Predictive Forcefield Trajectory Planning for Automated Vehicles

Dissertation
zur Erlangung des akademischen Grades

Doktoringenieur
(Dr.-Ing.)

vorgelegt der
Fakultät für Maschinenbau der
Technischen Universität Ilmenau

von Herrn
Dipl.-Ing. Tobias Wilhelm Hesse
geboren am 18.03.1979 in Herdecke

urn:nbn:de:gbv:ilm1-2013000684

Abstract

Driving is a complex task, as witnessed by millions of accidents each year which are mostly caused by human error. Increasing traffic density might even aggravate the challenge for the driver. Therefore, automotive industry and research strive to exploit recent advances in sensor technology as well as information processing and communication technologies to develop more and more driver assistance systems. While most of today's assistance systems still rely on purely reactive behavior or very simple short term planning, a more qualified assistance that is able to handle driving in more complex situations and for more distant planning horizons requires a detailed trajectory planning as a key ingredient.

This thesis contributes by devising a novel motion planning algorithm that could be used for autonomously driving vehicles or as a basis for future driver assistance systems. The new proposed motion planning approach is an integrated trajectory planning method that plans both path and velocity profile simultaneously. It centers on a predictive force field trajectory deformation algorithm which is combined with an A*-based trajectory initialization and thus joins the advantages of predictive potential field methods and A*-based graph search trajectory planning.

The most fundamental extension of the method compared to previous predictive potential field methods lies in the integrated planning of a velocity profile by additional degrees of freedom in the time dimension. Further, the dynamic state of the vehicle is now regarded in the planning, and limitations to low curvatures stemming from very rough street approximations have been lifted.

The initial solution is generated by an Anytime Weighted A* (AWA*) variant that has been extended to plan trajectories for road traffic in five dimensions (longitudinal position, lateral position, time, orientation, and velocity). The search space is discretized by an offline generated “state lattice” which allows an efficient coverage and provides a form of resolution-completeness.

Finally, an autonomous test vehicle has been set up to demonstrate the applicability of the devised motion planning approach in experimental tests. A BMW vehicle is equipped with GPS and inertial sensors. A sensor data fusion provides an estimation of the vehicle’s position and dynamic state. Steering, braking, and accelerating are automated by the construction and implementation of appropriate actuation modules. The designed motion control consists of several lower level control loops for the individual actuators and an integrated lateral and longitudinal control approach based on the method of nonlinear decoupling. Online trajectory planning and autonomous trajectory following have been demonstrated successfully in test drives.

Kurzfassung

Autofahren ist eine komplexe und herausfordernde Aufgabe, wie die unzähligen - meist auf menschliches Fehlverhalten zurückzuführenden - Unfälle Jahr für Jahr belegen. Daher arbeiten Automobilindustrie und Forschung fieberhaft daran, die stetig wachsenden technischen Möglichkeiten im Bereich der Sensorik, Datenverarbeitung und Kommunikation für den Fahrer nutzbar machen. Um dem Fahrer in Zukunft aber auch in komplexen Situationen eine qualifizierte Assistenz oder Automatisierung bieten zu können, die den Fahrer vorausschauend unterstützt oder gar selber handelt, ist eine detaillierte Trajektorienplanung unabdingbar.

Der Beitrag dieser Arbeit besteht hauptsächlich in dem neu entwickelten Trajektorienplanungsverfahren, das die Vorteile einer A*-basierten Graphensuche sowie die von vorausschauenden Potentialfeldansätzen in einem Initialisierungs- und einem Optimierungsschritt ausnutzt. Dabei werden Bahn und Geschwindigkeitsprofil simultan geplant.

Die grundlegende Weiterentwicklung der Trajektorienoptimierung liegt in einer Erweiterung, die die simultane Planung bzw. Optimierung von Weg und Geschwindigkeitsprofil ermöglicht. Im Unterschied zu früheren Entwicklungsständen der Methode wird hier der dynamische Fahrzeugzustand berücksichtigt. Außerdem wurden durch die Verwendung lokaler Koordinatensysteme viele vorherige Einschränkungen auf bestimmte Straßenverläufe eliminiert.

Die stichprobenbasierte Trajektorieninitialisierung basiert auf der Weiterentwicklung eines Anytime Weighted A* (AWA*) Verfahrens. Der Suchraum (Längsposition, Querposition, Zeit, Orientierung und Geschwindigkeit) wird hier durch einen offline generierten “State-lattice” diskretisiert, was eine effiziente Suche und eine Art der Auflösungs-Vollständigkeit gewährleistet.

Zusätzlich zu der Entwicklung des beschriebenen Trajektorienplanungsverfahrens wurde ein Versuchsfahrzeug aufgebaut, um die Anwendbarkeit des entwickelten Verfahrens für autonomes Fahren zu demonstrieren. Dazu wurde ein BMW mit der notwendigen Aktorik, Sensorik und Informationsverarbeitung ausgestattet. Die Zustandsbestimmung basiert auf der Datenfusion aus GPS und Inertialsensorik. Durch die Konstruktion geeigneter Adapter wurden Lenkung, Bremssystem und Antriebsstrang automatisiert und sind über einen PC steuerbar. Die Fahrzeugregelung besteht aus einer Reihe unterlagerter Regelkreise sowie einem Regleransatz zur integrierten Längs- und Querführung nach dem Prinzip der nichtlinearen Entkopplung. Die Anwendbarkeit des Planungsverfahrens zur Online-Trajektorienplanung, sowie die Funktionsfähigkeit der implementierten Hard- und Software zur Fahrzeugregelung wurden in Fahrversuchen erfolgreich demonstriert.

Contents

Abstract **iii**

Notation Principles and Abbreviations **xix**

1. Introduction and State of the Art **1**
- 1.1. Motion Planning . . . 6
  - 1.1.1. Search Space Representation . . . 7
  - 1.1.2. Application Domain . . . 9
  - 1.1.3. Quality . . . 9
  - 1.1.4. Reactive Approaches . . . 10
  - 1.1.5. Deliberative Approaches . . . 15
- 1.2. Motion Control . . . 35
  - 1.2.1. Decoupled Motion Control . . . 37
  - 1.2.2. Integrated Motion Control . . . 40
- 1.3. Thesis Contributions and Outline . . . 42

2. Trajectory Optimization **45**
- 2.1. External Force Field . . . 49
  - 2.1.1. Representation of Road . . . 52
  - 2.1.2. Representation of Obstacles . . . 61
- 2.2. Internal Force Field . . . 73
  - 2.2.1. Representation of Lateral Motion . . . 73
  - 2.2.2. Representation of Longitudinal Motion . . . 75
- 2.3. Preview Force Field . . . 77
- 2.4. Equilibrium Search . . . 78
  - 2.4.1. Application of Algorithm to Local Reference Frames . . . 81
  - 2.4.2. Constraints and Stabilizations . . . 83
- 2.5. Summary of Algorithm . . . 92

2.6. Choice of Parameters . . . 94
  2.6.1. General Planning Parameters . . . 94
  2.6.2. Force Field Parameters . . . 95

3. Trajectory Initialization 101

3.1. Definition of the Search Space . . . 105
  3.1.1. Search Space Discretization . . . 106
  3.1.2. Local Planning Method . . . 107
  3.1.3. A Note on Completeness . . . 110
3.2. Definition of the Goal Region . . . 112
3.3. Definition of the Starting Region . . . 112
3.4. Constraint Validation . . . 115
3.5. Cost Functions . . . 118

4. Experimental Setup 121

4.1. Information Processing . . . 121
  4.1.1. Hardware Interfaces in Test Vehicle . . . 122
  4.1.2. Software Structure . . . 123
4.2. Test Vehicle . . . 125
  4.2.1. Sensors for State Estimation . . . 127
  4.2.2. Drive Train Automation . . . 129
  4.2.3. Braking System Automation . . . 131
  4.2.4. Steering Automation . . . 134
4.3. Trajectory Following Control . . . 137
  4.3.1. Integrated Longitudinal and Lateral Motion Control . . . 138
  4.3.2. Subordinate Lateral Control . . . 151
  4.3.3. Subordinate Longitudinal Control . . . 151

5. Results 161

5.1. Trajectory Optimization . . . 161
  5.1.1. Overtaking and Vehicle Following . . . 162
  5.1.2. Overtaking with Oncoming Traffic . . . 168
  5.1.3. Cross Traffic . . . 176
5.2. Trajectory Initialization . . . 179
5.3. Experimental Results . . . 181
  5.3.1. Overtaking Static Obstacle . . . 181
  5.3.2. Overtaking with Oncoming Traffic . . . 189

6. Conclusion and Future Prospects 193

A. Research Vehicles 199

A.1. Boss, Carnegie Mellon University . . . 200
  A.1.1. Road-Based Motion Planner . . . 201
  A.1.2. Unstructured-Environment Motion Planner . . . 201
A.2. Junior, Stanford University . . . 202
  A.2.1. Road-Based Motion Planner . . . 202

A.2.2. Unstructured-Environment Motion Planner 203
A.3. Odin, Virginia Polytechnic Institute and State University 203
A.3.1. Motion Planner 205
A.4. AnnieWay, University of Karlsruhe 205
A.4.1. Road-Based Motion Planner 206
A.4.2. Unstructured-Environment Motion Planner 206
A.5. Conclusion 207

B. Calculations regarding Road Segments **209**
B.1. Clothoids 209
B.2. Relation of Arclengths of Paths with Lateral Offsets 212

Supervised Student Works **215**

My Related Publications **219**

Bibliography **221**

List of Tables

2.1. Parameters for virtual force field trajectory optimization 94

4.1. Main vehicle parameters according to [Presse, 2002] and own measurements . 126
4.2. Comparison of GPS and inertial measurements 127

List of Figures

0.1. Street representation and relevant reference frames xx

1.1. Development of road traffic and accidents 2
1.2. Roadmap for active and passive safety systems 2
1.3. Driving task acc. to Bernotat, Donges and Rasmussen 3
1.4. Combinatorial motion planning with Skeletons 17
1.5. Combinatorial motion planning with Vertical Cell Decomposition 18
1.6. General Structure of Sampling-Based Planning Algorithms 20
1.7. A* motion planning 22
1.8. RRT motion planning 26
1.9. Predictive potential field motion planning 32
1.10. Main components in motion control 36
1.11. General control loop 36
1.12. Input-Output Linearization 40

2.1. Illustration of workspace 46
2.2. Blockdiagram trajectory optimization 47
2.3. Blockdiagram external force field 49
2.4. Street representation and relevant reference frames 50
2.5. Representation of road 52
2.6. Road approximation with cubic polynomials 53
2.7. Road representation 54
2.8. Straight road segment 57
2.9. Circular road segment 58
2.10. Clothoid road segment 59
2.11. Blockdiagram Representation of Obstacles 61
2.12. Extrapolation of obstacles 64
2.13. Out-of-lane obstacle extrapolation 66

2.14. Distance to obstacles . . . 67
2.15. Temporal distance to obstacles . . . 69
2.16. Blockdiagram for calculation of temporal distance to obstacle . . . 69
2.17. External potential field $V^{\text{ext}}(x, y)$ for $t = 0$ . . . 72
2.18. External potential field $V^{\text{ext}}(x, y, t)$ . . . 72
2.19. Lateral acceleration forces . . . 75
2.20. Preview forces . . . 78
2.21. Example of pot. unstable region for $\mathbf{F}_t^{\text{acc}} + \mathbf{F}_t^{\text{dacc}}$ . . . 89
2.22. Schematic analysis of badly conditioned node configurations for $\mathbf{F}_t^{\text{acc}}$ . . . 90
2.23. Real and "virtual" road . . . 96

3.1. Blockdiagram Trajectory Initialization . . . 103
3.2. Construction of Maximum Turning Radius Curve . . . 109
3.3. Velocity profile for lattice generation . . . 110
3.4. Convergence of motion primitives . . . 111
3.5. Closed loop of trajectory planning and following . . . 113
3.6. Selection of Start Vertex . . . 115
3.7. Conservative Collision Testing . . . 117
3.8. Estimation of remaining path length . . . 120

4.1. Block diagram of hardware interfaces to PC . . . 122
4.2. Software structure . . . 123
4.3. Software structure for simulations with simple vehicle model . . . 125
4.4. Test vehicle . . . 126
4.5. Block diagram of state sensing and estimation . . . 128
4.6. Block diagram of principal drive train components . . . 130
4.7. Block diagram of drive train automation hardware . . . 131
4.8. Block diagram of principal braking system components . . . 133
4.9. Lever kinematic for braking system automation hardware . . . 133
4.10. Block diagram of braking system automation . . . 133
4.11. Block diagram of principal steering components . . . 134
4.12. Steering adapter . . . 135
4.13. Block diagram of steering automation hardware . . . 136
4.14. Steering angle sensor . . . 137
4.15. Blockdiagram Trajectory Following Control . . . 138
4.16. Block diagram of nonlinear decoupling controller . . . 140
4.17. Single track vehicle model . . . 143
4.18. Block diagram of implemented integrated motion control . . . 149
4.19. Block diagram of steering control . . . 151
4.20. Simplified model of longitudinal vehicle dynamics . . . 153
4.21. Characteristic maps model of longitudinal vehicle dynamics . . . 154
4.22. Experiment for braking system map . . . 155
4.23. Braking system map . . . 155
4.24. First experiment for drive train map . . . 156
4.25. Second experiment for drive train map . . . 157
4.26. Drive train map . . . 157

4.27. Block diagram of subordinate longitudinal control 158
4.28. Mode selector 159
4.29. Inverse maps for feedforward control 160

5.1. Overtaking static obstacle: Trajectory Optimization Iterations 163
5.2. Overtaking static obstacle: Replanning 164
5.3. Follow lead vehicle: Iterations 166
5.4. Follow lead vehicle: distance, velocity and acceleration 167
5.5. Overtake lead vehicle before oncoming traffic: Trajectory 169
5.6. Overtake lead vehicle before oncoming traffic: Snapshots 169
5.7. Overtake lead vehicle before oncoming traffic: Velocity and acceleration . . . 170
5.8. Overtake lead vehicle after oncoming traffic: Trajectory 173
5.9. Overtake lead vehicle after oncoming traffic: Snapshots 173
5.10. Overtake lead vehicle after oncoming traffic: Velocity and acceleration 174
5.11. Avoid collision with cross traffic: Trajectory 177
5.12. Avoid collision with cross traffic: Snapshots 177
5.13. Avoid collision with cross traffic: Velocity and acceleration profiles 178
5.14. Trajectory Initialization: Basic trajectory set for tree search 179
5.15. Trajectory Initialization: Explored edges and solution of grid-base tree-search 180
5.16. Initial and optimized trajectory for curved road with obstacles 180
5.17. Test drive overtaking LV: Snapshots 182
5.18. Test drive overtaking LV: Deviations from Trajectory 183
5.19. Test drive overtaking LV: a_x, α, s_B 184
5.20. Test drive overtaking LV: δ, ω_{SM} 185
5.21. Test drive overtaking LV: v_x, Δv_x 187
5.22. Test drive overtaking LV: a_x, Δa_x 187
5.23. Test drive overtaking LV: v_x, Δv_x 188
5.24. Test drives: Results in x-t-plane 189
5.25. Test drives: Snapshots Scenario A 190
5.26. Test drives: Snapshots Scenario B 191

B.1. Clothoid segment 210

List of Algorithms

1. Prototypical Sampling-based Planning Algorithm 20
2. A* Planing Algorithm 23

3. Armijo step size rule 91
4. Virtual Forcefield Trajectory Optimization 92
5. Calculation of Forces for Trajectory Optimization 93

6. AWA*-based Trajectory Initialization 104
7. Selection of start states for Trajectory Initialization 116

Notation Principles and Abbreviations

Notation Principles

Here, some general notation principles for this thesis are introduced to simplify reading. The meaning of all variables and symbols is explained wherever they are introduced in this thesis.

Objects and Constraints

$\mathcal{T}$	planned trajectory,
$\mathcal{O}$	obstacles,
$\mathcal{R}$	road,
$\mathcal{V}$	ego vehicle,
$\mathcal{C}$	dynamic constraints

Spaces

$\mathbb{W}_T$	augmented workspace $\langle x, y, t \rangle$,
$\mathbb{C}_T$	augmented configuration space $\langle x, y, \psi, t \rangle$,
$\mathbb{X}_T$	augmented phase space $\langle x, y, \psi, v, a, a_y, \dot{a}, \dot{a}_y, t \rangle$

Sets, Lists

Γ	goal set for trajectory initialization,
Λ	set of motion primitives for trajectory initialization,
OPEN	list of nodes to explore in trajectory initialization

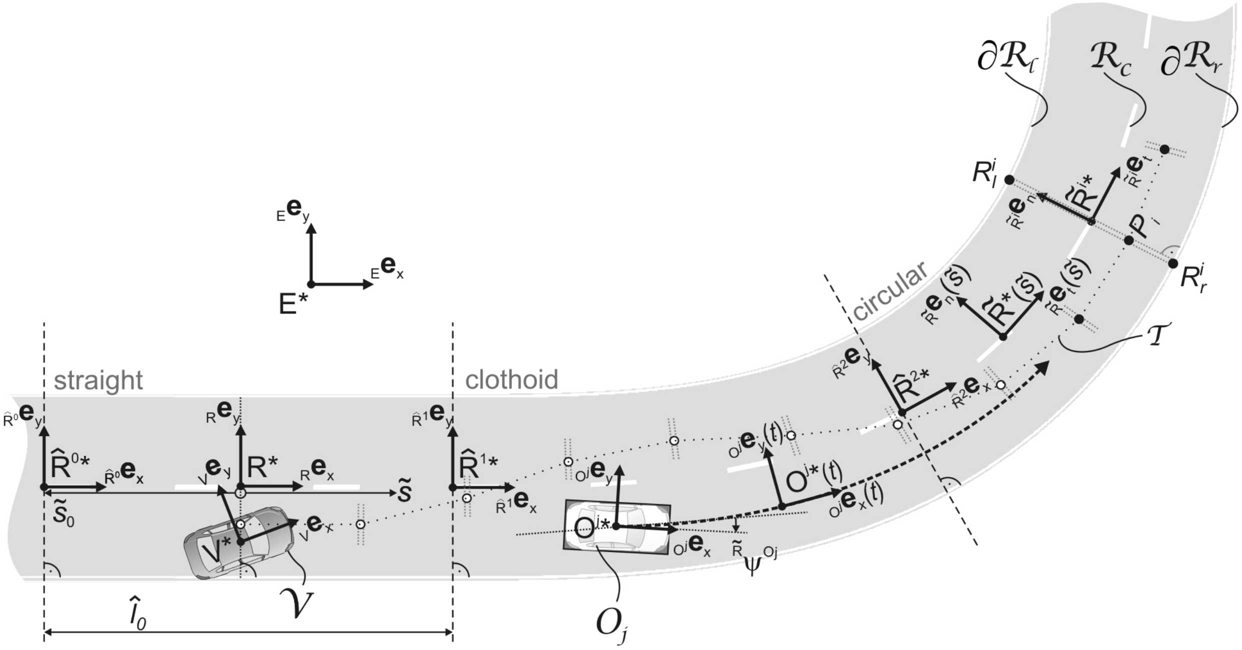


Figure 0.1.: Street representation and relevant reference frames. The reference frame $\underrightarrow{\text{CL}}$ is only used in the appendix and illustrated there.

Reference Frames

$\underrightarrow{\text{E}}$	**E**arth fixed reference frame $\{E^*, {}_E\mathbf{e}_x, {}_E\mathbf{e}_y, {}_E\mathbf{e}_t\}$
$\underrightarrow{\text{R}}$	**R**oad fixed reference frame fixed at planning time t_0 $\{R^*, {}_R\mathbf{e}_x, {}_R\mathbf{e}_y, {}_R\mathbf{e}_t\}$
$\underrightarrow{\tilde{R}(\tilde{s})}$	**R**oad centerline reference frame, shifted with the arclength $\tilde{s}$ $\left\{\tilde{R}^*, {}_{\tilde{R}}\mathbf{e}_x, {}_{\tilde{R}}\mathbf{e}_y, {}_{\tilde{R}}\mathbf{e}_t\right\}$
$\underrightarrow{\tilde{R}^i}$	**R**oad centerline reference frame for point P_i $\left\{\tilde{R}^{i*}, {}_{\tilde{R}^i}\mathbf{e}_x, {}_{\tilde{R}^i}\mathbf{e}_y, {}_{\tilde{R}^i}\mathbf{e}_t\right\}$
$\underrightarrow{\hat{R}^k}$	**R**oad segment fixed reference frame of segment $\hat{R}^k$ $\left\{\hat{R}^{k*}, {}_{\hat{R}^k}\mathbf{e}_x, {}_{\hat{R}^k}\mathbf{e}_y, {}_{\hat{R}^k}\mathbf{e}_t\right\}$
$\underrightarrow{\text{CL}}$	**Cl**othoid fixed auxiliary reference frame $\{CL^*, {}_{CL}\mathbf{e}_x, {}_{CL}\mathbf{e}_y, {}_{CL}\mathbf{e}_t\}$
$\underrightarrow{\text{V}}$	**V**ehicle fixed reference frame of ego-vehicle $\{V^*, {}_V\mathbf{e}_x, {}_V\mathbf{e}_y, {}_V\mathbf{e}_t\}$
$\underrightarrow{O_j}$	**O**bstacle fixed reference frame of obstacle O_j $\left\{O^*_j, {}_{O_j}\mathbf{e}_x, {}_{O_j}\mathbf{e}_y, {}_{O_j}\mathbf{e}_t\right\}$

The reference frames are illustrated in Figure 0.1.

Vector Notation Principles

${}_{V}\mathbf{r}^{A,B}$	vector from A to B, represented in V
${}^{E}_{V}\mathbf{v}^{A}$	velocity of A relative to E, represented in V
${}^{E}_{V}v^{A}_{x}$	x-component in V of velocity of point A relative to E,
${}^{E}_{V}\mathbf{a}^{A}$	acceleration of A relative to E, represented in V
${}^{E}_{V}a^{A}_{x}$	x-component in V of acceleration of A relative to E,
${}^{E}_{V}\boldsymbol{\omega}^{V}$	yawrate of V in E, represented in V
${}_{V}\mathbf{F}^{A}$	Force at A, represented in V
${}_{V}F^{A}_{x}$	x-component in V of force at A

Abbreviations

Abbreviation	Meaning
ABC	**A**ctive **B**ody **C**ontrol
ABS	**A**nti-Lock **B**raking **S**ystem
ACC	**A**daptive **C**ruise **C**ontrol
AD*	**A**nytime-**D**ynamic-A*
ADMA	**A**utomative **D**ynamic **M**otion **A**nalyzer
AFS	**A**ctive **F**ront **S**teering
AHW	**A**utomation **H**ard**W**are
ARA*	**A**nytime **R**epairing **A***
AWA*	**A**nytime **W**eighted **A***
BkRRT	**B**est of **k**-nearest neighbor **RRT**
CA	**C**ollision **A**voidance
CAN	**C**ontroller **A**rea **N**etwork
CEP	**C**ircle of **E**qual **P**robability
CMS	**C**ollision **M**itigation **S**ystem
CVM	**C**urvature **V**elocity **M**ethod
D*	**D**ynamic A*
D/A	**D**igital-to-**A**nalog
DARPA	**D**efense **A**dvanced **R**esearch **P**rojects **A**gency
DGPS	**D**ifferential **G**lobal **P**ositioning **S**ystem
DRRT	**D**ynamic **RRT**
DSC	**D**ynamic **S**tability **C**ontrol System
DUC	**D**ARPA **U**rban **C**hallenge
DWA	**D**ynamic **W**indow **A**pproach
EBD	**E**lectronic **B**rakeforce **D**istribution
EKF	**E**xtended **K**alman **F**ilter
ERRT	**E**xtended **RRT**
ESP	**E**lectronic **S**tability **P**rogram
ETC	**E**lectronic **T**raction **C**ontrol
FSA*	**F**ringe **S**aving **A***
FSM	**F**inite **S**tate **M**achine
GA	**G**enetic **A**lgorithm
GCC	**G**lobal **C**hassis **C**ontrol
GDWA	**G**lobal **D**ynamic **W**indow **A**pproach
GND	**G**lobal **N**earness **D**iagram
GPS	**G**lobal **P**ositioning **S**ystem
GSM	**G**lobal **S**ystem for **M**obile Communications
HC	**H**eading **C**ontrol
HIDS	**H**onda **I**ntelligent **D**river **S**upport System
HMI	**H**uman-**M**achine **I**nteraction
hRRT	**h**euristically-guided **RRT**

Abbreviation	**Meaning**
HW	**H**ard**W**are
IkRRT	**I**terative **k**-nearest neighbor **RRT**
IMU	**I**nertial **M**easurement **U**nit
LADAR	**LA**ser RA**DAR**
LCM	**L**ane **C**urvature **M**ethod
LIDAR	**LI**ght **D**etection **A**nd **R**anging
LDW	**L**ane **D**eparture **W**arning
LKA	**L**ane **K**eeping **A**ssistance
LKS	**L**ane **K**eeping **S**upport
LPA*	**L**ifelong **P**lanning **A***
LPM	**L**ocal **P**lanning **M**ethod
LQR	**L**inear **Q**uadratic **R**egulator
MPRRT	**M**ulti**P**artite **RRT**
ND	**N**earness **D**iagram
NF	**N**avigation **F**unction
NLVO	**N**on**L**inear **V**elocity **O**bstacles
NTG	**N**onlinear **T**rajectory **G**eneration
OBB	**O**riented **B**ounding **B**oxes
PID	**P**ropotional **I**ntegral **D**erivative
PRM	**P**robabilistic **R**oad**M**ap
RADAR	**RA**diation **D**etection **A**nd **R**anging
RAS	**R**ichtlinien für die **A**nlage von **S**traßen
RDT	**R**apidly Exploring **D**ense **T**rees
RNDF	**R**oad **N**etwork **D**efinition **F**ile
RRT	**R**apidly Exploring **R**andom **T**rees
RSC	**R**oll **S**tability **C**ontrol
RTK	**R**eal**T**ime **K**inematics
SbW	**S**teer **b**y **W**ire
SMPL	**S**traightforward **M**odular **P**rototyping **L**ibrary
SOR	**S**uccessive **O**ver **R**elaxation
TCC	**T**orque **C**onverter **C**lutch
UKF	**U**nscented **K**alman **F**ilter
VCD	**V**ertial **C**ell **D**ecomposition
VD	**V**oronoi **D**iagram
VDM	**V**ehicle **D**ynamics **M**anagement
VFH	**V**ector **F**ield **H**istogram
VO	**V**elocity **O**bstacles
VR	**V**oronoi **R**egion

CHAPTER 1

Introduction and State of the Art

Statistics show that the number of motor vehicles in Germany has increased by over 150% from 20.8 to 52.3 million from 1970 to 2010 and the traffic volume has multiplied even more tremendously over the last decades, thus probably increasing the average complexity of traffic situations, [Destatis, 2011*b*]. Improvements in infrastructure, better education of young drivers, enhanced emergency services, and many passive safety systems such as headrests, seatbelts, or airbags have tremendously decreased fatalities in accidents by almost 83% from 1970 to 2010, see Figure 1.1. At the same time the total number of accidents has even increased from less than 1.4 million in 1970 to over 2.4 million in 2010 [Destatis, 2011*a*]. Further, the general cause for accidents largely remains the same: Still about 84% of all traffic accidents are caused by human failure of the driver, [Destatis, 2011*b*], whereas technical failures are mentioned in only 0.9% of the accidents in the statistic.

Therefore, it seems obvious that traffic safety might be improved by actively assisting the driver with his task of driving, thereby helping him to cope with increasingly difficult traffic situations and aiding to actively prevent accidents rather than just mitigating their consequences. These systems are called active safety systems. [Thurner et al., 1998] already expected in 1998 the remaining safety potential of these accident avoiding active safety systems to be much higher than the almost fully exploited potential of passive safety systems, see Figure 1.2.

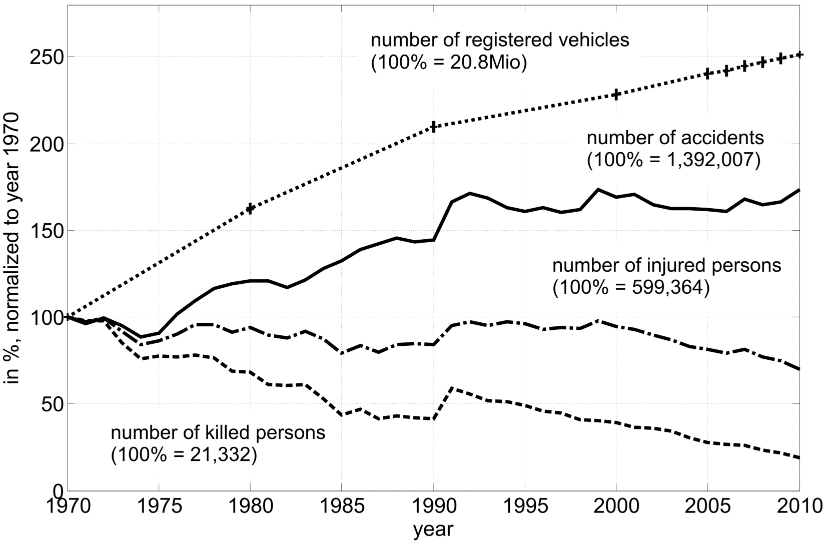


Figure 1.1.: Development of road traffic and accidents, [Destatis, 2011*a*,*b*]
While the traffic volume has multiplied, the number of casualties has decreased due to passive safety systems, regulations and improved infrastructure. The number of injuries, however, has decreased much less, and the number of accidents has even increased showing existing potential for active safety systems. (Until 1990 former territory of Federal Republic.)

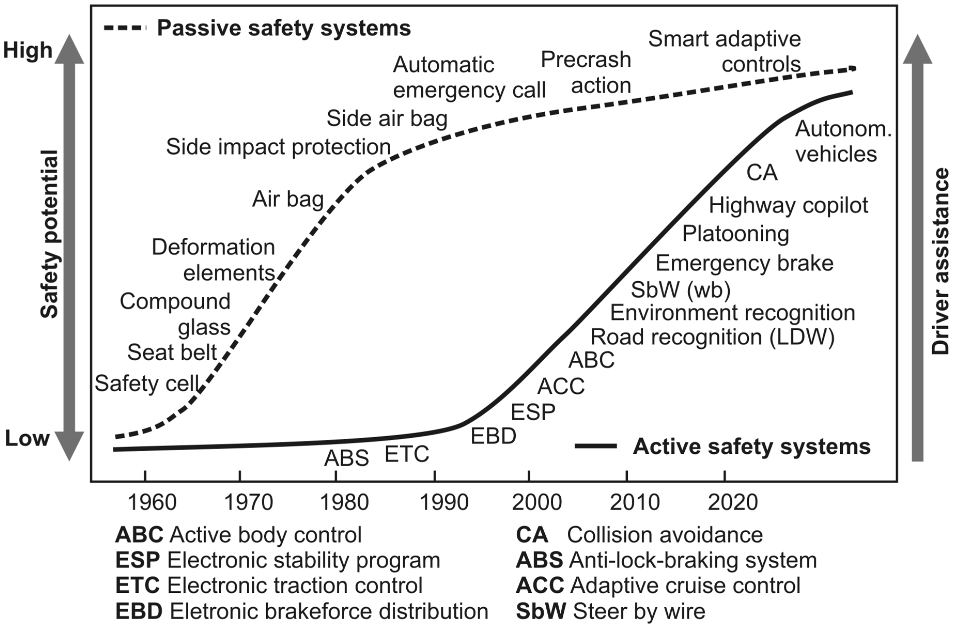


Figure 1.2.: Roadmap for active and passive safety systems, [Thurner et al., 1998]
While the safety potential of passive systems is almost fully exploited, active safety systems still offer a large potential for further improvements in traffic safety.

According to [Bernotat, 1970; Donges, 1982; Rasmussen, 1983], the task of driving can be subdivided into three levels or subtasks: navigation, guidance, and stabilization, see Figure 1.3. For higher automated systems, these subtasks are attributed to different layers in the system architecture, called strategic layer, tactical layer and reactive layer. They differ in their respective time horizons.

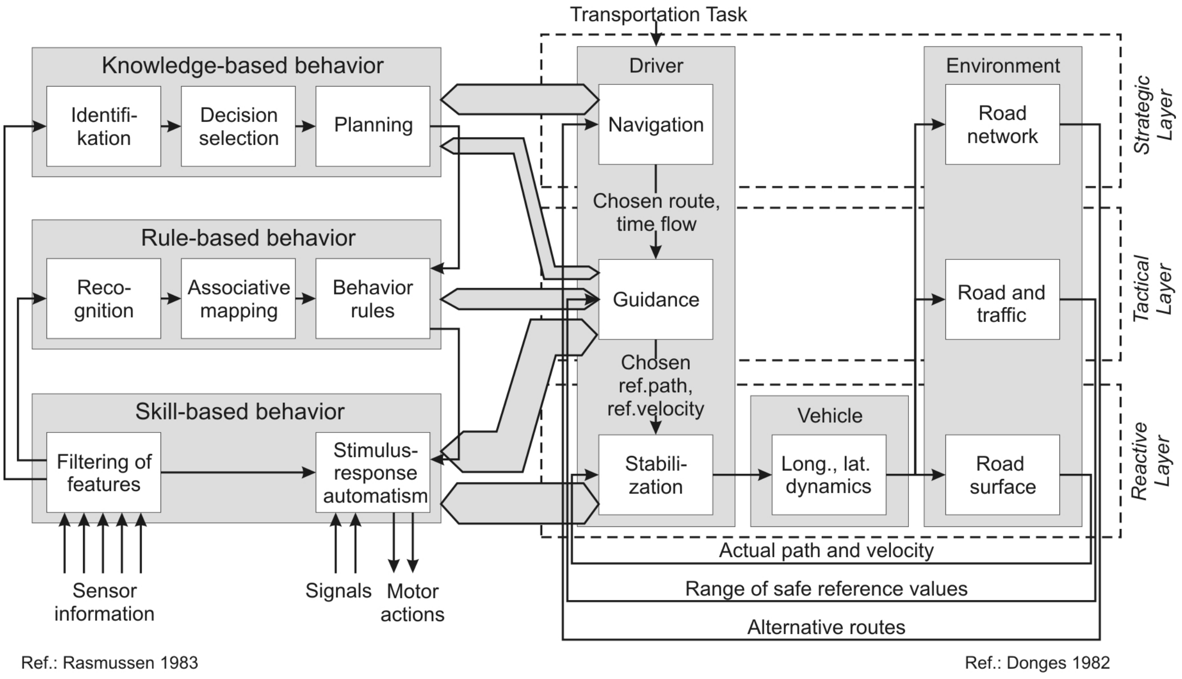


Figure 1.3.: Driving task acc. to Bernotat, Donges and Rasmussen akin to [Donges, 2009]
The driving task is divided into a strategic, a tactical, and a reactive layer. Knowledge-based, behavior-based, and skill-based behavior of a human driver acc. to [Rasmussen, 1983] can be linked to these layers and the different driving tasks acc. to [Donges, 1982] as displayed.

Navigational tasks on the *Strategic Layer* have the longest time horizon. Here, a route is planned from a starting point to a goal. In autonomous systems this plan is often also referred to as a mission.

The *Tactical Layer* is responsible for the guidance of the vehicle and contains the development and execution of short term plans for the upcoming maneuver(s). For example, it is decided whether or not to overtake another vehicle or which lane to choose to prevent a potentially dangerous situation from occurring.

The shortest time horizon is attributed to the stabilization in the *Reactive Layer*. This layer comprises the instantaneous control of the vehicle's dynamic state. It is responsible to compensate disturbances such as wind or a changing road surface and to prevent instabilities during the vehicle guidance such as skidding.

Driver assistance systems have the longest tradition for the strategic and reactive layer while still fewer assistance is available on the tactical layer. Assistance on the strategic level is provided by numerous navigation systems. On the reactive levels, among the most abundant are the Anti-Lock Braking System (ABS) and the Electronic Stability Program (ESP) by Bosch, introduced into the market by DaimlerChrysler in 1979 and 1995, respectively, see [Schuette and Waeltermann, 2005]. Beyond that, many more stabilization programs and functionalities have been devised, such as Roll Stability Control (RSC) or Active Front Steering (AFS). Newer developments like Global Chassis Control (GCC) [Andreasson and Bünte, 2006] or Vehicle Dynamics Management (VDM) [Trächtler, 2004] aim at the integration of several subsystems to achieve desired overall vehicle dynamics. On the guidance level, however, still fewer assistance is available to guide the driver and his vehicle through increasingly complex traffic situations.

With the advent of environmental sensors such as Radiation Detection and Ranging (RADAR), Light Detection and Ranging (LIDAR), and video cameras in passenger vehicles, first assistance systems for vehicle guidance were developed. These include Adaptive Cruise Control (ACC) for longitudinal guidance, see e.g. [Winner et al., 1996], and Lane Departure Warning (LDW) for lateral guidance, as described e.g. in [Suzuki and Jansson, 2003]. A general overview of early systems on the guidance level can further be found in [Vahidi and Eskandarian, 2003].

ACC was introduced by Mitsubishi in 1995 and Toyota in 1996. As first European carmakers, Mercedes and Jaguar followed in 1999 by introducing ACC in the S-Class (ACC is here called Distronic) and the Jaguar XKR, respectively, followed by BMW's 7-series in early 2000. These systems are based on RADAR or LIDAR sensors to measure the distance, velocity, and heading angle of preceding vehicles and adapt the velocity of the ego vehicle to follow leading vehicles at a safe distance. However, ACC is rather comfort oriented and not designed to provide safe longitudinal guidance control by itself since the applied deceleration is limited to a comfortable level, for example 2,5 m/s^2, [Özgüner et al., 2007].

Further developments include the extension of ACC to stop-and-go traffic and the integration of forward collision warning and avoidance, [Piao and McDonald, 2008]. Examples of such a system are the RADAR-based active brake assistance system which is available in the Mercedes S-class since 2005 or Honda's Collision Mitigation System (CMS). As other similar products, the CMS follows an escalation strategy to mitigate collisions. First, a warning signal is activated when the distance to the preceding vehicle becomes too small. Then the system brakes, stepwise increasing the deceleration to a maximum of about 6 m/s^2 as the danger of a collision increases. In addition, the system enhances any existing yet insufficient reactions by the driver in this situation. Even though such collision mitigation systems cannot avoid accidents autonomously, they provide an additional support to the driver and can reduce the collision severity, [Kodaka and Gayko, 2004]. Completely autonomous emergency brake systems that activate in time to prevent collisions have also already been researched, see e.g. [Kopischke, 2000]. Volvo has integrated such a functionality for low speeds in its City Safety System introduced in the XC60 which brakes and stops the car autonomously to prevent forward collisions at velocities of less than 30 km/h, [Avery and Weekes, 2008].

First driver assistance systems for lateral vehicle guidance were Lane Keeping Assistance (LKA) systems. First LKA systems where limited to warnings, therefore dubbed Lane Departure Warning (LDW) systems. LDW systems observe the lane marking of the road with video sensors and warn the driver if he leaves the lane unintendedly. They were first introduced in trucks, for example by Mercedes or MAN. Nissan introduced the first LKA system in a passenger car in the Cima in 2001. The interaction with the human driver varies and includes beeping sounds, blinking lights, vibrations in the steering wheel or seat, or combinations thereof. Newer LKA systems guide the driver closer and give feedback when he deviates from the middle of his lane and not only when he leaves the lane. This kind of support is often referred to as Lane Keeping Support (LKS) or Heading Control (HC). Newer systems include an integration with longitudinal assistance such as the Honda Intelligent Driver Support System (HIDS) which was introduced to the Japanese market in 2002. It combines ACC with LKS. Based on lane detection by a video sensor, an auxiliary supportive momentum is added to the driver's action, [Ishida et al., 2004].

Current developments focus on a further integration of different longitudinal and lateral guidance functions. This includes assistance for maneuvers that require a higher amount of active steering like assisted lane changes, assisted overtaking, or highly automated driving, such as in the EU projects HAVEit, [Hoeger et al., 2008]. In addition to assistance in "normal" traffic situations, automatic or assisted collision avoidance systems have been researched that do not only brake but are also capable of steering or combined steering and braking to prevent impending collisions. [Mildner, 2004] describes a method to compute trajectories for simultaneous braking and evasion. The research project PRORETA uses similarly planned trajectories to devise a system for automatic collision avoidance, see [Bender et al., 2007]. Current research efforts in active assistance and automation in normal and emergency situations include for example the EU project interactIVe, [Etemad, 2010].

One of the challenges for the development of driver assistance systems on the guidance level is posed by the sensory requirements. In order to plan ahead on a highway for five seconds at 100km/h, that is for about 140m, oncoming traffic would have to be detected at a distance of at least 280m. Environment recognition at these distances, however, is still subject to research and not reliably possible at the moment. Needless to note that a whole overtaking maneuver might even take considerably longer than the mentioned time frame.

Nevertheless, as the sensor ranges are increasing and additional environmental information becomes available through Car-to-Car and Car-to-Infrastructure communication, the sensors might no longer be the sole limiting factor for future driver assistance. With more reliable information the number of options for the vehicle increases since complex maneuvers might be performed inside the sensor range. Current assistance systems are all based on either no or a very limited maneuver planning. Lane change assistance for example only needs fixed predetermined trajectories for lane changes to the right and left. Collision avoidance systems are based only on simple maneuvers with certain lateral offsets that must be achieved. Even the current efforts in purely autonomous driving use sophisticated planning algorithms only for parking lot situations and not for road driving, see the review in the Annex A. This, too, can be related to still existing sensory limitations.

In order to provide a more qualified assistance in complex traffic situations and for more distant planning horizons, an underlying detailed trajectory planning is necessary. These planned trajectories can then be either executed autonomously or used as a basis for qualified driver assistance.

It is therefore the goal of this work to contribute by devising a new motion planning algorithm that could be used as a foundation for future autonomous driving or driver assistance systems. The applicability of the developed algorithm is demonstrated in experiments with an autonomously driving test vehicle. The following sections offer an extensive review of existing motion planning approaches (Section 1.1) and provide an overview over available path or trajectory tracking controllers (Section 1.2). Section 1.3 details the contributions and structure of this thesis.

1.1. Motion Planning

A key ingredient to the autonomous motion of a car or a qualified driver assistance is the planning of motion. Besides vehicle guidance, motion planning problems occur in many different disciplines such as in nautics, avionics, or robotics. Even though nautics and avionics provide for a large number of different approaches for motion planning and collision avoidance, these methods are generally not applicable in motion planning for vehicle guidance. In both areas, motions are far less restricted. The dynamics also differ much from that of an automobile. In avionics, even an additional dimension (elevation) is available. Therefore, the field of robotics is usually searched for motion planning approaches to be adapted for use in automotive vehicle guidance.

In their survey of over a hundred different motion planning algorithms in [Hwang and Ahuja, 1992], Hwang and Ahuja classify the different methods according to their *search space representation*, the *application domain*, and their *quality*. These aspects are discussed in the following Sections 1.1.1 - 1.1.3. Sections 1.1.4 and 1.1.5 give an overview over existing planning algorithms, dividing them with regards to their scope into reactive (Section 1.1.4) and deliberative algorithms (Section 1.1.5). The scope can be regarded as one aspect of the attribute quality, as discussed later.

1.1.1. Search Space Representation

The environment can be represented in various different ways, differing by the dimensions that define the *search space* and the *underlying paradigm* that is used to enable or facilitate its exploration.

Even though any combination of different dimensions can be used to define the *search space*, the most common search space representations are:

- Workspace,
- Configuration Space,
- Phase Space, and
- Command Space.

Most basic and intuitive alternative is to make the search space equal to the *workspace* $\mathbb{W}$. The workspace includes the physical dimensions that determine a certain location. Therefore it usually has three dimensions (x,y,z), or can be reduced to a two dimensional space (x,y) in case the robot is restricted to a plane.

Furthermore, the search can be performed in the *configuration space* $\mathbb{C}$, which was first presented in [Lozano-Perez and Wesley, 1979]. It allows to represent all points by just one coordinate: The dimensionality of the configuration-space $\mathbb{C}$ is equal to the degrees of freedom of the robot. The vector $\mathbf{q} \in \mathbb{C}$ specifies the robot's configuration and therefore its location in the configuration space.

In order to allow an easier representation of existing differential constraints, the *phase space* $\mathbb{X}$ can be used. Here temporal derivatives of configurations space dimensions are added as new dimensions until all differential constraints can be represented as first order terms [LaValle, 2006, p. 736].

One major disadvantage of high dimensional search spaces, such as configuration space or phase space, is that the size of the search space increases exponentially with the number of dimensions. Therefore, an extensive search of a highly dimensional space may take exceedingly long. In addition, many of the trajectories that are can be planned in the configuration space may not be realizable by the operational layer. In order to avoid this, Kelly and Stentz elected to represent trajectories implicitly in terms of actuator commands, [Kelly and Stentz, 1998]. This alternative search space is dubbed *command space*. The satisfaction of environment constraints can for example be verified by means of an additional forward simulation in the workspace.

In their survey of motion planning algorithms, Hwang and Ahuja identify four distinct *underlying paradigms* which differ in their environment representation and exploration:

- Skeletons,
- Cell Decompositions,
- Mathematical Programming, and
- Potential Fields.

A *Skeleton* denotes a one-dimensional, homeotopically equivalent representation of the free-space. *Cell Decompositions* consists of dissections of the free-space into simple volumes. In *Mathematical Programming*, constraints are represented through linear inequalities. Last, the physical metaphor of *Potential Fields* can be used to represent the environment in a potential field hazard map.

Clearly, additional representations exist such as in *behavior or rule-based approaches* or *genetic algorithms*, where the free space is represented as genes which are subject to mutation, recombination, and selection processes. This list might be augmented by other forms of representation for different algorithms. In addition, a number of combinations of these paradigms are possible.

1.1.2. Application Domain

The characterization of the application domain of a certain motion planning algorithm includes the specification of allowed robot properties and the classification of the overall problem.

The *allowed robot properties* encompasses all geometric, kinematic and dynamic restrictions that apply to the given robot.

With regards to the *problem class*, the survey [Kavraki and LaValle, 2008] discriminates between several classes ensuing from the level of abstraction and the set of constraints in deterministic motion planning: geometric planning problems, time varying problems, and problems with differential constraints.

In the *geometric path planning problem* a robot has to move without collision from an initial position to a goal position in a world of static and fully known obstacles. The world in which the robot operates, termed the workspace $\mathbb{W}$, is three-dimensional. Often the robot is constrained to a plane and the workspace is then assumed to be two-dimensional. The subset of the workspace occupied by the robot and the obstacles is given by the closed sets $\mathcal{A}$ and $\mathcal{O}$.

As an extension of the geometric planning problem, the *time varying planning problem* allows mobile obstacles. Obstacle regions are predicted for all points of time between the start-time of plan-execution and the arrival-time at any goal or intermediate position. The search space is augmented by the time dimension, so that validity can be checked against the predicted positions of the obstacles for each configuration $(q, t) \in \mathbb{X} = \mathbb{CT}$.

In addition to the global constraints imposed by the obstacles in the geometric planning problem, *problems with differential constraints* introduce local constraints arising when the system's motion is modeled by a differential equation. The search space, then termed phase space [LaValle, 2006, p. 736], contains dimensions for the configuration and for the time derivatives of the configuration: $x = (q, \dot{q})$.

1.1.3. Quality

The quality of an algorithm is determined by its *completeness* and *scope*.

The *completeness* of an algorithm depends on the algorithm's ability to guarantee termination with a valid solution. A complete algorithm finds a plan in finite time for environments, where plans between start and goal configuration exist. Weaker forms of completeness are resolution completeness and probabilistic completeness. A *resolution-complete* algorithm discretizes continuous quantities, such as obstacle-volume or robot configurations, and is able to increase accuracy by increasing resolution, so that completeness is gained in the limit. Randomized algorithms can be *probabilistically complete* if the probability to find a solution approaches one in the limit. Resolution- or probabilistically-complete methods may lack the ability to determine exactly whether a solution exists, but at least the feasibility of motion from start to goal is guaranteed when a solution is found.

The *scope* of a method describes how much environmental information is used as a decision basis in a search algorithm. Algorithms with a local scope usually navigate according to obstacles in the vicinity of the robot and are inherently incomplete. Due to the limited scope these local algorithms are rather reactive in a sense that they do not plan long ahead but can rather only react to the local environment. Global algorithms on the other hand take all available environmental information into account and allow for deliberate planning.

The different motion planning approaches that are presented in the following sections are divided with regards to their scope. First, Section 1.1.4 elaborates algorithms with a local scope, i.e. reactive approaches. Subsequently, deliberative approaches with a more global scope are detailed in Section 1.1.5.

1.1.4. Reactive Approaches

Local or reactive approaches are often also referred to as planning approaches. However, they usually only determine what to do in a certain instant and do not create plans that guarantee to arrive at a certain goal state. Even if a motion from start to goal results it is usually not optimal with regards to any global criteria and at times hard to predict [LaValle, 2006]. Their application can be found where local collision avoidance maneuvers are needed but no global information about the environment can be obtained. Further, reactive approaches can be used in combination with global plans, to alter them quickly when new local information for example about obstacles is available, see e.g. [Arkin, 1989].

There are many different reactive “planning” methods, ranging from workspace methods like *Potential Fields* or *Nearness Diagrams (ND)* to command space methods like *Curvature Velocity Methods (CVM)*, *Vector Field Histogram (VFH)* variants, or *Nonlinear Velocity Obstacles (NLVO)*.

Reactive Potential Fields The reactive potential field method was first introduced by Krogh [Krogh, 1984] and Khatib [Khatib, 1986] for realtime obstacle avoidance of manipulators and mobile robots. Introductory texts on this topic can be found for example in [Choset et al., 2005; Latombe, 1991; Siegwart and Nourbakhsh, 2004]. In this approach, the robot is treated like a point that performs gradient descent on an artificial potential field which is constructed such that obstacles generate repulsive forces and the goal exercises an attractive force. The virtual force on the robot is then translated to actuator controls to generate the desired motion. The modeling of the environment is very intuitive. One of the major drawbacks of this approach is the existence of local minima in which the robot can get stuck, see [Latombe, 1991]. Hence, modifications of this approach often aim at alleviating this drawback.

One attempt to rid the potential field methods of this problem is the introduction of *navigation functions* as potential fields. As defined in [Koditschek and Rimon, 1990; Rimon and Koditschek, 1992], a navigation function essentially is a potential field as a function of distance from the obstacles that is smooth and has only one unique minimum at the goal state.

In a different approach, [Feder and Slotine, 1997] uses *harmonic potentials* introduced by [Connolly and Grupen, 1993] to find closed-form solutions to construct collision-free paths given a known model of a dynamic environment. Expressing the environment in terms of harmonic potentials, however, is not straightforward and only guarantees convergence towards the goal for static environments.

In [Sullivan et al., 2003] Sullivan, Waydo and Campbell use *stream functions* to create potential fields that have saddle points but no local minima. Therefore the robot will never "get stuck" and always reaches its goal. Chang and Marsden add the notion of gyroscopic forces for obstacle avoidance, [Chang and Marsden, 2003]. In doing so the potential field itself does not have to regard the obstacles and thus can be created without local minima. The approach was extended and used for swarming of multi-agent systems in [Chang et al., 2003].

Reactive potential fields have also already been applied to autonomous vehicles. [Gutsche et al., 1993] uses this approach for the motion planning of an automatic indoor transport system. The potential fields are constructed based on a occupancy probability calculated from sensor input. Hennessey and others use a potential field formulation to introduce their concept of virtual bumpers combined with lane keeping, [Hennessey et al., 1995]. [Reichardt, 1996] adapts the reactive potential field approach to be used in a continuous guidance of automobiles in a dynamic environment. Multiple potential fields are used for the different lanes and for collision avoidance.

Nearness Diagram (ND), Global Nearness Diagram (GND) The Nearness Diagram (ND) uses a sectored (polar) environment representation that is used to express distances to obstacles and allows selecting an optimal valley, [Minguez and Montano, 2000, 2004]. As navigation strategy, five laws of motion are used, selected on the basis of an interpretation step. [Minguez et al., 2001] extends this method to a Global Nearness Diagram (GND) approach, adding global reasoning to the Nearness Diagram. It consists of a Mapping ND which integrates information in a model of the environment, and a Mapping-Planning ND which exploits connectivity information of free space using NF1 Navigation Functions. This is an example of using a relatively simple global planner in conjunction with high-performance reactive obstacle avoidance in order to avoid local minima. The main drawback of using NF1 is the type of paths it produces, which are not smooth and graze obstacles.

Curvature Velocity Method (CVM), Lane Curvature Method (LCM) The Curvature Velocity Method (CVM) is a reactive command space search algorithm for unstructured environments. As described in [Simmons, 1996], the search space is comprised of the translational and rotational velocity. It treats obstacle avoidance as a constrained optimization and determines the optimal translational and rotational velocity. The optimum is defined in terms of speed, safety, and goal-directness. The available velocities are constrained by the obstacle configurations and physical limitations of the ego-robot or vehicle.

The Lane Curvature Method (LCM) combines the CVM with the Lane Method which divides the environment into zones of direction called lanes, [Ko and Simmons, 1998]. The Lane Method then chooses the best lane to optimize travel along a desired heading. A local heading is determined for entering and following this lane.

Dynamic Window Approach (DWA), Global DWA (GDWA) The Dynamic Window Approach (DWA) is a reactive “planning” method for obstacle avoidance in unstructured dynamic environments. It is very similar to the CVM in the sense that is uses a constrained search in the velocity space to determine actuator commands, see [Fox et al., 1997]. It also trades off speed, safety, and goal-directedness. However, physical limitations can be included more easily and it uses a grid-based representation that makes it more straight-forward to compute velocity space obstacles, at the cost of the increased memory requirements.

The Global Dynamic Window Approach (GDWA) adds global reasoning to the local DWA similar to the way the GND extends the ND. As described in [Brock and Khatib, 1999], NF1 navigation functions are used to provide a global path towards the goal. This is somewhat inspired by the work of Koditschek and Rimon on reactive potential fields without local minima, as described above. The drawback, that this kind of navigation function produces paths that graze obstacles and thus have a high risk of collisions, is alleviated by the use of the DWA for local obstacle avoidance. Similarly, Ögren and Leonard also propose to use navigation functions and combine them with a DWA in a model predictive control approach to be able to prove the convergence towards the goal, [Ögren and Leonard, 2005].

Vector Field Histogram (VFH) Velocity Field Histogram (VFH) planning uses a two-stage reduction of a local histogram grid to calculate control commands that steer the robot towards a valley in a polar obstacle density histogram. The chosen valley usually is the one closest to the goal direction. When the robot drives around an obstacle, the choice is further influenced by the direction with which the obstacle is circumvented, see [Borenstein and Koren, 1991].

[Ulrich and Borenstein, 1998] extends this approach in several ways, calling it VFH+. The obstacles are enlarged by the robot radius and a security distance. Further, a hysteresis is applied on the polar histogram to reduce oscillations between valleys. In addition, valleys that require control inputs exceeding actuator limits are blocked. Finally, goal-directedness, path smoothness, and continuity of motor commands are traded off.

In [Ulrich and Borenstein, 2000] Ulrich and Borenstein add global reasoning to the method, combining VFH with an A* graph search to VFH*. (For further explanation of A*-based searches see also Section 1.1.5.2.) For each candidate valley, a projected position is calculated if this valley was chosen. For each position the predicted VFH+ is evaluated and the next candidate valleys and projected positions are generated. By repeating this process a tree of projected positions is created, until the goal point is reached. Then the initial valley that leads to the best path towards the goal can be chosen.

Velocity Obstacles (VO), Nonlinear Velocity Obstacles (NLVO) [Fiorini and Shiller, 1998] describes reactive motion planning using Velocity Obstacles (VO). All obstacles are assumed to have a constant velocity. They are transformed into the velocity space that consists of the two translational velocities (v_x, v_y). In this description it can be seen which velocities of the ego-robot would lead to collisions with obstacles and these can be excluded when the next velocities commands are chosen. Further, physical limitations can be regarded by introducing a set of admissible velocities which is defined by the robot's acceleration constraints.

In [Shiller et al., 2001] and [Large et al., 2002], a set of approximations is introduced that allows to extend the method of Velocity Obstacles of circular objects with constant velocities to non-circular objects on non-linear trajectories, the Nonlinear Velocity Obstacles (NLVO). Adding a notion of risk (imminent collisions are more dangerous than distant ones), this allows obstacle avoidance using a cost function on the robot's velocity.

This local concept can also be used for global or deliberative planning by building a search tree. This tree is acquired by sampling at certain time intervals and adding several possible velocities at each step, see [Fiorini and Shiller, 1998; Large et al., 2005].

As described, reactive approaches enable a fast reaction of a vehicle or robot to its immediate environment and determine actions to prevent imminent collisions. However, purely reactive approaches do not really plan ahead which results in several shortcomings. First and most prominently, reactive approaches might get stuck in dead ends or local minima since they only take instantaneous decisions which might be suboptimal in the long run. This also results in the second problem, that the resulting path is usually not optimal with regards to any cost function, like distance to obstacles, necessary accelerations, traveled distance, or time. Third, the resulting path towards the goal is unknown beforehand which is definitely a drawback if the planned path shall be communicated to a human driver in order to assist him. Forward simulations might be used to calculate this path if the environment is known completely, but this is rather time consuming and only offers the opportunity to check the results but no direct handle on how to modify them.

The first shortcoming has been addressed in many extensions, as mentioned. In some approaches like the use of navigation functions or harmonic potential fields for reactive potential fields, it can be guaranteed mathematically that the reactive approach always converges towards the goal state. However, this does not alleviate the second and third problem.

The same holds true when navigation functions are added to other reactive approaches to provide a global plan that is then used as a rough guideline in combination with a local reactive collision avoidance algorithm, as for example in GND or GDWA. This approach splits the task of planning into a global and a local planner. The used NF1 navigation functions describe the shortest path towards the goal, therefore, they tend to “graze” obstacles. Therefore, the reactive algorithm must deviate from the planned path to compensate these shortcomings. Unfortunately, this again results in the mentioned problems that the resulting path is neither known completely beforehand nor optimal.

Another approach mentioned above to add global reasoning to local reactive methods is to build a search tree from the current position towards the goal and to apply an adequate graph search algorithm to find the best path within the search tree, as for example in VFH* or NLVO. The reactive method is then mainly used to create the vertices from one node of the tree to the next. This approach truly integrates the reactive part into a global planner and at least mitigates all three problems that were states above. However, at this point some drawbacks of the applied tree search methods affect the result. These issues are addressed in more detail in Section 1.1.5.

In general, the performance of the algorithm and the nature of the overall planned path depend on the discretization in time and space that is chosen to build the tree. Here this applies to the length of the time intervals between one node and the next and also to the number of vertices the reactive algorithm produces at each node to generate subsequent ones. The quality of the planned path increases with a high discretization along with the necessary computational time. This problem increases for higher dimensionalities of the search space. Further, depending on the applied reactive algorithm that produces the vertices from on node to the next and the connections at the nodes, the final global path might not be smooth and thus hard to follow. In addition, some approaches as the VFH* only compute paths and do not plan a velocity profile. The generalization to a trajectory planning, i.e. an integrated path and velocity planning which is desired in this work, is not straight forward and sometimes maybe not possible.

1.1.5. Deliberative Approaches

In contrast to reactive approaches, deliberate approaches provide a "real" motion *planning* since they develop a plan through the search space from a start to a goal point or region. The deliberative motion planning approaches can be subdivided as mentioned above according to their completeness and underlying paradigm.

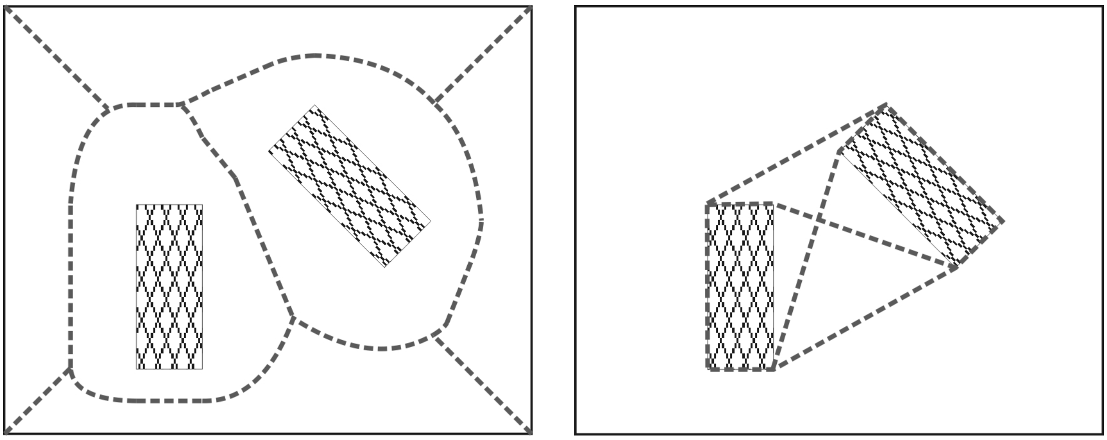
First, there exist a number of *combinatorial* approaches which are complete and build an explicit model of the free space as skeleton (often called roadmap) or by cell decomposition. Second, there are *sampling based* algorithms with the weaker forms of completeness termed resolution completeness and probabilistic completeness, as discussed above. These algorithms do not build an explicit model of the environment but check it on an as-needed basis to save time. Third, a number of incomplete *heuristic* algorithms have been devised based on geometric relations or a set of behavior rules. These algorithms work most of the time but cannot offer any guarantee that a solution is found if it exists. Finally, there are approaches in mathematical programming that consist of numerical single- or multi-objective *optimization* algorithms. In contrast to the other types of methods, they always require an initial guess as starting solution that is then deformed.

1.1.5.1. Combinatorial Motion Planning

Essentially, combinatorial algorithms create a complete representation of the free space mirroring its topology, i.e. maintaining the features accessibility, connectivity, and departability, [Canny, 1988]. In doing so, either the free space $\mathbb{X}_{free}$ is reduced to a skeleton, also called roadmap, or decomposed into cells, see Figures 1.4 and 1.5. In many theoretical papers the motion planning problem is considered to be solved once this representation can be successfully constructed. In order to generate a certain path or trajectory, both types or representation could then be used to create a graph that contains all topologically different paths through the environment. The graph is now searched for the best path in case several solutions exist. A cell representation can be transformed into a roadmap for example by connecting all centroids of the cells with the centroids of all neighboring cells by collision-free trajectories through their shared boundaries.

Skeletons Algorithms of the skeleton class can generate graphs through roadmaps by identifying arcs and points, which are prominent in the set $\mathcal{C}_{free}$ and capture the important topological and geometric properties of a robot's environment, [Choset, 1997]. Two different kinds of roadmaps are illustrated in Figure 1.4 and explained in the following paragraphs.

The *Visibility-Graph Algorithm* or *Shortest Path Roadmap* [LaValle, 2006, Section 6.2.4] illustrated in Figure 1.4b is one of the most common skeleton algorithms. It uses the observation that in a planar environment the shortest connection between two points must pass along obstacle corners, if the two points cannot be connected directly. A path in a two-dimensional environment can be constructed in $O(n^2 \lg(n))$, with n vertices. The algorithm cannot be directly extended to three or even more dimensions, as here shortest path edges may be deflected anywhere on a polyhedron's edge. Finding an exact shortest path in three dimensions is NP-hard, [Jiang et al., 1993], although approximation algorithms exist.



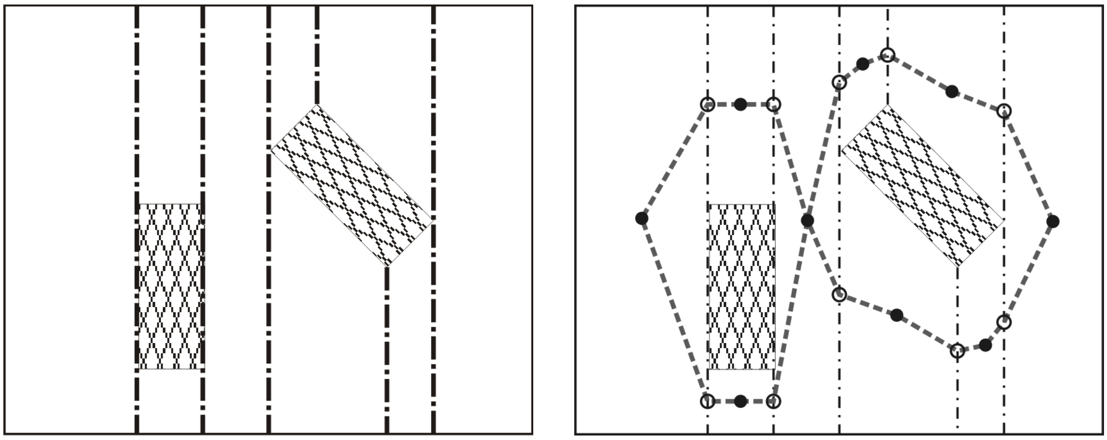
(a) Maximum Clearance Roadmap (b) Shortest Path Roadmap

Figure 1.4.: Combinatorial motion planning with Skeletons.
a) Maximum Clearance Roadmap (Voronoi Diagram),
b) Shortest Path Roadmap (Visibility Graph)

Instead of allowing near contact to obstacles, to minimizes path length, *Maximum Clearance Roadmaps* [LaValle, 2006, Section 6.2.3] illustrated in Figure 1.4a locally maximize the distance to the border of $\mathbb{C}_{free}$ for any point of the roadmap. Safe paths are in that way preferred over short paths. Maximum Clearance Roadmaps are based on Voronoi Diagrams (VD) [Aurenhammer and Klein, 1996]: For a set of obstacle-points S on a plane, the Voronoi region (VR) of a point $p \in S$ is defined to be the set of coordinates, which have a smaller distance to p than to any other point $q \in S$. The *Voronoi Diagram* of S is defined as the conjunction of the coordinates, which belong to the border of the Voronoi regions for all $p \in S$. Generalized Voronoi Diagrams can be constructed not only for points, but also for composite objects, such as polyhedra consisting of vertices and line-segments. Voronoi Diagrams retain the topology of the free-space, but reduce the dimensionality only be one. The VD of a two-dimensional space is a one-dimensional roadmap, whereas the VD of a three-dimensional space consists of two dimensional planes. According to [Choset, 1997] Voronoi Diagrams can be recursively defined on Voronoi Diagrams, until a one-dimensional structure is obtained, the so-called Generalized Voronoi Graph. These structures though are not guaranteed to be connected for higher dimensional spaces. A construction of connecting edges produces roadmaps which contain very unintuitive paths that are not applicable to higher velocity motion planning.

The algorithm proposed in [Canny, 1988] constructs roadmaps from silhouette curves of semi-algebraically defined obstacle-sets. The algorithm is applicable to search spaces of arbitrary dimension, as long as the obstacles can be described semi-algebraically, and was the first algorithm which solved the motion planning problem in general. Its run-time is theoretically polynomial in the number of polynomials and in their polynomial degree, as well as singly-exponential in the number of dimensions. Probably due to its complexity though, no robotic real-world application using this algorithm seems to exist.

Cell Decomposition Besides a skeleton approach, the motion planning problem can be solved by decomposing the free search space into primitive cells, inside of which planning is trivial. A path can be constructed by finding an ordering of interfacing cells between the start-cell and the goal-containing cell. The *Vertical Cell Decomposition (VCD)*, [LaValle, 2006, sec. 6.2.2] for example, can be used to decompose two-dimensional search spaces in time $O(n^2 \lg(n))$. The method is illustrated in Figure 1.5. The cells are trapezoids with vertically aligned parallel sides and can be constructed by ordering the vertices along the x-axis. At each vertex a vertical wall has to be erected in $\mathcal{C}_{free}$, which ends above and below the vertex at the next encountered edge, 1.5a. On the basis of the constructed cells, a roadmap can be generated by connecting the centroids of each cell via the midpoint of the boundaries between two cells, 1.5b.



(a) Vertical Cell Decomposition (b) Roadmap Construction

Figure 1.5.: Combinatorial motion planning with Vertical Cell Decomposition (VCD).
a) The cells are created by erecting vertical divisions for all the corners of obstacles.
b) The contructed cells can be used to generate a roadmap by connecting the centroids of each cell via the midpoint of the boundaries between two cells.

The method can be extended to higher dimensions, by sweeping a hyperplane across the search space and noting the critical points. However, highly dimensional cell decomposition can become quite complex and time consuming.

It can be concluded that the existing combinatorial motion planning algorithms are not well suited for online trajectory planning in higher dimensional spaces.

1.1.5.2. Sampling Based Motion Planning

In contrast to the combinatorial motion planning approaches, sampling based methods do not construct an explicit representation of the free space $\mathbb{X}_{free}$. Instead, they “try-out” the reachability of certain positions with the help of a collision detection module. Since many of the real world problems can be solved by exploring only a small part of the free space, this removes a significant overhead and was the main reason for the success of these algorithms according to [LaValle et al., 2004]. Sampling based algorithms in a way have a more natural comprehension of what makes planning difficult. Instead of the complexity of the *description* of the free space, i.e. the number of polygons or number of dimensions, it is now rather the *form* of the free space that stipulates the difficulty of the motion planning problem and the time the algorithm needs for execution.

Furthermore, general planners can be developed, which are not restricted to certain cost-functions, certain obstacle representations or classes of robots. As a result, sampling based algorithms seem to be the only feasible solution to planning problems with high numbers of dimensions and kinematic and dynamic constraints, also termed kinodynamic planning problems in [LaValle and Kuffner Jr, 2001]. Sampling-based planning algorithms also allow to easily incorporate the time dimension into the planning process, and are therefore predestined for time-varying environments.

A prototypical framework for single query, sampling-based motion planning is given in [LaValle, 2006, sec. 5.4.1]. In all algorithms of this class certain recurring software-modules appear. On the greatest level of abstraction, the motion planner accesses the two modules *Local Planning Method*, which is employed to generate a directly connecting trajectory between any two states x_1 and x_2, and a *Collision Detection* and *Cost Evaluation Module*, which assesses such a candidate trajectory for satisfaction of constraints and its objective value. As depicted in Figure 1.6, this abstraction allows to decouple the algorithmic details of the planning algorithm from the domain dependent models and representations of the robot and the environment.

An abstract algorithm complying to the framework is given in algorithm 1: The design freedom of such algorithms lies in the manner, in which the order of exploration is handled. In the vertex selection step in line 2, the algorithm may prefer to select the vertex, which for example has the smallest distance to the goal, or which bears the maximum potential

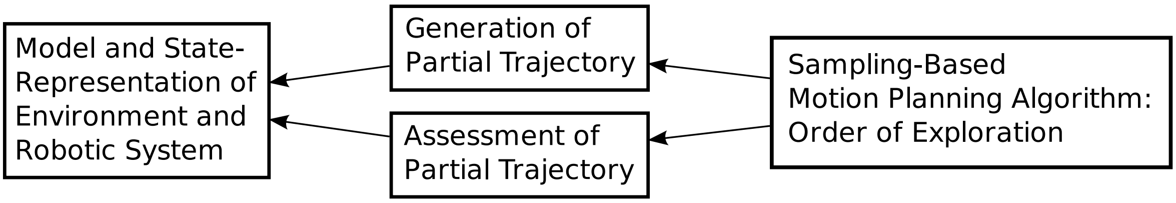

Model and State-Representation of Environment and Robotic System
Generation of Partial Trajectory
Assessment of Partial Trajectory
Sampling-Based Motion Planning Algorithm: Order of Exploration

Figure 1.6.: General structure of sampling-based motion planning algorithms (loosely after [LaValle, 2006, Figure 5.1])

for exploration. Additionally, an algorithm might decide to connect several x_{new} to one x_{cur} at once.

Algorithm 1 Prototypical Sampling-based Algorithm, after [LaValle, 2006, p. 217]

```
SampleAndSearch
 1  Initialization: Create a graph-represented search structure $\mathcal{G}(V, E)$,
        which will contain all accessible states in $V$ and the connecting trajectories in $E$.
        All states which the solution must pass along, can be initially inserted into $V$.
 2  Vertex Selection Method: Choose a vertex $x_{cur} \in V$ for expansion,
        according to the algorithms exploration strategy.
 3  Local Planning Method: For some intriguing state $x_{new} \in \mathbb{X}_{free}$:
 4      Let the Local Planning Module calculate a trajectory $\tau$ from $x_{cur}$ to $x_{new}$.
 5      Check that $\tau$ is collision free with the Collision Detection Module.
 6              If not collision free, go to line 2.
 7      Let the Cost Evaluation Module calculate the value of $\tau$.
 8  Insert into Graph: The state $x_{new}$ can be added to $V$ as reachable,
        and the trajectory $\tau$ can be added to $E$ as a connection between $x_{cur}$ and $x_{new}$.
 9  Check for Solution: If a solution path as been found, terminate.
10  Else, continue process, goto line 2.
```

The two strategies most widely employed in motion planning are A* based heuristic algorithms, which select the one unexplored x_{cur} for exploration, which yields the smallest estimated cost from the start to the goal, and Rapidly Exploring Random Trees (RRT) based randomized algorithms, which select x_{cur} in favor of exploration of unvisited free-space. An additional distinction between A* and RRT is the representation of the search space, which is constrained to a locally finite graph in case of the A* algorithm, and is represented by random samples in case of the RRT algorithm. Originally, these algorithms sample the work- or configuration-space, however the method is applicable to any kind of search space.

One special subclass of sampling-based algorithms is gained by sampling the command-space and directly selecting controls u that can be applied by the controller. Such a command-space search strategy is typically limited to a constant set of distinct alternatives that are evaluated one after each other.

A* Based Motion Planning A best-first search strategy maintains a set of candidate solutions and decides the next exploration step based on a numeric value f, which indicates the “utility” of an exploration-step. Pearl [Pearl, 1985, p. 48] gives three types of “utility” that might guide such a search: The difficulty of the remaining sub-problem, the estimated quality of complete solutions reachable from a candidate solution and the amount of information gained by the exploration of a candidate solution.

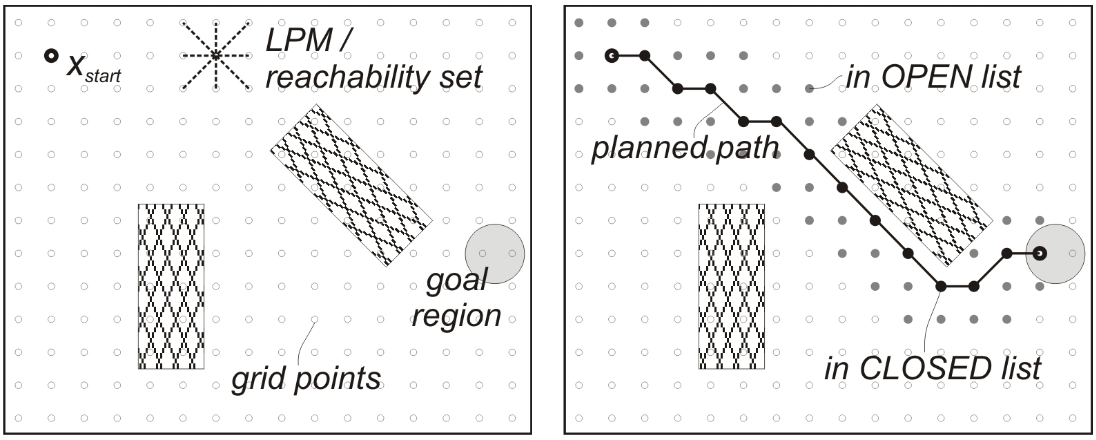
The most widely used best-first search is A*. It was developed in 1968 by Hart, Nilsson and Raphael [Hart et al., 1968] and estimates the quality of the best solution reachable from a given candidate solution. The estimate f is given in form of an additive cost measure, which sums up the already expended cost $g(n)$ of a partial solution n and the expected remaining cost $h(n)$. $h(n)$ is a heuristic function that makes cost-predictions based on domain knowledge.

Each search-step, the algorithm selects the candidate solution from all known candidate solutions, which minimizes f, and discovers all refined candidate solutions, which are directly reachable from that specific candidate solution. The property which defines this reachability is the dual of a directed graph, where vertices stand for subsets of candidate solutions and edges stand for specifications of candidate-solution sets to more refined sets.

In the context of deterministic motion planning, a candidate-solution is equivalent to a plan or trajectory starting at the current state of the system. A complete solution is a movement plan specifying the motion from the start-state of the investigated system, via all intermediate states to an end-state in a region which has been selected as goal-region.

The pseudo-code for the algorithm is given in algorithm 2. A* manages two sets of known candidate solutions: The OPEN-set contains all candidate-solutions, (nodes), which still have to be explored. The CLOSED-set contains all nodes that have been explored at least once. The algorithm starts by inserting the initial problem s, in the context of motion-planning the system’s start configuration, into the OPEN set. The subsequent operations, the *selection-* (line 2) and *expansion-step* (line 4-7), are repeated until a solution is found.

The selection-step retrieves the node n with minimum f-value from the OPEN-set, that is the candidate-solution who’s exploration is deemed most useful. In most implementations of



(a) A* initial problem (b) A* final result

Figure 1.7.: A* motion planning
In this simple example the reachable nodes are set up as a regular 8-connected 2 dimensional grid. In each step all possible next nodes are added to the OPEN list, the best one is explored further. All visited nodes are added to the CLOSED list. When the goal is reached, the planned path is constructed going backwards from the goal to each predecessor node until the starting node is reached.

A*, the OPEN set is represented by an implementation of the *priority-queue* abstract data type, for example a binary heap. The priority of a node is given by its evaluation function f. This allows for fast extraction of the node with minimum f. The expansion-step moves n from the OPEN- to the CLOSED-set and *generates* all successors of n. For each successor n_i the new expended cost $g'(n_i)$ is calculated as the cost of its potential predecessor $g(n)$ plus the cost that has to be spent to reach n_i from n, $c(n, n_i)$. The expected cost of a solution passing through n and n_i is calculated as expended cost plus expected remaining cost $h(n_i)$. The parent-relationship $parent'(n_i) := n$ creates a *pointer-path* leading from any known candidate-solution via all ancestors to the initial problem s.

$$g(n_i) := g(n) + c(n, n_i) \quad \text{expended cost} \tag{1.1}$$

$$f(n_i) := g(n_i) + h(n_i) \quad \text{expected cost} \tag{1.2}$$

If a successor n_i is already known, $(n_i \in OPEN \cup CLOSED)$, its associated values $g(n_i)$, $f(n_i)$ and $parent(n_i)$ are replaced by the new values $g'(n_i)$, $f'(n_i)$, $parent'(n_i)$ only if the new cost is better, $(g'(n_i) < g(n_i))$. If the f-value of node n_i is lowered, although it had previously been expanded, the f-values of its successors are invalid and n_i has to be *re-expanded*. The algorithm terminates as soon as the selection-step produces a node that is identified in line 3 to belong to the set of goal-states.

A* is guaranteed to terminate in finite time, if the search space is locally finite, [Pearl, 1985, Theorem 1]. A search space is locally finite, if each node is allowed only a finite number of successors. The size of the graph itself though may be unrestricted. According to [Pearl, 1985, Theorem 2], A* terminates with an optimal solution, if the heuristic h is admissible. An admissible heuristic is required under all circumstances to underestimate the exact remaining solution cost h^*: $h(n) \leq h^*(n)$. The closer the heuristic h comes to the exact remainder h^*, the more guidance it can give to an A* search. If two heuristics h_1 and h_2 are compared, the heuristic h_2 is said to dominate h_1 if the guidance of h_2 is better than the guidance of h_1 for all non-goal nodes, $h_2(n) > h_1(n)$. An A* algorithm using h_2 is guaranteed to be more efficient than an algorithm using h_1, according to [Pearl, 1985, Theorem 7].

Algorithm 2 A* Planning Algorithm, adapted from [Hart et al., 1968, p. 102]

```
 Input: A start-vertex $s$ and a set of goal vertices $\Gamma$.
 Output: A goal-vertex $n$ and an optimal cost $f$
1  Mark $s$ "open" and calculate $f(s)$.
2  Select the open node $n$ whose value of $f$ is smallest.
   Resolve ties arbitrarily, but always in favor of any node $n \in \Gamma$.
3      If $n \in T$, mark $n$ "closed" and terminate the algorithm with $n$ and $f(n)$.
4      Otherwise, mark $n$ closed and determine all successors $n_i \in succ(n)$.
5          Update $g(n_i)$, $f(n_i)$ and $parent(n_i)$ for each successor $n_i$.
6          Mark as open each successor not already marked closed.
7          Remark as open any closed node $n_i$ which is a successor of $n$
           and for which $f(n_i)$ is smaller now than it was when $n_i$ was marked closed.
8          Go to step 2.
```

A discretization of the naturally continuous search space has to be found. This can either be achieved by sampling the command space, i.e. deciding on a finite set of actions that may be applied to move through the configuration space, thereby implicitly discretizing the configuration space. Or by a direct discretization of the configuration space, which allows only actions to be taken that move from one discrete point in the configuration space to another point in a specific, finite neighborhood.

The most common methods of configuration space discretization are regular grids or lattice structures. The problem is here to choose the neighborhood in such a way that the system's action space is sampled densely enough while the neighborhood is kept as small as possible to prevent redundant node explorations.

The publication [Ferguson and Stentz, 2006] describes a replanning A* derivative, called *Field D**, with a new neighborhood selection method for two-dimensional grids. Instead of restricting the heading of edges to four or eight discrete angles, the Field D* algorithm is able to produce edges of arbitrary orientation by interpolating between adjacent grid-points.

The algorithm *Theta** is presented in [Nash et al., 2007] as a compromise between visibility graphs and four- or eight-connected two dimensional grids. It combines the advantage of minimal length paths in visibility graphs with the superior runtime of grid-based planners by adding direct line of sight connections to the standard grid-neighborhood.

In [Montemerlo et al., 2008, pp. 583ff] the utilization of a *Hybrid A** algorithm is reportet. The "hybrid" discretization in an A* planning algorithm is used for free-form navigation during the Defense Advanced Research Projects Agency (DARPA) Urban Challenge. The action space is discretized to create a finite set of controls. The system is simulated with each control in the set as a constant control input for a preselected amount of time Δt, each simulation generating one edge in the search graph. During execution of the search, reachable configuration space positions are grouped according to a predefined grid to prevent the search space from growing exponentially with the planning distance. By approaching the problem from the action space side, every solution plan is automatically realizable and contains a description of the controls necessary for execution. The authors note that the approach is not complete and fails to find solutions more often for large Δt.

In a similar approach presented in [Pivtoraiko and Kelly, 2005*a*] Pivtoraiko and Kelly use a "primitive path set generation" to create offline a problem specific neighborhood for grid-based search algorithms, which respects the system's motion constraints and at the same time supplies the algorithm with a set of discrete representations of states to reduce computational complexity. The problem of planning a path through a continuous function space is reduced to a problem of selecting a number of path-primitives from a finite set of alternatives. Pivtoraiko and Kelly refer to the search-space induced by the set of primitive paths as a *state lattice*. The primitive path set can be generated in such a way that the algorithm remains resolution complete, see [Pivtoraiko and Kelly, 2005*c*].

The utilization of a set of (offline generated) motion primitives to discretize the configuration space is already very similar to a sampling of the command space, as mentioned above. Other examples of implicit discretization of the configuration space by command space sampling can be found by combining local (reactive) "planning" methods introduced in Section 1.1.4 with the A* search paradigm. Depending on the type of local method, the local method may be used both as Local Planning Method to populate the OPEN set and also as Cost Evaluation Module to estimate the expended and/or expected cost for each candidate solution. This kind of combination leads to A* based variants such as VFH*, [Ulrich and Borenstein, 2000].

For the use of A*-based searches under real-time restrictions so called Anytime extensions exist. Their goal is therefore to quickly produce a highly suboptimal solution, and to improve the solution as long as time is remaining. The most common methods are *Anytime Repairing A* (ARA*)*, see [Likhachev et al., 2004], and *Anytime Weighted A* (AWA*)*, see [Hansen and Zhou, 2007]. Based on Weighted A* [Pearl, 1985, pp. 87ff], this is generally achieved by balancing the cost functions in such a way that it first favors exploration toward the goal more or less disregarding the optimality of the found path. Then the cost function is updated incrementally to produce more optimal results.

Another useful addition to A* methods can be found in incremental search algorithms. These assume that planning is a continuous process as a form of adaption to the changing environment. To improve the reaction time, these algorithms try to reuse results from previous planning cycles. Examples are Dynamic A*, also called *D** [Stentz, 1994] with variants such as *D* Lite* [König and Likhachev, 2002] and *Focused D** [Stentz, 1995], *Lifelong Planning A* (LPA*)* [König et al., 2004], *Adaptive A** [König and Likhachev, 2006], or *Fringe Saving A* (FSA*)* [Sun and Koenig, 2007; Sun et al., 2009].

RRT Based Motion Planning The class of randomized sampling based motion planners introduced in the 1990s enabled the solution of high-dimensional and dynamically constrained motion planning problems. The first well-known randomized sampling based algorithm was the Randomized Path Planner [Barraquand and Latombe, 1990] in 1990, which is built on potential fields and random walks. According to [Lindemann and LaValle, 2003], “for the following decade virtually every significant sampling-based motion planning algorithm used randomization”.

Rapidly Exploring Random Trees (RRT), also termed Rapidly Exploring Dense Trees (RDT) in [LaValle, 2006], constitute a family of randomized search algorithms that have been explicitly developed for motion planning. They solve the problem of search-space discretization on-line: A search tree is grown through the search-space in such a way that the resolution of the tree is iteratively refined.

The search space is sampled randomly. For each selected sample $\alpha(i)$ the nearest point x_{near} in the already existing search tree is found and a local planning method (LPM) tries to connect α to x_{near}, see Figure 1.8. The LPM can ensure that the planned path obey all kinematic and dynamic constraints to create only realizable solutions. If the direct link to the sample α is obstructed by an obstacle, only the first collision-free part from x_{near} to the last collision-free position x_{new}, is added to the tree.

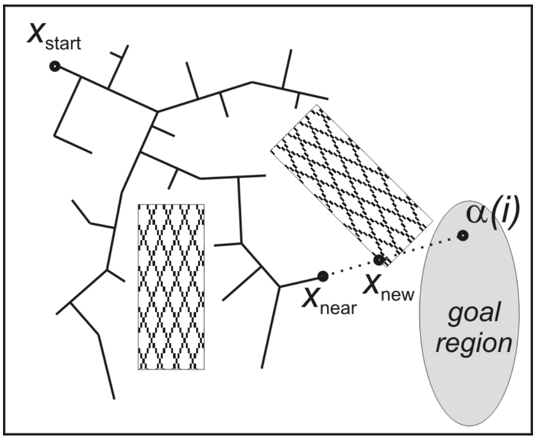


Figure 1.8.: RRT motion planning
A sample α is chosen randomly and the nearest point x_{near} on the explored tree is selected. Then a local planning method (LPM) is invoked to plan from x_{near} towards α either a certain distance or until a collision is detected. The endpoint of this LPM is added as new point x_{new} to the tree.

A speed-up of the planning process can often be gained, if the samples $\alpha(i)$ are not selected from a uniform distribution, but preferably from "important" regions of the search-space.

It can be argued that the goal-region is one of these "important" regions, since it must contain part of the solution. Therefore, a goal-bias can be introduced: Each sample is drawn with probability p from the usual sampling-space and with probability $(1-p)$ from the goal-region.

Further, a lot of ideas were published on sampling-biases regarding obstacles. These ideas target the enhanced performance on motion planning problems with "narrow passages" that are otherwise very difficult for RRT-based algorithms. [Geraerts and Overmars, 2006] details different alternatives such as *Gaussian-Sampling*, *Bridge-Test-Sampling*, *Obstacle-Based-Sampling*, *Nearest-Contact-Sampling*, or *Medial-Axis-Sampling*. However, [Geraerts and Overmars, 2006, Table 4] shows that these heuristic sampling-methods only gain speedups for specific problem-instances, but on average perform worse than uniform-sampling.

A basic RRT algorithm is exploration oriented and possesses no inherent method to prefer "better" over "worse" paths. In [Urmson and Simmons, 2003] three methods are studied which introduce such a cost control into the RRT-framework. The methods bias the selection-step towards higher quality predecessor nodes x_{near}. Heuristically-guided RRT (hRRT) is the simplest: It draws a sample $\alpha(i)$ and calculates the *nearest* neighbor x_{near}. If x_{near} passes the quality-based selection test, it is selected and explored. Otherwise new samples are drawn and new neighbors are calculated until a quality test succeeds. The algorithms Best of k-nearest neighbor RRT (BkRRT) and Iterative k-nearest neighbor RRT (IkRRT) require calculation of the k nearest neighbors for each search-space sample $\alpha(i)$. In *IkRRT* all k neighbors are tested in order of quality, the first successful is selected. *BkRRT* tests only the neighbor with the highest quality value and re-samples on failure. The algorithm BkRRT gives best results in terms of quality, but is up to a magnitude slower than the basic RRT.

RRTs are especially suitable for anytime implementations, because the first feasible solution can be generated with a total focus on exploration and might outperform smallest-cost first strategies as A* in situations where the 'small-is-quick' principle is misleading. In an *anytime RRT* algorithm presented in [Ferguson and Stentz, 2006] the improvement is enforced by adding only nodes to the tree with an expected cost smaller than the cost of the last solution.

Incremental search strategies are also possible. The most common are *Extended-RRT (ERRT)* [Bruce and Veloso, 2003], *Dynamic-RRT (DRRT)* [Ferguson et al., 2006], and *Multipartite-RRT (MPRRT)* [Zucker et al., 2007].

A very similar concept to RRTs are *Probabilistic Roadmap Planners (PRM)*. However, they create roadmaps to find multiple paths rather than just a single one. For more details see for example [Kavraki et al., 1996].

Incremental anytime RRT approaches can be formed by a combination of the described RRT extensions, exemplified by the combination of an Anytime-RRT and DRRT algorithm as proposed in [Ferguson and Stentz, 2007].

Although RRT based motion planning algorithms are rather easy to implement and the random sample selection provides some advantages for high-dimensional and dynamically constrained search spaces, this randomization also constitutes the major drawback: The outcome is neither predictable nor repeatable with regards to the result and the execution time, i.e. two consecutive planning updates might produce very different solutions in very different computation times for the same situation. Further (at least for most RRT variants) the resulting planned motion is not optimal with regards to any criterion.

1.1.5.3. Heuristic Motion Planning

Purely heuristic motion planning approaches are usually inherently incomplete. These algorithms work most of the time but cannot offer any guarantee that a solution is found if it exists. However, they have received increasing interest, especially in the area of automotive collision avoidance and autonomous driving. In these areas, completeness is a still a valuable feature but often not strictly required. Rather, it is of higher interest if certain paths or maneuvers are possible, like a lane-change to the left. If no viable alternative is found, a backup solution like automatic braking can often be applied.

Examples for such motion planning approaches are simple *geometric motion planning* and *rule- or behavior-based motion planning* methods.

Geometric Motion Planning The simplest geometrically planned path consist of linear segments. [Fraichard, 1991] presents a planning method for static, structured environments that smooths such a piecewise linear path to produce concatenated arcs and straight pieces. [Fraichard and Ahuactzin, 2001] discusses the possibility to further enhance the planning result by the use of clothoid pieces to create a continuous curvature along the path.

In a different geometric approach to create drivable, curvature continuous paths, splines can be used. In [Simon, 2003; Simon and Becker, 1999] Simon proposes path planning with Bezier splines through bounded corridors in static, structured environments.

The above approaches consider only static environments and deal with moving obstacles only by frequent replanning, but do not regard their motion within each planning step. The velocity is usually not planned simultaneously. It is either assumed to fixed or planned in a second step.

Much more specialized to collision avoidance, [Lammen, 1993] decides between braking and evading. A braking maneuver decelerates the vehicle to keep a constant distance to a leading obstacle or sufficiently to avoid a collision with cross traffic. The evasion maneuver consists of a quadratic curve at a constant velocity which results in a discontinuous lateral acceleration. If possible, an evasion maneuver is preferred over braking in this approach.

Similarly, [Ameling, 2002] decides between braking, evading left, and evading right. The evasion paths target a certain lateral offset. In this approach they are comprised of four segments of third-order polynomials to approximate clothoids. This leads to smoother lateral accelerations and steering rates. Further, the evasion maneuvers feature combined steering and braking and always end in a stand-still. A decision module chooses the maneuver that can be initiated last to still avoid the collision.

Weisen and Milder follow the same train of thought. In order to achieve a continuous and smooth lateral acceleration along the planned trajectory, they demand the lateral acceleration to have a sinusoidal form as function of the longitudinal distance x, [Mildner, 2004; Weisen, 2003]. Assuming a constant velocity this yields the path's curvature which can be approximated by a third order polynomial and integrated to obtain the desired collision avoidance path as a fifth order polynomial. Along the path the remaining tire adhesion potential that is not used for steering is used to decelerate the car. The fact that at a reduced velocity, smaller turning radii become possible is neglected to facilitate the calculations. The planned maneuver always ends in a stand still.

[Isermann et al., 2008] presents a very similar motion planning approach. Instead of a polynomial, the resulting path is given explicitly as a sigmoidal function, thereby reducing the distance that is traveled until stand still is reached.

Advantages of such geometric approaches can be found in their predictability and their short and reliable execution times. The major disadvantage lies in the fact, that the planned trajectory is not general enough to allow alternative or more complex maneuvers and solve more general motion planning problems.

Behavior- or Rule-Based Motion Planning A very different heuristic approach is behavior- or rule-based motion planning. The basic idea is to discretize possible motions into distinct maneuvers and to emulate human maneuver-based decision making. In order to do that, often fuzzy logic is used. Therein the situation, the possible maneuvers, and the decision rules are expressed in linguistic terms; for example: “IF the car ahead is *slower* THEN *brake*”.

One example of such an approach that uses fuzzy logic can be found in [Lages, 2001]. Lages designs a system that is capable of braking, steering and combined braking and steering in order to prevent a collision with static obstacles. The system depends heavily on specific experimental data of the used test vehicle to tune the rule-basis and decision thresholds.

Dickmann and others use a behavior and rule based approach that chooses a certain behavior that is to be executed according to the situation. This choice is performed by means of fuzzy logic. The approach regards that a behavior is only executable if all required lower level motion primitive or “skills” are available and offers a “backup behavior” in case the first choice becomes impossible because one of the associated skills is no longer accessible, [Brüdigam, 1994; Dickmanns, 2005; Pellkofer, 2003].

1.1.5.4. Optimization Motion Planning

In addition to the mentioned combinatorial, sampling-based, and heuristic approaches there exist a number of optimization methods which are generally also referred to as motion planning approaches when applied to a motion planning problem. Most of them can be attributed to the paradigm of mathematical programming, i.e. an "optimal solution" is acquired numerically under certain constraints which are formulated as inequalities or differential equations. Because of the numerical algorithm, an initial guess or start solution is necessary. Thus, the task of motion planning can be distributed between the initialization of the starting solution and its optimization. The necessary quality of the starting solution depends on the sophistication of the optimization algorithm.

Different optimization algorithms for motion planning can be classified as local or global. Whereas global optimizations search the global optimum and have the ability to "jump" from one local optimum to another, local optimizations merely find a single local optimum that is closest (for this algorithm) to a starting solution. The distinction sometimes blurs and some algorithms can also be tuned to be more local or global.

Both classes have their distinct advantages and disadvantages for an application in motion planning. A global optimization poses minimal requirements on the starting solution, since, apart from very ill-posed problems, any starting solution converges to the same global optimum. On the other hand, this might be undesirable if a certain type of solution is targeted, as for example to pass an obstacle on the left. In this case, only homeotopic deformations are desired and a local optimization might be better suited.

Optimal Control Motion Planning The Nonlinear Trajectory Generation (NTG) algorithm developed in [Milam, 2003] treats motion planning as a constrained optimal control problem. Among others the algorithm is applied to micro satellite formation flying.

A similar optimal control approach is adapted in [Bertolazzi et al., 2007] for real-time motion planning of a vehicle in traffic. The method achieves a speed-up compared to previous similar approaches by eliminating hard constraints and replacing them with penalty functions instead. The algorithm uses a combination of finite differences and a Newton-Broyden algorithm.

A comprehensive survey of optimal control methods in general can be found for example in [Betts, 2010].

Genetic Algorithm Motion Planning Genetic Algorithms (GA) or Evolutionary Algorithms are global optimization techniques that represent the subject, in this case a planned path or trajectory, by genes. A certain combinations of genes, i.e. a certain planned path, is called an individual and all currently "living" individuals together constitute the population. The algorithm then tries to find an optimum by emulating natural evolution in a repeated two-step process: reproduction and selection. First, new individuals are created by recombining the genes of individuals from the existing population (crossover) or selected mutation of certain genes. Second, all individuals of the population are evaluated with regards to a certain fitness function and only the fittest survive. This basic concept can be complemented by a number of variations. Especially, the genes of new individuals can be repaired before the evaluation, i.e. a planned path can be made collision-free by small local changes. An overview and a general introduction to genetic algorithms can be found for example in [Mitchell, 1998].

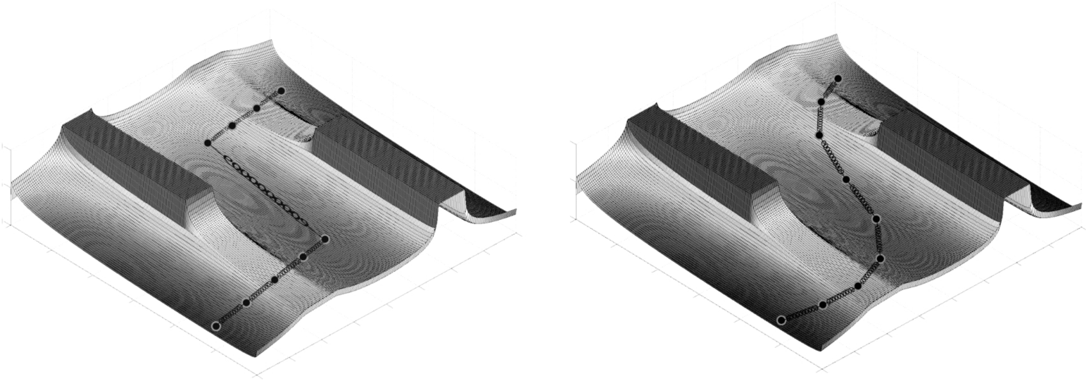
Applied to motion planning, the genes can be for example trajectories parts, IDs of visited nodes on a grid, or command space samples at certain times instances. These genes combine to represent the total planned trajectory. Depending on the number of different genes and the number of genes to a whole planned trajectory, the total number of possible combinations is very large and it becomes unfeasible to test all possible combinations. Therefore, Genetic Algorithms can be used to identify good combinations.

Sugihara proposes a GA for path planning and trajectory planning of an autonomous mobile robot, [Sugihara and Smith, 1997]. The environment is represented by a 2D occupancy grid that is considered static for each planning instant. Time varying environments are addressed only by adapting previous solutions in the next planning instant. In order to enhance the performance, each path is described by a fixed length binary string of genes which represent distances and directions of path pieces. To ensure this fixed length, all paths are required to be monotonous in at least one direction of the grid. [Castillo et al., 2006] extends Sugihara's GA algorithm by introducing a difficulty value to each grid cell and adding a second objective into the fitness function. Paths are selected with regards to minimal length and difficulty.

One of the outstanding characteristics of genetic algorithms is that they are inherently parallel and are therefore predestined for distributed computations. If not implemented in parallel, on the other hand, the large computational and memory requirements can become quite challenging, [Ismail et al., 2008], which leads the voluntary restrictions like monotony of the path as addressed above. In an attempt to decrease the necessary computational time, [Koryakovskiy et al., 2009] sketches the idea of a local map and does not plan the whole path to the goal at once but only to a receding horizon. Further, the article proposes an interpolation with Bezier splines to create smooth paths and suggests to regard the obstacle velocity in the calculation of a distance to obstacle-based fitness function.

Genetic Algorithms are known to be very robust for complex and ill-behaved optimization problems, [Goldberg, 1989]. However, if a good model of the problem exists and the optimization problem is not too ill-behaved, often superior optimization techniques exist that might for example exploit knowledge about the direction of path deformation that leads to a better path. So far it seems that Genetic Algorithms are not yet suitable for the application to the online planning of drivable trajectories in a quickly changing, dynamic environment or higher dimensional search spaces.

Predictive Potential Field Motion Planning While former potential field methods were purely reactive approaches, where resulting virtual forces acted directly on the mobile robot, Quinlan and Khatib developed a deliberative approach by letting the virtual forces act upon the planned path rather than on the robot. They introduced the concept of representing the planned path by an elastic band that is deformed by virtual forces that push the band away from obstacles, [Quinlan and Khatib, 1993*b*].



(a) initial path (b) deformed final path

Figure 1.9.: Predictive potential field motion planning, [Brandt, 2007]
The environment is modeled by a potential field, the planned path is represented by discrete nodes, interlinked by linear springs. Forces produced by the potential field and the linear springs deform the planned path.

The elastic-band method presented in [Quinlan and Khatib, 1993*a*] samples a given path and creates bubbles around these points. The size of these bubbles depends on the distance to the closest obstacle. During the deformation process the bubbles are maximized in size. The path is guaranteed to be collision-free as long as the bubbles overlap sufficiently along the path. In order to achieve a smooth path, the bubble centers are then connected with splines to form the final planned path.

The form of the bubbles depends on allowed movements of the planning robot or vehicle. While a circle, sphere or hypersphere results for non-restricted movements in cartesian spaces, the form of the bubble becomes more complex if non-holonomic constraints are regarded as described in [Khatib et al., 1997; Vendittelli et al., 1999]. Further, the size of the bubbles can be used as means of implicitly calculating safe velocities along the path, as illustrated in [Furtwängler et al., 1998].

However, even though kinematic constraints have been regarded by the form of the bubbles and the spline interpolation creates smooth paths, dynamic limitations were not considered within the this path planning. The major limitation, however, consists in the fact that all obstacles were considered static and no velocity profile was planned.

Later, the elastic band approach was adapted for automotive motion planning by Hilgert and others, see for example [Hilgert et al., 2003]. As in the original approach in [Quinlan and Khatib, 1993*b*], the elastic band is discretized into equally spaced nodes that are interconnected with linear springs. The first and last node are fixed. The obstacles are represented by a number of relatively small safety circles ($R = 1\text{m}$). Within these safety circles, potential fields are defined that increase quadratically toward the center. The virtual obstacle forces are derived as negative gradient of these potential fields and cause a deformation of the elastic band. [Hirsch et al., 2005] uses the same elastic band path planning and adds a second step where a linear bicycle model is used to evaluate the driveability of the path and to determine the necessary steering angle to follow it. This first adaption to automotive application still has some disadvantages, though. The main disadvantage lies in the representation of the obstacles. Road sides are not considered and all obstacles are considered to be static, which prevents a truly predictive planning. Further, the potential fields have an extremely limited area of influence and due to their quadratic form, this influence might be compensated completely by the forces of the linear springs inside the elastic band. Therefore, the final path cannot be guaranteed to be collision free even directly at the nodes.

In a different adaption, [Gehrig and Stein, 2007] uses an elastic band path planning for vehicle following. The last node of the band is attached to the preceding vehicle. The elastic band is also represented by virtual linear springs. However, these forces receive an offset such that the planned path matches the path of the preceding vehicle in the absence of obstacles. The ego vehicle is represented as rectangle, obstacles as polyhedra. Obstacles and road sides are modeled as potential fields that depend quadratically on the distance. Similar to [Hilgert et al., 2003], the obstacle potential fields are active only within a certain safety distance from the obstacles. The obstacle's motion is not extrapolated but the potential field of each each obstacle is modified around a static position depending on the relative velocity, as proposed by [Krogh, 1983] for the reactive potential field approach.

[Brandt, 2007] adapts the method of elastic bands further for spatially and temporally predictive motion planning. As in all previous approaches, the velocity is not planned but assumed to be constant or otherwise known. Brandt then determines the points of time for all nodes of the elastic band and extrapolates the motion of all obstacles for these points of time. The repulsive potential fields are then built around these extrapolated positions. The final path is defined as the equilibrium configuration, where all forces cancel each other out.

Differing from the elastic band formulations above mentioned, the obstacle and roadside potential fields are not limited in range, which leads a smoother overall potential field, weighing distance to obstacles versus deformation of the elastic band for all nodes and not only in the close vicinity of an obstacle. Furthermore, the potential fields are defined logarithmically with regards to the distance. This has the advantage that the repulsive forces tend to infinity as a node approaches an obstacle or roadside. Therefore, each node of the final path is guaranteed to be collision free. Brandt determines that for path planning on low curvature streets it is sufficient to shift the nodes only in lateral direction in a fixed reference frame and to remove one degree of freedom.

However, the unidimensional displacement of the nodes in a single direction and the used road model limit the approach to low curvature streets or very low planning distances. Furthermore, no kinematic or dynamic limitations are considered.

All predictive potential field approaches mentioned above are purely path planning or rather path deformation methods and no velocity profile is planned simultaneously. To integrate the velocity in the planning algorithm and to provide an integrated trajectory planning algorithm, the dimensionality can be increased.

Kurniawati and Fraichard suggest a trajectory deformation method that applies such an augmentation. The environment is modeled by potential fields in an x-y-t workspace $\mathbb{W}_T = \mathbb{W} \times \mathbb{T}$ and the trajectory representation and deformation takes place in a state space that is augmented by the time dimension $\mathbb{S}_T = \mathbb{S} \times \mathbb{T}$, [Kurniawati and Fraichard, 2007]. As in other approaches, virtual forces are calculated from the obstacle potential field which are then translated to forces in the augmented configuration space. Additional state space forces are determined to regard dynamic limitations. These forces push a node to the center of forward and backward reachability sets which depend on the dynamic model of the given mobile robot or vehicle, see [Fraichard and Delsart, 2009].

In each planning, the trajectory is deformed one step in the direction of the resulting total forces at the nodes. Because no equilibrium solution is sought, the results differ depending on the initial trajectory and the number of planning steps. Further, the high dimensional representation of the trajectory makes the algorithm computationally more expensive which might be a reason that no equilibrium configuration is computed. In addition, the reachability sets become very complex for realistic vehicle models. The idea is implemented for moving circular obstacles and a double integrator point mass vehicle model.

1.2. Motion Control

If a certain desired trajectory has been determined by a motion planning module, it still needs to be “executed”. This task of motion control is usually taken over by a separate module. Trajectory following consists of a lateral control component that keeps the vehicle on the planned path and a longitudinal control component that controls the velocity. Different motion control approaches for trajectory following can be separated in *decoupled motion control* where lateral and longitudinal control are realized in two separate controllers and *integrated motion control* where the two components are integrated in a single controller. Further differences can be found for example with regards to the controller type, the vehicle model that is used for the controller design, and the control variables.

Figure 1.10 illustrates the main components for motion control and some of the control variables that are commonly used. It is exemplified, that the control difference might not necessarily be evaluated at the center of gravity V^* of the ego vehicle but rather at a preview point PV in front of the vehicle.

A generalized control loop that covers a majority of possible control approaches is depicted in Figure 1.11. Its basic components include a feedback controller for general tracking, a feedforward controller to enhance the response time and counteract known disturbances,

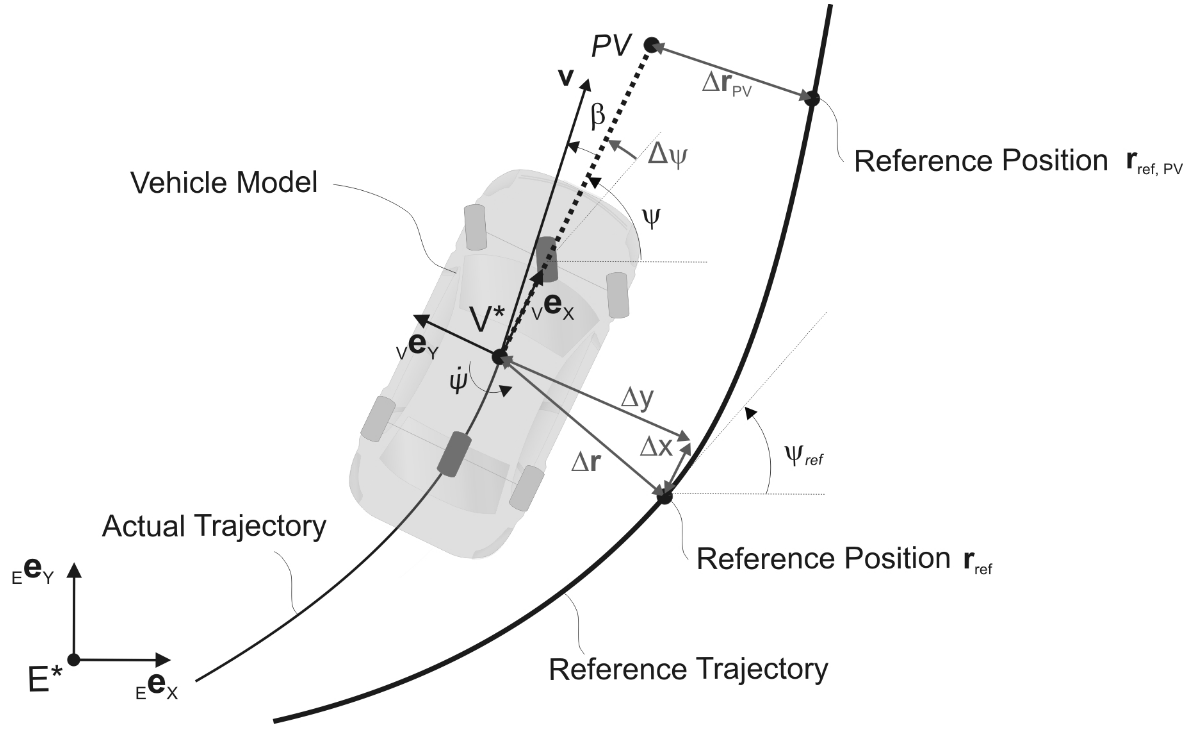


Figure 1.10.: Main components in motion control
Among others, motion control approaches differ with regards to the type of reference trajectory, the used vehicle model, and the used control variables.

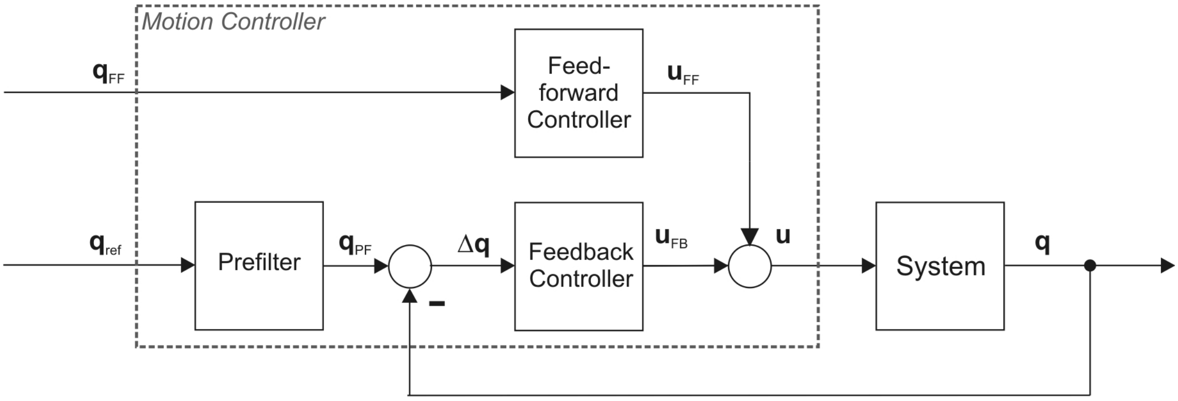


Figure 1.11.: General control loop
Most linear control approaches can be described by this control loop and its basic components: a feedback controller for general tracking, a feedforward controller to enhance the response time and counteract known disturbances, and a prefilter to further compensate slow or otherwise undesired closed loop dynamics.

and a prefilter to further compensate slow or otherwise undesired closed loop dynamics. The system consists of the vehicle and some subordinate control loops.

The reference $\mathbf{q}_{\text{ref}}$ may consists for example of the desired lateral position y_{ref} and a reference orientation ψ_{ref}. The input $\mathbf{u}$ to the system could be e.g. the desired steering angle δ, or a desired longitudinal acceleration which are then controlled by a subordinate control loop.

1.2.1. Decoupled Motion Control

In the majority of applications, decoupled motion control approaches are applied. They offer the advantage that each separate controller is less complex and has less control variables which facilitates the design process.

Longitudinal Control The longitudinal control usually aims to reach and maintain a certain desired velocity or a velocity profile. In automotive applications it has the longest tradition in the application to cruise control systems, where a constant velocity constitutes the control reference, [McMurray and Sniers, 1963]. Later longitudinal controllers for intelligent or adaptive cruise control applications or platooning were developed, where the main focus shifted to controlling a certain relative position to a lead vehicle, see [Swaroop et al., 2001; Xiao and Gao, 2010].

Other applications exist in automated driving either on test grounds or for collision avoidance. Here, often the longitudinal velocity control is combined with a lateral control for path following and the emphasis is placed mostly on the lateral control part. Longitudinal controllers differ in their design but also with regards to the reference variable. A velocity profile is usually given either with respect to space ($v(x)$) or time ($v(t)$). Additionally, the desired acceleration is often used to add a feed forward control component for a faster control response. However, if only the velocity (and its derivative(s)) are used as control variables, the longitudinal position error might accumulate over time. Therefore, the longitudinal position must be used as control variable for successful tracking of a certain position over time.

Lateral Control The lateral control component tries to keep the ego vehicle on a certain given path $y(x)$ and is not concerned with deviations from a desired velocity or with longitudinal position errors. Especially for collision avoidance, the lateral control has received a much higher attention than the longitudinal control, [Schorn et al., 2006]. The control variables for lateral control are usually the lateral deviation Δy and sometimes also the orientation error $\Delta\psi$. In addition, their derivatives might be used as well.

A vast variety of different controller designs exist, see [Lunze, 2008], many of which have been applied to lateral control of vehicles. The majority of approaches seems to use a linear single track model as vehicle model that is used in the controller design process. Some approaches use a nonlinear model and apply an input-output-linearization to be able to apply linear control techniques again.

Almost all controllers for lateral vehicle guidance rely on the steering angle δ or the steering rate $\dot{\delta}$ as sole actuating variable. Another option to influence the lateral vehicle dynamics would be differential braking, where only one wheel on the front or rear axle is braked as used in the ESP. However, this kind of actuation is mostly just used in an subordinate stabilization control, as in vehicle dynamics management (VDM), see [Trächtler, 2005].

Among the most basic controller types are PID controllers and their variants. Weilkes shows in [Weilkes et al., 2005] that a combination of a PI controller for the lateral deviation Δy and a P controller for the orientation error $\Delta\psi$ yields sufficient results for lane keeping. The controllers are augmented by a curvature dependent feedforward control. Similarly, Schorn and Isermann test a PD controller with feedforward control and gain scheduling to control the lateral deviation Δy in collision avoidance maneuvers, see [Schorn et al., 2006]. [Söhnitz, 2001] demonstrated satisfactory results for a PIDT$_2$ controller to control Δy for path following with high curvatures, also combining it with a curvature dependent feedforward control.

Adding a slightly different perspective, Switkes and Gerdes devise a potential field controller for lane keeping, [Switkes and Gerdes, 2005]. The lane is modeled by a potential field and any deviation in position or orientation causes a virtual force on the vehicle. In order to make the vehicle move as if this forces existed, a desired steering angle and differential braking are determined. The advantage of the approach is that Lyapunov functions can be derived to prove stability and error bounds for the controlled system. However, the control law itself comes down to a rather simple P controller for lateral deviation Δy and orientation error $\Delta\psi$.

This approach was adapted for path following in [Brandt, 2007], adding a curvature dependent feedforward control for the steering angle. To the author's knowledge the proof of stability on the other hand has not yet been extended beyond straight reference paths.

Taylor uses a Lead-Lag controller for autonomous highway driving. The controller sets the steering angle δ depending on the lateral deviation Δy_{PV} at a preview point PV in front of the ego vehicle, [Taylor et al., 1999]. Note that the distance l_{PV} from the vehicle's center of gravity V^* to the preview point PV can be used as an additional control parameter. For approximately straight reference paths and small orientation errors $\Delta\psi$, l_{PV} actually scales the current lateral deviation Δy versus the orientation error $\Delta\psi$ in the control law according to $\Delta y_{PV} \approx \Delta y + l_{PV}\Delta\psi$.

In an attempt to create superior results, additional variables can be fed back to the controller besides the lateral deviation Δy and the orientation error $\Delta\psi$. This leads to full state feedback controllers where the full state of the used (linear) vehicle model is either measured or observed and fed back to the controller. This approach might give some more freedom in the design of system dynamics and allows to place the poles anywhere, actuator limits permitting. Taylor applies this method, but in his case it yields inferior results to his Lead-Lag compensator design, see [Taylor et al., 1999].

Maurer applies a full state feedback controller design for autonomous highway driving where he switches between a third- and a fifth-order model depending on the velocity, [Maurer, 2000]. Besides the lateral position y and the yaw angle ψ, the fifth-order model regards the yaw rate $\dot{\psi}$, the steering angle δ, and the side slip angle β.

As exemplified by [König and Likhachev, 2006] and [Weilkes et al., 2005], the controller design for full state feedback approaches can be facilitated by optimization methods such as Linear Quadratic Regulator (LQR), see [König and Likhachev, 2006; Weilkes et al., 2005], or H_2, see [Söhnitz, 2001]. In these methods the poles of the controlled system are chosen such that a quadratic costfunction is minimized, thus reducing the error integral or the necessary control effort.

All the above approaches regard linear vehicle models. However, for certain situations, as e.g. for higher lateral accelerations, a linear single track model is oversimplified and no longer accurately represents the vehicle dynamics. Therefore, the performance of controllers that were designed using such a linear model might degrade in these cases. In order to regard more accurate nonlinear vehicle models and still be able to apply linear control theories, some authors have applied an input-output linearization, see Figure 1.12. In the illustrated example, the system, i.e. the nonlinear vehicle model, is combined with a nonlinear decoupling controller such that the closed loop transfer function $\frac{Y(s)}{W(s)}$ is linear and decoupled in w_i, y_i. Therefore, linear control methods such as pole placement can be used in the design process.

Taylor combines the input-output linearization with his Lead-Lag controller. However, in his experiments on low curvature paths the controller performed not as good as the pure

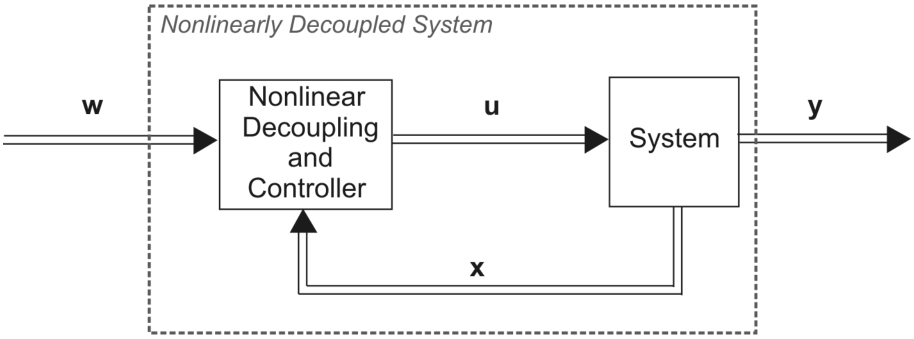

Nonlinearly Decoupled System
w
Nonlinear Decoupling and Controller
u
System
y
x

Figure 1.12.: Input-Output Linearization
A controller is designed such that the nonlinear system is decoupled and the closed loop behavior of controller and system yields a linear behavior.

Lead-Lag controller, [Taylor et al., 1999]. König reports good results of a similar approach to an LQR design with a linear model. The tests where performed on a 1:5 model vehicle, [König and Likhachev, 2006]. In [Söhnitz, 2001], an input-output linearization is combined with a linear integral control and a Kalman filter as observer to reduce the measurement noise. In [Schorn et al., 2006], an input-output linearization approach is applied successfully for collision avoidance. Test results show a maximum lateral deviation of 0.2 m during such a maneuver at 10 m/s.

It can be concluded that many different control approaches can be used successfully for lateral control of a vehicle. The choice of the controller type and the performance of the controlled system among others largely depend on the vehicle, its sensor and actuator limitations and the specific requirements (like high speed, high curvatures, slopes, wind or uneven terrain).

1.2.2. Integrated Motion Control

While a decoupled motion control usually is easier to design and to parametrize, integrated control approaches promise superior results, [Pham et al., 1997; Söhnitz, 2001]. While H. Pham compares several different control approaches for mostly low-curvature autonomous highway driving, [Pham et al., 1997], I. Söhnitz specifically targets trajectory following control for high curvature paths, [Söhnitz, 2001].

The control variables for integrated motion control approaches are largely the same as for decoupled motion control for trajectory following. The controller output usually consists of the steering angle δ as well as a certain accelerating or braking torque $T_{A/B}$ or longitudinal force F_x.

For autonomous highway driving Pham proposes a sliding mode controller based on a linear single track model for integrated lateral and longitudinal control. In order to control a certain position error Δx and Δy, the desired steering angle δ and the desired accelerating or braking torque $T_{A/B}$ are determined. The controller is tested only in simulations and performs well for the tested curvatures of about κ =1/200m.

A very similar approach is used in [Swaroop and Yoon, 1999], where a sliding mode controller is designed for vehicle following during an autonomous lane change maneuver. The control design is also based on a linear single track model. Given a certain desired distance to the preceding vehicle, the control variables consists of the position error Δx and Δy as well as the velocity error Δv_x, which are used to calculate the desired steering angle δ and the desired accelerating or braking torque $T_{A/B}$, as in [Pham et al., 1997]. The controller is also tested only in simulations and performs well for the tested curvatures of κ =1/300m.

As discussed before with regards to lateral control, the design based on linearized models is limited and for certain situations a nonlinear control design may yield superior results. Mayr, for example, concluded that among various controllers he tested for collision avoidance maneuvers, a nonlinear model-based design performed best, [Mayr, 1991]. The controller design is based on the principle of nonlinear decoupling which is further detailed in Section 4.3.1. In order to reduce oscillations, he determines the position error at a preview point PV in front of the vehicle instead of the car's center of gravity V^*. The controller performed well in simulations at $v = 100$km/h and curvatures of $\kappa \leq 1/100$m.

Lammen successfully applies the nonlinear controller design of Mayr to a test vehicle for autonomous collision avoidance, [Lammen, 1993]. However, due to limitations in available sensors and computational resources, the tests are limited to 15 km/h and lateral acceleration of about $a_y = 1\text{m/s}^2$.

As already mentioned for lateral control, the choice of the controller type and the performance of the controlled system among others largely depend on the vehicle, its sensor and actuator limitations and the specific requirements. The more information is available and used about the vehicle state, the vehicle model, or possible disturbances the better the potential performance but the more complex the controller design becomes. In addition, any error in used models might result in larger errors if the controller design depends heavily on their correctness.

1.3. Thesis Contributions and Outline

As detailed, collision avoidance systems are generally based only on simple maneuvers with certain lateral offsets that must be achieved. Even current efforts in purely autonomous driving use sophisticated planning algorithms only for parking lot situations and not for road driving, as detailed in the Annex A regarding the successful DARPA Urban Challenge participants. In part this can be related to still existing sensory limitations, which will most likely be alleviated in future. However, in order to provide a more qualified assistance in complex traffic situations and for more distant planning horizons, an underlying detailed trajectory planning is necessary.

Therefore, as stated above, it is the goal of this work to contribute by devising a novel motion planning algorithm that could be used for autonomously driving vehicles or as basis of future driver assistance systems.

The new proposed motion planning approach is an integrated trajectory planning method that plans both path and velocity profile simultaneously. It centers on a predictive force field trajectory deformation algorithm which is combined with an A*-based trajectory initialization and thus joins the advantages of predictive potential field methods and A*-based graph search trajectory planning. As described in Chapter 2, a given initial trajectory is deformed by a virtual force field. Based on the ideas of existing predictive potential field path planning methods such as [Brandt, 2007; Hesse and Sattel, 2007; Sattel and Brandt, 2008], a predictive virtual force field approach is devised for integrated lateral and longitudinal motion planning, see also [Hesse et al., 2010]. The most fundamental extension lies in the integrated planning of a velocity profile by additional degrees of freedom in the time dimension. In difference to [Brandt, 2007], other enhancements include that the initial state, especially the orientation of the vehicle, is regarded in the planning, and limitations to low curvatures stemming from very rough street approximations have been lifted. (The additional dimensionality is also one major reason, why a different method had to be found to create a viable starting solution.)

The description of the environment by a potential field hazard map and the definition of optimization objectives as virtual forces that “push” the trajectory away from undesired configurations is very intuitive. The deformation of the trajectory depends on the course of the road, the predicted movements of all detected obstacles and the desired traveling speed and minimizes necessary lateral and longitudinal accelerations. The virtual force field approach is a locally convergent method, and therefore depends on an initial guess, i.e. it cannot be executed without a trajectory initialization step. However, in combination with a good trajectory initialization the local convergence becomes a desirable characteristic, since the nature of the maneuver is preserved and initial and optimized trajectory are always homeotopically equivalent.

The initial solution is generated in an A* based trajectory initialization, see Chapter 3. This sampling based search algorithm provides (resolution) completeness and allows fast and efficient coverage of a higher dimensional search space to find a solution that is optimal with regards to a given cost function. In this work, existing A* based approaches are extended to plan trajectories for road traffic in five dimensions x, y, t, ψ, and v and create feasible starting solutions for the force field optimization. Therein, an AWA* variant with anytime characteristics (following [Hansen and Zhou, 2007]) is chosen, where the search space $\mathbb{S} : \langle x, y, t, \psi, v \rangle$ is discretized by an offline generated "state lattice", in an extension of [Pivtoraiko and Kelly, 2005*a*,*c*].

During this trajectory initialization, the best "type" of motion (e.g. follow, overtake before oncoming traffic, overtake after oncoming traffic, etc.) is selected regarding necessary accelerations and path length, and a viable starting solution is created, see Chapter 3. The initial trajectories are also collision free, but could graze them as the algorithm does not include spatial or temporal "distances" to obstacles as optimization criteria. The "optimality" of the solution might be further limited by the resolution of the search space discretization and the allotted execution time. Due to the discretization of the search space, graph searches also tend to create aliasing or "staircase" effects that result from the concatenation of motion primitives or vertices to connect grid points. These disadvantages are alleviated by the before mentioned trajectory optimization step.

As additional goal of this research work, an autonomous test vehicle was set up to demonstrate applicability of devised motion planning approach in experimental tests. Chapter 4 details the modifications, added actuators and sensors, the applied architecture and the implemented controllers. For the implementation of motion control within this thesis an integrated control approach is chosen. One reason lies in the fact that the motion planning provides a position reference $[x(t), y(t)]$ over time in lateral and longitudinal direction. In order to prevent a collision with moving obstacles it is essential not only to follow a certain path but also to be at a specific place at a certain point of time. Therefore, it seems reasonable to apply an integrated control approach.

Furthermore, a decoupled motion control approach was tested that used the potential field control devised by Gerdes and Brandt in [Brandt, 2007], but the performance was unsatisfactory. The vehicle was only stable up to a velocity of about 9 m/s when following a straight path at a constant velocity. The integrated approach proved more stable and yielded superior tracking results for the used test vehicle.

Chapter 5 discusses the results of planning simulations and first experimental tests. The work concludes with a summary and outlook in Chapter 6.

CHAPTER 2

Trajectory Optimization

The optimization of the trajectory is achieved by its submersion into a predictive virtual force field. The choice of method and some principal advantages and disadvantages compared to other state of the art motion planning methods have been elaborated in Chapter 1. This new approach is based on the ideas of existing predictive potential field path planning methods such as [Brandt, 2007; Hesse and Sattel, 2007; Sattel and Brandt, 2008] and other methods as detailed in Chapter 1.

One of the most fundamental extensions of this method lies in the integrated lateral and longitudinal motion planning, i.e. the simultaneous planning of path and velocity profile. This is achieved by additional degrees of freedom in the time dimension, see also [Hesse et al., 2010]. The development of this method has been supported by several student and diploma theses supervised by this author, where differing variants and possible extensions were implemented and tested, see for example [Kahl, 2007; Klötzer, 2007*a*,*b*; Neuhaus, 2008; Stefani, 2009].

The current chapter details the devised predictive force field trajectory optimization algorithm. Further characteristics and advantages are discussed within Chapters 5 (Results) and 6 (Conclusion and Future Prospects).

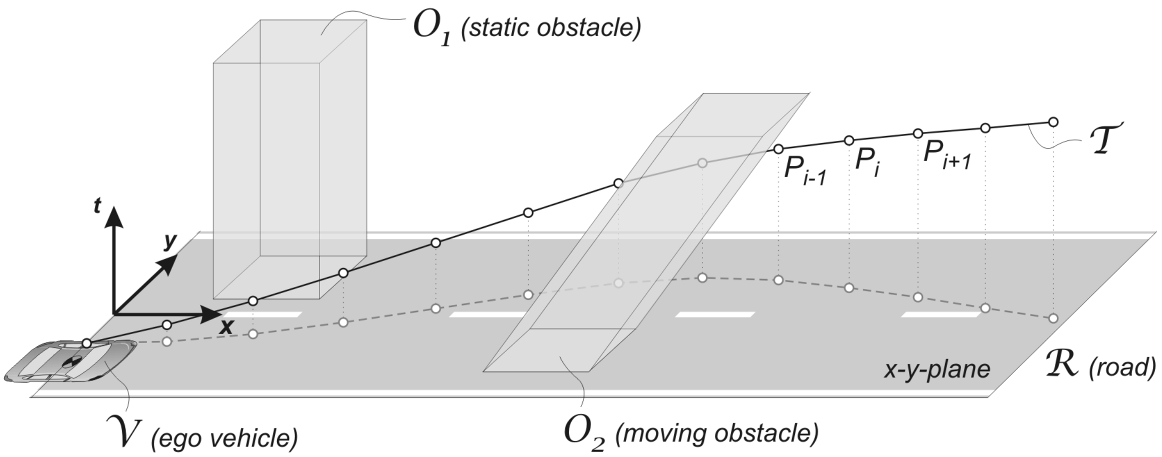


Figure 2.1.: Illustration of workspace
The trajectory optimization is carried out in the augmented workspace $\mathbb{W}_T = \langle x, y, t \rangle$ to limit the computational complexity. The obstacles $\mathcal{O}_|$ are represented as rectangles for each point of time, resulting e.g. in rhombohedra for obstacles moving in a straight line at a constant velocity, see $\mathcal{O}_\in$. The trajectory $\mathcal{T}$ is represented as discrete nodes $P_i = [x_i, y_i, t_i]^\mathsf{T}$ representing the planned states of the ego vehicle's $(\mathcal{V})$ center of gravity $CG = V^*$.

The trajectory $\mathcal{T}$ is given as discrete nodes P_i in the augmented workspace

$$\mathbb{W}_T \quad = \quad \langle x, y, t \rangle \, , \tag{2.1}$$

that includes the spatial dimensions x, y and regards the time t as third dimension as displayed in Figure 2.1. Therefore, a position vector in the augmented workspace $\mathbb{W}_T$ is defined as $\mathbf{r} = [x, y, t]^\mathsf{T}$.

The trajectory is now subjected to a predictive virtual force field $\mathbf{F}$ that pushes the nodes in certain directions and thus deforms the trajectory in order to enhance its safety and driveability. The virtual force field

$$\mathbf{F}\left(\mathcal{T}, \mathcal{R}, \mathcal{O}\right) = \mathbf{F}^{\text{ext}}\left(\mathcal{R}, \mathcal{O}\right) + \mathbf{F}^{\text{int}}\left(\mathcal{T}\right) + \mathbf{F}^{\text{prev}}\left(\mathcal{T}, \mathcal{R}, \mathcal{O}\right) \tag{2.2}$$

comprises three components as illustrated in Figure 2.2. The external force field $\mathbf{F}^{\text{ext}}$ represents the environment and ensures the safety of the trajectory by pushing it away from hazards posed by the sides of the road $\mathcal{R}$ and from obstacles $\mathcal{O}$ and their predicted future movements, see Section 2.1.

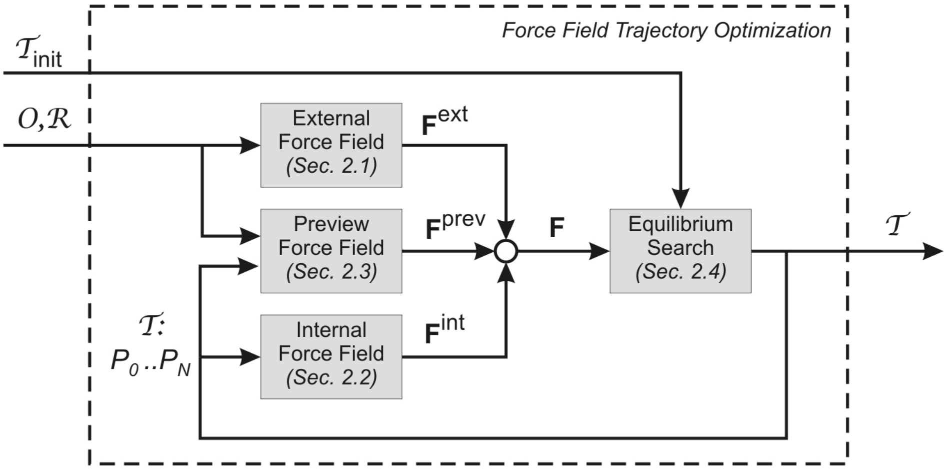

Force Field Trajectory Optimization
$\mathcal{T}_{\text{init}}$
$\mathcal{O},\mathcal{R}$
External Force Field (Sec. 2.1)
Preview Force Field (Sec. 2.3)
Internal Force Field (Sec. 2.2)
$\mathcal{T}$: $P_0..P_N$
$\mathbf{F}^{\text{ext}}$
$\mathbf{F}^{\text{prev}}$
$\mathbf{F}^{\text{int}}$
$\mathbf{F}$
Equilibrium Search (Sec. 2.4)
$\mathcal{T}$

Figure 2.2.: Blockdiagram trajectory optimization

The virtual forcefield $\mathbf{F}$ consists of three components, detailed in Sections 2.1, 2.2 and 2.3. It then deforms the trajectory $\mathcal{T}$ until an equilibrium configuration is found, as detailed in Section 2.4.

The internal force field $\mathbf{F}^{\text{int}}$ punishes dynamically unfavorable trajectories and depends solely on the planned trajectory $\mathcal{T}$ itself. This is detailed in Section 2.2. The preview force field $\mathbf{F}^{\text{prev}}$ introduces an additional look-ahead and depends on both the trajectory $\mathcal{T}$ and the environment $\mathcal{R}, \mathcal{O}$ as presented in Section 2.3.

The virtual force field $\mathbf{F}$ is evaluated at all nodes P_i of the trajectory $\mathcal{T}$ to produce certain forces $\mathbf{F}_i$. The solution to the trajectory optimization problem is defined as the equilibrium configuration of the trajectory in the augmented workspace $\mathbb{W}_T$, where at each node the forces produced by the different force field components $\mathbf{F}^{\text{ext}}$ (see Section 2.1), $\mathbf{F}^{\text{int}}$ (see Section 2.2), and $\mathbf{F}^{\text{prev}}$ (see Section 2.3) cancel each other out to yield

$$\mathbf{F}\left(P_i\right) = \mathbf{0} \quad \forall \quad P_i \in \mathcal{T}. \tag{2.3}$$

The equilibrium configuration must be obtained numerically, as detailed in Section 2.4.

While the external and preview forces $\mathbf{F}_i^{\text{ext}}$, $\mathbf{F}_i^{\text{prev}}$ at the nodes P_i depend on the trajectory's representation in the augmented configuration space

$$\mathbb{C}_T \quad = \quad \langle x, y, \psi, t \rangle \,, \tag{2.4}$$

because they depend also on the orientation ψ of the ego vehicle relative to the environment. The internal forces $\mathbf{F}_i^{\mathrm{int}}$ are defined with regards to the trajectory's representation in

$$\mathbb{X}_T = \langle x, y, \psi, v, a, a_y, \dot{a}, \dot{a}_y, t \rangle \,, \tag{2.5}$$

which shall be called augmented phase space. $\mathbb{X}_T$ also contains all derivatives of the dimensions in $\mathbb{W}_T$, $\mathbb{C}_T$ that are regarded in optimization criteria or restrictions for the trajectory. The trajectory optimization is performed in $\mathbb{W}_T$ instead of $\mathbb{X}_T$ to avoid the high number of dimensions and thereby reduce the computational effort.

Due to the discrete nature of representation, the relations between the augmented workspace $\mathbb{W}_T$, the augmented configuration space $\mathbb{C}_T$, and the augmented phase space $\mathbb{X}_T$ along the trajectory $\mathcal{T}$ are established via finite differences. For each discrete node $P_i = [x_i, y_i, t_i]^\mathsf{T}$ they are defined as

$$\psi_i := \arctan \frac{\Delta y_i}{\Delta x_i}, \tag{2.6}$$

$$v_i := \frac{\Delta s_i}{\Delta t_i}, \tag{2.7}$$

$$a_i := \frac{v_i - v_{i-1}}{\frac{1}{2}(t_i - t_{i-2})}, \tag{2.8}$$

$$a_{y,i} := \frac{v_{y,i} - v_{y,i-1}}{\frac{1}{2}(t_i - t_{i-2})}, \tag{2.9}$$

$$\dot{a}_i := \frac{a_{i+1} - a_{i-1}}{\frac{1}{3}(t_{i+1} + t_{i-2})}, \tag{2.10}$$

$$\dot{a}_{y,i} := \frac{a_{y,i+1} - a_{y,i-1}}{\frac{1}{3}(t_{i+1} + t_{i-2})}, \tag{2.11}$$

where

$$\Delta x_i = x_i - x_{i-1}, \qquad \Delta y_i = y_i - y_{i-1},$$

$$\Delta s_i = \sqrt{{\Delta x_i}^2 + {\Delta y_i}^2}, \qquad v_{y,i} = \frac{\Delta y_i}{\Delta t_i},$$

$$\Delta t_i = t_i - t_{i-1}.$$

All components above are assumed to be given with regards to the same cartesian reference frame. Using these relations, the ego vehicle is assumed to be aligned with the trajectory, neglecting a sideslip angle.

2.1. External Force Field

The external force field $\mathbf{F}^{\text{ext}}(\mathcal{R}, \mathcal{O})$ depends on the course of the road $\mathcal{R}$ and the obstacles $\mathcal{O}$. It is derived as the negative gradient of a potential field hazard map V^{ext} that represents the environment, analogously to existing potential field motion planning approaches, see Chapter 1. The external force $\mathbf{F}_i^{\text{ext}}$ that results at a certain node $P_i = [x_i, y_i, t_i]$ is therefore given by

$$\mathbf{F}_i^{\text{ext}} \quad := \quad -\nabla V_i^{\text{ext}} := \left[\ -\frac{\partial V^{\text{ext}}}{\partial x_i},\quad -\frac{\partial V^{\text{ext}}}{\partial y_i},\quad -\frac{\partial V^{\text{ext}}}{\partial t_i}\ \right]^{\mathsf{T}} \tag{2.12}$$

and has components in spatial (x, y) and temporal (t) directions.

As illustrated in Figure 2.3, the total external potential V^{ext} results as the sum of potentials from the road and obstacle representations $V^{\mathcal{R}}$ and $V^{\mathcal{O}}$,

$$V^{\text{ext}} \quad = \quad V^{\mathcal{R}} + V^{\mathcal{O}}. \tag{2.13}$$

The calculation of the road potential $V^{\mathcal{R}}$ is detailed in Section 2.1.1, while Section 2.1.2 introduces the definition of the obstacle potential $V^{\mathcal{O}}$.

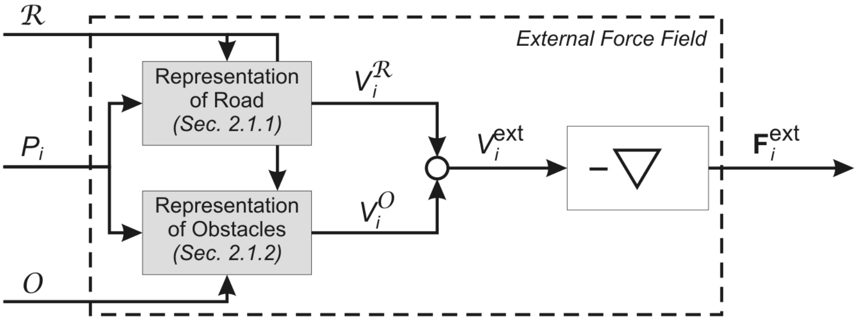

$\mathcal{R}$
External Force Field
Representation of Road (Sec. 2.1.1)
$V_i^{\mathcal{R}}$
P_i
V_i^{ext}
$-\nabla$
$\mathbf{F}_i^{\text{ext}}$
Representation of Obstacles (Sec. 2.1.2)
V_i^{O}
O

Figure 2.3.: Blockdiagram external force field
The road $\mathcal{R}$ and the obstacles $\mathcal{O}_j$ are mapped into an external potential field V^{ext}. The resulting external force $\mathbf{F}_i^{\text{ext}}$ at P_i is defined by the negative gradient ∇.

For the mathematical description of the environment, the following reference frames are used, see Figure 2.4:

$\underline{\text{E}}$:	$\{E^*, {}_E\mathbf{e}_x, {}_E\mathbf{e}_y, {}_E\mathbf{e}_t\}$	**E**arth fixed reference frame
$\underline{\text{R}}$:	$\{R^*, {}_R\mathbf{e}_x, {}_R\mathbf{e}_y, {}_R\mathbf{e}_t\}$	**R**oad fixed reference frame fixed at planning time t_0
$\underline{\tilde{R}\,(\tilde{s})}$:	$\left\{\tilde{R}^*, {}_{\tilde{R}}\mathbf{e}_x, {}_{\tilde{R}}\mathbf{e}_y, {}_{\tilde{R}}\mathbf{e}_t\right\}$	**R**oad centerline reference frame, shifted with the arclength $\tilde{s}$ along the centerline
$\underline{\tilde{R}^i}$:	$\left\{\tilde{R}^{i*}, {}_{\tilde{R}^i}\mathbf{e}_x, {}_{\tilde{R}^i}\mathbf{e}_y, {}_{\tilde{R}^i}\mathbf{e}_t\right\}$	**R**oad centerline reference frame for point P_i
$\underline{\hat{R}^k}$:	$\left\{\hat{R}^{k*}, {}_{\hat{R}^k}\mathbf{e}_x, {}_{\hat{R}^k}\mathbf{e}_y, {}_{\hat{R}^k}\mathbf{e}_t\right\}$	**R**oad segment fixed reference frame of segment $\hat{R}^k$
$\underline{\text{V}}$:	$\{V^*, {}_V\mathbf{e}_x, {}_V\mathbf{e}_y, {}_V\mathbf{e}_t\}$	**V**ehicle fixed reference frame of ego-vehicle
$\underline{O_j}$:	$\left\{O_j^*, {}_{O_j}\mathbf{e}_x, {}_{O_j}\mathbf{e}_y, {}_{O_j}\mathbf{e}_t\right\}$	**O**bstacle fixed reference frame of obstacle O_j

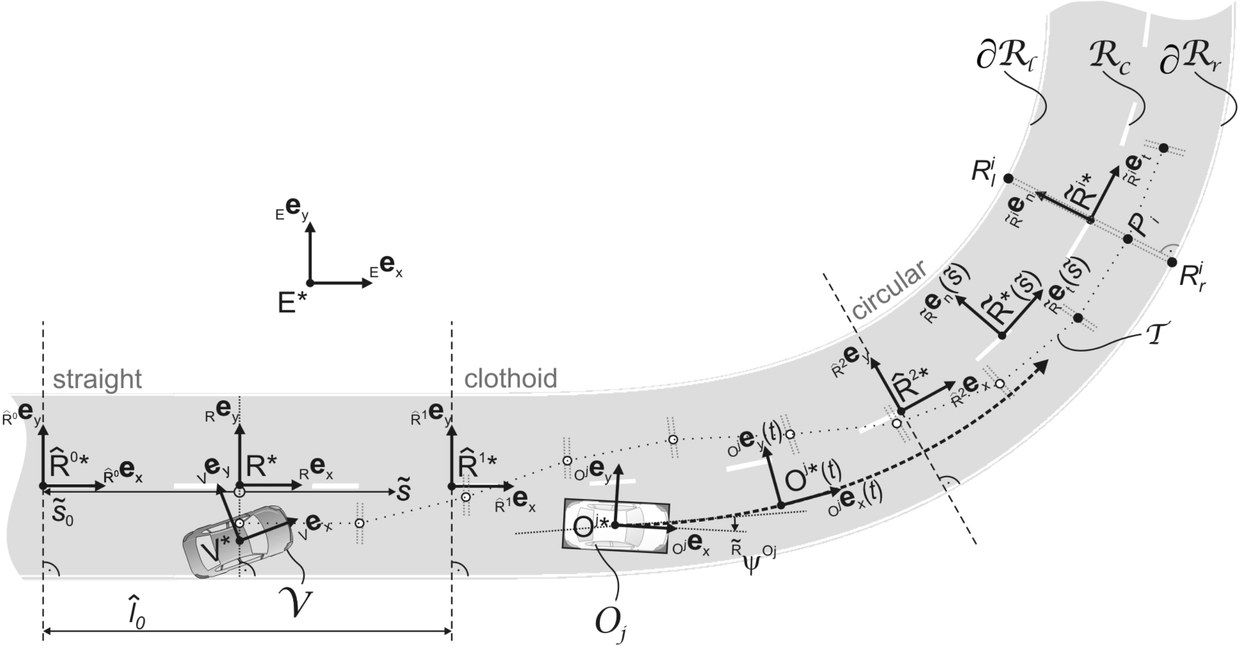


Figure 2.4.: Street representation and relevant reference frames
Besides the global earth fixed reference frame $\underline{\text{E}}$, there are object fixed reference frames for the ego-vehicle as well as for all obstacles, and reference frames at the road centerline, placed at the beginning of each segment and for each node P_i. The usage of different reference frames simplifies the description of motions and allows the reduction of degrees of freedom in search for the equilibrium solution.

Note that for all reference frames the third spatial dimension (z) is replaced by the temporal dimension (t). The relation between the different dimensions for any of the above reference frames is defined by $\frac{1}{\mathrm{m}}\mathbf{e}_x \times \frac{1}{\mathrm{m}}\mathbf{e}_x = \frac{1}{\mathrm{s}}\mathbf{e}_t$. Unless stated otherwise, position vectors are given with respect to $\underline{\text{R}}$.

The road centerline reference frame $\underrightarrow{\uparrow\tilde{R}(\tilde{s})}$ is defined depending on the arc length $\tilde{s}$ along the centerline, as illustrated in Figure 2.4. Its origin lies on the centerline $\mathcal{R}_c$ of the road, its ${}_{\tilde{R}}\mathbf{e}_x$-axis is tangent to the road and the ${}_{\tilde{R}}\mathbf{e}_y$-axis points to the left border of the road,

$$
{}_{\tilde{R}}\mathbf{e}_x := \frac{\mathbf{r}^{\tilde{R}^{*\prime}}}{\left\|\mathbf{r}^{\tilde{R}^{*\prime}}\right\|} = \frac{{}_{R}x^{\tilde{R}^{*\prime}}\,{}_{R}\mathbf{e}_x + {}_{R}y^{\tilde{R}^{*\prime}}\,{}_{R}\mathbf{e}_y}{\sqrt{\left({}_{R}x^{\tilde{R}^{*\prime}}\right)^2 + \left({}_{R}y^{\tilde{R}^{*\prime}}\right)^2}}, \qquad {}_{\tilde{R}}\mathbf{e}_y := \frac{-{}_{R}y^{\tilde{R}^{*\prime}}\,{}_{R}\mathbf{e}_x + {}_{R}x^{\tilde{R}^{*\prime}}\,{}_{R}\mathbf{e}_y}{\sqrt{\left({}_{R}x^{\tilde{R}^{*\prime}}\right)^2 + \left({}_{R}y^{\tilde{R}^{*\prime}}\right)^2}}, \tag{2.14}
$$

where $(\cdot)' = \frac{d(\cdot)}{d\tilde{s}}$ abbreviates the derivative with respect to the arc length $\tilde{s}$ along the centerline of the road.

For each node P_i a reference frame $\underrightarrow{\uparrow\tilde{R}^i} = \underrightarrow{\uparrow\tilde{R}(\tilde{s}_i)}$ is defined on the centerline of the road closest to P_i such that

$$
\tilde{s}_i \;=\; \underset{\tilde{s}_i}{\operatorname{argmin}} \left\| \mathbf{r}^{\tilde{R}^*(\tilde{s}_i),P_i} \right\|. \tag{2.15}
$$

This means that by definition the position vector of P_i in $\underrightarrow{\uparrow\tilde{R}^i}$ only has a lateral component and

$$
{}_{\tilde{R}^i}\mathbf{r}_x^{\tilde{R}^{i*},P_i} := 0. \tag{2.16}
$$

At each planning instant t_0, the centerline $\mathcal{R}_c$ is represented in the road fixed reference frame $\underrightarrow{\uparrow\mathrm{R}}$. $\underrightarrow{\uparrow\mathrm{R}}$ is identical to the reference frame $\underrightarrow{\uparrow\tilde{R}(0)}$ that is closest to the current position of the center of gravity V^* of the ego-vehicle $\mathcal{V}$. Therefore, the longitudinal position of the ego-vehicle at $t = t_0$ becomes

$$
{}_{R}\mathbf{r}_x^{R*,V*} \;=\; 0. \tag{2.17}
$$

At the beginning of each road segment $\hat{R}^k$ there is a segment fixed reference frame $\underrightarrow{\uparrow\hat{R}^k}$. It is identical to the road centerline reference frame $\underrightarrow{\uparrow\tilde{R}(\tilde{s}_k)}$ at that position. Finally, also the ego vehicle and all obstacles have their own reference frames $\underrightarrow{\uparrow\mathrm{V}}$, $\underrightarrow{\uparrow O_j}$.

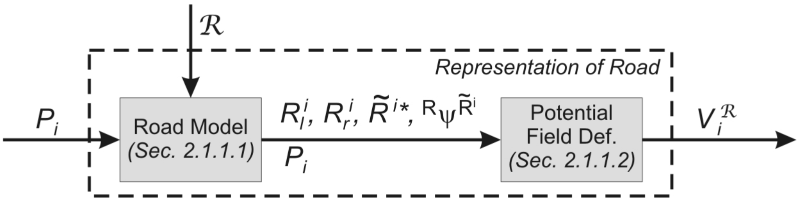


Figure 2.5.: Representation of road

For every point P_i on the street, the road model calculates the closest point on the centerline of the road $\tilde{R}^{i*}$, the corresponding points R_l^i and R_r^i, and the road orientation ${}^{R}\psi^{\tilde{R}^i}$ of the tangent at $\tilde{R}^{i*}$. From this information, the road potential $V_i^{\mathcal{R}}$ is calculated.

2.1.1. Representation of Road

The street is modeled in order to map its course into a potential field hazard map, as shown in Figure 2.5. For every point P_i on the street, the road model calculates the closest point on the centerline of the road, $\tilde{R}^{i*}$, the corresponding points R_l^i and R_r^i, and the road orientation ${}^{R}\psi^{\tilde{R}^i}$ of the tangent at $\tilde{R}^{i*}$ as illustrated in Figure 2.4. Using this information, the road potential $V_i^{\mathcal{R}}$ at P_i is calculated.

For the sake of the trajectory optimization algorithm, any kind of road model that produces the indicated information is feasible. Following in Section 2.1.1.1, this degree of freedom is explored and a suitable road model is discussed. The definition of the road potential is given in Section 2.1.1.2. In addition, this model provides a basis for the extrapolation of obstacle movements, see Section 2.1.2.1.

2.1.1.1. Road Model

The construction of roads in Germany is, as in most countries, governed by national guidelines as documented in “Richtlinien für die Anlage von Straßen” (RAS), [FGSV, 1999]. Among others, Germany’s national guidelines demand a continuous curvature for the course of the road. Therefore, straight pieces and circular arcs need to be connected by transition segments called clothoids, see Figure 2.4.

Multifarious approaches for modeling the road for motion planning or driver assistance systems can be found. In most cases, first information about the course of the road in the vicinity of the ego-vehicle is collected by the use of various sensors such as RADAR [Polychronopoulos et al., 2004], laser RADAR (LADAR) [Cremean and Murray, 2006; Wijesoma et al., 2004], video [Bellino et al., 2004; Koller et al., 1993], laser scanners [Sparbert et al., 2001], or combinations thereof [Cramer et al., 2004; Goyat et al., 2009]. Then a

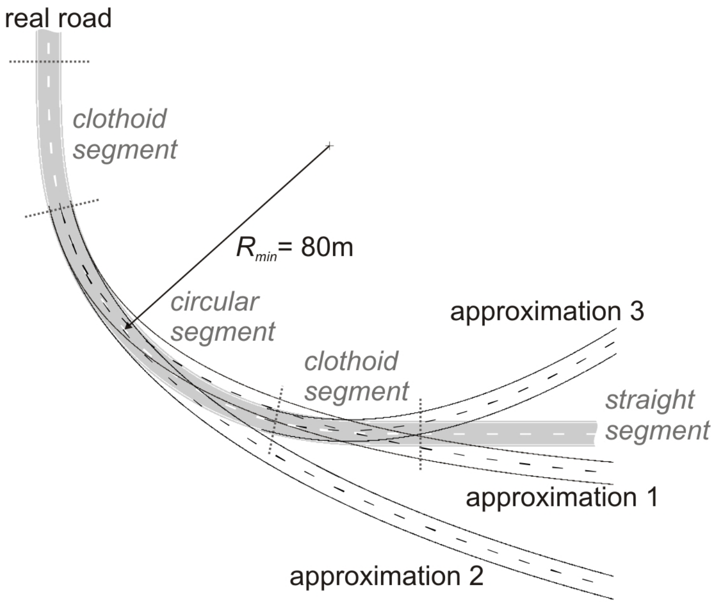


Figure 2.6.: Road approximation with cubic polynomials

mathematical model for the road is devised and the acquired data is fitted to this model. An overview of early approaches can be found in [Kluge and Thorpe, 1989].

With regards to the mathematical road model, the majority of approaches use flat (2D) road representations. In some publications these models have been extended to account for 3D shapes of roads, as for example in [Dickmanns and Mysliwetz, 1992; Nedevschi et al., 2004]. It is very common to use continuous models for the road centerline that give local approximations in the vicinity of the ego-vehicle. These models vary from linear [Lee, 2002] or circular [Ma et al., 2000] to parabolic [Kreucher and Lakshmanan, 1999] and cubic polynomial representations [Southall and Taylor, 2001]. The advantage of these rather simple continuous geometric models lies in the need for only very few parameters that can easily be estimated from sensor data.

However, even cubic models are only valid within a certain range that depends on the curvature of the road. Figure 2.6 shows a curve with a minimum radius of 80 m that obeys the national road construction guidelines for a velocity of 50 km/h [FGSV, 1999]. It consists of a concatenation of straight, clothoid and circular segments. The Figure shows three cubic approximations that start at different locations along the road. Approximation 1 starts right before the circular segment is reached, approximation 2 starts at the beginning of the circular segment and approximation 3 starts at the end of the circular segment. Figure 2.6 shows that the cubic polynomial approximation may exhibit large lateral offsets: in this example more than a lane width of 3.5 m after an arc length of less than 60 m.

In an attempt to enlarge the validity range of a road model, [Khosla, 2002] discusses the concatenation of two cubic polynomials as approximation for clothoids.

Following this train of thought, piecewise or segmented representations have the advantage that their accuracy can be chosen arbitrarily by their discretization. The arising models have no inherent range limitations. Each segment could for example be represented by a linear [Kwon and Lee, 2002] or circular [Kluge and Thorpe, 1995] function. The disadvantage of such a representation lies in the need for an increasingly high number of parameters that need to be estimated from sensor data. For an accurate representation of the road, an infinite number of linear or circular segments would be necessary. As a general trade-off, the simpler the segment model the more segments are necessary. Therefore, the used model of the road heavily depends on the purpose which defines the necessary range and accuracy of the model and the available (sensor) data to estimate the model parameters.

For the course of this work, the centerline $\mathcal{R}_c$ of the road is modeled segment-wise as given in the national construction guidelines by linear, circular and clothoid segments. The altitude information of the road is neglected and the road is modeled two dimensionally as projection to the x-y-plane. This yields an accurate model that is valid for arbitrary planning distances. The road model, i.e. the (partially redundant) parameters for each segment, $\hat{l}_i$, $\kappa_{k,0}$, κ'_k, $({}^{R}\psi^{\hat{R}^k}$, $\mathbf{r}^{R^*,\hat{R}^{k*}})$, are stored in $\mathcal{R}$ and are assumed to be known at least within the planning distance from a digital map and/or sensor data. For each P_i, the given road model is applied to recover R^i_l, R^i_r, $\tilde{R}^{i*}$, and ${}^{R}\psi^{\tilde{R}^i}$. The blockdiagram of the apllication of the road model is illustrated in Figure 2.7.

The road centerline model returns the position vector $\mathbf{r}^{R*,\tilde{R}^*(s_i)}$ and orientation ${}^{R}\psi^{\tilde{R}(s_i)}$ of the road centerline $\mathcal{R}_c$ relative to the road-fixed reference frame R for a given arc length $\tilde{s}_i$ (compare also Figure 2.4).

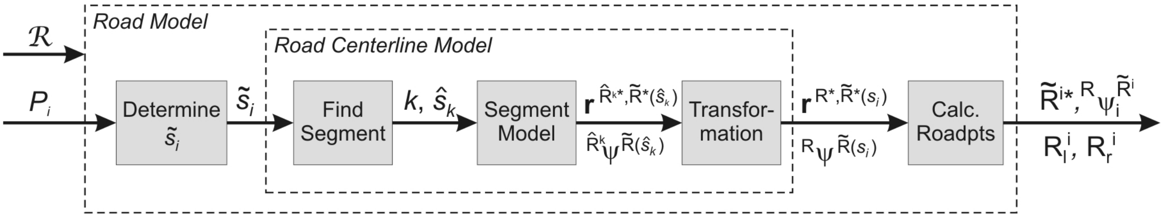

Road Model
Road Centerline Model
$\mathcal{R}$
P_i
Determine $\tilde{s}_i$
$\tilde{s}_i$
Find Segment
k, $\hat{s}_k$
Segment Model
$\mathbf{r}^{\hat{R}^{k*},\tilde{R}^*(\hat{s}_k)}$
${}^{\hat{R}^k}\psi^{\tilde{R}(\hat{s}_k)}$
Transfor-mation
$\mathbf{r}^{R^*,\tilde{R}^*(s_i)}$
${}^{R}\psi^{\tilde{R}(s_i)}$
Calc. Roadpts
$\tilde{R}^{i*}$, ${}^{R}\psi_i^{\tilde{R}^i}$
R_l^i, R_r^i

Figure 2.7.: Road representation

For every point P_i on the street, the road model calculates the closest point on the centerline of the road $\tilde{R}^{i*}$, the corresponding points R^i_l and R^i_r on the road sides, the road orientation ${}^{R}\psi^{\tilde{R}^i}$ of the tangent at $\tilde{R}^{i*}$, and the arc length $\tilde{s}_i$ along the centerline from R^* to $\tilde{R}^{i*}$

Calculate Roadpoints Using the output of the road centerline model, the correlating roadside points R_l^i and R_r^i on the left and right road side, are easily determined. The right and left border of the road $\partial\mathcal{R}_r, \partial\mathcal{R}_l$ are given by

$$\partial\mathcal{R}_{l/r} := \mathcal{R}_c \pm \frac{b}{2}\,{}_{\tilde{R}}\mathbf{e}_y, \tag{2.18}$$

where b denotes the width of the road. Therefore, the position of the roadside points is given by

$$\mathbf{r}^{R^*,R^i_{l/r}} = \mathbf{r}^{R^*,\tilde{R}^{i*}} \pm \frac{b}{2}\,{}_{\tilde{R}^i}\mathbf{e}_y, \tag{2.19}$$

where, as defined earlier, $\tilde{R}^{i*} = \tilde{R}^*(\tilde{s}_i)$ and ${}_{\tilde{R}^i}\mathbf{e}_y = -\,{}_R\mathbf{e}_x \sin {}^R\psi^{\tilde{R}^i} + {}_R\mathbf{e}_y \cos {}^R\psi^{\tilde{R}^i}$ with ${}^R\psi^{\tilde{R}^i} = {}^R\psi^{\tilde{R}(s_i)}$. Thus, all outputs $\tilde{R}^{i*}$, ${}^R\psi^{\tilde{R}^i}$, R_l^i, and R_r^i are known.

Determine $\tilde{\mathbf{s}}_\mathbf{i}$ In case the arc length $\tilde{s}_i$ is unknown for a certain point P_i, it must be determined in order to use the road centerline model as seen in Figure 2.7, but as will be shown, hardly ever an extensive search is necessary.

The search for $\tilde{s}_i$ for the nodes P_i of the planned trajectory can be divided into three different cases. In the first case, when no prior information about $\tilde{s}_i$ exists, the whole map must be searched for $\tilde{s}_i$ to fulfill Equation 2.15 such that the point $\tilde{R}(\tilde{s}_i)$ is the one on the road center line $\mathcal{R}_c$ that is closest to P_i. The best choice for the search algorithm depends heavily on the characteristics of the map, as for example its size. In most cases it makes sense to search first the road segment in which P_i lies and then inside this segment the closest point on the centerline.

The search for the relevant road segment can be enhanced by storing the segments in specialized data structures that allow time efficient retrieval of the closest element. The fastest search inside a segment again depends on its characteristics: While in a linear segment the arc length within the segment can be directly computed, in other types of segments e.g. a recursive search of segment parts could be implemented.

However, this first case where no prior information for $\tilde{s}_i$ exists is extremely rare and is usually only relevant for the first node P_0 of the planned trajectory for the very first planning instance $K = 0$.

In the second case, we are also searching for $\tilde{s}_0$ correlating to the first node P_0 of the planned trajectory. However, in difference to the first case, a trajectory had been planned already in a previous planning instance $K-1$. Knowing $\tilde{s}_0$ for the previous first node P_0^{K-1} and the approximate distance the ego vehicle has traveled since then, a very good initial guess for $\tilde{s}_0$ already exists to speed up the search.

In the third case, the search for $\tilde{s}_i$ is even simpler. In this case, we are searching for the arc length $\tilde{s}_i$ correlating to P_i with $i > 0$. Since $\tilde{s}_{i-1}$ can be assumed to be known at this point, as is the arc length $\Delta\tilde{s}_{i-1,i}$ between the nodes, $\tilde{s}_i$ is given as $\tilde{s}_i = \tilde{s}_{i-1} + \Delta\tilde{s}_{i-1,i}$.

The position $\mathbf{r}^{R^*,\tilde{R}^*}(\tilde{s})$ and orientation ${}^{R}\psi^{\tilde{R}}(\tilde{s})$ of the road centerline $\mathcal{R}_c$ are defined depending on the type of road segment. As illustrated in Figure 2.7, the road centerline model consists of three parts: First, a part to find the relevant segment, second, the segment model of this segment, and third, a coordinate transformation.

Find Segment In each segment $\hat{R}^k$ only the segment arc length

$$\hat{s}_k = (\tilde{s} - \tilde{s}_k) \tag{2.20}$$

along the centerline of the segment is regarded, where

$$\tilde{s}_k = \tilde{s}_0 + \sum_{i=0}^{k-1} \hat{l}_i \tag{2.21}$$

denotes the arc length along the road centerline from R^* to $\hat{R}^{k*}$. Therein, $\hat{l}_i$ represents the length of the centerline in each segment $\hat{R}^i$ and $\tilde{s}_0$ designates the arc length along the centerline from R^* to the beginning R^{0*} of the current road segment, see Figure 2.4. It is important to note that $\tilde{s}_0 \leq 0$. The segment $\hat{R}^k$ is determined by

$$k = \underset{k}{\operatorname{argmin}}(\tilde{s} - \tilde{s}_k \,|\tilde{s} - \tilde{s}_k > 0). \tag{2.22}$$

Transformation For each segment $\hat{R}^k$, the position $\mathbf{r}^{R^*,\tilde{R}^*}(\tilde{s})$ and orientation ${}^{R}\psi^{\tilde{R}}(\tilde{s})$ of the road centerline $\mathcal{R}_c$ are first computed in a road segment fixed reference frame $\underrightarrow{\uparrow\hat{R}^k} = \underrightarrow{\uparrow\tilde{R}(\tilde{s}_k)}$, that is fixed on the centerline $\mathcal{R}_c$ of the road at the beginning of each segment, and then transformed into $\underrightarrow{\uparrow\mathrm{R}}$ by applying

$$
\begin{aligned}
{}_{R}\mathbf{r}^{R^*,\tilde{R}^*}(\tilde{s}) &= {}_{R}\mathbf{r}^{R^*,\hat{R}^{k*}} + \mathbf{C}^{\hat{R}^{k*},R}\; {}_{\hat{R}^k}\mathbf{r}^{\hat{R}^{k*},\tilde{R}^*}(\hat{s}_k)\,, && (2.23)\\
{}^{R}\psi^{\tilde{R}}(\tilde{s}) &= {}^{R}\psi^{\hat{R}^k} + {}^{\hat{R}^k}\psi^{\tilde{R}}(\hat{s}_k)\,. && (2.24)
\end{aligned}
$$

The transformation matrix in Equation 2.23 reads

$$\mathbf{C}^{\hat{R}^{k*},R} = \begin{bmatrix} \cos{}^{R}\psi^{\hat{R}^k} & \sin{}^{R}\psi^{\hat{R}^k} \\ -\sin{}^{R}\psi^{\hat{R}^k} & \cos{}^{R}\psi^{\hat{R}^k} \end{bmatrix}, \tag{2.25}$$

where the start orientation ${}^{R}\psi^{\hat{R}^k}$ of each road segment $\hat{R}^k$ and the position vectors ${}_{R}\mathbf{r}^{R^*,\hat{R}^{k*}}$ of the segment fixed reference frames are known from a digital map and/or sensor data, as mentioned before. Relative to one another they are given as

$$\begin{aligned} {}^{R}\psi^{\hat{R}^k} &= {}^{R}\psi^{\hat{R}^{k-1}} + {}^{\hat{R}^{k-1}}\psi^{\tilde{R}}\left(\hat{l}_{k-1}\right), & (2.26) \\ {}_{R}\mathbf{r}^{R^*,\hat{R}^{k*}}\left(\tilde{s}_k\right) &= {}_{R}\mathbf{r}^{R^*,\hat{R}^{k-1*}} + \mathbf{C}^{\hat{R}^{k-1*},R}\,{}_{\hat{R}^{k-1}}\mathbf{r}^{\hat{R}^{k-1*},\tilde{R}^*}\left(\hat{l}_{k-1}\right). & (2.27) \end{aligned}$$

Segment Model for Straight Street Segments As illustrated in Figure 2.8, the centerline of straight segments is given by

$${}_{\hat{R}^k}\mathbf{r}^{\hat{R}^k,\tilde{R}^*}\left(\hat{s}_k\right) = \begin{bmatrix} \hat{s}_k \\ 0 \end{bmatrix}. \tag{2.28}$$

The road orientation remains constant at

$${}^{R^k}\psi^{\tilde{R}}\left(\hat{s}_k\right) = 0. \tag{2.29}$$

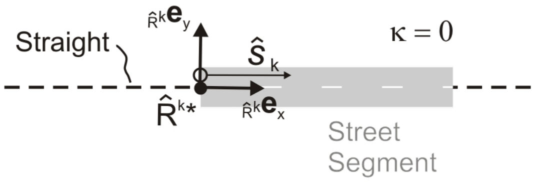
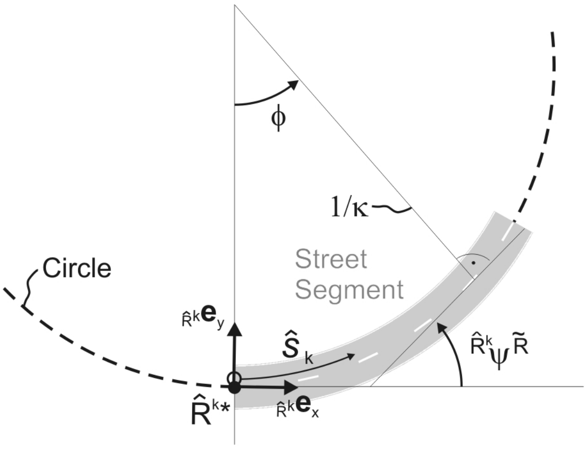


Figure 2.8.: Straight road segment



ϕ
$1/\kappa$
Circle
Street Segment
${}_{\hat{R}^k}\mathbf{e}_y$
$\hat{s}_k$
${}^{\hat{R}^k}\psi^{\tilde{R}}$
$\hat{R}^{k*}$
${}_{\hat{R}^k}\mathbf{e}_x$

Figure 2.9.: Circular road segment

Segment Model for Circular Street Segments Circular segments, as shown in Figure 2.9, are represented by

$$
{}_{\hat{R}^k}\mathbf{r}^{\hat{R}^k,\tilde{R}^*}\left(\hat{s}_k\right) = \frac{1}{\kappa_k}\begin{bmatrix} \sin\left(\phi\left(\hat{s}_k\right)\right) \\ 1-\cos\left(\phi\left(\hat{s}_k\right)\right) \end{bmatrix}, \tag{2.30}
$$

$$
{}^{R^k}\psi^{\tilde{R}}\left(\hat{s}_k\right) = \phi\left(\hat{s}_k\right) = \hat{s}_k\kappa_k, \tag{2.31}
$$

where κ_k denotes the curvature of the road segment.

Segment Model for Clothoid Street Segments For clothoid street segments, as illustrated in Figure 2.10, the curvature κ varies linearly with κ' over the arc length s

$$
\kappa\left(s\right) = \kappa' s, \tag{2.32}
$$

where κ' denotes the change in curvature which remains constant at $\kappa' = \kappa'_k$ within each clothoid road segment $\hat{R}^k$. In the road construction guidelines [FGSV, 1999], further restrictions on κ' exist. However, they shall not be regarded at this point, since they do not limit the validity of this road model. The start curvature $\kappa_{k,0}$ is the same as the end curvature of the previous segment, i.e. the curvature of two consecutive segments (for example a circular and a clothoid segment) is identical where they are connected.

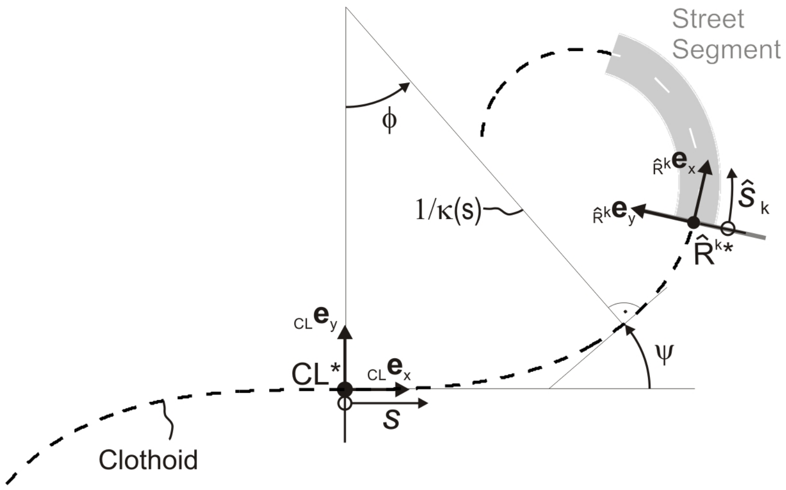


Figure 2.10.: Clothoid road segment

As detailed in B.1, clothoids are described as Fresnels integrals and cannot be solved in closed form, [Bronstein and Semendjajew, 1991]. However, they can be approximated with arbitrary accuracy using series expansions for the sine and cosine functions in the integrands, see for example [Wang et al., 2001].

This results in

$$ {}_{\hat{R}^k}\mathbf{r}^{\hat{R}^{k*},\tilde{R}^*}\left(\hat{s}_k\right) = \mathbf{C}^{CL,\hat{R}^k}\begin{bmatrix} \sum_{n=0}^{\infty}\frac{(-1)^n\left(\frac{1}{2}\kappa_k'\right)^{2n}}{(2n)!(4n+1)}\left(\left(\hat{s}_k+\frac{\kappa_{k,0}}{\kappa_k'}\right)^{4n+1}-\left(\frac{\kappa_{k,0}}{\kappa_k'}\right)^{4n+1}\right) \\ \sum_{n=0}^{\infty}\frac{(-1)^n\left(\frac{1}{2}\kappa_k'\right)^{2n+1}}{(2n+1)!(4n+3)}\left(\left(\hat{s}_k+\frac{\kappa_{k,0}}{\kappa_k'}\right)^{4n+3}-\left(\frac{\kappa_{k,0}}{\kappa_k'}\right)^{4n+3}\right) \end{bmatrix}. \tag{2.33} $$

$\mathbf{C}^{CL,\hat{R}^k}$ denotes the transformation matrix between $\uparrow\!\underrightarrow{\mathrm{CL}}$ and $\uparrow\!\underrightarrow{\hat{R}^k}$

$$ \mathbf{C}^{CL,\hat{R}^k} = \begin{bmatrix} \cos{}^{CL}\psi^{\hat{R}^k} & \sin{}^{CL}\psi^{\hat{R}^k} \\ -\sin{}^{CL}\psi^{\hat{R}^k} & \cos{}^{CL}\psi^{\hat{R}^k} \end{bmatrix}, \tag{2.34} $$

where ${}^{CL}\psi^{\hat{R}^k}$ represents the rotation angle between the two reference frames

$$ {}^{CL}\psi^{\hat{R}^k} = \frac{\kappa_{k,0}^2}{2\kappa_k'}. \tag{2.35} $$

The orientation of the road centerline of a clothoid segment with regards to the segment fixed reference frame is given (see B.1) by

$$ {}^{\hat{R}^k}\psi^{\tilde{R}}\left(\hat{s}_k\right) = \kappa_{k,0}\hat{s}_k + \frac{1}{2}\kappa'_k\hat{s}_k^2. \tag{2.36}$$

2.1.1.2. Definition of Road Potential

For the trajectory optimization the road is mapped into an external road potential $V^{\mathcal{R}}$, as mentioned before. This potential shall have the properties that it is well defined and twice continuously differentiable for any point on the road. In addition, both the potential and its first derivative shall go to infinity at the borders of the road. This property is demanded to ensure that the equilibrium always lies within the road borders even when obstacles are present. Further, there should be a unique minimum across the width of the road that can be placed to represent the desired path in the absence of obstacles, see Section 2.6. Since the road does not vary over time, the potential field shall not vary in time-direction.

Following the design considerations above and a previous successful approach for path planning, see [Brandt, 2007], the road potential is chosen as the sum of two logarithmic functions for each road side

$$V^{\mathcal{R}} := V^{\partial\mathcal{R}_r} + V^{\partial\mathcal{R}_l}, \tag{2.37}$$

$$V^{\partial\mathcal{R}_q} := -k^{\partial\mathcal{R}_q}\ln\left(d^{\partial R_q}\right), \qquad q \in \{r, l\} \tag{2.38}$$

where $d^{\partial R_q}$ denotes the distance to the road side $\partial\mathcal{R}_q$. The distance $d^{\partial R_q}$ does not depend on the time, since the roadsides are time-invariant. Further it is approximated to be independent of the ego-vehicle's orientation ψ to reduce computational complexity, since the planned motion is mostly longitudinal to the course of the road. This is especially true where the ego vehicle comes close to the road borders. Therefore, the final result should not differ significantly.

The road potential for a given road $\mathcal{R}$ therefore only depends on the spatial position,

$$V_i^{\mathcal{R}} = f\left(x_i, y_i\right) = f\left(\mathbf{r}_i\right). \tag{2.39}$$

The choice of the constants $k^{\partial\mathcal{R}_r}$ and $k^{\partial\mathcal{R}_l}$ is discussed in Section 2.6.

2.1.2. Representation of Obstacles

All obstacles are mapped to an obstacle potential $V^{\mathcal{O}}$ that can be evaluated for any point P. In principle, any obstacle geometry can be regarded, but certain approximations are advantageous because they simplify the calculations. Available sensor data would have to be fitted to these approximations to create the obstacle representations for the trajectory optimization. Easiest in terms of distance computations would be circles, however, for automotive vehicles, a circle is a rather poor approximation and an encompassing circle would be far too conservative for little available free space: Two vehicles on neighboring lanes could not pass each other anymore because they would seemingly collide if approximated by encompassing circles. Therefore, all obstacles are represented as rectangles. Different geometries are enlarged to rectangles.

The motion of all obstacles $\mathcal{O}_j, j = 1..M$ is extrapolated depending on their detected dynamic state at $t = t_0$ as illustrated in Figure 2.11 and detailed in Sections 2.1.2.1 and 2.1.2.2. Then the distances $d_i^{\mathcal{O}_j}$ from a point P_i to all obstacles $\mathcal{O}_j$ is determined, see Section 2.1.2.3, as well as the time to collision $\Delta t_i^{\mathcal{O}_j}$ (forward and backward in time) in case it exists for the position $[x_i, y_i]^\mathsf{T}$ of P_i, see Section 2.1.2.4. Finally, the obstacle potential $V^{\mathcal{O}}$ is computed depending on these spatial and temporal distances, see Section 2.1.2.5.

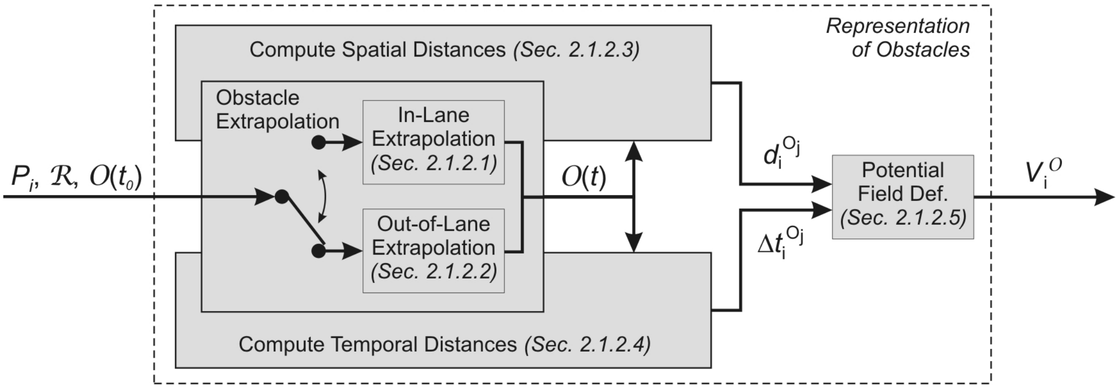

Representation of Obstacles
Compute Spatial Distances (Sec. 2.1.2.3)
Obstacle Extrapolation
In-Lane Extrapolation (Sec. 2.1.2.1)
Out-of-Lane Extrapolation (Sec. 2.1.2.2)
Compute Temporal Distances (Sec. 2.1.2.4)
P_i, $\mathcal{R}$, $O(t_0)$
$O(t)$
$d_i^{O_j}$
$\Delta t_i^{O_j}$
Potential Field Def. (Sec. 2.1.2.5)
V_i^O

Figure 2.11.: Blockdiagram Representation of Obstacles
The obstacle potential field depends on the spatial and temporal distances to all obstacles. In order to determine these distances for a certain point P_i, the motion of the obstacles is extrapolated either along the road (in-lane) or only due to the current dynamic state of the obstacle and ignoring the course of the road (out-of-lane).

The extrapolation of traffic objects and other obstacles is a complex matter and there have been many investigations. In many motion planning algorithms dynamic obstacles are extrapolated on linear trajectories with a constant velocity or acceleration [LaValle, 2006].

More sophisticated model-based approaches are used in object tracking applications [Hu et al., 2004]. Wang et al. incorporate traffic flow information in their model for in-lane vehicle motion prediction [Wang et al., 2005].

In addition, there exist behavior-based approaches that model the behavior of a human driver to predict the motion of the ego-vehicle [Toledo et al., 2007] and the surrounding traffic [Dagli and Reichardt, 2002]. Therein, probabilistic belief networks are used that operate with discrete motion concepts or maneuvers such as a lane-change [Dagli et al., 2003]. For an overview of behavior-based approaches see [Plöchl and Edelmann, 2007]. In case little previous knowledge about the behavior of obstacles exist, their motion patterns can also be learned over time, as presented in [Kruse et al., 1997; Vasquez et al., 2004].

The problem of the uncertainty of such predictions that are based on flawed or noisy sensor data can be addressed for example by Monte Carlo simulations, reachability set analyses [Althoff et al., 2008], or Markov chains [Rohrmüller et al., 2008].

Among others, the best extrapolation algorithm should be chosen depending on the available data. Within this virtual force field trajectory optimization, any (deterministic) extrapolation algorithm is feasible that predicts at least position and orientation of each obstacle for a given point of time to be able to compute the spatial and temporal distances $d_i^{O_j}$, $\Delta t_i^{O_j}$, see Figure 2.11. Data about obstacles can be obtained by the use of various sensors and data sources such as RADAR, LIDAR, laser scanners, cameras or car-to-x communication, see [Klein, 2001; Stiller et al., 2000].

For the course of this thesis, at the point of time of replanning $t = t_0$ the following data is assumed to be given for each obstacle O_j:

l^{O_j}, w^{O_j} $\hat{=}$ **l**ength and **w**idth of obstacle O_j
$\mathbf{r}^{R*,O_j^*}(t_0)$ $\hat{=}$ position of obstacle
${}^{R}\psi^{O_j}(t_0)$ $\hat{=}$ yaw angle of obstacle (if applicable, is included in geometric form)
${}^{R}v^{O_j}(t_0)$ $\hat{=}$ velocity of obstacle
${}^{R}a^{O_j}(t_0)$ $\hat{=}$ acceleration of obstacle
${}^{R}\omega^{O_j}(t_0)$ $\hat{=}$ yaw rate of obstacle

The obstacle's velocity ${}^{R}v^{O_j}(t)$ is extrapolated assuming that it maintains the constant acceleration ${}^{R}a^{O_j}(t_0)$ until either an assumed maximum velocity $v_{max}^{O_j}$ or a standstill is reached

$$\begin{aligned} {}^{R}v^{O_j}(t) &= \begin{cases} {}^{R}v_{lin}^{O_j}(t) & 0 < {}^{R}v_{lin}^{O_j}(t) < v_{max}^{O_j} \\ v_{max}^{O_j} \quad \text{if} & {}^{R}v_{lin}^{O_j}(t) > v_{max}^{O_j} \\ 0 & 0 > {}^{R}v_{lin}^{O_j}(t) \quad , \end{cases} \qquad (2.40) \\ {}^{R}v_{lin}^{O_j}(t) &= {}^{R}v^{O_j}(t_0) + {}^{R}a^{O_j}(t_0)(t - t_0). \qquad (2.41) \end{aligned}$$

The traveled arc length $s^{O_j}(t)$ is then given according to

$$s^{O_j}(t) = \int_{t_0}^{t} v^{O_j}(\tau)\, d\tau. \qquad (2.42)$$

For the course of this work, the extrapolation of an obstacle is divided into two categories, as illustrated in Figure 2.12. Depending on the orientation of the obstacle relative to the road centerline $\mathcal{R}_c$,

$${}^{\tilde{R}}\psi^{O_j} = {}^{R}\psi^{O_j}(t_0) - k\pi - {}^{R}\psi^{\tilde{R}}\left(\tilde{s}^{O_j}(t_0)\right), \qquad (2.43)$$

the obstacle is either assumed to follow the course of the road or to ignore it. k depends on the direction of motion relative to the ego-vehicle,

$$k = \begin{cases} 0 & \text{for parallel traffic} \\ 1 & \text{for oncoming traffic.} \end{cases} \qquad (2.44)$$

${}^{R}\psi^{\tilde{R}}\left(\tilde{s}^{O_j}(t_0)\right)$ is determined using Equations 2.24 and 2.26. The arc length $\tilde{s}^{O_j}(t_0)$ is defined as the arc length along the road centerline from R^* to the point on the centerline that is closest to the obstacle at $t = t_0$.

If ${}^{\tilde{R}}\psi^{O_j}$ is greater than a certain threshold ϵ_ψ, the obstacle is assumed to leave the road, otherwise it is assumed to follow it,

$$\mathbf{r}^{R^*,O_j^*}(t) = \begin{cases} \mathbf{r}_{\text{in-lane}}^{R^*,O_j^*}(t) & \text{if} \quad \left|{}^{\tilde{R}}\psi^{O_j}\right| \le \epsilon_\psi \\ \mathbf{r}_{\text{out-of-lane}}^{R^*,O_j^*}(t) & \text{otherwise.} \end{cases} \qquad (2.45)$$

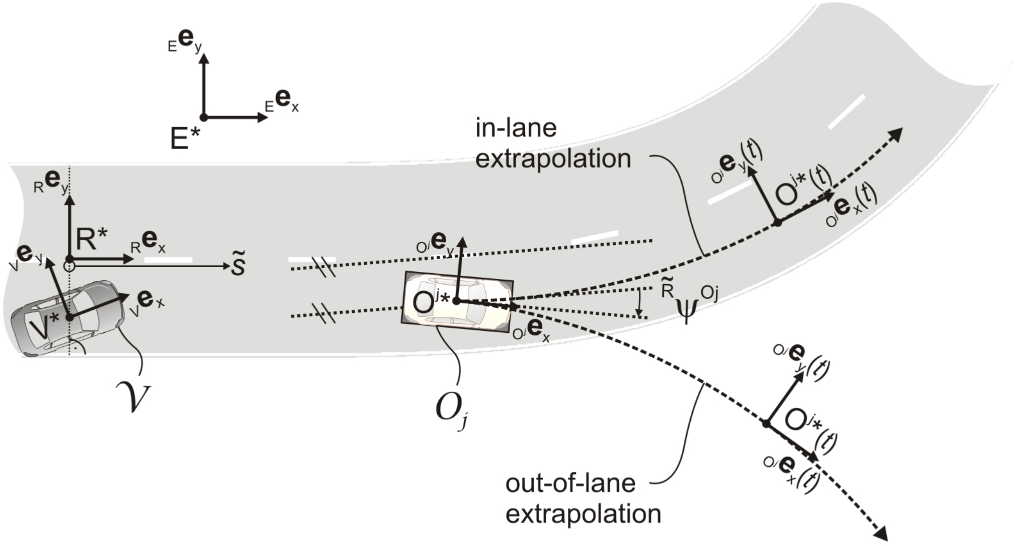


Figure 2.12.: Extrapolation of obstacles
Depending on the relative orientation to the road, the obstacle is either assumed to follow the course of the road (in-lane extrapolation) or to ignore the road (out-of-lane extrapolation).

2.1.2.1. In-Lane Extrapolation of Obstacles

If an obstacle follows the road, the lateral offset to the road centerline is assumed to remain constant such that

$$\mathbf{r}^{R^*,O_j^*}_{\text{in-lane}}(t) \quad = \quad \mathbf{r}^{R^*,\tilde{R}^*}\left(\tilde{s}^{O_j}(t)\right) + y^{O_j}_{\text{off}}\,{}_{\tilde{R}}\mathbf{e}_y\left(\tilde{s}^{O_j}(t)\right) \tag{2.46}$$

where $y^{O_j}_{\text{off}}$ equals the obstacle's current lateral offset to the center of the road

$$y^{O_j}_{\text{off}} \quad = \quad {}_{\tilde{R}}r_y^{\tilde{R}^*,O_j}\left(\tilde{s}^{O_j}(t_0)\right). \tag{2.47}$$

The orientation of the extrapolated obstacle is assumed to be equal to the tangent on the road centerline

$${}^{R}\psi^{O_j}\left(\tilde{s}^{O_j}\right) \quad = \quad {}^{R}\psi^{\tilde{R}}\left(\tilde{s}^{O_j}\right) + k\pi, \tag{2.48}$$

where k again represents the direction of motion relative to the ego-vehicle, see Equation 2.44. $\tilde{s}^{O_j}$ represents the arc length along the centerline of the road $\mathcal{R}_c$. The relation between

$\tilde{s}^{O_j}$ and s^{O_j} depends on the segment type. As derived in B.2, the final arc length relations are given by

$$s^{O_j} = \begin{cases} \tilde{s}^{O_j} & \text{straight road segments} \\ \tilde{s}^{O_j} + y_{\text{off}}^{O_j}\ \kappa_k\ \tilde{s}^{O_j} \qquad \text{for} & \text{circular road segments} \\ \tilde{s}^{O_j} + y_{\text{off}}^{O_j}\left(\kappa_{k,0}\ \tilde{s}^{O_j} + \kappa_k' \frac{\left(\tilde{s}^{O_j}\right)^2}{2}\right) & \text{clothoid road segments.} \end{cases} \tag{2.49}$$

2.1.2.2. Out-of-lane Extrapolation of Obstacles

In case the obstacle is assumed to ignore the course of the road, its motion is extrapolated depending only on the obstacle's current dynamic state at $t = t_0$. For the extrapolation, a kinematic singletrack model is used. The motion is extrapolated in the object-fixed reference frame $\underrightarrow{O_j^0} = \underrightarrow{O_j(t_0)}$ which is fixed at $t = t_0$ and then transformed into the road fixed reference frame $\underrightarrow{\mathrm{R}}$ by applying

$$ {}_{R}\mathbf{r}_{\text{out-of-lane}}^{R*,O_j^*} = {}_{R}\mathbf{r}^{R*,O_j^{0*}} + \mathbf{C}^{O_j^0,R}\ {}_{O_j^0}\mathbf{r}_{\text{out-of-lane}}^{O_j^{0*},O_j^*}, \tag{2.50}$$

$$ {}^{R}\psi^{O_j} = {}^{R}\psi^{O_j^0} + {}^{O_j^0}\psi^{O_j}. \tag{2.51}$$

For $\dot{\psi} = \omega = 0$, the obstacle moves on a straight line given by

$$ {}_{O_j^0}\mathbf{r}_{\text{out-of-lane}}^{O_j^{0*},O_j^*}(t) = \begin{bmatrix} s^{O_j}(t) \\ 0 \end{bmatrix}, \tag{2.52}$$

where the traveled arc length $s^{O_j}(t)$ is determined using Equations 2.40 to 2.42.

For $\dot{\psi} = \omega \neq 0$, the obstacle is assumed to maintain its curve radius. As illustrated in Figure 2.13, this results in a circular motion with the instantaneous center of rotation ICR lying besides the object's rear axle, see [Mitschke and Wallentowitz, 2004]. For this rotational movement of the point H_j on the rear axle of obstacle O_j, its center O_j^* can be determined for any point of time according to

$$ {}_{O_j^0}\mathbf{r}_{\text{out-of-lane}}^{O_j^{0*},O_j^*}(t) = {}_{O_j^0}\mathbf{r}^{O_j^{0*},H_j^0} + {}_{O_j^0}\mathbf{r}^{H_j^0,H_j}(t) + {}_{O_j^0}\mathbf{r}^{H_j,O_j^*}(t) \tag{2.53}$$

$$ = \begin{bmatrix} -l_H \\ 0 \end{bmatrix} + \begin{bmatrix} R_H \sin\phi(t) \\ R_H\left(1 - \cos\phi(t)\right) \end{bmatrix} + \begin{bmatrix} l_H \cos\phi(t) \\ l_H \sin\phi(t) \end{bmatrix}, \tag{2.54}$$

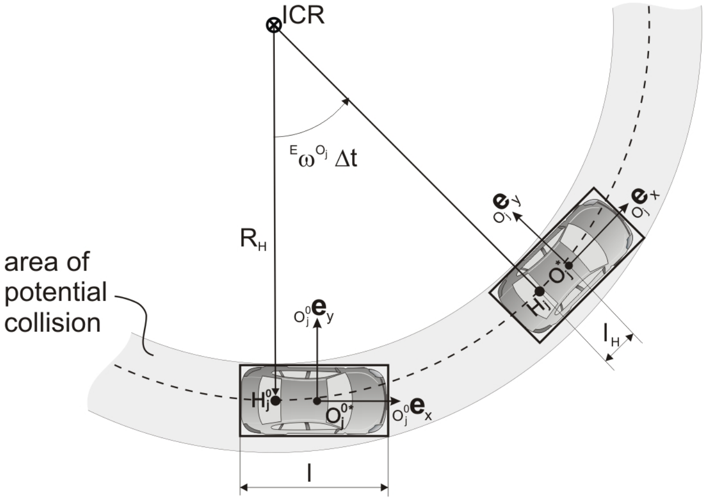


Figure 2.13.: Out-of-lane obstacle extrapolation
If the obstacle is not extrapolated along the road, it is extrapolated due to its current dynamic state at $t = t_0$, using a kinematic single track model. The area swept by the obstacle is denoted *area of potential collisions* which is used for the computation of the temporal distances.

where

$$\phi\left(t\right) = s^{O_j}\left(t\right)/R_H \tag{2.55}$$

and the radius R_H is given by

$$R_H = \left\| \mathbf{r}^{ICR,H_j^0} \right\| = \frac{\left\| {}^E\mathbf{v}^{H_j}\left(t_0\right) \right\|}{{}^E\omega^{O_j}\left(t_0\right)} = \frac{\left\| {}^E\mathbf{v}^{O_j^*}\left(t_0\right) + \mathbf{r}^{O_j^{0*},H_j^0} \times {}^E\boldsymbol{\omega}^{O_j}\left(t_0\right) \right\|}{{}^E\omega^{O_j}\left(t_0\right)}. \tag{2.56}$$

The obstacle's orientation can be calculated according to

$${}^R\psi^{O_j}\left(t\right) \;=\; {}^R\psi^{O_j^0} + \phi\left(t\right). \tag{2.57}$$

2.1.2.3. Computation of spatial distances d^{O_j}

For the evaluation of the obstacle potential (that is then used to determine the external forcefield) it is necessary to compute the spatial distances from the ego-vehicle $\mathcal{V}$ at any node $P_i, i = 0..N$ to all obstacles $\mathcal{O}_j, j = 1..M$ for the correlating points of time. Further, it is of interest to determine the closest point $\hat{O}_j$ on each obstacle. The position and orientation of the obstacles are given by the extrapolations, for example as described in Sections 2.1.2.1 and 2.1.2.2. This results in $N \times M$ necessary distance computations.

The distance $d_i^{O_j} = \sqrt{(\Delta x_i^{O_j})^2 + (\Delta y_i^{O_j})^2}$ is defined as the minimum distance between the two rectangles, $\mathcal{V}$ at P_i and $\mathcal{O}_j$ for $t = t_i$, see Figure 2.14. Such distance computations have received much attention because they serve as fundamental building blocks in many collision detection packages. A very simple and straight forward approach is to treat each node P_i separately. First, the motion of every obstacle is extrapolated for $t = t_i$, and second, the distance $d_i^{O_j}$ is computed.

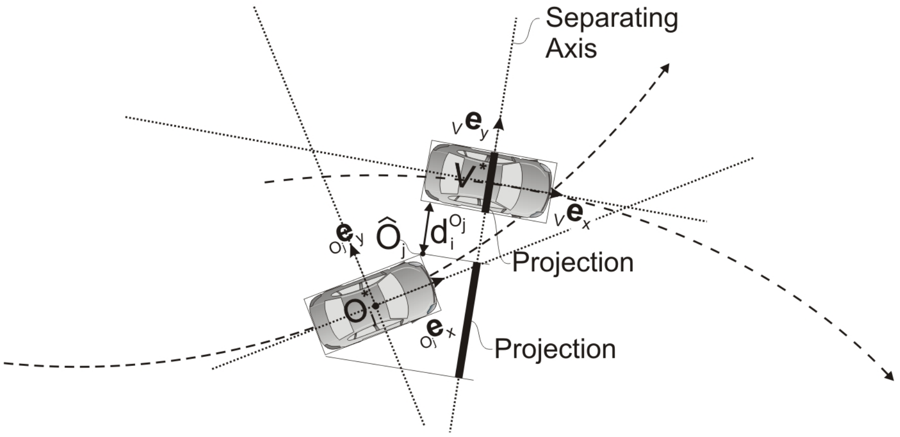


Figure 2.14.: Distance to obstacles
The distances $d_i^{O_j}$ are determined by simple vertice-edge distance computations. Collision checks are performed according to the separating axis theorem.

Early work on distance calculation includes a linear-time algorithm by Dobkin and Kirkpatrick to determine the distance between two static convex polyhedra, [Dobkin and Kirkpatrick, 1985]. More sophisticated and efficient approaches have been developed using hierarchical data structures and incremental distance computation by exploiting the coherence of motion. Lin and Canny deduct that for small time increments between two distance calculations the closest pair of features (vertices, edges, or faces) between two polyhedra will not vary much. In [Lin and Canny, 1991] they propose an algorithm that exploits this coherence for convex polyhedra and "walks" on the surface of polyhedra from a previously found closest pair to the next. Empirically, the query time was found to be almost constant.

Many modifications and extensions of this algorithm have hence been devised, as for example in [Cameron, 1997] or [Mirtich, 1998]. A combination of incremental distance computations with hierarchical data structures was suggested by [Guibas et al., 1999]. In [Ehmann and Lin, 2001], Ehmann and Lin devise a similar method that enables queries of varying generality such as collision detection, approximate and exact distance calculation and disjoint contact determination in a single framework. Even more general geometries, i.e. non-convex objects could be handled by an extension published in [Quinlan, 1994]. Non-convex objects are decomposed into several convex objects using a hierarchical bounding representation and spherical approximations.

Since for the course of this work the distance calculations are performed in the x-y-plane of the augmented workspace where all objects are approximated as simple rectangles, very general algorithms such as those for non-convex objects are not expected to significantly benefit the average computation time of distances. Further research would be necessary at this point in order to evaluate and adapt the most suitable method. Especially incremental algorithms seem promising due to the vast number of distance calculations and small time increments. For first implementations as used for the experimental results in Chapter 4, a very crude algorithm has been implemented that treats the extrapolation for all nodes P_i separately and does not use any special data structure but simply performs eight vertice-edge distance computations to yield the minimum distance and the closest point $\hat{O}_j$ on the obstacle $\mathcal{O}_j$ to P_i.

2.1.2.4. Computation of temporal distances Δt^{O_j}

While the spatial distance always exists, the temporal distance on the other hand is defined in such a way that it only exists with non-infinite value if there is a potential collision at a certain node P_i. A potential collision occurs if the extrapolation of obstacle $\mathcal{O}_j$ for any time from its detection $t^{O_j}_{\text{detect}}$ to the extrapolation horizon t_{ehorizon} collides with the ego-vehicle $\mathcal{V}$ at (x_i, y_i), i.e. if the projection of P_i to the x-y-plane lies within the area of potential collisions, see Figures 2.13, 2.15. The temporal distance $\Delta t_i^{O_j}$ from P_i to obstacle $\mathcal{O}_j$ is defined as

$$\Delta t_i^{O_j} \quad := \quad \begin{cases} \left|\hat{t}^{O_j}(x_i, y_i) - t_i\right| & \text{if potential collision exists for } P_i \\ \infty & \text{otherwise.} \end{cases} \tag{2.58}$$

For each planned position P_i of the ego-vehicle $\mathcal{V}$ that has a potential collision with an obstacle, there exists a certain period of time during which this potential collision occurs. It starts at $t_i^{OF_j}$ when the front of the obstacle $\mathcal{O}_j$ reaches the position of the ego-vehicle $\mathcal{V}(t_i)$ that is associated with P_i, and it finishes at $t_i^{OR_j}$ when the rear end of the obstacle

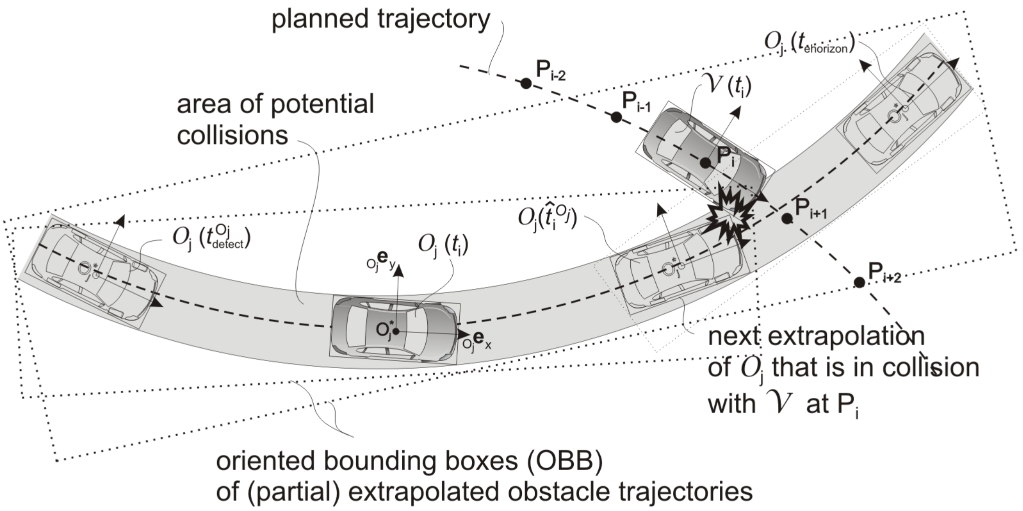

planned trajectory
area of potential collisions
P_{i-2}
P_{i-1}
P_i
P_{i+1}
P_{i+2}
O_j (t_{ehorizon})
$\mathcal{V}(t_i)$
$O_j(\hat{t}_i^{O_j})$
O_j ($t_{\text{detect}}^{O_j}$)
O_j (t_i)
$_{O_j}\mathbf{e}_y$
$_{O_j}\mathbf{e}_x$
O_j
next extrapolation of O_j that is in collision with $\mathcal{V}$ at P_i
oriented bounding boxes (OBB) of (partial) extrapolated obstacle trajectories

Figure 2.15.: Temporal distance to obstacles
If the ego vehicle $\mathcal{V}$ intersects ("collides") with the swath (area of potential collision) of the obstacle $\mathcal{O}_j$ for any node P_i, there exists a finite (positive or negative) time to collision $\Delta t_i^{O_j}$. The collision checking can be performed e.g. by oriented bounding boxes (OBB) of partial trajectories.

leaves the potential collision. $\hat{t}^{O_j}(x_i, y_i)$ denotes the point of time within this period that is closest to t_i. Setting $\Delta t_i^{O_j}$ to infinity lets the temporal part of the obstacle potential vanish, see Section 2.1.2.5. Figure 2.16 illustrates the two-step process of computing $\Delta t_i^{O_j}$ in a blockdiagram.

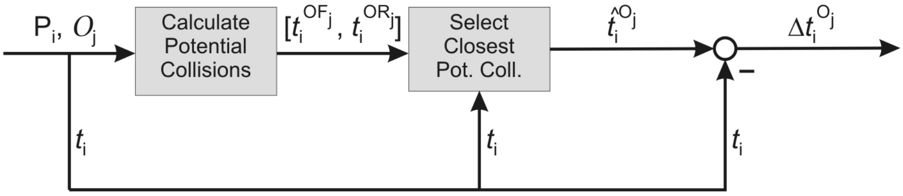

P_i, O_j
Calculate Potential Collisions
$[t_i^{OF_j}, t_i^{OR_j}]$
Select Closest Pot. Coll.
$\hat{t}_i^{O_j}$
$\Delta t_i^{O_j}$
–
t_i
t_i
t_i

Figure 2.16.: Blockdiagram for calculation of temporal distance to obstacle
For each node P_i and its time t_i and each obstacle $\mathcal{O}_j$ it is first checked whether a potential collision exists, see 2.15. Then the time interval $[t_i^{OF_j}, t_i^{OR_j}]$ is determined in which the obstacle is in collision with the ego vehicle if it was static at (x_i, y_i). Finally, from this time interval the shortest difference to t_i is calculated.

In order to calculate $\left[t_i^{OF_j}, t_i^{OR_j}\right]$ and finally $\hat{t}^{O_j}(x_i, y_i)$, a number of different approaches can be taken. To handle any method for obstacle extrapolation, a recursive numerical approach is proposed. The obstacles' extrapolated trajectories are recursively broken up into parts. Each partial trajectory is then tested for collision with the rectangle that represents ego-vehicle at P_i. If there is no collision, this part is disregarded. In case there exists a collision, the partial trajectory is further subdivided. This is repeated until either the time interval Δt of the examined partial trajectory drops below the threshold $\epsilon_{\Delta t}$ or all parts are found to be collision-free. $\epsilon_{\Delta t}$ determines the accuracy of the returned temporal distance.

For the collision detection of the partial trajectories, a number of different algorithms can be applied, among those many listed already in Section 2.1.2.3, see for example [Dobkin and Kirkpatrick, 1985; Mirtich, 1998]. Lin and Gottschalk give a survey of different classes and algorithms in [Lin and Gottschalk, 1998]. Efficient collision detection algorithms mostly rely on hierarchically structured approximations of different objects with suitable bounding volumes. In [Hubbard, 1996] for example, polyhedra are approximated with spheres for collision checking, while Klosowski et al. use hierarchically structured convex polytopes, [Klosowski et al., 1998]. Newer approaches further try to extend collision detection to dynamically deforming objects, see [Curtis et al., 2008; Teschner et al., 2005].

Since in this case all objects are represented as rectangles, the approximation by oriented bounding boxes (OBB) as presented in [Gottschalk et al., 1996] seems most appropriate, see Figure 2.15. The collision test between the rectangular ego-vehicle and the rectangularly bounded partial obstacle trajectory is performed by an efficient test that is based on the separating axis theorem which is frequently used in this context, [Gottschalk, 2000; LaValle, 2006; Van Den Bergen, 2005]. For rectangles this means that there exists a collision if and only if the projections on all four axes intersect, see Figure 2.14. These axes coincide with the directions of the rectangles' edges.

For the special obstacle extrapolations used within the course of this work, the case that there exists no potential collision can be found more directly, since only certain geometric types of areas are swept by the obstacles. If an obstacle follows the road, the test is whether or not the ego-vehicle at node P_i covers a certain width of the road. If an obstacle is extrapolated out of the lane, the swept area of potential collision is bounded either by a rectangle for ${}^{E}\omega^{O_j} = 0$ or two circles for ${}^{E}\omega^{O_j} \neq 0$, see Figure 2.13 and Sections 2.1.2.1, 2.1.2.2.

2.1.2.5. Definition of Obstacle Potential

For the trajectory optimization, the obstacles are represented by a potential field $V^{\mathcal{O}}$, from which the obstacle force field $\mathbf{F}^{\mathcal{O}}$ is computed. Analogously to the road potential, there is a logarithmic component for each obstacle that decays with growing distance to the obstacle and reaches infinity at the obstacles' borders

$$V^{\mathcal{O}} = \sum_{j=1}^{M} V^{O_j}, \tag{2.59}$$

$$V^{O_j} = -k_y^{O_j} \ln\left(\left\| d^{O_j} \right\|\right) - k_t^{O_j} \ln\left(\left\| \Delta t^{O_j} \right\|\right). \tag{2.60}$$

The spatial and temporal distances d^{O_j} and Δt^{O_j} to all obstacles O_j are determined as presented in the sections above.

The obstacle potential depends on the planned position $(x,\, y)$ and orientation ψ of the ego vehicle. Further, the obstacle potential varies over time for dynamic obstacles. Due to the relations established above between the phase space $\mathbb{X}$ and the work space $\mathbb{W}$ the potential at node P_i depends on the position of P_i and P_{i-1} in $\mathbb{W}$

$$V_i^{\mathcal{O}} = f\left(x_i, y_i, t_i, \psi_i\right) = f\left(\mathbf{r}_i, \mathbf{r}_{i-1}\right). \tag{2.61}$$

Figure 2.17 illustrates the external potential field for a straight road and two obstacles for a single point of time $t = 0$ and $\psi = 0$. It goes to infinity as the distance to the borders of the road or the obstacles approaches zero. As can be seen, areas closer to the border of the road than half of the ego vehicle width are prohibited. The minimum of the combined road and obstacle potential obviously does not always lie in the middle of the right lane anymore. In case an obstacle is not static but moves, the external potential also varies over time, as illustrated in Figure 2.18.

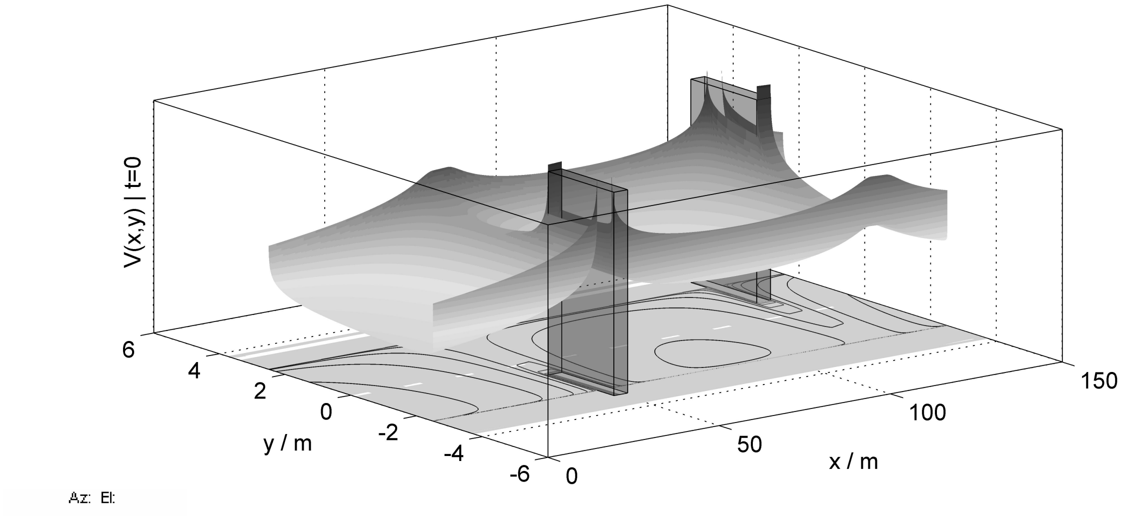


Figure 2.17.: External potential field $V^{\text{ext}}(x, y)$ for $t = 0$ and $\psi = 0$
The figure illustrates the external potential field for a straight road and two obstacles for a single point of time $t = 0$ and $\psi = 0$. It goes to infinity as the distance to the borders of the road or the obstacles approaches zero.

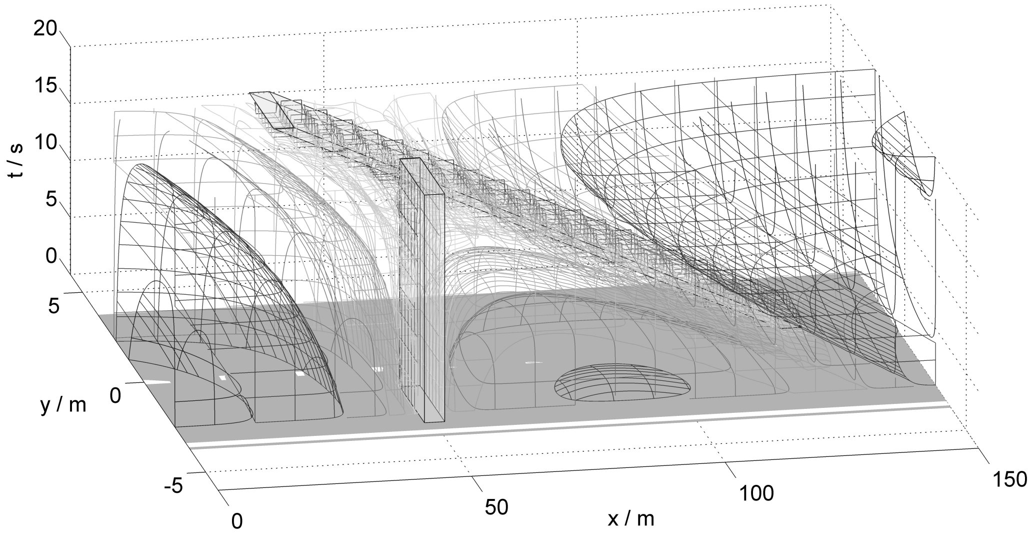


Figure 2.18.: External potential field $V^{\text{ext}}(x, y, t)$ for $\psi = 0$
The figure illustrates the external potential field for a straight road with one static obstacle and one obstacle moving with a constant velocity from right to left on the farther lane.

2.2. Internal Force Field

Depending on the trajectory's representation in the augmented phase space $\mathbb{X}_T$, a virtual internal force field $\mathbf{F}^{\text{int}}$ is defined that enhances the trajectory's drivability by punishing configurations that are dynamically unfavorable. Different models can be applied here. For forcefield path planning for example an inverse linear single track model has been applied, see [Hesse and Sattel, 2007]. Pending future incorporation of more detailed models in to this trajectory optimization method, the suitability of a trajectory is here quantified in terms of accelerations of a point-mass model in lateral and longitudinal road direction along the trajectory. In addition to the expected lateral and absolute acceleration a_y, a, also the change in acceleration, called jerk, is regarded. Further, the internal force field comprises another longitudinal component that abets a certain desired traveling speed v_{des}

$$\begin{aligned}\mathbf{F}^{\text{int}}\left(v, a, a_y, \dot{a}, \dot{a}_y\right) \quad = \quad & \left(\mathbf{F}_y^{\text{acc}}\left(a_y\right) + \mathbf{F}_y^{\text{dacc}}\left(\dot{a}_y\right)\right)\, {}_{\tilde{R}}\mathbf{e}_y \\ & + \left(\mathbf{F}_t^{\text{vel}}\left(v\right) + \mathbf{F}_t^{\text{acc}}\left(a\right) + \mathbf{F}_t^{\text{dacc}}\left(\dot{a}\right)\right)\, {}_{\tilde{R}}\mathbf{e}_t. \end{aligned} \tag{2.62}$$

The following sections define the forces that result when the virtual internal force field is evaluated at a certain node P_i. As mentioned above, the planned trajectory consists of discrete nodes P_i in the augmented workspace $\mathbb{W}_T = \langle x, y, t\rangle$. Using the relations between $\mathbb{W}_T$ and $\mathbb{X}_T$ that were stated at the beginning of this chapter, the dependencies of the internal force at a node P_i read

$$\begin{aligned}\mathbf{F}_i^{\text{int}}\left(\mathbf{r}_{i+1}, \mathbf{r}_i, \mathbf{r}_{i-1}, \mathbf{r}_{i-2}\right) \quad = \quad & \left(F_{y,i}^{\text{acc}}\left(\mathbf{r}_{i+1}, \mathbf{r}_i, \mathbf{r}_{i-1}\right) + F_{y,i}^{\text{dacc}}\left(\mathbf{r}_{i+1}, \mathbf{r}_i, \mathbf{r}_{i-1}, \mathbf{r}_{i-2}\right)\right)\, {}_{\tilde{R}}\mathbf{e}_y \\ & + \left(F_{t,i}^{\text{acc}}\left(\mathbf{r}_{i+1}, \mathbf{r}_i, \mathbf{r}_{i-1}\right) + F_{t,i}^{\text{dacc}}\left(\mathbf{r}_{i+1}, \mathbf{r}_i, \mathbf{r}_{i-1}, \mathbf{r}_{i-2}\right)\right)\, {}_{\tilde{R}}\mathbf{e}_t \\ & + F_{t,i}^{\text{vel}}\left(\mathbf{r}_i, \mathbf{r}_{i-1}\right)\, {}_{\tilde{R}}\mathbf{e}_t. \end{aligned} \tag{2.63}$$

2.2.1. Representation of Lateral Motion

The lateral motion along the planned trajectory is optimized by the forces $F_{y,i}^{\text{acc}}$ and $F_{y,i}^{\text{dacc}}$ in lateral road direction that depend on the lateral acceleration and rate of change of this acceleration, called lateral jerk, respectively.

The lateral acceleration force $F_{y,i}^{\text{acc}}$ at each node P_i works against high accelerations in lateral road direction $a_{y,i}$ at the node P_i, given by the backward difference quotient as defined at the beginning of this chapter in Equations 2.6ff,

$$\begin{aligned} F_{y,i}^{\text{acc}} &= -k_y^{\text{acc}} a_{y,i} && (2.64) \\ &= -k_y^{\text{acc}} 2 \frac{\frac{y_i - y_{i-1}}{t_i - t_{i-1}} - \frac{y_{i-1} - y_{i-2}}{t_{i-1} - t_{i-2}}}{t_i - t_{i-2}}. && (2.65) \end{aligned}$$

All components $(t_i, t_{i-1}, t_{i-2}, y_i, y_{i-1}, y_{i-2})$ are given with respect to the road centerline reference frame for P_i, $\underline{\uparrow\tilde{R}^i}$.

At this point it can be seen why it makes sense to define the lateral acceleration $a_{y,i}$ the way it is. The use of a central difference quotient as relation between the augmented workspace $\mathbb{W}_T$ and the phase space variable $a_{y,i}$ would , among others, not be well-suited to produce a low acceleration at the very beginning of the trajectory, see $F_{y,i}^{\text{acc, c}}$ in Figure 2.19. In case of a sharp bend at P_0 from P_{-1} (which results from the previously planned trajectory) to P_1, a central difference quotient might produce only a force at P_0. However, because this node is fixed, the acceleration force $F_{y,0}^{\text{acc, c}}$ would be rendered useless.

As can be seen in Figure 2.19, the use of a backward difference quotient in the same situation, on the other hand, produces the force $F_{y,1}^{\text{acc}}$ on P_1 that effectively reduces the curvature and hence the lateral acceleration at P_0.

The lateral jerk force $F_{y,i}^{\text{dacc}}$ is designed to produce trajectories with a constant lateral acceleration and to prevent high lateral jerks. It depends on the change in acceleration in lateral road direction $\dot{a}_{y,i}$ at P_i

$$\begin{aligned} F_{y,i}^{\text{dacc}} &= -k_y^{\text{dacc}} \dot{a}_{y,i} && (2.66) \\ &= -k_y^{\text{dacc}} \frac{6}{t_{i+1} - t_{i-2}} \left(\frac{\frac{y_{i+1} - y_i}{t_{i+1} - t_i} - \frac{y_i - y_{i-1}}{t_i - t_{i-1}}}{t_{i+1} - t_{i-1}} - \frac{\frac{y_i - y_{i-1}}{t_i - t_{i-1}} - \frac{y_{i-1} - y_{i-2}}{t_{i-1} - t_{i-2}}}{t_i - t_{i-2}} \right). && (2.67) \end{aligned}$$

As before, all components (y_i, y_{i-1}, y_{i-2}) are given with respect to the road centerline reference frame for P_i, $\underline{\uparrow\tilde{R}^i}$. The resulting trajectory can be tuned by scaling k_y^{acc} and k_y^{dacc}, see Section 2.6.

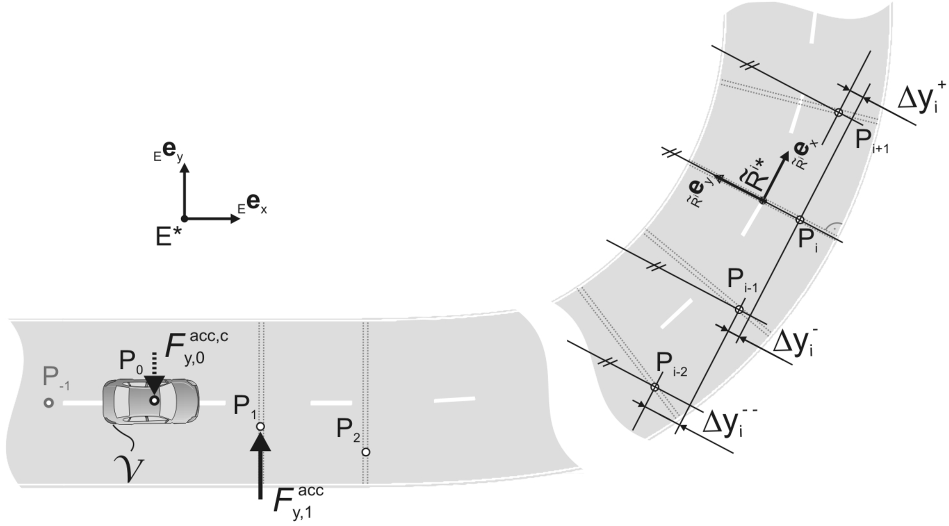


Figure 2.19.: Lateral acceleration forces
$F^{acc}_{y,i}$ depends on the lateral acceleration at P_i as defined by the backward difference quotient. The lateral offsets y_i, y_{i-1}, y_{i-2} therein are given with respect to $\tilde{R}^i$.

2.2.2. Representation of Longitudinal Motion

The total internal force in time direction at node P_i is composed of three parts: The acceleration force $F^{\mathrm{acc}}_{t,i}$, the jerk force $F^{\mathrm{dacc}}_{t,i}$ that depends on the change in acceleration, and finally the travel velocity force $F^{\mathrm{vel}}_{t,i}$ that depends on the deviation from the desired traveling speed.

The acceleration force $F^{\mathrm{acc}}_{t,i}$ in time-direction at node P_i is given by

$$F^{\mathrm{acc}}_{t,i} = k^{\mathrm{acc},1}_t a_i \tag{2.68}$$

$$= 2k^{\mathrm{acc}}_t \frac{\frac{\Delta s_i}{t_i - t_{i-1}} - \frac{\Delta s_{i-1}}{t_{i-1} - t_{i-2}}}{t_i - t_{i-2}}, \tag{2.69}$$

where the spatial distances are defined as before

$$\Delta s_i = \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}, \tag{2.70}$$

$$\Delta s_{i-1} = \sqrt{(x_{i-1} - x_{i-2})^2 + (y_{i-1} - y_{i-2})^2}. \tag{2.71}$$

As before, the components $(x_i, x_{i-1}, x_{i-2}, y_i, y_{i-1}, y_{i-2})$ given with respect to $\tilde{R}^i$.

Analogously to the representation of the lateral motion, there is also a force $F_{t,i}^{\text{dacc}}$ that depends on the change in acceleration $\dot{a}_i$ at the node P_i

$$F_{t,i}^{\text{dacc}} = k_t^{\text{dacc}} \dot{a}_i \tag{2.72}$$

$$= k_t^{\text{dacc}} \frac{6}{t_{i+1} - t_{i-2}} \left(\frac{\frac{\Delta s_{i+1}}{t_{i+1}-t_i} - \frac{\Delta s_i}{t_i - t_{i-1}}}{t_{i+1} - t_{i-1}} - \frac{\frac{\Delta s_i}{t_i - t_{i-1}} - \frac{\Delta s_{i-1}}{t_{i-1} - t_{i-2}}}{t_i - t_{i-2}} \right), \tag{2.73}$$

where the spatial distances are defined as

$$\Delta s_{i+1} = \sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}, \tag{2.74}$$

$$\Delta s_i = \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}, \tag{2.75}$$

$$\Delta s_{i-1} = \sqrt{(x_{i-1} - x_{i-2})^2 + (y_{i-1} - y_{i-2})^2}. \tag{2.76}$$

Again, the components $(x_{i+1}, x_i, x_{i-1}, y_{i+1}, y_i, y_{i-1})$ are given with respect to $\uparrow\!\underrightarrow{\tilde{R}^i}$.

The resulting trajectory can be tuned by scaling k_t^{acc} and k_t^{dacc}, see Section 2.6.

The velocity force $F_{t,i}^{\text{vel}}$ at the node P_i punishes a divergence from the desired traveling speed v_{des} and ensures that the vehicle returns to it when possible. The force depends on the difference between the current velocity v_i at node P_i and the desired one, v_{des}

$$F_{t,i}^{\text{vel}} = k^{\text{vel}} (v_i - v_{\text{des}}) \tag{2.77}$$

$$= k^{\text{vel}} \left(\frac{\Delta s_i}{t_i - t_{i-1}} - v_{\text{des}} \right), \tag{2.78}$$

where

$$\Delta s_i = \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}, \tag{2.79}$$

$$\tag{2.80}$$

with $(x_i, x_{i-1}, y_i, y_{i-1})$ are given with respect to $\uparrow\!\underrightarrow{\tilde{R}^i}$.

This force also determines the distance with which a leading obstacle is followed as discussed later in Section 2.6.

2.3. Preview Force Field

The force field framework as presented so far already leads to smooth, drivable, and collision-free trajectories, however, they can be improved further. The introduction of force fields depending on the acceleration and jerk creates trajectories that change the current dynamic state of the vehicle as little and as slowly as possible. The downside to this desirable characteristic is that planned trajectories start for example an overtaking maneuver rather late and the maximum lateral deviation is located behind the middle of the obstacle that is passed. Further, when returning the the right lane, the planned trajectory shows some overshoot.

Therefore, inspired by lookahead controllers, a lookahead is introduced such that the overtaking maneuver is initiated earlier. In order to introduce such a preview, for every node the external force field is evaluated at a preview point and the computed force is then applied at the node P_i as illustrated in Figure 2.20. The preview point P_i^{prev} for node P_i is determined by the orientation given by P_i and P_{i-1} and a certain preview distance l_{prev}

$$\mathbf{r}^{P_i^{\text{prev}}} = \mathbf{r}^{P_i} + \mathbf{r}^{P_i,P_i^{\text{prev}}}, \tag{2.81}$$

where

$$\mathbf{r}^{P_i,P_i^{\text{prev}}} = l_{\text{prev}} \frac{\mathbf{r}^{P_{i-1},P_i}}{\| \mathbf{r}^{P_{i-1},P_i} \|} = l_{\text{prev}} \begin{bmatrix} \sin \psi_i \\ \cos \psi_i \end{bmatrix}. \tag{2.82}$$

In case the preview point lies outside the road or inside an obstacle it is moved to a certain minimum distance ϵ_{prev} from the roadside or the border of the obstacle in the direction of the nodes P_{i+x} the same distance ahead.

The external force is computed as before as the negative gradient of the external potential field. Thus, the preview force $\mathbf{F}_i^{\text{prev}}$ at node P_i becomes

$$\mathbf{F}_i^{\text{prev}} = \mathbf{F}^{\text{ext}} \left(P_i^{\text{prev}}\right) = -\nabla_{\mathbf{r}_i} V^{\text{ext}} \left(P_i^{\text{prev}}\right) \tag{2.83}$$

$$= -\nabla_{\mathbf{r}_i} V^{\mathcal{R}} \left(P_i^{\text{prev}}\right) - \nabla_{\mathbf{r}_i} V^{\mathcal{O}} \left(P_i^{\text{prev}}\right) \tag{2.84}$$

The road and obstacle potentials at the preview point are evaluated according to the Equations 2.37, 2.38 and 2.59, 2.60.

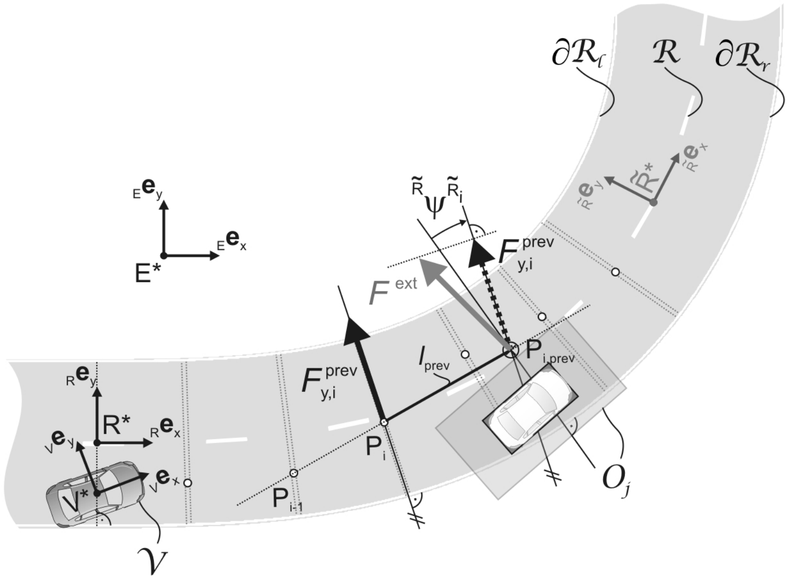


Figure 2.20.: Preview forces
The external forcefield F^{ext} is evaluated at a preview point P_i^{prev} for each node P_i to create a preview force F_i^{prev} that enhances the incorporation of the ego vehicle's orientation and leads to improved obstacle avoidance and lane following plans.

The preview force field enhances the trajectory optimization not only by deforming a trajectory "earlier" to avoid hazards smoothly but also further incorporates of the ego vehicle's orientation along the trajectory which leads to enhanced plans.

2.4. Equilibrium Search

The solution to the trajectory optimization is defined to be the equilibrium-configuration, where all internal, external, and preview forces cancel each other out, i.e. the total force at each node is zero.

All forces and node coordinates are each combined into a single $3\,(N+1)$-dimensional vector

$$\mathbf{z} = \begin{bmatrix} \mathbf{r}_0 \\ \mathbf{r}_1 \\ \mathbf{r}_2 \\ \vdots \\ \mathbf{r}_N \end{bmatrix} \quad \text{and} \quad \mathbf{F} = \begin{bmatrix} \mathbf{F}_0 \\ \mathbf{F}_1 \\ \mathbf{F}_2 \\ \vdots \\ \mathbf{F}_N \end{bmatrix}, \tag{2.85}$$

where

$$\mathbf{r}_i = \mathbf{r}^{R^*,P_i} = \begin{bmatrix} {}_{R}r_x^{R^*,P_i} \\ {}_{R}r_y^{R^*,P_i} \\ {}_{R}r_t^{R^*,P_i} \end{bmatrix}, \quad \mathbf{F}_i = \mathbf{F}^{P_i} = \begin{bmatrix} {}_{R}F_x^{P_i} \\ {}_{R}F_y^{P_i} \\ {}_{R}F_t^{P_i} \end{bmatrix}. \tag{2.86}$$

Therefore, the task is to solve the $3\,(N+1)$-dimensional nonlinear equation

$$\mathbf{F}\,(\mathbf{z}) = \mathbf{0}. \tag{2.87}$$

Because there is no analytical solution, a numerical approach must be taken. For an overview of existing methods for this purpose see [Ortega and Rheinboldt, 1970; Roos and Schwetlick, 1999; Schwetlick and Kretzschmar, 1991]. An analysis and comparison of some available numeric algorithms was performed in a bachelor thesis supervised by this author, [Stüve, 2007]. The vast number of algorithms mainly results from various combinations of algorithms for subproblems.

One main approach is to subdivide the multidimensional nonlinear problem into multiple scalar nonlinear equations. Each equation is solved separately in each iteration. For the next iteration all equations are solved again using the solutions of the separate equations from the previous iteration. Examples for this kind of algorithm are the nonlinear Jacobi, nonlinear Gauss-Seidel, or nonlinear Successive Over Relaxation (SOR) algorithms [Rheinboldt, 1998]. Often the solution of the nonlinear scalar equations also requires iterative numerical methods.

Another main approach, that is also chosen in this work, is to simplify the nonlinear problem by some kind of approximation at a certain starting point $\mathbf{z}_0$ to acquire a simpler system of equations that can be solved efficiently. In each iteration, this approximation is solved and then a new approximation is constructed where the previous approximation was zero, thus converging to a solution. For this work in each step the nonlinear problem is linearized

$$\mathbf{F}\,(\mathbf{z}_k) + \mathbf{A}\Delta\mathbf{z}_k = 0. \tag{2.88}$$

The new point of approximation is then iteratively found by

$$\mathbf{z}_{k+1} = \mathbf{z}_k + \Delta\mathbf{z}_k, \tag{2.89}$$

until the displacement $\Delta \mathbf{z}_k$ falls below a certain threshold $\epsilon_{\Delta z}$,

$$\Delta \mathbf{z}_k < \epsilon_{\Delta z}, \tag{2.90}$$

and the iteration is terminated successfully.

The different algorithms in this class differ with regards the kind of linearization and the solution of the linear subproblem. An extensive overview can be found for example in [Argyros, 2005; Deuflhard, 2006].

The standard Newton method uses an analytically given Jacobian $\mathbf{A} = \mathbf{J}(\mathbf{z}_k) = \frac{\partial \mathbf{F}}{\partial \mathbf{z}_k^\mathsf{T}}$ that is computed for each iteration k. The resulting linear approximation of Equation 2.87 at $\mathbf{z} = \mathbf{z}_k$ is given by

$$\mathbf{F}(\mathbf{z}_k) + \mathbf{J}(\mathbf{z}_k)\,\Delta \mathbf{z}_k = 0. \tag{2.91}$$

$\mathbf{J}$ represents the Jacobian

$$\mathbf{J} = \begin{bmatrix} \mathbf{J}_{0,0} & \mathbf{J}_{0,1} & \mathbf{J}_{0,2} & \cdots & \mathbf{J}_{0,N} \\ \mathbf{J}_{1,0} & \mathbf{J}_{1,1} & \mathbf{J}_{1,2} & \cdots & \mathbf{J}_{1,N} \\ \mathbf{J}_{2,0} & \mathbf{J}_{2,1} & \mathbf{J}_{2,2} & \cdots & \mathbf{J}_{2,N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{J}_{N,0} & \mathbf{J}_{N,1} & \mathbf{J}_{N,2} & \cdots & \mathbf{J}_{N,N} \end{bmatrix} = \begin{bmatrix} \frac{\partial \mathbf{F}_0}{\partial \mathbf{r}_0^\mathsf{T}} & \frac{\partial \mathbf{F}_0}{\partial \mathbf{r}_1^\mathsf{T}} & \frac{\partial \mathbf{F}_0}{\partial \mathbf{r}_2^\mathsf{T}} & \cdots & \frac{\partial \mathbf{F}_0}{\partial \mathbf{r}_N^\mathsf{T}} \\ \frac{\partial \mathbf{F}_1}{\partial \mathbf{r}_0^\mathsf{T}} & \frac{\partial \mathbf{F}_1}{\partial \mathbf{r}_1^\mathsf{T}} & \frac{\partial \mathbf{F}_1}{\partial \mathbf{r}_2^\mathsf{T}} & \cdots & \frac{\partial \mathbf{F}_1}{\partial \mathbf{r}_N^\mathsf{T}} \\ \frac{\partial \mathbf{F}_2}{\partial \mathbf{r}_0^\mathsf{T}} & \frac{\partial \mathbf{F}_2}{\partial \mathbf{r}_1^\mathsf{T}} & \frac{\partial \mathbf{F}_2}{\partial \mathbf{r}_2^\mathsf{T}} & \cdots & \frac{\partial \mathbf{F}_2}{\partial \mathbf{r}_N^\mathsf{T}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \mathbf{F}_N}{\partial \mathbf{r}_0^\mathsf{T}} & \frac{\partial \mathbf{F}_N}{\partial \mathbf{r}_1^\mathsf{T}} & \frac{\partial \mathbf{F}_N}{\partial \mathbf{r}_2^\mathsf{T}} & \cdots & \frac{\partial \mathbf{F}_N}{\partial \mathbf{r}_N^\mathsf{T}} \end{bmatrix}, \tag{2.92}$$

where each partial Jacobian $\mathbf{J}_{\nu,\mu}$ is

$$\mathbf{J}_{\nu,\mu} = \frac{\partial \mathbf{F}_\nu}{\partial \mathbf{r}_\mu^\mathsf{T}} = \begin{bmatrix} \frac{\partial\, {}_R F_x^{P_\nu}}{\partial\, {}_R r_x^{R^*,P_\mu}} & \frac{\partial\, {}_R F_x^{P_\nu}}{\partial\, {}_R r_y^{R^*,P_\mu}} & \frac{\partial\, {}_R F_x^{P_\nu}}{\partial\, {}_R r_t^{R^*,P_\mu}} \\ \frac{\partial\, {}_R F_y^{P_\nu}}{\partial\, {}_R r_x^{R^*,P_\mu}} & \frac{\partial\, {}_R F_y^{P_\nu}}{\partial\, {}_R r_y^{R^*,P_\mu}} & \frac{\partial\, {}_R F_y^{P_\nu}}{\partial\, {}_R r_t^{R^*,P_\mu}} \\ \frac{\partial\, {}_R F_t^{P_\nu}}{\partial\, {}_R r_x^{R^*,P_\mu}} & \frac{\partial\, {}_R F_t^{P_\nu}}{\partial\, {}_R r_y^{R^*,P_\mu}} & \frac{\partial\, {}_R F_t^{P_\nu}}{\partial\, {}_R r_t^{R^*,P_\mu}} \end{bmatrix}. \tag{2.93}$$

Simplified Newton methods, also called parallel-chords methods, only compute the Jacobian once, $\mathbf{A} = \mathbf{J}(\mathbf{z}_0)$, to save computational cost for the repeated evaluation of $\mathbf{J}$. Similarly, n-ary Newton methods update the Jacobian every n steps. However, the reduction of computational cost for each iteration in general results also in a reduced convergence rate and therefore a higher number of necessary iterations. Besides this approach, also otherwise approximated Jacobians are applicable. Where the Jacobian cannot be given analytically, there exist algorithms that approximate tangents or secants from discrete points of $\mathbf{F}$.

The convergence of the standard Newton method is quadratic. However, the convergence is local and cannot be guaranteed in general. In order to stabilize the Newton algorithm and increase the area of convergence, the class of damped Newton methods has been developed. Here, the computed step size in each iteration is reduced by a certain factor to stabilize the algorithm, rewriting Equation 2.89 to

$$\mathbf{z}_{k+1} = \mathbf{z}_k + \alpha_k \Delta \mathbf{z}_k. \tag{2.94}$$

One prominent solution is the application of an Armijo-type stepsize rule that iteratively finds an appropriate reduction factor that guarantees $\mathbf{F}(\mathbf{z}_{k+1}) < \mathbf{F}(\mathbf{z}_k)$, see for example [Bonnans and Lemaréchal, 2006]. But also other reductions are admissible to achieve a stabilization of the Newton method, see [Cohen, 1981].

The most efficient solution of the linear subproblem largely depends on the matrix structure of $\mathbf{A}$. Since the direct inversion of $\mathbf{A}$ is hardly ever a good idea, there exist several different Gauss algorithms that use row and column pivoting and different factorizations such as LR or Cholesky that solve linear problems efficiently. Besides Gauss, many other methods exist as for example splitting methods such as Jacobi, Gauss-Seidel or SOR can be used.

In order to solve for the equilibrium of the trajectory in the virtual force fields, a Newton method is chosen. The method is applied in local road-fixed reference frames and unnecessary degrees of freedom are removed, see Section 2.4.1. The Jacobian is approximated analytically and evaluated in every iteration, and a variable dampening is applied to stabilize the algorithm and to enforce a number of constraints, see Section 2.4.2.

2.4.1. Application of Algorithm to Local Reference Frames

In order to facilitate the calculations, the forces, position vectors, and displacements are represented in the local reference frames $\uparrow\!\underrightarrow{\tilde{R}^i}$ combining them into ${}_{\tilde{R}}\mathbf{F}$, ${}_{\tilde{R}}\mathbf{z}$, and ${}_{\tilde{R}}\mathbf{\Delta z}$, respectively. Therein, the position and forces for each node P_i are represented in the reference frame $\uparrow\!\underrightarrow{\tilde{R}^i}$ which is rotated around

$$\psi_i \quad = \quad {}^{R}\psi^{\tilde{R}_i} \tag{2.95}$$

with respect to $\underline{\mathrm{R}}$. The relations are given by

$$
\begin{aligned}
{}_{\tilde{R}}\mathbf{z} &= \mathbf{T}\left({}_{R}\mathbf{z}^{\tilde{R}^{i*}} + {}_{R}\mathbf{z}\right) && (2.96)\\
{}_{\tilde{R}}\Delta\mathbf{z} &= \mathbf{T}_{R}\Delta\mathbf{z} && (2.97)\\
{}_{\tilde{R}}\mathbf{F} &= \mathbf{T}_{R}\mathbf{F}, && (2.98)
\end{aligned}
$$

where ${}_{R}\mathbf{z}^{\tilde{R}^{i*}}$ contains the position vectors to the local reference frames

$$
{}_{R}\mathbf{z}^{\tilde{R}^{i*}} = \begin{bmatrix} {}_{R}\mathbf{r}^{R^*,\tilde{R}^{0*}} \\ {}_{R}\mathbf{r}^{R^*,\tilde{R}^{1*}} \\ {}_{R}\mathbf{r}^{R^*,\tilde{R}^{2*}} \\ \vdots \\ {}_{R}\mathbf{r}^{R^*,\tilde{R}^{N*}} \end{bmatrix} \qquad \text{with} \qquad {}_{R}\mathbf{r}^{R^*,\tilde{R}^{i*}} = \begin{bmatrix} {}_{R}r_x^{R^*,\tilde{R}^{i*}} \\ {}_{R}r_y^{R^*,\tilde{R}^{i*}} \\ {}_{R}r_t^{R^*,\tilde{R}^{i*}} \end{bmatrix} \tag{2.99}
$$

and the rotation matrix $\mathbf{T}$ is given by

$$
\mathbf{T} = \begin{bmatrix} \mathbf{T}_0 & 0 & 0 & \dots & 0 \\ 0 & \mathbf{T}_1 & 0 & \dots & 0 \\ 0 & 0 & \mathbf{T}_2 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \mathbf{T}_N \end{bmatrix} \qquad \text{with} \qquad \mathbf{T}_i = \begin{bmatrix} \cos\psi_i & -\sin\psi_i & 0 \\ \sin\psi_i & \cos\psi_i & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.100}
$$

Equation 2.91 can be reformulated to

$$
\mathbf{F} = -\mathbf{J}\Delta\mathbf{z}, \tag{2.101}
$$

where all forces $\mathbf{F}$ and displacements $\Delta\mathbf{z}$ are represented in $\underline{\mathrm{R}}$. Using the transformation matrix in 2.100 and exploiting $\mathbf{T}^{-1} = \mathbf{T}^{\mathsf{T}}$, this results in

$$
{}_{\tilde{R}}\mathbf{F} = -\underbrace{\mathbf{T}^{\mathsf{T}}\mathbf{J}\mathbf{T}}_{{}_{\tilde{R}}\mathbf{J}}\,{}_{\tilde{R}}\Delta\mathbf{z}. \tag{2.102}
$$

The Jacobian ${}_{\tilde{R}}\mathbf{J}$ now contains the partial derivatives of the forces with regards to the displacements in the local reference frames $\underrightarrow{\uparrow\tilde{R}^i}$

$$
{}_{\tilde{R}}\mathbf{J} = \begin{bmatrix} {}_{\tilde{R}}\mathbf{J}_{0,0} & {}_{\tilde{R}}\mathbf{J}_{0,1} & {}_{\tilde{R}}\mathbf{J}_{0,2} & \cdots & {}_{\tilde{R}}\mathbf{J}_{0,N} \\ {}_{\tilde{R}}\mathbf{J}_{1,0} & {}_{\tilde{R}}\mathbf{J}_{1,1} & {}_{\tilde{R}}\mathbf{J}_{1,2} & \cdots & {}_{\tilde{R}}\mathbf{J}_{1,N} \\ {}_{\tilde{R}}\mathbf{J}_{2,0} & {}_{\tilde{R}}\mathbf{J}_{2,1} & {}_{\tilde{R}}\mathbf{J}_{2,2} & \cdots & {}_{\tilde{R}}\mathbf{J}_{2,N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ {}_{\tilde{R}}\mathbf{J}_{N,0} & {}_{\tilde{R}}\mathbf{J}_{N,1} & {}_{\tilde{R}}\mathbf{J}_{N,2} & \cdots & {}_{\tilde{R}}\mathbf{J}_{N,N} \end{bmatrix} \tag{2.103}
$$

$$
= \begin{bmatrix} \frac{\partial\, {}_{\tilde{R}^0}\mathbf{F}_0}{\partial\, {}_{\tilde{R}^0}\mathbf{r}_0^{\mathrm{T}}} & \frac{\partial\, {}_{\tilde{R}^0}\mathbf{F}_0}{\partial\, {}_{\tilde{R}^1}\mathbf{r}_1^{\mathrm{T}}} & \frac{\partial\, {}_{\tilde{R}^0}\mathbf{F}_0}{\partial\, {}_{\tilde{R}^2}\mathbf{r}_2^{\mathrm{T}}} & \cdots & \frac{\partial\, {}_{\tilde{R}^0}\mathbf{F}_0}{\partial\, {}_{\tilde{R}^N}\mathbf{r}_N^{\mathrm{T}}} \\ \frac{\partial\, {}_{\tilde{R}^1}\mathbf{F}_1}{\partial\, {}_{\tilde{R}^0}\mathbf{r}_0^{\mathrm{T}}} & \frac{\partial\, {}_{\tilde{R}^1}\mathbf{F}_1}{\partial\, {}_{\tilde{R}^1}\mathbf{r}_1^{\mathrm{T}}} & \frac{\partial\, {}_{\tilde{R}^1}\mathbf{F}_1}{\partial\, {}_{\tilde{R}^2}\mathbf{r}_2^{\mathrm{T}}} & \cdots & \frac{\partial\, {}_{\tilde{R}^1}\mathbf{F}_1}{\partial\, {}_{\tilde{R}^N}\mathbf{r}_N^{\mathrm{T}}} \\ \frac{\partial\, {}_{\tilde{R}^2}\mathbf{F}_2}{\partial\, {}_{\tilde{R}^0}\mathbf{r}_0^{\mathrm{T}}} & \frac{\partial\, {}_{\tilde{R}^2}\mathbf{F}_2}{\partial\, {}_{\tilde{R}^1}\mathbf{r}_1^{\mathrm{T}}} & \frac{\partial\, {}_{\tilde{R}^2}\mathbf{F}_2}{\partial\, {}_{\tilde{R}^2}\mathbf{r}_2^{\mathrm{T}}} & \cdots & \frac{\partial\, {}_{\tilde{R}^2}\mathbf{F}_2}{\partial\, {}_{\tilde{R}^N}\mathbf{r}_N^{\mathrm{T}}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial\, {}_{\tilde{R}^N}\mathbf{F}_N}{\partial\, {}_{\tilde{R}^0}\mathbf{r}_0^{\mathrm{T}}} & \frac{\partial\, {}_{\tilde{R}^N}\mathbf{F}_N}{\partial\, {}_{\tilde{R}^1}\mathbf{r}_1^{\mathrm{T}}} & \frac{\partial\, {}_{\tilde{R}^N}\mathbf{F}_N}{\partial\, {}_{\tilde{R}^2}\mathbf{r}_2^{\mathrm{T}}} & \cdots & \frac{\partial\, {}_{\tilde{R}^N}\mathbf{F}_N}{\partial\, {}_{\tilde{R}^N}\mathbf{r}_N^{\mathrm{T}}} \end{bmatrix} . \tag{2.104}
$$

2.4.2. Constraints and Stabilizations

In order to enhance the quality and efficiency of the numerical solution, certain constraints and simplifications are introduced. First, unnecessary degrees of freedom are removed from the system as detailed in Section 2.4.2.1. Second, the numerical efficiency is further improved by the use of an approximate Jacobian and a decoupling of the spatial and temporal dimensions during each Newton iteration, see Section 2.4.2.2. Third, several constraints exist for the displacements so as to stabilize the Newton algorithm and prevent a divergence, see Section 2.4.2.3.

2.4.2.1. Eliminating degrees of freedom

As discussed before, certain displacements are restricted. The first node P_0 is fixed completely. All other nodes are fixed in longitudinal road direction ${}_{\tilde{R}}\mathbf{e}_x$ and may only be moved in ${}_{\tilde{R}}\mathbf{e}_y$- and ${}_{\tilde{R}}\mathbf{e}_t$-direction. In order to assure these constraints, constraint-forces $\mathbf{F}^{\mathrm{constr}}$ exist that cancel out the other forces in the direction of restricted movement. Since forces and displacements are zero for the restricted degrees of freedom, they can be eliminated completely from the equation and the corresponding rows and columns in the Jacobian can be removed. Thus, the dimension of the problem has been reduced from $3\,(N+1)$ to $2N$. The equation that is iteratively solved to find the equilibrium now reads

$$ {}_{\tilde{R}}\mathbf{F}\left({}_{\tilde{R}}\mathbf{z}_k\right) + {}_{\tilde{R}}\mathbf{J}\left({}_{\tilde{R}}\mathbf{z}_k\right){}_{\tilde{R}}\Delta\mathbf{z}_k = 0 \tag{2.105} $$

with

$$ {}_{\tilde{R}}\mathbf{z} = \begin{bmatrix} {}_{\tilde{R}^1}r_y^{\tilde{R}^{1*},P_1} \\ {}_{\tilde{R}^1}r_t^{\tilde{R}^{1*},P_1} \\ {}_{\tilde{R}^2}r_y^{\tilde{R}^{2*},P_2} \\ {}_{\tilde{R}^2}r_t^{\tilde{R}^{2*},P_2} \\ \vdots \\ {}_{\tilde{R}^N}r_y^{\tilde{R}^{N*},P_N} \\ {}_{\tilde{R}^N}r_t^{\tilde{R}^{N*},P_N} \end{bmatrix}, \qquad {}_{\tilde{R}}\mathbf{F} = \begin{bmatrix} {}_{\tilde{R}^1}F_y^{P_1} \\ {}_{\tilde{R}^1}F_t^{P_1} \\ {}_{\tilde{R}^2}F_y^{P_2} \\ {}_{\tilde{R}^2}F_t^{P_2} \\ \vdots \\ {}_{\tilde{R}^N}F_y^{P_N} \\ {}_{\tilde{R}^N}F_t^{P_N} \end{bmatrix}, \tag{2.106} $$

and

$$ {}_{\tilde{R}}\mathbf{J} = \begin{bmatrix} {}_{\tilde{R}}\mathbf{J}_{1,1} & {}_{\tilde{R}}\mathbf{J}_{1,2} & \cdots & {}_{\tilde{R}}\mathbf{J}_{1,N} \\ {}_{\tilde{R}}\mathbf{J}_{2,1} & {}_{\tilde{R}}\mathbf{J}_{2,2} & \cdots & {}_{\tilde{R}}\mathbf{J}_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ {}_{\tilde{R}}\mathbf{J}_{N,1} & {}_{\tilde{R}}\mathbf{J}_{N,2} & \cdots & {}_{\tilde{R}}\mathbf{J}_{N,N} \end{bmatrix}, \tag{2.107} $$

where each partial jacobian ${}_{\tilde{R}}\mathbf{J}_{\nu,\mu}$ is

$$ {}_{\tilde{R}}\mathbf{J}_{\nu,\mu} = \frac{\partial \mathbf{F}_\nu}{\partial \mathbf{r}_\mu^{\mathsf{T}}} = \begin{bmatrix} \dfrac{\partial\, {}_{\tilde{R}^\nu}F_y^{P_\nu}}{\partial\, {}_{\tilde{R}^\mu}r_y^{\tilde{R}^{\mu*},P_\mu}} & \dfrac{\partial\, {}_{\tilde{R}^\nu}F_y^{P_\nu}}{\partial\, {}_{\tilde{R}^\mu}r_t^{\tilde{R}^{\mu*},P_\mu}} \\ \dfrac{\partial\, {}_{\tilde{R}^\nu}F_t^{P_\nu}}{\partial\, {}_{\tilde{R}^\mu}r_y^{\tilde{R}^{\mu*},P_\mu}} & \dfrac{\partial\, {}_{\tilde{R}^\nu}F_t^{P_\nu}}{\partial\, {}_{\tilde{R}^\mu}r_t^{\tilde{R}^{\mu*},P_\mu}} \end{bmatrix}. \tag{2.108} $$

Note that the forces ${}_{\tilde{R}^\nu}\mathbf{F}^{P_\nu}$ depend on the position of several nodes $P_{\nu-2}, ..., P_{\nu+1}$

$$ {}_{\tilde{R}^\nu}\mathbf{F}^{P_\nu} = f\left(\mathbf{r}^{\tilde{R}^{\nu-2*},P_{\nu-2}},\, \mathbf{r}^{\tilde{R}^{\nu-1*},P_{\nu-1}},\, \mathbf{r}^{\tilde{R}^{\nu*},P_\nu},\, \mathbf{r}^{\tilde{R}^{\nu+1*},P_{\nu+1}}\right), \tag{2.109} $$

where the position vectors $\mathbf{r}^{\tilde{R}^{\nu*},P_\nu}$ can still be represented in any arbitrary reference frame. The forces ${}_{\tilde{R}^\nu}\mathbf{F}^{P_\nu}$ at node P_ν are formulated depending on position vectors which are also represented in the local reference frame $\uparrow\!\underrightarrow{\tilde{R}^{\nu}}$ of node P_{ν}

$$
{}_{\tilde{R}^{\nu}}\mathbf{F}^{P_{\nu}} = f\left({}_{\tilde{R}^{\nu}}\mathbf{r}^{\tilde{R}^{\nu-2*},P_{\nu-2}},\ {}_{\tilde{R}^{\nu}}\mathbf{r}^{\tilde{R}^{\nu-1*},P_{\nu-1}},\ {}_{\tilde{R}^{\nu}}\mathbf{r}^{\tilde{R}^{\nu*},P_{\nu}},\ {}_{\tilde{R}^{\nu}}\mathbf{r}^{\tilde{R}^{\nu+1*},P_{\nu+1}} \right). \tag{2.110}
$$

The partial derivatives from Equation 2.108 are therefore detailed

$$
{}_{\tilde{R}}\mathbf{J}_{\nu,\mu} = \frac{\partial \mathbf{F}_{\nu}}{\partial \mathbf{r}_{\mu}{}^{\mathsf{T}}} \quad = \quad \begin{bmatrix} \dfrac{\partial\, {}_{\tilde{R}^{\nu}}F_y^{P_{\nu}}}{\partial\, {}_{\tilde{R}^{\nu}}r_y^{\tilde{R}^{\mu*},P_{\mu}}} & \dfrac{\partial\, {}_{\tilde{R}^{\nu}}r_y^{\tilde{R}^{\mu*},P_{\mu}}}{\partial\, {}_{\tilde{R}^{\mu}}r_y^{\tilde{R}^{\mu*},P_{\mu}}} & \dfrac{\partial\, {}_{\tilde{R}^{\nu}}F_y^{P_{\nu}}}{\partial\, {}_{\tilde{R}^{\nu}}r_t^{\tilde{R}^{\mu*},P_{\mu}}} \\ \dfrac{\partial\, {}_{\tilde{R}^{\nu}}F_t^{P_{\nu}}}{\partial\, {}_{\tilde{R}^{\nu}}r_y^{\tilde{R}^{\mu*},P_{\mu}}} & \dfrac{\partial\, {}_{\tilde{R}^{\nu}}r_y^{\tilde{R}^{\mu*},P_{\mu}}}{\partial\, {}_{\tilde{R}^{\mu}}r_y^{\tilde{R}^{\mu*},P_{\mu}}} & \dfrac{\partial\, {}_{\tilde{R}^{\nu}}F_t^{P_{\nu}}}{\partial\, {}_{\tilde{R}^{\nu}}r_t^{\tilde{R}^{\mu*},P_{\mu}}} \end{bmatrix}, \tag{2.111}
$$

where

$$
\frac{\partial\, {}_{\tilde{R}^{\nu}}r_y^{\tilde{R}^{\mu*},P_{\mu}}}{\partial\, {}_{\tilde{R}^{\mu}}r_y^{\tilde{R}^{\mu*},P_{\mu}}} \quad = \quad \cos {}^{\tilde{R}_{\nu}}\psi^{\tilde{R}_{\mu}}. \tag{2.112}
$$

${}^{\tilde{R}_{\nu}}\psi^{\tilde{R}_{\mu}}$ denotes the angle between the reference frames $\uparrow\!\underrightarrow{\tilde{R}^{\nu}}$ and $\uparrow\!\underrightarrow{\tilde{R}^{\mu}}$.

2.4.2.2. Approximation of Jacobian and Decoupling of Dimensions

Instead of the correct Jacobian, also approximate Jacobian matrices can be used in order to speed up the algorithm, see for example [Broyden, 1967; Dennis Jr and More, 1977; Kelley, 2003]. It is important to note that the equilibrium itself remains unchanged and any solution acquired using an approximate Jacobian is also a valid solution for the original problem. An approximation can influence the convergence, however the reduction in computational complexity that can be achieved often leads to an overall speed-up in computation time.

Approximation of Obstacle Jacobian As described above, the ego-vehicle and the obstacles are modeled as rectangles. The obstacle force field is defined depending on the distance to the obstacle and depends on the position in x, y and t as well as on the relative orientation ${}^{V}\psi^{O_j}$ of ego-vehicle $\mathcal{V}$ and obstacle O_j. For the distance computation for each node P_i, a certain reference point $\hat{O}_j$ on each obstacle O_j is computed that has the lowest distance to the ego-vehicle $\mathcal{V}$ at P_i, see Figure 2.14. In order to enable the analytical calculation of the obstacle Jacobian $\mathbf{J}^{O_j}$, this reference point $\hat{O}_j$ is assumed to remain constant within one Newton iteration.

Decoupling of Time and Space In this case the Jacobian is approximated such that all cross-dimensional derivatives of forces are set to zero

$$\frac{\partial\ {}_{R}F_{x}^{P_\nu}}{\partial\ {}_{R}r_{t}^{R^*,P_\mu}} := 0, \frac{\partial\ {}_{R}F_{t}^{P_\nu}}{\partial\ {}_{R}r_{x}^{R^*,P_\mu}} := 0, \qquad \forall \quad \mu,\nu \in \{1,2,...,N\}\,. \tag{2.113}$$

This results in two decoupled N-dimensional equations for each iteration, while the overall trajectory optimization remains a coupled $2N$-dimensional problem in the augmented workspace $\mathbb{W}_T$.

As discussed before, see Section 2.4.2.1, Equation 2.108-2.111, the time-part Jacobian remains unaltered when transforming everything into $\uparrow\!\underrightarrow{\tilde{R}}$, while the transformed lateral-part Jacobian entries become

$${}_{\tilde{R}}\mathbf{J}_{\nu,\mu} = \frac{\partial\ {}_{\tilde{R}^\nu}\mathbf{F}_{\nu}}{\partial\ {}_{\tilde{R}^\nu}\mathbf{r}_{\mu}^{\mathsf{T}}} \cos{}^{\tilde{R}^\mu}\psi^{\tilde{R}^\nu}\,. \tag{2.114}$$

2.4.2.3. Restrictions of Displacements

There exist three kinds of restrictions for the trajectory during the iterative numerical solution with the Newton algorithm. First, there are forbidden areas in the augmented configuration space $\mathbb{C}_T = \langle x, y, t, \psi \rangle$, where the ego-vehicle would drive off the road $\mathcal{R}$ or collide with an obstacle $\mathcal{O}$. Second, there apply dynamic limitations for the ego-vehicle like limited allowable velocities or accelerations that pose restrictions on the usable portion of the augmented phase space $\mathbb{X}_T$. Finally, there exist node configurations that have an adverse effect on the numerical convergence of the Newton algorithm. All three kinds of configurations must be prevented. In order to do this, on the one hand the initial solution must already obey these constraints and on the other hand, the calculated displacements in each Newton iteration are checked and limited by scaling each displacement $\Delta\mathbf{z}_i$ by α_i

$$\Delta\hat{\mathbf{z}} = \boldsymbol{\alpha}\Delta\mathbf{z} = \begin{bmatrix} \alpha_1\Delta\mathbf{z}_1 \\ \alpha_2\Delta\mathbf{z}_2 \\ \vdots \\ \alpha_N\Delta\mathbf{z}_N \end{bmatrix}, \tag{2.115}$$

$$\boldsymbol{\alpha} = \operatorname{diag}(\alpha_i)\,, \qquad 0 < \alpha_i < 1 \quad \forall \quad i \in \{1,2,...,N\}\,. \tag{2.116}$$

This alters Equation 2.89 to

$$\mathbf{z}_{k+1} = \mathbf{z}_k + \Delta\hat{\mathbf{z}}_k. \tag{2.117}$$

The reduction factor $\boldsymbol{\alpha}$ regards all three kinds of restrictions mentioned above

$$\boldsymbol{\alpha} \;=\; \boldsymbol{\alpha}^{\text{ext}}\boldsymbol{\alpha}^{\text{int}}\boldsymbol{\alpha}^{\text{num}}, \tag{2.118}$$

where the external restrictions posed by the road $\mathcal{R}$ and the obstacles $\mathcal{O}$ are included in the external reduction factor $\boldsymbol{\alpha}^{\text{ext}}$, the dynamic limitations of e.g. velocity and accelerations are regarded by the reduction factor $\boldsymbol{\alpha}^{\text{int}}$, and the numerical restrictions produce $\boldsymbol{\alpha}^{\text{num}}$. These three factors are computed and applied after one another.

In computing each of these factors, there are two different options how to reduce the displacement to prevent forbidden movements. First, all displacements could be reduced by the same amount $\alpha_1 = \alpha_2 = ... = \alpha_N$ which only scales the length of the $2N$-dimensional displacement vector $\Delta\mathbf{z}$. Second, only the displacement of the nodes that would cause a violation could be reduced which allows to scale all displacements Δz_i individually by different α_i. This modifies both the size and the orientation of the displacement vector and affects the search direction in the $2N$-dimensional search space.

In most cases, the first method is to be preferred, because an alteration of the search direction of the Newton algorithm can cause a poor convergence. On the other hand, if the reduction is too large and scaling all displacements equally would lead to hardly any displacement, it might be advantageous to scale single displacements individually to increase convergence speed. Therefore, a threshold ϵ_{red} is chosen. If the necessary reduction would be less than this threshold, than the threshold reduction is applied and further necessary displacement reductions are applied only to the displacements of individual nodes,

$$\alpha_i \;=\; \begin{cases} \min\left(\epsilon_{\text{red}}, \alpha_{i,\max}\right) & \text{if} \quad \min\left(\alpha_{1,\max}, ..., \alpha_{N,\max}\right) < \epsilon_{\text{red}} \\ \min\left(\alpha_{1,\max}, ..., \alpha_{N,\max}\right) & \text{otherwise.} \end{cases} \tag{2.119}$$

$\alpha_{i,\max}$ denotes the least necessary reduction of displacement for the iteration such that the restrictions are obeyed. The reduction factors α_i are combined in a matrix

$$\boldsymbol{\alpha} \;=\; \operatorname{diag}\left(\alpha_i\right). \tag{2.120}$$

Prevent movement of nodes into forbidden areas by $\boldsymbol{\alpha}^{\mathbf{ext}}$ Even though the repelling forces become infinity as a node approaches a forbidden area, i.e., an obstacle or a border of the road, due to the local linear approximations the calculated displacements in one Newton iteration could still move a node into a forbidden area of the augmented configuration space $\mathbb{C}_T$ such that the ego-vehicle would be either off the road or in collision with an obstacle. Inside these forbidden areas there are no force fields defined. Therefore, in each iteration the calculated displacements must be checked and reduced in case a node would be moved into a forbidden area.

In case one or more nodes would leave the street, the maximum displacement $\Delta z_{i,\max}$ for these nodes is determined and the necessary reduction parameters $\alpha_{i,\max}^{\text{ext}}$ are computed and combined according to Equation 2.119.

If the displacement of the nodes would lead to a collision with an obstacle, it is more complicated to calculate the maximum allowable displacement $\Delta z_{i,\max}$. Therefore, here a numerical approach is taken and all α_i are iteratively decreased until no collision occurs anymore to determine

$$\alpha_{i,\max}^{\text{ext}} = \frac{\Delta z_{i,\max}}{\Delta z_i}. \tag{2.121}$$

Once the threshold ϵ_{red} is reached only the scaling parameters α_i^{ext} and $\alpha_{i-1}^{\text{ext}}$ are reduced further since the collision-checking at P_i depends on the position and orientation which is determined depending on the position of P_i and the previous node P_{i-1}.

Prevent violations of maximum velocity by $\boldsymbol{\alpha}^{\mathbf{int}}$ Besides environmental restrictions, there exist also dynamic limitations of the ego vehicle. Limitations can be implemented regarding the velocity, acceleration, and jerk, as well as regarding lateral acceleration. Especially critical is the limitation to an allowed velocity range, as this includes the restriction that the ego vehicle cannot drive backwards nor can it go back in time. This must be prevented, because no force fields are defined for these two cases. Displacements of nodes that would lead to violations of dynamic limitations are also restricted analogously to Equations 2.115 and 2.119.

Prevent badly conditioned configurations by $\boldsymbol{\alpha}^{\mathbf{num}}$ In addition to forbidden areas and dynamic limitations in $\mathbb{C}$ and $\mathbb{X}$, there exist unfavorable node configurations that are very likely to lead to a divergence of the Newton algorithm. These configurations must be prevented as well and displacements are checked and restricted analogously.

Formally, we demand that the force fields are all monotonically increasing or decreasing for all node coordinates. This means that the negative gradient on the force fields always points toward the zero which leads to a better convergence of the Newton algorithm.

Even though the used force fields are mostly designed to obey the condition of monotonic dependences, this is not true in some cases for the longitudinal acceleration force fields as a function of the temporal coordinates of the nodes.

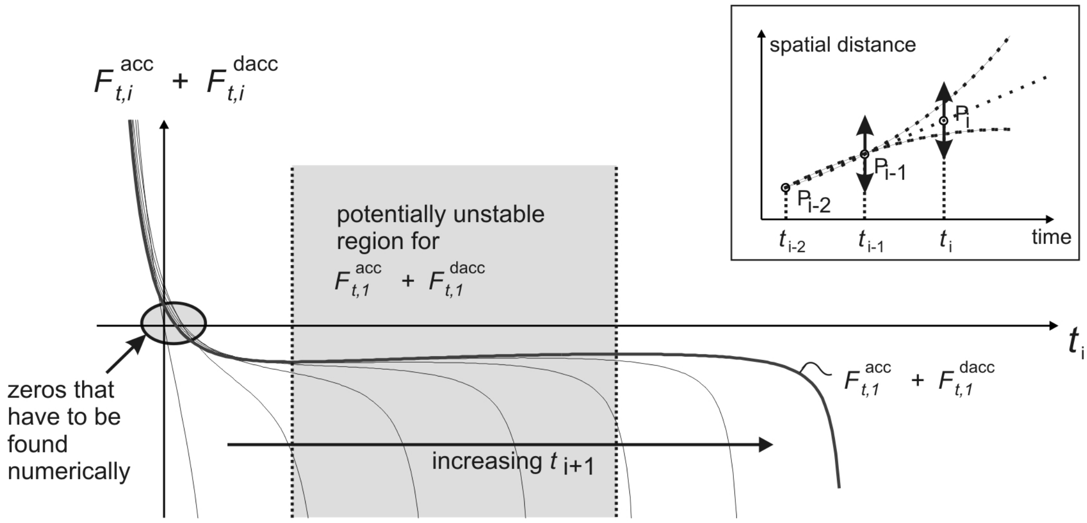


Figure 2.21.: Example of potentially unstable region for $\mathbf{F}_t^{\text{acc}} + \mathbf{F}_t^{\text{dacc}}$
The figure shows an array of force curves for $F_{t,i}^{\text{acc}} + F_{t,i}^{\text{dacc}}$ at P_i as a function of t_i for different t_{i-1}.

Figure 2.21 shows an array of force curves for $F_{t,i}^{\text{acc}} + F_{t,i}^{\text{dacc}}$ at P_i as a function of t_i for different t_{i-1}. As can be seen by this example, the force does not decrease monotonically. This creates a more difficult problem for the zero finding algorithm, since it uses local linear approximations which could, in this case lead to (even large) steps in the wrong direction.

This whole problem exists due to the discrete representation of the trajectory and the fact that the acceleration is approximated from finite differences from nodes in the augmented workspace $\mathbb{W}_T$. Figure 2.22 illustrates this problem with regards to the acceleration: A constant acceleration over three nodes is assumed, which effectively results in fitting a quadratic polynomial through three points. Even though there always exists a solution to this problem, the underlying assumption of constant acceleration does not always make sense. As can be seen in Figure 2.22, this could result in a motion where the vehicle is assumed to first drive from the second point P_{i-1} beyond the third point P_i with a constant negative acceleration and then finally drives backwards reaching the third point P_i. Similar problems exist for the jerk which result in fitting a cubic polynomial to four points.

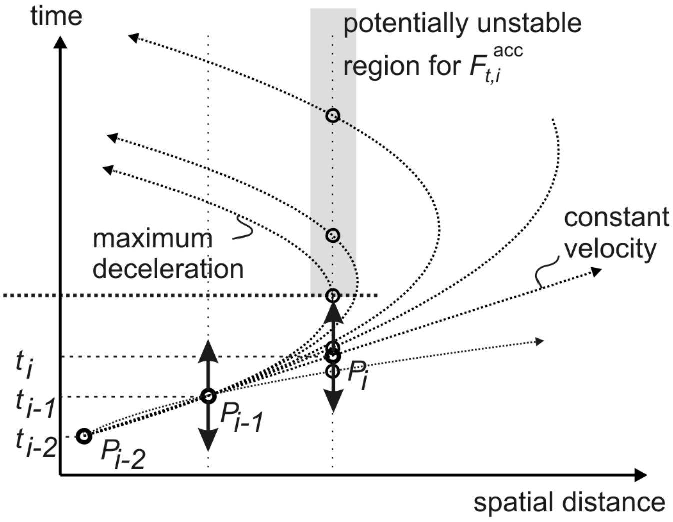


Figure 2.22.: Schematic analysis of badly conditioned node configurations for $\mathbf{F}_t^{\text{acc}}$
Due to the defined relations between $\mathbb{X}$ and $\mathbb{W}$ the acceleration a_i at P_i is determined by finite differences, which effectively results in fitting a quadratic polynomial through three points. If not prevented certain node configurations can yield unmeaningful results, where the direction of motion would be inverted between the second and third node.

Therefore, certain node configurations must be prevented to restrict the node configurations to monotonic regions. This can be achieved by analysis of local extrema and the derivation of appropriate limitations. The limitations are enforced by the application of the according scaling factor $\boldsymbol{\alpha}^{\text{num}}$ in Equation 2.118.

2.4.2.4. Stabilization of the Newton Algorithm

In addition to the specific modifications that have been discussed in the previous sections, more general measures can be taken to stabilize the Newton algorithm and ensure that the value of the function, in this case the forces, decreases in each step,

$$\|\mathbf{F}(\mathbf{z}_{k+1})\| = \|\mathbf{F}(\mathbf{z}_k + \boldsymbol{\Delta}\hat{\mathbf{z}}_k)\| \quad < \quad \|\mathbf{F}(\mathbf{z}_k)\|. \tag{2.122}$$

One of the most common methods to achieve this characteristic is to introduce Armijo step size control [Bonnans and Lemaréchal, 2006], given in algorithm 3. For each iteration, the Armijo rule calculates a secant $\mathbf{f}_{\text{secant}}$ at $\mathbf{F}(\mathbf{z}_k)$,

$$\mathbf{f}_{\text{secant}}(\mathbf{z}_k) \quad = \quad \mathbf{F}(\mathbf{z}_k) + \delta_{\text{Armijo}}\mathbf{J}(\mathbf{z}_k)\,\boldsymbol{\Delta}\hat{\mathbf{z}}_k \tag{2.123}$$

with $\delta_{\text{Armijo}} \in (0,1)$, and then reduces the step size $\|\boldsymbol{\Delta}\hat{\mathbf{z}}_k\|$ to $\|\boldsymbol{\Delta}\hat{\hat{\mathbf{z}}}_k\|$ by applying $\boldsymbol{\Delta}\hat{\hat{\mathbf{z}}}_k = \alpha_{\text{Armijo}}\boldsymbol{\Delta}\hat{\mathbf{z}}_k$ until the Armijo condition is met, i.e. until the forces decrease below this secant,

$$\|\mathbf{F}\left(\mathbf{z}_k + \boldsymbol{\Delta}\hat{\hat{\mathbf{z}}}_k\right)\| \quad < \quad \|\mathbf{F}\left(\mathbf{z}_k\right) + \delta_{\text{Armijo}}\mathbf{J}\left(\mathbf{z}_k\right)\boldsymbol{\Delta}\hat{\hat{\mathbf{z}}}_k\|. \tag{2.124}$$

The slope of the secant is $\mathbf{m}_{\text{secant}} = \delta_{\text{Armijo}}\mathbf{J}\left(\mathbf{z}_k\right)$ and thus given relative to the slope of the tangent $\mathbf{m}_{\text{tangent}} = \mathbf{J}\left(\mathbf{z}_k\right)$ with $0 < \delta_{\text{Armijo}} < 1$. The step size is reduced by iteratively

Algorithm 3 Armijo step size rule

```
$\alpha_{\text{Armijo}} \leftarrow 1$
while $\|\mathbf{F}(\mathbf{z}_k + \alpha_{\text{Armijo}}\boldsymbol{\Delta}\hat{\mathbf{z}}_k)\| \geq \|\mathbf{F}(\mathbf{z}_k) + \alpha_{\text{Armijo}}\delta_{\text{Armijo}}\mathbf{J}(\mathbf{z}_k)\boldsymbol{\Delta}\hat{\mathbf{z}}_k\|$
    $\alpha_{\text{Armijo}} \leftarrow \rho_{\text{Armijo}}\alpha_{\text{Armijo}}$
$\boldsymbol{\Delta}\hat{\hat{\mathbf{z}}}_k \leftarrow \alpha_{\text{Armijo}}\boldsymbol{\Delta}\hat{\mathbf{z}}_k$
```

decreasing the dampening factor α_{Armijo} by multiplying it with the reduction factor $0 < \rho_{\text{Armijo}} < 1$. The smaller ρ_{Armijo}, the quicker the Armijo condition is met, however, the step size $\boldsymbol{\Delta}\hat{\hat{\mathbf{z}}}_k$ might become unnecessarily small which leads to a slower convergence of the overall Newton method.

Since the norm of the force along the trajectory is a measure for the quality of the trajectory, the Armijo dampening guarantees that the quality of the trajectory monotonically increases with each step. Therefore, as a by-product, the trajectory optimization algorithm now has an anytime characteristic, meaning that it can be aborted anytime if no further computation time is available and the longer it runs the better the solution becomes. Equation 2.89 has been finally modified to read

$$\mathbf{z}_{k+1} \quad = \quad \mathbf{z}_k + \boldsymbol{\Delta}\hat{\hat{\mathbf{z}}}_k = \mathbf{z}_k + \alpha_{\text{Armijo}}\boldsymbol{\alpha}\boldsymbol{\Delta}\mathbf{z}_k. \tag{2.125}$$

The Armijo dampening is applied after the previously mentioned restrictions of displacements from Subsection 2.4.2.3, since the step size reduction from Subsection 2.4.2.3 could lead to a violation of the Armijo condition and therefore reduce the convergence. Further, all nodes must already be collision-free in order to be able to apply the Armijo step size rule.

2.5. Summary of Algorithm

The algorithms 4 and 5 summarize the virtual forcefield trajectory optimization algorithm as detailed in the previous sections of this chapter.

Algorithm 4 Virtual Forcefield Trajectory Optimization

```
ForcefieldTrajectoryOptimization($\mathcal{T}_{\text{init}}, \mathcal{R}, \mathcal{O}, \mathcal{C}$)
  Input:    Initial Trajectory $\mathcal{T}_{\text{init}}$,
            road model $\mathcal{R}$, obstacle description $\mathcal{O}$, dynamic constraints $\mathcal{C}$
  Output:   Planned Trajectory $\mathcal{T}$

$_{\tilde{R}}\mathbf{z}$ $\leftarrow$ SampleInitialTrajectory($\mathcal{T}_{\text{init}}, \mathcal{R}$)
$_{\tilde{R}}\mathbf{F}$ $\leftarrow$ CalculateForces($_{\tilde{R}}\mathbf{z}, \mathcal{R}, \mathcal{O}$)    [see Alg. 5]
$_{\tilde{R}}\mathbf{\Delta z}$ $\leftarrow$ $\infty$;   $\alpha_{\text{Armijo}} \leftarrow 1$;   $\boldsymbol{\alpha} \leftarrow \mathbf{I}$
while time is remaining and $\alpha_{\text{Armijo}} \cdot \boldsymbol{\alpha} \cdot {}_{\tilde{R}}\mathbf{\Delta z} \geq \epsilon_{\Delta z}$
    $_{\tilde{R}}\mathbf{J}$ $\leftarrow$ CalculateJacobian($_{\tilde{R}}\mathbf{z}, {}_{\tilde{R}}\mathbf{F}, \mathcal{R}$)    [see Sec. 2.4]
    $_{\tilde{R}}\mathbf{\Delta z}$ $\leftarrow$ Displacement($_{\tilde{R}}\mathbf{F}, {}_{\tilde{R}}\mathbf{J}$)    [see Eq. 2.101]
    $\boldsymbol{\alpha}$ $\leftarrow$ LimitDisplacement($_{\tilde{R}}\mathbf{z}, {}_{\tilde{R}}\mathbf{\Delta z}, \mathcal{R}, \mathcal{O}, \mathcal{C}$)    [see Sec. 2.4.2.3]
    $_{\tilde{R}}\mathbf{F}_{\text{next}}$ $\leftarrow$ CalculateForces($_{\tilde{R}}\mathbf{z} + \boldsymbol{\alpha} \cdot {}_{\tilde{R}}\mathbf{\Delta z}, \mathcal{R}, \mathcal{O}$)    [see Alg. 5]
    $\alpha_{\text{Armijo}}$ $\leftarrow$ 1
    while $\|{}_{\tilde{R}}\mathbf{F}_{\text{next}}\| \geq \|{}_{\tilde{R}}\mathbf{F} + \alpha_{\text{Armijo}} \cdot \delta_{\text{Armijo}} \cdot {}_{\tilde{R}}\mathbf{J} \cdot \boldsymbol{\alpha} \cdot {}_{\tilde{R}}\mathbf{\Delta z}\|$    [see Alg. 3]
        $_{\tilde{R}}\mathbf{F}_{\text{next}}$ $\leftarrow$ CalculateForces($_{\tilde{R}}\mathbf{z} + \alpha_{\text{Armijo}} \cdot \boldsymbol{\alpha} \cdot {}_{\tilde{R}}\mathbf{\Delta z}, \mathcal{R}, \mathcal{O}$)    [see Alg. 5]
        $\alpha_{\text{Armijo}}$ $\leftarrow$ $\rho_{\text{Armijo}} \cdot \alpha_{\text{Armijo}}$
    end
    $_{\tilde{R}}\mathbf{z}$ $\leftarrow$ $_{\tilde{R}}\mathbf{z} + \alpha_{\text{Armijo}}\boldsymbol{\alpha}{}_{\tilde{R}}\mathbf{\Delta z}$    [see Eq. 2.125]
    $_{\tilde{R}}\mathbf{F}$ $\leftarrow$ $_{\tilde{R}}\mathbf{F}_{\text{next}}$
end
$\mathcal{T}$ $\leftarrow$ ConstructTrajectory($_{\tilde{R}}\mathbf{z}, \mathcal{R}$)
```

Algorithm 5 Calculation of Forces for Trajectory Optimization

```
CalculateForces($_{\tilde{R}}\mathbf{z}, \mathcal{R}, \mathcal{O}$)
, Input:    $_{\tilde{R}}\mathbf{z} = [\mathbf{r}_1 \dots \mathbf{r}_N]^{\mathsf{T}}$ which represents the Trajectory $\mathcal{T}$,
            road model $\mathcal{R}$, obstacle description $\mathcal{O}$
  Output:   Forces $_{\tilde{R}}\mathbf{F}$ in local (road centerline) reference frames $\tilde{R}^i$

For $i = 1$ to $N$   $\mathbf{r}_i \leftarrow$ CoordinatesNodePi($_{\tilde{R}}\mathbf{z}, i$)   End
For $i = 1$ to $N$
    $R_l^i, R_r^i$  $\leftarrow$  RoadModel($\mathbf{r}_i, \mathcal{R}$)                              [see Sec. 2.1.1.1]
    $V_i^{\mathcal{R}}$  $\leftarrow$  PotentialFieldRoad($R_l^i, R_r^i$)                             [see Sec. 2.1.1.2]

    For all $\mathcal{O}_j \in \mathcal{O}$                                                    [see Sec. 2.1.2]
        If $\left|{}^{\tilde{R}}\psi^{O_j}\right| \leq \epsilon_\psi$                               [see Sec. 2.1.2.1, 2.1.2.3, 2.1.2.4]
            $d_i^{O_j}, \Delta t_i^{O_j}$  $\leftarrow$  InLaneExtrapolation($\mathbf{r}_i, O_j, \mathcal{R}$)
        Else                                                                    [see Sec. 2.1.2.2, 2.1.2.3, 2.1.2.4]
            $d_i^{O_j}, \Delta t_i^{O_j}$  $\leftarrow$  OutOfLaneExtrapolation($\mathbf{r}_i, O_j, \mathcal{R}$)
        End
        $V_i^{O_j} \leftarrow$  ObstaclePotentialField($d_i^{O_j}, \Delta t_i^{O_j}$)              [see Sec. 2.1.2.5]
    End
    $V_i^{\mathcal{O}}$  $\leftarrow$  $\sum_j V_i^{O_j}$                                            [see Eq. 2.59]

    $_{\tilde{R}}\mathbf{F}_i^{\text{ext}} \leftarrow$  $-\nabla\left(V_i^{\mathcal{O}} + V_i^{\mathcal{R}}\right)$                  [see Eq. 2.12, 2.13]

    $_{\tilde{R}}\mathbf{F}_i^{\text{int}} \leftarrow$  InternalForce($\mathbf{r}_{i-2}, \mathbf{r}_{i-1}, \mathbf{r}_i, \mathbf{r}_{i+1}$)          [see Sec. 2.2]

    $_{\tilde{R}}\mathbf{F}_i^{\text{prev}} \leftarrow$  PreviewForce($\mathbf{r}_{i-1}, \mathbf{r}_i, \mathcal{R}, \mathcal{O}$)                 [see Sec. 2.3]

    $_{\tilde{R}}\mathbf{F}_i$  $\leftarrow$  $_{\tilde{R}}\mathbf{F}_i^{\text{int}} + {}_{\tilde{R}}\mathbf{F}_i^{\text{prev}} + {}_{\tilde{R}}\mathbf{F}_i^{\text{ext}}$                 [see Eq. 2.2]
End
$_{\tilde{R}}\mathbf{F}$  $\leftarrow$  $[{}_{\tilde{R}}\mathbf{F}_1, \dots, {}_{\tilde{R}}\mathbf{F}_N]^{\mathsf{T}}$
```

2.6. Choice of Parameters

As presented, the virtual force field trajectory optimization depends a number of different parameters. While this constitutes the challenge to tune all parameters properly, this parameter sensitivity also presents a great potential to adapt the method to specific needs. One opportunity might be to make it driver-adaptive to yield certain characteristics a particular driver favors. This section presents some basic considerations and design guidelines to obtain a suitable parameter choice. The parameters can be grouped as shown in Table 2.1.

General planning parameters	
L	planning distance
ΔL	discretization of planned trajectory
N	number of nodes in trajectory

Force field parameters	
$k^{\partial\mathcal{R}_r}$	amplification of road force field of right road side
$k^{\partial\mathcal{R}_l}$	amplification of road force field of left road side
$k^{\mathcal{R}}$	amplification of road force field
$k^{\mathcal{O}}$	amplification of obstacle force field
k^{prev}	amplification of preview force field
l^{prev}	preview length for preview force field
k_y^{acc}	amplification of lateral acceleration force field
k_y^{dacc}	amplification of lateral jerk force field
k_t^{vel}	amplification of velocity force field
k_t^{acc}	amplification of longitudinal acceleration force field
k_t^{dacc}	amplification of longitudinal jerk force field

Newton parameters	
ϵ_{red}	reduction threshold
$\epsilon_{\Delta z}$	iteration abortion threshold

Table 2.1.: Parameters for virtual force field trajectory optimization

2.6.1. General Planning Parameters

There are a couple of parameters that affect the general characteristics of the planned and optimized trajectory such as its length and discretization.

2.6.1.1. Planning Horizon

The planning horizon L denotes the length of the planned trajectory along the course of the road. It is determined outside the trajectory optimization and depends on the distance or time interval that shall be planned. In practice it largely depends on the available sensors and data sources about the environment, since it does not seem prudent to execute or optimize planned maneuvers in a yet unknown region of the environment.

2.6.1.2. Discretization

The discretization ΔL of the optimized trajectory determines the number of nodes N. Currently, the nodes are equally distributed, resulting in

$$N = \frac{L}{\Delta L} \tag{2.126}$$

nodes that comprise the discretized trajectory in addition to the first node P_0 that is fixed. However, since the computation time depends approximately quadratically on the number of nodes, it appears prudent to save nodes where they are not needed. Therefore, the discretization could be adaptive to guarantee smoother reference for the controller at the beginning of a planned trajectory and finer discretization near obstacles in order to better avoid collisions. At these places a higher number of nodes could be placed. Lower discretizations could be chosen for the rest of the planned trajectory, especially further away from the current position, in order to save computational power.

2.6.2. Force Field Parameters

This section presents some basic considerations about the different force field parameters and their interactions.

2.6.2.1. $k^{\partial\mathcal{R}_r}$ vs. $k^{\partial\mathcal{R}_l}$

The coefficients $k^{\partial\mathcal{R}_q}$ can be scaled to shift the potential minimum of $V^{\mathcal{R}}$ to any desired course that should be taken in the absence of obstacles, for example to the middle of the right lane. In this case it can be assumed that the ego-vehicle's orientation does not differ much from that of the road. Therefore, as shown in Figure 2.23, the distances to the right and left road border $d^{\partial\mathcal{R}_r}$, $d^{\partial\mathcal{R}_l}$ are given by

$$d^{\partial\mathcal{R}_r} = \frac{1}{4}b - \frac{1}{2}w, \tag{2.127}$$

$$d^{\partial\mathcal{R}_l} = \frac{3}{4}b - \frac{1}{2}w, \tag{2.128}$$

where b denotes the road width and w stands for the width of the ego-vehicle. Therefore, the relation between the two roadside coefficients should be

$$\frac{k^{\partial\mathcal{R}_l}}{k^{\partial\mathcal{R}_r}} = \frac{3b - 2w}{b - 2w}. \tag{2.129}$$

The absolute value of each coefficient can then be defined via a single road potential coefficient $k^{\mathcal{R}}$ with

$$k^{\partial\mathcal{R}_l} := k^{\mathcal{R}}\left(3b - 2w\right) \tag{2.130}$$

$$k^{\partial\mathcal{R}_r} := k^{\mathcal{R}}\left(b - 2w\right). \tag{2.131}$$

The coefficient $k^{\mathcal{R}}$ can be used to scale the road potential versus other potentials and forcefields.

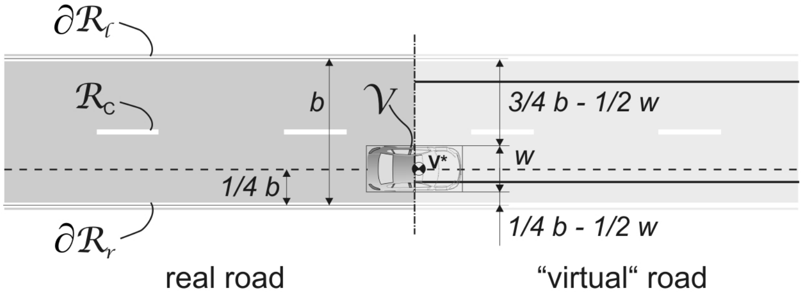


Figure 2.23.: Real and “virtual” road

Since the ego vehicle has a finite width w, it’s center of gravity (for which the trajectory is planned) cannot be allowed to touch the real road border, but must be restricted a narrower “virtual” road. If the vehicle is in the middle of the right lane of the real road, the center of gravity is not in the middle of the “virtual” road.

The relation between the two road side parameters could, however, be adjusted in order to achieve another preferred trajectory in the absence of obstacles. This could be useful for example to achieve lane changes or to perform other maneuvers that might be suggested from some other level of planning.

2.6.2.2. $k^{\mathcal{R}}$ vs. $k^{\mathcal{O}}$

The parameters $k^{\mathcal{O}}$ and $k^{\mathcal{R}}$ determine something akin to a threat level posed by the obstacles and roadsides. Since both kinds of collisions can have similar effects these two parameters can be chosen to be equal. If more information is available even each individual obstacle $\mathcal{O}_j$ could be scaled with regards to its hazard.

2.6.2.3. $k^{\mathcal{R}}$ vs. $k_y^{\mathbf{acc}}$

As an initial design consideration the road force field parameter $k^{\mathcal{R}}$ can be chosen relative to the lateral acceleration force field parameter k_y^{acc} such that the reduction of lateral acceleration is not acquired at the expense of cutting curves into oncoming traffic. For the parameter design we demand that in the absence of obstacles the equilibrium solution lies within the ego-vehicle's own lane and not cross into the oncoming traffic.

Even in the absence of obstacles the equilibrium solution cannot be found analytically. Therefore, this demand requires further approximations. First, the solution trajectory is assumed not to exceed the curvature of the road centerline anywhere. Because the internal force fields tend to reduce the curvature along the trajectory this assumption is plausible and usually met when no obstacles are present. Hence, the maximum road curvature κ_{max} along the planned trajectory is taken into account.

Second, at the apex of the planned trajectory where the lateral shift from the middle of the own lane is highest, the change of curvature of the planned trajectory is assumed to be negligible. This is usually the case, since the curvature increases until this point and decreases afterwards. Therefore, at the apex the change in curvature should be zero. Finally, the ego-vehicle is assumed to maintain its desired traveling velocity v_{des}.

In this case the force produced by the lateral acceleration force field reads

$$\begin{aligned} F_y^{\text{acc}} &= k_y^{\text{acc}} \kappa_i v_i^2 && (2.132) \\ &\leq k_y^{\text{acc}} \kappa_{max} v_{\text{des}}^2. && (2.133) \end{aligned}$$

In order to prevent the equilibrium solution from crossing into the other lane it is mandatory that the road force field and the lateral acceleration forcefield cancel each other out (latest) at the center of the road $y = y_C$

$$\mathbf{F}_y^{\mathrm{acc}}(y_C) = F^{\mathcal{R}}(y_C). \tag{2.134}$$

This results in a design choice for $k^{\mathcal{R}}$

$$k^{\mathcal{R}} = k_y^{\mathrm{acc}} \frac{\kappa_{max} v_{\mathrm{des}}^2}{2 - 2\frac{w}{b}}, \tag{2.135}$$

where w denotes the width of the ego-vehicle and b the width of the road.

Note that the preview force field $\mathbf{F}^{\mathrm{prev}}$ might have an adverse effect and may foster the cutting of curves. The size of its effect depends on several parameters, such as l_{prev}, k_{prev}, κ, b, and $\dot{\kappa}$. To counteract $\mathbf{F}^{\mathrm{prev}}$ in curves, $k^{\mathcal{R}}$ can be increased further to build a “reserve”. Regarding the empirical choice of l_{prev}, k_{prev} see Section 2.6.2.5.

2.6.2.4. k^{vel} vs. k_t^{acc}

The parameters k^{vel} and k_t^{acc} can be scaled versus one another to achieve a certain desired safety distance between the ego-vehicle and a leading obstacle at a constant velocity. For an obstacle with a constant velocity the equilibrium solution will follow the obstacle eventually also at the same constant speed and therefore at a constant distance. Since in this case the acceleration and jerk forces become zero, the equilibrium solution in time-direction is determined only by the travel velocity force field and the obstacle force field of the leading obstacle. These two must cancel each other out which leads to

$$k^{\mathrm{vel}}(v_i - v_0) = -k_t^{O_j} \frac{1}{t_i - \hat{t}_i^{O_j}}. \tag{2.136}$$

Substituting $\Delta s_{\mathrm{follow}} = v_i \left(t_i - \hat{t}_i^{O_j}\right)$ and $v_i = v^{O_j}$ in Equation 2.136 yields the resulting safety distance $\Delta s_{\mathrm{follow}}$

$$\Delta s_{\mathrm{follow}} = \frac{k_t^{O_j}}{k^{\mathrm{vel}}} \frac{v^{O_j}}{v_0 - v^{O_j}}. \tag{2.137}$$

This can be used to calculate the necessary relation between k^{vel} and $k_t^{O_j}$ to ensure a certain safety distance for vehicle following.

2.6.2.5. l_{prev} and k_{prev}

The preview forces depend on the preview distance l_{prev} and the amplification factor k_{prev}. The preview distance must be chosen such that no obstacle is "missed". This could be the case if the preview distance is too large. As a design goal the overshoot during alane change back to the right lane should be eliminates and the maximum lateral displacement when overtaking could be placed the middle of the obstacle. The amplification factor k_{prev} must be chosen empirically, balancing it versus the external and internal force fields.

As options for further improvement, the preview distance could be chosen adaptive depending on the distance to the next obstacle. Alternatively, multiple previews could be introduced.

CHAPTER 3

Trajectory Initialization

The virtual forcefield trajectory optimization algorithm presented in Chapter 2 optimizes a given trajectory and easily adapts an existing plan to smaller changes in the environment. As most numerical optimization algorithms it requires an initial guess. Since the algorithm for trajectory optimization is locally convergent, the final solution is always homeotopically equivalent to this initial guess. This is generally a desired characteristic since it makes the outcome of the optimization step controllable. On the other hand this property increases the importance of the trajectory initialization, since the fundamental type of maneuver is chosen here.

In an application to an advanced driver assistance system, the desired maneuver can be chosen depending on detected driver intentions, which could be detected for example with Artificial Neural Networks, Hidden Markov Models, Dynamic Belief Networks or others, [Dagli et al., 2003; Liu and Pentland, 2002; Oliver and Pentland, 2000; Zong et al., 2009]. However, most research in this area still focuses on a limited number of very short maneuvers, such as go straight / turn left / turn right or follow lane / change lane, and hardly predict longer or more complex maneuvers such as avoiding several obstacles or multiple lane changes.

For the demonstration in autonomous driving, in this chapter an alternative means of trajectory initialization is devised that does not require any input from a human driver. It could also be combined with driver intention recognition as a basis for maneuver arbitration between driver and assistance system, see for example [Löper et al., 2008]. The basic trajectory chosen together by human and machine can then be detailed by the introduced trajectory optimization approach and finally used for driver assistance in vehicle guidance.

The trajectory initialization must meet several demands: It must provide a viable initial solution to the short term trajectory planning problem. A single solution is desired in order to reduce computational demands. It would be generally infeasible to plan all possible homeotopically different trajectories, since their number grows exponentially with the number of obstacles. Further, the initial trajectory must obey a number of constraints to prevent collisions and to enhance the numerical stability of the trajectory optimization, see Section 2.4.2.3. Since the trajectory initialization has to be executed online and in frequent planning updates, anytime and incremental characteristics are of interest. Anytime algorithms aim at finding a solution quickly and spend the remaining time refining it. Therefore, they are well-suited when only limited time is allotted and the computational requirements vary depending on the scenario. Incremental algorithms try to achieve a speed-up by reusing previously planned trajectories or parts thereof. They are especially successful when changes are limited from one planning update to the next.

Based on the literature review in Chapter 1, a state-space oriented, A* derived search algorithm that includes the above characteristics is devised as trajectory initialization. Recalling this review, combinatorial approaches proved difficult and computationally expensive for more than two dimensions. Therefore, the class of sampling-based algorithms was selected. Among those, randomized planning approaches such as RRT-based approaches were tested and compared to A*-based algorithms in a related diploma thesis, [Hess, 2009]. In this comparison the implemented RRT-based approaches proved to be slower on average for most test scenarios. Moreover, the unpredictability of execution time and type of initial trajectory posed the most significant drawback, since for frequent online replanning only a limited time is available for each planning update and frequent changes in the type of initial trajectory are undesireable. Therefore, an A*-based approach was chosen, extended, and implemented, supported by a supervised diploma thesis, see [Hess, 2009; Hesse et al., 2010].

The basic structure of the employed algorithm is illustrated in Figure 3.1 and given in the form of pseudo-code in Algorithm 6: It is a direct application of AWA* to the domain of motion planning, assuming that transitions in the search space are defined by a finite set of motion primitive trajectories. (For some more basic explanations about A* in general see Chapter 1 or related literature such as [Pearl, 1985].)

None of the mentioned incremental algorithm extensions (e.g. LPA*, FSA* or Adaptive A*) for A* are used for several reasons: For the overhead of the incremental algorithm (esp. LPA*, FSA*) to pay off, changes in the environment have to be limited and occur towards the leafs of the search-tree, so that major parts of the previous search tree can be reused. These requirements are often not fulfilled, as obstacle extrapolations can vary significantly depending on current sensor inputs which might effect large portions of the search space. The combination of AWA* with Adaptive A* seems attractive, since it requires hardly any preprocessing overhead. Further it could potentially eliminate Adaptive A*'s dependency on increasing edge costs to maintain admissibility. AWA* could improve the solution incrementally until all decreased edge costs have been addressed. However, AWA* might be terminated prematurely and the learned heuristic $h'(x) = f^*_{t-1} - g_{t-1}(x)$ could then degrade to a non-admissible estimate, which could have unforeseen effects on the search behavior. Hence, for this work, incremental A* extensions are not applied.

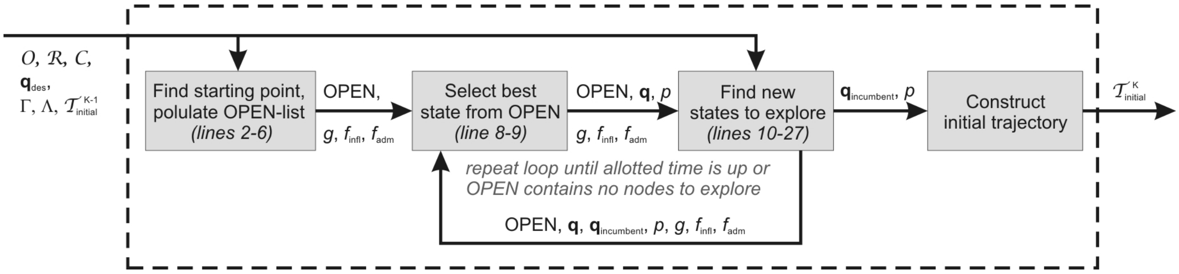

$\mathcal{O}$, $\mathcal{R}$, $\mathcal{C}$,
$\mathbf{q}_{\text{des}}$,
Γ, Λ, $\mathcal{T}^{K-1}_{\text{initial}}$
Find starting point, polulate OPEN-list (lines 2-6)
OPEN, g, f_{infl}, f_{adm}
Select best state from OPEN (line 8-9)
OPEN, $\mathbf{q}$, p
g, f_{infl}, f_{adm}
Find new states to explore (lines 10-27)
$\mathbf{q}_{\text{incumbent}}$, p
Construct initial trajectory
$\mathcal{T}^{K}_{\text{initial}}$
repeat loop until allotted time is up or OPEN contains no nodes to explore
OPEN, $\mathbf{q}$, $\mathbf{q}_{\text{incumbent}}$, p, g, f_{infl}, f_{adm}

Figure 3.1.: Blockdiagram Trajectory Initialization

As illustrated in Figure 3.1, the input to the AWA*-based trajectory initialization algorithm (Algorithm 6) is comprised of the desired vehicle state $\mathbf{q}_{\text{des}}$, the road model $\mathcal{R}$, the obstacles $\mathcal{O}$, existing dynamic constraints $\mathcal{C}$, the goal set Γ, and a set of motion primitives Λ. The result is the currently best solution vertex $\mathbf{q}_{\text{incumbent}}$ and a pointer path defined by p, from which the initial trajectory $\mathcal{T}^k_{\text{initial}}$ is then constructed.

In lines 2 to 6 the $OPEN$ list is populated with starting positions based on the given inputs, see Section 3.3. $g(\mathbf{q})$ denotes the cost for each vertex - note that in general each initial state also has an associated nonzero cost, 3.3. f_{infl} and f_{adm} represent solution cost estimates for each states, where the inflated cost estimate f_{infl} is allowed to overestimate the solution cost and the admissible cost estimate f_{adm} is guaranteed to underestimate the solution cost.

The main body of the algorithm, lines 8 to 27, is repeated until either the allotted time is up or $OPEN$ contains no nodes to explore (see line 7). In each cycle, first the best candidate $\mathbf{q}$ is selected from $OPEN$ (line 8) and tested if worth exploring (line 9). Then, in lines 10 to 27, new nodes $\mathbf{q}_{\text{new}}$ are discovered and added to $OPEN$. Note, that the inflated cost

Algorithm 6 AWA*-based anytime algorithm, compare [Hess, 2009].

```
UpdatePathAWA($\mathbf{q}_{\text{des}}, \mathcal{R}, \mathcal{O}, \mathcal{C}, \Gamma, \Lambda$)
  Input:   Desired vehicle state $\mathbf{q}_{\text{des}}$, road model $\mathcal{R}$, obstacle description $\mathcal{O}$,
           dynamic constraints $\mathcal{C}$, goal set $\Gamma$, set of motion primitives $\Lambda$.
  Output:  Currently best solution vertex $\mathbf{q}_{\text{incumbent}}$, pointer path def. by $p$
  $\mathbf{q}_{\text{incumbent}} \leftarrow \perp; f(\mathbf{q}_{\text{incumbent}}) \leftarrow \infty$
  $OPEN, g \leftarrow$ CalculateStartPositions($\Lambda, \mathbf{q}_{\text{des}}, \mathcal{R}, \mathcal{O}, \mathcal{C}$)
  For each $\mathbf{q} \in OPEN$
      $f_{\text{infl}}(\mathbf{q}) \leftarrow g(\mathbf{q}) +$ EstimateCost($\mathbf{q}, \Gamma$)
      $f_{\text{adm}}(\mathbf{q}) \leftarrow g(\mathbf{q}) +$ EstimateCostAdmissibly($\mathbf{q}, \Gamma$)
      $p(\mathbf{q}) \leftarrow \perp$
  End
  While time is remaining and $OPEN \neq \emptyset$
      $\mathbf{q} \leftarrow \text{argmin}\{f_{\text{infl}}(\mathbf{q}_i) : \mathbf{q}_i \in OPEN\}$
      If $f_{\text{adm}}(\mathbf{q}) < f(\mathbf{q}_{\text{incumbent}})$
          For all $\tau \in \Lambda$ with $\tau(0) \equiv \mathbf{q}$
              $\tau_{\text{fit}} \leftarrow$ transform $\tau$ in such a way that $\tau_{\text{fit}}(q_t) = \mathbf{q}$
              $\mathbf{q}_{\text{new}} \leftarrow \tau_{\text{fit}}(end)$
              $c \leftarrow$ CalculateCost($\tau_{\text{fit}}$)
              $h_{\text{infl}} \leftarrow$ EstimateCost($\mathbf{q}_{\text{new}}, \Gamma$)
              $h_{\text{adm}} \leftarrow$ EstimateCostAdmissibly($\mathbf{q}_{\text{new}}, \Gamma$)
              If $\mathbf{q}_{\text{new}} \notin OPEN \vee g(\mathbf{q}_{\text{new}}) > g(\mathbf{q}) + c$
                  If $g(\mathbf{q}_{\text{incumbent}}) > g(\mathbf{q}) + c + h_{\text{adm}}$
                      If Validate($\tau_{\text{fit}}, \mathcal{R}, \mathcal{O}, \mathcal{C}$)
                          $p(\mathbf{q}_{\text{new}}) \leftarrow \mathbf{q}$
                          $g(\mathbf{q}_{\text{new}}) \leftarrow g(\mathbf{q}) + c$
                          $f_{\text{infl}}(\mathbf{q}_{\text{new}}) \leftarrow g(\mathbf{q}_{\text{new}}) + h_{\text{infl}}$
                          $f_{\text{adm}}(\mathbf{q}_{\text{new}}) \leftarrow g(\mathbf{q}_{\text{new}}) + h_{\text{adm}}$
                          If $\mathbf{q}_{\text{new}} \in \Gamma$
                              $\mathbf{q}_{\text{incumbent}} \leftarrow \mathbf{q}_{\text{new}}$
                              $f(\mathbf{q}_{\text{incumbent}}) \leftarrow g(\mathbf{q}_{\text{new}})$
                          Else
                              $OPEN \leftarrow OPEN \cup \{\mathbf{q}_{\text{new}}\}$
                          End
                      End End End End End
```

estimate f_{infl} is used for the selection of $\mathbf{q}$ and therefore determines the "search direction", while the admissible cost estimate f_{adm} serves to eliminate nodes which are known to have a higher cost than already known solutions ($f(\mathbf{q}_{\text{incumbent}})$).

In line 11 all motion primitives τ from Λ are transformed to "fit" the selected node $\mathbf{q}$. A new node $\mathbf{q}_{\text{new}}$ is found at the endpoint of every transformed primitive τ_{fit} (line 12). In lines 13 to 15 the cost c from $\mathbf{q}$ to $\mathbf{q}_{\text{new}}$ and the estimated remaining costs h_{infl} and h_{adm} to the goal region Γ are calculated.

The new node $\mathbf{q}_{\text{new}}$ is only regarded further, if it has not been in $OPEN$ before or in case a known node has now been reached at a lower cost ($g(\mathbf{q}) + c$) than before ($g(\mathbf{q}_{\text{new}})$), line 16. In addition, in line 17 it is tested whether there is a chance that the new node can improve an already existing solution $\mathbf{q}_{\text{incumbent}}$, i.e. whether the underestimated cost to the goal region via the new node ($g(\mathbf{q}) + c + h_{\text{adm}}$) is lower than the cost of the current solution $g(\mathbf{q}_{\text{incumbent}})$. As last (because most time consuming) test, the trajectory τ_{fit} is checked to stay on the road $\mathcal{R}$, not to hit any obstacles $\mathcal{O}$ and to obey all dynamic constraints $\mathcal{C}$ (line 18).

If the new node passes all of the above tests, a pointer $p(\mathbf{q}_{\text{new}})$ is set to the predecessor node $\mathbf{q}$ (line 19) to be able to reconstruct the solution trajectory and the required costs are recorded (lines 20-22). Here, the goal-criterion is tested directly upon discovery (line 23), differing from A*'s delayed termination and testing upon selection from $OPEN$. In case the new node $\mathbf{q}_{\text{new}}$ lies in the goal region Γ, the previous solution $\mathbf{q}_{\text{incumbent}}$ is replaced (lines 24, 25), since the new one must be better (because it passed the test in line 17). Otherwise $\mathbf{q}_{\text{new}}$ is added to the $OPEN$ list for further exploration (line 27).

The algorithm consists of several subproblems, which are detailed in the following sections. Section 3.1 defines an adequate planning space that allows to plan useful initial trajectories that regard all necessary constraints. Section 3.2 discusses how to determine the goal states. Section 3.3 details considerations concerning the selection of the starting states and discusses why it was decided not to select the vehicle's actual position as starting state. Sections 3.4 and 3.5 describe the constraint validation and the used cost functions that determine which initial trajectory is chosen.

3.1. Definition of the Search Space

The performance of the algorithm depends heavily on the selection of the searched dimensions. As deliberated in the literature review in Chapter 1 and further detailed in Annex

A, most state space search-algorithms used during the DARPA Urban Challenge (DUC) did not include the longitudinal velocity as full dimension. Instead, a decoupled planning approach was mostly sought: First, a path through a static environment was planned and afterwards, the velocity along the path was calculated to avoid also mobile obstacles. Such a decoupled planning is less flexible than an integrated path and velocity planning approach and could yield very unfavorable maneuvers if applied to regular road traffic. For example, an overtaking maneuver would be planned for all preceding vehicles, if they were considered as static obstacles in the path planning step.

In this case, the task is to create an initial trajectory for the virtual forcefield optimization which represents trajectories as discrete nodes in $\mathbb{W}_T : \langle x, y, t \rangle$. Planning the initial trajectory also in x, y, t allows an implicit velocity planning. However, in order to generate smooth initial velocity profiles it is necessary to add the longitudinal velocity v as an extra dimension. Otherwise two states at the same place (x, y) and time t would be considered identical even if they were reached at different velocities, and this velocity would not be regarded in the selection of the subsequent trajectory element. Analogously, the orientation ψ is added as fifth dimension to enforce finite changes in orientation along the trajectory. Furthermore, the addition of v and ψ as search dimensions facilitates the restriction to viable orientations and velocities. It is important to keep in mind that the initial trajectory merely provides a starting guess for the subsequent trajectory optimization. Further optimization with regards to acceleration a_x, a_y and jerk $\dot{a}_x, \dot{a}_y$ is not yet necessary at this point but left to the trajectory optimization. Therefore, the trajectory initialization is performed within the searchspace $\mathbb{S} : \langle x, y, t, \psi, v \rangle$.

3.1.1. Search Space Discretization

The performed literature review on the search space discretization for A*-based algorithms did not reveal any methods for integrated planning of acceleration and steering. Therefore, it was necessary to modify one of the lower dimensional methods. Field-D* is likely to cause exceedingly high interpolation costs on the cell-border when extended to higher dimensions ($n > 2$), since it is necessary to find the minimum of an interpolated $n-1$-dimensional cost-function. Theta* itself relies only on a regular eight-connected grid, even though it's parent-connection attempts remain generally applicable to other search space discretization. If needed, it could be used to reduce the stair-effects that result from small motion primitive sets. State-Lattice and Hybrid-A* algorithms seem both applicable. However, unfortunate combinations of exploration order and obstacle configurations might cause a Hybrid-A* search to become stuck. Therefore, a State-Lattice primitive trajectory set is used. The form of the primitive trajectories is independent of the absolute position in x, y and the time t. It only depends on the vehicle's orientation ψ and velocity v, the primitive trajectory set has to contain trajectories for all possible combinations of v and ψ. The original motion primitive set generation algorithm would therefore calculate all trajectories between any start position $[0, 0, i \cdot \Delta v, j \cdot \Delta\psi, 0]$ and any goal position $[k \cdot \Delta x, l \cdot \Delta y, m \cdot \Delta v, n \cdot \Delta\psi, o \cdot \Delta t]$, which seems unfeasible.

Rather, this huge, high-dimensional set is reduced by limiting goal velocities and orientations to small sets around the initial values. As suggested in [Ferguson et al., 2008], the discretization of the orientation is selected depending on the velocity. Moreover, the generated primitive trajectories are limited to a maximum length in the time dimension. Only trajectories that obey the dynamic constraints are allowed. All generated trajectories are further tested for path-decomposition and path-equivalence to eliminate unnecessary motion primitives.

In addition to the mentioned state-lattice approach that is used in this work, the potential edges could also be generated online by sampling the (a,δ)-command space using the same local planning method, see [Hesse et al., 2010].

3.1.2. Local Planning Method

The generation of a state lattice requires a local planning method (LPM) that creates primitive trajectories from an initial state $\mathbf{q}_i = [0, 0, 0, \psi_i, v_i]^\mathsf{T}$ to a goal state $\mathbf{q}_g = [x_g, y_g, t_g, \psi_g, v_g]^\mathsf{T}$ in an obstacle-free environment, while respecting all kinematic and dynamic constraints. In general, this involves the solution of a two point boundary value problem. For constraints in the form of differential equation, mostly this necessitates a numerical solution. However, in this case a precise modeling of vehicle dynamics is not necessary. The important constraints rather arise from considerations concerning the numerical stability of the subsequent virtual forcefield optimization. Therefore, a simple nonholonomic vehicle model is applied, as already used for modeling other traffic participants in Chapter 2.

The control input comprises the bounded longitudinal acceleration $a_{\text{min}} \leq a \leq a_{\text{max}}$, as well as the limited steering angle $|\delta| < \delta_{\text{max}}$. This simple model allows a closed form solution for the mentioned boundary value problem similar to Dubins curves. Dubins creates minimum length paths that consist of circular arcs with upper-bounded curvature and straight lines, [Dubins, 1957].

However, if the lateral acceleration is restricted, the minimum turning radius becomes velocity dependent. Instead, $\mathbf{q}_i$ and $\mathbf{q}_g$ are connected by maximum-curvature paths which allow the highest possible velocity. This simplifies to check whether dynamic constraints can be met. Then the velocity profile is calculated to meet the boundary conditions at $\mathbf{q}_i$ and $\mathbf{q}_g$.

The construction of maximum curvature paths excludes the combination of curved with straight pieces. Even further, since only forward motion is of interest, the construction of the paths is limited to a single right and a single left turn, respectively denoted "R" and "L". If the curvature falls below the minimum curvature caused by the limitations in the steering angle, the goal configuration $\mathbf{q}_g$ is declared unachievable.

Figure 3.2 shows the construction of an example "LR" path: Relative to the start configuration $\mathbf{q}_i$, the target configuration is at $\mathbf{q}_g = [x, y, \psi]^{\mathsf{T}}$. Knowing that the distance of the centers of rotation A and B is given as $\| \mathbf{r}^{A,B} \| = 2r$ it can be deducted that

$$(2 \cdot r)^2 = (x + r \cdot \sin(\psi))^2 + (y - r \cdot (\cos(\psi) + 1))^2. \tag{3.1}$$

Hence, the turning radius r for LR is

$$r = -p/2 + \sqrt{(p/2)^2 + q} \tag{3.2}$$

with

$$p = 2\frac{x \cdot \sin(\psi) - y \cdot (\cos(\psi) + 1)}{\sin^2(\psi) + (\cos(\psi) + 1)^2 - 4}, \tag{3.3}$$

$$q = \frac{x^2 + y^2}{\sin^2(\psi) + (\cos(\psi) + 1)^2 - 4}. \tag{3.4}$$

The length l of the "LR" path is

$$l = r\,(\alpha + \beta) \tag{3.5}$$

with

$$\alpha = \frac{\pi}{2} + \gamma, \tag{3.6}$$

$$\beta = \alpha - \psi. \tag{3.7}$$

The “RL” path is constructed analogously. In case both “LR” and “RL” paths are possible, the one of shorter length is selected. The selected options depending on the relative goal orientation are depicted in Figure 3.2 on the right.

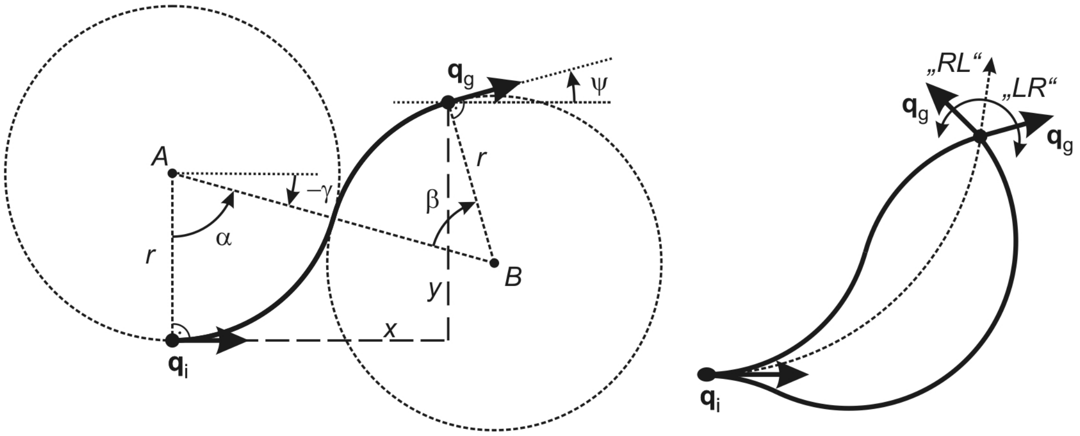


Figure 3.2.: Construction of maximum turning radius curve
The maximum turning radius path between an initial state $\mathbf{q}_i$ and a goal state $\mathbf{q}_g$ consists of a maximum of two concatenated circular arcs, one turning left (“L”) and one right (“R”). (Figure left shows the construction of the “LR” example.) Depending on the relative orientation at the goal, either the combination “LR” or “RL” is to be preferred, see Figure right.

The velocity profile has to fulfill the given boundary conditions regarding the initial and goal velocity v_i, v_g and the duration t_g for the given path length l

$$s(0) = 0, \qquad s(t_g) = l, \tag{3.8}$$

$$\dot{s}(0) = v_i, \qquad \dot{s}(t_g) = v_g, \tag{3.9}$$

where s is the arc length along the path. To satisfy these constraints, a simple triangluar velocity profile is chosen with two phases of constant acceleration. At a turn-around time t_c, the acceleration direction is inverted as illustrated in Figure 3.3. The acceleration is therefore given as

$$\ddot{s}(t) = \begin{cases} a, & 0 < t \leq t_c \\ -a, & t_c < t \leq t_g \end{cases} \tag{3.10}$$

with

$$a = \frac{v_g - v_i}{2t_c - t_g}. \tag{3.11}$$

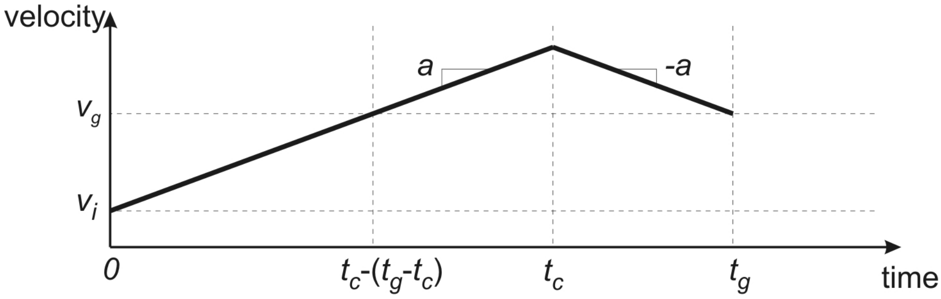


Figure 3.3.: Velocity profile for lattice generation
The velocity profile for each motion primitive consists of a triangular profile with piece-wise constant acceleration $\ddot{s} = \pm a$ to satisfy the goal velocity v_g for $t = t_g$.

The turn-around time t_c can be solved by solving $s(t_g) = l$

$$v_i t_g + a t_c \cdot (t_g - t_c) - 0.5a \cdot (t_g - t_c)^2 = l, \tag{3.12}$$

resulting in

$$t_c = -p/2 \pm \sqrt{(p/2)^2 + q} \tag{3.13}$$

with

$$p = -\frac{l - t_g v_g}{(2t_c - t_g) \cdot (v_g - v_i)}, \tag{3.14}$$

$$q = \frac{2l t_g - t_g^2 v_g - t_g^2 v_i}{(4t_c - 2t_g) \cdot (2v_g - 2v_i)}. \tag{3.15}$$

At least one of the two possible turn-around times must be in $[0, t_g]$. If both are valid, the velocity profile with the lower acceleration is chosen to minimize longitudinal acceleration.

3.1.3. A Note on Completeness

In [Pivtoraiko and Kelly, 2005*b*] a similar, yet lower dimensional state-lattice approach is taken. Therein, motion primitives are generated for the primitive set until it converges, i.e., until no more motion primitives can be found that cannot be decomposed into already

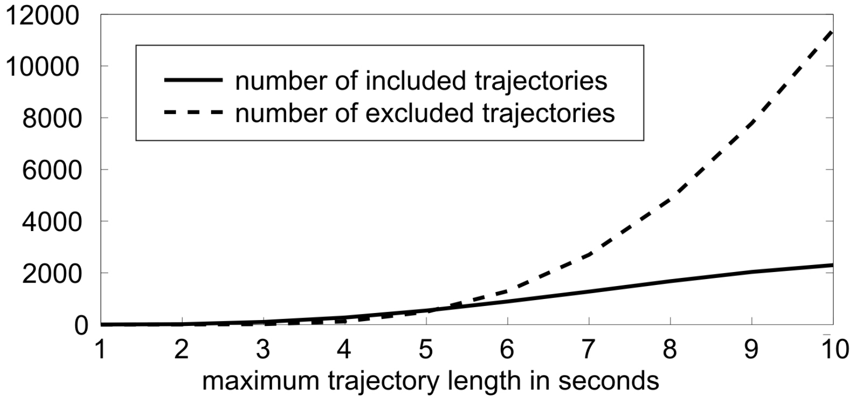


Figure 3.4.: Convergence of motion primitives as investgated in [Hess, 2009]
The figure shows the number of included motion primitives versus the number of superfluous trajectories that could be excluded, because they could be decomposed into smaller, already existing primitives. (Motion primitive generation was executed with x- and y-discretization $1m$, angular discretization $1/16 \cdot \pi$ in angular range $[-1/4, +1/4] \cdot \pi$, time discretization $0.5s$, and constant velocity $10m/s$.)

existing ones. It is then believed that the resulting set of motion primitives guarantees a form of completeness, where in the absence of obstacles any grid position is reachable.

In order to test whether the devised motion primitive set in the defined higher dimensional search space also converges, a motion primitive generation is started with the following boundary conditions and parameters: start and goal velocities equal at $v_i = v_g = 10\text{m/s}$, maximum allowed offset in y-direction $y_{\max} = 10\text{m}$ angular resolution $\Delta\psi = \pi/16$, angular range $\psi \in [-1/8\pi, +1/8\pi]$, resolution of the grid is $\Delta x = \Delta y = 1\text{m}$ and $\Delta t = 0.5\text{s}$.

Figure 3.4 displays the number of included (non-decomposable) and excluded (decomposable) trajectories. While the number of included trajectories increases very quickly, the number of the excluded trajectories seems to be level off. However, for any tested maximum trajectory length still a number of new, non-decomposable trajectories could be found. These non-decomposable trajectories seem to occur at the dynamic limits, where all potential composite trajectories just overstep the given dynamic constraints.

For the trajectory initialization algorithm, it was decided to generate a motion primitive set with a fixed maximum length in time as a compromise between efficiency and resolution. Therefore, it was not possible to guarantee completeness for the primitive set. The use of a limited time-length state-lattice can though be treated as another form of resolution-completeness.

3.2. Definition of the Goal Region

The goal region Γ is defined as a receding horizon instead of a fixed global (navigation) goal, as it is common for tactical layer planning algorithms. In general, the receding horizon could be defined in terms of temporal or spatial distance. A fixed time-horizon, however, creates problems to find appropriate cost functions: While a time-based cost measure would become impossible, path length-based measure would then always prefer short paths with very low velocity over longer (and faster) ones. Hence, a fixed time horizon might be applicable for planning "last chance" maneuvers in emergency situations but is not suitable for continuous driving in general.

Therefore, the goal region Γ is defined by a "finish line" in the x-y-plane, placed a certain planning distance ahead of the vehicle's current position. Such a planar horizon fits well to existing perception ranges of vehicle sensors and allows free maneuvers within the given distance. The limited planning distance further restricts the number of nodes in the subsequent forcefield optimization and thus prevents exceedingly high execution times.

As alternatives to a moving horizon, a higher level planner could also select a gap in traffic or a target velocity to define the goal region to create for example emergency evasion or stopping maneuvers.

3.3. Definition of the Starting Region

At first glance, it seems to be most obvious to select the vehicle's current position as starting point for the motion planning. This is indeed suitable when a single trajectory is planned in static situations and re-planning is not necessary. When the vehicle is moving while the motion planner is executed, the current vehicle state might not be the optimal start state anymore. Rather, the predicted vehicle state at the point of time of the planning algorithm's deadline might be chosen.

In the intended application to provide guidance to a vehicle through moving traffic, frequent re-planning is essential to account for changes in the dynamic environment. In this case, planning from the predicted vehicle state is not always the best option either. Here, the following properties shall be required: On the one hand, the selection of a starting position should be flexible enough that plans differing from an old plan can be developed in order not to hinder the reaction to unforeseen environment or vehicle developments. In this case it seems prudent to consider the predicted state of the vehicle as start state.

On the other hand, a previous plan or at least its first part may remain usable if changes in the environment are limited. If additionally the vehicle's position did not diverge too far from the plan, the starting state of a new search should be selected as to ensure consistency with the previous plan. Essentially, during the trajectory initialization is has to be decided whether to follow up the previous plan or to develop a completely new one.

In case the former option is selected, all inputs to the trajectory following controller should be as continuous as possible, since sudden discontinuities may adversely affect both stability and comfort. This becomes obvious when the closed loop of trajectory re-planning and trajectory following controller as displayed in Figure 3.5 is considered. Moreover, it becomes evident that the selection of the vehicle state as start state for each re-planning would effectively cancel out any control deviation. As the re-planning frequency approaches the execution frequency of the outer control loop of the trajectory following controller, the vehicle becomes uncontrollable.

For this reason, the desired vehicle state $\mathbf{q}_{\text{des}}$ according to the previously planned initial trajectory $\mathcal{T}^{K-1}$ is calculated for the point of time corresponding to the planning algorithm's deadline

$$\mathbf{q}_{\text{des}} = \mathcal{T}^{K-1}(t+T) = \left[x_{\text{des}}, y_{\text{des}}, t_{\text{des}}, \psi_{\text{des}}, v_{\text{des}}\right]^{\mathsf{T}}, \tag{3.16}$$

where T denotes the length of the re-planning interval.

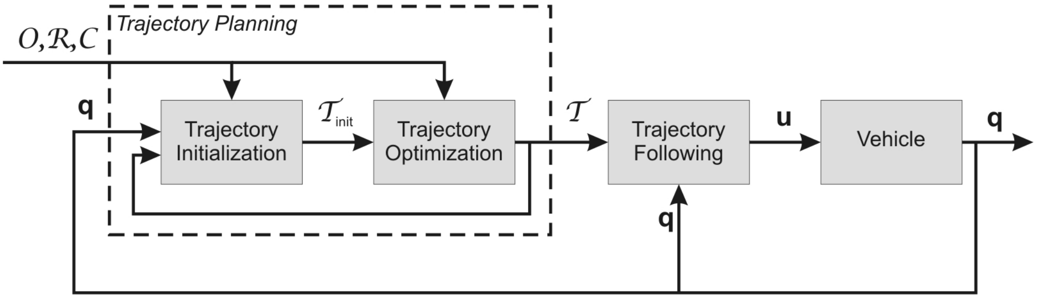


Figure 3.5.: Closed loop of trajectory planning, trajectory following and vehicle
The selection of the vehicle state as starting state would cancel out the control deviation and might introduce instabilities.

In order to allow variable start states that are selected due to a certain cost function, often the search direction is inverted and the plan is developed from a static goal state towards the mobile robot. However, since a moving target region rather than a static target point is used for the application to road driving, the inversion of the search direction is not practical.

Therefore, a method for the selection of start states is developed, see Algorithm 7, which is able to consider every possible state on a motion primitive. It is essential that the start state is not restricted to the (coarse) grid in which the search space is discretized, but that it can be connected to a grid point to assure the compatibility to the devised trajectory initialization algorithm. This connection shall be achieved by the use of partial motion primitives.

First, for each motion primitive τ_i in the used lattice Λ, reference states are identified such that the motion primitive ends at a time that belongs to the time grid. The relative time from the beginning of each motion primitive τ_i to the reference states on τ_i is given by

$$t_{\mathrm{rel},j} \quad = \quad j \cdot \Delta t + \mathrm{rem}(t_{\mathrm{des}}, \Delta t) \ \forall \ j = 0, 1, 2 ... \text{ with } t_{\mathrm{rel}} \in \mathrm{dom}(\tau). \tag{3.17}$$

Depending on its length in time and the grid discretization Δt, each motion primitive contains several reference states $\tau_i(t_{\mathrm{rel},j})$, see Algorithm 7, line 4.

The start states $\mathbf{q}_{\mathrm{start}}$ are generates from all reference states on the motion primitives as follows, (see also Figure 3.6): First, the motion primitives τ are translated to τ_{fit} such that the reference states are located at the x, y-position of the desired start state $\mathbf{q}_{\mathrm{des}}$ (Figure 3.6a and Algorithm 7, line 5). Second, the endpoints of the motion primitives are snapped to the nearest grid point, translating the primitives τ_{fit} to τ_{grid} (Figure 3.6b and line 6 f).

The grid points at the end of the translated motion primitives represent the start states $\mathbf{q}_{\mathrm{start}}$. They become elements of the set X_s that is used to initiate the $OPEN$ list if the corresponding motion primitive τ_{grid} fulfill the given constraints, e.g. to be collision-free and remain on the road. Since every start state $\mathbf{q}_{\mathrm{start}}$ can exist multiple times for different τ_{grid}, only the ones with the lowest cost are added to X_s, (see also Figure 3.6c) and Algorithm 7, line 9-(line 10)). The cost represents several, linearly weighted aspects and is defined as

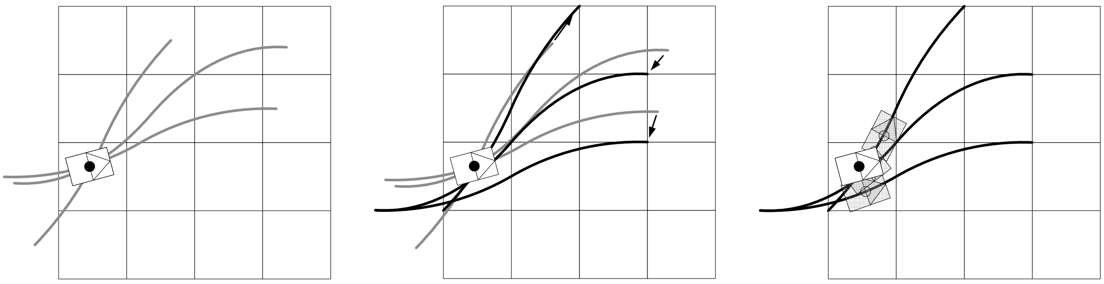
$$g_0(\mathbf{q}_{\mathrm{start}}) = w_c \cdot \frac{t_{\mathrm{end}} - t_{\mathrm{rel}}}{t_{\mathrm{end}}} \cdot c(\tau) + w_p \cdot c_p(\tau) + w_\varepsilon \cdot c_\varepsilon(q, \tau(t_{\mathrm{rel}})), \tag{3.18}$$

where w_c, w_p, and w_ε represent the weighting factors for the added costs.

$c(\tau)$ denotes the general cost measure that is used in the subsequent trajectory search algorithm. It is applied proportional (regarding the time dimension) to the used part of the motion primitive, $c(\tau)\frac{t_{\text{end}}-t_{\text{rel}}}{t_{\text{end}}}$.

The cost c_p favors motion primitives that create trajectories consistent with the previous one in order to prevent oscillations between different, equally suitable trajectories. The current motion primitive is compare with the first two segments of the previous trajectory. A diverging primitive receives c_p as a penalty.

$c_\varepsilon(q, \tau(t_{\text{rel}}))$ is the control error cost that penalizes the distance between the start state $\mathbf{q}_{\text{start}}$ and the desired vehicle state $\mathbf{q}_{\text{des}}$ at the point of time of the algorithm's deadline (see Figure 3.6c). This distance causes undesired discontinuities in the control error for the trajectory following controller. the control error cost is currently implemented as the L^2 norm in the dimensions x, y, ψ and v.



(a) Place arcs so that error in x, y, t is zero

(b) Snap endpoints to grid

(c) Evaluate position error

Figure 3.6.: Selection of start-vertices for a grid-based planner, compare [Hess, 2009].

3.4. Constraint Validation

The validity of potential trajectory parts is determined by a separate module that ensures that all constraints are satisfied. As mentioned, these constraints consist of the requirements to drive on the road, to evade static and mobile obstacles, and to respect kinodynamic constraints on the motion to foster the numerical convergence of the subsequent trajectory optimization. While the latter can be easily enforced already within the local planning method proposed in Section 3.1.2, this section therefore focuses on the question how the obstacle-constraints can be tested.

The obstacles can be represented in different ways. Most commonly, they are represented either explicitly, in an object oriented fashion or an implicit, approximate grid-based ap-

Algorithm 7 Selection of start states in accordance with a set of motion primitives, compare [Hess, 2009]

```
CalculateStartPositions($\Lambda, \mathbf{q}_{\text{des}}, \mathcal{R}, \mathcal{O}, \mathcal{C}$)
  Input:     A set of motion primitives $\Lambda$, a desired start state $\mathbf{q}_{\text{des}}$,
             sets of obstacles and constraints $\mathcal{R}, \mathcal{O}, \mathcal{C}$
  Output:    A set of start positions $X_s$, a cost function $g_0 : X_s \to \mathbb{R}$
  Constant:  Discretization of the grid, $\Delta x, \Delta y, \Delta t$
  $X_s \leftarrow \emptyset; g_0 \leftarrow \emptyset.$
  $t_{\text{off}} \leftarrow \text{rem}(t_{\text{des}}, \Delta t)$
  For all $\tau \in \Lambda$
      For all $t_{\text{rel}} \in \text{dom}(\tau_i)$ with $\text{rem}(t_{\text{rel}}, \Delta t) = t_{\text{off}}$
          $\tau_{\text{fit}} \leftarrow$ translate $\tau$ in $x$ and $y$, so that $[\tau_{\text{fit},x}(t_{\text{rel}}), \tau_{\text{fit},y}(t_{\text{rel}})] = [x_{\text{des}}, y_{\text{des}}]$
          $\mathbf{q}_{\text{start}} \leftarrow \text{round}(\tau_{\text{fit}}(t_{\text{end}}), \Delta x, \Delta y)$ to grid-point
          $\tau_{\text{grid}} \leftarrow$ translate $\tau_{\text{fit}}$ in $x$ and $y$, so that $\tau_{\text{grid}}(t_{\text{end}}) = \mathbf{q}_{\text{start}}$
          If Validate($\tau_{\text{grid}}, \mathcal{R}, \mathcal{O}, \mathcal{C}$) Then
              $c \leftarrow$ CalculateCost($\mathbf{q}_{\text{des}}, \tau_{\text{grid}}$)
              If $\mathbf{q}_{\text{start}} \notin X_s \lor c < g_0(\mathbf{q}_{\text{start}})$ then
                  $X_s \leftarrow X_s \cup \{\mathbf{q}_{\text{start}}\}; g_0(\mathbf{q}_{\text{start}}) \leftarrow c.$
              End
          End
      End
  End
```

proach is taken. In [Urmson et al., 2008], an object oriented approach is chosen for on-road navigation, but on a grid-based representation is applied for navigation in parking lots.

Occupancy grids have the advantage of constant query times for collision checking independent of the number of obstacles and are easy to generate from sensor data for static obstacles in an x-y-map. For higher dimensions and moving obstacles though the computational effort for preprocessing increases exponentially. Due to this reason and in order to make the trajectory initialization consistent with the subsequent optimization, the same object oriented approach is taken as in Chapter 2.

To verify in an object oriented fashion that a trajectory is collision-free, the volumes by the vehicle and all obstacles have to be polygonally represented in the augmented workspace $\mathbb{W}_T : \langle x, y, t \rangle$ and tested for overlap. The implemented collision checking algorithm follows a hierarchical *Divide & Conquer* approach as already described in Chapter 2, similar to e.g. [Ferguson et al., 2008]: The trajectory is recursively split into half along the time dimension, which is easy as movement in t-direction is monotonous. For each trajectory part a rough test with bounding boxes is performed that can guarantee correctness of negative results, (no collision found).

For the current implementation the bounding box is an axis aligned box, which extends through the complete time interval covered by the trajectory, and which would contain the object at any rotation (see Figure 3.7, left). In narrow situations any type of closer fitting container, such as OBBs proposed by [Gottschalk, 2000] (see Figure 3.7, right), could reduce the necessary recursion depth, see Chapter 2.

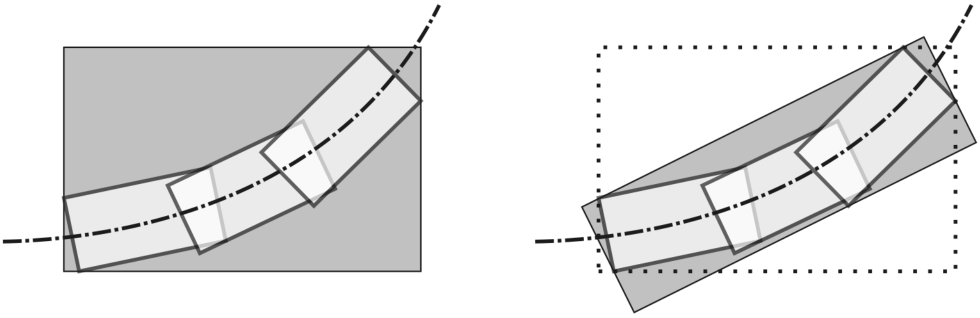


Figure 3.7.: Conservative Collision Test
Rough and conservative collision tests are executed on bounding-boxes around partial trajectories. On the left is displayed the current method of axis aligned bounding boxes. An improvement to the precision of the conservative test would be to use tighter bounding boxes as displayed on the right.

Intersection can be tested with the help of the separating axis test described in [Gottschalk et al., 1996], as already mentioned in Chapter 2, Figure 2.14.

An exceedingly high recursion depth can be avoided by switching to an iterative test method when the recursion limit is reached. The iterative test method samples the remaining sub-trajectories of both obstacle and vehicle at an adequate resolution in time, and simply compares the position of obstacle and vehicle for each point of time.

As mentioned, the initial trajectory (which is the result of the trajectory initialization described in this chapter) must obey the constraints posed by the trajectory optimization presented in Chapter 2. The safest way to achieve this is to apply the same models for road and obstacles. Regarding the obstacles, this applies especially to obstacles' motion extrapolation. While this is indeed the case for the obstacles, regarding the road, currently still a piecewise linear approximation of the road is used for the trajectory initialization and its collision checking. However, it can be tuned to ensure that all nodes of the discretized trajectory for the optimization lie inside the road boundaries. The more detailed road model is left to the trajectory optimization.

3.5. Cost Functions

The cost function essentially controls the chosen initial trajectory, as it assigns preference of one “route” over another in the search algorithm. Therefore, the cost functions should reflect the main demands to the initial trajectory, which can be summed up with the keywords safety and comfort.

A cost function which implements the *safety* requirement is rather elusive: Safety can be divided into, first, safety from collisions with obstacles and road sides and, second, safety from unstable vehicle states to avoid overturns or other uncontrollable behavior. Since no sophisticated vehicle model is used within this search, the latter kind of safety can only be regarded by a cost function that penalizes high acceleration values.

The safety with regards to obstacles refers to the probability of collision. In many planning algorithms the collision probability is approximated by the distance to the obstacles. However, the computation of distances (spatial, temporal or both) to all obstacles along each potential trajectory segment is very time consuming, even if relatively fast approaches for a hardware-based computation of distance maps exist, see [Hoff III et al., 2000]. Therefore, as common to a number of approaches, safety during the trajectory initialization is approximated by the requirement to keep a certain minimum distance to obstacles. To ensure this minimum distance, the obstacles are enlarged accordingly.

According to [ISO2631, 1997; Smith et al., 1978], *comfort* for the driver is highly correlated with the experienced accelerations. Even though other factors such as jerk, see e.g. [Chee et al., 1994], seem to contribute to the measure of comfort, many trajectory planning approaches rely on penalizing or limiting accelerations, see for example [Solea and Nunes, 2006].

For the trajectory initialization algorithm devised in this work, the direct consideration of jerk is not possible due to the setup of the search space and the used local planning method. The comfort is regarded by a cost function c_{acc} which penalizes high absolute accelerations

$$c_{\text{acc}}(\tau) = \int\limits_{t_{\text{start}}}^{t_{\text{end}}} \sqrt{\tau_v(t)^2/|r| + a_{\text{long}}^2} \quad dt. \tag{3.19}$$

An additional consideration of the jerk is left to the subsequent trajectory optimization step.

This cost can be precomputed for all motion primitives and looked-up in constant time. However, it is hard to acquire an admissible estimate for the remaining acceleration cost. Unfortunately, without an admissible non-zero estimate for the remaining cost the A* search would degrade to an uninformed best-first search. Therefore, the acceleration cost is complemented by a path length based cost measure to a pseudo cost c_{pseudo}, which allows to calculate a non-zero estimate of the remaining cost,

$$c_{\text{pseudo}}(\tau) = c_{\text{length}}(\tau) + c_{\text{acc}}(\tau). \tag{3.20}$$

As depicted in Figure 3.8, the remaining length of the path is estimated as the sum of the minimum length paths through all remaining street segments. The minimum lengths $l_{i,\min}$ for a straight, circular, or clothoid segment $\hat{R}^i$ as indicated by bold lines in Figure 3.8 can be computed according to Equation 2.49 resulting in

$$l_{i,\min} \quad = \begin{cases} \hat{l}_i & \phantom{\text{for}} \quad \text{straight road segments} \\ \hat{l}_i - \frac{b}{2}\,\kappa_i\,\hat{l}_i & \text{for} \quad \text{circular road segments} \\ \hat{l}_i - \frac{b}{2}\left(\kappa_{i,0}\,\hat{l}_i + \kappa_i' \frac{\left(\hat{l}_i\right)^2}{2}\right) & \phantom{\text{for}} \quad \text{clothoid road segments,} \end{cases} \tag{3.21}$$

where $\hat{l}_i$ denotes the segment's known length along the centerline, b the street width, κ_i the curvature of a circular segment, $\kappa_{i,0}$ the initial curvature of a clothoid segment, and κ_i' the segment's rate of change of curvature.

For the current implementation of the trajectory initialization, the street is discretized into a piecewise-linear representation.

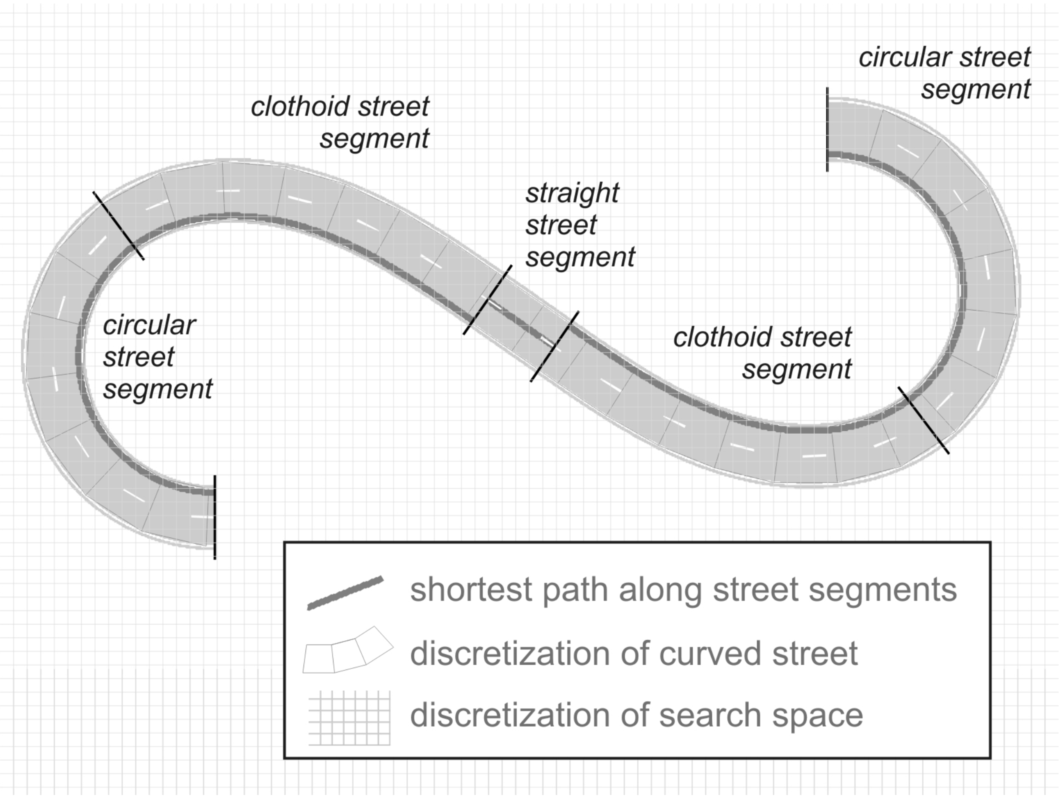


Figure 3.8.: Estimation of remaining path length
The shortest path to cross each segment is indicated by bold lines. In the current implementation of the trajectory initialization, a piecewise-linearly discretized version of the street is used.

CHAPTER 4

Experimental Setup

In order to demonstrate the applicability of the developed trajectory planning approach presented in Chapters 2 and 3, it is tested in simulations and test drives. Section 4.1 describes the information processing and gives an overview over the related hardware and the implemented software structure. Section 4.2 covers the setup of the test vehicle and details how it has been equipped with additional sensors, actuators, power supply, and information processing capabilities for automated driving. An integrated trajectory following controller with several cascaded control loops has been devised that controls lateral and longitudinal deviations from the planned trajectory and provides input to the additional actuators, see Section 4.3.

4.1. Information Processing

Except for some subordinate control-loops and the sensor fusion of the satellite-based Global Positioning System (GPS) and inertial sensors that are discussed in Section 4.2, the information processing is performed by a regular PC or laptop. As described in detail in Section 4.1.1, the PC is equipped with several Controller Area Network (CAN) interfaces and a digital-to-analog (D/A) converter to communicate with the different sensing and automation hardware modules introduced in Section 4.2.

Besides the hardware structure and interfaces, also a suitable software structure had to be devised that is flexible and provides support for simulations, simulator studies, and experiments, as detailed in Section 4.1.2.

4.1.1. Hardware Interfaces in Test Vehicle

A PC[1] was chosen as the central processing and control unit due to cost limitations and the simplicity of implementation. The PC is connected via CAN busses and a D/A converter to the different sensing and automation hardware modules, see Figure 4.1.

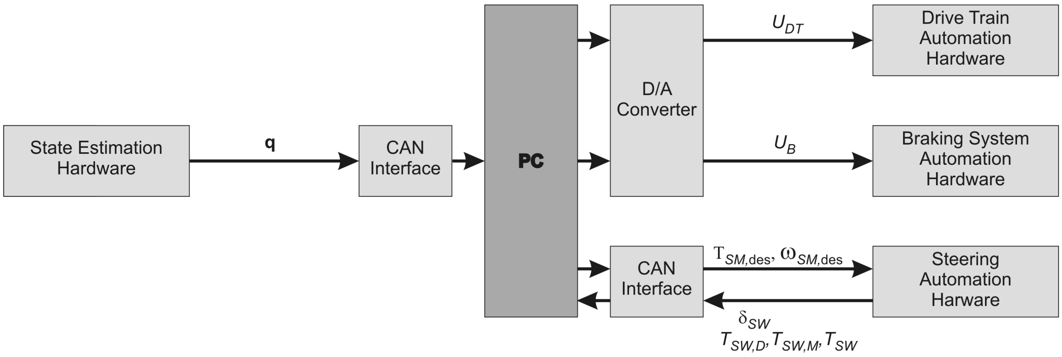

State Estimation Hardware
q
CAN Interface
PC
D/A Converter
U_{DT}
Drive Train Automation Hardware
U_B
Braking System Automation Hardware
CAN Interface
$T_{SM,\mathrm{des}}, \omega_{SM,\mathrm{des}}$
Steering Automation Harware
δ_{SW}
$T_{SW,D}, T_{SW,M}, T_{SW}$

Figure 4.1.: Block diagram of hardware interfaces to PC
The PC as the central processing and control unit is connected via CAN busses and a D/A converter to the different sensing and automation hardware modules. As input, the PC receives the estimated position $\mathbf{q}$ and the current steering wheel angle δ_{SW} and steering wheel torques $T_{SW,M}, T_{SW,D}, T_{SW}$. The output from the PC consists of two control voltages U_{DT} and U_B for a desired acceleration and braking, respectively, and the desired steering torque $T_{SM,\mathrm{des}}$ or steering rate $\omega_{SM,\mathrm{des}}$.

As input, the PC receives the estimated vehicle state $\mathbf{q}$ from the state estimation module. Further, the PC receives feedback about the current steering wheel angle δ_{SW} and steering wheel torques caused by the steering motor ($T_{SW,M}$), the human driver ($T_{SW,D}$), as well as the total steering wheel torque T_{SW}. Each automation hardware module is driven by one control variable. For the acceleration automation hardware module this is the voltage U_A that represents a certain desired throttle flap angle. With regards to braking, the control variable is the voltage U_B that represents a certain desired position of the brake pedal. The steering automation hardware module receives, depending on the used modus, either the desired steering torque $T_{SM,\mathrm{des}}$ or the desired steering rate $\omega_{SM,\mathrm{des}}$ via the steering CAN.

---

[1] Laptop: Intel Core 2 Duo Mobile T5500; 2x1,67GHz, 1GB RAM

4.1.2. Software Structure

It is important to implement a software structure that provides support during all phases of development and testing. More specifically, the software structure should be usable in simulations, simulator studies, and experiments in a test vehicle and allow the usage of identical software modules such as the trajectory planning in all cases without any changes to the code.

The devised software structure consists of separate modules that all communicate via a single shared memory, as illustrated in Figure 4.2. The foundation to this software structure is called Straightforward Modular Prototyping Library (SMPL) and was acquired through a cooperation with the Institute of Transportation Systems of the German Aerospace Center in Braunschweig.

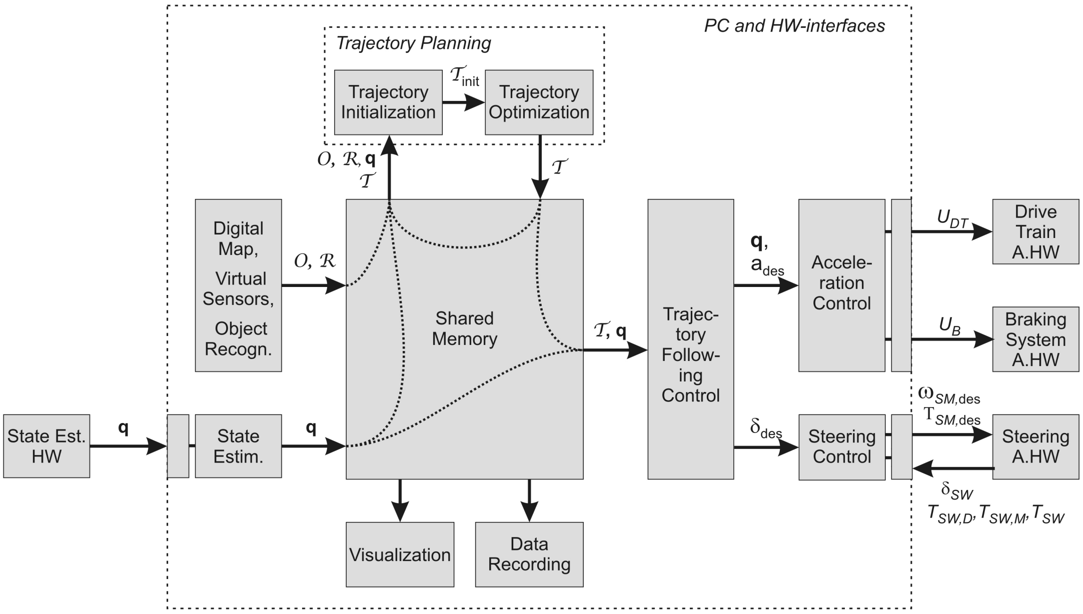


Figure 4.2.: Software structure
All modules, i.e. the state estimation, the virtual sensors, the trajcetory planning, the trajectory following controller, and secondary modules such as the visualization and the data recording, communicate over a shared memory.

The communication is asynchronous which allows each software module to run at a different frequency as necessary. Further, each software module can even be executed on a different computer as long as they are connected, e.g. by ethernet. The shared memory is then synchronized between all computers by a socket-based communication, serializing and then deserializing all variables.

Figure 4.2 shows the configuration used in the test vehicle. The information about the environment that consists of obstacles $\mathcal{O}$ and the road $\mathcal{R}$ is generated based on a digital map. In future applications, this data can be acquired using external sensors such as RADAR, camera, or laser scanners. The current dynamic state $\mathbf{q}$ of the vehicle is measured by the state estimation hardware, and simply written to the shared memory by the state estimation software module. Alternatively, the data fusion of GPS and inertial measurement unit (IMU) data can be performed in the state estimation software module.

The trajectory planning uses the information about the environment $(\mathcal{O}, \mathcal{R})$, the dynamic state $\mathbf{q}$, and the previously planned trajectory $\mathcal{T}$. First, an initial trajectory $\mathcal{T}_{\text{init}}$ is generated, according to the trajectory initialization method presented in chapter 3. Second, $\mathcal{T}_{\text{init}}$ is optimized as described in Chapter 2 to provide the new planned trajectory $\mathcal{T}$.

The trajectory following controller always reads the current planned trajectory $\mathcal{T}$ and the current dynamic state $\mathbf{q}$ from the shared memory and determines the desired longitudinal acceleration steering angle a_{des} and δ_{des}, respectively.

The acceleration is controlled in a subordinate module that decides between using the brake or the throttle valve to achieve the desired acceleration and then determines the proper voltages U_A and U_B to drive the respective hardware modules.

The desired steering angle δ_{des} is controlled in the subordinate steering control module, where the desired steering rate, i.e. the rotational velocity $\omega_{SM,\text{des}}$ for the steering motor is determined based on the control error between the desired and the actual steering wheel angle δ_{SW}. Additionally to what is displayed in Figure 4.2, also a desired motor torque could be set. The actual steering torques measured by the strain gauges are omitted in this diagram, since they are not used in any control loop.

In addition, secondary software modules exist to realize the visualization of the current situation and to record all data for later evaluation.

In order to perform simulations instead of experiments in the test vehicle, basically, only the real vehicle must be replaced by a vehicle model. Depending on the depth of the model, some subordinate control loops also have to be eliminated. For example, if the vehicle model does not contain any drive train model, there is no point in determining a driving voltage U_A to set a certain throttle flap angle. Rather, the desired longitudinal acceleration a_{des} can be set directly in the model. Figure 4.3 shows the configuration for use with a simple model.

For simulator studies the steering control can be added again to drive the steering hardware module of the simulator.

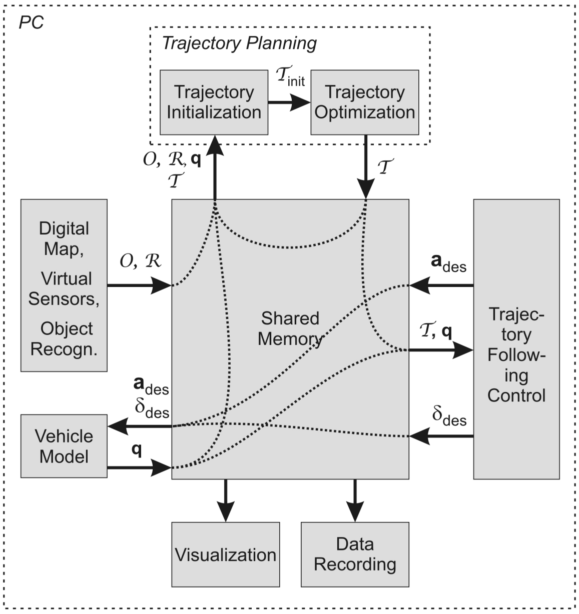

PC
Trajectory Planning
Trajectory Initialization
$\mathcal{T}_{\text{init}}$
Trajectory Optimization
$\mathcal{O}, \mathcal{R}, \mathbf{q}$
$\mathcal{T}$
$\mathcal{T}$
Digital Map, Virtual Sensors, Object Recogn.
$\mathcal{O}, \mathcal{R}$
Shared Memory
$\mathbf{a}_{\text{des}}$
$\mathcal{T}, \mathbf{q}$
Trajec-tory Follow-ing Control
$\mathbf{a}_{\text{des}}$
δ_{des}
δ_{des}
Vehicle Model
$\mathbf{q}$
Visualization
Data Recording

Figure 4.3.: Software structure for simulations with simple vehicle model
Compared to the configuration for test drives, the test vehicle is replaced by a model and some subordinate control loops have been eliminated.

4.2. Test Vehicle

The test vehicle is a BMW 540i Touring that has been equipped with additional sensors and actuators, some of which are displayed in Figure 4.4. Some of the main vehicle parameters are listed in Table 4.1.

In order to find the best concept of automation, a list of requirements was drawn up, all options to achieve the desired functionality were considered, different feasible automation concepts were designed based on the state of the art found in current literature, and finally short utility analyses were performed to select the best concept for our purpose. The following Subsections 4.2.1 - 4.2.4 detail the different additions to the vehicle and briefly discuss the main reasons for the choice of concept.

Note, that no external sensors such as RADAR, laser scanners, or cameras have been implemented. Instead, external sensor information is simulated online based on the location of the ego-vehicle in a detailed digital map. This allows to decouple the development, testing, and demonstration the developed motion planning approach from the quality of any particular sensors.

Parameter	Value
Vehicle model	BMW 540i Touring
Motor power	210kW
Mass m_{veh}	1,845kg
Length l_{veh}	4.805m
Width w_{veh}	1.800m
Height h_{veh}	1.440m
Distance from center of gravity to front axle l_F	1.46m
Distance from center of gravity to rear axle l_R	1.37m
Steering ration $i = \delta_{SW}/\delta$	17.9
Transmission	5 gear, automatic

Table 4.1.: Main vehicle parameters according to [Presse, 2002] and own measurements

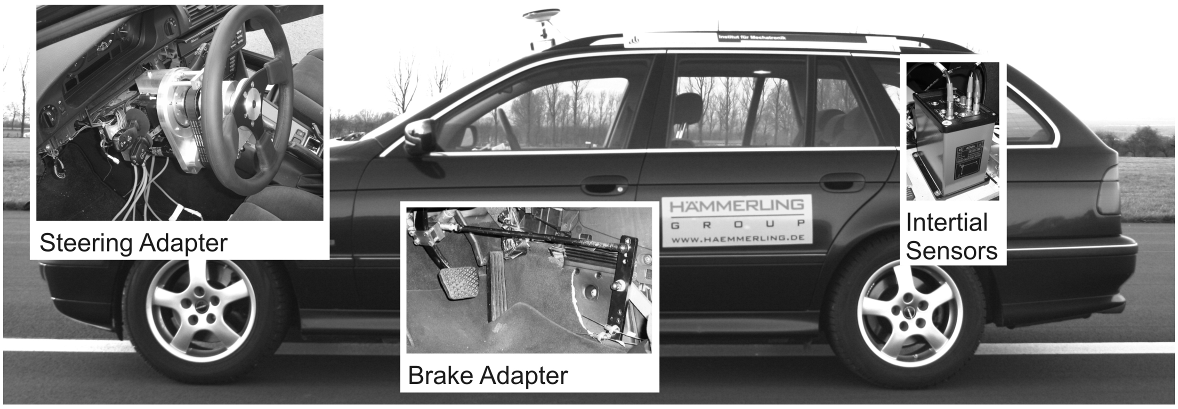


Figure 4.4.: Test vehicle
The BMW 540i Touring is equipped with sensors for state estimation and additional actuators and sensors for automated steering, braking and accelerating.

4.2.1. Sensors for State Estimation

For reliable trajectory planning and following it is essential to measure the relevant current dynamic state (i.e. position, velocity, acceleration, yaw angle and yaw rate) of the vehicle accurately at a high update rate.

In order to achieve this goal, a number of different sensors can be used, such as sensors for satellite navigation (using e.g. GPS), optical velocity sensors (e.g. Correvit), inertial sensors such as accelerometers and gyroscopes, wheel speed sensors, or even a steering angle sensor.

All sensors can also be used in combination with each other and with prior knowledge about the vehicle (such as a vehicle model) or the environment (such as a digital map). The combination of several sensors is called sensor fusion and has the advantage that data from multiple sources can be used to acquire a better estimate of the vehicle's dynamic state. Moreover, that the individual drawbacks of different sensors can be eliminated by the combination of sensors of complementary characteristics.

Most of the sensors mentioned above cannot measure the vehicle's absolute position directly. Rather, measurements of velocities, accelerations, or turning rates must be integrated over time which requires knowledge of an initial reference state and leads to accumulating errors. Therefore, the usage of GPS as absolute reference is necessary.

For this test vehicle, the dynamic state is determined by sensor fusion of GPS data with inertial measurements. This combination has proved successful in a number of applications, see e.g. [Rezaie et al., 2007; Schubert et al., 2008; Wendel, 2007], due to the complementary characteristics of the sensors as displayed in Table 4.2.

	GPS measurements	inertial measurements
Short term error	medium	low
Long term error	low	high
Update rate	low	high

Table 4.2.: Comparison of GPS and inertial measurements

Inertial measurements can be acquired at a very high update rate and give reliable results for short periods of time. Unfortunately, for longer periods of time, the integration of inertial measurements leads to accumulating position errors. GPS on the other hand yields errors in individual position measurements, e.g. due to multipath erros (caused by reflections from buildings, etc.), atmospheric effects, clock inaccuracies, and satellite visibility [Kaplan and

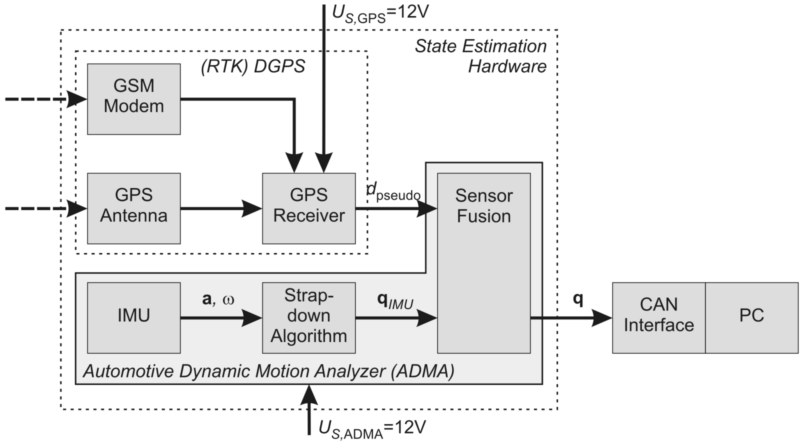


Figure 4.5.: Block diagram of state sensing and estimation, adapted from [GeneSys Elektronik GmbH, 2008]
The input to the state estimation are GPS satellite signals and correction data for differential GPS (DGPS) via a GSM modem. The ouput is the current estimation of the vehicle's dynamic state $\mathbf{q}$.

Hegarty, 2006], and is only available at a rather low update rate. However, the position error does not accumulate over time. Therefore, the GPS measurements can be used to correct accumulating errors from inertial measurements while the usage of inertial measurements can provide a higher update rate and mitigate errors in individual GPS measurements.

Figure 4.5 gives an overview of elements involved in the state estimation of the vehicle. The used GPS receiver is a Novatel ProPak-V3 which uses dual frequency differential GPS with realtime kinematics (RTK DGPS) to provide measurements with a nominal accuracy of 1cm + 1ppm at 10-20Hz depending on the provided values and the receiver's configuration, see [Novatel, 2006*a*,*b*]. If the rover is e.g. 10km away from the stationary reference station for the differential GPS corrections, 1ppm equals 1cm, [Novatel, 2006*c*]. In addition, it is important to note that the accuracy is given for a circle of equal probability (CEP), which means that in the given example 50% of all measurements lie within a circle of 2cm radius around the real value.

The correction data for the differential GPS is provided by ASCOS, a commercial service by Axio-net GmbH, via a GSM modem. The position and velocity measurements can be acquired from the receiver via a serial port. For a detailed explanation of the underlying principles of differential GPS see for example [Wendel, 2007].

The inertial measurements stem from three accelerometers and three fiberoptic gyroscopes which are combined into an IMU that provides accelerations $\mathbf{a}$ and turning rates $\boldsymbol{\omega}$ in all three axes. The IMU used in the test vehicle is located inside the Automotive Dynamic

Motion Analyzer (ADMA) from GeneSys GmbH. Most errors such as misalignment errors or temperature drift are corrected internally. Measurements are provided at an update rate of up to 400 Hz via a CAN bus interface, [GeneSys Elektronic GmbH, 2008].

A strapdown algorithm corrects for accelerations and turning rates due to gravity, the earth turning rate, and Coriolis terms when the vehicle is moving and integrates the measurements to obtain velocity, position, and orientation measurements. GPS and IMU data is fused by means of an extended Kalman filter (EKF). Both the strapdown algorithm and the Kalman filter run inside the ADMA which provides readily fused measurements for the vehicle's dynamic state at an update rate of 100-200Hz via a CAN bus. The implemented fusion is called "tightly coupled", since the determined pseudoranges d_{pseudo}, i.e. the estimated distances to the satellites in view, are used to enhance the results even for only few visible satellites. For more details refer to [GeneSys Elektronic GmbH, 2008; GeneSys Elektronik GmbH, 2008].

Alternatively to the commercial version of sensor fusion, several different Kalman filter based sensor fusion algorithms (a basic Kalman filter, an error state-space EKF and an unscented Kalman filter (UKF)) have been developed in related student theses with promising results, see [Arronte Arroyuelos, 2006; Lauhoff, 2009; Moreno Schneider, 2007; Othmani, 2009].

4.2.2. Drive Train Automation

Figure 4.6 shows the principal components involved in the acceleration of the vehicle and offers several points of influence where the vehicle could be altered to automate the acceleration.

Changes or additions to the later part of the drive train were not considered due to infeasibility of the necessary constructions. Access to the motor control (as done e.g. in [Schröder et al., 2006]) was also impossible. The main options included actuating or exchanging the gas pedal (as e.g. in [Kirchgässner and Tröster, 2006]) or changing the Adaptive Cruise Control (ACC) motor control. Even though, some disadvantages applied with regards to the possible development of human-machine interactions (HMI) in the future, it was chosen to use the ACC motor to automate the vehicle's acceleration, mainly because it was easier to implement, fulfilled all safety requirements, and offered the freedom of additional changes to the gas pedal for HMI design later.

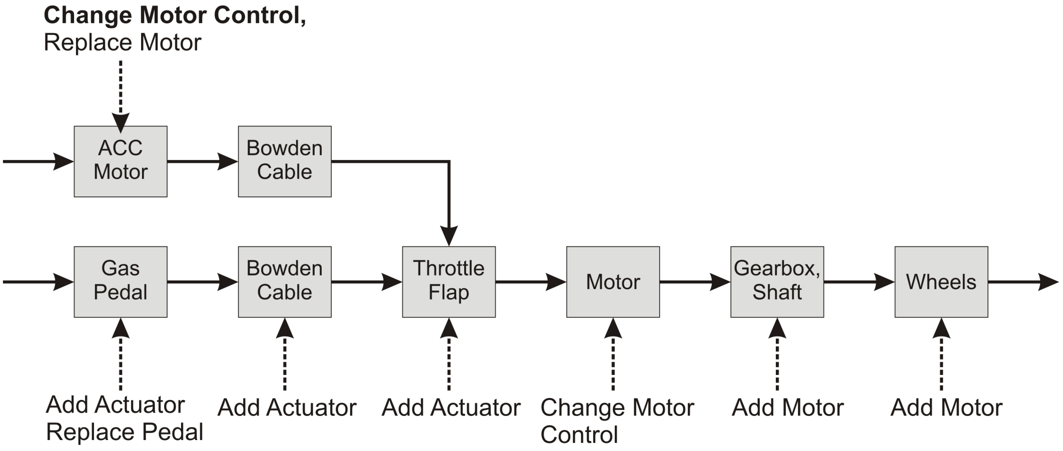


Figure 4.6.: Block diagram of principal drive train components
Possible points of influence are shown to achieve an automated acceleration of the vehicle. As indicated in bold, it was chosen to change the motor control of the existing ACC motor in order to automate the drive train.

The existing ACC throttle flap motor was disconnected from the ACC controls and instead controlled by an analog control circuit to actuate the throttle valve from the PC, as displayed in Figure 4.7.

The desired throttle flap angle α_{des} is given as a voltage U_{DT} by a D/A converter, while the current throttle flap angle α is measured as the voltage U_{pot} by a potentiometer. To control α, an analog PID controller is used. The control output U_{TFM} in amplified by an operational amplifier circuit to provide sufficient power to drive the throttle flap motor. This control loop was implemented outside the PC in order to increase speed and save computational resources. The analog implementation was chosen mainly due to lower costs and less implementation effort if compared to an implementation on a separate micro controller.

The throttle flap motor is supplied with up to 24V DC (via the operational amplifier) which is about twice the motor's original specification to increase the possible reaction times to set a certain throttle flap angle. The voltage input to the potentiometer had to be stabilized to a controlled supply voltage of $U'_{S,\text{pot}} = 5\text{V}$ as indicated in Figure 4.7, because the voltage of 12 V provided by the vehicle itself is not steady enough to provide the same voltage range as provided by the D/A converter.

The throttle flap is also still connected to the driver's gas pedal, enabling the driver to open the throttle flap further and increase the vehicle's acceleration. Further, there is a clutch between the throttle flap motor and the throttle flap which opens when the power is cut. Because the throttle flap is spring loaded it then closes to the position given by the

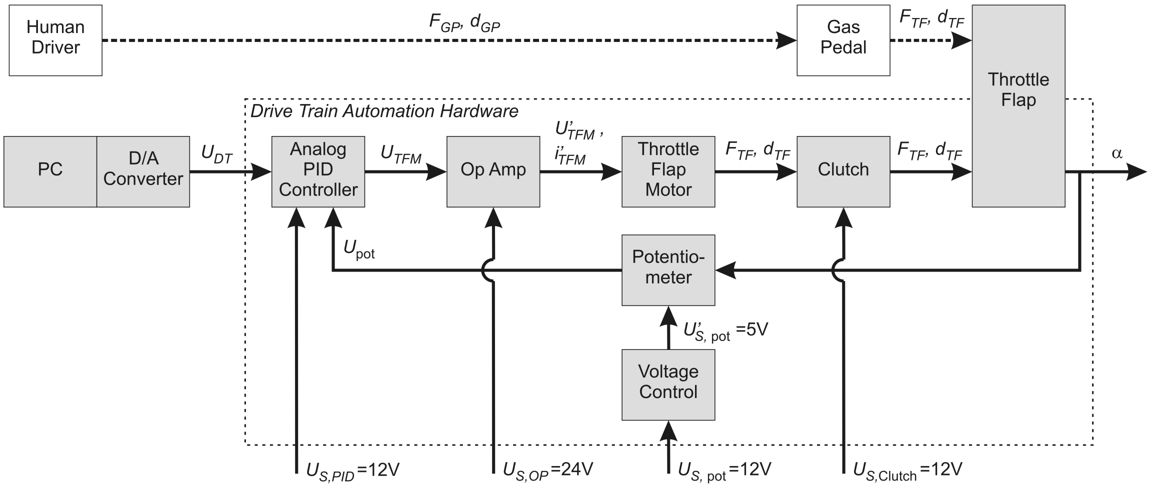


Figure 4.7.: Block diagram of drive train automation hardware
Input to the drive train automation hardware is an drive train voltage U_{DT} that is produced by a D/A converter and is proportional to the desired throttle flap angle α. The output is a force F_{TF} that causes a displacement d_{TF} of a bowden cable that opens the throttle flap by α to accelerate the car.

gas pedal. To shut off the acceleration automation, an emergency switch cuts the power to the clutch and activates another relay that shuts off the power supply to the throttle flap motor. If the throttle motor is disconnected, the vehicle is still allowed to drive on public streets and does not loose its official approval and homologation for road service.

4.2.3. Braking System Automation

Figure 4.8 shows the principal components involved in the braking of the vehicle and offers several points of influence where the vehicle could be altered to achieve automated braking.

While the addition of extra brakes or additional actuators directly at the brakes was not considered due to technical difficulties and legal problems, many other viable options were taken into account. Concerning a manipulation of the existing Dynamic Stability Control (DSC) system, the available brake pressure of about 160 bar that can be found in descriptions should suffice even for an emergency brake. However, investigations at Karlsruhe University revealed that (at least in their case) the equivalent Electronic Stability Program (ESP) could not produce a sufficiently large deceleration, [Schröder et al., 2006]. Additionally, in oder to gain access to the DSC controls, the control unit would have had to be replaced with unforeseen consequences for the other connected systems.

A regulation of the brake pressure by additional pipes, valves, and pumps was readily designed and taken into account. However, it proved hard to find suitable valves with sufficient reaction times that could withstand the corrosive brake fluid. Further, the technical realization would have been quite costly and time consuming.

An alternative that is used in many automated vehicles is to influence the brake pressure by changes to the main brake cylinder (see e.g. [Gerdes, 1996]), or the brake booster (see e.g. [Mauciuca, 1997]). However, it was chosen to actuate the braking pedal instead (similar to [Schröder et al., 2006]), due to an easier technical realization in our case.

A number of different actuators and constructions were considered and evaluated such as linear or rotatory-type actuators, hydraulic, pneumatic, electromechanical, or even piezoelectric actuators, and different types of linkage between the actuator and the pedal. As the best solution in our case, the brake is actuated by a DC brake motor via a bowden cable and a lever kinematic as displayed in Figures 4.4 and 4.9.

The motor originates from a handicapped vehicle control system. A second braking motor exists in parallel to the first to provide redundancy and additional force if needed, but is currently not in use.

The brake motor possesses an internal position control for the bowden cable, where the desired position is given by an analog input signal. This signal originates from a D/A converter and a customized circuit board for necessary voltage adaptations, as illustrated in Figure 4.10.

Depending on the desired deceleration of the vehicle, a braking voltage U_B is produced by a D/A converter, which is then adapted to a suitable range to the voltage U'_B. This voltage encodes a desired displacement of the bowden cables by the braking motor, which is controlled inside the motor. The motor produces a force F_{BC} to achieve the displacement of the bowden cables d_{BC} which is then translated via the lever kinematic to a brake pedal displacement d_{BP} and finally the displacement input to the brake booster s_B. Together with the vehicle's brake booster, the force F_{BP} at the brake pedal produces a certain brake pressure p_B that activates the brakes to decelerate the car.

The brake motor is supplied with 24V DC which is provided from a 12 to 24 V converter. The voltage input to the brake control circuit board is stabilized, because the voltage of 12 V provided by the vehicle itself is not steady enough.

Because the bowden cables from the brake motor to the lever kinematic can apply forces only in one direction, the brake pedal can always be pushed down further by a human driver to increase the deceleration. Additional analog input signals from the D/A converter to

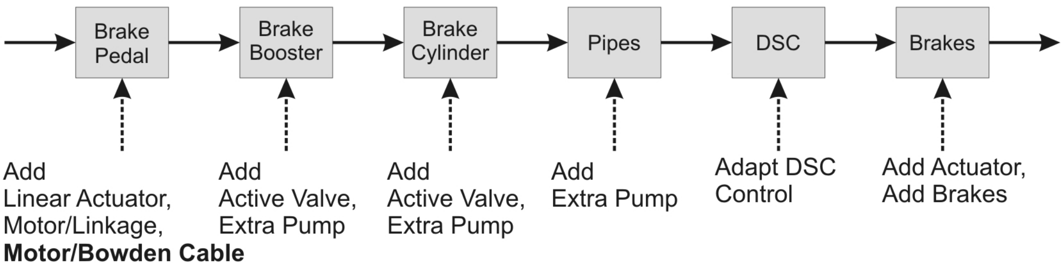
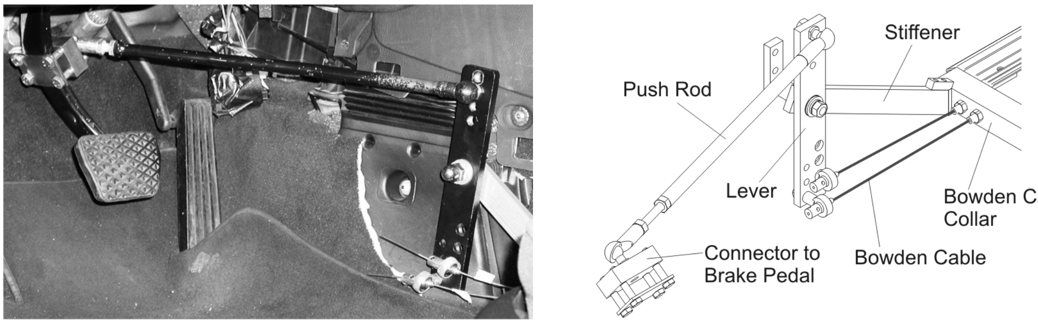

Brake Pedal
Brake Booster
Brake Cylinder
Pipes
DSC
Brakes
Add Linear Actuator, Motor/Linkage, **Motor/Bowden Cable**
Add Active Valve, Extra Pump
Add Active Valve, Extra Pump
Add Extra Pump
Adapt DSC Control
Add Actuator, Add Brakes

Figure 4.8.: Block diagram of principal braking system components
Possible points of influence are shown to achieve an automated braking of the vehicle. As indicated in bold, it was chosen to actuate the brake pedal by an additional motor via bowden cables.



(a) Foto (b) Drawing

Figure 4.9.: Lever kinematic for braking system automation hardware
One or two DC motors pull bowden cables that turn the displayed vertical lever counterclockwise to push down the brake pedal.

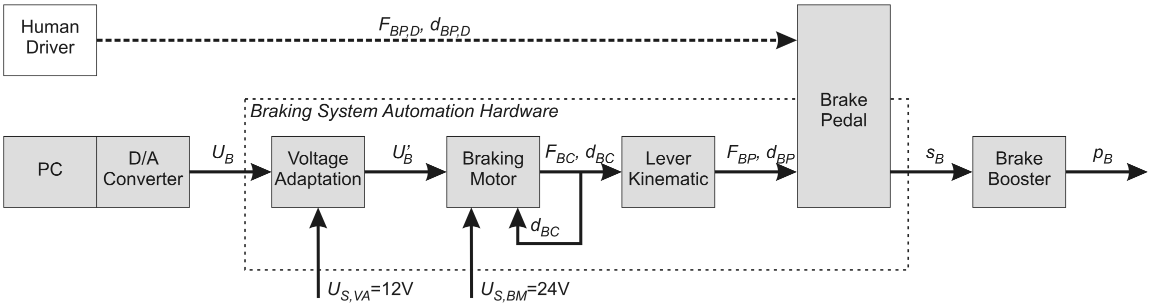


Figure 4.10.: Block diagram of braking system automation
Input to the braking system automation hardware is a braking voltage U_B that is produced by a D/A converter and the output is a force F_{BP} on the brake pedal that causes a brake pedal displacement d_{BP}, which in turn translates to the brake booster input displacement s_B. The brake booster then produces a certain brake pressure p_B that activates the brakes to decelerate the car.

the brake motor deactivate it. Therefore, the braking automation can be switched on and off by software. The emergency switch activates a relay that cuts the power supply to the brake motor, rendering control to a human driver. The lever kinematic can be disconnected manually from the brake pedal. If this mechanical connection is severed, the vehicle is still allowed to drive on public streets and does not loose its official approval and homologation for road service.

4.2.4. Steering Automation

Figure 4.11 shows the principal components involved in the steering of the vehicle and offers several points of influence where the vehicle could be altered to achieve automated steering.

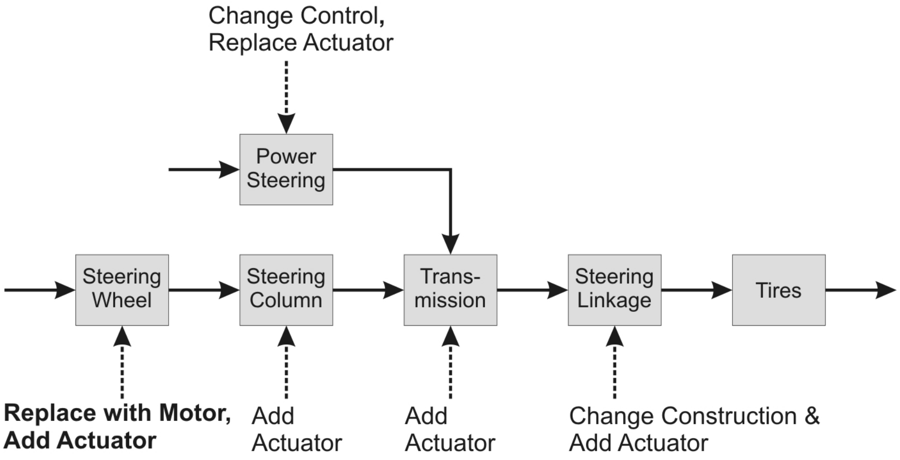
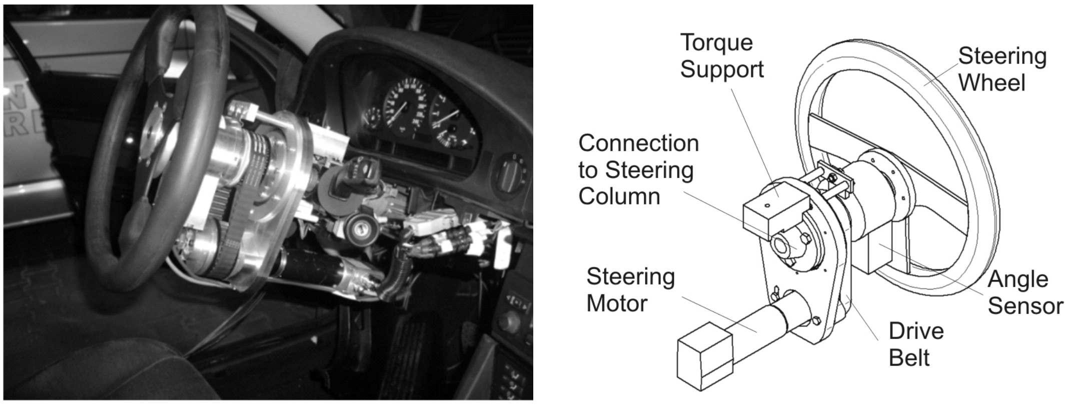


Figure 4.11.: Block diagram of principal steering components
Possible points of influence are shown to achieve an automated steering of the vehicle. As indicated in bold, it was chosen to automate the steering by replacing the steering wheel with an actuated steering adapter.

An alteration or addition to the steering linkage, the steering transmission or the power steering was infeasible. Furthermore, the steering wheel could not be eliminated completely, since the driver had to take over control in case of an automation failure. A motor connected to the steering column was difficult to mount and would have had legal implications.

Therefore, automated steering is achieved by the introduction of a steering adapter that can be mounted instead of the regular steering wheel, see Figures 4.4 and 4.12. The detailed mechanical construction was supported by a supervised student thesis, [Nawroth, 2007]. The main components are a brushless DC steering motor and a steering wheel angle sensor



(a) Foto (b) Drawing

Figure 4.12.: Steering adapter, see also [Nawroth, 2007]
The steering adapter consists mainly of a steering motor and a steering wheel angle sensor. It can be attached or removed to restore the original steering wheel.

to control the steering angle. The rotating part of the steering adapter is fixed to the steering column while the part that includes the motor itself is mounted to the dash board carrier to provide torque support.

The steering motor is combined with a transmission and provides up to 15 Nm of torque at about 1.5 rotations per second to turn the steering wheel. It is driven by a motor controller which combines the necessary power electronics with a microcontroller that can be set to control either the torque or the rotational velocity of the motor, see Figure 4.13. The mentioned additionally attached sensor measures the current steering wheel angle at 1 kHz and a resolution of 0.04°, [IVO GmbH & Co. KG, 2008]. Both the sensor and motor driving microcontroller communicate via an additional CAN bus (the “steering CAN”).

In addition, strain gauges have been applied to the steering adapter to both sides of the pinion where the motor torque is applied. This way the torque applied by the human driver and the steering motor can be measured as well as the torque “feedback” from the wheels the driver would ordinarily experience when holding the steering wheel. The strain gauges are connected to measurement amplifiers which also provide the current measurements as 32 bit variables to the steering CAN at an update rate of about 1kHz, [HBM, 2008]. Unfortunately, the torque measurements are only accurate to about ±0.1Nm. Therefore, they cannot be used for feedback control but only for example to recognize when large torque control errors occur or whether the driver tries to turn the wheel himself.

The steering motor is supplied with 24V DC supplied by a 12V to 24V DC/DC converter. In case a human driver turns the steering wheel, power might be generated inside the motor which must be absorbed in a chopper module in order not to damage the motor controller.

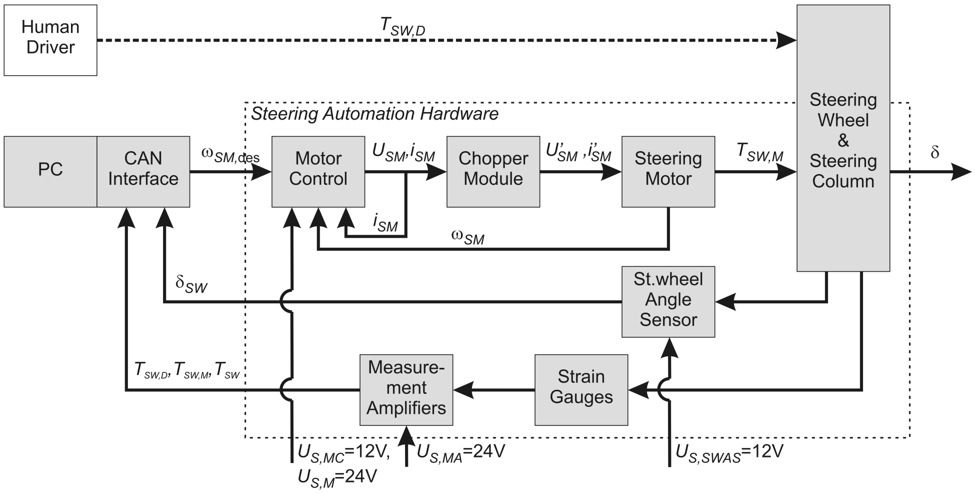


Figure 4.13.: Block diagram of steering automation hardware
The input is a desired torque $T_{SM,\mathrm{des}}$ or rotational velocity $\omega_{SM,\mathrm{des}}$ of the steering motor which is given via the steering CAN. The ouput is a certain torque T_{SM} that acts on the steering adapter and turns the steering wheel. Further, feedback about the current steering wheel angle δ_{SW} and the applied steering torques by the steering motor $T_{SW,M}$, the human driver $T_{SW,D}$, and the total torque T_{SW} is also provided via the steering CAN bus.

The steering wheel angle sensor and the strain gauge measurement amplifiers both use a 12V DC supply voltage from the car battery.

The steering motor can be throttled at the motor controller either manually or via the steering CAN to decrease the maximum torque. If the maximum torque is low enough (<5Nm), a human driver can forcibly overrule the steering decisions. The steering motor can be enabled and disabled both by a software switch and a hardware switch at the motor controller. Further, the emergency switch cuts the power supply to the steering motor via a relay. Steering is then left to a human driver.

The whole steering adapter can be removed and replaced with the original steering wheel. Thus, the vehicle is still allowed to drive on public streets and does not loose its official approval and homologation for road service.

Preliminary tests have shown that the steering angle δ at the tires and the steering wheel angle δ_{SW} have no static relation. Instead, the steering adapter, steering column and transmission between δ and δ_{SW} have a significant compliance and it is hard to determine the real steering angle δ just by measuring δ_{SW}. Therefore, an additional steering angle sensor was attached directly at the tires, as shown in Figure 4.14.

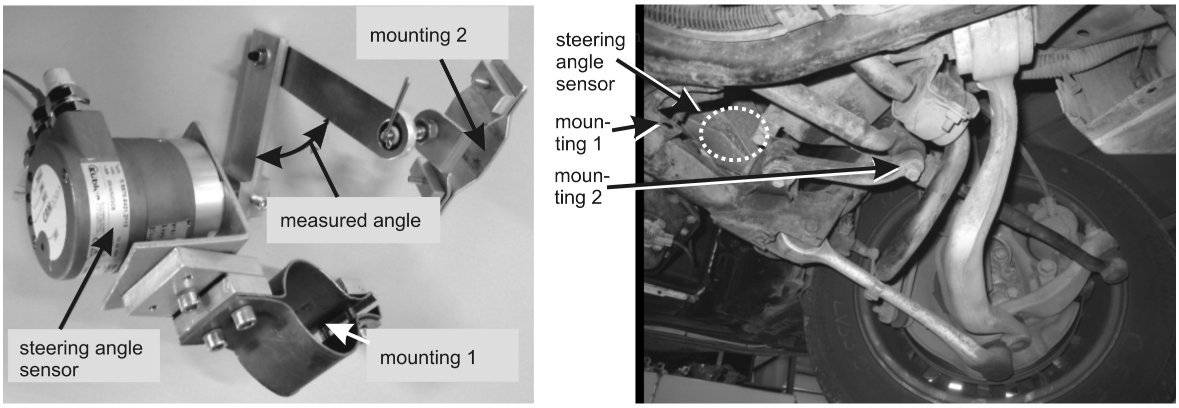


Figure 4.14.: Steering angle sensor
The steering angle sensor is attached close to the wheel underneath the car in a parallelogram-construction due to space constraints. It measures the current steering angle δ very directly, compared to the steering wheel angle sensor in the steering adapter which measures δ_{SW}.

4.3. Trajectory Following Control

In order to show the applicability of the devised trajectory planning method in test drives, a trajectory following controller is designed and implemented. As depicted in Figure 4.15, the controller is structured into several cascaded control loops. In the outermost control loop, an integrated lateral and longitudinal control generates the desired steering angle δ_{des} and longitudinal acceleration $a_{x,\mathrm{des}}$ depending on the vehicle's deviation from the planned trajectory $\mathcal{T}$, see Section 4.3.1. The subordinate lateral and longitudinal controllers are detailed in Sections 4.3.2 and 4.3.3.

As needed for the controller design, appropriate vehicle models haven been built or selected. In general, one can differentiate between theoretical and experimental modeling. Theoretical models (often referred to as *white box models*) represent the system by (differential) equations that stem from physical relations and may be very complex. Experimental models (*black box models*) on the other hand do not offer any insight into the structure of the system but merely model the overall transfer behavior. The advantage of such models is that they are often easier to acquire, [Börner, 2003].

The combination of black and white box models, i.e. of theoretical and experimentally acquired submodels is often referred to as a *gray box model*, [Schorn, 2007, p. 11]. Generally, a suitable model is as simple as possible while reproducing all relevant dynamics as accurately as necessary. Therefore, it is a promising approach to firstly build a theoretical model which can be deconstructed into smaller parts and then to replace some rather com-

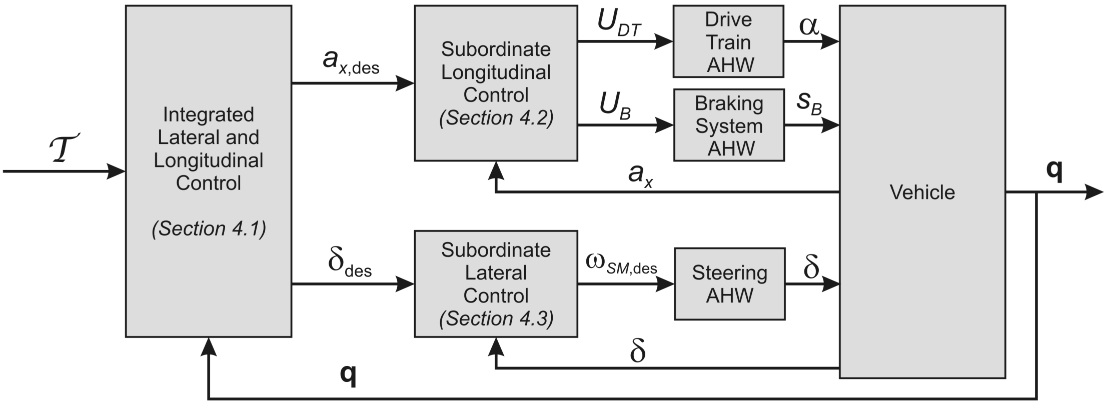


Figure 4.15.: Blockdiagram Trajectory Following Control
The trajectory following control is organized in several cascaded control loops. The outer control loop consists of an integrated lateral and longitudinal control (see Section 4.3.1). Subordinate control loops exist for the control of the steering angle δ and the longitudinal acceleration a_x (see Sections 4.3.2 and 4.3.3) which then drive the respective automation hardware (AHW) modules.

plex submodels with experimental models that are easier to obtain, [Zambou, 2005, p. 43]. This way, the model becomes simpler and easier to obtain while the basic structure of the overall system is conserved.

The used models are described in the following sections along with the developed controllers. While for the outer control loop for the integrated lateral and longitudinal control a simple white box model (single track model) is used, gray or black box models are used applied in the modeling of the longitudinal vehicle dynamics.

4.3.1. Integrated Longitudinal and Lateral Motion Control

Based on the literature search presented in Section 1.2 on the state of the art and some own experiments with different types of controllers (e.g. in related student theses supervised by this author [Hertkorn, 2008; Niggemann, 2007; Roeser, 2008; Schellenberg, 2007; Wesemeyer, 2008]), it was decided to pursue an integrated lateral and longitudinal control approach. The chosen controller is based on the method of nonlinear decoupling, which is similiar to feedback linearization [Freund and Mayr, 1997], see also [Föllinger, 2008; Khalil, 2002] for further information. This method needs the system dynamics to be transformed from the state space representation

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{a}(\mathbf{x}) + \mathbf{B}(\mathbf{x})\mathbf{u}, && (4.1)\\ \mathbf{y} &= \mathbf{c}(\mathbf{x}) && (4.2)\end{aligned}$$

to the equivalent form [Föllinger, 2008]

$$\tilde{\mathbf{y}} = \tilde{\mathbf{c}}(\mathbf{x}) + \tilde{\mathbf{D}}(\mathbf{x})\mathbf{u}(t), \tag{4.3}$$

$$\tilde{\mathbf{y}} = \begin{bmatrix} \overset{(\delta_1)}{y_1} \\ \vdots \\ \overset{(\delta_n)}{y_n} \end{bmatrix}; \tilde{\mathbf{c}}(\mathbf{x}) = \begin{bmatrix} N^{\delta_1} c_1 \\ \vdots \\ N^{\delta_n} c_n \end{bmatrix}; \tilde{\mathbf{D}}(\mathbf{x}) = \begin{bmatrix} \frac{\partial}{\partial \mathbf{x}} N^{\delta_1 - 1} c_1 \mathbf{B} \\ \vdots \\ \frac{\partial}{\partial \mathbf{x}} N^{\delta_n - 1} c_n \mathbf{B} \end{bmatrix}; \mathbf{u}(t) = \begin{bmatrix} u_1 \\ \vdots \\ u_n \end{bmatrix}.$$

Therein, δ_i represents the differential order for each output y_i,

$$\delta_i = \min\{j : \frac{\partial}{\partial \mathbf{x}}(N^{j-1} c_i)\mathbf{B}(\mathbf{x}) \neq 0; j = 1, 2, \ldots, n\}, \tag{4.4}$$

which signifies the lowest derivative $\overset{(\delta_i)}{y_i}$ that depends explicitly on $\mathbf{u}$. The operator $N^k c_i$ is recursively defined as

$$N^k c_i = [\frac{\partial}{\partial \mathbf{x}} N^{k-1} c_i(\mathbf{x})]\mathbf{a}(\mathbf{x}) \quad \text{mit} \quad N^0 c_i = c_i(\mathbf{x}). \tag{4.5}$$

For more details regarding this representation and the determination of the differential order see for example [Föllinger, 2008].

The basic idea of a nonlinear decoupling controller is to linearize and decouple the dynamic system such that each output y_i or $\tilde{y}_i$ is only influenced by a single reference input w_i. In order to achieve this property the control law depicted in Figure 4.16 is chosen,

$$\mathbf{u}(t, \mathbf{x}) = \boldsymbol{\alpha}(\mathbf{x}) + \boldsymbol{\beta}(\mathbf{x})\mathbf{w}(t), \tag{4.6}$$

where $\boldsymbol{\alpha}(\mathbf{x})$ is similar to a controller matrix and $\boldsymbol{\beta}(\mathbf{x})$ to a prefilter in linear systems.

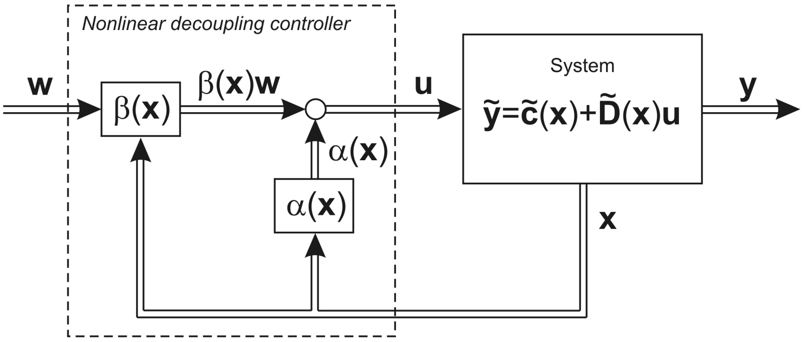


Figure 4.16.: Block diagram of nonlinear decoupling controller for integrated motion control The controller linearizes and decouples the dynamic system such that each output y_i or $\tilde{y}_i$ is only influenced by a single reference input w_i. The controller consists of a prefilter matrix $\boldsymbol{\beta}$ and feedback gain matrix $\boldsymbol{\beta}$.

Substituting the control law in Equation 4.6 into the uncontrolled dynamic system as described in Equation 4.3 results in the closed loop behavior

$$\tilde{\mathbf{y}} = \tilde{\mathbf{c}}(\mathbf{x}) + \tilde{\mathbf{D}}(\mathbf{x})\boldsymbol{\alpha}(\mathbf{x}) + \tilde{\mathbf{D}}(\mathbf{x})\boldsymbol{\beta}(\mathbf{x})\mathbf{w}(t). \tag{4.7}$$

It can be seen that the system output $\tilde{\mathbf{y}}$ is influenced by the reference input $\mathbf{w}(t)$ via $\tilde{\mathbf{D}}(\mathbf{x})$ and $\boldsymbol{\beta}(\mathbf{x})$. In order to obtain the desired characteristic that each $\tilde{y}_i$ is only influenced by w_i, the following property must hold:

$$\tilde{\mathbf{D}}(\mathbf{x})\boldsymbol{\beta}(\mathbf{x})\mathbf{w}(t) = \begin{bmatrix} \lambda_1 w_1(t) \\ \vdots \\ \lambda_q w_q(t) \end{bmatrix} = \tilde{\mathbf{\Lambda}}\mathbf{w}(t). \tag{4.8}$$

Thus, the prefilter matrix $\boldsymbol{\beta}(\mathbf{x})$ is given as

$$\boldsymbol{\beta}(\mathbf{x}) = \tilde{\mathbf{D}}(\mathbf{x})^{-1}\tilde{\mathbf{\Lambda}} \tag{4.9}$$

and the closed loop behavior becomes

$$\tilde{\mathbf{y}} = \tilde{\mathbf{c}}(\mathbf{x}) + \tilde{\mathbf{D}}(\mathbf{x})\boldsymbol{\alpha}(\mathbf{x}) + \tilde{\mathbf{\Lambda}}\mathbf{w}. \tag{4.10}$$

In Equation 4.10 it can be seen that $\tilde{y}_i$ might still be coupled with other output and reference elements via the state-dependent terms $\tilde{\mathbf{c}}(\mathbf{x})$ and $\tilde{\mathbf{D}}(\mathbf{x})\boldsymbol{\alpha}(\mathbf{x})$. To eliminate this possible coupling and to create a decoupled linear input-output relationship, the following ansatz is chosen, [Föllinger, 1993, Eq. 7.65f,7.86]:

$$\begin{aligned} \lambda_i w_i &= \overset{(\delta_i)}{y_i} + \alpha_{i\delta_i-1}\overset{(\delta_i-1)}{y_i} + \dots + \alpha_{i1}\dot{y}_i + \alpha_{i0}y_i \qquad (4.11) \\ & \quad \overset{(\delta_i)}{y_i} + \alpha_{i\delta_i-1}N^{\delta_i-1}c_i + \dots + \alpha_{i1}N^1 c_i + \alpha_{i0}N^0 c_i. \qquad (4.12) \end{aligned}$$

For the whole system of equations the ansatz function therefore reads

$$\tilde{\mathbf{y}} \;=\; -\underbrace{\begin{bmatrix} \sum_{k=0}^{\delta_1-1}\alpha_{1k}N^k c_1 \\ \vdots \\ \sum_{k=0}^{\delta_n-1}\alpha_{nk}N^k c_n \end{bmatrix}}_{\tilde{\mathbf{M}}(\mathbf{x})} + \tilde{\mathbf{\Lambda}}\mathbf{w}. \tag{4.13}$$

Comparing Equations 4.10 and 4.13 delivers the necessary form for $\boldsymbol{\alpha}(\mathbf{x})$ to achieve the desired linear decoupled closed loop system behavior,

$$\boldsymbol{\alpha}(\mathbf{x}) = -\tilde{\mathbf{D}}(\mathbf{x})^{-1}[\tilde{\mathbf{M}}(\mathbf{x}) + \tilde{\mathbf{c}}(\mathbf{x})]. \tag{4.14}$$

Having derived $\boldsymbol{\alpha}(\mathbf{x})$ and $\boldsymbol{\beta}(\mathbf{x})$, the control law in Equation 4.6 can be rewritten to

$$\mathbf{u}(t,\mathbf{x}) = \tilde{\mathbf{D}}(\mathbf{x})^{-1}[-\tilde{\mathbf{c}}(\mathbf{x}) + \tilde{\mathbf{\Lambda}}\mathbf{w}(t) - \tilde{\mathbf{M}}(\mathbf{x})] \tag{4.15}$$

with

$$\tilde{\mathbf{c}}(\mathbf{x}) = \begin{bmatrix} N^{\delta_1}c_1 \\ \vdots \\ N^{\delta_n}c_n \end{bmatrix}; \quad \tilde{\mathbf{D}}(\mathbf{x}) = \begin{bmatrix} \frac{\partial}{\partial\mathbf{x}}N^{\delta_1-1}c_1\mathbf{B}(\mathbf{x}) \\ \vdots \\ \frac{\partial}{\partial\mathbf{x}}N^{\delta_n-1}c_n\mathbf{B}(\mathbf{x}) \end{bmatrix}; \quad \mathbf{u} = \begin{bmatrix} u_1 \\ \vdots \\ u_q \end{bmatrix}$$

$$\tilde{\mathbf{\Lambda}} = \begin{bmatrix} \lambda_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \lambda_q \end{bmatrix}; \quad \tilde{\mathbf{M}}(\mathbf{x}) = \begin{bmatrix} \sum_{k=0}^{\delta_1-1}\alpha_{1k}N^k c_1(x) \\ \vdots \\ \sum_{k=0}^{\delta_n-1}\alpha_{nk}N^k c_n(x) \end{bmatrix}.$$

The following sections detail a suitable vehicle model and the application of the nonlinear decoupling to this model in order to achieve an integrated lateral and longitudinal trajectory following control.

4.3.1.1. Vehicle Model

The selected integrated lateral and longitudinal trajectory following control requires the modeling of the principal vehicle dynamics in the x-y-plane. Vertical dynamics as well as rolling or pitching motions are neglected. The dynamics of the drive train and the braking system are regarded only in design the subordinate acceleration controller.

If vertical, rolling, and pitching dynamics are neglected, planar vehicle models result, such as the two track model. Here the center of gravity is assumed to lie on the surface of the road and hence only three degrees of freedom remain for the vehicle body. A further reduction can be achieved by assuming a symmetry along the vehicle's longitudinal axis which results in the single track model, where the right and left tires are lumped together into a single tire per axle, see Figure 4.17. In particular, the single track model eliminates the distinction between a right and left steering angle δ_r, δ_l and only considers a single steering angle δ.

The Newton-Euler equations of motion for the single track model expressed in the Frenet frame $\underline{\mathrm{T}}$: $(T^*,\ {}_T\mathbf{e}_t,\ {}_T\mathbf{e}_n,\ {}_T\mathbf{e}_b)$ of reference, that is rotated by β with respect to $\underline{\mathrm{V}}$, are given by

$$\begin{aligned}
m\,{}^{E}_{T}\dot{v}^{CG}_{t} &= \left({}_{WR}F^{WR*}_{x} - {}_{V}F^{\mathrm{Wind}}_{x}\right)\cos(\beta) + {}_{WF}F^{WF*}_{x}\cos(\delta-\beta) \qquad (4.16)\\
&\quad +\left({}_{WR}F^{WR*}_{y} - {}_{V}F^{\mathrm{Wind}}_{y}\right)\sin(\beta) - {}_{WF}F^{WF*}_{y}\sin(\delta-\beta),\\
m\,{}^{R}_{V}\dot{v}^{CG}_{n} &= \left(-{}_{WR}F^{WR*}_{x} + {}_{V}F^{\mathrm{Wind}}_{x}\right)\sin(\beta) + {}_{WF}F^{WF*}_{x}\sin(\delta-\beta) \qquad (4.17)\\
&\quad +\left({}_{WR}F^{WR*}_{y} - {}_{V}F^{\mathrm{Wind}}_{y}\right)\cos(\beta) + {}_{WF}F^{WF}_{y}\cos(\delta-\beta),\\
\Theta^{CG}_{b}\ddot{\psi} &= -{}_{WR}F^{WR*}_{y}\,l_{R} + {}_{WF}F^{WF*}_{y}\,l_{F}\cos(\delta) + {}_{WF}F^{WF*}_{x}\,l_{F}\sin(\delta). \qquad (4.18)
\end{aligned}$$

Using the relation

$${}^{R}_{V}\dot{v}^{CG}_{n} = {}^{E}_{T}v^{CG}_{t}\left(\dot{\beta}+\dot{\psi}\right) \Rightarrow \dot{\beta} = \frac{{}^{R}_{V}\dot{v}^{CG}_{n}}{{}^{E}_{T}v^{CG}_{t}} - \dot{\psi} \tag{4.19}$$

to rewrite Equations 4.16-4.18, the dynamics of the single track model can be formulated in state space form, where the state vector

$$\mathbf{q} = \begin{bmatrix} q_1 & q_2 & q_3 & q_4 & q_5 & q_6 \end{bmatrix}^{\mathsf{T}} = \begin{bmatrix} \beta & \psi & \dot{\psi} & {}^{E}_{T}v^{CG}_{t} & {}_{E}x^{CG} & {}_{E}y^{CG} \end{bmatrix}^{\mathsf{T}} \tag{4.20}$$

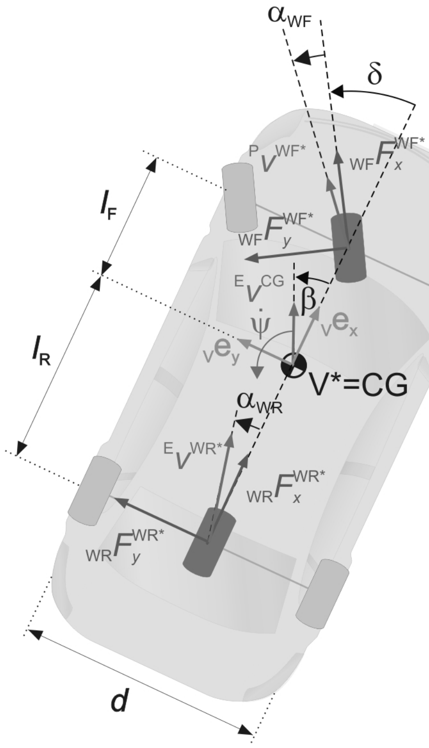


Figure 4.17.: Single track vehicle model
Vertical, rolling, and pitching dynamics are neglected, the center of gravity is assumed to lie on the surface of the road. Further, the right and left tires are lumped together into a single tire per axle, eliminating the distinction between the right and left steering angle.

describes the the system in terms of velocities and attitude, see [Mayr, 1991]:

$$\dot{\mathbf{q}} = \underbrace{\begin{bmatrix} \frac{1}{mq_4}\left[(-{}_{V}F_y^{\text{Wind}} + {}_{WR}F_y^{WR*})cos(q_1) + {}_{V}F_x^{\text{Wind}}\sin(q_1) + {}_{WF}F_x^{WF*}\sin(\delta - q_1)\right] - q_3 \\ q_3 \\ \frac{l_F}{\Theta_z^{CG}}\,{}_{WF}F_x^{WF*}\sin(\delta) - \frac{l_R}{\Theta_z^{CG}}\,{}_{WR}F_y^{WR*} \\ -\frac{1}{m}\,{}_{V}F_x^{\text{Wind}}cos(q_1) + \frac{1}{m}({}_{WR}F_y^{WR*} - {}_{V}F_y^{\text{Wind}})\sin(q_1) + \frac{1}{m}\,{}_{WF}F_x^{WF*}\cos(\delta - q_1) \\ q_4\cos(q_1 + q_2) \\ q_4\sin(q_1 + q_2) \end{bmatrix}}_{\mathbf{a}(\mathbf{q})}$$

$$+ \underbrace{\begin{bmatrix} \frac{\cos(\delta - q_1)}{mq_4} & -\frac{\sin(q_1)}{mq_4} \\ 0 & 0 \\ \frac{l_F\cos(\delta)}{\Theta_z^{CG}} & 0 \\ -\frac{\sin(\delta - q_1)}{m} & \frac{\cos(q_1)}{m} \\ 0 & 0 \\ 0 & 0 \end{bmatrix}}_{\mathbf{B}(\mathbf{q})}\mathbf{u}. \tag{4.21}$$

The input $\mathbf{u} = \begin{bmatrix} {}_{WF}F_y^{WF*} & {}_{WR}F_x^{WR*} \end{bmatrix}^T$ to the system is given by the lateral force at the front axle, ${}_{WF}F_y^{WF*}$, that results from steering and the longitudinal force at the rear axle, ${}_{WR}F_x^{WR*}$, that results from accelerating or braking.

In order to use this model in the controller design, Equation 4.21 is simplified and partly linearized analogously to [Freund and Mayr, 1997; Mayr, 1991] under the following assumptions: The angles β and δ are small, drive train and braking forces are applied only to the rear axle, and the lateral tire forces ${}_{WR}F_y^{WR*}$, ${}_{WF}F_y^{WF*}$ have no direct influence on the longitudinal acceleration ${}^E_T\dot{v}_t^{CG}$. The resulting simplified nonlinear system of equations is given by

$$\dot{\mathbf{q}} = \mathbf{a}\left(\mathbf{q}\right) + \mathbf{B}\left(\mathbf{q}\right)\mathbf{u}, \tag{4.22}$$

$$\mathbf{y} = \mathbf{c}\left(\mathbf{q}\right) \tag{4.23}$$

where $\mathbf{q}$ and $\mathbf{u}$ are the same as before, and $\mathbf{a}\left(\mathbf{q}\right)$, $\mathbf{B}\left(\mathbf{q}\right)$, and $\mathbf{c}\left(\mathbf{q}\right)$ are simplified to (compare [Mayr, 1991, Eq. 4.73], [Freund and Mayr, 1997, Eq. 2])

$$\mathbf{a}\left(\mathbf{q}\right) = \begin{bmatrix} \frac{1}{mq_4}\,{}_{WR}F_y^{WR*} + \frac{1}{mq_4}\,{}_{V}F_x^{\text{Wind}} q_1 - q_3 \\ q_3 \\ -\frac{l_R}{\Theta_z^{CG}}\,{}_{WR}F_y^{WR*} \\ -\frac{1}{m}\,{}_{V}F_x^{\text{Wind}} \\ q_4\cos(q_1+q_2) \\ q_4\sin(q_1+q_2) \end{bmatrix}, \mathbf{B}\left(\mathbf{q}\right) = \begin{bmatrix} \frac{1}{mq_4} & -\frac{q_1}{mq_4} \\ 0 & 0 \\ \frac{l_F}{\Theta_z^{CG}} & 0 \\ 0 & \frac{1}{m} \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \mathbf{c}\left(\mathbf{q}\right) = \begin{bmatrix} q_5 \\ q_6 \end{bmatrix} \tag{4.24}$$

4.3.1.2. Controller

In order to apply the control principle of nonlinear decoupling from Section 4.3.1, the simplified single track model in Equation 4.24 is first transformed into the form of Equation 4.3 resulting in

$$\tilde{\mathbf{y}} = \tilde{\mathbf{c}}(\mathbf{q}) + \tilde{\mathbf{D}}(\mathbf{q})\mathbf{u}(t) \tag{4.25}$$

with

$$\begin{aligned}
\mathbf{u} &= \begin{bmatrix} {}_{WR}F_x^{WR^*} & {}_{WF}F_y^{WF^*} \end{bmatrix}^{\mathsf{T}}, \\
\tilde{\mathbf{y}} &= \begin{bmatrix} {}_E\ddot{x}^{CG} & {}_E\ddot{y}^{CG} \end{bmatrix}^{\mathsf{T}}, \\
\tilde{\mathbf{c}}(\mathbf{q}) &= \begin{bmatrix} \frac{1}{m}(\cos(q_1+q_2) - q_1\sin(q_1+q_2))\,{}_VF_x^{\text{Wind}} - \frac{1}{m}\sin(q_1+q_2)\,{}_{WR}F_y^{WR*} \\ \frac{1}{m}(\sin(q_1+q_2) + q_1\cos(q_1+q_2))\,{}_VF_x^{\text{Wind}} + \frac{1}{m}\cos(q_1+q_2)\,{}_{WR}F_y^{WR*} \end{bmatrix}, \\
\tilde{\mathbf{D}}(\mathbf{q}) &= \begin{bmatrix} -\frac{1}{m}\sin(q_1+q_2) & \frac{1}{m}(q_1\sin(q_1+q_2) + \cos(q_1+q_2)) \\ \frac{1}{m}\cos(q_1+q_2) & \frac{1}{m}(-q_1\cos(q_1+q_2) + \sin(q_1+q_2)) \end{bmatrix},
\end{aligned}$$

Then the derived control law from Section 4.3.1, Equation 4.15 is applied yielding

$$\mathbf{u}(t,\mathbf{q}) = \tilde{\mathbf{D}}(\mathbf{q})^{-1}[-\tilde{\mathbf{c}}(\mathbf{q}) + \tilde{\mathbf{\Lambda}}\mathbf{w}(t) - \tilde{\mathbf{M}}(\mathbf{q})] \tag{4.26}$$

with

$$\begin{aligned}
\mathbf{u} &= \begin{bmatrix} {}_{WR}F_x^{WR^*} & {}_{WF}F_y^{WF^*} \end{bmatrix}^{\mathsf{T}}, \\
\mathbf{w} &= \begin{bmatrix} {}_Ex_{\text{ref}}^{CG} & {}_Ey_{\text{ref}}^{CG} \end{bmatrix}^{\mathsf{T}}, \\
\tilde{\mathbf{D}}^{-1}(\mathbf{q}) &= \begin{bmatrix} -m(-q_1\cos(q_1+q_2) + \sin(q_1+q_2)) & m(q_1\sin(q_1+q_2) + \cos(q_1+q_2)) \\ m\ \cos(q_1+q_2) & m\ \sin(q_1+q_2) \end{bmatrix}, \\
\tilde{\mathbf{c}}(\mathbf{q}) &= \begin{bmatrix} \frac{1}{m}(\cos(q_1+q_2) - q_1\sin(q_1+q_2))\,{}_VF_x^{\text{Wind}} - \frac{1}{m}\sin(q_1+q_2)\,{}_{WR}F_y^{WR*} \\ \frac{1}{m}(\sin(q_1+q_2) + q_1\cos(q_1+q_2))\,{}_VF_x^{\text{Wind}} + \frac{1}{m}\cos(q_1+q_2)\,{}_{WR}F_y^{WR*} \end{bmatrix}, \\
\tilde{\mathbf{M}}(\mathbf{q}) &= \begin{bmatrix} \alpha_{10}q_5 + \alpha_{11}q_4\cos(q_1+q_2) \\ \alpha_{20}q_6 + \alpha_{21}q_4\sin(q_1+q_2) \end{bmatrix}, \\
\tilde{\mathbf{\Lambda}} &= \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}
\end{aligned}$$

The combination of the controller and the vehicle is depicted later in Figure 4.18. As derived earlier in Section 4.3.1, the closed loop input-output behavior is given by

$$\tilde{\mathbf{y}} \;=\; -\tilde{\mathbf{M}}(\mathbf{q}) + \tilde{\mathbf{\Lambda}}\mathbf{w}, \tag{4.27}$$

which in this case yields

$${}_E\ddot{x}^{CG} + \alpha_{11}\,{}_E\dot{x}^{CG} + \alpha_{10}\,{}_Ex^{CG} \;=\; \lambda_1\,{}_Ex_{\text{ref}}^{CG}, \tag{4.28}$$

$${}_E\ddot{y}^{CG} + \alpha_{21}\,{}_E\dot{y}^{CG} + \alpha_{20}\,{}_Ey^{CG} \;=\; \lambda_2\,{}_Ey_{\text{ref}}^{CG}. \tag{4.29}$$

As can be seen in Equations 4.28, the closed loop system dynamics are now linear and lateral and longitudinal motions have been successfully decoupled. The equations still contain six parameters that have to be chosen to tune the trajectory following controller that can be used to ensure a certain system behavior. As first approach, lateral and longitudinal dynamics can be weighted equally,

$$\lambda_1 = \lambda_2 = \lambda, \tag{4.30}$$

however, λ_1 and λ_2 will also be considered separately further on to preserve the possibility to tune lateral vs. longitudinal dynamics later on. Further, in order to obtain stationary accuracy, it must be demanded that

$$\alpha_{10} = \lambda_1 \quad (= \lambda), \tag{4.31}$$
$$\alpha_{20} = \lambda_2 \quad (= \lambda). \tag{4.32}$$

Finally, the system behavior can be analyzed analogously to a general PT2 element which has the form

$$\ddot{y}_i + D\omega_0 \dot{y}_i + \omega_0^2 y = \omega_0^2 w_i. \tag{4.33}$$

Following this comparison, critical damping $D = 1$ is desired, yielding

$$\alpha_{11} = 2\sqrt{\lambda_1} \quad (= 2\sqrt{\lambda}), \tag{4.34}$$
$$\alpha_{21} = 2\sqrt{\lambda_2} \quad (= 2\sqrt{\lambda}). \tag{4.35}$$

Following this proposed parametrization, the system has two double Eigenvalues at $-\sqrt{\lambda_1}$ and $-\sqrt{\lambda_2}$ on the negative real axis and only two parameters λ_1 and λ_2 have to be tuned. (In case $\lambda_1 = \lambda_2 = \lambda$, a quadruple Eigenvalue at $-\sqrt{\lambda}$ results and only a single parameter λ has to be tuned.) Therefore, (within the limited posed by the underlying assumptions of the model) the system is input-output stable as long as $\lambda_1 > 0 \wedge \lambda_2 > 0$, [Föllinger, 2008]. Within given hardware constraints, the reactivity of the system can be tuned by adapting λ_1, λ_2.

Note, that since the differential degree $\delta = 4$ is smaller than the order of the nonlinear system $n = 6$, there exist "hidden" internal dynamics which are not observable anymore in the linearized and decoupled closed-loop system, [Khalil, 2002, p. 517], which are not covered by the stability analysis above. A special case of these internal dynamics is also known as zero dynamics, [Svaricek, 2006, Def. 3.1]. As suggested for practical applications for this case by [Föllinger, 1993, p. 315], the feasibility of the control approach was tested in simulations and experiments and no unbounded increase in any of the system states was observed. Future research could examine this in more depth.

The trajectory following controller as designed so far produces a desired linear and decoupled closed loop system behavior and tracks a given reference $\left[\,{}_{E}x^{CG}_{\text{ref}} \quad {}_{E}y^{CG}_{\text{ref}}\right]^{\mathsf{T}}$ without stationary error for ${}^{E}_{V}v^{CG}_{t} = 0$, ${}^{E}_{V}\dot{v}^{CG}_{t} = 0$. However, depending on velocity and acceleration, a lag error e_{lag} occurs in both lateral and longitudinal direction,

$$
\begin{aligned}
e_{\text{lag},x} \quad &= \quad {}_{E}x^{CG}_{\text{ref}} - {}_{E}x^{CG} = \frac{1}{\lambda_1}\left({}^{E}_{V}\ddot{x}^{CG} + \alpha_{11}\,{}^{E}_{V}\dot{x}^{CG}\right), && (4.36)\\
e_{\text{lag},y} \quad &= \quad {}_{E}y^{CG}_{\text{ref}} - {}_{E}y^{CG} = \frac{1}{\lambda_2}\left({}^{E}_{V}\ddot{y}^{CG} + \alpha_{21}\,{}^{E}_{V}\dot{y}^{CG}\right). && (4.37)
\end{aligned}
$$

This lag error can be compensated by adapting the reference variables and replacing ${}_{E}x^{CG}_{\text{ref}}$ and ${}_{E}y^{CG}_{\text{ref}}$ in the control law by ${}_{E}\hat{x}^{CG}_{\text{ref}}$ and ${}_{E}\hat{y}^{CG}_{\text{ref}}$, respectively, where

$$
\begin{aligned}
{}_{E}\hat{x}^{CG}_{\text{ref}} \quad &= \quad {}_{E}x^{CG}_{\text{ref}} + \frac{1}{\lambda_1}\left({}^{E}_{V}\ddot{x}^{CG}_{\text{ref}} + \alpha_{11}\,{}^{R}_{V}\dot{x}^{CG}_{\text{ref}}\right), && (4.38)\\
{}_{E}\hat{y}^{CG}_{\text{ref}} \quad &= \quad {}_{E}y^{CG}_{\text{ref}} + \frac{1}{\lambda_2}\left({}^{E}_{V}\ddot{y}^{CG}_{\text{ref}} + \alpha_{21}\,{}^{E}_{V}\dot{y}^{CG}_{\text{ref}}\right). && (4.39)
\end{aligned}
$$

Reformulating Equations 4.38 and 4.39 in terms of the vehicle's reference tangential velocity ${}^{E}v^{CG}_{t,\text{ref}}$ yields

$$
\begin{aligned}
{}^{E}\hat{x}^{CG}_{\text{ref}} \quad &= {}^{E}x^{CG}_{\text{ref}} + \tfrac{\alpha_{11}}{\lambda_1}\,{}^{E}\mathbf{v}^{CG}_{t,\text{ref}}\cos\left(\nu_{\text{ref}}\right) + \tfrac{1}{\lambda_1}\,{}^{E}\dot{v}^{CG}_{t,\text{ref}}\cos\left(\nu_{\text{ref}}\right) - \tfrac{1}{\lambda_1}\,{}^{E}v^{CG}_{t,\text{ref}}\dot{\nu}_{\text{ref}}\sin\left(\nu_{\text{ref}}\right), && (4.40)\\
{}^{E}\hat{y}^{CG}_{\text{ref}} \quad &= \underbrace{{}^{E}y^{CG}_{\text{ref}}}_{\text{position}} + \underbrace{\tfrac{\alpha_{21}}{\lambda_2}\,{}^{E}v^{CG}_{t,\text{ref}}\sin\left(\nu_{\text{ref}}\right)}_{\text{velocity}} + \underbrace{\tfrac{1}{\lambda_2}\,{}^{E}\dot{v}^{CG}_{t,\text{ref}}\sin\left(\nu_{\text{ref}}\right)}_{\text{long. acceleration}} + \underbrace{\tfrac{1}{\lambda_2}\,{}^{E}v^{CG}_{t,\text{ref}}\dot{\nu}_{\text{ref}}\cos\left(\nu_{\text{ref}}\right)}_{\text{lat. acceleration}}, && (4.41)
\end{aligned}
$$

where ν_{ref} denotes the reference course angle of the planned trajectory. Applying Equations 4.34, 4.35 the closed loop system dynamics (compare Equations 4.28f) now read

$$
\begin{aligned}
{}_{E}\ddot{x}^{CG} + 2\sqrt{\lambda_1}\,{}_{E}\dot{x}^{CG} + \lambda_1\,{}_{E}x^{CG} \quad &= \quad \lambda_1\,{}_{E}\hat{x}^{CG}_{\text{ref}}\,, && (4.42)\\
{}_{E}\ddot{y}^{CG} + 2\sqrt{\lambda_2}\,{}_{E}\dot{y}^{CG} + \lambda_2\,{}_{E}y^{CG} \quad &= \quad \lambda_2\,{}_{E}\hat{y}^{CG}_{\text{ref}}\,. && (4.43)
\end{aligned}
$$

As can be seen, the control law in Equation 4.26ff contains the longitudinal tire force at the rear axle ${}_{WR}F_x^{WR^*}$ as well as the lateral tire forces at front and rear axle ${}_{WF}F_y^{WF^*}$, ${}_{WR}F_y^{WR^*}$. Even though these forces can be expressed in dependence of the system state $\mathbf{q}$ using a tire model, such a substitution has not been necessary until this point. This property is valuable since at this point any desired tire model can be used.

One way to describe the tire characteristics is to employ experimental tire data. The drawback of this method is that data for all environmental conditions are necessary. For this reason, the use of models that describe the physics of tire road interaction is common and many different tire models for different fields of application exist and are still subject to numerous research activities.

Frequently used models include the Pacejka's magic tire formula, see [Bakker et al., 1989; Pacejka, 2006; Pacejka and Bakker, 1992], which essentially is a curve fitting approach to tire test data with a rather high number of parameters to be fitted. Another widespread approach, with less parameters, is the tire model of Burckhardt, see [Burckhardt, 1993]. Even less parameters (and, according to a comparison made in [Halfmann, 2003], less computational effort) are necessary for the Dugoff model, a physical model of the tire road interaction introduced by Dugoff, Fancher and Segel, [Dugoff et al., 1970].

In this case, an even simpler linear tire model is used for the lateral tire forces,

$$\alpha_{Wq} \;=\; \frac{1}{C_\alpha^q}\,{}_{Wq}F_y^{Wq^*}, \qquad q \in \{R, L\}\,. \tag{4.44}$$

In unstable situations, where the rear tire slip angle α_{WR} becomes high and leaves the linear region, the lateral tire force at the rear axle ${}_{WR}F_y^{WR^*}$ in the control law is overestimated, however, according to [Mayr, 1991] this has a positive effect on the controlled vehicle, since $\partial u_1/\partial\, {}_{WR}F_y^{WR^*} < 0$. Therefore u_1 is reduced, the vehicle becomes more stable and the tire slip angle leaves the nonlinear region.

The front tire slip angle α_{WF} is substituted according to the kinematics

$$\alpha_{WF} = \beta + \frac{l_F}{{}_V^R v_t^{CG}}\dot{\psi} - \delta. \tag{4.45}$$

Substituting Equation 4.44 into 4.45 yields

$$\delta_{\text{des}} \;=\; \beta + \frac{l_F}{{}_V^R v_t^{CG}}\dot{\psi} - \frac{1}{C_\alpha^F}u_1. \tag{4.46}$$

The conversion between $u_2 = {}_{WR}F_x^{WR^*}$ and $a_{x,\text{des}}$ is given by

$$a_{x,\text{des}} = \frac{1}{m} u_2 . \tag{4.47}$$

Note, that the real relation between the two would have to include additional components such as the rotational inertia of wheels and axles or the transfer function of the torque converter, see Section 4.3.3.1. However, since the controller was designed based on the single track model, Equation 4.47 produces the desired acceleration that coincides with the desired closed loop vehicle dynamics.

As a further substitution, the force ${}_V F_x^{\text{Wind}}$, which is caused by the air resistance, can be expressed in terms of the vehicle's longitudinal velocity as

$${}_V F_x^{\text{Wind}} = \frac{1}{2} c_w \rho_{\text{air}} A_w \left({}^E v_t^{CG} \cos(\beta) \right)^2 . \tag{4.48}$$

Therein, c_w denotes the drag coefficient, ρ_{air} the air density, and A_w denotes the reference area.

The complete control structure is displayed in Figure 4.18 and the integrated lateral and longitudinal control is summarized in the following box.

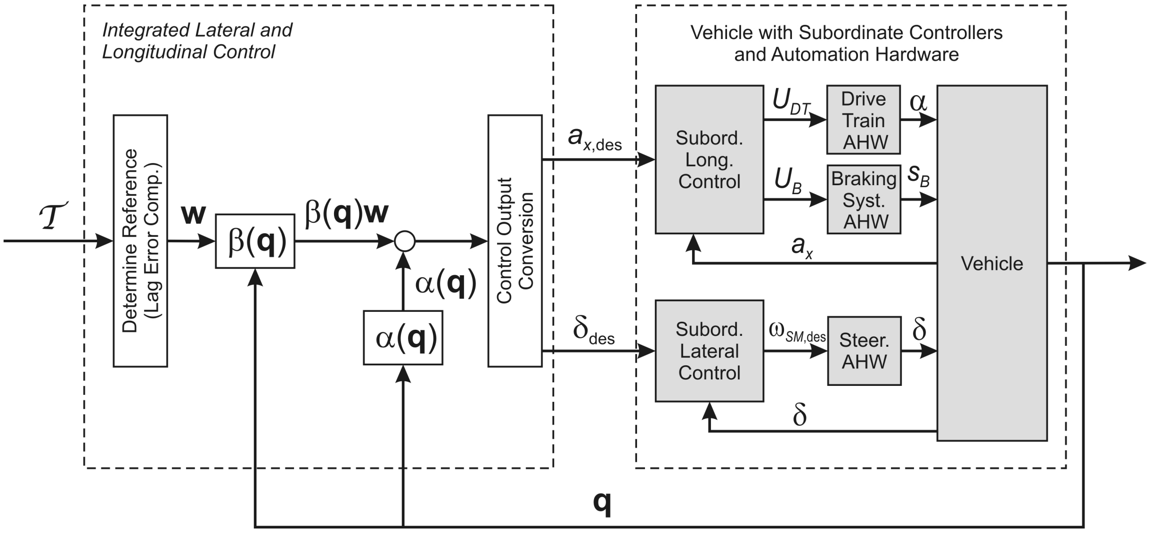


Figure 4.18.: Block diagram of implemented integrated motion control
The controller linearizes and decouples the dynamic system. The controller consists of a prefilter matrix $\boldsymbol{\beta}$ and feedback gain matrix $\boldsymbol{\beta}$.

Inputs to Trajectory Following Controller

$$\begin{aligned} \text{reference:} \quad & {}_{E}x^{CG}_{\mathrm{ref}},\ {}_{E}y^{CG}_{\mathrm{ref}},\ {}^{E}v^{CG}_{t,\mathrm{ref}},\ {}^{E}\dot{v}^{CG}_{t,\mathrm{ref}}, \nu_{\mathrm{ref}}, \dot{\nu}_{\mathrm{ref}} \quad \text{from} \quad \mathcal{T} \\ \text{vehicle state:} \quad & \mathbf{q} = \begin{bmatrix} {}_{E}x^{CG} & {}_{E}y^{CG} & {}^{E}v^{CG}_{t} & \psi & \dot{\psi} & \beta \end{bmatrix}^{\mathsf{T}} \end{aligned}$$

Outputs of Trajectory Following Controller

reference for subordinate control: $\delta_{\mathrm{des}}, a_{x,\mathrm{des}}$

Parameters

$$\begin{aligned} \text{vehicle parameters:} \quad & m, l_F, l_R, C^F_\alpha, C^R_\alpha, c_w, A_w \\ \text{controller parameters:} \quad & \lambda_1, \lambda_2 \\ \text{other parameters:} \quad & \rho_{\mathrm{air}} \end{aligned}$$

Reference Input

$$\begin{aligned} \hat{w}_1 &= {}_{E}\hat{x}^{CG}_{\mathrm{ref}} = {}_{E}x^{CG}_{\mathrm{ref}} + \frac{2}{\sqrt{\lambda_1}}\,{}^{E}v^{CG}_{t,\mathrm{ref}} \cos\left(\nu_{\mathrm{ref}}\right) + \frac{1}{\lambda_1}\,{}^{E}\dot{v}^{CG}_{t,\mathrm{ref}} \cos\left(\nu_{\mathrm{ref}}\right) - \frac{1}{\lambda_1}\,{}^{E}v^{CG}_{t,\mathrm{ref}}\dot{\nu}_{\mathrm{ref}} \sin\left(\nu_{\mathrm{ref}}\right) \\ \hat{w}_2 &= {}_{E}\hat{y}^{CG}_{\mathrm{ref}} = {}_{E}y^{CG}_{\mathrm{ref}} + \frac{2}{\sqrt{\lambda_2}}\,{}^{E}v^{CG}_{t,\mathrm{ref}} \sin\left(\nu_{\mathrm{ref}}\right) + \frac{1}{\lambda_2}\,{}^{E}\dot{v}^{CG}_{t,\mathrm{ref}} \sin\left(\nu_{\mathrm{ref}}\right) + \frac{1}{\lambda_2}\,{}^{E}v^{CG}_{t,\mathrm{ref}}\dot{\nu}_{\mathrm{ref}} \cos\left(\nu_{\mathrm{ref}}\right) \end{aligned}$$

Control Law

$$\begin{aligned} u_1 &= {}_{WF}F^{WF*}_{y} = -C^R_\alpha \left(\beta - \frac{l_R}{{}^{E}v^{CG}_{t}}\dot{\psi}\right) + m[(\beta\cos(\beta+\psi) - \sin(\beta+\psi))y_{1pp} \\ &\quad +(\beta\sin(\beta+\psi) + \cos(\beta+\psi))y_{2pp}] \\ u_2 &= {}_{WR}F^{WR*}_{x} = \frac{1}{2}c_w\rho_{\mathrm{air}}A_w\left({}^{E}v^{CG}_{t}\cos(\beta)\right)^2 + m[\cos(\beta+\psi)y_{1pp} + \sin(\beta+\psi)y_{2pp}] \end{aligned}$$

with

$$\begin{aligned} y_{1pp} &= \lambda_1\left({}_{E}\hat{x}^{CG}_{\mathrm{ref}} - {}_{E}x^{CG}\right) - 2\sqrt{\lambda_1}\,{}^{E}v^{CG}_{t}\cos(\psi+\beta) \\ y_{2pp} &= \lambda_2\left({}_{E}\hat{y}^{CG}_{\mathrm{ref}} - {}_{E}y^{CG}\right) - 2\sqrt{\lambda_2}\,{}^{E}v^{CG}_{t}\sin(\psi+\beta) \end{aligned}$$

Translation of Controller Output

$$\begin{aligned} \delta_{\mathrm{des}} &= \beta + \frac{l_F}{{}^{E}_{V}v^{CG}_{t}}\dot{\psi} - \frac{1}{C^F_\alpha}\,{}_{WF}F^{WF*}_{y} = \beta + \frac{l_F}{{}^{E}_{V}v^{CG}_{t}}\dot{\psi} - \frac{1}{C^F_\alpha}u_1 \\ a_{x,\mathrm{des}} &= \frac{1}{m}\,{}_{WR}F^{WR*}_{x} = \frac{1}{m}u_2 \end{aligned}$$

4.3.2. Subordinate Lateral Control

A subordinate control loop ensures that the desired steering angle δ_{des} is set. As shown in Figure 4.19, depending on the difference e_δ between the desired steering angle δ_{des} and the measured one δ, a desired rotational motor velocity $\omega_{SM,\text{des}}$ is set as the input to the steering automation hardware (AHW) module. There, two further cascaded control loops exist to control the motor torque $T_{SW,M}$ in such a way that the desired rotational velocity is achieved.

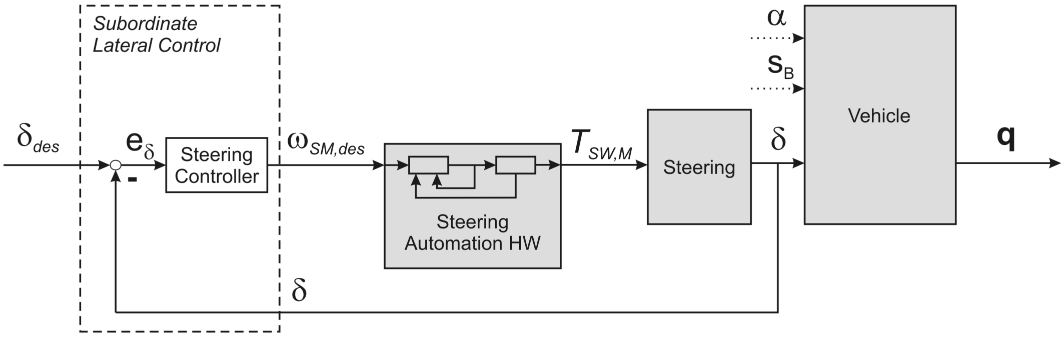


Figure 4.19.: Block diagram of steering control
The subordinate lateral control comprises a steering controller that sets a desired rotational motor velocity $\omega_{SM,\text{des}}$ depending on the steering angle error e_δ.

Even though the steering hardware shows some compliance, the connection is still very direct and the closed loop dynamics of steering hardware and steering automation hardware module are sufficiently fast. For the design of the steering controller, a detailed modeling was not performed. Therefore, a PID-type controller was selected and tuned experimentally supported by a student thesis, see [Hesse, 2007].

4.3.3. Subordinate Longitudinal Control

The model of a vehicle's longitudinal dynamics can be separated into three main parts: the vehicle body, the drive train, and the braking system.

For the design of the integrated lateral and longitudinal control in the outer control loop only simplified dynamics of the vehicle body have been regarded.In order to implement an effective subordinate longitudinal control, however, that controls the desired longitudinal acceleration $a_{x,\text{des}}$, also the dynamics of the drive train and the braking system have to be included.

4.3.3.1. Drive Train and Braking System Model

A detailed theoretical modeling of the longitudinal dynamics can be found for example in [Germann, 1997]. Germann concludes that the resulting model is much too complicated to be used for most applications. Therefore, he devises a simplified longitudinal model which is shown in Figure 4.20.

Three main parts comprise the model: the drive train, the braking system, and the vehicle body. Further, the slope of the street is regarded as external influence. The central idea is to determine the different longitudinal forces that act on the vehicle body and accelerate it. Besides the gravity force F_g for slopes $\alpha_{\text{slope}} \neq 0$, drive train and braking system generate the forces F_{DT} and F_B depending on the throttle flap angle α and the brake pressure p_B in the main brake cylinder as respective model inputs. In addition, the vehicle body itself causes a force F_{VB} due to its inertia and wind and roll resistance. The output of the model is the vehicles longitudinal acceleration a_x that is then integrated over time to acquire the current velocity v_x.

The *drive train* model includes the dynamics of the throttle flap, the intake manifold, and the motor. At low velocities, when the torque converter clutch (TCC) is still open, the torque converter is also accounted for. The stationary transfer behaviors of intake manifold/motor and converter are modeled as nonlinear maps depending on the rotational velocities ω_M, ω_T of motor (driveshaft) and turbine (driven shaft), respectively. If the TCC is closed, the turbine torque T_T is equal to the motor torque T_M. The dynamics are modeled by PT2 second order delay elements. The torque T_T is converted into a force F_{DT} depending on the gear ratio i_{Gear}, the rear axle transmission ratio i_{RAT}, and the dynamic roll radius R_{dyn}.

The *braking system* is modeled linearly as a PT2 second order delay element. The dynamic roll radius R_{dyn} translates the braking torque T_B into the braking force F_B that decelerates the car.

The total longitudinal force F_x, comprised of F_B, F_{DT}, F_g, and F_{VB}, is the input to the *vehicle body*. To calculate the acceleration a_x of the vehicle body from the total force at the vehicle tires, a proportional element is used that contains the effective vehicle mass m_{eff}, the moment of inertia Θ^* of the motor, the gear ratio i_{Gear}, and the torque converter efficiency $F\left(\omega_T/\omega_M\right)$. The velocity depending vehicle body force F_{VB} models the aerodynamic resistance $c_x v_x^2$ and the roll resistance $k_R v_x$ which decelerate the vehicle, where k_R and $c_x = c_w \frac{\rho A}{2}$ represent the roll and drag coefficients, respectively. $\Theta^* = \frac{i_{RAT}^2}{R_{\text{dyn}}^2} \Theta_M$ depends on the motor moment of inertia Θ_M, i_{RAT}, and R_{dyn}. The effective mass m_{eff} allows to

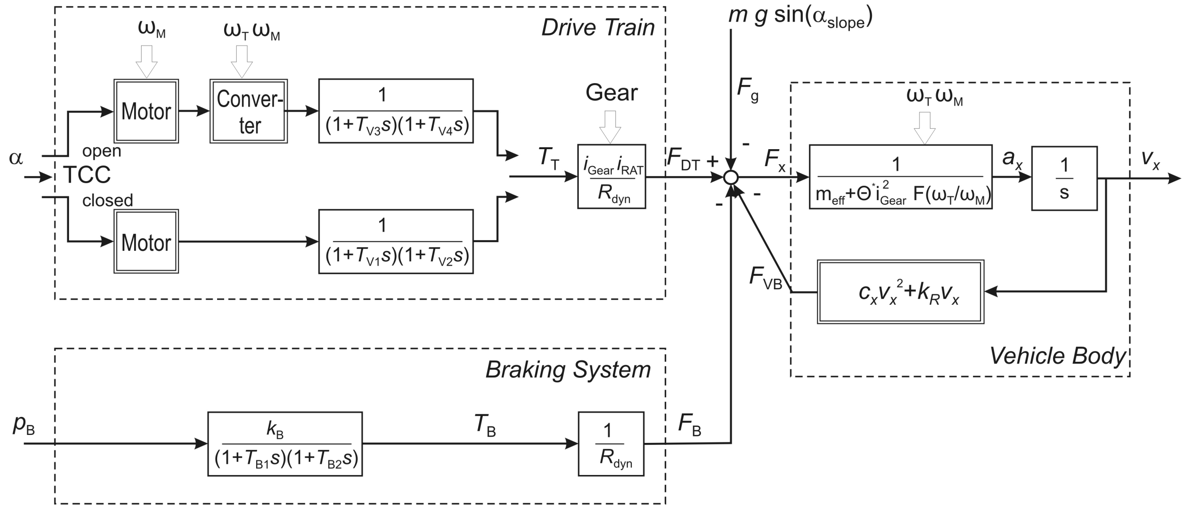


Figure 4.20.: Simplified theoretical model of longitudinal vehicle dynamics according to [Germann, 1997, p. 77]
The model determines the acceleration resulting from different longitudinal forces by the drive train, the braking system, the slope of the street, and the vehicle body (wind and friction forces).

describe the longitudinal inertia of the vehicle in a simple single term. It lumps together the real vehicle mass and the moments of inertia of all rotating elements that are directly connected to the wheels.

The above model offers some valuable insight into the (simplified) structure of the longitudinal dynamics. Unfortunately, it was not possible to obtain data about the nonlinear characteristics of the engine and the torque converter for the BMW 540i test vehicle. As investigated in supervised student theses, they could not be measured separately and the model was simplified even further in to an almost purely experimental model, [Büchsenschütz, 2009; Hesse, 2008].

The transfer characteristic between the inputs U_{DT} (to the drive train) and U_B (to the braking system) and the acceleration a_x as output is modeled by velocity dependent characteristic maps, as displayed in Figure 4.21.

The *drive train map* models the static transfer characteristic from U_{DT} to a_x when only the drive train is used and $U_B = 0$. It includes the drive train AHW, the drive train, and the vehicle body. The map differs for each gear and it is velocity dependent for two reasons: First, it includes the velocity dependent wind resistance that is modeled in the vehicle body in Figure 4.20. Second, the torque converter behavior and the point where the TCC closes depend on the rotational velocities ω_M, ω_T of motor (driveshaft) and turbine (driven shaft),

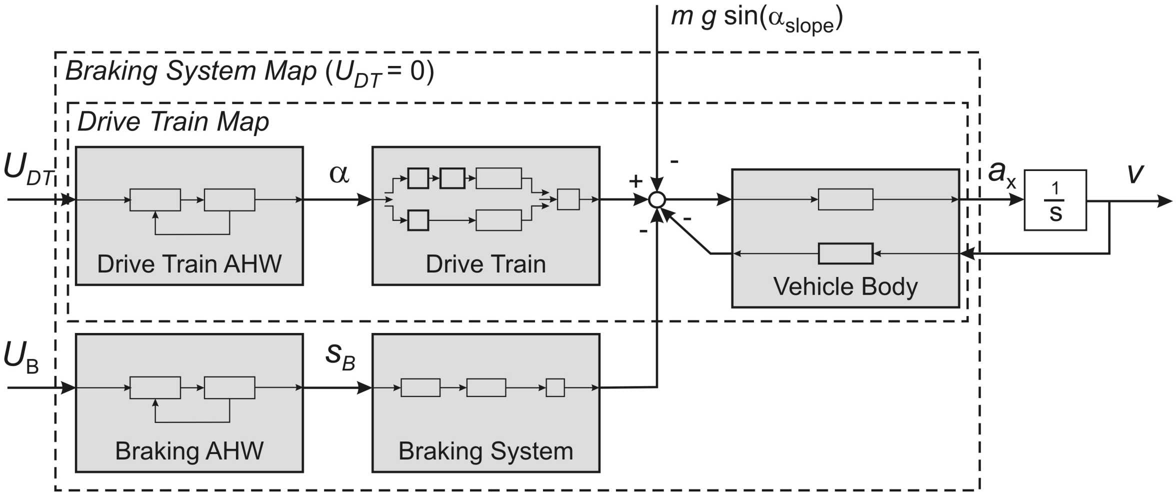


Figure 4.21.: Characteristic maps model of longitudinal vehicle dynamics
The experimentally recorded characteristic maps model the static transfer characteristics of different parts of the longitudinal dynamics.

respectively. These in turn mainly depend on the throttle flap angle α which depends on the input U_{DT} and the vehicle's velocity v.

The *braking system map* models the static transfer characteristic from U_B to a_x when the brake is used and $U_{DT} = 0$. In contrast to the drive train map, $U_{DT} = 0$ does not mean that the drive train has no effect anymore, because it is still connected to the tires. Therefore, it decelerates the car at higher velocities (engine brake). Further, the drive train still accelerates the car at low velocities even at $U_{DT} = 0$, $\alpha = 0$ due to the idling mixture supply. (Even when the throttle flap is closed completely, a small stream of air accesses the motor through a bypass to prevent the motor from stalling.) The braking system map includes the drive train AHW ($U_{DT} = 0$), the drive train ($\alpha = 0$), the braking system AHW, the braking system, and the vehicle body. Compared to the model of Germann in Figure 4.20, the braking system includes not only the pipes and brakes but also the brake pedal, brake booster and main brake cylinder. Therefore, the input to this part is now the brake pedal position s_B instead of the brake pressure p_B.

The influence of the current slope is not included in either map as both maps model $\alpha_{\text{slope}} = 0$. Instead, the slope of the track is regarded as a disturbance.

The maps have been acquired by extensive experiments carried out, see [Büchsenschütz, 2009]. To obtain the braking system map, test drives where made with step inputs of varying amplitude $U_B = U_{B,i}$ at a certain initial velocity, see Figure 4.22.

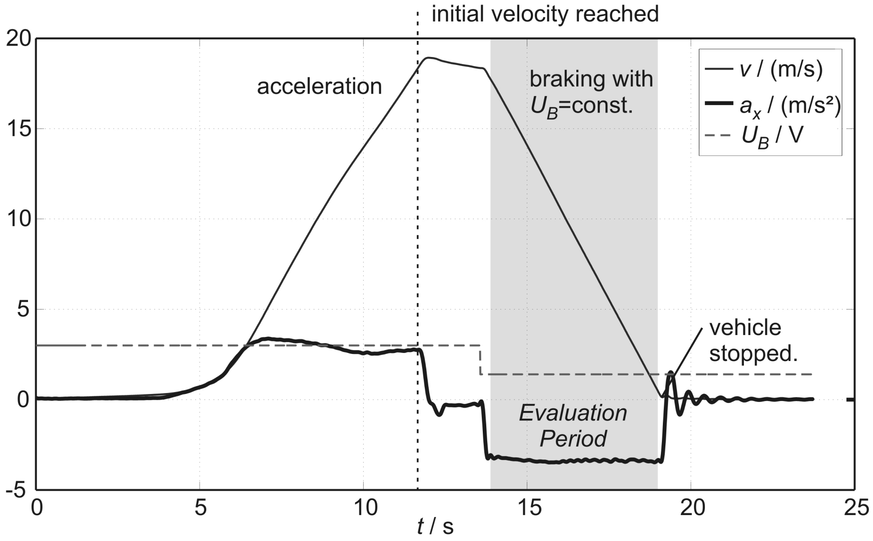


Figure 4.22.: Experiment for braking system map

During braking at constant input voltage U_B, the vehicle's (negative) acceleration is recorded depending on the velocity.

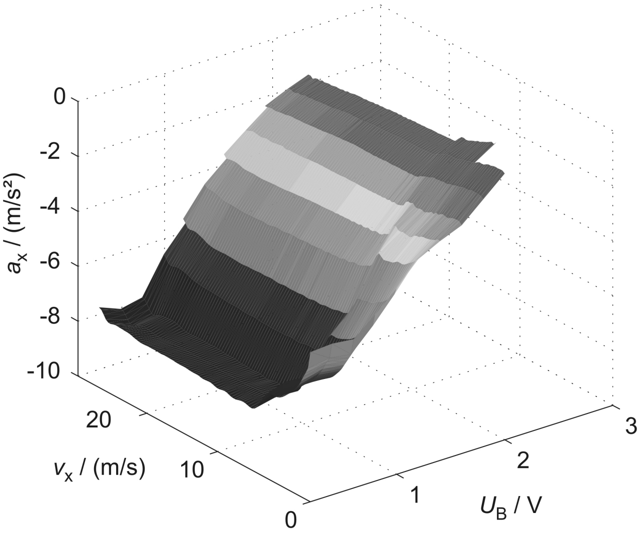


Figure 4.23.: Braking system map for second gear

The recorded curves for $a_x(v)$ for different constant voltages U_B are put together to a three dimensional map $a_x(v, U_B)$.

The vehicle decelerates until it stops. The initial transient and the measured oscillations after stopping, which are due to pitching movements of the vehicle body, are disregarded. The remaining measurements within the evaluation period provide map data for the velocity

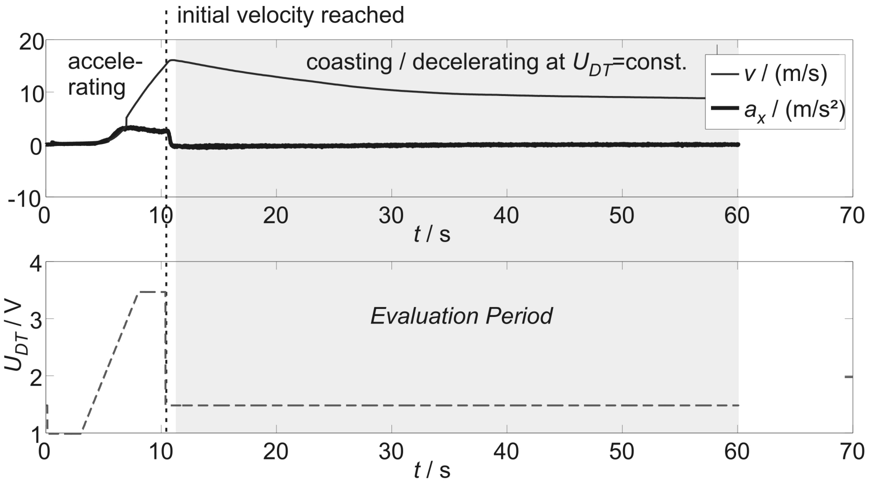


Figure 4.24.: First experiment for drive train map
The vehicle decelerates at a constant voltage U_{DT} and the vehicle's (negative) acceleration is recorded depending on the velocity.

dependent (negative) acceleration for a constant $U_B = U_{B,i}$. Combining the measurements for several amplitudes and interpolating between them provides a three dimensional map as shown for second gear in Figure 4.23.

In order to obtain the drive train map, two different kinds of experiments had to be performed, since the drive train can cause positive and negative accelerations depending on the current velocity and the throttle flap angle. For high velocities and small throttle flap angles, the vehicle decelerates. This part of the drive train map is explored as shown in Figure 4.24.

First the vehicle accelerates to an initial velocity by fully opening the throttle flap. Then the throttle flap is closed again partly according to a step downwards to a certain $U_{DT} = U_{DT,i}$ which is then kept constant. As a consequence, the vehicle slows down until the deceleration declines and a final velocity is approached. Again the initial transient is disregarded. The measurements recorded during the evaluation period provide map data for the velocity dependent (negative) acceleration for a constant input $U_{DT} = U_{DT,i}$.

The second experiment investigates positive accelerations due to the drive train, see Figure 4.25. This time the vehicle drives slowly due to the idling mixture supply at $U_{DT} = 0$ and then a step function with the amplitude $U_{DT} = U_{DT,i}$ is applied to accelerate the vehicle until a final velocity is approached or the experiment is aborted because the end of the test track is reached. Again discarding the initial transient, the measured data provides velocity dependent accelerations for a constant input $U_{DT} = U_{DT,i}$.

The data from these two types of experiments is joined and interpolated to create a three dimensional map as shown for second gear in Figure 4.26.

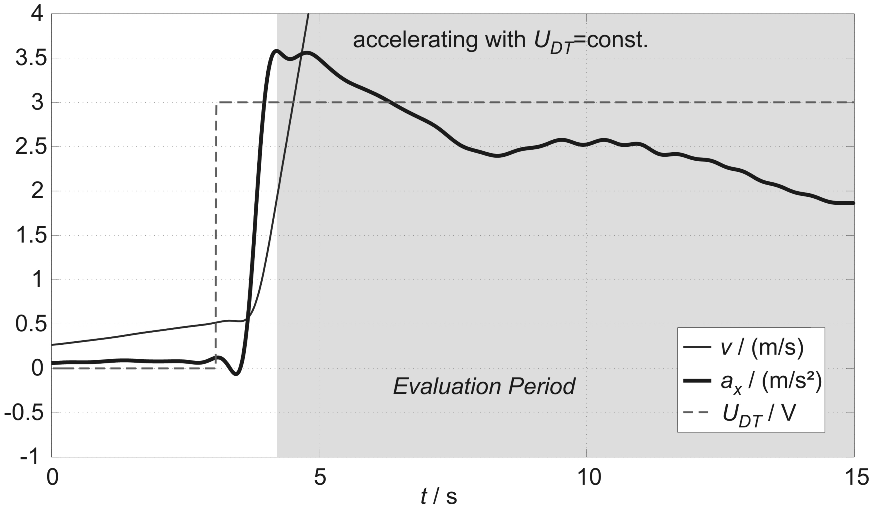


Figure 4.25.: Second experiment for drive train map

The vehicle accelerates at a constant voltage U_{DT} and the vehicle's acceleration is recorded depending on the velocity.

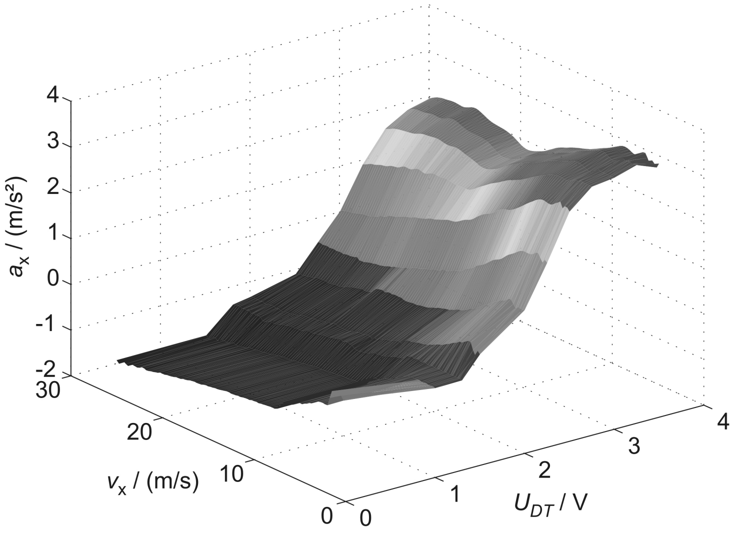


Figure 4.26.: Drive train map for second gear

The recorded curves for $a_x(v)$ for different constant voltages U_{DT} are put together to a three dimensional map $a_x(v, U_{DT})$.

4.3.3.2. Controller

The subordinate longitudinal control consists of an acceleration controller and a mode selector.

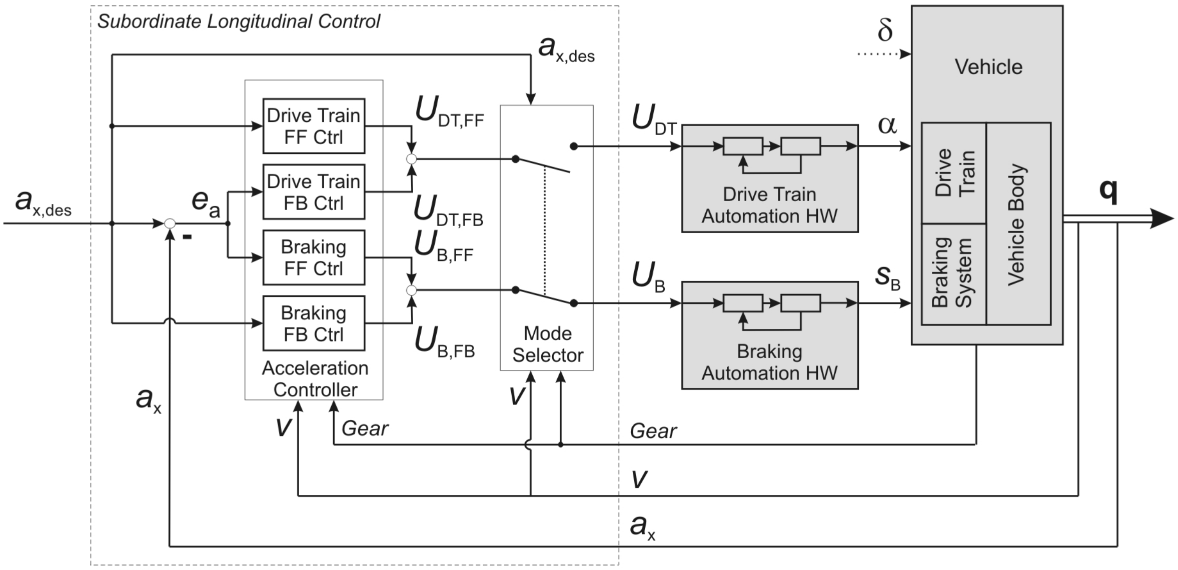
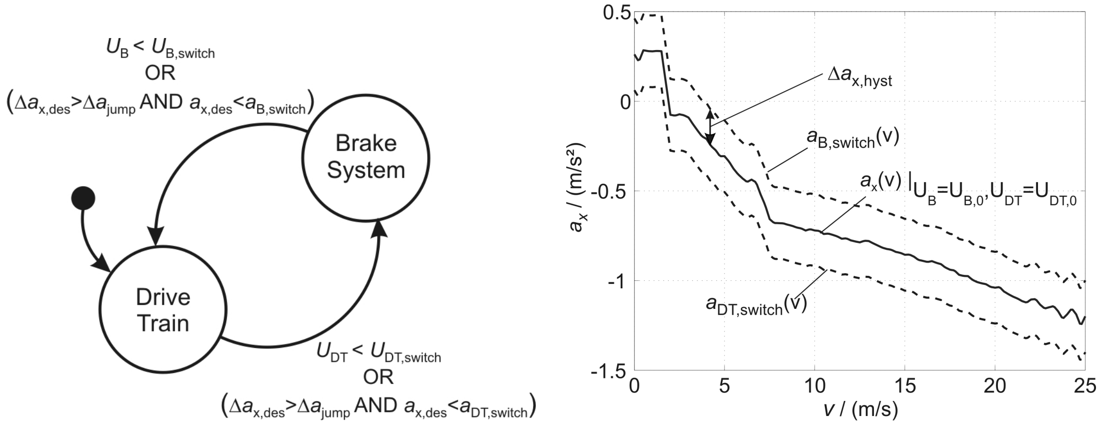


Figure 4.27.: Block diagram of subordinate longitudinal control
The subordinate longitudinal control consists of an acceleration controller and a mode selector. The *mode selector* decides if the desired longitudinal acceleration $a_{x,\text{des}}$ is achieved using the drive train or the braking system. The acceleration controller has separate controllers for drive train and braking system and complements the feedback control with a feedforward part each to enhance reaction times.

For example, a small negative acceleration at higher speeds can also be achieved without braking, just "getting of the gas" and closing the throttle valve. On the other hand, on a downhill slope an acceleration can be achieved by only braking less. Hence, the mode selection between braking system and drive train depends not only on $a_{x,\text{des}}$, but also on the current velocity v and external factors such as the slope of the track and wind.

With regards to the velocity, the inverse extended drive train and braking system maps introduced earlier can be used to check whether they contain a valid input $U_{DT/B}$ for the desired acceleration $a_{x,\text{des}}$ at the given velocity v. However, the external factors such as the slope or wind currently cannot be measured.

Therefore, the mode selection is twofold. For sudden steps in $a_{x,\text{des}}$, the mentioned maps are used as a decision basis, collapsing the maps down to a single decision curve, displayed in Figure 4.28. The curve is identical to coasting with neither braking nor opening the throttle flap. A hysteresis is added not to change hastily from one mode to the other.



(a) Mode selector state diagram (b) Mode selector decision curve

Figure 4.28.: Mode selector

For gradually varying control inputs, this decision curve is no longer used. The curve is valid for certain conditions during the experiments such as an even terrain and hence it is possible that for changed conditions, another mode must be chosen to achieve the desired acceleration. For example, if the vehicle is driving downhill, the low deceleration that would usually require only to close the throttle flap now necessitates the use of the braking system.

Therefore, the mode is selected depending on the control outputs $U_{DT/B}$ as depicted in Figure 4.28. As soon as U_{DT} drops below $U_{DT,\text{switch}} = U_{DT,\min} - \Delta U$, this means that $a_{x,\text{des}}$ cannot be achieved by means of the drive train any longer and the mode selector switches to the braking system. (The integrator in the brake feedback control is reset at this point.) Also, if U_B exceeds $U_{B,\text{switch}} = U_{B,\max} + \Delta U$, this means that $a_{x,\text{des}}$ cannot be achieved by means of the braking system and the mode selector switches to the drive train instead.

The *acceleration controller* determines the current input voltage U_{DT} or U_B for the drive train or braking automation hardware, respectively. This voltage results as the sum of a feedback and a feedforward part

$$U_q = U_{q,FB} + U_{q,FF}, \qquad q \in \{DT, B\}. \tag{4.49}$$

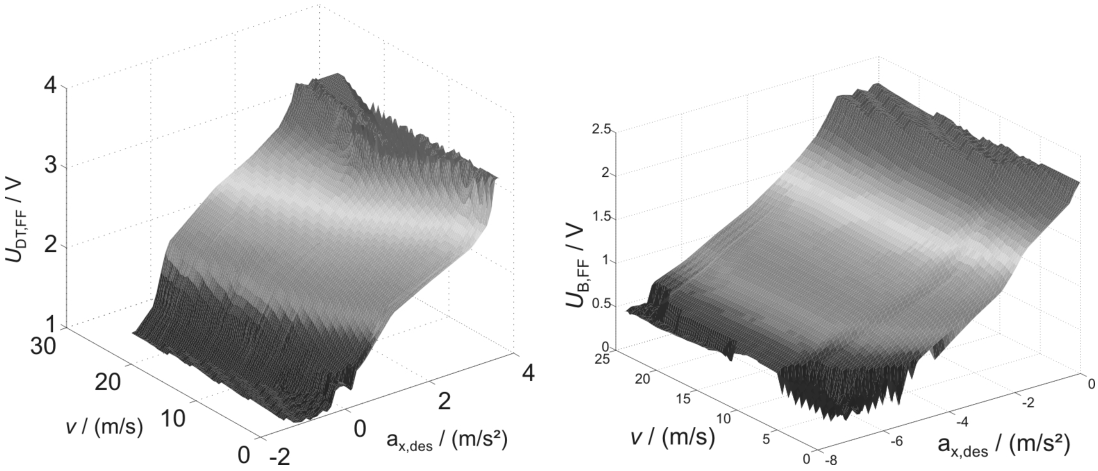
The feedback part $U_{q,FB}$ results from a PID control law applied to the acceleration error

$$U_{q,FB} = k_P e_a + k_D \dot{e}_a + k_I \int e_a dt. \tag{4.50}$$

The feedforward part $U_{q,FF}$ is taken from the inverse of the drive train or braking system map that was built in Section 4.3.3.1

$$U_{q,FF} = M_q^{-1}\left(a_{x,\text{des}}, v, Gear\right). \tag{4.51}$$

The inverse maps are illustrated in Figure 4.29; for their use in the feedforward control they have been slightly smoothed to enhance the control stability.



(a) Inverse drive train map M_{DT}^{-1} (b) Inverse braking system map M_B^{-1}

Figure 4.29.: Inverse maps for feedforward control (shown for second gear)
The illustrated inverse drive train and braking system maps for second gear are built from the maps shown in Figure 4.23 and 4.26. For their use in the feedforward control part ($U_{q,FF}$) they are slightly smoothed to enhance control stability.

CHAPTER 5

Results

This chapter shows and discusses some results of the devised trajectory planning algorithm. Therein, emphasis is placed on the proposed force field trajectory optimization, see Section 5.1. Section 5.2 contains some results for the trajectory initialization, while experimental results for demonstration of applicability of the taken approach can be found in Section 5.3.

5.1. Trajectory Optimization

This section demonstrates the functionality and results of the virtual force field trajectory optimization for several example scenarios. In each scenario, the initial trajectories are manually given such that they are not smooth and may have sharp corners to better show the effect of the optimization and its principle functionality. First, Section 5.1.1 demonstrates and discusses some basic features in an overtaking and a vehicle-following scenario where either a pure path or a velocity planning alone would suffice and no integrated lateral and longitudinal planning would be necessary. Then in Sections 5.1.2 and 5.1.3, more complicated scenarios are demonstrated, where a pure path planning would be insufficient and a combined lateral and longitudinal planning approach is needed. This includes overtaking scenarios with oncoming traffic and the avoidance of cross traffic.

5.1.1. Overtaking and Vehicle Following

First, some basic properties of the devised force field trajectory optimization algorithm shall be demonstrated for the evasion of a static obstacle and the following of a lead vehicle. Among others, these two scenarios are suitable to show the forces and deformations in the x-y-plane for the evasion maneuver and the x-t-plane for the following maneuver.

5.1.1.1. Evasion of Static Obstacle

For the evasion of a static obstacle, a pure path planning would still suffice, however, this example scenario is suitable to demonstrate the forces resulting from the different virtual force fields and the effect they have over the course of several iterations to deform the trajectory from an initial to an optimized one, see Figure 5.1. Further, for this scenario some replanning characteristics are discussed, see Figure 5.2.

In Figure 5.1, the optimization of a given initial trajectory is shown in several iterations in the x-y-plane. For each of the discrete nodes of the trajectory the internal and external forces are displayed as gray and black arrows, respectively. The external forces here include the preview forces. As can be seen, the initial trajectory is deliberately chosen to be discontinuous which causes high internal forces at the two places where the trajectory changes the lane. In front of and behind the obstacle $\mathcal{O}_1$, the effect of the road side force fields can be seen that produces forces that push the trajectory back to the middle of the right lane. Further, the obstacle forces push the trajectory away from the obstacle in lateral direction. Due to the preview forces, the maximum lateral external forces appear in front of the obstacle $\mathcal{O}_1$.

It can be seen that the trajectory in this scenario is already very smooth after only a single iteration and then smaller incremental changes are applied during the remaining iterations, until after seven iterations an optimized trajectory is obtained, where the external and internal forces cancel each other out at all nodes.

From the results in Figure 5.1 it can be seen how the different forces are superimposed and how a smooth equilibrium solution is successfully reached in only few iterations even for a very rough initial trajectory. Further it can be observed that in this scenario the trajectory optimization could still produce meaningful results if it had to be aborted prematurely for example due to time constraints, since the main deformation of the trajectory was achieved in the first iteration.

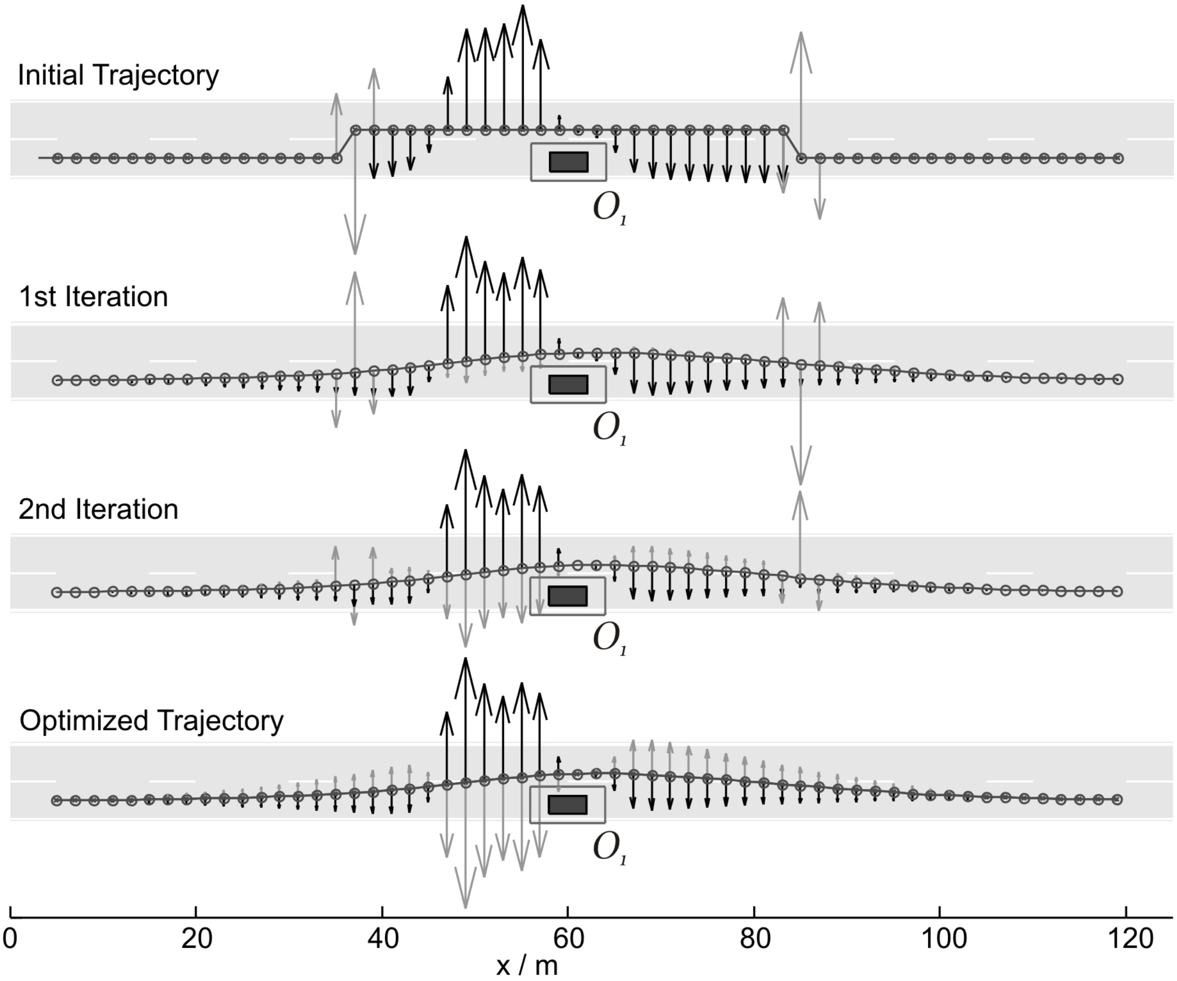


Figure 5.1.: Overtaking static obstacle: Iterations of the Trajectory Optimization
Internal forces are displayed as gray and and external forces as black arrows. The external forces in this figure include the preview forces. The rectangular obstacle (inner black rectangle) is enlarged by the length and width of the ego vehicle (outer rectangle) to better illustrate the distance in which the ego vehicle is planned to pass the obstacle.

While Figure 5.1 only illustrated the optimization of a single trajectory, it is interesting to analyze the behavior of the force field trajectory optimization for repeated replanning. The effect of replanning and different planning distances is demonstrated in Figure 5.2 for the same scenario as before, where the overtaking maneuver of a static obstacle is planned.

As shown in Figure 5.2, a new trajectory is planned about every twelve meters to pass the static obstacle $\mathcal{O}_1$. All these single trajectories (thin lines) have a length of $L = 100$m. Each new trajectory starts from the new position along the previously planned one. Again very simple starting solutions were used, similar to Figure 5.1. The used parts of the different trajectories are concatenated to create the combined trajectory (thick line) that would be the reference trajectory for a trajectory following controller.

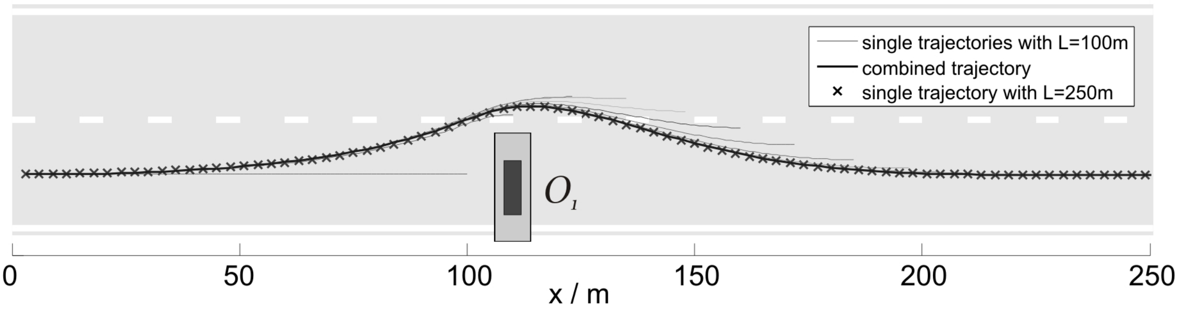


Figure 5.2.: Overtaking static obstacle: Replanning
A new trajectory is planned about every twelve meters to pass the static obstacle $\mathcal{O}_1$. All these single trajectories (thin lines) have a length of $L = 100$m. Each new trajectory starts from the new position along the previously planned one. The used parts of the different trajectories are concatenated to create the combined trajectory (thick line) that would be the reference trajectory for a trajectory following controller. Note, that the last 100 m of this combined trajectory consist of the last single trajectory. This combined trajectory from multiple subsequently planned single trajectories is compared to a single trajectory with a length of $L = 250$m (x-markers).

It can be seen that as soon as the obstacle lies within the planning range, an evasion maneuver is planned. The endings of the single trajectories differ somewhat due to different forces on these trajectories, but the combined trajectory is very smooth. This is also due to the fact, that not only the new starting positions are fixed on the previously planned trajectory but also a starting orientation is regarded within the optimization that is equivalent to the orientation of the previously planned trajectory at the new starting point.

In Figure 5.2 the combined trajectory from multiple subsequently planned single trajectories (where the last 100m consist of the last single trajectory) is compared to a single trajectory with a length of $L = 250$m (x-markers). Although it is not a proved general characteristic, it can be observed that these two trajectories are almost identical for many scenarios. This characteristic of the force field trajectory optimization is very desirable since for frequent replanning the resulting combined reference trajectory does not heavily depend on the length L of each trajectory.

Nevertheless, a sufficient planning distance and replanning frequency is necessary in order to develop a meaningful plan, as can be seen by the first single trajectory with $L = 100$m which is not yet long enough to pass the obstacle. There, first no evasion maneuver is planned and the planned maneuver changes significantly from the first to the second single trajectory.

5.1.1.2. Following of Lead Vehicle

In this section, the basic properties in the added dimension, the time-dimension, are demonstrated using a simple vehicle following scenario. Analogously to the evasion scenario in Section 5.1.1.1, the forces and deformations of the trajectory from the initial to the optimized trajectory are shown, see Figure 5.3. Further, the resulting profiles of distance to lead vehicle, velocity and acceleration are shown in Figure 5.4, and the effect of different discretizations is discussed.

Figure 5.3 shows the initial and optimized trajectories in the x-t-plane and the resulting external (black) and internal (gray) virtual forces in time-direction. Again, a rather poor initial solution is selected to better show the effect of the optimization. The initial solution has a very sharp bend where the ego vehicle suddenly changes velocity to avoid a collision with the slower lead vehicle (obstacle $\mathcal{O}_1$). Here, high internal forces result (which are scaled down here to fit in the Figure).

For the trajectory optimization algorithm, the lead vehicle exists from the point of time when it is first detected, in this example scenario $t = 0$s, about 60m ahead of the ego vehicle. All nodes above the representation of the lead vehicle are affected by an external force that pushes them away from the obstacle and shifts the nodes to later points of time to slow the ego vehicle down.

After eight iterations, the optimized trajectory is smooth, the distance to the lead vehicle has increased and the internal and external forces cancel each other out at all nodes. Note that the internal forces here consist not only of acceleration or jerk forces to counteract longitudinal accelerations but also the forces $\mathbf{F}_t^{\text{vel}}$ to return to the desired traveling velocity. Without the latter force component, following at a constant distance would be impossible, since for a constant velocity and a constant distance, no internal forces would exist to cancel out the external forces that push the trajectory away from the obstacle, see Section 2.6.

Whereas Figure 5.3 shows the optimized trajectory in the x-t-plane, it is hard to judge distances, velocities or even acceleration. Therefore, these values are displayed in Figure 5.4 to further analyze the properties of the optimized solution. Distance to the lead vehicle, velocity of the ego vehicle and acceleration of the ego vehicle along the planned trajectory are shown for the initial trajectory (broken lines) and the optimized trajectories for different discretizations of $\Delta L = 1$ m (solid line) and $\Delta L = 5$ m (x-markers). This allows further to shed some light on the effect of the discretization for this scenario.

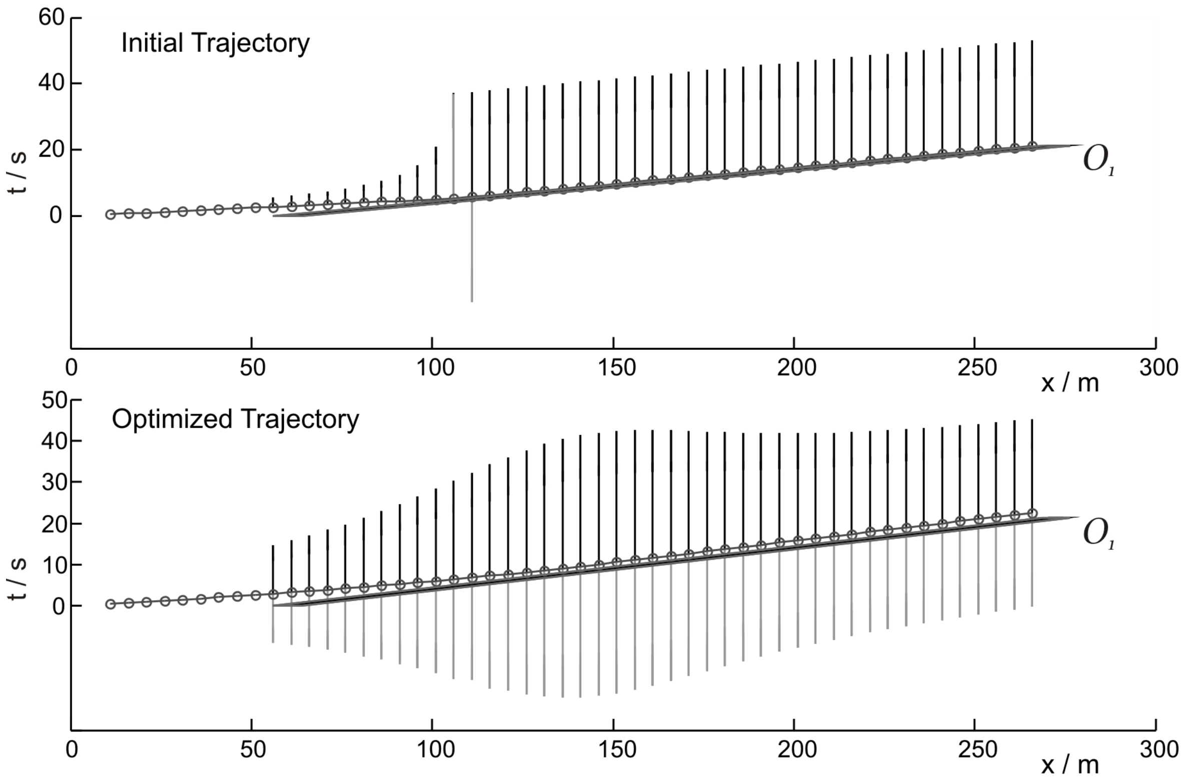


Figure 5.3.: Follow lead vehicle: Iterations
The figure shows the initial and optimized trajectories in the x-t-plane and shows the resulting external (black) and internal (gray) virtual forces in time-direction. The obstacle (lead vehicle) exists from the point of time when it is first detected, in this example scenario $t = 0$ s, about 60 m ahead of the ego vehicle. The obstacle representation is enlarged by the ego vehicle's length to better illustrate the distances between ego vehicle and obstacle.

In Figure 5.4, it can be seen that distance, velocity, and acceleration profiles are smoothed during the optimization from a deliberately non-smooth initial trajectory. A stationary distance of about 20 m to the lead vehicle is approached, however, this distance first falls below this value, which allows less deceleration. The velocity smoothly changes from 20 m/s to 10 m/s to match the speed of the lead vehicle, while the maximum deceleration is about -1.7 m/s^2.

This example scenario is used to show the effect of different discretization. As can be seen, the optimized trajectories for the two different discretizations are almost identical. Only at the beginning, where the nodes are not yet directly affected by obstacle forces, small discrepancies can be detected. This result is true for all vehicle following scenarios. For other maneuvers, there can be slightly larger differences, especially if only small portions of the trajectory are in the vicinity of an obstacle. In these cases a change in discretization has a large impact on the number of nodes with high obstacle forces. Therefore, slightly different trajectories can result.

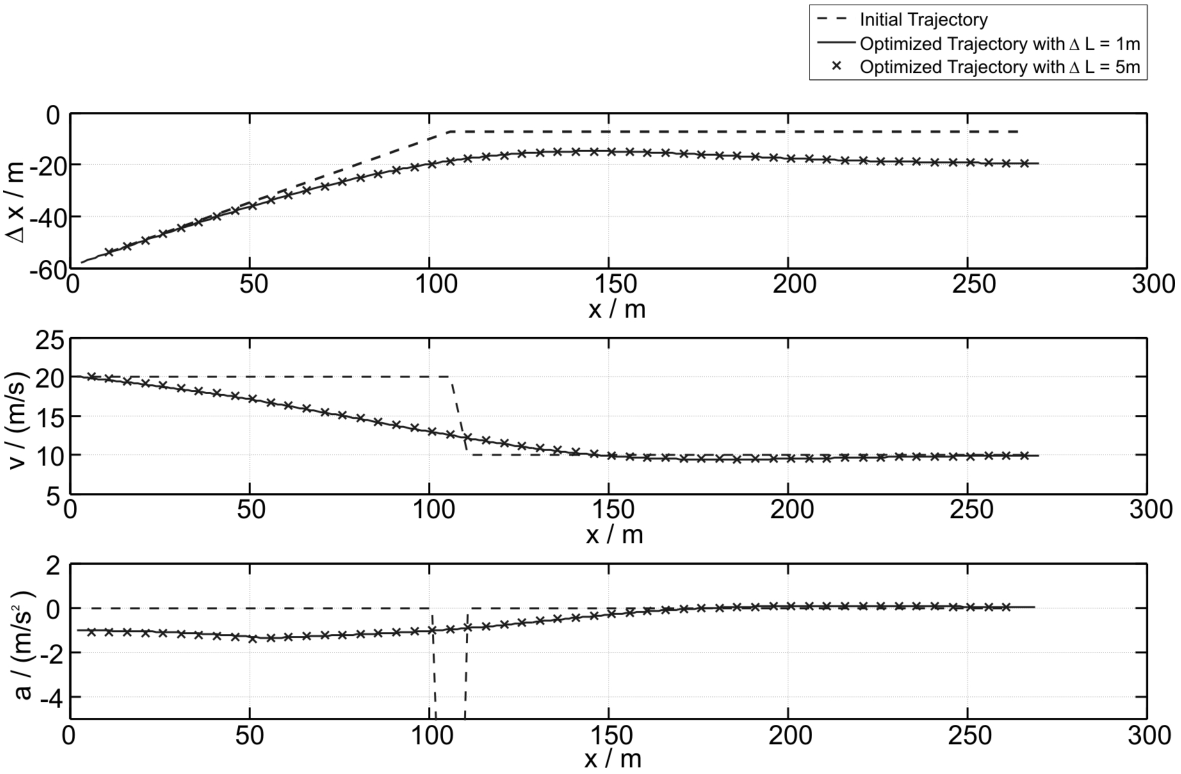


Figure 5.4.: Follow lead vehicle: distance, velocity and acceleration
Distance to the lead vehicle, velocity of the ego vehicle and acceleration of the ego vehicle along the planned trajectory are shown for the initial trajectory (broken lines) and the optimized trajectories for different discretizations of $\Delta L = 1$ m (solid line) and $\Delta L = 5$ m (x-markers).

Note, that the planned trajectory only follows the lead vehicle because the initial trajectory does. In case the initial trajectory would overtake the lead vehicle, a different optimized trajectory would result.

5.1.2. Overtaking with Oncoming Traffic

Building on the results shown in the previous section, now more complicated scenarios are discussed that demonstrate the usefulness of an integrated trajectory planning versus a pure path planning or a sequential path and velocity planning. As example scenario, an overtaking maneuver of a slower lead vehicle with oncoming traffic is chosen. In this context also the effect of homeotopically different initial trajectories on the optimized trajectory is shown: Whether the lead vehicle is passed before or after the oncoming vehicle, solely depends on the initial solution. These two alternatives are elaborated in Sections 5.1.2.1 and 5.1.2.2. Further, if the initial solution would follow the lead vehicle, so would the optimized trajectory as already shown above in Section 5.1.1.2.

5.1.2.1. Overtaking Before Oncoming Traffic

In this section an overtaking scenario with oncoming traffic is analyzed. It consists of a lead vehicle and an oncoming vehicle that drive at constant velocity. The ego vehicle overtakes the lead vehicle before the oncoming vehicle approaches. The scenario is shown in the augmented workspace $\langle x, y, t\rangle$ in Figure 5.5 and then analyzed in snapshots for different points of time in Figure 5.6. Figure 5.7 shows the velocity and acceleration profiles for the resulting optimized trajectory.

Figure 5.5 shows the representation of the optimized trajectory $\mathcal{T}$ and the two obstacles $\mathcal{O}_1$ (lead vehicle) and $\mathcal{O}_2$ (oncoming vehicle) in the augmented workspace $\langle x, y, t\rangle$. Further, the projection of the trajectory to the x-y-plane is shown.

As can be seen, both lead vehicle and oncoming vehicle move at constant velocity. The optimized trajectory is not in collision with either of the obstacles and is “below” the oncoming vehicle $\mathcal{O}_2$, which means that it overtakes the lead vehicle $\mathcal{O}_1$ before $\mathcal{O}_2$ arrives.

Figure 5.6 shows how the scenario progresses in a sequence of snapshots to illustrate that the planned trajectory is collision free. The scenario is illustrated in the x-y-plane at four consecutive points of time t_1 to t_4. The planned trajectory for this scenario is represented by a solid line. As comparison, the planned trajectory for the same scenario (and the same initial trajectory) but without the oncoming vehicle $\mathcal{O}_2$ is shown as broken line.

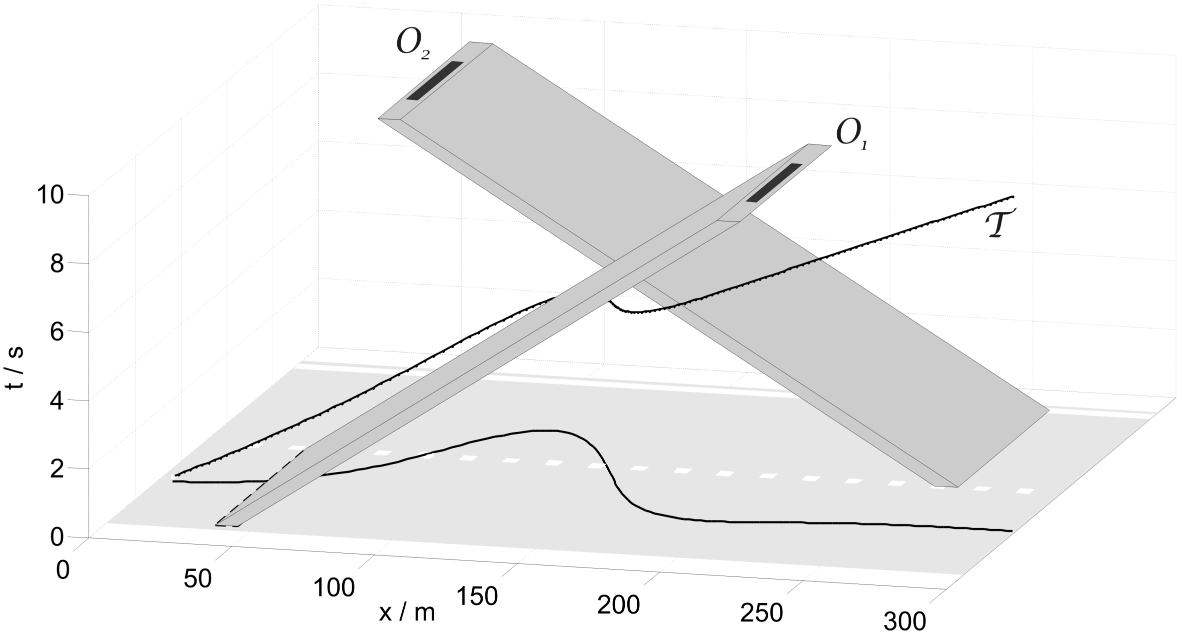


Figure 5.5.: Overtake lead vehicle before oncoming traffic: Trajectory
The figure shows the optimized trajectory $\mathcal{T}$ and the two obstacle $\mathcal{O}_1$ (lead vehicle) and $\mathcal{O}_2$ (oncoming vehicle) in the augmented workspace $\langle x, y, t\rangle$ and the projection of $\mathcal{T}$ to the x-y-plane. The rectangular obstacles (inner black rectangles) are enlarged by the length and width of the ego vehicle (outer gray rectangles) to better illustrate the distance in which the ego vehicle is planned to pass the obstacles.

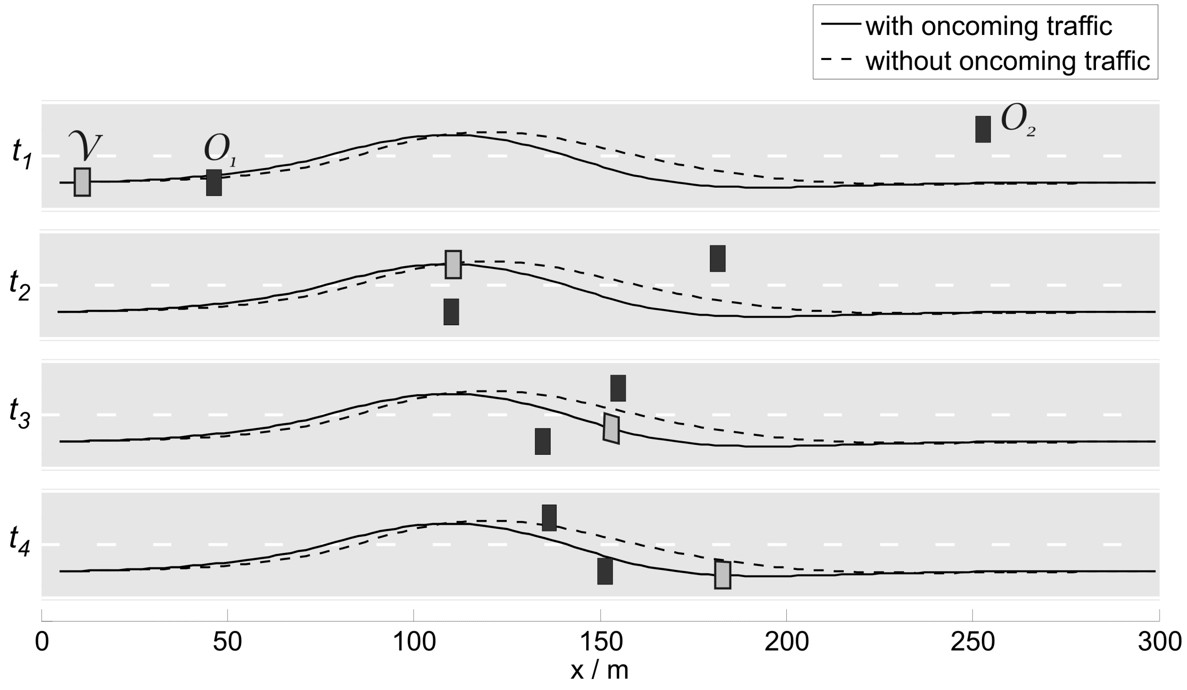


Figure 5.6.: Overtake lead vehicle before oncoming traffic: Snapshots

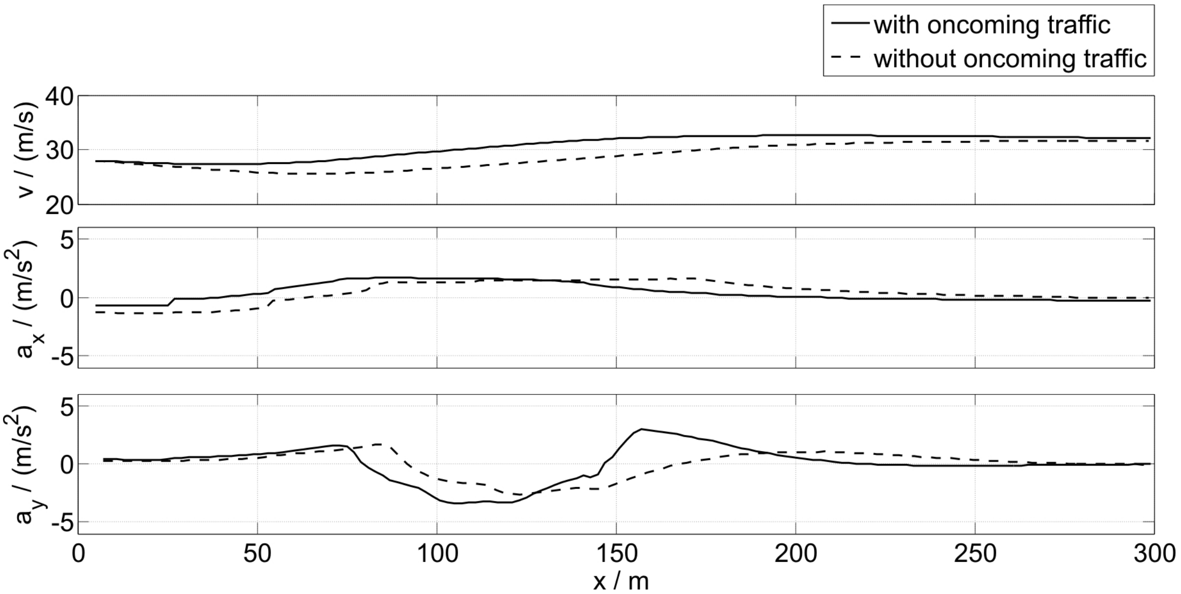


Figure 5.7.: Overtake lead vehicle before oncoming traffic: Velocity and acceleration
The solid lines represent the planned profiles for the overtaking scenario with oncoming traffic, as shown in Figure 5.5, while the broken lines show the planned profiles in case the oncoming vehicle was not there as comparison.

At t_1, the ego vehicle is still in its lane behind the lead vehicle, but the overtaking maneuver is already planned, taking the movements of both obstacles into account. Then the overtaking maneuver starts and the snapshot at t_2 shows the point of time when the ego vehicle is right beside $\mathcal{O}_1$. It can be seen that the trajectory's maximum lateral deviation from the right lane is reached where the lead vehicle is passed, as expected. Next the ego vehicle returns to the right lane before the oncoming vehicle approaches, as illustrated in snapshots t_3, where it can be observed that a head-on collision with the oncoming vehicle is successfully avoided. In snapshot t_4, the ego vehicle is back to the right lane and the overtaking maneuver is completed.

The difference in the planned trajectory that is caused by the oncoming vehicle, i.e. the difference between the solid and broken line, is evident. The overtaking maneuver is initiated a little earlier and is primarily shorter, such that the ego vehicle returns to the right earlier to avoid a collision with the oncoming traffic, which would still occur for the broken line as can be seen for t_3.

While Figure 5.6 illustrates the planned path as the trajectory's projection to the x-y-plane for different points of time, it does not contain any information about the velocity or acceleration along the trajectory. These are detailed in Figure 5.7.

Figure 5.7 shows the planned velocity v, longitudinal acceleration a_x, and lateral acceleration a_y over the distance along the street x. The solid lines represent the planned profiles for the overtaking scenario with oncoming traffic, as shown in Figure 5.5, while the broken lines show the planned profiles in case the oncoming vehicle was not there as comparison.

As can be seen, *without* the oncoming vehicle (broken lines), the initial velocity of 28 m/s is first reduced a little when the ego vehicle approaches the lead vehicle. When the overtaking maneuver starts and the ego vehicle changes to the left lane at about 70 m, the ego vehicle accelerates to overtake. As can be seen by the broken line, for the overtaking scenario without oncoming traffic, a maximum velocity of about 32 m/s is reached at about 250 m when the ego vehicle is already back to the right lane.

For the scenario *with* the oncoming vehicle (solid lines), the overtaking maneuver starts a little bit earlier and the ego vehicle's velocity is hardly reduced at all before the lane change is initiated. Then the ego vehicle accelerates earlier and a little bit stronger to reach a maximum velocity of about 33 m/s.

The longitudinal and lateral acceleration profiles are very similar with or without the oncoming vehicle, they are mainly shifted because the overtaking maneuver starts a little earlier with oncoming traffic and is also finished earlier. The maximum longitudinal acceleration of about 1.6 m/s^2 is only minimally higher with than without oncoming vehicle. The main difference between the two scenarios can be seen in the lateral acceleration when the ego vehicle returns to the right lane after the overtaking maneuver. Since this must happen earlier and more abruptly with oncoming traffic, there is a higher peak of about 2.9 m/s^2, compared to only about 1 m/s^2 without the oncoming traffic.

It can be seen that the force field optimization successfully generates a trajectory with a smooth path and velocity profile and continuous acceleration profiles for an overtaking maneuver with oncoming traffic. As given by the initial solution, the lead vehicle is passed before the oncoming vehicle approaches. The integrated planning in the $\langle x, y, t\rangle$ workspace results in a meaningful maneuver which does not require much higher velocities or accelerations compared to the same scenario without oncoming traffic. This is one of the advantages of integrated planning of path and velocity. In case the velocity was planned in a second step, taking the path represented by the broken line in Figure 5.6, much higher accelerations would have been required to prevent a head-on collision with the oncoming traffic.

5.1.2.2. Overtaking After Oncoming Traffic

In this section again an overtaking scenario with oncoming traffic is analyzed, as in Section 5.1.2.1, however in this case, the oncoming vehicle is closer and an initial trajectory is selected such that the ego vehicle overtakes the lead vehicle after the oncoming vehicle has passed. The scenario is shown in the augmented augmented workspace $\langle x, y, t \rangle$ in Figure 5.8 and then analyzed in snapshots for different points of time in Figure 5.9. Figure 5.10 shows the velocity and acceleration profiles for the resulting optimized trajectory.

Figure 5.8 shows the representation of the optimized trajectory $\mathcal{T}$ and the two obstacless $\mathcal{O}_1$ (lead vehicle) and $\mathcal{O}_2$ (oncoming vehicle) in the augmented workspace $\langle x, y, t \rangle$. Further, the projection of the trajectory to the x-y-plane is shown. As can be seen, the optimized trajectory is not in collision with either of the obstacles and is “above” the oncoming vehicle $\mathcal{O}_2$, which means that is overtakes the lead vehicle $\mathcal{O}_1$ after $\mathcal{O}_2$ arrives.

Figure 5.9 shows in a sequence of snapshots how the scenario progresses to illustrate that the planned trajectory is collision free. The scenario is illustrated in the x-y-plane at five consecutive points of time t_1 to t_5. The planned trajectory for this scenario is represented by a solid line. As comparison, the planned trajectory for the same scenario but without the oncoming vehicle $\mathcal{O}_2$ is shown as broken line.

At t_1, the ego vehicle is still in its lane behind the lead vehicle, but the overtaking maneuver is already planned, taking the movements of both obstacles into account. From t_1 to t_2, the ego vehicle decelerates and follows the lead vehicle until the oncoming vehicle has passed to prevent a head-on collision. Then the overtaking maneuver starts and the snapshot at t_3 shows the point of time when the ego vehicle is beside $\mathcal{O}_1$. Next, the ego vehicle returns to the right lane as illustrated in snapshot t_4. Snapshot t_5 shows the ego vehicle back to the right lane and the overtaking maneuver is completed.

The difference in the planned trajectory that is caused by the oncoming vehicle, i.e. the difference between the solid and broken line, is evident. The overtaking maneuver is initiated much later and a little more abruptly. The higher curvature of this lane change to the left results partly from the lower velocity which allows higher curvatures and partly by the necessity to change the lane more quickly since the ego vehicle is already rather close to the lead vehicle. The lane change back to the right lane happens later than without the oncoming traffic but is very similar.

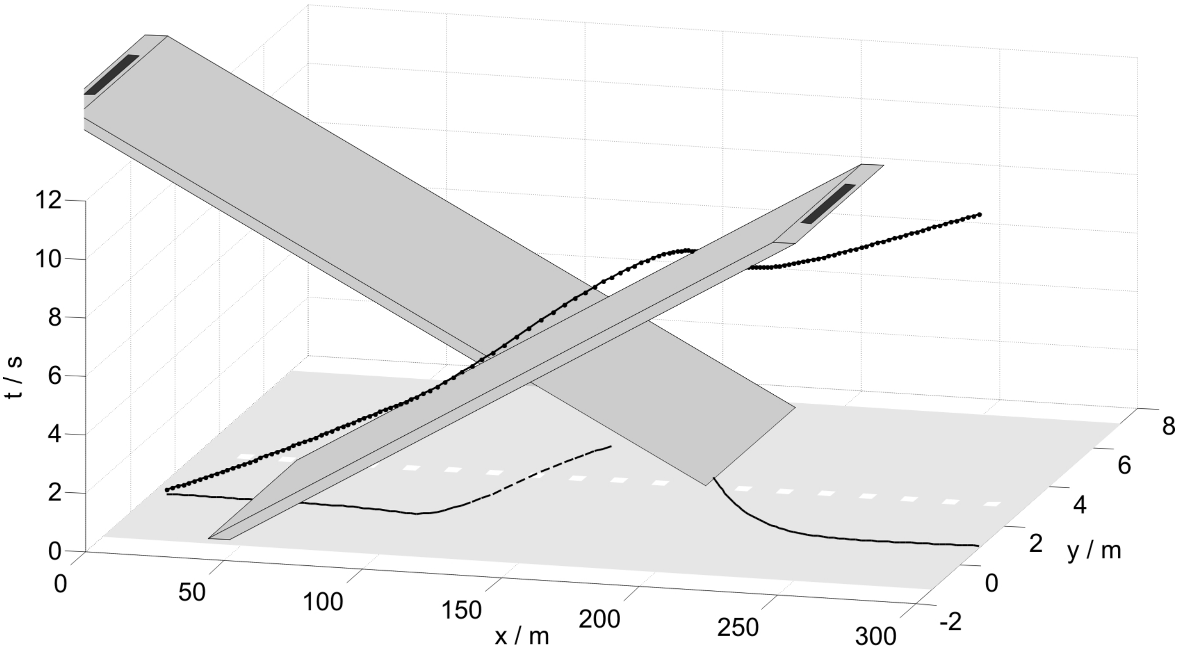


Figure 5.8.: Overtake lead vehicle after oncoming traffic: Trajectory
The rectangular obstacles are enlarged by the length and width of the ego vehicle to better illustrate the distance in which the ego vehicle is planned to pass the obstacles.

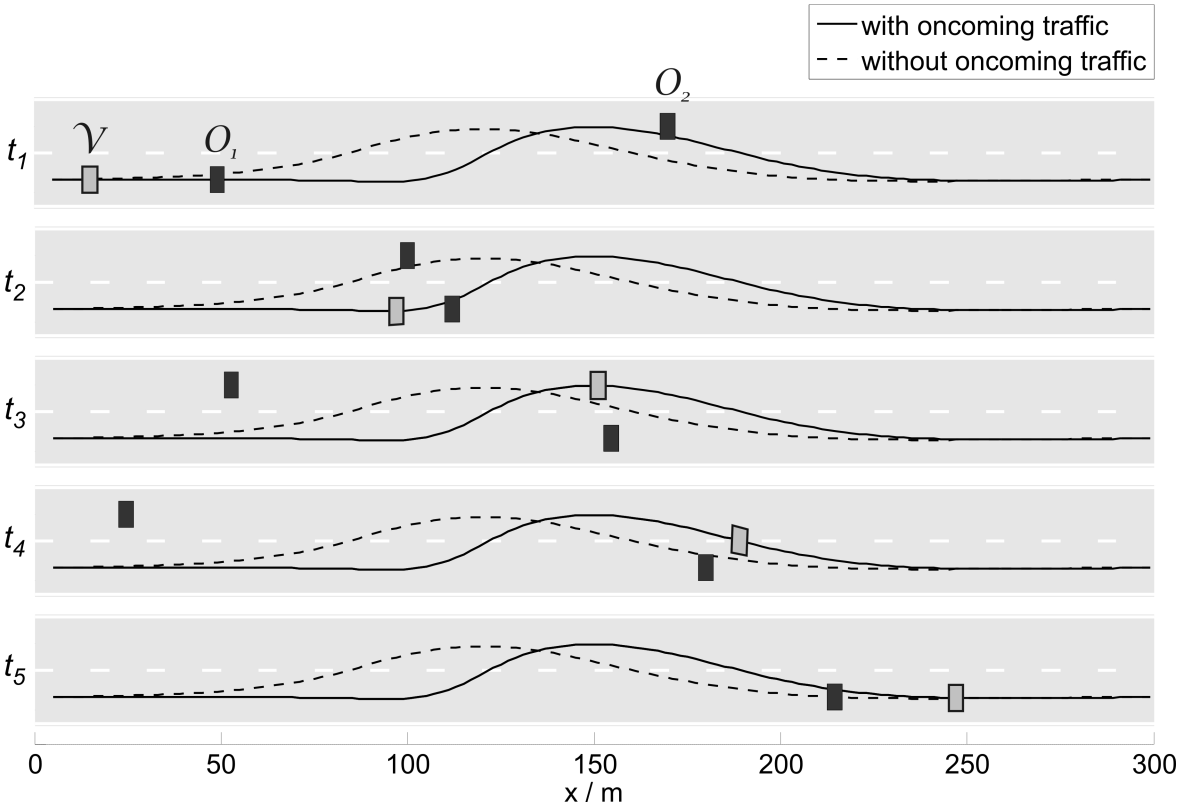


Figure 5.9.: Overtake lead vehicle after oncoming traffic: Snapshots

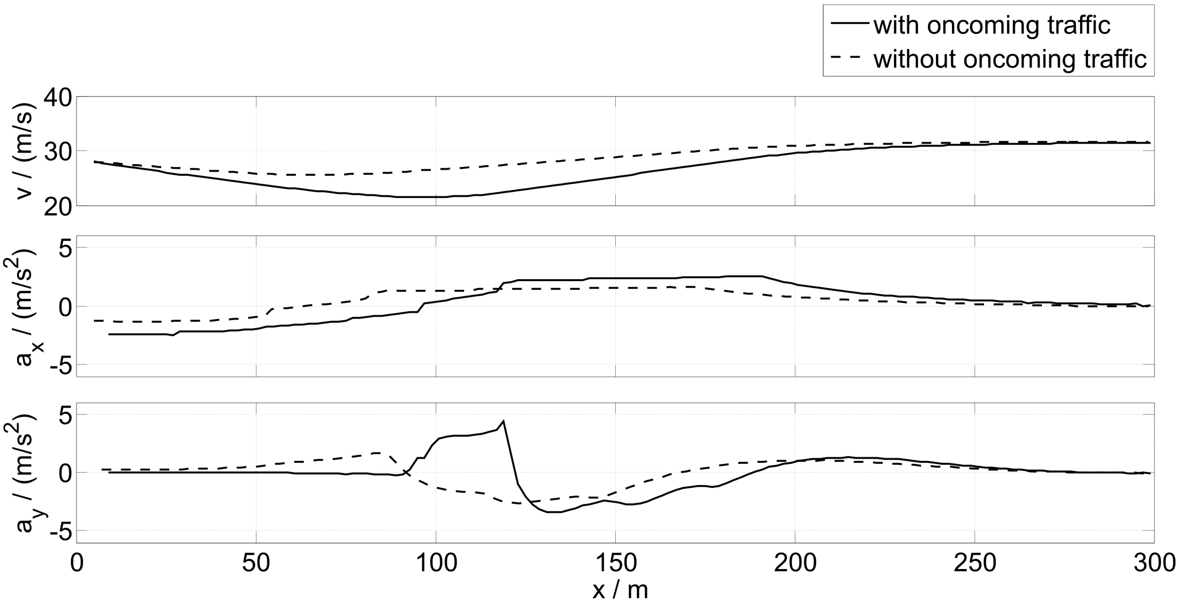


Figure 5.10.: Overtake lead vehicle after oncoming traffic: Velocity and acceleration
The solid lines represent the planned profiles for the overtaking scenario with oncoming traffic, as shown in Figure 5.8, while the broken lines show the planned profiles in case the oncoming vehicle was not there as comparison.

While Figure 5.9 illustrates the planned path as the trajectory's projection to the x-y-plane for different points of time, it does not contain any information about the velocity or acceleration along the trajectory. These are detailed in Figure 5.10.

Figure 5.10 shows the planned velocity v, longitudinal acceleration a_x, and lateral acceleration a_y over the distance along the street x. The solid lines represent the planned profiles for the overtaking scenario with oncoming traffic, as shown in Figure 5.8, while the broken lines show the planned profiles in case the oncoming vehicle was not there as comparison.

As can be seen, *without* the oncoming vehicle (broken lines), the initial velocity of 28 m/s is first reduced a little when the ego vehicle approaches the lead vehicle. When the overtaking maneuver starts and the ego vehicle changes to the left lane at about 70 m, the ego vehicle accelerates to overtake. As can be seen by the broken line, for the overtaking scenario without oncoming traffic, a maximum velocity of about 32 m/s is reached at about 250 m when the ego vehicle is already back to the right lane.

For the scenario *with* the oncoming vehicle (solid lines), the ego vehicle first decelerates stronger with a maximum deceleration of about 2.4 m/s^2 compared to about 1.4 m/s^2 to reduce the velocity to about 22 m/s. Then at about 110 m the overtaking maneuver starts and as the ego vehicle changes the lane to the left it also begins to accelerate. The maximum longitudinal acceleration is about 2.6 m/s^2 to accelerate to about 32 m/s at the end of the overtaking maneuver.

The lateral acceleration shows a high peak of about 4.3 m/s^2 when the ego vehicle changes to the left lane. As discussed before, this higher lateral acceleration results from the fact, that the ego vehicle changes the lane with a rather high curvature to avoid a rear-end collision with the lead vehicle without having to brake much more and first increase the distance to the lead vehicle. However, the lateral acceleration profile still shows room for improvement as it is not very smooth during the overtaking maneuver.

It can be seen that the force field optimization successfully generates a trajectory with a smooth path and velocity profile and continuous acceleration profiles for an overtaking maneuver with oncoming traffic. As given by the initial solution, the lead vehicle is passed after the oncoming vehicle approaches. In order to do that, the ego vehicle is first slowed down to follow the lead vehicle and then initiate the lane change at a later point when the oncoming vehicle has passed. Again the integrated planning in the $\langle x, y, t\rangle$ workspace demonstrates the advantage of integrated planning of path and velocity. In case the velocity was planned in a second step, taking the path represented by the broken line in Figure 5.9, much higher decelerations and accelerations would have been required to prevent a head-on collision with the oncoming traffic and still overtake after the oncoming traffic has passed. A pure path planning would not have been able to generate a valid solution at all in this case.

As shown by the combination of Figures 5.5-5.10, the qualitative solution of the force field optimization depends on the initial solution. As demonstrated, two homeotopically different initial solutions produced two different optimized trajectories, overtaking a lead vehicle either before or after an oncoming vehicle. As mentioned before, this is a very desirable characteristic since the optimization regards the choice given by the trajectory initialization. This way, either several different initial trajectories for different maneuvers could be optimized and compared or a certain maneuver can be achieved, e.g. if the driver has a certain preference.

5.1.3. Cross Traffic

After the demonstration of the trajectory optimization in challenging scenarios with longitudinal road traffic, now a scenario with lateral traffic is discussed, where again a pure path planning would not suffice to generate a collision free motion plan. Figure 5.11 shows the representation of the scenario in the augmented workspace $\langle x, y, t \rangle$, including the obstacle $\mathcal{O}_1$ that crosses the street and the resulting optimized trajectory $\mathcal{T}$. Further, the projection of the trajectory to the x-y-plane is shown.

As can be seen in Figure 5.11, the optimized trajectory is not in collision with the crossing obstacle but passes "above" its representation in the augmented workspace $\langle x, y, t \rangle$, which means that it passes after the obstacle has crossed the street.

Figure 5.12 shows in a sequence of snapshots how the scenario progresses to illustrate that the planned trajectory is collision free. The scenario is illustrated in the x-y-plane at three consecutive points of time t_1 to t_3. The ego vehicle $\mathcal{V}$ and the cross traffic $\mathcal{O}_1$ are drawn as rectangles. The planned trajectory for this scenario is represented by a solid line. As comparison, for the second snapshot t_2, the position of the ego vehicle is indicated as empty rectangle if it followed the lane at constant velocity.

At t_1, the obstacle $\mathcal{O}_1$ is detected and the according trajectory is planned. At this point the vehicle is moving at a velocity of 28 m/s in the right lane. The planned path itself is hardly changed and almost remains in the middle of the right lane. Only a slight lateral deviation to the right can be seen. But the velocity is adapted: Between t_1 and t_3 the ego vehicle decelerates to let the crossing obstacle pass and avoid a collision. Snapshot t_3 shows that the collision is successfully avoided. Snapshot t_2 shows that a collision would result if the ego vehicle does not decelerate, as indicated by the empty rectangle.

To complement the above illustrations, Figure 5.13 shows the planned velocity v, longitudinal acceleration a_x, and lateral acceleration a_y over the distance along the street x. As soon as the crossing obstacle is detected, i.e. from the beginning of this planned trajectory, the ego vehicle brakes with a constant deceleration of about -5 m/s^2 to avoid the collision. It thereby decreases its velocity from 28 m/s to about 20 m/s. As soon as the obstacle has passed, the ego vehicle slowly accelerates back to its desired traveling velocity. Hardly any lateral acceleration is planned, as only a slight swerve to the right is planned to help in avoiding the collision.

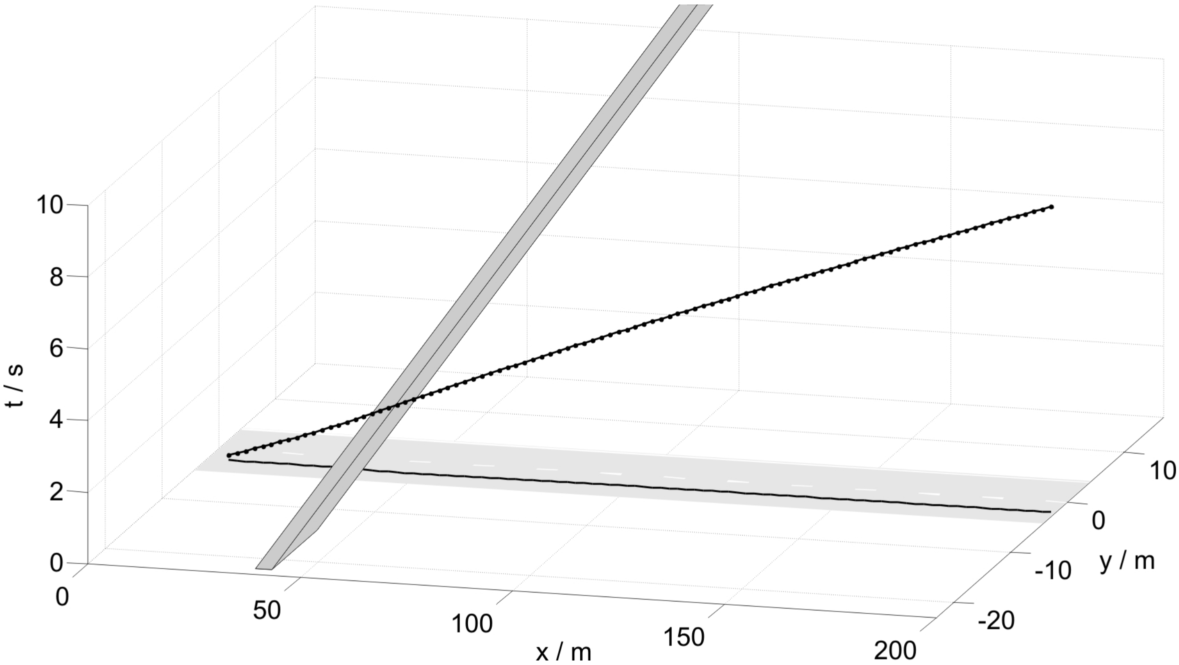


Figure 5.11.: Avoid collision with cross traffic: Trajectory
The figure shows the optimized trajectory $\mathcal{T}$ and the obstacle $\mathcal{O}_1$ that crosses the street in the augmented workspace $\langle x, y, t \rangle$ and the projection of the trajectory to the x-y-plane is shown. The rectangular obstacle is enlarged by the length and width of the ego vehicle to better illustrate the distance in which the ego vehicle is planned to pass the obstacle.

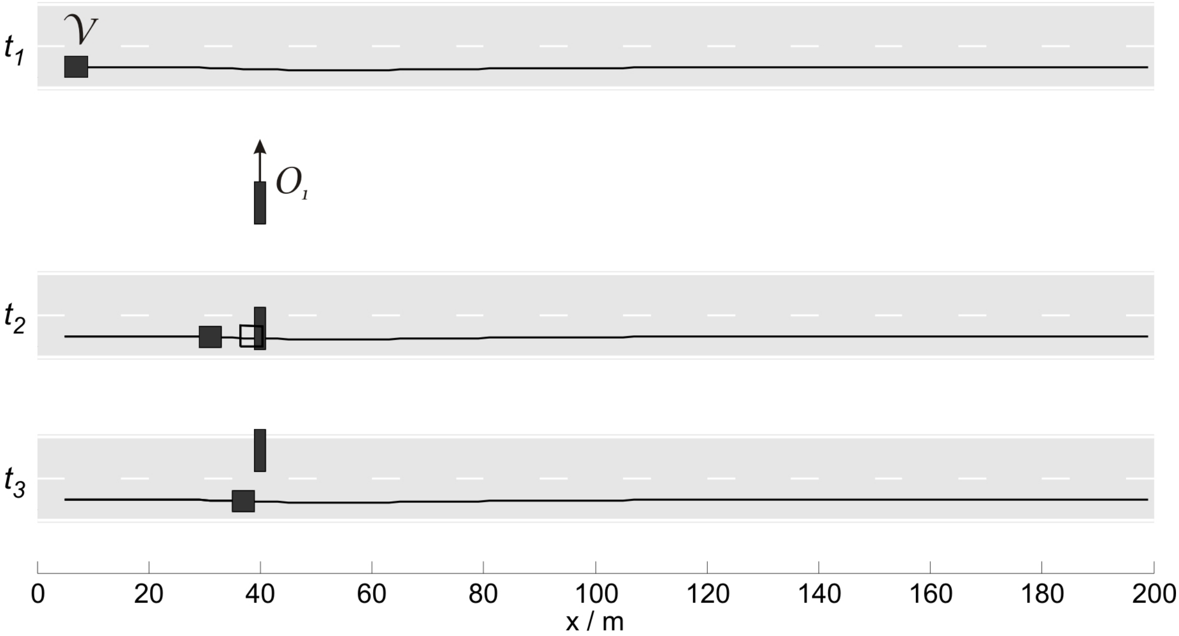


Figure 5.12.: Avoid collision with cross traffic: Snapshots
The planned trajectory for this scenario is represented by a solid line. As comparison, for the second snapshot t_2, the position of the ego vehicle is indicated as empty rectangle if it followed the lane at constant velocity.

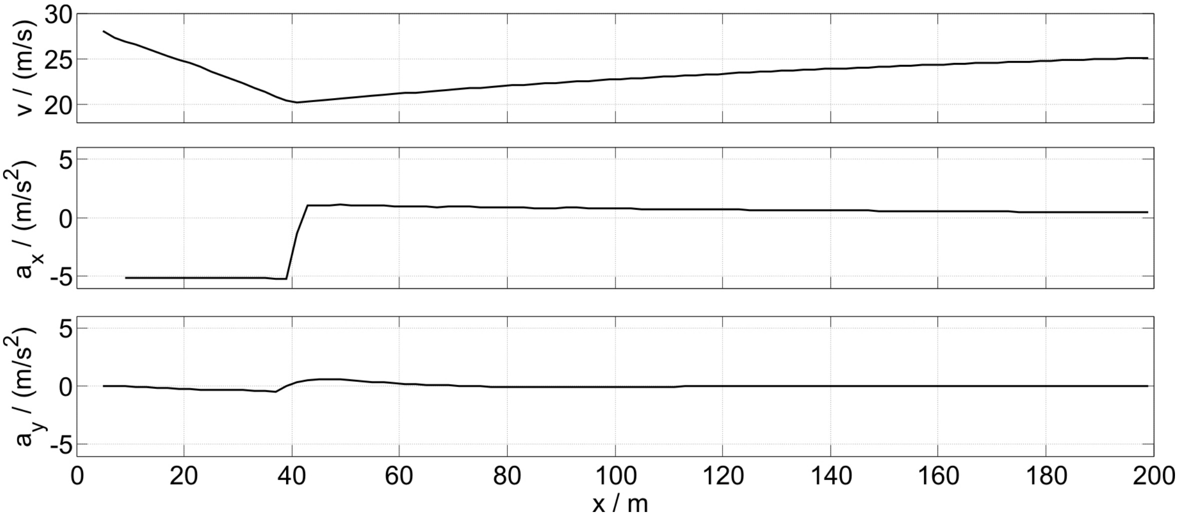


Figure 5.13.: Avoid collision with cross traffic: Velocity and acceleration profiles

As demonstrated, the force field trajectory optimization successfully adapted the trajectory to avoid a collision with a crossing obstacle. To do so the velocity profile was altered significantly and also a slight adaptation of the path aided in the collision avoidance maneuver. In order to prevent the collision, a relatively high deceleration was planned, however, as can be seen by the constant profile this deceleration is necessary to achieve the collision avoidance with the demonstrated safety margin.

It can be seen that a path planning (at constant velocity) alone would not have sufficed to avoid a collision, since even for extreme maneuvers to the right or left in this case a collision would have resulted.

The jerk, i.e. the change of acceleration could be improved as there are large changes at the very beginning and at about 40 m. The large jump at the very beginning is caused by the fact that no initial acceleration is regarded since only the first two nodes are fixed and therefore only initial position, orientation, and velocity are taken into account.

5.2. Trajectory Initialization

As mentioned before, the force field trajectory optimization algorithm requires an initial guess. Therefore, the motion planning approach that has been devised in this work is comprised of two steps, as detailed before: In the first step, the trajectory initialization, a collision-free and dynamically favorable trajectory $\mathcal{T}_{\text{init}}$ is generated by a grid-based tree-search algorithm. In the second step, the trajectory optimization, local trajectory deformations are applied with the help of the virtual force field algorithm, resulting in an improved solution trajectory $\mathcal{T}$. As previously discussed, this initial trajectory determines the nature of the optimized solution, e.g. whether to pass a lead vehicle or whether to follow it.

Even though the trajectory initialization is not the main focus of this work, some exemplary results shall be presented at this point. Figure 5.14 depicts the basic trajectory set for all valid edges of one velocity profile for a given starting point and orientation from which edges are selected in each step of the tree search. Some of the depicted edges must still be removed after collision checks with the sides of the road.

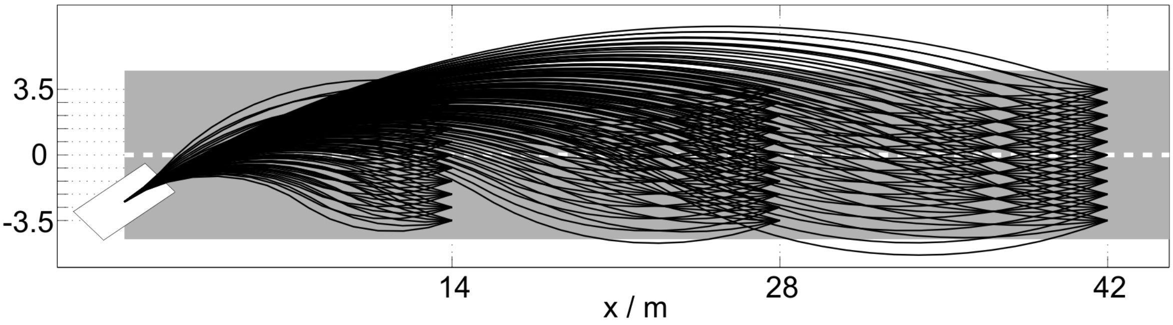


Figure 5.14.: Trajectory Initialization: Basic trajectory set for tree search

Figure 5.15 shows the result of the trajectory initialization for a curved road and two obstacles as projection to the x-y-plane. The edge color signifies the total accumulated cost along the planned trajectory, where darker colors symbolize a higher cost. It can be seen that the road is searched thoroughly and a meaningful and smooth initial trajectory is produced.

Figure 5.15 also illustrates the anytime-characteristics of the devised AWA*-based starting solution. The intermediate solution shown in the figure represents the first found solution. During the remainder of the allotted time it was enhanced until finally the indicated solution trajectory resulted. As can be seen, a premature termination of the trajectory initialization would have still produced a viable input to the trajectory optimization step, even though a larger deformation and therefore maybe more iterations would have been necessary.

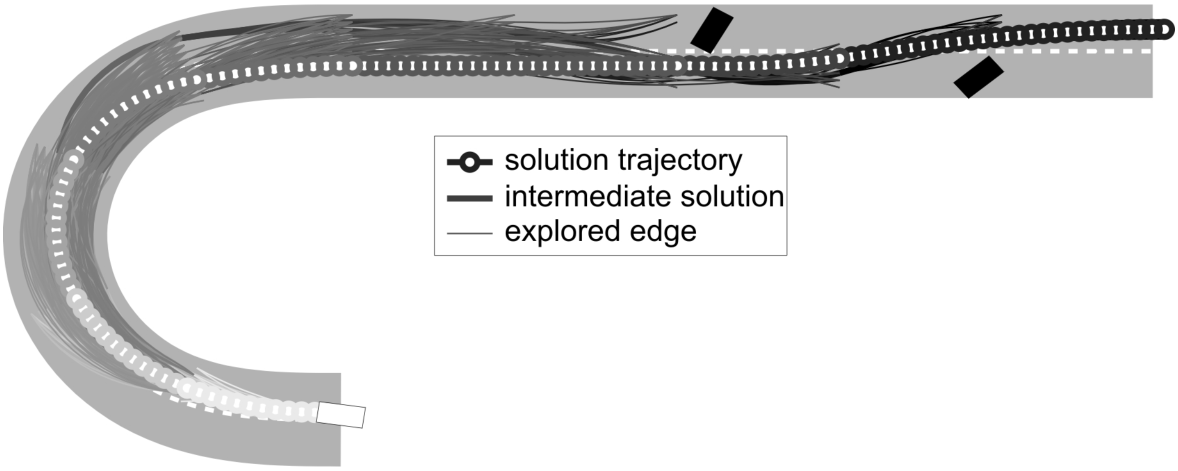


Figure 5.15.: Trajectory Initialization: Explored edges and solution of grid-base tree-search
The edge color corresponds to accumulated path cost: the darker the higher the accumulated cost

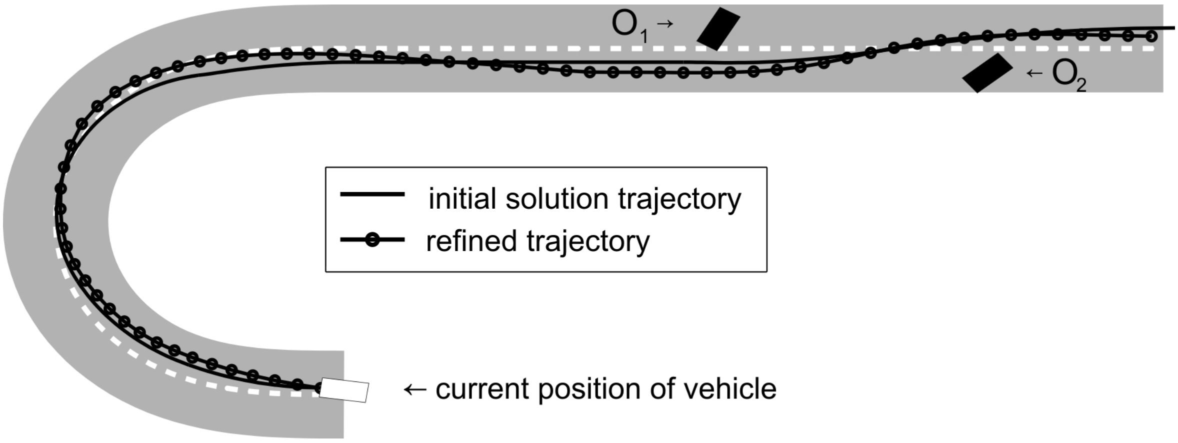


Figure 5.16.: Initial and optimized trajectory for curved road with obstacles

Figure 5.16 then shows how this initial trajectory is deformed by the trajectory optimization algorithm. It can be seen that the optimized trajectory passes the obstacle $\mathcal{O}_1$ with a greater safety margin. In the curve, the optimized trajectory briefly leaves the right lane to reduce the lateral acceleration. This effect could be eliminated, i.e. the trajectory could be forced to stay in the right lane if the force field parameters were tuned accordingly. In addition to the evident effects of the trajectory optimization, some discontinuities or “steps” (often called aliasing) in curvature and velocity profile in the staring solution were eliminated to create a smoother whole trajectory. These “steps” stem from the creation of the initial trajectory as concatenation of certain trajectory pieces.

5.3. Experimental Results

In order to show the applicability of the devised trajectory planning method - the virtual force field trajectory optimization in combination with the AWA*-based tree search trajectory initialization, test drives with a fully automated passenger car were carried out. The following test vehicle setup and system architecture have been used (for more details refer to Section 4.2):

The test vehicle is a modified BMW 540i Touring. For automated lateral guidance a steering wheel-adapter has been designed and integrated, and for automation of the braking system an additional actuator moves the pedal via levers and bowden cables. The throttle valve is activated via a modified cruise control system. The vehicle state, (position, orientation, yaw-rate and accelerations), is estimated by a Kalman filter, fusing measurements from an inertial navigation system and a DGPS receiver. Position estimates are available with a nominal accuracy of about 0.01-0.02 m at a rate of up to 400 Hz.

All test drives are carried out on an empty air-strip, (which is the reason why curved roads could until now only be tested in simulations). Obstacles are simulated and virtually inserted into the environment estimation of the autonomous vehicle.

The tree-search algorithm used for the trajectory initialization is run with a 0.2 s upper bound on computation time, the force-field based optimization is limited to five iterations. Re-planning is initiated as soon as the previous planning process has terminated. The planning horizon is set to 140 m in front of the current vehicle position.

For trajectory tracking the nonlinear decoupling controller presented in Section 4.3.1 generates steering angle and acceleration set values, which are executed by subordinate controllers for steering, throttle and brake, see Chapter 4.

5.3.1. Overtaking Static Obstacle

The test scenario consists of a straight two-lane road with a single lead vehicle $\mathcal{O}_1$ that moves with a constant velocity of 5 m/s. The desired traveling velocity of the ego vehicle $\mathcal{V}$ is set to $v_{\text{des}} = 14$ m/s. The ego vehicle starts at $v = 0$, accelerates to v_{des}, then approaches the lead vehicle $\mathcal{O}_1$ and finally overtakes it. Figure 5.17 illustrates the overtaking maneuver in five snapshots. As can be seen, the overtaking maneuver is planned and executed successfully. Both the planned and the driven path are collision-free and smooth.

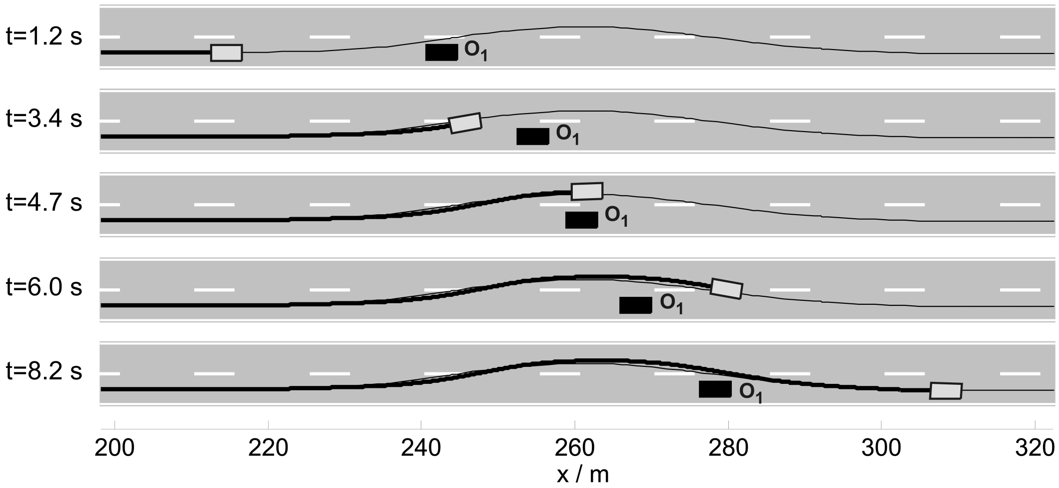


Figure 5.17.: Test drive overtaking moving lead vehicle: Snapshots
The autonomous vehicle safely overtakes the lead vehicle (obstacle $\mathcal{O}_1$). The thin line represents the planned trajectory (as concatenation of the parts used as input to the trajectory following controller), the thick line represents the actually driven trajectory.

Figures 5.18, 5.19, and 5.20 present the desired and measured values for the trajectory following control and the subordinate control loops.

The trajectory following control is analyzed by comparison of the planned and driven trajectory in Figure 5.18. Since the planned trajectory is repeatedly updated during the test drive, the (total) planned trajectory is composed of those parts of the planned trajectories that were used as reference for the trajectory following controller. The middle and bottom diagrams of the figure show the control error in lateral (Δy) and longitudinal (Δx) direction evaluated for the trajectory's given time reference for each point (x, y, t).

As can be seen, the planned trajectory is followed very well with a maximum position error of little over 0.5 m at a velocity of about 50 km/h. The lateral control error Δy is greatest at the beginning of each lane change (first to the left lane and then back to the right). The course of the lateral control error is smooth. This is remarkable because the reference trajectory changes frequently due the performed planning updates. It shows that the selection of the start states in the trajectory initialization (see Section 3.3) to reduce discontinuities in the control error was successful.

The longitudinal control error Δx is also remarkably small, especially since the reference velocity is not constant, see Figure 5.21. However, it is not as smooth as the lateral control error and small discontinuities of several centimeters can be seen for many of the planning updates, wherever a newly planned trajectory is taken as new reference.

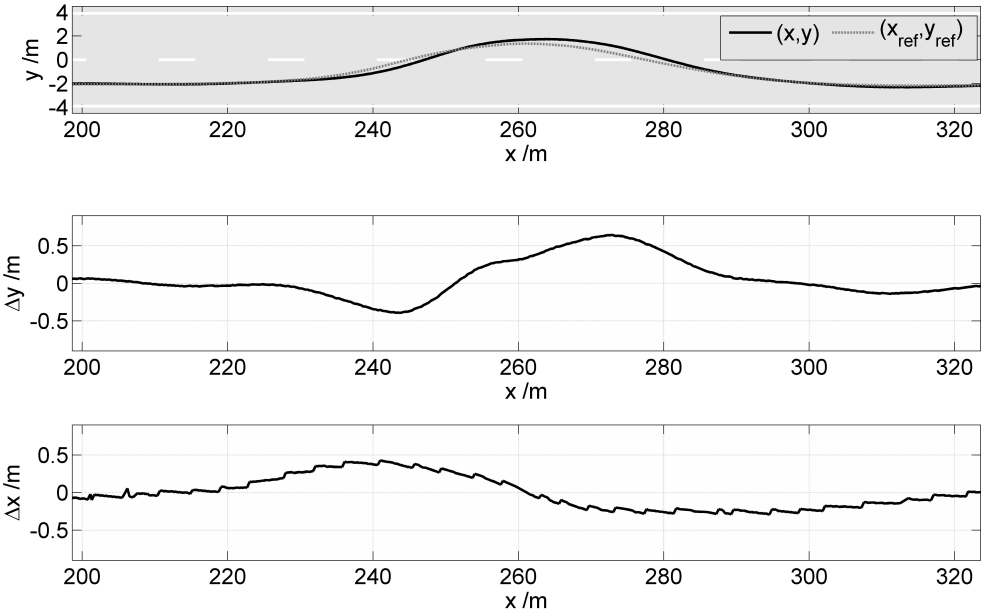


Figure 5.18.: Test drive overtaking moving lead vehicle: Deviations from planned trajectory Upper diagram shows planned and driven trajectory in x-y plane. The planned trajectory is composed of those parts of the planned trajectories that were used as reference for the trajectory following controller. Middle and bottom diagrams show the control error in lateral (Δy) and longitudinal (Δx) direction evaluated for the trajectory's given time reference for each point (x, y, t).

In order to compensate any control error in x and y, the integrated lateral and longitudinal trajectory following controller generates a desired longitudinal acceleration $a_{x,\mathrm{des}}$ and a desired steering angle δ_{des}. The longitudinal acceleration is realized by the subordinate longitudinal controller by setting desired values for the throttle flap angle α_{des} and brake pedal position $s_{B,\mathrm{des}}$ as inputs to the control of the drive train and braking system, respectively.

The desired throttle flap angle α_{des} is given in terms of the control voltage U_{DT}, where a higher voltage correlates to a larger throttle flap angle and therefore a higher acceleration of the vehicle, see Figure 5.19. The desired brake pedal position $s_{B,\mathrm{des}}$ is given in terms of the control voltage U_B, where a lower voltage correlates to stronger braking of the vehicle.

As shown in Figure 5.19, the longitudinal acceleration is rather low and barely exceeds 1 m/s^2. The ego vehicle slows down a little bit before the overtaking maneuver is initiated and during the lane change to the left lane. During the lane change back to the right the ego vehicle accelerates back to the desired traveling velocity. The vehicle's actual longitudinal acceleration follows the desired value, even though a time delay of 0.5 to 1 s can be observed

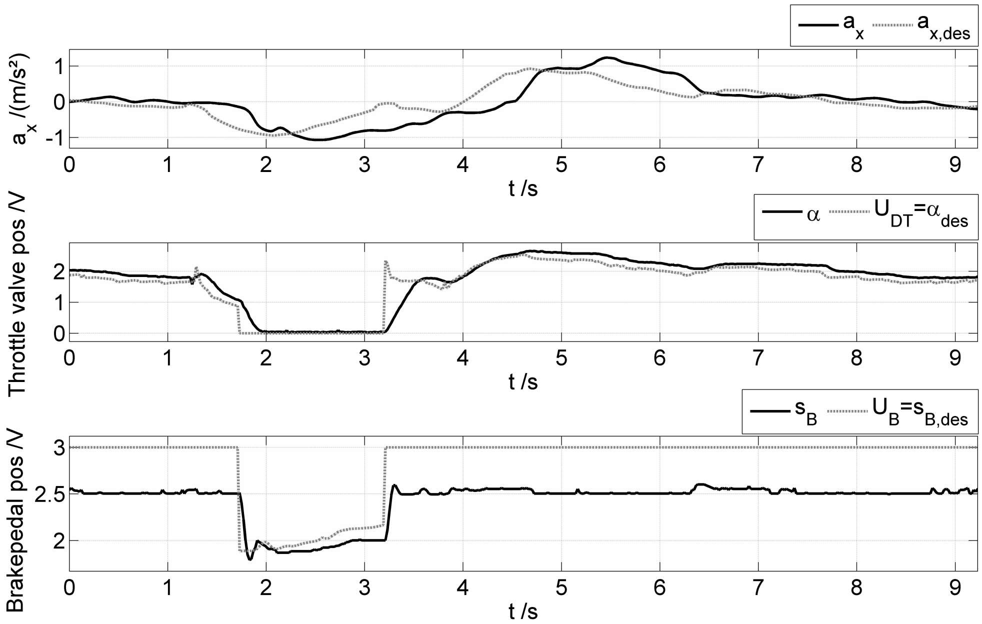


Figure 5.19.: Test drive overtaking moving lead vehicle: Desired and measured longitudinal acceleration a_x, throttle flap angle α and brake pedal position s_B
The desired longitudinal acceleration $a_{x,\text{des}}$ is an output of the trajectory following controller. In order to achieve $a_{x,\text{des}}$, the subordinate controller generates desired values for the throttle flap angle α_{des} and brake pedal position $s_{B,\text{des}}$ as inputs to the control of the drive train and braking system, respectively.

that results from the dynamic limitations of the drive train and the delay of the controller until switching to the braking system for low decelerations.

In order to decelerate, the throttle flap is closed ($U_{DT} = 0$ V) and the brake pedal is pushed ($U_B < 2.5$ V). The middle and bottom diagrams of Figure 5.19 show that the subordinate control loops make α and s_B follow the desired values well. Note, that any value of U_B greater than about 2.5 V has no further effect. Due to inaccuracies in the analogue measurements of the brake pedal position s_B, the desired value $U_B = s_{B,\text{des}}$ is set to 3.0 V in case no braking is desired. As the middle diagram shows, the dynamical limitations of the throttle flap motor cause greater control errors where step-like changes occur in U_{DT}.

The desired steering angle δ_{des} is controlled in the subordinate lateral control loop. The results are displayed in Figure 5.20. To compensate the control error $\delta_{\text{des}} - \delta$ a desired steering motor velocity $\omega_{SM,\text{des}}$ is set as input to the steering automation hardware module, see Figure 4.19 in Section 4.3.2.

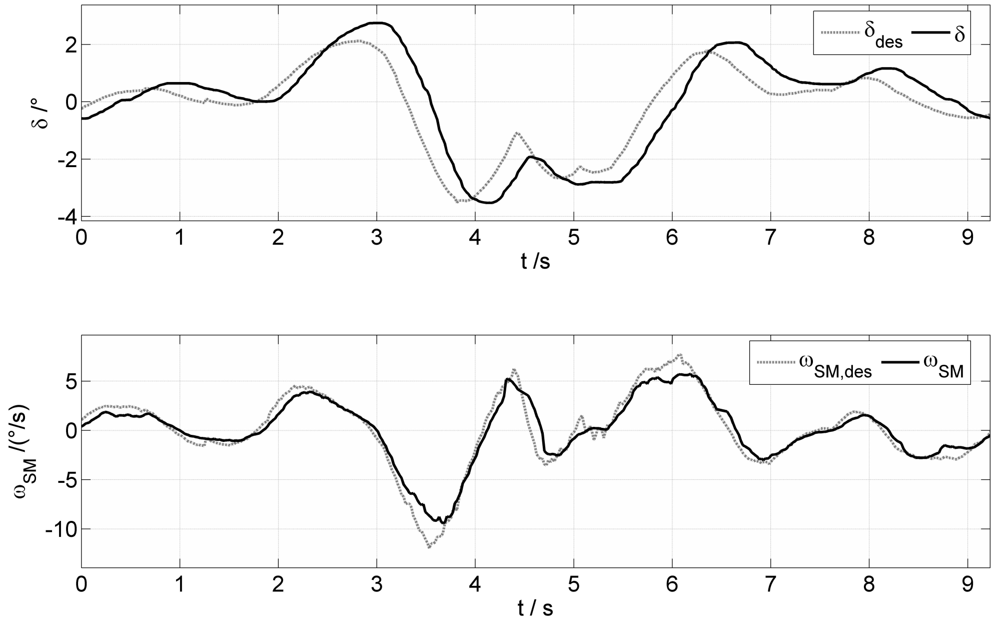


Figure 5.20.: Test drive overtaking moving lead vehicle: Desired and measured steering angle δ and steering motor velocity ω_{SM}

The desired steering angle δ_{des} is an output of the trajectory following controller. In order to achieve $\delta_{SM,\text{des}}$ the subordinate steering controller generates a desired steering motor velocity $\omega_{SM,\text{des}}$.

As can be seen in the bottom diagram of Figure 5.20, the desired steering motor velocity is followed well. Only at some peaks a relevant deviation is observable. Maybe, in these instances the required torque exceeded the steering motor's limit to cause the deviations.

The measured steering angle δ follows the desired steering angle δ_{des}, even though a certain time delay of about 200ms can be seen. Part of the reason for this effect lies in a lack of mechanical stiffness of the steering adapter construction, see torque support in Figure 4.12 in Section 4.2.4. The resulting distortion hinders quick reactions, especially when the steering direction is reversed.

To eliminate any resulting measurement error in the actual steering angle, the additional steering angle sensor was placed next to the tires, see Figure 4.14. Therefore, the mechanical distortion does not cause any error in the displayed steering angle measurement and only degrades the steering dynamics. Future improvement could include an enhancement of the mechanical construction to include additional torque support and thereby increase to overall mechanical stiffness.

While the overall lateral control quality is satisfactory and produced low deviations from the planned trajectory during the overtaking maneuver, it was not very comfortable due to a slightly “twitchy” behavior of the steering wheel with a high frequency of corrections of the steering angle. Further, during lane following of a straight road, slight oscillations of the lateral vehicle movement could be observed in some of the test drives that could cause an instable behavior at higher velocities. As potential alternative that could be explored for future improvement, for example Werling suggests not to use the steering angle as control variable but to control a certain steering rate $\dot{\delta}$ instead [Werling, 2010].

Besides the analysis of the control variables and deviations at various levels of the cascaded control loops as performed above, the measured velocity and acceleration shall be compared to the planned velocity and acceleration profiles, see Figures 5.21 and 5.22. The planned trajectory, and therefore also the planned longitudinal velocity and acceleration $v_{x,\text{traj.}}$, $a_{x,\text{traj.}}$, is composed of those parts of the planned trajectories that were used as reference for the trajectory following controller during the test drive.

It is important to note that $v_{x,\text{traj.}}$ and $a_{x,\text{traj.}}$ are no direct reference inputs to the trajectory controller, see Section 4.3.1. They are only regarded in the prefilter-part of the trajectory following controller and the compensation of accumulated position errors in x or y might necessitate deviations from the planned velocity and acceleration profiles.

The planned velocity profile is followed with a maximum deviation of about 0.5 m/s. The planned velocity profile shows only minimal discontinuities where the planned trajectory was updated.

These discontinuities cause small spikes in the planned acceleration profile, see Figure 5.22. An additional discontinuity of the planned acceleration profile occurs where the slope of the planned velocity profile is reversed at around $t = 4$ s upon a planning update. Besides the small “spikes” that occur at planning updates, the measured acceleration profile resembles the planned one, especially before and after the overtaking maneuver.

Concluding, the trajectory following control proved very successful and allowed only deviations in position of less than 0.5 m from the planned trajectory. The mentioned discontinuities in planned velocity and acceleration that occur at planning updates influence the trajectory following control as they are an input to the pre-filter, see Section 4.3.1. Therefore, if the same or a similar trajectory following controller is used, future improvement could target to reduce the observed discontinuities at planning updates.

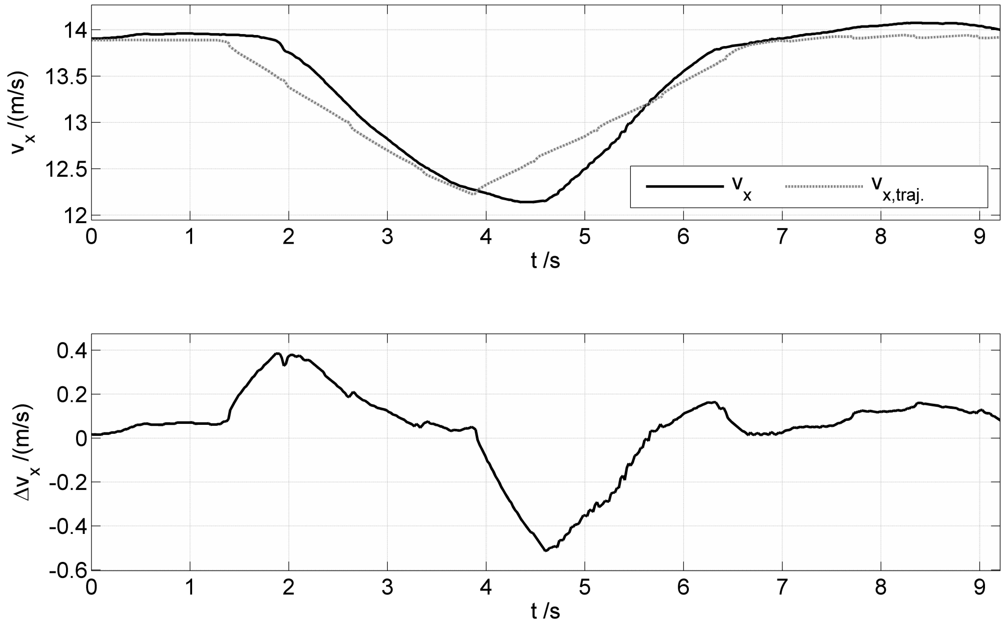


Figure 5.21.: Test drive overtaking moving lead vehicle: Planned and measured longitudinal velocity v_x and difference Δv_x

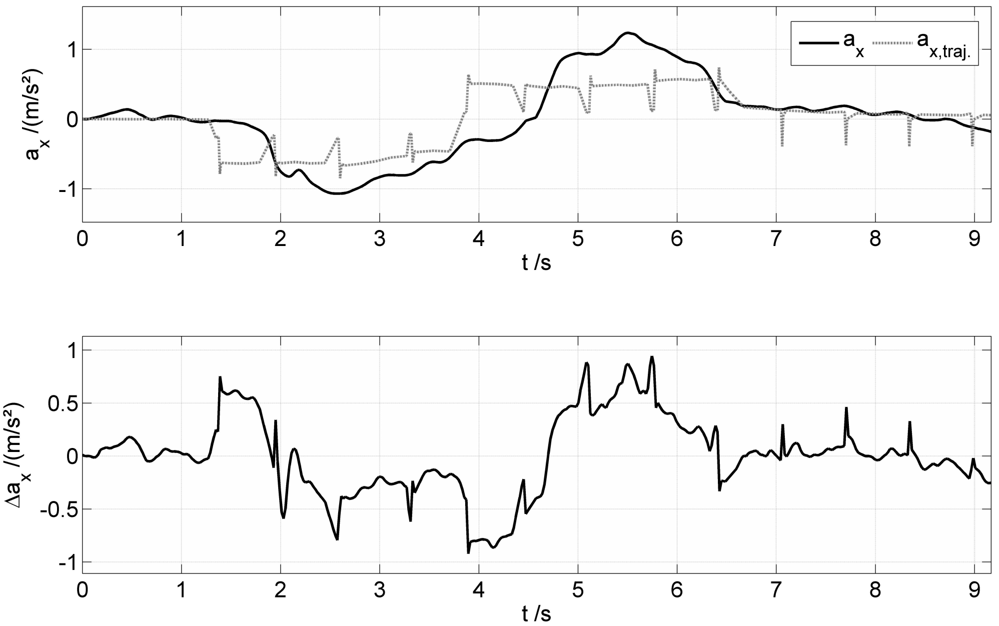


Figure 5.22.: Test drive overtaking moving lead vehicle: Planned and measured longitudinal acceleration a_x and difference Δa_x

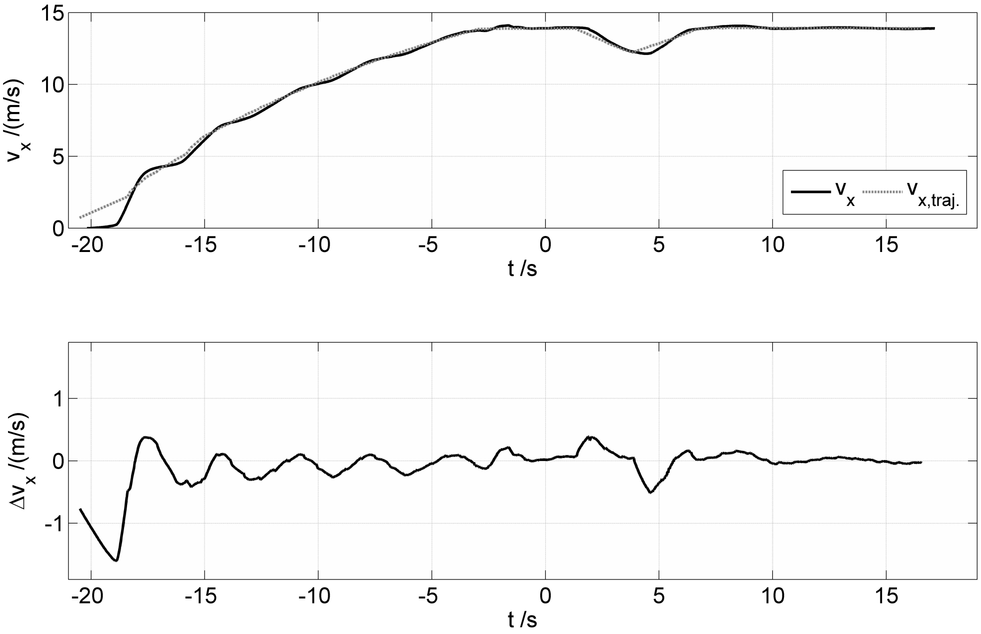


Figure 5.23.: Test drive overtaking moving lead vehicle: Planned and measured longitudinal velocity v_x and difference Δv_x

In addition, while even velocity (and acceleration) follow the planned profiles, the longitudinal control still shows suboptimal results in some cases, as can be seen during first phase of the test drive while accelerating from standstill to the desired traveling velocity, see Figure 5.23.

While accelerating to the desired traveling velocity, the vehicle's velocity oscillates around the planned ramping profile. The amplitude of the oscillation decreases and the velocity finally settles after only a small overshoot. Nevertheless, the oscillation and overshoot produce unnecessary accelerations which reduce the comfort and show room for potential future improvement.

5.3.2. Overtaking with Oncoming Traffic

In addition to the simple overtaking scenario presented in the previous section, the applicability of the devised trajectory planning algorithm and implemented trajectory following is demonstrated in two additional overtaking scenarios with oncoming traffic.

The test scenario consists of a straight two-lane road with one lead vehicle $\mathcal{O}_2$ and one oncoming vehicle $\mathcal{O}_1$. $\mathcal{O}_1$ moves with a constant velocity of 15 m/s, $\mathcal{O}_2$ with 5 m/s. The driven trajectories of these two obstacles and the ego vehicle are shown in Figure 5.24 as projections to the x-t-plane, where the black line indicates the movement of the ego vehicle.

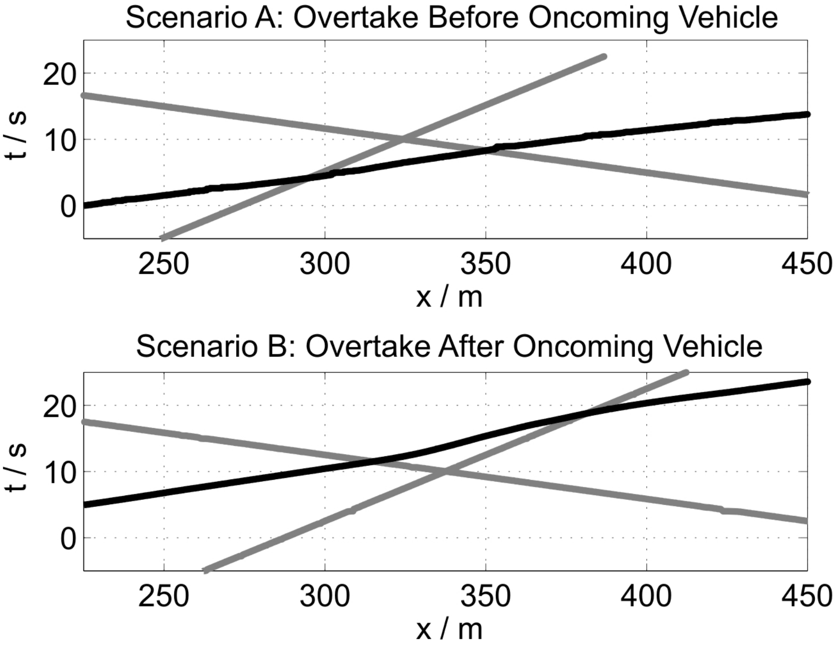


Figure 5.24.: Test drives: Results in x-t-plane
Driven trajectory of autonomous vehicle and obstacle movements are projected unto the x-t-plane. The autonomous vehicle is depicted in black, obstacle swaths are gray.

In this representation a lower slope corresponds to a higher velocity. Obstacle $\mathcal{O}_2$ was set to drive slower than the desired traveling speed 25 m/s of the ego vehicle, therefore the lead vehicle should be passed. Depending on the timing when the ego vehicle starts, it is advantageous to either drive somewhat slower when following the lead vehicle and wait until the oncoming vehicle has passed, or to accelerate and pass the lead vehicle before the oncoming traffic has approached. These two options were tested and are shown in Figure 5.24.

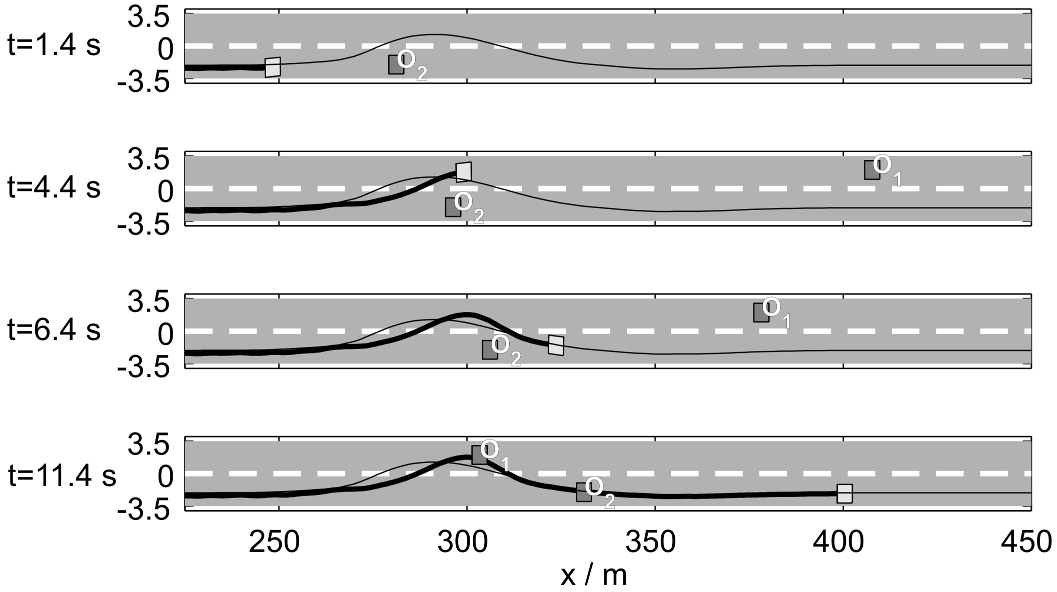


Figure 5.25.: Test drives: Snapshots Scenario A
The autonomous vehicle overtakes obstacle $\mathcal{O}_2$ while avoiding collisions with the oncoming traffic $\mathcal{O}_1$.

Figures 5.25 and 5.26 show snapshots that detail the driven trajectory of the ego vehicle in the x-y-plane (thick line) as well as the total planned trajectory (thin line) as concatenation of the planned trajectories from one planning update to the next. The illustrations at different instances of time show the scenario progress.

In scenario A, Figure 5.25, it can be seen that the ego vehicle successfully overtakes the lead vehicle before the oncoming vehicle approaches, while in scenario B, Figure 5.26, it can be seen that the ego vehicle plans and executes its motion such that it waits for the oncoming traffic and then overtakes.

The two-step trajectory planning successfully developed drivable and collision-free trajectories to follow the road and overtake a slower lead vehicle while avoiding oncoming traffic. In doing so, the trajectory planning created suitable paths and velocity profiles to either accelerate or decelerate before and during the overtaking maneuver.

In scenario A the test vehicle achieves a traveling velocity of about 15 m/s at $t = 0$ before it arrives behind obstacle $\mathcal{O}_2$. To execute the overtaking maneuver, the test vehicle accelerates to 18 m/s at $t = 3.2$ s. Right after having passed $\mathcal{O}_2$, the vehicle brakes down to 11.5 m/s at $t = 5.7$ s to avoid a collision with the oncoming traffic while swerving to the right lane. After completion of the maneuver and until the end of the experiment the test vehicle accelerates to 22 m/s at $t = 22$ s.

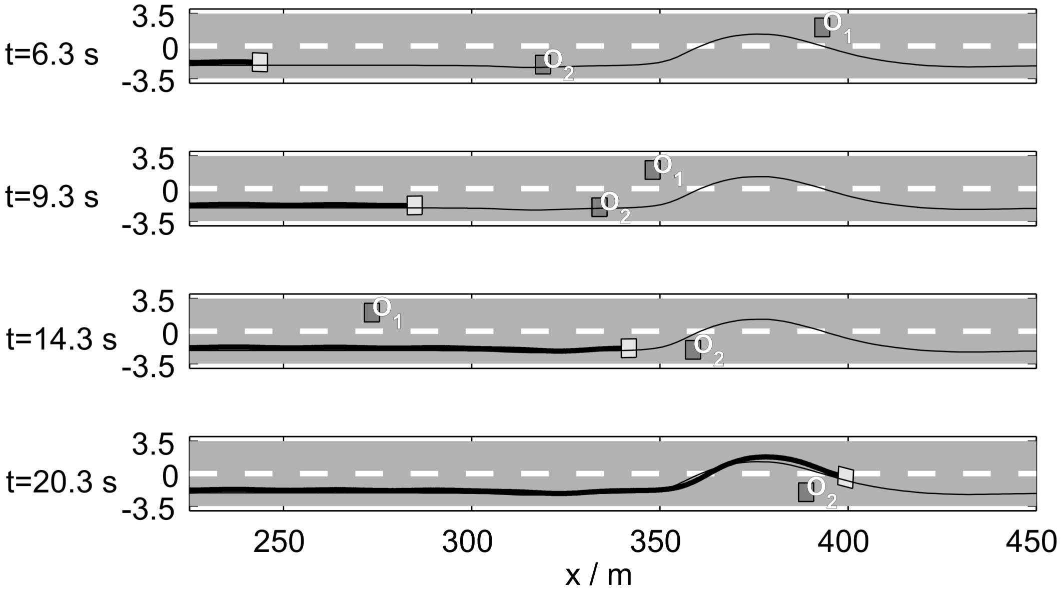


Figure 5.26.: Test drives: Snapshots Scenario B
The autonomous vehicle overtakes obstacle $\mathcal{O}_2$ while avoiding collisions with the oncoming traffic $\mathcal{O}_1$.

Scenario B starts with a higher traveling velocity. The maximum traveling velocity of 25 m/s is attained at $t = 0.6$ s before the test vehicle closes up to the preceding obstacle $\mathcal{O}_2$. The vehicle has to slow down to a minimal velocity during this test drive of 7.5 m/s, waiting to let the oncoming traffic $\mathcal{O}_1$ pass until $t = 15$ s. To overtake obstacle $\mathcal{O}_2$ afterwards, the test vehicle again accelerates and achieves a velocity of 13.5 m/s at $t = 20$ s.

As can be seen in Figures 5.25 and 5.26, the trajectory following controller manages to make the vehicle follow the planned trajectory, but a maximum lateral deviation of 1.4 m can be observed at the beginning of the overtaking maneuver in Figure 5.25. As the analysis shows, here, the trajectory following controller still has to be improved.

CHAPTER 6

Conclusion and Future Prospects

As evidenced, the increasingly complex task of driving might be alleviated by further assistance and automation systems. For the development of such future assistance systems and vehicle automation a motion planning algorithm is a key module. Most of today's planning algorithms for road-based autonomous vehicle navigation plan ahead simple swerving maneuvers and select longitudinal velocity profiles in a decoupled manner. For complex maneuver execution these algorithms have to rely on emergent behavior and are thus limited in solution-space and foresightedness.

In this contribution, a trajectory planner specialized to the domain of road-based navigation is presented, which facilitates the generation of complex long distance maneuvers, including combined planning of steering and throttle / brake. To maximize the exploitation of a-priori domain knowledge, a two-step approach is employed: An initial trajectory is computed in x, y, ψ, v and t using an A* derivative, afterwards a force-field based local optimization algorithm deforms a discrete x-y-t-representation of the trajectory depending on the representation in the augmented phase space $\mathbb{X}_T$ in x, y, t, ψ, v, a_x, a_y, $\dot{a}_x$ and $\dot{a}_y$.

The two-step algorithm combines the advantages of an efficient coverage of the search space and the resolution completeness of the A* based graph search in the trajectory initialization with the intuitiveness of the environment representation of potential field approaches in the optimization step.

During the optimization step, the planned trajectory remains homeotopically equivalent, i.e. the “kind” of trajectory is selected in the initialization. Hence, the devised planning algorithm can be easily extended to plan ahead several alternative trajectory simultaneously or regard higher level decisions in the selection of the planned maneuver. These decisions could stem from central traffic management centers, “negotiations” with other vehicles in the context of cooperative driving, or detected driver intentions. The possibility of a combination with a behavior-based maneuver decision module or a rule-based approach for cooperative riving of several vehicles or robots has been demonstrated in several supervised student works, [Knolle, 2007, 2008; Welzel, 2009].

Compared to many other potential field or force field methods, the approach is predictive, since the force field acts on the planned trajectory and not the vehicle itself. The added dimensionality allows an integrated lateral and longitudinal, i.e. an integrated path and velocity planning.

Both the trajectory initialization and optimization have anytime characteristics which allows to set constraints on the algorithms execution time even in environments of varying complexity. For the trajectory initialization this is realized by extending an Anytime Weighted A* (AWA*) variant. For the trajectory force-field optimization, an Armijo stepsize control guarantees a monotonous decrease of the forces with each iteration.

The trajectory planning algorithm regards dynamic obstacles and extrapolates their motion, either along or exiting the road. The restriction to limited curvatures has been lifted by the application to local reference frames along the road.

A further enhancement compared to earlier predictive potential field trajectory optimization algorithms like the elastic bands, the vehicle dynamics are taken into consideration in terms of lateral and longitudinal velocity, acceleration, and jerk. This is true also at the beginning of the trajectory, where the initial orientation and velocity are regarded. This characteristic is important also for frequent planning updates.

As shown, the formulation of the internal forcefields in terms of these dynamic variables also allows a variable discretization without a large effect on the planned trajectory. Also it was demonstrated that the resulting trajectory does depend heavily on the chosen planning distance.

The characteristics of the planned trajectory depend on a number of parameters. In order to simplify the tuning problem, some basic considerations and rules have been devised in this work.

Potential future improvement of the trajectory planning approach include the a reduction of the needed computational resources to speed up the motion planning. In this work the implementation of the trajectory planning was not optimized in this regard and a speedup by a factor of 5-10 should easily achievable. Further improvement could be accomplished by employing a more detailed motion model in the optimization step or even a closed-loop "pre-simulation" could be tested inside the trajectory planning. For a pure path planning variant of this force field trajectory planning (without the extension to the time dimension and the integrated planning of a velocity profile), the exploitation of an inverse vehicle model to define the internal force fields and to test for drivability has already been devised and tested successfully, supported by a diploma thesis, [Kahl, 2007].

In addition to the development of the motion planning approach, a test vehicle was set up. A series vehicle was equipped with the necessary sensors, actuators and information processing capabilities for autonomous driving.

The dynamic state of the vehicle is determined by a Kalman-filter based data fusion of a differential GPS with inertial measurements. External sensors for the detection of the environment are simulated online based on digital maps. In order to allow automated steering, a removable steering adapter was constructed. The braking system was automated by a construction of levers and bowden cables to actuate the brake pedal. The drive train was automated by an adaptation of the existing cruise control units.

A trajectory following controller was implemented. Upon a comparison of several different controllers an integrated lateral and longitudinal control approach based on the concept of nonlinear decoupling was chosen. A subordinate acceleration controller decides between the use of the braking system versus the drive train and exploits recorded maps to approximate their static transfer behavior. For steering, braking system, and drive train automation hardware modules further subordinate control loops were implemented.

The devised trajectory planning approach was implemented ans tested in various scenarios, planning single trajectories as exemplified in Sections 5.1 and 5.2 as well as in closed-loop simulations using the configuration illustrated in Section 4.1.2, Figure 4.3. In these simulations a large variety of scenarios, including curved roads such as in the example in Figures 5.15 and 5.16 and higher velocities of the ego vehicle could be tested successfully, planning and executing drivable, collision-free trajectories and the properties described above could be demonstrated.

The algorithm was then deployed in the built test vehicle and test drives with simulated static and moving obstacles were carried out. The suitability to online planning was successfully demonstrated, including traffic situations where combined manipulation of both steering and throttle/brake was necessary to resolve imminent conflicts. This constitutes a considerable improvement compared to similar earlier algorithms with sequential path and velocity planning.

During the test drives, the reoccurring update of the planned trajectory produced a continuous reference in lateral direction and only very small discontinuities on the order of several centimeters in longitudinal direction. The resulting planned velocity profile is rather smooth, however, these small discontinuities in the planned position caused some spikes in the planned acceleration. As a potential future improvement, these small discontinuities might be reduced or the planned acceleration that is an input to the prefilter part of the trajectory following controller, could be filtered to create a smoother control input.

The trajectory following behavior of the test vehicle was satisfactory, but still shows some room for future improvement. While the control error in the avoidance of a static obstacle was less than 70 cm in lateral direction and below 50 cm in longitudinal direction, this performance is not yet very consistent, especially at higher velocities, as evidenced in other test runs for overtaking with oncoming traffic. Here, a delay in vehicle response led to a larger lateral deviation of 1.4 m from the path (although a collision was not entailed). The oscillations observed during the acceleration phase from standstill further show potential for improvements for the longitudinal control.

As detailed, the performance of the trajectory following control might be enhanced also by improvements in the construction of the automation hardware modules. For the steering, the mechanical stiffness of the steering adapter could be increased to reduce the mechanical distortions. The steering motor could be replaced by a direct drive, to eliminate the transmission and thus reduce mechanical friction and play. For the acceleration, the cruise control motor that was used to operate the throttle flap could be replaced by an actuator of higher performance to enable quicker reaction times.

For applications like future driver assistance systems, the proposed trajectory planning approach could possibly be used as a basis to achieve a more flexible and foresighted artificial decision making and trajectory planning - especially in more complex situation where a simple braking or lane change maneuver is not sufficient anymore.

Nonetheless, for the application in future driver assistance systems, it would be important to regard not only the technical applicability and performance, but to include the human driver in the system design. To enable an optimal interplay between driver and automation, the automation (including the trajectory planning and following) should be compatible with the drivers' expectations. In order to achieve this, further adaptation of the trajectory planning might be necessary, e.g. to increase safety distances or make the trajectory optimization more comfort-oriented. The implemented trajectory following controller was only intended for the demonstration of fully automatic driving and would probably have to be redesigned if a human driver entered the control loop.

APPENDIX A

Research Vehicles

Organized by the American Defense Advanced Research Projects Agency (DARPA) the robotic competition Urban Challenge was held in the year 2007. The contestants had to build autonomous vehicles that were able to drive in an urban scenario with other moving traffic-participants. The three major situations the cars had to handle were

- driving along roads,
  for which GPS-coordinate based road definitions were supplied in a road network definition file (RNDF),
- correct handling of preference rules at crossroads and intersections,
- navigation in unstructured areas, such as parking lots.

The following sections review the first three cars of the competition:

- Boss, Carnegie Mellon University,
- Junior, Stanford University
- Odin, Virginia Polytechnic Institute and State University

and additionally

- Annieway, University of Karlsruhe,

one of the two German cars, which made it to the finals of the competition.

They stem from [Hess, 2009], a diploma thesis supervised by the author. It has to be pointed to the fact that the publications of the four groups have different levels of detail for the different aspects. Therefore, this review cannot give a one-to-one comparison of the concepts.

A.1. Boss, Carnegie Mellon University

The soft- and hardware-components of the DARPA Urban Challenge (DUC) winning vehicle Boss (Carnegie-Mellon University) are described in [Urmson et al., 2008]. The behavioral architecture of Boss aims to reduce the complexity of planning, by identifying the behavior which is most fitting to the current situation, (the "driving context"). A restricted motion planning problem is solved, which takes into account only the environmental constraints relevant to the "driving context". The behaviors used to handle the situations generate goals, which are then sent to one of the two motion-planning subsystems responsible for road-based navigation and navigation in unstructured environments.

The behaviors for road-following are concerned with distance keeping to preceding vehicles and lane-change planning. Distance keeping is achieved with a linear control-law, setting the commanded velocity proportional to the difference between desired and actual vehicle distance. The lane-change planner is rule-based and analyzes for single merge maneuvers into all available gaps in traffic, so-called slots, whether time, velocity and acceleration constraints can be met and whether safety distances to other traffic participants can be maintained.

All traffic-participants' movements are predicted along their associated lane. In uni-directional multi-lane roads a feasible merge which is furthest down the road is selected for execution, to make fast progress. For bi-directional two-lane roads the closest feasible slot is chosen, to minimize the time spent in a wrong-direction road. For such standard situations the road-following behaviors generate commanded velocity and a goal position in a lane as input to the road-based motion planner. Error recovery behaviors allow to identify blocked roads. In such a case the motion planner for unstructured environments is used to calculate U-turns or to wander indefinitely along the road.

The behaviors for intersection handling analyze the geometric structure of intersections for precedence rules and monitor the traffic in intersections to generate goals which comply to the traffic rules. If an intersection is free and if Boss has precedence over other waiting vehicles, goals are generated for the road-based navigation system. When the intersection is occupied, a time-out allows to identify deadlocks caused be broken-down vehicles. In such a case goals for the unstructured motion planner are generated.

During zone-navigation the behavior-system specifies only a single goal-position, such as a parking slot. The plan from the current vehicle position to the goal-position has to be created by the motion planner for unstructured environments.

A.1.1. Road-Based Motion Planner

Instead of sampling the command space, as in the case of the other three reviewed cars , around the lane-center trajectory, the state space along the lane is sampled. The motion planner generates curves to a set of goals at a certain planning distance along the lane, varying the lateral offset from the lane-center. For each goal one smooth and one sharp trajectory are generated. The smooth trajectory has an initial curvature as predicted by the vehicle state, whereas the sharp trajectory exhibits a constant curvature offset to produce a quick initial action when the vehicle starts following the trajectory.

A constant, linear, ramp or trapezoidal velocity profile is calculated for each trajectory, taking into account the decision of the behavioral system, the road's speed limit, the limitation by the curvature of the trajectory and the desired goal velocity. Each trajectory is evaluated against proximity to static and dynamic obstacles, lateral offset from the lane-center, and smoothness. The feasibility is verified with the help of an accurate vehicle model.

A.1.2. Unstructured-Environment Motion Planner

As described in [Ferguson et al., 2008], the algorithm Anytime-Dynamic-A* (AD*), a combination of ARA* and D*Lite (see Section 1.1), is employed for planning of complex, long distance maneuvers in unstructured environments. As the search is performed in a backward direction from the target location and orientation towards the vehicle state, the changes perceived in the environment of the vehicle can be efficiently integrated into existing plans. The vehicle state used during planning, (x, y, ψ, v), contains position and heading of the car, as well as the longitudinal velocity. To represent possible movements, the lattice generation algorithm offline generates one high-resolution and one low-resolution primitive path set. The high-resolution set is applied in the vicinity of vehicle and goal, whereas the low-resolution set saves computation time in between.

A combination of two heuristics guides the search: The first heuristic takes the motion model of the vehicle into account and disregards obstacles. The cost of a feasible path from the robot to any position in a limited environment is precomputed offline with an uninformed search algorithm and stored in a look-up-table for on-line access. The second heuristic gives the cost of a 2D path from the robot to any position in the zone, regarding the perceived obstacles. It is calculated before execution of the AD* algorithm by a Dijkstra's search. The final heuristic of the AD* algorithm is the maximum of both heuristics. To avoid moving obstacles, the complete short term prediction of any obstacle's path in the robot's vicinity is marked blocked in the static obstacle map.

A.2. Junior, Stanford University

Junior's navigation module consists of several motion planners plus a hierarchical finite state machine (FSM), which invokes different behaviors and prevents deadlocks, [Montemerlo et al., 2008]. The FSM contains states for intersection handling, one state for road-based navigation (forward driving, lane keeping and obstacle avoidance), which is preferred when not in parking lots, several states for handling of blocked roads (U-turns, divider crossing and free-form navigation to handle difficult road-blocks) as well as one state for navigation in parking lots.

A.2.1. Road-Based Motion Planner

As base trajectories the lane-center trajectories defined by the RNDF in global coordinates are used. For each lane and base trajectory a set of trajectories with differing lateral shifts at the end-points are generated by simulating a vehicle model for different steering parameters. The preference between the different trajectories is decided based on the required execution time, cost along the trajectory and remaining cost to the goal-location, which is estimated by a global planning system that analyzes the RNDF. Trajectories are generated for all accessible roads splitting at a junction to allow the local execution costs to influence the global routing of the car. The planner was found to work well in "well-defined traffic situations", planning smooth motion along unobstructed roads and handling simple passing and evasion maneuvers.

A.2.2. Unstructured-Environment Motion Planner

A free-form planner is responsible for navigation in parking lots, U-turns and traversal of blocked intersections and one-way roads. The hybrid A* method is employed to plan a path towards a static goal location, such as the next GPS waypoint or a parking box. The vehicle is modeled by a four-dimensional state-vector (x, y, ψ, v) containing the x- and y-position, the orientation of the vehicle and the longitudinal direction of motion. (The velocity v has only two possible values: forward and backward.)

Two heuristics help the A* planner to find a path to the goal: A “holonomic-with-obstacle” heuristic (h1) ignores the vehicle-model’s nonholonomic constraints and generates a path in the x- and y-dimension as an admissible estimate of the remaining distance to the goal. This heuristic is better informed than the euclidean-distance estimate, as it prevents the planner to enter local deadends. A “nonholonomic-without-obstacles” heuristic (h_2) helps to estimate the amount of maneuvering necessary to approach a goal-position with the required orientation. The heuristic h_1 is calculated on-line with a dynamic-programming method to incorporate the environmental constraints perceived by the vehicle. The heuristic h_2 is context-insensitive and thus can be accessed from a pre-computed data structure.

The free-form planner has to post-process a path to the goal as the small number of discrete actions available to the hybrid A* algorithm results in rapid steering angle changes. A not precisely specified “conjugate gradient smoother” modifies controls and moves way points locally, using an objective function which penalizes steering-wheel motion and curvature.

A.3. Odin, Virginia Polytechnic Institute and State University

The planning component of Odin’s software suite is described in [Bacha et al., 2008] with the help of a hierarchical architecture. Ordered from global and strategic to local and tactical, the layers of the architecture are:

1. A deliberative route planner working on the RNDF, which generates a sequence of lane-exits leading to the goal location.
2. A set of reactive driving behaviors producing a short-term goal. The short-term goal comprises six target locations in global coordinates, a desired maximum speed, zone

and safety-area flags, and for each location the travel lane, the driving direction, a desired heading, as well as a stop flag.

3. A deliberative motion planner, which analyzes feasible trajectories and sends motion commands to the vehicle interface that are best fitted to reach the short-term goals.

The reactive second layer consists of seven driving behaviors that send their desired short-term goal to an arbitration unit. The arbitration unit selects the short-term goal with the maximum benefit. The driving behaviors *Route*, *Passing* and *Blockage Driver* handle road-based navigation, intersections are managed by the *Precedence*, *Left Turn* and *Merge Driver*, for navigation in parking lots and unstructured environments the *Zone Driver* is responsible.

The Route Driver follows a lane according to the route planner, when no obstacles or traffic are encountered. In case of slow moving obstacles the Passing Driver analyzes whether the situation's parameters such as left lane type (oncoming or forward lane) permit an overtaking maneuver and generates the necessary short-term goals for the motion planner, (the algorithm is not specified). The Blockage Driver analyzes the drivable state of all available lanes and requests the route planner to update the sequence of lane-exits, when all forward lanes are blocked. The details of how a U-turn has to be executed seem to be handled by the third motion planning layer.

The behaviors responsible for intersections monitor critical areas for traffic and calculate whether turning maneuvers can be executed safely. If the vehicle has to wait, the stop flag is set for a target-point before the entrance of the intersection. The Merging Driver controls the vehicle speed in such a way that the vehicle is able to merge into gaps in the traffic.

For parking lot navigation it was initially attempted to automatically infer a regular structure from the a priori environment information, which would have then enabled a behavior to drive on lanes between parking rows. Because that approach was not judged realizable, the team had to manually specify “control points” in zones. The route planning layer generates a list of control points leading to the goal location on the parking lot. The Route Driver simply follows the list of control points. The Zone Driver detects situations in which the goal is not reachable via the list of control points, then disconnects links between control points and requests a new route. Obstacle avoidance is solely managed by the motion planning layer.

A.3.1. Motion Planner

Odin relies on a single motion planner which is configured differently for road- and zone-based navigation. For road-based navigation planning of acceleration and steering are decoupled. A speed limiting module analyzes the traffic in the future path of the vehicle and takes stop commands into account to generate a maximum velocity. This enables Odin to wait for the passage of oncoming traffic or to follow preceding cars.

Inside zones, the speed-limiter is disabled and the motion planner has to handle dynamic obstacles all by itself. An ego-graph approach is employed to plan a series of motion commands, which specify desired curvature, curvature rate of change, desired velocity, maximum acceleration and time duration. Sets of trajectories are precomputed for initial steering angles varied at 0.25° increments. Possible commands are steering angle velocities of $\{0, 6, 12, 18\}\ \frac{\circ}{s}$ combined with longitudinal velocities of $\{2, 5.5, 9, 12.5\}\ \frac{m}{s}$.

The trajectory search seems to use an objective function based on execution time and obstacles distance. “While traveling in segments, trajectory search chooses goals that travel down a lane. In contrast, zone traversal is guided by target points along with goal criteria and search heuristics to produce different behaviors.” ([Bacha et al., 2008])

A.4. AnnieWay, University of Karlsruhe

Team AnnieWay was one of the two German teams qualified for the DUC finals. The publication [Kammel et al., 2008] describes the responsibility for motion control in the autonomous vehicle as sub-divided between a reactive and a deliberative component. The reactive component receives trajectories from the deliberative component and decides whether any trajectory can be safely executed given the latest environment information. If such a trajectory is not obstacle-free, the reactive component resorts to a command-space approach.

Similar to all three above mentioned autonomous vehicles, AnnieWay’s deliberative component comprises a *Mission Planner*, which generates a global RNDF-based strategic plan, and a *Maneuver Planner*, which handles the local planning by following the RNDF-based trajectory, applying traffic rules and deciding for driving maneuvers. The Maneuver Planner is implemented as a hierarchical state machine that groups and organizes driving behaviors. The three top-most behaviors responsible for driving realize the division into road-driving, zone-driving and intersection-handling, familiar from above mentioned vehicles. Road-driving includes special behaviors for lane-following, lane-changing and direction-changing with a “k-turn”. Zone-driving includes approaching, entering, parking and driving to the exit. The intersection behavior manages approaching, queuing, waiting and traversal.

A.4.1. Road-Based Motion Planner

For road-based navigation through moving traffic a special algorithm is employed, which calculates velocity profiles for merging and intersection-crossing. Velocity profiles are restricted to a set of two-phase linear ramp profiles, where a desired velocity v_d is reached by applying the maximum acceleration a_{sat} in phase one and v_d is held constant in phase two. The resulting velocity profiles have one degree of freedom, v_d. The algorithm, of which a detailed description can be found in [Werling et al., 2008], operates on a set of obstacles, (other traffic participants with right of way), whose movement is extrapolated along the middle of their associated lane with constant velocity.

Traffic scenarios are represented by equivalence graphs which model conflict free parts of extrapolated paths (time independent) as edges associated with their respective length, and conflict positions, where paths intersect and possible collisions could occur, as vertices. To ensure that paths are collision free, the passage of two vehicles along the same equivalence graph vertex is inspected for time and spacial gaps. Spacial gaps are more relevant at lower velocities, whereas time gaps ensure safety at higher velocities. A maneuver of overtaking one moving obstacle can be represented in an equivalence graph by considering not absolute, but relative velocities between the planning vehicle and the obstacle, effectively abstracting the dynamic passing maneuver to a static passing maneuver.

A.4.2. Unstructured-Environment Motion Planner

In unstructured environments an A* planning algorithm[Ziegler and Werling, 2008] with command-space discretization is used. The vehicle state is modeled by a four dimensional vector $(x, y, \psi, delta)$, which contains the position in the plane, the orientation and the steering angle, respectively. The steering angle dimension is discretized as $D = \{\delta_1, \ldots, \delta_n\}$. For each known configuration q the set of reachable states is enumerated by simulating a kinematic vehicle model with q as initial state and a constant steering rate $\dot{\delta} = (q_\delta - \delta_i)/s$ for the fixed arc-length s and any terminal steering angle $\delta_i \in D$. The resulting clothoid-like arcs are precomputed for all possible steering angles and accessed on-line by translation and rotation of the arc to fit the position and orientation of an initial state.

The cost function employed by the A* algorithm seems to be based on a pseudo-cost combination of path length and obstacle distance: A Voronoi-diagram of the grid-represented, two dimensional workspace is created and a graph search algorithm is used to trace the shortest distance along Voronoi-edges from the goal towards any position on a Voronoiedge. The cost of any position outside a Voronoi-edge is calculated with the help of the nearest position on the Voronoi-edge to create a cost function which is sloped towards the Voronoi-edges. The obstacle-aware distance-to-goal computed from the Voronoi-diagram is combined with a “non-holonomic-without-obstacles”-distance by taking the maximum of both distances. How the aggregated path cost of a clothoid-arc is actually calculated, (whether from all positions in the arc’s swath, or only from the arc’s endpoint), is not apparent.

A.5. Conclusion

The similarities between the approaches are striking: Similar architectures were used to create a hierarchy of software components for vehicle control. At the highest layer always resides a behavior- or rule-based decision unit, which selects the appropriate planner according to the context. These contexts are roughly road-based driving, intersection handling, road-blockage resolution and driving in unstructured environments.

In all cases the road-based driving is solved with a motion planner that generates simple trajectories for road-following. In case of AnnieWay, Odin, and Junior, command-space search algorithms were employed. In the vehicle Boss a state-space search method was employed, which though produces trajectories of great resemblance to the trajectories of a road-based command-space method.

AnnieWay, Junior, and Boss solved planning in unstructured environments and during road-blockages with an A* state-space search, which employs the combination of a kinematic distance estimate and an obstacle-regarding 2D distance estimate as a heuristic. Odin applied the command-space search even in unstructured environments. Junior applied a “conjugate gradient smoother” as post processing or second planning step that modifies controls and moves way points locally.

APPENDIX B

Calculations regarding Road Segments

B.1. Clothoids

As illustrated in Figure B.1, for each infinitesimal segment of a clothoid the following relation holds

$$d\phi\left(s\right) \quad = \quad \kappa\left(s\right)ds. \tag{B.1}$$

Integrating Equation B.1 from $s_0 = 0$ to s for a clothoid that starts in CL^* in ${}_{CL}\mathbf{e}_x$-direction, $\phi\left(0\right) = 0$, with $\kappa\left(0\right) = 0$ using Equation 2.32 yields

$$\phi\left(s\right) \quad = \quad \int\limits_0^s \kappa\left(\bar{s}\right)d\bar{s} = \frac{1}{2}\kappa' s^2. \tag{B.2}$$

As can be seen in Equation B.2 and Figure B.1, the angle $\phi\left(s\right)$ is equal to the orientation ψ of the clothoid with regards to ${}_{CL}\mathbf{e}_x$

$$\psi\left(s\right) \quad = \quad \phi\left(s\right). \tag{B.3}$$

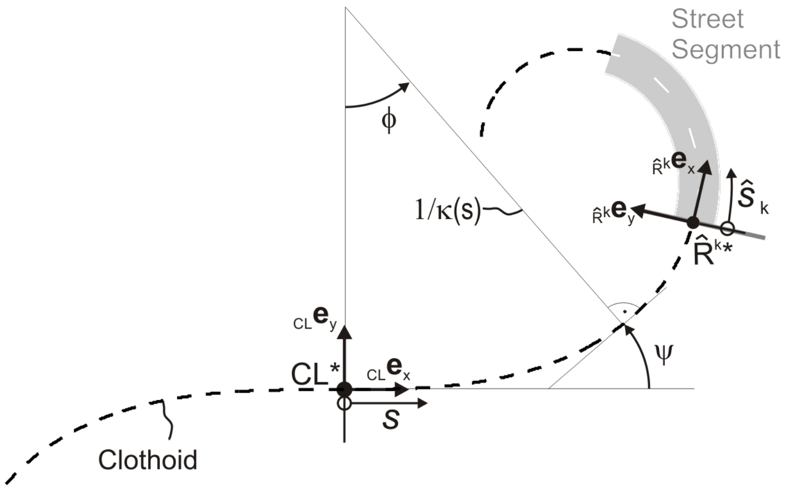


Figure B.1.: Clothoid segment

The differential changes $_{CL}d\mathbf{r} = [_{CL}dx, {_{CL}dy}]^{\mathsf{T}}$ in the clothoid coordinates are given by

$$_{CL}d\mathbf{r}\left(s\right) = \begin{bmatrix} _{CL}dx\left(s\right) \\ _{CL}dy\left(s\right) \end{bmatrix} = \begin{bmatrix} \cos\left(\phi\left(s\right)\right)ds \\ \sin\left(\phi\left(s\right)\right)ds \end{bmatrix}. \tag{B.4}$$

Therefore, the position vector $_{CL}\mathbf{r}^{CL^*,P} = [_{CL}x, {_{CL}y}]^{\mathsf{T}}$ in CL to any point P on the clothoid is acquired by integration of Equation B.4 from $\phi\left(0\right) = 0$ to ϕ, exploiting Equation B.2. This yields

$$_{CL}\mathbf{r}^{CL^*,P}\left(s\right) = \begin{bmatrix} _{CL}x\left(s\right) \\ _{CL}y\left(s\right) \end{bmatrix} = \begin{bmatrix} \int_0^s \cos\left(\frac{1}{2}\kappa'\bar{s}^2\right)d\bar{s} \\ \int_0^s \sin\left(\frac{1}{2}\kappa'\bar{s}^2\right)d\bar{s} \end{bmatrix}. \tag{B.5}$$

The integrals in Equation B.5 are known as Fresnels integrals and cannot be solved in closed form, [Bronstein and Semendjajew, 1991]. However, they can be approximated with an arbitrary accuracy by using series expansions for the sine and cosine functions in the integrands, see for example [Wang et al., 2001], which yields

$$_{CL}\mathbf{r}^{CL^*,P}\left(s\right) = \begin{bmatrix} _{CL}x\left(s\right) \\ _{CL}y\left(s\right) \end{bmatrix} = \begin{bmatrix} \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n)!(4n+1)} s^{4n+1} \left(\frac{1}{2}\kappa'\right)^{2n} \\ \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)!(4n+3)} s^{4n+3} \left(\frac{1}{2}\kappa'\right)^{2n+1} \end{bmatrix}. \tag{B.6}$$

Since a clothoid street segment does not necessarily start with $\kappa = 0$, the reference frames

$\underrightarrow{\uparrow\mathrm{CL}}$ and $\underrightarrow{\uparrow\hat{R}^k}$ are generally not the same, as shown in Figure B.1. The curvature of a clothoid segment $\hat{R}^k$ is therefore given by

$$\kappa_k\left(\hat{s}_k\right) \quad = \quad \kappa_{k,0} + \kappa_k' \hat{s}_k. \tag{B.7}$$

Comparing Equations 2.32 and B.7 yields the relation between the clothoid arc length s and the segment arc length $\hat{s}_k$

$$s\left(\hat{s}_k\right) \quad = \quad \hat{s}_k + \frac{\kappa_{k,0}}{\kappa_k'}. \tag{B.8}$$

Knowing the starting point $\hat{R}^{k*}$ of the clothoid segment relative to the clothoid reference frame $\underrightarrow{\uparrow\mathrm{CL}}$, the segment's centerline is first computed in $\underrightarrow{\uparrow\mathrm{CL}}$ and than transformed into the segment fixed reference frame $\underrightarrow{\uparrow\hat{R}^k}$ by

$${}_{\hat{R}^k}\mathbf{r}^{\hat{R}^{k*},\tilde{R}^*}\left(\hat{s}_k\right) \quad = \quad \mathbf{C}^{CL,\hat{R}^k}\underbrace{\left({}_{CL}\mathbf{r}^{CL^*,\tilde{R}^*} - {}_{CL}\mathbf{r}^{CL^*,\hat{R}^{k*}}\right)}_{{}_{CL}\mathbf{r}^{\hat{R}^{k*},\tilde{R}^*}\left(\hat{s}_k\right)}. \tag{B.9}$$

Both ${}_{CL}\mathbf{r}^{CL^*,\hat{R}^{k*}}$ and ${}_{CL}\mathbf{r}^{CL^*,\tilde{R}^*}$ can be found by the evaluation of Equation B.6 for $s = \frac{\kappa_{k,0}}{\kappa_k'}$ and $s = \hat{s}_k + \frac{\kappa_{k,0}}{\kappa_k'}$, respectively, see Equation B.8. This results in

$${}_{CL}\mathbf{r}^{\hat{R}^{k*},\tilde{R}^*}\left(\hat{s}_k\right) \quad = \quad \begin{bmatrix} \sum_{n=0}^{\infty}\frac{(-1)^n}{(2n)!(4n+1)}\left(\left(\hat{s}_k + \frac{\kappa_{k,0}}{\kappa_k'}\right)^{4n+1} - \left(\frac{\kappa_{k,0}}{\kappa_k'}\right)^{4n+1}\right)\left(\frac{1}{2}\kappa_k'\right)^{2n} \\ \sum_{n=0}^{\infty}\frac{(-1)^n}{(2n+1)!(4n+3)}\left(\left(\hat{s}_k + \frac{\kappa_{k,0}}{\kappa_k'}\right)^{4n+3} - \left(\frac{\kappa_{k,0}}{\kappa_k'}\right)^{4n+3}\right)\left(\frac{1}{2}\kappa_k'\right)^{2n+1} \end{bmatrix} \tag{B.10}$$

$\mathbf{C}^{CL,\hat{R}^k}$ denotes the transformation matrix between $\underrightarrow{\uparrow\mathrm{CL}}$ and $\underrightarrow{\uparrow\hat{R}^k}$

$$\mathbf{C}^{CL,\hat{R}^k} \quad = \quad \begin{bmatrix} \cos{}^{CL}\psi^{\hat{R}^k} & \sin{}^{CL}\psi^{\hat{R}^k} \\ -\sin{}^{CL}\psi^{\hat{R}^k} & \cos{}^{CL}\psi^{\hat{R}^k} \end{bmatrix}, \tag{B.11}$$

where ${}^{CL}\psi^{\hat{R}^k}$ represents the rotation angle between the two reference frames. It can be found by exploiting the clothoid properties given in Equations B.2 and B.3 for $s = \frac{\kappa_{k,0}}{\kappa_k'}$, see

Equation B.8, to be

$$ {}^{CL}\psi^{\hat{R}^k} = \frac{\kappa_{k,0}^2}{2\kappa_k'}. \tag{B.12} $$

The orientation at the end of the clothoid segment is given by

$$ {}^{CL}\psi^{\hat{R}^{k+1}} = \frac{\left(\kappa_{k,0} + {\kappa_k}'\hat{l}_k\right)^2}{2\kappa_k'}. \tag{B.13} $$

The orientation of the road centerline of a clothoid segment with regards to the segment fixed reference frame can be found by integrating Equation B.7 to

$$ {}^{\hat{R}^k}\psi^{\tilde{R}}\left(\hat{s}_k\right) = \kappa_{k,0}\hat{s}_k + \frac{1}{2}\kappa_k'\hat{s}_k^2. \tag{B.14} $$

B.2. Relation of Arclengths of Paths with Lateral Offsets

For straight segments $\tilde{s}^{O_j}$ and s^{O_j} are identical. For any curved segments the differential changes in arc length of the centerline $\tilde{s}^{O_j}$ and the offset path s^{O_j} are governed by

$$ d\tilde{s}^{O_j} = \left(\frac{1}{\kappa_k\left(\tilde{s}^{O_j}\right)}\right) d\phi \qquad \text{and} \tag{B.15} $$

$$ ds^{O_j} = \left(\frac{1}{\kappa_k\left(\tilde{s}^{O_j}\right)} + y_{\text{off}}^{O_j}\right) d\phi, \tag{B.16} $$

where $\frac{1}{\kappa\left(\tilde{s}^{O_j}\right)}$ represents the instantaneous radius of curvature of the road centerline $\mathcal{R}_c$. Plugging Equation B.15 into Equation B.16 yields the relation

$$ ds^{O_j} = \left(1 + \kappa_k\left(\tilde{s}^{O_j}\right)\ y_{\text{off}}^{O_j}\right) d\tilde{s}^{O_j}. \tag{B.17} $$

For circular segments the curvature is constant, $\kappa_k\left(\tilde{s}^{O_j}\right) = \kappa_k = const.$, while for clothoid segments the curvature varies linearly with the arc length, $\kappa_k\left(\tilde{s}^{O_j}\right) = \kappa_{k,0} + \kappa_k'\tilde{s}^{O_j}$, compare

Equation B.7. Using these expressions for the curvature, Equation B.17 can be integrated from $\tilde{s}_0^{O_j} = 0$ to $\tilde{s}^{O_j}$ and $s_0^{O_j} = 0$ to s^{O_j}.

Supervised Student Works

S. Gausemeier. Planung einer kollisionsfreien Ausweichbahn mit Balkenmodellen für ein Fahrerassistenzsystem zur Unfallvermeidung. Studienarbeit, Universität Paderborn, 2006.

B. Arronte Arroyuelos. Sensor fusion of differential gps and inertial measurement unit to measure state of test vehicle. Bachelor thesis, Universität Paderborn, 2006.

A. Bloch. Adaptive Filterung und Skalierung dynamischer Verkehrsobjekte in der potentialfeldbasierten Bahnplanung. Studienarbeit, Universität Paderborn, 2007.

S. Chu. Ansatz für die Bahnplanung in Fahrerassistenzsystemen auf stark gekrümmten Strassen mit Kreissegmenten. Studienarbeit, Universität Paderborn, 2007.

D. Hess. Implementation of an automotive vehicle guidance system for the hni-minirobot. Studienarbeit, Universität Paderborn, 2007.

B. Hesse. Entwurf einer unterlagerten Lenkregelung zur automatischen Fahrzeugführung. Studienarbeit, Universität Paderborn, 2007.

S. Kahl. Entwicklung von Bahnplanungsverfahren unter Berücksichtigung der Fahrzeugdynamik. Diplomarbeit, Universität Paderborn, 2007.

P. Klötzer. Ansätze zur Trajektorienplanung mit elastischen Bändern zur kombinierten Längs- und Querführung. Studienarbeit, Universität Paderborn, 2007a.

P. Klötzer. Erweiterung eines Trajektorienplanungsverfahrens für beliebige Strassenverläufe. Studienarbeit, Universität Paderborn, 2007b.

N. Knolle. Verhaltensbasierte Ansätze in der Bahnplanung. Studienarbeit, Universität Paderborn, 2007.

J. Moreno Schneider. Implementation in C++ of kalman filter for sensor fusion of differential GPS and inertial measurement unit. Bachelor thesis, Universität Paderborn, 2007.

J. Nawroth. Konstruktiver Entwurf einer Lenkaktorik für das autonome Fahren mit einem Kraftfahrzeug. Studienarbeit, Universität Paderborn, 2007.

P. Niggemann. Reglerentwurf zur kombinierten Längs- und Querführung von Kraftfahrzeugen. Diplomarbeit, Universität Paderborn, 2007.

W. Schellenberg. Nichtlineare Stabilitätsuntersuchungen zum Entwurf von Fahrzeug Führungsreglern. Diplomarbeit, Universität Paderborn, 2007.

F. Stüve. Untersuchung numerischer Verfahren für die Berechnung der Gleichgewichtslage beim Verfahren der elastischen Bänder. Bachelor thesis, TU Ilmenau, 2007.

N. Arada Perez. Characterization of inertial measurement unit. Bachelor thesis, Universität Paderborn, 2008.

T. Gaukstern. Adaptiver Auflösung für die Trajektorienplanung mit elastischen Bändern. Studienarbeit, Universität Paderborn, 2008.

R. Giebl. Strassenmodellierung und Bahnplanung für Fahrerassistenzsysteme bei stark gekrümmten Strassen mit Hilfe krummliniger Koordinaten. Diplomarbeit, Universität Paderborn, 2008.

J. Hertkorn. Entwurf einer Lenkwinkelregelung für die autonome Querführung von Kraftfahrzeugen. Diplomarbeit, Universität Paderborn, 2008.

B. Hesse. Entwicklung eines mechatronischen Systems zur automatischen Längsführung von Kraftfahrzeugen. Diplomarbeit, Universität Paderborn, 2008.

N. Knolle. Fuzzy-basierte Verhaltensentscheidungen für eine prädiktive Bahnplanung mit elastischen Bändern. Diplomarbeit, Universität Paderborn, 2008.

A. Korsmeier. Detektierung der Drehmomente bei der haptischen Fahrer-Fahrzeug-Interaktion. Studienarbeit, Universität Paderborn, 2008.

J. Neuhaus. Kombinierte Quer- und Längsplanung mit elastischen Bändern in Weg-Zeit-Koordinaten. Diplomarbeit, Universität Paderborn, 2008.

D. Roeser. Integrierte Längs- und Querführung eines Versuchsfahrzeugs entlang vorgegebener Trajektorien mit dem Verfahren der nichtlinearen Entkopplung. Studienarbeit, Universität Paderborn, 2008.

D. Wesemeyer. Reglerauslegung für ein autonomes Fahrzeug. Diplomarbeit, Universität Paderborn, 2008.

M. Büchsenschütz. Entwurf und Implementierung der Längsregelung eine autonomen Versuchsfahrzeuges. Diplomarbeit, Universität Paderborn, 2009.

A. Elsafadi. Modellbasierte systemidentifikation eines versuchsfahrzeuges anhand von differential-gps und inertialsensoren. Master's thesis, Universität Paderborn, 2009.

D. Hess. Evaluation of motion planning strategies for autonomous cars. Diplomarbeit, Universität Paderborn, 2009.

A. Lauhoff. Entwurf eines erweiterten Kalmanfilters zur Positions- und Orientierungsbestimmung von Kraftfahrzeugen. Studienarbeit, Universität Paderborn, 2009.

K. Othmani. Entwurf eines erweiterten Kalmanfilters zur Positions- und Orientierungsbestimmung von Kraftfahrzeugen. Master's thesis, Universität Paderborn, 2009.

J. Welzel. Konzeption einer kooperativen Manöverplanung für Kraftfahrzeuge und deren Implementierung in einer Testumgebung mit Minirobotern. Diplomarbeit, Universität Paderborn, 2009.

A. Albarran. Vehicle setup for test drives. Internship report, L-LAB, Universität Paderborn, 2007.

M. Bachmann. Electrical installations in test vehicle. Internship report, L-LAB, Universität Paderborn, 2008.

T. Florez. A* introduction and implementation. Internship report, Universität Paderborn, 2008.

S. Jayaprakash. Integration of different path planning algorithms for ADAS to include vehicle dynamics with newton damping and a potential optimization algorithm. Internship report, Universität Paderborn, 2008.

P. Mishra. Roadbuilder in matlab. Internship report, Universität Paderborn, 2008.

T. Stahl. Implementation of throttle flap control. Internship report, L-LAB, Universität Paderborn, 2008.

E. Wangui Gituku. Improvement of curvature and feedforward steering angle calculations. Internship report, Universität Paderborn, 2008.

J. Stefani. Integration of trajectory planning algorithm matlab code as DLL in C++ framework. Internship report, L-LAB, Universität Paderborn, 2009.

My Related Publications

C. Löper, T. Hesse, T. Brandt, and T. Sattel. Developing driver assistance systems using carsim as a reference model. In *Proceedings of 1st European CarSim User Conference*, September 2005.

T. Hesse, H. Shadeed, M. Götz, S. Strauss, and J. Wallaschek. Concept of an active front-lighting driver assistance system. In *Proceedings of 4th IFAC-Symposium on Mechatronic Systems*, Heidelberg, 2006.

T. Hesse, T. Sattel, J. L. Du, and U. Witkowski. Application of automotive motion planning algorithms on non-holonomic mobile minirobots. In *4th International Symposium on Autonomous Minirobots, AMiRE*, Buenos Aires, Argentina, October 2007.

T. Hesse and T. Sattel. Path-planning with virtual beams. In *Proceedings of the American Control Conference (ACC)*, pages 3904–3905, New York City, USA, July 2007a. ISBN 1-4244-0989-6.

T. Hesse and T. Sattel. An approach to integrate vehicle dynamics in motion planning for advanced driver assistance systems. In *2007 IEEE Intelligent Vehicles Symposium*, pages 1240–1245, 2007b.

T. Sattel, T. Hesse, and C. Sondermann-Wölke. Experimental validation of a potential field based assistance system for collision avoidance. In *National Conference on Control of Vehicles and Motors (AUTOREG)*, Wiesloch, Germany, February 2008a.

T. Sattel, T. Hesse, and C. Sondermann-Wölke. Automatic evasion in dynamic environments for collision avoidance driver assistance systems (in german). In *Conference on Active Safety by Driver Assistance*. TÜV Süd, Munich, Germany 2008b.

T. Hesse, C. Sondermann-Wölke, and T. Sattel. Towards an artificial potential field framework for automotive collision avoidance assistance systems. In *FISITA Automotive World Congress*, 2008.

C. Sondermann-Wölke, T. Hesse, T. Sattel, and T. Hemsel. Menschliche unzuverlässigkeit als grundlage für den entwurf von kollisionsvermeidungssystemen. In *24. Tagung Technische Zuverlässigkeit (TTZ) - Entwicklung und Betrieb zuverlässiger Produkte*, Leonberg, 2009.

T. Sattel, T. Hesse, C. Sondermann-Wölke, and T. Hüfner. Potenzialfeldmethoden für fahrerassistenzsysteme zur automatisierten fahrzeugführung. *Mechatronik Mobil*, 1, 2009.

T. Hesse, D. Hess, and T. Sattel. Motion planning for passenger vehicles - force field trajectory optimization for automated driving. In *IASTED International Conference Robotics and Applications (RA)*, pages 284–294, 2010.

T. Hesse, E. Engström, E. Johansson, G. Varalda, M. Brockmann, A. Rambaldini, N. Fricke, F. Flemisch, F. Köster, and L. Kanstrup. Towards user-centred development of integrated information, warning, and intervention strategies for multiple adas in the eu project interactive. In *14th International Conference on Human-Computer Interaction (HCI)*, 2011.

F. Flemisch, M. Heesen, T. Hesse, J. Kelsch, A. Schieben, and J. Beller. Towards a dynamic balance between humans and automation: authority, ability, responsibility and control in shared and cooperative control situations. *Cognition, Technology & Work*, pages 1–16, 2011.

Bibliography

Althoff, M., Stursberg, O. and Buss, M. [2008], Stochastic reachable sets of interacting traffic participants, *in* ‘Proceedings of IEEE Intelligent Vehicles Symposium’, pp. 1086–1092.

Ameling, C. [2002], *Steigerung der aktiven Sicherheit von Kraftfahrzeugen durch ein Kollisionsvermeidungssystem*, number 510 *in* ‘12’, VDI-Verlag, Düsseldorf.

Andreasson, J. and Bünte, T. [2006], ‘Global chassis control based on inverse vehicle dynamics models’, *Vehicle System Dynamics* **44**, 321–328.

Argyros, I. [2005], *Newton Methods*, Nova Science Pub Inc.

Arkin, R. [1989], Towards the unification of navigational planning and reactive control, *in* ‘Mobile Robot Laboratory Publications’, Georgia Institute of Technology.

Arronte Arroyuelos, B. [2006], Sensor fusion of differential gps and inertial measurement unit to measure state of test vehicle, Bachelor thesis, Universität Paderborn.

Aurenhammer, F. and Klein, R. [1996], *Voronoi diagrams*, Karl-Franzens-Universität Graz & Technische Universität Graz.

Avery, M. and Weekes, A. [2008], Volvo city safety–collision avoidance technology and its potential to reduce whiplash injuries, *in* ‘Conference of Neck Injuries in Road Traffic and Prevention Strategies, Munich, Germany’.

Bacha, A., Bauman, C., Faruque, R., Fleming, M., Terwelp, C., Reinholtz, C., Hong, D., Wicks, A., Alberi, T., Anderson, D. et al. [2008], ‘Odin: Team victortango’s entry in the darpa urban challenge’, *Journal of Field Robotics* **25**(8).

Bakker, E., Pacejka, H. and Lidner, L. [1989], ‘A new tire model with an application in vehicle dynamics studies’, *SAE transactions: Journal of Passenger Cars* **98**(6), 101–113.

Barraquand, J. and Latombe, J. [1990], A monte-carlo algorithm for path planning with many degrees of freedom, *in* ‘1990 IEEE International Conference on Robotics and Automation, 1990. Proceedings.’, pp. 1712–1717.

Bellino, M., de Meneses, Y. L., Ryser, P. and Jacot, J. [2004], Lane detection algorithm for an onboard camera, *in* ‘SPIE proceedings of the first Workshop on Photonics in the Automobile’, Vol. 30.

Bender, E., Darms, M., Schorn, M., Stählin, U., Isermann, R., Winner, H. and Landau, K. [2007], ‘Das Antikollisionssystem PRORETA–auf dem Weg zum unfallvermeidenden Fahrzeug’, *Automobiltechnische Zeitung ATZ* **2**(04).

Bernotat, R. [1970], ‘Operation functions in vehicle control’, *Ergonomics* **13**(3), 353–377.

Bertolazzi, E., Biral, F. and Lio, M. D. [2007], ‘Real-time motion planning for multibody systems’, *Multibody System Dynamics* **17**(2), 119–139.

Betts, J. [2010], *Practical methods for optimal control and estimation using nonlinear programming*, Society for Industrial & Applied Mathematics.

Bonnans, J. and Lemaréchal, C. [2006], *Numerical optimization: theoretical and practical aspects*, 2 edn, Springer-Verlag New York Inc.

Borenstein, J. and Koren, Y. [1991], ‘The vector field histogram–fast obstacle avoidance for mobile robots’, *IEEE Journal of Robotics and Automation* **7**(3), 278–288.

Börner, M. [2003], Adaptive Querdynamikmodelle für Personenkraftfahrzeuge - Fahrzustandserkennungund Sensorfehlertolleranz, PhD thesis, Technische Universität Darmstadt.

Brandt, T. [2007], A predictive potential field concept for shared vehicle guidance, PhD thesis, Universität Paderborn.

Brock, O. and Khatib, O. [1999], High-speed navigation using the global dynamic window approach, *in* ‘Proceedings of the IEEE International Conference on Robotics and Automation’, Vol. 1, pp. 341–346.

Bronstein, I. and Semendjajew, K. [1991], *Taschenbuch der Mathematik*, 25 edn, Verlag Harri Deutsch, Frankfurt/Main.

Broyden, C. [1967], 'Quasi-Newton methods and their application to function minimisation', *Mathematics of Computation* **21**(99), 368–381.

Bruce, J. and Veloso, M. [2003], 'Real-time randomized path planning for robot navigation', *Lecture Notes in Computer Science* pp. 288–295.

Brüdigam, C. [1994], Intelligente Fahrmanöver sehender autonomer Fahrzeuge in autobahnähnlicher Umgebung, PhD thesis, Universität der Bundeswehr München.

Büchsenschütz, M. [2009], Entwurf und Implementierung der Längsregelung eine autonomen Versuchsfahrzeuges, Diplomarbeit, Universität Paderborn.

Burckhardt, M. [1993], *Fahrwerktechnik: Radschlupf-Regelsysteme*, Vogel-Verlag, Germany.

Cameron, S. [1997], 'A comparison of two fast algorithms for computing the distance between convex polyhedra', *Robotics and Automation, IEEE Transactions on* **13**(6), 915–920.

Canny, J. [1988], *The complexity of robot motion planning*, MIT press.

Castillo, O., Trujillo, L. and Melin, P. [2006], 'Multiple objective genetic algorithms for path-planning optimization in autonomous mobile robots', *Soft Computing-A Fusion of Foundations, Methodologies and Applications* **11**(3), 269–279.

Chang, D. and Marsden, J. [2003], 'Gyroscopic forces and collision avoidance with convex obstacles', *New Trends in Nonlinear Dynamics and Control and their Applications* pp. 145–160.

Chang, D., Shadden, S., Marsden, J. and Olfati-Saber, R. [2003], Collision avoidance for multiple agent systems, *in* 'Proceedings of the IEEE Conference on Decision and Control', Citeseer.

Chee, W., Tomizuka, M., California, of Transportation Studies, I., for Advanced Transit, P., (Calif, H., of Transportation, D. and of Mechanical Engineering, D. [1994], *Vehicle Lane Change Maneuver in Automated Highway Systems*, California PATH Program, Institute of Transportation Studies, University of California at Berkeley.

Choset, H. [1997], Incremental construction of the generalized voronoi diagram, the generalized voronoi graph, and the hierarchical generalized voronoi graph, *in* '1st CGC Workshop on Computation Geometry', Vol. 2.

Choset, H., Hutchinson, S., Lynch, K., Kantor, G., Burgard, W., Kavraki, L. and Thrun, S. [2005], *Principles of Robot Motion: Theory, Algorithms, and Implementations*, The MIT Press.

Cohen, A. [1981], 'Stepsize analysis for descent methods', *Journal of Optimization Theory and Application* **2**, 187–205.

Connolly, C. and Grupen, R. [1993], 'The applications of harmonic functions to robotics', *Journal of Robotic Systems* **10**, 931–946.

Cramer, H., Scheunert, U. and Wanielik, G. [2004], A new approach for tracking lanes by fusing image measurements with map data, *in* 'Proceedings of the IEEE Intelligent Vehicles Symposium', pp. 607–612.

Cremean, L. and Murray, R. [2006], Model-based estimation of off-highway road geometry using single-axis LADAR and inertial sensing, *in* 'Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)', pp. 1661–1666.

Curtis, S., Tamstorf, R. and Manocha, D. [2008], Fast collision detection for deformable models using representative-triangles, *in* 'Proceedings of the 2008 symposium on Interactive 3D graphics and games', ACM New York, NY, USA, pp. 61–69.

Dagli, I., Brost, M. and Breuel, G. [2003], 'Action recognition and prediction for driver assistance systems using dynamic belief networks', *Lecture Notes in Computer Science* **2592/2003**, 179–194.

Dagli, I. and Reichardt, D. [2002], Motivation-based approach to behavior prediction, *in* 'Intelligent Vehicle Symposium', Vol. 38.

Dennis Jr, J. and More, J. [1977], 'Quasi-Newton methods, motivation and theory', *Siam Review* **19**(1), 46–89.

Destatis [2011*a*], 'Road traffic accidents, casualty number', Internet Webpage. Last updated on 06 July 2011.
URL: *http://www.destatis.de/jetspeed/portal/cms/Sites/destatis/Internet/EN/Content/Statistics/TimeSeries/LongTermSeries/Transport/Content100/lrvkr002a, templateId=renderPrint.psml*

Destatis [2011*b*], *Unfallentwicklung auf deutschen Strassen 2010*, Statistisches Bundesamt, Wiesbaden.

Deuflhard, P. [2006], *Newton Methods for Nonlinear Problems*, Springer.

Dickmanns, E. [2005], *Fahrerassistenzsysteme mit maschineller Wahrnehmung*, Springer, chapter Vision: Von Assistenz zum autonomen Fahren, pp. 203–238.

Dickmanns, E. and Mysliwetz, B. [1992], 'Recursive 3-D road and relative ego-state recognition', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **14**(2), 199–213.

Dobkin, D. and Kirkpatrick, D. [1985], 'A linear algorithm for determining the separation of convex polyhedra', *Journal of algorithms* **6**(3), 381–392.

Donges, E. [1982], 'Aspekte der aktiven Sicherheit bei der Führung von Personenkraftwagen', *Automobil-Industrie* **2**, 183–190.

Donges, E. [2009], *Fahrerverhaltensmodelle*, Springer, chapter 2.

Dubins, L. [1957], 'On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents', *American Journal of Mathematics* **79**(3), 497–516.

Dugoff, H., Fancher, P. S. and Segel, L. [1970], 'An analysis of tire traction properties and their influence on vehicle dynamic performance', *SAE papers* **700377**, 1219–1243.

Ehmann, S. and Lin, M. [2001], Accurate and fast proximity queries between polyhedra using convex surface decomposition, *in* 'Computer Graphics Forum', pp. 500–510.

Etemad, A. [2010], interactive - accident avoidance by active intervention for intelligent vehicles, Technical report, Ford Research & Advanced Engineering Europe.
URL: *www.interactIVe-ip.eu*

Feder, H. and Slotine, J. [1997], Real-time path planning using harmonic potentials in dynamic environments, *in* 'IEEE International Conference on Robotics and Automation', pp. 874–881.

Ferguson, D., Howard, T. and Likhachev, M. [2008], 'Motion planning in urban environments', *Journal of Field Robotics* **25**, 939–960.

Ferguson, D., Kalra, N. and Stentz, A. [2006], 'Replanning with RRTs', *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* pp. 1243–1248.

Ferguson, D. and Stentz, A. [2006], Anytime RRTs, *in* 'Proceedings of the IEEE International Conference on Intelligent Robots and Systems (IROS)'.

Ferguson, D. and Stentz, A. [2007], Anytime, dynamic planning in high-dimensional search spaces, *in* 'Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)'.

FGSV [1999], *Richtlinien für die Anlage von Strassen, Teil: Linienführung (RAS-L)*, berichtigter nachdruck edn, Kirschbaum. Forschungsgesellschaft für Strassen und Verkehrswesen (FGSV).

Fiorini, P. and Shiller, Z. [1998], 'Motion planning in dynamic environments using velocity obstacles', *The International Journal of Robotics Research* **17**(7), 760–772.

Föllinger, O. [1993], *Methoden der Regelungs- und Automatisierungstechnik: Nichtlineare Regelungen II*, 7 edn, Oldenbourg.

Föllinger, O. [2008], *Regelungstechnik. Einführung in die Methoden und ihre Anwendung*, 10 edn, Hüthig Jehle Rehm GmbH, Heidelberg, Germany.

Fox, D., Burgard, W. and Thrun, S. [1997], 'The dynamic window approach to collision avoidance', *IEEE Robotics & Automation Magazine* **4**(1), 23–33.

Fraichard, T. [1991], Smooth trajectory planning for a car in a structured world, *in* 'Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)', Vol. 1, pp. 318–323.

Fraichard, T. and Ahuactzin, J. [2001], Smooth path planning for cars, *in* 'Proceedings of the IEEE International Conference on Robotics and Automation', Vol. 4, pp. 3722–3727.

Fraichard, T. and Delsart, V. [2009], 'Navigating dynamic environments with trajectory deformation', *Journal of Computing and Information Technology* **17**(1), 27–36.

Freund, E. and Mayr, R. [1997], 'Nonlinear path control in automated vehicle guidance', *IEEE Transactions on Control and Automation* **13**(1), 49–60.

Furtwängler, R., Hanebeck, U. D. and Schmidt, G. [1998], Dynamische Bänder zur Bewegungsplanung für mobile Manipulatoren, *in* 'Autonome Mobile Systeme 1998, 14. Fachgespräch', Springer-Verlag, London, UK, pp. 164–171.

Gehrig, S. and Stein, F. [2007], 'Collision avoidance for vehicle-following systems', *IEEE Transactions on Intelligent Transportation Systems* **8**(2), 233–244.

GeneSys Elektronic GmbH [2008], 'User manual adma'.

GeneSys Elektronik GmbH [2008], 'Technical documentation adma'.

Geraerts, R. and Overmars, M. [2006], 'Sampling and node adding in probabilistic roadmap planners', *Robotics and Autonomous Systems* **54**(2), 165–173.

Gerdes, J. C. [1996], Decoupled Design of Robust Controllers for Nonlinear Systems - As Motivatied by and Applied to Coordinated Throttle and Brake Control for Automated Highways, PhD thesis, University of California at Berkeley.

Germann, S. [1997], Modellbildung und modellgestützte Regelung der Fahrzeuglängsdynamik, PhD thesis, Technische Universität Darmstadt, Düsseldorf.

Goldberg, D. [1989], *Genetic Algorithms in Search and Optimization*, Addison-Wesley.

Gottschalk, S. [2000], Collision queries using oriented bounding boxes, PhD thesis, University of North Carolina at Chapel Hill.

Gottschalk, S., Lin, M. and Manocha, D. [1996], OBBTree: A hierarchical structure for rapid interference detection, *in* ‘Proceedings of the 23rd annual conference on Computer graphics and interactive techniques’, ACM New York, NY, USA, pp. 171–180.

Goyat, Y., Chateau, T. and Trassoudaine, L. [2009], ‘Tracking of vehicle trajectory by combining a camera and a laser rangefinder’, *Machine Vision and Applications* **online**.

Guibas, L. J., Hsu, D. and Zhang, L. [1999], H-walk: hierarchical distance computation for moving convex bodies, *in* ‘SCG ’99: Proceedings of the fifteenth annual symposium on Computational geometry’, ACM, New York, NY, USA, pp. 265–273.

Gutsche, R., Laloni, C. and Wahl, F. [1993], Fine motion planning in self overlapping driving channels and multiple mobile vehicle coordination, *in* ‘IEEE/RSJ International Conference on Intelligent Robots and Systems’, Vol. 3, pp. 2249–2256.

Halfmann, C. [2003], *Adaptive Modelle für die Kraftfahrzeugdynamik*, Springer.

Hansen, E. and Zhou, R. [2007], ‘Anytime heuristic search’, *Journal of Artificial Intelligence Research* **28**, 267–297.

Hart, P., Nilsson, N. and Raphael, B. [1968], ‘A formal basis for the heuristic determination of minimum cost paths’, *IEEE transactions on Systems Science and Cybernetics* **4**(2), 100–107.

HBM [2008], ‘Datenblatt digiCLIP DF30CAN’.

Hennessey, M., Shankwitz, C. and Donath, M. [1995], Sensor-based virtual bumpers for collision avoidance: configuration issues, *in* ‘Proceedings of the SPIE’, Vol. 2592, pp. 48–59.

Hertkorn, J. [2008], Entwurf einer Lenkwinkelregelung für die autonome Querführung von Kraftfahrzeugen, Diplomarbeit, Universität Paderborn.

Hess, D. [2009], Evaluation of motion planning strategies for autonomous cars, Diplomarbeit, Universität Paderborn.

Hesse, B. [2007], Entwurf einer unterlagerten Lenkregelung zur automatischen Fahrzeugführung, Studienarbeit, Universität Paderborn.

Hesse, B. [2008], Entwicklung eines mechatronischen Systems zur automatischen Längsführung von Kraftfahrzeugen, Diplomarbeit, Universität Paderborn.

Hesse, T., Hess, D. and Sattel, T. [2010], Motion planning for passenger vehicles - force field trajectory optimization for automated driving, *in* 'Proceedings of the IASTED International Conference Robotics and Applications RA', Cambridge, Massachusetts, pp. 284–294.

Hesse, T. and Sattel, T. [2007], An approach to integrate vehicle dynamics in motion planning for advanced driver assistance systems, *in* '2007 IEEE Intelligent Vehicles Symposium', pp. 1240–1245.

Hilgert, J., Hirsch, K., Bertram, T. and Hiller, M. [2003], Emergency path planning for autonomous vehicles using elastic band theory, *in* 'Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)', Vol. 2, pp. 1390–1395 vol.2.

Hirsch, K., Hilgert, J., Lalo, W., Schramm, D. and Hiller, M. [2005], Optimization of emergency trajectories for autonomous vehicles with respect to linear vehicle dynamics, *in* 'Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)', pp. 528–533.

Hoeger, R., Amditis, A., Kunert, M., Hoess, A., Flemisch, F., Krueger, H., Bartels, A., Beutner, A. and Pagle, K. [2008], Highly automated vehicles for intelligent transport: HAVEit approach, *in* 'ITS World Congress. New York. USA'.

Hoff III, K., Culver, T., Keyser, J., Lin, M. and Manocha, D. [2000], Interactive motion planning using hardware-accelerated computationof generalized voronoi diagrams, *in* 'Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)', Vol. 3.

Hu, W., Tan, T., Wang, L. and Maybank, S. [2004], 'A survey on visual surveillance of object motion and behaviors', *IEEE Transactions on Systems, Man, and Cybernetics* **34**(3), 334–352.

Hubbard, P. [1996], ‘Approximating polyhedra with spheres for time-critical collision detection’, *ACM Transactions on Graphics (TOG)* **15**(3), 179–210.

Hwang, Y. K. and Ahuja, N. [1992], ‘Gross motion planning – a survey’, *ACM Comput. Surv.* **24**(3), 219–291.

Isermann, R., Schorn, M. and Stählin, U. [2008], ‘Anticollision system proreta with automatic braking and steering’, *Vehicle System Dynamics* **46**, 683–694.

Ishida, S., Gayko, J. and Tochigi, J. [2004], Development, evaluation and introduction of a lane keeping assistance system, *in* ‘Proceedings of the IEEE Intelligent Vehicles Symposium’, pp. 943–944.

Ismail, A., Sheta, A. and Al-Weshah, M. [2008], ‘A mobile robot path planning using genetic algorithm in static environment’, *Journal of Computer Science* **4**(4), 341–344.

ISO2631 [1997], ‘Mechanical vibration and shock – evaluation of human exposure to whole-body vibration – part 1: General requirements’.

IVO GmbH & Co. KG [2008], ‘Absolute Drehgeber - modulare Bushauben, Multiturn-Drehgeber 13 Bit ST / 16 Bit MT’.

Jiang, K., Seneviratne, L. and Earles, S. [1993], Finding the 3D shortest path with visibility graph and minimumpotential energy, *in* ‘Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).’, Vol. 1.

Kahl, S. [2007], Entwicklung von Bahnplanungsverfahren unter Berücksichtigung der Fahrzeugdynamik, Diplomarbeit, Universität Paderborn.

Kammel, S., Ziegler, J., Pitzer, B., Werling, M., Gindele, T., Jagzent, D., Schroder, J., Thuy, M., Goebl, M., von Hundelshausen, F. et al. [2008], ‘Team annieway’s autonomous system for the darpa 2007 urban challenge’, *Journal of Field Robotics* **25**, 615–639.

Kaplan, E. and Hegarty, C. [2006], *Understanding GPS: Principles and Applications*, second edn, Artech House, Boston, MA.

Kavraki, L. and LaValle, S. [2008], ‘Motion planning, chapter 5’.

Kavraki, L., Svestka, P., Latombe, J.-C. and Overmars, M. [1996], ‘Probabilistic roadmaps for path planning in high-dimensional configuration spaces’, *IEEE transactions on Robotics and Automation* **12**(4), 566–580.

Kelley, C. [2003], *Solving nonlinear equations with Newton’s method*, Society for Industrial Mathematics.

Kelly, A. and Stentz, A. [1998], An approach to rough terrain autonomous mobility, *in* 'International Conference on Mobile Planetary Robots', pp. 129–198.

Khalil, H. [2002], *Nonlinear Systems*, 3 edn, Prentice Hall, Upper Saddle River, NJ.

Khatib, M., Jaouni, H., Chatila, R. and Laumond, J. [1997], Dynamic path modification for car-like nonholonomic mobile robots, *in* 'Proceedings of the IEEE International Conference on Robotics and Automation', Vol. 4, pp. 2920–2925.

Khatib, O. [1986], 'Real-time obstacle avoidance for manipulators and mobile robots', *The International Journal of Robotics Research* **5**(1), 90–98.

Khosla, D. [2002], Accurate estimation of forward path geometry using two-clothoid road model, *in* 'Proceedings of the IEEE Intelligent Vehicle Symposium', Vol. 1, pp. 154–159 vol.1.

Kirchgässner, D. and Tröster, F. [2006], 'Autonomes fahren an der hochschule heilbronn. das acc (automotive competence center) projekt fahrautomat', *horizonte* **28**, 10–19.

Klein, L. [2001], *Sensor technologies and data requirements for ITS*, Artech House.

Klosowski, J., Held, M., Mitchell, J., Sowizral, H. and Zikan, K. [1998], 'Efficient collision detection using bounding volume hierarchies of k-DOPs', *IEEE transactions on Visualization and Computer Graphics* **4**(1), 21–36.

Klötzer, P. [2007*a*], Ansätze zur Trajektorienplanung mit elastischen Bändern zur kombinierten Längs- und Querführung, Studienarbeit, Universität Paderborn.

Klötzer, P. [2007*b*], Erweiterung eines Trajektorienplanungsverfahrens für beliebige Strassenverläufe, Studienarbeit, Universität Paderborn.

Kluge, K. and Thorpe, C. [1989], Explicit models for robot road following, *in* 'Proceedings of the IEEE International Conference on Robotics and Automation', pp. 1148–1154 vol.2.

Kluge, K. and Thorpe, C. [1995], 'The yarf system for vision-based road following', *Mathematical and Computer Modelling* **22**(4-7), 213–233.

Knolle, N. [2007], Verhaltensbasierte Ansätze in der Bahnplanung, Studienarbeit, Universität Paderborn.

Knolle, N. [2008], Fuzzy-basierte Verhaltensentscheidungen für eine prädiktive Bahnplanung mit elastischen Bändern, Diplomarbeit, Universität Paderborn.

Ko, N. Y. and Simmons, R. G. [1998], The lane-curvature method for local obstacle avoidance, *in* ‘Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems’, Vol. 3.

Kodaka, K. and Gayko, J. [2004], Intelligent systems for active and passive safety-collision mitigation brake system, *in* ‘Proceedings of the ATA EL Conference’, Parma, Italy.

Koditschek, D. E. and Rimon, E. [1990], ‘Robot navigation functions on manifolds with boundary’, *Advances in Applied Mathematics* **11**(4), 412–442.

Koller, D., Nagel, H. and Danilidis, K. [1993], ‘Model-based object tracking in monocular image sequences of road traffic scenes’, *International Journal of Computer Vision* **10**, 257–281.

König, S. and Likhachev, M. [2002], D* lite, *in* ‘Proceedings of the National Conference on Artificial Intelligence’, Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, pp. 476–483.

König, S. and Likhachev, M. [2006], ‘A new principle for incremental heuristic search: Theoretical results’, *Proceedings of the International Conference on Automated Planning and Scheduling* pp. 410–413.

König, S., Likhachev, M. and Furcy, D. [2004], ‘Lifelong planning A*’, *Artificial Intelligence* **155**(1), 93–146.

Kopischke, S. [2000], Entwicklung einer Notbremsfunktion mit Rapid Prototyping Methoden, PhD thesis, TU Darmstadt.

Koryakovskiy, I., Hoai, N. and Lee, K. [2009], A genetic algorithm with local map for path planning in dynamic environments, *in* ‘Proceedings of the 11th Annual conference on Genetic and evolutionary computation’, pp. 1859–1860.

Kreucher, C. and Lakshmanan, S. [1999], ‘LANA: a lane extraction algorithm that uses frequency domainfeatures’, *IEEE Transactions on Robotics and Automation* **15**(2), 343–350.

Krogh, B. [1983], Feedback obstacle avoidance control, *in* ‘Proceedings of the 21st Allerton Conference’, Vol. 111, pp. 325–334.

Krogh, B. [1984], A generalized potential field approach to obstacle avoidance control, *in* ‘International Robotics Research Conference’.

Kruse, E., Gutsche, R. and Wahl, F. [1997], Acquisition of statistical motion patterns in dynamic environments and their application to mobile robot motion planning, *in* 'Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems', Vol. 2, pp. 712–717 vol.2.

Kurniawati, H. and Fraichard, T. [2007], From path to trajectory deformation, *in* 'Proceedings of the IEEE-RSJ International Conference on Intelligent Robots and Systems, San Diego, CA (US)', Citeseer.

Kwon, W. and Lee, S. [2002], 'Performance evaluation of decision making strategies for an embedded lane departure warning system', *Journal of Robotic Systems* **19**(10), 499–509.

Lages, U. [2001], Collision avoidance system for fixed obstacles-fuzzy controllernetwork for robot driving of an autonomous vehicle, *in* 'Proceedings of the IEEE Intelligent Transportation Systems', pp. 489–491.

Lammen, B. [1993], Automatische Kollisionvermeidung für Kraftfahrzeuge, PhD thesis, Universität Dortmund.

Large, F., Laugier, C. and Shiller, Z. [2005], 'Navigation among moving obstacles using the nlvo: Principles and applications to intelligent vehicles', *Autonomous Robots* **19**(2), 159–171.

Large, F., Sekhavat, S., Shiller, Z. and Laugier, C. [2002], Towards real-time global motion planning in a dynamic environment using the nlvo concept, *in* 'Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems', Vol. 1.

Latombe, J. C. [1991], *Robot Motion Planning*, Springer.

Lauhoff, A. [2009], Entwurf eines erweiterten Kalmanfilters zur Positions- und Orientierungsbestimmung von Kraftfahrzeugen, Studienarbeit, Universität Paderborn.

LaValle, S., Branicky, M. and Lindemann, S. [2004], 'On the relationship between classical grid search and probabilistic roadmaps', *The International Journal of Robotics Research* **23**(7-8), 673–692.

LaValle, S. and Kuffner Jr, J. [2001], 'Randomized kinodynamic planning', *The International Journal of Robotics Research* **20**(5), 378–400.

LaValle, S. M. [2006], *Planning algorithms*, Cambridge University Press.
URL: *planning.cs.uiuc.edu*

Lee, J. [2002], 'A machine vision system for lane-departure detection', *Computer Vision Image Understanding* **86**(1), 52–78.

Likhachev, M., Gordon, G. and Thrun, S. [2004], ‘ARA*: Anytime A* with provable bounds on sub-optimality’, *Advances in Neural Information Processing Systems* **16**.

Lin, M. and Canny, J. [1991], A fast algorithm for incremental distance calculation, *in* ‘Proceedings of the IEEE International Conference on Robotics and Automation’, Vol. 2, pp. 1008–1014.

Lin, M. and Gottschalk, S. [1998], Collision detection between geometric models: A survey, *in* ‘Proceedings of the IMA Conference on Mathematics of Surfaces’.

Lindemann, S. and LaValle, S. [2003], Incremental low-discrepancy lattice methods for motion planning, *in* ‘Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)’, Vol. 3.

Liu, A. and Pentland, A. [2002], Towards real-time recognition of driver intentions, *in* ‘Proceedings of the IEEE Conference on Intelligent Transportation System (ITSC)’, pp. 236–241.

Löper, C., Kelsch, J. and Flemisch, F. [2008], ‘Kooperative, manöverbasierte automation und arbitrierung als bausteine für hochautomatisiertes fahren’, *AAET 2008-Automatisierungssysteme, Assistenzsysteme und eingebettete Systeme für Transportmittel* pp. 215–237.

Lozano-Perez, T. and Wesley, M. [1979], ‘An algorithm for planning collision-free paths among polyhedral obstacles’, *Communications of the ACM* **22**(10), 560–570.

Lunze, J. [2008], *Regelungstechnik 2*, 5 edn, Springer.

Ma, B., Lakshmanan, S. and Hero, A. O. [2000], ‘Simultaneous detection of lane and pavement boundaries using model-based multisensor fusion’, *IEEE Transactions on Intelligent Transportation Systems* **1**(5), 135–147.

Mauciuca, D. B. [1997], Nonlinear Robust and Adaptive Control with Application to Brake for Automated Highway Systems, PhD thesis, University of California at Berkeley.

Maurer, M. [2000], Flexible Automatisierung von Straßenfahrzeugen mit Rechnersehen, PhD thesis, Armed Forces University, Munich.

Mayr, R. [1991], Verfahren zur Bahnfolgeregelung für ein automatisch geführtes Fahrzeug, PhD thesis, University of Dortmund.

McMurray, E. and Sniers, R. C. [1963], ‘Vehicle speed warning and cruise control system’, US-Patent.

Milam, M. [2003], Real-time optimal trajectory generation for constrained dynamical systems, PhD thesis, California Institute of Technology.

Mildner, F. [2004], Untersuchungen zur Erkennung und Vermeidung von Unfällen für Kraftfahrzeuge, PhD thesis, Helmut Schmidt University, University of the Federal Armed Forces, Hamburg, Germany.

Minguez, J. and Montano, L. [2000], Nearness diagram navigation (ND): a new real time collision avoidance approach, *in* 'Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)', Vol. 3, pp. 2094–2100.

Minguez, J. and Montano, L. [2004], 'Nearness diagram (ND) navigation: collision avoidance in troublesome scenarios', *IEEE Transactions on Robotics and Automation* **20**(1), 45–59.

Minguez, J., Montano, L., Simeon, T. and Alami, R. [2001], Global nearness diagram navigation (GND), *in* 'Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)', Vol. 1, pp. 33–39.

Mirtich, B. [1998], 'V-clip: Fast and robust polyhedral collision detection', *ACM Transactions on Graphics* **17**(3), 177–208.

Mitchell, M. [1998], *An introduction to genetic algorithms*, The MIT press.

Mitschke, M. and Wallentowitz, H. [2004], *Dynamik der Kraftfahrzeuge*, Springer.

Montemerlo, M., Becker, J., Bhat, S., Dahlkamp, H., Dolgov, D., Ettinger, S., Haehnel, D., Hilden, T., Hoffmann, G., Huhnke, B. et al. [2008], 'Junior: The stanford entry in the urban challenge', *Journal of Field Robotics* **25**(9), 569–597.

Moreno Schneider, J. [2007], Implementation in C++ of kalman filter for sensor fusion of differential GPS and inertial measurement unit, Bachelor thesis, Universität Paderborn.

Nash, A., Daniel, K., Koenig, S. and Felner, A. [2007], 'Theta*: Any-angle path planning on grids', *Proceedings of the National Conference on Artificial Intelligence* **22**(2), 1177.

Nawroth, J. [2007], Konstruktiver Entwurf einer Lenkaktorik für das autonome Fahren mit einem Kraftfahrzeug, Studienarbeit, Universität Paderborn.

Nedevschi, S., Schmidt, R., Graf, T., Danescu, R., Frentiu, D., Marita, T., Oniga, D. and Pocol, C. [2004], 3d lane detection system based on stereovision, *in* 'Proceedings of the IEEE Intelligent Transportation Systems Conference, Washington DC', pp. 161–166.

Neuhaus, J. [2008], Kombinierte Quer- und Längsplanung mit elastischen Bändern in Weg-Zeit-Koordinaten, Diplomarbeit, Universität Paderborn.

Niggemann, P. [2007], Reglerentwurf zur kombinierten Längs- und Querführung von Kraftfahrzeugen, Diplomarbeit, Universität Paderborn.

Novatel [2006*a*], 'OEMV family user manual - volume 1, installation and operation'.

Novatel [2006*b*], 'OEMV family user manual - volume 2, command and log reference'.

Novatel [2006*c*], 'ProPak-V3 Manual'.

Ögren, P. and Leonard, N. [2005], 'A convergent dynamic window approach to obstacle avoidance', *IEEE Transactions on Robotics* **21**(2), 188–195.

Oliver, N. and Pentland, A. P. [2000], Driver behavior recognition and prediction in a SmartCar, *in* 'Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series', Vol. 4023, pp. 280–290.

Ortega, J. and Rheinboldt, W. [1970], *Iterative Solution of Nonlinear Equations in Several Variables*, 1st ed. edn, Academic Press, New York.

Othmani, K. [2009], Entwurf eines erweiterten Kalmanfilters zur Positions- und Orientierungsbestimmung von Kraftfahrzeugen, Master's thesis, Universität Paderborn.

Özgüner, Ü., Stiller, C. and Redmill, K. [2007], 'Systems for safety and autonomous behaviour in cars: The darpa grand challenge experience', *Proceedings of the IEEE* **95**(2), 397–412.

Pacejka, H. [2006], *Tyre and vehicle dynamics*, Vol. 642, 2 edn, Butterworth-Heinemann.

Pacejka, H. and Bakker, E. [1992], 'The magic formula tyre model', *Vehicle System Dynamics* **21**(S1), 1–18.

Pearl, J. [1985], *Heuristics - Intelligent Search Strategies for Computer Problem Solving*, Addison-Wesley Reading, Mass.

Pellkofer, M. [2003], Verhaltensentscheidung für autonome Fahrzeuge mit Blickrichtungssteuerung, PhD thesis, Universität der Bundeswehr München.

Pham, H., Tomizuka, M. and Hedrick, J. K. [1997], Integrated maneuvering control for automated highway systems based on a magnetic reference/sensing system, Technical report, UC Berkeley: California Partners for Advanced Transit and Highways (PATH)., Retrieved from: http://escholarship.org/uc/item/7hk255bn.

Piao, J. and McDonald, M. [2008], ‘Advanced driver assistance systems from autonomous to cooperative approach’, *Transport Reviews* **28**(5), 659–684.

Pivtoraiko, M. and Kelly, A. [2005*a*], ‘Constrained motion planning in discrete state spaces’, *Field and Service Robotics* **25**, 269–280.

Pivtoraiko, M. and Kelly, A. [2005*b*], Efficient constrained path planning via search in state lattices, *in* ‘International Symposium on Artificial Intelligence, Robotics, and Automation in Space’, Vol. 8.

Pivtoraiko, M. and Kelly, A. [2005*c*], ‘Generating near minimal spanning control sets for constrained motion planning in discrete state spaces’, *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* pp. 3231–3237.

Plöchl, M. and Edelmann, J. [2007], ‘Driver models in automobile dynamics application’, *Vehicle System Dynamics* **45**(7 & 8), 699–741.

Polychronopoulos, A., Amditis, A., Floudas, N. and Lind, H. [2004], ‘Integrated object and road border tracking using 77 ghz automotive radars’, *Radar, Sonar and Navigation, IEE Proceedings -* **151**(6), 375–381.

Presse, B. [2002], ‘Technische daten bmw 5er touring 540i’.

Quinlan, S. [1994], Efficient distance computation between non-convex objects, *in* ‘International Conference on Robotics and Automation’, pp. 3324–3329 vol.4.

Quinlan, S. and Khatib, O. [1993*a*], Elastic bands: Connecting path planning and control, *in* ‘Proceedings of the IEEE International Conference on Robotics and Automation’, Vol. 2, pp. 802–802.

Quinlan, S. and Khatib, O. [1993*b*], *Experimental Robotics 2*, Springer-Verlag, Berlin Heidelberg, chapter Towards real-time execution of motion tasks, pp. 241–241.

Rasmussen, J. [1983], ‘Skills, rules, and knowledge; signals, signs, and symbols, and other distinctions in human performance models’, *IEEE transactions on systems, man, and cybernetics* **13**(3), 257–266.

Reichardt, D. [1996], Kontinuierliche Verhaltenssteuerung eines autonomen Fahrzeugs in dynamischer Umgebung, PhD thesis, Technical University of Kaiserslautern, Germany.

Rezaie, J., Moshiri, B., Araabi, B. and Asadian, A. [2007], Gps/ins integration using nonlinear blending filters, *in* ‘SICE, 2007 Annual Conference’, pp. 1674–1680.

Rheinboldt, W. [1998], *Methods for Solving Systems of Nonlinear Equations*, number 70 *in* 'CBMS-NSF Regional Conference Series in Applied Mathematics', 2 edn, Society for Industrial and Applied Mathematics, Philadelphia.

Rimon, E. and Koditschek, D. [1992], 'Exact robot navigation using artificial potential functions', *Robotics and Automation, IEEE Transactions on* **8**(5), 501–518.

Roeser, D. [2008], Integrierte Längs- und Querführung eines Versuchsfahrzeugs entlang vorgegebener Trajektorien mit dem Verfahren der nichtlinearen Entkopplung, Studienarbeit, Universität Paderborn.

Rohrmüller, F., Althoff, M., Wollherr, D. and Buss, M. [2008], Probabilistic mapping of dynamic obstacles using markov chains for replanning in dynamic environments, *in* 'Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)', pp. 2504–2510.

Roos, H.-G. and Schwetlick, H. [1999], *Numerische Mathematik*, Mathematik für Ingenieure und Naturwissenschaftler, 1 edn, B. G. Teubner, Stuttgart.

Sattel, T. and Brandt, T. [2008], 'From robotics to automotive: Lane-keeping and collision avoidance based on elastic bands', *Vehicle System Dynamics* **46**(7), 597–619.

Schellenberg, W. [2007], Nichtlineare Stabilitätsuntersuchungen zum Entwurf von Fahrzeug Führungsreglern, Diplomarbeit, Universität Paderborn.

Schorn, M. [2007], Quer- und Längsregelung eines Personenkraftwagens für ein Fahrerassistenzsystem zur Unfallvermeidung, PhD thesis, Technische Universität Darmstadt.

Schorn, M., Stählin, U., Khanafer, A. and Isermann, R. [2006], Nonlinear trajectory following control for automatic steering of a collision avoiding vehicle, *in* 'American Control Conference, 2006'.

Schröder, J., Müller, U. and Dillmann, R. [2006], Smart roadster project: Setting up drive-by-wire or how to remote-control your car, *in* 'Proceedings of the 9th International Conference on Intelligent Autonomous Systems', pp. 383–390.

Schubert, R., Mattern, N. and Wanielik, G. [2008], An evaluation of nonlinear filtering algorithms for integrating gnss and inertial measurements, *in* 'Proceedings of the IEEE/ION Position, Location and Navigation Symposium', pp. 25–29.

Schuette, H. and Waeltermann, P. [2005], Hardware-in-the-loop testing of vehicle dynamics controllers- a technical survey, *in* 'SAE World Congress Detroit', Society of Automotive Engineers, 400 Commonwealth Dr, Warrendale, PA, 15096, USA.

Schwetlick, H. and Kretzschmar, H. [1991], *Numerische Verfahren für Naturwissenschaftler und Ingenieure*, 1 edn, Fachbuchverlag Leipzig.

Shiller, Z., Large, F. and Sekhavat, S. [2001], Motion planning in dynamic environments: Obstacles moving along arbitrary trajectories, *in* 'Proceedings of the IEEE International Conference on Robotics and Automation', Vol. 4, pp. 3716–3721.

Siegwart, R. and Nourbakhsh, I. [2004], *Introduction to autonomous mobile robots*, The MIT Press.

Simmons, R. [1996], The curvature-velocity method for local obstacle avoidance, *in* 'Proceedings of the IEEE International Conference on Robotics and Automation', Vol. 4.

Simon, A. [2003], *Führung eines autonomen Strassenfahrzeugs mit redundanten Sensorsystemen*, VDI Verlag.

Simon, A. and Becker, J. [1999], Vehicle guidance for an autonomous vehicle, *in* 'Proceedings of the IEEE/IEEJ/JSAI International Conference on Intelligent Transportation Systems', pp. 429–434.

Smith, C., McGehee, D. and Healey, A. [1978], 'The prediction of passenger riding comfort from acceleration data', *Journal of Dynamic Systems, Measurement, and Control* **100**(1), 34–41.

Söhnitz, I. [2001], Querregelung eines autonomen Strassenfahrzeugs, PhD thesis, Technische Universität Braunschweig.

Solea, R. and Nunes, U. [2006], Trajectory planning with velocity planner for fully-automated passenger vehicles, *in* 'Proceedings of the IEEE Intelligent Transportation Systems Conference (ITSC)'.

Southall, B. and Taylor, C. J. [2001], Stochastic road shape estimation, *in* 'Proceedings of the International Conference on Computer Vision, Vancouver, BC, Canada', pp. 205–212.

Sparbert, J., Dietmayer, K. and Streller, D. [2001], Lane detection and street type classification using laser range images, *in* 'Intelligent Transportation Systems', pp. 454–459.

Stefani, J. [2009], Integration of trajectory planning algorithm matlab code as DLL in C++ framework, Internship report, L-LAB, Universität Paderborn.

Stentz, A. [1994], Optimal and efficient path planning for partially-known environments, *in* 'Proceedings of the IEEE International Conference on Robotics and Automation', pp. 3310–3317.

Stentz, A. [1995], The focussed D* algorithm for real-time replanning, *in* 'International Joint Conference on Artificial Intelligence', Vol. 14, pp. 1652–1659.

Stiller, C., Hipp, J., Rössig, C. and Ewald, A. [2000], 'Multisensor obstacle detection and tracking', *Image and Vision Computing* **18**(5), 389 – 396.

Stüve, F. [2007], Untersuchung numerischer Verfahren für die Berechnung der Gleichgewichtslage beim Verfahren der elastischen Bänder, Bachelor thesis, TU Ilmenau.

Sugihara, K. and Smith, J. [1997], Genetic algorithms for adaptive motion planning of an autonomousmobile robot, *in* 'Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)', pp. 138–143.

Sullivan, J., Waydo, S. and Campbell, M. [2003], Using stream functions for complex behavior and path generation, *in* 'AIAA Guidance, Navigation, and Control Conference', Citeseer.

Sun, X. and Koenig, S. [2007], The fringe-saving A* search algorithm - a feasibility study, *in* 'Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)', pp. 2391–2397.

Sun, X., Yeoh, W. and Koenig, S. [2009], Dynamic fringe-saving A*, *in* 'Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)'.

Suzuki, K. and Jansson, H. [2003], 'An analysis of driver's steering behaviour during auditory or haptic warnings for the designing of lane departure warning system', *JSAE Review* **24**(1), 65–70.

Svaricek, F. [2006], 'Zero dynamics of linear and nonlinear systems: Definitions, properties and applications', *at–Automatisierungstechnik* **54**(7), 310–322.

Swaroop, D., Hedrick, J. and Choi, S. [2001], 'Direct adaptive longitudinal control of vehicle platoons', *IEEE transactions on vehicular technology* **50**(1), 150–161.

Swaroop, D. and Yoon, S. M. [1999], 'Integrated lateral and longitudinal vehicle control for an emergency lane change manoeuvre design', *International Journal of Vehicle Design* **21**(2), 161–174.

Switkes, J. and Gerdes, C. [2005], Guaranteeing lanekeeping performance with tire saturation using computed polynomial lyapunov functions, *in* 'Proceedings of IMECE, ASME International Mechanical Engineering Congress and Exposition'.

Taylor, C. J., Kosecka, J., Blasi, R. and Malik, J. [1999], 'A comparative study of vision-based lateral control strategies for autonomous highway driving', *The International Journal of Robotics Research* **18**(5), 442–453.

Teschner, M., Kimmerle, S., Heidelberger, B., Zachmann, G., Raghupathi, L., Fuhrmann, A., Cani, M., Faure, F., Magnenat-Thalmann, N., Strasser, W. et al. [2005], Collision detection for deformable objects, *in* 'Computer Graphics Forum', Vol. 24, Blackwell Publishing Ltd., pp. 61–81.

Thurner, T. et al. [1998], 'X-by-wire: Safety related fault-tolerant systems in vehicles', *XByWire* **DB-6/6-25**.

Toledo, T., Koutsopoulos, H. and Ben-Akiva, M. [2007], 'Integrated driving behavior modeling', *Transportation Research Part C* **15**(2), 96–112.

Trächtler, A. [2004], 'Integrated vehicle dynamics control using active brake, steering and suspension systems', *International journal of vehicle design* **36**(1), 1–12.

Trächtler, A. [2005], 'Integrierte Fahrdynamikregelung mit ESP, aktiver Lenkung und aktivem Fahrwerk', *at–Automatisierungstechnik* **53**(1), 11–19.

Ulrich, I. and Borenstein, J. [1998], VFH+: Reliable obstacle avoidance for fast mobile robots, *in* 'Proceedings of the IEEE international conference on robotics and automation', pp. 1572–1577.

Ulrich, I. and Borenstein, J. [2000], VFH*: Local obstacle avoidance with look-ahead verification, *in* 'Proceedings of the IEEE international conference on robotics and automation', Vol. 3, pp. 2505–2511.

Urmson, C., Anhalt, J., Bagnell, D., Baker, C., Bittner, R., Clark, M., Dolan, J., Duggins, D., Galatali, T., Geyer, C. et al. [2008], 'Autonomous driving in urban environments: Boss and the urban challenge', *Journal of Field Robotics* **25**(8), 425–466.

Urmson, C. and Simmons, R. [2003], Approaches for heuristically biasing RRT growth, *in* 'Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)', Vol. 2.

Vahidi, A. and Eskandarian, A. [2003], 'Research advances in intelligent collision avoidance and adaptive cruise control', *IEEE Transactions on Intelligent Transportation Systems* **4**(3), 143–153.

Van Den Bergen, G. [2005], 'Efficient collision detection of complex deformable models using AABB trees', *Graphics Tools: The Jgt Editors' Choice* **2**(4), 131–144.

Vasquez, D., Large, F., Fraichard, T. and Laugier, C. [2004], High-speed autonomous navigation with motion prediction for unknown moving obstacles, *in* 'Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems', Vol. 1, pp. 82–87.

Vendittelli, M., Laumond, J.-P. and Nissoux, C. [1999], 'Obstacle distance for car-like robots', *IEEE Transactions on Robotics and Automation* **15**(4), 678–691.

Wang, J., Liu, R. and Montgomery, F. [2005], 'Car-following model for motorway traffic', *Transportation Research Record: Journal of the Transportation Research Board* **1934**, 33–42.

Wang, L., Miura, K., Nakamae, E., Yamamoto, T. and Wang, T. [2001], 'An approximation approach of the clothoid curve defined in the interval [0, $\pi/2$] and its offset by free-form curves', *Computer-Aided Design* **33**(14), 1049–1058.

Weilkes, M., Bürkle, L., Rentschler, T. and Scherl, M. [2005], 'Future vehicle guidance assistance–combined longitudinal and lateral control (in german)', *at–Automatisierungstechnik* **53**(1/2005), 4–10.

Weisen, R. [2003], Gekoppelte Quer- und Längsregelung eines Personenkraftwagens im fahrphysikalischen Grenzbereich, PhD thesis, Helmut Schmidt University, University of the Federal Armed Forces, Hamburg, Germany.

Welzel, J. [2009], Konzeption einer kooperativen Manöverplanung für Kraftfahrzeuge und deren Implementierung in einer Testumgebung mit Minirobotern, Diplomarbeit, Universität Paderborn.

Wendel, J. [2007], *Integrierte Navigationssysteme: Sensordaten, GPS und Inertiale Navigation*, Oldenbourg Wissenschaftsverlag.

Werling, M. [2010], *Ein neues Konzept für die Trajektoriengenerierung und -stabilisierung in zeitkritischen Verkehrsszenarien*, Vol. 34 of *Schriftenreihe des Institus für angewandte Informatik / Automatisierungstechnik am Karlsruher Institut für Technologie*, KIT Scientific Publishing.

Werling, M., Gindele, T., Jagszent, D. and Groll, L. [2008], A robust algorithm for handling moving traffic in urban scenarios, *in* 'Proceedings of the IEEE Intelligent Vehicles Symposium', pp. 1108–1112.

Wesemeyer, D. [2008], Reglerauslegung für ein autonomes Fahrzeug, Diplomarbeit, Universität Paderborn.

Wijesoma, W., Kodagoda, K. and Balasuriya, A. [2004], 'Road-boundary detection and tracking using ladar sensing', *IEEE Transactions on Robotics and Automation* **20**(3), 456–464.

Winner, H., Witte, S., Uhler, W. and Lichtenberg, B. [1996], 'Adaptive cruise control system aspects and development trends', *SAE transactions* **105**(6), 1412–1421.

Xiao, L. and Gao, F. [2010], 'A comprehensive review of the development of adaptive cruise control systems', *Vehicle System Dynamics* **48**(10), 1167–1192.

Zambou, N. [2005], Lagrange-basierte und modellgestützte Regelungsstrategie für die automatische Fahrzeugführung im Konvoi, PhD thesis, RWTH Aachen.

Ziegler, J. and Werling, M. [2008], Navigating car-like robots in unstructured environments using an obstacle sensitive cost function, *in* 'Proceedings of the IEEE Intelligent Vehicles Symposium', pp. 787–791.

Zong, C., Wang, C., Yang, D. and Yang, H. [2009], Driving intention identification and maneuvering behavior prediction of drivers on cornering, *in* 'Proceedings of the International Conference on Mechatronics and Automation (ICMA)', pp. 4055–4060.

Zucker, M., Kuffner, J. and Branicky, M. [2007], Multipartite RRTs for rapid replanning in dynamic environments, *in* 'Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)', pp. 1603–1609.

Erklärung

(gemäß Anlage 1 der Siebten Änderung der Promotionsordnung der TU Ilmenau - Allgemeine Bestimmungen)

Ich versichere, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen direkt oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet.

Bei der Auswahl und Auswertung folgenden Materials haben mir die nachstehend aufgeführten Personen in der jeweils beschriebenen Weise unentgeltlich geholfen:

- Die Arbeit wurde durch zahlreiche studentische Arbeiten unterstützt, die von mir betreut wurden. Eine vollständige Liste findet sich am Ende der Arbeit. Die inhaltlich relevanten Arbeiten sind zusätzlich im Text entsprechend referenziert.
- Daniel Hess und Thorsten Hüfner halfen bei der Vorbereitung und Durchführung der letzten Fahrversuche an der TU Ilmenau.

Weitere Personen waren an der inhaltlich-materiellen Erstellung der vorliegenden Arbeit nicht beteiligt. Insbesondere habe ich hierfür nicht die entgeltliche Hilfe von Vermittlungs- bzw. Beratungsdiensten (Promotionsberater oder anderer Personen) in Anspruch genommen. Niemand hat von mir unmittelbar oder mittelbar geldwerte Leistungen für Arbeiten erhalten, die im Zusammenhang mit dem Inhalte der vorgelegten Dissertation stehen.

Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form einer Prüfungsbehörde vorgelegt.

Ich bin darauf hingewiesen worden, dass die Unrichtigkeit der vorstehenden Erklärung als Täuschungsversuch bewertet wird und gemäß § 7 Abs. 10 der Promotionsordnung den Abbruch des Promotionsverfahrens zur Folge hat.

Braunschweig, den 08.03.2012