Entwicklung und Implementierung eines Algorithmus zur Berechnung kooperativer Fahrmanöver von autonomen Fahrzeugen

Masterarbeit zur Erlangung des Grades M. Sc.

an der Fakultät für Maschinenwesen der Technischen Universität München

Aufgabensteller	Univ.-Prof. Dr.-Ing. Markus Lienkamp
	Lehrstuhl für Fahrzeugtechnik
Betreuer	**Christian Knies**, **M. Sc.**
	Lehrstuhl für Fahrzeugtechnik
Eingereicht von	Leonhard Hermansdorfer, B. Sc.
	Matrikelnummer: 03628509
Ausgabe am	23.07.2017
Eingereicht am	22.01.2018

Lehrstuhl für Fahrzeugtechnik
Fakultät für Maschinenwesen
Technische Universität München

Aufgabenstellung

Entwicklung und Implementierung eines Algorithmus zur Berechnung kooperativer Fahrmanöver von autonomen Fahrzeugen

Kooperatives Fahren bedeutet, dass sich Fahrzeuge mittels V2X-Kommunikation über ihre geplanten Fahrmanöver abstimmen und sich dabei gegenseitig unterstützen. Dadurch soll die natürliche Kooperation zwischen Autofahrern verbessert oder, im Fall des automatisierten Fahrens, ersetzt werden.

Im Rahmen dieser Masterarbeit sollen bestehende Ansätze zur kooperativen Trajektorienplanung recherchiert, zusammengefasst und bewertet werden. Der bestbewertete Ansatz soll in Matlab simulativ umgesetzt werden.

Folgende Punkte sind in der Masterarbeit zu bearbeiten:

1. Literaturrecherche:
   - Umfassende Analyse bestehender Ansätze zur kooperativen Trajektorienplanung
   - Mathematische Verfahren, die in der kooperativen Trajektorienplanung Anwendung finden
2. Bewertung:
   - Kategorisierung und Analyse der Vor- und Nachteile der einzelnen Ansätze
   - Bewertung und Auswahl des geeignetsten Ansatzes
3. Umsetzung:
   - Programmierung einer Testumgebung in Matlab
   - Implementierung der bestbewerteten Trajektorienplanung
4. Evaluation
   - Evaluation des Trajektorienplaners anhand unterschiedlicher Szenarien in der erstellten Testumgebung

Die Ausarbeitung soll die einzelnen Arbeitsschritte in übersichtlicher Form dokumentieren. Der Kandidat verpflichtet sich, die Semesterarbeit selbständig durchzuführen und die von ihm verwendeten wissenschaftlichen Hilfsmittel anzugeben.

Die eingereichte Arbeit verbleibt als Prüfungsunterlage im Eigentum des Lehrstuhls und darf Dritten nur unter Zustimmung des Lehrstuhlinhabers zugänglich gemacht werden.

Ausgabe: 23.07.2017 Abgabe: 22.01.2018

Prof. Dr.-Ing. M. Lienkamp

Betreuer: **Christian Knies**, M. Sc.

Lehrstuhl für Fahrzeugtechnik
Fakultät für Maschinenwesen
Technische Universität München

Geheimhaltungsverpflichtung

Herr **Hermansdorfer, Leonhard**

Im Rahmen der Angebotserstellung und der Bearbeitung von Forschungs- und Entwicklungsverträgen erhält der Lehrstuhl für Fahrzeugtechnik der Technischen Universität München regelmäßig Zugang zu vertraulichen oder geheimen Unterlagen oder Sachverhalten industrieller Kunden, wie z.B. Technologien, heutige oder zukünftige Produkte, insbesondere Prototypen, Methoden und Verfahren, technische Spezifikationen oder auch organisatorische Sachverhalte.

Der Unterzeichner verpflichtet sich, alle derartigen Informationen und Unterlagen, die ihm während seiner Tätigkeit am Lehrstuhl für Fahrzeugtechnik zugänglich werden, strikt vertraulich zu behandeln.

Er verpflichtet sich insbesondere

- derartige Informationen betriebsintern zum Zwecke der Diskussion nur dann zu verwenden, wenn ein ihm erteilter Auftrag dies erfordert,
- keine derartigen Informationen ohne die vorherige schriftliche Zustimmung des betreffenden Kunden an Dritte weiterzuleiten,
- keine Fotografien, Zeichnungen oder sonstige Darstellungen von Prototypen oder technischen Unterlagen hierzu anzufertigen,
- auf Anforderung des Lehrstuhls für Fahrzeugtechnik oder unaufgefordert spätestens bei seinem Ausscheiden aus dem Lehrstuhl für Fahrzeugtechnik alle Dokumente und Datenträger, die derartige Informationen enthalten, an Lehrstuhl für Fahrzeugtechnik zurückzugeben.

Eine besondere Sorgfalt gilt im Umgang mit digitalen Daten:

- Kein Dateiaustausch über Dropbox, Skydrive o.ä.
- Keine vertraulichen Informationen unverschlüsselt über Email versenden.
- Wenn geschäftliche Emails mit dem Handy synchronisiert werden, darf dieses nicht in die Cloud (z.B. iCloud) synchronisiert werden, da sonst die Emails auf dem Server des Anbieters liegen.
- Die Kommunikation sollte nach Möglichkeit über die (my)TUM-Mailadresse erfolgen. Diese Emails dürfen nicht an Postfächer anderer Emailprovider (z.B.: gmail.com) weitergeleitet werden.

Die Verpflichtung zur Geheimhaltung endet nicht mit dem Ausscheiden aus dem Lehrstuhl für Fahrzeugtechnik, sondern bleibt 5 Jahre nach dem Zeitpunkt des Ausscheidens in vollem Umfang bestehen.

Der Unterzeichner willigt ein, dass die Inhalte seiner Studienarbeit in darauf aufbauenden Studienarbeiten und Dissertationen mit der nötigen Kennzeichnung verwendet werden dürfen.

Datum: 23.07.2017

Unterschrift: _______

Lehrstuhl für Fahrzeugtechnik
Fakultät für Maschinenwesen
Technische Universität München

Erklärung

Ich versichere hiermit, dass ich die von mir eingereichte Abschlussarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Garching, den 22.01.2018

_____

Leonhard Hermansdorfer, B. Sc.

Inhaltsverzeichnis

Abkürzungsverzeichnis

ABS	Antiblockiersystem
BKV	Bremskraftverstärker
Fzg	Fahrzeug
GUI	Graphical user interface
IDM	Intelligent Driver Model
Lkw	Lastkraftwagen
MCTS	Monte-Carlo Tree Search
MILP	Mixed-Integer Linear Programming
NEFZ	Neuer Europäischer Fahrzyklus
Pkw	Personenkraftwagen
PMP	Partial Motion Planning
RRT	Rapidly-exploring Random Tree
V2I	Vehicle-to-Infrastructure (Kommunikation)
V2V	Vehicle-to-Vehicle (Kommunikation)
V2X	Vehicle-to-X (Kommunikation)

Formelzeichen

Formelzeichen	Einheit	Beschreibung
$\boldsymbol{A}_{Veh}$	-	Aktionsmatrix aller Fahrzeuge eines Verkehrsszenarios
$\boldsymbol{a}_{act,i}$	-	Aktionsvektor mit allen zulässigen Aktionen von Fahrzeug i
$\boldsymbol{a}_{act,def}$	-	Aktionsvektor mit allen standardmäßig wählbaren Aktionen
$a_{act,ij}$	-	Konkrete Aktion j von Fahrzeug i
$a_{act,res}$	-	Permutation aus Aktionsmatrix
$\boldsymbol{a}_i$	m/s²	Beschleunigungsvektor mit x- und y-Werten von Fahrzeug i
a_i	m/s²	Beschleunigung in Längsrichtung von Fahrzeug i
$a_{=0}$	m/s²	Standardwert für Fahren mit konstanter Geschwindigkeit
$a_{>0}$	m/s²	Standardwert für Beschleunigung
$a_{<0}$	m/s²	Standardwert für Verzögerung
a_{IDM}	m/s²	Standard-Beschleunigungswert aus IDM
a_{wunsch}	m/s²	gewünschte/komfortable Beschleunigung (IDM)
b_{FSP}	m	Breite einer Fahrspur
b_{max}	m/s²	maximale Verzögerung (IDM)
b_{wunsch}	m/s²	gewünschte/komfortable Verzögerung (IDM)
C_p	-	Konstante zur Beeinflussung des Expansion-Exploration-Dilemmas (MCTS-Algorithmus)
$\boldsymbol{f}_{long,i}$	-	Longitudinale Kraft auf einen Knoten von Fahrzeug i (Methode der elastischen Bänder)
$\boldsymbol{f}_{lat,i}$	-	Laterale Kraft auf einen Knoten von Fahrzeug i (Methode der elastischen Bänder)
$\boldsymbol{f}_{road,i}$	-	Straßenrandkraft auf einen Knoten von Fahrzeug i (Methode der elastischen Bänder)
$\boldsymbol{f}_{coop,i}$	-	Kraft zwischen kooperierenden Fahrzeugen auf einen Knoten von Fahrzeug i (Methode der elastischen Bänder)
$\boldsymbol{f}_{obst,i}$	-	Hinderniskraft auf einen Knoten von Fahrzeug i (Methode der elastischen Bänder)
f_i	-	Zustandsübergangsfunktion von Fahrzeug i
i_{FSP}	-	Anzahl der Fahrspuren
J_a	-	Kostenwert für Beschleunigung und Verzögerung
$J_{FSPrechts}$	-	Kostenwert für Nicht-Einhalten des Rechtsfahrgebots
J_{ges}	-	Kostenwert aus Summe der Einzelkosten
J_{Koll}	-	Kostenwert für Kollisionen
$J_{safedis}$	-	Kostenwert für Unterschreitung des Sicherheitsabstandes
J_{SPW}	-	Kostenwert für einen Spurwechsel
J_v	-	Kostenwert für Abweichung von der Wunschgeschwindigkeit

$j_{treedepth}$	-	Wert für die relative Baumtiefe, von der die aktuell betrachteten Kosten stammen
k_{def}	-	Anzahl an standardmäßig wählbaren Aktionen
$l_{Fzg,i}$	m	Radstand von Fahrzeug i
n	-	Anzahl der Spiele des aktuell betrachteten Knotens (MCTS-Algorithmus)
n_{parent}	-	Anzahl der Spiele des Elternknotens (MCTS-Algorithmus)
P	-	Menge an Fahrzeugen eines Verkehrsszenarios
p_i	-	Eigenschaften von Fahrzeug i
SPW_{links}	-	Standardaktion für Spurwechsel nach links
SPW_{rechts}	-	Standardaktion für Spurwechsel nach rechts
S_{FBL}	-	geometrische Repräsentation des Fahrbahnlayouts
S_{VS}	-	geometrische Repräsentation des Verkehrsszenarios
s_0	m	Minimalabstand zu vorausfahrendem Fahrzeug (linear Anteil aus IDM)
s_1	m	Minimalabstand zu vorausfahrendem Fahrzeug (nichtlinearer Anteil aus IDM)
s_{gap}	m	realer Abstand zu vorausfahrendem Fahrzeug
s_{gap}^*	m	gewünschter Abstand zu vorausfahrendem Fahrzeug (IDM)
T_{gap}	s	Folgezeit zu vorausfahrendem Fahrzeug (IDM)
T_{SimIDM}	s	Simulationszeit in Simulation-Phase für IDM-Berechnung
t	s	Zeit
Δt	s	Zeitschritt
Δt_{SimIDM}	s	Simulationsschrittweite in Simulation-Phase für IDM-Berechnung
t_0	s	initialer Zeitpunkt
t_{end}	s	Endzeitpunkt/Planungshorizont
t_{start}	s	Startzeitpunkt
$\boldsymbol{v}_i$	m/s	Geschwindigkeitsvektor mit x- und y-Werten von Fahrzeug i
v_i	m/s	Geschwindigkeit in Längsrichtung von Fahrzeug i
Δv	m/s	Geschwindigkeitsdifferenz zu vorausfahrendem Fahrzeug
$v_{max,i}$	m/s	Maximalgeschwindigkeit von Fahrzeug i
$v_{wunsch,i}$	m/s	Wunschgeschwindigkeit von Fahrzeug i
$w_{FzgTyp,(\cdot)}$	-	Gewichtungsfaktor für die Kostenfunktion $(\cdot)$ und den jeweiligen Fahrzeugtyp
$\boldsymbol{X}$	-	Zustandsraum
$\boldsymbol{X}_{Veh}$	-	Zustandsmatrix aller Fahrzeuge eines Verkehrsszenarios
$\bar{X}$	-	gemittelter Gewinn pro Spiel (MCTS-Algorithmus)
$\bar{X}_{Knoten}$	-	angepasster Wert für die „gemittelten“ Kosten in der Selection-Phase
$\boldsymbol{x}_i$	-	Zustandsvektor von Fahrzeug i
$\boldsymbol{x}_{pos,i}$	m	Vektor mit x- und y-Koordinaten von Fahrzeug i
x_i	m	x-Position von Fahrzeug i
x_{BSP}^{start}	m	Startposition der Beschleunigungsspur
x_{BSP}^{end}	m	Endposition der Beschleunigungsspur

x_{vFSP}^{start}	m	Startposition der verengten Fahrspur
x_{vFSP}^{end}	m	Endposition der verengten Fahrspur
y_i	m	y-Position von Fahrzeug i
$y_{FSP,i}$	-	Aktuelle Fahrspur von Fahrzeug i
δ_{IDM}	-	Beschleunigungsexponent (IDM)
δ_i	°	Lenkwinkel an der Vorderachse von Fahrzeug i
$\delta_{max,i}$	°	Maximaler Lenkwinkel an der Vorderachse von Fahrzeug i
θ_i	°	Orientierung bezogen auf globale x-Achse von Fahrzeug i
Ω^{obst}	-	geometrische Repräsentation der Hindernisse
Ω^{road}	-	geometrische Repräsentation der Fahrbahn
Ω^{veh}	-	geometrische Repräsentation der Fahrzeuge

1 Einleitung

Das selbstfahrende Automobil – die technischen Fortschritte im Bereich des autonomen Fahrens und vor allem die vielversprechenden Ankündigungen vieler Unternehmen lassen vermuten, dass dessen Markteinführung kurz bevorsteht. Vielen Menschen ist dabei nicht bewusst, dass der Ausdruck „selbstfahrendes Automobil“ eine Tautologie darstellt, also eine Aussage doppelt wiedergibt [1]. Der Begriff Automobil setzt sich aus dem griechischen Wort *autós* („selbst, eigen, persönlich“) [2] und dem lateinischen Wort *mobilis* („beweglich“) [3] zusammen und impliziert bereits die Fähigkeit sich selbst bzw. eigenständig zu bewegen. Woher kommt die Diskrepanz zwischen der Bedeutung des Begriffs und seiner tatsächlichen Ausprägung, wie sie täglich bei Millionen von Menschen in Deutschland zum Tragen kommt?

Die Antwort auf diese Frage findet sich am Anfang der Automobilentwicklung vor über 125 Jahren. Zu dieser Zeit war das Pferdefuhrwerk das überwiegend genutzte Verkehrsmittel [4, S. 2]. Die Einführung des Automobils bedeutete eine zunehmende Unabhängigkeit von Fuhrwerken. Es konnte sich ohne die Unterstützung von Tieren und damit „selbst bewegen“, worauf die ursprüngliche Bedeutung des Begriffs Automobil abzielt. Ironischerweise Weise ging mit der Erfindung des Automobils „(…) eine gewisse Form der Autonomie des Gefährts verloren (…)“, wie Maurer [5, S. 2] feststellt. Weiter führt er aus, dass Kutschenpferden durch Dressur beigebracht wurde, Verkehrsregeln zu beachten und in bestimmten Situationen entsprechend zu reagieren. Unter Umständen konnten sie das Gespann sogar ohne fahrtauglichen oder komplett ohne Kutscher bewegen. Letztendlich waren sie auch in der Lage, das Gespann in einen sicheren Zustand zu überführen. Er schließt mit der Feststellung ab, dass „Das autonome Automobil […] dem Fahrzeug seine verloren gegangene Autonomie zurückgeben, ja die historische Form noch weit übertreffen [will]“ [5, S. 3].

Auf dem Weg zum autonomen Fahren werden die heutigen Fahrzeuge mit einer stetig anwachsenden Zahl an Fahrerassistenzsystemen ausgestattet. Audi stellte bspw. 2017 die Oberklasse-Limousine A8 vor, die als erstes Serienfahrzeug das SAE-Level 3 erfüllen und bis 60 km/h selbstständig in Staus fahren können soll [6]. Weitere Hersteller werden in absehbarer Zukunft auch das SAE-Level 3 anbieten können, sodass der Anteil solcher Fahrzeuge im Straßenverkehr kontinuierlich steigt. Je mehr Fahrzeuge sich ohne (aufmerksamen) Fahrer auf den Straßen befinden, desto wichtiger wird es, die Kommunikation zwischen den Verkehrsteilnehmer sicher zu stellen. Auf Konfliktsituationen im Verkehrsgeschehen, die bisher zwischenmenschlich auf nonverbaler Ebene gelöst wurden, müssen auch von Fahrzeugen unterschiedlicher Automatisierungsgrade adäquate Antworten gefunden werden. Aktuell werden solche Situationen durch die Fahrerassistenzsysteme eines Fahrzeugs individuell gelöst. Eine Absprache unter mehreren Verkehrsteilnehmern findet nicht statt. Damit der Verkehr reibungslos ablaufen kann, ist jedoch eine Zusammenarbeit aller Beteiligten zwingend erforderlich [7, S. 1143]. Zusätzlich ergeben sich daraus Chancen, das Fahren sicherer und effizienter zu machen [8]. Damit dieses technisch anspruchsvolle Vorhaben gelingen kann, ist ein Teilaspekt der Zusammenarbeit besonders wichtig: der der Abstimmung über und Entscheidung für bestimmte Fahrmanöver, mit denen verschiedenen Verkehrssituationen

bestmöglich begegnet werden kann. Diese Masterarbeit setzt sich mit der Planung solcher kooperativen Fahrmanöver auseinander. Im Rahmen dieser Masterarbeit werden bestehende Ansätze zur Planung kooperativer Fahrmanöver recherchiert, zusammengefasst und bewertet. Im Anschluss erfolgt eine Anpassung und Weiterentwicklung des am besten bewerteten Verfahrens, das dann in Matlab implementiert wird. Der Fokus liegt auf Verkehrsszenarien im normal fließenden Verkehr auf mehrspurigen Fahrbahnen mit höheren Fahrgeschwindigkeiten, wie sie auf Autobahnen auftreten. Gegenverkehr und Kreuzungssituationen werden nicht behandelt.

Zu Beginn der Arbeit wird der Stand der Technik der Methoden zur kooperativen Trajektorienplanung untersucht (Kapitel 2). Erläutert werden die Motivation und die Ziele des kooperativen Fahrens sowie die Herausforderungen, die es zu meistern gilt (Unterkapitel 2.1). Die Grundlagen der Trajektorienplanung in der Robotik und für Einzelfahrzeuge werden aufgegriffen (Unterkapitel 2.2), bevor bestehende Methoden zur kooperativen Trajektorienplanung speziell für Straßenfahrzeuge analysiert werden (Unterkapitel 2.3). Ziel ist es, auf Basis dieser Analyse eine Bewertung der recherchierten Verfahren durchzuführen, um das für die Weiterverfolgung am besten geeignete Verfahren zu bestimmen (Unterkapitel 3.1). Anschließend folgt eine Beschreibung der Kernelemente des darauf aufbauenden Algorithmus (Unterkapitel 3.2) und dessen Umsetzung in Matlab (Unterkapitel 3.3). Mit Kapitel 4 werden die Ergebnisse des entwickelten Verfahrens vorgestellt und diskutiert. Ausgewählte Fahrszenarien werden erläutert (Unterkapitel 4.1) sowie die Ergebnisse der Berechnungen dargelegt (Unterkapitel 4.2) und analysiert (Unterkapitel 4.3). Abschließend wird die Performance des implementierten Algorithmus untersucht (Unterkapitel 4.4) und mögliche Ansatzpunkte für dessen Weiterentwicklung aufgezeigt (Unterkapitel 4.5). Die Arbeit schließt in Kapitel 5 mit einer Zusammenfassung und einem Ausblick auf weitere Entwicklungen in diesem Themenbereich ab.

2 Stand der Technik – Kooperative Trajektorienplanung

Das zweite Kapitel dieser Arbeit beschäftigt sich mit dem Stand der Technik des Themenbereichs kooperatives Fahren. Zuerst wird die Thematik der kooperativen Trajektorienplanung vorgestellt und die Ziele sowie die damit verbundenen aktuellen Herausforderungen erläutert. Anschließend erfolgt ein kurzer Überblick über die Grundlagen der Trajektorienplanung in der Robotik und für automobile Anwendungen mit Einzelfahrzeugen. In Unterkapitel 2.3 werden verschiedene Verfahren zur Planung kooperativer Fahrmanöver vorgestellt und detailliert untersucht. Auf Grundlage dieser Untersuchung soll in Kapitel 3 eine Bewertung durchgeführt werden, um das für die Problemstellung am besten passende Verfahren zu bestimmen.

2.1 Kooperative Fahrmanöver – Definition, Ziele und Herausforderungen

Der Begriff „Kooperation“ beschreibt in der Psychologie eine Form der gesellschaftlichen Zusammenarbeit, bei der eine Gruppe von Akteuren durch bewusstes und planvolles Handeln versucht, die jeweiligen, i.d.R. unterschiedlichen Zielvorstellungen bestmöglich zu erfüllen. Entscheidend ist, dass eine Zielabstimmung und konstruktive Problemdiskussion zwischen den Akteuren erfolgt. Voraussetzung dafür ist eine Möglichkeit zum gegenseitigen Informationsaustausch und dem Vorhandensein öffentlich zugänglicher Regeln und Richtlinien [9]. Im Bereich der Robotik liegt kooperatives Verhalten nach Cao [10] vor, wenn ein System von mehreren Robotern so handelt, dass sich der Gesamtnutzen des Systems (engl.: total utility of the system) erhöht. Die Roboter führen dabei eine vom Entwickler festgelegte Aufgabe aus und halten sich an ein zugrundeliegendes Regelwerk. Die Erhöhung des Nutzens findet dabei gegenüber dem rein kollektiven Verhalten der Gruppe statt [10, S. 8]. Eine Kontroll- und Kommunikationsstruktur muss vorhanden sein, um kooperatives Verhalten zu ermöglichen.

Kooperatives Verhalten im Straßenverkehr ist allgegenwärtig und wird meist instinktiv angewendet. Vor allem Verkehrssituationen auf mehrspurigen Straßenabschnitten und an Fahrbahnkreuzungen ohne Lichtsignalanlage erfordern ein hohes Maß an Kooperation, damit der Verkehrsfluss aufrechterhalten werden kann. Ein Beispiel für nicht funktionierende Kooperation sind sogenannte „Phantomstaus“, die entstehen, wenn Fahrzeuge im dichten Verkehr Manöver ausführen, die andere Fahrzeuge zum Abbremsen zwingen. Alle hinterherfahrenden Fahrzeuge müssen immer stärker als das jeweils vorausfahrende Fahrzeug verzögern. Das passiert so lange, bis ein Fahrzeug bis zum Stillstand abbremsen muss und so den Verkehrsfluss ins Stocken bringt. Eines dieser unangepassten Manöver reicht aus, um Minuten später und an anderer Stelle zu einer Stauung aller Fahrspuren zu führen [11]. Es entsteht ein Stau, ohne dass ein auf den ersten Blick ersichtlicher Grund vorliegt.

Trotz der wachsenden Anzahl und stetigen Verbesserungen moderner Fahrerassistenzsysteme sind kooperative Aspekte bisher kaum berücksichtigt [7, S. 1151]. Das liegt vor allem daran, dass die Standards für eine herstellerübergreifende Fahrzeugkommunikation (V2V) und die dafür nötige Infrastruktur nicht vorliegen. Bei der Vernetzung von Fahrzeugen spricht man von V2X-Kommunikation, wobei das „V“ für Vehicle und das „X“ für den jeweiligen Kommunikationspartner steht. Häufig verwendet wird V2V (Vehicle-to-Vehicle Kommunikation) und V2I (Vehicle-to-Infrastructure Kommunikation) [5, S. 72]. Weil eine solche Infrastruktur derzeit kaum vorhanden ist, sind die bestimmenden Formen der Kommunikation im Straßenverkehr die Gestik und das Verhalten von Verkehrsteilnehmern. Signaleinrichtungen, die sich im und am Fahrzeug befinden (bspw. Fahrtrichtungsanzeiger, Bremsleuchten und Signalhorn) und die die Infrastruktur bereitstellt (bspw. Verkehrsschilder, Fahrbahnmarkierungen und Lichtsignalanlagen), spielen ebenfalls eine wichtige Rolle [7, S. 1157]. Ulbrich [12] kategorisiert die unterschiedlichen Ausprägungen des kooperativen Handelns nach „Kooperativen Fähigkeiten“ und nach „Kommunikationsart und Bewusstseinsgrad“. In Abb. 2.1 sind Beispiele für kooperatives Verhalten dargestellt, die typisch für die beiden Kategorien sind.

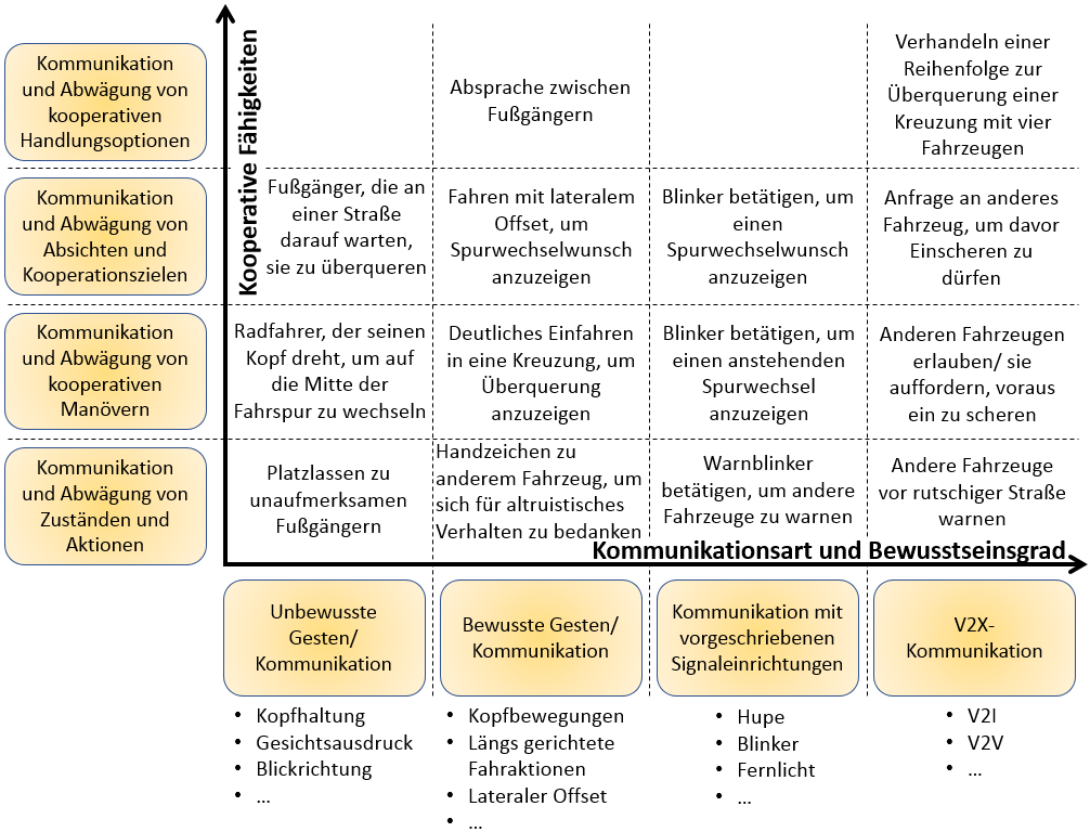


Abb. 2.1 Kategorisierung unterschiedlicher Ausprägungen des kooperativen Handelns nach kooperativen Fähigkeiten und Kommunikationsart und Bewusstseinsgrad nach [12, S. 2160]

Auf der vertikalen Achse sind die kooperativen Fähigkeiten aufgetragen, die die Verkehrsteilnehmer aufweisen. Diese reichen vom Erkennen bestimmter Verkehrssituationen und dem Ableiten konkreter Aktionen bis zum Erkennen und Abwägen von mehreren kooperativen Handlungsoptionen. Die horizontale Achse repräsentiert verschiedene Arten der Kommunikation und wie bewusst diese erfolgen. Unbewusste Gesten sind das niedrigste Level an Kommunikation, weil sie unbeabsichtigt stattfinden und wenig Informationsgehalt besitzen. V2X-Funktionen bilden hingegen das maximal erreichbare Level an Kommunikation ab, weil theoretisch alle Fahrzeuginformationen für andere bereitgestellt werden können. Bewegen sich alle Verkehrsteilnehmer komplett automatisiert fort (z.B. auf der Autobahn), können die

ersten drei Level der Kommunikation entfallen und der Informationsaustausch zwischen den Fahrzeugen muss vollständig über V2X-Kommunikation stattfinden. Ziel des kooperativen Fahrens im vollständig automatisierten Verkehr ist, die gesamte Kommunikation der Verkehrsteilnehmer und die Entscheidungsfindung von Fahrmanövern maschinell durchzuführen. Ein aktiver Informationsaustausch mit menschlichen Verkehrsteilnehmern soll nur noch stattfinden, wenn keine V2X-Kommunikation möglich ist (z.B. zwischen Fußgängern und Fahrzeugen). Erreicht werden sollen damit vor allem eine Erhöhung der Verkehrssicherheit und der Kapazität der Straßen. Beide Gesichtspunkte werden im Folgenden näher ausgeführt:

Ein wichtiger Aspekt zur Erhöhung der Sicherheit im Straßenverkehr ist die kollektive Perzeption [8]. Dadurch können Fahrzeuge ihre gesammelten Sensordaten anderen Verkehrsteilnehmern zur Verfügung stellen, die damit Zugriff auf für sie eigentlich nicht erfassbare Informationen bekommen. Als Folge ist es möglich, dass Fahrzeuge nicht mehr – wie bisher – individuell auf (Gefahren-)Situationen reagieren müssen, sondern präventiv und im Kollektiv handeln und ggf. ein Fahrmanöver ausführen können [8]. Speziell in kritischen Situationen mit mehreren beteiligten Fahrzeugen ist eine optimale Reaktion der Fahrzeugführer sehr anspruchsvoll. Ein entsprechendes Fahrmanöver wird durch mehrere Faktoren erschwert [13, S. 3]. Die menschliche Reaktionszeit und die fehlende Möglichkeit zur Kommunikation mit den anderen Beteiligten stellen das Hauptproblem dar. Außerdem sind Fahrmanöver im Voraus schwer auf ihre fahrdynamische Durchführbarkeit zu bewerten und Gefahrensituationen treten nur selten auf, wodurch die Routine des Fahrers fehlt. Durch eine vom System berechnete kooperative Handlung der Fahrzeuge kann solchen gefährlichen Situationen entsprechend begegnet werden. Kritische Situationen und Unfälle können komplett vermieden oder zumindest die Unfallschwere gemindert werden [13, S. 1-2].

Durch ein koordiniertes Fahrverhalten kann eine deutliche Effizienzsteigerung des Verkehrs erreicht werden. Bei rein autonomem Verkehr stellt Mauerer [5, S. 346] eine Kapazitätserhöhung von 40 % auf Fahrbahnen im Stadtverkehr und 80 % auf Autobahnabschnitten fest. Die Kapazität eines Straßenabschnitts wird in Fahrzeugen pro Stunde gemessen und gibt die maximale Anzahl an Fahrzeugen an, die diesen Abschnitt während einer Zeiteinheit passieren können [5, S. 339]. Eine weitere Kennzahl ist die Stabilität des Verkehrs, die bei rein autonomem Verkehr ebenfalls steigt. Je größer die Stabilität des Verkehrs ist, desto weniger schwankt die Kapazität des Straßenabschnitts und desto gleichmäßiger fließt der Verkehr [5, S. 343-344]. Es kommt folglich zu keinen Kapazitätseinbrüchen, wie sie in stockendem Verkehr auftreten. Eine Fahrzeuggruppe, die autonom fährt und vernetzt ist, hat gegenüber einer mit menschlichen Fahrern mehrere Vorteile. Durch den Entfall der Reaktionszeit kann das Beschleunigen und Verzögern des vorausfahrenden Fahrzeugs antizipiert werden. Das ermöglicht eine kleinere Zeitlücke und einen stets konstanten Abstand zwischen den Fahrzeugen, bei gleichbleibend komfortabler Fahrweise. Gleichzeitig kann die Fahrgeschwindigkeit, bei für Menschen viel zu geringem Abstand, erhöht werden, was zu einer Kapazitätserhöhung führt [5, S. 347-348]. Letztendlich können Fahrmanöver von menschlichen Fahrern, die kontraproduktiv für einen stabilen Verkehrsfluss sind, vollständig unterbunden werden. Ein Beispiel sind die bereits genannten Phantomstaus. Häufig werden sie dadurch ausgelöst, dass Fahrer spontan beschließen, auf die vermeintlich schnellere benachbarte Fahrspur zu wechseln [11]. Als Folge müssen Fahrzeuge auf dieser Fahrspur abbremsen, um dem einscherenden Fahrzeug Platz zu machen. Diese Bremsvorgänge bewirken stockenden oder stillstehenden Verkehr.

Dass vernetzte autonome Fahrzeuge den Verkehrsfluss stabilisieren und die Kapazität der Fahrbahn erhöhen, ist unbestritten. Was die Einführung von autonomen Fahrzeugen und

deren wachsender Anteil am Verkehrsgeschehen bedeuten, wird allerdings kontrovers diskutiert. Auf der einen Seite gibt es optimistische Einschätzungen, die die bereits genannten Vorteile der Unfallvermeidung und des positiven Einflusses auf den Verkehr nennen. Auf der anderen Seite gehen einige Experten davon aus, dass die Einführung autonomer Fahrzeuge höchstens eine geringfügige Verbesserung des Verkehrsproblems mit sich bringt. Im schlimmsten Fall verschlechtert sich die Verkehrssituation sogar [14]. Ein Grund dafür ist, dass sich eine höhere Anzahl an Fahrzeuge auf den Straßen befinden wird. Ein gewisser Anteil davon wird komplett ohne Passagiere fahren. Es geht dabei nicht um Fahrten, die eine notwendige Fahrt ersetzen (z.B. in eine Autowerkstatt). Vielmehr sind die Leerfahrten entscheidend, die aufgrund der Technologie erst ermöglich werden (z.B. Absetzen und Abholen der Passagiere mit zwischenzeitlicher Heimfahrt oder selbstständiger Parkplatzsuche) [15, S. 11]. Experten gehen auch von einer höheren Anzahl an Pendlern und einer gesteigerten Reisebereitschaft aus, weil der Innenraum komfortabler ausgestattet wird und das Reisen für andere Tätigkeiten genutzt werden kann [14]. Weiterhin erlaubt das autonome Fahren Menschen, die aufgrund ihres Alters keinen Zugang zu einem eigenen Fahrzeug haben, die Teilnahme am Straßenverkehr. Das sind bspw. Personen, die noch keine Fahrerlaubnis besitzen oder aufgrund ihres Alters nicht mehr fahrtüchtig sind [15, S. 11]. Als Folge steigt das Verkehrsaufkommen deutlich an. Kurz- und mittelfristig (bis ca. 2035) ist davon auszugehen, dass sich durch die Einführung immer leistungsfähigerer Fahrerassistenzsysteme und Fahrzeugen mit SAE-Level 3 und 4 die allgemeine Verkehrssituation verschlechtert [15, S. 10]. Das ist vor allem dem Mischverkehr aus konventionell gesteuerten und automatisierten Fahrzeugen geschuldet. Wie sich diese Technologie und ihre Marktdurchdringung auf den Straßenverkehr auswirkt, ist in Abb. 2.2 dargestellt.

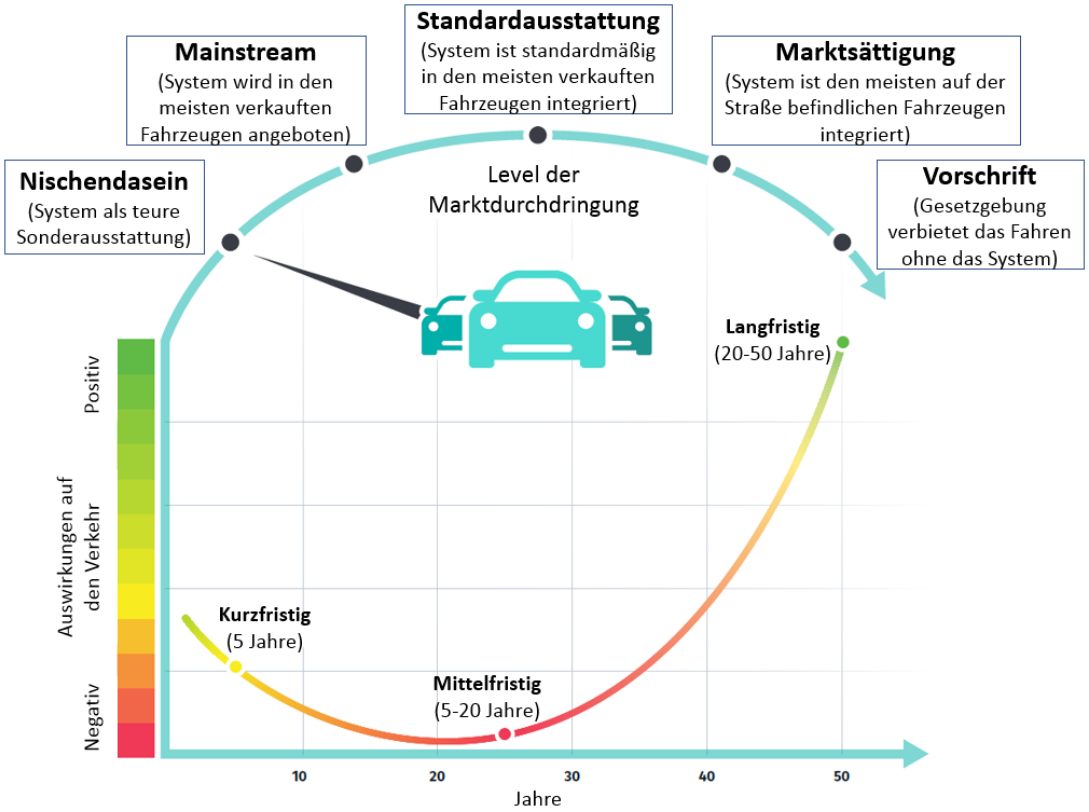


Abb. 2.2 Einfluss des autonomen Fahrens auf die Verkehrssituation in Abhängigkeit der Marktdurchdringung [15, S. 9]

Erst wenn die meisten auf der Straße befindlichen Fahrzeuge autonom und vernetzt fahren können, wird sich der positive Effekt dieser Technologie auf den Verkehr einstellen. Zuvor

überwiegen die negativen Auswirkungen der erhöhten Anzahl an Verkehrsteilnehmern und dem Mix aus konventionell gesteuerten und autonomen Fahrzeugen. Vor allem dadurch, dass sich beide Arten von Fahrzeugen im Verkehr befinden, entstehen große Herausforderungen.

Erstens kann das unterschiedliche Fahrverhalten von autonomen und konventionellen Fahrzeugen zu mehr Unfällen führen, weil Fahrer durch das nicht-menschliche Verhalten überrascht werden. Besonders die Tatsache, dass menschliche Fahrer die Verkehrsregeln situationsentsprechend häufig brechen, um den Verkehrsfluss aufrecht zu erhalten, gestaltet den Mischbetrieb schwierig. Ein Fahrzeug, das sich stets an die Verkehrsregeln hält, wird in vielen Situationen den Verkehr behindern [16]. Solange es keine wirkliche künstliche Intelligenz gibt, die jede Verkehrssituation verstehen kann, muss von den Entwicklern entschieden werden, wann und wie es zulässig ist, Regeln zu brechen. Angesichts der Unmenge an möglichen Situationen, kann dieses Vorgehen nie alle Szenarien abdecken [16].

Zweitens entfallen wichtige Kommunikationskanäle, wie sie in Abb. 2.1 zu finden sind. Eine Abstimmung zwischen Fahrer und autonomem Fahrzeug über die ersten drei Kommunikationsarten ist eingeschränkt oder nicht mehr möglich. Zum einen muss das autonome Fahrzeug seine Absichten den anderen Fahrzeugen ohne V2X-Kommunikation mitteilen können. Zum anderen muss das autonome Fahrzeug die Intentionen der menschlichen Fahrer detektieren und interpretieren können. Das gestaltet sich bei allen drei Kommunikationsarten als äußert komplex [12, S. 2161]. Bisher ist es technisch kaum umsetzbar, dass Bremslichter- oder Blinker-Signale von anderen Fahrzeugen zuverlässig detektiert werden. Die Erkennung von Gesten, Kopfbewegungen, Gesichtsausdrücken und Augenkontakt ist bisher nicht in brauchbarem Maße möglich [12, S. 2161, 17]. Zuletzt kann auch das Vertrauen der Passagiere in die Technik so groß werden, dass die Achtsamkeit nachlässt, was im gemischten Verkehr in mehr Unfällen resultiert [15, S. 10-11].

Für diese Übergangsphase und die Zeit des komplett autonomen Verkehrs müssen Lösungen entwickelt werden, damit die negativen Auswirkungen abgeschwächt werden und der Übergang zum vollständig autonomen Verkehr reibungslos verlaufen kann. Lösungsansätze sind unter anderem das gesetzliche Verbieten von menschlichen Fahrern oder das Einführen einer Steuer oder Maut. Ersteres kann vor allem im Berufsverkehr und auf stark ausgelasteten Streckenabschnitten stattfinden, um den kompletten Verkehr zentral koordinieren zu können [15, S. 12]. Letzteres dient dazu, der steigenden Anzahl an Fahrzeugen entgegenzuwirken und bspw. Leerfahrten einzuschränken. In Studien hat sich das Erheben von Maut oder Steuern als wirksamen Mittel zur positiven Beeinflussung des Verkehrsaufkommens und der Reisezeit herausgestellt [18, S. 42-43].

Letztendlich ist zur Umsetzung dieser Funktionen eine V2X-Kommunikation mit herstellerübergreifendem Standard erforderlich. Die Infrastruktur muss dafür ausgebaut werden und vor allem das 5G-Mobilfunknetz muss flächendeckend und zuverlässig verfügbar sein. Im Gegensatz zu den umstrittenen Auswirkungen, die autonome Fahrzeuge auf den Straßenverkehr haben werden, ist die Tatsache, dass dafür umfangreiche V2X-Kommunikation zwingend erforderlich ist, unbestritten. Für den zukünftigen Verkehr ist eine koordinierte Fahrweise mit aufeinander abgestimmten Trajektorien essentiell. Die Kommunikation, die heute der Fahrzeugführer übernimmt, muss vollständig ersetzt und maschinell dargestellt werden. Voraussetzung dafür ist, dass die Kommunikationsinfrastruktur ausgebaut wird.

2.2 Grundlagen der Trajektorienplanung

In diesem Unterkapitel werden die Grundlagen der Trajektorienplanung vorgestellt, um ein Verständnis für die typischen Fragestellungen zu schaffen. Speziell in der Robotik spielt die Planung von, im Sinne eines Gütemaßes, optimalen Trajektorien eine große Rolle. Mit der Entwicklung modernen Fahrerassistenzsysteme konzentriert sich die Forschung verstärkt auf Verfahren zur Trajektorienplanung von Einzelfahrzeugen. Arbeiten mit Schwerpunkt auf kooperative Trajektorienplanung von Straßenfahrzeugen sind bisher nur vereinzelt zu finden. Meist behandeln diese Arbeiten die Verkehrsflussoptimierung an Kreuzungen [13, S. 14].

Trajektorienplanung beinhaltet sowohl Bahnplanung als auch Bewegungsplanung. Geht es bei der Bahnplanung rein um das Finden eines zulässigen (i.d.R. kollisionsfreien) Weges, so berücksichtigt die Bewegungsplanung die dynamischen Restriktionen des sich bewegenden Objekts und den zeitlichen Verlauf der Bewegung [13, S. 18]. Grundsätzlich besteht ein Planungsproblem aus den immer gleichen Bestandteilen [19, S. 17-19]:

- **Zustandsraum:** In einem Planungsproblem beinhaltet der Zustandsraum alle möglichen Zustände, die auftreten können. Ein Zustand ist zum Bespiel die Position, die Ausrichtung und die Geschwindigkeit eines Fahrzeugs. Alle möglichen Konfigurationen, die alle beteiligten Fahrzeuge einnehmen können, werden durch den Zustandsraum abgebildet. Die Zustände können diskret oder kontinuierlich sein. In den meisten Fällen kann der Zustandsraum nicht explizit angegeben werden, weil die Anzahl der möglichen Zustände zu groß ist. Die Formulierung des Zustandsraums erfolgt dann implizit durch den Planungsalgorithmus.

- **Zeit:** Für ein Planungsproblem kann die Zeit explizit oder implizit modelliert werden. Explizit wird sie eingebunden, wenn es bspw. darum geht, das definierte Ziel schnellstmöglich zu erreichen. Interessiert die konkrete Zeitdauer nicht, wird die Zeit implizit berücksichtigt. Verwendet wird dies für Planungsprobleme, bei denen Aktionen von denen vorheriger Zeitschritte abhängen oder wenn die Aktionen nur hintereinander und nicht parallel ablaufen können.

- **Aktionen:** Durch Aktionen wird der Input in das System beschrieben, der zu einer Zustandsänderung führt. Im Vorherein muss festgelegt werden, wie sich eine bestimmte Aktion auf den Zustand des Systems auswirkt. Aktionen können diskret oder kontinuierlich definiert und deren Auswirkung auf den Zustand über gewichtete Funktionen oder gewöhnliche Differentialgleichungen bestimmt werden. Ein Beispiel für eine Aktion ist die Beschleunigung oder Verzögerung eines Fahrzeugs, die sich unmittelbar auf den Zustand Geschwindigkeit auswirkt.

- **Start- und Endzustand:** Der Start- und der Endzustand des Systems sind im Zustandsraum enthalten. Der Startzustand repräsentiert die Konfiguration, mit der das Planungsproblem beginnt. Für den Endzustand kann genau eine oder mehrere zulässige Konfigurationen definiert werden. Das Ziel ist, unter Anwendung der spezifizierten Aktionen vom Startzustand zum Endzustand zu gelangen.

- **Gütemaß:** Das Gütemaß legt fest, welche Aktionen und Zustände auf dem Weg zum Endzustand bevorzugt werden sollen. Grundsätzlich wird zwischen Umsetzbarkeit und Optimalität entschieden. Bei der Umsetzbarkeit wird nur gefordert, dass die Abfolge von Aktionen und Zuständen zulässig ist. Vor allem die

Kollisionsfreiheit der gefundenen Trajektorie steht hier im Fokus. Bei der Optimalität ist die Umsetzbarkeit eine Grundvoraussetzung. Zusätzlich werden Kostenfunktionen definiert, die die gewählten Aktionen und die daraus resultierenden Zustände bewerten. In diesem Fall ist die gefundene Trajektorie umsetzbar und optimal im Sinne des Gütemaßes. Häufig verwendete Gütemaße für Straßenfahrzeuge sind die Forderung nach möglichst geringer Beschleunigung und Verzögerung, geringer Abweichung von der Wunschgeschwindigkeit und Einhaltung des Sicherheitsabstands zu anderen Fahrzeugen [13, S. 88-90, 20, S. 46-48, 21, S. 450]. In kritischen Situationen kann der Fokus auf das Finden einer kollisionsfreien Trajektorie, unter Wegfall der Forderung nach Optimalität, gesetzt werden. Die Forderung nach Optimalität erhöht die Komplexität des Planungsproblems und damit den Rechenaufwand meist deutlich.

- **Planungsstrategie:** Durch die Planungsstrategie wird festgelegt, wie der Zustandsraum nach der Lösung durchsucht wird. Ein weiterer Bestandteil der Planungsstrategie ist, wie mit bekannten und noch unbekannten Zuständen umgegangen wird. Eng zusammen damit hängt die Formulierung des Gütemaßes, weil die Planungsstrategie unmittelbar auf die dadurch berechneten Werte zurückgreift.

Eines der einfacheren Planungsprobleme in der Robotik ist das Suchen des kürzesten Weges im zweidimensionalen Raum, in dem sich nur statische Hindernisse befinden [19, S. 27-29]. Zum einen existiert eine eindeutige Definition des Zielzustandes, zum anderen ist eine eindeutige Heuristik zur Bewertung des aktuellen Zustands verfügbar (minimaler euklidischer Abstand). Der Zustandsraum verändert sich, bis auf den Zustand des sich bewegenden Objekts, nicht über der Zeit. Häufig wird die Dynamik in Roboteranwendungen aufgrund der geringen Geschwindigkeiten und/oder der geringen Massen vernachlässigt. Diese Vereinfachung kann bei Straßenfahrzeugen nicht durchgeführt werden [13, S. 21]. Ein weiterer Unterschied zum Straßenfahrzeug ist, dass Roboter meist keine nicht-holonomen Einschränkungen besitzen (abhängig von der Bauart). Ein holonomes System liegt vor, wenn dessen Freiheitsgrade unabhängig voneinander oder nur durch eine bestimmte Anzahl an Zwangsbedingungen miteinander verbunden sind. Diese Zwangsbedingungen dürfen nur von den Freiheitsgraden selbst und nicht von deren Ableitungen abhängen. Auftretende Ableitungen müssen durch Integration eliminiert werden können, die Zwangsbedingung muss „integrierbar“ sein. Ist dies nicht der Fall, besitzt das System nicht-holonome Einschränkungen [19, S. 722-726, 22, S. 93-96]. Bei einem Straßenfahrzeug ist der Lenkwinkel an der Vorderachse eines Fahrzeugs beschränkt und die Hinterachse kann i.d.R. nicht lenken. Daraus resultieren Zwangsbedingungen für die Bewegung, die abhängig von der Geschwindigkeit sind. Bei der Betrachtung eines kleinen Zeitintervalls der Fahrzeugbewegung ist die Richtung derselbigen genau gleich der Ausrichtung der Räder an der Hinterachse [19, S. 722-724]. Eine Bewegung senkrecht dazu ist ausgeschlossen. Könnte jedes einzelne Rad um 90° nach links und rechts eingeschlagen werden und das Fahrzeug ließe sich aus dem Stand in jede beliebige Richtung bewegen, dann wäre dieses System holonom.

Vergleichbar mit dem beschriebenen Planungsproblem in der Robotik ist bei Straßenfahrzeugen das langsame Fahren in statischer, unstrukturierter Umgebung, zum Beispiel bei der Parkplatzsuche auf einer Parkfläche [20, S. 1]. Durch das langsame Fahren ist die Trägheit vernachlässigbar, weil unmittelbar zum Stillstand gekommen werden kann. Der Zielzustand ist klar definiert (Parken in Parklücke). Bei dieser Aufgabe handelt es sich

überwiegend um ein kombinatorisches Problem, bei dem die nicht-holonomen Eigenschaften des Fahrzeugs zum Tragen kommen.

Ein Zustandsraum mit bewegten Hindernissen erschwert das Finden einer Lösung nur geringfügig, solange die Bewegung der Hindernisse bekannt oder vorhersagbar ist [19, S. 311]. Der Zustandsraum wird um die Zeitachse erweitert und solange die Bewegung der Hindernisse bekannt ist, verhält sich dieses Planungsproblem nahezu wie ein statisches. Wichtig ist, dass die Zustandsraumvariablen dann Funktionen der Zeit sind [20, S. 10]. Wie sich der Zustandsraum bei der Erweiterung um eine Zeitachse verändert, ist in Abb. 2.3 zu erkennen. Die rote (jeweils links) und orange (jeweils rechts) Markierung repräsentieren zwei Hindernisse, die sich mit der Zeit bewegen. Der Parameter $\boldsymbol{x}(t)$ beschreibt die Trajektorie des Roboters.

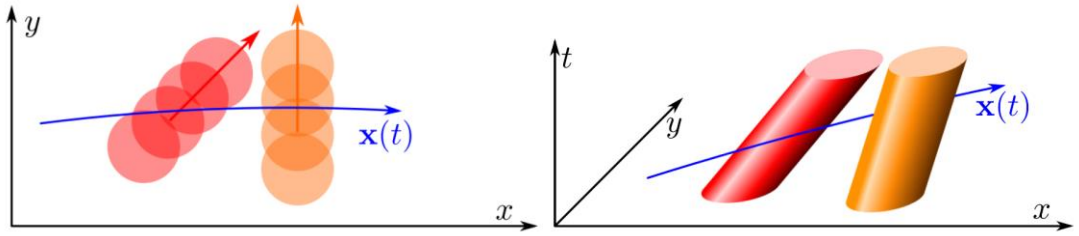


Abb. 2.3 Veränderung des ursprünglichen Zustandsraums (links) bei der Erweiterung um eine Zeitachse (rechts) [20, S. 10]

Für Straßenfahrzeuge zählt darunter das Fahren in fließendem Verkehr auf mehrspurigen Fahrbahnen. Die Umgebung ist dynamisch und strukturiert [20, S. 1]. Im Vergleich zur statischen Umgebung müssen bei höheren Geschwindigkeiten die Trägheit und die dynamischen Einschränkungen der Fahrzeuge beachtet werden. Aufgrund der starken Strukturierung durch Fahrspurmarkierungen und Fahrbahngeometrie ist der zulässige Fahrweg so stark eingeschränkt, dass kombinatorische Ansätze nicht zielführend sind [20, S. 1-2]. Es ist ausreichend, nur einen geringen Anteil aller theoretisch möglichen Aktionen zu berücksichtigen (geringer Lenkeinschlag und geringe Beschleunigung/Verzögerung), wenn das Ziel ist, die Wunschgeschwindigkeit oder -Fahrspur zu halten. Hinzu kommt, dass im fließenden Verkehr die Planungsrate im Vergleich zur Umgebung mit statischen Hindernissen deutlich erhöht werden muss. Ebenso müssen alle bewegten Objekte berücksichtigt und deren zukünftige Trajektorien abgeschätzt werden, um das in der Planung der eigenen Trajektorie zu berücksichtigen [20, S. 1-2]. In der Wahrnehmung eines solchen dynamischen Umfeldes konnten bereits große Fortschritte erzielt werden [7, S. 1146-1147]. Äußerst schwierig gestaltet sich jedoch das zuverlässige Prädizieren von Fahrzeugbewegungen über einen längeren Zeitraum, wenn die Absichten anderer Verkehrsteilnehmer geschätzt werden müssen [7, S. 703-704]. Speziell im gemischten Verkehr (menschliche Fahrer und autonom) verschärft sich diese Thematik, weil es kaum möglich ist, die Absichten eines menschlichen Fahrers zu erfassen (Unterkapitel 2.1). Sind die Bewegungen nicht mehr oder nur mit großer Unsicherheit vorhersagbar, so gestaltet sich die Lösung des Planungsproblems deutlich anspruchsvoller [19, S. 311]. Als Verfahren zur Trajektorienplanung von Einzelfahrzeugen wurden folgende entwickelt und angewendet:

- **RRT (rapidly-exploring random tree):** Dieses Verfahren exploriert den Zustandsraum zufällig. Gestartet werden kann im Start- oder Endzustand, aber auch in beiden Zuständen gleichzeitig. Von dort aus wird der Suchbaum inkrementell und zufällig erweitert, wobei zuerst eine grobe Abtastung des Zustandsraums erfolgt. Erst mit steigender Iterationszahl wird die Auflösung feiner. Der große Vorteil des Verfahrens ist, dass dynamische und kinematische Einschränkungen auch bei Problemstellungen mit einer hohen Anzahl an Freiheitsgraden berücksichtigt werden können. Das macht

das Verfahren vor allem für die Anwendung im Fahrzeug mit seinen nicht-holonomen Zwangsbedingungen interessant. In der Literatur werden RRTs unter anderem ausführlich von LaValle [19, S. 228-237], [23] behandelt. Es ist möglich, die Performance des RRT-Algorithmus zu verbessern, indem offline bereits Bewegungsprimitive berechnet und diese dann in der Suche berücksichtigt werden [19, S. 836].

- **Kombination von Bewegungsprimitiven:** Bewegungsprimitive werden im Voraus eines konkreten Planungsproblems offline berechnet. Das resultierende Set an möglichen Trajektorien-Stücken wird anschließend online kombiniert, um eine zulässige Trajektorie für das vorliegende Problem zu generieren. LaValle [19, S. 808-809], Heß [24] und Frazzoli [25] behandeln diese Thematik ausführlich. Angewendet werden Bewegungsprimitive unter anderem von Manzinger [26] und Schwarting [27].

  Vor allem beim Fahren in strukturierter Umgebung (bspw. Autobahn) ist die Anzahl an relevanten Manövern gering und daher das Arbeiten mit Bewegungsprimitiven vorteilhaft. Damit kann eine starke Verbesserung der Performance des Planungsalgorithmus erreicht werden. Problematisch ist die Planung von Trajektorien in kritischen Fahrsituationen, weil die notwendige feine zeitliche Auflösung nur mit hohem Aufwand realisiert werden kann [19, S. 809]. Eng mit der Anwendung von Bewegungsprimitiven verbunden ist das Parametrieren von im Voraus festgelegten Bahn- und Bewegungsverläufen.

- **Parametrierung von Bahn- und Bewegungsverläufen:** Dieses Verfahren kann unmittelbar nach der Kombination von Bewegungsprimitiven oder bei bereits vollständig vordefinierten Trajektorien angewendet werden. Die gespeicherten Trajektorien werden in Abhängigkeit der detektierten Verkehrssituation parametriert. Dieses Verfahren wird überwiegend für Spurwechselmanöver angewendet, bei dem meist ein Polynom fünften Grades parametriert wird. Vorteilhaft ist, dass das Problem stark vereinfacht wird. Allerdings kann dieses Verfahren nicht auf komplexere Fahrsituationen angewendet werden, bei dem eine deutlich größere Anzahl an möglichen Trajektorien berücksichtigt werden muss [13, S. 73]. Anwendungsbeispiele für die Parametrierung von Bahn- und Bewegungsverläufen finden sich bei Resende [28], Papadimitriou [29] und Hansen [30].

- **Elastische Bänder:** Das Konzept der elastischen Bänder wurde 1993 von Quinlan [31] vorgestellt. Das Verfahren ist dazu fähig, eine vorgegebene Trajektorie in Echtzeit auf ein sich veränderndes Umfeld anzupassen, um so auf bewegte Hindernisse zu reagieren. Allerdings muss dafür zuerst eine initiale Trajektorie berechnet und vorgegeben werden können. Ein großer Vorteil der Methode der elastischen Bänder ist, dass ihr Rechenaufwand im Gegensatz zu vielen anderen Bewegungsplanungsalgorithmen nicht exponentiell mit der Anzahl der zu berücksichtigenden Fahrzeuge wächst [32, S. 4]. Aus diesem Grund ist die Methode besonders für die kooperative Trajektorienplanung interessant und wird daher in Unterkapitel 2.3 detailliert erklärt.

- **Partial Motion Planning (PMP):** Der PMP-Algorithmus zeichnet sich durch Echtzeitfähigkeit aus. Außerdem soll durch ihn gewährleistet werden, dass keine Trajektorien vorgegeben werden, die zu einem Zustand führen, in dem eine Kollision nicht mehr vermeidbar ist. Es wird keine vollständige Trajektorie zum Ziel geplant, dafür erfolgt die Planung abschnittsweise so, dass unter allen Umständen eine Kollision verhindert wird oder das Ego-Fahrzeug zumindest bis zum Zeitpunkt einer Kollision zum Stillstand kommen kann. Literatur zu diesem Thema findet sich von Petti [33], Resende [28] und Bouraine [34].

Die Trajektorienplanung in Situationen mit mehreren Robotern stellt sich deutlich komplexer dar. Anstelle einer einzigen Trajektorie müssen mehrere gefunden werden, die nicht kollidieren dürfen und ggf. entsprechend eines Gütemaßes optimiert werden müssen. Die Dimension des Zustandsraums erhöht sich linear mit der Anzahl der beteiligten Roboter. Für einen vollständigen Planungsalgorithmus bedeutet das, dass die notwendige Rechenzeit exponentiell mit der Dimension des Zustandsraums ansteigt [19, S. 320].

Ein Vorteil von mobilen Robotern gegenüber Straßenfahrzeugen ist, dass häufig keine nichtholonomen Zwangsbedingungen und dynamische Einschränkungen berücksichtigt werden müssen. Das ermöglicht Verfahren wie die Entkopplung der Bahn- und Geschwindigkeitsplanung, wodurch sich die Komplexität des Problems erheblich verringern lässt [13, S. 22-23]. Dabei wird zuerst für jeden beteiligten Roboter eine optimale Bahn gesucht, die stationäre Hindernisse umgeht und die anderen Roboter vorerst ignoriert. Anschließend wird ein Geschwindigkeitsprofil für die jeweilige Bahn berechnet, sodass keine Kollisionen zwischen den Trajektorien der Roboter auftreten. Zu Nutze macht man sich dabei, dass aufgrund der geringen Trägheit instantan beschleunigt und verzögert werden kann. Für Straßenfahrzeuge trifft diese Annahme nicht zu [13, S. 23-24]. Häufig wird für Roboter auch die Entscheidung ohne Vorausplanung angewandt. Die Entscheidung, welche Aktion durchzuführen ist, wird instantan und nur anhand des aktuellen Zustands entschieden. Vorteilhaft sind der dadurch gering gehaltene Rechenaufwand und die gute Anpassungsfähigkeit an das veränderliche Umfeld. Negativ zu bewerten sind die mangelhafte Berücksichtigung der Dynamik und die Gefahr, keine verklemmungsfreien Bewegungen zu erhalten [13, S. 25]. Eine vorausschauende Handlungsweise kann dieses Verfahren nicht erzeugen.

Grundsätzlich wird bei der kooperativen Manöverplanung von Robotern häufig mit vereinfachenden Annahmen gearbeitet, die die Komplexität des Planungsproblems verringern und damit die Lösungsfindung erleichtern. Ein Großteil dieser Annahmen lässt sich nicht oder nur in geringem Maße auf die kooperative Trajektorienplanung für Straßenfahrzeuge übertragen, sodass sich damit gesondert auseinandergesetzt werden muss.

2.3 Analyse von Methoden zur kooperativen Trajektorienplanung

In diesem Unterkapitel werden die Funktionsweise und die Besonderheiten von Methoden zur Planung kooperativer Trajektorien erarbeitet, die sich speziell für die Anforderungen von Straßenfahrzeugen eigenen. Wie in dem vorherigen Unterkapitel gezeigt, fallen dabei viele Verfahren weg, die in der Robotik Anwendung finden. Begonnen wird mit der Formulierung des Problems, unabhängig von der konkreten Methode. Dabei wird die Komplexität des vorliegenden Planungsproblems aufgezeigt. Im Anschluss daran werden aussichtsreiche Verfahren zur kooperativen Trajektorienplanung von Straßenfahrzeugen detailliert untersucht und deren Vor- und Nachteile herausgearbeitet. Untersucht wird die prioritätsbasierte Planung, die Methode der gemischt-ganzzahligen linearen Programmierung, die Methode der elastischen Bänder und die Baumsuche. Die Baumsuche wird nochmals unterteilt in die Suchstrategien A* (A-Star), Branch-and-Bound und Monte-Carlo Baumsuche.

2.3.1 Allgemeine Problemformulierung und Komplexität des Planungsproblems

In diesem Abschnitt wird das Problem der kooperativen Trajektorienplanung allgemeingültig formuliert, um so die Basis für die detaillierte Untersuchung der verschiedenen Methoden zu schaffen. Dabei werden die Grundbausteine eines Planungsproblems aus Unterkapitel 2.2 nochmals aufgegriffen. Die im folgenden vorgestellte Notation orientiert sich überwiegend an Frese [13] und Lenz [21].

Grundsätzlich besteht ein Verkehrsszenario aus einer Anzahl von m Fahrzeugen p. Daraus ergibt sich eine Menge von Fahrzeugen $P = \{p_1, p_2, \dots, p_m\}$. Jedes Fahrzeug aus P kann unterschiedliche Eigenschaften haben. Beispiele sind der Fahrzeugtyp (Pkw, Lkw), die geometrischen Maße des Fahrzeugs (Länge, Breite), die maximale Geschwindigkeit sowie die maximale Beschleunigung und Verzögerung. Hinzu kommen fahrerabhängige Eigenschaften, wie die Wunschgeschwindigkeit und die komfortable Beschleunigung/Verzögerung, die ebenfalls berücksichtigt werden können. Der Zustandsvektor $\boldsymbol{x}_i$ des i-ten Fahrzeugs (mit $i = 1, \dots, m$) beschreibt dessen Zustand, der sich aus der x-Position x_i, der y-Position y_i, der Fahrzeugorientierung θ_i und der Fahrzeuggeschwindigkeit v_i zusammensetzt (Gl. (2.1)).

$$\boldsymbol{x}_i = (x_i, y_i, \theta_i, v_i)^T \tag{2.1}$$

Wie sich diese Größen am Fahrzeug wiederfinden, ist in Abb. 2.4 dargestellt. Die Räder in der Mitte der Vorder- und Hinterachse stellen das Einspurmodell dar, wie es von LaValle [19, S. 723] und Frese [13, S. 78] verwendet wird. Den Abstand zwischen der Vorder- und Hinterachse (Radstand) drückt $l_{Fzg,i}$ aus.

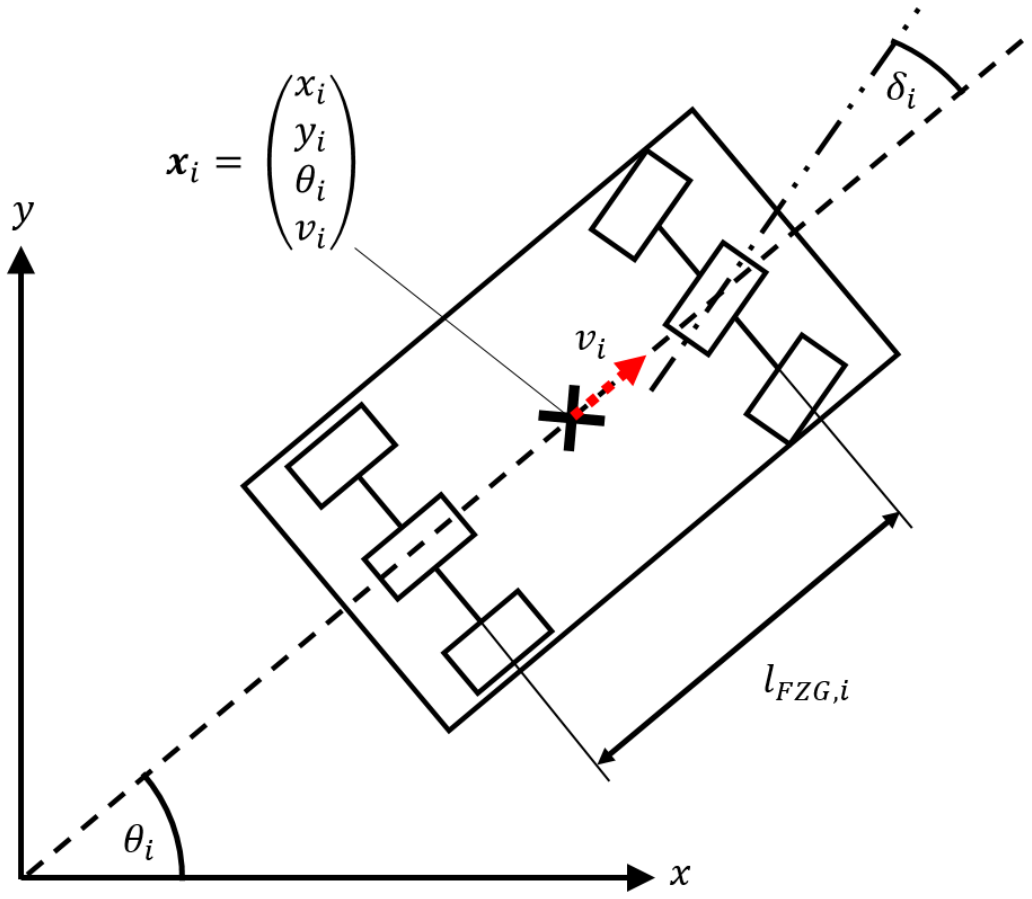


Abb. 2.4 Darstellung der relevanten Größen für den Fahrzeug-Zustandsvektor am Einspurmodell

Alle Größen beziehen sich auf den geometrischen Mittelpunkt des Fahrzeugs. Die x- und y-Position wird im globalen Koordinatensystem ermittelt. Der Winkel der Fahrzeugorientierung θ_i beschreibt die Verdrehung der Fahrzeuglängsachse zur globalen x-Achse. Die Geschwindigkeit des Fahrzeugs v_i wird im Mittelpunkt in Richtung der Fahrzeuglängsachse angegeben. Der Lenkwinkel des Einspurmodells wird durch δ_i beschrieben, findet aber im Zustandsvektor keine Verwendung. Den aktuellen Zustand des Systems repräsentiert $\boldsymbol{X}_{Veh}(t) = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_m)$, in dem alle Einzelzustände $\boldsymbol{x}_i(t)$ der m Fahrzeuge enthalten sind. Ist $\boldsymbol{X}$ der Zustandsraum des Planungsproblems, so gilt stets $\boldsymbol{X}_{Veh}(t) \in \boldsymbol{X}\ \forall\ t \in [t_{start}, t_{end}]$.

Neben der Position des Ego-Fahrzeugs müssen auch die Fahrbahn und die dazugehörigen Fahrspuren, Hindernisse und die anderen Fahrzeuge berücksichtigt werden. Angelehnt an Frese [13, S. 70] werden die geometrischen Maße der Fahrzeuge durch $\Omega_i^{veh}, i = 1, \dots, m$ ausgedrückt. Gleiches gilt für statische Hindernisse und deren geometrischer Repräsentation Ω^{obst} sowie für die Eigenschaften der Fahrbahn mit Ω^{road}. Diese sind unter anderem der Kurvenradius, die Steigung, die Anzahl und Breite der Fahrspuren sowie der Fahrspur-Typ (normale Fahrspuren, Beschleunigungs- und Verzögerungsspuren). Zusammen mit statischen Hindernissen, wie einer Fahrbahnverengung, kann daraus ein statisches Umfeld konstruiert werden. Dieses wird in der vorliegenden Arbeit als Fahrbahnlayout S_{FBL} bezeichnet und ist die Vereinigungsmenge von Ω^{road} und Ω^{obst} (Gl. (2.2)).

$$\Omega^{road} \cup\ \Omega^{obst} = S_{FBL} \tag{2.2}$$

Während die Fahrbahn und die statischen Hindernisse unabhängig von der Zeit sind, nehmen die Fahrzeuge zu jedem Zeitpunkt t eine andere räumliche Konfiguration ein. Die Fahrzeuge befinden sich zu Beginn der Manöverplanung zum Zeitpunkt t_0 im jeweiligen Startzustand $\boldsymbol{x}_i^{start}, i = 1, \dots, m$. Sie belegen den geometrischen Bereich $\Omega^{veh}(t = t_0)$ und besitzen eine initiale Geschwindigkeit und Orientierung. Werden die Fahrzeuge im Startzustand der statischen Umgebung hinzugefügt, so resultiert daraus das initiale Verkehrssituation S_{VS}^{start}, die das Verkehrsszenario für die Planung darstellt. Unterschieden wird zwischen Verkehrsszenario und Verkehrssituation. Das Verkehrsszenario wird durch die initiale Verkehrssituation beschrieben, für die die Planung der kooperativen Trajektorien stattfinden soll und die als Input für die Berechnung dient. Die Verkehrssituation selbst verändert sich nach jedem Zeitschritt aufgrund der sich verändernden Fahrzeugzustände. Die Verkehrssituationen beinhaltet die Zustände, die im Laufe der Berechnung auftreten und erst in Realität vorliegen, wenn der gefundene Plan ausgeführt wird. Allgemein ist die Verkehrssituation $S_{VS}(t)$ die Vereinigungsmenge des Fahrbahnlayouts S_{FBL} und der aktuell belegten Bereiche der Fahrzeuge $\Omega^{veh}(t)$ (Gl. (2.3)). Wird $\Omega^{veh}(t = t_0)$ anstatt $\Omega^{veh}(t)$ eingesetzt, ergibt sich das initiale Verkehrsszenario S_{VS}^{start}.

$$S_{FBL} \cup\ \Omega^{veh}(t) = S_{VS}(t) \tag{2.3}$$

Im Verkehrsszenario S_{VS}^{start} sind alle Bestandteile des zu lösenden Problems enthalten. Es beinhaltet die Umgebung inklusive der statischen Hindernisse und die bewegten Fahrzeuge. Die individuellen Eigenschaften jedes Fahrzeugs sind in P hinterlegt. Die Stellgrößen eines Fahrzeugs sind die Beschleunigung (positiv oder negativ) und der Lenkwinkel. Diese Stellgrößen können in ihrer Höhe verändert und miteinander kombiniert werden, wodurch ein Set an k Aktionen $\boldsymbol{a}_{act,i}(S_{VS}(t)) = \left(a_{act,i1}, a_{act,i2}, \dots, a_{act,ik}\right)^T$ für jedes Fahrzeug i definiert werden kann. Eine Handlung/Aktion $a_{act,ij}$ (mit $j = 1, \dots, k$) besteht immer aus einem Wertepaar von Beschleunigung und Lenkwinkel (a_{ij}, δ_{ij}). Das Set an Aktionen für jedes Fahrzeug $\boldsymbol{a}_{act,i}(S_{VS}(t))$ ist abhängig von der aktuellen Verkehrssituation, weil diese manche

Handlungen verbietet. Aus diesem Grund ist die Anzahl an Aktionen k nicht für jedes Fahrzeug identisch. Die Menge der Aktionen aller Fahrzeuge zum Zeitpunkt t ist die Aktionsmatrix $\boldsymbol{A}_{Veh}(S_{VS}(t)) = (\boldsymbol{a}_{act,1}, \boldsymbol{a}_{act,2}, \dots, \boldsymbol{a}_{act,m})$ und ebenfalls abhängig von der aktuellen Verkehrssituation. Befindet sich das Fahrzeug bspw. auf der linken Fahrspur, dann ist eine Aktion, die das Fahrzeug nach links lenken würde, nicht erlaubt, weil das zu einem Verlassen der Fahrbahn führen würde. Findet keine Diskretisierung der Stellgrößen statt und werden sie kontinuierlich abgebildet, so gibt es unendlich viele Aktionen. Mit einer Aktion $a_{act,ij}$ aus dem Set an Aktionen $\boldsymbol{a}_{act,i}$ kann der Zustand eines Fahrzeugs $\boldsymbol{x}_i(t)$ in einen neuen Zustand des Zeitpunkts $\boldsymbol{x}_i(t + \Delta t)$ überführt werden. Das geschieht mithilfe einer Zustandsübergangsfunktion f_i , wie aus Gl. (2.4) hervorgeht [13, S. 82, 19, S. 29].

$$\boldsymbol{x}_i(t + \Delta t) = f_i(\boldsymbol{x}_i(t), a_{act,ij}, \Delta t) \tag{2.4}$$

Für die Zustandsübergangsfunktion wird der Zeitschritt Δt , die jeweilige Position und Geschwindigkeit des Zeitpunkts t aus dem Zustandsvektor $\boldsymbol{x}(t)$ und die Aktion $a_{act,ij}$ benötigt. Wird der Geschwindigkeits- und Beschleunigungsvektor in seine x- und y-Koordinaten aufgespalten, lässt sich die Geschwindigkeit $\boldsymbol{v}(t + \Delta t)$ des nächsten Zeitschritts $t + \Delta t$ mit Gl. (2.5) berechnen [13, S. 78]. Der Einfluss der Orientierungsänderung wird hier aus Anschaulichkeitsgründen nicht berücksichtigt.

$$\boldsymbol{v}(t + \Delta t) = \begin{pmatrix} v_x(t + \Delta t) \\ v_y(t + \Delta t) \end{pmatrix} = \begin{pmatrix} v_x(t) \\ v_y(t) \end{pmatrix} + \begin{pmatrix} a_x(t) \\ a_y(t) \end{pmatrix} \Delta t \tag{2.5}$$

Die Fahrzeugposition $\boldsymbol{x}_{pos}(t + \Delta t)$ für den nächsten Zeitschritt lässt sich mithilfe von Gl. (2.6) berechnen [13, S. 78].

$$\boldsymbol{x}_{pos}(t + \Delta t) = \begin{pmatrix} x(t + \Delta t) \\ y(t + \Delta t) \end{pmatrix} = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} + \begin{pmatrix} v_x(t) \\ v_y(t) \end{pmatrix} \Delta t + \frac{1}{2} \begin{pmatrix} a_x(t) \\ a_y(t) \end{pmatrix} (\Delta t)^2 \tag{2.6}$$

Für die Fahrzeugorientierung $\theta(t + \Delta t)$ lässt sich der Zusammenhang aus Gl. (2.7) angeben [13, S. 78]. Verwendete Größen sind, wie in Abb. 2.4 notiert, der Lenkwinkel δ, die Geschwindigkeit v, die Beschleunigung a und die Orientierung des Fahrzeugs θ jeweils zum aktuellen Zeitschritt t. Der Radstand ist l_{Fzg}. Die Eingangsgrößen, die eine Veränderung des Zustands bewirken, sind die Beschleunigung a und der Lenkwinkel δ.

$$\theta(t + \Delta t) = \theta(t) + \frac{\tan \delta(t)}{l_{Fzg}} \left(v(t) + \frac{1}{2}\, a(t)\, \Delta t\right) \Delta t \tag{2.7}$$

Die Stellgrößen werden durch kinematische und dynamische Einschränkungen festgelegt. Für den Lenkwinkel gilt $\delta = [-\delta_{max}, \delta_{max}]$, was durch die kinematische Auslegung der Vorderachsaufhängung bestimmt wird [13, S. 78]. Die maximalen Werte für die Längs- und Querbeschleunigung ergeben sich aus der Betrachtung des Kamm'schen Kreises [4, S. 705-706]. Abhängig vom aktuellen Fahrzustand eines Fahrzeuges ergeben sich daraus maximal erreichbare Kraftschlusswerte für Aktionen, bei denen gleichzeitig Längs- und Querkräfte auftreten.

Die Schwierigkeit bei der Planung kooperativer Fahrmanöver liegt darin, dass die Dimension des Zustandsraumes linear mit der Anzahl der beteiligten Fahrzeuge anwächst. Das bedeutet für vollständige Algorithmen wiederum einen exponentiellen Anstieg der Rechenzeit [19, S. 320]. Eine Vorstellung über das Ausmaß des Planungsproblems gibt auch folgendes Beispiel: Sei die Anzahl der beteiligten Fahrzeuge m = 3, die Anzahl der Aktionen pro Fahrzeug k = 8 und der Planungshorizont, bis zu dem die Berechnung durchgeführt werden soll,

t_{end} = 10 s mit Δt = 1 s. Mithilfe von Gl. (2.8) lässt sich die Anzahl der möglichen Kombinationen berechnen, wenn stets alle acht Aktionen ausgewählt werden können [13, S. 92].

$$Anzahl\ der\ möglichen\ Kombinationen = k^{m\,\frac{t_{end}}{\Delta t}} \tag{2.8}$$

Für das genannte Beispiel existieren $1{,}24 \cdot 10^{27}$ mögliche Kombinationen für ein kooperatives Fahrmanöver. Es wird deutlich, dass es spezieller Verfahren bedarf, um dieses komplexe Problem lösen zu können.

2.3.2 Prioritätsbasierte Planung

Neben der Entkopplung von Bahn- und Geschwindigkeitsplanung stellt die prioritätsbasierte Planung eine weitere Variante der entkoppelten Bewegungsplanung für Probleme mit mehreren Agenten dar. Bei diesem Verfahren wird den beteiligten Robotern eine Priorität zugewiesen. Gestartet wird mit der Trajektorienplanung für den höchstpriorisierten Roboter. Alle beteiligten Roboter, die sich in der Priorität darunter befinden, werden komplett ignoriert. Anschließend erfolgt die Trajektorienplanung für den nächsten, niedriger priorisierten. Dabei wird der höher priorisierte Roboter als Hindernis, das sich entlang der bereits berechneten Trajektorie bewegt, berücksichtigt. Alle anderen werden, wie beschrieben, für die Planung ignoriert. Auf diese Weise wird bis zum letzten Roboter fortgefahren, der sich folglich nur in einer Umgebung mit bewegten Hindernissen befindet. Das Verfahren wird ausführlich von LaValle [19, S. 322-323] beschrieben. Für die Planung der Trajektorien können die Verfahren angewandt werden, die bereits für den Einsatz bei Einzelfahrzeugen im Umfeld mit bewegten Hindernissen erläutert wurden (Unterkapitel 2.2).

Im Vergleich zur Entkopplung von Bahn- und Geschwindigkeitsplanung kann bei der prioritätsbasierten Planung die Dynamik der Fahrzeuge berücksichtigt werden [13, S. 24]. Eine Eigenschaft der Entkopplungsmaßnahmen ist, dass dadurch die Rechenzeit der Planungsalgorithmen stark reduziert werden kann [13, S. 73]. Nachteilig bei dem Konzept der Entkopplung ist die Tatsache, dass die Vollständigkeit der Suchalgorithmen verloren geht und damit die Anzahl an verfügbaren Fahrmanövern eingeschränkt wird. Ein anschauliches Beispiel bietet Abb. 2.5.

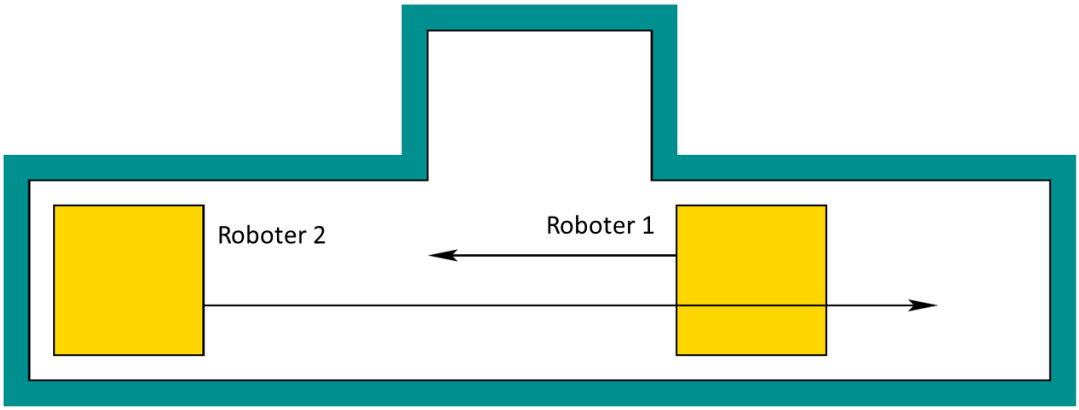


Abb. 2.5 Prioritätsbasierte Planung: Roboter 1 ignoriert bei der Trajektorienplanung Roboter 2; das Problem kann nicht gelöst werden (nach LaValle [19, S. 323])

Das Ziel der beiden Roboter ist, aneinander vorbeizufahren. Roboter 1 ignoriert bei der Planung der eigenen Trajektorie Roboter 2 und wählt den kürzest möglichen Weg zur gegenüberliegenden Seite. Roboter 2 sieht anschließend Roboter 1 als bewegtes Hindernis, auf das kein Einfluss genommen werden kann. Roboter 2 kann nicht ausweichen und es

kommt zur Kollision. Das Problem kann mit der prioritätsbasierten Planung nicht gelöst werden, obwohl eine Lösung des Problems möglich ist. Einer der Roboter müsste lediglich in die Aussparung am oberen Rand fahren und warten, bis der andere Roboter den Durchgang nicht mehr blockiert.

In Abb. 2.6 ist ein weiteres Beispiel von Frese [13, S. 23] abgebildet, wie Entkopplungsmaßnahmen die Lösung einschränken oder keine zulässige hervorbringen. Bild (a) zeigt das zugrundeliegende Verkehrsszenario. Es handelt sich dabei um zwei aufeinander zufahrende Fahrzeuge auf einer einspurigen Fahrbahn. Bei der Entkopplung der Bahn- und Geschwindigkeitsplanung (b) fahren die Fahrzeuge geradewegs aufeinander zu. Nur die Geschwindigkeit wird durch Verzögerung angepasst, mit dem Ziel, eine Kollision zu vermeiden. In diesem Szenario ist eine rechtzeitige Abbremsung nicht möglich, weshalb es zu einer Kollision kommt. Die prioritätsbasierte Planung (c) führt dazu, dass das höher priorisierte Fahrzeug mit konstanter Geschwindigkeit dem Straßenverlauf folgt und dabei das andere Fahrzeug komplett ignoriert. Das niedrig priorisierte Fahrzeug sieht das andere als bewegtes Hindernis auf sich zu kommen und führt ein Ausweichmanöver durch, das sogar zum Verlassen der Straße führt. Es wird deutlich, dass die Entkopplungsmaßnahmen für die kooperative Manöverplanung nicht zielführend eingesetzt werden können. Ein kooperatives Fahrmanöver (d) ist in der Lage, eine Kollision und ein Verlassen der Straße komplett zu vermeiden. Die Fahrzeuge weichen einander erfolgreich aus.

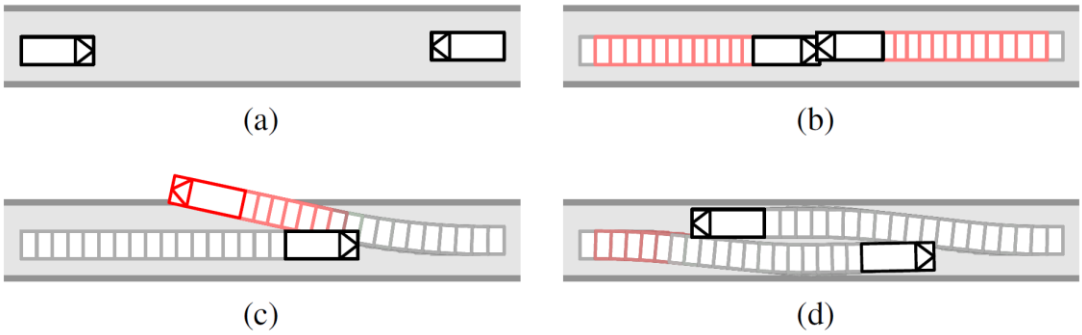


Abb. 2.6 Vergleich verschiedener Ansätze zur Bewegungsplanung in einem Gegenverkehrsszenario [13, S. 23]: (a) Anfangszustand, (b) Entkopplung von Bahn- und Geschwindigkeitsplanung, (c) prioritätsbasierte Planung, (d) kooperative Bewegungsplanung

Schwierig gestaltet sich bei der prioritätsbasierten Planung die Zuteilung der Prioritäten, wie Fall (c) zeigt. Obwohl Heuristiken und Optimierungsverfahren entwickelt wurden, ist die optimale Zuteilung der Prioritäten stark situationsabhängig. Beim autonomen Fahren kommen zusätzlich rechtliche Aspekte ins Spiel, vor allem wenn es darum geht, bei einem Fahrmanöver zur Unfallvermeidung Fahrzeugen eine Priorität zuzuteilen.

2.3.3 Gemischt-ganzzahlige lineare Programmierung

Das gemischt-ganzzahlige lineare Programmieren (engl.: mixed-integer linear programming, MILP) ist ein Verfahren zur Optimierung einer Zielfunktion unter Nebenbedingungen. Es wurde 1996 von Sierksma [35] vorgestellt. Gemischt-ganzzahlig bedeutet, dass alle oder zumindest ein Teil der Freiheitsgrade des zu optimierenden Systems ganzzahlig oder diskret sein müssen [36, S. 1-2]. Die Freiheitsgrade werden auch Optimierungsvariablen genannt. Die Zielfunktion und deren Nebenbedingungen müssen lineare Optimierungsvariablen besitzen.

Klassische Anwendungsfälle des MILP-Verfahrens zur Optimierung finden sich in den Bereichen Transport und Logistik, Finanzen, Kommunikation und Produktionsplanung. Das bekannteste Beispiel ist das Problem des Handlungsreisenden (engl.: traveling salesman problem), das mithilfe dieses Verfahrens gelöst werden kann [36, S. 2]. Das gemischt-ganzzahlige lineare Programmieren wurde bisher in einigen Arbeiten für die kooperative Bewegungsplanung von Flugzeugen angewandt [37]. Es finden sich auch Arbeiten, die das Verfahren für die Koordination von Schiffen [38] oder von autonomen Fahrzeugen an einer Kreuzung ohne Lichtsignalanlage [39] anpassen. Es werden allerdings nicht die Trajektorien der Fahrzeuge berechnet, sondern nur das Timing der Kreuzungsquerung. Speziell auf die Berechnung kooperativer Trajektorien von Straßenfahrzeugen wird das MILP-Verfahren von Frese [13, S. 106-118] zugeschnitten. Optimierungsvariablen sind Freiheitsgrade des Systems, die optimiert werden sollen. Die Werte, die sie annehmen können, werden durch Nebenbedingungen beschränkt. Generell wird zwischen mehreren Arten von Variablen unterschieden [36, S. 10-12]:

- **Kontinuierliche Variablen** werden durch den Bereich $0 \leq X^- \leq x \leq X^+$ definiert. Sie sind nicht-negativ und reell. In den meisten Fällen gilt $X^- = 0$. Die obere Grenze X^+ kann beliebige Werte bis unendlich annehmen.
- **Freie oder unbeschränkte Variablen** sind kontinuierliche Variablen, die beliebige positive und negative Werte annehmen können.
- **Ganzzahlige Variablen** sind Werte der Menge der ganzen Zahlen $\mathbb{Z}$. Sie können positive und negative Werte im Bereich $x = \{\dots, -2, -1, 0, 1, 2, \dots\}$ annehmen.
- **Binäre Variablen** können nur die Werte 0 und 1 annehmen und sind damit ein Sonderfall der ganzzahligen Variablen. Mit ihnen können logische Zustände, wie einfache Ja-/Nein-Entscheidungen oder Ein-/Aus-Zustände, beschrieben werden.
- **Diskrete Variablen** können nur diskrete Werte annehmen, die zuvor in einer Wertemenge festgelegt werden. Die Variable x kann bspw. nur folgende Werte annehmen: $x = \{-5.2, -2, 0, 1.3, 10\}$. Die konkreten Werte der definierten Wertemenge müssen weder ganzzahlig noch binär sein.

Halbstetige bzw. semi-kontinuierliche und partiell-ganzzahlige Variablen und strukturierte Variablenmengen werden nicht aufgeführt, weil sie in dem Entwurf von Frese [13] keine Rolle spielen. Detailliert werden sie von Kallrath [36, S. 11-12] beschrieben.

Nebenbedingungen oder Randbedingungen beschreiben den zulässigen Bereich, den die Optimierungsvariablen einnehmen dürfen. Die Begrenzung dieses Bereichs wird mit Gleichungen und Ungleichungen formuliert. Die Ungleichungen mit $<$ und $>$ müssen dabei unter Zuhilfenahme von zusätzlichen Parametern auf die Relationen $\leq$ und $\geq$ zurückgeführt werden [36, S. 13]. Bei der linearen Optimierung dürfen die Nebenbedingungen auf beiden Seiten der Gleichungen und Ungleichungen nur lineare Ausdrücke enthalten. Die Variablen dürfen nur mit konstanten Koeffizienten multipliziert werden. Produkte, Potenzen, e-Funktionen und Winkelfunktionen sind Bestandteil der nichtlinearen Optimierung [36, S. 13]. Die Zielfunktion ist der entscheidende Bestandteil eines Optimierungsproblems [36, S. 13]. Sie bestimmt, wie sich die Freiheitsgrade des Systems verändern müssen, damit ein optimales Ergebnis erreicht wird. Die Zielfunktion hat die Aufgabe, die Abhängigkeiten der Optimierungsvariablen des Systems untereinander auszudrücken [36, S. 1]. Die Variablen werden vom Algorithmus derart berechnet, dass die Zielfunktion einen optimalen Wert annimmt, welcher entweder das Minimum oder das Maximum der Zielfunktion darstellt.

Im Folgenden wird das von Frese [13] entwickelte Konzept zur Anpassung des MILP-Verfahrens auf die kooperative Trajektorienplanung von Straßenfahrzeugen vorgestellt. Frese setzt den Schwerpunkt auf das Berechnen von Trajektorien zur unmittelbaren Unfallvermeidung und nicht zur Kooperation im normal fließenden Verkehrsgeschehen [13, S. 2].

Die Optimierungsvariablen bei der kooperativen Trajektorienplanung sind die Position, die Geschwindigkeit und die Beschleunigung der Fahrzeuge zu diskreten Zeitpunkten. Die Position wird in einem globalen x-y-Koordinatensystem bestimmt. Gleiches gilt für die Geschwindigkeit und die Beschleunigung, die im selben Koordinatensystem vektoriell angegeben werden. Die Orientierung des Fahrzeugs ist zum Startzeitpunkt bekannt und wird nur implizit durch die Abfolge der Geschwindigkeits- und Beschleunigungsvektoren mitgeführt [13, S. 107]. Die Beschleunigungswerte der Fahrzeuge müssen nicht diskretisiert werden. Sie stellen zusammen mit der Geschwindigkeit und der Position die kontinuierlichen Optimierungsvariablen des Systems dar. Da das Einspurmodell, wie es in Abb. 2.4 abgebildet ist, Nichtlinearitäten aufgrund von Winkelfunktionen enthält, muss die Beschreibung der Fahrzeugbewegung linearisiert werden. Das Zustandsraummodell kann in linearen Gleichungen überführt werden. Benötigt wird dazu der Zeitschritt Δt, die jeweilige Position und Geschwindigkeit des vorherigen Zeitpunkts $t - \Delta t$ und die aktuelle Beschleunigung $\boldsymbol{a}(t)$. Die Position $\boldsymbol{x}_{pos}(t)$ und die Geschwindigkeit $\boldsymbol{v}(t)$ des aktuellen Zeitschritts lässt sich ähnlich zu Gl. (2.5) und Gl. (2.6) mit einer diskretisierten Zustandsübergangsfunktion berechnen [13, S. 107].

Die Geschwindigkeit und die Beschleunigung werden durch Nebenbedingungen beschränkt. Für die Geschwindigkeit kann ein zulässiger Bereich angegeben werden, der im Intervall $[0, v_{max}]$ liegt, wobei v_{max} die Höchstgeschwindigkeit beschreibt und Rückwärtsfahren ausgeschlossen ist [13, S. 108]. Die Nebenbedingungen für die Beschleunigung sind deutlich aufwändiger zu definieren, weil das Kraftschlusspotential der Reifen berücksichtigt werden muss. Frese führt dazu eine lineare Approximation des Kamm'schen Kreises durch, bezogen auf die Anfangsorientierung des jeweiligen Fahrzeugs θ_i. Diese muss über die gesamte Fahrzeugbewegung verwendet werden, weil die Fahrzeugorientierung während der geplanten Bewegung im linearen Fahrzeugmodell nicht berechnet werden kann. Nach einer Orientierungsänderung, bspw. durch eine Lenkbewegung, können die fahrdynamischen Begrenzungen nur noch näherungsweise abgebildet werden [13, S. 107-108].

Durch die Zielfunktion werden vordefinierte Variablen des Systems optimiert, zum Beispiel die Beschleunigung oder die Abweichung von der Wunschgeschwindigkeit. Frese definiert die Zielfunktion als Betrag der Beschleunigungen aller Fahrzeuge, den es zu minimieren gilt [13, S. 108-109]. Alle anderen Anforderungen, wie Kollisionsfreiheit oder Nicht-Verlassen der Straße, werden über harte Nebenbedingungen durchgesetzt und müssen deswegen nicht in der Zielfunktion berücksichtigt werden [13, S. 108].

Für die Vermeidung von Kollisionen mit anderen Fahrzeugen und statischen Hindernissen wird der Grundriss der Fahrzeuge überapproximiert. Zusätzlich können Sicherheitsabstände mit einbezogen werden. Diese Grundrissflächen werden mit Ungleichungen auf eine Kollision mit einer anderen Fläche zum jeweils gleichen Zeitpunkt überprüft. Mithilfe binärer Variablen werden diese Nebenbedingungen formuliert [13, S. 109-110]. Im Vergleich zur Anwendung bei der Koordination von Flugzeugen erfordert das Fahren im Straßenverkehr deutlich kleinere Sicherheitsbereiche. Die Größe der Sicherheitsbereiche längs zur Fahrbahnrichtung (Sicherheitsabstand zum vorausfahrenden Fahrzeug) und quer zur Fahrbahnrichtung (Sicherheitsabstand zum Fahrzeug auf der benachbarten Fahrspur) unterscheiden sich zudem stark voneinander. Für die Anwendung im Straßenverkehr muss das Geometriemodell

genauer ausgearbeitet werden, weil sonst bspw. ein Überholmanöver auf der benachbarten Fahrspur nicht möglich wäre [13, S. 110-111]. Zur Lösung dieses Problems schlägt Frese vor, die Position des vorherigen Zeitschritts zu verwenden und die des nächsten abzuschätzen, um daraus einen Bereich zu konstruieren, in den andere Fahrzeuge im aktuellen Zeitschritt nicht eindringen dürfen [13, S. 111-112]. Ein weiterer Ansatz ist, die Fahrzeuggeometrie durch ein Rechteck mit der Orientierung des Fahrzeugs zum Startzeitpunkt abzubilden. Beide Ansätze haben gemein, dass sie eine Orientierungsänderung des Fahrzeugs während der Manöverplanung nicht berücksichtigen, weil dies nicht mit linearen Funktionen ausgedrückt werden kann. Beide Ansätze gehen davon aus, dass sich die Orientierung während des Fahrmanövers nur geringfügig ändert [13, S. 113].

Die Modellierung der Straßengeometrie erfolgt auf ähnliche Weise [13, S. 113-114]. Die Straße wird mit Rechtecken approximiert, die je nach Kurvenradius immer feiner zerlegt werden. Es wird, ähnlich der Kollisionsüberprüfung, mit Ungleichungen gearbeitet, um festzustellen, ob sich die Fahrzeuge auf der Fahrbahn befinden. Dafür werden binäre Variablen verwendet. Befindet sich der Mittelpunkt des Fahrzeugs in einem der Rechtecke, so wird die Überprüfung aller anderen Rechtecke ausgesetzt. Wie auch bei den Formulierungen zuvor ist durch die lineare Darstellung des Zustandsraummodells keine Berechnung der aktuellen Fahrzeugorientierung möglich. Abschließend formuliert Frese [13, S. 115-118] obere Schranken für die Fahrzeugposition, um den Rechenaufwand des MILP-Verfahrens zu verringern. Mit den maximal und minimal möglichen Längs- und Querbeschleunigungen werden Bereiche bestimmt, die die Fahrzeuge im aktuellen Zeitschritt im Extremfall erreichen können. Alle anderen Fahrzeuge, deren ebenfalls auf diese Weise berechneter Bereich außerhalb liegt, werden bei einer Kollisionsabfrage nicht berücksichtigt. Eine Kollision zu diesem Zeitpunkt kann ausgeschlossen werden, weil die maximalen Beschleunigungs- und Verzögerungswerte nicht überschritten werden können.

Es existieren viele verschiedene Ansätze zur Lösung von linearen gemischt-ganzzahligen Optimierungsproblemen. Eine Unterteilung lässt sich vornehmen in Verfahren mit Heuristiken und in sogenannte „exakte“ Methoden. Nur letztere sind in der Lage, eine optimale Lösung zu finden. Einen detaillierten Überblick über diese exakten Methoden und deren Eigenschaften gibt Kallrath [36, S. 83-105].

Zu den Vorteilen des gemischt-ganzzahligen linearen Programmierens zählt, dass die Optimierungsvariablen der Fahrzeuge nicht diskretisiert werden müssen. Das ermöglicht eine kontinuierliche Bestimmung der Beschleunigungswerte, die die Stellgrößen für das Fahrzeug darstellen. Mit den kontinuierlichen Werten ergeben sich keine Einschränkungen der Lösung durch eine zu grobe Diskretisierung. Durch die großen Fortschritte in der Entwicklung neuer Algorithmen und der damit verbundenen Performancesteigerung haben MILP-Verfahren in den letzten Jahren eine erhebliche Verbreitung gefunden [36, S. 2].

Nachteilig ist, dass durch die Linearisierung des Zustandsraummodells wichtige Informationen verloren gehen. Vor allem über einen längeren Simulationszeitraum ist das Fehlen der Fahrzeugorientierung problematisch. Hinzu kommt, dass die Kollisionsprüfung nur nach festen Zeitschritten Δt stattfindet und dort durch die definierten Nebenbedingungen durchgesetzt wird. Diese Abtastintervalle müssen dementsprechend eng sein und an die geringen räumlichen Abstände sowie die auftretenden Fahrzeuggeschwindigkeiten angepasst werden [13, S. 154]. Hinzu kommt, dass Optimierungsprobleme ohne die Forderung nach ganzzahligen Ergebnissen für die Optimierungsvariablen deutlich effizienter gelöst werden können als solche mit dieser Forderung. Die Algorithmen, die für das MILP-Verfahren verwendet werden müssen, sind deutlich rechenaufwendiger und damit langsamer [35, S. 326-328].

2.3.4 Elastische Bänder

Die Methode der elastischen Bänder wurde bereits in Unterkapitel 2.2 eingeführt, weil sie häufig für die Berechnung von Einzelfahrzeugtrajektorien angewendet wird. Die Methode wurde von Quinlan [31] für Roboter als Brückenschlag zwischen der globalen Trajektorienplanung und deren echtzeitfähiger, sensorbasierter Anpassung an das Umfeld vorgeschlagen. Zuerst berechnet ein Algorithmus zur Trajektorienplanung eine initiale Trajektorie für das gegebene Szenario. Anschließend wird diese Trajektorie in Echtzeit an Abweichungen von der vorgegebenen Trajektorie, Veränderungen des Umfelds oder unerwartet auftretende Hindernisse angepasst [31, S. 802]. Vor allem wenn der Roboter von der ursprünglichen Trajektorie abweicht, muss kein komplett neuer Weg zum Ziel berechnet werden, weil die Methode der elastischen Bänder die ursprüngliche Trajektorie nie verwirft, sondern nur anpasst. Die globale Information, wie das Ziel zu erreichen ist, geht nicht verloren [31, S. 802].

Die Idee hinter der Methode der elastischen Bänder ist, dass virtuelle Kräfte auf die ursprüngliche Trajektorie wirken und diese der Umgebung entsprechend verformen. Dabei wirken zum einen Kräfte, die die Trajektorie geradeziehen, ähnlich wie die Spannung in einem gedehnten Gummiband, und damit unstetige Verläufe beseitigen. Zum anderen rufen Hindernisse Kräfte hervor, die das elastische Band davon abstoßen [31, S. 803]. Die wirkenden Kräfte verformen das elastische Band solange, bis ein Kräftegleichgewicht vorliegt. Dann ist die lokal angepasste Trajektorie gefunden. Diese Methode wurde bereits mehrfach in Fahrzeugen angewendet und auf die speziellen Anforderungen hin angepasst. Sattel [40] greift die Idee der elastischen Bänder auf und modifiziert sie mit Fokus auf Kollisionsvermeidung. Dazu werden die Positionswerte $\boldsymbol{x}_{pos}$, die die initiale Trajektorie zu bestimmten Zeitpunkten t vorgibt, mit virtuellen Federn verbunden. Zwischen den Positionswerten, auch Knoten bezeichnet, treten dadurch Kräfte auf, durch die sich die Knoten voneinander abstoßen oder gegenseitig anziehen. Das bewirkt, dass die Bewegung hinsichtlich der Beschleunigung und damit des Geschwindigkeitsverlaufs geglättet wird. Zusätzlich werden Kräfte formuliert, die das Fahrzeug vom Fahrbahnrand weg in Richtung der Fahrbahnmitte stoßen. Bei der Suche nach dem Kräftegleichgewicht an jedem Knoten werden diese so lange verschoben, bis das Gleichgewicht erreicht ist. Abschließend wird die Trajektorie mit kubischen Splines durch die neuen Positionsknoten interpoliert [40, S. 4992-4993]. Auch Hilgert [41] verwendet ein ähnliches Verfahren zur Bestimmung der wirkenden Kräfte. Aktuellere Arbeiten zu diesem Thema stammen von Keller [42] und Götte [43].

Explizit auf die kooperative Trajektorienplanung schneidet Frese [13, S. 118-128] das Verfahren der elastischen Bänder zu. Er verwendet dabei das Konzept der virtuellen Federn und definiert Kräfte, die zwischen den Knoten innerhalb einer Fahrzeugtrajektorie, zwischen den Knoten der Fahrzeugtrajektorien von kooperierenden Fahrzeugen und zwischen Knoten und der Umgebung wirken. Weil diese Knoten nur Informationen über die jeweilige Fahrzeugposition $\boldsymbol{x}_{pos}$ enthalten, müssen die Geschwindigkeit v und die Fahrzeugorientierung θ immer davon abgeleitet werden. Für die Positionsknoten wird ein gleichmäßiger zeitlicher Abstand Δt gefordert. Durch diese Zeitdiskretisierung lassen sich aus der aktuellen und der vorherigen Position die beiden Werte Geschwindigkeit und Fahrzeugorientierung berechnen [13, S. 119]. Die Fahrzeugzustände zum Startzeitpunkt sind bekannt.

Auf den Positionsknoten des Zeitpunktes t einer Trajektorie des i-ten Fahrzeugs wirkt eine virtuelle Kraft $\boldsymbol{f}_{Typ,i}(t)$, die sich aus mehreren Einzelkräften zusammensetzt. Tiefergestellt wird der Typ der wirkenden Kraft notiert. Die resultierende virtuelle Kraft $\boldsymbol{f}_{Typ,i}(t)$ ist vektoriell in x- und y- Komponenten des globalen Koordinatensystems geteilt. Die Erklärung der

auftretenden Einzelkräfte orientiert sich vorwiegend an Frese [13, S. 120-126]. Er unterscheidet generell zwischen inneren Kräften (longitudinale und laterale Kraft), äußeren Kräften (Straßenrandkraft und Hinderniskraft) und kooperativen Kräften (Kraft zwischen kooperierenden Fahrzeugen). Auf den Fahrzeugindex i wird, sofern nicht zwingend erforderlich, verzichtet und nur der allgemeine Fall für ein Fahrzeug betrachtet.

- **Longitudinale Kraft:** Der Vektor der longitudinalen Kraft greift am Knoten des aktuellen Zeitschritts $\boldsymbol{x}_{pos}(t)$ stets in Richtung des Knotens des vorherigen Zeitschritts $\boldsymbol{x}_{pos}(t-\Delta t)$ oder des nächsten Zeitschritts $\boldsymbol{x}_{pos}(t+\Delta t)$ an, wie aus Abb. 2.7 hervorgeht. Beschleunigt das Fahrzeug, wird der Knoten $\boldsymbol{x}_{pos}(t)$ zu dem Knoten des vorherigen Zeitschritts gezogen. Verzögert das Fahrzeug, dann erfolgt eine Abstoßung der beiden Knoten und die Kraft wirkt in Richtung des Knotens des nächsten Zeitschritts. Bei einer Bewegung mit konstanter Geschwindigkeit treten keine longitudinalen Kräfte auf. Mithilfe einer Kostenfunktion können zu hohe Beschleunigungs- und Verzögerungswerte bestraft und die Einhaltung der fahrdynamischen Grenzwerte erzwungen werden. Die longitudinalen Kräfte glätten den Verlauf der Längsbewegung. Zusammen mit den lateralen Kräften wird die maximal mögliche Gesamtbeschleunigung mit dem Kamm'schen Kreis bestimmt [13, S. 120-121].
- **Laterale Kraft:** Die lateralen Kräfte wirken an den jeweiligen Knoten quer zu den longitudinalen Kräften, also quer zur direkten Verbindung zwischen aktuellem Knoten und dem jeweils vorausgehenden oder nachfolgenden. Die Wirkrichtung der lateralen Kraft ist in der unteren Hälfte von Abb. 2.7 dargestellt.

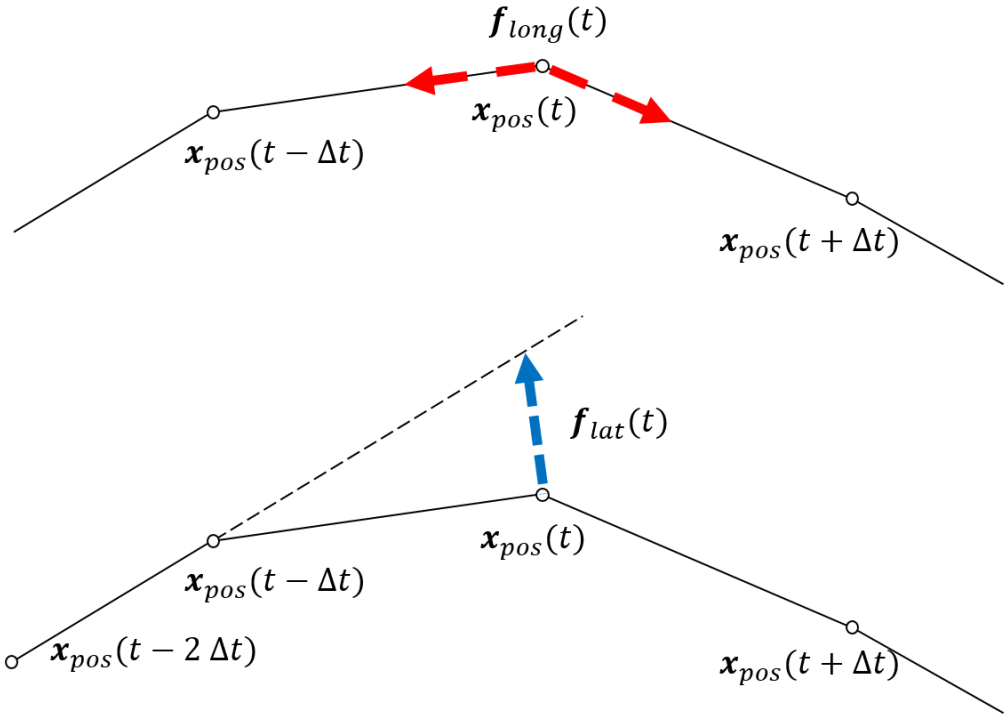


Abb. 2.7 Longitudinale (oben) und laterale (unten) Kraft auf den aktuellen Knoten nach Frese [13, S. 121]

Angenommen wird, dass die longitudinale Kraft in Richtung des Knotens zum vorherigen Zeitpunkt wirkt. Deshalb steht die laterale Kraft senkrecht auf der Verbindungslinie zwischen aktuellem und vorherigem Knoten. Durch die laterale

Kraft wird die Querbeschleunigung abgebildet und die Einhaltung der fahrdynamischen Beschränkungen durchgesetzt. Sie wirkt stets in die Richtung, in die sich das Fahrzeug bewegten müsste, um geradeaus zu fahren. Die Richtung der Geradeausfahrt ist in diesem Fall die verlängerte Gerade zwischen den Positionsknoten des vorletzten Zeitschritts $\boldsymbol{x}_{pos}(t - 2\,\Delta t)$ und des letzten Zeitschritts $\boldsymbol{x}_{pos}(t - \Delta t)$. Liegen drei aufeinanderfolgende Knoten auf einer Geraden, so erfährt der dritte Knoten keine laterale Kraftwirkung. Dadurch, dass die laterale Kraft die Knoten senkrecht zur longitudinalen Kraft verschiebt, überlagern sich beide Kraftwirkungen nicht. Über eine Kostenfunktion werden hohe Querbeschleunigungswerte bestraft und nicht zulässig hohe Werte verhindert. Zusammen mit der longitudinalen Kraft eines Knotens ergibt sich die resultierende Gesamtkraft, die innerhalb des fahrdynamisch zulässigen Bereichs liegen muss. Die nicht-holonome Kinematik wird vernachlässigt, weil bei hohen Geschwindigkeiten dynamische Einschränkungen überwiegen [13, S. 122].

- **Straßenrandkraft:** Die Straßenrandkraft wirkt normal zum Fahrbahnrand in Richtung der Fahrbahnmitte. Befindet sich ein Knoten zu nahe am Fahrbahnrand, entsteht eine Kraft, die den entsprechenden Knoten davon wegdrückt. Das gleiche gilt für einen Knoten, der nicht auf der Straßenfläche liegt. Der minimale Abstand kann als feste Schranke definiert werden, bei deren Unterschreitung die Kraft zu wirken beginnt [13, S. 123]. Alternativ kann die Fahrbahn durch ein Potentialfeld abgebildet werden, dessen Minimum in der Fahrbahnmitte liegt [40, S. 4992]. Je stärker der Positionsknoten von der Fahrbahnmitte abweicht, desto größer wird die wirkende Straßenrandkraft. Anstatt die abstoßenden Kräfte nur an den Fahrbahnrändern erscheinen zu lassen, können diese auch für jede Fahrspur definiert werden. Auf diese Weiße orientieren sich Fahrzeuge nach einem Spurwechsel immer in der Mitte der Fahrspur. Ebenfalls kann ein Potentialfeld erzeugt werden, dass von links nach rechts abfällt. Dadurch orientieren sich die Fahrzeuge bei freier Fahrt möglichst weit rechts auf der Fahrbahn [13, S. 123].

  In Abb. 2.8 ist dargestellt, wie die Straßenrandkräfte auf die jeweiligen Knoten wirken. In der oberen Darstellung ist ein Fahrzeug zu sehen, das an zwei Hindernissen vorbeifährt. Die Straßenrandkraft wirkt senkrecht zum Fahrbahnrand (grün/ausgefüllter Pfeil).

- **Kraft zwischen kooperierenden Fahrzeugen:** Dieser Kraft-Typ wird speziell dazu eingeführt, um kooperative Fahrmanöver zu ermöglichen [13, S. 124]. In anderen Arbeiten kann nur das Ego-Fahrzeug Aktionen durchführen und alle weiteren beteiligten Fahrzeuge sind bewegte Hindernisse, deren Bewegung abgeschätzt werden kann. Mit der Kraft zwischen kooperierenden Fahrzeugen eröffnet sich die Möglichkeit, dass sich mehrere Fahrzeuge gegenseitig beeinflussen. Entscheidend für die Höhe der wirkenden Kraft ist der euklidische Abstand zwischen zwei Positionsknoten. Die Fahrzeugmaße werden in der Berechnung des Abstands berücksichtigt. Zu beachten ist, dass die Kräfte zwischen den kooperierenden Fahrzeugen nur zwischen den Knoten mit gleichem Zeitindex wirken. Andernfalls würden zeitlich frühere Knoten die aktuellen Knoten abstoßen. Ein Überfahren der gleichen Fahrbahnfläche zu unterschiedlichen Zeitpunkten wäre dann nicht möglich. Außerdem werden aufeinander zufahrende Fahrzeuge stärker voneinander abgestoßen als Fahrzeuge, die sich in dieselbe Richtung bewegen [13, S. 124-125].

Im unteren Teil von Abb. 2.8 sind die Trajektorien von zwei aufeinander zufahrenden Fahrzeugen abgebildet. Die wirkenden kooperativen Kräfte (orange/nicht ausgefüllter Pfeil) sind zu den jeweiligen Zeitpunkten $t = 6$ und $t = 7$ zu erkennen. Sie sind betragsmäßig gleich groß und wirken entgegengesetzt auf die jeweiligen Knoten.

- **Hinderniskraft:** Ähnlich wie bei der Kraft zwischen kooperierenden Fahrzeugen ist bei den Kräften, die Hindernisse auf Positionsknoten ausüben, der Abstand der entscheidende Parameter. Der Abstand eines Positionsknotens zum Hindernis wird explizit in eine Formel eingesetzt. Die Hinderniskraft kann alternativ über ein das Hindernis umgebendes Potentialfeld bestimmt werden [40, S. 4992-4993]. Berücksichtigt werden können statische und bewegte Hindernisse [13, S. 125-126]. In Abb. 2.8 ist im oberen Teil die Situation mit einem Einzelfahrzeug, das zwischen zwei Hindernissen manövriert, dargestellt. Die Hinderniskraft (dunkelrot/nicht ausgefüllter Pfeil) stößt die Positionsknoten des elastischen Bandes ab, wodurch die Trajektorie von den Hindernissen weggeschoben wird.

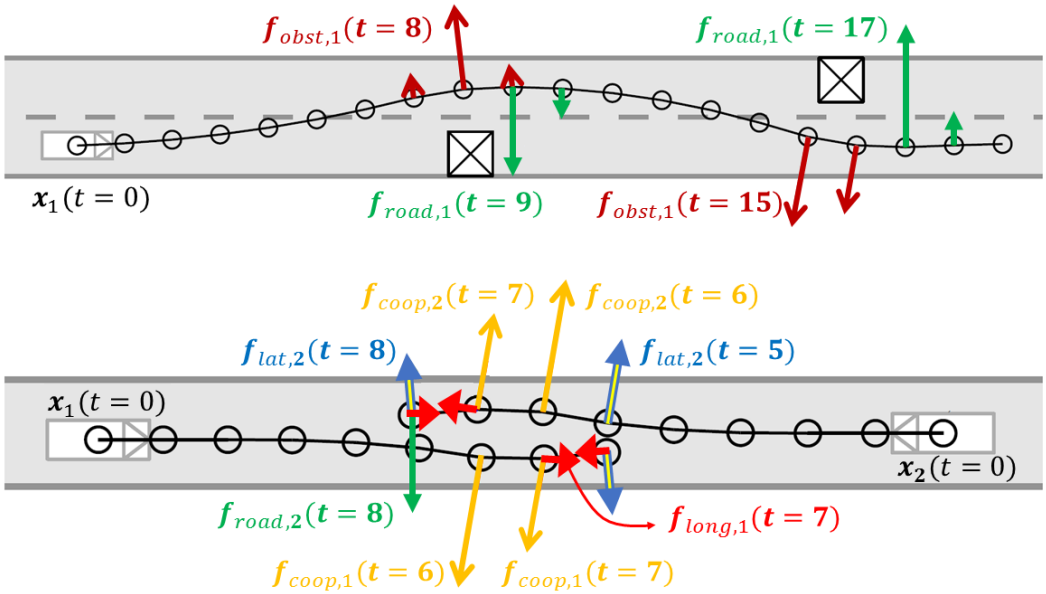


Abb. 2.8 Visualisierung der elastischen Bänder und ausgewählte, darauf wirkende Kräfte bei Hindernissen (oben) und bei einem kooperativen Fahrmanöver (unten) nach Frese [13, S. 126]

Frese [13, S. 127] erzeugt die notwendige Ausgangs-Trajektorie, indem er den Straßenverlauf berücksichtigt und eine konstante Geschwindigkeit annimmt. Auf Grundlage dieser Trajektorie wird die Methode der elastischen Bänder angewendet. Während der Bewegungsplanung werden die Positionsknoten der ursprünglichen Trajektorie verschoben, indem die virtuellen Kräfte aufgebracht werden. Die Kräfte an einem Knoten werden vektoriell aufsummiert und jeweils mit Gewichtungsfaktoren multipliziert [13, S. 127]. Die Verschiebung der Knoten und die Berechnung der daraus resultierenden Kraft geschieht iterativ so lange, bis sich ein Kräftegleichgewicht einstellt. Kräftegleichgewicht bedeutet, dass die resultierende Kraft an jedem Knoten verschwindet. Wenn dieses vorliegt, dann ist das kooperative Fahrmanöver gefunden. Das kooperative Fahrmanöver stellt sozusagen einen Kompromiss aus allen vorliegenden Einzelkräften dar [13, S. 128]

Ein großer Vorteil des Verfahrens ist, dass der Rechenaufwand nicht, wie bei den meisten anderen Verfahren, exponentiell mit der Anzahl der Fahrzeuge ansteigt, sondern nur

polynomiell [13, S. 128]. Die Methode zeichnet sich durch eine gute Performance in der Berechnung von Einzelfahrzeugtrajektorien aus und ist eine der wenigen Verfahren, die die gestellten Anforderungen unter Echtzeitbedingungen erfüllen kann [32, S. 196]. Eine weitere Stärke ist, dass auf plötzlich veränderte Umgebungsbedingungen reagiert werden kann, ohne dass der zuvor berechnete, globale Pfad verworfen und neu kalkuliert werden muss. Außerdem erfolgt keine Diskretisierung der Werte für die Fahrzeugbeschleunigung und den Lenkwinkel, sodass diese besser im realen Fahrzeug umgesetzt werden können.

Nachteilig ist vor allem die Tatsache, dass das Verfahren der elastischen Bänder nur lokale Trajektorien optimiert. Das schränkt die Lösung problemabhängig deutlich ein, weil das Verfahren unter Umständen keine Lösung finden kann, obwohl dies prinzipiell möglich ist [31, S. 803]. In solchen Fällen und zur Initialisierung ist stets ein anderer Algorithmus zur Trajektorienplanung nötig, um die Methode der elastischen Bänder anwenden zu können.

2.3.5 Baumsuche

Ein Graph besteht aus Kanten und Knoten (Ecken). Eine Kante verbindet jeweils zwei Knoten miteinander, wobei von einem Knoten mehrere Kanten zu anderen Knoten ausgehen können [44, S. 2-3]. Ein Baum ist ein besonderer Graphen-Typ, der keine sogenannten Kreise enthält. Als Kreis wird ein geschlossener Kantenzug bezeichnet, der jede seiner Ecken höchstens einmal enthält [44, S. 46]. Anschaulich bedeutet das, dass die Blätter an der Spitze eines Baumes den Stamm berühren und dadurch einen Kreis herstellen. Dann wäre der Graph kein Baum mehr. In der Graphentheorie wird prinzipiell zwischen ungerichteten und gerichteten Graphen unterschieden. Bei einem gerichteten Graphen, einem sogenannten Digraphen (engl.: directed graph), können die Kanten zwischen zwei Knoten nur in eine Richtung durchschritten werden. Bei einem ungerichteten Graphen sind beide Richtungen zulässig [44, S. 133]. Eine gerichtete Baumstruktur eignet sich besonders zur Abbildung hierarchischer Abhängigkeiten [45, S. 52]. Ein Beispiel dafür sind Handlungsanweisungen, die aus mehreren Schritten bestehen und nacheinander abgearbeitet werden sollen [44, S. 133]. Durch die gerichteten Kanten ist die Reihenfolge der Handlungen klar bestimmt. Durch die Baumstruktur ist garantiert, dass der Graph keinen Kreis enthält, durch den ein bereits abgearbeiteter Handlungsschritt erreicht werden könnte. Der letzte Knoten des Graphen stellt das Ende der Handlungsanweisungen dar. Vor allem wenn die Knoten nicht die Reihenfolge an sich festlegen, sondern eine zeitliche Abfolge von Aktionen, wird deutlich, dass sich die Baumstruktur besonders anbietet. Ein Rückschritt in der Zeit durch eine ungerichtete Kante oder durch einen Kreis ist allein aus physikalischer Sicht unmöglich.

Die Knoten eines Baumes werden in Abhängigkeit ihrer Position im Baum und der ein- und ausgehenden Kanten verschieden bezeichnet. Die Nomenklatur unterscheidet zwischen Elternknoten (engl.: parents), Kindknoten (engl.: children), Geschwisterknoten (engl.: siblings), Blattknoten (engl.: leaves) und einem Wurzelknoten (engl.: root) [45, S. 51-52]. In einem Baum kann es nur einen Knoten geben, der als Wurzelknoten bezeichnet wird. Das ist i.d.R. der erste Knoten von dem aus die Baumstruktur aufgebaut wird. Dann hat der Wurzelknoten nur ausgehende und keine eingehenden Kanten. Elternknoten sind diejenigen Knoten, die mindestens eine ausgehende Kante haben. Kindknoten sind wiederum Knoten, die mindestens eine eingehende Kante haben. Auf die meisten Knoten eines Baumes trifft beides zu, weil sie eine eingehende und mindestens eine ausgehende Kante besitzen. Ob der Knoten als Eltern- oder Kindknoten bezeichnet wird, hängt von der konkreten Betrachtungsweise ab. Knoten, die denselben Elternknoten besitzen, sind Geschwisterknoten. Blattknoten werden

Knoten genannt, die keine ausgehenden Kanten und damit keine Kindknoten besitzen. Sie befinden sich am „Ende“ des Baumes. Alle Knoten, die weder Wurzelknoten noch Blattknoten sind, werden als innere Knoten (engl.: interior nodes) zusammengefasst [45, S. 52].

Bei der Anwendung der Baumstruktur zur Planung von kooperativen Fahrmanövern repräsentieren die einzelnen Knoten jeweils einen Zustand des Systems. Dieser Systemzustand $\boldsymbol{X}_{Veh}(t)$ beinhaltet alle Einzelfahrzeugzustände $\boldsymbol{x}_i(t)$ (mit $i = 1, \dots, m$) der m Fahrzeuge. Der Systemzustand kann mithilfe der Zustandsübergangsfunktion aus Gl. (2.4) in einen Zustand des nächsten Zeitschritts $\boldsymbol{X}_{Veh}(t + \Delta t)$ überführt werden. Die Zustandsübergangsfunktion basiert auf den verwendeten Fahrzeugmodellen. Es können beliebige Modelle für Geometrie, Kinematik und Dynamik der Fahrzeuge implementiert werden, womit unter anderem die nicht-holonome Fahrzeugkinematik problemlos berücksichtigt werden kann [13, S. 76-77]. In diesem Fall wird auf das Einspurmodell aus Abschnitt 2.3.1 zurückgegriffen. Die Berechnung der Position und Geschwindigkeit zum nächsten Zeitschritt erfolgt mit Gl. (2.5) und (2.6). Dadurch, dass die Orientierung der Fahrzeuge bei der Baumsuche explizit mitgeführt wird, sind geringfügige Anpassungen an der Berechnung von Position und Geschwindigkeit vorzunehmen. Bei der Vorstellung der Gleichungen wurde die Orientierung nicht berücksichtigt [13, S. 78]. Die Orientierung kann durch nichtlineare Zusammenhänge mit Gl. (2.7) ausgedrückt werden. Die Eingangsgrößen, die eine Veränderung des Zustands bewirken, sind die Beschleunigung a und der Lenkwinkel δ. Für die Baumsuche müssen diese diskretisiert und zusammengefasst werden, um konkrete Handlungen $a_{act,ij} = (a_{ij}, \delta_{ij})$ zu definieren. Diese Auswahl an Aktionen $\boldsymbol{a}_{act,i}$ stehen jedem Fahrzeug grundsätzlich zur Verfügung. Für den Lenkwinkel kann ein Intervall mit zulässigen Werten $[-\delta_{max}, \delta_{max}]$ angegeben werden, die sich aus der Fahrzeugkinematik ergeben. Die maximal mögliche Beschleunigung ist i.d.R. durch die Motorcharakteristik des Fahrzeuges gegeben. Die maximalen Werte für Verzögerung und Querbeschleunigung werden mithilfe des Kamm’schen Kreises ermittelt, in dem alle auftretenden Beschleunigungen berücksichtigt werden [13, S. 78-79]. Im Sinne der Problemlösung wäre eine möglichst feine Diskretisierung optimal, weil andernfalls der Lösungsraum verkleinert wird. Wie allerdings Gl. (2.8) zeigt, hat die Anzahl der angebotenen Aktionen einen großen Einfluss auf die Anzahl der Permutationen und damit unmittelbaren Einfluss auf den Rechenaufwand. Die Anzahl aller Permutationen entspricht bei einer Baumstruktur der Anzahl der Blattknoten bzw. der möglichen unterschiedlichen Pfade vom Wurzelknoten zu einem Blattknoten. Bei $m = 1$ Fahrzeugen, $k = 2$ wählbaren Aktionen und $t_{end} / \Delta t = 3$ Zeitschritten bis zum Planungshorizont ergeben sich nach Gl. (2.8) acht mögliche Pfade, wie sie in Abb. 2.9 gezeigt werden.

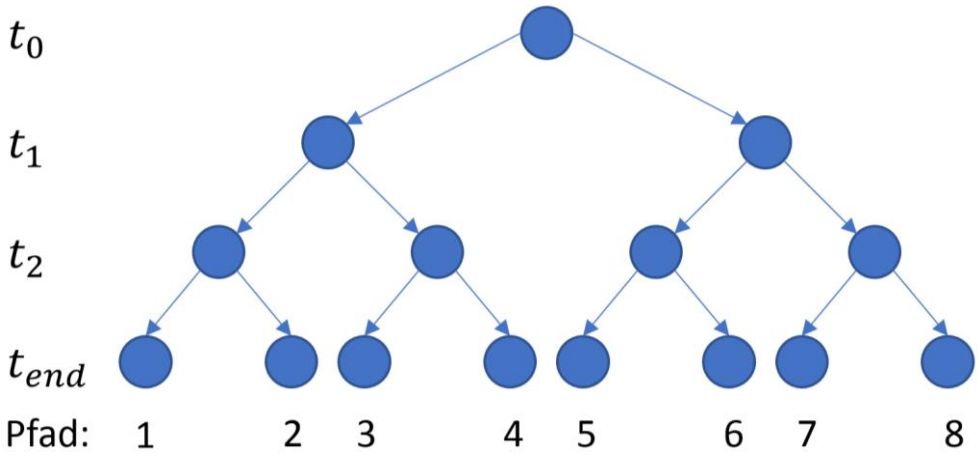


Abb. 2.9 Darstellung aller möglichen Pfade bei einer Baumstruktur

Es muss folglich ein Kompromiss gefunden werden zwischen möglichst feiner Diskretisierung und möglichst wenigen wählbaren Aktionen. Frese [13, S. 80-82] definiert insgesamt neun verschiedene Handlungen, die sich überwiegend aus Maximalwerten zusammensetzen. Er geht davon aus, dass in Extremsituationen zur Unfallvermeidung maximale Aktionen optimal sind. Lenz [21, S. 450] beschränkt sich auf sieben Handlungen, die jedoch auf einen gleichmäßig fließenden Verkehrsfluss fokussiert sind und nur moderate Beschleunigungswerte annehmen. Der Lenkwinkel wird nicht diskretisiert. Stattdessen wird ein Spurwechsel mit konstanter Geschwindigkeit als eine Aktion vorgegeben. Die Beschleunigungswerte orientieren sich an einem Fahrermodell oder sind fest vorgeben.

Damit ein Knoten expandiert werden kann, müssen zuerst für jedes Fahrzeug aus den wählbaren Aktionen diejenigen verworfen werden, die aufgrund der Verkehrssituation nicht zulässig sind. Die Verkehrssituation $S_{VS}(t)$ enthält die Fahrbahn, die Hindernisse und die Fahrzeuge (Gl. (2.2) und (2.3)). Daraus ergibt sich eine Menge an Aktionen $\boldsymbol{A}_{Veh}$, die sich aus den jeweilig zulässigen Aktionen $\boldsymbol{a}_{act,i}$ aller beteiligten Fahrzeuge zusammensetzt. Jede der resultierenden Permutationen dieser Einzelfahrzeugaktionen $a_{act,res}$ bringen durch die Zustandsübergangsfunktion f einen neuen Zustand $\boldsymbol{X}_{Veh}$ zum Zeitpunkt $t + \Delta t$ hervor. Diese neuen Zustände sind die Kindknoten des Zustandes zum Zeitpunkt t. Die Anzahl der Kindknoten des aktuellen Zustands berechnet sich aus den Permutationen, die aus der Menge der Aktionen aller Fahrzeuge $\boldsymbol{A}_{Veh}$ gebildet werden können. In Gl. (2.9) ist dieser Zusammenhang dargestellt. Die Anzahl der Fahrzeuge ist m. Die Variable k_i entspricht der Anzahl an zulässigen Aktionen für jedes Fahrzeug zum betrachteten Zeitpunkt.

$$Anzahl\ der\ Permutationen = \prod_{i=1}^{m} k_i \tag{2.9}$$

Bei dem Beispiel aus Abb. 2.9 entspricht das jeweils zwei resultierenden Kindknoten, weil die Anzahl der Fahrzeuge $m = 1$ und die Anzahl an zulässigen Aktionen $k_1 = 2$ ist. In Abb. 2.10 ist beispielhaft dargestellt, wie die Expansion des Wurzelknotens zum Zeitpunkt t_0 abläuft. Für Fahrzeug 1 stehen drei Aktionen $k_1 = 3$ und für Fahrzeug 2 zwei Aktionen $k_2 = 2$ zur Auswahl. Die Anzahl der Permutationen $a_{act,res}$ entspricht $k_1\ k_2 = 6$ nach Gl. (2.9).

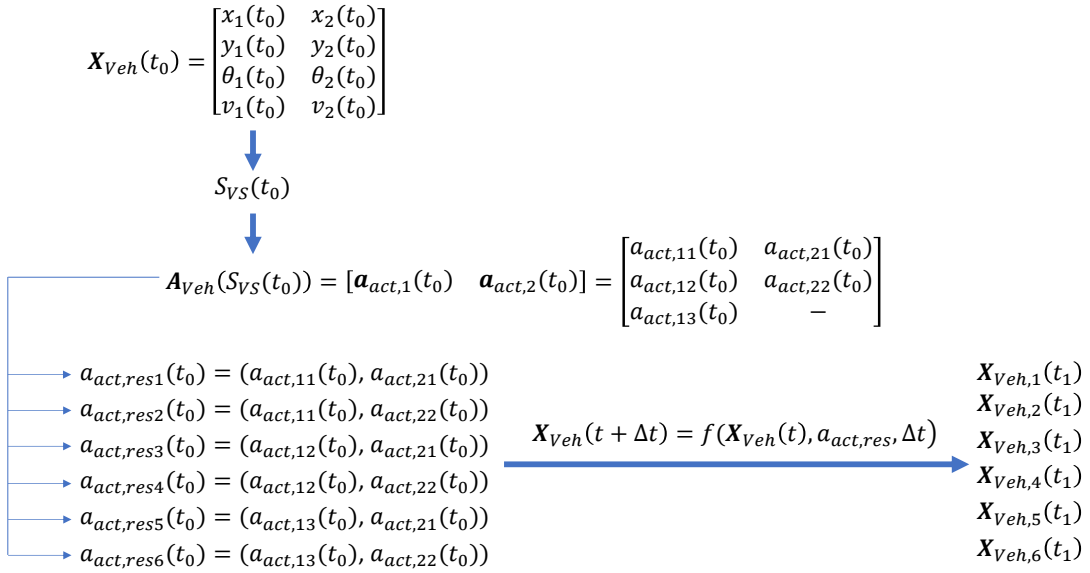


Abb. 2.10 Vorgehen bei der Expansion eines Zustands/Knotens

Mit der Zustandsübergangsfunktion, in der Abbildung nicht auf die Einzelfahrzeugzustände, sondern auf den Zustand aller Fahrzeuge bezogen, ergeben sich sechs neue Zustände zum Zeitpunkt $t_1 = t + \Delta t$. Die neuen Zustände können anschließend auf die gleiche Weise expandiert werden. Das geschieht so lange, bis alle Blattknoten auf der Zeitebene des Planungshorizonts zum Zeitpunkt t_{end} liegen. Ist dies der Fall, dann ist der komplette Suchbaum aufgebaut. Zu diesem Zeitpunkt liegt die optimale Abfolge von Handlungen im Suchbaum vor, allerdings muss sie erst gefunden werden. Dies erfordert eine Bewertung, welche Zustände und Aktionen gewünscht sind oder bevorzugt werden und welche unerwünscht oder unzulässig sind. Mithilfe von Kostenfunktionen lassen sich Aktionen und die resultierenden Zustände bewerten. Damit wird eine Möglichkeit geschaffen, Zustände miteinander zu vergleichen. Jedem Knoten wird ein positiver Wert zugeordnet. Dieser hängt ab vom Zustand, den der Knoten repräsentiert, und der Aktion, die zu diesem Zustand geführt hat. Der zugeordnete Wert wird als „Kosten“ bezeichnet. Sie setzen sich aus mehreren Einzelkosten zusammen, die je nach Bedarf definiert werden. Sie erfüllen den Zweck, den die Gewichtungsfaktoren bei der Methode der elastischen Bänder haben. Frese [13, S. 82-90], dessen Schwerpunkt auf Unfallvermeidung liegt, setzt die Kosten aus den folgenden Bestandteilen zusammen: Kosten für Kollisionen mit anderen Fahrzeugen und Hindernissen, Kosten, die die Schwere einer Kollision berücksichtigen, und Kosten für das Verlassen der Fahrbahn. Außerdem definiert er Kosten, die die Ziel- und Regelkonformität der Fahrzeughandlungen bewerten. Zielkonformität bedeutet, dass Kosten anfallen, wenn von der vorgegebenen Route zum Fahrziel abgewichen wird. Kosten für die Regelkonformität setzen Verkehrsregeln durch, wie das Rechtsfahrgebot oder Geschwindigkeitsbegrenzungen. Die letzte Kostenart bewertet die Aktionen der Fahrzeuge hinsichtlich Komfort und Energieverbrauch. Geringe Werte für Beschleunigung und Verzögerung sowie für den Lenkwinkel werden dadurch favorisiert.

Bei Lenz [21, S. 450] werden andere Kostenterme festgelegt, weil der Schwerpunkt auf der Berechnung kooperativen Trajektorien im fließenden Verkehr und nicht in Extremsituationen liegt. Die Einzelkosten werden berechnet aus der Abweichung zwischen aktueller und gewünschter Geschwindigkeit, der Höhe der Beschleunigung und Verzögerung und dem längswertigen Sicherheitsabstand zum vorausfahrenden Fahrzeug. Kosten fallen außerdem für einen durchgeführten Spurwechsel und, wie bei Frese, für eine Kollision mit anderen Fahrzeugen und Hindernissen an. Wichtig ist in beiden Fällen, dass die Einzelkosten, in Abhängigkeit ihrer Bedeutung für die kooperative Fahrmanöverplanung, eine hierarchische Abstufung erhalten [13, S. 89]. Ein Beispiel sind die Kosten für eine Kollision: die niedrigsten Kosten, die dafür anfallen können, müssen immer größer sein als alle anderen Einzelkosten. Andernfalls wäre es bspw. möglich, dass anstatt einer sehr großen Abweichung von der Wunschgeschwindigkeit eine Kollision bevorzugt wird. Dieser Fall darf nicht auftreten und kann mit Gewichtungsfaktoren, die den jeweiligen Einzelkosten zugewiesen sind, verhindert werden. Zur Bewertung der Knoten können die jeweiligen Kosten herangezogen werden. Für das Finden des optimalen Pfades müssen alle Kosten der Knoten aufsummiert werden, die auf dem Weg vom Wurzelknoten zu einem Blattknoten durchschritten werden. Daraus ergibt sich für jeden Pfad ein Kostenwert (Anzahl der möglichen Pfade entspricht der Anzahl der Blattknoten). Der Pfad mit den niedrigsten Kosten ist der gesuchte, optimale Pfad.

Das Aufbauen einer Baumstruktur stellt an sich keine Schwierigkeit dar. Problematisch wird das allerdings, wenn es so viele Knoten gibt, dass das komplette Aufbauen des Suchbaumes nicht realisierbar ist. Bereits bei einer überschaubaren Anzahl an Fahrzeugen wird der resultierende Suchbaum extrem groß, wie am Ende des Abschnitts 2.3.1 mit Gl. (2.8) gezeigt wird. Das komplette Aufbauen des Suchbaums erfordert eine hohe Rechenzeit. Zusätzlich

kann der Speicherbedarf die verfügbaren Ressourcen weit übersteigen. Vor allem im Hinblick auf Echtzeitfähigkeit ist es nicht praktikabel, den Suchbaum vollständig aufzubauen. Damit trotzdem die optimale Abfolge an Handlungen gefunden werden kann, muss ein Baumsuchalgorithmus eingesetzt werden. Dessen Aufgabe ist, den optimalen Pfad durch den Suchbaum zu finden, obwohl nur ein Bruchteil von diesem wirklich aufgebaut wird. Einfache Suchverfahren wie Tiefensuche (engl.: Depth-First) oder Breitensuche (engl.: Breadth-First), die den Suchbaum ohne Systematik nur der Reihenfolge nach aufbauen, führen bei der Größe des vorliegenden Suchbaumes zu keinem zufriedenstellenden Ergebnis [19, S. 35-36]. Es müssen Algorithmen verwendet werden, die in der Lage sind, vielversprechende Bereiche des Suchbaumes näher zu untersuchen und dafür Teile des Baumes zu vernachlässigen, die wenig Aussicht auf Erfolg bieten. Treten unzulässige Zustände auf, können die nachfolgenden Knoten alle verworfen werden. Drei Suchalgorithmen mit den genannten Eigenschaften werden nachfolgend vorgestellt: Branch-and-Bound, A* (A-Star) und Monte-Carlo Baumsuche (engl.: Monte-Carlo Tree Search, MCTS).

- **Branch-and-Bound:** Der Branch-and-Bound-Algorithmus stellt eine Spezialform der Tiefensuche dar. Zum einen untersucht er, wie die Tiefensuche, jeden möglichen Pfad. Er besucht also jeden Blattknoten. Zum anderen speichert er die Kosten des bisher gefundenen besten Pfades ab [45, S. 246-247]. Erreicht der Algorithmus das erste Mal einen Knoten des Zeitpunktes t_{end}, werden die Kosten für diese Handlungsabfolge gespeichert. Anschließend beginnt der Algorithmus, sich vom Wurzelknoten aus wieder in Richtung des nächsten Blattknotens zum Zeitpunkt t_{end} zu arbeiten. Bei jedem Knoten, den er dabei erreicht, werden die aktuellen Kosten mit den gespeicherten verglichen. Stellt sich heraus, dass die aktuellen Kosten bereits größer sind als die bisher optimalen für eine komplette Handlungssequenz, wird der Knoten verworfen (und damit auch der Teilbaum, der von diesem Knoten ausgegangen wäre). Der Algorithmus startet wieder am Wurzelknoten. Zusätzlich können untere Schranken definiert werden, die angeben, mit welchen Kosten von einem bestimmten Knoten aus mindestens zu rechnen ist. Ist die Summe aus aktuellen und mindestens zu erwartenden Kosten größer als die bisher gefunden optimalen Kosten, so kann auch dieser Knoten/Teilbaum verworfen werden [13, S. 102-103].

  Der Vorteil des Verfahrens ist vor allem der geringe Speicheraufwand, weil nur die aktuell beste Handlungssequenz und deren Kosten gespeichert werden müssen [13, S. 103]. Außerdem liegt damit eine verlässliche obere Schranke für die Kosten vor, weil diese nicht geschätzt, sondern direkt aus einer konkreten Handlungssequenz ermittelt wird. Nachteilig ist die Tatsache, dass der Algorithmus im schlechtesten Fall alle Blattknoten besuchen muss, um die optimale Lösung zu finden. Außerdem ist die Definition und Berechnung einer unteren Schranke mit zusätzlichem Speicher- und Rechenaufwand verbunden, was sich negativ auf die Performance des Algorithmus auswirkt [13, S. 101].

- **A*:** Der A*-Algorithmus basiert auf dem Dijkstra-Algorithmus und ist in der Lage, die Anzahl der besuchten Knoten im Vergleich zu diesem deutlich zu verringern [19, S. 37]. Dazu wird eine untere Schranke für jeden Knoten bestimmt, die angibt, welche Kosten bis zum Ziel mindestens zu erwarten sind. Voraussetzung ist, dass diese Heuristik die tatsächlich auftretenden Kosten immer unterschätzt. Im Vergleich zu Branch-and-Bound werden beim A*-Algorithmus nicht nur die aktuell beste gefundene Handlungssequenz und deren Kosten gespeichert.

Vielmehr werden alle bereits expandierten Knoten gespeichert. Zusätzlich dazu wird die Summe der Kosten hinterlegt, die vom Wurzelknoten bis zu dem jeweiligen Knoten aufgetreten sind. Damit ist es möglich, immer den Knoten auszuwählen, dessen Summe aus aktuellen und mindestens zu erwartenden Kosten am geringsten ist. Dieser Knoten stellt den aussichtsreichsten Zustand dar und wird expandiert. Die Kosten des neuen Knotens werden angepasst und die mindestens zu erwartenden Kosten vom neuen Knoten bis zum Ziel gespeichert. Anschließend wird wieder der beste und aussichtsreichste Knoten gesucht, ausgewählt und expandiert. Das geschieht so lange, bis der Algorithmus zu einem Blattknoten vorgedrungen ist. Der erste auf dieser Ebene gefundene Knoten und die dahin führende Handlungsabfolge sind optimal. Im Vergleich zu Branch-and-Bound wird der Baum nicht der Reihe nach abgesucht, sondern die Suche wird stets mit dem aussichtsreichsten Knoten fortgesetzt.

Mit der Wahl einer Heuristik, die die Kosten zum Ziel immer unterschätzt, garantiert der A*-Algorithmus das Finden der optimalen Lösung [19, S. 37]. Er bietet sich für Anwendungsfälle an, bei denen eine gültige Heuristik problemlos gefunden werden kann, bspw. bei der Berechnung des kürzesten Weges. Die Heuristik ist dann der Luftlinienabstand, weil er die tatsächlichen Kosten immer unterschätzt. Ein großer Nachteil des Verfahrens ist, dass es in vielen Fällen nicht möglich ist, eine entsprechende Heuristik zu definieren, die die Anforderungen erfüllt. Außerdem erfordert das Speichern der Kosten aller besuchten Knoten sowie deren unterer Schranke einen hohen Speicheraufwand. Hinzu kommt, dass die untere Schranke zuvor berechnet werden muss, was, je nach Größe des Suchbaums, einen nicht unerheblichen Rechenaufwand bedarf. Im Vergleich zu Branch-and-Bound sucht der A*-Algorithmus in der Breite. Bei einem sehr hohen Verzweigungsgrad kann das einen großen Zeitraum erfordern, bis der Algorithmus in tiefere Baumschichten absteigt und letztendlich einen Blattknoten erreicht [13, S. 104].

- **Monte-Carlo Tree Search (MCTS):** Die Monte-Carlo Baumsuche hat in den letzten Jahren enorme Aufmerksamkeit erhalten, vor allem durch die Erfolge des Programms AlphaGo des Google-Unternehmens DeepMind [46]. Es war in der Lage, das traditionelle chinesische Brettspiel Go gegen den besten menschlichen Spieler zu gewinnen. Das Programm basiert auf dem MCTS-Verfahren in Verbindung mit neuronalen Netzwerken. Die Komplexität des Spiels resultiert nicht aus den Regeln, sondern an der immensen Anzahl an möglichen Spiel-Konfigurationen. Es gibt ca. 10^{170} mögliche Spiel-Zustände, was es zu einem der schwierigsten Spiele für künstliche Intelligenz macht [46]. Ein Verfahren wie die zwei bereits genannten wäre bei dieser Zahl an Möglichkeiten chancenlos. Im Gegensatz dazu hat das MCTS-Verfahren in solchen Problemstellungen überzeugende Ergebnisse geliefert.

  Entscheidend ist, dass kein domainspezifisches Wissen und somit keine Heuristik notwendig ist und der Algorithmus einen asymmetrischen Baum aufbaut. Dabei werden vielversprechende Bereiche bevorzugt untersucht und andere dafür vernachlässigt. Der Algorithmus kann außerdem zu jedem Zeitpunkt abgebrochen werden, weil er nicht die vollständige, optimale Handlungsabfolge sucht, sondern nur die optimale Handlung für den nächsten Zeitschritt [47, S. 8-9]. Der aus dieser Handlung resultierende Zustand/Knoten ist dann der

Wurzelknoten eines neuen Suchbaums. Das geschieht so lange, bis der Planungshorizont t_{end} oder das Ende des Spiels erreicht ist. Der Algorithmus expandiert den Suchbaum iterativ. Eine Iteration besteht aus den vier Schritten Selection, Expansion, Simulation und Backpropagation. Nach einer definierten Anzahl an Iterationen wird der beste Knoten/die beste Handlung ausgewählt. In Abb. 2.11 ist das Verfahren schematisch abgebildet.

In der nachfolgenden Beschreibung des Verfahrens wird nicht auf den speziellen Anwendungsfall der kooperativen Fahrmanöver eingegangen, sondern allgemein auf ein Spiel mit den Kosten 0 für eine Niederlage und 1 für einen Gewinn. Das Ziel ist also, die Kosten (in diesem Fall besser: Ertrag) zu maximieren. Jeder Knoten enthält die Information, wie oft er bereits ausgewählt („gespielt“) wurde und wie viele Spiele davon gewonnen wurden.

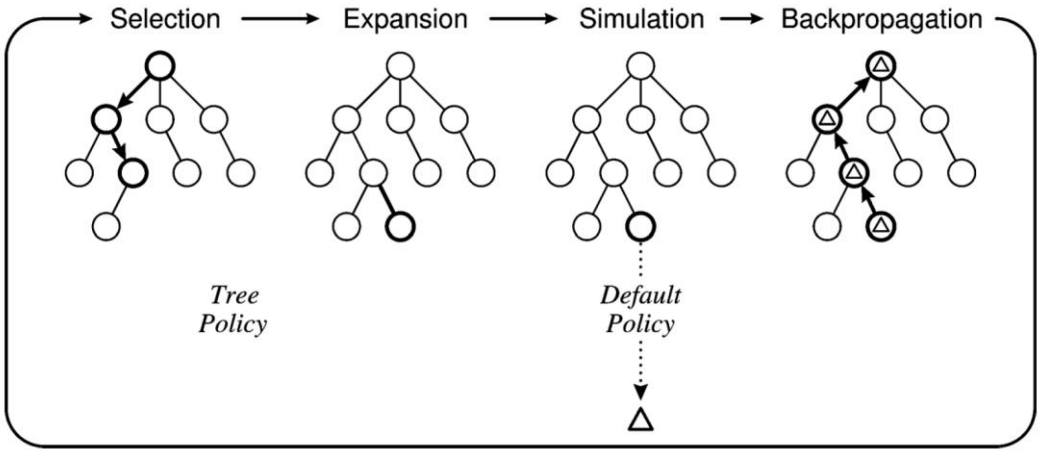


Abb. 2.11 Eine Iteration der Monte-Carlo Tree Search Verfahrens [47, S. 6]

Eine Iterationsschleife beginnt stets mit dem Selection-Prozess. Dabei wird vom Wurzelknoten aus der Suchbaum abgestiegen, bis ein Knoten erreicht wird, der entweder ein Blattknoten ist oder ein Knoten, der noch nicht expandiert wurde. Im Falle eines Blattknotens bricht der Algorithmus ab, weil dann der beste Knoten gefunden wurde. Ist der ausgewählte Knoten kein Blattknoten, so wird er expandiert [47, S. 5]. Die Logik, nach der die Auswahl der jeweiligen Knoten in der Selektion-Phase getroffen wird, heißt tree policy. Dafür hat sich eine Formel nach Gl. (2.10) etabliert, die UCT (engl.: Upper confidence bounds for trees) genannt wird [47, S. 6-7]. Dieser Wert wird für alle Kindknoten berechnet und der maximale UCT-Wert bestimmt den als Nächstes zu betrachtenden Knoten.

$$UCT = \bar{X} + 2\,C_p \sqrt{\frac{2 \ln n_{parent}}{n}} \tag{2.10}$$

Der gemittelte Gewinn pro Spiel ist $\bar{X}$ und berechnet sich aus der Anzahl an Gewinnen geteilt durch die Anzahl an Spielen. Der gemittelte Gewinn ist stets ein Wert im Intervall $[0,1]$. Der erste Term ist der Expansion-Term, weil er dem besten Knoten den maximalen UCT-Wert zuordnet. Das führt dazu, dass aussichtsreiche Bereiche des Baums bevorzugt expandiert werden. Im zweiten Term, dem Exploration-Term, ist C_p eine positive Konstante, die je nach Problemstellung angepasst werden kann. Die Anzahl der Spiele des Elternknotens

vom aktuell betrachteten Knoten ist n_{parent} und die Anzahl der Spiele des aktuellen Knotens (Kindknoten) ist n. Der Exploration-Term bewirkt, dass häufig gespielte Knoten abgewertet werden und Bereiche des Baumes, die noch nicht expandiert wurden, untersucht werden. Mit der Konstante C_p lässt sich dieses Expansion-Exploration-Dilemma je nach Bedarf anpassen [47, S. 6-7]. Wird die Konstante $C_p = 0$ gesetzt, entspricht das einer „best-first“ Suche.

In der Expansion-Phase wird der ausgewählte Knoten expandiert. Wie bereits beschrieben, werden die möglichen Aktionen angewandt und dadurch ein neuer Zustand und damit Knoten hervorgerufen. Es gibt unterschiedliche Ansätze, bei denen nur ein, mehrere oder alle Kindknoten in einem Schritt expandiert werden [47, S. 5].

In der Simulation-Phase wird der neue Kindknoten bewertet. Das geschieht mit der sogenannten default policy. Sie definiert ein Standard-Verhalten der beteiligten Spieler, das vom aktuell untersuchten Knoten aus angewendet wird. Das kann für eine bestimmte Anzahl an Zeitschritten geschehen oder bis zum Erreichen des Endzustands des Spiels [21, S. 449]. Anhand einer Kostenfunktion wird das Simulationsergebnis bewertet. Im einfachsten Fall wird ermittelt, ob das simulierte Spiel gewonnen ($Ertrag = 1$) oder verloren ($Ertrag = 0$) wurde. Im Vergleich zu den oben beschriebenen Suchalgorithmen muss beim MCTS-Verfahren keine untere Schranke oder Heuristik definiert werden.

Im letzten Schritt, der Backpropagation, wird der Suchbaum mit dem Ergebnis der Simulation aktualisiert. Für jeden bei der Selection-Phase ausgewählten Knoten wird der gespeicherte Wert für die Anzahl der Spiele um 1 erhöht. Außerdem wird der hinterlegte Wert für die Anzahl der Gewinne je nach Simulationsergebnis ebenfalls erhöht [47, S. 7-8].

Nachdem der Suchbaum aktualisiert wurde, startet eine neue Iterationsschleife und beginnt wieder mit der Auswahl des nächsten Knotens (Selection-Phase). Der Vorgang wird so lange wiederholt, bis eine vordefinierte Beschränkung erreicht wird. Das kann die Rechenzeit, der Speicherbedarf oder die Anzahl an Iterationen sein [47, S. 5]. Dann wird die optimale Handlung für den nächsten Zeitschritt ausgegeben und als Wurzelknoten für einen neuen Suchbaum eingesetzt. Ein Beispiel zur Veranschaulichung für den beschriebenen Ablauf eines Iterationsschritts mit Zahlenwerten ist in Abb. 2.12 dargestellt.

Ein entscheidender Vorteil des MCTS-Verfahrens ist, dass keine Heuristik benötigt wird. Dadurch bietet sich das Verfahren zum einen für Anwendungen an, bei denen keine gültige Heuristik gefunden werden kann. Zum anderen ist das Verfahren prädestiniert für große Suchbäume mit vielen möglichen Varianten, weil die Berechnung eines Heuristik-Wertes für jeden Knoten und damit der Rechenaufwand entfällt [47, S. 34]. Außerdem kann der Algorithmus jederzeit unterbrochen werden und ein gültiges Ergebnis liefern. Jede weitere Rechenzeit führt stets zu einer Verbesserung des Ergebnisses [47, S. 1].

Nachteilig ist, dass durch die Simulation und Bewertung eines Standard-Verhaltens nur eine Abschätzung möglich ist, wie gut der Knoten zu bewerten ist. Der tatsächliche Wert des Knotens ist unbekannt. Weil keine Heuristik vorliegt, die die minimal zu erwartende Kosten garantiert, kann nicht so zielgerichtet wie bspw. mit dem A*-Algorithmus nach der optimalen Lösung gesucht werden. In

Anwendungsfällen in denen eine Heuristik existiert, schneidet das MCTS-Verfahren deutlich schlechter ab als Suchalgorithmen, die eine Heuristik verwenden [47, S. 9]. Insbesondere bei sehr großen Suchbäumen müssen Anpassungen an den MCTS-Algorithmus vorgenommen werden, weil sonst keine signifikanten Verbesserungen zu erwarten sind [47, S. 34-35]. Generell muss der MCTS-Algorithmus immer an das vorliegende Problem adaptiert werden, um sein Potential ausspielen zu können [47, S. 1-2].

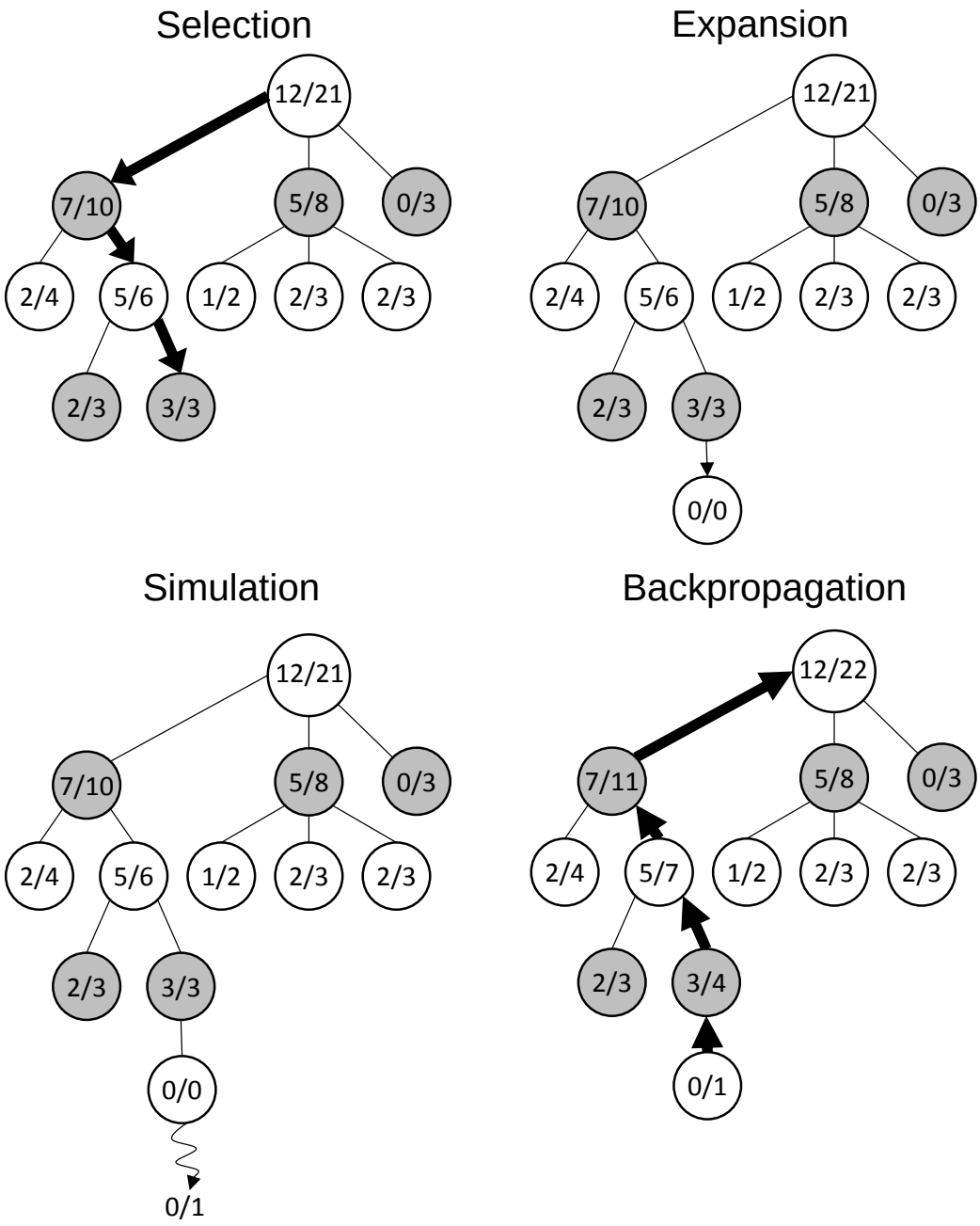


Abb. 2.12 Beispielhafter Ablauf einer Iteration des MCTS-Algorithmus nach [48]

Der Aufbau eines Suchbaums hat, unabhängig von der darauf angewandten Suchstrategie, den Vorteil, dass die angewandten Fahrzeugmodelle beliebig komplex sein können. Wie die Zustandsübergangsfunktion gestaltet wird, hat keinen negativen Einfluss auf das Berechnungsergebnis [13, S. 76]. Außerdem können die Kostenfunktion und ihre Einzelkosten beliebig erweitert oder angepasst werden, ohne dass größere Eingriffe in die

Programmstruktur vorzunehmen sind. Das ist bei dem MILP-Verfahren und der Methode der elastischen Bänder nicht ohne größeren Aufwand möglich [13, S. 175].

Nachteilig ist die Diskretisierung, die vorgenommen werden muss. Dadurch ist die Vollständigkeit des Algorithmus nicht mehr gewährleistet. Theoretisch ist eine unendlich feine Diskretisierung der Aktionen und der Zeit notwendig, um die Vollständigkeit sicherzustellen. Das ist wegen des enormen Rechenaufwands nicht möglich [13, S. 106]. Außerdem steigt die Rechenzeit exponentiell mit der Anzahl der möglichen Handlungsabfolgen und damit der Blattknoten an. Das kann je nach Problemstellung zu einem sehr großen Suchbaum führen.

3 Entwicklung und Implementierung eines Algorithmus zur kooperativen Trajektorienplanung

Dieses Kapitel beginnt mit der Bewertung der in Unterkapitel 2.3 vorgestellten Methoden zur kooperativen Trajektorienplanung. Die betrachteten Methoden werden hinsichtlich ihrer Eignung für die vorliegende Problemstellung bewertet. Anschließend wird der beste Ansatz ausgewählt. Im darauffolgenden Unterkapitel 3.2 werden die Basiskomponenten des Algorithmus, wie er in dieser Arbeit zu kooperativen Trajektorienplanung entwickelt wird, vorgestellt. Darauf aufbauend wird erläutert, wie der Algorithmus konkret in Matlab umgesetzt ist und wie die Berechnung der kooperativen Fahrmanöver erfolgt. Die besonderen Eigenschaften werden herausgearbeitet und vorgestellt.

3.1 Bewertung und Kategorisierung der ausgewählten Methoden

Bei der Vorstellung der Methoden zur kooperativen Trajektorienplanung wurden bereits die Vor- und Nachteile des jeweiligen Verfahrens herausgearbeitet. Diese werden aufgegriffen, um die Verfahren untereinander zu vergleichen und in Bezug auf die Problemstellung zu bewerten. Es wird nicht jeder Vor- und Nachteil nochmals aufgeführt. Vielmehr werden nur die Vor- und Nachteile erwähnt, die ein Verfahren besonders hervorheben oder ausschließen. Das Ziel ist, eines auszuwählen, das implementiert wird oder zumindest als Grundlage für weitere Anpassungen und Entwicklungen verwendet werden kann. Die Methoden prioritätsbasierte Planung, gemischt-ganzzahlige lineare Programmierung, elastische Bänder und Baumsuche werden auch von Frese [13] analysiert und bewertet. Da in seiner Arbeit alle Verfahren implementiert und getestet wurden, wird im Rahmen der Bewertung auf seine Ergebnisse zurückgegriffen. Zu beachten ist, dass der Schwerpunkt dabei auf der kooperativen Trajektorienplanung in Extremsituationen liegt und nicht im fließenden Verkehr. In der vorliegenden Arbeit werden die Anforderungen Sicherheit und Komfort an die zu findenden Fahrmanöver gestellt. Außerdem ist der betrachtete Planungszeitraum deutlich größer. Zur Unfallvermeidung reichen wenige Sekunden aus [13, S. 152]. Für die Kooperation im fließenden Verkehr muss vorausschauend gefahren werden, weshalb in den meisten Fällen mindestens 15 bis 20 Sekunden Planungshorizont notwendig sind.

Prioritätsbasierte Planung

Die prioritätsbasierte Planung kann für die vorliegende Aufgabenstellung verworfen werden. Durch die Priorisierung der Fahrzeuge findet keine wirkliche Kooperation statt. Vielmehr müssen die niedrig-priorisierten Fahrzeuge auf die höher-priorisierten Rücksicht nehmen. Dass

das in einigen Fällen nicht zielführend ist, wird in Abschnitt 2.3.2 gezeigt. Hinzu kommt, dass die Zuweisung von Prioritäten an Fahrzeuge meist nicht ohne erheblichen Aufwand durchgeführt werden kann [13, S. 167]. Trotzdem kann die prioritätsbasierte Planung angewendet werden, um in einer unkritischen Verkehrssituation einzelnen Fahrzeugen eine vordefinierte Bewegung vorzugeben und sie so aus dem Planungsproblem auszuschließen. Ein Fahrzeug mit fest vorgegebener Handlungsabfolge muss während der Trajektorienplanung der restlichen Fahrzeuge nur als bewegtes Hindernis berücksichtigt werden. Bei der Berechnung der möglichen Kombinationen geht das entsprechende Fahrzeug nur mit dem Faktor 1 ein. Ein Beispiel dafür ist ein Fahrzeug, das langsamer als die anderen Verkehrsteilnehmer fährt und sich auf der rechten Fahrspur befindet. In den meisten Fällen kann diesem Fahrzeug problemlos eine konstante Geschwindigkeit aufgezwungen werden, ohne die Option eines Spurwechsels zu erlauben.

Gemischt-ganzzahlige lineare Programmierung (MILP)

Die gemischt-ganzzahlige lineare Programmierung (MILP) wurde aus der Planung von Flugzeugbahnen übernommen und für die Anwendung mit Fahrzeugen angepasst. Bei der Anwendung des MILP-Verfahrens ist ein enges Abtastintervall erforderlich, weil nur zu den diskreten Zeitpunkten die Kollisionsfreiheit überprüft wird. Das ist akzeptabel für die Unfallvermeidung, weil sich diese in einem kleinen Zeitrahmen abspielt. Kooperative Handlungen in fließendem Verkehr müssen deutlich vorausschauender geplant werden, was sich in einem großen Planungshorizont ausdrückt. Hierfür ist eine enge Zeitdiskretisierung hinderlich, weil dadurch der Rechenaufwand stark ansteigt [13, S. 152]. Eine Verringerung des Abtastintervalls muss auch vorgenommen werden, wenn höhere Fahrgeschwindigkeiten auftreten [13, S. 169]. Grundsätzlich sind beim MILP-Verfahren große Schwankungen in der Rechenzeit im Vergleich zu den anderen Verfahren bei identischen Szenarien festzustellen [13, S. 168-170]. Erweiterungen beim MILP-Verfahren müssen stets die Linearitätsanforderungen erfüllen. Werden zusätzliche ganzzahlige Variablen hinzugefügt, ist ein deutlicher Anstieg der Rechenzeit zu erwarten [13, S. 175].

Methode der elastischen Bänder

Die Methode der elastischen Bänder besitzt, im Vergleich zu den anderen Verfahren, den großen Vorteil, dass die Anzahl der Fahrzeuge nur polynomiell in die Rechenzeit eingeht. Daher ist das Verfahren bei mehreren beteiligten Fahrzeugen besser [13, S. 172-173]. Nachteilig sind die impliziten Geometrie- und Dynamikmodelle sowie die Optimierung, die nur lokal stattfindet [13, S. 172]. Bei der Verschiebung der Knoten durch virtuelle Kräfte erfolgt nur eine lokale Optimierung einer initialen, globalen Trajektorie. Diese Trajektorie muss zuvor entsprechend eines Planungsalgorithmus bestimmt werden und hat einen großen Einfluss auf die Lösung [13, S. 174]. Vor allem da im vorliegenden Problem mit dem großen Planungshorizont eine vorausschauende Fahrweise notwendig ist, muss bei der Planung global nach einer Lösung gesucht werden. Insbesondere Fahrmanöver, die sich erst über einen langen Zeitraum als optimal herausstellen, werden von den elastischen Bändern nicht gefunden. Nachträgliche Erweiterungen sind bei diesem Verfahren nur mit hohem Entwicklungsaufwand verbunden, weil für weitere Nebenbedingungen neue virtuelle Kräfte definiert werden müssen [13, S. 175].

Baumsuche

Bei der Baumsuche ist die notwendige Diskretisierung problematisch [13, S. 173]. Für die Anwendung muss ein Kompromiss zwischen einer möglichst engen Abtastrate und einer möglichst geringen Anzahl an Zeitschritten bis zum Planungshorizont gefunden werden. Die zeitliche Diskretisierung kann dafür beliebig gewählt werden. Sie ist nach unten begrenzt durch

den Rechenaufwand und nach oben durch eine zu grobe Abtastrate, die keine Lösung mehr finden kann. Die optimale Abfolge von Handlungen wird ebenfalls in diskreten Werten zurückgegeben. Bei dem MILP-Verfahren und den elastischen Bändern erfolgt dies kontinuierlich. Bei der Baumsuche können Erweiterungen und Anpassungen am einfachsten vorgenommen werden. Beliebige Fahrzeugmodelle können implementiert und mit bestehenden ausgetauscht werden. Außerdem kann die Kostenfunktion um beliebige Einzelkosten erweitert werden, ohne dass die Programmstruktur geändert werden muss [13, S. 175]. Die Rechenzeit steigt bei Frese ab vier oder mehr Fahrzeugen stark an und erreicht inakzeptable Werte [13, S. 169].

Hinsichtlich des Suchalgorithmus stellt sich das MCTS-Verfahren als äußerst interessant heraus, weil es auf eine Heuristik verzichtet. Eine Heuristik-Funktion für das Problem der kooperativen Fahrmanöverplanung zu finden ist schwierig und mit hohem Aufwand verbunden. Frese [13, S. 97-101] berechnet dazu die auftretenden Zustände im Voraus. Das ist für den geringen Planungshorizont der Unfallvermeidung teilweise umsetzbar [13, S. 145]. Ohne Vorberechnung führen Verkehrsszenarien mit vier beteiligten Fahrzeugen bereits zu einem Speicherüberlauf (bei A*-Algorithmus) oder benötigen mehrere Stunden Rechenzeit (bei Branch-and-Bound) [13, S. 145]. Für den deutlich längeren Planungshorizont, wie er in dieser Arbeit üblich sein wird, kann die Vorausberechnung nicht mehr realisiert werden. Außerdem hat das MCTS-Verfahren bewiesen, dass es in Problemen mit sehr großem Suchbaum gute Ergebnisse erzielen kann. In Tabelle 3.1 werden die Eigenschaften der unterschiedlichen Verfahren gegenübergestellt. Die prioritätsbasierte Planung wird nicht mehr berücksichtigt, weil sie aus genannten Gründen nicht interessant ist.

Tabelle 3.1 Vergleichende Gegenüberstellung der drei kooperativen Bewegungsplanungsalgorithmen nach [13, S. 176]

	Baumsuche	MILP	Elastische Bänder
Handlungen	grob diskretisiert	kontinuierlich	kontinuierlich
Zeitdiskretisierung	grob	fein	fein
Optimierung	global	global	lokal
Modelle	explizit, beliebig	explizit, linear	implizit
Erweiterbarkeit	einfach	aufwändig	aufwändig
Einfluss: Anzahl Fzg. auf Rechenzeit	exponentiell	>> linear	polynomiell
Einfluss: Höhe des Planungshorizonts auf Rechenzeit	abhängig von Diskretisierung	groß	groß

Als Basis für die Entwicklung eines Algorithmus zur kooperativen Trajektorienplanung wird die Methode der Baumsuche gewählt. Vor allem aufgrund der Tatsache, dass die Planung über einen vergleichsweise großen Planungshorizont erfolgen muss, scheint die Baumsuche das vielversprechendste Verfahren zu sein. Neben der freien Gestaltung (Fahrzeugmodelle, Kostenfunktionen) des Algorithmus ist auch die gute Erweiterbarkeit ein großer Pluspunkt. Für das MCTS-Verfahren sprechen die genannten Vorteile. Außerdem wurde es für eine Anwendung dieser Art bisher nur von Lenz [21] untersucht, allerdings ohne eine explizite Kommunikation zwischen den Fahrzeugen zu berücksichtigen. Das macht die Implementierung mit der Annahme eines vollständigen Informationsaustauschs zusätzlich interessant.

Vereinfachende Annahmen können getroffen werden, weil die Fahrmanöverplanung im fließenden Verkehr deutlich unkritischer ist als die für das Vermeiden von Unfällen. Ein Beispiel dafür ist die Definition einer Spurwechselaktion, anstatt verschiedene Lenkwinkelwerte zu diskretisieren [21, S. 450]. Damit wird die Anzahl möglicher Varianten gesenkt und somit auch der Rechenaufwand. Diese getroffenen Annahmen werden im nächsten Unterkapitel bei der Vorstellung der Kernkomponenten des Algorithmus vorgestellt und erklärt.

3.2 Kernelemente des Algorithmus zur kooperativen Trajektorienplanung

In diesem Unterkapitel wird zuerst auf die Definition des Fahrbahnlayouts und der initialen Verkehrssituation eingegangen. Ferner werden die dabei getroffenen Annahmen erläutert. Anschließend wird das verwendete Fahrzeug- und Fahrermodell, die dafür notwendigen mathematischen Gleichungen und dabei getroffene Vereinfachungen vorgestellt. In Abschnitt 3.2.4 wird das Baumsuchverfahren erarbeitet, das im Rahmen dieser Arbeit Verwendung findet. Dieses basiert auf der Methode, die aus der Gegenüberstellung und Auswahl in Unterkapitel 3.1 hervorgeht, der Monte-Carlo Baumsuche. Abschließend werden Kostenfunktionen eingeführt, mithilfe derer das beste kooperative Fahrmanöver bestimmt wird.

3.2.1 Definition des Fahrbahnlayouts und des Verkehrsszenarios

Das Fahrbahnlayout S_{FBL} stellt das statische Umfeld der Fahrmanöverplanung dar und setzt sich aus der Fahrbahn und den statischen Hindernissen zusammen. Die Fahrbahn besteht aus mindestens einer Fahrspur und unter Umständen einer Beschleunigungsspur. Diese gibt den auffahrenden Fahrzeugen die Möglichkeit, auf die Geschwindigkeit des umgebenden Verkehrs zu beschleunigen und dann mit möglichst geringer Differenzgeschwindigkeit auf die normale Fahrbahn zu wechseln. Teil des Fahrbahnlayouts sind auch statische Hindernisse, die eine oder mehrere Fahrspuren über eine gewisse Länge blockieren. Diese blockierten Bereiche sind für Fahrzeuge nicht zugänglich. Zusammen mit den initialen Fahrzeugzuständen bildet das Fahrbahnlayout das Verkehrsszenario. Eine beispielhafte Darstellung eines Verkehrsszenarios mit einem beteiligten Fahrzeug ist in Abb. 3.1 zu sehen.

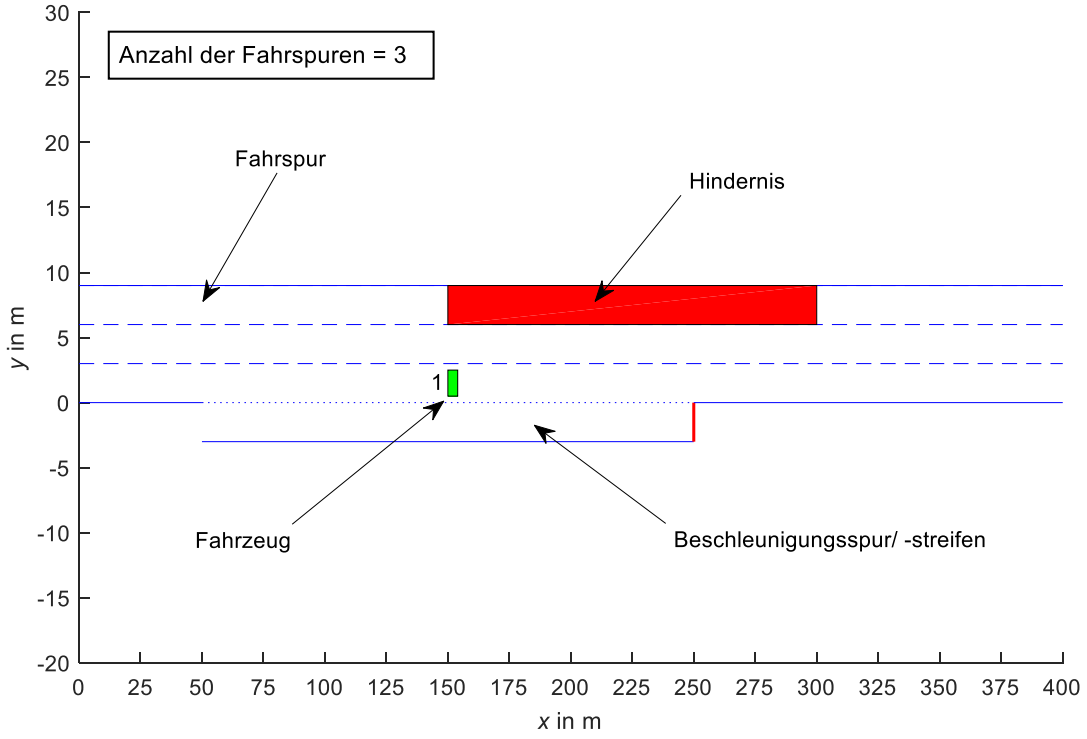


Abb. 3.1 Verkehrsszenario bestehend aus Fahrbahnlayout und einem Fahrzeug

Bei der Berechnung kooperativer Trajektorien werden gekrümmte Fahrbahnverläufe und solche mit Steigung nicht explizit betrachtet. Weil bei Fahrmanövern im fließenden Verkehr

i.d.R. nur geringe Längs- und Querbeschleunigungswerte auftreten, wird die Problemstellung auf einen geraden und ebenen Fahrbahnverlauf beschränkt. Die berechneten Ergebnisse können mithilfe einer Transformation auf Fahrbahnverläufe mit Krümmung angewandt werden. Außerdem werden Situationen mit Verzögerungsspur nicht betrachtet. Aufgrund dieser Annahmen kann die Fahrbahn vollständig in einem globalen x-y-Koordinatensystem dargestellt werden, wie es in Abb. 3.1 zu finden ist. Der Nulldurchgang auf der x-Achse liegt dabei am Beginn des betrachteten Fahrbahnabschnitts. Der Nulldurchgang auf der y-Achse befindet sich auf Höhe der rechten Fahrbahnmarkierung der rechten Fahrspur. Der Beschleunigungsstreifen besitzt somit einen negativen y-Wert und wird nicht als Fahrspur gezählt. Deshalb ist die Anzahl der Fahrspuren in diesem Beispiel $i_{FSP} = 3$. Mit dem gegebenen Koordinatensystem kann das Fahrbahnlayout durch Parameter beschrieben werden, wie sie in Abb. 3.2 dargestellt sind.

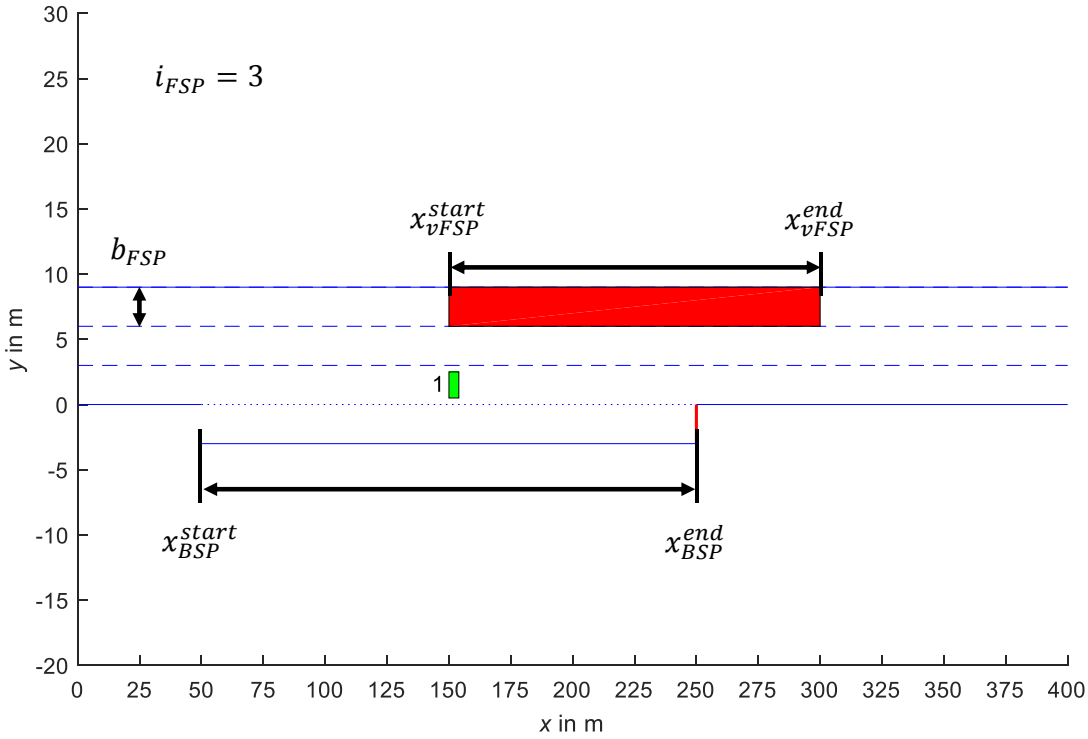


Abb. 3.2 Parameter zur Beschreibung des Fahrbahnlayouts

Bei der Anzahl der Fahrspuren i_{FSP} zählen nur die Fahrspuren der Haupt-Fahrbahn und nicht der Beschleunigungsstreifen. Die Breite jeder Fahrspur wird durch b_{FSP} gegeben und ist für alle Fahrspuren identisch. Die Beschleunigungsspur wird durch ihre Startposition x_{BSP}^{start} und ihre Endposition x_{BSP}^{end} festgelegt. Gleiches gilt für das Hindernis und dessen Startposition x_{vFSP}^{start} sowie dessen Endposition x_{vFSP}^{end}. Da die Hindernisse in den meisten Fällen Fahrspuren über eine längere Distanz blockieren, wie bspw. Baustellen oder gesperrte Fahrspuren, werden sie auch verengte Fahrspuren genannt (x_{vFSP}).

Die initialen Fahrzeugzustände $\boldsymbol{X}_{Veh}(t_0) = (\boldsymbol{x}_1^{start}, \boldsymbol{x}_2^{start}, \dots, \boldsymbol{x}_m^{start})$ legen die Anfangsposition der beteiligten Fahrzeuge fest. Wie mit Abb. 2.4 und Gl. (2.1) eingeführt, bestehen die Fahrzeugzustände jeweils aus der x- und y-Position, der Orientierung bezogen auf die x-Achse und der initialen Geschwindigkeit. Damit können die Fahrzeuge im Fahrbahnlayout eindeutig platziert werden und bilden zusammen das Verkehrsszenario für die Manöverplanung. Das Fahrbahnlayout ist zu allen Zeitpunkten gleich dem initialen Zustand. Die Fahrzeuge bewegen sich in dem gegebenen statischen Umfeld.

3.2.2 Fahrzeug- und Fahrermodell

Das Fahrzeugmodell beschreibt, wie sich der Zustand des Fahrzeugs über die Zeit verändert und welche Auswirkungen bestimmte Aktionen darauf haben. Für den Aufbau eines Suchbaums müssen diskrete Zeitschritte vorgegeben werden. Aus diesem Grund muss auch das Fahrzeugmodell diskretisiert werden. Das geschieht in Gl. (2.5) für die Position und in Gl. (2.6) für die Geschwindigkeit. Analog dazu kann eine Gleichung für die Berechnung der Fahrzeugorientierung bestimmt werden (Gl. (2.7)) [13, S. 78]. Die Fahrzeug-Stellgrößen – Beschleunigung und Lenkwinkel – werden jeweils über einen Zeitschritt Δt als konstant angenommen. Mithilfe der Zustandsübergangsfunktion (Gl. (2.4)) wird der neue Zustand berechnet. Die Planung kooperativer Fahrmanöver im fließenden Verkehr ist deutlich unkritischer hinsichtlich fahrdynamischer Einschränkungen als die Planung von Fahrmanövern zur Unfallvermeidung. Aus diesem Grund können mehrere Vereinfachungen für das in dieser Arbeit verwendete Fahrzeugmodell getroffen werden.

Erstens wird die Orientierungsänderung der Fahrzeuge während des Fahrmanövers vernachlässigt. Die Fahrzeuglängsachse zeigt im initialen Zustand stets in die Richtung der x-Achse des globalen Koordinatensystems. Diese Orientierung wird bis zum Ende der Berechnung konstant gehalten. Das ist zulässig, weil der Straßenverlauf zum einen keine Krümmung aufweist. Außer bei einem Spurwechsel bewegen sich die Fahrzeuge daher immer ausschließlich in x-Richtung. Zum anderen kann bei einem Spurwechsel in fließendem Verkehr und bei höheren Geschwindigkeiten, wie sie in dieser Arbeit vorliegen, die Orientierungsänderung vernachlässigt werden. Bei einem Spurwechsel dauert das Überschreiten der Fahrspurmarkierung ca. zwei Sekunden [49, S. 50-51]. Das entspricht nicht der gesamten Dauer des Spurwechsel-Manövers, sondern nur dem Zeitintervall, in der das Fahrzeug beide Fahrspuren belegt. Bei Fahrversuchen wird beobachtet, dass die Spurwechseldauer unabhängig von der Fahrgeschwindigkeit ist und somit als annähernd konstant angenommen werden kann [49, S. 84]. Der Versatz in y-Richtung entspricht der Fahrspurbreite und ist i.d.R. zwischen drei und vier Meter groß. Bei einer Geschwindigkeit von 80 km/h (ca. 22 m/s) werden in der Zeit des Spurwechsels von zwei Sekunden ungefähr 44 m in x-Richtung zurückgelegt. Mit der Tangens-Winkelbeziehung lässt sich daraus ein Fahrzeug-Orientierungswinkel während des Spurwechsels von ca. 5° berechnen. Da im Normalfall höhere Geschwindigkeiten auftreten, wird der Winkel noch geringer ausfallen. Aus diesem Grund werden nicht unterschiedliche Lenkwinkel als Aktionen angeboten, sondern lediglich eine Aktion für einen Spurwechsel nach links und eine Aktion für einen Spurwechsel nach rechts. Dieses Vorgehen wird auch von Lenz [21, S. 450] für Spurwechsel angewendet. Die Fahrzeuglängsachse ist dabei parallel zur x-Achse des globalen Koordinatensystems. Das Fahrzeug hat während des Spurwechsels eine Geschwindigkeit in y-Richtung und bewegt sich senkrecht zur Fahrzeuglängsachse. Nicht-holonome Zwangsbedingungen spielen bei diesen geringen Orientierungsänderungen keine Rolle. Auch die Querbeschleunigung ist so gering, dass sie nicht explizit aus der Orientierungsänderung berechnet und überprüft werden muss.

Die zweite Vereinfachung zielt auf diese geringen Beschleunigungswerte ab. Weil bei den Fahrmanövern nur geringe Längs- und Querbeschleunigungen auftreten, muss die Einhaltung der fahrdynamischen Grenzwerte nicht explizit, bspw. durch den Kamm'schen Kreis, berücksichtigt werden. Der maximale Querbeschleunigungswert bei Spurwechselmanövern liegt laut Freyer [49, S. 82-83] im Durchschnitt bei 0,75 m/s². Eine Abhängigkeit zur Fahrgeschwindigkeit kann nicht festgestellt werden. Die Einhaltung der fahrdynamischen Grenzwerte wird dadurch garantiert, dass die Aktionen und Beschleunigungswerte, die die Fahrzeuge ausführen können, weit unter diesen Grenzwerten liegen. Für Verzögerungen kann

ein Wert von -1,5 m/s² beobachtet werden, ab dem die Aktion von Insassen als unkomfortabel und sogar gefährlich eingestuft wird [49, S. 52-53]. Für komfortable und gewünschte Beschleunigungswerte wird häufig der Bereich zwischen 0,7 und 1,0 m/s² angegeben [50, S. 29, 51, S. 366-367]. Mit diesen geringen Werten für Längs- und Querbeschleunigung kann die vereinfachende Annahme getroffen werden, dass die geplanten Manöver aus fahrdynamischen Gesichtspunkten problemlos durchführbar sind, ohne dies explizit zu überprüfen.

Drittens wird vorausgesetzt, dass sich die Fahrzeuge immer mittig ihrer jeweiligen Fahrspur befinden. Außer bei Spurwechselmanövern folgen die Fahrzeuge exakt dem Fahrbahnverlauf. Das Spurhalten der Fahrzeuge muss daher nicht explizit sichergestellt werden, weil sich die Fahrzeuge nur mittig der Fahrspuren bewegen können. Die Fahrzeuge belegen dabei stets die komplette Fahrspur. Bei einem Spurwechsel werden beide Fahrspuren, die Ausgangsfahrspur und die Zielfahrspur, belegt. Wenn die jeweilige Fahrspur immer komplett blockiert ist, entfällt die Erstellung eines komplexen Geometriemodells von Fahrzeugen und Hindernissen.

Mit diesen Vereinfachungen kann der Zustand eines Fahrzeuges anstatt mit bisher vier Parametern mit den drei Parametern x-Position, y-Position und aktuelle Geschwindigkeit ausgedrückt werden. Die jeweiligen Spurwechsel-Aktionen werden so modelliert, dass die Zeitdauer, in der beide Fahrspuren belegt sind, konstant ist. Damit lässt sich aus der Fahrspurbreite eine Geschwindigkeit in y-Richtung berechnen. Die auftretende Beschleunigung wird vernachlässigt und es wird angenommen, dass die y-Geschwindigkeit sofort bei Beginn des Spurwechsels vorliegt. Das vollständige Spurwechselmanöver dauert länger als zwei Sekunden. Freyer [49, S. 85] nennt einen Wert von durchschnittlich 7,4 s. Die meiste Zeit entfällt auf das langsame Aus- und Einscheren aus/in die jeweilige Fahrspur. In diesem Zeitraum ist nur eine Fahrspur blockiert. Interessant ist jedoch nur der Zeitraum, in dem beide Fahrspuren blockiert sind. Aus diesem Grund werden in der vorliegenden Arbeit die Spurwechselmanöver auf den Zeitraum verkürzt, in dem ausschließlich beide Fahrspuren blockiert sind. Der entscheidende Wert bei einem Spurwechsel ist damit die y-Geschwindigkeit. Ist dieser Wert negativ, bedeutet das einen Spurwechsel nach rechts. Ein positiver Geschwindigkeitswert stellt einen Spurwechsel nach links dar.

Zusätzlich werden Aktionen definiert, die ein konstantes Beschleunigen und Verzögern ermöglichen. Für einen Pkw wird dafür die Beschleunigung $a_{>0} = 2{,}5$ m/s² und die Verzögerung $a_{<0} = -1{,}5$ m/s² definiert. Die Möglichkeit, mit konstanter Geschwindigkeit zu fahren, bietet der Wert $a_{=0} = 0$ m/s². Nicht berücksichtigt ist bisher, dass das Verhalten im fließenden Verkehr stark vom jeweils vorausfahrenden Fahrzeug abhängt. Dafür sind situationsabhängige Beschleunigungs- und Verzögerungswerte notwendig, die mithilfe eines Fahrermodells, dem sogenannten IDM (engl.: Intelligent-Driver Model), berechnet werden. Das IDM wird zusätzlich für die Simulation-Phase des MCTS-Verfahrens als default policy verwendet, denn dadurch kann das „Standard-Verhalten“ eines Fahrers simuliert werden. Mit dem Intelligent-Driver Model lässt sich ein kontinuierlicher Wert für die Beschleunigung a_{IDM} berechnen. Das Ergebnis hängt ab von der aktuellen Geschwindigkeit v, dem Abstand s_{gap} und der Geschwindigkeitsdifferenz Δv jeweils bezogen auf das Fahrzeug/Hindernis voraus. Die Formel zur Berechnung des Beschleunigungswertes a_{IDM} entspricht Gl. (3.1).

$$a_{IDM} = a_{wunsch}\left[1 - \left(\frac{v}{v_{wunsch}}\right)^{\delta_{IDM}} - \left(\frac{s^*_{gap}(v, \Delta v)}{s_{gap}}\right)^2\right] \tag{3.1}$$

Je größer der Beschleunigungsexponent δ_{IDM} ist, desto länger wird mit einem hohen Wert auf Wunschgeschwindigkeit beschleunigt. Der Standardwert für den Beschleunigungsexponenten ist $\delta_{IDM} = 4$ [52, S. 163]. Für das Fahrermodell müssen im Voraus die komfortable/gewünschte Beschleunigung a_{wunsch} $(= a_{>0})$, die Wunschgeschwindigkeit v_{wunsch} und der gewünschte Sicherheitsabstand zum vorausfahrenden Fahrzeug s^*_{gap} bestimmt werden. Dieser Sicherheitsabstand hängt ab von der aktuellen Geschwindigkeit v sowie der Geschwindigkeitsdifferenz Δv zum vorausfahrenden Fahrzeug. Die Geschwindigkeitsdifferenz ist positiv, wenn das vorausfahrende Fahrzeug langsamer fährt als das Ego-Fahrzeug und negativ im anderen Fall. Außerdem werden vordefinierte Parameter benötigt. Der gewünschte Sicherheitsabstand wird durch Gl. (3.2) bestimmt.

$$s^*_{gap}(v, \Delta v) = s_0 + s_1 \sqrt{\frac{v}{v_{wunsch}}} + v\, T_{gap} + \frac{v\, \Delta v}{2\sqrt{a_{wunsch}\, b_{wunsch}}} \tag{3.2}$$

Für die Berechnung des gewünschten Abstands zum vorausfahrenden Fahrzeug s^*_{gap} werden folgende Parameter benötigt: der Abstand zum vorausfahrenden Fahrzeug bei Stillstand s_0, ein nichtlinearer Abstandsparameter s_1, der gewünschte zeitliche Abstand zum vorausfahrenden Fahrzeug T_{gap}, der oben genannte Wert für die gewünschte Beschleunigung a_{wunsch} und zusätzlich ein Wert für die gewünschte/komfortable Verzögerung b_{wunsch} $(= a_{<0})$. Der Parameter s_1 wird in den meisten Fällen zu Null gesetzt, weil er keinen anschaulichen Beitrag zum IDM hat [50, S. 5-6]. Mit dem Parameter b_{wunsch} wird nur ein gewünschter Wert für die Verzögerung gegeben, der allerdings keinerlei begrenzende Eigenschaft auf den resultierenden IDM-Wert a_{IDM} hat. Während die gewünschte Beschleunigung a_{wunsch} gleichzeitig die maximal mögliche Beschleunigung darstellt, kann die Verzögerung bei einer starken Unterschreitung des Sicherheitsabstandes unendlich groß werden. Aus diesem Grund muss die maximale Verzögerung explizit vorgegeben und deren Einhaltung überwacht werden. Zusätzlich kann der Fall auftreten, dass die Geschwindigkeitsdifferenz so weit im negativen Bereich liegt, dass daraus ein negativer gewünschter Abstand resultiert. Weil dieser Wert quadratisch in die IDM-Beschleunigung eingeht, kann ein hoher negativer Abstandswunsch eine unlogische Verzögerung hervorrufen. Aus diesem Grund wird in der Literatur die Summe der letzten beiden Terme von s^*_{gap} gleich Null gesetzt, wenn sie negativ ausfällt [52, S. 162].

Durch das IDM können zum einen unterschiedliche Fahrertypen simuliert werden. Sportliche Fahrer haben im Vergleich zu komfortablen Fahrern bspw. eine deutlich höhere Wunschgeschwindigkeit und akzeptieren stärkere Beschleunigungs- und Verzögerungswerte. Zum anderen können durch das IDM verschiedene Fahrzeugtypen definiert werden, bspw. Pkw und Lkw, die jeweils unterschiedliche Ausprägungen der IDM-Parameter besitzen. In dieser Arbeit wird nicht zwischen Fahrertypen unterschieden, dafür werden IDM-Parameter jeweils für Pkw und Lkw definiert. In Tabelle 3.2 wird eine Übersicht über die einzelnen Werte gegeben.

Tabelle 3.2 IDM-Parameter für Pkw und Lkw

Formelzeichen	Einheit	Beschreibung	Pkw	Lkw
a_{wunsch}	m/s²	Wunschbeschleunigung	2,5	1,5
δ_{IDM}	-	Beschleunigungsexponent	4	4
s_0	m	Minimalabstand (linear)	2	2
s_1	m	Minimalabstand (nicht-lin.)	0	0
T_{gap}	s	Folgezeit	2	2
b_{wunsch}	m/s²	Wunschverzögerung	-1,5	-1,5

In der Literatur werden die IDM-Parameter mit ähnlichem Wertebereich von Treiber [50, S. 29, 52, S. 163] und Helbing [51, S. 367] verwendet. Lediglich die gewünschte Beschleunigung ist für beide Fahrzeugtypen etwas höher angesetzt im Vergleich zu den Werten aus der Literatur. Beide verzichten zudem auf den nichtlinearen Abstandsterm, indem sie $s_1 = 0$ wählen. Die Wunschgeschwindigkeit v_{wunsch} kann für jedes Fahrzeug unterschiedlich definiert werden, solange es unterhalb der Maximalgeschwindigkeit v_{max} des jeweiligen Fahrzeugtyps liegt. Die maximal erreichbare Verzögerung b_{max} wird ebenfalls für jeden Fahrzeugtyp separat festgelegt. In Tabelle 3.3 sind die für die Berechnung angenommenen Werte gezeigt. Eine Überschreitung der maximalen Geschwindigkeit und eine Unterschreitung des minimalen Verzögerungswertes wird bei der Verwendung des IDM stets überprüft.

Tabelle 3.3 Grenzwerte für die maximale Geschwindigkeit und Verzögerung von Pkw und Lkw

Formelzeichen	Einheit	Pkw	Lkw
v_{max}	m/s	50 (= 180 km/h)	27,78 (= 100 km/h)
b_{max}	m/s²	-7	-7

Mit den festgelegten Aktionen kann für alle Fahrzeuge die Menge an standardmäßig ausführbaren Aktionen $\boldsymbol{a}_{act,def}$ gegeben werden (Gl. (3.3)).

$$\boldsymbol{a}_{act,def} = \begin{pmatrix} a_{=0} \\ a_{>0} \\ a_{<0} \\ a_{IDM} \\ SPW_{links} \\ SPW_{rechts} \end{pmatrix} \tag{3.3}$$

Die Anzahl an Aktionen für jeden Zustand ist unabhängig vom Fahrzeugtyp und beträgt maximal $k_{def} = 6$. Je nach Fahrsituation werden einzelne Aktionen ausgeschlossen.

3.2.3 Kostenfunktionen zur Bewertung von Fahrmanövern

Bei der Vorstellung der Baumsuche in Abschnitt 2.3.5 wird bereits erwähnt, dass Bewertungskriterien notwendig sind, um die einzelnen Knoten untereinander vergleichen zu können. Mit den definierten Kostenfunktionen soll die optimale Handlungsabfolge ermittelt werden, die jeweils einen Systemzustand zu jedem Zeitschritt beschreibt. Diese Kostenfunktionen decken dabei Aspekte des Planungsproblems ab, die gewissen Einschränkungen unterliegen oder einen bevorzugten Wertebereich besitzen. Eine Bewertung muss rein anhand der Zustandsgrößen Position und Geschwindigkeit erfolgen können. Der Kostenwert muss stets positiv sein. In dieser Arbeit werden folgende Kosten berücksichtigt:

- **Kosten für Spurwechsel:** Führt ein Fahrzeug einen Spurwechsel durch, so ist dies mit den Kosten J_{SPW} verbunden. Bei einem Spurwechsel ist $J_{SPW} = 1$. Folgt das Fahrzeug der aktuellen Fahrspur, gilt $J_{SPW} = 0$.
- **Kosten für Abweichung von der Wunschgeschwindigkeit:** Jedes Fahrzeug besitzt eine individuelle Wunschgeschwindigkeit. Weicht die aktuelle Geschwindigkeit von der Wunschgeschwindigkeit ab, verursacht das die Kosten J_v. Diese Kosten können durch die Verwendung von Gl. (3.4) berechnet werden.

$$J_v = |v - v_{wunsch}| \tag{3.4}$$

Eine weitere Möglichkeit ist, die Differenz aus aktueller und gewünschter Geschwindigkeit zu quadrieren. Weiterhin kann eine asymmetrische Kostenfunktion vorgegeben werden, die das Überschreiten der Wunschgeschwindigkeit bei gleicher Geschwindigkeitsdifferenz stärker bestraft als ein Unterschreiten. Dazu wird angenommen, dass eine Überschreitung der Wunschgeschwindigkeit von Fahrzeuginsassen deutlich unangenehmer wahrgenommen wird als eine Unterschreitung.

- **Kosten für die Beschleunigung:** Die Kosten für die auftretende Fahrzeugbeschleunigung J_a können wie bei der Berechnung der Kosten für die Abweichung von der Wunschgeschwindigkeit mit einem Betragswert oder einem quadratischen Term (Gl. (3.5)) bestimmt werden. Das ist notwendig, weil dieser Wert positiv und negativ sein kann.

$$J_a = a^2 \tag{3.5}$$

  Auch bei der Bestimmung der Kosten für die Beschleunigung ist eine asymmetrische Kostenfunktion denkbar, die bspw. Verzögerung stärker bestraft als Beschleunigung.

- **Kosten für Unterschreitung des Sicherheitsabstandes:** Die Kosten bei Unterschreitung des Sicherheitsabstandes $J_{safedis}$ treten nur dann auf, wenn das Fahrzeug den berechneten Abstand zum vorausfahrenden Fahrzeug unterschreitet. Die Berechnung des einzuhaltenden Sicherheitsabstands $s_{safedis}$ orientiert sich an Gl. (3.2), mit der bereits der gewünschte Sicherheitsabstand für das IDM bestimmt wurde. Der Unterschied dabei ist, dass die gewünschte Beschleunigung a_{wunsch} und Verzögerung b_{wunsch} durch den maximal zu erreichenden Verzögerungswert b_{max} ausgetauscht werden. Der Sicherheitsabstand $s_{safedis}$ ist folglich der Wert, der in einer kritischen Situation ausreichend groß ist, um zum Stillstand zu kommen. Er wird mit Gl. (3.6) bestimmt.

$$s_{safedis} = s_0 + v\,T_{gap} + \frac{v\,\Delta v}{b_{max}} \tag{3.6}$$

  Mit diesem Abstandswert werden die Kosten für eine Unterschreitung des Sicherheitsabstands durch Gl. (3.7) bestimmt. Ist der aktuelle Abstand s_{gap} größer als der geforderte Sicherheitsabstand resultieren daraus keine Kosten.

$$J_{safedis} = \max\left(0, -\left(\frac{s_{gap}}{s_{safedis}}\right) + 1\right) \tag{3.7}$$

  Da der Sicherheitsabstand in seiner Größe situationsabhängig ist und deswegen stark schwankt, werden die Kosten normiert, um die Vergleichbarkeit sicherzustellen. Die Kosten bewegen sich im Intervall $[0, 1]$. Die Kostenfunktion kann zusätzlich quadriert werden, um ein geringes Unterschreiten des Sicherheitsabstands zu tolerieren und im weiteren Verlauf eine stärkere Steigung zu erlangen.

- **Kosten für Nicht-Einhalten des Rechtsfahrgebots:** Mit dem Kostenterm für das Nicht-Einhalten des Rechtsfahrgebotes $J_{FSPrechts}$ wird das Fahren auf einer Fahrspur, außer der rechten, mit einem Kostenwert bestraft. Für die davon linksgelegene Fahrspur gilt $J_{FSPrechts} = 1$, für die zweite, weiter links liegende Fahrspur $J_{FSPrechts} = 2$ usw. Der Kostenwert der Beschleunigungsspur ist ebenfalls $J_{FSPrechts} = 1$, wobei eine stärkere Gewichtung des Fahrens auf dieser Fahrspur denkbar ist.

- **Kosten für Kollisionen:** Kosten für Kollisionen J_{Koll} treten auf, wenn ein Fahrzeug mit anderen Fahrzeugen, Hindernissen oder mit dem Ende der Beschleunigungsspur

kollidiert. Eine Möglichkeit ist, die Kosten bei einer Kollision so hoch zu setzen, dass sie unter keinen Umständen durch andere Kostenterme überboten werden können. Eine weitere Variante ist, den Kosten bei einer Kollision keinen konkreten Zahlenwert zu zuweisen, sondern dem Knoten einen Hinweis anzuhängen, dass dieser ungültig/nicht zulässig ist. Mit beiden Varianten wird erreicht, dass Aktionen und die Knoten, die zu einer Kollision führen, nicht mehr weiter expandiert werden. Der tatsächliche Wert des Kostenterms ist für Kollisionen $J_{Koll} = 1$ und andernfalls $J_{Koll} = 0$.

Weil einige Kostenterme nur Werte zwischen Null und Eins annehmen können, muss über Gewichtungsfaktoren eine Hierarchie erzeugt werden. Das ist erforderlich, da bspw. eine Kollision unter allen Umständen ausgeschlossen werden muss und nicht gegenüber einer hohen Abweichung von der Wunschgeschwindigkeit bevorzugt werden kann. Die absoluten Kosten besitzen keine Aussagekraft, deshalb müssen sie nur relativ zueinander vergleichbar sein. Aus diesen Gründen werden Gewichtungsfaktoren $w_{FzgTyp,(\cdot)}$ festgelegt. Sie unterscheiden sich je nach Fahrzeugtyp zwischen Pkw und Lkw. Der Notation von Lenz [21, S. 450] folgend, lassen sich die Kosten für ein Fahrzeug mit Gl. (3.8) berechnen.

$$J_{ges,i} = \sum w_{FzgTyp,(\cdot)}\ J_{i,(\cdot)} \tag{3.8}$$

Die gesamten Kosten J_{ges} bildet die Summe der Einzelkosten jedes Fahrzeugs $J_{ges,i}$. Unter Berücksichtigung der maximalen Werte der jeweiligen Kostenterme wurden die Gewichtungsfaktoren ausgewählt. Sie sind in Tabelle 3.4 aufgeführt.

Tabelle 3.4 Gewichtungsfaktoren für die Kostenfunktionen nach jeweiligem Fahrzeugtyp unterteilt

Kostenterm	Intervall	Pkw $w_{Pkw,(\cdot)}$	Lkw $w_{Lkw,(\cdot)}$
J_{SPW}	$[0,1]$	15	15
J_v	$[0, v_{max}]$	15	15
J_a	$[0, b_{max}^2]$	1	2
$J_{safedis}$	$[0, 1]$	15000	15000
$J_{FSPrechts}$	$[0, i_{FSP} - 1]$	20	30
J_{Koll}	$[0, 1]$	50000/nicht zulässig	50000/nicht zulässig

Die Kosten für einen Spurwechsel liegen unter denen, die ein Abweichen von der rechten Fahrspur hervorruft. Dadurch wird sichergestellt, dass es immer attraktiver ist, nach rechts zu wechseln anstatt auf der aktuellen Fahrspur weiter zu fahren. Außerdem sind die Kosten für das Nicht-Einhalten des Rechtsfahrgebotes von Lkw höher als die von Pkw, weil Lkw im Vergleich zu Pkw tendenziell nur selten auf die linken Fahrspuren wechseln. Lkw beschleunigen und verzögern weniger stark als Pkw, was in den entsprechenden Gewichtungsfaktoren zum Tragen kommt. Eine Abweichung von der Wunschgeschwindigkeit wird gleich gewichtet. Denkbar ist eine höhere Gewichtung auf Seiten der Lkw, die durch die hohen Anforderungen an Kraftstoffeffizienz begründet werden kann. Diese wird durch jeden Beschleunigungs- und Abbremsvorgang verschlechtert. Die Kosten für eine Kollision müssen entweder ausreichend hoch sein oder die Aktion für ungültig erklären. Beides wird in der Implementierung untersucht. Grundsätzlich gilt, dass die Gewichtungsfaktoren anhand von einfachen Überlegungen definiert wurden und im Laufe der Berechnung angepasst werden. Je nachdem, welches Fahrverhalten von den beiden Fahrzeugtypen verlangt wird, können die Gewichtungsfaktoren ohne großen Aufwand angepasst werden.

3.2.4 Baumsuchverfahren

Das MCTS-Verfahren in seiner ursprünglichen Form besitzt bei der Anwendung auf das Problem der Planung kooperativer Fahrmanöver einige Schwächen und Nachteile. Aus diesem Grund müssen zum einen Anpassungen vorgenommen und zum anderen einige Abschnitte des Verfahrens neu entwickelt werden. Der entscheidende Unterschied zwischen der Planung kooperativer Fahrmanöver und der sonst üblichen Anwendung zur Berechnung von Spielzügen ist, dass sich bei Ersterem alle beteiligten Fahrzeuge stets vorhersehbar verhalten. Bei Spiel-Anwendungen ist hingegen nie mit absoluter Sicherheit bekannt, welchen Spielzug der Gegenspieler durchführen wird. Deshalb haben bei der Expansion eines Knotens die besten und die schlechtesten Kindknoten einen Einfluss. Dadurch wird berücksichtigt, dass der Knoten neben den guten Spielzügen auch zu welchen führen kann, die sehr schlecht im Sinne des Spielgewinnes sind. In Abb. 2.12 werden daher für die jeweiligen Elternknoten die Anzahl der Gewinne aller Kindknoten summiert. Damit werden alle möglichen Spielzüge des Gegners berücksichtigt. Bei der kooperativen Manöverplanung ist der Zustand aller Fahrzeuge bekannt und die Einhaltung der berechneten Aktionen ist zumindest während der Berechnung garantiert. Aus diesem Grund müssen schlechte Kindknoten nicht weiter betrachtet werden, weil sichergestellt ist, dass diese nicht auftreten können.

Anschaulich ist das Beispiel von zwei parallel fahrenden Fahrzeugen auf zwei nebeneinanderliegenden Fahrspuren mit identischer und konstanter Geschwindigkeit. Eines der beiden Fahrzeuge ist um zehn Meter nach hinten versetzt und beide haben ihre Wunschgeschwindigkeit bereits erreicht. Mit diesen Angaben ist die plausibelste Aktion, dass beide Fahrzeuge ihre Geschwindigkeit konstant halten. Das Rechtsfahrgebot wird für dieses Beispiel vernachlässigt. Neben unterschiedlich starken Beschleunigungswerten steht beiden Fahrzeugen ein Spurwechsel auf die Fahrspur des jeweils anderen Fahrzeugs zur Option. Bei der Expansion des aktuellen Zustandes werden alle möglichen Aktionen angewandt und die sich ergebenden Kindknoten bewertet. Das Fahren mit konstanter Geschwindigkeit erzeugt die geringsten Kosten. Eine Spurwechsel-Aktion ruft dagegen hohe Kosten hervor, weil sich dann beide Fahrzeuge auf der gleichen Fahrspur befinden und der Abstand nur zehn Meter beträgt. Der Sicherheitsabstand wird deutlich unterschritten. Werden anschließend alle Kosten der resultierenden Kindknoten summiert und während der Backpropagation-Phase dem Elternknoten zugewiesen, besitzt dieser hohe Kosten. Die eigentlich beste Aktion der konstanten Geschwindigkeit wird stark abgewertet. Über mehrere Iterationsschritte hinweg, führt das dazu, dass die Fahrzeuge das vermeintlich sicherste Fahrmanöver wählen. Das ist in den meisten Fällen ein großer Abstand zueinander.

Bei der Anwendung in Spielen ist gewünscht, dass für das eigene Spielziel schlechte Züge berücksichtigt werden und nur der sicherste und gleichzeitig erfolgversprechendste Zug gewählt wird. Bei der Planung kooperativer Fahrmanöver, bei der das Einhalten der vorgegebenen Aktionen der Fahrzeuge sichergestellt ist, müssen mögliche schlechte Handlungen nicht berücksichtigt werden. Dieses Problem wird gelöst, indem jeweils nur der beste, mit den niedrigsten Kosten bewertete Kindknoten an den Elternknoten weitergegeben wird. Das repräsentiert die Annahme, dass die Fahrzeuge stets die beste Aktion ausführen werden. Neben der Backpropagation-Phase wurde auch die Selection-Phase an diese unterschiedliche Anforderung angepasst. Im Folgenden wird der Verlauf einer Iteration erläutert. Dabei wird auf die Unterschiede zum ursprünglichen MCTS-Verfahren eingegangen und der Verlauf des MCTS-Verfahren mit seinen vier Schritten mit Beispielen erklärt.

Ausgangspunkt der Erläuterung ist ein initiales Verkehrsszenario aus Abb. 3.3 mit Fahrzeug 1 und der sich daraus ergebende Wurzelknoten des Suchbaums. Der Selection-Prozess bricht

dabei auf der Suche nach dem nächsten zu expandierenden Knoten sofort ab, weil im initialen Zustand nur der Wurzelknoten existiert. Dieser muss als Nächstes expandiert werden. In der Expansion-Phase werden zuerst alle zulässigen Aktionen der beteiligten Fahrzeuge bestimmt.

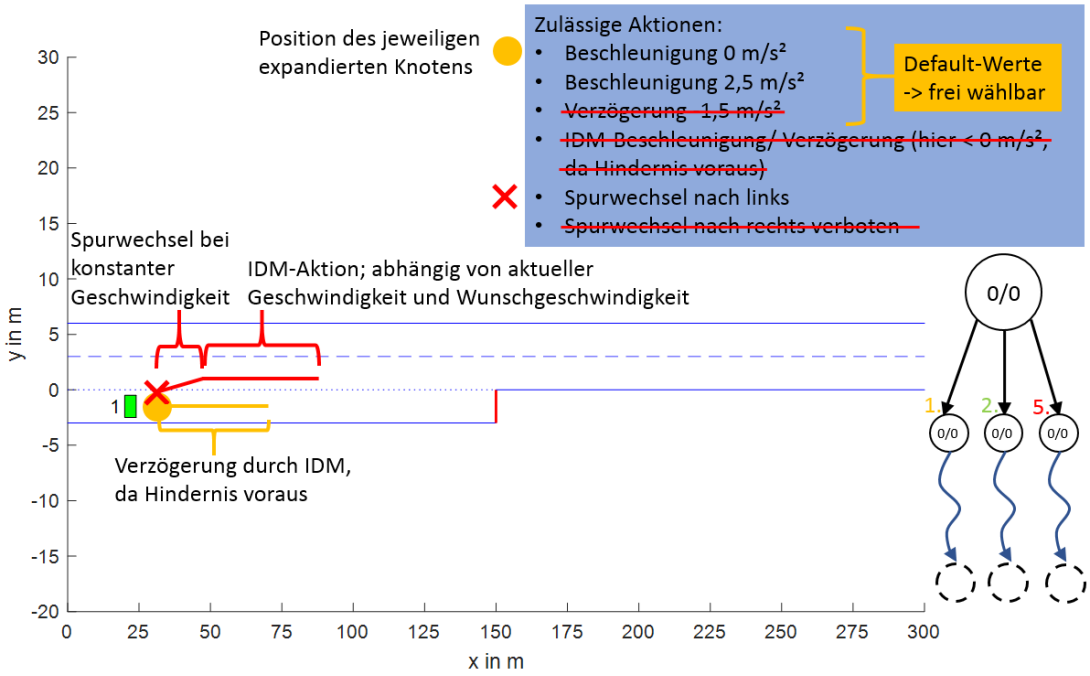


Abb. 3.3 Verkehrsszenario zur Erläuterung des Baumsuchverfahrens während der Simulation-Phase

Für das Fahrzeug sind im Beispiel nur Fahrt mit konstanter Geschwindigkeit, Beschleunigung und Spurwechsel nach links zulässig. Ein Spurwechsel wird auf der rechten Seite durch den Fahrbahnrand unterbunden. Das Verzögern auf der Beschleunigungsspur wird per Definition verboten, weil das gerade auf dieser Fahrspur zu gefährlichen Situationen führen kann. Die Anzahl möglicher Aktionen für das Fahrzeug ist $k = 3$. Daraus resultieren nach Gl. (2.9) drei zu expandierende Kindknoten. Dieser Vorgang erfolgt analog zu der Darstellung in Abb. 2.10. In Abb. 3.3 werden der Übersichtlichkeit wegen nur die expandierten Kindknoten mit konstanter Geschwindigkeit (Aktion $a_{=0}$) und Spurwechsel nach links (SPW_{links}) dargestellt.

Die drei neuen Zustände werden anschließend in der Simulation-Phase mithilfe der default policy, in diesem Fall dem IDM, untersucht und als Grundlage für die Bewertung durch die Kostenfunktionen bereitgestellt. Mit dem IDM wird ein Standardverhalten jedes Fahrzeugs simuliert. Die Fahrzeuge können nur beschleunigen oder verzögern, aber nicht die Fahrspur wechseln. Im Vergleich zum ursprünglichen MCTS-Algorithmus wird diese Simulation nicht bis zum Planungshorizont durchgeführt, sondern nur eine gewisse Zeitspanne T_{SimIDM}, die mit einem Zeitschritt Δt_{SimIDM} diskretisiert ist. Das ist notwendig, weil Spurwechsel im IDM nicht enthalten sind. Bei der Simulation bis zum Planungshorizont würden dann mögliche Spurwechsel, die zu einer optimalen Handlungsabfolge führen würden, komplett ausgeblendet. Durch das Simulieren von nur wenigen Sekunden können die Aktionen und deren unmittelbare Auswirkungen bewertet werden. Für langfristige Aussagen ist das IDM nicht geeignet. Weil die Zeitdiskretisierung des Suchbaumes bei maximal einer Sekunde liegt, kann mit dem IDM und dessen Simulation über wenige Sekunden eine ausreichend gute Situationsbewertung erreicht werden. Der große Vorteil gegenüber anderen Baumsuch-Algorithmen, dass keine Heuristik benötigt wird, bringt aber Einschränkungen mit sich. Aufgrund der Tatsache, dass das IDM nur das Standardverhalten ohne Spurwechsel berücksichtigt, können der „tatsächliche“ Wert und die Kosten einer Aktion nicht bestimmt werden. Es ist lediglich möglich, eine Abschätzung über

die Auswirkungen der Aktion zu erhalten. Eine Heuristik bietet dagegen einen verlässlichen Wert zur Orientierung. In einigen Fällen kann das dazu führen, dass die optimale Lösung nicht gefunden wird, weil die Abschätzung nicht genau genug ist oder einzelne Aspekte nicht erfasst werden. Ein Beispiel ist das genannte Vernachlässigen von Spurwechsel-Aktionen. Außerdem erlaubt das IDM nur die sinnvolle Simulation eines kurzen Zeitrahmens. Längerfristige Abschätzungen werden immer ungenauer und liefern irreführende Ergebnisse, weil sich das kooperative Handeln eben genau durch das Abweichen vom Standardverhalten auszeichnet. Eine Simulation bis zum Planungshorizont, wie in den Spiel-Anwendungen des MCTS-Verfahrens, ist nicht möglich. Durch das IDM werden $T_{SimIDM} \,/\, \Delta t_{SimIDM}$ neue Zustände für jeden der drei Kindknoten erzeugt. Diese neuen Zustände repräsentieren das Ausführen der situationsabhängigen IDM-Beschleunigung und die daraus resultierende Fahrzeugposition und -geschwindigkeit. Im gegebenen Beispiel wird das Fahrzeug im Zustand nach der Aktion $a_{=0}$ bei der IDM-Simulation eine starke Verzögerung ausführend, weil es mit hoher Geschwindigkeitsdifferenz auf ein stehendes Hindernis zufährt. Im Falle der Aktion SPW_{links} simuliert das IDM zuerst einen Spurwechsel mit konstanter Geschwindigkeit und dann abhängig von der aktuellen Geschwindigkeit und der Wunschgeschwindigkeit eine Beschleunigung oder Verzögerung. Bei einem Spurwechsel befindet sich das Hindernis nicht mehr vor dem Fahrzeug. Es hat folglich keinen Einfluss mehr auf die IDM-Aktion.

In der Backpropagation-Phase werden die Zustände, die sich aus der Simulation mit dem IDM ergeben, bewertet. Das geschieht für jeden der Kindknoten einzeln. Einer der Kindknoten besitzt $T_{SimIDM} \,/\, \Delta t_{SimIDM}$ Zustände, die mit den Kostenfunktionen bewertet werden. Für jeden Zeitschritt der IDM-Simulation ergibt sich ein Wert für die in Abschnitt 3.2.3 definierten Kosten, weil die Kostenfunktionen nur einen Zustand zur Berechnung benötigen. Zuerst werden alle Einzelkosten für ein Fahrzeug addiert (Gl. (3.10)). Die Summe der Kosten aller Fahrzeuge ergibt die Kosten eines Zustands von insgesamt $T_{SimIDM} \,/\, \Delta t_{SimIDM}$ Zuständen, die sich aus der IDM-Simulation ergeben haben. Abschließend werden diese aufsummiert. Daraus resultieren die Kosten, die dem Kindknoten zugeordnet werden. Im gegebenen Beispiel bewirkt die Aktion $a_{=0}$ hohe Kosten, weil zum einen wegen des Hindernisses stark verzögert wird. Zum anderen wird der Abstand immer geringer und sobald dieser den geforderten Sicherheitsabstand unterschreitet, erzeugt das hohe Kosten. Zusätzlich steigt durch die Verzögerung die Abweichung von der Wunschgeschwindigkeit an. Im Gegensatz dazu sind die Kosten bei Ausführung eines Spurwechsels niedrig. Dieser verursacht zwar an sich Kosten, diese sind aber aufgrund der hierarchischen Abstufung durch die Gewichtungsfaktoren deutlich geringer als die einer Unterschreitung des Sicherheitsabstandes. Lediglich die Abweichung von der Wunschgeschwindigkeit verursacht in diesem Fall Kosten. Die eine Aktion, die nicht beschrieben wurde, ruft ähnlich hohe Kosten wie die Aktion $a_{=0}$ hervor, weil sich stets das Hindernis vor dem Fahrzeug befindet. Nur der Spurwechsel sorgt dafür, dass das Hindernis umgangen werden kann. Deshalb besitzt der Knoten, der durch den Spurwechsel aus dem Wurzelknoten (initialem Zustand) resultiert, die niedrigsten Kosten.

Der ursprüngliche MCTS-Algorithmus weist jedem Knoten seine Kosten und die Anzahl, wie oft der Knoten besucht wurde, zu. Mit diesen beiden Werten werden die durchschnittlichen „Kosten pro Spiel“ errechnet, die im Selection-Prozess benötigt werden. Bei der Anpassung des Verfahrens an die kooperative Fahrmanöverplanung reichen die beiden Werte nicht mehr aus, weil nicht alle Kindknoten-Kosten summiert werden, sondern nur der Beste an den Elternknoten weitergegeben wird. Da die Anzahl der Spiele durch den jeweiligen Knoten nicht mehr aussagekräftig ist, wird eine neue Backpropagation-Methode entwickelt, die die neuen Anforderungen berücksichtigt. Dazu wird neben den Kosten des Knotens und dessen Anzahl an Spielen der ursprüngliche Kostenwert und die Baumtiefe des Knotens, auf dem der aktuelle

beste Pfad liegt, gespeichert. Die ursprünglichen Kosten sind diejenigen, die durch das IDM bei der erstmaligen Untersuchung des Knotens resultieren. Wird ein Knoten expandiert, so werden zu seinen ursprünglichen die neuen Kosten der Kindknoten addiert. Der ursprüngliche Kostenwert geht dann verloren. In dem entwickelten Verfahren werden sie behalten. Die Tiefe des aktuell besten Pfades entspricht dem Zeitschritt des Suchbaumes, aus dem die Kosten hervorgehen. In Abb. 3.4 ist das Verfahren beispielhaft für einen teilweise expandierten Suchbaum dargestellt.

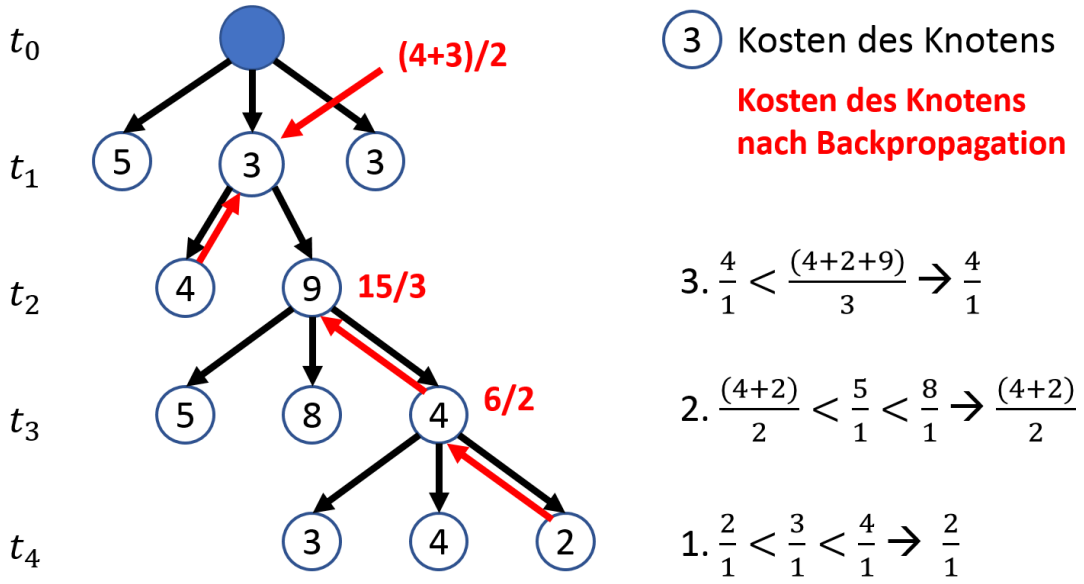


Abb. 3.4 Beispielhafter Ablauf der weiterentwickelten Backpropagation-Phase

Die Kosten in den Kreisen sind dabei die ursprünglichen Kosten, die fest hinterlegt werden. Ergibt sich ein neuer Wert für die Kosten im Laufe der Backpropagation-Phase, ist dieser in roter Schrift neben dem entsprechenden Knoten notiert. Bei der Betrachtung eines Knotens werden die aktuellen, nicht die ursprünglichen Kosten, betrachtet und jeweils durch die Baumtiefe, aus der die Kosten hervorgehen, geteilt. Ist ein Knoten noch nicht expandiert worden, so ist die Baumtiefe $j_{treedepth} = 1$ und die aktuellen Kosten entsprechen den ursprünglichen. Im vorliegenden Beispiel wurde als Letztes der Knoten expandiert, der seine Kindknoten auf der Baumebene t_4 hat. Wie bereits beschrieben, wird nur der Wert mit den niedrigsten Kosten weiter berücksichtigt. In diesem Fall wird die Ungleichung (1.) untersucht und der Knoten mit den Kosten $J = 2$ gewählt. Diese Kosten werden auf die ursprünglichen Kosten des Elternknotens addiert, die hinterlegte Baumtiefe beträgt für diesen Elternknoten anschließend $j_{treedepth} = 2$. Im nächsten Schritt erfolgt wieder eine Suche nach dem besten Knoten mit den niedrigsten Kosten (2.). Die gerade gebildete Summe wird durch die dazugehörige Baumtiefe geteilt, um einen vergleichbaren Wert zu bilden. Die Kosten des optimalen Knotens werden anschließend eine Baumebene nach oben propagiert. Dabei erfolgt wieder eine Summenbildung der ursprünglichen Kosten des darüberliegenden Elternknotens und der niedrigsten Kosten des Kindknotens. Die dazugehörige Baumtiefe ist $j_{treedepth} = 3$. Der gleiche Schritt wird wiederholt (3.). In dem Beispiel ist nun der noch nicht expandierte Knoten mit den Kosten $J = 4$ besser, weil er die niedrigeren Kosten aufweist. Aus diesem Grund wird er weiterverfolgt. Der Knoten ist noch nicht expandiert und hat daher die zugehörige Baumtiefe $j_{treedepth} = 1$. Auf der Baumebene t_1 werden die Kosten des darunterliegenden besten Kindknotens auf die ursprünglichen Kosten des Elternknotens addiert. Die hinterlegte Baumtiefe wird um 1 erhöht. Es handelt sich im eigentlichen Sinne

nicht um die absolute Baumtiefe, die fest mit der Baumebene gekoppelt ist, sondern um die Baumtiefe der Kosten relativ zum betrachteten Knoten. Weil der nächste Elternknoten im Beispiel bereits der Wurzelknoten ist, ist die Backpropagation-Phase beendet. Ein neuer Iterationsschritt beginnt mit der Selection-Phase. Die hinterlegten Daten am Beispiel des Knotens mit den ursprünglichen Kosten $J = 9$ sind folgende: die ursprünglichen Kosten aus dem IDM, die Anzahl, wie oft der Knoten gespielt wurde (analog dem MCTS-Verfahren), die aktuellen Kosten des Knotens (in diesem Fall 15) und die relative Baumtiefe, aus der die aktuell besten Kosten stammen (in diesem Fall 3). Jeder Knoten besitzt damit vier hinterlegte Werte.

Der Selection-Prozess beginnt stets am Wurzelknoten des Baumes. Von dort arbeitet er sich mithilfe der tree policy solange durch den Suchbaum, bis er den nächsten zu expandierenden Knoten ermittelt hat. Durch die Änderungen der Backpropagation-Phase muss auch die Selection-Phase angepasst werden. Vor allem die neuen gespeicherten Werte der Knoten sind relevant. Verwendet wird ebenfalls die UCT-Gleichung (Gl. (2.10)). Der Exploration-Term wird komplett übernommen. Der Expansion-Term muss angepasst werden, weil der gemittelte Gewinn aufgrund der unterschiedlich hohen Kosten nicht mehr im Intervall $[0, 1]$ liegt. Dieses Problem wird in Gl. (3.9) adressiert.

$$\bar{X}_{Knoten} = \left(1 - \frac{aktuelle\ Kosten\ des\ Knotens}{max\{aktuelle\ Kosten\ aller\ Geschwisterknoten\}}\right) \tag{3.9}$$

Durch die Umformung bekommt der Knoten mit den maximalen Kosten den Wert 0 zugewiesen und der Knoten mit den geringsten Kosten erhält den höchsten Wert, maximal jedoch den Wert 1. Verwendet werden stets die aktuellen Kosten, die ursprünglichen finden keine Beachtung in der Selection-Phase. Im oben dargestellten Beispiel würde der rechte Knoten auf Baumebene t_1 expandiert werden, weil seine aktuellen Kosten am geringsten sind ($3 < \frac{7}{2} < 5$).

Als Abbruchbedingung für die Iterationsschleife wird die Anzahl der Iterationen im Voraus bestimmt. Wird diese Zahl erreicht, dann wird der bisher beste Kindknoten des Wurzelknotens ausgewählt. Bei der Verwendung des UCT-Gleichung ist der optimale Knoten derjenige, der am häufigsten gespielt wurde. Denkbar wäre auch das Auswählen des Knotens mit den niedrigsten Kosten. Der ausgewählte Knoten wird anschließend als neuer Wurzelknoten eingesetzt und die Iterationsschleife beginnt von vorne. Dies geschieht so lange, bis der ausgewählte nächste Wurzelknoten auf der Baumebene des Planungshorizontes liegt. In diesem Fall ist der beste letzte Knoten gefunden. Die Abfolge der bis dahin gewählten Knoten und die damit verbundenen Aktionen stellen die optimale Handlungsabfolge der Fahrzeuge dar. Wird während der Baumsuche bis auf die unterste Baumebene vorgedrungen, dann kann die Suche abgebrochen werden. Wenn das der Fall ist, dann wurde der beste Pfad bereits gefunden. Eine erneute Iterationsschleife bringt dann keine besseren Knoten hervor.

Ein großer Vorteil ist, dass bereits berechnete Ergebnisse wiederverwendet werden können. Weil alle Fahrzeugbewegungen vorhersehbar sind und kein „Gegenspieler“ existiert, sind die Ergebnisse der vorherigen Iterationsschleife unmittelbar wiederverwendbar. Anstatt dass ein komplett neuer Suchbaum unter dem neuen Wurzelknoten aufgebaut wird, kann der alte Suchbaum weiter benutzt werden. Der Selection-Prozess startet lediglich bei dem ausgewählten neuen Wurzelknoten und nicht beim ursprünglichen Wurzelknoten zum Zeitpunkt t_0. Bei dem Beispiel aus Abb. 3.3 erweist sich das unmittelbare Spurwechseln und Fahren auf der rechten Fahrspur als optimale Handlung. Das spiegelt die definierten Kostenfunktionen und das angepasste Suchverfahren auf Basis des MCTS-Algorithmus wider.

3.3 Umsetzung des Algorithmus in Matlab

Der beschriebene Algorithmus wird in der Matlab-Programmierumgebung des Software-Unternehmens Mathworks implementiert. Die Programmstruktur und die angelegte Ordnerstruktur werden in Abschnitt 3.3.1 erläutert. Dabei wird auf einige Besonderheiten der Programmierung in Matlab eingegangen und der angewandte Programmierstil erklärt. Die Umsetzung des Algorithmus wird in drei Teile gegliedert. Das Preprocessing behandelt die Eingabe und Visualisierung von Daten, die in die Berechnung der kooperativen Fahrmanöver eingehen. Vor allem das Abfragen der Inputdaten per graphischer Benutzeroberfläche, kurz GUI (engl.: graphical user interface), und deren Verarbeitung für die Berechnung stehen im Mittelpunkt. Den Hauptteil stellt die Berechnung des kooperativen Fahrmanövers dar und wird in Abschnitt 3.2.4 gezeigt. Dort findet sich die Umsetzung des erwähnten neuen Verfahrens. Das Postprocessing beschäftigt sich damit, wie die Ergebnisse der Berechnungen mittels einer GUI visualisiert und ausgewertet werden.

3.3.1 Aufbau der Programmstruktur und Definition von Klassen mittels OOP

Der Algorithmus wird in mehrere einzelne Matlab-Skripte unterteilt, die je nach Aufgabe und Stelle im Ablauf der Simulation in verschiedenen Ordnern abgelegt sind. Die Ordnerstruktur ist in Abb. 3.5 visualisiert. Auf der obersten Ordnerebene sind 01_Hauptsoftware, 02_erstellte-Szenarien, 03_Simulationsergebnisse und 04_GIFs im Ordner mit dem Namen „Matlab“ zu finden. 01_Hauptsoftware beinhaltet den kompletten Programmcode des Algorithmus. Dieser wird unterteilt in 01_Pfadverwaltung, 02_GUI, 03_Verkehrsszenario, 04_Simulation_InputData, 05_Simulation_classdef, 06_Simulation_Result, Initialisierung.m und run_Hauptskript.m. 01_Pfadverwaltung beinhaltet das Skript zur Pfadverwaltung. Es bildet die gesamte Ordnerstruktur unterhalb des Matlab-Ordners ab. Dadurch ist es ausreichend, diesen Ordner und alle darunterliegenden an einen beliebigen Speicherort zu kopieren. Mit der Pfadverwaltung werden die neuen Pfade automatisch angepasst und das Programm kann problemlos ausgeführt werden.

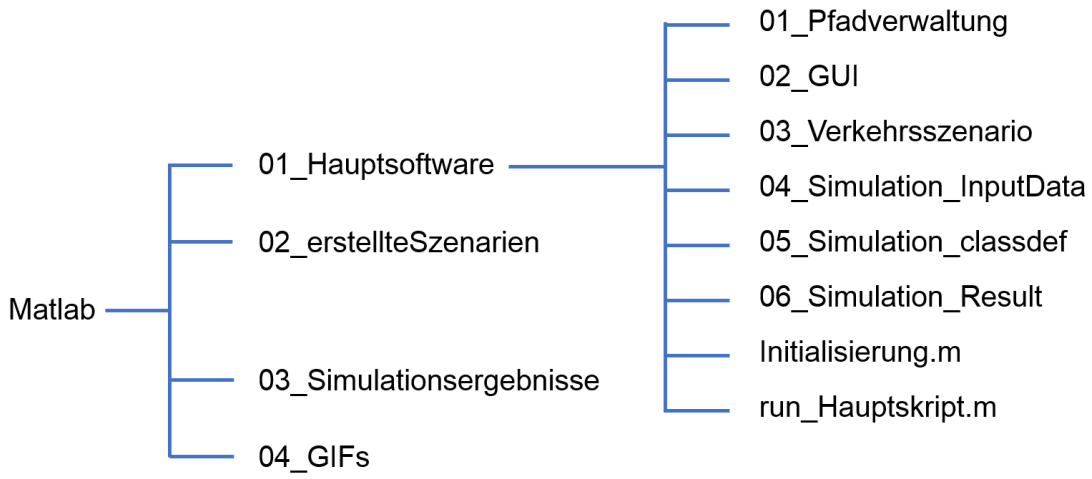


Abb. 3.5 Visualisierung der Ordnerstruktur des Matlab-Programms

In 02_GUI sind alle Skripte hinterlegt, die für den Aufbau und die Benutzung der GUI notwendig sind. Neben der Erzeugung der GUI-Elemente, wie Buttons oder Textfelder, schließt das auch

die Plots des erzeugten Verkehrsszenarios und der Simulationsergebnisse ein. Ebenfalls in dem Ordner enthalten sind Funktionen, die die eingegebenen Nutzerdaten umwandeln und in einer für die Simulation verwendbaren Form abspeichern. Abgelegt werden die umgeformten Eingabedaten im Ordner 04_Simulation_InputData. Dort befinden sich die GUI-Daten (savedGUIdata.mat), die umgeformten Daten des Verkehrsszenarios für die Simulation (Simulation_Inputdaten.mat), die aus den GUI-Daten gewonnen werden, und die eingegebenen Simulations-Parameter (SimTime_Parameter.mat). In dem Ordner mit dem Namen 03_Verkehrsszenario befindet sich ein Skript, mit dem Verkehrsszenarien direkt, ohne GUI, erstellt werden können. Für den eigentlichen Programmablauf hat dies jedoch keine Bedeutung mehr. In 05_Simulation_classdef sind die Klassendefinitionen MCTS_Tree, MCTS_Simulation, Traffic_Scenario und die dazugehörige IDM-Klasse zu finden. Die Klassen wurden mithilfe des objektorientierten Programmierens implementiert. Außerdem gespeichert sind in den jeweiligen Unterordnern die Parameter des MCTS-Verfahrens, wie die Gewichtungsfaktoren und die Parameter für das IDM im Traffic_Scenario-Ordner. Das Ergebnis der Berechnung wird in den Ordner 06_Simulation_Result gespeichert. Die Datei heißt SimRes.mat und entspricht stets der letzten durchgeführten Simulation. Bei jeder erneuten Simulation wird diese Datei überschrieben. Soll ein Simulationsergebnis behalten werden, muss es explizit abgespeichert werden. Der zugehörige Ordner ist eine Ebene unterhalb des Matlab-Ordners und heißt 03_Simulationsergebnisse. Ebenfalls auf dieser Ebene befinden sich 02_erstellte Szenarien, in den gespeicherte Verkehrsszenarien abgelegt werden können, und 04_GIFs, in den die einzelnen Bilder zur Erzeugung einer GIF-Datei gespeichert werden.

Das Skript 01_Hauptsoftware\Initialisierung.m startet das Programm. Dazu muss es in Matlab ausgeführt werden, wodurch die Haupt-GUI geöffnet wird. Von dort kann in die drei Abschnitte Verkehrsszenario-Editor, Simulation und Auswertung & Plot gewechselt werden, wie sie mit Abb. 3.6 veranschaulicht werden. Im ersten Teil findet das Preprocessing statt. Der Nutzer erstellt hier ein neues Verkehrsszenario oder bearbeitet ein altes, das abgespeichert wurde. In der zweiten GUI werden Simulationsparameter festgelegt und die Simulation gestartet, was den Hauptteil des Algorithmus darstellt. Bei Start der Simulation wird run_Hauptskript.m ausgeführt, das den Ablauf der Simulation steuert, die Anzahl der Iterationsschleifen überwacht und abschließend das Ergebnis speichert. Die dritte Benutzeroberfläche dient dazu, das Postprocessing durchzuführen und die Ergebnisse der Simulation zu visualisieren und auszuwerten.

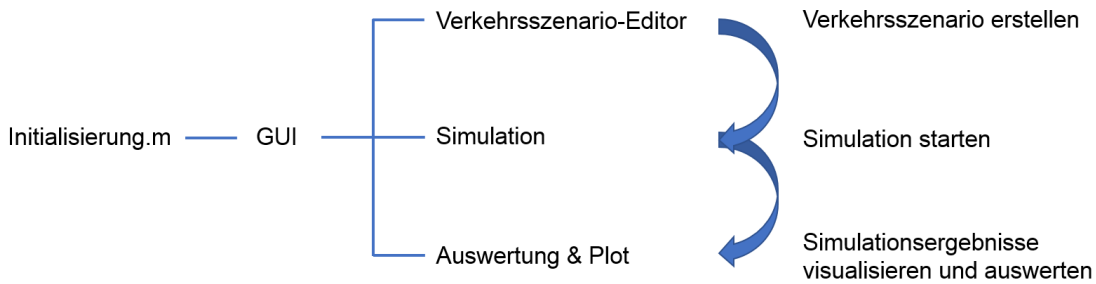


Abb. 3.6 Visualisierung des Programmablaufs zur Berechnung kooperativer Fahrmanöver

Das objektorientierte Programmieren (OOP) wurde von Programmiersprachen wie Java oder C++ übernommen und für die Anwendung in Matlab angepasst. Dadurch können Klassen definiert und standardmäßige objektorientierte Designmuster, wie die Wiederverwendung von Code, Vererbung, Kapselung und das Referenzverhalten, angewendet werden [53].

Beim objektorientierten Programmieren werden konkrete Objekte erzeugt, indem Klassen instanziiert werden. Die Objekte zeigen das für die Klasse festgelegte Verhalten, repräsentieren aber das anwendungsspezifische Objekt. In dieser Arbeit werden der Suchbaum, das Verkehrsszenario und die MCTS-Simulation als Objekte definiert. Der Vorteil ist, dass die jeweilige Klasse nur allgemein implementiert werden muss. Durch die Instanziierung erfolgt erst die Zuweisung konkreter Werte. Ein gutes Beispiel ist das Verkehrsszenario, das je nach Nutzereingabe ständig andere Werte besitzt. Die dahinterliegende Klasse ist jedoch immer identisch. Klassen weisen feste Eigenschaften und Methoden auf. Die Eigenschaften, wie bestimmte Zahlenwerte, müssen bei der Instanziierung definiert werden und sind dann fest mit dem erzeugten Objekt verbunden. Methoden sind Operationen, die das erzeugte Objekt ausführen kann. Bei der Klasse für den Suchbaum ist das unter anderem das Hinzufügen neuer Knoten.

Die beiden Klassen für den Suchbaum und die MCTS-Simulation wurden als Handle-Klassen implementiert. Das bedeutet, dass Operationen, die auf Variablen einer dieser Klassen zugreifen, nicht die Variable selbst kopieren. Denn im Normalfall wird eine Variable, wenn sie an eine andere Funktion übergeben wird, kopiert. Das gleiche trifft auf den Rückgabewert der Funktion zu. Mit Handles kann darauf verzichtet werden. Es wird lediglich eine Referenz auf die originale Variable übergeben. Mit dieser kann die Funktion direkt auf die Variable zugreifen und diese verändern. Ein Rückgabewert aus der Funktion ist nicht mehr notwendig [54]. Durch den Wegfall der direkten Variablenübergabe mittels Kopieren kann die Performance des Programms gesteigert werden. Vor allem bei der Baumstruktur mit ihrer hohen Anzahl an Knoten und der daraus resultierenden Datenmenge ist das ein großer Vorteil.

Die Klasse für das Verkehrsszenario Traffic_Scenario.m wird mit der Nutzereingabe initialisiert. Sie enthält alle Daten, die über die komplette Simulation identisch bleiben. Das sind das Fahrbahnlayout, die Fahrzeugparameter (Fahrzeugtyp, Wunschgeschwindigkeit usw.) und die IDM-Parameter. Die IDM-Parameter werden an eine Unterklasse weitergereicht, die nur für die IDM-Berechnung zuständig ist. Die Klasse für den Suchbaum MCTS_Tree.m wird mit den Daten für den initialen Zustand des Verkehrsszenarios erzeugt. Zusätzlich wird ein Wert für die initiale Speicherbelegung hinterlegt, der angibt, wie viele Knoten gespeichert werden können. In die Klasse für die Durchführung der Berechnung MCTS_Simulation.m gehen alle Simulationsparameter, die der Nutzer bestimmt, und die Gewichtungsfaktoren für die Kostenfunktionen mit ein.

3.3.2 Preprocessing: Definition des Verkehrsszenarios und der Berechnungsparameter

Bevor die Simulation gestartet werden kann, müssen das Fahrbahnlayout und die beteiligten Fahrzeuge definiert werden. Das geschieht mit dem Verkehrsszenario-Editor, einer graphischen Benutzeroberfläche, die über die Haupt-GUI aufgerufen wird. Der Aufbau dieses Verkehrsszenario-Editors ist in Abb. 3.7 erklärt. Im oberen, linken Teil wird das Fahrbahnlayout bestimmt. Einzugeben sind die Fahrspurbreite b_{FSP} in Meter (bei Markierung 1) und die Anzahl der Fahrspuren i_{FSP} (bei Markierung 2). Der Beschleunigungsstreifen zählt nicht dazu, er wird über eine Checkbox separat aktiviert (3). Im nächsten Textfeld wird seine Startposition x_{BSP}^{start} und Endposition x_{BSP}^{end} festgelegt (4). Gleiches gilt für das Hindernis, das durch die Angabe der blockierten Fahrspur (5) und dessen Startposition x_{vFSP}^{start} und Endposition x_{vFSP}^{end} angegeben werden muss (6). Es können mehrere Hindernisse, auch auf der gleichen Fahrspur, definiert werden. Neben den Textfeldern befinden sich jeweils Buttons mit Fragezeichen. Bei einem

Klick werden Informationen angezeigt, wie die Eingabe richtig durchzuführen ist und ob Einschränkungen bzgl. der einzugebenden Werte vorliegen. Bei der Fahrspurbreite ist ein Mindestwert von drei Metern gefordert. Hindernisse müssen mindestens 100 Meter lang sein.

Im unteren Teil der GUI werden die initialen Fahrzeugzustände bestimmt. Mit der Eingabe der Anzahl an Fahrzeugen (7) erfolgt automatisch eine Anpassung der beiden Tabellen. Mit der linken Tabelle (8) werden der Anfangszustand und Fahrzeugparameter abgefragt. Anzugeben sind die x-Position, die initiale Geschwindigkeit, die Wunschgeschwindigkeit und der Fahrzeugtyp. Anstatt der direkten Angabe der y-Position wird die Fahrspur abgefragt. Ein Wert von Null bedeutet Beschleunigungsspur. Ansonsten erfolgt die Nummerierung der Fahrspuren wie bereits vorgestellt. Mit der rechten Tabelle (9) kann ausgewählt werden, welche Aktionen aus dem Set an Standardaktionen $\boldsymbol{a}_{act,def}$ dem jeweiligen Fahrzeug zur Verfügung stehen sollen. Im gezeigten Beispiel kann Fahrzeug 4 nur $a_{=0}$ ausführen, was eine konstante Geschwindigkeit bedeutet. Mit dieser Einstellung kann die Rechenzeit teilweise deutlich reduziert werden.

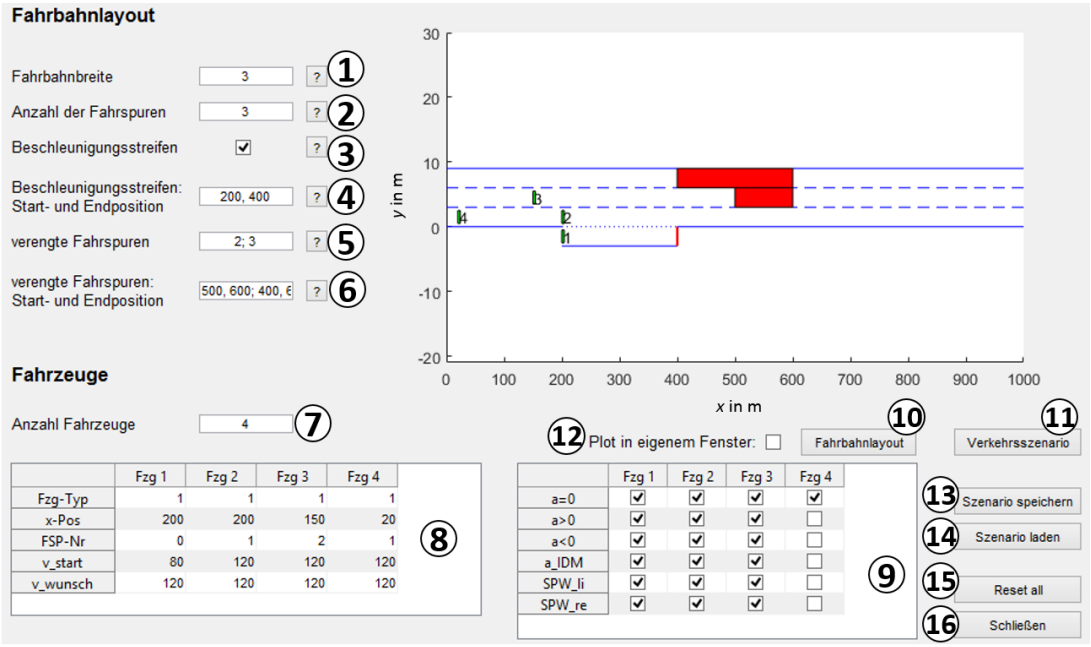


Abb. 3.7 Aufbau des Verkehrsszenario-Editors zur Definition eines Verkehrsszenarios

Die Buttons auf der rechten Seite der GUI haben folgende Funktionen: Mit dem Button „Fahrbahnlayout“ (10) lässt sich das Fahrbahnlayout alleine, ohne die Fahrzeuge, visualisieren. Mit der Schaltfläche rechts daneben (11) kann das Verkehrsszenario komplett angezeigt werden. Der Plot kann innerhalb der GUI erfolgen, wie es in der Abbildung dargestellt ist, oder in einem separaten Fenster. Das kann mittels der Checkbox „Plot in eigenem Fenster“ (12) bestimmt werden. Weiterhin kann das Szenario gespeichert (13) oder ein altes, bereits gespeichertes geladen (14) werden. Standardmäßig wird dazu der Ordner Matlab\02_erstellteSzenarien geöffnet. Bei der Verwendung aller vier genannten Buttons erfolgt automatisch eine Überprüfung der Nutzeingaben. Das beinhaltet das Überprüfen von Beschränkungen (bspw. $b_{FSP} \geq 3\,m$) sowie das Testen auf gültige Eingabewerte (keine Buchstaben o.Ä.) und Zahlenwerte (Zahlenwerte > 0). Außerdem wird überprüft, ob sich die Fahrzeuge und Hindernisse auf der Fahrbahn befinden. Nicht geprüft wird hingegen, ob Fahrzeuge bereits mit anderen Fahrzeugen oder Hindernissen kollidieren. Ist die Überprüfung erfolgreich abgeschlossen, kann der Speichervorgang fortgesetzt werden. Beim Laden von GUI-Daten ist darauf zu achten, dass die Dateien die Endung „_savedGUIdata.mat“ besitzen, weil nur sie in die GUI eingelesen werden können. Weitere Buttons erlauben das Zurücksetzen der GUI auf den letzten

gespeicherten Zustand (15) und das Schließen der GUI (16). Sobald ein Verkehrsszenario gespeichert oder ein altes geladen wurde, wird dieses im Zwischenspeicher abgelegt. Auf diesen Zwischenspeicher greift auch die graphische Benutzeroberfläche „Simulation“ zu. Wird sie über die Haupt-GUI geöffnet, wird automatisch das aktuelle Verkehrsszenario geplottet. Die GUI für die Eingabe der Simulationsparameter ist in Abb. 3.8 dargestellt.

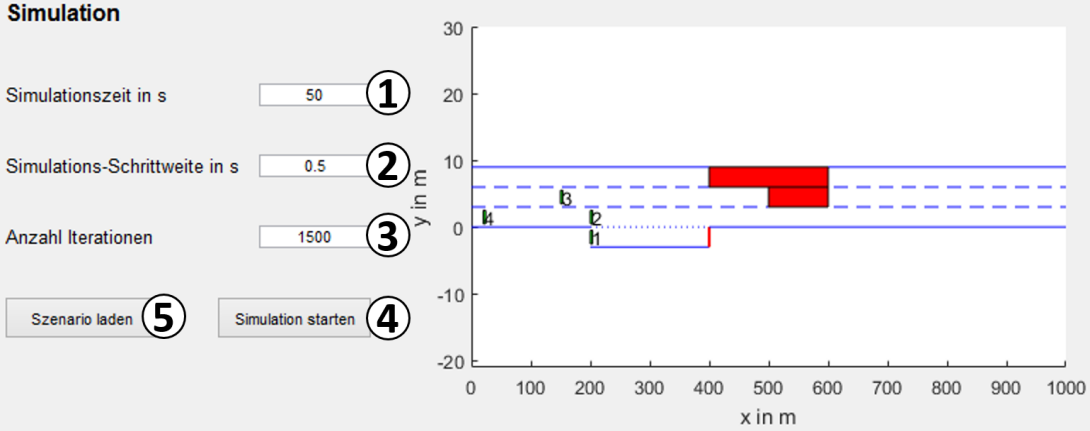


Abb. 3.8 Aufbau der graphischen Benutzeroberfläche zur Eingabe der Simulationsparameter

Die Simulationsparameter, die vom Nutzer abgefragt werden, sind der Planungshorizont der Simulation (bei Markierung 1), die Schrittweite der Simulation (2) und die Anzahl der Iterationen (3). Der Planungshorizont und die Schrittweite legen gleichzeitig die Tiefe des Suchbaums fest. Beide Parameter haben einen großen Einfluss auf die Rechenzeit. Die Schrittweite entspricht der zeitlichen Diskretisierung des Planungsproblems und bestimmt, über welche Zeitspanne die Fahrzugaktionen ausgeführt und die jeweils resultierenden, neuen Zustände berechnet werden. Die Anzahl der Iterationen kann frei gewählt werden. Diese hat ebenfalls einen großen Einfluss auf die Rechenzeit und auf die Güte des Simulationsergebnisses. Je mehr Iterationen pro Baumebene durchgeführt werden, desto besser wird das resultierende Ergebnis.

Bevor die Simulation gestartet werden kann, werden bei Betätigung der Schaltfläche (4) die Nutzereingaben auf ihre Gültigkeit überprüft, ähnlich wie es im Verkehrsszenario-Editor geschieht. Zum Laden von bereits definierten Verkehrsszenarios, die nicht dem aktuellen entsprechen, kann die linke Schaltfläche (5) betätigt werden. Dann öffnet sich analog zum Verkehrsszenarien-Editor das Ladefenster im Ordner Matlab\02_erstellteSzenarien. Hierbei gilt ebenfalls, dass nur die Dateien mit der Endung „_savedGUIdata.mat“ gelesen werden können. Das aktuelle Verkehrsszenario wird immer in dem Plot-Fenster innerhalb der GUI angezeigt.

Wird die Simulation gestartet, werden in der command line der Matlab-Benutzeroberfläche der Zustandsvektor aller Fahrzeuge sowie die erlaubten Aktionen in Matrixdarstellung angezeigt. Das entspricht den Werten der beiden Tabellen im Verkehrsszenario-Editor. Die Werte, die festlegen, ob eine Aktion zur Verfügung steht, sind die Zahl 1 für zulässig oder der Matlab-interne Wert „NaN“ für nicht erlaubt. „NaN“ steht für Not-a-Number und existiert nur innerhalb von Matlab. Außerdem erscheint die Nachricht „Simulation gestartet...“ in der command line. Die GUI ruft zum Start der Simulation das Skript run_Hauptskript.m auf, das den weiteren Verlauf der Simulation organisiert.

3.3.3 Berechnung der kooperativen Fahrmanöver

Bevor der Algorithmus startet, müssen die Objekte aus den Klassen MCTS_Tree, MCTS_Simulation und Traffic_Scenario erzeugt werden. Dazu werden die Nutzereingaben aus dem Verkehrsszenario-Editor zusammengefasst und umgeformt. Die Daten werden in Matrixform angeordnet und verarbeitet, weil diese Form der Darstellung in Matlab viele Vorteile bietet. Es werden Matrizen angelegt, die das Fahrbahnlayout und die konstanten Parameter der Fahrzeuge (Fahrzeugtyp, Wunschgeschwindigkeit usw.) abbilden. Beide beinhalten nur Werte, die über die gesamte Simulation unverändert benutzt werden. Mit diesen Werten wird ein Objekt der Klasse Traffic_Scenario erzeugt. Dieses Objekt enthält alle Methoden, die notwendig sind, um das Verhalten der Fahrzeuge in der jeweiligen Fahrsituation zu bestimmen. Dazu zählt die Ermittlung der Fahrzeuge, die sich im unmittelbaren Umfeld des Ego-Fahrzeugs befinden, die Berechnung der IDM-Beschleunigung und das Verwerfen von nicht zulässigen Aktionen.

Mit dem Wegfall der Orientierungsänderung lässt sich die Darstellung des Fahrzeugzustandes auf die x- und y-Position sowie die Geschwindigkeit beschränken. Für die Auswahl durchzuführender Aktionen wird nur jeweils der aktuelle Zustand herangezogen. Daraus ergibt sich das Problem, dass anhand der drei genannten Größen bei einem gerade ablaufenden Spurwechsel die Spurwechselrichtung nicht erkannt werden kann. Die y-Position enthält keine Information, ob das Fahrzeug von links nach rechts wechselt oder umgekehrt. Aus diesem Grund muss eine zusätzliche Zustandsgröße eingeführt werden. Die neue, zusätzliche Zustandsgröße beschreibt die y-Position nicht in Metern, sondern bezogen auf die Fahrspur. Ein Fahrzeug, dass auf Fahrspur 1 fährt besitzt die y-Koordinate $y_{FSP} = 1$. Zur Erkennung der Spurwechselrichtung wird eingeführt, dass die y-Koordinate y_{FSP} bei einem gerade ablaufenden Spurwechsel die Ausgangsfahrspur beibehält, aber zusätzlich eine Nachkommastelle bekommt. Ein Spurwechsel von links nach rechts bedeutet einen Nachkommawert von 0,2, ein Spurwechsel von rechts nach links einen Nachkommawert von 0,8. Wechselt ein Fahrzeug von Fahrspur 1 auf Fahrspur 2 ergibt das $y_{FSP} = 1{,}8$, für einen Spurwechsel von Fahrspur 2 auf Fahrspur 1 resultiert eine y-Koordinate $y_{FSP} = 1{,}2$. Der neue Zustandsvektor $\boldsymbol{x}_i$ für ein Fahrzeug i wird mit Gl. (3.10) beschrieben, aufbauend auf Gl. (2.1).

$$\boldsymbol{x}_i = \left(x_i, y_i, y_{FSP,i}, v_i\right)^T \tag{3.10}$$

Durch die Einführung dieser Zustandsgröße ergeben sich viele Vereinfachungen im Programmcode, vor allem bei der Erkennung blockierter Fahrspuren, dem Finden der vorausfahrenden Fahrzeuge und der Berücksichtigung mehrerer blockierter Fahrspuren während Spurwechsel-Aktionen. Aus der Eingabe der initialen Fahrzeugzustände im Verkehrsszenario-Editor wird eine Matrix generiert, die pro Spalte den jeweiligen neu definierten Zustandsvektor enthält. Mit diesem initialen Zustand wird ein Objekt der Klasse MCTS_Tree erzeugt. Zusätzlich erforderlich ist ein Wert für die Anzahl der Knoten, die in der initialen Speicherbelegung berücksichtigt werden sollen. Die konkrete Wahl des Wertes ist nicht relevant, weil bei Mehrbedarf der Speicherplatz dynamisch erweitert wird. Der initiale Zustand stellt den Wurzelknoten des Suchbaumes dar. Die Simulationsparameter, die vom Nutzer in der GUI festgelegt wurden und die im Skript MCTS_Simulation_Parameter.m definiert sind, erzeugen ein Objekt der Klasse MCTS_Simulation. Dieses Objekt beinhaltet alle Bestandteile einer Iteration des MCTS-Verfahrens, nämlich Selection, Expansion, Simulation und Backpropagation. Das Skript enthält den UCT-Parameter C_p, die Gewichtungsfaktoren für die Kostenfaktoren und die Angabe für den Zeitraum, über den die default policy in der Simulation-Phase angewendet werden soll. Das erzeugte Objekt ist konstant über die Simulation und arbeitet nur mit Knoten des Suchbaumes aus dem Objekt der Klasse MCTS_Tree.

Sind alle drei Objekte erzeugt, beginnt die erste Iterationsschleife. Wie bereits erwähnt, wird der komplette Berechnungsprozess im Skript run_Hauptskript.m koordiniert. Von dort aus werden die einzelnen Objekte und Funktionen aufgerufen. In diesem Skript gibt es zwei ineinander verschachtelte Iterationsschleifen. Beide sind mit ihren wichtigsten Operationen in gekürztem Matlab-Code in Abb. 3.9 dargestellt.

```
while i_SimTime <= MCTS_Sim.T_final/MCTS_Sim.t_timestep

    i_Zaehler = 0;

    while i_Zaehler <= Anzahl_Iterationen

        nextNode = MCTS_Sim.TreePolicy(newTree, root_node);

        if isnan(nextNode) || isinf(nextNode)
            break
        end

        MCTS_Sim.expandTree(newTree, traffic, nextNode)

        i_Zaehler = i_Zaehler + 1;

    end

    optimaler_Knoten = ...

    if isnan(nextNode)
        ...
    end

    if isinf(nextNode)
        ...
        i_SimTime = 1;
        optimaler_Knoten = 1;
        root_node = optimaler_Knoten;

        X_Fahrzeuge_solution = [];
        newTree.setOnes_MCTSdata();
        continue

    end
    ...
    root_node = optimaler_Knoten;
    ...
    X_Fahrzeuge_solution = [X_Fahrzeuge_solution; X_Fahrzeuge_opt];
    ...
    i_SimTime = i_SimTime + 1;

end
```

Abb. 3.9 Gekürzter Matlab-Code der beiden Iterationsschleifen für die Simulation

Die äußere Iterationsschleife läuft so lange, bis der optimale Knoten auf Ebene des Planungshorizontes gefunden wurde. Die innere Schleife arbeitet für jeden Zeitschritt die vorgegebene

Anzahl an Iterationen ab. In der inneren Schleife wird zu Beginn die tree policy angewandt, mit der der nächste zu expandierende Knoten ermittelt wird. Ausgabewert ist die Nummer des Knotens, der expandiert werden soll. Anschließend folgt für diesen Knoten die Expansion-, Simulation- und Backpropagation-Phase. Sind alle Iterationen eines Zeitschritts durchlaufen, wird der optimale Knoten, das ist der Knoten mit den meisten „Spielen“, bestimmt. Dieser Knoten wird als der neue Wurzelknoten festgelegt und der zugehörige Zustand wird in der Lösungsmatrix abgespeichert. Danach beginnt die äußere Iterationsschleife wieder von vorne.

Das erste if-Statement in der inneren Iterationsschleife dient dazu, bei einem ungültigen Wert für den nächsten zu expandierenden Knoten aus der inneren while-Schleife auszusteigen. Es erfolgt abhängig vom Wert eine Fehlerbehandlung. Die Knotenwerte werden in der tree policy absichtlich auf „NaN“ oder „Inf“ (=Unendlich) gesetzt, um sie einfach detektieren zu können. Der NaN-Wert bedeutet in diesem Fall, dass die Kosten eines Knotens gegen unendlich gehen und daher mit der tree policy keine sinnvolle Aussage mehr getroffen werden kann. Das ist bei unendlichen Kosten deswegen der Fall, weil alle Kosten der Geschwisterknoten durch den auftretenden maximalen Kostenwert geteilt werden, um die UCT-Wert zu berechnen. Bei unendlichen Kosten bedeutet das für alle anderen Kosten den Wert 0. Eine richtige Entscheidung ist nicht mehr möglich. In diesem Fall wird die innere while-Schleife abgebrochen und der beste Knoten anhand der niedrigsten Kosten und nicht anhand der meisten Spiele gewählt. Der Knoten mit den meisten Spielen ist i.d.R. der Knoten, dessen Kosten gegen unendlich gehen. Dieser Fall tritt selten auf und wird nur der Vollständigkeit halber abgefangen. Wird während der Ausführung der tree policy dem Knotenindex der Wert „Inf“ zugewiesen, erfolgt ebenfalls ein Abbruch der inneren Iterationsschleife. Tritt dieser spezielle Wert auf, bedeutet das, dass alle Kindknoten des aktuellen Wurzelknotens einen unzulässigen Kostenwert, wie er bei Kollisionen auftritt, aufweisen. Dieser Fall tritt ein, wenn vom aktuellen Wurzelknoten aus keine Aktion möglich ist, mit der sich eine Kollision noch verhindern lässt. Wird ein vermeintlich guter Pfad nachverfolgt, der sich allerdings bei genauerer Untersuchung als ungültig herausstellt, muss dieses Ereignis abgefangen werden. Passiert das bereits beim initialen Zustand, so kann keine Lösung für das Planungsproblem gefunden werden. Tritt dieser Fall erst bei Knoten auf, die sich auf unteren Baumebenen befinden, also die äußere Iterationsschleife schon mehrmals durchlaufen wurde, kann in den meisten Fällen eine Lösung gefunden werden. Dazu wird der aktuelle „Wurzelknoten“ wieder auf den initialen Zustand (t_0) gesetzt und der Suchbaum nochmal „von oben“ untersucht, allerdings unter Verwendung der bisher berechneten Ergebnisse. Damit wird erreicht, dass der Bereich, in dem die Baumsuche keine gültige Lösung mehr finden konnte, umgangen wird. Das Suchverfahren hat dadurch die Möglichkeit, andere Pfade zu untersuchen, die zuvor wegen des vermeintlich besten Pfades vernachlässigt wurden. Damit die Entscheidung nicht wieder zugunsten der gleichen Knoten fällt, wird die Anzahl der Spiele für den gesamten Suchbaum auf den Anfangszustand zurückgesetzt.

Die Expansion- und Simulation-Phase erfolgt wie in Abschnitt 3.2.4 beschrieben. Beide sind Methoden des Objektes der Klasse MCTS_Simulation. Die Backpropagation-Phase wird in zwei Teile unterteilt. Der eine Teil ist die Berechnung der Kosten, die ebenfalls eine Methode des gerade genannten Objektes ist. Dort wird das Ergebnis der Simulation mit dem IDM verwendet. Der Kostenwert errechnet sich mithilfe der Kostenfunktionen und der dazugehörigen Gewichtungsfaktoren. Die bereits erläuterten Kostenfunktionen werden in der Umsetzung des Algorithmus teilweise erweitert. Die Berechnung der Kosten für die Abweichung von der Wunschgeschwindigkeit wird asymmetrisch gestaltet, so dass eine aktuelle Geschwindigkeit, die über der Wunschgeschwindigkeit liegt, stärker bestraft wird als wenn sie um die gleiche Differenz unterhalb liegen würde. Zusätzlich wird bei den Kosten für das Nicht-Einhalten des

Rechtsfahrgebotes eingeführt, dass das Fahren auf der Beschleunigungsspur übermäßig viel kostet, um ein frühzeitiges Spurwechseln zu fördern. Die Kosten werden für alle Kindknoten des expandierten Knotens berechnet und in einer Matrix abgelegt. Der zweite Teil der Backpropagation ist eine Methode des Objektes der Klasse MCTS_Tree, an die die Matrix mit den Kostenwerten übergeben wird. Von der Eingangsmatrix wird der minimale Kostenwert ermittelt und dieser wird an den expandierten Knoten (Elternknoten) weitervermittelt. Die Methode führt das angepasste Backpropagation-Verfahren durch, wie es in Abschnitt 3.2.4 beschrieben ist. Alle Kindknoten des expandierten Knotens werden in diesem Schritt dem Suchbaum hinzugefügt. Ist die Backpropagation-Phase abgeschlossen, beginnt eine neue Iteration der inneren while-Schleife. In Abb. 3.10 ist beispielhaft dargestellt, welche Daten für die Kindknoten des initialen Knotens abgelegt werden. Das berechnete Verkehrsszenario ist das aus Abb. 3.3. Für das Fahren auf dem Beschleunigungsstreifen werden Bremsmanöver generell verboten, da das auch im heutigen Straßenverkehr selten auftritt und zu Gefahrensituationen, vor allem für die hinterherfahrenden Fahrzeuge, führen kann. Das gleiche gilt auch für die IDM-Beschleunigung. Diese ist aufgrund des Hindernisses, was dem Ende der Beschleunigungsspur entspricht, negativ und wird deshalb verworfen. Außerdem entfällt der Spurwechsel nach rechts. Damit reduzieren sich die durchführbaren Aktionen auf konstante Geschwindigkeit (Zeile 2 in der Abbildung), Standard-Beschleunigung (Zeile 3) und Spurwechsel nach links (Zeile 4). In den Spalten sind die ursprünglichen Kosten des Knotens (Spalte 1), die Anzahl an Spielen (Spalte 2), die Summe aus den ursprünglichen Kosten und den besten Kosten der unteren Ebenen (Spalte 3) und die Tiefe der Ebene, aus deren Kosten sich die Kostensumme zusammensetzt (Spalte 4). Der sofortige Spurwechsel nach links stellt die beste Aktion dar, wie an den ursprünglichen Kosten zu sehen ist. Dieser Knoten wird weiter untersucht. Weil keine weiteren Fahrzeuge oder Hindernisse im Verkehrsszenario sind, ist der sofortige Spurwechsel so niedrig in den Kosten, dass die anderen Varianten nicht untersucht werden. Der Such-Algorithmus dringt bis auf die unterste Ebene des Suchbaumes mit der Tiefe 30 vor und bricht dann ab. Die optimale Lösung wurde bereits innerhalb der ersten Iterationsschleife gefunden. Wenn das der Fall ist, bricht der Suchalgorithmus ab und gibt den Pfad als Lösung aus. In der obersten Zeile befindet sich der Wurzelknoten, von dem nur die Gesamtzahl an Spielen verfolgt wird. Die anderen Parameter spielen für ihn keine Rolle.

	1	2	3	4
1	0	87	0	0
2	4.0949e+04	1	4.0949e+04	1
3	5.3640e+04	1	5.3640e+04	1
4	5.1002e+03	85	5.1005e+03	30

Abb. 3.10 Die für jeden Knoten hinterlegten Daten zur Durchführung der Backpropagation

Nachdem der beste Knoten zum Zeitpunkt des Planungshorizontes t_{end} gefunden wurde, wird auch die äußere while-Schleife verlassen. Der Lösungsvektor, der die Zustände der optimalen Handlungsabfolge enthält, wird im Ordner 06_Simulation_Result\temp als mat-File abgespeichert. Während der Simulation wird in der command line nach jedem Durchlauf der inneren while-Schleife angezeigt, wie viele Knoten expandiert wurden, wie lange der Durchlauf gedauert hat und wie viele Knoten pro Sekunde expandiert wurden. Zusätzlich wird berechnet, wie oft der gewählte, optimale Knoten gespielt wurde im Verhältnis zu der Spielzahl seines Elternknotens. Das kann einen Hinweis darauf geben, wie deutlich ein Knoten überlegen/besser war. Der Wert wäre für das oben gezeigte Beispiel die Anzahl der Spiele des besten Kindknotens mit 85 geteilt durch die Anzahl der Spiele des Elternknotens 87. Das ergibt einen Spielanteil

von 98 % und deutet darauf hin, dass die Entscheidung eindeutig ist und nicht mehrere ähnlich gute Aktionen existieren. Im schlechtesten Fall können alle Kindknoten gleich viele Spiele aufweisen. Deshalb kann keine eindeutige Entscheidung getroffen werden. Das spiegelt sich in einem niedrigen Prozentwert wider. Der ermittelte Prozentwert ist allerdings nicht immer ein verlässliches Indiz für die Qualität der Entscheidungsfindung. Wurde bspw. lange der „falsche“ Pfad nachverfolgt und erst spät der richtige ausgespielt, hat dieser trotzdem einen niedrigen Prozentwert. Brauchbar ist der Wert dennoch, weil damit immer bestimmt werden kann, wie eindeutig die Entscheidung ausgefallen ist.

3.3.4 Postprocessing: Visualisierung und Auswertung der Berechnungsergebnisse

Die graphische Benutzeroberfläche zur Visualisierung und Auswertung der Berechnungsergebnisse lässt sich über den unteren Button der Haupt-GUI aufrufen. Es öffnet sich die „Auswertung & Plot“-GUI, wie sie in Abb. 3.11 dargestellt ist. Nach einer Simulation wird automatisch das dazu erstellte Verkehrsszenario abgebildet. Werden alte Berechnungsergebnisse geladen, wird das zugehörige Verkehrsszenario angezeigt. Wird die GUI geschlossen und neu geöffnet, ist immer das zuletzt geladene Verkehrsszenario das aktuelle. Außer es wurde in der Zwischenzeit eine Simulation durchgeführt.

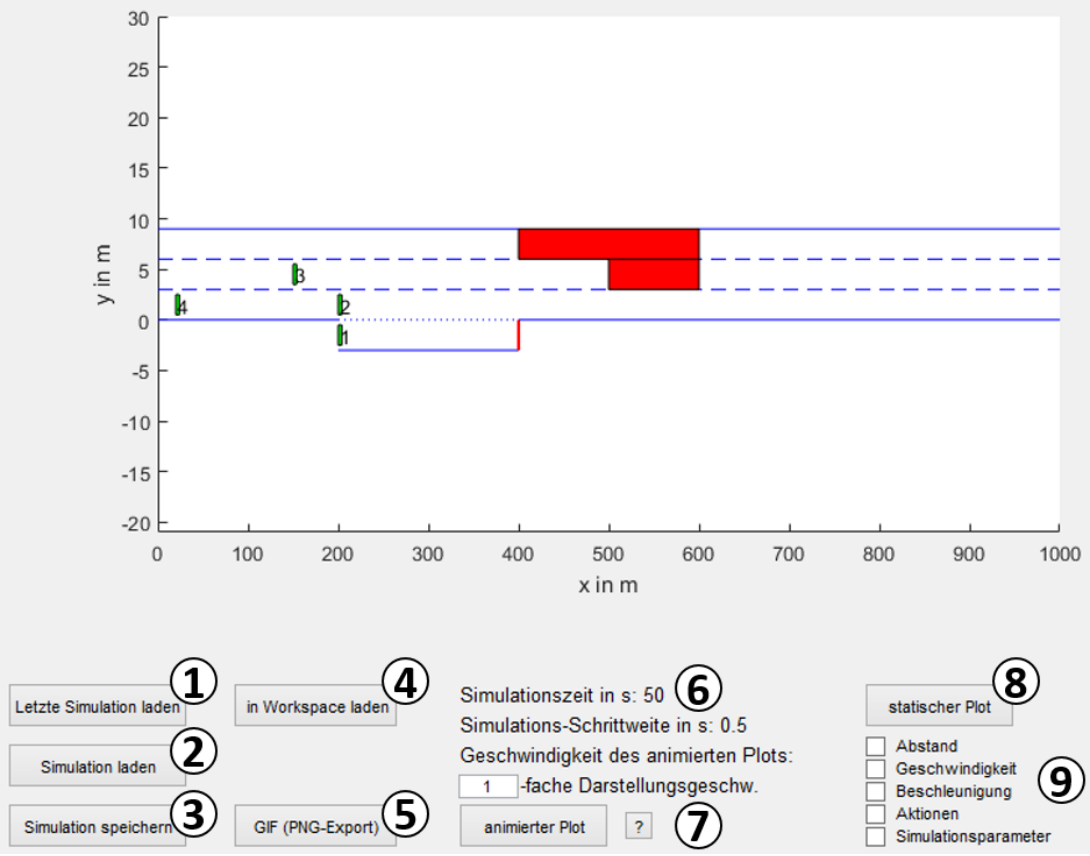


Abb. 3.11 Aufbau der GUI zur Visualisierung und Auswertung der Berechnungsergebnisse

Mit dem obersten Button auf der linken Seite (bei Markierung 1) lassen sich stets die Ergebnisse der letzten durchgeführten Simulation laden. Dazu ist es nicht nötig, diese abzuspeichern, bevor alte Ergebnisse geladen werden. Damit können die aktuellen Ergebnisse schnell und einfach mit alten verglichen werden, ohne dass diese erst gespeichert

und anschließend wieder geladen werden müssen. Werden die neuen Ergebnisse nicht explizit abgespeichert, werden sie bei der nächsten Simulation überschrieben. Die darunterliegende Schaltfläche (2) dient dazu, alte Simulationsergebnisse in die GUI zu laden. Standardmäßig werden Simulationsergebnisse im Ordner Matlab\03_Simulationsergebnisse gespeichert, weshalb das Lade-Fenster in diesem Ordner öffnet. Bei der Auswahl von alten Simulationsergebnissen ist wichtig, dass nur die Dateien mit der Endung „_SimRes.mat“ geöffnet werden können. Mit der letzten Schaltfläche in der linken Reihe (3) lässt sich das aktuell in der GUI geladene Simulationsergebnis abspeichern. Soll das neue Ergebnis gespeichert werden und befindet sich aber aktuell ein altes Ergebnis in der GUI, muss erst „Letzte Simulation laden“ angeklickt werden. Anschließend kann das neue Simulationsergebnis gespeichert werden. Der Standardname, der für das Speichern vorgegeben wird, setzt sich aus dem Datum und der Uhrzeit des Speicherzeitpunktes und der Endung „_SimRes.mat“ zusammen. Zusätzlich wird die dazugehörige GUI-Datei abgelegt, ebenfalls mit Datum und Uhrzeit sowie der Endung „_savedGUIdata.mat“. Auf diese Weise kann ein bereits berechnetes Verkehrsszenario ohne Umwege in den Verkehrsszenario-Editor oder in die Simulationsparameter-GUI geladen werden. Mit dem Button „in Workspace laden“ (4) werden die Variablen, die sich in der Datei mit den Berechnungsergebnissen befinden, in den Matlab-Workspace geladen. Damit kann eine detaillierte Untersuchung des Lösungsvektors und aller mitgeloggten Daten des Simulationsverlaufs stattfinden. Das sind unter anderem die Anzahl der expandierten Knoten und die Zeitdauer für den Durchlauf einer Iterationsschleife. Alle Simulationsparameter, Gewichtungsfaktoren und IDM-Parameter können ebenfalls eingesehen werden.

Das Verkehrsszenario mit dem berechneten kooperativen Fahrmanöver kann als Einzelbilder exportiert werden. Die Bilder werden im png-Format in einen Ordner mit dem Datum und der Uhrzeit als Ordnernamen kopiert. Der Pfad ist Matlab\04_GIFs. Aus diesen Einzelbildern kann mit einer entsprechenden Software eine GIF-Datei erzeugt werden, um sie bspw. in PowerPoint einzubinden. Vor dem Export der Bilder ist anzugeben, in welchem zeitlichen Abstand diese erzeugt werden sollen. Empfohlen wird, maximal die Schrittweite der Simulation selbst auszuwählen. Ein kleinerer Zeitschritt ist problemlos möglich, solange sich der Simulationszeitschritt durch den gewählten ohne Rest teilen lässt. Bei größeren Zeitschritten als der Simulationsschrittweite für den Bildexport können Fehler auftreten. In der GUI werden die Parameter Simulationszeit und Simulationsschrittweite angezeigt (6). Sie dienen als Orientierung für die animierte Darstellung der kooperativen Fahrmanöver (7). In dem Textfeld über der Schaltfläche kann angegeben werden, ob die Visualisierung in Echtzeit (Eingabe: 1) oder bspw. nur halb so schnell (Eingabe: 0.5) oder doppelt so schnell (Eingabe: 2) ablaufen soll. Abhängig von der Performance des PCs wird bei einer gewissen Darstellungs-geschwindigkeit das Limit des Matlab-Programms hinsichtlich der Plot-Geschwindigkeit erreicht. Schneller kann dann die Fahrzeugbewegung nicht mehr dargestellt werden. Die kooperativen Fahrmanöver können in der GUI zudem statisch angezeigt werden (8). Dazu werden die Trajektorien der Fahrzeuge übereinander abgebildet und der Fahrweg jedes Fahrzeugs über den gesamten Simulationszeitraum wird sichtbar. Mit den darunterliegenden Checkboxen können weitere Größen ausgewählt werden. Der Abstand für jedes Fahrzeug ist die Distanz zwischen dem Ego-Fahrzeug und dem nächsten vorausliegenden Fahrzeug oder Hindernis. Bei einem Spurwechsel wird der geringere Abstand zum vorausliegenden Fahrzeug/Hindernis beider Spuren zum jeweiligen Zeitpunkt der Simulation angezeigt. Sprünge im Abstands-Plot lassen sich darauf zurückführen, dass entweder das Ego-Fahrzeug oder ein Fahrzeug vor ihm die Spur wechselt. Befindet sich kein Fahrzeug oder Hindernis voraus, wird der Abstand mit 1000 m angegeben, um zu vermeiden, dass ein Abstand von

Unendlich angezeigt wird. Der Abstand wird für jedes Fahrzeug in einem separaten Fenster angezeigt. Die Geschwindigkeit der Fahrzeuge wird ebenfalls über die Simulationszeit auf der x-Achse aufgetragen. Alle Fahrzeuggeschwindigkeiten werden in einem Fenster abgebildet, weil sich die Werte im gleichen Größenbereich bewegen und so eine gute Vergleichbarkeit möglich ist. Die Beschleunigungs- und Verzögerungswerte werden für jedes Fahrzeug separat in einem Fenster dargestellt, weil darunter sonst die Übersichtlichkeit aufgrund der gleichen Standardaktionen pro Fahrzeugtyp leidet. Die Werte werden über der Simulationszeit auf der x-Achse aufgetragen. Wird die Checkbox „Aktionen“ ausgewählt, wird für jedes Fahrzeug angezeigt, welche Aktion aus den wählbaren Standardaktionen $\boldsymbol{a}_{act,def}$ zu jedem Simulationsschritt jeweils durchgeführt wurde. Zusätzlich dazu wird ein Tortendiagramm für jedes Fahrzeug erstellt, dass den Anteil angibt, wie oft eine bestimmte Standardaktion bezogen auf die gesamte Simulation ausgeführt wurde. Zu beachten ist, dass der ermittelte Anteil nicht auf die Simulationszeit bezogen wird, sondern auf die getroffenen Entscheidungen für eine bestimmte Aktion. Das heißt, dass ein Spurwechsel, der mehrere Sekunden und damit über mehrere Zeitschritte der Simulation andauert, nur als eine Aktion zählt und nicht pro Zeitschritt in dem er stattfindet. Das Tortendiagramm für Fahrzeug 1 aus dem Verkehrsszenario in Abb. 3.3 ist in Abb. 3.12 dargestellt.

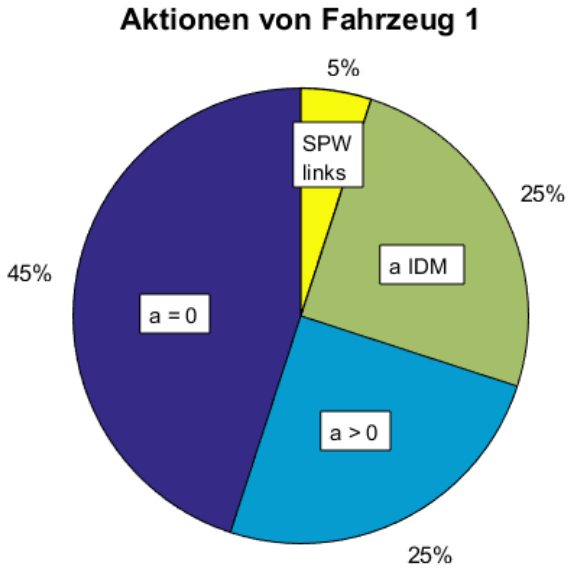


Abb. 3.12 Tortendiagramm mit dem Anteil der jeweiligen Standardaktion an der Menge aller durchgeführten Aktionen während der Simulation

Mit der letzten Checkbox „Simulationsparameter“ wird die Anzahl der expandierten Knoten pro Iterationsschleife auf der x-Achse und die Zeitdauer pro Iterationsschleife auf der y-Achse geplottet. Daraus ergibt sich eine Übersicht über die Geschwindigkeit des Algorithmus, die sich für Performance-Vergleiche eignet. Weiterhin lässt sich damit feststellen, ob es Einbrüche oder Veränderungen in der Rechengeschwindigkeit gegeben hat.

4 Ergebnisse und Diskussion

Im den Hauptteil abschließenden Kapitel dieser Arbeit werden die Berechnungsergebnisse aus der Anwendung des entwickelten Algorithmus zur kooperativen Trajektorienplanung vorgestellt und analysiert. Dazu werden in Unterkapitel 4.1 zuerst einige beispielhafte Verkehrsszenarien definiert. Mit diesen soll in Unterkapitel 4.2 zum einen die grundsätzliche Funktionsfähigkeit des Algorithmus gezeigt werden. Zum anderen sollen Lösungen für typische Situationen im Straßenverkehr berechnet und detailliert erläutert werden. In Unterkapitel 4.3 werden die Ergebnisse analysiert. Vor allem die besonderen Eigenschaften des Algorithmus und deren Auswirkung auf die Berechnungsergebnisse werden aufgegriffen. Daran anschließend wird in Unterkapitel 4.4 die Performance des Matlab-Programms mit unterschiedlichen Simulationsparametern bewertet. Das letzte Unterkapitel (Unterkapitel 4.5) beschäftigt sich mit möglichen Ansatzpunkten zur Weiterentwicklung des vorgestellten Algorithmus zur kooperativen Trajektorienplanung. Das beinhaltet potentielle Verbesserungen und denkbare zusätzliche Funktionen sowie das weitere Vorgehen für die Problemstellung der kooperativen Trajektorienplanung.

4.1 Ausgewählte Verkehrsszenarien und deren Relevanz im Straßenverkehr

Zur Darstellung und Erläuterung der Funktionsfähigkeit des entwickelten Algorithmus werden vier beispielhafte Verkehrsszenarien festgelegt. Zwei davon sind einfach gehalten, um die grundsätzlichen Vorgänge anschaulich darlegen zu können. Außerdem kann damit gezeigt werden, in welcher Weise die Fahrzeuge aufeinander abgestimmte Fahrmanöver durchführen. Einfach bedeutet, dass nur wenige Fahrzeuge aktiv beteiligt sind. Die weiteren zwei Verkehrsszenarien sind deutlich komplexer und enthalten mehrere Fahrzeuge. Mit ihnen wird aufgezeigt, wie der Algorithmus diese Verkehrsszenarien löst und warum bestimmte Aktionen in bestimmten Situationen durchgeführt werden. Alle Verkehrsszenarien werden im vorgestellten Verkehrsszenario-Editor erstellt. Alle verwendeten Einstellungen und Parameter sind auf dem beiliegenden Datenträger zu finden. Im Folgenden werden nur die wichtigsten erwähnt.

Mit Verkehrsszenario 01 soll das Finden eines kooperativen Fahrmanövers anschaulich und übersichtlich dargestellt werden. Deshalb sind nur zwei Fahrzeuge beteiligt. Betrachtet wird eine Situation, in der Fahrzeug 1 auf einer Fahrspur fährt, auf der sich ein Hindernis befindet. In diesem Fall handelt es sich um die Beschleunigungsspur, die an einer gewissen x-Position endet. Auf der links danebenliegenden Fahrspur (rechte Fahrspur der Fahrbahn) fährt Fahrzeug 2. Auf dieser Fahrspur befindet sich kein Hindernis. Die initiale Geschwindigkeit von Fahrzeug 1 beträgt 80 km/h und die Wunschgeschwindigkeit 120 km/h. Fahrzeug 2 startet mit 140 km/h, was auch der gewünschten Geschwindigkeit entspricht. Fahrzeug 2 ist zu Beginn 50 m hinter Fahrzeug 1. Für die Berechnung werden vier Varianten dieses Verkehrsszenarios erstellt. Zwei Varianten enthalten zwei Fahrspuren, so dass Fahrzeug 2 auf diese ausweichen

könnte. Die anderen beiden Varianten beinhalten nur die eine Fahrspur, auf der sich Fahrzeug 2 bereits befindet. Der Unterschied zwischen zwei Varianten mit gleicher Anzahl an Fahrspuren ist die x-Position des Fahrzeugs 2. Diese ist jeweils um 50 m nach hinten verschoben. Das Verkehrsszenario mit zwei Fahrspuren und der ursprünglichen x-Position von Fahrzeug 2 ist in Abb. 4.1 dargestellt.

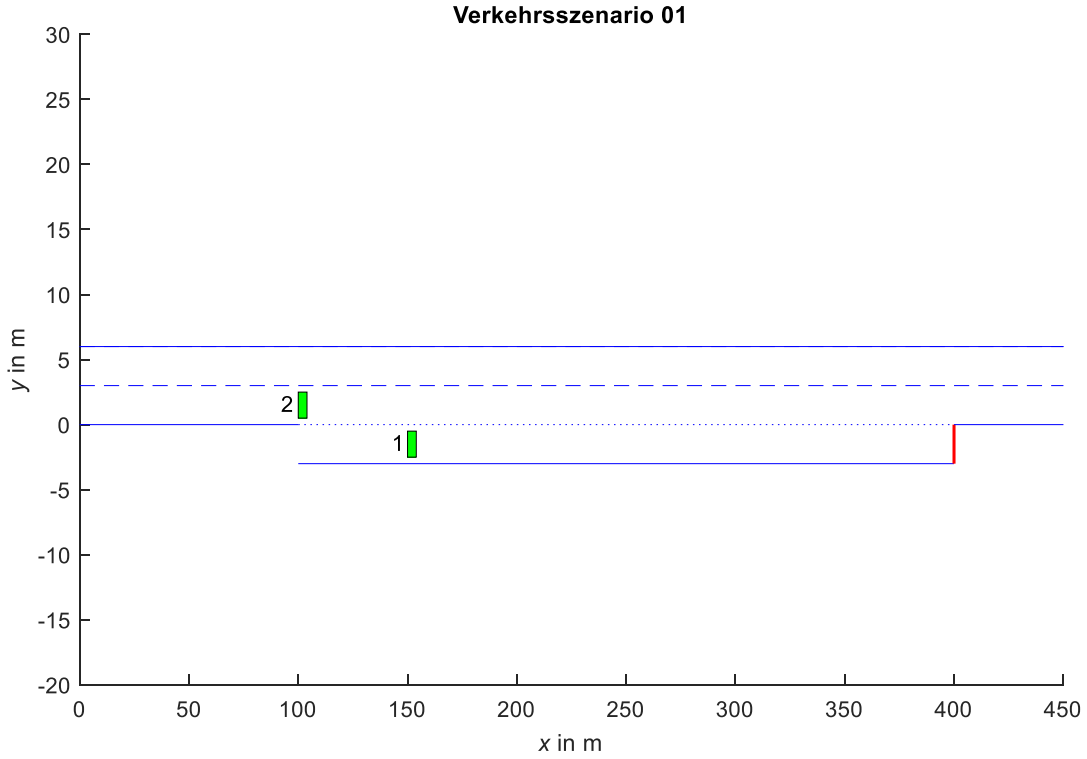


Abb. 4.1 Verkehrsszenario 01 mit Fahrzeug 1 und 2

Die initialen Fahrzeugzustände der Fahrzeuge 1 und 2 und die eingestellten Simulationsparameter für das Verkehrsszenario 01 sind in Tabelle 4.1 aufgelistet.

Tabelle 4.1 Initiale Fahrzeugzustände und Simulationsparameter für Verkehrsszenario 01

Formelzeichen	Einheit	Fahrzeug 1	Fahrzeug 2
$x_i(t_0)$	m	150	100 (50)
$y_{FSP,i}(t_0)$	-	0	1
$v_i(t_0)$	km/h	80	140
$v_{wunsch,i}$	km/h	120	140
t_{end}	s	20	
Δt	s	0,5	
Anzahl Iterationen	-	1500	

Situationen wie in Verkehrsszenario 01 treten auf mehrspurigen Fahrbahnen häufig auf. Das Auffahren auf ein langsameres – oder sogar stehendes – Hindernis und das Rücksichtnehmen auf den rückwärtigen Verkehr für einen notwendigen Spurwechsel sind ein wichtiger Aspekt der Kooperation zwischen den Fahrzeugen.

Auf eine ähnliche Situation zielt auch Verkehrsszenario 02 ab. Das Fahrbahnlayout ist identisch der Variante von Verkehrsszenario 01 mit zwei Fahrspuren. Beteiligt sind drei Fahrzeuge, davon sind zwei Pkw und eines ein Lkw. Pkw werden in der graphischen Darstellung jeweils durch grüne Rechtecke dargestellt, Lkw durch gelbe Rechtecke. Nur die beiden Pkw sind aktiv an dem Verkehrsszenario beteiligt. Dem Lkw (Fahrzeug 3) wird nur die Aktion $a_{=0}$ zur Auswahl

gestellt, ihm wird folglich eine konstante Geschwindigkeit fest zugewiesen. Diese Geschwindigkeit beträgt 100 km/h und entspricht der Wunschgeschwindigkeit. Die Startposition in x-Richtung ist bei 250 m. Fahrzeug 1 und Fahrzeug 2 befinden sich an der gleichen Position wie in Verkehrsszenario 01. Für Fahrzeug 2 ist die Start- und Wunschgeschwindigkeit jetzt allerdings bei 120 km/h. Das definierte Verkehrsszenario 02 ist in Abb. 4.2 dargestellt.

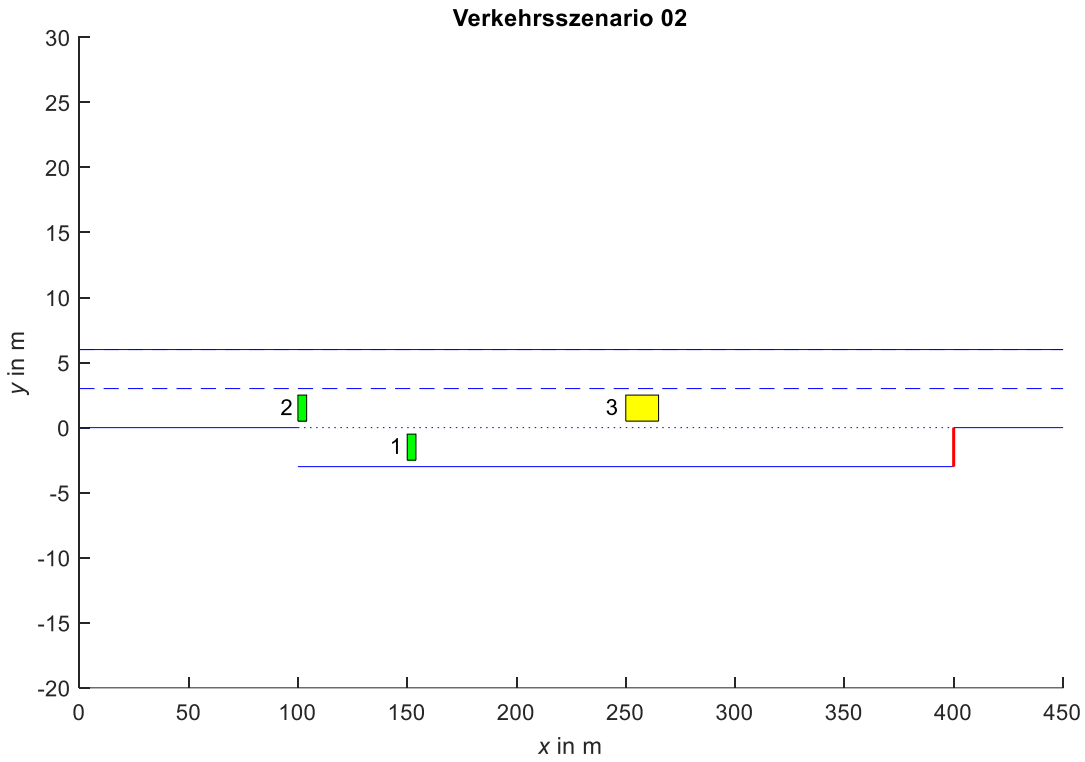


Abb. 4.2 Verkehrsszenario 02 mit Fahrzeug 1 bis 3

Die initialen Fahrzeugzustände der Fahrzeuge 1 bis 3 und die eingestellten Simulationsparameter für das Verkehrsszenario 02 sind in Tabelle 4.2 aufgelistet.

Tabelle 4.2 Initiale Fahrzeugzustände und Simulationsparameter für Verkehrsszenario 02

Formelzeichen	Einheit	Fahrzeug 1	Fahrzeug 2	Fahrzeug 3
$x_i(t_0)$	m	150	100	250
$y_{FSP,i}(t_0)$	-	0	1	1
$v_i(t_0)$	km/h	80	120	100
$v_{wunsch,i}$	km/h	120	120	100
t_{end}	s	50		
Δt	s	0,5		
Anzahl Iterationen	-	1500		

In diesem Verkehrsszenario spielt ebenfalls das Auffahren auf ein langsameres Hindernis eine große Rolle. Zum einen befindet sich Fahrzeug 1 auf der Beschleunigungsspur, die an einem gewissen Punkt endet. Zum anderen bewegte sich Fahrzeug 3 langsamer als die beiden anderen Fahrzeuge und muss somit überholt werden. Dadurch, dass nur eine Fahrspur für den Überholvorgang frei ist, müssen sich die Fahrzeuge 1 und 2 abstimmen und miteinander kooperieren. Fahrzeug 3 hat keine Möglichkeit zu reagieren, weil es nur eine Aktion wählen kann.

Verkehrsszenario 03 beschreibt eine Situation, die eine Kooperation von mehreren Fahrzeugen erfordert. Das Fahrbahnlayout entspricht denen der beiden vorherigen Verkehrsszenarien.

Beteiligt sind sechs Fahrzeuge, davon sind vier Fahrzeuge Pkw und zwei Fahrzeuge Lkw. Fahrzeug 4 ist ein Lkw, der nur die Aktion $a_{=0}$ durchführen kann und deshalb mit einer konstanten Geschwindigkeit von 60 km/h fährt. Dieser befindet sich vor allen anderen Fahrzeugen und stellt ein bewegtes Hindernis dar. Fahrzeug 1 befindet sich auf der Beschleunigungsspur und hat eine Startgeschwindigkeit von 80 km/h sowie eine Wunschgeschwindigkeit von 160 km/h. Alle Fahrzeuge bis auf das Fahrzeug 4 haben die Möglichkeit, aus dem vollständigen Satz an Aktionen zu wählen. Das Verkehrsszenario 03 ist in Abb. 4.3 dargestellt.

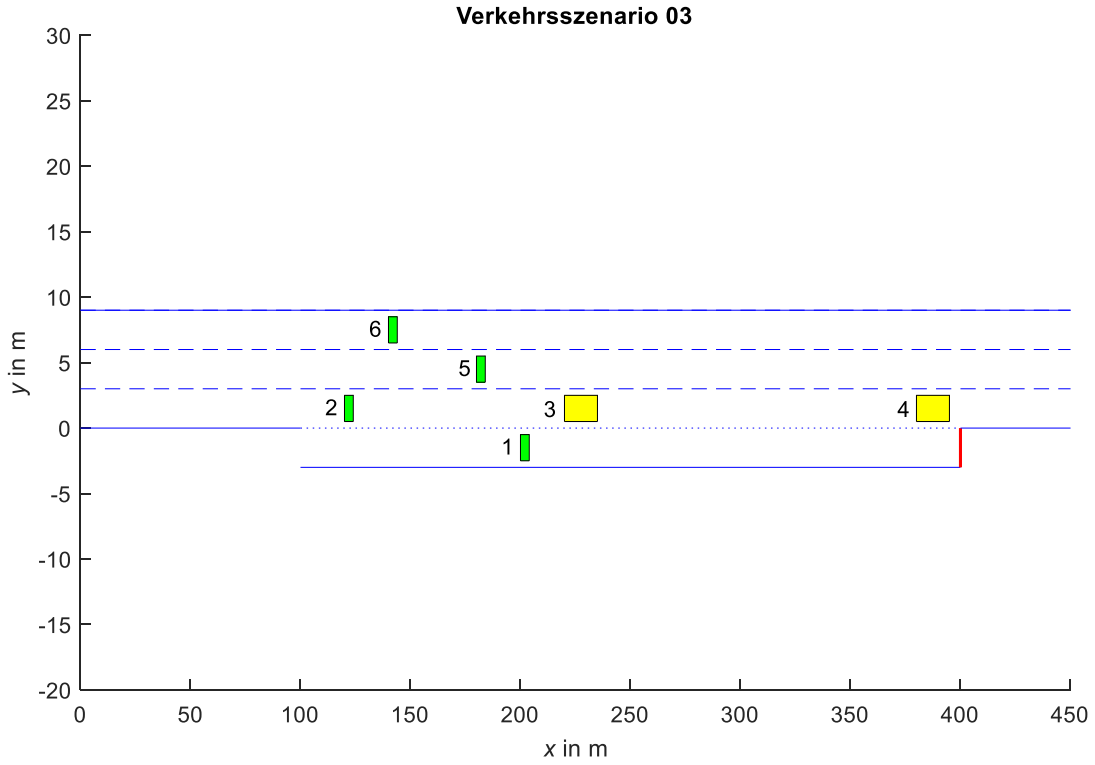


Abb. 4.3 Verkehrsszenario 03 mit Fahrzeug 1 bis 6

Die initialen Fahrzeugzustände der Fahrzeuge 1 bis 6 und die eingestellten Simulationsparameter für das Verkehrsszenario 03 sind in Tabelle 4.3 aufgelistet.

Tabelle 4.3 Initiale Fahrzeugzustände und Simulationsparameter für Verkehrsszenario 03

Formelzeichen	Einheit	Fzg 1	Fzg 2	Fzg 3	Fzg 4	Fzg 5	Fzg 6
$x_i(t_0)$	m	200	120	220	380	180	130
$y_{FSP,i}(t_0)$	-	0	1	1	1	2	3
$v_i(t_0)$	km/h	80	120	100	60	120	150
$v_{wunsch,i}$	km/h	160	120	100	60	120	150
t_{end}	s	40					
Δt	s	1					
Anzahl Iterationen	-	50					

In Verkehrsszenario 03 muss Fahrzeug 1 von der Beschleunigungsspur auf die rechte Fahrspur wechseln und hat gleichzeitig den höchsten Wert der Wunschgeschwindigkeit. Weil Fahrzeug 4 am langsamsten fährt, müssen alle anderen beteiligten Fahrzeuge diesen Lkw passieren. Gleiches gilt für alle Pkw, die schneller fahren und eine höhere Wunschgeschwindigkeit besitzen als beide Lkw. Die Anzahl der Iterationen wurde im Vergleich zu den vorherigen Berechnungen reduziert, weil durch die gestiegene Anzahl an Fahrzeugen der Rechenaufwand erheblich ansteigt.

Das letzte betrachtete Verkehrsszenario 04 greift das Vorgehen von Lenz [21] auf. In seiner Arbeit wird ein kooperatives Fahrmanöver für eine Einfädelsituation ähnlich der aus Verkehrsszenario 01 berechnet, jedoch mit mehreren Fahrzeugen auf den beiden Fahrspuren. Bei ihm können nur das Ego-Fahrzeug und andere ausgewählte Fahrzeuge ihre Aktionen selbst bestimmen. Die anderen beteiligten Fahrzeuge sind lediglich in der Lage, nach dem IDM und dem daraus resultierenden Beschleunigungs- oder Verzögerungswert zu handeln. Sie können also nicht agieren, sondern nur auf die aktuelle Verkehrssituation reagieren. Im definierten Verkehrsszenario 04 können Fahrzeug 1 und Fahrzeug 2 aus dem vollständigen Set an Aktionen wählen. Alle anderen Fahrzeuge können nur die durch das IDM vorgegebene Beschleunigung ausführen. Alle Pkw bis auf Fahrzeug 1 besitzen eine Start- und Wunschgeschwindigkeit von 120 km/h. Beide Lkw besitzen eine Start- und Wunschgeschwindigkeit von 100 km/h. Fahrzeug 1 startet mit 80 km/h und hat eine gewünschte Geschwindigkeit von 140 km/h. Das Verkehrsszenario 04 ist in Abb. 4.4 dargestellt.

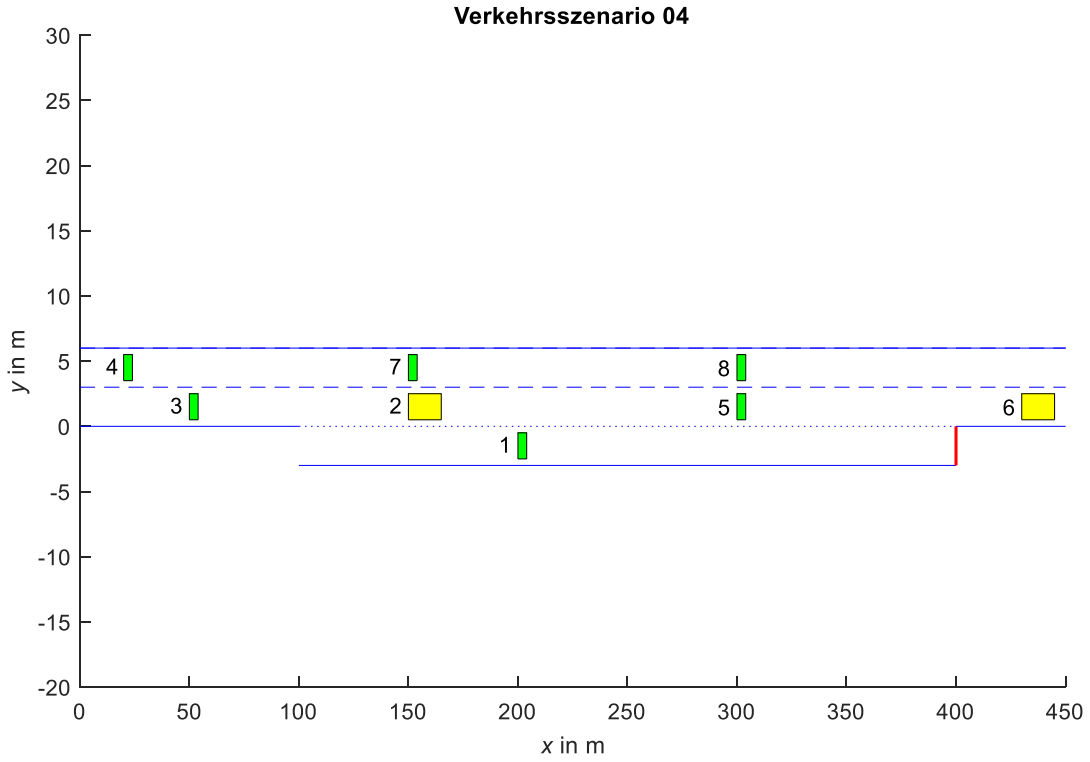


Abb. 4.4 Verkehrsszenario 04 mit Fahrzeug 1 bis 8

Die initialen Fahrzeugzustände der Fahrzeuge 1 bis 8 und die eingestellten Simulationsparameter für das Verkehrsszenario 04 sind in Tabelle 4.4 aufgelistet.

Tabelle 4.4 Initiale Fahrzeugzustände und Simulationsparameter für Verkehrsszenario 04

Formelzeichen	Einheit	Fzg 1	Fzg 2	Fzg 3	Fzg 4	Fzg 5	Fzg 6	Fzg 7	Fzg 8
$x_i(t_0)$	m	200	150	50	20	300	430	150	300
$y_{FSP,i}(t_0)$	-	0	1	1	2	1	1	2	2
$v_i(t_0)$	km/h	80	100	120	120	120	100	120	120
$v_{wunsch,i}$	km/h	140	100	120	120	120	100	120	120
t_{end}	s	30							
Δt	s	1							
Anzahl Iterationen	-	1000							

Mit Verkehrsszenario 04 soll gezeigt werden, wie der umliegende Verkehr auf eine Störung im Verkehrsfluss reagiert, wenn nur das IDM benutzt wird. Durch die höhere gewünschte Geschwindigkeit von 140 km/h des Fahrzeugs 2 soll diese zusätzliche Störung eingebracht werden. Auf diese Weise soll anschaulich gezeigt werden, wie die Fahrzeuge darauf reagieren. Weil für alle Fahrzeuge die Start- gleich der Wunschgeschwindigkeit ist, ist der initiale Beschleunigungswert für alle Fahrzeuge außer Fahrzeug 1 und 2 nur vom Abstand zum vorausfahrenden Fahrzeug abhängig. Bei einem Spurwechselvorgang in fließendem Verkehr mit geringen Geschwindigkeitsdifferenzen ist die ausschließliche Verwendung des IDM außerdem interessant, weil es auch für Verkehrssimulationen verwendet wird [50].

4.2 Vorstellung der Berechnungsergebnisse

In diesem Unterkapitel werden die Ergebnisse der Berechnung der kooperativen Fahrmanöver für die Verkehrsszenarien aus Unterkapitel 4.1 vorgestellt. Die resultierenden Trajektorien werden beschrieben und Auffälligkeiten hervorgehoben. Eine Analyse, Bewertung und Interpretation der Simulationsergebnisse erfolgt erst in Unterkapitel 4.3. Die detaillierten Plots der Fahrzeugzustände und alle aus der Simulation resultierenden Daten sind dem beiliegenden Datenträger zu entnehmen. Vor allem bei Verkehrsszenarien mit mehreren Fahrzeugen wird darauf verwiesen, weil eine übersichtliche Darstellung der Fahrzeugzustände, insbesondere der Trajektorien, nicht mehr möglich ist. Die Diagramme aller Fahrzeuggrößen, die in der Ausarbeitung erwähnt werden, sind zusätzlich im Anhang dieser Arbeit hinterlegt.

Für Verkehrsszenario 01 wurden die vier beschriebenen Varianten mit den identischen Simulationsparametern berechnet. Bei den beiden Varianten mit zwei Fahrspuren wechselt das Fahrzeug 2 stets auf die linke Fahrspur, um Fahrzeug 1 Platz zu machen. Die Trajektorien der Fahrzeuge für die ursprüngliche Variante des Verkehrsszenario 01 sind in Abb. 4.5 gezeigt.

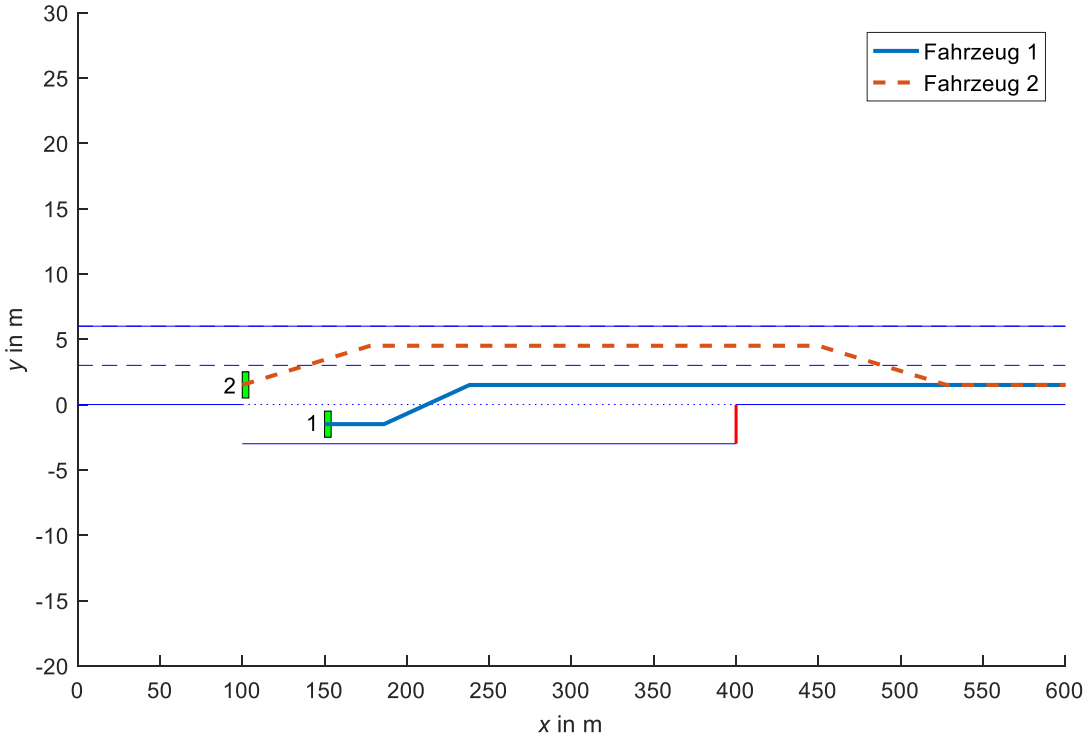


Abb. 4.5 Berechnete Fahrzeug-Trajektorien für Verkehrsszenario 01 (zwei Fahrspuren, $x_2(t_0) = 100$ m)

Fahrzeug 1 kann aufgrund des Spurwechsels von Fahrzeug 2 selbst einen Spurwechsel von der Beschleunigungsspur auf die rechte Fahrspur der Fahrbahn durchführen, ohne einen geringen Abstand zwischen beiden Fahrzeugen hervorzurufen. Fahrzeug 2 behält während des gesamten Fahrmanövers seine initiale Geschwindigkeit bei, was gleichzeitig der Wunschgeschwindigkeit entspricht (Plot im Anhang). Fahrzeug 1 beschleunigt von 80 km/h auf die Wunschgeschwindigkeit 120 km/h. Eine Unterbrechung ist nur während des Spurwechsels zu erkennen, weil dieser per Definition mit konstanter Geschwindigkeit durchgeführt wird. Der Spurwechsel beginnt kurz bevor Fahrzeug 2 die rechte Fahrspur verlassen hat. In der Zeit bis zum Spurwechsel beschleunigt Fahrzeug 1 auf der Beschleunigungsspur. Erkennbar ist das Ausweichen von Fahrzeug 2 auf die freie, linke Fahrspur im Wert für den Abstand s_{gap} zwischen Fahrzeug 1 und dem nächsten Fahrzeug/Hindernis. Nach dem Überholen von Fahrzeug 1 schert Fahrzeug 2 wieder auf die rechte Fahrspur ein. Der Spurwechsel von Fahrzeug 1 zurück auf die ursprüngliche Fahrspur beginnt bei ca. 40 m Abstand zwischen beiden Fahrzeugen, wie in Abb. 4.6 zu erkennen ist.

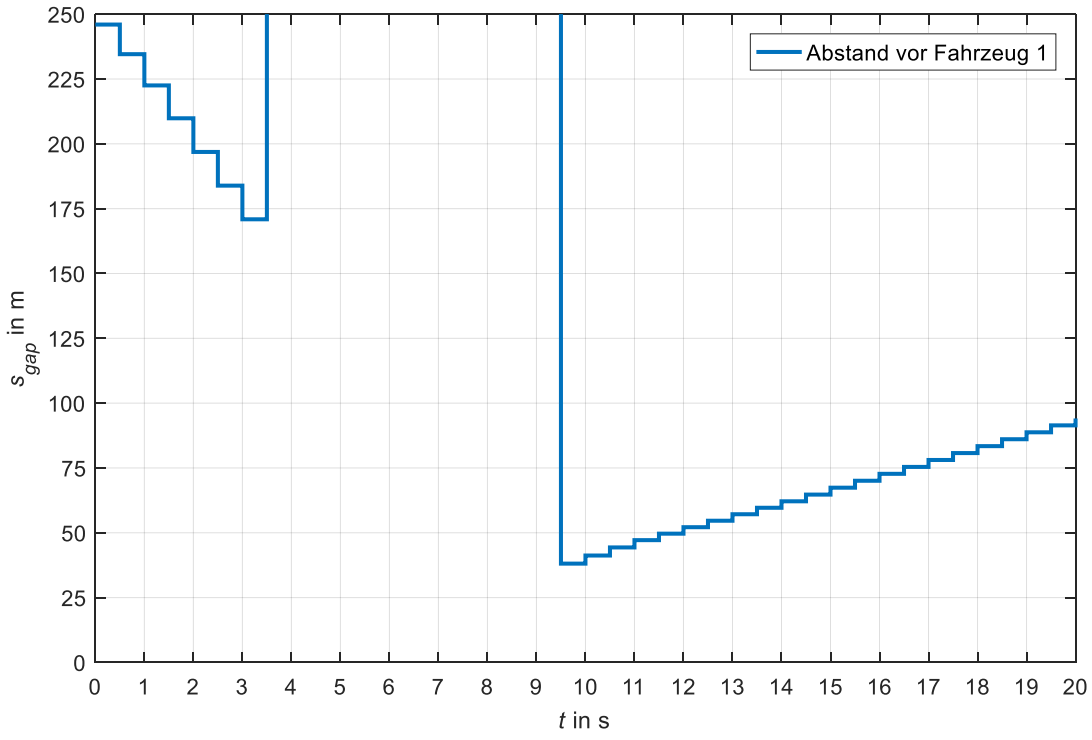


Abb. 4.6 Abstandswert s_{gap} von Fahrzeug 1 in Verkehrsszenario 01 (zwei Fahrspuren, $x_2(t_0) = 100$ m)

Der Abstand s_{gap} wird unmittelbar nach dem Spurwechsel zurück stetig größer, weil Fahrzeug 2 um 20 km/h schneller fährt als Fahrzeug 1. Am Anfang der Simulation befindet sich das Ende der Beschleunigungsspur vor Fahrzeug 1, was durch den geringer werdenden Abstand zwischen 0 s und 4 s zu erkennen ist. Der Bereich oberhalb von 250 m Abstand wurde aus Gründen der Übersichtlichkeit abgeschnitten. Er ist nicht relevant, weil in diesem Fall ein y-Wert von 1000 m vorliegt, was bedeutet, dass sich kein Fahrzeug voraus befindet.

Im Falle von nur einer Fahrspur ist eine stärkere Kooperation der beiden Fahrzeuge notwendig (Plots im Anhang). Bei der Konfiguration mit einer x-Startposition von 100 m für Fahrzeug 2 fährt dieses weiterhin mit konstanter Geschwindigkeit auf der einzig verbleibenden Fahrspur. Fahrzeug 1 reagiert darauf mit einem längeren Fahren auf der Beschleunigungsspur. Es beschleunigt bis ca. 100 km/h, fährt dann solange mit dieser Geschwindigkeit, bis Fahrzeug 2 vorbeigefahren ist und wechselt anschließend die Fahrspur. Nach erfolgtem Spurwechsel

beschleunigt Fahrzeug 1 auf die Wunschgeschwindigkeit von 120 km/h. Der Abstand vor Fahrzeug 1 zum jeweiligen nächsten Hindernis ist in Abb. 4.7 abgebildet. Sichtbar wird das längere Fahren auf dem Beschleunigungsstreifen und dem dadurch geringeren Abstand zu dessen Ende. Der Spurwechsel beginnt bei ca. 100 m Abstand vor dem Endpunkt. Der starke Sprung im Abstand resultiert aus dem Spurwechsel, der dazu führt, dass Fahrzeug 1 hinter Fahrzeug 2 in die Fahrspur einschert. Aufgrund der Geschwindigkeitsdifferenz wächst der Abstand stetig an. Zu Beginn wächst er stärker, weil Fahrzeug 1 mit 100 km/h von der Beschleunigungsspur kommt und erst auf der neuen Fahrspur weiter auf die Wunschgeschwindigkeit beschleunigen kann. Diese ist bei ca. 10 s erreicht, ab da wächst der Abstand in konstantem Maße an. Ab diesem Zeitpunkt fahren die beiden Fahrzeuge mit konstanter Geschwindigkeitsdifferenz hintereinander her.

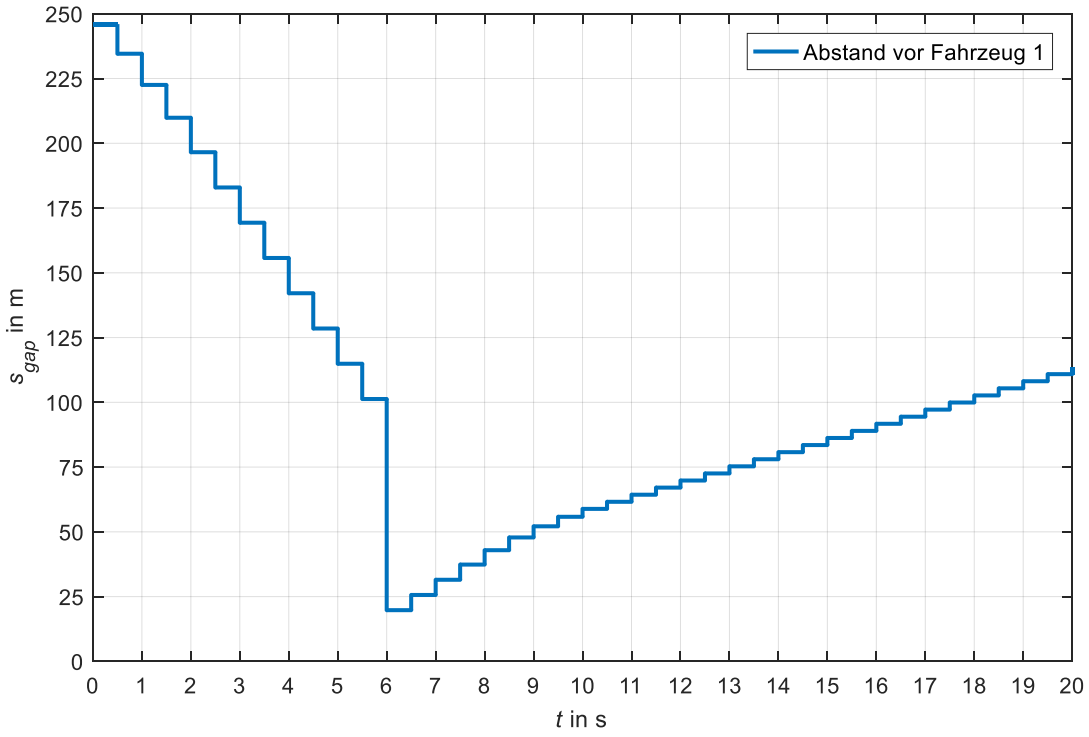


Abb. 4.7 Abstandswert s_{gap} von Fahrzeug 1 in Verkehrsszenario 01 (eine Fahrspur, $x_2(t_0) = 100$ m)

Bei der Variante mit einer Fahrspur, bei der Fahrzeug 2 um 50 m nach hinten versetzt ist, resultiert eine komplett unterschiedliche Trajektorie. In diesem Fall fährt Fahrzeug 2 nicht vorbei, sondern verzögert, um Fahrzeug 1 einfädeln zu lassen. Während Fahrzeug 2 verzögert, beschleunigt Fahrzeug 1 auf der Beschleunigungsspur. Auf diese Weise findet eine Reduzierung der Differenzgeschwindigkeit statt, so dass sie zum Ende des Spurwechselvorgangs nur noch ca. 4 km/h beträgt. Der Geschwindigkeitsverlauf ist in Abb. 4.8 dargestellt. Nachdem Fahrzeug 1 mit konstanter Geschwindigkeit die Spur gewechselt hat, erfolgt ein Beschleunigungsvorgang. Allerdings liegt der angestrebte Geschwindigkeitswert über der Wunschgeschwindigkeit von 120 km/h. Fahrzeug 2 folgt der Beschleunigung von Fahrzeug 1, bleibt jedoch unterhalb der gewünschten Geschwindigkeit von 140 km/h. Ab ca. 17 s fahren beide Fahrzeuge mit nahezu identischer, konstanter Geschwindigkeit. Der resultierende Abstand pendelt sich bei ungefähr 70 m ein. Fahrzeug 2 kann das andere Fahrzeug nicht überholen, weil nur eine Fahrspur vorhanden ist. Es muss gezwungenermaßen hinter dem langsameren Fahrzeug 1 herfahren. Auffällig sind die „Überschwinger“ in der Fahrzeuggeschwindigkeit, die bei Fahrzeug 1 nach ca. 7 s und bei Fahrzeug 2 nach ca. 7 s, 11 s und 16 s auftreten.

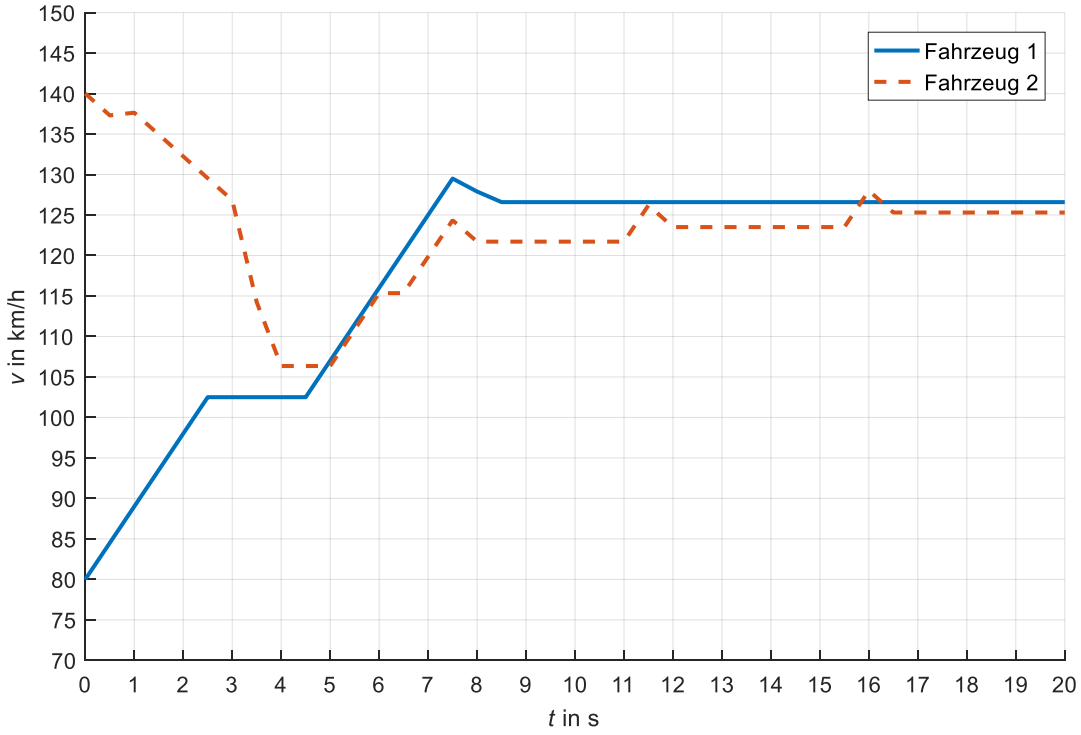


Abb. 4.8 Geschwindigkeitsverlauf der beiden Fahrzeuge in Verkehrsszenario 01 (eine Fahrspur, $x_2(t_0) = 50$ m)

Aus der Simulation für Verkehrsszenario 02 resultieren die Trajektorien, die in Abb. 4.9 gezeigt werden. Fahrzeug 1 und Fahrzeug 2 haben die gleichen initialen Zustände wie in Verkehrsszenario 01, bis auf die Start- und Wunschgeschwindigkeit von Fahrzeug 2, die jetzt 120 km/h ist. Fahrzeug 3, ein Lkw, fährt mit konstanter Geschwindigkeit auf der rechten Fahrspur.

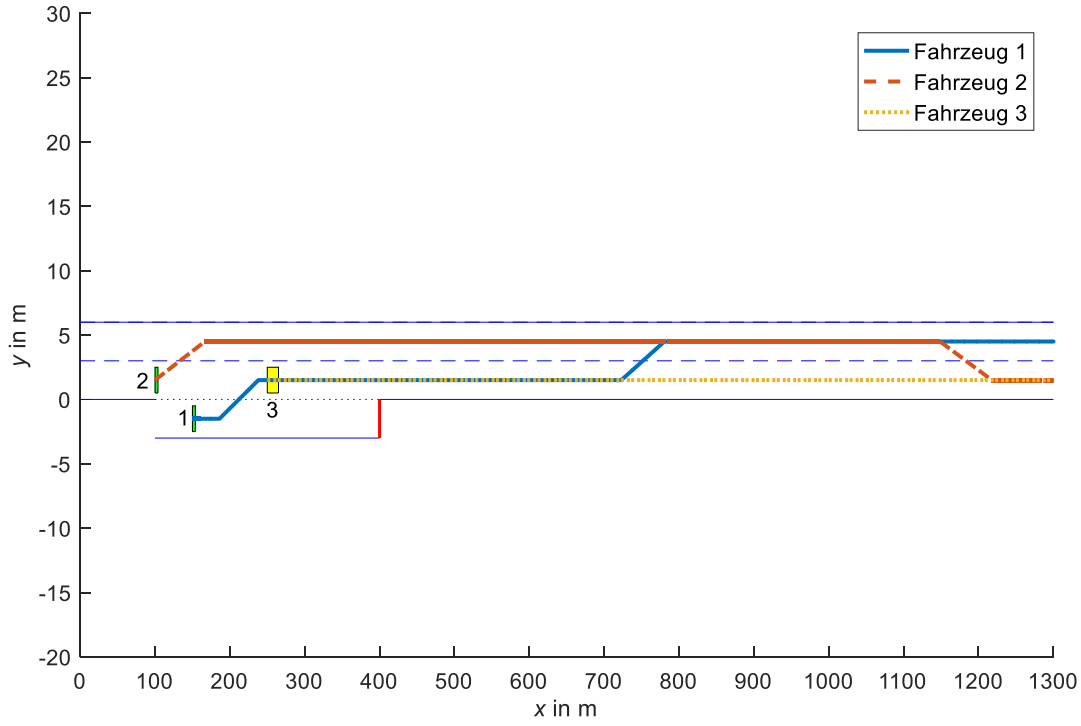


Abb. 4.9 Berechnete Fahrzeug-Trajektorien für Verkehrsszenario 02

Der Beginn der Trajektorien von Fahrzeug 1 und Fahrzeug 2 ist identisch mit Verkehrsszenario 01. Fahrzeug 2 wechselt sofort auf die linke Fahrspur, um Fahrzeug 1 den Platz zum

Spurwechsel zu bieten. Fahrzeug 2 fährt in Verkehrsszenario 02 nur 120 km/h, was auch der Wunschgeschwindigkeit entspricht. Letzteres trifft auch auf Fahrzeug 1 zu, das nach dem Spurwechsel weiter beschleunigt. Beide Fahrzeuge fahren fast nebeneinander mit gleicher Geschwindigkeit auf Fahrzeug 3 zu. Aufgrund des die linke Fahrspur blockierenden Fahrzeugs 2 beginnt Fahrzeug 1 zu verzögern und seine Geschwindigkeit nähert sich der des Lkw an. Zu diesem Zeitpunkt (bei ca. 20 s) erhöht Fahrzeug 2 die Geschwindigkeit über die eigene Wunschgeschwindigkeit. Nachdem für Fahrzeug 1 auf der linken Spur ausreichend Platz und Abstand zu Fahrzeug 2 ist, wechselt es die Fahrspur. Beide Fahrzeuge fahren hintereinander. Hat Fahrzeug 2 den Lkw überholt und besteht ein ausreichender Abstand, vollzieht es einen Spurwechsel zurück auf die rechte Fahrspur. Daran anschließend wird die Geschwindigkeit wieder auf die Wunschgeschwindigkeit reduziert. Fahrzeug 1 fährt mit der erreichten Wunschgeschwindigkeit von 120 km/h auf Fahrspur 2. Ein Spurwechsel zurück auf die rechte Fahrspur findet nicht statt. Der beschriebene Geschwindigkeitsverlauf ist in Abb. 4.10 dargestellt. Wie bereits bei dem Geschwindigkeitsverlauf aus Verkehrsszenario 01 fallen „Überschwinger“ und zackige Verläufe auf.

Der Abstandswert für Fahrzeug 1 in Verkehrsszenario 02 ist in Abb. 4.11 dargestellt. Er ist zu Beginn des Fahrmanövers ähnlich zu Verkehrsszenario 01, weil sich das Ende der Beschleunigungsspur vor Fahrzeug 1 befindet. Nach dem Spurwechsel ist Fahrzeug 3 das relevante Hindernis. Am Abstand ist zu erkennen, dass die Geschwindigkeit von Fahrzeug 1 zuerst kleiner ist als von Fahrzeug 3 und dann größer. Der Abstand wächst zunächst an, bevor er geringer wird. Die Verringerung der Geschwindigkeitsdifferenz spiegelt sich im Abstand derart wider, dass die Rate mit der der Abstand abnimmt, geringer wird.

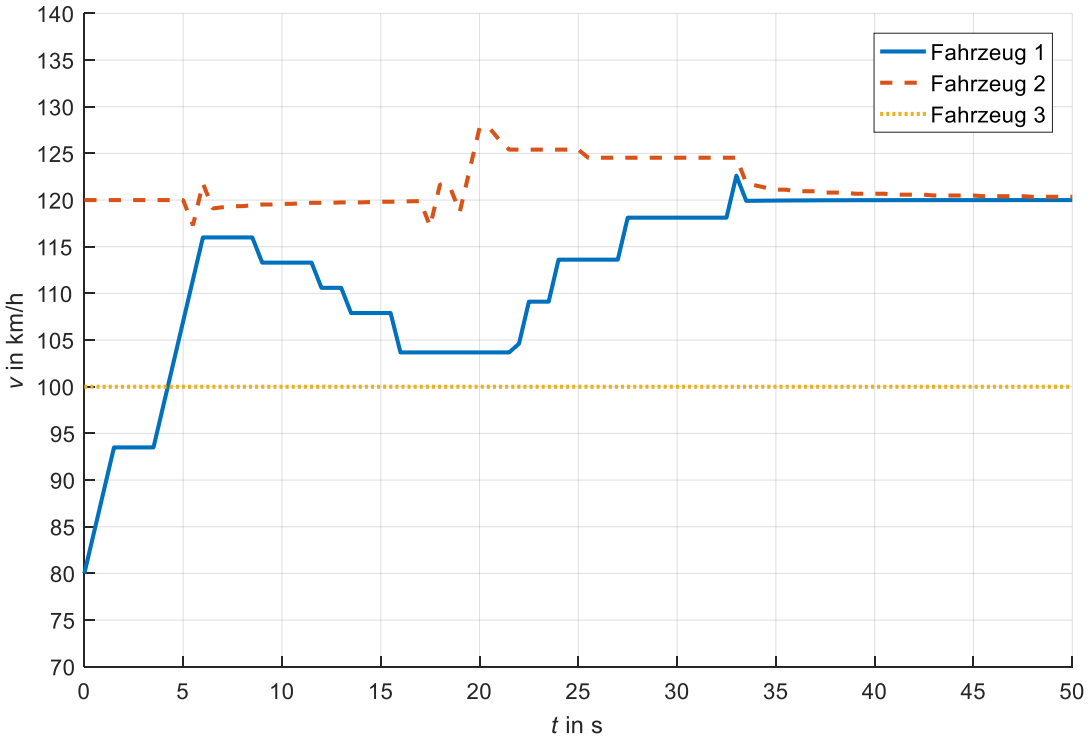


Abb. 4.10 Geschwindigkeitsverlauf der beteiligten Fahrzeuge in Verkehrsszenario 02

Nach dem Spurwechsel auf die linke Fahrspur nimmt der Abstand sprunghaft ab, weil nun Fahrzeug 2 das Hindernis ist, das zu Fahrzeug 1 näher ist. Durch die große Geschwindigkeitsdifferenz steigt der Abstand trotz der Beschleunigung von Fahrzeug 1 an. Mit Abnahme der Geschwindigkeitsdifferenz sinkt auch die Steigerungsrate des Abstandes.

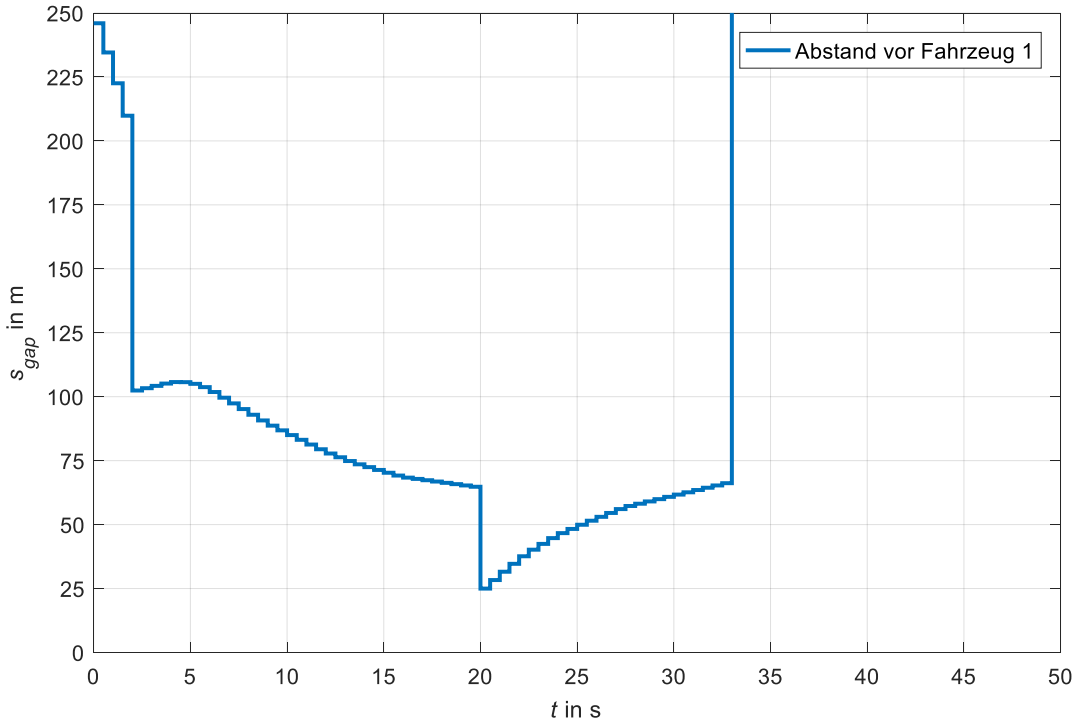


Abb. 4.11 Abstandswert s_{gap} von Fahrzeug 1 in Verkehrsszenario 02

Bei 33 s hat Fahrzeug 2 das Spruchwechselmanöver zurück auf die rechte Fahrspur beendet. Dadurch befindet sich kein Hindernis mehr vor Fahrzeug 1. Der Wert für den Abstand von Fahrzeug 1 zum nächsten Hindernis springt auf die vordefinierten 1000 m. Aus diesem Grund ist der Bereich der y-Achse oberhalb von 250 m nicht mehr relevant und daher abgeschnitten.

Verkehrsszenario 03 stellt das komplexeste, hier vorgestellte Verkehrsszenario dar. Zum einen sind sechs Fahrzeuge beteiligt, von denen fünf die Möglichkeit haben, aus dem vollständigen Set an Aktionen zu wählen. Nur Fahrzeug 4 besitzt eine konstante Geschwindigkeit. Zum anderen ermöglicht die dritte Fahrspur viel mehr Kombinationsmöglichkeiten als eine Fahrbahn mit nur zwei Fahrspuren. Aus beiden Gründen resultiert eine sehr hohe Anzahl an Kindknoten für jeden untersuchten Knoten. Deshalb wurde die Iterationszahl bei diesem Verkehrsszenario auf 50 Iterationen abgesenkt. Die Fahrzeug-Trajektorien, die sich aus der Simulation ergeben, sind in Abb. 4.12 dargestellt. Eine größere und übersichtlichere Darstellung kann dem beiliegenden Datenträger entnommen werden. Der zugehörige Verlauf der einzelnen Fahrzeug-Geschwindigkeiten ist Abb. 4.13 zu entnehmen. Fahrzeug 1 beginnt unmittelbar zu Beginn, die Beschleunigungsspur zu verlassen und auf die rechte Fahrspur zu wechseln. Auf der Fahrspur angekommen, erfolgt allerdings noch kein Beschleunigungsmanöver, sondern ein Fahren mit konstanter Geschwindigkeit und direkt im Anschluss ein weiterer Spurwechsel auf die mittlere Fahrspur. Erst dort erfolgt die Beschleunigung auf die Wunschgeschwindigkeit von 160 km/h. Nachdem alle langsameren Fahrzeuge auf der rechten Fahrspur überholt sind, wechselt Fahrzeug 1 auf diese zurück. Fahrzeug 2 behält während des gesamten Fahrmanövers seine Wunschgeschwindigkeit bei. Es wechselt direkt zu Beginn mit einem zweifachen Spurwechsel auf die linke Fahrspur und fährt dort so lange, bis es von Fahrzeug 1 rechts überholt wurde. Dann wechselt Fahrzeug 2 auf die mittlere Fahrspur. Nachdem genügend Abstand zum Lkw (Fahrzeug 4) vorahnden ist, erfolgt der Spurwechsel zurück auf die rechte Fahrspur.

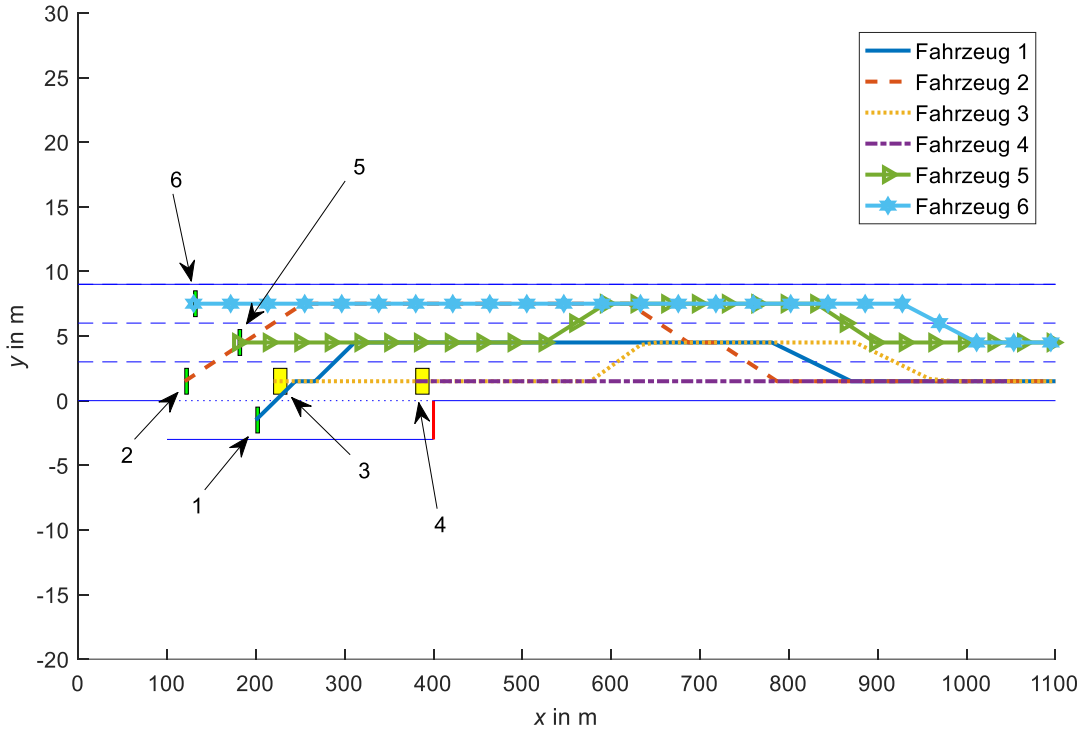


Abb. 4.12 Berechnete Fahrzeug-Trajektorien für Verkehrsszenario 03

Der zweite Lkw (Fahrzeug 3) verzögert hinter dem vorderen Lkw und wartet mit dem Spurwechsel, bis die Fahrzeuge auf der mittleren Fahrspur überholt haben. Dann folgt der Spurwechsel nach links. Erst danach beschleunigt der Lkw (Fahrzeug 3) wieder auf seine Wunschgeschwindigkeit. Sobald ausreichend Abstand zum überholten Lkw (Fahrzeug 4) vorliegt, wechselt er zurück auf die rechte Fahrspur. Fahrzeug 5 beschleunigt zu Beginn, als Fahrzeug 2 hinter ihm auf dieselbe Fahrspur wechselt.

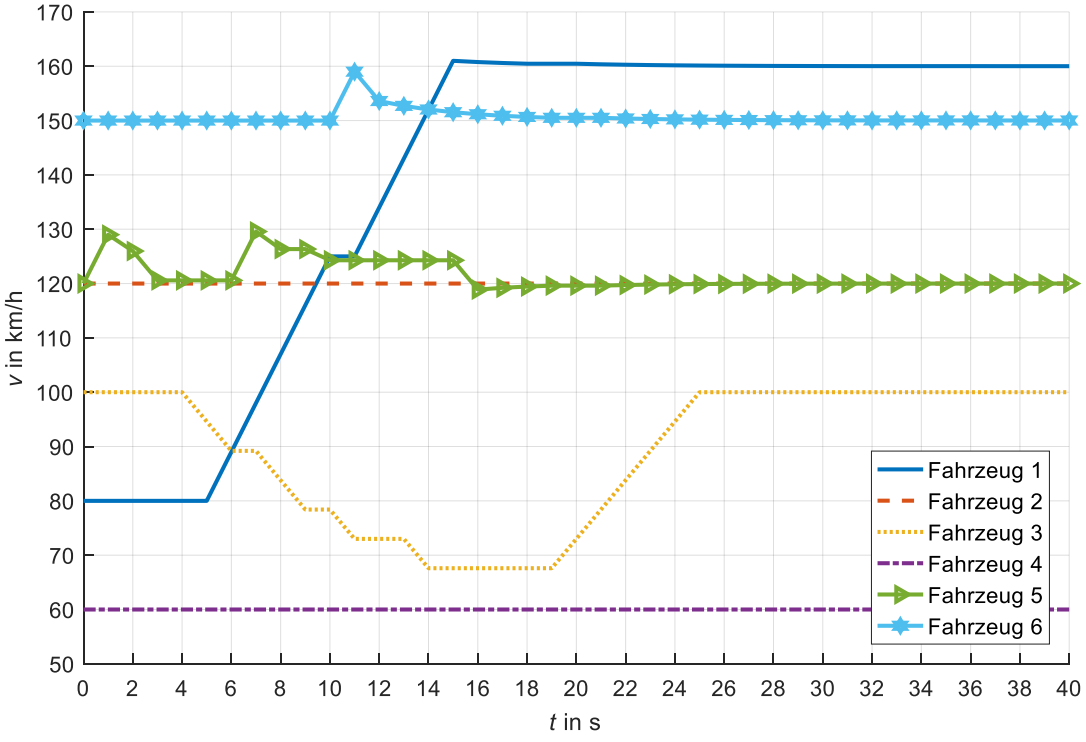


Abb. 4.13 Geschwindigkeitsverlauf der beteiligten Fahrzeuge in Verkehrsszenario 03

Der zweite Beschleunigungsvorgang von Fahrzeug 5 geschieht zu dem Zeitpunkt, zu dem Fahrzeug 1 auf die mittlere Fahrspur gewechselt hat und ebenfalls beginnt zu beschleunigen. Fahrzeug 5 wechselt auf die linke Fahrspur hinter Fahrzeug 6 und verzögert wieder auf die Wunschgeschwindigkeit von 120 km/h. Nachdem Fahrzeug 1 rechts überholt hat, wechselt Fahrzeug 5 wieder auf die mittlere Fahrspur. Fahrzeug 6 fährt die ganze Zeit mit konstanter Geschwindigkeit auf der linken Fahrspur, bis die mittlere frei ist. Parallel zu Fahrzeug 5 führt es einen Spurwechsel durch. Ab ca. 25 s Simulationszeit haben alle Fahrzeuge ihre Wunschgeschwindigkeit wieder erreicht.

Im Folgenden werden die Aktionen von Fahrzeug 1 näher betrachtet. Dazu werden die durchgeführten Aktionen für jeden Zeitschritt in Abb. 4.14 dargestellt. Auf der y-Achse befinden sich von unten nach oben die Aktionen $a_{=0}$, $a_{>0}$ und $a_{<0}$ für die vordefinierten, festen Beschleunigungswerte. Darüber ist die Beschleunigung a_{IDM}, die sich aus dem IDM berechnet. Die letzten beiden y-Achsen-Markierungen stehen für einen Spurwechsel nach links (SPW-L) und nach rechts (SPW-R). Die konkreten Werte in diesem Diagramm beschreiben die Aktion, die jeweils vom Zeitschritt auf der x-Achse zum nächsten Zeitschritt ausgeführt wird. Der Wert für Spurwechsel nach links bei t_0 sagt bspw. aus, dass im Zeitraum von $t = 0$ s bis $t = 1$ s ein Spurwechselmanöver ausgeführt wird. Weil Spurwechsel nur mit konstanter Geschwindigkeit durchgeführt werden können und Fahrzeug 1 bei Simulationsschritt 2 s keine Beschleunigung wählt, resultiert der bereits beschriebene, konstante Geschwindigkeitsverlauf bis $t = 5$ s. Danach wird, bis auf eine kurze Unterbrechung, die vordefinierte Beschleunigung von 1,5 m/s² ausgeführt. Im Anschluss daran erfolgt eine langsame Anpassung der Geschwindigkeit auf die genaue Wunschgeschwindigkeit mittels des IDM. Dabei treten sehr geringe Beschleunigungswerte auf. Das geschieht so lange, bis die Wunschgeschwindigkeit erreicht ist. Obwohl das der Fall ist, treten trotzdem zum Ende vier IDM-Beschleunigungen auf. Im Zeitraum zwischen $t = 18$ s und $t = 20$ s wechselt das Fahrzeug 1 die Fahrspur nach rechts.

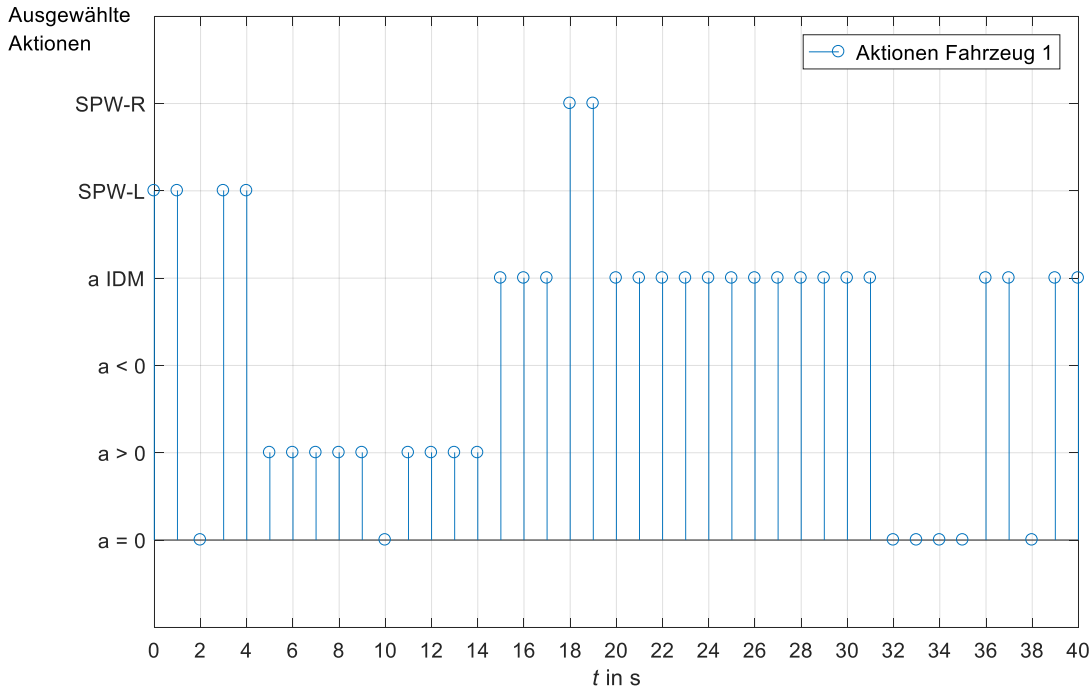


Abb. 4.14 Ausgeführte Aktionen von Fahrzeug 1 in Verkehrsszenario 03

Im Vergleich zu anderen Verkehrsszenarien tritt kein „Überschwinger“ auf. Lediglich im Geschwindigkeitsverlauf von Fahrzeug 6 ist ein plötzlicher Anstieg zu erkennen. Dieser tritt

allerdings zu dem Zeitpunkt auf, zu dem Fahrzeug 2 hinter Fahrzeug 6 auf die linke Fahrspur wechselt. Es beschleunigt also, um den Abstand zu Fahrzeug 2 zu erhöhen.

In Verkehrsszenario 04 ist hauptsächlich interessant, wie Fahrzeug 1 versucht, seine im Vergleich zu allen anderen Fahrzeugen hohe Wunschgeschwindigkeit umzusetzen und wie die anderen Fahrzeuge darauf reagieren. Die Fahrzeuge 1 und 2 können aus dem vollständigen Set an Aktionen wählen, für alle anderen steht nur das IDM zu Verfügung. Der Geschwindigkeitsverlauf der Fahrzeuge aus Verkehrsszenario 04 ist in Abb. 4.15 gezeigt.

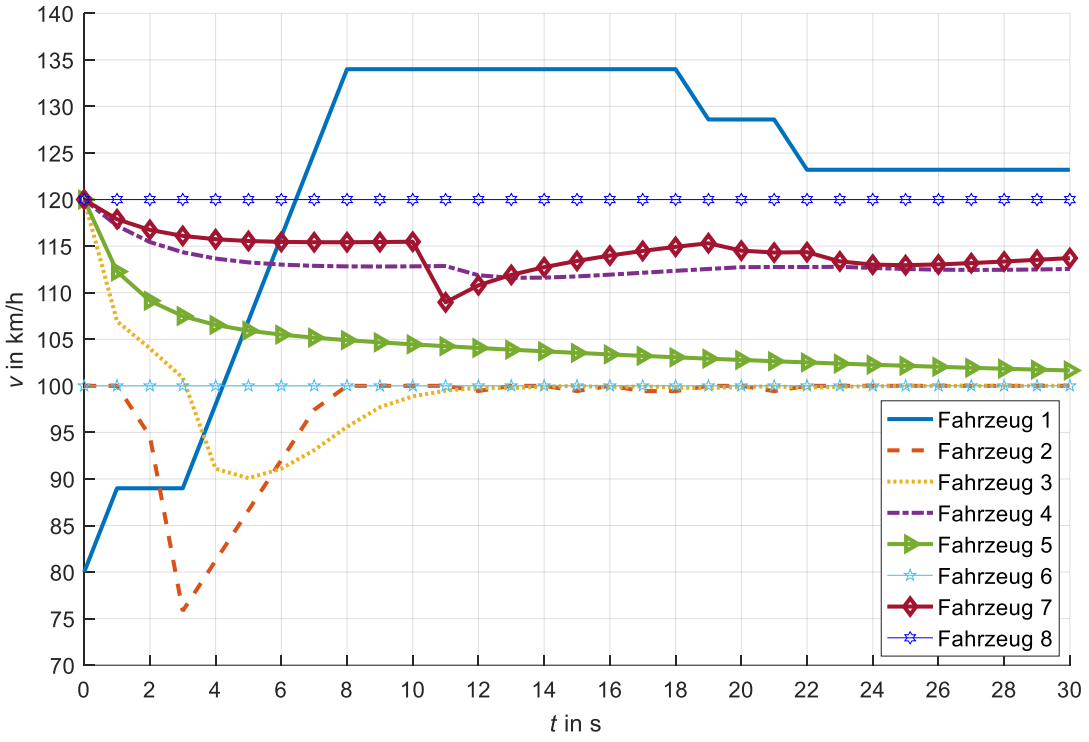


Abb. 4.15 Geschwindigkeitsverlauf der beteiligten Fahrzeuge in Verkehrsszenario 04

Zunächst beschleunigt Fahrzeug 1 auf der Beschleunigungsspur und beginnt nach einer Sekunde den Spurwechsel. Der Spurwechsel führt dazu, dass Fahrzeug 2 abbremst, um einen angemessenen Abstand zu dem neuen, vorausfahrenden Hindernis (Fahrzeug 1) einzuhalten. Durch die unmittelbare weitere Beschleunigung von Fahrzeug 1 kann auch Fahrzeug 2 wieder auf seine Wunschgeschwindigkeit beschleunigen, bis sie nach ca. 8 s erreicht ist. Fahrzeug 3 verzögert direkt zu Beginn. Der Geschwindigkeitsverlauf folgt grob dem des vorausfahrenden Fahrzeugs 2, allerdings mit geringem Zeitversatz. Nach dem Abbremsen auf ca. 90 km/h beschleunigt Fahrzeug 3 auf die Geschwindigkeit des vorausfahrenden Fahrzeugs. Die Wunschgeschwindigkeit von 120 km/h wird nicht erreicht. Die Fahrzeuge 6 und 8 fahren mit konstanter Geschwindigkeit. Fahrzeug 5 startet mit einer Geschwindigkeit von 120 km/h, beginnt aber sofort zu verzögern. Die Geschwindigkeit nähert sich im Laufe der Simulationszeit der des vorausfahrenden Fahrzeugs 6 von 100 km/h an, liegt jedoch auch unter der gewünschten Geschwindigkeit von 120 km/h. Nachdem Fahrzeug 1 auf der rechten Fahrspur beschleunigt hat, erfolgt ein Spurwechsel auf die linke Fahrspur in die Lücke zwischen Fahrzeug 6 und 7. Das bewirkt eine Verzögerung von Fahrzeug 7 bei 10 s Simulationszeit. In Abb. 4.16 sind die zugehörigen Beschleunigungswerte von Fahrzeug 7 und dem dahinterfahrenden Fahrzeug 4 abgebildet. Der Abstand von Fahrzeug 4 und 7 zum jeweils nächsten Hindernis ist in Abb. 4.17 zu sehen. Der Spurwechsel von Fahrzeug 1 erzeugt einen Sprung im Abstandswert für Fahrzeug 7. Daraus resultiert die beschriebene Verzögerung, wie sie im

Geschwindigkeits- und Beschleunigungsverlauf zu erkennen ist. Das nachfolgenden Fahrzeug 4 verzögert zeitversetzt und in geringerem Maße.

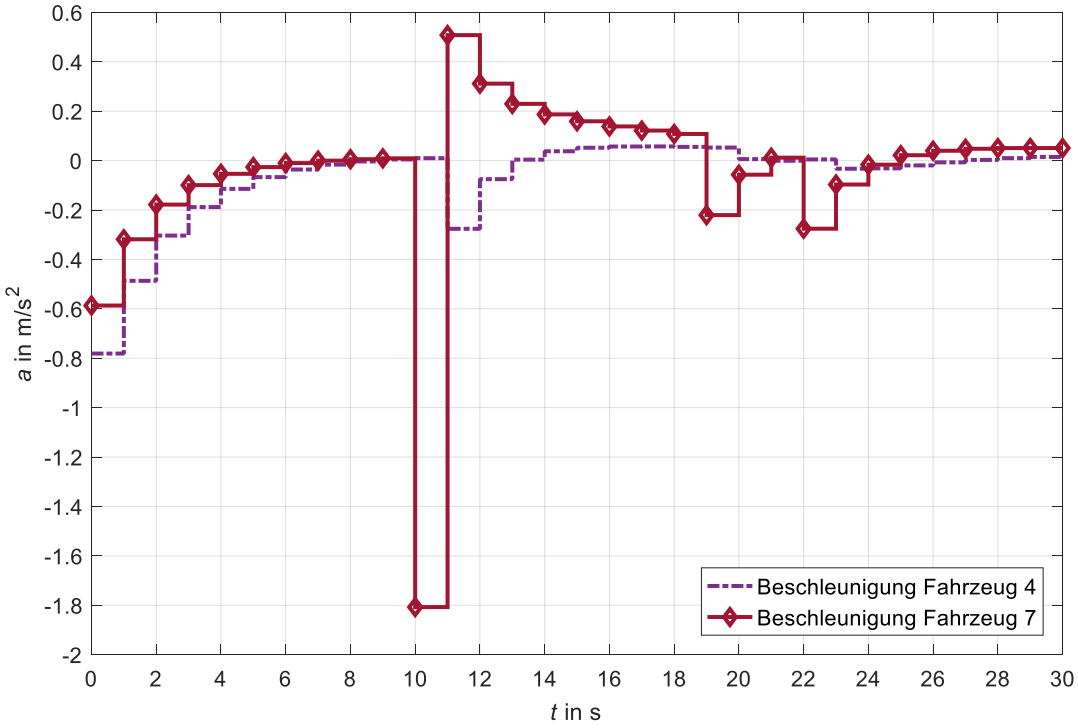


Abb. 4.16 Beschleunigungsverlauf der Fahrzeuge 4 und 7 in Verkehrsszenario 04

Auffällig ist, dass Fahrzeug 4 sichtbar weniger stark auf die Geschwindigkeitsänderung von Fahrzeug 7 reagiert, als Fahrzeug 7 auf das Verhalten von Fahrzeug 1 anspricht.

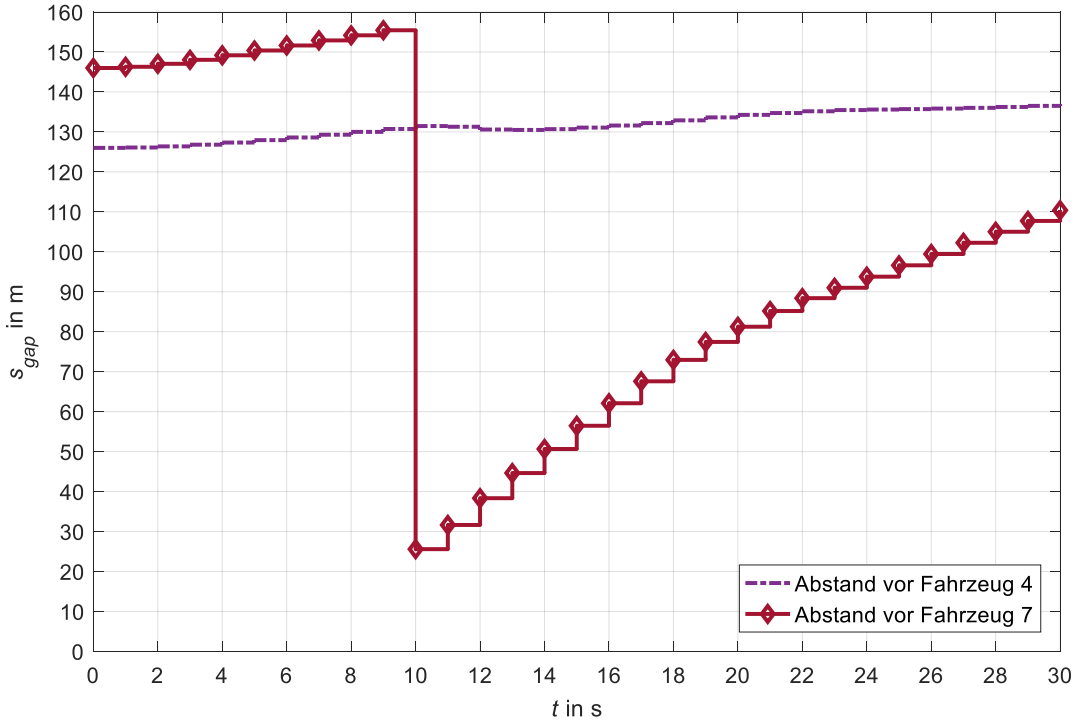


Abb. 4.17 Abstandswert s_{gap} von Fahrzeug 4 und 7 in Verkehrsszenario 04

Deutlich im Beschleunigungsverlauf von Fahrzeug 7 ist das Abbremsen von Fahrzeug 1 bei 18 s und 21 s Simulationszeit zu sehen. Fahrzeug 1 macht dort Anpassungen der eigenen

Geschwindigkeit an die des vorausfahrenden Fahrzeugs 8. Fahrzeug 7 verzögert um einen Zeitschritt versetzt. Bei Fahrzeug 4 ruft diese Verzögerung kaum eine Aktion hervor. Während der Abstand zwischen Fahrzeug 1 und 7 nach dem Abbremsen von Fahrzeug 7 wieder ansteigt, verändert sich der zwischen Fahrzeug 4 und 7 nur gering. Die Geschwindigkeit der Fahrzeuge 4 und 7 liegt zum Ende der Simulation bei ca. 115 km/h, was nicht der Wunschgeschwindigkeit von 120 km/h entspricht. Zu beobachten ist, dass die IDM-Beschleunigung stets gegen den Nullwert strebt. Außerdem kann festgestellt werden, dass die Fahrzeuge auf der linken Fahrspur direkt zu Beginn verzögern, obwohl die Startgeschwindigkeit bereits der Wunschgeschwindigkeit entspricht und alle Fahrzeuge gleich schnell fahren.

4.3 Analyse und Bewertung der Berechnungsergebnisse

Die vorgestellten Berechnungsergebnisse zu den Verkehrsszenarien 01 bis 04 werden in diesem Unterkapitel näher analysiert und bewertet. Besonders wird auf festgestellte Besonderheiten und Auffälligkeiten eingegangen. Im Rahmen der Analyse werden gegebenenfalls Simulationsparameter variiert, um die Aussagekraft der erhaltenen Ergebnisse einordnen und vergleichen zu können. Zusätzlich zu den Simulationsparametern wie Schrittweite, Planungshorizont und Anzahl der Iterationen können die initialen Zustandsgrößen x und v und die IDM-Parameter variiert werden, um die Aussagekraft und Qualität der erhaltenen Ergebnisse beurteilen zu können. Anhand der vier Verkehrsszenarien werden häufig festgestellte Phänomene, die während der Entwicklungs- und Testphase aufgetreten sind, erläutert. Grundsätzlich findet jede Entscheidung für oder gegen eine Aktion anhand der ermittelten Kosten statt. Es handelt sich um konkrete Kostenwerte, die miteinander verglichen werden. Während der Selection-Phase werden die jeweils besten Knoten gewählt. Die Kosten werden stets auf die gleiche Weise ermittelt und mit festgelegten Gewichtungsfaktoren multipliziert. Durch die konkreten Werte tritt häufig der Fall auf, dass bereits geringe Änderungen bspw. der Anfangszustände zu einem komplett anderen Fahrmanöver führen. In Verkehrsszenario 01 ist das der Fall. Die 50 m, die Fahrzeug 2 bei der einspurigen Variante in negative x-Richtung verschoben wird, sind entscheidend dafür, ob Fahrzeug 1 das Fahrzeug 2 vorbeifahren lässt oder davor einschert. Theoretisch kann bereits eine Verschiebung des Fahrzeugs 2 um einen Meter diese gänzlich unterschiedliche Lösung hervorrufen.

Auffällig im Geschwindigkeitsverlauf aus Abb. 4.8 ist, dass Fahrzeug 1 über seine Wunschgeschwindigkeit hinaus beschleunigt. Dieses Phänomen lässt sich auf die Kosten zurückführen. Fahrzeug 2 will auf 140 km/h beschleunigen, Fahrzeug 1 nur auf 120 km/h. Dadurch, dass die Kosten für eine Überschreitung der Geschwindigkeit höher liegen als für eine Unterschreitung, stellen sich die niedrigsten Kosten unterhalb von 130 km/h ein. Die tatsächlichen Geschwindigkeiten beider Fahrzeuge sind sozusagen der Kompromiss aus den beiden Wunschgeschwindigkeiten. Interessant ist, dass Fahrzeug 1 von der Wunschgeschwindigkeit abweicht, obwohl sie problemlos zu erreichen wäre. Die Kooperationsleistung besteht darin, dass Fahrzeug 1 einen „schlechteren“ Zustand in Kauf nimmt, um Fahrzeug 2 entgegen zu kommen und dadurch die Gesamtkosten zu minimieren.

Die festgestellten „Überschwinger“ treten in den Geschwindigkeitsverläufen häufig auf, vor allem dann, wenn nur geringe Geschwindigkeitsanpassungen notwendig sind. Das ist meistens der Fall, wenn ein hinterherfahrendes Fahrzeug nicht seine Wunschgeschwindigkeit erreichen kann. Dann passt das Fahrzeug die eigene Geschwindigkeit stetig an den Vordermann an, um

so nahe wie möglich an der Wunschgeschwindigkeit zu sein und trotzdem den geforderten Abstand einzuhalten. Dieses schrittweise Anpassen erfordert nur sehr geringe Beschleunigungs- oder Verzögerungswerte. Das IDM gibt in solchen Folge-Situationen meist einen negativen Wert aus. Die einzige Möglichkeit für eine Geschwindigkeitsanpassung ergibt sich aus dem Standard-Beschleunigungswert $a_{>0}$. Dieser Wert (für Pkw: $a_{>0} = 2{,}5$ m/s²) ist für eine geringe Geschwindigkeitserhöhung zu groß. Der Standard-Verzögerungswert für Fahrzeuge beträgt $a_{<0} = -1{,}5$ m/s². Damit die Geschwindigkeit nur in geringem Maße angehoben wird, führt das Fahrzeug 2 zuerst eine Standard-Beschleunigung $a_{>0}$ und direkt im Anschluss eine Standard-Verzögerung $a_{<0}$ aus. Dadurch entstehen die beobachteten „Überschwinger“, wie sie in Abb. 4.18 dargestellt sind.

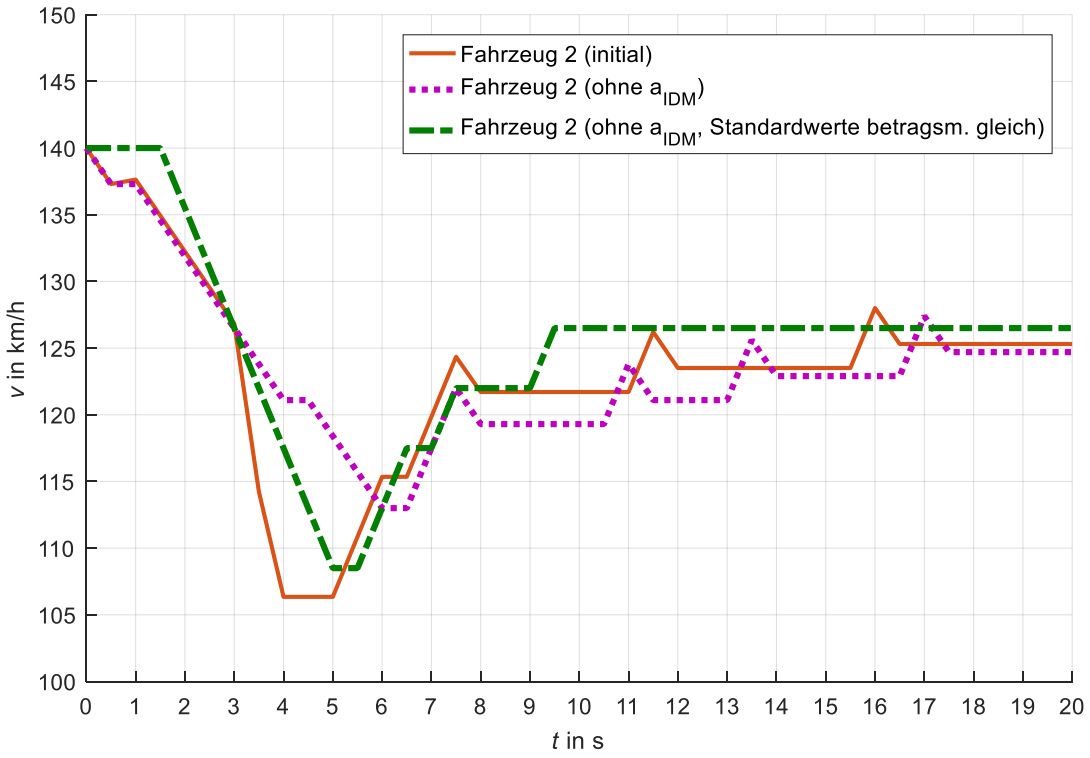


Abb. 4.18 Geschwindigkeitsverlauf von Fahrzeug 2 in Verkehrsszenario 01 (eine Fahrspur, $x_2(t_0) = 50$ m); jeweils mit IDM-Beschleunigung nicht zur Auswahl

In dieser Abbildung ist der Geschwindigkeitsverlauf von Fahrzeug 2 aus Verkehrsszenario 01 in der ursprünglichen Form und mit zwei Varianten abgebildet. Die orange, durchgehende Linie zeigt den ursprünglichen Verlauf der Geschwindigkeit von Fahrzeug 2, wie er bereits beschrieben wurde. Die lila, gepunktete Linie zeigt ebenfalls die Geschwindigkeit von Fahrzeug 2 im gleichen Verkehrsszenario, allerdings wurde die IDM-Aktion deaktiviert, so dass sie nicht mehr ausgewählt werden kann. Das dient dazu, ein klares Ergebnis zu bekommen, das nicht durch die im Voraus unbekannten IDM-Werte beeinflusst wird. Die „Überschwinger“ treten in beiden Geschwindigkeitsverläufen auf. Bei der zweiten Variante wurden neben dem Deaktivieren der IDM-Beschleunigung a_{IDM} die Standardwerte der Beschleunigung und Verzögerung betragsmäßig gleich groß definiert, also $a_{>0} = 2{,}5$ m/s² und $a_{<0} = -2{,}5$ m/s². Mit diese Werten lässt sich keine Geschwindigkeitsänderung erreichen, wenn beide Aktionen hintereinander ausgeführt werden. Die Geschwindigkeit nach diesem Manöver ist gleich der davor. Der Geschwindigkeitsverlauf der zweiten Variante bestätigt diese Annahme, weil trotz ansonsten identischer Startbedingungen keine „Überschwinger“ auftreten. Bei aktivierter IDM-Aktion a_{IDM} treten sie auch bei betragsmäßig gleichen Standardwerten auf, weil das IDM Werte ausgeben kann, die genau einen solchen „Überschwinger“ provozieren. Grundsätzlich treten sie

aufgrund der Diskretisierung der Aktionen auf. Erst bei einer sehr feinen Diskretisierung der Beschleunigungs- und Verzögerungswerte würden sie sich vollständig vermeiden lassen. Dann wäre stets die passende Beschleunigung anwählbar und eine Kombination aus Beschleunigung und Verzögerung wäre nicht notwendig. Aus genannten Gründen ist eine feinere Diskretisierung allerdings nicht durchsetzbar, weshalb die „Überschwinger“ toleriert werden müssen. Sie treten auch auf, wenn ein schnelleres Fahrzeug hinter einem langsameren fährt und dabei ständig die Geschwindigkeit verringert, um den Abstand einzuhalten.

Ein bedeutender Stellhebel zur Beeinflussung der Berechnungsergebnisse ist die Simulationsschrittweite Δt. Je kleiner sie gewählt wird, desto öfter kann ein Fahrzeug bei konstantem Planungshorizont entscheiden, welche Aktion die geringsten Kosten hervorruft. Die Ergebnisqualität steigt folglich an. Allerdings führt das Verringern der Simulationsschrittweite nicht immer zu einer Verbesserung des Simulationsergebnisses. Je kleiner die Simulationsschrittweite ist, desto tiefer ist der Suchbaum bei gleichbleibendem Planungshorizont. Die Anzahl der Ebenen eines Suchbaumes bestimmt sich aus dem Planungshorizont geteilt durch die Simulationsschrittweite. Dieser Term geht in den Exponenten der Formel zur Berechnung der maximalen Anzahl an möglichen Kombinationen ein (Gl. (2.8)). Die Verringerung der Simulationsschrittweite bringt zwei große Nachteile mit sich. Zum einen erhöht sich mit der Tiefe des Suchbaumes auch die Anzahl der notwendigen Durchläufe der „äußeren“ Iterationsschleife (pro Baumebene ein Durchlauf mit je der festgelegten Anzahl an Iterationen aus den Simulationsparametern). Zum anderen muss die Anzahl der Iterationen pro Iterationsschleife deutlich erhöht werden. Durch die geringe Schrittweite Δt wird mit jedem neuen expandierten Knoten auf der nächsten, tieferen Baumebene auch nur der Zustand zum Zeitpunkt $t + \Delta t$ untersucht und bewertet. Für eine Simulationsschrittweite von $\Delta t = 0{,}2$ s muss bspw. fünfmal tiefer in den Baum vorgedrungen werden als für ein $\Delta t = 1$ s, damit die gleiche Aussage getroffen werden kann. Zwar wird in der Simulation-Phase mit dem IDM unabhängig von der Simulationsschrittweite ein Zeitraum von $T_{SimIDM} = 4$ s simuliert und bewertet, allerdings kann diese Abschätzung das tatsächliche Explorieren der Zustände nicht ersetzen. Langfristig ungünstige Fahrmanöver werden erst spät und nach höherem Rechenaufwand erkannt. Wird bspw. ein Bereich des Baumes untersucht, der sich erst nach vielen Expansionsschritten als unzulässig herausstellt, geschieht dies mit der groben Diskretisierung schneller als mit der feinen. Mit der feinen Diskretisierung müssen mehr Knoten expandiert werden, bis festgestellt wird, dass der Teilbaum zu einer schlechten Lösung führt. Aus diesem Grund ist eine verringerte Simulationsschrittweite i.d.R. mit einer deutlichen Erhöhung der Iterationsanzahl auszugleichen. Weil die Baumtiefe und die notwendige Anzahl an Iterationen ansteigen, erfordert eine verringerte Simulationsschrittweite einen erheblichen Mehraufwand an Rechenkapazität. Für Verkehrsszenario 01 findet eine Simulation mit dem Zeitschritt $\Delta t = 0{,}2$ s und ansonsten identischen Startbedingungen eine schlechte Lösung, wie sie in Abb. 4.19 zu sehen ist. Beide Fahrzeuge fahren bis kurz vor Ende der Beschleunigungsspur nebeneinander (Plot im Anhang). Dann muss Fahrzeug 1 einen Spurwechsel ausführen, damit eine Kollision vermieden wird. Dieser Spurwechsel zwingt Fahrzeug 2 in ein starkes Verzögerungsmanöver, weil ihm Fahrzeug 1 in geringem Abstand vorausfährt. Die ursprünglichen Geschwindigkeitsverläufe der Simulation mit dem Zeitschritt $\Delta t = 0{,}5$ s finden sich in Abb. 4.8.

Anhand dieses Beispiels ist deutlich zu sehen, dass die notwendige, vorausschauende Fahrweise bei einer Verringerung der Simulationsschrittweite und gleichbleibender Anzahl an Iterationen verloren geht. Die Fahrzeuge reagieren nur und treffen keine vorausschauenden Entscheidungen.

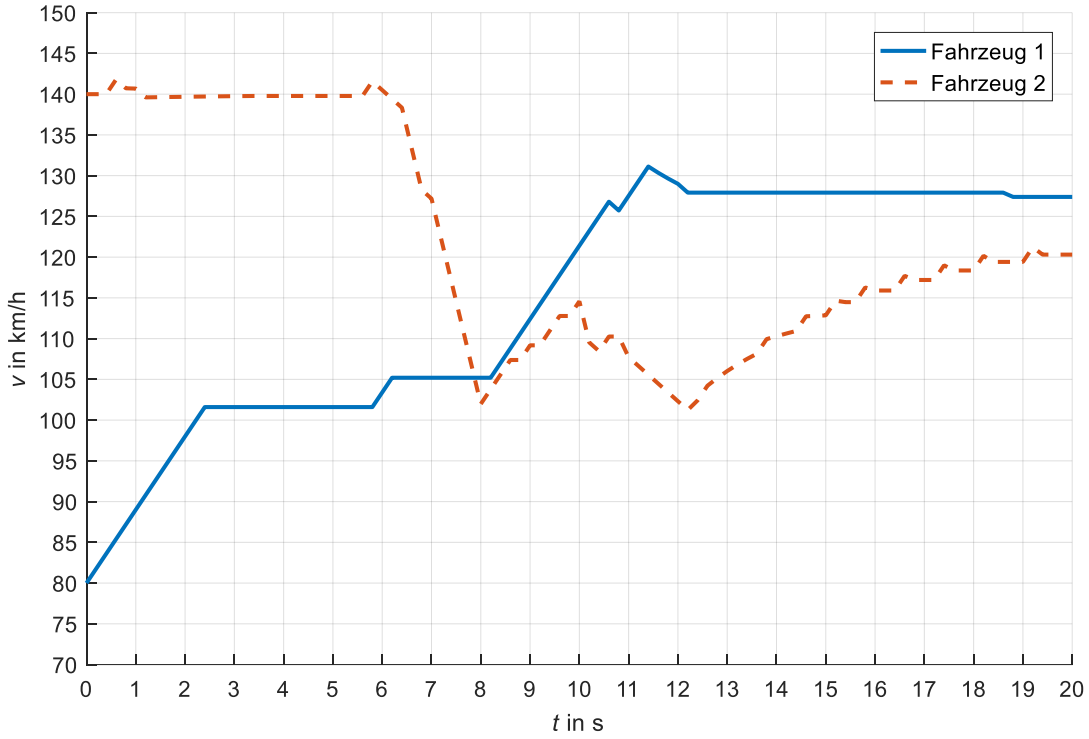


Abb. 4.19 Geschwindigkeitsverlauf der beiden Fahrzeuge in Verkehrsszenario 01 (eine Fahrspur, $x_2(t_0) = 50$ m) mit Simulationsschrittweite $\Delta t = 0{,}2$ s

Im Verlauf der Fahrzeuggeschwindigkeiten bei Verkehrsszenario 02 (Abb. 4.10) sind auch die erwähnten „Überschwinger“ zu erkennen. Auffällig ist allerdings der zackige Geschwindigkeitsverlauf bei $t = 5$ s. Dort scheint Fahrzeug 2 ohne Grund von seiner Wunschgeschwindigkeit abzuweichen. Es befindet sich kein anderes Fahrzeug oder Hindernis auf der gleichen Fahrspur. Dieses Phänomen tritt nur in Verkehrsszenarien auf, in denen mehrere Fahrzeuge beteiligt sind. Der Auslöser sind ein oder mehrere Fahrzeuge, die sehr hohe Kosten verursachen, und ein Fahrzeug, dass durch mehrere, verschiedene Aktionen im Verhältnis dazu nur geringe Kosten verursacht. Werden die geringen Kosten des einen Fahrzeuges zu den restlichen, hohen Kosten addiert, kann der Fall auftreten, dass durch internes Runden in Matlab die geringen Kosten des einen Fahrzeugs „verschwinden“. Der Algorithmus kann anschließend nicht mehr zwischen den unterschiedlichen Knoten differenzieren, weil die Kosten gleich groß sind. Der gewählte Knoten kann dann ein Knoten sein, der für das Fahrzeug mit den nahezu gleichwertigen Aktionen eine davon auswählt, die nicht der Erwartung entspricht. Bei dem Beispiel aus Verkehrsszenario 02 entspricht das einer Verzögerung, einer anschließenden Beschleunigung und anschließend dem langsamen Annähern an die Wunschgeschwindigkeit. Die maximale Abweichung von der Wunschgeschwindigkeit beträgt 2,7 km/h, was einen IDM-Beschleunigungswert von 0,24 m/s² hervorrufen würde. Fahrzeug 1 verursacht zu diesem Zeitpunkt hohe Kosten, weil es ca. 15 km/h unterhalb seiner Wunschgeschwindigkeit fährt, Fahrzeug 3 vor sich hat und mit höherer Geschwindigkeit auf dieses zu fährt. Im Verhältnis dazu sind die Kosten von Fahrzeug 2 so gering, dass sie durch Runden verloren gehen. Bei der Simulation eines einzelnen Fahrzeugs, das mit der Wunschgeschwindigkeit auf einer freien Fahrbahn fährt, treten keine Störungen auf. Gleiches gilt für mehrere Fahrzeuge, wenn keine Situation auftritt, die große Unterschiede in den einzelnen Fahrzeug-Kostenwerten hervorruft. Aufgrund der Tatsache, dass dieses Verhalten nur bei unbedeutenden Aktionen, im Sinne von geringen Kosten, auftritt, wird es in den Berechnungsergebnissen akzeptiert. Wenn eine solche Aktion Einfluss auf andere Fahrzeuge hätte, wären die Kosten wieder groß genug, dass sie berücksichtigt werden.

Der vermeintliche „Überschwinger“ im Geschwindigkeitsverlauf von Fahrzeug 6 aus Verkehrsszenario 03 wurde bereits erklärt. Fahrzeug 6 beschleunigt, um den Abstand zum dahinter einscherenden Fahrzeug 2 zu vergrößern. Ein weiterer Anhaltspunkt, dass es sich um keinen „Überschwinger“ wie in den anderen Fällen handelt, ist die Tatsache, dass sich vor Fahrzeug 6 zu dem Zeitpunkt kein Fahrzeug befindet. Nur Fahrzeug 2 ist der Auslöser des Beschleunigungsmanövers. Interessant sind bei Verkehrsszenario 03 die Geschwindigkeitsverläufe der Fahrzeuge 1, 3 und 5. Diese sind in Abb. 4.20 dargestellt.

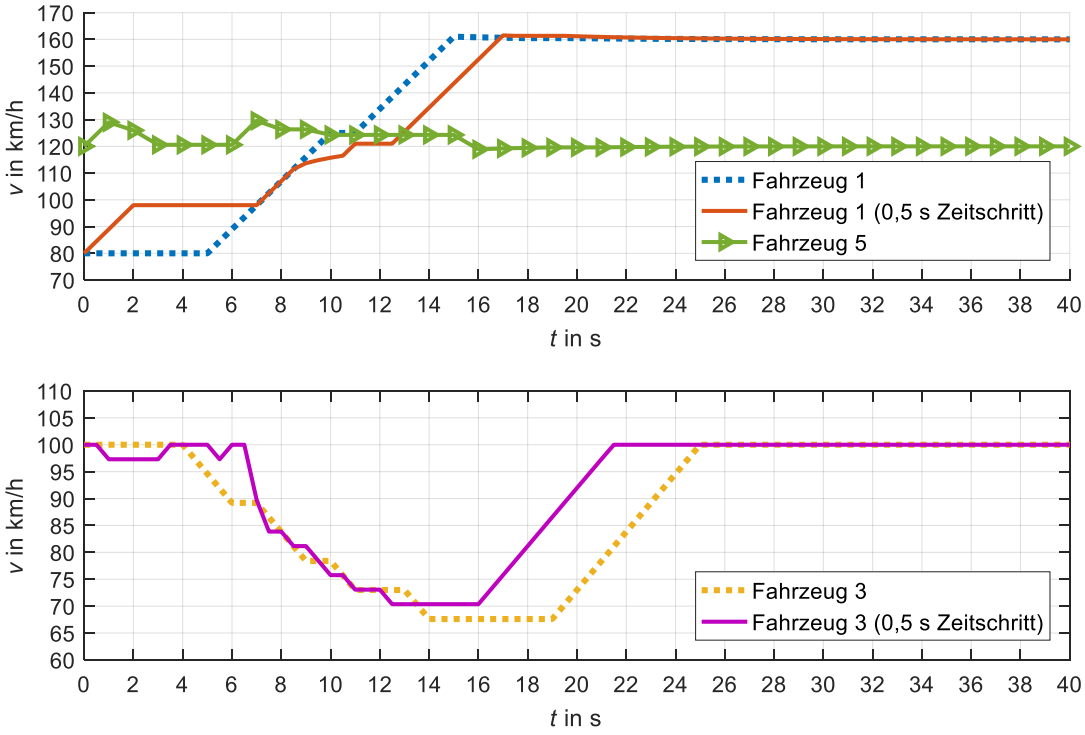


Abb. 4.20 Geschwindigkeitsverlauf der Fahrzeuge 1, 3 und 5 in Verkehrsszenario 03

Zum Vergleich wurde zusätzlich die identische Simulation mit der Schrittweite $\Delta t = 0{,}5$ s durchgeführt. Fahrzeug 1 beginnt im Vergleich zur ursprünglichen Simulation sofort mit dem Beschleunigungsvorgang und wechselt danach erst auf die rechte Fahrspur. Der restliche Geschwindigkeitsverlauf ist ähnlich. Das verschiedene Verhalten am Anfang kann darauf zurückzuführen sein, dass bei der geringeren Simulationsschrittweite das Ende des Beschleunigungsstreifens erst „später“ bemerkt wird. Dieser Effekt hat im vorliegenden Verkehrsszenario keine negativen Auswirkungen, kann aber mit einer höheren Anzahl an Iterationen theoretisch ausgeglichen werden. Der Geschwindigkeitsverlauf von Fahrzeug 3 verhält sich in beiden Fällen ähnlich. Im Fall der geringeren Simulationsschrittweite wechselt es ca. 3 s früher die Fahrspur. Bei $t = 1$ s und $t = 5$ s treten bei Fahrzeug 3 die beschriebenen, sehr geringen Abweichungen von der Wunschgeschwindigkeit auf, die dann wieder korrigiert werden. Weil die Kosten der Aktionen $a_{=0}$, $a_{>0}$ und $a_{<0}$ gering und fast identisch sind, können sie nach der Rundung nicht mehr unterschieden werden. Dieser Effekt wird durch die vielen beteiligten Fahrzeuge und die daraus resultierenden Gesamtkosten hervorgerufen. Die Auswirkungen und damit die Relevanz dieses Effektes sind vernachlässigbar klein.

Bei Betrachtung des Verlaufs der Geschwindigkeit von Fahrzeug 5 fällt auf, dass dieser nur über der Wunschgeschwindigkeit liegt, wenn sich ein Fahrzeug dahinter auf der gleichen Fahrspur befindet. Ist dies nicht der Fall, verzögert Fahrzeug 5 sofort wieder auf die Wunschgeschwindigkeit. Es muss zweimal beschleunigen und reagiert anscheinend nur auf andere Fahrzeuge und agiert nicht im Sinne einer langfristig günstigen Lösung. Dieses

Verhalten ist ein Zeichen dafür, dass die Iterationsanzahl höher gewählt werden muss. Vor allem aufgrund der hohen Anzahl an Fahrzeugen und damit dem Suchbaum mit großem Verzweigungsfaktor ist das eine plausible Annahme. Der Einfluss der Iterationsanzahl auf die kooperative Fahrmanöverplanung lässt sich mithilfe des zu Erläuterungszwecken eingeführten Verkehrsszenarios 05 aus Abb. 4.21 erläutern.

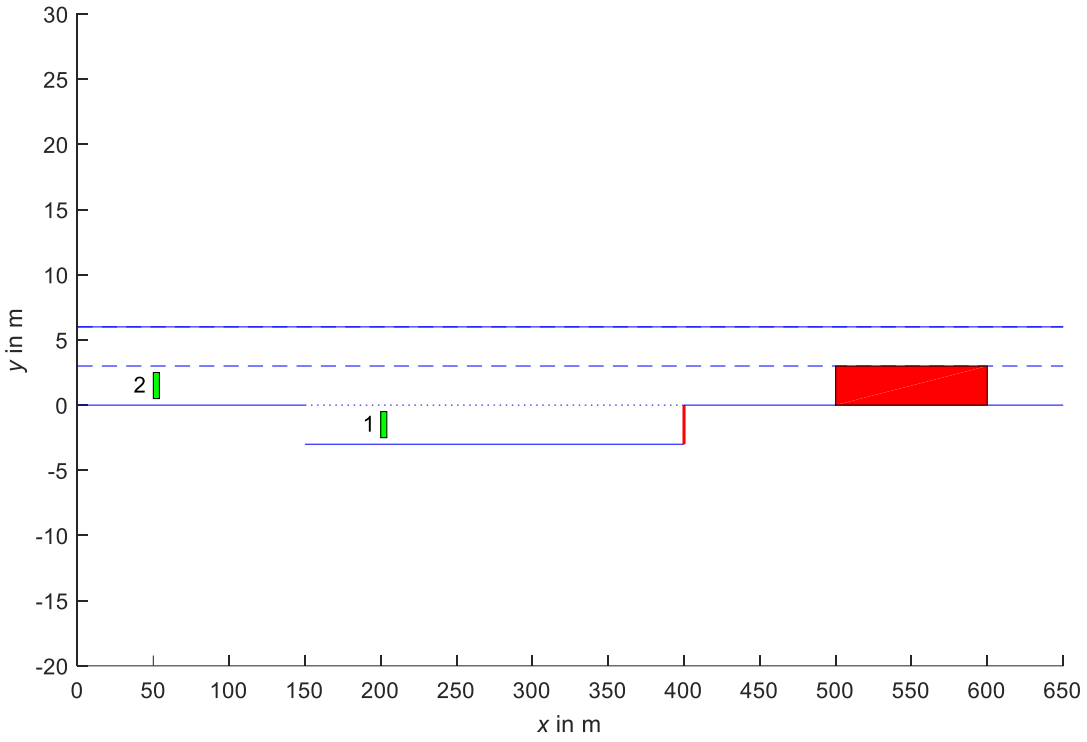


Abb. 4.21 Verkehrsszenario 05 zur Erläuterung des Einflusses der Anzahl an Iterationen

Für die jeweils identischen Fahrzeugzustände und IDM-Parameter wird das Verkehrsszenario 05 simuliert. Die Schrittweite beträgt in beiden Fällen $\Delta t = 1$ s. Die Anzahl der Iterationen ist in der einen Variante 1 000 und in der anderen 5 000. Der Geschwindigkeitsverlauf der beiden Fahrzeuge, der sich aus den zwei verschiedenen Varianten ergibt, ist in Abb. 4.22 sehen. Die Trajektorien und Geschwindigkeitsverläufe aller Varianten sind im Anhang zu finden. Bei der Simulation mit nur 1 000 Iterationen kann ein weniger vorausschauendes Fahren festgestellt werden. Fahrzeug 1 beschleunigt nach dem Spurwechsel zunächst, beginnt aber dann wieder abzubremsen. Ein Spurwechsel auf die linke Fahrspur ist nicht möglich, weil sich dort Fahrzeug 2 befindet. Nachdem dieses überholt hat, kann Fahrzeug 1 die Fahrspur wechseln und auf die Wunschgeschwindigkeit beschleunigen. Bei der Simulation mit 5 000 Iterationen beschleunigt Fahrzeug 1 im Vergleich zu der ursprünglichen Simulation länger bis $t = 7$ s. Dann wechselt es auf die linke Fahrspur und beschleunigt weiter, sogar über die eigene Wunschgeschwindigkeit hinaus, um den Abstand zu Fahrzeug 2 aufzubauen. Nach dem Spurwechsel wird die Geschwindigkeit an die Wunschgeschwindigkeit angenähert. Fahrzeug 2 beginnt bereits vor dem Spurwechsel von Fahrzeug 1 die Geschwindigkeit zu reduzieren, damit dieses einscheren kann. Nachdem der Spurwechsel durchgeführt wurde und Fahrzeug 1 beschleunigt, hebt auch Fahrzeug 2 die Geschwindigkeit wieder an. Sobald die Fahrspur vor Fahrzeug 2 frei ist, beschleunigt es auf die eigene Wunschgeschwindigkeit von 140 km/h. Der Unterschied in beiden Varianten tritt hauptsächlich zwischen den Zeitpunkten $t = 5$ s und $t = 18$ s auf. In diesem Zeitraum wird das vorausschauende Planen des Fahrmanövers sichtbar: zum einen, weil Fahrzeug 1 trotz des Endes der Beschleunigungsspur beschleunigt, zum anderen, weil Fahrzeug 2 ohne Hindernis voraus verzögert, um dann Fahrzeug 1

einscheren zu lassen. Beide Fahrmanöver sind für sich gesehen mit hohen Kosten verbunden, allerdings werden die Gesamtkosten des Verkehrsszenarios dadurch minimiert. Ein weiterer Hinweis darauf, dass mehr Iterationen zu einer besseren Lösung führen, ist der Geschwindigkeitsverlauf von Fahrzeug 2 mit 1 000 Iterationen zwischen Zeitpunkt $t = 8$ s und $t = 14$ s. Ohne ersichtlichen Grund verzögert und beschleunigt es. Daraus lässt sich folgern, dass die Kriterien zur Entscheidungsfindung eine schlechte Qualität haben. Eine Verbesserung kann über mehr Iterationen erreicht werden, weil dadurch die einzelnen Aktionen und deren Auswirkungen genauer untersucht werden. Die Entscheidung kann aufgrund besserer/genauerer Daten, was die längerfristigen Auswirkungen einer Aktion angeht, getroffen werden.

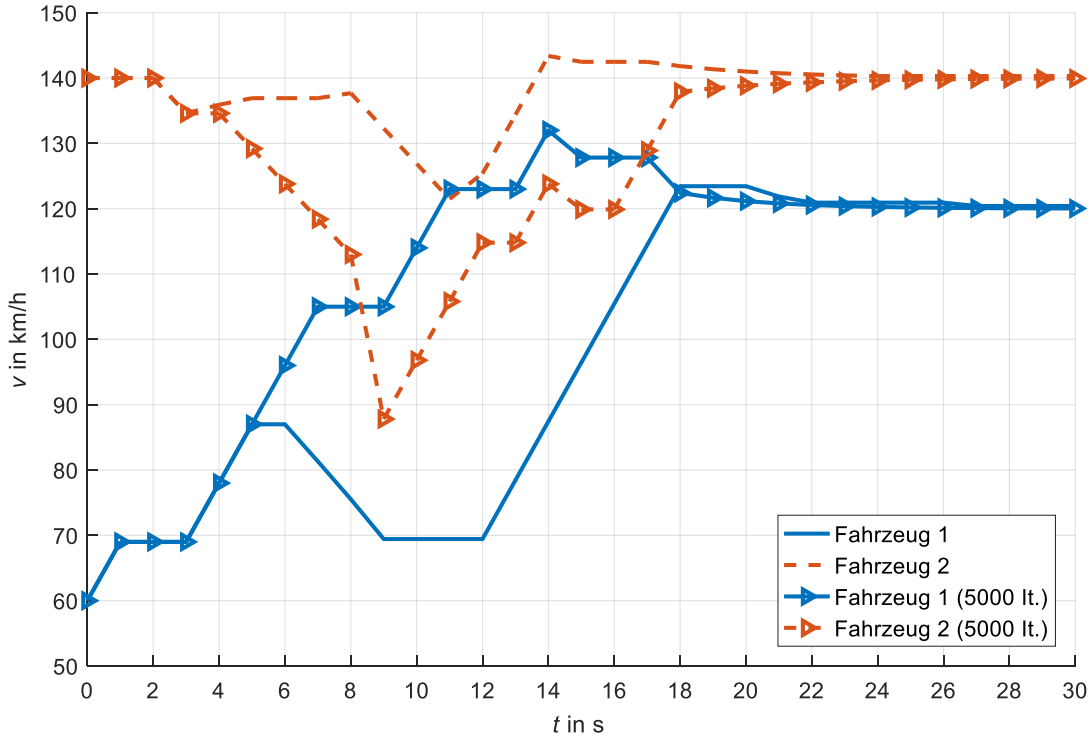


Abb. 4.22 Geschwindigkeitsverlauf der beiden Fahrzeuge in Verkehrsszenario 05

Bei der Vorstellung der Aktionen des Fahrzeugs 1 in Abb. 4.14 wurde bereits festgehalten, dass gegen Ende der Simulationszeit IDM-Beschleunigungen stattfinden, obwohl die Beschleunigung zwischenzeitlich Null ist. Die Wunschgeschwindigkeit ist zu diesem Zeitpunkt scheinbar erreicht. Bei genauer Betrachtung ist bei Zeitpunkt $t = 36$ s eine Abweichung von der Wunschgeschwindigkeit von ca. 0,02 km/h festzustellen. Mit dem IDM wird ein kontinuierlicher Wert für die Beschleunigung berechnet, der erst Null wird, wenn sich kein Hindernis voraus befindet und die Wunschgeschwindigkeit exakt erreicht ist. Letzteres ist in diesem Beispiel nicht der Fall. Daraus gehen geringe IDM-Beschleunigungswerte für die Aktion a_{IDM} hervor. Dass dennoch zwischenzeitlich die Aktion $a_{=0}$ ausgewählt wird, hat damit zu tun, dass die Kosten relativ zu den Gesamtkosten zu gering sind. Die geringen Abweichungen von der Wunschgeschwindigkeit erzeugen so geringe Kosten, dass diese nicht mehr von denen unterschieden werden können, die auftreten, wenn nicht weiter beschleunigt wird. Aus diesem Grund ist die Wahl der konkreten Aktion, ob $a_{=0}$ oder a_{IDM}, zufällig. In diesem Verkehrsszenario ist die Auswirkung dieses Effektes so gering, dass er vernachlässigt werden kann.

In Verkehrsszenario 04 fällt zu Beginn der Simulation auf, dass die Fahrzeuge 3, 4, 5 und 7 verzögern. Bei allen vier Fahrzeugen entspricht die initiale Geschwindigkeit der Wunschgeschwindigkeit. Weil sie nur auf das IDM als Aktion zurückgreifen können, kann nur ein zu geringer Abstand oder eine Geschwindigkeitsdifferenz eine Verzögerung auslösen. Bei den

Fahrzeugen 4 und 7 ist der zu geringe Abstand der Auslöser. Eine Geschwindigkeitsdifferenz liegt nicht vor. Würde der initiale Abstand zwischen beiden Fahrzeugen größer gewählt werden, kann eine Verzögerung zu Beginn vermieden werden. Für die Fahrzeuge 3 und 5 trifft beides zu. Fahrzeug 3 fährt auf Fahrzeug 2 auf, dass 20 km/h langsamer fährt. Der geringer werdende Abstand und die Geschwindigkeitsdifferenz von 20 km/h führen zu einem negativen IDM-Beschleunigungswert. Gleiches gilt für Fahrzeug 5.

Bei Betrachtung des Beschleunigungsverlaufes (Abb. 4.16) und des Abstandes zwischen Fahrzeug 4 und 7 sowie 7 und dem nächsten Hindernis (Abb. 4.17), kann ein Zusammenhang erkannt werden. Zum Startzeitpunkt ist der Unterschied der Geschwindigkeiten von Fahrzeug 4, 7 und 8 gering. Deshalb ist der Abstand für die beiden Fahrzeuge 4 und 7 die entscheidende Größe bei der Berechnung des IDM-Beschleunigungswertes. Aus Abb. 4.17 wird ersichtlich, dass der Abstand zwischen Fahrzeug 7 und 8 größer ist als der zwischen Fahrzeug 4 und 7. Deshalb verzögert Fahrzeug 4 stärker als Fahrzeug 7 (Abb. 4.16). Der Sprung im Abstand durch den Spurwechsel von Fahrzeug 1 zum Zeitpunkt $t = 10$ s ruft einen negativen IDM-Beschleunigungswert hervor. Die Geschwindigkeitsdifferenz spielt keine Rolle, weil das vorausfahrende Fahrzeug 1 schneller fährt als Fahrzeug 7. Fahrzeug 4 reagiert einen Zeitschritt versetzt auf die Verzögerung des vorausfahrenden Fahrzeugs 7. Auffällig ist, dass Fahrzeug 4 sichtbar weniger stark verzögert. Das liegt zum einen an der weiterhin geringen Geschwindigkeitsdifferenz, zum anderen daran, dass der Abstand zwischen beiden Fahrzeugen ausreichend groß ist. Der große Abstand hat in diesem Fall eine dämpfende Wirkung, weil dadurch ein gewisser Spielraum ermöglicht wird. Dieser führt dazu, dass sich die Aktionen des vorausfahrenden Fahrzeuges nur in abgeschwächter Form auf das Fahrzeug dahinter auswirken. Der gleiche Effekt ist zu den Zeitpunkten $t = 19$ s und $t = 22$ s im Beschleunigungsverlauf zu sehen. Die durchgeführte Verzögerung von Fahrzeug 7 ($a_{IDM}(t = 19\,\mathrm{s}) = 0{,}2$ m/s², $a_{IDM}(t = 22\,\mathrm{s}) = 0{,}3$ m/s²) ist bereits deutlich geringer als die von Fahrzeug 1 mit der Standard-Aktion $a_{<0} = -1{,}5$ m/s² zu beiden Zeitpunkten. Die Reaktion von Fahrzeug 4 im darauffolgenden Zeitschritt ist nicht mehr zu erkennen. Festzuhalten ist außerdem, dass die gleiche Simulation mit geringerer Schrittweite $\Delta t = 0{,}5$ s dazu führt, dass das Fahrzeug 1 den zweiten Spurwechsel nicht vollzieht (Plot im Anhang). Es wechselt auf die rechte Fahrspur und bleibt dort hinter Fahrzeug 6. Dieses Verhalten kann wieder auf den bereits erwähnten Effekt zurückgeführt werden, dass bei geringerer Schrittweite deutlich mehr Iterationen notwendig sind, um ein Ergebnis gleicher Güte zu erhalten.

Zusammenfassend lassen sich einige Punkte festhalten, die bei der Berechnung von kooperativen Fahrmanövern für definierte Verkehrsszenarien zu beachten sind. Vor allem bei der Bestimmung der Simulationsparameter und der Auswertung der Berechnungsergebnisse sind einige Aspekte zu beachten.

- Die sogenannten „Überschwinger“ treten auf, weil die Diskretisierung nicht fein genug gewählt wurde. Das ist theoretisch möglich, führt aber zu einer stark erhöhten Anzahl an wählbaren Fahrzeugaktionen und damit auch zu einem erheblichen Mehraufwand an Rechenkapazität. Die Auswirkungen auf den Simulationsverlauf und die Aussagekraft der Ergebnisse sind vernachlässigbar klein, so dass sich keine Einschränkungen daraus ergeben.
- Scheinbar grundlose Beschleunigungen oder Verzögerungen eines Fahrzeugs lassen sich darauf zurückführen, dass bei diesem Fahrzeug die wählbaren Aktionen nur geringe Kostenunterschiede aufweisen. Sind die resultierenden Kosten im Vergleich zu den gesamten Kosten mit allen beteiligten Fahrzeugen klein, kann der Algorithmus im Extremfall nicht mehr zwischen den einzelnen, ähnlichen Aktionen unterscheiden. Das

kann dazu führen, dass anstatt einer konstanten Geschwindigkeit eine geringe Beschleunigung oder Verzögerung gewählt wird. Diese wird i.d.R. unmittelbar danach ausgeglichen. Ebenfalls gilt hier, dass die Auswirkungen auf den Simulationsverlauf und auf die Aussagekraft der Ergebnisse vernachlässigbar klein sind. Aus diesem Grund ergeben sich durch den beschriebenen Effekt keine Einschränkungen.

- Eine Verfeinerung der Simulationsschrittweite zur Verbesserung des Ergebnisses ist stets kritisch zu hinterfragen. Theoretisch erhöht die Verfeinerung der Schrittweite Δt immer die Qualität des Simulationsergebnisses. Allerdings muss dazu die Anzahl an Iterationen entsprechend angehoben werden. Beide Maßnahmen wirken sich negativ auf den Rechenaufwand aus und erhöhen die notwendige Rechenzeit erheblich. Vor der Verfeinerung der Schrittweite ist dieser Aspekt stets hinzuzuziehen und im Hinblick auf eine schnelle Rechenzeit abzuwägen.

4.4 Untersuchung der Performance des Algorithmus

Dieses Unterkapitel dient dazu, die berechneten Verkehrsszenarios hinsichtlich der Rechenzeit und des Rechenaufwands zu untersuchen. Auftretende Abhängigkeiten zwischen einzelnen Parametern und dem Rechenaufwand werden dazu herausgearbeitet und erklärt. Für alle vier vorgestellten Verkehrsszenarien und dem zusätzlich eingeführten Verkehrsszenario 05 werden die Simulationszeit und die expandierten Knoten dargestellt, jeweils für die gesamte Simulation und für einzelne Durchläufe der Iterationsschleifen.

Für Verkehrsszenario 01 sind in Tabelle 4.5 die Anzahl der insgesamt expandierten Knoten und die dafür benötigte Rechenzeit eingetragen. Daraus berechnet sich der ebenfalls eingetragene Wert für die durchschnittliche Anzahl an expandierten Knoten pro Sekunde.

Tabelle 4.5 Gesamte Rechenzeit, expandierte Anzahl an Knoten und die durchschnittliche Anzahl expandierter Knoten pro s für die berechneten Varianten des Verkehrsszenario 01

	Gesamtzahl an expandierten Knoten	gesamte Rechenzeit in s	durchschnittliche Anzahl expandierter Knoten pro s
Variante 1 (zwei Fahrspuren, $x_2 = 100$ m)	1 380	1,1	1 255
Variante 2 (eine Fahrspur, $x_2 = 100$ m)	86 940	70,8	1 228
Variante 3 (zwei Fahrspuren, $x_2 = 50$ m)	583	0,5	1 088
Variante 4 (eine Fahrspur, $x_2 = 50$ m)	41 702	30,6	1 356

Jede der vier Varianten wurde mit 1500 Iterationen simuliert. Unterschiedlich ist nur die Anzahl der Fahrspuren und die x-Startposition von Fahrzeug 2. Sind zwei Fahrspuren vorhanden, findet der Suchalgorithmus im Vergleich zu den Varianten mit nur einer Fahrspur viel schneller die optimale Lösung. Die Rechenzeit beträgt in beiden Varianten nur jeweils ungefähr 1,6 % im Vergleich zu denen mit einer Fahrspur. Die Anzahl der expandierten Knoten ist ebenfalls geringer. Das liegt daran, dass bei zwei Fahrspuren mit dem Spurwechsel von Fahrzeug 2 keine Notwendigkeit einer direkten Kooperation mehr vorliegt. Die beiden Fahrzeuge beeinflussen sich nicht negativ, wodurch sich schnell eine optimale Lösung finden lässt. Der Suchalgorithmus gelangt bereits in der ersten Iterationsschleife bis zur tiefsten Baumebene.

Die optimale Handlungsabfolge ist somit gefunden. Alle weiteren Iterationsschleifen werden übersprungen, weil eine weitere Suche nicht mehr nötig ist. In Abb. 4.23, in der die Anzahl der expandierten Knoten pro Iterationsschleife dargestellt sind, befinden sich deswegen alle weiteren Markierungen von Variante 1 im Ursprung. Nur die erste Markierung ist an der Position mit 1380 expandierten Knoten und der dafür benötigten Zeitdauer von 1,1 s. Die Rechenzeit und die expandierten Knoten bei Variante 3 verhalten sich im Vergleich dazu ähnlich. Die noch geringeren Werte können damit begründet werden, dass Fahrzeug 2 früher auf die linke Fahrspur wechseln kann, weil es 50 m früher beginnt, und damit noch weniger Konflikte mit Fahrzeug 1 besitzt als in Variante 1.

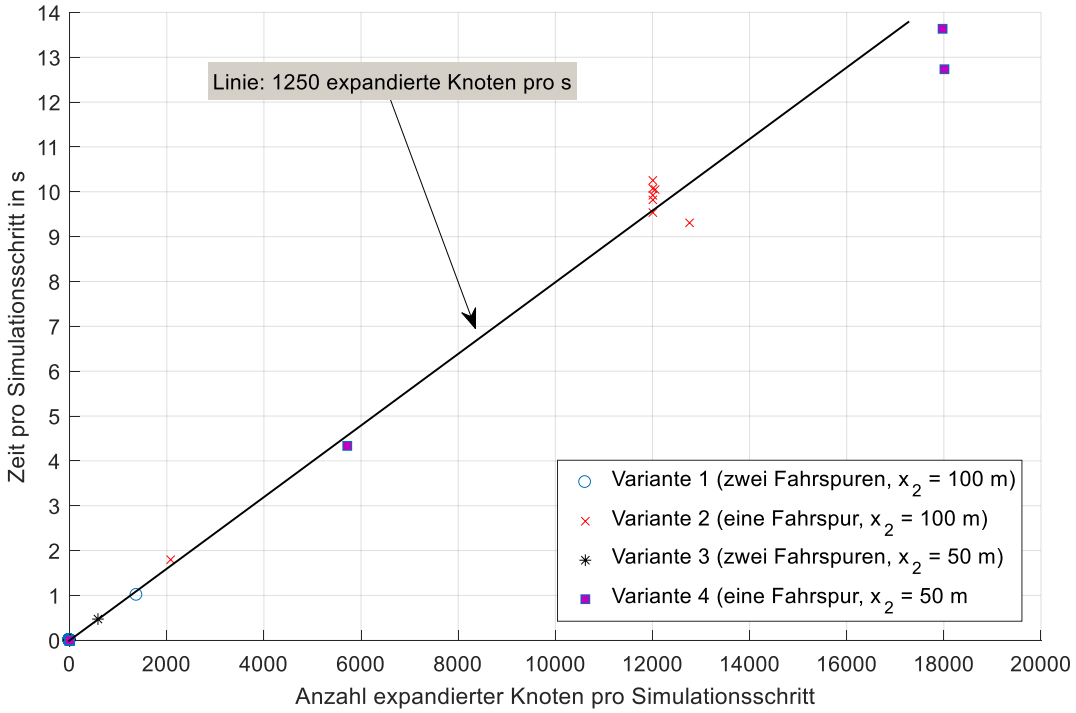


Abb. 4.23 Rechenzeit für den Durchlauf jeweils einer Iterationsschleife und Anzahl der dabei expandierten Knoten für Verkehrsszenario 01

Variante 4 ist schneller als Variante 2, weil weniger Iterationsschleifen durchlaufen werden. In der Abbildung ist zu sehen, dass Variante 4 nur zwei Iterationsschleifen komplett durchläuft und bereits in der dritten zur untersten Ebene des Suchbaumes gelangt. Die restlichen Markierungen befinden sich im Ursprung des Koordinatensystems. Variante 2 gelangt erst im achten Iterationsdurchlauf dort hin. Das liegt daran, dass in Variante 2 Fahrzeug 2 erst an Fahrzeug 1 vorbeifährt, bevor letzteres auf die rechte Fahrspur wechselt. Bis zu diesem Spurwechsel werden vom Algorithmus ständig alle möglichen Aktionen untersucht, darunter auch ein Spurwechsel. In Variante 4 ist ab dem frühen Spurwechsel von Fahrzeug 1 die Anzahl an möglichen Aktionen geringer, weil keines der beiden Fahrzeuge mehr die Fahrspur wechseln kann. Dadurch verringert sich auch der Rechenaufwand und damit die Rechenzeit. Die durchschnittliche Anzahl an expandierten Knoten pro Sekunde beträgt ca. 1250 und wird im Folgenden als Referenzwert verwendet.

Für Verkehrsszenario 02 wurde bei gleichbleibender Iterationsanzahl die Schrittweite der Simulation halbiert. Das bedeutet, dass doppelt so viele Iterationsschleifen durchlaufen werden müssen. In Tabelle 4.6 sind die Daten der Simulation dargestellt und in Abb. 4.24 ist die graphische Darstellung der Anzahl der expandierten Knoten pro Iterationsschleife zu sehen. In Variante 2 werden ungefähr dreimal so viele Knoten expandiert wie in Variante 1, für die Rechenzeit gilt ein ähnliches Verhältnis.

Tabelle 4.6 Gesamte Rechenzeit, expandierte Anzahl an Knoten und die durchschnittliche Anzahl expandierter Knoten pro s für die berechneten Varianten des Verkehrsszenario 02

	Gesamtzahl an expandierten Knoten	gesamte Rechenzeit in s	durchschnittliche Anzahl expandierter Knoten pro s
Variante 1 (1500 Iterationen, 1 s Zeitschritt)	301 211	286,3	1052
Variante 2 (1500 Iterationen, 0,5 s Zeitschritt)	849 620	1 005,7	845

Die Anzahl an expandierten Knoten pro Zeitschritt ist für beide Varianten größtenteils identisch, wie sich aus Abb. 4.24 entnehmen lässt. Das ist plausibel, weil das Verkehrsszenario an sich gleich bleibt.

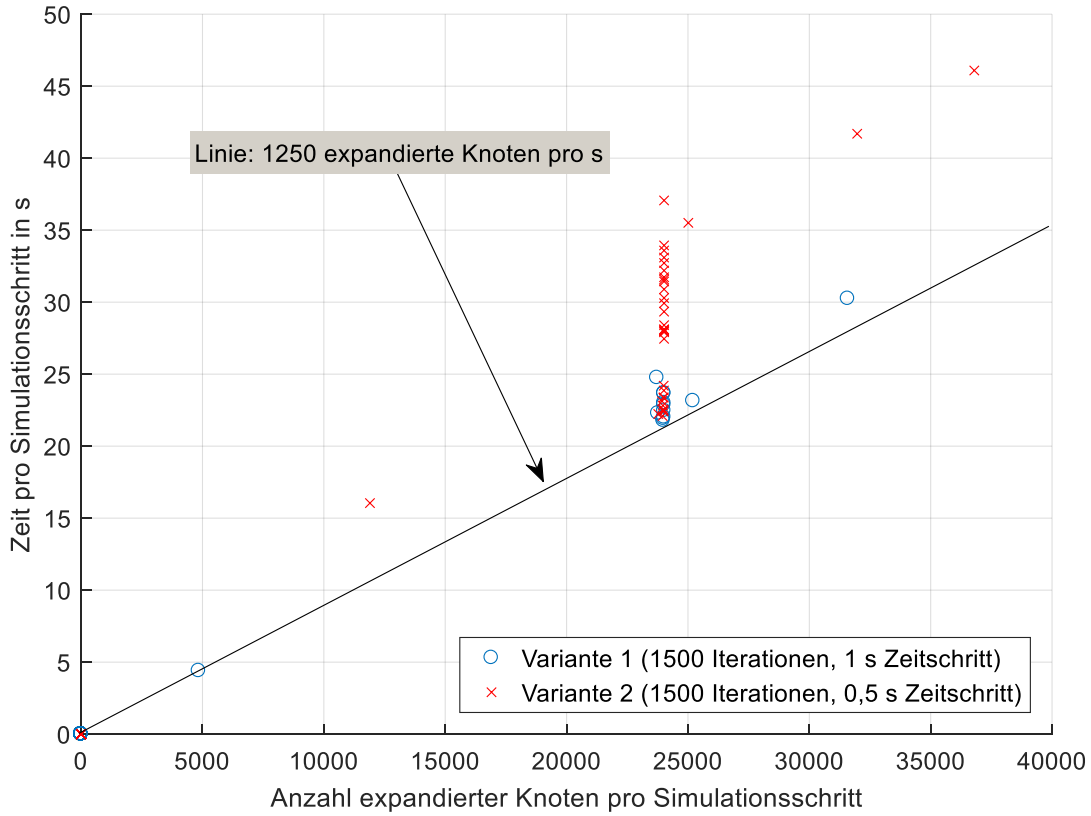


Abb. 4.24 Rechenzeit für den Durchlauf jeweils einer Iterationsschleife und Anzahl der dabei expandierten Knoten für Verkehrsszenario 02

Nur in wenigen Fällen unterscheiden sich die Anzahl der expandierten Knoten stark. Auffallend ist, dass die Rechenzeit für gleich viele zu expandierende Knoten nicht konstant ist. Für beide Varianten driftet die benötigte Zeit nach oben ab. Die Ursache ist die steigende Zahl an gespeicherten Knoten und deren Zuständen. Es muss eine wachsende Datenmenge verwaltet werden. Außerdem muss die Speicherbelegung gegebenenfalls erweitert werden, was zusätzlich Zeit erfordert. Insgesamt ist die durchschnittliche Anzahl an expandierten Knoten pro Sekunde mit 1 052 bzw. 845 unterhalb dem zuvor festgelegten Referenzwert. Im Vergleich zu Verkehrsszenario 01 ist das Fahrzeug 3 für die Simulation-Phase relevant, obwohl es nur eine konstante Geschwindigkeit ausführen kann. Die Bewertung mittels des IDM wird trotzdem durchgeführt, was zusätzlichen Aufwand bedeutet.

Verkehrsszenario 03 ist das komplexeste berechnete Verkehrsszenario, das in dieser Arbeit vorgestellt wird. Jeder Knoten, der expandiert wird, hat eine maximale Anzahl an Kindknoten von $6^5 = 7776$. Die Basis des Ausdrucks ist die Anzahl der möglichen wählbaren Aktionen, im

Exponent steht die Anzahl der Fahrzeuge. Fahrzeug 4 hat keinen Einfluss, weil es nur eine Aktion ausführen kann. Das entspricht einer Multiplikation mit 1. Durch die Tatsache, dass für einen ausgewählten Knoten stets alle seine Kindknoten expandiert werden, werden bereits bei einem Iterationsschritt viel mehr Knoten expandiert als bei einem Szenario mit weniger Fahrzeugen. Aus diesem Grund führt schon eine geringe Anzahl an Iterationen, in diesem Fall 50, zu einem hohen Rechenaufwand. In Tabelle 4.7 sind die gesamte Rechenzeit, die gesamte expandierte Anzahl an Knoten und die durchschnittliche Anzahl expandierter Knoten pro Sekunde eingetragen. In Variante 1 werden mit nur 50 Iterationen mehr Knoten expandiert als in Verkehrsszenario 02 mit 1500 Iterationen. Die durchschnittliche Anzahl expandierter Knoten pro Sekunde ist in Verkehrsszenario 03 geringer. Daraus folgt, dass die Geschwindigkeit, mit der eine gewisse Anzahl an Knoten expandiert werden kann, nicht von der Höhe der Knoten-Anzahl selbst abhängen kann. Sonst wäre dieser Wert in beiden Verkehrsszenarien ähnlich. Entscheidend ist die Anzahl der beteiligten Fahrzeuge. Jedes Fahrzeug bedeutet einen zusätzlichen Aufwand in der Expansion- und Simulation-Phase, die unabhängig von der Anzahl an wählbaren Aktionen für jedes beteiligte Fahrzeug gleich abläuft.

Tabelle 4.7 Gesamte Rechenzeit, expandierte Anzahl an Knoten und die durchschnittliche Anzahl expandierter Knoten pro s für die berechneten Varianten des Verkehrsszenario 03

	Gesamtzahl an expandierten Knoten	gesamte Rechenzeit in s	durchschnittliche Anzahl expandierter Knoten pro s
Variante 1 (50 Iterationen, 1 s Zeitschritt)	389 230	715,2	544
Variante 2 (50 Iterationen, 0,5 s Zeitschritt)	3 510 039	6 734,5	521

Durch die Halbierung des Zeitschritts wird die Tiefe des Suchbaumes doppelt so tief. Das äußert sich in einer höheren Rechenzeit und einer größeren Anzahl an expandierten Knoten von Variante 2 gegenüber Variante 1. Beide Werte sind bei Variante 2 ca. um den Faktor 10 größer. In Abb. 4.25 ist die Anzahl der expandierten Knoten pro Iterationsschleife dargestellt.

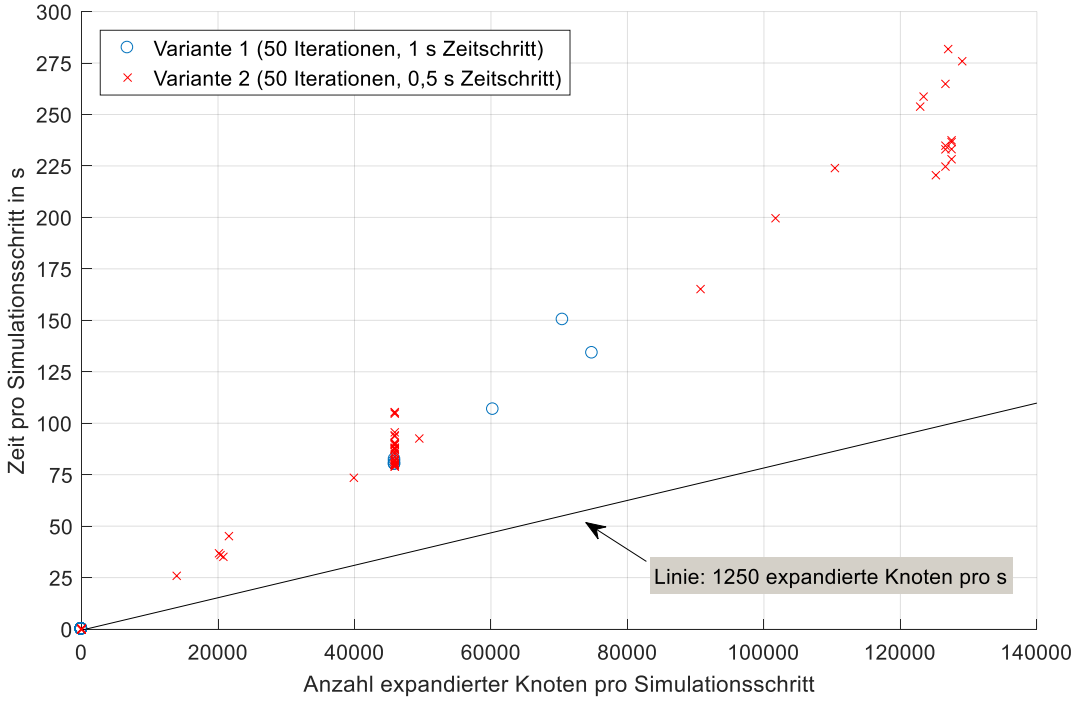


Abb. 4.25 Rechenzeit für den Durchlauf jeweils einer Iterationsschleife und Anzahl der dabei expandierten Knoten für Verkehrsszenario 03

Während bei Variante 1 die höchste Anzahl an expandierten Knoten in einem Zeitschritt bei ca. 78 000 liegt, sind es bei Variante 2 ungefähr 130 000 Knoten. In diesem Größenbereich befinden sich viele Iterationsschleifen bei Variante 2, wodurch die insgesamt hohe Anzahl aller Knoten zu Stande kommt. Variante 1 erreicht in der siebten Iterationsschleife (von insgesamt 40) die unterste Ebene des Suchbaumes nach einer Rechenzeit von 715 s. Bei Variante 2 ist das erst in der 53. Iterationsschleife (von insgesamt 80) nach 6 735 s der Fall.

Verkehrsszenario 04 beinhaltet zwar mehr Fahrzeuge als Verkehrsszenario 03, allerdings können nur zwei von den acht Fahrzeugen aus dem vollständigen Set an Aktionen wählen. Alle anderen Fahrzeuge müssen den IDM-Beschleunigungswert ausführen. Das spiegelt sich in der Gesamtzahl der expandierten Knoten aus Tabelle 4.8 wider.

Tabelle 4.8 Gesamte Rechenzeit, expandierte Anzahl an Knoten und die durchschnittliche Anzahl expandierter Knoten pro s für die berechneten Varianten des Verkehrsszenario 04

	Gesamtzahl an expandierten Knoten	gesamte Rechenzeit in s	durchschnittliche Anzahl expandierter Knoten pro s
Variante 1 (1000 Iterationen, 1 s Zeitschritt)	453 788	1305,0	348
Variante 2 (1000 Iterationen, 0,5 s Zeitschritt)	1 065 332	2 945,5	362

Pro expandiertem Knoten gibt es deutlich weniger Kindknoten. Die maximale Anzahl der neuen Zustände berechnet sich mit $6^2 = 36$. Alle Fahrzeuge, die nur eine Aktion ausführen können, entsprechen einer Multiplikation mit 1. In Abb. 4.26 ist die Anzahl der expandierten Knoten pro Iterationsschleife graphisch dargestellt.

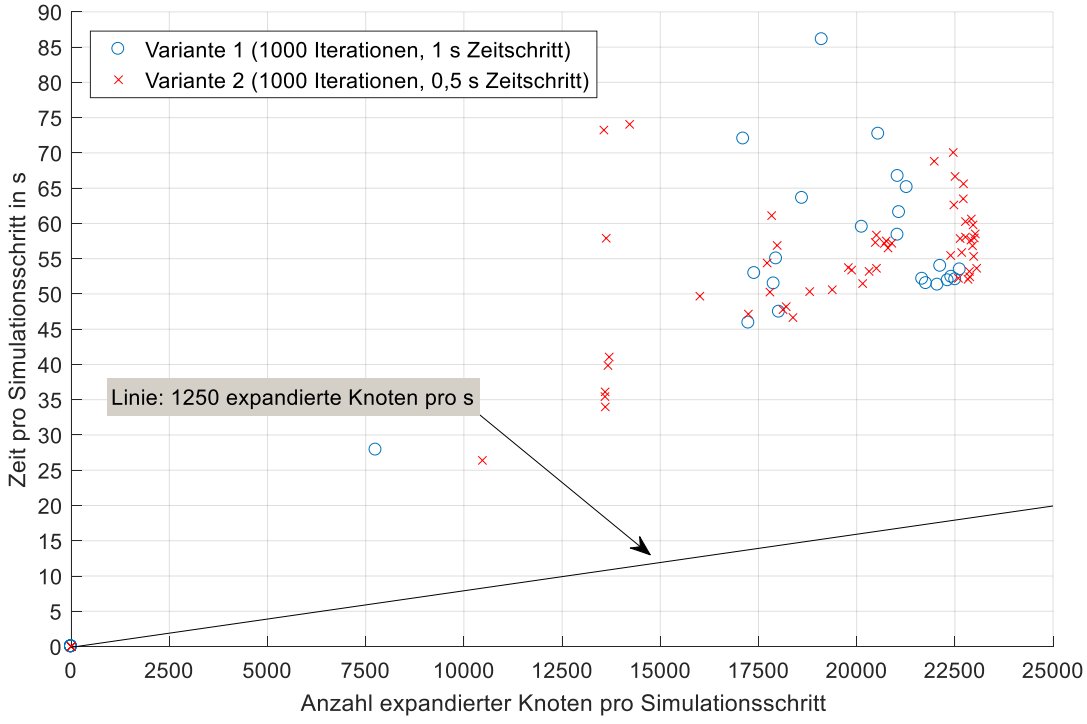


Abb. 4.26 Rechenzeit für den Durchlauf jeweils einer Iterationsschleife und Anzahl der dabei expandierten Knoten für Verkehrsszenario 04

Auffällig ist, dass die Anzahl an expandierten Knoten pro Sekunde noch niedriger ist als in Verkehrsszenario 03. Der Grund dafür ist, dass jedes Fahrzeug, unabhängig wie viele Aktionen es zur Verfügung hat, in der Expansion- und Simulation-Phase berücksichtigt werden muss. In der Expansion-Phase muss für jedes Fahrzeug der neue Zustand berechnet werden.

In der Simulation-Phase wird für jedes Fahrzeug mit dem IDM ein vordefinierter Zeitraum simuliert, damit der neue Zustand durch die Kostenfunktionen bewertet werden kann. Das führt zu einem Mehraufwand, der die Geschwindigkeit, mit der die Knoten expandiert werden, senkt. Auch ist für beide Varianten ein Abdriften der Geschwindigkeit, mit der die Knoten expandiert werden, zu erkennen. Eine gleiche oder ähnliche Anzahl an Knoten beansprucht in manchen Iterationsschleifen mehr Zeit. Grundsätzlich liegt die Anzahl der expandierten Knoten pro Simulationsschritt für beide Varianten in einem ähnlichen Wertebereich. Variation 1 gelangt in der 23. Iterationsschleife ans Ende des Suchbaumes (nach 1305 s), für Variation 2 ist dies in der 54. Iterationsschleife (nach 2946 s) der Fall.

Verkehrsszenario 05, das nur zur Erläuterung in Unterkapitel 4.3 kurz eingeführt wurde, wird ebenfalls für die Performance-Analyse untersucht. Der Hauptunterschied zu den beiden vorherigen Verkehrsszenarios ist die geringe Anzahl von nur zwei Fahrzeugen. Mit den vorherigen Beobachtungen muss sich daraus eine höhere Anzahl an expandierten Knoten pro Sekunde ergeben. Dass das der Fall ist, kann Tabelle 4.9 entnommen. In Abb. 4.27 ist die Anzahl der expandierten Knoten pro Iterationsschleife graphisch dargestellt.

Tabelle 4.9 Gesamte Rechenzeit, expandierte Anzahl an Knoten und die durchschnittliche Anzahl expandierter Knoten pro s für die berechneten Varianten des Verkehrsszenario 05

	Gesamtzahl an expandierten Knoten	gesamte Rechenzeit in s	durchschnittliche Anzahl expandierter Knoten pro s
Variante 1 (1000 Iterationen, 1 s Zeitschritt)	174 057	180,0	967
Variante 2 (5000 Iterationen, 1 s Zeitschritt)	477 000	442,6	1 078
Variante 3 (5000 Iterationen, 0,5 s Zeitschritt)	2 011 787	1 855,4	1 084
Variante 4 (10000 Iterationen, 0,5 s Zeitschritt)	3 933 158	4 068,0	967

Variante 1 benötigt am wenigsten Zeit, weil die Iterationszahl am geringsten ist. Gut miteinander vergleichbar sind Variante 2 und 3. Der einzige Unterschied ist die Größe des Zeitschritts mit $\Delta t = 1$ s bei Variante 2 und $\Delta t = 0{,}5$ s bei Variante 3. Zu beobachten ist zum einen, dass bei Variante 3 wieder die Geschwindigkeit, mit der die Knoten expandiert werden, nach oben abdriftet trotz gleichbleibender Anzahl an Knoten. Zum anderen kann festgestellt werden, dass Variante 3 genau 13 Iterationsschleifen mehr durchlaufen muss als Variante 2, bis der Suchalgorithmus zur untersten Ebene des Suchbaumes vordringt. Variante 4 besitzt bis auf einige Ausreißer eine ähnliche Geschwindigkeit, mit der die Knoten expandiert werden. Die Ausreißer können dadurch entstehen, dass bei der großen Anzahl an Knoten in diesen Zeitschritten die Speicherreservierung erweitert wurde. Insgesamt wurden bei doppelter Iterationsanzahl ca. doppelt so viele Knoten expandiert, wobei zwischen den beiden Werten kein direkter Zusammenhang besteht.

Zusammenfassend kann festgehalten werden, dass die Geschwindigkeit, mit der die Knoten expandiert werden, nicht von der Iterationszahl oder den wählbaren Aktionen der Fahrzeuge abhängt. Als entscheidend hat sich die Anzahl der am Verkehrsszenario beteiligten Fahrzeuge herausgestellt, unabhängig davon, wie viele Aktionen sie zur Wahl haben. Der negative Einfluss einer Verfeinerung des Zeitschritts wurde in den Beispielen sichtbar. Der Rechenaufwand erhöht sich in den meisten Fällen stark und damit auch die Rechenzeit. Der notwendige Ausgleich durch die Steigerung der Iterationszahl wurde dabei noch nicht berücksichtigt. Weiterhin hängt der Rechenaufwand stark von der jeweiligen initialen Verkehrssituation ab. Bereits geringe Änderungen in den Anfangszuständen der Fahrzeuge

können große Unterschiede im Rechenaufwand hervorrufen. Ist ein Verkehrsszenario so weit berechnet, dass die Notwendigkeit zur Kooperation stark absinkt oder verschwindet, so findet der Suchalgorithmus meist schnell zur untersten Ebene des Suchbaums. Auf diese Weise kann die optimale Handlungsabfolge bereits vor dem Durchlaufen der letzten Iterationsschleife gefunden werden. Das äußert sich in einer Zeitschritt-Dauer von nahe Null. Die Matlab-interne command line zeigt dann die Ausgabe „Bester Knoten gefunden“ an. Simulationen mit grober Zeitdiskretisierung dringen i.d.R. schneller bis zur untersten Baumebene vor als solche mit feiner Zeitdiskretisierung.

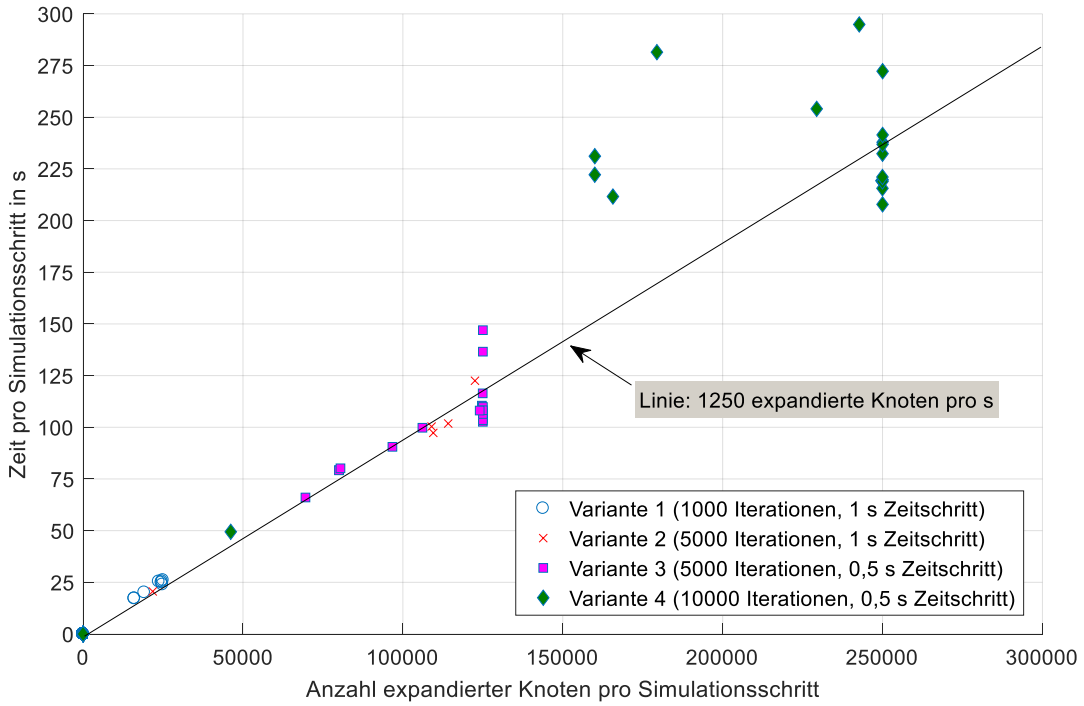


Abb. 4.27 Rechenzeit für den Durchlauf jeweils einer Iterationsschleife und Anzahl der dabei expandierten Knoten für Verkehrsszenario 05

Für die Berechnungen wurde ein PC mit einem Intel Core i5-4200U CPU@1.60 GHz Prozessor (x64-basiert) und 8 GB Arbeitsspeicher (RAM) verwendet.

4.5 Potentielle Ansatzpunkte zur Weiterentwicklung des Verfahrens

Zum Abschluss des vierten Kapitels werden Ansatzpunkte für eine Weiterentwicklung des vorgestellten Verfahrens und des dahinterliegenden Algorithmus aufgezeigt. Außerdem wird ein Vergleich zu den Arbeiten von Frese und Lenz gezogen, die eine ähnliche Problemstellung bearbeiten. Daran anschließend werden noch offene Fragen und mögliche zukünftige Entwicklungsrichtungen erläutert.

Den Rechenaufwand so gering wie möglich zu halten und dennoch zuverlässige Ergebnisse zu erhalten, stellt eine der großen Herausforderungen der Problemstellung rund um die Berechnung kooperativer Fahrmanöver dar. Interessante Verkehrsszenarien enthalten i.d.R. mehrere Fahrzeuge und müssen langfristig geplant werden, weil erst dann ein optimales kooperatives Fahrmanöver ermittelt werden kann. Genau diese beiden Anforderungen treiben

den Rechenaufwand stark nach oben. Hinzu kommt die notwendige Diskretisierung der Aktionen und der Simulationszeit bis zum Planungshorizont. Gewünscht ist eine möglichst feine Diskretisierung, weil sich dadurch eine höhere Qualität der Berechnungsergebnisse erzielen lässt. Das ist an sich umsetzbar, hat aber eine stark negative Auswirkung auf den Rechenaufwand. Bei Reduktion der Größe des Zeitschritts steigt der Rechenaufwand für gleiche Ergebnisse deutlich an, weil pro expandiertem Knoten auf Zeitebene weniger weit „in die Zukunft“ geschaut werden kann. Deswegen sind mehr Iterationen notwendig, um diesen Nachteil auszugleichen. Beide Effekte (kleinerer Zeitschritt, mehr Iterationen) erhöhen den Rechenaufwand erheblich. Vor allem Verkehrsszenarien, die einen weiten Vorausblick erfordern, sind mit einer groben Zeitdiskretisierung besser zu lösen. Ein wichtiger Bestandteil des Algorithmus ist die aus dem MCTS-Verfahren übernommene und angepasste Simulation-Phase. Durch das mit dem IDM festgelegten Standardverhalten der Fahrzeuge (default policy) wird erst ermöglicht, eine Baumstruktur dieser Größe nach der besten Handlungsabfolge abzusuchen. Wegen der Tatsache, dass es keine Heuristik gibt, mit der sich ein neuer Zustand mit seinen „Erfolgsaussichten“ abschätzen lässt, bietet die Simulation mit dem IDM erhebliche Vorteile. Problematisch ist, dass es bei der Planung kooperativer Fahrmanöver nicht möglich ist, ein eindeutiges Ziel zu definieren, das mit der Planung erreicht werden soll. Aus diesen Gründen ist eine Suche mit Verfahren, die eine Heuristik benötigen, wie bspw. dem A*-Algorithmus, nicht möglich. Mit der Simulation mithilfe des IDM und der anschließenden Bewertung des Ergebnisses lassen sich diese Probleme umgehen. Nachteilig an dieser Methode ist, dass es sich nur um das Standardverhalten der Fahrzeuge handelt. Alle Aktionen, die den Fahrzeugen eigentlich zur Verfügung stehen, wie Standardbeschleunigung und -verzögerung sowie die Möglichkeit zum Spurwechsel, werden nicht berücksichtigt. Das hat zur Folge, dass Situationen tendenziell zu schlecht bewertet werden. Unter Umständen kann sich das auf die Lösungsfindung negativ auswirken. Interessant ist daher eine Anpassung oder Weiterentwicklung der während der Simulation-Phase verwendeten default policy. Das Standardverhalten wird durch das IDM bestimmt und reagiert nur auf das unmittelbar vorausfahrende Fahrzeug. Vor allem ein Fahrermodell, das auch Spurwechsel berücksichtigt, scheint für das vorliegende Problem wünschenswert. Damit kann der jeweilige expandierte Knoten und der Zustand, den dieser repräsentiert, genauer bewertet werden.

Der Rechenaufwand lässt sich verringern, indem die Diskretisierung der Aktionen sowie der Simulationszeit grober gestaltet wird. Jedoch wurde bei der Auswahl der Aktionen bereits die minimale Anzahl verwendet. Dies beinhaltet jeweils eine Aktion mit einem festen Beschleunigungs- und Verzögerungswert, eine Aktion zum Beibehalten der aktuellen Geschwindigkeit, die IDM-Beschleunigung und die beiden Spurwechselaktionen nach links und rechts. Für die Simulationszeit ist ein Zeitschritt von einer Sekunde bereits die obere Grenze. Eine noch gröbere Schrittweite verschlechtert das Finden einer annehmbaren Lösung stark. Aus diesen Gründen kann das Verändern von Parametern, die unmittelbar in die Anzahl der Kombinationen eingehen, nicht weiterverfolgt werden.

Zum einen können Performance-Verbesserungen auf Seiten des Algorithmus erreicht werden. Frese [13, S. 176] führt dazu die gute Parallelisierbarkeit von Baumsuchalgorithmen an, wodurch bspw. einzelnen Prozessoren unterschiedliche Teilbäume zugewiesen werden können. Mit einer weiteren Performance-Steigerung ist zu rechnen, wenn der Algorithmus in einer maschinennahen Programmiersprache wie bspw. C++ implementiert wird.

Zum anderen können Maßnahmen getroffen werden, die nach der Definition des Verkehrsszenarios und vor oder während der Simulation Anwendung finden. Es handelt sich dabei um Verfahren, die die Komplexität des Planungsproblems durch zulässige Vereinfachungen

verringern. Die prioritätsbasierte Planung wurde bereits im Stand der Technik vorgestellt. Eine Anwendung auf das Planungsproblem stellte sich aus genannten Gründen als nicht erfolgsversprechend heraus. Dennoch ist der Ansatz interessant, einzelnen Fahrzeugen eine vordefinierte Trajektorie zuzuweisen und diese Fahrzeuge während der Berechnung der anderen Trajektorien nur als bewegte Hindernisse zu berücksichtigen. Das hat zur Folge, dass dieses Fahrzeug für das Planungsproblem wegfällt, weil seine Aktionen zu jeder Zeit festgelegt sind. Der Rechenaufwand verringert sich, weil die Anzahl an möglichen Kombinationen sinkt. Im aktuellen Stand des Algorithmus ist die Möglichkeit, einzelnen Fahrzeugen eine vordefinierte Aktion vorzugeben, enthalten. Es können nur eine Aktion oder aber mehrere erlaubt werden. Dies muss jedoch vom Nutzer explizit festgelegt werden. In vielen Fällen ist eine erkennbare, zulässige Einschränkung im Voraus schwer zu bestimmen. Außerdem bietet sich nicht die Möglichkeit, für ein Fahrzeug bestimmte Aktionen zu bestimmten Zeitpunkten vorzugeben. Es kann nur zu Beginn ausgewählt werden, welche Aktionen während der gesamten Simulationszeit zur Verfügung stehen sollen. Als Ansatzpunkt zur Verringerung des Rechenaufwands ist ein Algorithmus denkbar, der das vom Nutzer bestimmte Verkehrsszenario für die Berechnung aufbereitet. Den darin enthaltenen Fahrzeugen wird, wenn möglich, eine feste Trajektorie im Voraus zugewiesen oder das Set an Aktionen eingeschränkt. Daraus resultiert im ersten Fall nur ein bewegtes Hindernis, das nicht am Planungsprozess eingeschlossen ist, oder im zweiten Fall ein Fahrzeug, das weniger mögliche Kombinationen hervorruft. In beiden Fällen reduziert sich die Anzahl der möglichen Kombinationen und damit die Größe des Suchbaums und der Rechenaufwand.

Während der Simulation bieten sich ebenfalls Möglichkeiten, den Rechenaufwand zu verringern. Ein Beispiel, das Frese [13, S. 49-50] entwickelt, ist die Bildung von kooperativen Gruppen. Die Fahrzeuge in einem Verkehrsszenario werden in sogenannte kooperative Gruppen eingeteilt. Diese beinhaltet nur Fahrzeuge, die miteinander interagieren können und einer Verkehrssituation gegenüberstehen, die eine Kooperation erfordert. Aus dem Planungsproblem ausgeschlossen werden somit Fahrzeuge, die nicht berücksichtigt werden müssen. Das können bspw. Fahrzeuge auf der gegenläufigen Fahrbahn sein oder Fahrzeuge, die sich so weit weg von den anderen Fahrzeugen befinden, dass sie nicht relevant sind. Vor allem bei Fahrzeugen, die zu Beginn der Planung berücksichtigt werden müssen, aber im späteren Verlauf einen Abstand zu den restlichen Fahrzeugen aufbauen, ist die Berechnung von kooperativen Gruppen interessant. Sie können auf diese Weise, ähnlich der prioritätsbasierten Planung vor Beginn der Simulation, vom Planungsproblem ausgeschlossen werden. Das betrifft bspw. Fahrzeuge, die sich mit hoher Geschwindigkeit von der kooperativen Gruppe entfernt haben oder durch eine geringe Geschwindigkeit so weit zurückgefallen sind, dass sie nicht mehr mit den anderen Fahrzeugen interagieren können. Im aktuellen Stand des Suchalgorithmus werden alle Fahrzeuge über den gesamten Simulationszeitraum identisch behandelt und eine Unterscheidung findet nicht statt. Ausgewählte Einschränkungen für das Set an Aktionen haben über den gesamten Simulationszeitraum Bestand.

Während Frese [13] kritische Fahrsituationen betrachtet und die Kooperation zwischen Fahrzeugen zur Unfallvermeidung dient, werden in der vorliegenden Arbeit unkritische Fahrsituationen behandelt. Der Schwerpunkt bei Frese liegt auf der Betrachtung kurzer Zeiträume von wenigen Sekunden und einer großen Vielfalt an möglichen Fahrmanövern. Starke Richtungsänderungen und sogar das Verlassen der Straße sind bei ihm nicht ausgeschlossen. In dieser Arbeit werden Situationen im normalen, fließendem Verkehr bei autobahntypischer Geschwindigkeit betrachtet. Viele Vereinfachungen wurden bzgl. des Fahrzeugmodells und der durchführbaren Aktionen getroffen. Diese Vereinfachungen und der im Vergleich zu Frese lange Planungszeitraum stechen als Unterschiede besonders hervor. Weiterhin können mit

dem entwickelten Algorithmus mehr beteiligte Fahrzeuge berücksichtigt werden, ohne dass die Rechenzeit in inakzeptable Bereiche ansteigt. Außerdem kann bestimmten Fahrzeugen im Voraus ein eingeschränktes Set an Aktionen zur Wahl gestellt werden.

Das von Lenz [21] vorgestellte Verfahren zu kooperativen Fahrmanöverplanung wurde in dieser Arbeit aufgegriffen und weiterentwickelt. Viele Elemente, wie bspw. die Anwendung des MCTS-Algorithmus und des IDM, wurden übernommen und angepasst oder neu entwickelt. Der Unterschied zu Lenz besteht darin, dass alle Fahrzeuge miteinander kommunizieren können. Lenz schließt das explizit aus und konzentriert sich auf die Fahrmanöverplanung des Ego-Fahrzeugs unter Berücksichtigung aller umgebenden Fahrzeuge. In der vorliegenden Arbeit findet eine V2V-Kommunikation explizit statt, wodurch sich die Fahrzeuge über kooperative Fahrmanöver abstimmen können. Angenommen wird, dass alle Fahrzeuge die Möglichkeit haben, an dieser Absprache teilzunehmen. Interessant für zukünftige Entwicklungsrichtungen aufbauend auf dem entwickelten Algorithmus sind folgende Aspekte:

- Performance-Steigerung des Algorithmus
- Bestimmung von festen Trajektorien für bestimmte Fahrzeuge bereits im Voraus der eigentlichen Berechnung
- Einteilung von Fahrzeugen eines Verkehrsszenarios in kooperative Gruppen
- Ausarbeitung der bereits implementierten Kostenfunktionen und Entwicklung von neuen Gütemaßen zur genaueren Bewertung der Fahrmanöver
- Berücksichtigung von Unsicherheiten bzgl. der Fahrzeugzustände Position und Geschwindigkeit
- Anwendung von weiteren spieltheoretischen Ansätzen und Nutzung des rasanten Entwicklungsfortschritts in diesem Bereich

5 Zusammenfassung und Ausblick

Für die Entwicklung des Algorithmus zur Planung kooperativer Fahrmanöver wurde zu Beginn der vorliegenden Arbeit eine umfassende Literaturrecherche durchgeführt. Diese wird in Kapitel 2 „Stand der Technik“ ausgearbeitet. Das beinhaltet Grundlagen der Trajektorienplanung in der Robotik allgemein und speziell im Hinblick auf Einzelfahrzeuge. Darauf aufbauend werden Verfahren vorgestellt, die sich besonders für die kooperative Trajektorienplanung von Straßenfahrzeugen eignen. Kapitel 2 behandelt außerdem die Motivation für kooperatives Fahren und die aktuellen Herausforderungen. Auf die Komplexität des vorliegenden Planungsproblems wird ebenfalls eingegangen. Basierend auf der detaillierten Vorstellung der Verfahren zur kooperativen Trajektorienplanung werden in Unterkapitel 3.1 deren Vor- und Nachteile miteinander verglichen, um das beste Verfahren für die Problemstellung auszuwählen und weiter zu verfolgen. Die Wahl fällt auf die Monte-Carlo Baumsuche. Die Kernelemente der Monte-Carlo Baumsuche werden in Unterkapitel 3.2 erläutert und vorgenommene Anpassungen oder Neuentwicklungen einzelner Elemente werden vorgestellt. Im Anschluss daran wird die konkrete Implementierung des Algorithmus in Matlab dargelegt. Eingegangen wird auf Besonderheiten der Matlab-Software und generell auf die Ausgestaltung der Programmstruktur sowie den Aufbau der graphischen Benutzeroberfläche. Kapitel 4 beschäftigt sich mit der Vorstellung und Diskussion der Berechnungsergebnisse des entwickelten Verfahrens. Zuerst werden beispielhafte Verkehrsszenarien definiert und anschließend die Ergebnisse der Berechnungen dargestellt. Diese werden in Unterkapitel 4.3 analysiert, bewertet und interpretiert. Besonderheiten und auffällige Merkmale in den Ergebnissen werden mit Bezug auf den Algorithmus erklärt. In den letzten beiden Unterkapiteln wird die Performance des entwickelten Algorithmus untersucht und auf potentielle Ansatzpunkte zur Weiterentwicklung des Verfahrens eingegangen.

Mit dieser Arbeit wird ein neues Verfahren vorgestellt, mit dem kooperative Fahrmanöver von Fahrzeugen geplant werden können. Dazu wird die Monte-Carlo Baumsuche auf das Problem der kooperativen Fahrmanöverplanung angepasst. Die behandelten Verkehrsszenarien finden in fließendem Verkehr bei höheren Geschwindigkeiten auf mehrspurigen Fahrbahnen, wie bspw. Autobahnabschnitten, statt. Mit dem entwickelten Algorithmus werden Handlungsabfolgen für alle beteiligten Fahrzeuge einer Situation, die eine Kooperation erfordert, bestimmt. Ziel ist stets, ein optimales Fahrmanöver, im Sinne des definierten Gütemaßes, zu finden. Die Kooperation entsteht dadurch, dass diese Kosten nicht für jedes Fahrzeug einzeln, sondern die Gesamtkosten des Systems minimal sein müssen. Bisher entwickelte Verfahren beschäftigen sich überwiegend mit der Trajektorienplanung in kritischen Verkehrssituationen und zur Unfallvermeidung. Der Planungshorizont beträgt dabei nur wenige Sekunden. Für die Trajektorienplanung in unkritischen Verkehrssituationen ist eine deutlich längerfristige Planung nötig, weil sich die meisten Fahrmanöver erst nach längeren Zeiträumen als optimal herausstellen. Das wird mit dem entwickelten Algorithmus berücksichtigt. Der Planungshorizont beträgt dabei meist mehr als 20 Sekunden. Mit der im Rahmen dieser Arbeit entwickelten Software können Verkehrsszenarien schnell erstellt und einfach verwaltet werden. Die Ergebnisse lassen sich nach der Berechnung in einer graphischen Benutzeroberfläche visualisieren und auswerten.

Für die zukünftige Entwicklung der kooperativen Fahrmanöverplanung mit dem neuen Verfahren scheint vor allem die Steigerung der Performance interessant. Das beinhaltet zum einen die Verwendung einer leistungsfähigeren Programmiersprache, einer optimierten Verwaltung der großen Datenmengen und vor allem den Einsatz deutlich leistungsfähigerer Hardware. Zum anderen können durch die Zuweisung von festen Trajektorien an bestimmte Fahrzeuge oder durch die Einschränkung der wählbaren Aktionen bereits vor der eigentlichen Fahrmanöverplanung große Einsparungen im Rechenaufwand erreicht werden. Die dazu notwendigen Algorithmen scheinen ebenfalls ein interessanter Aspekt für weitere Entwicklungen zu sein.

Letztendlich ist für den Einsatz eines solchen Verfahrens der Ausbau der Intrastruktur zur Kommunikation zwischen Fahrzeugen zwingend erforderlich. Voraussetzung ist zudem, dass die Fahrzeuge, die autonom fahren können und in der Lage sind, miteinander zu kommunizieren und sich abzustimmen, einen sehr hohen Marktanteil besitzen. Die Annahme dieser Arbeit ist, dass an den Verkehrsszenarien nur Fahrzeuge beteiligt sind, die an der Abstimmung der Fahrmanöver teilnehmen können und sich daran strikt halten. Nicht betrachtet werden Fahrzeuge, die keinen Zugriff darauf haben oder von einem menschlichen Fahrer gesteuert werden. Wie sich bspw. alleine ein Fahrzeug ohne das System auf den Planungsvorgang auswirkt, wird nicht untersucht.

Mit steigendem Anteil an autonomen Fahrzeugen im Verkehrsgeschehen wächst die Notwendigkeit, dass Fahrzeuge nicht nur auf sich bezogen, sondern aufeinander abgestimmt handeln. Eine koordinierte Fahrweise zwischen allen Verkehrsteilnehmern wird unumgänglich, um den Verkehrsfluss aufrecht zu erhalten. Diese Koordination fiel selbst den bereits erwähnten Vorgängern des Automobils schwer. Die Aussage, dass „Das autonome Automobil […] dem Fahrzeug seine verloren gegangene Autonomie zurückgeben, ja die historische Form noch weit übertreffen [will]“, trifft zu. Vielmehr stellt es jedoch eine zwingende Notwendigkeit dar, dass die historische Form weit übertroffen wird. Andernfalls rücken die Visionen des vollautomatisierten Verkehrs der Zukunft in weite Ferne. Diese Arbeit soll einen Beitrag dazu leisten, dass das heutige Automobil wieder seiner ursprünglichen Bedeutung gerecht wird und die Vision des Verkehrs der Zukunft ein Stück mehr Wirklichkeit wird.

Abbildungsverzeichnis

Tabellenverzeichnis

Literaturverzeichnis

[1] Dudenredaktion (o. J.), *„Tautologie“ auf Duden online.* [Online] Verfügbar: https://www.duden.de/node/758064/revisions/1389897/view. Gefunden am: 02. Nov. 2017.

[2] Dudenredaktion (o. J.), *„Auto“ auf Duden online.* [Online] Verfügbar: https://www.duden.de/node/720712/revisions/1267391/view. Gefunden am: 02. Nov. 2017.

[3] Dudenredaktion (o. J.), *„Mobil“ auf Duden online.* [Online] Verfügbar: https://www.duden.de/node/644057/revisions/1670376/view. Gefunden am: 02. Nov. 2017.

[4] H.-H. Braess und U. Seiffert, Hrsg, *Vieweg Handbuch Kraftfahrzeugtechnik*, 7. Auflage. Wiesbaden: Springer Vieweg, 2013.

[5] M. Maurer, J. C. Gerdes, B. Lenz, und H. Winner, *Autonomes Fahren: Technische, rechtliche und gesellschaftliche Aspekte*, 1. Auflage. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015.

[6] S. Viehmann, *Echte Roboter-Funktionen statt Mogelpackung: Neuer A8 im Autobahn-Test.* [Online] Verfügbar: http://www.focus.de/auto/fahrberichte/audi-a8-stau-pilot-im-test-echte-roboter-funktionen-statt-mogelpackung-neuer-a8-im-autobahn-test_id_7539414.html. Gefunden am: 02. Nov. 2017.

[7] H. Winner, S. Hakuli, F. Lotz, und C. Singer, Hrsg, *Handbuch Fahrerassistenzsysteme: Grundlagen, Komponenten und Systeme für aktive Sicherheit und Komfort*, 3. Auflage. Wiesbaden: Springer Vieweg, 2015.

[8] IMAGinE Konsortium, *IMAGinE – Lösungen für kooperatives Fahren.* [Online] Verfügbar: https://imagine-online.de/home/. Gefunden am: 02. Nov. 2017.

[9] E. Spieß, *"Kooperation" aus Dorsch – Lexikon der Psychologie.* [Online] Verfügbar: https://portal.hogrefe.com/dorsch/kooperation/. Gefunden am: 03. Nov. 2017.

[10] Y. U. Cao, A. S. Fukunaga, und A. B. Kahng, “Cooperative Mobile Robotics: Antecedents and Directions,” *Autonomous Robots*, Bd. 4, S. 7–27.

[11] M. Randelhoff, *Die drei Haupttheoreme der Stauforschung: Der Schmetterlingseffekt, unsichtbare Wellen (= Phantomstau) und die Tragik des Zufalls.* [Online] Verfügbar: https://www.zukunft-mobilitaet.net/3344/strassenverkehr/wie-entstehen-staus-phantomstau/. Gefunden am: 03. Nov. 2017.

[12] S. Ulbrich *et al,* “Structuring Cooperative Behavior Planning Implementations for Automated Driving,” in *2015 IEEE 18th International Conference on Intelligent Transportation Systems (ITSC 2015),* Las Palmas, Gran Canaria, Spain, 2015, S. 2159–2165.

[13] C. Frese, *Planung kooperativer Fahrmanöver für kognitive Automobile*. Dissertation. Karlsruhe, Hannover: KIT Scientific Publishing; Technische Informationsbibliothek u. Universitätsbibliothek, 2012.

[14] D. Muoio, *Self-driving cars could be terrible for traffic - here's why.* [Online] Verfügbar: http://www.businessinsider.de/self-driving-cars-traffic-congestion-2017-6?r=US&IR=T. Gefunden am: 04. Nov. 2017.

[15] C. Cox und A. Hart, *How autonomous vehicles could relieve or worsen traffic congestion: Whitepaper of Here and SBD.* [Online] Verfügbar: https://www.sbdautomotive.com/c/here/traffic_congestion_autonomous_vehicles/media/index.html. Gefunden am: 03. Nov. 2017.

[16] D. Charanzová, *How to teach Automated Cars to break the law: the bumpy road ahead for regulation of Connected Vehicles.* [Online] Verfügbar: https://www.2025ad.com/latest/driverless-cars-regulation/. Gefunden am: 05. Nov. 2017.

[17] A. Marshall, *Puny Humans Still See the World Better Than Self-Driving Cars.* [Online] Verfügbar: https://www.wired.com/story/self-driving-cars-perception-humans/. Gefunden am: 05. Nov. 2017.

[18] G. Duranton und M. A. Turner, “The Fundamental Law of Road Congestion: Evidence from US cities,” National Bureau of Economic Research, Cambridge, MA, USA. [Online] Verfügbar: http://www.nber.org/papers/w15376. Gefunden am: 04. Nov. 2017.

[19] S. M. LaValle, *Planning algorithms*, 1. Auflage. Cambridge: Cambridge University Press, 2006.

[20] J. Ziegler, “Optimale Bahn- und Trajektorienplanung für Automobile,” Dissertation, Fakultät für Maschinenbau, Karlsruher Instituts für Technologie, Karlsruhe, 2015.

[21] D. Lenz, T. Kessler, und A. Knoll, “Tactical Cooperative Planning for Autonomous Highway Driving using Monte-Carlo Tree Search,” in *2016 IEEE Intelligent Vehicles Symposium (IV),* Gothenburg, Sweden, 2016, S. 447–453.

[22] N. Straumann, *Theoretische Mechanik: Ein Grundkurs über klassische Mechanik endlich vieler Freiheitsgrade*, 2. Aufl. Berlin: Springer Spektrum, 2015.

[23] S. M. LaValle und J. J. Kuffner, JR, “Randomized Kinodynamic Planning,” in *1999 IEEE International Conference on Robotics and Automation,* May 1999, S. 473–479.

[24] D. Heß, M. Althoff, und T. Sattel, “Formal Verification of Maneuver Automata for Parameterized Motion Primitives,” in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems,* Chicago, Illinois, USA, 2014, S. 1474–1481.

[25] E. Frazzoli, M. A. Dahleh, und E. Feron, “Maneuver-based motion planning for nonlinear systems with symmetries,” *IEEE Trans. Robot,* Bd. 21, Rn. 6, S. 1077–1091.

[26] S. Manzinger, M. Leibold, und M. Althoff, “Kooperative Bewegungsplanung autonomer Fahrzeuge unter Verwendung von Manöver-Templates,” in *AAET - Automatisiertes und Vernetztes Fahren: Beiträge zum gleichnamigen 18. Braunschweiger Symposium vom 8. und 9. Februar 2017*Braunschweig: ITS automotive nord e.V, 2017.

[27] W. Schwarting und P. Pascheka, “Recursive Conflict Resolution for Cooperative Motion Planning in Dynamic Highway Traffic,” in *2014 IEEE 17th International Conference on Intelligent Transportation Systems (ITSC),* Qingdao, China, 2014, S. 1039–1044.

[28] P. Resende und F. Nashashibi, “Real-time Dynamic Trajectory Planning for Highly Automated Driving in Highways,” in *2010 13th International IEEE Conference on Intelligent Transportation Systems (ITSC),* Funchal, Madeira Island, Portugal, 2010, S. 653–658.

[29] I. Papadimitriou und M. Tomizuka, “Fast Lane Changing Computations using Polynomials,” in *Proceedings of the 2003 American Control Conference,* Denver, Colorado, USA, 2003, S. 48–53.

[30] T. Hansen, M. Schulz, M. Knoop, und U. Konigorski, “Trajektorienplanung für automatisierte Fahrstreifenwechsel,” *ATZ - Automobiltechnische Zeitschrift*, Bd. 118, Rn. 7-8/2016, S. 66–71.

[31] S. Quinlan und O. Khatib, “Elastic bands: Connecting path planning and control,” in *1993 IEEE International Conference on Robotics and Automation,* Atlanta, GA, USA, 1993, S. 802–807.

[32] C. Frese, T. Batz, und J. Beyerer, *Kooperative Bewegungsplanung zur Unfallvermeidung im Straßenverkehr mit der Methode der elastischen Bänder.* [Online] Verfügbar: https://www.researchgate.net/publication/226067667_Kooperative_Bewegungsplanung_zur_Unfallvermeidung_im_Strassenverkehr_mit_der_Methode_der_elastischen_Bander. Gefunden am: 09. Nov. 2017.

[33] S. R. Petti, “Safe navigation within dynamic environments: a partial a partial motion planning approach,” Dissertation, École Nationale Supérieure des Mines de Paris, Paris, 2007.

[34] S. Bouraine, T. Fraichard, O. Azouaoui, und H. Salhi, “Passively safe partial motion planning for mobile robots with limited field-of-views in unknown dynamic environments,” in *2014 IEEE International Conference on Robotics and Automation,* Hong Kong, China, 2014, S. 3576–3582.

[35] G. Sierksma, *Linear and integer programming: Theory and practice*, 2. Auflage. New York, NY, USA: Marcel Dekker, 2002.

[36] J. Kallrath, *Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis; mit Fallstudien aus Chemie, Energiewirtschaft, Papierindustrie, Metallgewerbe, Produktion und Logistik*, 2. Auflage. Wiesbaden: Springer, 2013.

[37] A. Richards und J. P. How, “Aircraft trajectory planning with collision avoidance using mixed integer linear programming,” in *Proceedings of the 2002 American Control Conference,* Anchorage, AK, USA, 2002, 1936-1941.

[38] J. Leng, J. Liu, und H. Xu, “Online Path Planning Based on MILP for Unmanned Surface Vehicles,” in *Oceans - San Diego, 2013,* San Diego, California, USA, 2013.

[39] S. A. Fayazi, A. Vahidi, und A. Luckow, “Optimal scheduling of autonomous vehicle arrivals at intelligent intersections via MILP,” in *2017 American Control Conference (ACC),* Seattle, WA, USA, 2017, S. 4920–4925.

[40] T. Sattel und T. Brandt, “Ground vehicle guidance along collision-free trajectories using elastic bands,” in *Proceedings of the 2005, American Control Conference, 2005,* Portland, OR, USA, 2005, S. 4991–4996.

[41] J. Hilgert, K. Hirsch, T. Bertram, und M. Hiller, “Emergency path planning for autonomous vehicles using elastic band theory,” in *2003 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM 2003),* Kobe, Japan, 2003, S. 1390–1395.

[42] M. Keller, F. Hoffmann, C. Hass, T. Bertram, und A. Seewald, “Planning of Optimal Collision Avoidance Trajectories with Timed Elastic Bands,” *IFAC Proceedings Volumes*, Bd. 47, Rn. 3, S. 9822–9827.

[43] C. Götte *et al,* “A model predictive combined planning and control approach for guidance of automated vehicles,” in *2015 IEEE International Conference on Vehicular Electronics and Safety (ICVES),* Yokohama, Japan, 2015, S. 69–74.

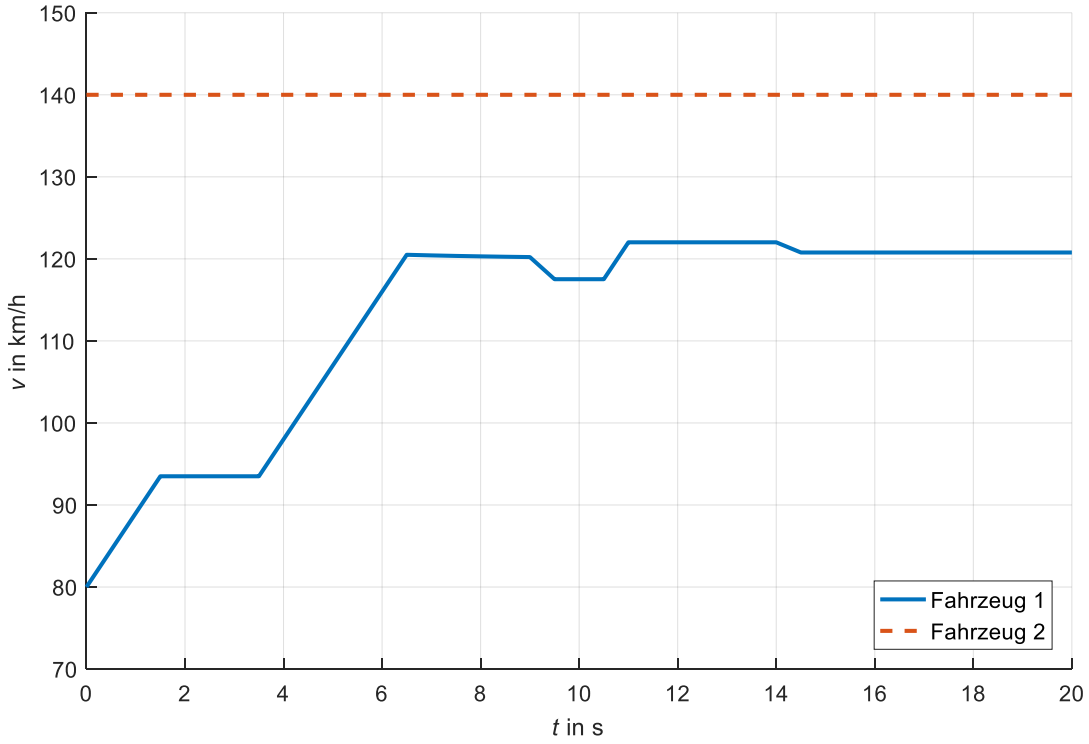
[44] M. Nitzsche, *Graphen für Einsteiger: Rund um das Haus vom Nikolaus*, 3. Auflage. Wiesbaden: Vieweg+Teubner Verlag / GWV Fachverlage GmbH Wiesbaden, 2009.

[45] K. Mehlhorn und P. Sanders, *Algorithms and Data Structures: The Basic Toolbox*. Berlin, Heidelberg: Springer-Verlag Berlin Heidelberg, 2008.

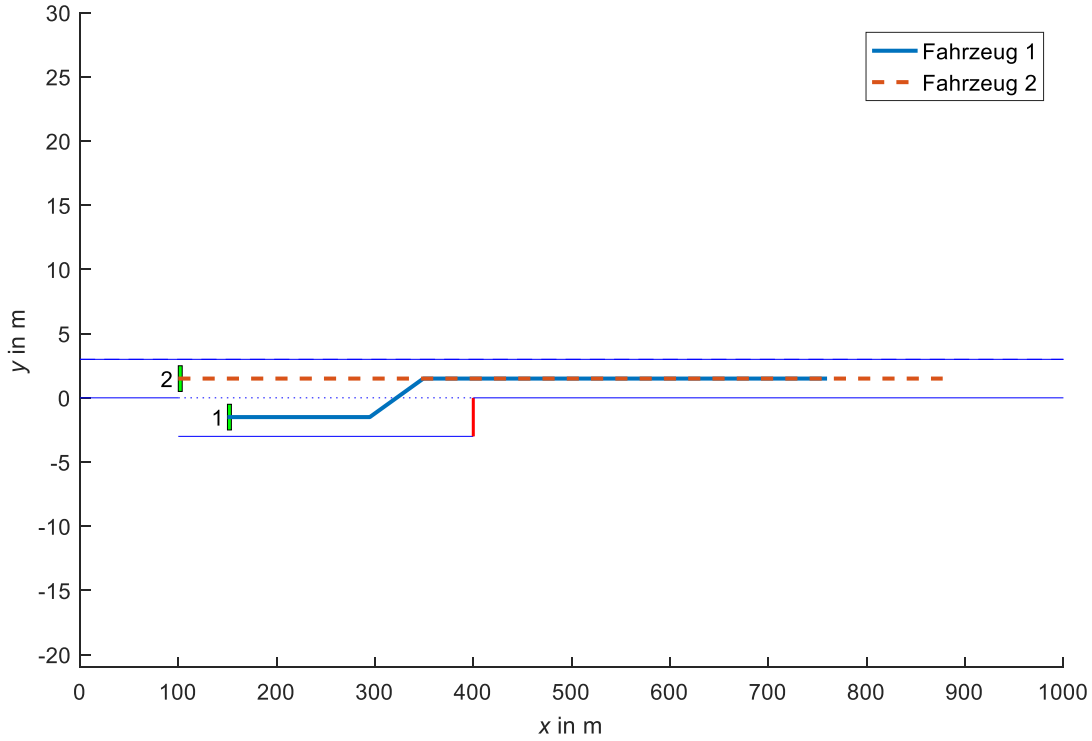
[46] DeepMind Technologies Limited, *AlphaGo - Research.* [Online] Verfügbar: https://deepmind.com/research/alphago/. Gefunden am: 16. Nov. 2017.
[47] C. B. Browne *et al,* “A Survey of Monte Carlo Tree Search Methods,” *IEEE Trans. Comput. Intell. AI Games*, Bd. 4, Rn. 1, S. 1–43.
[48] Wikipedia Foundation, *Monte Carlo tree search.* [Online] Verfügbar: https://en.wikipedia.org/wiki/Monte_Carlo_tree_search. Gefunden am: 17. Nov. 2017.
[49] J. Freyer, *Vernetzung von Fahrerassistenzsystemen zur Verbesserung des Spurwechselverhaltens von ACC*. Dissertation. München, Universität der Bundeswehr, 1. Auflage. Göttingen: Cuvillier, 2008.
[50] M. Treiber, A. Hennecke, und D. Helbing, *Congested Traffic States in Empirical Observations and Microscopic Simulations*. Stuttgart, Germany: University of Stuttgart, 2008.
[51] D. Helbing, H.J. Herrmann, M. Schreckenberg, und D.E. Wolf, Hrsg, *Traffic and Granular Flow ’99: Social, Traffic, and Granular Dynamics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000.
[52] M. Treiber und A. Kesting, *Verkehrsdynamik und -simulation*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010.
[53] The MathWorks Inc, *Objektorientiertes Programmieren in MATLAB.* [Online] Verfügbar: https://de.mathworks.com/discovery/object-oriented-programming.html. Gefunden am: 27. Nov. 2017.
[54] The MathWorks Inc, *Documentation - Handle Object Behavior.* [Online] Verfügbar: https://de.mathworks.com/help/matlab/matlab_oop/comparing-handle-and-value-classes.html. Gefunden am: 21. Nov. 2017.

Anhang

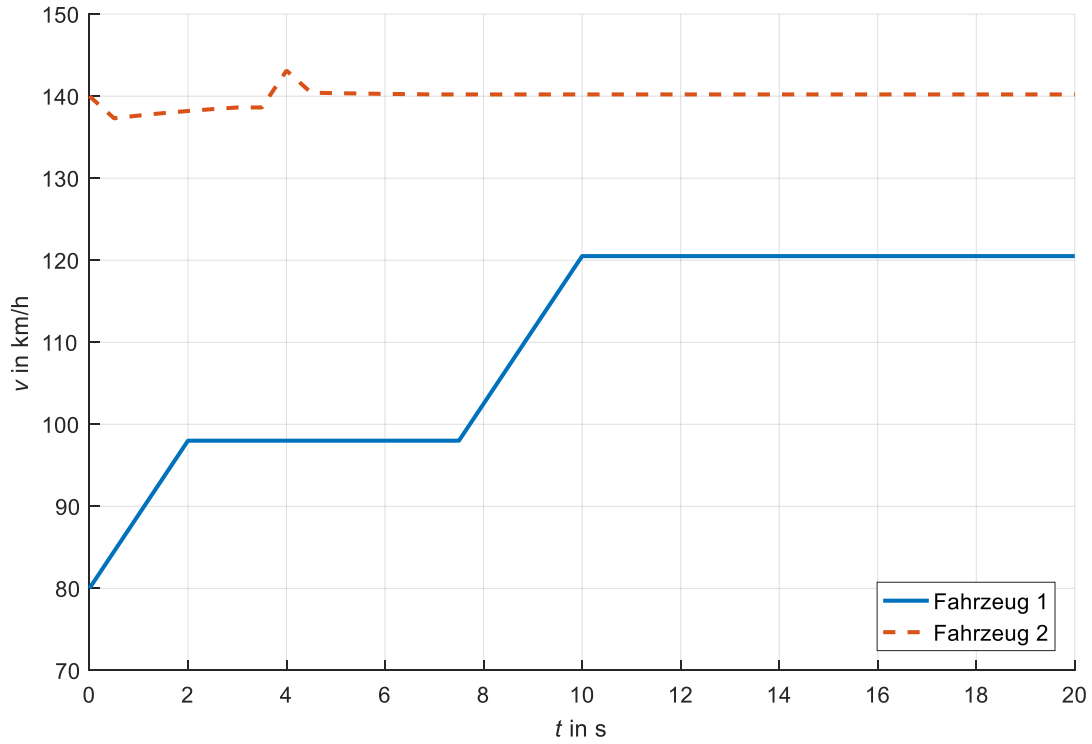
A 4.2



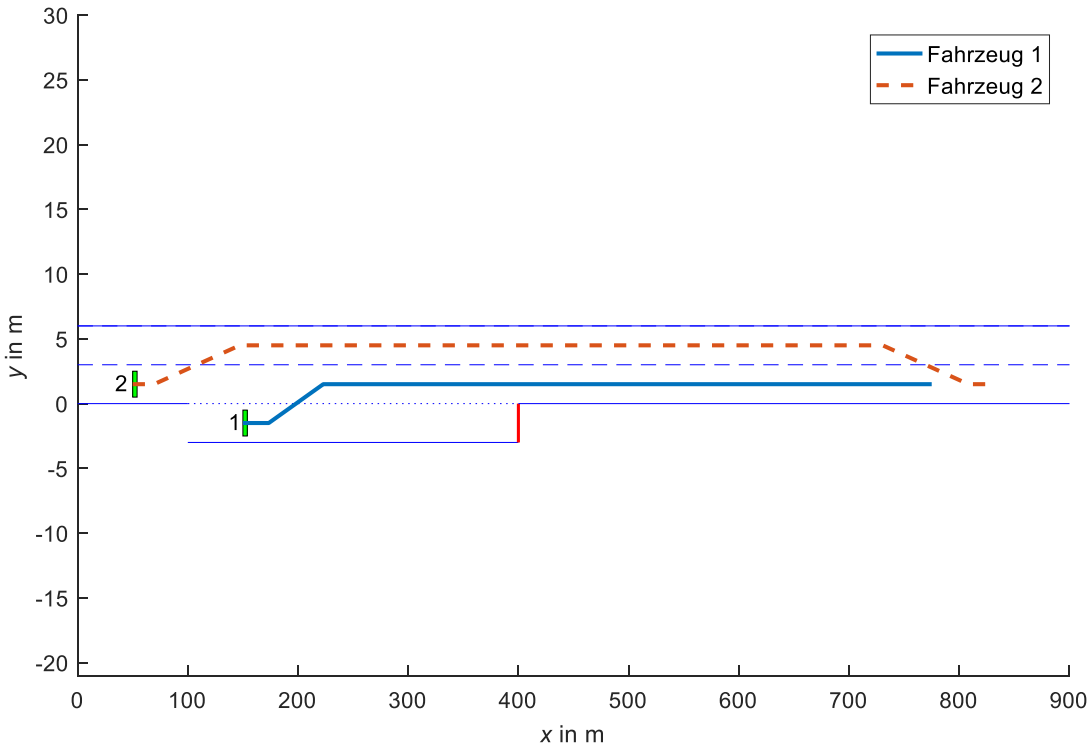
Geschwindigkeitsverlauf der beiden Fahrzeuge in Verkehrsszenario 01 (zwei Fahrspuren, $x_2(t_0) = 100$ m)



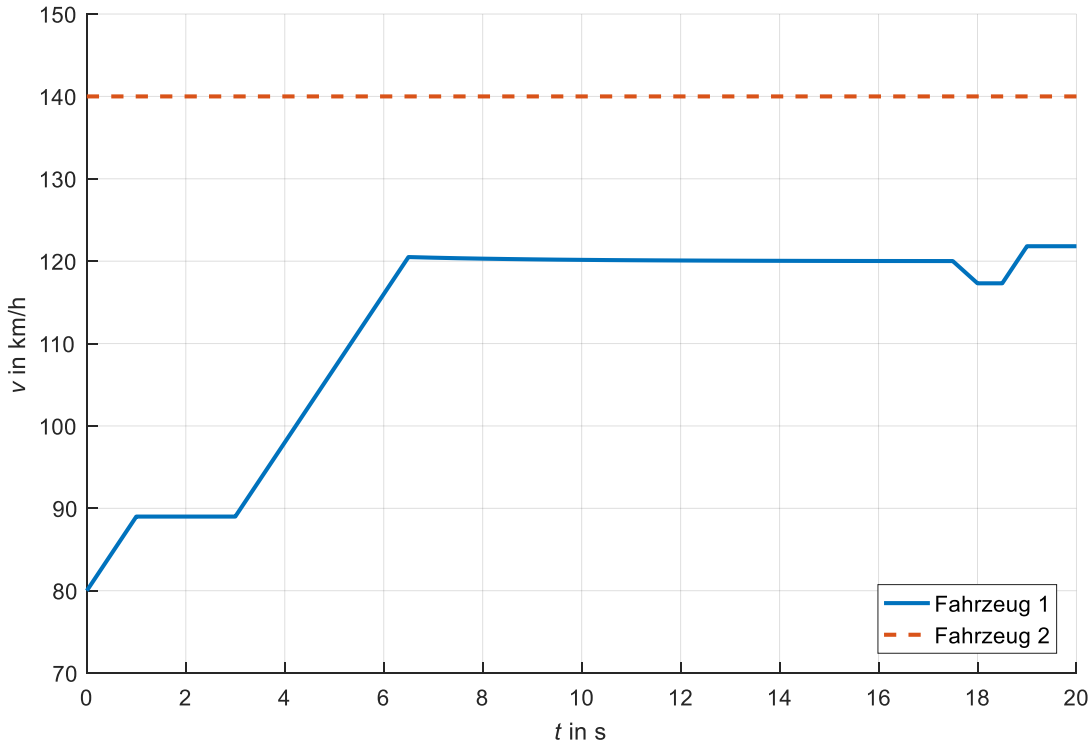
Berechnete Fahrzeug-Trajektorien für Verkehrsszenario 01 (eine Fahrspur, $x_2(t_0) = 100$ m)



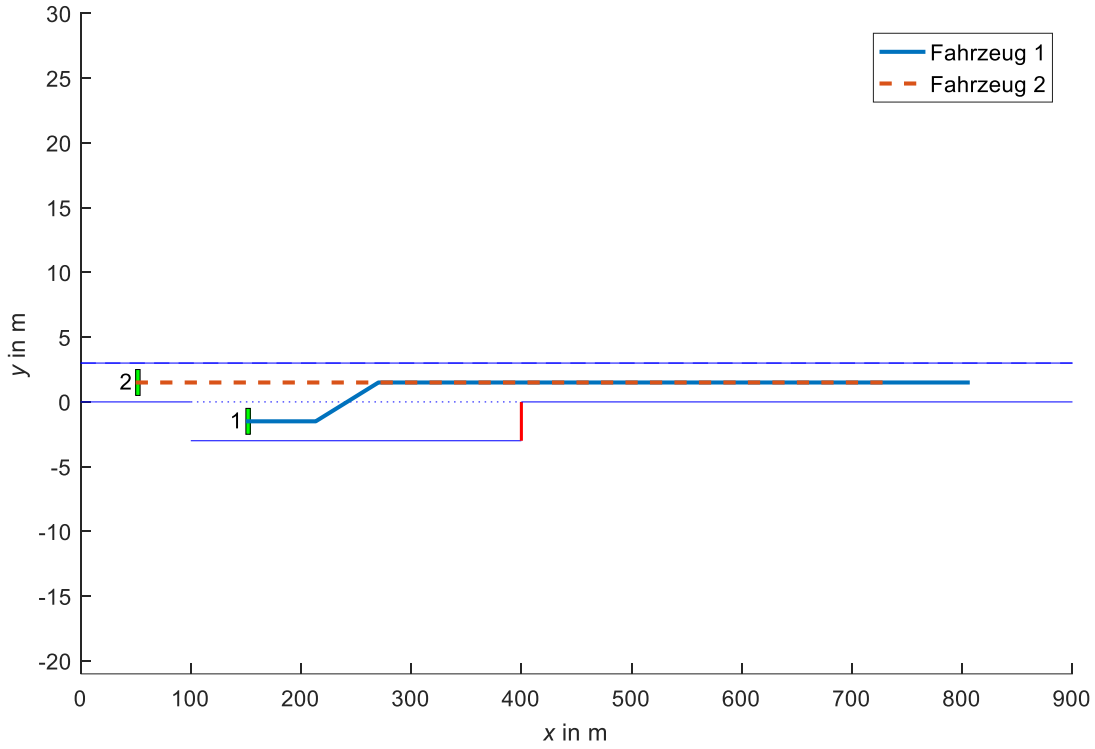
Geschwindigkeitsverlauf der beiden Fahrzeuge in Verkehrsszenario 01 (eine Fahrspur, $x_2(t_0) = 100$ m)



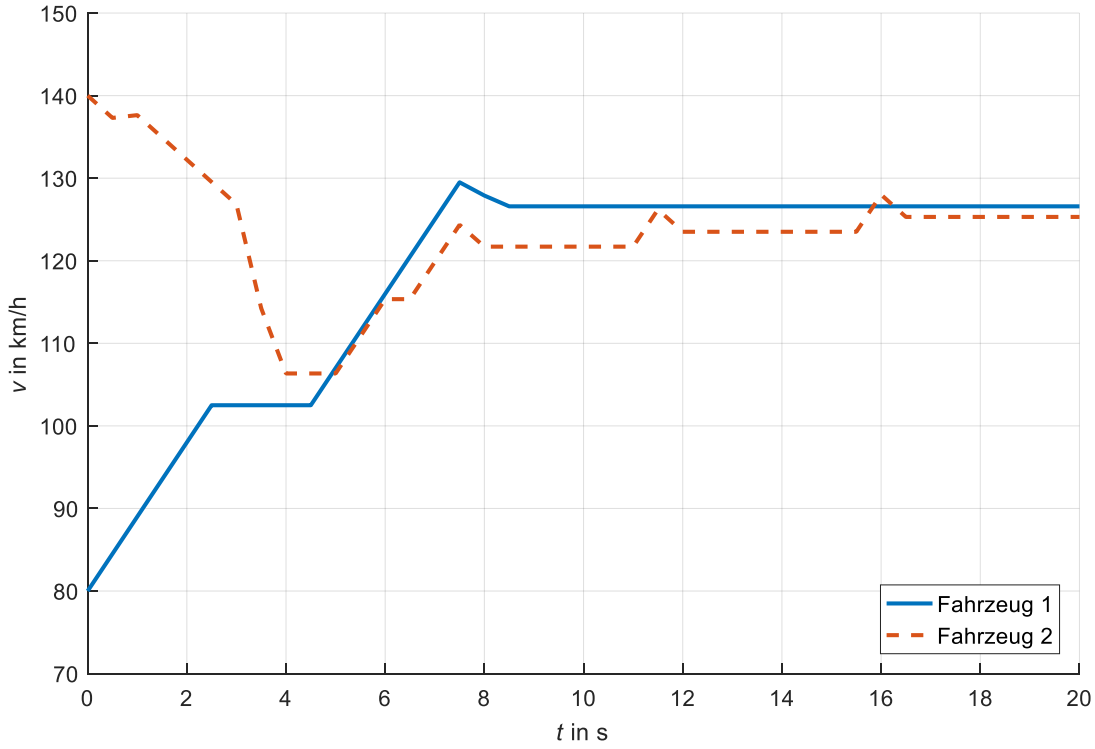
Berechnete Fahrzeug-Trajektorien für Verkehrsszenario 01 (zwei Fahrspuren, $x_2(t_0) = 50$ m)



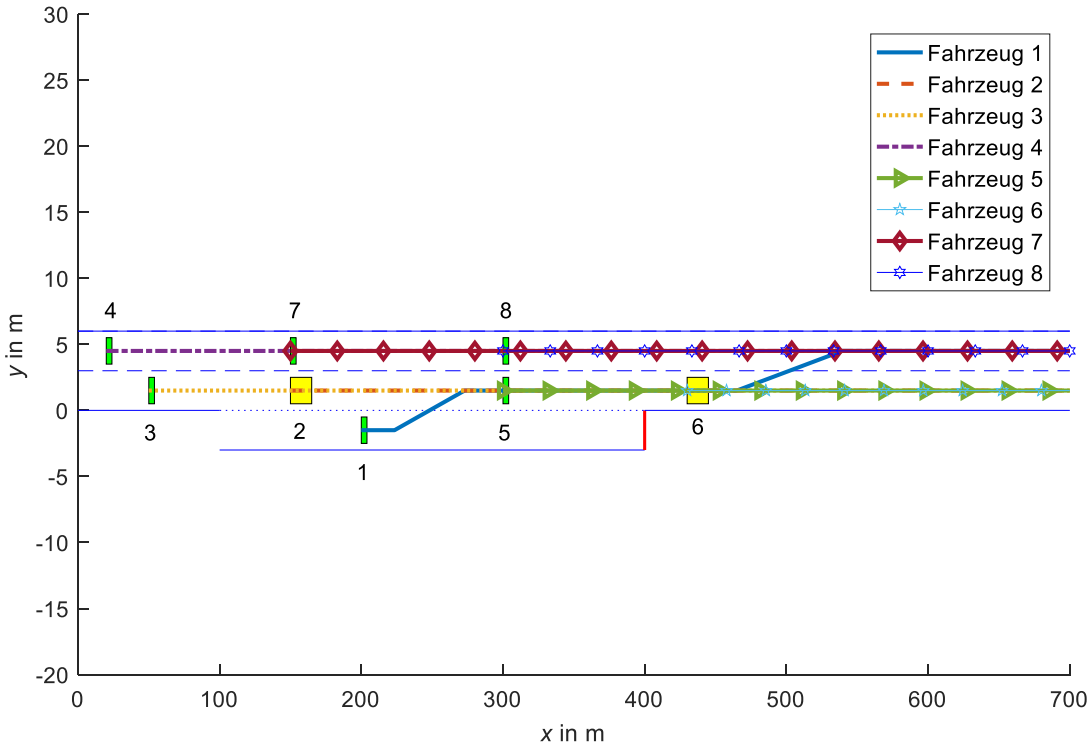
Geschwindigkeitsverlauf der beiden Fahrzeuge in Verkehrsszenario 01 (zwei Fahrspuren, $x_2(t_0) = 50$ m)



Berechnete Fahrzeug-Trajektorien für Verkehrsszenario 01 (eine Fahrspur, $x_2(t_0) = 50$ m)

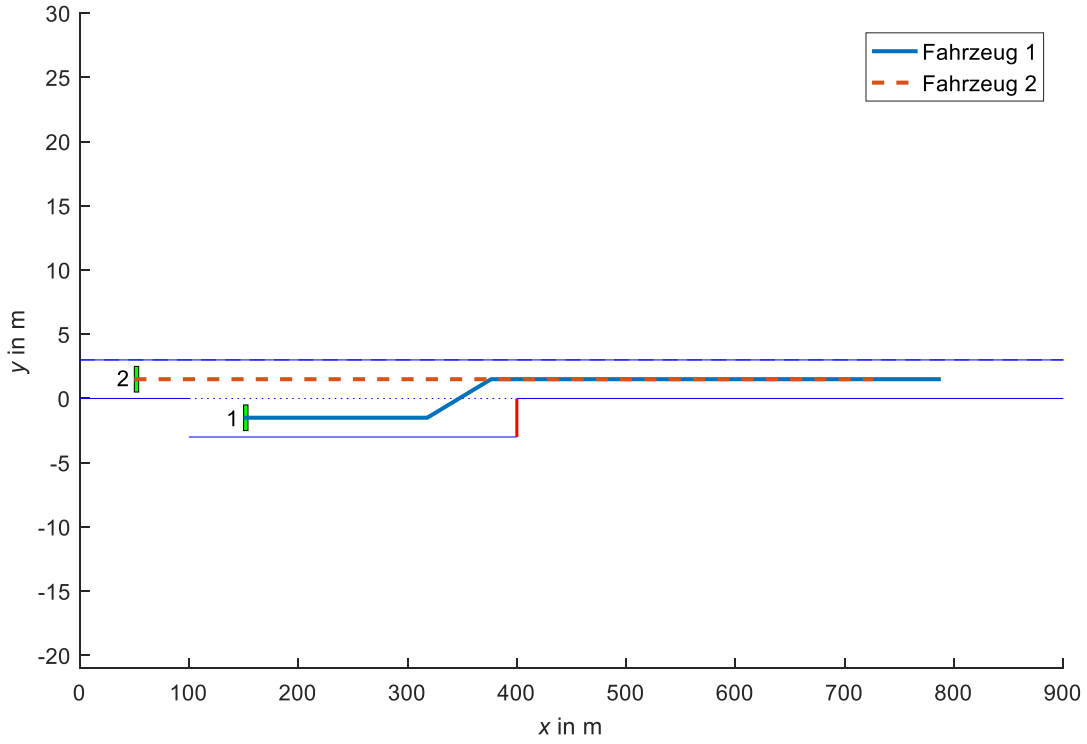


Geschwindigkeitsverlauf der beiden Fahrzeuge in Verkehrsszenario 01 (eine Fahrspur, $x_2(t_0) = 50$ m)

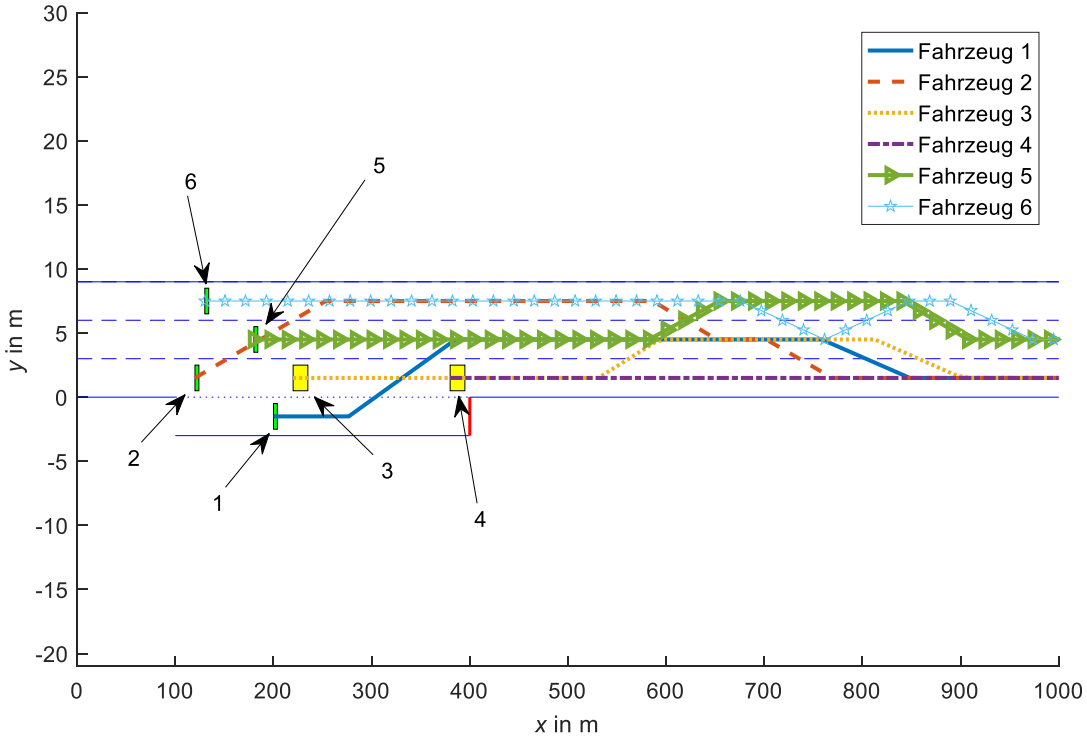


Berechnete Fahrzeug-Trajektorien für Verkehrsszenario 04

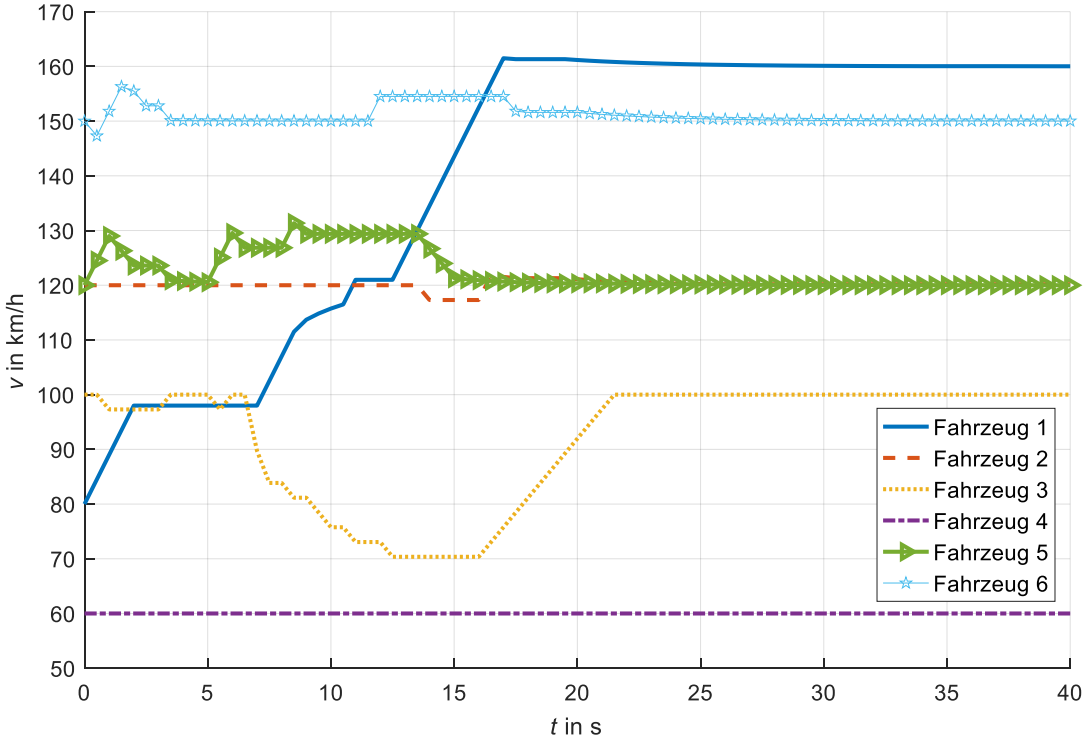
A 4.3



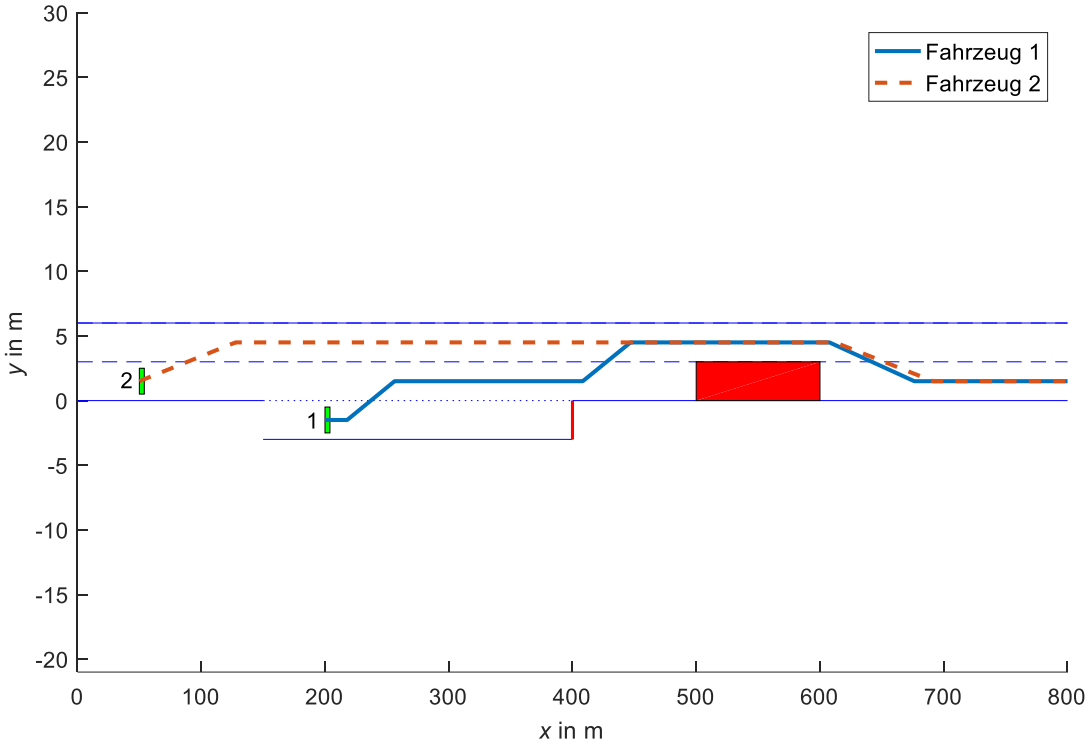
Berechnete Fahrzeug-Trajektorien für Verkehrsszenario 01 (eine Fahrspur, $x_2(t_0) = 50$ m) mit Simulationsschrittweite $\Delta t = 0{,}2$ s



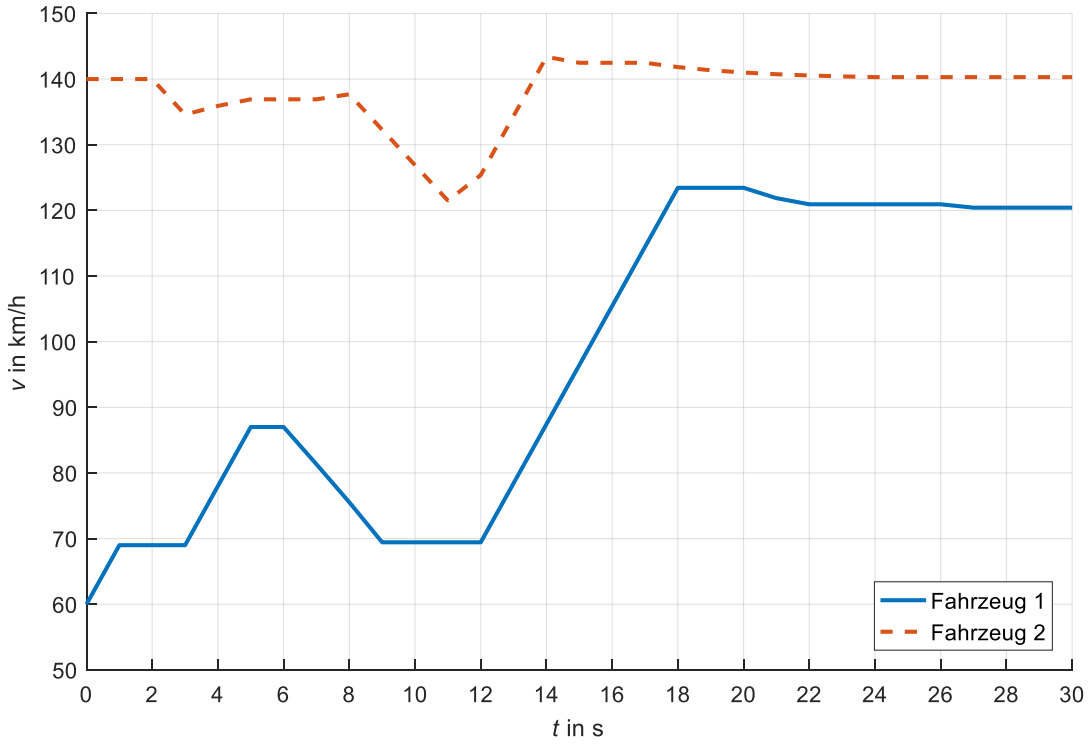
Berechnete Fahrzeug-Trajektorien für Verkehrsszenario 03 mit Simulationsschrittweite $\Delta t = 0{,}5$ s



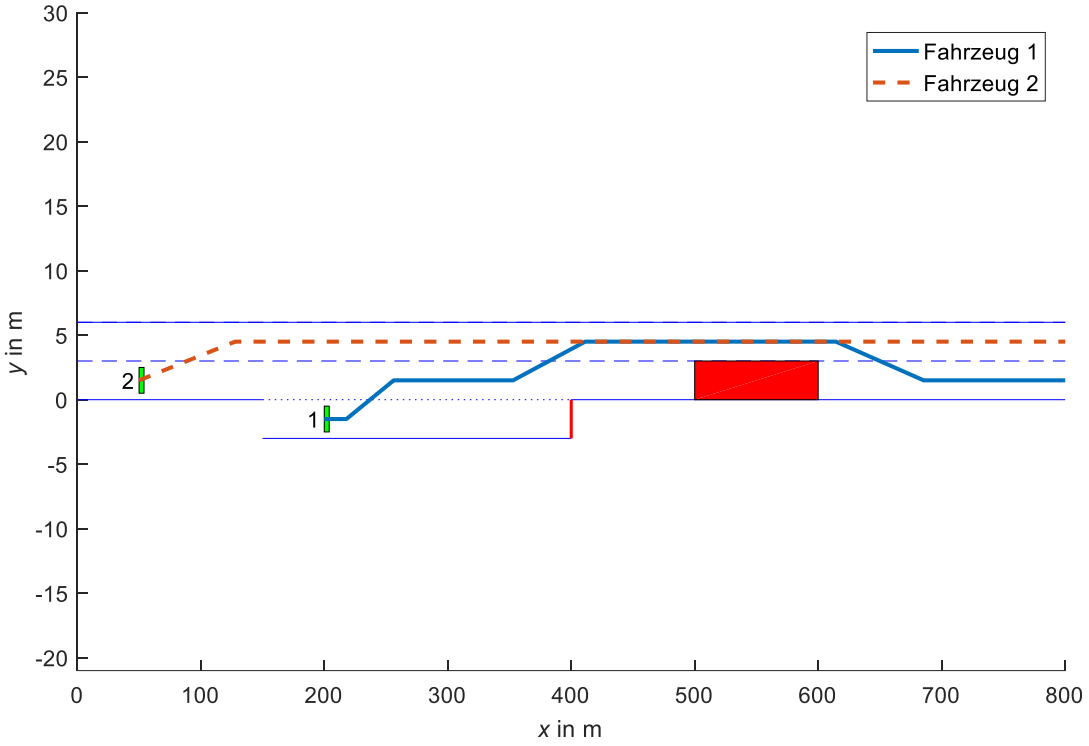
Geschwindigkeitsverlauf der beteiligten Fahrzeuge in Verkehrsszenario 03 mit Simulationsschrittweite $\Delta t = 0{,}5$ s



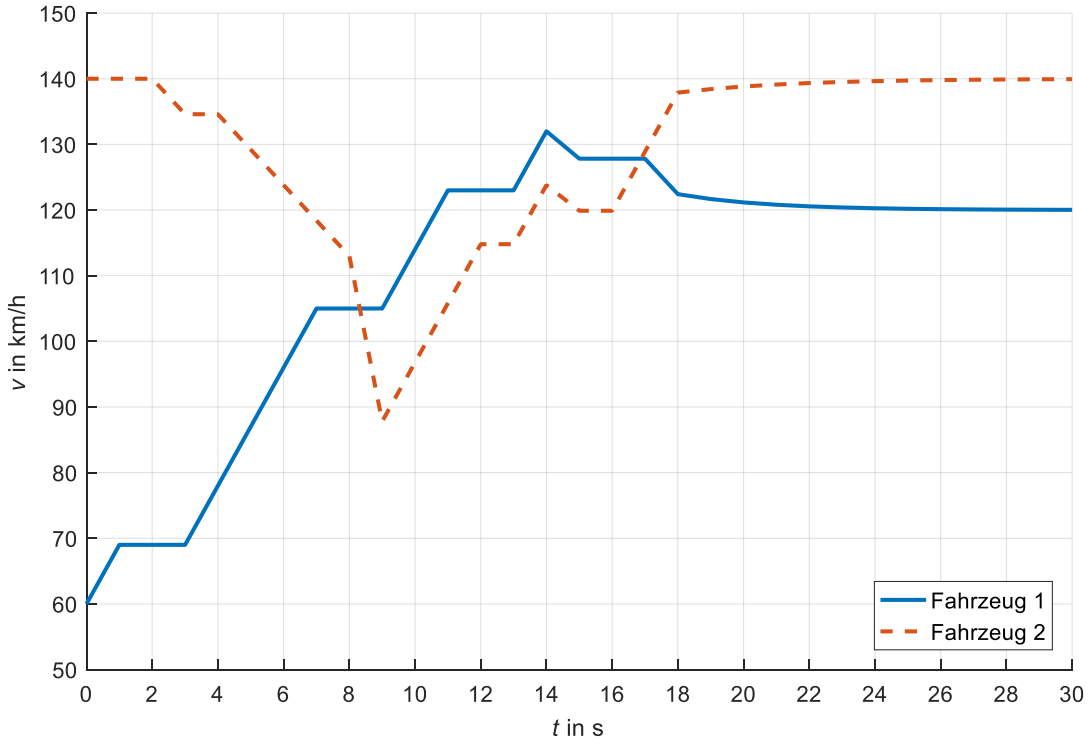
Berechnete Fahrzeug-Trajektorien für Verkehrsszenario 05 (mit 1 000 Iterationen und Simulationsschrittweite $\Delta t = 1{,}0$ s)



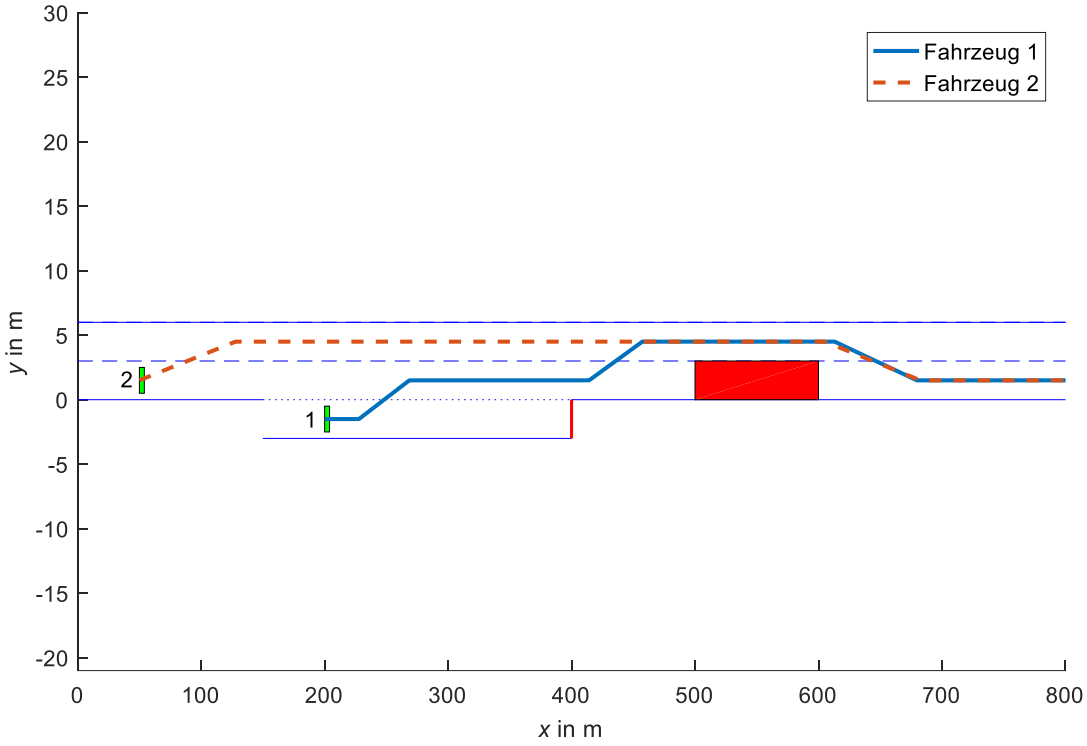
Geschwindigkeitsverlauf der beiden Fahrzeuge in Verkehrsszenario 05 (mit 1 000 Iterationen und Simulationsschrittweite $\Delta t = 1{,}0$ s)



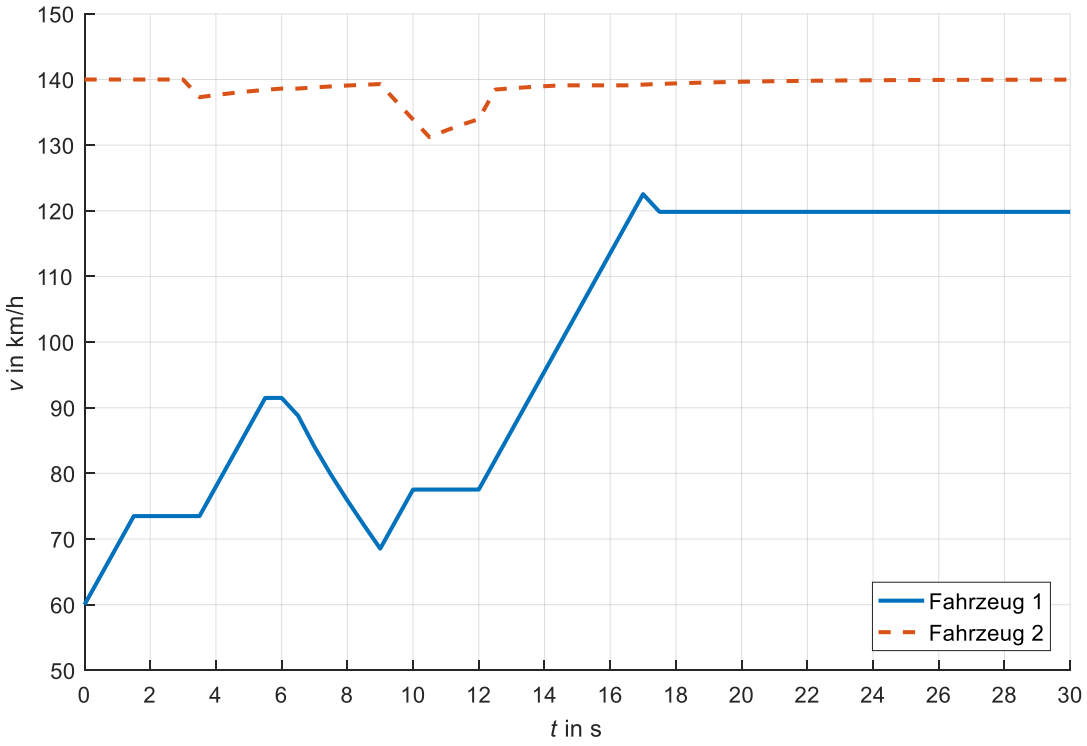
Berechnete Fahrzeug-Trajektorien für Verkehrsszenario 05 (mit 5 000 Iterationen und Simulationsschrittweite $\Delta t = 1{,}0$ s)



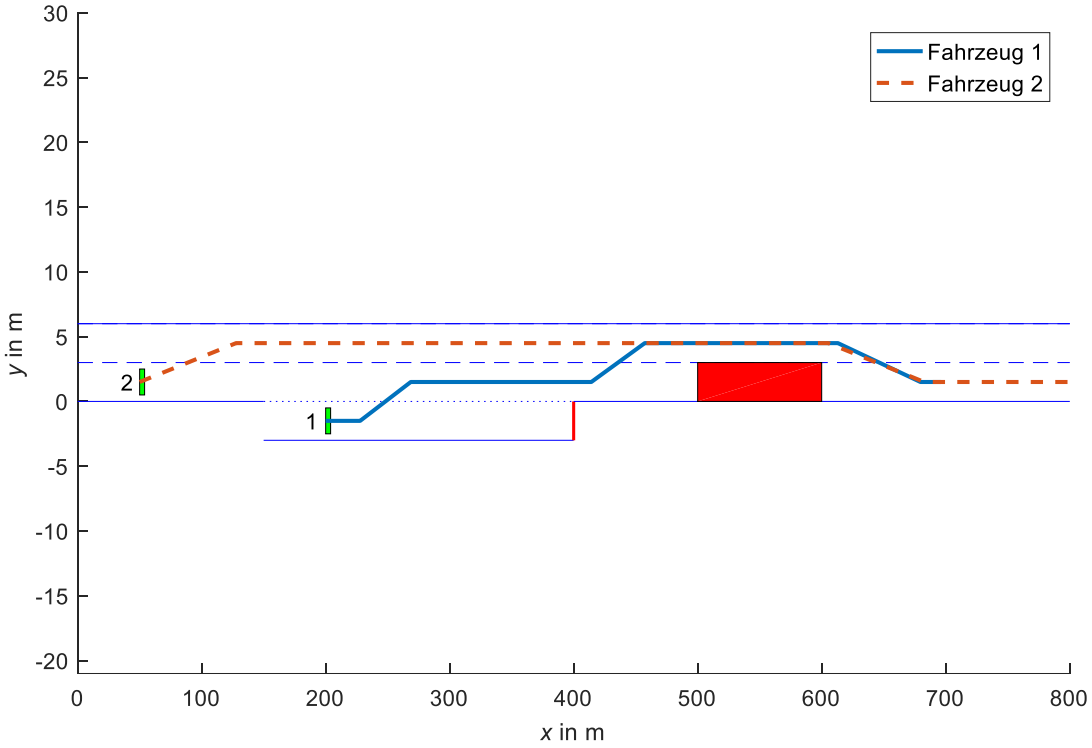
Geschwindigkeitsverlauf der beiden Fahrzeuge in Verkehrsszenario 05 (mit 5 000 Iterationen und Simulationsschrittweite $\Delta t = 1{,}0$ s)



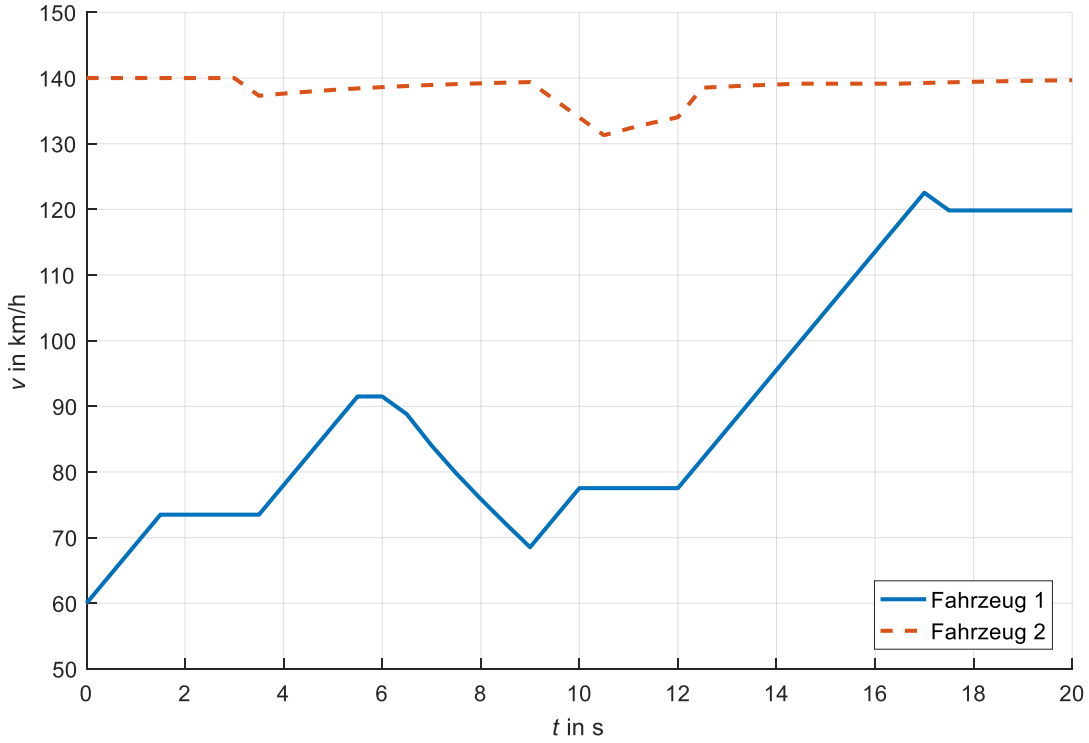
Berechnete Fahrzeug-Trajektorien für Verkehrsszenario 05 (mit 5 000 Iterationen und Simulationsschrittweite $\Delta t = 0{,}5$ s)



Geschwindigkeitsverlauf der beiden Fahrzeuge in Verkehrsszenario 05 (mit 5 000 Iterationen und Simulationsschrittweite $\Delta t = 0{,}5$ s)



Berechnete Fahrzeug-Trajektorien für Verkehrsszenario 05 (mit 10 000 Iterationen und Simulationsschrittweite $\Delta t = 0{,}5$ s)



Geschwindigkeitsverlauf der beiden Fahrzeuge in Verkehrsszenario 05 (mit 10 000 Iterationen und Simulationsschrittweite $\Delta t = 0{,}5$ s)