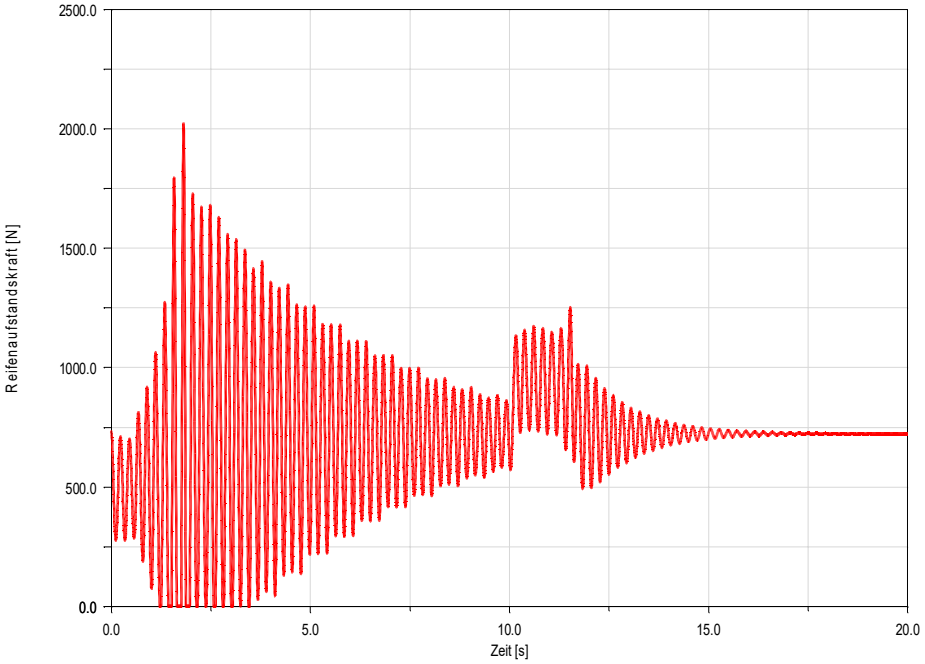


Abbildung 4.93.: Fahrmanöver *Beschleunigen-Bremsen*: Verlauf der Reifenaufstandskraft an einem Rad der Vorderachse

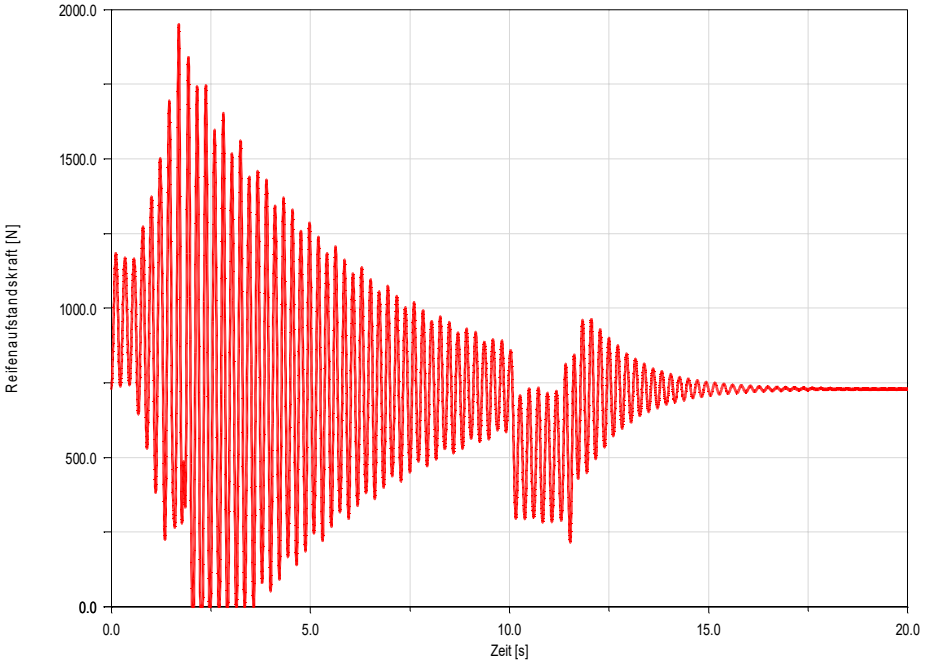


Abbildung 4.94.: Fahrmanöver *Beschleunigen-Bremsen*: Verlauf der Reifenaufstandskraft an einem Rad der Hinterachse

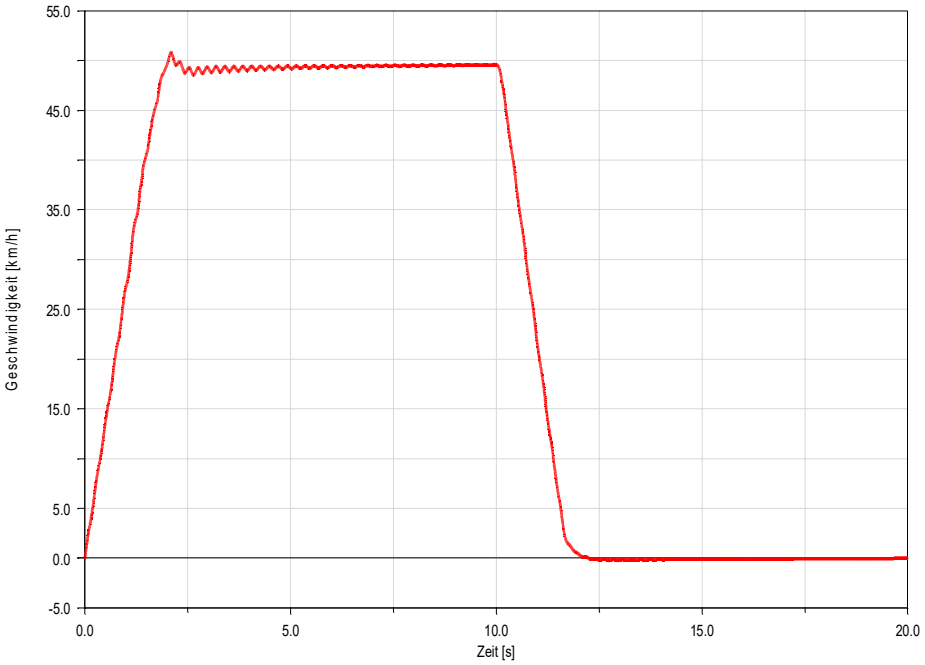


Abbildung 4.95.: Fahrmanöver *Beschleunigen-Bremsen*: Verlauf der Geschwindigkeit des Roboters

4.5. Diskussion der Ergebnisse der Gesamtsimulation

Aus den durchgeführten fahrdynamischen Manövern lassen sich nun einige Aussagen ableiten. Bei zwei der durchgeführten Szenarien, bei dem Drehzahlsprung (Kapitel 4.4.2) und auch bei dem Beschleunigungsmanöver (Kapitel 4.4.8) zeigte sich, dass die Antriebsmotoren dazu in der Lage sind, den Roboter, während eines Beschleunigungsvorganges, an der Vorderachse aufzuheben, ein Umstand der im realen Betrieb umbedingt vermieden werden sollte. Desweiteren wurde die Wichtigkeit einer möglichst ohne Ecken definierten Trajektorie ersichtlich. Im PKW-Modus ist ohne Veränderung einer solchen Trajektorie ein zufriedenstellendes Befahren nicht möglich. Hier muss bereits bei Erstellung der Trajektorie darauf geachtet werden, nicht zu viele Unstetigkeiten zu erzeugen. Die im Zuge der durchgeführten fahrdynamischen Voruntersuchung festgelegte maximale Querbeschleunigung konnte mit dem realisierten Konzept erreicht werden, selbstverständlich steigt jedoch die Ungenauigkeit, je näher man dem Grenzbereich, vor allem des Reifens, kommt.

Ein großer Hebel für die Stabilität, sowie auch die Genauigkeit des umgesetzten Konzeptes, stellt die Vorschaudistanz dar. Dieser Umstand wird auch in den Ergebnissen ersichtlich, da der Robotermodus, begründet durch die größeren Lenkwinkel, bei gleicher Vorschaudistanz, immer größere Abweichungen produziert als der PKW-Modus. Das bedeutet im Umkehrschluss, dass bereits bei der Planung der Bewegung des Roboters, darauf Acht gegeben werden sollte, in welchem Modus dieser Betrieben werden soll. Weiters zeigt sich, besonders bei Trajektorien mit Unstetigkeiten, wie beispielsweise Ecken, dass große Vorschaudistanzen ein stabiles Befahren ermöglichen, jedoch mit größeren Abweichungen zu rechnen ist, als bei geringeren Vorschaudistanzen, welche jedoch zu Instabilität und letztlich noch größeren Abweichungen neigen können.

5. Zusammenfassung und Ausblick

Die vorliegende Arbeit befasst sich mit der fahrdynamischen Auslegung eines allradgetriebenen Roboters mit quasi-omnidirektionaler Lenkung, welcher einer Trajektorie folgen soll.

Da sich der betrachtete Roboter in einer sehr frühen Phase der Entwicklung befindet, wurde zunächst ein möglichst einfaches Mehrkörpersimulationsmodell erstellt. Bei diesem Modell wurde die Vorderachse, wie bei einem PKW, mittels Ackermann-Kinematik realisiert, die Hinterachse ist ungelenkt. Trotz der getroffenen Vereinfachungen und vorhandenen Unsicherheiten, bezüglich der für das Modell verwendeten Parameter, ist die grundsätzliche Dynamik des Systems in ausreichendem Maße abgebildet worden.

Das hauptsächliche Ziel der Voruntersuchung war es herauszufinden, ob der Roboter für den angedachten realen Betrieb eine dezidierte Federung, sowie eine Dämpfung, ausgeführt beispielsweise als Mc-Pherson Radaufhängung, benötigt. Hierzu wurde die Bewegung des erstellten Robotermodells auf einer unebenen Straße simuliert, der Roboter bewegte sich dabei mit seiner angedachten Höchstgeschwindigkeit von 50 $\frac{km}{h}$. Die Ergebnisse zeigen, dass zwar mit erhöhtem Schwingen des Systems zu rechnen ist, was bei einer Kurvefahrt zu schnellerem Kraftschlussverlust führt, jedoch findet bei der Geradeausfahrt kein Abheben der Räder statt. Ein potentieller Lenkeingriff wegen eines Hindernisses ist somit auch in diesem Szenario noch möglich. Dies bedeutet im Umkehrschluss, dass auf eine zusätzliche Feder- oder Dämpferwirkung verzichtet werden kann, da in diesem Fall der damit einhergehende Komfortverlust vertretbar ist. Diese gewonnene Information konnte in dem parallel laufenden Entwicklungsprozess eingebunden werden, anschließend erfolgte die Entwicklung eines zweiten MKS-Modells, mit welchem es möglich ist alle vier Räder unabhängig voneinander zu lenken. Dieses Modell wurde auch für die erstellte Co-Simulation zwischen MATLAB/Simulink und ADAMS/Car verwendet. Für die, dem Roboter vorgegebene Trajektorie, können beliebige Punkte definiert werden, im nächsten Schritt werden diese automatisch zu einer punktweise-linearen Trajektorie zusammengesetzt. Für eine größere Genauigkeit werden zusätzlich Zwischenpunkte generiert, da die Beseitigung einer potentiellen Abweichung des Roboters von der Trajektorie, darauf basiert, einen Punkt, welcher vor der Fahrtrichtung des Roboters liegt, anzuvisieren.

Mittels dem in der Co-Simulation umgesetzten Konzeptes, ist es dem Roboter möglich, innerhalb seiner fahrdynamischen Grenzen bleibend, dieser Vorgabe zu folgen und zusätzlich dazu, seinen Roboteraufbau wie gewünscht zu bewegen. Weiters konnte gezeigt werden, dass die Verwendung eines kinematischen Modells, welches für die Bestimmungs der Lenkwinkel und Raddrehzahlen des Roboters verwendete wurde, ausreicht um Bewegungen im normalen Betrieb zu realisieren. Sollen vermehrt Bewegungen des Roboters realisiert werden, welche entweder im Grenzbereich liegen, oder

das Fahrzeug wird auf losem Untergrund bewegt, ist als nächster Schritt ein dynamisches Modell für die Berechnung der Lenkwinkel und Raddrehzahlen zu verwenden, da dann die für das kinematische Modell zu Grunde gelegte Annahme einer reinen Festkörperbewegung ihre Gültigkeit verliert. Um das Schneiden von einzelnen Bereichen der Trajektorie zu verhindern, begründet durch die Vorausschau des Roboters entlang der Trajektorie selbst, muss entweder die Trajektorie dahingehend optimiert werden (weniger Knicke, möglichst keine Ecken, sondern Radien), oder es muss eine andere Strategie für die Richtungsbestimmung des Roboters verwendet werden.

Anhang A.

Anhang: Fahrdynamische Voruntersuchung

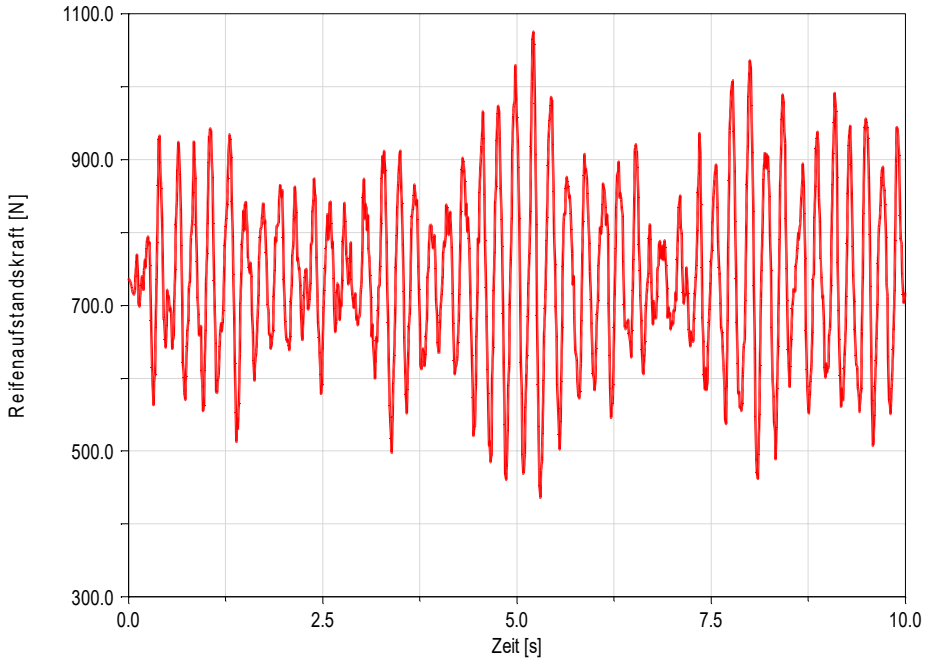


Abbildung A.1.: Verlauf der Radaufstandskraft hinten links

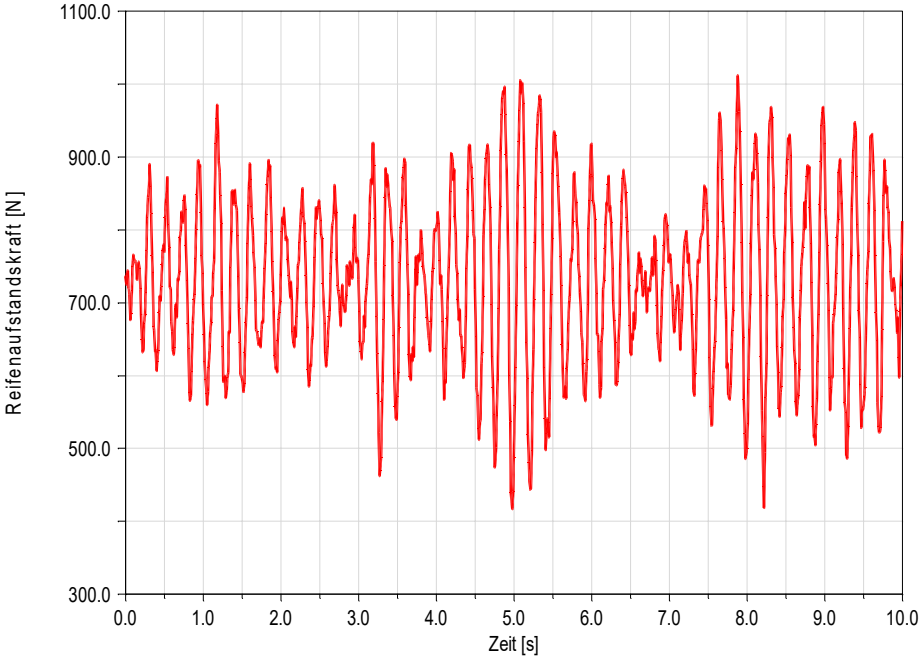


Abbildung A.2.: Verlauf der Radaufstandskraft vorne rechts

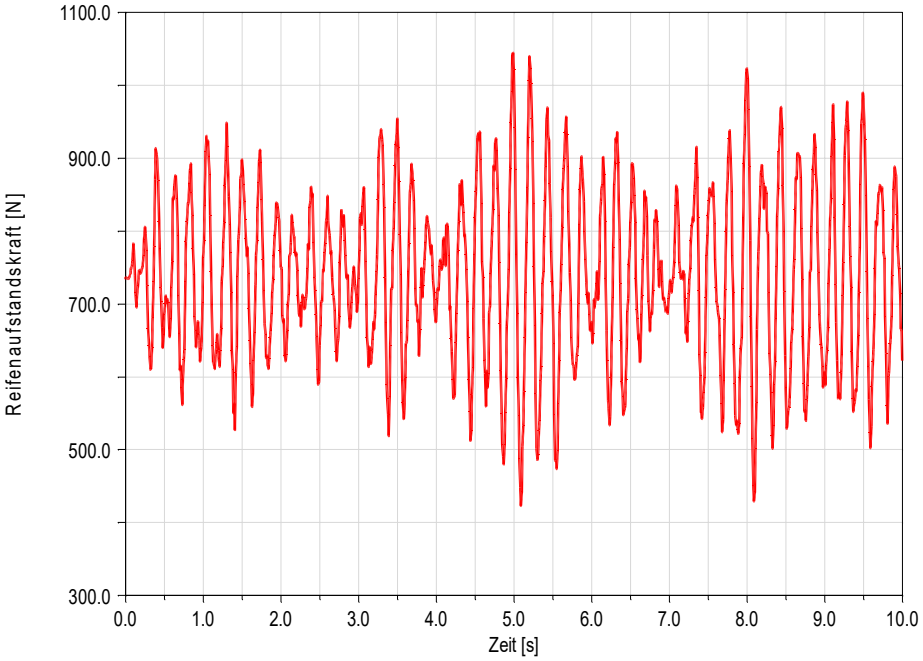


Abbildung A.3.: Verlauf der Radaufstandskraft hinten rechts

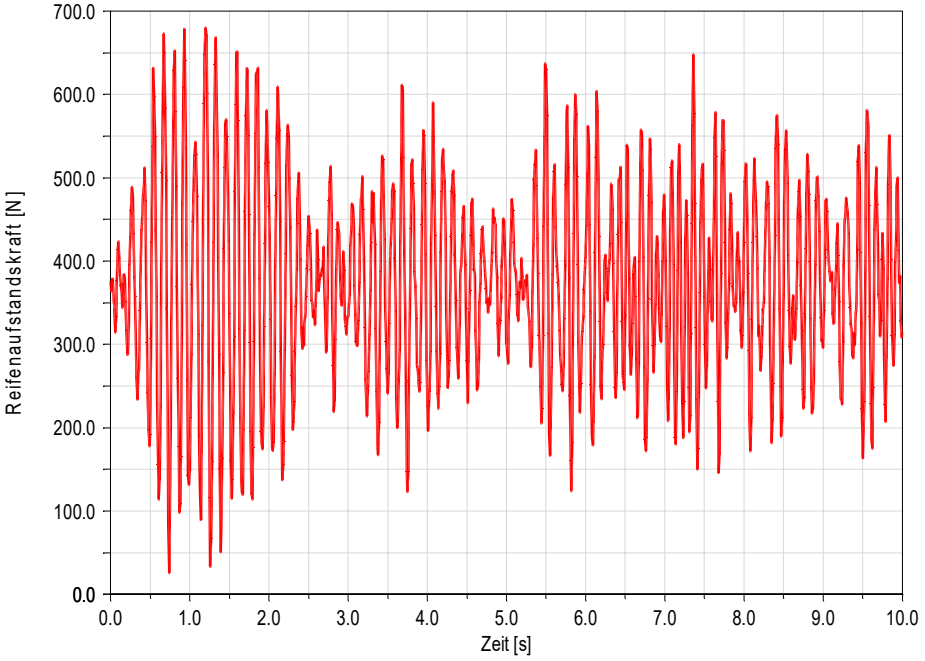


Abbildung A.4.: Verlauf der Radaufstandskraft vorne rechts bei halber Masse des Roboters

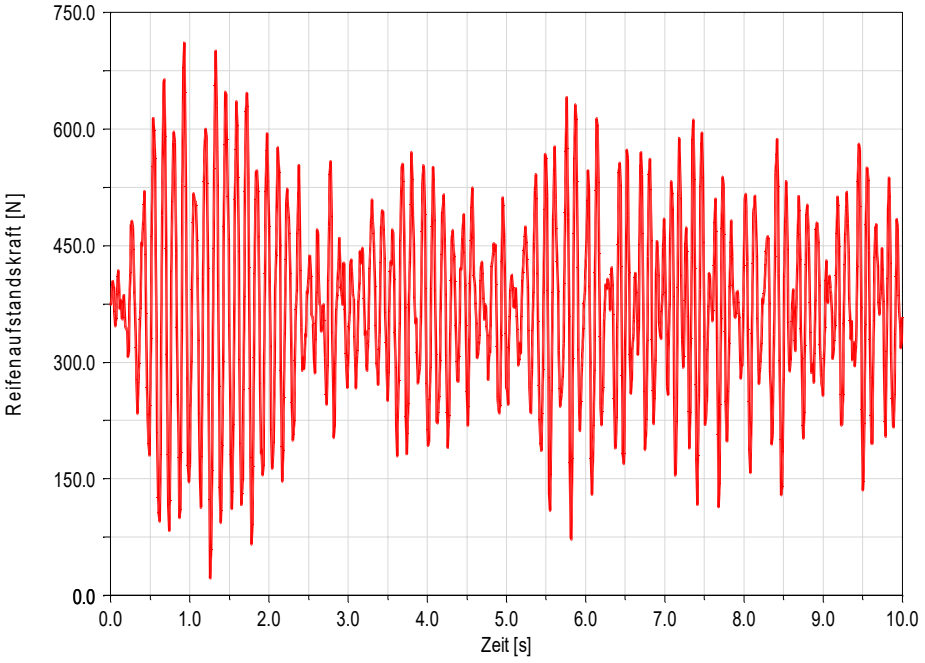


Abbildung A.5.: Verlauf der Radaufstandskraft vorne links bei halber Masse des Roboters

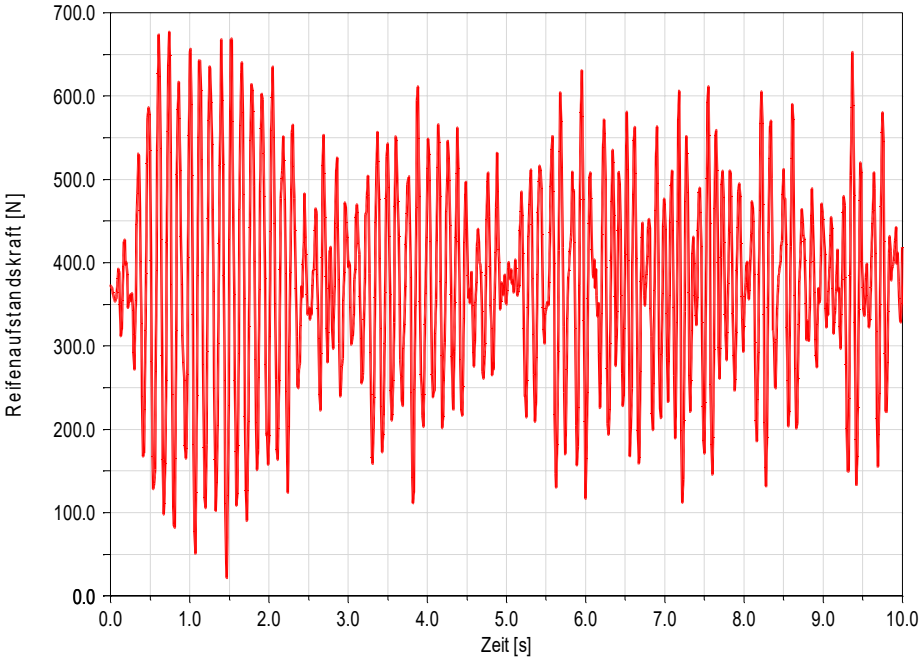


Abbildung A.6.: Verlauf der Radaufstandskraft hinten links bei halber Masse des Roboters

Anhang B.

Anhang Gesamtsimulation

B.1. Kreisradius bei drei gegebenen Punkten bestimmen

Es werden drei Punkte einer Ebene betrachtet, welche nicht auf einer gemeinsamen Linie liegen. Es soll nun der zugehörige Kreis zu diesen Punkten gefunden werden. Die allgemeine Kreisgleichung ist in Gleichung B.1 zu sehen. Jeder Punkt $P(x, y)$ welcher auf einem Kreis mit dem Mittelpunkt (x_m, y_m) liegt, muss diese Gleichung erfüllen und einen Abstand von r zu dem Kreismittelpunkt besitzen.

$$(x - x_m)^2 + (y - y_m)^2 = r^2 \tag{B.1}$$

Im weiteren wird nun Gleichung B.1 umgestellt um Gleichung B.2 zu bekommen.

$$x^2 - 2 \cdot x_m + x_m^2 + y^2 - 2 \cdot y \cdot y_m + y_m^2 = r^2 \tag{B.2}$$

In der Gleichung B.2 befinden sich die drei unbekannten Variablen (x_m, y_m, r), welche es zu bestimmen gilt. In Gleichung B.3 werden die Variablen neu geordnet, unbekannte Variablen befinden sich nun auf der linken Seite der Gleichung.

$$x_m^2 + y_2^2 - r^2 - 2 \cdot x_m \cdot x - 2 \cdot y_m \cdot y = -(x^2 + y^2) \tag{B.3}$$

Im nächsten Schritt werden drei neue Variablen (A, B, C) definiert, wie in den Gleichungen B.4, B.5 und B.6 zu sehen ist. Unter Berücksichtigung dieser neu definierten Variablen ergibt sich Gleichung B.7.

$$A := x_m^2 + y_m^2 - r^2 \tag{B.4}$$

$$B := 2 \cdot x_m \tag{B.5}$$

$$C := 2 \cdot y_m \tag{B.6}$$

$$A + B \cdot (-x) + C \cdot (-y) = -(x^2 + y^2) \tag{B.7}$$

Mit den drei gegebenen Punkten kann nun ein lineares Gleichungsystem aufgestellt werden, bestehend aus den Gleichungen B.8, B.9 sowie B.10. Dies kann nun gelöst werden, um den gewünschten Kreismittelpunkt (x_m, y_m) , sowie den Kreisradius r zu erhalten.

$$A + B \cdot (-x_1) + C \cdot (-y_1) = -(x_1^2 + y_1^2) \tag{B.8}$$

$$A + B \cdot (-x_2) + C \cdot (-y_2) = -(x_2^2 + y_2^2) \tag{B.9}$$

$$A + B \cdot (-x_3) + C \cdot (-y_3) = -(x_3^2 + y_3^2) \tag{B.10}$$

B.2. Berechnung des Vorschaupunktes

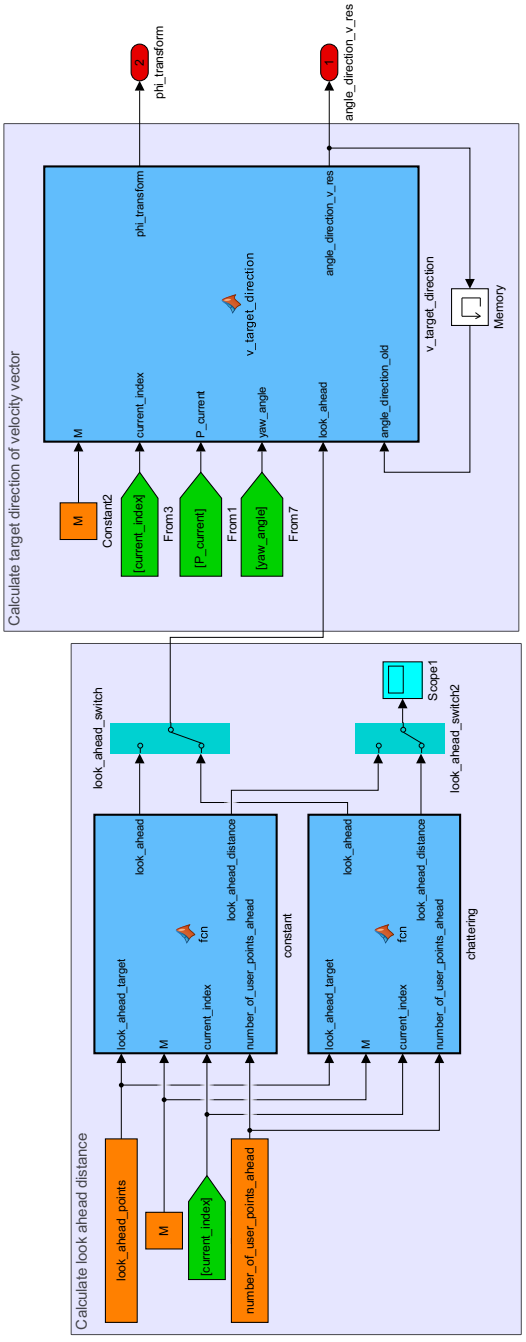


Abbildung B.1.: Berechnung des Vorschaupunktes

B.3. Twist des Roboters bestimmen

Die Tabelle B.1 beschreibt die in dem Block *Roboter-Twist bestimmen* verwendeten Parameter und den jeweiligen Wert. Dazu gehören die notwendigen Parameter für die Geschwindigkeitsregelung, die Regelung des Roboteraufbauwinkels, die Vorschaudistanz, welche sich aus der Anzahl an Punkten vor dem Roboter ergibt, sowie die maximal zugelassenen Abweichungen zwischen Soll- und Istwerten.

Modul Motion-Controller			
Variablenname	Bedeutung	Wert	Einheit
max_velocity_deviation_error	max. Geschw.-abweichung	5	$\frac{m}{s}$
p_v	Parameter für Geschw.-regelung	-0,8	-
k_v	Parameter für Geschw.-regelung	0,5	-
T_pt1_v	Zeitkonstante für PT1-Glied	0,1	s
T_pt2_v	Zeitkonstante für PT1-Glied	0,1	s
look_ahead_points	Anzahl an Vorschaupunkten	250	-
number_of_user_points_ahead	Nutzerpunkte in der Vorschau	4	-
max_phi_error	max. Abweichung im Aufbauwinkel	0,5	rad
epsilon	Parameter für Aufbauwinkelregelung	0,01	-
U	Parameter für Aufbauwinkelregelung	0,65	-

Tabelle B.1.: Parameter für den Block *Roboter-Twist bestimmen* in der Gesamtsimulation (Teil 1)

In der Tabelle B.2 wird der Parameter p für die Korrektur der Richtung des Geschwindigkeitsvektors bestimmt. In Abhängigkeit von der Abweichung zur Trajektorie *S_e*, wird der Parameter linear zwischen angegebenen Werten interpoliert. Ist die Abweichung größer als 2 Meter, wird für die Bestimmung des Parameters p linear extrapoliert. Der Parameter *ratio* wird nach demselbem Schema bestimmt, abhängig von dem Verhältnis zwischen Winkelgeschwindigkeit der Punktmasse und der Winkelgeschwindigkeit des Roboteraufbaus, bezeichnet als *omega_ratio*. Dieses Verhältnis liegt immer zwischen null und eins, dazwischen wird zur Bestimmung des Parameters *ratio* linear interpoliert.

Modul Motion-Controller			
S_e [m]	p [-]	omega_ratio [-]	ratio [-]
0	0	0	0,3
2	4	1	1

Tabelle B.2.: Parameter für den Block *Roboter-Twist bestimmen* in der Gesamtsimulation (Teil 2)

B.3.1. Regelung des Roboter-Aufbauwinkels

In diesem MATLAB-Simulink Modell, Abbildung B.2, wird die Variable *omega_ratio* bestimmt, welche für die proportionale Regelung des Geschwindigkeitvektors verwendet wird. Prinzipiell wird die Variable *omega_ratio* nach Gleichung B.11 bestimmt. Die

Variable *omega_trajectory_current* steht dabei für die Winkelgeschwindigkeit, welcher ein Massenpunkt entlang der Trajektorie erfährt und *omega_b* steht für die Winkelgeschwindigkeit des Roboteraufbaus. Fährt der Roboter wie ein PKW entlang der Trajektorie ($omega\_b = omega\_trajectory\_current$) ergibt sich $omega\_ratio = 1$, dreht sich der Roboteraufbau nicht mit, ergibt sich $omega\_ratio = 0$.

$$omega\_ratio = \frac{omega\_b}{omega\_trajectory\_current} \tag{B.11}$$

Bewegt sich der Roboter entlang eines Geradenstücks gilt $omega\_trajectory\_current = 0$, die Variable *omega_ratio* behält den Wert, den sie in der letzten Kurve hatte bei und ändert sich erst wieder bei befahren der nächsten Kurve. Dieser Fall ist in Abbildung B.3 zu sehen und als *Subsystem* in das Modell in Abbildung B.2 integriert. In diesem *Subsystem* - Modell wird der Variable *omg_traj* der Wert null zugewiesen, wenn ihr Betrag größer null ist (entspricht einer Kurvenfahrt), da für diesen Fall jener Block keine Relevanz besitzt, ist die Variable *omg_traj* null, wird ihr der Wert eins zugewiesen. Wenn diese Variable nun von null auf eins springt, von einem Zeitschritt zum nächsten, wird die Variable *omega_ratio* aktualisiert und bleibt für das befahrene Geradenstück auf diesem Wert.

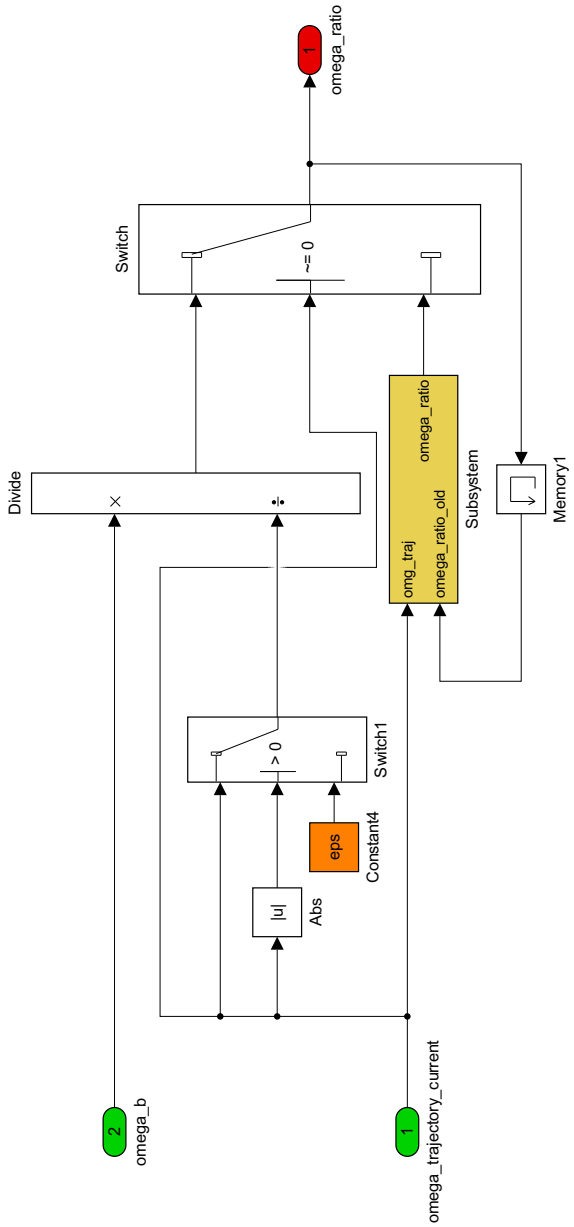


Abbildung B.2.: Berechnung des Verhältnisses der Winkelgeschwindigkeit

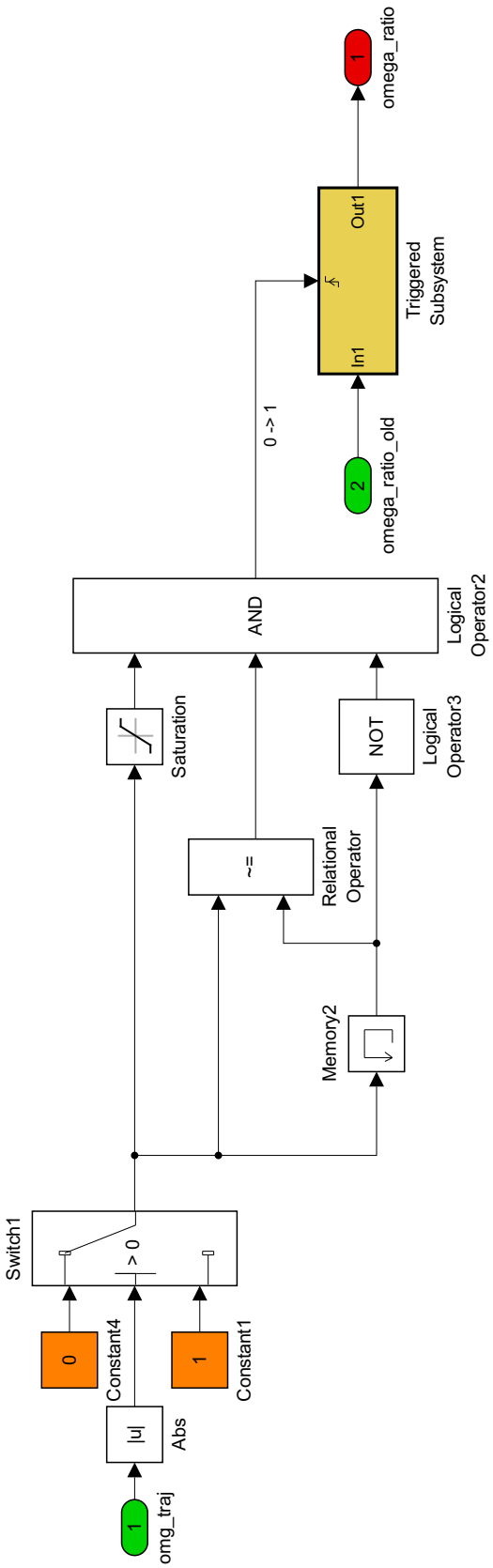


Abbildung B.3.: Berechnung des Verhältnisses des Winkelgeschwindigkeit - Subsystems

B.4. Modul Motion-Controller

In der Tabelle B.3 werden die im Block *Motion Controller* verwendeten Parameter aufgelistet. Es handelt sich dabei um die Zeitkonstanten der verwendeten PT1-Glieder, die Eigenwerte der Beobachter, die maximal zulässigen Abweichungen zwischen Soll- und Istwert, sowie die Parameter für die Lenkwinkel- und Antriebsdrehzahlregelung. Die Parameter *lambda_s* und *W_s*, welche im Block *Motion Controller* für die Regelung des Lenkwinkels verwendet werden, hängen wie in Gleichung B.12, sowie B.13 zu sehen ist, von dem in Tabelle B.3 definierten Parameter *U_s* ab.

$$lambda\_s = \sqrt{U\_s} \tag{B.12}$$

$$W\_s = 1,1 \cdot U\_s \tag{B.13}$$

Modul Motion-Controller			
Variablenname	Bedeutung	Wert	Einheit
T_csim	Zeitkonstante für PT1-Glied	0,005	s
T_current_rotation	Zeitkonstante für PT1-Glied	0,1	s
T_current_angle	Zeitkonstante für PT1-Glied	0,1	s
T_steering_angle	Zeitkonstante für PT1-Glied	0,05	s
$\lambda_{B\delta 1,2,3}$	Eigenwerte für Lenk-Beobachter	-5	$\frac{1}{s}$
max_steering_torque	max. Lenkmoment	60	Nm
min_steering_torque	min. Lenkmoment	-60	Nm
max_steering_angle_error	max. Lenkwinkelabweichung	30	Grad
max_steering_angle	max. Lenkwinkel	135	Grad
min_steering_angle	min. Lenkwinkel	135	Grad
U_s	Paramter für Lenkwinkelregelung	0,409	-
p	Parameter für Lenkwinkelregelung	1	-
P_motion	Parameter für Antriebsdrehzahlregelung	2	-
I_motion	Parameter für Antriebsdrehzahlregelung	0,5	-
a_w_motion	Parameter für Antriebsdrehzahlregelung	0,05	-
$\lambda_{B\omega 1,2}$	Eigenwerte für Raddrehzahl-Beobachter	-25	$\frac{1}{s}$
max_powertrain_torque	max. Antriebsmoment (ohne Übersetzung)	20	Nm
min_powertrain_torque	min. Antriebsmoment (ohne Übersetzung)	20	Nm

Tabelle B.3.: Parameter für den Block *Motion Controller* in der Gesamtsimulation

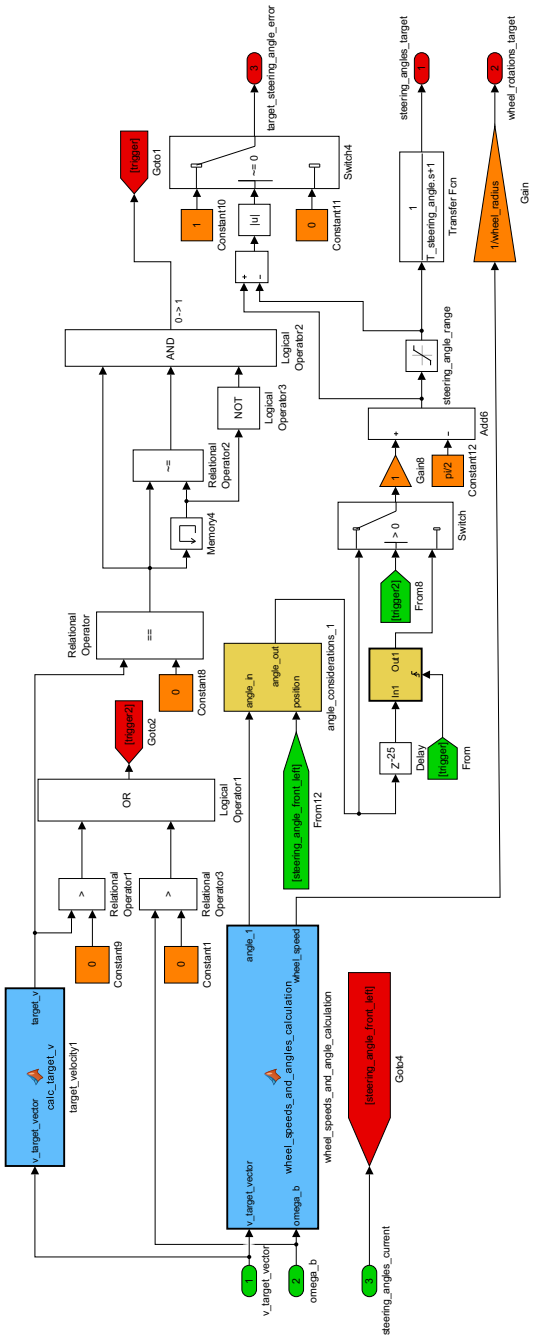


Abbildung B.4.: Berechnung der Raddrehzahlen und der Lenkwinkel des Roboters

B.4.1. Beobachter - Allgemeine Betrachtung

Ausgehend von einer Regelstrecke der Form

$$\frac{d\underline{x}}{dt} = \underline{\underline{A}} \cdot \underline{x} + \underline{b} \cdot u \ , \ y = \underline{c}^T \cdot \underline{x} \tag{B.14}$$

besitzt nach Horn & Dourdoumas (2004) der Beobachter das Modell

$$\frac{d\underline{\hat{x}}}{dt} = \left(\underline{\underline{A}} - \underline{\hat{b}} \cdot \underline{c}^T\right) \underline{\hat{x}} + \underline{b} \cdot u + \underline{\hat{b}} \cdot y, \tag{B.15}$$

wobei es nun durch die Einführung einer fiktiven Messgröße

$$\hat{y} = \underline{c}^T \cdot \underline{\hat{x}} \tag{B.16}$$

umgeformt wird:

$$\frac{d\underline{\hat{x}}}{dt} = \underline{\underline{A}} \cdot \underline{\hat{x}} + \underline{b} \cdot u + \underline{\hat{b}} \cdot (y - \hat{y}) \, . \tag{B.17}$$

Die Regelstrecke ist beobachtbar wenn sich die Eigenwerte der Systemmatrix

$$\underline{\underline{\hat{A}}} = \underline{\underline{A}} - \underline{\hat{b}} \cdot \underline{c}^T \tag{B.18}$$

beliebig platzieren lassen. Es gilt nun also den Parameter $\underline{\hat{b}}$ so festzulegen, dass $\underline{\underline{\hat{A}}}$ n vorgegebene Eigenwerte in der linken offenen s-Ebene besitzt. Es muss folglich gelten

$$det\left[s \cdot \underline{\underline{E}} - \underline{\underline{\hat{A}}}\right] = \prod_{i=1}^{n} (s - \zeta_i) \, . \tag{B.19}$$

Wie in Horn & Dourdoumas (2004) ausgeführt, lässt sich Beobachtbarkeit nachweisen, wenn folgende Matrix

$$\underline{\underline{B_y}} = \begin{pmatrix} c^T \\ c^T \cdot \underline{\underline{A}} \\ . \\ . \\ c^T \cdot \underline{\underline{A}}^{n-1} \end{pmatrix} \tag{B.20}$$

regulär ist. In den nächsten beiden Kapiteln B.4.2, sowie B.4.3, wird das jeweils verwendete Beobachtermodell vorgestellt.

B.4.2. Beobachter für das Lenk - Rückstellmoment

In Anlehnung an Moseberg (2016) erfolgt der Entwurf des Beobachters für das Rückstellmoment beim Lenken des Rades, nach der Modellvorstellung in Gleichung 4.38. Das Beobachtermodell ergibt sich somit zu

$$\underline{\hat{\dot{x}}}_\delta = \underline{\underline{A}}_\delta \cdot \underline{\hat{x}}_\delta + \underline{B}_\delta \cdot M_\delta + \underline{L}_\delta \cdot \left(y_\delta - \underline{C}_\delta \cdot \underline{\hat{x}}_\delta\right). \tag{B.21}$$

$$\underline{\underline{A}}_\delta = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & \frac{-1}{I_{zz}} \\ 0 & 0 & 0 \end{bmatrix} \tag{B.22}$$

$$\underline{B}_\delta = \begin{bmatrix} 0 \\ \frac{1}{I_{zz}} \\ 0 \end{bmatrix} \tag{B.23}$$

$$\underline{L}_\delta = \begin{bmatrix} l_{\delta 1} \\ l_{\delta 2} \\ l_{\delta 3} \end{bmatrix} \tag{B.24}$$

$$\underline{C}_\delta = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}^T \tag{B.25}$$

$$y_\delta = \delta \tag{B.26}$$

$$\underline{\hat{x}_\delta} = \begin{bmatrix} \hat{\delta} \\ \dot{\hat{\delta}} \\ \hat{M}_{r\delta} \end{bmatrix} \tag{B.27}$$

Die Elemente des Vektors $\underline{L}_\delta$ gilt es nun zu Bestimmen, die Beobachtbarkeitsmatrix

$$\underline{\underline{B}}_{y_\delta} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \frac{-1}{I_{zz}} \end{bmatrix} \tag{B.28}$$

ist regulär, da gilt

$$det\left(\underline{\underline{B}}_{y_\delta}\right) \neq 0 \rightarrow Beobachtbar! \tag{B.29}$$

Das MATLAB-Simulink Modell des erstellten Beobachtermodells ist in Abbildung B.5 zu sehen. Nach der Gleichung B.19 lässt sich mittels

$$det\left(s \cdot \underline{\underline{E}} - \underline{\underline{A}}_\delta + l_\delta \cdot \underline{C}_\delta^T\right) \stackrel{!}{=} (s - \lambda_{B\delta 1})(s - \lambda_{B\delta 2})(s - \lambda_{B\delta 3}) \tag{B.30}$$

unter Vorgabe der Eigenwerte $\lambda_{B\delta1}$,$\lambda_{B\delta2}$,$\lambda_{B\delta3}$ die Paramter des Vektors $\underline{L}_\delta$ folgendermaßen bestimmen:

$$l_{\delta1} = -(\lambda_{B\delta1} + \lambda_{B\delta2} + \lambda_{B\delta3}) \tag{B.31}$$

$$l_{\delta2} = \lambda_{B\delta1} \cdot \lambda_{B\delta2} + \lambda_{B\delta2} \cdot \lambda_{B\delta3} + \lambda_{B\delta1} \cdot \lambda_{B\delta3} \tag{B.32}$$

$$l_{\delta3} = J_\delta \cdot \lambda_{B\delta1} \cdot \lambda_{B\delta2} \cdot \lambda_{B\delta3} \tag{B.33}$$

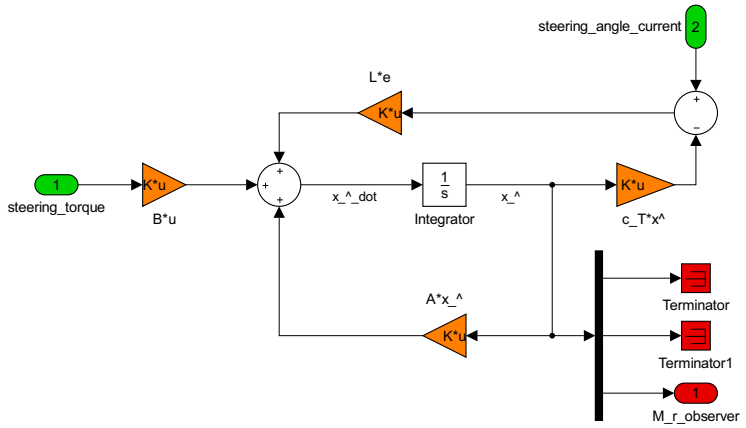


Abbildung B.5.: Luenberger-Beobachter für die Regelung des Lenkmoments

B.4.3. Beobachter für das Antriebs - Rückstellmoment

In Anlehnung an Moseberg (2016) erfolgt der Entwurf des Beobachters für das Rückstellmoment beim Antreiben des Rades, nach der Modellvorstellung in Gleichung 4.42. Das Beobachtermodell ergibt sich somt zu

$$\dot{\underline{\hat{x}}}_\omega = \underline{\underline{A_\omega}} \cdot \underline{\hat{x}}_\omega + \underline{B_\omega} \cdot M_\omega + \underline{L_\omega} \cdot \left(y_\omega - \underline{C_\omega} \cdot \underline{\hat{x}}_\omega\right). \tag{B.34}$$

$$\underline{\underline{A_\omega}} = \begin{bmatrix} 0 & \frac{-1}{I_{yy}} \\ 0 & 0 \end{bmatrix} \tag{B.35}$$

$$\underline{B_\omega} = \begin{bmatrix} \frac{1}{I_{yy}} \\ 0 \end{bmatrix} \tag{B.36}$$

$$\underline{L_\omega} = \begin{bmatrix} l_{\omega 1} \\ l_{\omega 2} \end{bmatrix} \tag{B.37}$$

$$\underline{C_\omega} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}^T \tag{B.38}$$

$$y_\omega = \omega \tag{B.39}$$

$$\underline{\hat{x}} = \begin{bmatrix} \hat{\omega} \\ \hat{M}_{r\omega} \end{bmatrix} \tag{B.40}$$

Die Elemente des Vektors $\underline{L_\omega}$ gilt es nun zu Bestimmen, die Beobachtbarkeitsmatrix

$$\underline{\underline{B_{y_\omega}}} = \begin{pmatrix} 1 & 0 \\ 0 & \frac{-1}{I_{yy}} \end{pmatrix} \tag{B.41}$$

ist regulär, da gilt

$$det\left(\underline{\underline{B_{y_\omega}}}\right) \neq 0 \rightarrow Beobachtbar! \tag{B.42}$$

Das MATLAB-Simulink Modell des erstellten Beobachtermodells ist in Abbildung B.6 zu sehen. Nach der Gleichung B.19 lässt sich mittels

$$det\left(s \cdot \underline{\underline{E}} - \underline{\underline{A}}_\omega + \underline{l}_\omega \cdot \underline{C}_\omega^T\right) \stackrel{!}{=} (s - \lambda_{B\omega 1})(s - \lambda_{B\omega 2}) \tag{B.43}$$

unter Vorgabe der Eigenwerte $\lambda_{B\omega 1}$, $\lambda_{B\omega 1}$ die Parameter des Vektors L_ω folgendermaßen bestimmen:

$$l_{\omega 1} = -(\lambda_{B\omega 1} + \lambda_{B\omega 2}) \tag{B.44}$$

$$l_{\omega 2} = -J_\omega \cdot \lambda_{B\omega 1} \cdot \lambda_{B\omega 2} \tag{B.45}$$

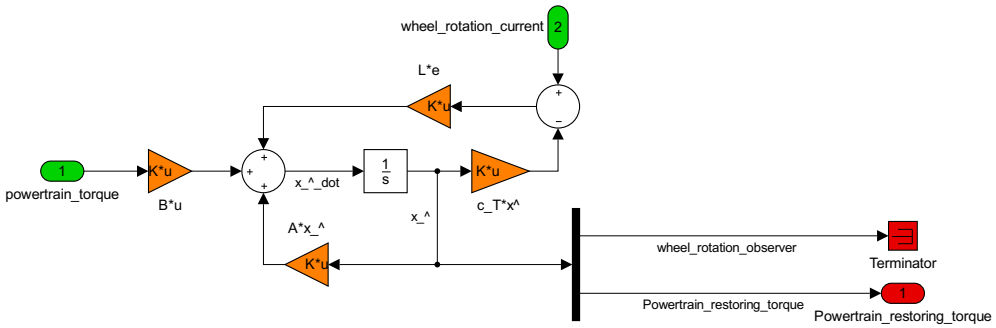


Abbildung B.6.: Luenberger-Beobachter für die Regelung des Antriebsmoment

B.5. Modul ADAMS/Car

In der Tabelle B.4 sind alle Parameter aufgelistet, welche für die Gesamtsimulation benötigt werden. Dazu gehören die Masse des Fahrzeuges, die relevanten Trägheitsmomente, der verwendete Reifenradius, sowie die Position der Räder im Roboterkoordinatensystem (Ursprung im Schwerpunkt des Roboters). Da diese thematisch dem Fahrzeug zugerechnet werden, sind diese unter dem Modul *ADAMS/Car* aufgeführt.

Modul ADAMS/Car			
Variablenname	Bedeutung	Wert	Einheit
mass	Masse des Fahrzeuges	296.09	kg
I_zz_wheel	z-Haupträgheitsmoment des Rades	3,2674	kg m²
I_yy_wheel	y-Hauptträgheitsmoment des Rades	1,3154	kg m²
wheel_radius	undeformierter Reifenradius	0,215	m
x_wheel_front_left	x-Koord. des linken vorderen Rades	-0,5021	m
y_wheel_front_left	y-Koord. des linken vorderen Rades	0,5	m
x_wheel_front_right	x-Koord. des rechten vorderen Rades	0,5021	m
y_wheel_front_right	y-Koord. des rechten vorderen Rades	0,5	m
x_wheel_rear_left	x-Koord. des linken hinteren Rades	-0,4979	m
y_wheel_rear_left	y-Koord. des linken hinteren Rades	-0,5	m
x_wheel_rear_right	x-Koord. des rechten hinteren Rades	0,4979	m
y_wheel_rear_right	y-Koord. des rechten hinteren Rades	-0,5	m

Tabelle B.4.: Parameter für den Block *ADAMS/Car* in der Gesamtsimulation

B.6. Hessesche Normalform

In diesem Kapitel wird nun erläutert wie man den vorzeichenbehafteten Abstand eines Punktes zu einer Geraden ermitteln kann, wie in Kapitel 4.3.4.3 benötigt. Ausgegangen wird zuerst von zwei gegebenen Punkten P_1 und P_2, siehe Gleichung B.46. Aus diesen Punkten wird eine Gerade definiert, siehe Gleichung B.47, der Richtungsvektor dieser Geraden wird in Gleichung B.48 als $\overrightarrow{v}$ bezeichnet. In Gleichung B.49 wird nun der Normalenvektor auf den Richtungsvektor gebildet. Die Gleichung B.50 liefert die Bedingung für die Bestimmung des normierten Normalenvektors, der Vektor $\overrightarrow{p}_1$ ist dabei der Vektor des Punktes P_1 zum jeweiligen Koordinatenursprung. Somit kann mit der Gleichung B.51 die allgemeine Form der Hesseschen Normalenform definert werden. Es hat nun jeder Verbindungsvektor eines Punktes auf der Gerade g mit dem Koordinatenursprung den Abstand c, wenn ein Skalarprodukt mit dem normierten Normalenvektor berechnet wird. Möchte man nun den Abstand eines Punktes zu dieser Geraden bestimmen, kann man wie in Gleichung B.52 zu sehen, einen Vektor $\overrightarrow{q}$ definieren, welcher zwischen dem beliebigen Punkt Q und dem Koordinatenursprung gelegt wurde und den Abstand d zur Gerade g bestimmen.

$$2\ Punkte : P_1 = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix},\ P_2 = \begin{pmatrix} x_2 \\ y_2 \end{pmatrix} \tag{B.46}$$

$$Gerade\ g :\ \overrightarrow{x} = P_1 + r \cdot \overrightarrow{P_2 - P_1} \tag{B.47}$$

$$\overrightarrow{v} = \overrightarrow{P_2 - P_1} \tag{B.48}$$

$$\overrightarrow{n} = \begin{pmatrix} v_2 \\ -v_1 \end{pmatrix},\ \overrightarrow{v} \perp \overrightarrow{n} \tag{B.49}$$

$$\overrightarrow{n_0} = \begin{cases} +\frac{\overrightarrow{n}}{|\overrightarrow{n}|} & \text{wenn } \overrightarrow{p_1} \cdot \overrightarrow{n} \geq 0 \\ -\frac{\overrightarrow{n}}{|\overrightarrow{n}|} & \text{wenn } \overrightarrow{p_1} \cdot \overrightarrow{n} < 0 \end{cases} \tag{B.50}$$

$$\overrightarrow{n_0} \cdot \overrightarrow{p_1} = c \tag{B.51}$$

$$\overrightarrow{n_0} \cdot \overrightarrow{q} - c = d \tag{B.52}$$

Appendix

Literaturverzeichnis

ADAMS/Car®, 2001: *ADAMS/Car Training Guide*.

Andersen H., Chong Z.J., Eng Y.H., Pendleton S., Jr. M.H.A., 2016: *Geometric Path Tracking Algorithm for Autonomous Driving in Pedestrian Environment*, in: IEEE-International Conference on Advanced Intelligent Mechatronics (AIM), S. 1669–1674.

Bartolini G., Ferrara A., Levant A., Usai E., 1999: *On second order sliding mode controllers*, in: Lecture Notes in Control and Information Sciences.

de Brabandt I., 2015: *Mathematik in der Oberstufe*. `http://www.mathematik-oberstufe.de/vektoren/a/abstand-punkt-gerade-formel.html` (Zugriff: 2017-07-12)

Campion G., Bastin G., D'Andréa-Novel B., 1996: *Structural Properties and Classification of Kinematic and Dynamic Models of Wheeled Mobile Robots*, in: IEEE-Transactions on Robotics and Automation 12, S. 47–62.

Connette C., 2013: *Kinematische Modellierung und Regelung omnidirektionaler, nicht-holonomer Fahrwerke*, Dissertation.

Corke P., 2011: *Robotics, Vision and Control*, Springer-Verlag Berlin Heidelberg.

Coulter R.C., 1992: *Implementation of the Pure Pursuit Tracking Algorithm*, in: Robotics Institute of the Carnegie Mellon University in Pittsburgh, Pennsylvania.

Dietrich A., Wimböck T., Albu-Schäffer A., Hirzinger G., 2011: *Singularity Avoidance for Nonholonomic, Omnidirectional Wheeled Mobile Platforms with Variable Footprint*, in: IEEE International Conference on Robotics and Automation, Shanghai International Conference Center.

Efimov D., Polyakov A., Fridman L., Perruquetti W., Richard J.P., 2015: *Delayed Sliding Mode Control*, in: Automatica, Elsevier.

Gfrerrer A., 2. Ausgabe 2014: *Mobile Robots - Lectur notes of the kinematic part of the lecture*, in: Institute of Geometry. Graz, University of Technology.

Giordano P.R., Fuchs M., Albu-Schäffer A., Hirzinger G., 2009: *On the Kinematic Modeling and Control of a Mobile Platform Equipped with Steering Wheels and Movable Legs*, in: IEEE International Conference on Robotics and Automation.

Graf B., Hans M., Schraft R., 2004: *Caro-O-bot II - Development of a Next Generation Robotic Home Assitant*, in: Autonomous Robots 16, S. 193–205.

Gruber C., Hofbaur M., 2016: *Remarks on the classification of wheeled mobile robots*, in: Mechanical Sciences - Open Access.

Heinzmann, 2017: *Synchronmotoren*. http://www.heinzmann.com/de/elektromotoren/scheibenlaeufermotor/synchronmotor (Zugriff: 2017-07-27)

Hägele M., 2009: *World Robotics 2009: Service Robots*. http://www.worldrobotics.org. (Zugriff: 2017-08-11)

Hirschberg W., Waser H., 2016: *Fahrzeugdynamik - Sommersemester 2016*, in: Institut für Fahrzeugtechnik.

Hoffmann G., Tomlin C., Montemerlo M., Thrun S., 2007: *Autonomous Automobile Trajectory Tracking for Off-Road Driving: Controller Design, Experimental Validation and Racing*, in: IEEE-Proceedings of the 2007 American Control Conference, S. 2296–2301.

Horn M., 2015: *Regelungstechnik*, in: Institut für Regelungs- und Automatisierungstechnik.

Horn M., Dourdoumas N., 2004: *Regelungstechnik - Rechnerunterstützer Entwurf zeitkontinuierlicher und zeitdiskreter Regelkreise*, Pearson Studium.

ISO 3888-2:2011, 2011: *Passenger cars - Test track for a severe lane-change manoeuvre - Part 2: Obstacle avoidance*, Standard, International Organization for Standardization, Geneva, CH.

ISO 8608:2016, 2016: *Mechanical vibration – Road surface profiles – Reporting of measured data*, Standard, International Organization for Standardization, Geneva, CH.

Lundgren M., 2003: *Path Tracking and Obstacle Avoidance for a Miniature Robot*, Diplomarbeit.

Moseberg J.E., 2016: *Regelung der Horizontalbewegung eines überaktuierten Fahrzeugs unter Berücksichtigung von Realisierungsanforderungen*, Dissertation.

Moseberg J.E., Roppenecker G., 2014: *Steuerung und Regelung der horizontalen Fahrzeugbewegung mit Einzelradaktorik*, in: at - Automatisierungstechnik 2014, 62(3): 216-225.

Nikolov I., 2011: *Maschinelles Lernen zur Optimierung einer autonomen Fahrspurführung*, Diplomarbeit.

Oubbati M., 2009: *Einführung in die Robotik*, Institut für Neuroinformatik, Universität Ulm, Albert-Einstein-Allee-11.

Pacejka H., Besselink I., 1997: *Magic Formula Tyre Model with Transient Properities*, in: Vehicle System Dynamics, S. 234–249.

Rauh J., Mössner-Beigel M., 2008: *Tyre simulation challenges*, in: Vehicle System Dynamics, 46:S1, S. 49–62.

Rill G., Schaeffer T., 2014: *Grundlagen und Methodik der Mehrkörpersimulation*, Band 2.Auflage, Springer Fachmedien Wiesbaden.

Schraft R., Hägele M., Wegener K., 2004: *Servicerobotik*, in: Service Roboter Vision - München.

Schramm D., Hiller M., Bardini R., 2013: *Modellbildung und Simulation der Dynamik von Kraftfahrzeugen*, Springer-Verlag GmbH.

Siciliano B., Khatib O., (Hrsg.), 2008: *Springer Handbook of Robotics*, Springer-Verlag Berlin-Heidelberg.

Snider J.M., 2009: *Automatic Steering Methods for Autonomous Automobile Path Tracking*, in: Robotics Institute of the Carnegie Mellon University in Pittsburg, Pennsylvania, (CMU-RI-TR-09-08).

Straumann N., 2015: *Theoretische Mechanik*, Springer Spektrum, 2 Auflage.

Todoran G., Bader M., 2016: *Expressive Navigation and Local Path-Planning of Independent Steering Autonomous Systems*, in: , S. 4742–4749.