



# Interaction-Aware Probabilistic Behavior Prediction of Traffic Participants in Urban Environments

Jens Claudius Schulz

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

**Doktor der Naturwissenschaften (Dr. rer. nat.)**

genehmigten Dissertation.

**Vorsitzender:**

Prof. Dr. Nassir Navab

**Prüfende der Dissertation:**

1. Prof. Dr.-Ing. Darius Burschka
2. Prof. Anca Dragan,  
University of California, Berkeley

Die Dissertation wurde am 03.08.2020 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 02.02.2021 angenommen.



# Abstract

Anticipating how a traffic situation will evolve in the future is essential for making reasonable and foresighted decisions as a driver. The aim of this thesis is to develop a method for interaction-aware and probabilistic behavior prediction of traffic participants, enabling autonomous vehicles to forecast how surrounding agents are going to act in order to better plan their own actions. The prediction of an agent’s future behavior is inherently *uncertain* and *multi-modal*, as humans might act differently in the same or in similar situations. Those actions are generally based on internal *intentions* (such as turning right at the next intersection), introducing the need to estimate these latent variables over time for improving long-term prediction performance. Furthermore, the behavior of drivers strongly depends on the *situational context*, i.e., what they observe in their surroundings such as the road geometry, traffic signs, and other agents. As the future situational context also depends on what other agents are going to do, a *mutual influence* between multiple agents arises. Due to this interdependency of their future trajectories, there is a *combinatorial nature* to the problem of predicting the behavior of multiple interacting agents (e.g., if A will turn left, B will have to brake). Neglecting this combinatorial aspect potentially results in inconsistent prediction hypotheses—a shortcoming this thesis addresses. When an autonomous vehicle tries to predict the scene for deriving its own behavior, it should additionally account for its *own influence* on the surrounding agents.

In contrast to most existing works that only address a subset of the aforementioned aspects, this thesis presents an approach that accounts for all of them in combination. We model the development of a complete traffic situation in a dynamic Bayesian network (DBN) that represents the context-dependent behavior of all agents in a probabilistic manner. We find the main reasons for the multi-modality in prediction are given by the agents’ desired routes and mutual collision avoidance constraints. Thus, we define two types of discrete driver intentions that are based on the concept of trajectory homotopy, allowing to subdivide the prediction problem into single modes. Conditioning the agents’ continuous actions on these intentions reduces complexity of the behavior models, enables the utilization of more efficient inference methods, and allows to infer the intention probabilities over time. By training a neural network-based behavior model and embedding it into the DBN, we demonstrate how modern deep learning approaches can be combined with classical Bayesian inference methods. Our evaluation shows that the deep learning model is capable of picking up subtleties such as drivers cutting curves and context-dependent prediction uncertainty which are hard to model by hand, but has more difficulty with the compounding error problem than our rule-based baseline model. By including all agents in the state space and iteratively predicting them into the future, we are able to account for the combinatorial interdependencies between multiple agents.

## *Abstract*

Deriving and predicting all possible combinations of intentions of all agents in a scene can however quickly become computationally prohibitive for more complex urban scenarios. Therefore, we investigate how to determine only the most likely combinations and to predict the corresponding trajectories without the need to estimate the probabilities of all possible combinations.

Finally, we propose two ego-vehicle motion planning approaches that leverage the presented prediction approach and enable autonomous vehicles to effectively interact with other agents. We demonstrate that accounting for how other agents are going to react to one's own future behavior allows to drive less conservatively, which is crucial for the acceptance and success of autonomous vehicles in mixed traffic.

# Zusammenfassung

Um als Verkehrsteilnehmer sinnvolle und vorausschauende Entscheidungen treffen zu können, ist es unerlässlich, zu antizipieren, wie sich eine Verkehrssituation in der Zukunft entwickeln wird. Ziel dieser Arbeit ist es, eine Methode zur interaktionsbewussten und probabilistischen Verhaltensprädiktion von Verkehrsteilnehmern zu entwickeln, welche es autonomen Fahrzeugen ermöglicht, das Verhalten von Agenten in deren Umgebung vorherzusagen um damit das eigene Handeln besser planen zu können. Die Vorhersage des zukünftigen Verhaltens von Verkehrsteilnehmern ist von Natur aus *unsicher* und *multimodal*, da sich Menschen in der gleichen oder in ähnlichen Situationen oft unterschiedlich verhalten. Da deren Verhaltensweisen von internen *Intentionen* abhängen (wie z. B. an der nächsten Kreuzung rechts abzubiegen), sollten diese latenten Variablen über der Zeit geschätzt werden um die Langzeit-Prädiktion zu verbessern. Darüber hinaus ist das Verhalten von Fahrern äußerst *situationsabhängig* und wird durch das beeinflusst, was sie in ihrer Umgebung wahrnehmen, wie beispielsweise die Straßengeometrie, Verkehrsschilder und andere Verkehrsteilnehmer. Da eine in der Zukunft liegende Situation auch davon abhängt wie sich andere Agenten verhalten werden, kann eine *gegenseitige Beeinflussung* zwischen mehreren Agenten entstehen. Aufgrund dieser wechselseitigen Abhängigkeit der zukünftigen Trajektorien mehrerer interagierender Agenten besitzt das Problems der Verhaltensprädiktion *kombinatorischen Charakter* (z. B. wenn A links abbiegen wird, muss B bremsen). Diesen kombinatorischen Aspekt zu vernachlässigen kann zu inkonsistenten Prädiktionshypothesen führen–ein Schwachpunkt, den diese Arbeit adressiert. Wenn ein autonomes Fahrzeug versucht die Entwicklung der aktuellen Szene vorherzusagen um sein eigenes Verhalten planen zu können, sollte es zudem auch seinen *eigenen Einfluss* auf die Agenten in seiner Umgebung berücksichtigen.

Im Gegensatz zu den meisten bestehenden Arbeiten, welche nur gewisse der oben genannten Aspekte adressieren, präsentiert diese Arbeit einen Ansatz der alle Aspekte in Kombination berücksichtigt. Wir modellieren die Entwicklung einer gesamten Verkehrssituation in einem dynamischen Bayes'schen Netzwerk (DBN), welches das kontextabhängige Verhalten aller Agenten auf probabilistische Weise repräsentiert. Wir beobachten, dass die Hauptgründe für die Multimodalität der Prädiktion durch die von den Agenten geplanten Routen und durch Bedingungen der gegenseitigen Kollisionsvermeidung gegeben sind. Aus diesem Grund definieren wir zwei Arten von diskreten Fahrerintentionen, welche auf dem Konzept der Trajektorien-Homotopie beruhen und es erlauben das Prädiktionsproblem in einzelne Modi aufzuteilen. Das kontinuierliche Verhalten der Agenten wird als abhängig von deren Intentionen modelliert. Dadurch reduziert sich zum einen die Komplexität der Verhaltensmodelle, zum anderen ermöglicht es den Einsatz effizienterer Inferenzmethoden und die Bestimmung der Intentionswahrscheinlichkeiten über der Zeit. Wir zeigen wie moderne Deep Learning Methoden mit klassi-

## *Zusammenfassung*

schen Bayes'schen Inferenzmethoden kombiniert werden können, indem wir ein auf einem neuronalen Netz basierendes Verhaltensmodell trainieren und in das DBN einbetten. Aus unserer Evaluation wird ersichtlich, dass das Deep Learning Modell in der Lage ist, händisch schwer zu modellierende Feinheiten des Fahrverhaltens wie Kurvenschneiden und kontextabhängige Prädiktionsunsicherheit zu lernen, jedoch mehr Schwierigkeiten mit sich zeitlich akkumulierenden Fehlern hat als das von uns entwickelte regelbasierte Verhaltensmodell. Durch die Einbeziehung aller Agenten in den Zustandsraum und durch deren iterative Prädiktion in die Zukunft sind wir in der Lage, die kombinatorischen Wechselwirkungen zwischen den einzelnen Agenten zu berücksichtigen. Alle möglichen Kombinationen von Intentionen aller Agenten in einer Szene zu bestimmen und zu präzisieren kann jedoch bei komplexeren städtische Szenarien schnell zu Rechenzeitproblemen führen. Daher analysieren wir, wie lediglich die wahrscheinlichsten Kombinationshypothesen bestimmt und die entsprechenden Trajektorien präzisiert werden können, ohne die Wahrscheinlichkeiten aller möglichen Kombinationen bestimmen zu müssen.

Schließlich schlagen wir zwei Ansätze zur Bewegungsplanung von autonomen Fahrzeugen vor, welche den vorgestellten Prädiktionsansatz nutzen um effektiv mit anderen Agenten zu interagieren. Wir zeigen, dass die Berücksichtigung von möglichen Reaktionen anderer Agenten auf das eigene zukünftige Verhalten es erlaubt, weniger konservativ zu fahren. Dies ist entscheidend für die Akzeptanz und den Erfolg von autonomen Fahrzeugen in gemischtem Verkehr mit menschlichen Verkehrsteilnehmern.

# Acknowledgements

This thesis presents the research conducted during my time as a PhD student in a cooperation of the Technical University of Munich and the BMW Group. I would like to thank everyone that joined me on my journey and contributed in making this possible.

Firstly, I would like to express my sincere gratitude to my advisor Prof. Dr.-Ing. Darius Burschka for the continuous support during this time, for his valuable advice, his patience and motivation, and not least for his humorous attitude. His guidance helped me in all the time of research, writing of publications, and writing of this thesis. I really could not have imagined having a better advisor and mentor for my PhD study. Furthermore, I would like to thank Prof. Anca Dragan for hosting my research visit at the InterACT lab of the University of California, Berkeley, and for becoming the co-examiner for this thesis. I am deeply grateful for having had the opportunity to broaden my research experience and to gain new perspectives, and I feel honored having an expert in the field of robotic systems interacting with humans review this thesis.

I would like to thank my supervisor, Julian Löchner, for his thorough guidance throughout each stage of the process, for always being optimistic, extremely involved and enthusiastic about my research, and for the great ideas and hour-long whiteboard discussions. Another big thank you goes to my second supervisor, Moritz Werling, for helping me take this path, guiding me with his strong academic experience and making me part of his team at BMW. Moreover, I thank my fellow PhD student colleagues, namely Constantin Hubmann, Sascha Steyer, Christian Pek, Branka Mirchevska, Kai Stiens, Thomas Barowski, and Tobias Rehder, for their great support, the profound discussions and foremost for the fun time working together. You always made me laugh, and together, we both enjoyed the many good times but also pushed through the difficult ones. This time would have never been the same without you. A great thank you goes to all my colleagues at BMW, especially to Daniel Althoff for always taking care and for his critical but extremely valuable advice. Furthermore, I want to express my gratitude to all students of the InterACT lab, who have been extremely welcoming and fun to work with. I would like to say thank you to my students I have had the chance to supervise, Kira Hirsenkorn, Nikolai Morin, and Artsiom Belahlazau, I really enjoyed it and you helped me look at problems from various different angles. I am thankful to the BMW Group, the Technical University of Munich, and the University of California, Berkeley, for giving me the opportunity to conduct this research.

A heartfelt thank you goes to my family, my parents and siblings, who have always been supportive throughout my entire life and helped me become the person I am today.

Finally, I want to thank Lena, for her strong support and understanding, her long-lasting patience, and, moreover, for the distraction from work in times needed.

*Jens Schulz*





# Contents

<b>Abstract</b>	<b>iii</b>
<b>Zusammenfassung</b>	<b>v</b>
<b>Acknowledgements</b>	<b>vii</b>
<b>Contents</b>	<b>ix</b>
<b>Acronyms</b>	<b>xiii</b>
<b>Notation and Symbols</b>	<b>xv</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	1
1.2 Approach Overview and Research Questions . . . . .	6
1.3 Related Work . . . . .	8
1.3.1 Intention Estimation . . . . .	8
1.3.2 Physics-based Prediction . . . . .	10
1.3.3 Reachable Set-based Prediction . . . . .	10
1.3.4 Rule-based Prediction . . . . .	10
1.3.5 Planning-based Prediction . . . . .	11
1.3.6 Machine Learning-based Prediction . . . . .	13
1.3.7 Motion Planning Accounting for Interrelation with Prediction . . . . .	22
1.3.8 Discussion . . . . .	24
1.4 Problem Statement . . . . .	27
1.5 Contributions . . . . .	28
1.6 Outline . . . . .	30
<b>2 Combined Intention Estimation and Trajectory Prediction</b>	<b>33</b>
2.1 Modeling the Development of a Traffic Situation in a Dynamic Bayesian Network . . . . .	33
2.2 Interdependencies Between Multiple Agents . . . . .	36
2.3 Measurement Model . . . . .	37
2.4 Decision Making Process of an Agent . . . . .	38
2.4.1 Route Intention . . . . .	38
2.4.2 Maneuver Intention . . . . .	40
2.4.3 Action . . . . .	41

<b>3</b>	<b>Behavior Models</b>	<b>43</b>
3.1	Rule-based Behavior Model . . . . .	44
3.1.1	Approach . . . . .	44
3.1.2	Rotate-Translate Kinematics Model . . . . .	44
3.1.3	Action Model . . . . .	45
3.2	Deep Learning-based Behavior Model . . . . .	48
3.2.1	Approach . . . . .	49
3.2.2	Kinematic Bicycle Model . . . . .	49
3.2.3	Action Model . . . . .	49
3.2.3.1	Target Generation . . . . .	51
3.2.3.2	Feature Generation . . . . .	53
3.2.3.3	Loss Function . . . . .	55
<b>4</b>	<b>Inference Methods</b>	<b>57</b>
4.1	Fundamentals . . . . .	57
4.1.1	Probability Theory . . . . .	57
4.1.2	Recursive Bayesian Estimation . . . . .	58
4.2	Sequential Monte Carlo . . . . .	60
4.2.1	Approach . . . . .	60
4.3	Multiple Model Unscented Kalman Filter . . . . .	64
4.3.1	Approach . . . . .	66
4.3.1.1	Multiple Models . . . . .	66
4.3.1.2	Single Model UKF . . . . .	68
4.4	Combinatorial Complexity . . . . .	71
4.4.1	Pruning in Full-Combinatorial DBN . . . . .	71
4.4.2	Interacting Single-Agent DBNs . . . . .	72
4.4.2.1	Filtering with Interacting Single-Agent DBNs . . . . .	72
4.4.2.2	Forward Simulation of Most Likely Combinations . . . . .	73
<b>5</b>	<b>Interrelated Ego Motion Planning and Prediction of Surrounding Agents</b>	<b>77</b>
5.1	Partially Observable Markov Decision Process . . . . .	79
5.1.1	Solution Method . . . . .	80
5.2	Multi-Agent Planning . . . . .	83
5.2.1	Maneuver Determination . . . . .	85
5.2.1.1	Multi-Agent System . . . . .	86
5.2.1.2	Homotopy Classes . . . . .	87
5.2.1.3	Formation Tree . . . . .	90
5.2.2	Trajectory Planning . . . . .	94
5.2.2.1	Kinematics Model . . . . .	94
5.2.2.2	Hard Constraints . . . . .	94
5.2.2.3	Cost Function . . . . .	97
5.2.3	Maneuver Estimation . . . . .	98
5.2.4	Decision Making . . . . .	99

<b>6 Experiments</b>	<b>101</b>
6.1 Environment Model . . . . .	101
6.2 Rule-based Behavior Models . . . . .	103
6.2.1 Intention Estimation . . . . .	104
6.2.2 Trajectory Prediction . . . . .	107
6.3 Deep Learning-based Behavior Models . . . . .	109
6.3.1 Models and Hyperparamters . . . . .	110
6.3.2 Training and Validation Data . . . . .	110
6.3.3 Comparison of Model Architectures . . . . .	111
6.3.4 Feature Importance . . . . .	113
6.3.5 Route Intention Estimation . . . . .	114
6.4 Comparison of Inference Methods . . . . .	116
6.5 Interacting Single-Agent DBNs . . . . .	120
6.6 Interactive Motion Planning . . . . .	127
6.6.1 Partially Observable Markov Decision Process . . . . .	127
6.6.2 Multi-Agent Planning . . . . .	129
<b>7 Discussion</b>	<b>133</b>
7.1 Conclusion . . . . .	133
7.2 Future Research Directions . . . . .	140
<b>Bibliography</b>	<b>145</b>



# Acronyms

ABT	adaptive belief tree
BC	behavior cloning
BN	Bayesian network
CNN	convolutional neural network
CRF	conditional random field
CTRA	constant turn rate and acceleration
CTRV	constant turn rate and velocity
CVAE	conditional variational autoencoder
DBN	dynamic Bayesian network
EKF	extended Kalman filter
EM	expectation maximization
GAIL	generative adversarial imitation learning
GAN	generative adversarial network
GP	Gaussian process
GRU	gated recurrent unit
HMM	hidden Markov model
IDM	intelligent driver model
IL	imitation learning
IMM	interacting multiple model
IOC	inverse optimal control
IRL	inverse reinforcement learning
L-BFGS	limited-memory Broyden-Fletcher-Goldfarb-Shanno
LSTM	long short-term memory
MCTS	Monte Carlo tree search
MDP	Markov decision process
MIQP	mixed-integer quadratic programming
MM-UKF	multiple model unscented Kalman filter

## *Acronyms*

MPC	model predictive control
POMDP	partially observable Markov decision process
RL	reinforcement learning
RMSE	root mean square error
RNN	recurrent neural network
SMC	sequential Monte Carlo
SVM	support vector machine
TRPO	trust region policy optimization
TTC	time-to-collision
UCB	upper confidence bound
UKF	unscented Kalman filter

# Notation and Symbols

## Notation

$(\cdot)^i$	value of agent $V^i$
$(\cdot)^{[n]}$	value of sample $n$
$(\cdot)_t$	value at time step $t$
$(\cdot)_{t_1:t_2} = (\cdot)_{t_1}, \dots, (\cdot)_{t_2}$	values at time steps $t_1$ until $t_2$
$\hat{(\cdot)}$	estimated mean value
$\bar{(\cdot)}$	arithmetic mean value
$\widehat{(\cdot)}$	approximated value

## Probability Theory

$p(\cdot)$	probability
$\mathcal{L}$	likelihood
$\mu$	mean
$\Sigma$	covariance
$\sigma$	standard deviation
$\mathcal{N}(\mu, \Sigma)$	normal distribution with mean $\mu$ and covariance $\Sigma$
$\mathcal{U}(\cdot)$	uniform distribution over a set or range $(\cdot)$
$\eta$	normalization constant

## General

$t$	discrete time index
$T$	prediction / planning horizon
$\Delta T$	time step size
map	topological and geometric map with traffic infrastructure

## Notation and Symbols

$\mathcal{V} = \{V^1, \dots, V^K\}$	set of agents to be predicted
$S = [X, R, M, A]$	scene state including all agents' variables
$\epsilon$	root weighted square error
<b>Kinematics</b>	
$(x, y)$	Cartesian position of an agent
$\theta$	orientation of an agent (yaw angle)
$v$	longitudinal velocity of an agent
$\mathbf{x} = [x, y, \theta, v]^\top$	kinematic state of an agent
$X = [\mathbf{x}^1, \dots, \mathbf{x}^K]$	kinematic states of all agents
$a$	acceleration of an agent
$\dot{\theta}$	yaw-rate of an agent
$\delta$	steering angle of an agent
$\mathbf{a} = [a, \dot{\theta}]^\top$ or $\mathbf{a} = [a, \delta]^\top$	action of an agent
$\mathbf{z} = [x_z, y_z, \theta_z, v_z]^\top$	measurement of kinematic state of an agent
$Z = [\mathbf{z}^1, \dots, \mathbf{z}^K]$	measurements of kinematic states of all agents
$\mathbf{Q}$	system noise covariance
$\mathbf{R}$	measurement noise covariance
$\mathbf{w} = [w_x, w_y, w_\theta, w_v]$	transition noise of rotate-translate kinematics model
<b>Intentions</b>	
$l_H$	metric route horizon
$r$	route intention of an agent
$\mathcal{R} = \{r_1, \dots, r_{ \mathcal{R} }\}$	set of possible route intentions of an agent
$R = [r^1, \dots, r^K]$	route intentions of all agents
$m$	maneuver intention of an agent
$\mathcal{M} = \{m_1, \dots, m_{ \mathcal{M} }\}$	set of possible maneuver intentions of an agent
$M = [m^1, \dots, m^K]$	maneuver intentions of all agents
$\iota$	generalized intention (subsuming route and maneuver)



$\mathcal{I} = \{\iota_1, \dots, \iota_{ \mathcal{I} }\}$	set of possible generalized intentions of an agent
$\zeta = [\iota^1, \dots, \iota^K]$	one possible combination of intentions of all agents
$\mathcal{C} = \{\zeta_1, \dots, \zeta_{ \mathcal{C} }\}$	set of all possible intention combinations of all agents
<b>Filtering</b>	
$S^{[i]}$	particle representing the scene state
$w^{[i]}$	weight of a particle
$\mathcal{P} = \{\langle S^{[1]}, w^{[1]} \rangle, \dots\}$	set of weighted particles representing the belief
$U$	unscented Kalman filter representing one mode of the belief
$p(U)$	probability of an unscented Kalman filter
$\mathcal{U} = \{\langle U^{[1]}, p(U^{[1]}) \rangle, \dots\}$	set of unscented Kalman filters representing the belief
$P$	state covariance
$\chi$	sigma point
$W_s, W_c$	state and covariance weights
$\alpha, \beta, \kappa$	parameters of an unscented Kalman filter
<b>Motion Planning</b>	
$V^0$	ego-vehicle
$\mathcal{V} = \{V^0, \dots, V^K\}$	set of agents including ego-vehicle
$F$	formation, a relative ordering of agents
$J$	cost function
$\mathcal{K}$	hard constraints of planning problem
$r$	reward
$\mathcal{R}$	reward function
$\gamma$	discount factor
$\pi$	policy
$b$	belief
$\mathcal{T}$	transition model
$\mathcal{O}$	observation model
$Q$	Q-value function



# 1 Introduction

Autonomous driving is considered to be one of the key technologies of future mobility. It has the potential to heavily reduce the number of accidents and the resulting human injuries and fatalities, as today, about 94 percent of accidents are caused by human driver error [122]. The shift of responsibility of safely controlling the car from the human driver to the autonomous vehicle enables people without the ability to drive themselves to use cars and allows for various activities of the passengers while driving. By more foresighted driving, more intelligent routing, and more accessible ride-sharing, it furthermore has the ability to strongly reduce traffic congestion [59].

Although research on self-driving cars has been conducted for decades [118], it is still considered one of the most complex challenges in the automotive and robotics industry worldwide. A key component for them to succeed in real, mixed traffic is the assessment of human behavior, especially the anticipation of their future motion. Without knowing what surrounding agents intend to do, it is difficult to drive in a foresighted and cooperative manner. Thus, the prediction of their motion and the estimation of their intentions is crucial for the large-scale success of self-driving vehicles.

## 1.1 Motivation

While autonomous driving has already been pioneered in the 1980s by universities such as Carnegie Mellon [136] and the Bundeswehr University Munich [155], it is still considered a challenge to integrate autonomous vehicles into real traffic. A major difficulty is given by the interaction with human drivers [29]. Autonomous vehicles need to estimate the intentions of humans and anticipate their future behavior in order to plan collision-free trajectories and drive in a foresighted, efficient and cooperative manner. As intentions cannot be measured directly and humans exhibit individual and highly complex behaviors, predictions will always be afflicted with uncertainty. On the one hand, different drivers might act differently, and on the other hand, even the same person might act differently if they encounter similar situations multiple times. Thus, two situations that are indistinguishable from an observer's point of view might result in completely different outcomes. This makes it impossible to tell in advance how the single agents in a scene are going to act exactly, i.e., to derive one accurate deterministic prediction that an autonomous vehicle could rely on for its own motion planning. A more favorable approach is to account for this *uncertainty* when modeling how a traffic situation is going to evolve. Generally, this prediction uncertainty is not spread uniformly in the space of possible predictions, but there might be multiple clusters of high probability and regions with very low probability. To understand this so-called *multi-modality*, let us look at the prediction problem on different levels of abstraction.

## 1 Introduction

High-level motion categories allow to subdivide the continuous space of possible predictions into logical statements such as “turning right”, “changing the lane”, or “stopping at a red light”. As these categories often describe what a driver intends to do, they are commonly called *intentions* in the literature. The prediction multi-modality is mainly caused by these varying high-level driver intentions, which typically depend on the prevalent traffic rules. For example, it is likely for an agent to drive on either one of two adjacent lanes, but it is rather unlikely to permanently drive in between two lanes. Introducing these high-level intentions allows to subdivide the prediction problem and improves interpretability of the prediction results. Low-level motion hypotheses then fall into any of these categories and describe the actual motion in a continuous manner, i.e., the change of the kinematic states of the agents. The estimation of high-level intentions can for example be utilized in driver assistance systems (risk assessment, collision warning, etc.) or by autonomous vehicles for high-level maneuver planning. The prediction of low-level trajectories allows for example a utilization in continuous motion planning algorithms to plan collision-free and foresighted trajectories, or to model traffic participants for microscopic traffic simulation. As driver intentions are generally persistent for some period of time and the corresponding behavior typically exhibits temporal consistency (e.g., a driver will continuously brake for a couple of seconds to reduce velocity before making a turn), it is beneficial to not only consider the current observation of the situation for estimating the intention, but also to consider the history of observations which characterizes the previous behavior of the agents. It follows that having a good estimate of the intention of a driver in turn allows for a more accurate prediction of the future continuous states. The estimation of discrete intentions and the prediction of continuous future motion are thus highly coupled problems.

Naturally, prediction uncertainty does exist in both of these levels of abstraction: It is evident that drivers may come to different high-level decisions (e.g., whether to stop for a traffic light that just turned orange or not), but also exhibit different driving styles or habits such as being more aggressive or more defensive (e.g., keep more or less distance to the preceding car). Therefore, one can distinguish between high-level intention uncertainty (describing the number and likelihood of the different clusters) and low-level action uncertainty (describing the shape and spread of the single clusters). When designing a prediction approach, uncertainty can either be neglected (one deterministic trajectory), be modeled only in the high-level intention (one deterministic trajectory per intention), only in the low-level prediction (unimodal trajectory distribution) or in both levels (multi-modal trajectory distribution). A typical prediction approach as depicted in Fig. 1.1 is to define a set of possible high-level intentions, estimate a probability distribution over them, and to predict the continuous motion given each of these intentions using a prototype trajectory or a distribution over trajectories.

Besides modeling the prediction problem probabilistically, one can also utilize the concept of so-called *reachability analysis*. Reachability analysis does not state how likely a specific prediction is but rather divides the space of predictions into two subsets, one that is possible (or reachable) and one that is impossible (or unreachable) given some constraints. A comparison of these approaches is depicted in Fig. 1.2. Deterministic approaches neglect uncertainty completely, potentially resulting in over-confident and



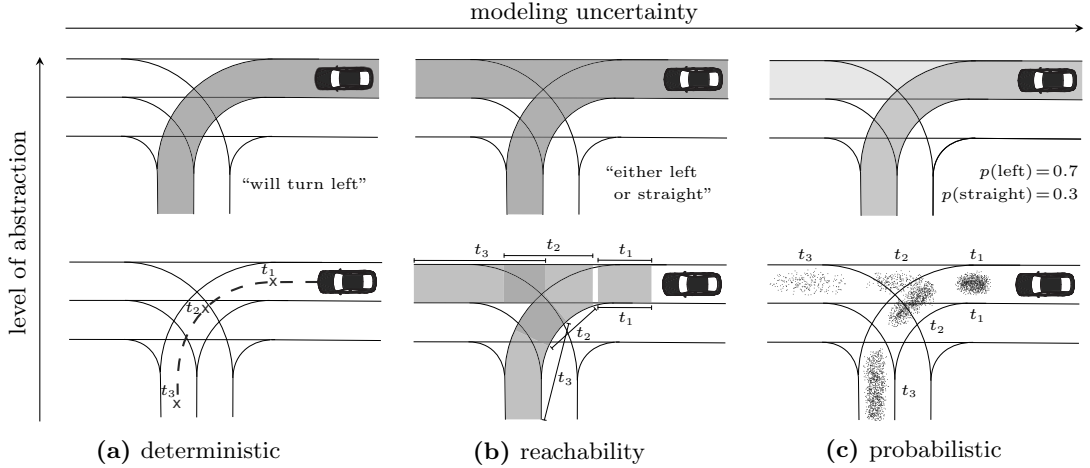
**Figure 1.1.** Possible prediction hypotheses anticipated by a Google self-driving car. Graphic taken from [144]. © Google 2015

thus unsafe behavior of the ego-vehicle. Reachability-based (also known as set-based) approaches, on the other hand, have the advantage of being able to give formal guarantees (usually given some assumptions), but in turn typically result in rather conservative (or under-confident) behavior. Neither of these approaches exploits the available information that allows to infer the probability of different possible prediction hypotheses. Probabilistic approaches, although not suited for proving formal safety, allow to exploit the shape of the existing uncertainty in order to realize more interactive, comfortable, and less conservative behavior, while still being able to account for unlikely but critical future events.

The prediction approach presented in this thesis falls into the category of probabilistic approaches that models uncertainty in both a driver's intentions as well as in their low-level actions. It is apparent that human drivers accept some risk to be able to drive more efficiently. Utilizing probabilistic approaches induces a certain risk as well, as they do not allow for formal verification. In order to still arrive at a behavior that is considered safe, we argue that one either has to distinguish between a comfort and a safety layer, where the comfort layer may be based on probabilistic approaches whereas the safety layer does not, or to investigate other methods for providing safety specifications that are rather based on statistics than on formal methods.

When defining prediction models, different *contextual information* which can give important clues for improving prediction accuracy can be utilized. There are many different methods to actually derive prediction hypotheses: Simple physics-based prediction approaches such as constant turn rate and velocity (CTRV) may be sufficient for short term

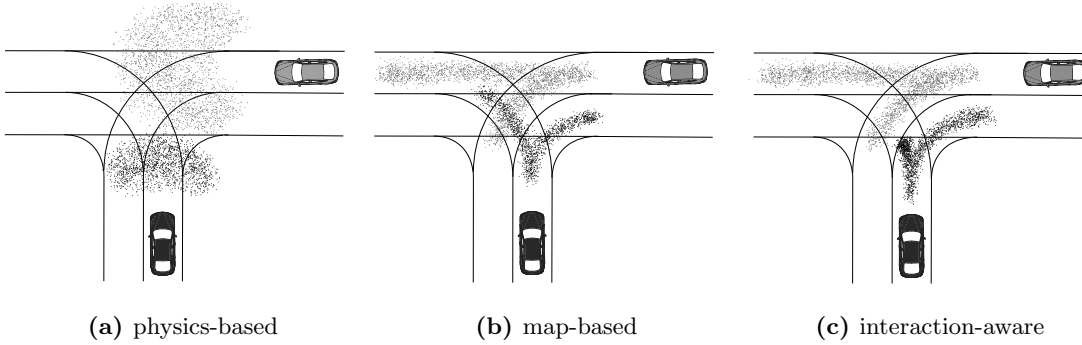
## 1 Introduction



**Figure 1.2.** Different prediction types distinguished by level of abstraction (first row: high-level intentions, second row: low-level states) and consideration of uncertainty: (a) Deterministic prediction assumes perfect knowledge and neglects uncertainty completely. (b) Reachable set prediction accounts for anything that could happen (typically given some constraints such as vehicle dynamics and staying on lanes), without distinguishing different probabilities. (c) Probabilistic prediction models the uncertainty probabilistically. Trajectories are here shown as particle distributions, but any other type of distributions such as Gaussians are also possible.

predictions and slowly changing, non-interactive situations and can be found in many of today’s driver assistance systems. However, as they do not incorporate any information about the situational context, they quickly come to a limit in complex urban scenarios, in which the behavior of a driver is dictated by the quickly changing road geometry, the traffic infrastructure (including road markings, traffic signs, and traffic lights), and its surrounding agents. Urban environments tend to be less structured than highways, often containing more options of what drivers might do in the future, such as following various lanes at intersections including U-turns, turning into driveways or stopping at the side of the road. Furthermore, they typically contain a higher variety of right of way rules such as at four-way stops, priority roads, and traffic lights. Partially blocked lanes due to parked vehicles and the existence of vulnerable road users such as pedestrians and bicyclists further complicate the prediction problem. Complex road topologies with a mixture of adjacent, crossing, merging and diverging lanes and the corresponding traffic rules create a stronger need to consider *interactions between traffic participants*. As shown in the work *unfreezing the robot* [141], humans typically engage in so-called *joint collision avoidance* in order to navigate in dense environments, resulting in a high interdependency between the motion of multiple agents.

A typical prediction problem at an intersection is depicted in Fig. 1.3, showing the different outcomes of a probabilistic prediction depending on whether it is only based on the history of kinematic states (physics-based), does also include the road geometry and topology (map-based), or even accounts for interaction between traffic participants (interaction-aware). Considering these interactions allows for a more realistic prediction,

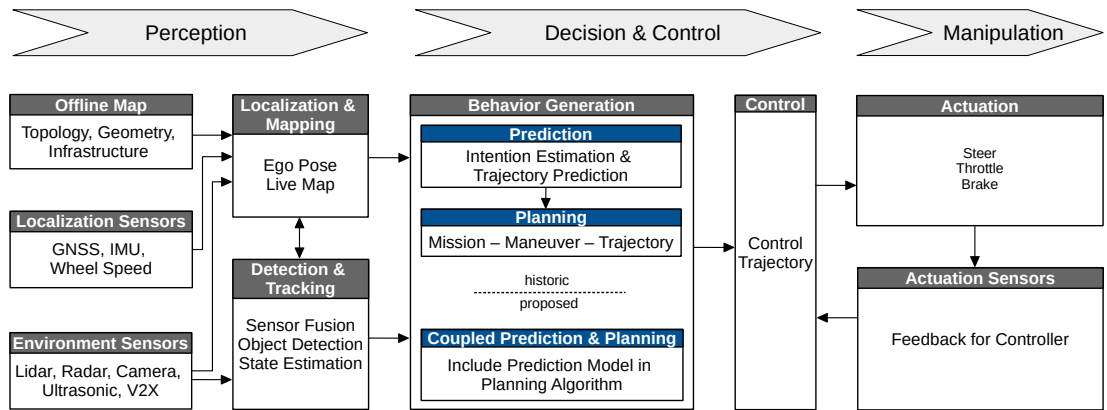


**Figure 1.3.** Probabilistic position prediction utilizing different contextual information. (a) Only the current kinematic state is used, neglecting information about the road and the presence of other agents. (b) Additional utilization of the road geometry, resulting in the predicted trajectories to stay on lanes. (c) Additional modeling of the interaction between both agents, predicting the black agent to yield to the gray agent.

as the behavior of a driver does not only depend on their intentions and the map, but also on the existence and the state of surrounding agents. This interaction does not only play a role at the current point in time, but will also affect the decisions of agents further in the future. Thus, the prediction of one agent becomes interdependent of the prediction of another agent. Let’s imagine the black agent in the depicted scene wants to turn left. Traffic rules demand it to yield to the gray agent. If the gray agent was to go straight, it will likely drive faster and thus impede the black agent only for a rather short time. If the gray agent was to turn left however, it had to slow down first, thus making the black agent having to wait even longer before turning left. The future motion of the black agent thus highly depends on the future behavior of the gray agent (and potentially also vice-versa). To be able to account for *future mutual influence* and to achieve a *consistent* predicted scene state (state including all agents), prediction should be done jointly for all interacting agents, describing how the *complete situation* evolves over time. For more complicated situations that contain many potentially interacting agents, there can be complicated chains of interaction that need to be taken into account. If one describes each agent’s behavior using discrete high-level intentions, it becomes apparent that there is a *combinatorial complexity* of how a situation might evolve, which grows exponentially with the number of considered agents. This introduces a great difficulty to interaction-aware motion prediction, still representing a great challenge today [81].

An autonomous vehicle that tries to predict the scene in order to better plan its own trajectory should not predict the scene in isolation, but should account for its *own influence* on the surrounding agents’ behaviors. Traditionally, the prediction of surrounding traffic participants is determined independently of the ego-vehicle’s future trajectory and the results are used as an input for the motion planning module that then plans collision-free trajectories given these predictions [63], [94], [145]. However, as the motion of the ego-vehicle might influence the motion of surrounding agents and vice-versa, the problems of prediction and planning are generally highly interdependent. Not accounting for this interdependency might result in too conservative behavior that has already

## 1 Introduction



**Figure 1.4.** Typical architecture of an autonomous vehicle. Historically, prediction is done before planning, neglecting the influence of the ego-vehicle on its surrounding agents. This thesis proposes to include the interactive prediction model into the planning algorithm to account for possible interdependencies.

shown to strongly reduce acceptance of autonomous cars in mixed traffic [29]. For example, if an autonomous vehicle wants to merge onto a lane with dense traffic, there might not be enough space to merge right away. Without anticipating that other agents are going to react to an attempt to merge, prediction will show that gap to remain too small resulting in the ego-vehicle being stuck. When modeling potential reactions to the ego-vehicle, it can be inferred that squeezing into a narrow gap (which is indeed often done by human drivers) actually is a feasible option, as other agents might open up a gap of sufficient size once they realize ego’s intent to merge. To illustrate the specific role of prediction and planning of an autonomous vehicle, a common software and hardware architecture is depicted in Fig. 1.4, showing the “historic” approach of separated prediction and planning and the more advanced approach of a coupled prediction and planning module which is also proposed in this thesis.

## 1.2 Approach Overview and Research Questions

This thesis presents a probabilistic framework for combined intention estimation and motion prediction based on the concept of Bayesian inference. Bayesian inference allows to intuitively model uncertainty of observations as well as of human behavior on both levels of abstraction, discrete high-level intentions and continuous low-level motion. We investigate what types of driver intentions explain the multi-modality of the possible prediction outcomes best and thus allow to effectively subdivide the space of possible predictions into single modes (i.e., clusters of high probability). We show that introducing these driver intentions and conditioning the continuous behavior models on them increases interpretability of the prediction and allows for a more efficient representation of the predicted belief and thereby for more efficient inference methods.



## 1.2 Approach Overview and Research Questions

Given a specific intention, it is analyzed what the most important parts of the situational context are that dictate a driver’s actions in typical urban environments, allowing for a more informed prediction compared to solely physics-based models. Besides the importance of the static context such as the road geometry or traffic infrastructure, the influence of interaction between multiple drivers is examined. This thesis tries to answer the question on how one can account for the combinatorial interaction between traffic participants in long-term prediction. To this end, we propose to include all agents in the same state space and to solve the prediction problem by employing an incremental probabilistic forward simulation of all agents’ behaviors, thus deriving a distribution over complete scene developments. We define an agent’s behavior at the current time to only depend on the current situation but not on the upcoming actions of other agents, thus avoiding cyclic dependencies between agents. During forward simulation, the state of all agents is updated after each time step such that each agent can base its upcoming action again on the newest available context. This effectively disaggregates complex long-term multi-agent interactions into multiple consecutive single-step single-agent actions. As the behavior models can still be anticipatory (i.e., assuming what others are going to do, in contrast to being purely reactive), sophisticated interactions can still be represented by iterating over time.

An important question is how to cope with the combinatorial complexity of interaction-aware prediction when considering scenes with many agents. This issue is addressed by providing different methods to prune the number of considered hypotheses allowing to concentrate computational efforts at high-probability outcomes and thus to provide a reasonable compromise between complexity and accuracy.

Another crucial part is the utilization of the presented prediction approach by an autonomous vehicle: How can the ego-vehicle predict a situation while considering the influence of its own future actions on the surrounding agents, without previously knowing its own actions? Expressed differently, how to solve the interdependency between the ego-vehicle’s motion planning and the prediction of surrounding traffic participants? To this end, this thesis proposes to include both the ego-vehicle and all surrounding agents in the same state space, allowing to model the interdependencies between all agents. The actions of the ego-vehicle can then be derived using optimization techniques that account for potential actions of other agents and thus solve prediction and planning jointly, while accounting for mutual influence and ensuring the ego-vehicle motion plan to be consistent with the motion prediction of other agents. Two distinct motion planning algorithms are presented that utilize an ego-aware prediction, one based on a partially observable Markov decision process (POMDP) and one based on the concept of cooperative multi-agent planning.

Altogether, this thesis proposes a *probabilistic* prediction approach that both estimates *high-level driver intentions* and predicts *low-level trajectories* while accounting for the current kinematics, available *contextual information* (such as the road geometry) and *interaction* between multiple traffic participants. Furthermore, the presented prediction approach is utilizable for interactive motion planning algorithms, allowing to account for the *interdependency of planning and prediction*.

## 1.3 Related Work

In the area of autonomous vehicles, intention estimation and trajectory prediction of traffic participants have been widely studied. Although these problems are highly coupled, in the existing literature, they are often tackled separately. The first of the following sections is devoted to literature that solely focuses on the estimation of discrete high-level intentions, generally falling into the category of *classification*.

Subsequently, a detailed analysis of different approaches for continuous trajectory prediction is conducted, which generally falls into the category of *regression*. Some of these trajectory prediction approaches additionally estimate high-level intentions either implicitly or explicitly in order to improve prediction accuracy and interpretability. These mixed approaches are also covered in the subsequent sections.

### 1.3.1 Intention Estimation

Popular methods for estimating driver intentions usually either fall into the category of discriminative classifiers such as support vector machines (SVMs) [5], [75], random forests [15], naïve Bayes [117], and neural networks [102], or into the category of probabilistic graphical models such as conditional random fields (CRFs) [139], hidden Markov models (HMMs) [18], [131], Bayesian networks (BNs) [67], [164], [86], and dynamic Bayesian networks (DBNs) [45], [80]. For this purpose, the set of possible intentions is typically predefined by hand and the models are either hand-tuned or learned for these fixed number of classes. For highway scenarios, this set usually consists of “lane change left”, “lane change right”, and “keep lane” (e.g., [11], [75]). Given a specific lane, it is furthermore often distinguished between different longitudinal modes such as “free flow” or “car following” [143]. A more detailed description has been used by Gindele et al. [45], where an overtaking maneuver is composed of a series of behaviors: “acceleration phase”, “sheer out”, “overtake”, and “sheer in”.

For intersection scenarios, the desired route is mostly represented by the predefined turning directions *left*, *right*, and *straight* (e.g., [86], [131], [67], [80], [101], [139]), thus restricting application to only a subset of possible intersections. Besides the intention of a lane change or the desired route, more detailed intentions can be distinguished. A longitudinal classification whether a vehicle is going to yield at an intersection or not, or whether it is going to violate or comply with the traffic rules at a red light has already been investigated (e.g., [5], [6], [15], [67]). Lefèvre et al. [80] determine the set of possible route alternatives and the respective yield positions online using a map and estimate the corresponding intentions.

Interactions between traffic participants are often neglected (e.g., [5], [15], [58], [75], [131]). If the motion of multiple vehicles is interdependent, however, this may result in inaccurate intention estimates (e.g., if a vehicle approaching an intersection has to decelerate because of a slow preceding vehicle, without considering interactions, it might be misleadingly inferred that it intends to turn and slows down for an upcoming curvature). Investigating the so-called *freezing robot problem*, Trautman and Krause [141] have shown that agents typically engage in *joint collision avoidance* and cooperatively

make room for one another to create feasible trajectories. Therefore, possible future interactions between agents should be taken into account both for estimating the current intentions as well as for predicting the future trajectories.

Other works on intention estimation have already explicitly modeled interdependencies between agents: Liebner et al. [86] and Klingelschmitt et al. [67] consider the dependency on the directly preceding vehicle in order to improve the route estimation at intersections. Promising results have been shown by Phillips et al. [102] using a long short-term memory (LSTM)-based route classification for intersections, which considers the states of up to seven surrounding vehicles, therefore respecting possible interactions implicitly. Although these works do account for interactions between vehicles within their behavior or classification models, their maneuver representations typically lack the descriptive power to specify the relative motion of vehicles in detail. When, for example, two oncoming vehicles are both planning to leave an intersection on the same exit, their maneuver representations provide no way to distinguish between who turns first. Including these relations in the maneuver representation allows for a more detailed differentiation of high-level maneuvers that might be beneficial for scenarios with close interaction.

In the area of robot motion *planning*, more sophisticated maneuver representations based on the concept of homotopy have already been proposed [13], [17], [19]–[21], [31], [49], [73]. Maneuvers are defined by clustering paths or trajectories into homotopy classes, which would also allow to distinguish the aforementioned case in the area of *prediction*. Kuderer et al. [73] present a framework to generate homotopically distinct navigation paths online for a mobile robot navigating hallways with the help of Voronoi diagrams. By using homotopy classes they avoid sticking to local minima during the path optimization. Bender et al. [17] propose to distinguish different maneuvers of autonomous vehicles in relation to nearby dynamic obstacles based on homotopy classes, allowing to determine the best trajectory given any of those classes using local optimization techniques. Enumerating all these classes and solving the respective optimization problems allows to pick the maneuver with lowest cost. Gu et al. [49] automatically discover tactical maneuver patterns based on sampled trajectories. Using pseudo-homotopy, they can extract distinct maneuver classes and decide for a suitable maneuver according to their cost function. All these works focus on ego motion planning and assume other agents to have known motion profiles such as constant velocity. Thus, these approaches are not suitable for predicting the maneuvers of other drivers and don't allow for interactive multi-agent environments which account for the fact that vehicles react to each other and that their decisions may be coupled. Still, for estimating intentions of multiple interacting agents, a maneuver representation based on homotopy classes being generated at runtime (given a topological map and the states of surrounding agents) might be advantageous over a predefined fixed set of intention.

Most of the previous works solely focus on classifying an agent's future behavior into discrete classes, but do not predict continuous trajectories which are typically needed for motion planning algorithms to plan collision-free trajectories. The following sections present different methods for predicting such continuous trajectories.

### 1.3.2 Physics-based Prediction

The most simple trajectory prediction methods are so-called *physics-based* and do not consider the situational context (such as the road geometry) or interaction with other traffic participants, as also described in the survey of Lefèvre et al. [81]. These models extrapolate the current kinematic state of an agent using some assumptions such as maintaining a constant velocity or acceleration. Schubert et al. [119] compare different motion models such as constant turn rate and velocity (CTRV) and constant turn rate and acceleration (CTRA) which are typically used for vehicle tracking, but, naturally, can also be utilized for forward simulation to predict trajectories multiple steps into the future.

Instead of applying such simple assumptions, it is also possible to learn more accurate models given the history of the kinematic state of an agent. Wiest et al. [154] learn a non-variational and a variational Gaussian mixture model based on the expectation maximization (EM) algorithm to predict vehicle states in a probabilistic fashion, only incorporating the previously observed trajectory snippet of a vehicle. They show that the variational model achieves promising results in predicting a single vehicle driving on a curved track, outperforming the non-variational model for predictions of 2 s into the future. However, especially at intersections and for long prediction horizons, these context-independent and interaction-unaware models tend to have low predictive performance due to the high dependency of the drivers' actions on the road geometry, traffic rules and interactions to surrounding vehicles. The trajectories of multiple vehicles in a traffic scene are often highly interdependent because drivers must avoid collisions, comply with traffic rules, and thus react to other drivers' actions.

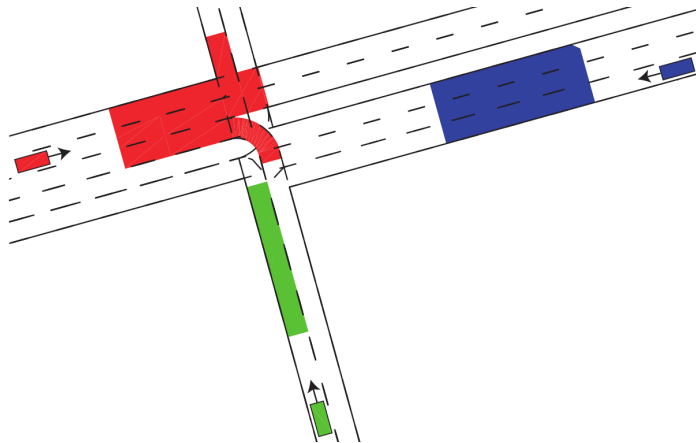
### 1.3.3 Reachable Set-based Prediction

Althoff, Koschi, and Magdici [3], [69] calculate the reachable set of future states of an agent by considering anything from full acceleration to full braking as well as all possible routes at intersections and arbitrary lane changes, resulting in an over-approximation of all possible occupancies, as depicted in Fig. 1.5. By considering physical constraints and assuming that the traffic participants abide by the traffic rules they can reduce the dimensions of these occupancies, allowing for less conservative predictions.

Predictions based on reachability analysis, as mentioned in Sec. 1.1, have the benefit of allowing formal verification and thus providing safety guarantees, but do not exploit additional information that allows to infer the probability of specific future hypotheses which is beneficial for comfortable and more efficient driving. Thus, these set-based predictions are generally said to result in rather conservative behavior.

### 1.3.4 Rule-based Prediction

So-called *rule-based* or heuristics-based behavior models define a (in most cases deterministic) mapping from situation and potentially a predefined intention of a driver to an action resulting in a state transition to the subsequent time step. In contrast to solely



**Figure 1.5.** Reachable set prediction (given some constraints) at an intersection for a specific future time interval. Graphic taken from [69]. © IEEE 2017

physics-based models, they do incorporate some additional contextual information such as the road geometry, traffic rules and the presence of surrounding agents.

The traffic simulator SUMO [71] comes with different rule-based behavior models built in. It distinguishes between models for car-following, lane-changing, and traversing intersections. The most well-known car-following models are the so-called Krauß-model [72] and the intelligent driver model (IDM) [142]. Typically, these models come with tunable parameters such as desired time headway or acceleration and deceleration ability. They define a deterministic mapping from relative distances and velocities to accelerations. Similarly, the lane-changing and intersection models can also be parameterized, for example in terms of the maximum lateral acceleration in a curve, the minimum time gap to other vehicles at overlapping lanes, or the tendency to ignoring the right of way. Thanks to its simplicity, the IDM is widely applied for driver intention estimation (e.g., [86], [166]) and for modeling how others react to specific plans of an autonomous vehicle (e.g., during forward simulation using Monte Carlo tree search [83]). Although these simple heuristics-based models are well suited for specific scenarios and controlled traffic simulation, they tend to be not detailed and realistic enough for accurate prediction of diverse human driving in real environments.

### 1.3.5 Planning-based Prediction

Another possibility of incorporating contextual information in the prediction of traffic participants is to model the prediction problem as a planning problem by modeling drivers as agents optimizing some sort of context-dependent cost function. Such methods are also called *Theory-of-Mind* [48, pp. 257 ff.] methods, as one assumes a human conducts rational actions given some objective.

Generally, any type of motion planning algorithm can be utilized for prediction purposes as well. There is an enormous variety of different planning algorithms, ranging from physically motivated methods such as potential fields [16] and elastic bands [107]

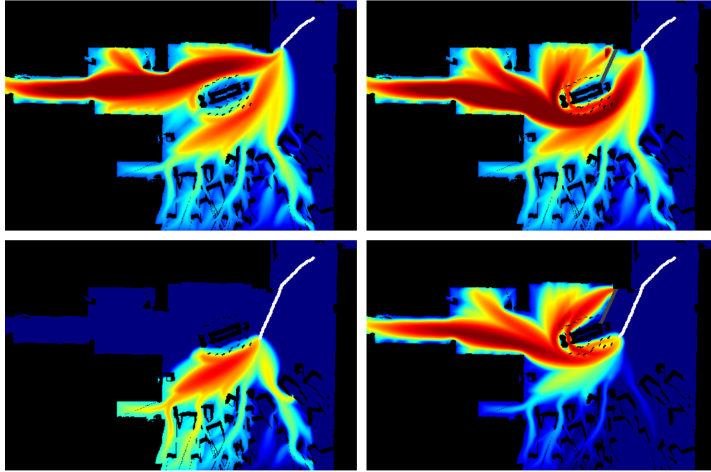
## 1 Introduction

over the optimization of hand-designed cost functions to learned cost functions such as generated by inverse reinforcement learning (IRL). As the subject of motion planning itself is out of the scope of this thesis, the interested reader is referred to the books by LaValle [78] and Latombe [77] that cover the fundamentals of motion planning.

Eichhorn et al. [38] model human drivers as optimizers of an optimal control problem and utilize cost-to-go gradients for different route hypotheses to infer the most likely route at intersections. Rehder and Kloeden [108] introduce the pedestrian’s destination as a latent variable in order to convert the prediction problem into a planning problem. They utilize the unicycle model for the pedestrian’s motion and apply it to both a forward prediction of the current position and a backward prediction of an assumed destination goal. Discretizing the state space into a grid representation allows for utilizing convolution for both forward and backward prediction using a convolution filter mask (representing the motion) and the belief at a specific time step. They show improved long-term prediction by this goal-directed prediction with goal inference.

When planning approaches are used for prediction, they are often combined with machine learning techniques, as it is desired to model how humans are *actually* behaving and not what the developer of the model might define to be the optimal behavior. Thus, in a later approach, Rehder et al. [109] propose to first predict multiple goal regions with an LSTM-based mixture density network and use a planning-based prediction approach that is based on IRL and a Markov decision process (MDP) that is realized as a convolutional neural network (CNN). Ding and Shen [35] utilize an LSTM for estimating the predefined driver intentions *forward*, *yield*, *turn left*, *turn right* in urban environments. Given the most likely intention, they generate a trajectory using non-linear least squares optimization given a constructed cost-map that includes contextual information such as the lane geometry and static and dynamic obstacles. As the optimization procedure includes multiple interacting vehicles, they iteratively optimize the trajectories of all agents, always assuming all other agents’ trajectories to be fixed and given by the values of the previous iteration. They show that their planning-based approach outperforms two regression methods, one simple least square polynomial method and one recurrent neural network (RNN) encoder-decoder architecture. Ziebart et al. [159] predict the motion of pedestrians by assuming people behave like planners and modeling their behavior in MDPs. For that purpose, they learn an obstacle-dependent cost function using maximum entropy IRL [160] and model different possible goal locations. Their approach generalizes well to unseen environments, as depicted in Fig. 1.6. Similarly, Gonzalez et al. [46] model each agent in a highway scenario in an MDP and employ maximum entropy IRL to learn the corresponding cost function. Sadigh et al. [115] model human drivers as agents that optimize a reward function that they also learn with IRL. Nested optimization based on the numerical optimization method L-BFGS allows to embed the assumed planning process of surrounding humans into the planning process of the ego-vehicle. This nested optimization allows for leveraging the effects of the autonomous vehicle’s behavior on the actions of the other traffic participants.

Ma et al. [89] base the problem of predicting humans in a crowd on game theory, modeling the intertwined decision making processes of multiple pedestrians as a collaborative multi-player game. They model each pedestrian to take a path based on their own



**Figure 1.6.** Planning-based pedestrian prediction by Ziebart et al., showing the generalization capabilities by comparing original predictions (left) with the prediction after adding an additional object (right). Graphic taken from [159]. © IEEE 2009

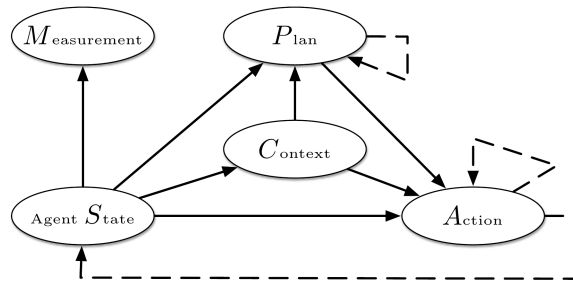
predictions of how other pedestrians will move. Using fictitious play allows to simulate likely future motion.

### 1.3.6 Machine Learning-based Prediction

A very popular approach to the problem of trajectory prediction is the utilization of machine learning methods, ranging from traditional regression models such as linear regression [104], Gaussian processes (GPs) [7], [140], random forests [44], or shallow neural networks [138] to more recent deep learning methods [2], [32], [50], [79]. Machine learning allows for high predictive performance without the need of cumbersome and error-prone hand-tuning.

**Traditional Machine Learning:** Armand et al. [7] learn velocity profiles for stopping at an intersection using GPs with heteroscedastic variance. They include knowledge about the upcoming intersection, but do not consider surrounding vehicles. Tran et al. [140] define multiple GPs conditioned on the intentions of a driver (“turn-left”, “turn-right”, “go-straight”, “stop-and-go”). Using these GPs as transition models of a particle filter allows them to estimate the intentions and to predict continuous trajectories. As contextual information, such as the existence of surrounding traffic participants, is not incorporated, the framework is not suited for multi-agent scenarios.

A two-staged approach for prediction at intersections is employed by Platho et al. [103], in which they first classify the traffic situation for each vehicle into one of multiple predefined driving situations (“stopped by red traffic light”, “stopped by leading car”, “stopped by intersection”, “not influenced”) and then predict the velocity profile using situation-specific models learned with random forests. As these profiles only depend on features of the current situation (e.g., states of preceding vehicles), but do not incorporate the prediction of the surrounding vehicles, future interdependencies are ignored. Another



**Figure 1.7.** Dynamic Bayesian Network proposed by Gindele et al. Graphic taken from [44].  
© IEEE 2013

two-staged approach to infer intentions and predict trajectories is proposed by Bahram et al. [11], where highway maneuvers are first estimated based on multi-agent simulations and then used as input for a continuous trajectory prediction. Thus, both works solve the two problems of intention estimation and motion prediction separately, but improve their trajectory prediction by their intention estimates.

Modeling the development of a traffic situation as a stochastic process conditioned on the agents’ hidden intentions allows to infer these intentions by incorporating measurements and to predict future trajectories by iteratively applying the transition models (forward simulation). Such Bayesian frameworks handle the problems of intention estimation and motion prediction in a combined way, utilizing the same generative behavior models. Gindele et al. [44] propose such a method in which they learn context-dependent action models of traffic participants using random forests that are conditioned on a driver’s route intention and embed them into the DBN shown in Fig. 1.7. They estimate the multi-modal, hybrid (both discrete and continuous variables) belief using sequential Monte Carlo (SMC) inference and predict the future scene development based on a non-linear transition model. Although they do not explicitly evaluate the route estimation capabilities, they show that their trajectory prediction outperforms a CTRV-based model in simulated intersection scenarios.

Wheeler et al. [153] present a survey on learned probabilistic driver behavior models for highways scenarios, comparing different traditional machine learning models including random forests, linear Gaussian, Gaussian mixture and Bayesian networks on the context classes of “free-flow”, “car following” and “lane change”. All these models allow an interaction-aware forward simulation and an estimation of discrete intentions when included in a Bayesian framework. However, the authors state that such probabilistic action models based on traditional supervised learning methods are reaching their limit, as small inaccuracies of the learned model compound over time during forward simulation, which is known as the so-called compounding error problem: As the output of the learned policy at one time step influences the environment serving as an input to the policy in the subsequent time step, any potentially small error accumulates over time, eventually leading to situations not encountered during training [110]. This result-



ing covariate-shift in turn typically results in even poorer predictions that can cause a feedback cycle known as cascading errors [9].

The compounding error problem is generally known in the context of learning from demonstrations in which an agent tries to learn a policy that resembles expert behavior and then applies that policy for a longer horizon. However, it does also apply in the case of prediction when a learned policy model is called iteratively in the process of forward simulation in order to predict longer trajectories. One major difference in prediction (compared to a real agent executing actions in an environment), however, is that in each time step, the initial state from which the prediction is initiated “resets” to the observation of (or the belief over) the agent’s actual current state. Thus, the negative impact of the compounding error depends on the horizon of the prediction, not on the duration of an agent (that uses such a prediction model) acting in an environment. Many *deep learning*-based action models do suffer from the same problem in theory, but are generally more accurate and thus are able to reduce this error significantly. Another possibility to reduce this problem is to solve the prediction task not using forward simulation of one-step models, but to predict full trajectories up to a desired horizon at once, allowing the learning algorithm to account for the deviations multiple steps into the future within the loss function. Similarly, Asadi et al. [8] have shown successes in multi-step planning (utilizing the outcome of a sequence of actions) over one-step planning in the context of reinforcement learning (RL).

**Deep Learning:** The field of deep learning has revolutionized the area of machine learning with the availability of large amounts of data and high-end compute. Most of recent trajectory prediction approaches utilize some sort of deep learning technique, as performance scales well with the amount of training data used, the need for feature engineering is reduced, and complex problems can be modeled within a single training objective.

Already simple deterministic feed-forward models have shown to achieve good results in the area of trajectory prediction and are able to outperform classical machine learning methods. Sarkar et al. [116] estimate traffic participants’ paths and predict their trajectories using a two-staged approach. They segment the road into different clusters representing motion patterns such as “turning right” based on recorded trajectories. For the velocity prediction, they propose a feed-forward neural network and compare it to a modified version of the DBN from [44], both utilizing the extracted clusters. They show that their deep neural network-based approach reduces prediction error compared to the DBN which was trained with random forests.

Lenz et al. [82] compare different deep neural network architectures for probabilistic Markovian action prediction in highway scenarios. As the action models are not conditioned on a driver’s intention such as lane changing or lane keeping, the authors model the action as a Gaussian mixture distribution to account for future multi-modality. They find that a fully connected feed-forward network outperforms recurrent architectures on the domain at hand. Those models can be utilized for an interaction-aware forward simulation of a whole traffic scene and the authors plan to integrate them in their motion planning framework in future work.

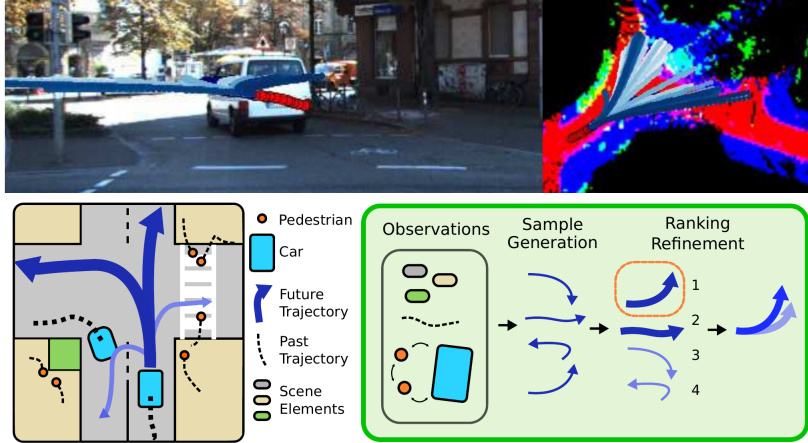
## 1 Introduction

Despite the success of feed-forward deep learning models, it is often beneficial to account for temporal dependencies using recurrent architectures. A major breakthrough in recurrent neural networks (RNNs) has been achieved by the introduction of the long short-term memory (LSTM) by Hochreiter and Schmidhuber [55]. The so-called gated recurrent unit (GRU) proposed by Cho et al. [25] represents a simplification of an LSTM with fewer parameters and has shown to have benefits on smaller datasets. Pool et al. [105] propose a GRU-based method to predict the trajectory of a bicyclist based on road information, the time-to-collision (TTC) with a following vehicle, and whether the bicyclist is indicating a turn with their arm. Although they are able to predict a Gaussian position distribution 1 s into the future with good accuracy, the specifics of the scenario (one bike, one vehicle following, the fixed route options left and straight) are not encoded in the input features, but are rather learned by heart. Only one intersection is used for training and evaluation and the input features have been designed in a scenario-specific way, making it difficult to analyze the generalizability of the approach.

**Sequence-to-Sequence Models:** A typical drawback of standard deep learning models is that input and output vectors have to be of fixed size. Sequence-to-sequence models eliminate this requirement as they consist of an RNN-based encoder-decoder architecture, where the encoder encodes an input sequence (e.g., a sentence) and the decoder uses this encoded input and generates an output sequence (e.g., the sentence in another language). The available input information is compressed into a context vector of fixed length which is expected to be a good representation of the relevant parts of the input sequence. Thus, arbitrarily long input and output vectors can be produced. The fixed length context vector, however, can be a bottleneck especially for very long sequences, resulting in decreased performance.

Encoding the history of past states of an agent using a recurrent network is a common way to compress an agent’s past trajectory and extract relevant information from it. These compressed representations can then be utilized as an input to a corresponding decoder for the forecasting of future trajectories. Park et al. [98] utilize such an LSTM-based encoder-decoder model combined with a so-called *beam-search* to produce a fixed number of most likely future trajectories. Their sequence-to-sequence model allows to utilize an arbitrarily long history and to predict up to an arbitrary horizon. Deo et al. [33] also utilize an LSTM encoder-decoder framework to predict vehicle trajectories on highways. They first classify six predefined maneuver classes (lane changes left and right as well as lane keeping, each having two different longitudinal velocity subclasses) with an LSTM. Secondly, they encode the history of states of an agent in another LSTM and use both the maneuver encoding and the trajectory encoding as an input for their decoder to predict the future. The decoder outputs a bivariate Gaussian position distribution per maneuver for each time step, showing promising results on both NGSIM highway datasets [26], [53].

Although most works encode and decode temporal sequences (e.g., a trajectory of an agent), naturally, other types of sequences can be represented. One alternative is to represent a spatial list of agents as the input sequence and a list of corresponding future trajectories as the output sequence, thus allowing for an arbitrary number of agents.



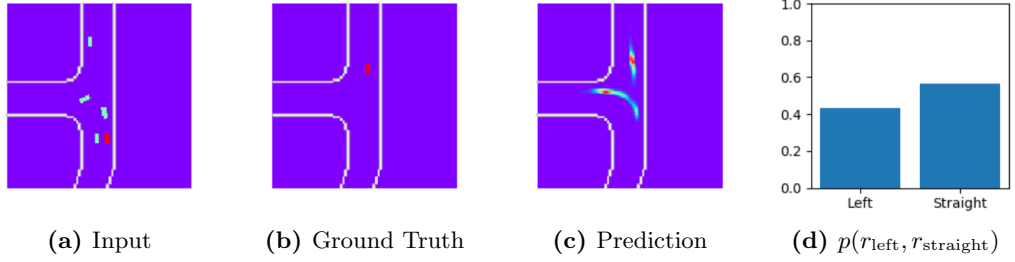
**Figure 1.8.** Exemplary future prediction (top) and workflow (bottom) of the DESIRE framework. Graphic taken from [79]. © IEEE 2017

**Deep Generative Models:** One important aspect of motion prediction is the representation of uncertainty. In contrast to learning the parameters of a parametric distribution (e.g., Gaussians as in the previous works), deep generative models such as conditional variational autoencoders (CVAEs) (proposed by Sohn et al. [125]) or generative adversarial networks (GANs) (proposed by Goodfellow et al. [47]) allow to represent arbitrary distributions. The idea is that these networks learn a mapping from a simple parametric input noise distribution to the desired output distribution. This allows to sample from the output distribution indirectly by sampling from the parametric input distribution and applying a forward-pass through the network.

Hu et al. [60] utilize a CVAE for predicting multi-modal trajectories while considering *pairwise* interaction. The deep learning-based multi-agent trajectory prediction framework by Lee et al. called DESIRE [79] aims to achieve interaction-awareness, account for the uncertainty and multi-modality of the future (e.g., induced by different possible routes) and achieve long-term accuracy—all within one training procedure. The model consists of multiple modules handling the trajectory sample generation (based on a CVAE), the trajectory ranking and refinement (based on inverse optimal control (IOC)) and the scene context fusion (based on a CNN encoding the scene context), as depicted in Fig. 1.8. The CVAE allows to create samples of multi-agent trajectories, accounting for the uncertainty and multi-modality. The ranking and refinement module reduces accumulating error and thus improves long-term prediction. The scene context fusion allows to incorporate contextual information. They achieve promising results on the KITTI dataset and the Stanford drone dataset.

**Graph Neural Networks:** One possibility to represent interdependencies between multiple interacting agents is given by so-called graph neural networks. A comprehensive survey on graph neural networks has been conducted by Wu et al. [156]. Diehl et al. [34] utilize graph neural networks for modeling a traffic scene as a graph of interacting vehicles. This allows for a flexible and abstract model of interactions. However, it is not

## 1 Introduction



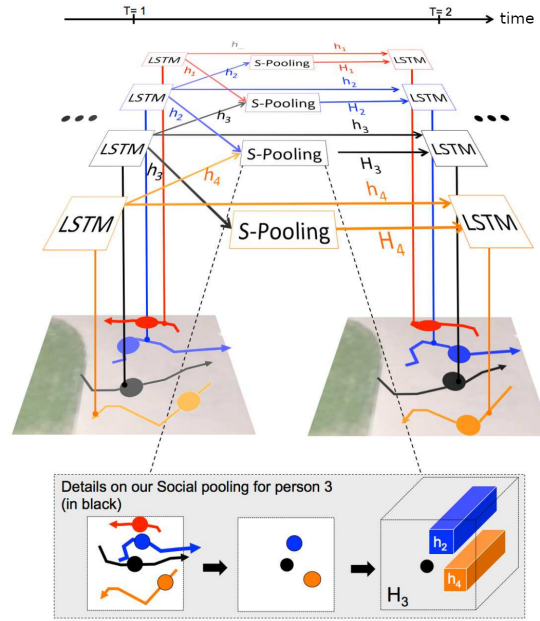
**Figure 1.9.** Mänttari et al. propose to utilize a schematic top-down view of a traffic scene as sole input to a CVAE for predicting the pose of multiple agents 6 s into the future. The ground truth and prediction are only shown for a single agent (red) to improve visualization. The color-coding of the prediction is generated by drawing multiple random samples from the CVAE. Graphic taken from [90]. © IEEE 2019

trivial to decide which agents will interact in advance, thus, finding relevant connections in this graph remains an open problem.

**Top-Down Image:** Utilizing a top-down image of a traffic situation allows to represent a variable number of agents and arbitrary road layouts with a fixed size input. Furthermore, the number of agents does not affect the computational complexity and there is no need to predefine specific types of interaction.

Djuric et al. [36] propose to use a rasterized actor-centric top-down view of a traffic scene as an input for a CNN-based trajectory prediction. They achieve promising results for short horizons but run into problems for longer horizons because of the neglected multi-modality and combinatorial nature of motion prediction. In their follow-up work, Cui et al. [27] improve on this problem by proposing to predict a fixed number of modes and utilizing a multi-modal loss function which they call “Multiple-Trajectory Prediction (MTP)” loss. This MTP loss only penalizes the mismatch of the closest predicted mode (given some distance function) and forces this mode’s probability to be close to one. This loss successfully prevents the problem of *mode collapse* which is typical for multi-modal predictions with classical loss functions such as the mean squared error loss. A remaining problem is the choice of the fixed number of modes, which has to be defined before training. Mänttari and Folkesson [90] also propose to utilize top-down images of a traffic scene as an input for a CVAE-based trajectory prediction, as depicted in Fig. 1.9. Using a CVAE naturally allows to represent arbitrary distributions and thus to also represent multi-modality. Currently, they do not incorporate any other context than road boundaries and surrounding agents, but road markings, traffic signs and traffic lights could potentially be added to the image. The state and trajectory outputs are discretized according to the limited image size. Although they are showing promising results, the evaluation uses a stationary top-down view of a single intersection, making it difficult to analyze generalizability to different road layouts. Similarly, Hoermann et al. [57] represent the scene as an occupancy grid and use a CNN to predict the scene in a probabilistic fashion.

**Social Pooling:** The so-called *social pooling* as proposed by Alahi et al. [2] for human trajectory prediction in crowded spaces became a popular method to model interactions

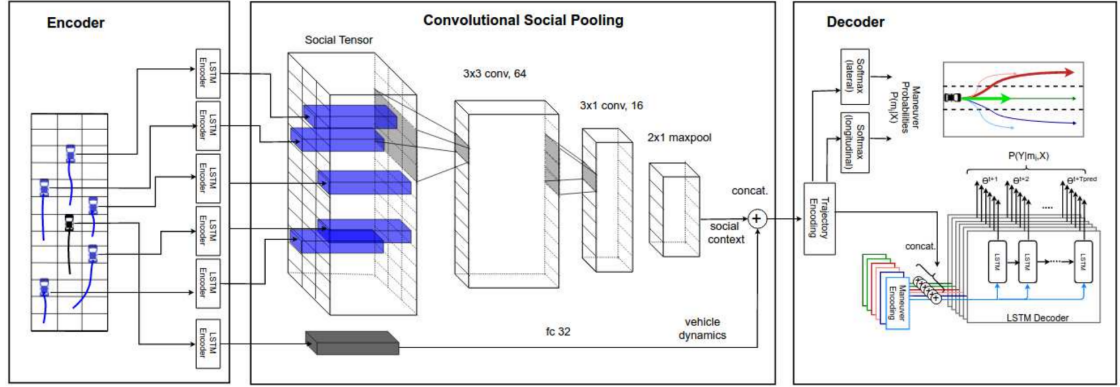


**Figure 1.10.** Social pooling proposed by Alahi et al. to account for common sense rules and social conventions when predicting pedestrians. Separate LSTMs for each agent’s trajectory are connected to each other through a social pooling layer to share information between spatially proximal agents. The hidden state  $h_i$  of an LSTM captures the latent representation of the  $i$ th person. This representation is shared with neighbors by building a social hidden-state tensor  $H_i$ . Graphic taken from [2]. © IEEE 2016

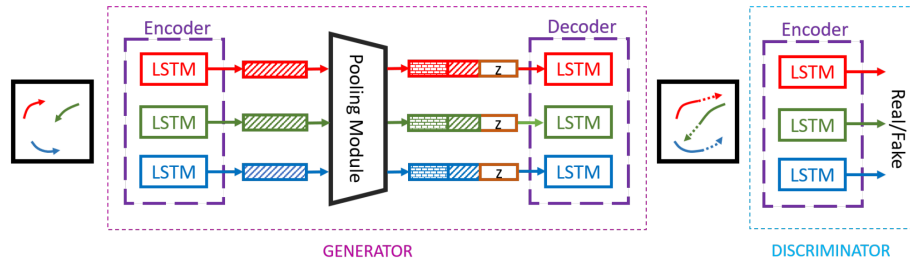
between nearby agents. Their method is depicted in Fig. 1.10 and consists of one LSTM per agent encoding its trajectory. All LSTMs are then connected using social pooling layers in order to share information between spatially proximal agents, allowing to create relationships between an agent and its close neighborhood. They show promising results in the area of pedestrian prediction in crowded spaces, but do not account for the inherent multi-modality of trajectory prediction.

Deo and Trivedi [32] build on this idea and extend their prior work [33] by proposing to use an LSTM encoder-decoder model that utilizes convolutional social pooling, as depicted in Fig. 1.11, for robustly learning interdependencies of vehicle motion in highway scenarios. Their model uses convolutional connections as opposed to fully connected layers that more robustly model spatial configurations of interacting agents in a scene. The LSTM decoder further estimates the probability distribution over six predefined maneuvers and predicts corresponding trajectories, resulting in a multi-modal prediction. However, they do not account for the interdependency of the future trajectories of multiple vehicles, neglecting the combinatorial nature of trajectory prediction. Zhao et al. [158] also utilize both the past trajectories of agents which are encoded into single LSTMs and a scene context top-down image which allows to capture position-invariant spatial interactions using convolutional operations. They use an adversarial loss with a

## 1 Introduction



**Figure 1.11.** Convolutional social pooling framework proposed by Deo et al. Graphic taken from [32].  
© IEEE 2018



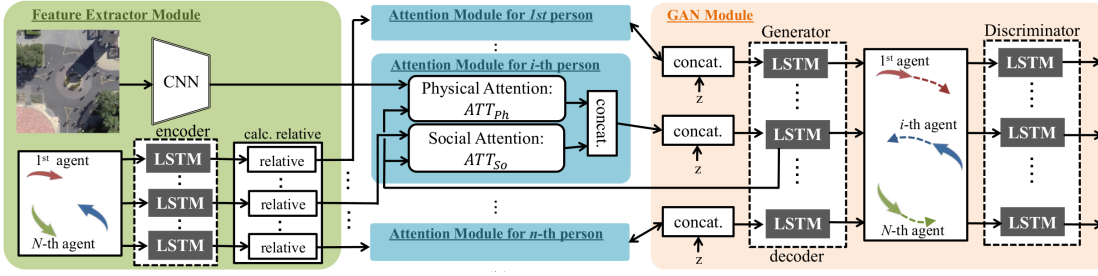
**Figure 1.12.** Social GAN architecture proposed by Gupta et al. Graphic taken from [50].  
© IEEE 2018

generator and a discriminator as in GANs in order to model the multi-modal uncertainty of the prediction.

Gupta et al. [50] propose a pedestrian prediction framework called Social-GAN, combining LSTM-based sequence-to-sequence models, observing motion histories and predicting future trajectories, with a novel pooling mechanism to aggregate information across people. They also model the problem as a GAN as depicted in Fig. 1.12. Similarly to [27], they encourage diverse sample generation by defining their loss only on the “best prediction” (in terms of euclidean distance) of a set of sampled predictions.

**Attention Mechanism:** So-called *attention mechanisms* which originally became famous in the fields of vision and natural language processing [10], [88], [93], [146] allow to focus on parts of the available input information that currently matters most and thus improve efficiency drastically. Utilizing this *selected focus*, they have been shown to reduce problems that recurrent sequence-to-sequence networks typically encounter with very long sequences.

Vemula et al. [147] present *social attention* which relaxes the assumption of social influence to be only local, i.e., available between spatial neighbors. They learn an attention model that includes possible influences of all available pedestrians in a crowd and show that this model outperforms locally constrained social pooling. Similarly, Mes-saoud et al. [92] tackle the problem of locality in social-pooling. They propose to use



**Figure 1.13.** SoPhie framework by Sadeghian et al. Graphic taken from [112]. © IEEE 2019

non-local social pooling to better capture spatial interactions between vehicles on a highway using an LSTM encoder-decoder framework. Furthermore, following the attention mechanism by [146], they mimic human reasoning, focusing attention selectively on a subset of agents that most influence the targets behavior, considering other agents less. They show improved performance on multiple highway datasets compared to methods not utilizing non-local social pooling.

Sadeghian et al. [113] propose to utilize raw top-down images with a visual attentive recurrent component that learns where to look in a large image of the scene. In [112] they extend their idea to model social and physical attention mechanisms together with a GAN-based trajectory generation module. Their framework which they call SoPhie is depicted in Fig. 1.13.

**Imitation Learning:** The task of learning from an expert is called imitation learning, apprenticeship learning, or learning from demonstration [1]. Using expert demonstrations to learn a policy is a well-known approach in the fields of robotics and AI. Besides training an agent what to do, these techniques can also be employed for prediction purposes. Behavior cloning (BC) treats imitation learning as a supervised learning problem, fitting a model to a fixed dataset of expert state-action pairs. Many of the previously presented deep learning approaches fall into the class of BC. A typical problem of such supervised learning approaches is given by the so-called *covariate shift* which is a result of the compounding error: Small inaccuracies compound during simulation, potentially leading to states that are underrepresented in the training data. This in turn results in even poorer predictions which might result in invalid or unseen situations (e.g., driving off-road) [110]. One way to improve on this problem are data augmentation techniques such as *data as demonstrator* [148] or *dataset aggregation (DAgger)* [111] or generally using a more diverse dataset.

Another approach to imitation learning is to recover a cost function that best explains the expert’s behavior using inverse reinforcement learning (IRL), typically assuming the expert follows an optimal policy. This cost function can in turn be used by planning methods or forward reinforcement learning (RL) to produce corresponding actions (cf. Sec. 1.3.5). This approach does not suffer from the compounding error and tends to generalize well to unseen situations. However, IRL also tends to be computationally expensive.

## 1 Introduction

The so-called generative adversarial imitation learning (GAIL) proposed by Ho and Ermon [54] represents a third approach to imitation learning. It is a method for direct policy optimization without first recovering a cost function. It basically solves IRL followed by RL by utilizing an adversarial method similar to GANs that iterates between a gradient step that updates the discriminator (trying to tell whether sampled trajectories from the learned policy are expert trajectories or not) and a trust region policy optimization (TRPO) step that updates the policy. Similar to IRL, GAIL-based models also avoid common pitfalls of BC or supervised methods such as the compounding error.

Kuefler et al. [74] utilize GAIL for learning human policies for highway driving and extend the concept to the optimization of recurrent policies. An extension of GAIL to the multi-agent case was proposed by Song et al. [126], as the single agent version does not consider the responses of other agents during interaction and thus results in errors in multi-agent scenarios. A special case of this multi-agent GAIL that assumes all agents are homogenous was introduced by Bhattacharyya et al. [22] in the context of highway driving. All agents share the same policy and receive rewards from the same critic. Thus, their approach is called parameter sharing GAIL (PS-GAIL). They show improved performance compared to the single-agent version. As PS-GAIL averages out individual differences among different drivers, Si et al. [121] propose an online adaptation framework of these offline learned models to recover individual differences for improved prediction. Their approach which they call adaptable generative prediction networks (AGen) utilizes the recursive least square parameter adaptation algorithm and outperforms standard homogenous PS-GAIL models in highway driving scenarios.

### 1.3.7 Motion Planning Accounting for Interrelation with Prediction

Most existing literature treats the problems of prediction and planning separately. Typically, prediction is solved first without accounting for the influence of the ego-vehicle, then, the planning module utilizes the determined prediction as a fixed input [63], [94], [145]. More advanced approaches account for the interrelation of the prediction and planning problems. As shown by Trautman and Krause [141], considering other agents' reactions to the ego-vehicle's planned trajectory helps to circumvent the so-called *freezing robot problem* and leads to less conservative actions which might be needed in dense and interactive environments.

Sadigh et al. [115] model human drivers as optimal planners that optimize a cost function that accounts for the existence of the ego-vehicle and other contextual information such as the road geometry. The parameters of this cost function are acquired using inverse reinforcement learning (IRL) and human demonstrations. They formulate the problem as a nested optimization problem: the optimization of the ego-vehicle's trajectory includes an assumed trajectory optimization of the surrounding agents that in turn depends on the ego-vehicle's planned trajectory. They show that they can leverage effects of the ego-vehicle's actions on the actions of surrounding agents in order to achieve the robot's goal more efficiently. However, their approach is conceptually limited to a single other agent, is computationally expensive due to the nested optimization, and does not account for uncertainty or multi-modality. Building on their previous work,



they show in [114] that this effect of the ego-behavior allows to conduct information gathering actions to better estimate the surrounding drivers' intentions. Sun et al. [133] build on the same concept of nested optimization and propose courteous (or cooperative) autonomous cars by adding a cost term that accounts for the costs of other drivers caused by the ego-vehicle.

Swamy et al. [169] investigate different methods for interaction-aware robot decision making based on machine learning methods in terms of their sample complexity. They evaluate two model-based RL approaches (neural network-based prediction and planning-based prediction) and a model-free RL approach, all accounting for the interdependency of the ego-vehicle and a surrounding agent. Both model-based approaches are utilized in a nested optimization scheme to plan ego-vehicle trajectories that comply with the corresponding prediction models (as proposed by Sadigh et al. [115]). They show that the planning-based prediction (so-called *Theory-of-Mind*), that utilizes a cost function learned with IRL needs less training data than the neural network-based prediction learned in a supervised fashion. Additionally, the planning-based prediction approach does not require closed-loop human-robot interaction data but can be learned with pre-recorded human-human data. Expectably, both model-based approaches outperform a model-free RL approach (which does not explicitly learn to predict) in terms of sample efficiency by multiple orders of magnitude.

Fisac et al. [40] account for the coupling between the ego-vehicle's planning and its predictions of surrounding agents by utilizing game theory. They introduce a novel online-capable game-theoretic trajectory planning algorithm that hierarchically decomposes the underlying dynamic game into a long-horizon strategic game and a short-horizon tactical game. The long-horizon game relies on simplified dynamics but utilizes the full information structure, whereas the short-horizon game uses full dynamics but a simplified information structure. The computed value function of the long-horizon game serves as a guiding term in the short-horizon trajectory optimization. Due to computational complexity, they limit their approach to two vehicles, the ego-vehicle and one other agent.

Lenz et al. [83] utilize Monte Carlo tree search (MCTS) for planning ego-vehicle trajectories on highways, modeling other agents to follow the intelligent driver model (IDM). As the IDM depends on the current situation including the ego-vehicle's state, they account for the reaction of other agents on possible future ego trajectories, allowing for interactive maneuvers. Additionally, they account for the costs of surrounding agents in order to drive in a more cooperative manner. Paxton et al. [100] formulate the problem as a reinforcement learning problem utilizing neural networks to learn both low-level control policies as well as task-level option policies. In order to explore future worlds of a complex and interactive environment, they utilize MCTS over the high-level options (represented by the learned control policies) with an option selection based on the learned high-level policy.

Cunningham et al. [28] and Galceran et al. [41] present an interaction-aware and online-capable decision making framework for autonomous vehicles which they call *Multi-Policy Decision Making*. They assume that both the autonomous vehicle and other agents execute actions from one policy of a set of plausible closed-loop policies and

## 1 Introduction

decide in each time step for the best available policy for the ego-vehicle. These policies are engineered based on domain knowledge and represent high-level actions such as driving along a lane, changing lanes or executing a parking maneuver. As they describe the behaviors of others in the form of policies that depend on the own agent’s state, they can account for the ego-vehicle’s influence on its surrounding traffic participants.

A popular method to account for mutual interaction and for prevailing uncertainty is to model both the ego-vehicle and surrounding vehicles within the same state space of a partially observable Markov decision processes (POMDPs) [12], [24], [87], [120], [127]. Sezer et al. [120] model an autonomous vehicle’s decision making with a POMDP in which human drivers have an unknown route (straight or turn), an unknown driving style (aggressive or compliant), and actions that depend on the ego-vehicle’s position. They utilize an offline solver and only consider one other vehicle. Brechtel et al. [24] also include other agents within the state space of their POMDP to account for possible interaction. They define Markovian action models for predicting the surrounding agents depending on the ego-vehicle’s state and thus account for mutual interaction. Furthermore, they explicitly model occlusions and are able to decrease the uncertainty about other vehicles’ positions by executing information gathering actions. They utilize an offline planner and a scenario-specific learned discretization of the state space. Bai et al. [12] propose an online-capable POMDP approach that utilizes an equidistant discretization of the state space in order to reduce complexity. They introduce a fixed set of predefined hidden driver intentions consisting of routes and driver types. Song et al. [127] estimate the intentions of other traffic participants with a deterministic HMM, and utilize these intentions within an online-capable POMDP. As the intentions are assumed to be deterministic, no intention uncertainty is considered during planning and no information gathering actions are performed. Liu et al. [87] model driver intentions based on predefined routes and discrete driving styles and distinguish between three discrete acceleration values that are chosen based on an interaction-aware and intention-dependent model.

### 1.3.8 Discussion

The presented literature gives an impression of the magnitude of different methods for solving the prediction and intention estimation problems. Unfortunately, there are multiple reasons why it is difficult to compare the performances of the single approaches with each other and thereby derive what could be called the state-of-the-art in motion prediction. One of the biggest problems is the lack of publicly available datasets that are both detailed enough and diverse enough. This also results in many approaches being tailored to specific scenarios and thus not being universally applicable. Such a dataset should contain relevant contextual information as the road geometry, road topology and traffic infrastructure (e.g., traffic lights, traffic signs, road markings) and contain both urban and highway environments with multiple interacting agents. Another problem is the lack of widely adopted, unified evaluation metrics, resulting in many works reporting different quantities to measure performance. The unavailability of the proposed methods’ source code often forbids the exact re-implementation of the algorithms and thus

complicates a fair comparison. Nevertheless, each of the distinct approaches does have its specific advantages and disadvantages, and there are some general conclusions that one can draw:

Solely estimating high-level intentions and not considering how the continuous state of an agent will change over time allows for a utilization in risk assessment, in assistance systems (e.g., to warn a human driver of upcoming collisions), or for high-level maneuver planning of autonomous vehicles. However, high-level intentions alone do not suffice as an input for continuous motion planning algorithms. The prediction of continuous trajectories is needed to plan collision-free and foresighted trajectories. Tackling both problems in a combined fashion has the benefits of improved interpretability, allows to subdivide the prediction problem into manageable subproblems, and can help to improve overall prediction performance.

*Physics-based* prediction approaches neglect the contextual information and thus are only suited for short-term predictions. *Reachable set-based* prediction over-approximates the possible future occupancies of the considered agents and thus allows to give formal guarantees, but also typically results in conservative or defensive behavior of the ego-vehicle. *Rule-based* prediction methods are composed of various hand-tuned heuristics and belong to the simplest methods that allow to incorporate contextual information, but generally cannot compete with more advanced approaches in terms of prediction performance. *Planning-based* approaches assume humans are rational agents that optimize their actions to achieve some sort of objective. The corresponding cost functions can incorporate contextual information and can also be learned from data, e.g., using inverse reinforcement learning (IRL). Planning-based prediction generalizes well to unseen environments and does not suffer from compounding error. Thanks to the availability of high-end compute and huge amounts of data, *deep learning* methods have the potential to outperform traditional machine learning methods due to their higher complexity and hierarchical structure, allowing them to learn high-level features from data in an incremental manner, eliminating the need for feature engineering based on expert knowledge.

It can be distinguished between one-step prediction models that are iteratively applied to generate full trajectories (forward simulation) and approaches that output complete trajectories, i.e., multiple time steps at once. Although learning complete trajectories does not suffer as much from the compounding error problem, such multi-step models—unlike one-step models—cannot be integrated in sequential decision making frameworks such as MDPs to account for the interdependencies between the prediction of surrounding agents and the ego-vehicle motion planning. It has to be noted that multi-step models can generally still be employed for interactive ego-vehicle planning, e.g., as done by Sadigh et al. [115] with nested optimization. However, this is typically significantly more involved: The prediction has to take a potential ego-vehicle plan as input and is called iteratively during the interactive ego-vehicle planning process. This is computationally expensive and requires a prediction that is specifically designed for this purpose. Existing literature that predicts complete trajectories typically does not account for the future motion of an ego-vehicle and thus can only be utilized for the “historic system architecture” of *prediction first, then planning* (cf. Fig. 1.4).

Table 1.1

TRAJECTORY PREDICTION LITERATURE REVIEW							
Approach	Method	Environment	C	I	U	M	S
Schubert et al. [119]	physics-based	road			✓	✓	
Wiest et al. [154]	physics-based, ML	road			✓	✓	
Althoff and Magdici [3]	reachable-set	highway, urban	✓		(✓) <sup>1</sup>	(✓) <sup>1</sup>	✓
Koschi and Althoff [69]	reachable-set	highway, urban	✓		(✓) <sup>1</sup>	(✓) <sup>1</sup>	✓
Treiber et al. [142]	rule-based	car following	✓	✓			✓
von Eichhorn et al. [38]	planning	urban	✓			✓	
Redher and Kloeden [108]	planning, ML	pedestrian	✓		✓	✓	✓
Rehder et al. [109]	planning, DL	pedestrian	✓		✓	✓	✓
Ding and Shen [35]	planning, DL	urban	✓	✓			
Ziebart et al. [159]	planning, IRL	pedestrian	✓		✓	✓	✓
Gonzalez et al. [46]	planning, IRL	highway	✓	(✓) <sup>2</sup>	✓	✓	✓
Sadigh et al. [115]	planning, IRL	highway, urban	✓	✓			
Ma et al. [89]	planning, DL, IRL	pedestrian	✓	(✓) <sup>2</sup>	✓	✓	✓
Armand et al. [7]	ML (GP)	intersection	✓		✓		✓
Tran et al. [140]	ML (GP)	intersections	✓		✓	✓	✓
Platho et al. [103]	ML (BN, RF)	intersections	✓	(✓) <sup>3</sup>			
Bahram et al. [11]	ML (BN), planning	highway	✓	(✓) <sup>2</sup>	✓		✓
Gindele et al. [44]	ML (DBN, RF)	intersections	✓	✓	✓	✓	✓
Wheeler et al. [153]	ML (BN, GM, RM)	highway	✓	✓	✓	✓	✓
Sarkar et al. [116]	DL (FC)	intersections	✓	✓			✓
Lenz et al. [82]	DL (FC, LSTM)	highway	✓	✓	✓	✓	✓
Pool et al. [105]	DL (GRU)	bicyclist	✓	(✓) <sup>3</sup>	✓		✓
Park et al. [98]	DL (seq2seq)	highway	✓	(✓) <sup>3</sup>		✓	✓
Deo and Trivedi [33]	DL (seq2seq)	highway	✓	(✓) <sup>3</sup>	✓	✓	
Hu et al. [60]	DL (CVAE)	urban	✓	(✓) <sup>7</sup>	✓	✓	
Lee et al. [79]	DL (CVAE)	urban	✓	(✓) <sup>6</sup>	✓	✓	
Diehl et al. [34]	DL (graph NN)	highway	✓	(✓) <sup>4</sup>			✓
Djuric et al. [36]	DL (top-down image)	urban	✓	(✓) <sup>3</sup>	✓		
Cui et al. [27]	DL (top-down image)	urban	✓	(✓) <sup>3</sup>	✓	✓	
Mänttari and Folkesson [90]	DL (top-down image)	intersection	✓	✓	✓	✓	
Hoermann et al. [57]	DL (top-down image)	urban	✓	(✓) <sup>5</sup>	✓	✓	
Alahi et al. [2]	DL (social pooling)	pedestrian	✓	(✓) <sup>3</sup>	✓		✓
Deo and Trivedi [32]	DL (social pooling)	highway	✓	(✓) <sup>3</sup>	✓	✓	
Zhao et al. [158]	DL (social pooling)	highway, ped.	✓	(✓) <sup>3</sup>	✓	✓	
Gupta et al. [50]	DL (social pooling)	pedestrian	✓	✓	✓	✓	
Vemula et al. [147]	DL (attention)	pedestrian	✓	✓	✓		✓
Messaoud et al. [92]	DL (attention)	highway	✓	(✓) <sup>3</sup>	✓		
Sadeghian et al. [112]	DL (attention)	pedestrian	✓	✓	✓	✓	
Kuefler et al. [74]	DL (GAIL)	highway	✓	✓	✓	✓	✓
Bhattacharyya et al. [22]	DL (GAIL)	highway	✓	✓	✓	✓	✓
Si et al. [121]	DL (GAIL)	highway	✓	✓			✓
This thesis	rule-based / DL	urban	✓	✓	✓	✓	✓

Legend: Context-dependency, Interaction-awareness, Uncertainty, Multi-modality, Single-step model

1) not a distribution but reachable-set

2) not combinatorial: features depend on complete distribution / possible occupancy of other agents

3) not combinatorial: prediction based on features of current time only

4) not combinatorial: no uncertainty considered

5) not combinatorial: occupancy distribution predicted

6) non-interactive sample generation, interaction-aware ranking and refinement

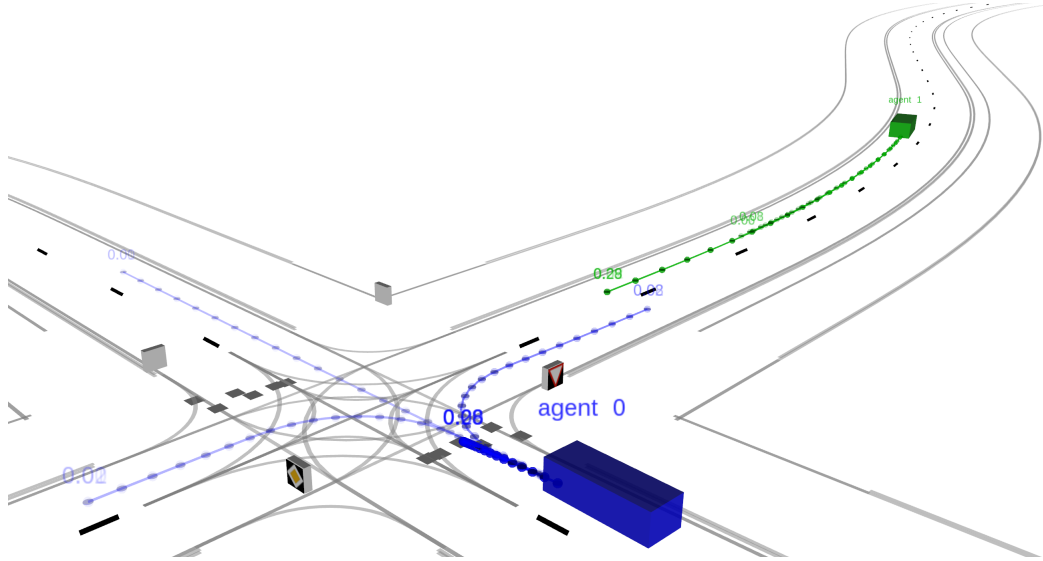
7) interaction only between vehicle pairs

By analyzing the large number of available publications, a conclusion can be drawn about important attributes that the prediction method of an autonomous vehicle should have: A full-fledged prediction approach should consider relevant contextual information, be universally applicable (i.e., not tailored to a specific situation), account for the interaction between multiple agents, model the uncertainty of human behavior, model the multi-modality of possible future trajectories, and be utilizable for interactive ego-vehicle motion planning. These characteristics can generally be achieved by a variety of different methods, no matter whether they are rule-based, planning-based, machine learning-based, or a combination thereof. An overview of which of the aforementioned criteria are met by the literature reviewed in this chapter can be found in Tab. 1.1.

## 1.4 Problem Statement

A traffic scene  $S$  consists of a set of  $K$  agents  $\mathcal{V} = \{V^1, \dots, V^K\}$ , with  $K \in \mathbb{N}$ , in an environment which is fully described by a given map. This map consists of a road network with topological, geometric and infrastructure information (yield lines, traffic signs, traffic lights, etc.) as well as the prevailing traffic rules. The state of traffic lights which is considered to be part of the map is updated in each time step. We assume discrete time steps, a continuous state space, and a continuous action space. At time step  $t$ , the set of agents  $\mathcal{V}$  is represented by their kinematic states  $X_t = [\mathbf{x}_t^1, \dots, \mathbf{x}_t^K]^\top$ , their route intentions  $R_t = [r_t^1, \dots, r_t^K]^\top$ , and their maneuver intentions  $M_t = [m_t^1, \dots, m_t^K]^\top$ . The kinematic state  $\mathbf{x}_t^i = [x_t^i, y_t^i, \theta_t^i, v_t^i]^\top$  of agent  $V^i$  consists of the Cartesian position, the heading or yaw-angle, and the velocity in driving direction. Each agent is approximated with a bounding box of rectangular shape, for which the position  $(x, y)$  defines the front center. The scene is projected to the  $xy$ -plane in order to simplify the problem. The agents' lengths and widths are considered to be given deterministically by the most recent measurement and, for the sake of brevity, are not included within  $\mathbf{x}^i$ . The route intention  $r_t^i$  defines a path through the road network the agent desires to follow. It is given by a sequence of lane segments that are topologically connected according to the map and thus can for example represent the exit an agent desires to take at an intersection or whether it is going to change from one lane to another. The maneuver intention  $m_t^i$  defines the desired order relative to other agents in cases of intersecting or merging routes where the agents' lanes overlap (see Sec. 2.4 for detailed definitions). Other types of maneuvers such as overtaking are not explicitly considered within this work.

At each time step, each agent executes an action  $\mathbf{a}_t^i$  that depends on its intentions, the map and the kinematic states of all agents, transforming the current kinematic state  $\mathbf{x}_t^i$  to a new state  $\mathbf{x}_{t+1}^i$ . The actions of all agents are denoted as  $A = [\mathbf{a}_t^1, \dots, \mathbf{a}_t^K]^\top$ . The complete dynamic part of a scene is thus described by  $S_t = [X_t, R_t, M_t, A_t]^\top$ . At each time step, a noisy measurement  $Z_t = [z_t^1, \dots, z_t^K]^\top$  with  $z_t^i = [x_{z,t}^i, y_{z,t}^i, \theta_{z,t}^i, v_{z,t}^i]^\top$  containing information about the kinematic states of all agents is received, which is distributed according to the measurement distribution  $p(Z_t|X_t)$ . The number of agents,



**Figure 1.14.** Typical situation with two interacting vehicles at an unsignalized intersection. Agent 0 (blue) has multiple high-level options: it can either turn left, right, or go straight and it can either merge or cross before or after agent 1 (green). The circles in the trajectories represent discrete time steps with  $\Delta T = 0.2$  s, the numbers show the probabilities of the respective combinations of intentions (e.g., agent 1 goes straight and agent 2 turns left before agent 1).

possible routes and maneuvers is arbitrary and may change over time: agents may appear or disappear and their possible routes and maneuvers need to be adapted accordingly.

The objective of this work is twofold: one part is to estimate the route intentions  $R$  and maneuver intentions  $M$  of all agents at the current time, mathematically  $p(R_t, M_t | Z_{0:t})$ , the other part is to predict the future kinematic states  $X$  up to an arbitrary temporal horizon  $T$  with step size  $\Delta T$ , mathematically  $p(X_{t:t+T} | Z_{0:t})$ , as exemplarily depicted in Fig. 1.14.

## 1.5 Contributions

This thesis proposes a trajectory prediction and intention estimation approach for autonomous vehicles that considers relevant contextual information, is universally applicable (not tailored to a specific situation), accounts for the interaction between multiple agents, models the uncertainty of measurement and human behavior, models the multimodality of possible future trajectories, and is utilizable for interactive ego-vehicle motion planning. The focus of the presented approach is on vehicles that drive guided by lanes such as cars, trucks, motorbikes, and potentially cyclists. The main contributions of this thesis are:

- Proposing a **hierarchical decision making process** of an agent, based on two types of discrete high-level intentions and a continuous low-level action. The intentions are based on the road topology, physical constraints, traffic rules and the

concept of trajectory homotopy and allow for a subdivision of the space of possible predictions into interpretable, logical statements. They are automatically generated at runtime given the current situation, removing the need to predefine a fixed set of intentions. An arbitrary number of intentions and arbitrary road layouts can be represented.

- Presenting a **combined approach** for driver **intention estimation** and interaction-aware **trajectory prediction** by modeling the interdependent behavior of multiple agents (based on the aforementioned decision making process) in a dynamic Bayesian network (DBN) that represents the evolution of the complete scene as a stochastic process. Intentions are estimated and trajectories are predicted using the same generative model. Cyclic dependencies between agents are dissolved by allowing independent actions for small time steps. Complete combinatorial scene developments including an arbitrary number of interacting agents are generated using incremental forward simulation up to arbitrarily long prediction horizons (thus also applicable as traffic simulator). It is accounted for uncertainty in measurements and human behavior, including the multi-modality of future trajectories.
- Proposing a context-dependent and probabilistic **rule-based behavior model** for urban environments, describing how humans act given a specific high-level intention and the current situation. In contrast to common existing rule-based models such as the intelligent driver model (IDM) that are typically only suited for a very specific type of scenario such as car-following, our model is able to cope with the complexity of urban scenarios by accounting for the influence of the road geometry, the traffic infrastructure, the prevailing traffic rules, and the interactions with surrounding agents. Furthermore, it is able to account for the uncertainty in human behavior.
- Proposing **deep learning-based behavior models** for urban environments that, besides learning the context-dependent behavior of traffic participants, are also able to learn the context-dependent uncertainty in a driver’s actions. Furthermore, the models are able to pick up subtleties in driver behavior which are difficult to model by hand such as drivers cutting curves. By embedding this deep neural network into the aforementioned DBN, we demonstrate how modern deep learning approaches can be combined with classical Bayesian inference and planning-based approaches.
- Presenting a sequential Monte Carlo (SMC)-based and a multiple model unscented Kalman filter (MM-UKF)-based **inference method** for the aforementioned DBN, demonstrating the trade-offs between flexibility, accuracy and runtime.
- Introducing **interactive ego-vehicle planning** methods that embed the presented prediction model, allowing to account for the interdependency of ego-vehicle motion planning and the prediction of surrounding agents. The presented approaches enable an autonomous vehicle to leverage the effects of its own actions on the behavior of other agents and thus to drive in a less conservative manner.

## 1.6 Outline

The remainder of this thesis is divided into six chapters, which are briefly described in the following.

*Chapter 2 – Combined Intention Estimation and Trajectory Prediction:* In this chapter, the main framework of the thesis is presented, formulating the Bayesian model used for both intention estimation and trajectory prediction. Different high-level intentions are introduced and included in the decision making process of an agent, allowing to explicitly represent the main multi-modalities of the predicted belief. Furthermore, this chapter addresses the interdependencies of the prediction of multiple interacting agents by breaking up the cyclic dependencies between agents: We utilize incremental forward simulation of small time steps for which actions of multiple agents can be assumed independent. Mutual influence is still represented by anticipatory behavior models and by updating the scene state after each forward simulation step.

*Chapter 3 – Behavior Models:* This chapter takes a detailed look at the relevant situational context (e.g., road geometry, traffic signs, etc.) and how that context in combination with a high-level intention influences the continuous action of a driver. Two distinct behavior models are introduced, one rule-based model that is defined using expert knowledge, and one machine learning-based model that employs different deep neural network architectures. Both models can easily be plugged into the DBN from Chapter 2, thus completing the prediction framework.

*Chapter 4 – Inference Methods:* In this chapter, two inference methods for solving the recursive Bayesian estimation and prediction problem defined in Chapter 2 are presented. The first method is based on sequential Monte Carlo (SMC) and thus can represent arbitrary distributions and non-linear system dynamics, but comes with high computational complexity. The second method is based on a multiple model unscented Kalman filter (MM-UKF) which uses a Gaussian approximation of the belief while still being able to represent the non-linear system dynamics by utilizing a smaller number of deliberately chosen sample points (so-called sigma points). Lastly, the chapter analyzes the combinatorial complexity of the problem domain and proposes a method to find a reasonable compromise between runtime and accuracy by only considering the most likely hypotheses without the need of tracking all possible combinations.

*Chapter 5 – Interrelated Ego Motion Planning and Prediction of Surrounding Agents:* This chapter closes the loop of prediction and planning, proposing different methods to handle the interdependency of ego-vehicle motion planning and estimation and prediction of other agents. It shows how the ego-vehicle can account for its own influence on the future actions of other agents and thus to drive less conservatively. The first proposed method is based on a partially observable Markov decision process (POMDP) whereas the second one is based on cooperative multi-agent planning.



*Chapter 6 – Experiments:* This chapter first explains the experiment setup and then evaluates the proposed prediction framework and the interaction-aware planning methods. The importance of the situational context and of accounting for interdependencies is analyzed utilizing the rule-based behavior model having full access to all extracted features and comparing it to a physics-based and a map-based baseline model. Furthermore, the deep learning-based model is evaluated and compared to the rule-based baseline, showing its capability of picking up subtle driver behavior while having problems with the compounding error for longer prediction horizons. We also demonstrate both inference methods and analyze how much runtime we can gain using the Gaussian assumption of the MM-UKF method compared to the SMC method. Furthermore, it is shown how the complexity reduction method significantly reduces tracking runtime and allows to derive the most likely hypotheses used for prediction to be able to focus computational efforts on high likelihood outcomes.

*Chapter 7 – Discussion:* In this chapter, the general approach of the thesis is discussed while referring to the results of the evaluation. We analyze the assumptions made, the advantages and disadvantages of the proposed methods, and state potential improvements. The main conclusions are drawn and ideas for future work are proposed.



## 2 Combined Intention Estimation and Trajectory Prediction

As the problems of driver intention estimation and trajectory prediction are highly coupled and only reflect different abstractions of a driver’s decision making process, this thesis proposes to solve them in a combined manner instead of having two completely distinct models or approaches. In this work, the development of a traffic situation is modeled as a stochastic process consisting of multiple interacting agents. Each agent can execute actions at every time step, thus transforming its own state over time. The actions of an agent are modeled to be dependent on internal intentions and the current context of the traffic scene including road infrastructure (traffic signs, traffic lights, etc.), road geometry, road topology, traffic rules, and other agents. Updating the belief over the current situation using incoming measurements allows to infer hidden variables such as the agents’ intentions. Iteratively applying the probabilistic transition models of all agents results in a probabilistic and interaction-aware multi-agent trajectory prediction. This chapter presents the overall prediction framework of this thesis. We address how the complete traffic situation is modeled jointly in the same state space, how the decision making process of a single agent is constructed, and how we handle the interdependencies between multiple agents. This chapter is based on the author’s previous publications [166] and [167].

### 2.1 Modeling the Development of a Traffic Situation in a Dynamic Bayesian Network

We describe the development of a traffic situation as a Markov process consisting of multiple interacting agents. Modeling this process in a dynamic Bayesian network (DBN) allows to explicitly specify relations between agents, define causal as well as temporal dependencies, and handle the uncertainty of measurements and human behavior. Furthermore, DBNs allow the definition of domain specific hierarchies within the decision making process of human drivers and the modeling of interdependencies between multiple agents. Dynamic Bayesian networks are directed, acyclic graphical models that define how random variables of a causal process relate with each other over the different steps of a sequence. For a temporal sequence that follows the Markov assumption, a DBN is given by a two-time-slice Bayesian network, such that the arcs within one slice represent “instantaneous” correlation between variables and the inter-slice arcs the temporal dependencies [95]. DBNs are a generalization of hidden Markov models (HMMs) by allowing an arbitrary number of continuous and discrete random variables (instead of

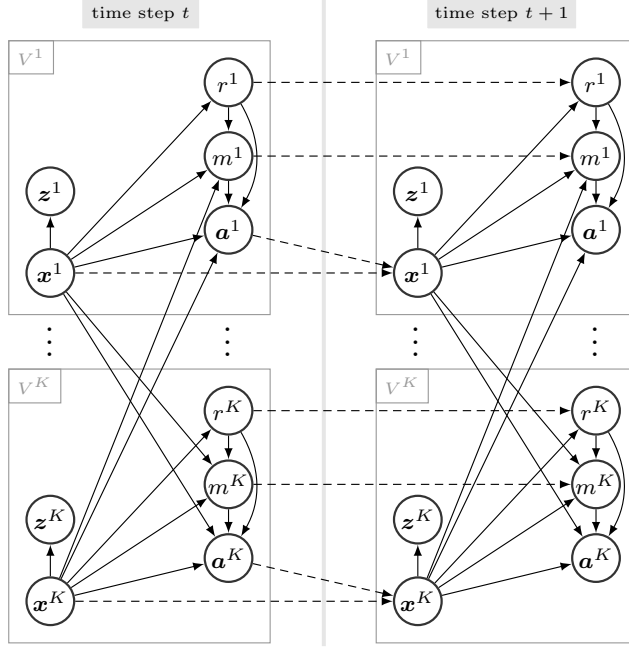
a single discrete random variable) and a generalization of Kalman filters by not restricting the process to be unimodal linear-Gaussian. Thus, DBNs have a greater “expressive power” [95]. They are well suited for modeling time-series, as time flows forward and dependencies can thus be modeled with directed arcs. Therefore, inference can be done sequentially for one time step after the other, meaning that it falls into the category of recursive Bayesian estimation. For further information about DBNs, we refer to [30].

Consider a set of  $K$  agents  $\mathcal{V} = \{V^1, \dots, V^K\}$  represented in a DBN as depicted in Fig. 2.1, showing the dependencies of the random variables of all agents. Each agent is represented by a bounding box with a kinematic state  $\mathbf{x} = [x, y, \theta, v]^\top$  consisting of the Cartesian position  $(x, y)$  located at the front center of the rectangle, its heading  $\theta$ , and its longitudinal velocity  $v$ . For the sake of brevity, the length and width of the agents are not included in the state, but are assumed to be given deterministically with the most recent measurement. As the slopes in the map used for evaluation are only minor, all agents in a scene and the corresponding map are projected to the 2D plane of  $(x, y)$ , neglecting differences in height in order to simplify the problem. The decision making process of each agent is composed of three hierarchical layers: the route intention  $r$ , i.e., the path defined on the basis of lanes the agent desires to follow, the maneuver intention  $m$ , i.e., whether it is going to pass a conflict area at an intersection before or after another agent, and the continuous action  $\mathbf{a}$  (see Sec. 2.4 for detailed definitions). Furthermore, measurements  $\mathbf{z}$  that depend on the kinematic state of the agent can be received in each time step. One time slice of the DBN thus consists of the dynamic states  $X = [\mathbf{x}^1, \dots, \mathbf{x}^K]$ , the route intentions  $R = [r^1, \dots, r^K]$ , the maneuver intentions  $M = [m^1, \dots, m^K]$ , the actions  $A = [\mathbf{a}^1, \dots, \mathbf{a}^K]$ , and potential measurements  $Z = [\mathbf{z}^1, \dots, \mathbf{z}^K]$  of all agents. The upper case symbols represent the vectorized concatenation of the single agent variables denoted in lower case letters. The complete scene state is denoted as  $S = [X, R, M, A]$ .

The continuous action of each agent is conditioned on its route and maneuver intentions, the states of all agents, and on the current environment given by the map. The map is considered to be given deterministically and is not depicted in Fig. 2.1 for improved visualization. It contains both topological and geometric information, the traffic infrastructure (stop lines, traffic signs, traffic lights, etc.), and the traffic rules. It is updated (e.g., the states of traffic lights) according to an environment model in each time step, but is considered to be constant during prediction. The random variables of route intentions, maneuver intentions, and actions depend on this map.

Given the kinematic states of all agents, first the set of possible routes and maneuvers is determined according to the road topology and geometry. Each agent’s continuous action is then derived using context-dependent behavior models given its route and maneuver intentions. These actions together with a corresponding motion model allow to predict the belief of the DBN one time step into the future. Alternatingly predicting this belief one time step ahead and then updating it with observations of the agents’ poses and velocities enables the drivers’ intentions and states to be inferred (so-called *filtering*). As DBNs are generative models, i.e., they can generate values of any of their random variables given their parents, it is also possible to iteratively predict the current belief *multiple steps* into the future (so-called *probabilistic forward simulation*) applying the same prediction model as within the filtering process. By starting this forward

## 2.1 Modeling the Development of a Traffic Situation in a Dynamic Bayesian Network



**Figure 2.1.** DBN showing the interdependencies between agents. Random variables are drawn as circles, causal and temporal dependencies as solid and dashed arrows, respectively. The random variables  $r$ ,  $m$ , and  $\mathbf{a}$  also depend on the map, which is not shown in order to improve visualization. Previously published in [166]. © IEEE 2018

simulation from the estimated belief of the current state, the predicted trajectories are weighted according to the probability estimates of the drivers' intentions. As the forward simulation is done one time step at a time for all agents in the state space, it is possible to respect the *current and future* interdependencies between all agents.

In order to use this recursive Bayesian estimation scheme, the conditional probability distributions of all variables given their parents have to be defined. For the sake of brevity, the superscript  $(\cdot)^i$  denoting the single agent is omitted in the following (e.g.,  $\mathbf{x}$  instead of  $\mathbf{x}^i$ ; capital letters such as  $X$  still denote the set of all agents). The following conditional probability distributions are introduced and explained in detail in the remainder of this chapter:

$$\begin{aligned}
 p(\mathbf{x}_0) &= f_{\mathbf{x}_0}, & t &= 0 \\
 p(\mathbf{z}_t|\mathbf{x}_t) &= f_{\mathbf{z}_0} = f_{\mathbf{z}}, & t &= 0, 1, \dots \\
 p(r_0|\mathbf{x}_0, \text{map}) &= f_{r_0}, & t &= 0 \\
 p(m_0|X_0, r_0, \text{map}) &= f_{m_0}, & t &= 0 \\
 p(\mathbf{a}_t|X_t, r_t, m_t, \text{map}) &= f_{\mathbf{a}_0} = f_{\mathbf{a}}, & t &= 0, 1, \dots \\
 p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{a}_{t-1}) &= f_{\mathbf{x}}, & t &= 1, 2, \dots \\
 p(r_t|r_{t-1}, \mathbf{x}_t, \text{map}) &= f_r, & t &= 1, 2, \dots \\
 p(m_t|m_{t-1}, X_t, r_t, \text{map}) &= f_m, & t &= 1, 2, \dots
 \end{aligned}$$

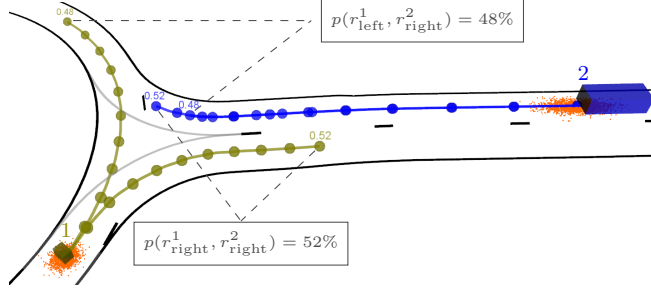
The functions with subscript  $(\cdot)_0$  are initialization functions and only apply for the time step of first detection of an object, the functions without subscript for all subsequent time steps. Although an agent might be added to the state space on later time steps ( $t > 0$ ), the initialization functions are still denoted here with  $t = 0$  for improved readability.

In order to account for changing situations, the network structure is adapted online (creating and deleting agents as well as route and maneuver hypotheses). Thus, it can be applied to varying situations with an arbitrary number of agents, intention hypotheses and different road layouts. Object-oriented DBNs are employed and parameters between multiple agents are shared.

### 2.2 Interdependencies Between Multiple Agents

The trajectories of multiple agents in a scene are generally interdependent. By including all agents within the same state space, it is possible to express interdependencies, account for interaction, and thus the combinatorial aspect of motion prediction in multi-agent environments. For that reason, this thesis proposes to formulate the decision making processes of all agents in a single DBN.

A model that predicts all agents in a combined manner multiple time steps into the future *at once* while accounting for these interdependencies is hard to formulate. Another possibility is to dissolve the interdependencies between agents by predicting them *independently* for a single time step: As can be seen in Fig. 2.1, the action of each agent at a specific time step is modeled to be independent of the actions or intentions of the other agents of the same time step given their kinematic states and its own intentions. Thus, cyclic dependencies between agents are avoided and one prediction step from  $t$  to  $t + 1$  of an agent can be executed independently of the prediction steps of other agents. For small time steps, this is a mild assumption, as one can assume that drivers don't know the future actions of other drivers when deciding for their own actions. In addition to the action models being able to react to the current context, they can also be *anticipatory*, i.e., make implicit predictions on what is going to happen and consider this for deriving what a good interaction strategy might be. Thus, even the current action might already take into account what other agents are likely going to do (but not what the forward simulation will actually predict them to do, as the actions of the different agents are derived independently). How such action models can be defined is addressed in Chapter 3, where we present one rule-based and one deep learning-based model. The rule-based model is anticipatory in the sense that it assumes likely behavior of others (e.g., time of arrival at lane merge position) and the deep learning-based model is implicitly anticipatory as it is based on how humans anticipate how a scene evolves in order to decide on their own actions. After each time step, the state of the complete situation is updated with all agents' actions such that for the decision making process of the subsequent time step, the context is up-to-date again and all agents observe how other agents actually have behaved (given the predicted forward simulation). By iteratively predicting all agents from one time step to another, it is possible to account for interaction gradually: As the current context depends on other agents' past actions,



**Figure 2.2.** Possible combinations of routes  $[r_{\text{left}}^1, r_{\text{right}}^2]$  and  $[r_{\text{right}}^1, r_{\text{right}}^2]$  at a roundabout: Although  $V^2$  is currently not influenced by  $V^1$ , it has to slow down in the future if  $V^1$  stays inside the roundabout, but can go faster in case  $V^1$  leaves the roundabout. This influence has to be taken into account for the trajectory prediction using forward simulation. Previously published in [166]. © IEEE 2018

an interdependency between their trajectories emerges *over time*, as shown in Fig. 2.2. Thus, when simulating the scene forward, long term interactions can be represented.

## 2.3 Measurement Model

High-level cuboid objects are used as measurements for the DBN. These measurements contain the kinematic states and the widths and lengths of the objects. The dimensions of the agents are considered to be given deterministically with the most recent measurement and are not included in the state or measurement vectors for the sake of brevity. Object detection and tracking given low-level sensor measurements is handled by a separate algorithm and is considered to be given within this work. Hence, low-level sensor specifics are abstracted and a variety of different sensor types (such as lidar, radar, or camera) and tracking methods can be used in combination with this framework.

The measurement  $\mathbf{z}_t = [x_{z,t}, y_{z,t}, \theta_{z,t}, v_{z,t}]^\top$  is distributed according to

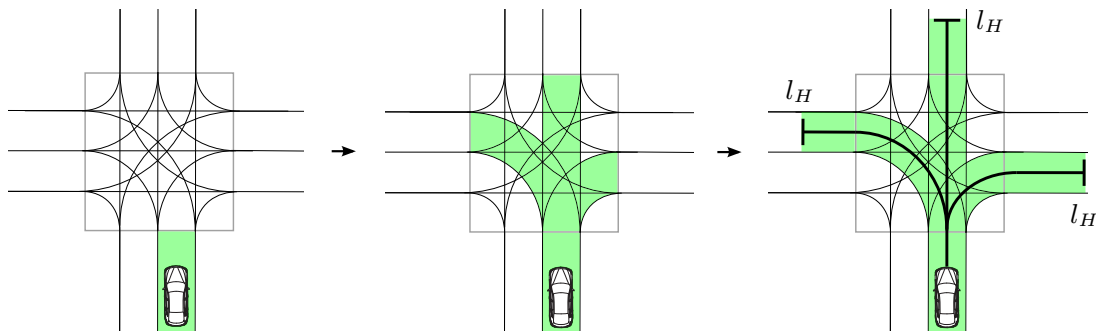
$$p(\mathbf{z}_t | \mathbf{x}_t) = f_{\mathbf{z}} = \mathcal{N}(\hat{\mathbf{z}}_t, \mathbf{R}_t) \quad (2.1)$$

with  $\hat{\mathbf{z}}_t = \mathbf{x}_t$  and  $\mathbf{R}_t = \text{diag}(\sigma_{x_z}^2, \sigma_{y_z}^2, \sigma_{\theta_z}^2, \sigma_{v_z}^2)$ . Thus, the kinematic state  $\mathbf{x}$  is measured with zero-mean Gaussian noise. For the initial time step of an agent (after it has been detected for the first time), no prior information about its kinematic state is assumed. This is achieved by an infinite variance of its kinematic state and an arbitrary mean (here 0):

$$p(\mathbf{x}_0) = f_{\mathbf{x}_0} = \mathcal{N}(\mathbf{0}, \text{diag}(\infty, \infty, \infty, \infty)) \quad (2.2)$$

Given the first measurement  $\mathbf{z}_0$ , the initial belief over the kinematic state is then given with Bayes' rule as

$$p(\mathbf{x}_0 | \mathbf{z}_0) = \frac{p(\mathbf{z}_0 | \mathbf{x}_0)p(\mathbf{x}_0)}{p(\mathbf{z}_0)} = \mathcal{N}(\mathbf{z}_0, \mathbf{R}_0). \quad (2.3)$$



**Figure 2.3.** Breadth-first search for possible routes of length  $l_H$  on the lane topology graph in a breadth-first manner. In this example, the route intention can be any of the three different paths through the intersection. Previously published in [166]. © IEEE 2018

## 2.4 Decision Making Process of an Agent

Incorporating high-level driver intentions into the decision making process of an agent allows to subdivide the space of possible predictions and to improve interpretability. The main cause for multi-modality in prediction in urban environments is given by the different possible decisions at intersections such as turning left or right or merging before or after another agent and by discrete lanes which agents desire to follow. Conditioning the prediction on such intentions allows to use simpler action models and approximate the predicted belief (given an intention) by a unimodal distribution. Unimodal distributions have the advantage that they can be approximated more easily by closed-form distributions such as Gaussians which allow more efficient calculations compared to general purpose particle distributions.

All agents are modeled to follow a decision making process that consists of three hierarchical layers: the route intention  $r$ , the maneuver intention  $m$ , and the continuous action  $\mathbf{a}$ . Each layer is conditioned on the layers of higher abstractions, such that the maneuver depends on the route and the action depends on the route and the maneuver. The action, in combination with a kinematics model, allows to transform the agent's current kinematic state to the subsequent time step. The uncertainty over possible future motion is thus divided into a high-level discrete intention uncertainty, and a low-level action uncertainty given a specific intention. The dependency graph of these random variables of the DBN is depicted in Fig. 2.1.

### 2.4.1 Route Intention

The route intention  $r \in \mathcal{R}$  defines the first layer of an agent's decision making process and serves as a path that guides its behavior. It is represented by a sequence of topologically connected lane segments. In every time step, the set of possible routes  $\mathcal{R}$  of an agent is determined given its pose and the topological map. We use breadth-first search on the lane graph starting from the current lane matching to find all possible hypotheses up to the metric horizon  $l_H$  as exemplarily depicted in Fig. 2.3. As each route has a different



geometry and may imply different traffic rules and relations to other agents, the route directly influences a driver's actions. In this thesis, the route of an agent mainly serves two purposes: Firstly, it allows to define and extract relevant features along an agent's planned path such as the upcoming road curvature or longitudinal distances to stop lines, utilized by the agent's action model (see Chapter 3). Secondly, the routes of multiple agents allow to build relationships between agents on complex road layouts. Two routes can be related with one another by dividing them into parts that either *merge*, *diverge*, *cross*, are *identical*, or have no relevant relation at all. Different road junction types such as roundabouts, intersections or highway entrances can thus be broken down into these types of relations, allowing for a better generalization of the action models. Typical features that describe the relations between agents consist of distances to merging or crossing areas of their routes and corresponding right of way rules.

Initially, the desired route  $r_0$  is distributed given the set of possible routes  $\mathcal{R}_0 = \{r_{0,1}, \dots, r_{0,|\mathcal{R}_0|}\}$  according to a prior distribution. In this work, a uniform prior

$$p(r_0|\mathbf{x}_0, \text{map}) = f_{r_0} = \mathcal{U}\{\mathcal{R}_0\} \quad (2.4)$$

or in different notation

$$p(r_{0,i}|\mathbf{x}_0, \text{map}) = |\mathcal{R}_0|^{-1}, \quad \forall r_{0,i} \in \mathcal{R}_0 \quad (2.5)$$

is assumed, but any other prior could be employed, in order to incorporate traffic statistics (such as it being more likely to stay on the main road, etc.).

Due to the fact that the route is only considered up to a specific horizon, in each new time step, a (potentially slightly) different set of possible routes may be determined. A binary matching function  $s_r(r_t, r_{t-1}) : \mathcal{R}_t \times \mathcal{R}_{t-1} \rightarrow \{0, 1\}$  is used to determine which of the routes  $r_{t,j} \in \mathcal{R}_t$  of the current time step are possible successors of a previous route hypothesis  $r_{t-1,k}$  (i.e., imply the same decisions at each topological split) and which are not. This allows to define the set of successor route hypotheses of a specific route hypothesis  $r_{t-1,k}$  as

$$\mathcal{R}_{t,\text{succ}(r_{t-1,k})} = \{r_{t,j} \mid \forall r_{t,j} \in \mathcal{R}_t : s_r(r_{t,j}, r_{t-1,k}) = 1\}. \quad (2.6)$$

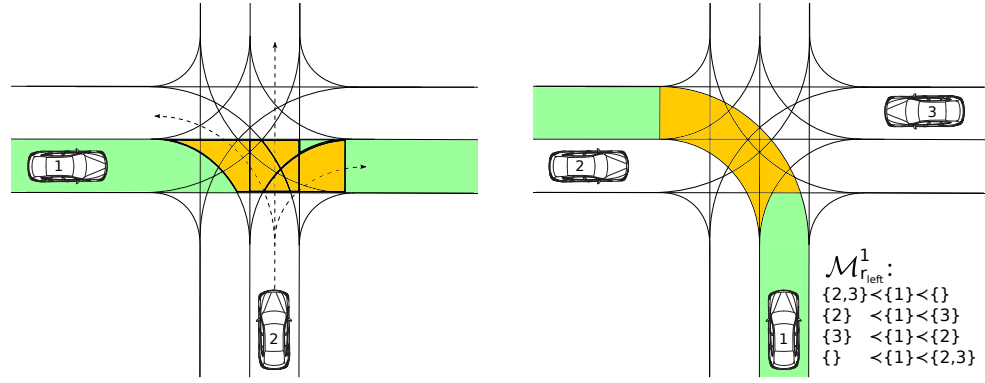
If there are multiple candidates ( $|\mathcal{R}_{t,\text{succ}(r_{t-1,k})}| > 1$ ), e.g., in case of the route horizon  $l_H$  reaching a topological split, again, the probabilities are distributed uniformly among them:

$$p(r_t|r_{t-1} = r_{t-1,k}, \mathbf{x}_t, \text{map}) = f_r = \mathcal{U}\{\mathcal{R}_{t,\text{succ}(r_{t-1,k})}\}, \quad (2.7)$$

or in different notation

$$p(r_{t,j}|r_{t-1} = r_{t-1,k}, \mathbf{x}_t, \text{map}) = \begin{cases} |\mathcal{R}_{t,\text{succ}(r_{t-1,k})}|^{-1}, & \text{for } r_{t,j} \in \mathcal{R}_{t,\text{succ}(r_{t-1,k})} \\ 0, & \text{for } r_{t,j} \notin \mathcal{R}_{t,\text{succ}(r_{t-1,k})} \end{cases}. \quad (2.8)$$

If there is no matching candidate ( $|\mathcal{R}_{t,\text{succ}(r_{t-1,k})}| = 0$ ), the agent has not followed the assumed route intention and this hypothesis is deleted. It is possible to include a (generally low) probability for switching the route to a non-matching candidate, corresponding to a driver changing their mind.



(a) Possible conflict areas from  $V^1$ 's perspective for going straight, resulting from the three route hypotheses of  $V^2$ . The actual route of  $V^2$  is unknown to  $V^1$ . (b) Four possible maneuvers for  $V^1$  turning left, representing the sequence of agents passing the conflict areas.

**Figure 2.4.** Conflict areas depicted in yellow and corresponding possible maneuvers. Previously published in [166]. © IEEE 2018

The set of possible routes is queried at runtime given a digital map, such that the intentions do not need to be predefined. As an input for the action model, each route intention is described distinctively from other possible routes with a variety of continuous features (such as the course of the centerline, the road curvature, longitudinal distances to traffic lights, etc.) rather than using one discrete value for, e.g., *turning left*. This enables the handling of an arbitrary and varying number of hypotheses, thus allowing for arbitrary road layouts.

### 2.4.2 Maneuver Intention

The maneuver intention  $m \in \mathcal{M}_r$  forms the second layer of the decision making process and describes the desired sequence in which agents are going to merge or cross at intersections. A maneuver is always defined given a specific route of the respective agent. Different routes may implicate the possibility of different maneuvers (e.g., for some routes, there might not be any conflicts whereas for another there might be multiple ones). For the definition of maneuvers, we introduce the notion of conflict areas: Given two agents on two routes, their conflict area is defined by the area at which both routes intersect, i.e., the area in which their lanes overlap. It is assumed that agents do not know which routes other agents are going to follow, thus, all possible conflict areas are considered (see Fig. 2.4(a)).

In order to avoid collisions at conflict areas, agents have to schedule their passing sequence. A maneuver of agent  $V^i$  states for all pairs  $\langle V^i, V^j \rangle$  that have a *potential* conflict (i.e., at least one route hypothesis of agent  $V^j$  has a conflict with  $V^i$ 's intended route), whether  $V^i$  will pass their conflict area first ( $V^i < V^j$ ) or not ( $V^i > V^j$ ). This definition is based on the concept of pseudo-homotopy of trajectories. Our evaluation data suggests

that vehicles that have right of way are typically only insignificantly influenced by other vehicles approaching the intersection. Thus, we only consider different maneuvers for vehicles that do not have right of way.

The set of possible maneuvers  $\mathcal{M}_r$  can be derived given the agent's route  $r$ , the map, and the kinematic states of all agents  $X$ . An example can be seen in Fig. 2.4(b). For the sake of brevity, the subscript  $(\cdot)_r$  referring to the corresponding route is omitted for the remainder of this section. The desired maneuver  $m_0$  is initially distributed according to a prior distribution among the set of possible maneuvers  $\mathcal{M}_0 = \{m_{0,1}, \dots, m_{0,|\mathcal{M}_0}|\}$ . In this work, a uniform prior

$$p(m_0|X_0, r_0, \text{map}) = f_{m_0} = \mathcal{U}\{\mathcal{M}_0\} \quad (2.9)$$

or in different notation

$$p(m_{0,i}|X_0, r_0, \text{map}) = |\mathcal{M}_0|^{-1}, \quad \forall m_{0,i} \in \mathcal{M}_0 \quad (2.10)$$

is assumed, but it is possible to learn the prior from data, considering quantities such as time gaps at the conflict areas. As situations change over time, the set of possible maneuvers may change as well (e.g., when a new agent is detected or when an existing agent has traversed a conflict area). Hence, for future time steps, a binary matching function  $s_m(m_t, m_{t-1}) : \mathcal{M}_t \times \mathcal{M}_{t-1} \rightarrow \{0, 1\}$  determines which of the new maneuvers  $m_{t,j} \in \mathcal{M}_t$  of the current time step are possible successors of a previous maneuver hypothesis  $m_{t-1,k}$  (i.e., there are no contradictory passing sequences):

$$\mathcal{M}_{t,\text{succ}(m_{t-1,k})} = \{m_{t,j} | \forall m_{t,j} \in \mathcal{M}_t : s_m(m_{t,j}, m_{t-1,k}) = 1\} \quad (2.11)$$

If there are multiple matching candidates (e.g., a new agent is detected which can either merge or cross before or after the predicted agent), again, the maneuver is sampled uniformly across candidates:

$$p(m_t | m_{t-1} = m_{t-1,k}, X_t, r_t, \text{map}) = f_m = \mathcal{U}\{\mathcal{M}_{t,\text{succ}(m_{t-1,k})}\}, \quad (2.12)$$

or in different notation

$$\begin{aligned} & p(m_{t,j} | m_{t-1} = m_{t-1,k}, X_t, r_t, \text{map}) \\ &= \begin{cases} |\mathcal{M}_{t,\text{succ}(m_{t-1,k})}|^{-1}, & \text{for } m_{t,j} \in \mathcal{M}_{t,\text{succ}(m_{t-1,k})} \\ 0, & \text{for } m_{t,j} \notin \mathcal{M}_{t,\text{succ}(m_{t-1,k})} \end{cases}. \end{aligned} \quad (2.13)$$

In order to allow for changing maneuver intentions, a small probability of switching to non-matching candidate hypotheses can be added. If there is no matching candidate, the maneuver intention became impossible, such that this hypothesis is removed.

### 2.4.3 Action

The action model forms the third layer of an agent's decision making process and states how an agent acts given its intentions and the current situation. The action  $\mathbf{a}$  thus

## 2 Combined Intention Estimation and Trajectory Prediction

depends on the agent’s intended route and maneuver, the kinematic states of all agents, and the map:  $p(\mathbf{a}^i | X, r^i, m^i, \text{map})$ . It should be noted that these actions are not control inputs to the system, but are modeled as random variables that are to be estimated, as they belong to the human controlled agents that are to be predicted. If, however, the ego-vehicle is included in the state space, the ego-vehicle’s action would serve as a control input, opening up the possibility of interaction-aware motion planning within the belief state of the DBN (see Chapter 5).

Within this dissertation, two different types of action models are proposed: a rule-based (or heuristics-based) action model that outputs a distribution over acceleration  $a$  and yaw-rate  $\dot{\theta}$  ( $\mathbf{a} = [a, \dot{\theta}]^\top$ ) (see Sec. 3.1), and a neural network-based action model that outputs a distribution over acceleration  $a$  and steering angle  $\delta$  ( $\mathbf{a} = [a, \delta]^\top$ ) (see Sec. 3.2). The rule-based model is coupled with a *rotate first, then translate* kinematics model, whereas the neural network-based model is coupled with a kinematic bicycle model. For the combination of an action and a kinematics model, the term *behavior model* is used, as it tells how an agent behaves given a specific situation and its intentions. Many different behavior models can be incorporated in the proposed framework. In order not to go beyond the scope of this chapter, the proposed rule-based and deep learning-based models are presented in Chapter 3.

### 3 Behavior Models

The term *behavior model* is widely used within the motion prediction literature. In this thesis, a behavior model describes how an agent acts given a specific situation and its route and maneuver intentions and how its state changes over time given this action. Thus, such a behavior model comprises an action model  $p(\mathbf{a}_t^i | X_t, r_t^i, m_t^i, \text{map})$  and a kinematics model  $p(\mathbf{x}_{t+1}^i | \mathbf{x}_t^i, \mathbf{a}_t^i)$ . The action model forms the third layer of an agent’s decision making process. It is the most complex, as it has to model the subtleties of human driving in a continuous manner.

Human drivers have individual and complex behavior characteristics which describe how they act in a specific situation. Driver behavior models are essential for many applications in the field of autonomous driving, ranging from microscopic traffic simulation, trajectory prediction and intention estimation, to interactive and cooperative motion planning. The need for detail and accuracy typically varies depending on the purpose of application. Models used for traffic flow analysis, for example, tend to be less detailed, as individual driver behavior patterns do not matter as much, whereas models used for estimating driver intentions, or for predicting how humans react to an autonomous vehicle, should be able to capture human behavior patterns in more detail. Although existing simple approaches such as the intelligent driver model (IDM) [142] are well suited for high-level traffic flow modeling or deterministic trajectory prediction in car following scenarios, they are neither capable of capturing the complex decision making process of human drivers in more diverse scenarios, nor of representing the uncertainty of the potential future behavior.

This chapter presents two different behavior models, one rule-based (or heuristics-based) action model coupled with a rotate-translate kinematics model, and a deep learning-based action model coupled with a kinematic bicycle model. The rule-based model serves as a baseline and implements anticipatory and reactive behavior and is able to represent basic driving principles in urban scenarios such as keeping distance to preceding agents, slowing down before curves, stopping for red lights, or merging into gaps. The deep learning-based model is trained on complex simulation agents and on real human driving data and is able to pick up more fine-grain subtleties such as drivers cutting curves and context-dependent variance, which are difficult to model by hand. These models allow the integration into various kinds of state-of-the-art algorithms such as forward simulation-based motion planning algorithms (e.g., Monte Carlo tree search (MCTS) [83], partially observable Markov decision processes (POMDPs) [162]), or intention estimation and trajectory prediction algorithms (e.g., dynamic Bayesian networks (DBNs) such as the one presented in Chapter 2). As the models we present in this chapter are intended to be integrated into sampling-based algorithms, the input to the models is deterministic (a sample of the belief), whereas the output is a probability

distribution over actions, from which one can again draw samples, if desired. Stepwise forward simulation of these models for the different possible route and maneuver intentions of all agents allows for multi-modal and interaction-aware scene predictions at arbitrary road layouts. This chapter is based on the author’s previous publications [166] and [168].

## 3.1 Rule-based Behavior Model

Rule-based or heuristics-based behavior models are models that capture how humans act based on traffic rules, traffic statistics, and domain knowledge. As they are designed and parameterized by domain experts, they typically have fewer parameters than data-driven models and are thus not able to capture the subtleties of human behavior as detailed. A big advantage of rule-based models is, however, that they remain interpretable, are quick to implement, don’t need lots of data, and usually generalize better to unseen scenarios.

### 3.1.1 Approach

In contrast to existing rule-based behavior models, this section proposes a model that can handle multiple influences such as speed limits, traffic lights, preceding agents, road curvature, or conflicts at intersections in a combined manner, allowing the utilization for prediction in urban scenarios. The model is conditioned on both the driver’s route and maneuver intentions, subdividing the prediction problem and reducing design complexity. Furthermore, the proposed model considers the uncertainty in human behavior and thus defines a probabilistic mapping from a driver’s intentions and the current situation to a distribution over actions.

The proposed rule-based behavior model comprises a set of influence-based acceleration submodels, each determining a *range of reasonable accelerations* given a specific influence (such as the road curvature). Besides applying the IDM for keeping appropriate distances to preceding vehicles, new heuristics-based models for stopping at stop signs and red traffic lights, slowing down before curvatures, and approaching gaps at intersections are proposed. These additional components successfully extend the usage of rule-based models to complex urban scenarios. In comparison to deep learning-based models, this model remains interpretable and extendable by additional influences and generalizes well to unseen situations. The action of each agent given the rule-based action model is defined as  $\mathbf{a} = [a, \dot{\theta}]^\top$  with the longitudinal acceleration  $a$  and the yaw rate  $\dot{\theta}$ .

### 3.1.2 Rotate-Translate Kinematics Model

The motion of an agent is described by a point mass model which couples  $x$  and  $y$  via the heading  $\theta$ . This model first rotates the agent according to the yaw-rate and then translates it along its new orientation according to the longitudinal velocity and acceleration. Gaussian noise is added to both the rotation and the translation parts, resulting in the typical “banana-shaped” uncertainty. The transition of the kinematic

**Table 3.1**INFLUENCES, FEATURES, AND ACTION RANGES FOR AGENT  $V^i$ . PREVIOUSLY PUBLISHED IN [166]

Influence	Features	Action Range
vehicle dynamics	-	$[a_{\text{vd}}^{\min}, a_{\text{vd}}^{\max}]$
speed limit	$d_{v_{\text{lim}}}, v_{\text{lim}}, v^i$	$[-\infty, a_{\text{IDM}}^{\max}]$
preceding agent $V^p$	$d^p, v^p, v^i$	$[-\infty, a_{\text{IDM}}^{\max}]$
red traffic light	$d_{\text{tl}}, v^i$	$[-\infty, a_{\text{tl}}^{\max}]$
stop sign	$d_{\text{stop}}, v^i$	$[-\infty, a_{\text{stop}}^{\max}]$
road curvature	$d_{\rho}, \rho, v^i$	$[-\infty, a_{\text{curve}}^{\max}]$
conflicting agent $V^c$	$\chi^{i,c}, d_{\text{entry}}^c, d_{\text{exit}}^c, v^c,$ $d_{\text{yield}}^i, d_{\text{entry}}^i, d_{\text{exit}}^i, v^i$	$[a_{\text{conf}}^{\min}, a_{\text{conf}}^{\max}]$

state given an action is described by the probability distribution  $p(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{a}_t)$  with the new state given as

$$\mathbf{x}_{t+1} = \begin{pmatrix} x_{t+1} \\ y_{t+1} \\ \theta_{t+1} \\ v_{t+1} \end{pmatrix} = \begin{pmatrix} x_t + v_t \Delta T \cos(\theta_{t+1}) + \frac{1}{2} a_t \Delta T^2 \cos(\theta_{t+1}) + w_x \\ y_t + v_t \Delta T \sin(\theta_{t+1}) + \frac{1}{2} a_t \Delta T^2 \sin(\theta_{t+1}) + w_y \\ \theta_t + \dot{\theta}_t \Delta T + w_\theta \\ v_t + a_t \Delta T + w_v \end{pmatrix}, \quad (3.1)$$

with the transition noise  $\mathbf{w} = [w_x, w_y, w_\theta, w_v]^\top \sim \mathcal{N}(\mathbf{0}, \mathbf{Q})$  and  $\mathbf{Q} = \text{diag}(\sigma_x^2, \sigma_y^2, \sigma_\theta^2, \sigma_v^2)$ . Although this model is simplistic, it is argued that it is sufficient for prediction purposes.

### 3.1.3 Action Model

The action  $\mathbf{a} = [a, \theta]$  of an agent depends on its route and maneuver intentions, the kinematic states of all agents, and the map. It forms the third layer of the decision making process. The acceleration model is a non-linear mapping from features to a Gaussian distribution  $p(a|r, m, X, \text{map}) = \mathcal{N}(\mu_a, \sigma_a^2)$ , where  $\mu_a$  is a function of  $(r, m, X, \text{map})$  and  $\sigma_a$  is constant. The set of features is derived given the current scene and the agent's intentions and describes the current context in the agent's perspective. In order to narrow down the large number of dependencies, a set of submodels is defined, each handling one so-called *influence*. Each influence consists of a subset of the available features and constrains the acceleration to a range  $[a^{\min}, a^{\max}]$  that is plausible (e.g., not leading to collisions or violations of traffic rules) given that specific influence. These ranges can either have a lower bound (dictating a minimum acceleration), an upper bound (dictating a maximum acceleration) or both. Tab. 3.1 shows the influences considered within this work for agent  $V^i$  with their corresponding features and action ranges. These influences represent the context on which an agent's actions are based on and are derived deterministically given the variables of the DBN.

The influence *vehicle dynamics* restricts the range of possible accelerations to the constant range  $[a_{\text{vd}}^{\min}, a_{\text{vd}}^{\max}]$  and reflects the maximum braking and acceleration capabilities of a vehicle. *Speed limits* are defined by a set of pairs of speed limit  $v_{\text{lim}}$  and distance

### 3 Behavior Models

along the route  $d_{v_{\text{lim}}}$  where they become effective. A *preceding agent*  $V^p$  is described by its relative distance  $d^p$  and its velocity  $v^p$ . For the currently active speed limit ( $d_{v_{\text{lim}}} \leq 0$ ) and the directly preceding agent, the IDM presented by Treiber et al. [142] is employed, dictating a maximum reasonable acceleration

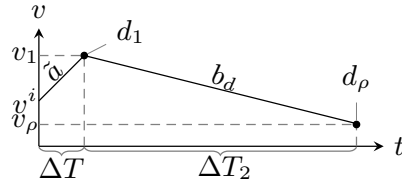
$$a_{\text{IDM}}^{\text{max}} = a_d \left( 1 - \left( \frac{v^i}{v_{\text{lim}}} \right)^\delta - \left( \frac{d_d + v^i T_d + \frac{v^i(v^i - v^p)}{2\sqrt{|a_d b_d|}}}{d^p} \right)^2 \right) \quad (3.2)$$

depending on the agent's velocity  $v^i$ , the speed limit  $v_{\text{lim}}$ , the preceding agent's velocity  $v^p$  and relative distance  $d^p$ . If there is no preceding agent, the distance  $d^p$  is set to infinity, resulting in the agent's velocity converging to the prevailing speed limit. The parameters minimum spacing  $d_d$ , desired time headway  $T_d$ , comfortable acceleration  $a_d$ , braking deceleration  $b_d$ , and acceleration exponent  $\delta$  have to be specified. The influences *red traffic light* and *stop sign* are also handled using the IDM formula from Eq. (3.2) by setting  $v^p = 0$  and  $d^p$  to the corresponding distance.

As the IDM was primarily designed for highway scenarios, it does neither consider the curvature of the road nor the existence of merging or intersecting lanes. Thus, we define the following models allowing for a prediction in urban scenarios: The model for the influence *curvature* is based on a desired maximum lateral acceleration  $a_{\text{lat}}^{\text{max}}$  that implies a maximum velocity  $v_\rho = \sqrt{\rho a_{\text{lat}}^{\text{max}}}$  at a given curve radius  $\rho$ . Given these constraints, we want to determine the maximum acceleration of  $V^i$  for the current time step such that these constraints will not be violated. This maximum acceleration for one time step  $\Delta T$  that still allows to reach the velocity  $v_\rho$  at the corresponding distance  $d_\rho$  with a comfortable braking deceleration  $b_d$  is then given by

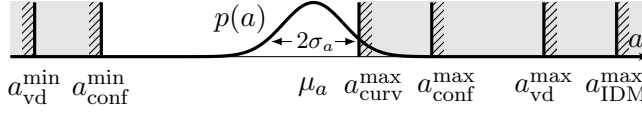
$$\tilde{a} := a_{v_\rho, d_\rho}^{\text{max}} = \frac{-2v^i + \Delta T b_d + \sqrt{4v^i \Delta T b_d + \Delta T^2 b_d^2 - 8b_d d_\rho + 4v_\rho^2}}{2\Delta T}, \quad (3.3)$$

which can be determined as follows: First, we assume a constant acceleration phase starting from the current velocity  $v^i$  with acceleration  $\tilde{a}$  (which we want to determine) for one time step  $\Delta T$ . Then, we assume a constant deceleration phase starting from the resulting velocity  $v_1$  with the comfortable braking deceleration  $b_d$  (which is the maximum we want to allow) for the time period of  $\Delta T_2$ , such that the agent exactly reaches the desired velocity  $v_\rho$  at distance  $d_\rho$ . This leads to the least constraining acceleration upper bound which still allows to fulfill the constraints. With the know variables  $(v^i, b_d, d_\rho, v_\rho, \Delta T)$  and the following equations, it is possible to derive this maximum possible acceleration  $\tilde{a}$  that the agent can perform:



$$\begin{aligned} v_1 &= v^i + \tilde{a}\Delta T \\ v_\rho &= v_1 + b_d\Delta T_2 \\ d_1 &= v^i \Delta T + \frac{1}{2}\tilde{a}\Delta T^2 \\ d_\rho &= d_1 + v_1\Delta T_2 + \frac{1}{2}b_d\Delta T_2^2 \end{aligned}$$





**Figure 3.1.** Example of possible upper and lower bounds of the acceleration submodels of the single influences, used to define the overall acceleration probability distribution. Previously published in [166] © IEEE 2018.

This is calculated for all curvature distance pairs along the route and the smallest allowed acceleration is used as an upper bound, as it is the most constraining one. This results in a foresighted curvature approach. Upcoming speed limits can also be approached in a foresighted way by utilizing the same formula from Eq. (3.3).

The *conflict* model is based on conflict areas at overlapping lanes where vehicles have to coordinate a specific sequence of passing. A conflict of agent  $V^i$  with another agent  $V^c$  is described by the right of way  $\chi^{i,c}$ , the agents' velocities, their distances to entering and exiting the conflict area  $d_{\text{entry}}$  and  $d_{\text{exit}}$ , and their distances to potential yield lines  $d_{\text{yield}}$ . If agent  $V^i$  has right of way, we assume that it is not influenced by the other agent ( $[a_{\text{conf}}^{\min}, a_{\text{conf}}^{\max}] = [-\infty, \infty]$ ). If agent  $V^i$  has to yield, it acts according to its desired maneuver  $m^i$ : Each agent that is going to pass before  $V^i$  introduces an upper bound of acceleration ( $a_{\text{conf}}^{\max}$ ), each agent that is going to pass after  $V^i$  introduces a lower bound ( $a_{\text{conf}}^{\min}$ ). These accelerations are determined such that a minimum time gap between the two passing vehicles at the overlapping areas is ensured, assuming  $V^i$  predicts the other agent  $V^c$  to drive with constant velocity.

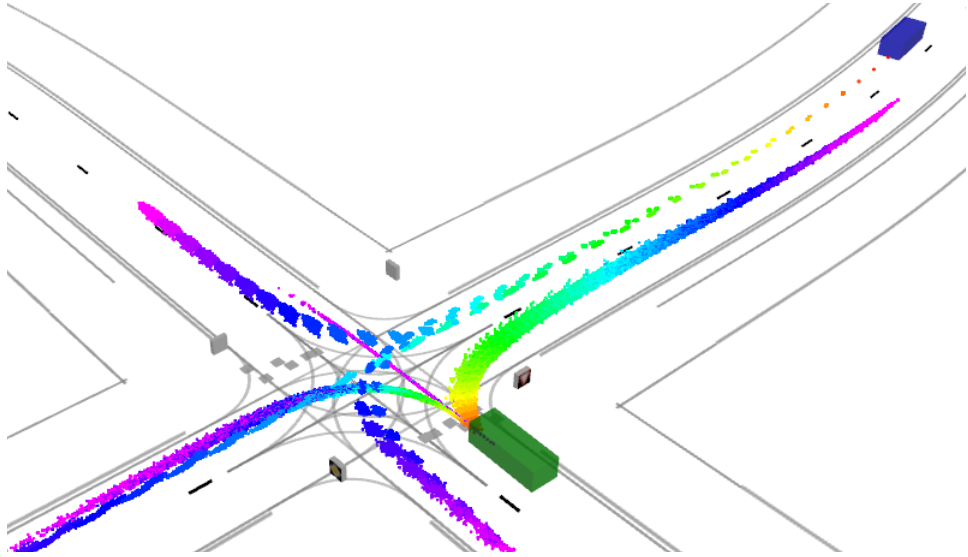
The ranges of feasible accelerations of the single influences are combined as shown in Fig. 3.1 to the overall range

$$a_{\max} = \min\{a_{\text{vd}}^{\max}, a_{\text{IDM}}^{\max}, a_{\text{tl}}^{\max}, a_{\text{stop}}^{\max}, a_{\text{curve}}^{\max}, a_{\text{conf}}^{\max}\}, \quad (3.4)$$

$$a_{\min} = \max\{a_{\text{vd}}^{\min}, a_{\text{conf}}^{\min}\}. \quad (3.5)$$

Our measurement data suggests that drivers tend to minimize driving time while not exceeding the plausible acceleration range. Thus, accelerations are sampled from the distribution  $p(a|r, m, X, \text{map}) = \mathcal{N}(\mu_a, \sigma_a^2)$ , with a mean close to the lowest maximum bound:  $\mu_a = a_{\max} - \sigma_a$  (see Fig. 3.1).

The yaw rate of an agent is distributed according to the Gaussian  $p(\dot{\theta}|r, \mathbf{x}, a, \text{map}) = \mathcal{N}(\mu_{\dot{\theta}}, \sigma_{\dot{\theta}}^2)$ , with constant variance  $\sigma_{\dot{\theta}}$  and a mean  $\mu_{\dot{\theta}}$  which is calculated based on simple heuristics such that it keeps the agent close to the center of its desired lane. Given the already calculated acceleration  $a$ , the distance the agent will travel in one time step is given by  $d_{\Delta T} = v\Delta T + \frac{1}{2}a\Delta T^2$ . The desired position the agent should reach is then calculated as the point on the centerline that is  $d_{\Delta T}$  further along the lane as the projection of the agent's current position onto the centerline. The mean yaw-rate is then calculated such that the agent's new orientation points to this desired position and the agent ends up in the close proximity of the centerline. Although this model is rather simplistic, it allows the agent to follow its desired lane given a reasonably small time step (in this thesis:  $\Delta T = 0.2$  s).



**Figure 3.2.** Learned probabilistic and interaction-aware driver behavior model based on a deep neural network, iteratively applied to generate possible scene predictions (colors indicate prediction horizon of up to 7 s). The green vehicle yields for the hypotheses of turning left and going straight, but is allowed to drive on for turning right. The variance in the driver’s actions highly depends on the situation, which can be seen in a higher lateral uncertainty for turning maneuvers. Previously published in [168]. © IEEE 2019

### 3.2 Deep Learning-based Behavior Model

Designing behavior models by hand is cumbersome and inaccurate, especially in urban environments, with their high variety of situations and the corresponding diversity in human behavior. In order to reduce the design complexity, one generally has to make assumptions which may be violated by real traffic participants. For example, many rule-based driver behavior models follow the concept of driving as fast as possible while accounting for the speed limit and ensuring reasonable safety distances, such that each agent will be able to avoid speeding and collisions with other agents. This is a strong assumption which does not always hold in real traffic, as many drivers do not maintain necessary safety distances or do violate speed limits from time to time. Rule-based models often consist of multiple interdependent parameters that are hard to tune by hand. Despite this perceived complexity, they are still typically not able to distinguish minor subtleties in human behavior such as how exactly drivers cut curves nor to account for the fact that the variance of human behavior is highly situation-dependent.

Learning how humans act from recorded scenarios is a promising way to overcome these problems and to achieve high accuracy models that are capable of representing subtle nuances of driver behavior. However, predicting complete trajectories at once is challenging, as one needs to account for multiple hypotheses and long-term interactions between multiple agents. Thus, this section proposes to learn a driver’s one-step behavior model from data to be integrated into the previously presented DBN, allow-

ing to probabilistically predict interaction-aware multi-agent trajectories, as depicted in Fig. 3.2: the learned model captures the context-dependent variance of a driver’s actions and accounts for the mutual influence of the different agents.

### 3.2.1 Approach

In contrast, to the presented related work in Chapter 1, this section proposes to learn probabilistic Markovian action models for urban environments with deep neural networks that are conditioned on a driver’s route intention (such as turning left or right) and the situational context. Learning to predict only one time step ahead given a specific route reduces learning complexity, such that simpler and faster models are obtained. This enables the integration into sampling-based algorithms for interactive motion planning (such as MCTS) or trajectory prediction and intention estimation (e.g., using particle filtering), which commonly still rely on simplistic hand-tuned models. Deep learning of behavior models thus allows for a combination of machine learning with planning-based and Bayesian algorithms. As the models might be called thousands of times per time step, the focus is on simple and fast models that are still able to accurately capture the variety that is present in urban scenarios.

### 3.2.2 Kinematic Bicycle Model

Instead of directly learning a state transition model, the kinematics of the agents are restricted to a kinematic bicycle model [68], such that the neural network only learns a two dimensional action distribution comprising acceleration  $a$  and steering angle  $\delta$ . This reduces learning complexity and enforces the nonholonomic vehicle constraints. The kinematic state transition is defined deterministically given these actions as

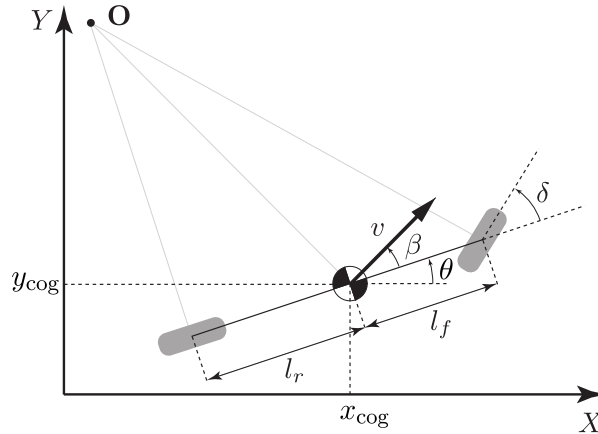
$$\dot{\mathbf{x}}_{\text{cog}} = \begin{pmatrix} \dot{x}_{\text{cog}} \\ \dot{y}_{\text{cog}} \\ \dot{\theta} \\ \dot{v} \end{pmatrix} = \begin{pmatrix} v \cos(\theta + \beta) \\ v \sin(\theta + \beta) \\ \frac{v}{l_r} \sin(\beta) \\ a_t \end{pmatrix}, \quad (3.6)$$

with  $\beta = \arctan\left(\frac{l_r}{l_f + l_r} \tan(\delta)\right)$ . The parameters  $l_f$  and  $l_r$  define the distances from center of gravity  $(x_{\text{cog}}, y_{\text{cog}})$  to the front and rear axis respectively, as depicted in Fig. 3.3. They are determined experimentally by minimizing the motion reconstruction error (see Sec. 3.2.3.1). The position of the front center of the vehicle  $(x, y)$  is determined with the respective transformations. In between two discrete time steps, it is assumed that the acceleration and the steering angle are kept constant.

### 3.2.3 Action Model

The action  $\mathbf{a} = [a, \delta]^\top$  is modeled to be normally distributed given a specific route intention  $r^i$  and the current situational context given by  $(X, \text{map})$  as

$$p(\mathbf{a}^i | r^i, X, \text{map}) = \mathcal{N} \left( \begin{bmatrix} \mu_a \\ \mu_\delta \end{bmatrix}, \begin{bmatrix} \sigma_a^2 & 0 \\ 0 & \sigma_\delta^2 \end{bmatrix} \right), \quad (3.7)$$

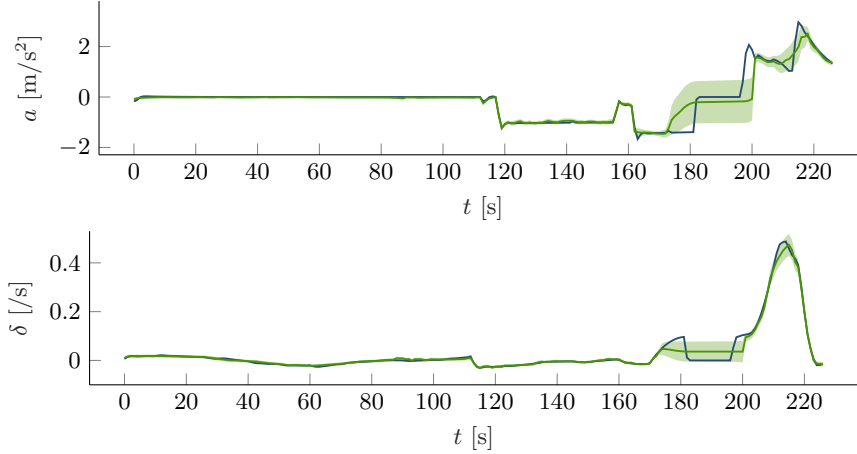


**Figure 3.3.** Kinematic bicycle model. Graphic taken and adapted from [68]. © IEEE 2015

whereas the agent’s kinematics are given by the kinematic bicycle model defined above. Given the uncertainty about the different possible route intentions and the uncertainty of the kinematic states, the overall predicted belief will still be multi-modal. The variance in driver behavior strongly depends on the situation. Thus, not only the mean of the actions are learned, but also the context-dependent or so-called *heteroscedastic variance*, which is difficult to model by hand. A validation plot depicting the learned mean and variance and the corresponding ground truth trajectory can be seen in Fig. 3.4, showing the strong context-dependency of the variance.

In contrast to the rule-based model, this deep learning-based model does not depend on a maneuver intention. The reason for that is, that it is hard to obtain ground truth data for a maneuver intention, as it is likely that a driver changes its intended maneuver over time. The route intention, however, is assumed to be constant, i.e., drivers do not continually redecide on whether to turn left or right. This allows to automatically label the ground truth route intention for all time steps after having observed which route an agent actually followed. This thesis argues that this assumption is mild for the route intention, but might be problematic for other types of intentions such as the maneuver intention. Leaving out the explicit modeling of maneuver intentions still allows to predict either merging (and crossing) before or after other agents, but that decision is handled by the neural network rather than being predefined by the maneuver hypothesis on which the rule-based model is conditioned on. The maneuver is thus learned implicitly and is given by the predicted trajectory as a result of the forward simulation of the learned actions. Conditioning the neural network on the maneuver intention without knowing the ground truth could be achieved with so-called *learning from incomplete data* with methods such as expectation maximization (EM). For more information, the interested reader is referred to [42], [43].

The conditioning on a driver’s route intention mainly serves two purposes: Firstly, it reduces the neural network’s prediction task complexity by eliminating the need to predict for multiple route hypotheses in a joint manner and thus to cope with the multi-



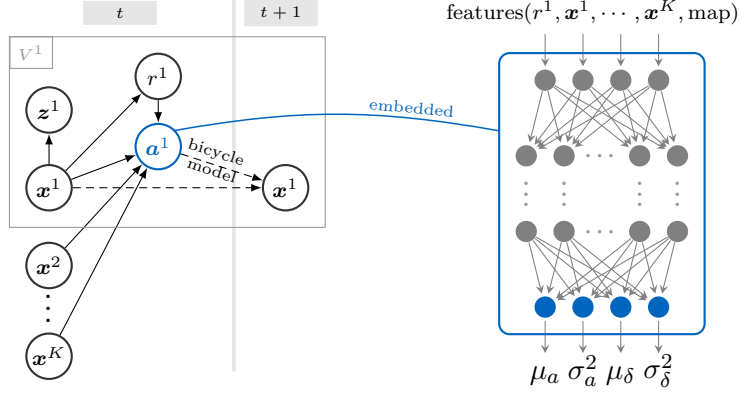
**Figure 3.4.** Heteroscedastic variance in human driver behavior: depicted is a validation plot of a learned model with mean and variance (green) and one trajectory of the validation dataset (blue). It can be seen that the variance of the learned actions strongly depends on the situation. Previously published in [168]. © IEEE 2019

modality induced by different route options. Furthermore, the fact that the number of possible routes is varying over time (depending on the road topology) makes it difficult to model all routes together with a neural network of fixed input and output size. In our case, the DBN identifies the various different route intentions first and then conditions the neural network on a specific intention to query the respective action distribution. The second purpose of conditioning on the route intention is that it allows to define relevant features along the planned path, such as the upcoming road curvature or longitudinal distances to stop lines, and to build simplified relationships between agents on complex road layouts. This enables the use of simple input and output representations, such as in our case a list of route-conditioned input features (see also Sec. 3.2.3.2) and the parameters of a Gaussian distribution as output. The integration of the neural network-based action model into the DBN presented in Chapter 2 is depicted in Fig. 3.5.

In order to be able to train a deep neural network-based action model, the ground truth actions (so-called *targets*) and corresponding input features have to be derived from recorded scenes and a suitable loss function has to be defined, which is explained in the following sections.

### 3.2.3.1 Target Generation

To determine the targets of the learning task, i.e., the “ground truth” actions of the agents, we first need to define the inverse of the kinematics bicycle model, which allows the calculation of the acceleration and steering angle given two consecutive kinematic states  $\mathbf{x}_t$  and  $\mathbf{x}_{t+1}$ . Given a sequence of states, the targets can then be determined. As the system of equations for the inverse model is over-determined, the components of  $x$



**Figure 3.5.** Embedding of neural network action model (right) into DBN (left), exemplarily depicted for one agent. The action of an agent depends on its route intention, the states of the surrounding agents, and the map, which are used to determine the input features of the neural network. The output of the network is a Gaussian action distribution comprising acceleration  $a$  and steering angle  $\delta$ . Previously published in [168]. © IEEE 2019

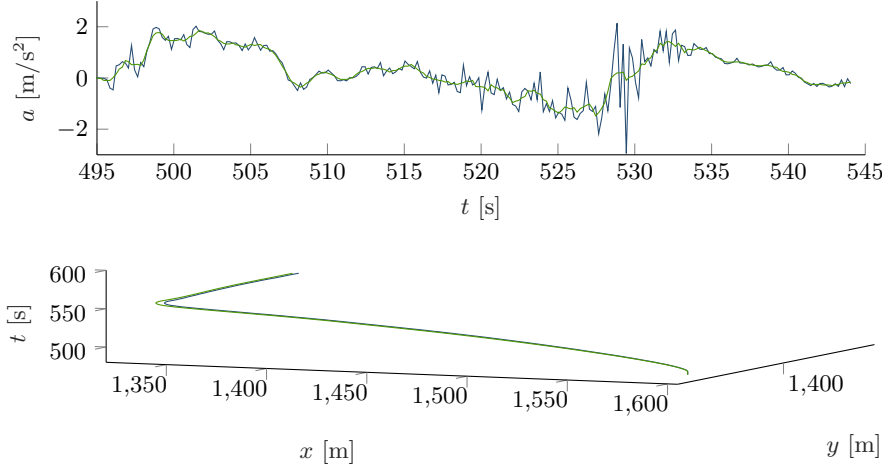
and  $y$  are ignored and it is relied on  $v$  and  $\theta$  to generate the targets:

$$a_t = \frac{v_{t+1} - v_t}{\Delta T} \quad (3.8)$$

$$\delta_t = \text{sgn} \left( \frac{\Delta \theta}{\bar{v}} \right) \arctan \left( \frac{l_f + l_r}{\sqrt{(\frac{\bar{v}}{\Delta \theta})^2 - l_r^2}} \right), \quad (3.9)$$

with  $\bar{v} = \frac{v_t + v_{t+1}}{2}$  being the mean velocity between two time steps. The steering angle is considered to be zero if an agent stands still or the term under the square root is negative. Before calculating the targets, the data sequences are resampled to  $\Delta T = 0.2$  s by linearly interpolating the kinematic states and features. To reduce the negative impact of observation noise during the target generation, the velocity and yaw angle are smoothed with moving median and moving mean (both with window size 1 s) before applying the inverse motion model (see first plot of Fig. 3.6 for a comparison of reconstructed acceleration with filtered and unfiltered velocity). To analyze the accuracy of the motion model, recorded trajectories can be reconstructed (using the initial kinematic state, the actions and the forward motion model) and compared to the original trajectories. Furthermore, this reconstruction allows to derive the vehicle model parameters  $l_f$  and  $l_r$  by searching for the minimum reconstruction error of a sample trajectory using exhaustive search. A typical example of the original data and the reconstructed states can be seen in the second plot of Fig. 3.6.

For the learning procedure, the smoothed trajectories (and corresponding targets) and the actually driven routes are assumed to be ground truth, thus learning can be done with complete data without the need to utilize methods for incomplete data such as EM.



**Figure 3.6.** First plot: determination of acceleration using inverse motion model with unfiltered (blue) and filtered (green) velocity and yaw-angle (real data). The filter applies a moving median and a moving mean with window sizes 1 s. Second plot: corresponding motion reconstruction (green) and original motion (blue). As the reconstruction with unfiltered velocity and yaw-angle does not differ noticeably from the filtered one, it is not depicted. Previously published in [168]. © IEEE 2019

### 3.2.3.2 Feature Generation

The input features of the neural network summarize the current situational context including the geometry of the road, the traffic rules, the road infrastructure, and the most relevant surrounding agents, enabling to learn the situation-dependent behavior of human drivers. Most of these features are conditioned on a specific route intention of an agent. The route intention itself is however *not* an input feature to the neural network, but rather it is used to derive a set of features that uniquely represents the specific route. These route-conditioned features describe for example the shape of the route (e.g., upcoming curvature and lane width), longitudinal distances to things like traffic lights, stoplines or the preceding agent, and the prevailing traffic rules such as the speed limit. The route of an agent furthermore enables to set it into relation to other agents and their respective possible future routes, allowing to define features such as distances to common conflict areas where their routes overlap (e.g., merge or cross) and which agent has right of way. The route intention is thus *encoded by a multitude of continuous and discrete features*, rather than given by a single discrete option such as “turn right”. When embedding this neural network into our DBN from Chapter 2, it is possible to handle arbitrary road layouts with a different number of possible route options, as the neural network is only queried conditioned on a single route at a time, allowing to always determine the same set of input features for deriving the action distribution.

In our training data (see Sec. 6.3), we found that the directly preceding agent and the closest conflicting agent are the two most influencing agents. In order to reduce complexity, only these two surrounding agents are included within the feature set used for our experiments. Considering more and potentially even a varying number of agents

**Table 3.2**  
 FEATURES OF DEEP LEARNING-BASED MODELS. PREVIOUSLY PUBLISHED IN [168]. © IEEE 2019

Predicted Vehicle	
velocity of predicted vehicle $V^i$	$v^i$
lateral position in lane	$d_{\text{lat},0}$
Route	
curvature values ahead at distances	$[c_0, c_5, \dots, c_H]$
reasonable acc given curvature ahead	$a_{\text{curve}}$
relative angle to point on centerline at distances	$[\phi_0, \phi_1, \dots, \phi_{15}]$
relative angle to direction of centerline	$\gamma_0$
width of lane	$w_0$
Traffic Rules	
speed limit	$v_{\text{lim}}$
distance to next traffic light	$d_{\text{tl}}$
next traffic light state	$s_{\text{tl}}$
distance to next stop line	$d_{\text{stop}}$
distance to next yield line	$d_{\text{yield}}$
distance to next intersection	$d_{\text{int}}$
whether always right of way at next intersection	$\chi_{\text{always}}$
Interaction	
velocity of preceding agent $V^p$	$v^p$
distance to preceding agent $V^p$	$d^p$
velocity of closest conflicting agent $V^c$	$v^c$
distance $V^c$ to conflict area (entry)	$d_{\text{entry}}^c$
distance $V^c$ to conflict area (exit)	$d_{\text{exit}}^c$
distance $V^i$ to conflict area (entry)	$d_{\text{entry}}^i$
distance $V^i$ to conflict area (exit)	$d_{\text{exit}}^i$
right of way for conflict	$\chi^{i,c}$

should be part of future research. A complete list of used features can be seen in Tab. 3.2. Different combinations of features are tried during evaluation to determine their importance (see Sec. 6.3).

There are two features that are not self-explanatory, thus they are shortly explained: The so-called *reasonable acceleration* given the upcoming curvature  $a_{\text{curve}}$  is a pre-computed feature based on domain knowledge. It is intended to subsume the set of upcoming curvature values and is calculated according to a desired maximum lateral acceleration. Given this lateral acceleration, we determine the maximum acceleration for one time step that still allows the agent to brake in time. This heuristic is also used in the rule-based model for predicting how agents are slowing down depending on the upcoming curvature (see Sec. 3.1 for more details). The boolean feature  $\chi_{\text{always}}$  specifies whether an agent does always have right of way on its route at the next intersection (no matter if other agents are present or not, nor what routes they are going to take). This feature was added, as agents on priority lanes tend to not slow down before intersections



at all, whereas agents that might have to yield slow down even in case there are no other agents around.

As the model is conditioned on the driver’s intended route, this intention has to be known during training in order to set the features accordingly. It is assumed that the route intentions are constant, i.e., that drivers do not change their minds about the routes they desire to follow. This allows to automatically label the route intention in the training data after observing which routes were actually taken. All tracks of agents that were lost before they have completely traversed an intersection are disregarded from training.

### 3.2.3.3 Loss Function

To be able to integrate the action model into various sampling-based algorithms, it is supposed to learn a mapping from a deterministic set of features to a probability distribution over actions. For the sake of problem simplification, the covariance between acceleration and steering angle is assumed to be zero.

The loss function given the predicted Gaussian with mean  $\boldsymbol{\mu} = [\mu_a, \mu_\delta]^\top$  and covariance matrix  $\boldsymbol{\Sigma} = \text{diag}(\sigma_a^2, \sigma_\delta^2)$  and the targets  $\mathbf{t} = [a^t, \delta^t]^\top$  is set to the negative log likelihood of the targets given the predicted Gaussian distribution. First, the likelihood is given by the probability density function of the multivariate Gaussian

$$f_{\mathcal{N}} = (2\pi)^{-\frac{k}{2}} |\boldsymbol{\Sigma}|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{t} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{t} - \boldsymbol{\mu})\right) \quad (3.10)$$

with  $k = 2$  being the dimensionality of the distribution and  $|\boldsymbol{\Sigma}|$  being the determinant of the covariance matrix. The negative log likelihood can then be derived to

$$\begin{aligned} -\log(f_{\mathcal{N}}) &= -\left[\log((2\pi)^{-\frac{k}{2}}) + \log(|\boldsymbol{\Sigma}|^{-\frac{1}{2}}) - \frac{1}{2}(\mathbf{t} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{t} - \boldsymbol{\mu})\right] \\ &= \log((2\pi)^{\frac{k}{2}}) + \log(\sqrt{|\boldsymbol{\Sigma}|}) + \frac{1}{2}(\mathbf{t} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{t} - \boldsymbol{\mu}) \end{aligned} \quad (3.11)$$

As the first term is constant and constant terms do not matter within a loss function, the actual loss is reduced to

$$l_{\text{NLL}}(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \mathbf{t}) = \log(\sqrt{|\boldsymbol{\Sigma}|}) + \frac{1}{2}(\mathbf{t} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{t} - \boldsymbol{\mu}). \quad (3.12)$$

The outputs of the neural network corresponding to the variances of the action are transformed with an exponential function before calculating the loss to guarantee positive values (i.e., the actual outputs of the network are  $\log(\sigma_a^2)$  and  $\log(\sigma_\delta^2)$ , respectively). This loss enables to learn both the mean and variance in a single training procedure.



# 4 Inference Methods

In this thesis, two different inference methods are proposed and compared in regards of their accuracy and runtime in the context of driver intention estimation and probabilistic trajectory prediction using the previously presented DBN from Chapter 2. Inference in a DBN generally falls into the category of recursive Bayesian estimation. Given a stochastic process that contains both hidden as well as observable random variables, recursive Bayesian estimation is the process of inferring the probability distributions of the hidden variables recursively over time. The estimates are derived using Bayesian statistics given an initial belief, an assumed transition model, an assumed measurement model, and measurements of the observable quantities. There are various different possibilities to solve a recursive Bayesian estimation problem, depending on the type of distribution and the transition and measurement models. A detailed overview of different possible inference algorithms for specific variants of DBNs has been presented by Murphy [95]. In the special case of discrete-only hidden variables or when transition and observation models are conjugate (e.g., linear-Gaussian), exact inference is generally possible. However, for non-linear DBNs with hybrid state space (i.e., the state consists of both discrete and continuous hidden variables), such as presented within this work, approximate inference becomes necessary.

In the remainder of this section, a sequential Monte Carlo (SMC)-based method and a multiple model unscented Kalman filter (MM-UKF)-based method are presented. This chapter is based on the author's previous publications [166] and [167].

## 4.1 Fundamentals

### 4.1.1 Probability Theory

To be able to derive the inference methods for the presented DBN, first some fundamentals of probability theory are revisited, namely *Bayes' rule*, the *theorem of total probability* and the *Markov assumption*. Note that the variable names of this and the subsequent section are chosen to be in line with prevalent literature and do not correspond to the variables used throughout the rest of this thesis. Given the definition of conditional probability

$$p(x|y) = \frac{p(x, y)}{p(y)}, \quad (4.1)$$

## 4 Inference Methods

it is possible to derive *Bayes' rule*, which relates a conditional probability  $p(x|y)$  to its inverse  $p(y|x)$ , as

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)}, \quad (4.2)$$

with  $p(y) > 0$ . In robotics or machine learning,  $p(x)$  is typically denoted as the prior probability distribution,  $y$  as the data, and  $p(x|y)$  as the posterior probability distribution. It is possible to condition Bayes' rule on another random variable as follows:

$$p(x|y, z) = \frac{p(y|x, z)p(x|z)}{p(y|z)}. \quad (4.3)$$

The *theorem of total probability* also follows from the definition of conditional probability and is given for the continuous case by

$$p(x) = \int p(x|y)p(y)dy. \quad (4.4)$$

For the discrete case, one sums over all the possible values of  $y$  instead of integrating over them.

The state of a stochastic process  $x_t$  is denoted *complete* (complete state assumption), if no variables prior to  $x_t$  may influence the stochastic evolution of future states (unless this dependence is mediated through the state  $x_t$ ). If the state is complete, the process follows the *Markov property*, i.e., the subsequent state  $x_{t+1}$  is conditionally independent of previous states  $x_{t-i} \forall i \in \mathbb{N}^+$  given the current state  $x_t$ , or formally

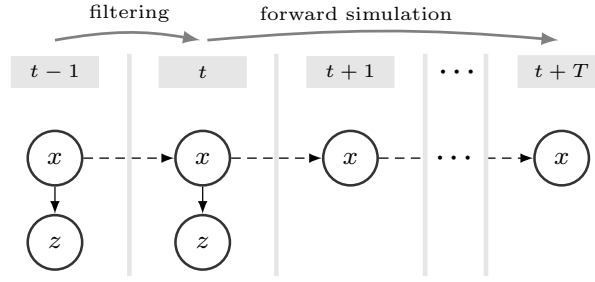
$$p(x_{t+1}|x_{0:t}) = p(x_{t+1}|x_t). \quad (4.5)$$

Furthermore, if a state  $x_t$  is complete, measurements  $z_t$  are independent of all other quantities given that state, i.e., they only depend on the state of the current time, and not on any previous states or measurements [137, p. 21]:

$$p(z_t|x_{0:t}, z_{0:t-1}) = p(z_t|x_t). \quad (4.6)$$

### 4.1.2 Recursive Bayesian Estimation

Assume a stochastic process that satisfies the complete state assumption and thus the Markov property and consists of a hidden state  $x_t$  and an observable quantity  $z_t$  as depicted in Fig. 4.1. Although the true state is not fully observable, the observable quantity depends on the true state, defined by an observation model  $z_t = h(x_t)$ , such that one can reason about the true state of the process. In recursive Bayesian estimation, one is generally interested in the belief over the true state  $x_t$  given all observations received over time  $z_{0:t}$ , mathematically speaking  $bel(x_t) := p(x_t|z_{0:t})$ . Using Bayes' rule,



**Figure 4.1.** Recursive Bayesian estimation with filtering and forward simulation. State transition model depicted as dashed arcs, measurement model depicted as solid arcs.

the Markov assumption and the law of total probability, it follows that

$$\begin{aligned}
 \underline{\underline{bel}}(x_t) &:= p(x_t|z_{0:t}) && \stackrel{\text{Bayes}}{=} && \frac{p(z_t|x_t, z_{0:t-1}) p(x_t|z_{0:t-1})}{p(z_t|z_{0:t-1})} \\
 & && = && \eta p(z_t|x_t, z_{0:t-1}) p(x_t|z_{0:t-1}) \\
 & \stackrel{\text{complete state/Markov}}{=} && \eta p(z_t|x_t) p(x_t|z_{0:t-1}) \\
 & \stackrel{\text{total probability}}{=} && \eta p(z_t|x_t) \int p(x_t|x_{t-1}, z_{0:t-1}) p(x_{t-1}|z_{0:t-1}) dx_{t-1} \\
 & \stackrel{\text{complete state/Markov}}{=} && \eta p(z_t|x_t) \int p(x_t|x_{t-1}) p(x_{t-1}|z_{0:t-1}) dx_{t-1} \\
 & = && \eta p(z_t|x_t) \int p(x_t|x_{t-1}) \underline{\underline{bel}}(x_{t-1}) dx_{t-1} \quad (4.7)
 \end{aligned}$$

with  $\eta$  being a normalization constant that is independent of  $x_{0:t}$ . Instead of calculating the belief of the complete trajectory at once, the belief of the state is calculated *sequentially* using a predict and update cycle starting with an initial belief  $\underline{\underline{bel}}(x_0) = p(x_0)$ . Each prediction step only uses the previous belief  $\underline{\underline{bel}}(x_{t-1})$  and the transition model  $p(x_t|x_{t-1})$  to calculate the predicted belief

$$\overline{\underline{\underline{bel}}}(x_t) := p(x_t|z_{0:t-1}) = \int p(x_t|x_{t-1}) \underline{\underline{bel}}(x_{t-1}) dx_{t-1}. \quad (4.8)$$

Each measurement update uses this predicted belief, the current measurement  $z_t$ , and the measurement model  $p(z_t|x_t)$  to generate the updated belief

$$\underline{\underline{bel}}(x_t) = \eta p(z_t|x_t) \overline{\underline{\underline{bel}}}(x_t) = \eta p(z_t|x_t) p(x_t|z_{0:t-1}), \quad (4.9)$$

This allows for a low computational cost per time step and enables online filtering algorithms.

Besides the process of filtering, mathematically  $p(x_t|z_{0:t})$ , it is possible to use the given models for prediction, mathematically  $p(x_{t+T}|z_{0:t})$  with  $T$  being the prediction horizon. Having defined the transition model  $p(x_{t+1}|x_t)$ , it is possible to extrapolate the current belief  $p(x_t)$  into the future by iteratively applying the transition model without

incorporating any new measurements. This process is known as *forward simulation* and allows to probabilistically predict the trajectory of the complete process for an arbitrary number of steps, as depicted in Fig. 4.1.

## 4.2 Sequential Monte Carlo

The presented DBN describes a hybrid, non-linear system with a multi-modal, non-Gaussian belief. Sequential Monte Carlo (SMC) methods are capable of representing arbitrary probability distributions and can handle highly non-linear system dynamics, which make them suitable for any type of DBN. They represent the belief over the state by a large number of (potentially weighted) deterministic state instances, so-called samples or particles. Areas of high probability contain more samples (or samples with larger weights) than areas of low probability. Major advantages of sample-based inference methods are, that there is no need for differentiation of the transition models and that one does not have to define the conditional probability distributions analytically, but only as a function that samples from the desired distribution. This allows to embed libraries and functions that expect deterministic input and may not be differentiable (such as for example a lane matching module).

SMC has already been successfully applied for DBN inference in the context of behavior prediction, e.g., by Gindele et al. [44]. In this thesis, a standard sequential importance resample (SIR) particle filter, also known as the *bootstrap filter*, with *low-variance resampling* is utilized.

### 4.2.1 Approach

Initially, a set of  $N$  weighted particles

$$\mathcal{P}_0 = \{\langle S_0^{[1]}, w_0^{[1]} \rangle, \dots, \langle S_0^{[N]}, w_0^{[N]} \rangle\} \quad (4.10)$$

is sampled according to the initial measurement  $Z_0$  and the map, with  $S^{[n]} = [X^{[n]}, R^{[n]}, M^{[n]}, A^{[n]}]$  representing the complete scene with  $K$  agents:

$$S_0^{[n]} \sim p(X_0, R_0, M_0, A_0 | Z_0, \text{map}) \quad (4.11)$$

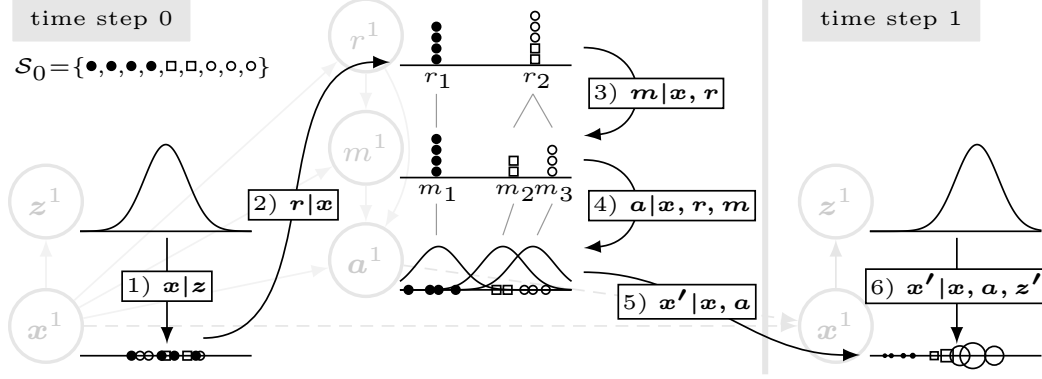
$$= p(X_0 | Z_0) p(R_0 | X_0, \text{map}) p(M_0 | R_0, X_0, \text{map}) p(A_0 | R_0, M_0, X_0, \text{map}) \quad (4.12)$$

$$= p(\mathbf{x}_0^1 | \mathbf{z}_0^1) \cdots p(\mathbf{x}_0^K | \mathbf{z}_0^K)$$

$$p(r_0^1 | \mathbf{x}_0^1, \text{map}) \cdots p(r_0^K | \mathbf{x}_0^K, \text{map})$$

$$p(m_0^1 | r_0^1, X_0, \text{map}) \cdots p(m_0^K | r_0^K, X_0, \text{map})$$

$$p(\mathbf{a}_0^1 | r_0^1, m_0^1, X_0, \text{map}) \cdots p(\mathbf{a}_0^K | r_0^K, m_0^K, X_0, \text{map}) \quad (4.13)$$



**Figure 4.2.** Exemplary initial sample generation (1-4), motion prediction (5), and particle weighting (size of particles represents weight) (6), shown for a single agent. Distributions are depicted simplified as being one dimensional. One particle represents the complete state space, i.e., kinematic states, routes, maneuverers, and actions of all agents in the scene. Previously published in [166]. © IEEE 2018

The corresponding weights are set to  $w_0^{[n]} = N^{-1}$ . For subsequent time steps, each particle is predicted according to the transition probability:

$$S_{t+1}^{[n]} \sim p(S_{t+1}|S_t^{[n]}, \text{map}) \quad (4.14)$$

$$= p(X_t|X_{t-1}^{[n]}, A_{t-1}^{[n]}) p(R_t|R_{t-1}^{[n]}, X_t^{[n]}, \text{map}) p(M_t|M_{t-1}^{[n]}, R_t^{[n]}, X_t^{[n]}, \text{map}) \\ p(A_t|R_t^{[n]}, M_t^{[n]}, X_t^{[n]}, \text{map}) \quad (4.15)$$

$$= p(x_t^1|x_{t-1}^{1,[n]}, a_{t-1}^{1,[n]}) \cdots p(x_t^K|x_{t-1}^{K,[n]}, a_{t-1}^{K,[n]}) \\ p(r_t^1|r_{t-1}^{1,[n]}, x_t^{1,[n]}, \text{map}) \cdots p(r_t^K|r_{t-1}^{K,[n]}, x_t^{K,[n]}, \text{map}) \\ p(m_t^1|m_{t-1}^{1,[n]}, r_t^{1,[n]}, X_t^{[n]}, \text{map}) \cdots p(m_t^K|m_{t-1}^{K,[n]}, r_t^{K,[n]}, X_t^{[n]}, \text{map}) \\ p(a_t^1|r_t^{1,[n]}, m_t^{1,[n]}, X_t^{[n]}, \text{map}) \cdots p(a_t^K|r_t^{K,[n]}, m_t^{K,[n]}, X_t^{[n]}, \text{map}). \quad (4.16)$$

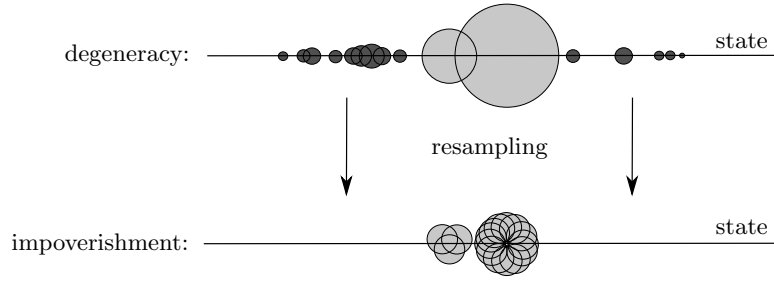
Due to the conditional independencies between the nodes, the sampling procedure can be done sequentially, node by node. This sequential procedure of the SMC inference method is exemplarily depicted for the initial sample generation in Fig. 4.2. Steps 1-4 represent Eq. (4.11). As soon as a new measurement is available, the state is predicted to the new time step (step 5) and the particle weights get updated according to the measurement likelihood (step 6):

$$\tilde{w}_t^{[n]} = p(Z_t|X_t^{[n]}) w_{t-1}^{[n]} \quad (4.17)$$

$$= p(z_t^1|x_t^{1,[n]}) \cdots p(z_t^K|x_t^{K,[n]}) w_{t-1}^{[n]} \quad (4.18)$$

and normalized

$$w_t^{[n]} = \frac{\tilde{w}_t^{[n]}}{\sum_{m=1}^N \tilde{w}_t^{[m]}}. \quad (4.19)$$



**Figure 4.3.** Problems of sample degeneracy and sample impoverishment in SMC methods: Top row shows weighted particles (weight illustrated with diameter) with most particles having negligible weight as they represent the state poorly. Bottom row shows the set of particles after resampling with a great loss of diversity, as only two samples have been reproduced (depicted slightly displaced for improved visualization). Graphic taken and adapted from [84].

Over the course of tracking, at some point most particles will inevitably end up with small weights because they represent the state poorly and only few will have non-negligible weight. This so-called *sample degeneracy* is caused by inaccurate models and prediction uncertainty. In order to concentrate the particles in areas of high likelihood and don't waste computational power for very unlikely particles, one can employ what is called *importance resampling*. The idea of resampling is, that new particles are drawn with replacement from the existing set of particles according to their weights. Thus, particles with high weights are more likely to be drawn than particles with low weights, reducing the sample degeneracy. After resampling, the weights of all  $N$  particles are reset to  $w^{[n]} = N^{-1}$  in order to not introduce bias to the distribution. The resampling, however, induces loss of sample diversity, and should only be done when necessary to avoid the problem of *sample impoverishment*, meaning most of the particles represent exactly the same values, as depicted in Fig. 4.3. A detailed analysis of these problems can be found in [84].

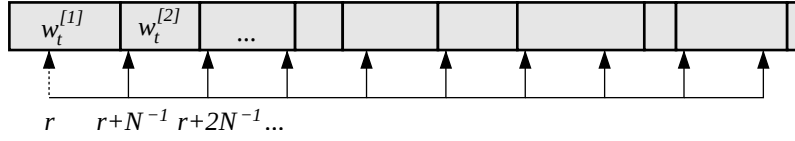
In this thesis, *low variance resampling* [137, p. 110] is utilized, whenever the *effective sample size*  $ESS$  is lower than  $N/2$ , as proposed by [37]:

$$ESS = \frac{1}{\sum_{n=1}^N (w_t^{[n]})^2} < N/2. \quad (4.20)$$

The particle filter algorithm is given in Algorithm 1 and the method of low variance resampling in Algorithm 2 and Fig. 4.4.

To save computational resources, first, only the kinematic state  $X_t$  is sampled according to  $p(X_t|X_{t-1}, A_{t-1})$  instead of sampling the complete state space according to Eq. (4.14). After the measurement update and (potential) resampling, the remaining variables of the state space  $R_t$ ,  $M_t$ , and  $A_t$  are sampled according to Eq. (4.14). This is possible as the measurement likelihood is independent of the routes, maneuvers, and actions given the kinematic states and restricts the computation of routes, maneuvers and actions to particles that have survived the resampling.





**Figure 4.4.** Principle of low variance resampling procedure. Graphic taken and adapted from [137, p. 111].

---

**Algorithm 1** Particle Filter (adapted from [137, p. 98])

---

**Input**  $\mathcal{P}_{t-1} = \{\langle S_{t-1}^{[1]}, w_{t-1}^{[1]} \rangle, \dots, \langle S_{t-1}^{[N]}, w_{t-1}^{[N]} \rangle\}$   $\triangleright$  set of particles at time  $t-1$   
**Output**  $\mathcal{P}_t = \{\langle S_t^{[1]}, w_t^{[1]} \rangle, \dots, \langle S_t^{[N]}, w_t^{[N]} \rangle\}$   $\triangleright$  set of particles at time  $t$

- 1: **procedure** PARTICLEFILTER
- 2:    $\tilde{\mathcal{P}}_t = \tilde{\mathcal{P}}_t = \emptyset$
- 3:   **for**  $n = 1$  to  $N$  **do**  $\triangleright$  predict and update
- 4:     sample  $S_t^{[n]} \sim p(S_t | S_{t-1}^{[n]}, \text{map})$
- 5:      $w_t^{[n]} = p(Z_t | S_t^{[n]}) \cdot w_{t-1}^{[n]}$
- 6:      $\tilde{\mathcal{P}}_t = \tilde{\mathcal{P}}_t \cup \langle S_t^{[n]}, w_t^{[n]} \rangle$
- 7:   **if** resampling desired **then**
- 8:     **for**  $n = 1$  to  $N$  **do**  $\triangleright$  resampling (optional)
- 9:      draw  $m$  with probability  $\propto w_t^m$
- 10:      $\mathcal{P}_t = \mathcal{P}_t \cup \langle S_t^{[m]}, \frac{1}{N} \rangle$
- 11:   **else**
- 12:      $\mathcal{P}_t = \tilde{\mathcal{P}}_t$

---



---

**Algorithm 2** Low Variance Resampling (adapted from [137, p. 110])

---

**Input**  $\tilde{\mathcal{P}}_t$   
**Output**  $\mathcal{P}_t$

- 1: **procedure** RESAMPLE
- 2:    $\mathcal{P}_t = \emptyset$
- 3:   sample  $r \sim \mathcal{U}(0; N^{-1})$   $\triangleright$  draw random number between 0 and  $N^{-1}$
- 4:    $c = w_t^{[1]}$
- 5:    $m = 1$
- 6:   **for**  $n = 1$  to  $N$  **do**
- 7:      $U = r + (n-1) \cdot N^{-1}$
- 8:     **while**  $U > c$  **do**
- 9:       $m = m + 1$
- 10:      $c = c + w_t^{[m]}$
- 11:      $\mathcal{P}_t = \mathcal{P}_t \cup \langle S_t^{[m]}, \frac{1}{N} \rangle$

---

The probability of a set of intentions  $(R_t, M_t)$  is determined by the proportion of the sum of the weights of the respective particles to the sum of the weights of all available particles after the measurement update of a given time step

$$p(R_t, M_t) = \frac{\sum_{j \in \mathcal{J}} w_t^{[j]}}{\sum_{n=1}^N w_t^{[n]}}, \mathcal{J} := \{j \mid R_t^{[j]} = R_t, M_t^{[j]} = M_t\}. \quad (4.21)$$

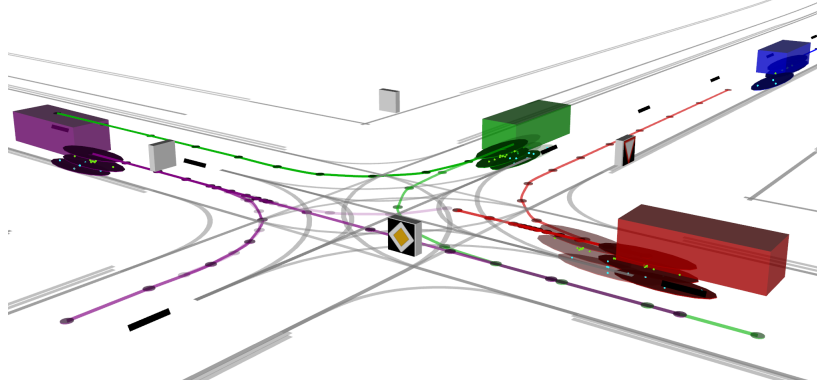
The intention of a single agent can be derived through marginalization of the belief. For the prediction into the future, steps 1-5 can be repeated without incorporating new measurements, predicting one time step at a time. This forward simulation can be executed until an arbitrary desired prediction horizon  $T$  resulting in the predicted particle distribution  $\mathcal{P}_{t+T}$ . This prediction incorporates the already determined probabilities of the different intention hypotheses and accounts for the interaction between all agents in the state space.

Depending on the prediction requirements, this forward simulation does not necessarily have to be done for each particle, but can also be done for only a subset of particles in order to reduce complexity. One way to choose this subset is to create one particle per route and maneuver *combination*  $(R, M)$  of all agents and set its kinematic state to the mean kinematic state given this set of intentions. This allows to keep the high-level intention uncertainty and the multi-modality of the prediction distribution while neglecting the low-level action uncertainty. For this purpose, for each available combination  $(R, M)$  within the set of particles  $\mathcal{P}$ , the mean kinematic state is calculated and one multi-agent trajectory is generated and weighted with the corresponding probability  $p(R, M)$ . Due to the interdependencies of multiple agents' future trajectories and the strong impact of the intentions on the future behavior, the combinatorial aspect of the high-level intentions should not be neglected within the prediction of the scene development.

### 4.3 Multiple Model Unscented Kalman Filter

To account for non-linear system dynamics and hybrid state spaces, SMC methods are often the inference method of choice. They come with the advantage of being able to represent arbitrary distributions (hybrid, non-Gaussian, multi-modal) and being applicable to highly non-linear systems. However, in state estimation problems with high uncertainty in the belief, e.g., as a result of unknown intentions, varying human behavior, and noisy measurements, they typically suffer from either high complexity or low accuracy, depending on the number of samples used. Many particles may be needed to approximate the belief appropriately and to reduce the variance caused by randomness in the sampling process. Furthermore, they suffer from the problems of sample degeneracy and sample impoverishment.

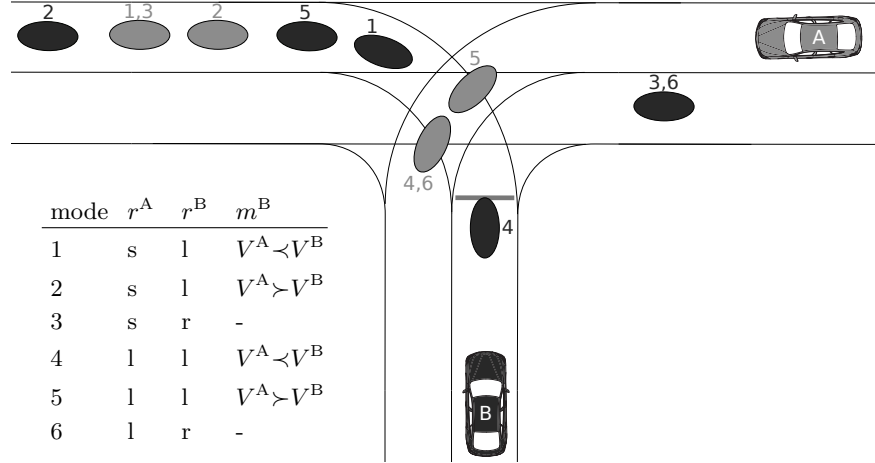
Another way of representing a multi-modal belief is by defining distinct modes of high probability and approximate each mode by a closed-form distribution such as a Gaussian. The main reasons for the multi-modality of the future development of a traffic situation are the different possible high-level intentions of all agents, i.e., their desired



**Figure 4.5.** Estimated belief represented by a mixture of Gaussians (depicted as ellipses in vehicle color) and corresponding sigma points (depicted as green dots) and interaction-aware trajectory prediction (depicted as lines in vehicle color) for the different possible intentions. Previously published in [167]. © IEEE 2018

routes and maneuvers. In order to circumvent the aforementioned disadvantages of SMC, this section presents a multiple model unscented Kalman filter (MM-UKF)-based inference method, which approximates the belief with a mixture of Gaussians. For each combination of high-level intentions, one mixture component is created. This method aims to reduce complexity, while still keeping the benefits of sample-based evaluation of complex, non-linear, and non-continuous transition models.

In contrast to the extended Kalman filter (EKF) which linearizes the transition functions using the first-order Taylor approximation, the unscented Kalman filter (UKF) does not require the differentiation of transition models, making it simpler and more widely applicable [95]. The UKF directly uses the non-linear transition and measurement functions, but approximates the Gaussian belief by a set of deterministically chosen samples. These so-called *sigma points* are then transformed using the non-linear functions and a new predicted or corrected Gaussian is calculated based on the transformed point estimates. The transition models proposed in this thesis include non-differentiable parts such as lane-matching, which forbids the use of EKFs. Furthermore, UKFs have often been shown to be superior to EKFs, especially in estimating the variance of the Gaussian belief [4], [62], [150]. Andersen et al. [4] compared EKFs, UKFs, and SMC methods, highlighting the benefits of UKF-based inference. Bitzer [23] conducted an in-depth analysis of UKF and SMC, demonstrating that the advantages and disadvantages of both inference methods in terms of complexity and accuracy depend on the dimensionality of the state space and the problem at hand. UKFs come with similar advantages as SMC methods, as they are also using samples of the belief to evaluate the transition models. A typical scenario is depicted in Fig. 4.5, showing the Gaussian mixture distribution and the corresponding sigma points.



**Figure 4.6.** Example of how different modes may result in different clusters of the continuous state (here shown as Gaussians of the positions) at a future time step, due to the interdependencies between agents. The routes are abbreviated as straight, left, and right. Previously published in [167]. © IEEE 2018

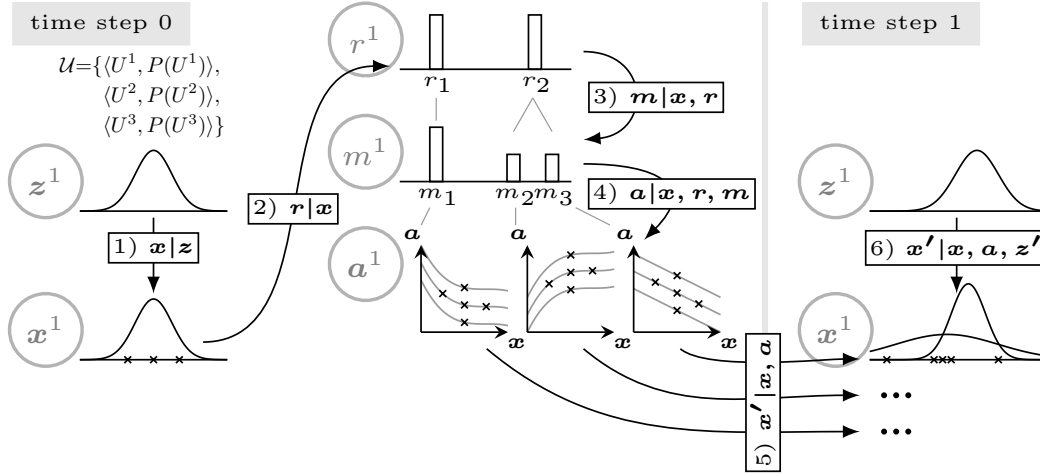
### 4.3.1 Approach

The existence of different route and maneuver hypotheses generally leads to multi-modality in the predicted trajectory distribution. As agents interact with each other and therefore their trajectories become interdependent, the discrete intentions of one agent may also lead to multi-modality in the prediction of *another* agent. Fig. 4.6 depicts a typical motion prediction situation illustrating this combinatorial aspect: depending on the intended route and maneuver of agent  $V^A$ , agent  $V^B$  may completely change its future behavior and vice-versa, resulting in different high probability clusters of their continuous states. For example, when  $V^A$  turns left,  $V^B$  has to wait longer at the intersection before it can turn left than when  $V^A$  would have driven straight.

As unscented Kalman filters can only represent unimodal and continuous distributions (multivariate Gaussians), but the presented DBN has a hybrid state space with a (potentially) multi-modal belief, this section proposes to use *multiple models*, representing the discrete variables by the different modes of the MM-UKF. To account for interdependencies between multiple agents, all agents are included in the state space of the DBN. Thus, the discrete route and maneuver intentions of all agents have to be considered in a combinatorial manner: each of the possible combinations of intentions of all agents ( $R, M$ ) is represented by one mode, whereas the continuous states ( $X, A$ ) are represented by one multivariate Gaussian given each mode. Thus, the overall belief is represented by a mixture of Gaussians, where each mode forms one mixture component.

#### 4.3.1.1 Multiple Models

Given the route and maneuver of *each agent*, the evaluation data of this thesis suggests that the belief can reasonably be approximated by a single multivariate Gaussian. Thus,



**Figure 4.7.** MM-UKF inference exemplarily shown for a single vehicle with three different modes:  $(r_1, m_1), (r_2, m_2), (r_2, m_3)$ . Distributions are depicted simplified as being one dimensional. One UKF represents the complete state space, i.e., kinematic states, routes, maneuvers, and actions of all agents in the scene. To account for the non-additive process noise induced by the uncertain actions, the sigma points (depicted as little crosses) are generated using an augmented state that includes  $\mathbf{x}$  and  $\mathbf{a}$  (see action distribution after step 4). The steps of the algorithm are marked from 1 to 6. Previously published in [167]. © IEEE 2018

one UKF represents one possible *combination* of route and maneuver intentions of all agents. The number of possible combinations defines the number of modes and is given by the multiplication of the number of intentions of the single agents

$$|\mathcal{C}| = \prod_{i=1}^K \sum_{r \in \mathcal{R}^i} \sum_{m \in \mathcal{M}_r^i} 1, \quad (4.22)$$

with  $\mathcal{C}$  being the set of all possible combinations of intentions of all agents. Hence, the belief of the DBN is tracked using a set of  $|\mathcal{C}|$  weighted UKFs

$$\mathcal{U} = \{\langle U^1, p(U^1) \rangle, \dots, \langle U^{|\mathcal{C}|}, p(U^{|\mathcal{C}|}) \rangle\}. \quad (4.23)$$

Each UKF  $U^j$  represents the complete scene  $S = [X, R, M, A]$  including all agents and has attached the corresponding mode probability  $p(U^j)$ .

The general prediction and update cycle of the MM-UKF is exemplarily depicted in Fig. 4.7: Initially, a multivariate Gaussian distribution of the kinematic state  $X_0$  is derived from the first measurement  $Z_0$  according to  $p(X|Z)$  (step 1). Then, for each agent, the set of possible routes  $\mathcal{R}_0$  and the set of possible maneuvers  $\mathcal{M}_{0|r}$  for each possible route  $r \in \mathcal{R}_0$  is determined given the agent's own state, the map, and all other agents' states (step 2-3). For each possible combination of  $(R_0^j, M_0^j)$  of all agents, one UKF  $U_0^j$  is created and initialized to the corresponding  $R_0^j$  and  $M_0^j$  and the Gaussian distribution  $X_0^j = X_0$ . Furthermore, it is assigned with the mode probability

$$p(R_0^j, M_0^j) = p(U_0^j) = p(R_0^j | X_0^j, \text{map}) p(M_0^j | R_0^j, X_0^j, \text{map}), \quad (4.24)$$

## 4 Inference Methods

which depends on the route and maneuver priors, which are assumed to be uniformly distributed in this thesis.

For each UKF, the continuous part of the belief  $U_t^j$  (which is assumed to be Gaussian) is temporarily represented by sigma points and predicted to  $U_{t+1|t}^j$  (step 4-5). The predicted sigma points are used to determine the predicted Gaussian which is then updated with the measurement to  $U_{t+1}^j$  (step 6). This procedure of the predict and update cycle of a single UKF is explained in more detail in the next section. The probability of each mode is updated according to the measurement likelihood  $\mathcal{L}(X_{t+1|t}^j|Z_{t+1})$ :

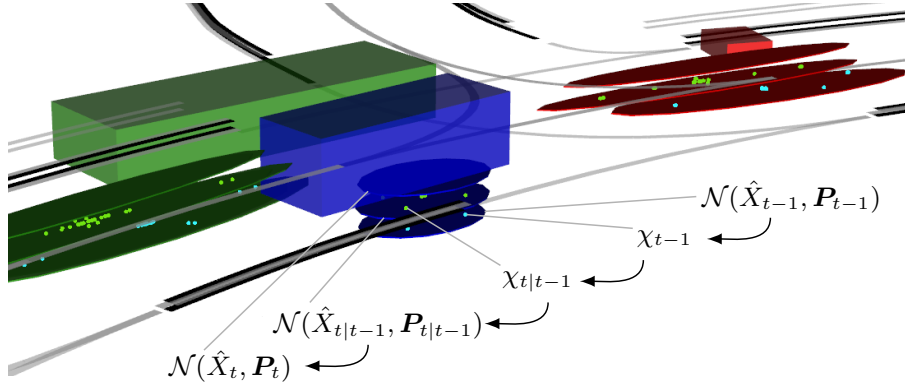
$$p(U_{t+1}^j) = \frac{p(U_t^j)\mathcal{L}(X_{t+1|t}^j|Z_{t+1})}{\sum_{l=1}^{|\mathcal{C}|} p(U_t^l)\mathcal{L}(X_{t+1|t}^l|Z_{t+1})}. \quad (4.25)$$

The Gaussian mixture that represents all modes, i.e., possible combinations of routes and maneuvers of all agents, is then given by the set of UKFs and corresponding probabilities. In case the number of possible combinations is too high to be computationally feasible for real time application, it is possible to only select a subset of modes in the beginning based on their prior probabilities and replace modes of low likelihood with new modes over time. This concept is similar to the resampling process of the SMC method, but it corresponds to a sampling without replacement, as one UKF already contains the information about the uncertainty (i.e., there is no benefit in tracking the same mode multiple times).

As new agents appear (i.e., they are represented for the first time in a new measurement) or new routes or maneuvers emerge due to the route horizon reaching a route split or a new conflict area, the existing modes are duplicated accordingly in order to represent the new agents or the new possible intentions, and the probabilities are split uniformly. When routes or maneuvers become impossible or when agents disappear, the corresponding modes of the MM-UKF are removed or merged. In addition, unlikely modes can be disregarded to decrease runtime complexity (however, this option was disabled within the evaluation chapter of this dissertation to ensure a fair comparison with the SMC-based inference). As the number of modes may change over time, the proposed inference method belongs to the type *multiple model with variable structure* [85]. For the evaluation of the MM-UKF, the intentions of a driver are modeled as constant over time, i.e., the probability of switching of the current mode is assumed to be zero. To allow for a possibility of mode switches, an interacting multiple model filter could be applied as described in [14, pp. 24-29].

### 4.3.1.2 Single Model UKF

Generally, a UKF determines a deterministically chosen set of *sigma points* that capture the mean and covariance of the original distribution and propagates them through the non-linear transition functions of the prediction and update steps. Then, a new Gaussian distribution is fitted to the resulting transformed points. As in the case of this thesis the measurement is Gaussian and the measurement function is linear, sigma points are only utilized for the prediction step to be able to capture the non-linear system dynamics,



**Figure 4.8.** Estimated belief and annotations for the Gaussians of the last posterior, the current prior and current posterior as well as the sigma points used for the prediction step. The posterior of the last time step and the prior of the current time step are drawn at lower height to improve visualization. Previously published in [167]. © IEEE 2018

whereas the measurement update is equivalent to the one of a standard Kalman filter. This procedure is depicted in Fig. 4.8 and will be explained in the remainder of this section. For the sake of brevity, the superscript  $(\cdot)^j$  denoting the respective mode is omitted within this section as the steps are identical for any mode of the MM-UKF.

Typically, a set of  $2L + 1$  sigma points is chosen, with  $L$  being the dimensionality of the state, such that one sigma point is placed at the mean of the Gaussian state and the others are symmetrically spread around it to capture the variance. If the system is affected by non-additive process noise, the mean estimate of the kinematic state  $\hat{\mathbf{x}}$  and the covariance  $\mathbf{P}$  of the Gaussian have to be augmented by one dimension per non-additive noise term resulting in  $\hat{\mathbf{x}}_{\text{aug}}$  and  $\mathbf{P}_{\text{aug}}$ , such that it can be accounted for by the sigma points. Additive noise, on the other hand, can simply be added to the fitted Gaussian after the transformation. Thus, for additive noise, the state and covariance do not have to be augmented further.

In this work, for each agent, the augmentation accounts for the non-additive process noise terms given by the uncertain actions of each agent. The augmented state and covariance thus comprise the kinematic state  $\mathbf{x}$  and the action  $\mathbf{a}$ , resulting in  $\hat{\mathbf{x}}_{\text{aug}}^i = [\hat{\mathbf{x}}^i \ \boldsymbol{\mu}_a^i]^\top$  and  $\mathbf{P}_{\text{aug}}^i = \text{diag}(\mathbf{P}^i, \boldsymbol{\Sigma}_a)$ . where  $\boldsymbol{\mu}_a$  is the mean and  $\boldsymbol{\Sigma}_a$  the covariance of the function  $p(\mathbf{a}|\hat{X}, r, m, \text{map})$  which is defined by the used behavior model (see Chapter 3). The overall augmented distribution consisting of all  $K$  agents is thus given by a Gaussian with mean  $\hat{X}_{\text{aug}} = [\hat{\mathbf{x}}_{\text{aug}}^1, \dots, \hat{\mathbf{x}}_{\text{aug}}^K]^\top$  and covariance  $\mathbf{P}_{\text{aug}} = \text{diag}(\mathbf{P}_{\text{aug}}^1, \dots, \mathbf{P}_{\text{aug}}^K)$ .

Various types of action models can be applied with this UKF. As both models we present in Chapter 3 consist of a two-dimensional action vector, the following equations assume an augmented state of size  $4 + 2$ . For  $K$  agents, the state thus has a size of  $L = K(4 + 2)$ , resulting in a total of  $2L + 1 = 12K + 1$  sigma points being created. The

#### 4 Inference Methods

sigma points are given by

$$\chi_t^0 = \hat{X}_{\text{aug},t} \quad (4.26)$$

$$\chi_t^i = \hat{X}_{\text{aug},t} + \left( \sqrt{(L + \lambda) \mathbf{P}_{\text{aug},t}} \right)_i, \quad i = 1, \dots, L \quad (4.27)$$

$$\chi_t^i = \hat{X}_{\text{aug},t} - \left( \sqrt{(L + \lambda) \mathbf{P}_{\text{aug},t}} \right)_i, \quad i = L + 1, \dots, 2L, \quad (4.28)$$

with  $\left( \sqrt{(L + \lambda) \mathbf{P}_{\text{aug},t}} \right)_i$  being the  $i$ th column of the matrix square root of  $(L + \lambda) \mathbf{P}_{\text{aug},t}$ . After their creation, each sigma point  $\chi_t^i$  is propagated through the transition function of the kinematic state which is defined by the behavior model (step 5 of Fig. 4.7). The resulting sigma points  $\chi_{t+1|t}^i$  are then used to derive the predicted Gaussian with mean and covariance

$$\hat{X}_{t+1|t} = \sum_{i=0}^{2L} W_s^i \chi_{t+1|t}^i \quad (4.29)$$

$$\mathbf{P}_{t+1|t} = \mathbf{Q} + \sum_{i=0}^{2L} W_c^i [\chi_{t+1|t}^i - \hat{X}_{t+1|t}] [\chi_{t+1|t}^i - \hat{X}_{t+1|t}]^\top \quad (4.30)$$

with the state and covariance weights

$$\begin{aligned} W_s^0 &= \frac{\lambda}{L + \lambda}, & W_c^0 &= W_s^0 + (1 - \alpha^2 + \beta), \\ W_s^i &= W_c^i = \frac{1}{2(L + \lambda)}, & i &= 1, \dots, 2L \end{aligned} \quad (4.31)$$

and  $\lambda = \alpha^2 \kappa - L$ . As the measurement noise is Gaussian and the measurement function is linear, the measurement step is performed according to the standard Kalman filter (step 6), correcting the mean and covariance of the belief without the need of utilizing sigma points. The measurement update further allows the determination of the measurement likelihood  $\mathcal{L}(X_{t+1|t}^j | Z_{t+1})$  of the specific mode, which is used to update the mode's probability.

The choice of how far to spread sigma points away from the mean is extensively discussed by Bitzer [23]. The author recommends the selection of sigma points that are not too close to the mean because otherwise, nonlinear effects away from the mean are not accounted for, although this may be required by the actual spread of the distribution. In this thesis, best results could be achieved when choosing what the author refers to as the *Gauss set* which corresponds to the parameters being set to  $\alpha = 1$ ,  $\beta = 0$ ,  $\kappa = 3$ .

The forward simulation of the MM-UKF is handled similarly to the one of the SMC method. For each UKF and thus for each mode, one multi-agent trajectory is generated and weighted with the corresponding probability  $p(U_i^j)$ .



## 4.4 Combinatorial Complexity

The trajectory prediction of multiple interacting agents, each having multiple hypotheses (or intentions), is combinatorial by nature. Specifying which agents are going to interact with each other in the near future and which aren't is cumbersome and error-prone, as there are often chains of dependencies that are hard to foresee. Thus, it is impractical to divide the set of agents into non-interacting subsets. Including all agents in one DBN (as proposed previously) allows to model all potential interactions and predict the scene in a combinatorial manner. However, the number of possible intention combinations  $|\mathcal{C}|$  in this *full combinatorial* model grows exponentially with the number of considered agents. It is given by the multiplication of the number of single agent intentions for all  $K$  agents:

$$|\mathcal{C}| = \prod_{i=1}^K \sum_{r \in \mathcal{R}^i} \sum_{\mathcal{M}_r^i} 1. \quad (4.32)$$

For the special case of each agent having the same number of possible routes  $|\mathcal{R}|$  and the same number of possible maneuvers  $|\mathcal{M}|$  given each route, mathematically

$$|\mathcal{R}| = |\mathcal{R}^i|, \quad |\mathcal{M}| = |\mathcal{M}_r^i| \quad \forall i \in [1, K], \forall r \in \mathcal{R}^i, \quad (4.33)$$

this can be simplified as

$$|\mathcal{C}| = (|\mathcal{R}||\mathcal{M}|)^K. \quad (4.34)$$

In complex and crowded traffic situations, the enumeration and the tracking of all these combinations quickly becomes computationally infeasible. Explicitly considering all of these combinations thus often limits estimation and prediction algorithms to either non-real-time applications or to scenarios with only few agents having few distinct hypotheses. Furthermore, subsequent software modules such as ego-vehicle behavior generation might also be unable to cope with all possible combinations. Thus, reasonable pruning is of high importance.

### 4.4.1 Pruning in Full-Combinatorial DBN

As the number of hypotheses that can be tracked in real-time is limited, it can be necessary to only consider a subset of all combinations in complex situations. One way of choosing this subset is by initially picking hypotheses based on their prior probabilities, whereas all other combinations are neglected at first. However, for a high number of possible combinations (and insufficient number of tracked hypotheses), the probability that the correct combination has also been picked only based on prior information can be rather low, as demonstrated with the following exemplary calculation:

Assume a complex traffic situation with ten agents, each having three different routes and two maneuvers given each route, such that each agent has six distinct high-level intentions in total. The total number of possible combinations is  $|\mathcal{C}| = 6^{10} \approx 6 \cdot 10^7$ . The maximum number of tracked hypotheses in parallel to still meet the runtime requirements depends on the complexity of the inference method (e.g., SMC, MM-UKF), the

used transition models (e.g., rule-based, neural network-based), the desired prediction horizon and the number of agents to be predicted. If we assume about 1000 hypotheses can be tracked in parallel, the probability of initially also picking the correct hypothesis (considering a uniform prior) is only 0.001654%. Therefore, replacing unlikely combinations with more likely hypotheses is key to successful long-term tracking and estimation. Over the course of tracking, hypotheses that became less likely can gradually be replaced with new, so far untracked ones. Such a pruning method can be employed by the previously presented full-combinatorial DBN for both the SMC method (within the initial particle sampling and resampling steps) and the MM-UKF method (within the initial mode selection and replacing of modes steps), allowing it to handle more complex urban scenarios despite runtime limitations.

One downside of this method, however, is that one does not know which of the untracked hypotheses is the most likely given the past observations, as they have not been tracked yet and their posterior probabilities have not been estimated. Thus, the replacement of unlikely hypotheses with untracked ones can only be done based on prior probabilities or at random. Therefore, it stands to reason that determining relevant combinations in a more informed manner without the need of tracking all possible combinations is needed.

### 4.4.2 Interacting Single-Agent DBNs

As the actions of the single agents are modeled to be conditionally independent, each agent can be predicted on its own for one time step given the current kinematic states of all other agents. This allows to divide the filtering problem into multiple *interacting single-agent estimators* that share statistics about their agents' current kinematic states. For each agent, one DBN tracks only the belief of this single agent while incorporating information about the pose and velocity of all other agents needed to determine the respective interaction-aware actions. After each time step, these shared statistics are updated according to the respective single-agent DBNs' updated state. Thus, instead of estimating the intentions in a combined manner (e.g.,  $V^i$  turns right while  $V^j$  turns left), the intention probabilities of the single agents are determined individually (e.g.,  $V^i$  turns right independently of what other agents are going to do). This allows to greatly reduce tracking complexity while still accounting for possible interactions between any of the considered agents.

The following sections shows how based on the estimated single-agent probabilities, the most relevant combinations of the multi-agent setting can be determined in an informed way without the need of explicitly deriving and tracking all possible combinations. These multi-agent combinations can then be simulated forward to generate the most likely interaction-aware probabilistic scene developments in a combinatorial fashion again.

#### 4.4.2.1 Filtering with Interacting Single-Agent DBNs

For each agent, one DBN is created which estimates the agent's kinematic state and intentions over time. In order to account for possible interaction when determining the

action of an agent, the kinematic states of the other agents have to be provided to this DBN. In each time step, each single-agent filter thus broadcasts a statistic about its agent’s pose and velocity to the other filters. This statistic consists of the mean kinematic state given the most likely route and maneuver of the agent. As the route and maneuver are the strongest causes for multi-modality, the belief over the kinematic state given a specific route and maneuver intention stays mostly unimodal. Thus, the mean given a route and maneuver is actually a representative statistic, in contrast to, e.g., the mean over all possible routes and maneuvers, which for example might be located in between two lanes.

It has to be noted, that these statistics do not incorporate the prevailing uncertainty in the kinematic states of the other agents, which is in fact neglected within the filtering stage. However, as the kinematic states can be measured with low noise and is filtered over time, their uncertainty is rather small compared to the intention and action uncertainty. The main uncertainty in the filtered state (at the current time step) is in fact given by the agents’ intentions, as intentions cannot be measured directly but are estimated indirectly given uncertain behavior models and measurements of the kinematic state. For the one-step predictions needed for filtering, the intentions of other agents are modeled to be irrelevant given their kinematic states (cf. Fig. 2.1).

Within the single-agent DBNs, the correlation between the intentions of multiple agents is not determined. For the measurement update, the correlation between the states of multiple agents is ignored as well. Although using this statistic is a strong simplification, the evaluation indicates that it still results in a reasonable approximation of the full combinatorial solution.

After each prediction step, the kinematic states and the intention probabilities are updated for each agent according to the new measurement. This allows to continuously track all single-agent hypotheses with linear complexity and determine the most likely single-agent intentions (regardless of the intentions of all other agents). The number of hypotheses to be tracked using the single-agent DBNs (again, given the same number of possible routes and maneuvers for all agents) is given by

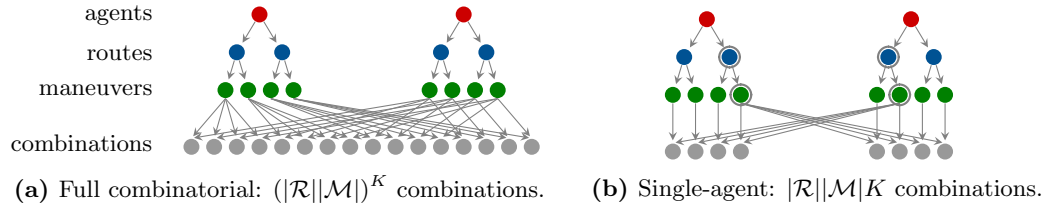
$$|\mathcal{C}| = \sum_{i=1}^K \sum_{r \in \mathcal{R}^i} \sum_{m \in \mathcal{M}_r^i} 1 = |\mathcal{R}| |\mathcal{M}| K, \quad (4.35)$$

which is linear in the number of considered agents compared to the exponential complexity of the full combinatorial tracking. An illustration of the tracking complexities of the full-combinatorial filter and the single-agent filter is shown in Fig. 4.9.

#### 4.4.2.2 Forward Simulation of Most Likely Combinations

As the actions of multiple agents might become interdependent for prediction horizons of more than one time step, the forward simulation still has to be executed in a combinatorial fashion: The intentions of an agent will affect its *future* kinematic states and in turn may also affect the *future* actions of other agents (as they depend on the kinematic states of all agents). Thus, all agents’ intentions need to be considered during forward

#### 4 Inference Methods



**Figure 4.9.** Tracking complexities of the *full-combinatorial* and the *single-agent* models for the special case of each of the  $K$  agents having the same number of  $|\mathcal{R}|$  routes and  $|\mathcal{M}|$  maneuvers per route (here  $K = |\mathcal{R}| = |\mathcal{M}| = 2$ ). For subfigure (b), the most likely single agent intentions are marked with circles.

simulation and a recombination of the single-agent filters is necessary. The number of possible combinations to be predicted is still the same as within the full combinatorial DBN ( $|\mathcal{C}| = (|\mathcal{R}||\mathcal{M}|)^K$ ), which might be too many to allow for real time application. But using the single-agent intention probabilities, it is possible to draw conclusions about the likelihood of the combinations, allowing for an informed selection (opposed to a random one or solely based on priors). A subset of all possible combinations can be chosen to reduce computational costs without the need to explicitly track all hypotheses. As the correlation between intentions has been neglected within the single-agent filtering stage, the recombination is done assuming independence of the intentions of different agents. Evidently, this implies that the recombination probabilities do not necessarily coincide with the actual combination probabilities (if tracked in a full-combinatorial fashion).

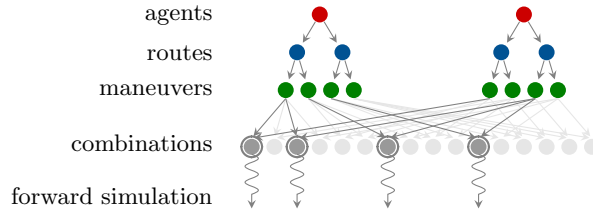
For the sake of clarity, the different types of intentions of an agent are subsumed within the *generalized intention*  $\iota$  of an agent. It specifies all intention types of interest in a combined manner, such as route and maneuver intentions in this work. For example, if there are two possible routes and two maneuvers for the first route and only one maneuver for the second route, then there are three different generalized intention possibilities  $\iota_1 = (r_1, m_1), \iota_2 = (r_1, m_2), \iota_3 = (r_2, m_1)$ . Each agent  $V^i$  has an unknown intention  $\iota^i$  which is part of the set of possible distinct intentions  $\mathcal{I}^i = \{\iota_1^i, \dots, \iota_{|\mathcal{I}^i|}^i\}$ . The estimated probability of an intention  $\iota_j^i$  is denoted as  $p(\iota_j^i)$ . For each agent, the probability distribution over all intentions is given by  $p(\iota^i) = [p(\iota_1^i), \dots, p(\iota_{|\mathcal{I}^i|}^i)]^\top$ .

A combination  $\zeta_j \in \mathcal{C}$  of intentions of agents  $[V^1, \dots, V^K]$  describes a vector of generalized intentions of all agents  $\zeta_j = [\zeta_j^1, \dots, \zeta_j^K]$ , with the indices  $\zeta_j^i \in \{1, \dots, |\mathcal{I}^i|\}$ . The estimated probability of such a combination is given by

$$p(\zeta_j) = p(\iota_{\zeta_j^1}^1) p(\iota_{\zeta_j^2}^2) \cdots p(\iota_{\zeta_j^K}^K), \quad (4.36)$$

assuming independence between the intentions of different agents. An example of the recombination of the most likely hypotheses based on the estimated single-agent probabilities used for the forward simulation is illustrated in Fig. 4.10.

For improved readability, the following algorithm only uses the respective *indices* of the generalized intentions  $\zeta_j = [\zeta_j^1, \dots, \zeta_j^K]$  to uniquely identify a combination, with



**Figure 4.10.** Forward simulation of  $n$  (here  $n = 4$ ) most likely combinations for two agents with single-agent intention probabilities  $p(\iota^1) = [0.4, 0.3, 0.2, 0.1]$  for the left agent and  $p(\iota^2) = [0.3, 0.1, 0.5, 0.1]$  for the right agent (intentions ordered from left to right). The resulting most likely combinations are given by  $\zeta_1 = [\iota_1^1, \iota_3^2]$ ,  $\zeta_2 = [\iota_2^1, \iota_3^2]$ ,  $\zeta_3 = [\iota_1^1, \iota_1^2]$ , and  $\zeta_4 = [\iota_3^1, \iota_3^2]$ .

**Table 4.1**

SINGLE-AGENT INTENTION PROBABILITIES AND CORRESPONDING MOST LIKELY COMBINATIONS

(a) SINGLE-AGENT PROBABILITIES					(b) MOST LIKELY COMBINATIONS	
	$V^1$	$V^2$	$V^3$	$V^4$	Combination	Probability
$p(\iota_1)$	0.7	0.6	0.4	0.5	$\zeta_1 = [1, 1, 1, 1]$	$p(\zeta_1) = 0.0672$
$p(\iota_2)$	0.2	0.3	0.3	0.2	$\zeta_2 = [1, 1, 2, 1]$	$p(\zeta_2) = 0.0504$
$p(\iota_3)$	0.1	0.1	0.3	0.2	$\zeta_3 = [1, 1, 3, 1]$	$p(\zeta_3) = 0.0504$
$p(\iota_4)$	-	0.1	-	0.1	$\zeta_4 = [1, 2, 1, 1]$	$p(\zeta_4) = 0.0420$
					$\vdots$	$\vdots$
					$\zeta_{144} = [3, 4, 3, 4]$	$p(\zeta_{144}) = 0.0003$

each index representing the intention of the respective agent  $V^i$ . In order to determine the  $n$  most likely combinations, a graph search in the space of possible combinations (or intention indices)  $\mathcal{C}$  is applied. For this purpose, in a first step, the intentions of the single agents are sorted by descending probability. The combination of highest probability is directly given by  $\zeta_1 = [1, \dots, 1]$ . Given a specific combination, a *child* relation is defined by having exactly the same indices except for a single index that is incremented by one (e.g.,  $[2, 1, 3, 2]$  is a child of  $[2, 1, 3, 1]$ ). The function  $children(\zeta)$  determines the set of all possible children of  $\zeta$ , which form a valid combination of intentions (no index is out of range). As the intentions of the single agents are sorted by descending probability, a child cannot have higher probability than its parent:

$$\forall \zeta_{\text{child}} \in children(\zeta) : p(\zeta_{\text{child}}) \leq p(\zeta) \quad (4.37)$$

Thus, in order to find the best descendant of any combination, it is sufficient to only check its direct children. This allows to find the  $n$  most likely combinations without the need to calculate the probabilities of all possible combinations (which on its own, although a simple calculation, might already fail to meet runtime requirements). An example of this procedure is depicted in Tab. 4.1 with the sorted single-agent intention probabilities and the resulting most likely combinations. The corresponding search algorithm to determine the  $n$  most likely combinations can be found in Algorithm 3.

#### 4 Inference Methods

The main advantages of the single-agent intention estimation and the presented re-combination of hypotheses are as follows: The filtering complexity is reduced from exponential to linear in the number of considered agents. The number of considered intention combinations for forward simulation can be adjusted according to runtime requirements, while neglecting the other, less likely hypotheses. This subset of combinations is selected in each time step in an informed manner based on the estimated single-agent intention probabilities (instead of selecting at random or solely based on priors), without the need to track all of them explicitly. This furthermore implies that instead of pruning possible combinations irrevocably from the beginning or over the course of tracking, pruning is done in each time step, allowing to reconsider combinations that became more likely again.

---

**Algorithm 3** Determine  $n$  most likely combinations

---

**Input**  $n, p_{\mathcal{I}^1}, \dots, p_{\mathcal{I}^K}$   
**Output**  $\mathcal{C}_n = [\zeta_1, \dots, \zeta_n]$ ,  
with  $p(\zeta_a) \geq p(\zeta_b) \geq p(\zeta_c), \quad \forall a < b \leq n, \quad a, b, n \in \mathbb{N}^+, \quad \zeta_c \notin \mathcal{C}_n$

- 1: **procedure** GETNMOSTLIKELYCOMBINATIONS
- 2:    $\zeta_1 \leftarrow [1, 1, \dots, 1]$  ▷ initialize search with best combination
- 3:    $\mathcal{C}_n \leftarrow [\zeta_1]$
- 4:    $openset \leftarrow children(\zeta_1)$
- 5:    $i \leftarrow 1$
- 6:   **while**  $i < n$  **do**
- 7:     **if**  $openset \neq \emptyset$  **then**
- 8:        $p_{best} \leftarrow 0$
- 9:       **for**  $\zeta \in openset$  **do** ▷ find best combination in openset
- 10:         **if**  $p(\zeta) > p_{best}$  **then**
- 11:            $\zeta_{best} \leftarrow \zeta$
- 12:            $p_{best} \leftarrow p(\zeta)$
- 13:        $\mathcal{C}_n \leftarrow [\mathcal{C}_n, \zeta_{best}]$  ▷ add combination and adjust openset
- 14:        $openset \leftarrow openset \setminus \zeta_{best}$
- 15:        $openset \leftarrow openset \cup children(\zeta_{best})$  ▷ no duplicates, as is a set
- 16:        $i \leftarrow i + 1$
- 17:     **else** ▷ no combination left
- 18:     **break** ▷ early stop, less than  $n$  combinations

---

## 5 Interrelated Ego Motion Planning and Prediction of Surrounding Agents

For an autonomous vehicle to drive in a foresighted and cooperative manner, it is beneficial to incorporate the intended motion of surrounding agents within its own motion planning. However, as intentions cannot be measured directly and the future motion of surrounding agents often highly depends on the future motion of the ego-vehicle, this incorporation has proven challenging. Traditionally, the problems of planning and prediction are handled separately, such that the prediction is assumed to be independent of the ego-motion and is provided to the motion planning algorithm as a fixed input. Especially in situations where decisions of traffic participants are highly coupled, the assumption of independent motion becomes invalid. This motion interdependency of the ego-vehicle and surrounding vehicles yields specific challenges, as it is not possible to separate the problems of prediction and planning anymore.

The previously presented estimation and prediction framework is able to both infer hidden intentions of traffic participants and to predict their trajectories in an interaction-aware manner. However, a justifiable question remains: how can this prediction and estimation approach be used to plan interactive ego-vehicle trajectories by an autonomous vehicle? This chapter presents two frameworks for combined ego-vehicle motion planning and prediction of surrounding traffic participants that account for interaction between all agents using such a Bayesian estimation framework. We include all agents including the ego-vehicle within the state space to account for possible interaction and the influence the ego-vehicle's actions have on the surrounding agents. Both approaches utilize a continuous state space, are online-capable (i.e., do not need to pre-compute scenario-specific information) and can handle an arbitrary number of agents and an arbitrary road layout. Furthermore, they consider uncertainty in measurement and human behavior including the prevailing multi-modality given by the drivers' intentions.

The first approach formulates the problem as a partially observable Markov decision process (POMDP), in which the ego-vehicle is taking actions sequentially to optimize a reward function and the surrounding agents are predicted sequentially utilizing a DBN as presented in Chapter 2 with behavior models such as the ones from Chapter 3. Incorporating the DBN within the POMDP allows to solve the interrelated problems of ego-motion planning and the prediction of others in a combined way by utilizing incremental forward simulation, while accounting for interaction between all agents. POMDPs naturally allow to handle uncertainties in both measurements as well as in the intentions and behavior of other agents. We employ a point-based solver based on the upper confidence bound (UCB) algorithm, which effectively transforms the problem into a Monte Carlo tree search, allowing arbitrary belief distributions. To achieve real-time capabil-

ity, we discretize the action space into discrete accelerations and employ a path-velocity decomposition based on the lane geometry. Lateral accelerations are optionally added for allowing lane changes. One major advantage of the POMDP approach is that it accounts for potential future observations during planning. In the forward simulation phase (or planning phase), the algorithm knows that observations will be coming in at certain points and that those observations will yield information about the state of other agents, allowing additional branching depending on what observation might be received. For example, at the current time, we might not know whether another agent is going to turn left or go straight at the next intersection, but we know that in 5 s we will have observations that allow us to infer the correct route with high probability. This allows the ego-vehicle to drive less conservatively.

In the second approach, we formulate the problem as a cooperative multi-agent planning problem that is divided into a high-level plan (a multi-agent maneuver or *collective maneuver* based on homotopy classes) and a low-level plan (a multi-agent trajectory) given a specific homotopy class. The prediction problem is thus transformed into a planning problem, assuming other agents try to optimize a specific cost function and choose their actions accordingly. This effectively circumvents the compounding error problem which can strongly affect machine learning-based single-step models. A Bayesian multiple model filter (similar to the DBN from Chapter 2) is used to estimate the intention probabilities of the other agents utilizing the derived trajectories as transition models (opposed to using the one-step behavior models from Chapter 3). The ego-vehicle can then either comply with the most likely maneuver, or try to convince other agents of a more beneficial maneuver (e.g., with lower overall costs). In contrast to the POMDP, this method allows us to optimize in a continuous and two dimensional action space and thus results in smoother trajectories that can already be executed using model predictive control (MPC). Another major difference is that the multi-agent planning approach explicitly enumerates the possible homotopy classes and retrieves optimal trajectories for all of them, whereas the POMDP tries to find the best option implicitly by only optimizing accelerations, not caring about suboptimal options. This makes the multi-agent planning approach more explicit by providing intention probabilities and costs for all homotopy options, allowing to select suboptimal maneuvers (given the cost function) to support other agents' suboptimal choices (given our intention estimation). One simplification is given by how this approach deals with uncertainty: Given a specific homotopy class, the low-level uncertainty is neglected during the trajectory optimization and only the mean state is utilized. Gaussian noise is then added to the state to account for inaccuracies in prediction. This allows for an efficient optimization considering deterministic actions and allows for a computationally cheap Kalman-filter-based intention estimation, but comes at the cost of less realistic uncertainty estimates. Another difference is that it assumes cooperation of other agents, as their behavior models are based on the optimization of a joint cost function. On the one hand, this allows for more interactive scenarios (e.g., creating lateral space on one's lane for letting other agents pass more easily), but also represents a strong assumption which is not always true. A major downside is that the multi-agent planning approach does not reason about future observations.



Thus, both approaches come with their advantages and disadvantages. Given future advances in computational power, some of the advantages of the multi-agent planning approach (e.g., continuous action space, longer planning horizon) could also be achieved with the POMDP approach. Furthermore, by adding cost-terms for making other agents deviate from their intentions, one could account for deciding for “suboptimal” maneuvers (w.r.t. the original reward function) to support other agents’ decisions.

Interaction-aware and cooperative motion planning for autonomous vehicles is an enormous open field of research. As the main focus of this thesis is the prediction of surrounding traffic participants, this chapter mainly intends to show how the presented prediction approach (and similar approaches) can be utilized for ego-vehicle planning, but does not go into much detail regarding trajectory and behavior planning algorithms in general. This chapter is based on the author’s previous publications [161]–[163], [165]. The interested reader is referred to these publications for further information about interaction-aware motion planning for autonomous vehicles.

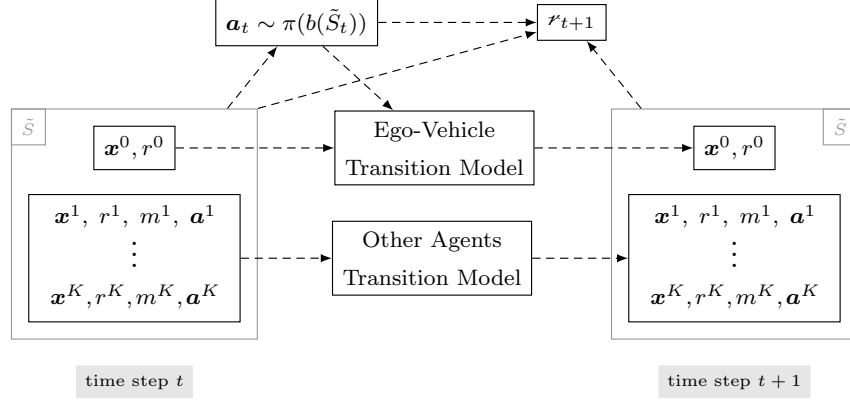
## 5.1 Partially Observable Markov Decision Process

Let’s consider the ego-vehicle, denoted as  $V^0$ , to be part of the state space consisting of all vehicles in a scene  $\mathcal{V} = \{V^0, \dots, V^K\}$ . At each time step  $t$ , the ego-vehicle has to choose an action  $\mathbf{a}_t^0 \in \mathcal{A}^0$  that transforms its current state  $\mathbf{x}_t^0$  to the subsequent time step  $\mathbf{x}_{t+1}^0$  and for which it receives a reward  $r_{t+1}^0$ . To improve readability, the superscript  $(\cdot)^0$  for the action and reward of the ego-vehicle are omitted for the rest of this section. Broadly speaking, the goal of the ego-vehicle is to optimize its actions in order to receive high rewards.

We propose two distinct discrete action spaces. The first one employs a path velocity decomposition and allows for different discrete longitudinal accelerations on the pre-selected route of the ego-vehicle, resulting in a lane following agent. The second one additionally allows for discrete lateral velocities in order to enable the ego-vehicle to conduct lane changes and lateral merges.

In contrast to the other agents, there is no uncertainty in the route intention of the ego-vehicle and the maneuver is given implicitly by the planned actions that are the result of the optimization. Thus, the state of the scene is given by  $\tilde{S} = [X, R, M^\theta, A^\theta] \in \tilde{\mathcal{S}}$ , with  $M^\theta, A^\theta$  denoting the maneuvers and actions of all agents but the ego-vehicle and  $X, R$  the kinematic states and routes of all agents including the ego-vehicle. The map is assumed to be given deterministically and is not included within the scene state for improved visualization. As the state of the scene is not fully observable, the ego-vehicle tracks a belief distribution  $b_t$  over all possible states at time  $t$ , starting with the initial belief  $b_0$ . The observations  $Z \in \mathcal{Z}$  and the corresponding observation probability  $\mathcal{O}(\tilde{S}, Z) := p(Z|\tilde{S}) = p(Z|X)$  are analogous to the ones of the DBN in Chapter 2, but additionally include the ego-vehicle.

The transition probability  $\mathcal{T}(\tilde{S}_{t+1}, \tilde{S}_t, \mathbf{a}_t, \text{map}) := p(\tilde{S}_{t+1}|\tilde{S}_t, \mathbf{a}_t, \text{map})$  defines how the complete scene state changes over time given the action of the ego-vehicle. The previously proposed DBN can be utilized to track and predict the scene state by adding the ego-



**Figure 5.1.** Schematic view of POMDP with integrated DBN. The DBN state space is extended by the ego-vehicle such that the transition models of the surrounding agents take the ego-vehicle into account. The action of the ego-vehicle is determined by the POMDP policy  $\pi$  that depends on the complete state space. The reward is determined by a function of old state, new state and chosen action.

vehicle to the state space and adding a transition model that depends on the chosen action as depicted in Fig. 5.1. The reward the ego-vehicle receives for action  $\mathbf{a}_t$  is defined as a function of the old state, the chosen action, the resulting new state and the map as  $r_{t+1} = \mathcal{R}(\tilde{\mathcal{S}}_{t+1}, \tilde{\mathcal{S}}_t, \mathbf{a}_t, \text{map})$ . In order to set a higher importance on short-term rewards compared to rewards that are in the distant future, typically a so-called discount factor  $\gamma \in [0, 1]$  is utilized.

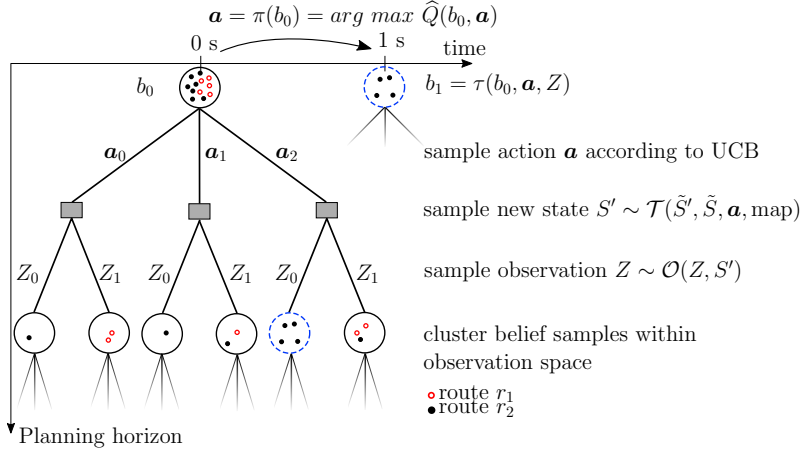
The POMDP is defined completely by the tuple  $\langle \tilde{\mathcal{S}}, \mathcal{A}, \mathcal{T}, \mathcal{Z}, \mathcal{O}, \mathcal{R}, b_0, \gamma \rangle$ . The so-called policy  $\pi$  in a POMDP describes how an agent acts and is a mapping from belief state to action,  $\pi : b \mapsto \mathbf{a}$ . The objective of the ego-vehicle is to maximize its expected cumulative discounted future reward by choosing actions  $(\mathbf{a}_0, \mathbf{a}_1, \mathbf{a}_2, \dots)$  according to the optimal policy

$$\pi^* := \arg \max_{\pi} \mathbb{E} \left[ \sum_{\tau=0}^{\infty} \gamma^{\tau} \mathcal{R}(\tilde{\mathcal{S}}_{\tau+1}, \tilde{\mathcal{S}}_{\tau}, \pi(b_{\tau}), \text{map}) \mid b_0, \pi \right] \quad (5.1)$$

Typical reward functions consist of terms that punish collisions, traffic law violations, and account for smoothness and efficiency (e.g., by punishing accelerations and jerk). As the belief transition function includes the prediction models of the surrounding agents, which each depends on the states of all agents including the ego-vehicle, the resulting policy of the autonomous vehicle accounts for the interdependency of all agents in the scene and thus also for its own influence on the actions of others.

### 5.1.1 Solution Method

There is a big variety of possible solution methods for POMDPs. As autonomous driving requires decision making in real time, typically, approximative methods are employed.

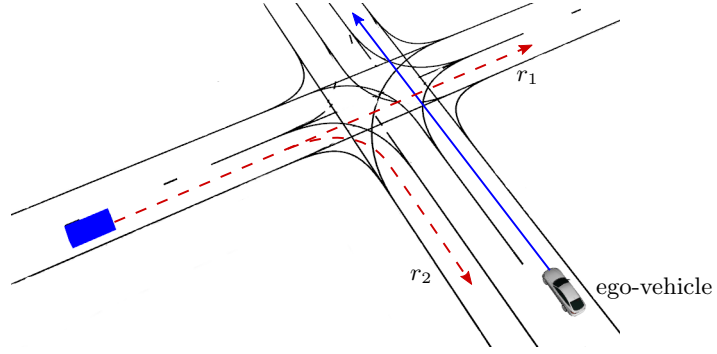


**Figure 5.2.** Exemplary approximation of the belief tree of scene from Fig. 5.3 using Monte Carlo sampling of episodes starting at the current belief. The route of the other agent is uncertain and influences the transitions and possible observations. Previously published in [162]. © IEEE 2018

As proposed in our previous work [162], one method that is well suited for the aforementioned POMDP is the so-called adaptive belief tree (ABT) [76]. With ABT, the current belief state is represented by a set of particles of possible state instances. The optimal solution is approximated by sampling potential future episodes according to an action-choosing regime and the given transition probabilities, allowing to determine the corresponding rewards. After sampling many episodes, it is possible to evaluate which trajectories are most likely to result in high rewards, given the probabilistic models of the surrounding agents. Utilizing such a point-based solver effectively transforms the problem into a Monte Carlo tree search (MCTS) problem and allows for a seamless integration of the previously presented sampling-based prediction models. The sampled episodes span a belief tree and are used to approximate the  $Q$ -value function  $Q(b, \mathbf{a})$  that represents the expected return of a specific action  $\mathbf{a}$  in a specific belief state  $b$  followed by actions of the optimal policy and thus describes how good a specific action is given a belief.

Such a belief tree is exemplarily depicted in Fig. 5.2 with the corresponding scene in Fig. 5.3. The ego-vehicle has an uncertain belief over the route intention of the other vehicle, thus it needs to be able to handle both hypotheses. Modeling this problem as a POMDP allows to account for the fact that the ego-vehicle will get further observations in the future, enabling it to drive in a less conservative manner. The forward simulation is executed similarly to the one presented in Chapter 2, but additionally includes the ego-vehicle that chooses its actions during the forward simulation according to the UCB algorithm for trees. Furthermore, possible future observations are taken into account. Similar beliefs are clustered within the observation space as depicted in Fig. 5.2.

Creating the search tree by sampling actions according to the UCB algorithm allows to trade off between exploration and exploitation within the forward simulation stage and together with efficient rollout policies, it is possible to get a grip on the exploding



**Figure 5.3.** Exemplary scene with the ego-vehicle driving straight over the intersection, having an uncertain belief of the other agent's route. Previously published in [162]. © IEEE 2018

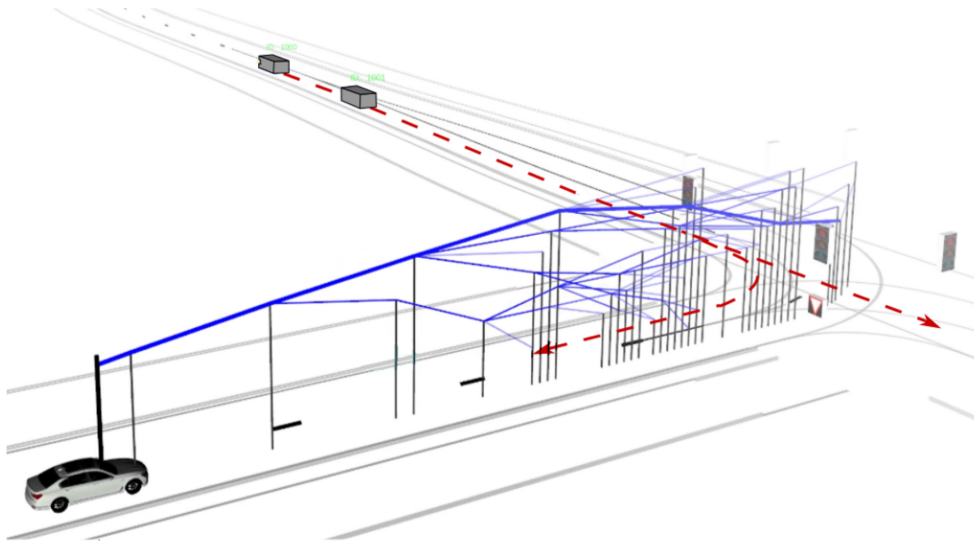
search tree. Initially, the actions are sampled according to a uniform distribution of all actions that have not been sampled yet, denoted as  $\mathcal{A}'$ . After each action has been sampled at least once, a trade-off between how good an action has performed so far (determined by the approximated  $Q$ -function  $\widehat{Q}(b, \mathbf{a})$ ) versus how often an action has been tried already ( $H_{(b, \mathbf{a})}$ ) compared to the number of total episodes starting from that belief ( $H_b$ ) determines which action to choose next. Altogether, the actions are sampled according to

$$\mathbf{a} = \begin{cases} \sim \mathcal{U}(\mathcal{A}') & , \text{ if } \mathcal{A}' \neq \emptyset \\ \arg \max_{\mathbf{a} \in \mathcal{A}} \left( \widehat{Q}(b, \mathbf{a}) + c \sqrt{\frac{\log(|H_b|)}{|H_{(b, \mathbf{a})}|}} \right) & , \text{ otherwise.} \end{cases} \quad (5.2)$$

The parameter  $c$  allows to put focus on either exploration or exploitation. Given enough runtime, UCB asserts convergence to the optimal  $Q$ -function. In order to cope with the exploding width of the search tree, so-called rollout policies can be employed that choose actions according to some heuristic. In our previous work [162], we proposed to combine the Monte Carlo sampling with a rollout policy that combines a three-step  $A^*$  rollout followed by a constant velocity rollout. Choosing well suited rollout policies or learning value functions from data are active fields of research and can help improve convergence time drastically. The more episodes are sampled, the better the approximated  $Q$ -values get. Thus, ABT is an anytime algorithm, allowing for real time application. When the ego-vehicle has to choose an action to be actually executed, it picks this action greedily according to the approximated  $Q$ -value function  $\widehat{Q}(b, \mathbf{a})$  to maximize its expected cumulative discounted future reward:

$$\pi(b) := \arg \max_{\mathbf{a} \in \mathcal{A}} \widehat{Q}(b, \mathbf{a}) \quad (5.3)$$

In each new time step, the belief is updated with an actual measurement using simple rejection sampling: unlikely particles are rejected whereas likely particles are kept. Thus, the belief is tracked over time in an unweighted particle filter fashion. The belief transition from one time step to another given an observation  $Z$  and an executed action  $\mathbf{a}$  is denoted as  $b' = \tau(b, \mathbf{a}, Z)$ . This tracking allows to preserve already determined branches



**Figure 5.4.** Approximated optimal policy visualized by the velocity profile (shown in blue) along the route of the ego-vehicle. The magnitude of the longitudinal velocity corresponds to the z-axis of the plotted trajectories. The policy branches in cases in which different possible future observations lead to different optimal actions. Graphic taken from [61].

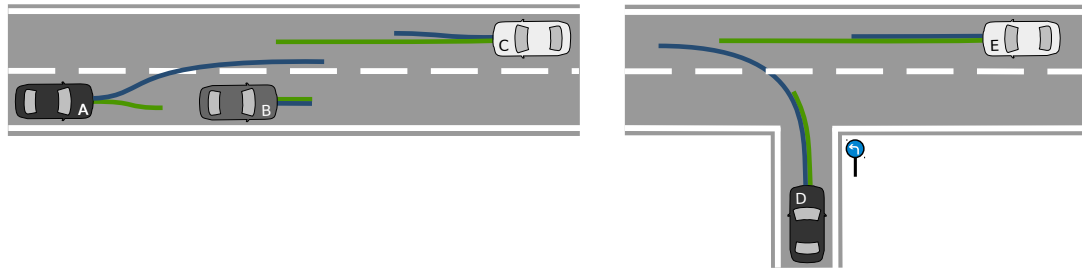
of the belief tree containing possible future episodes and corresponding approximated  $Q$ -values.

A visualization of the resulting policy is depicted in Fig. 5.4, showing the optimal actions depending on the possible future observations. It can be seen that the policy is not limited to a specific maneuver or homotopy class, but may choose a different maneuver for different observations.

As this solution method is sample-based, it comes with similar benefits as the previously presented SMC inference for the DBN. The reward function does not need to be differentiable and arbitrary distributions and non-linear transition functions can be represented. However, this sample-based nature also comes with the drawback that the approach cannot be considered to be completely safe. Thus, in order to deploy it in real traffic, we argue that one should add additional safety mechanisms that check for unlikely but critical outcomes that might have been missed during random sampling.

## 5.2 Multi-Agent Planning

This section proposes another approach for handling the interrelated problems of ego-vehicle motion planning and the prediction of surrounding agents in a combined fashion. In contrast to the previously presented POMDP which utilizes one-step prediction models (such as the ones presented in Chapter 3), this method is based on multi-agent planning and thus utilizes a planning-based prediction. Planning-based prediction of human behavior assumes actions to be rational and being the result of optimizing a cost function.

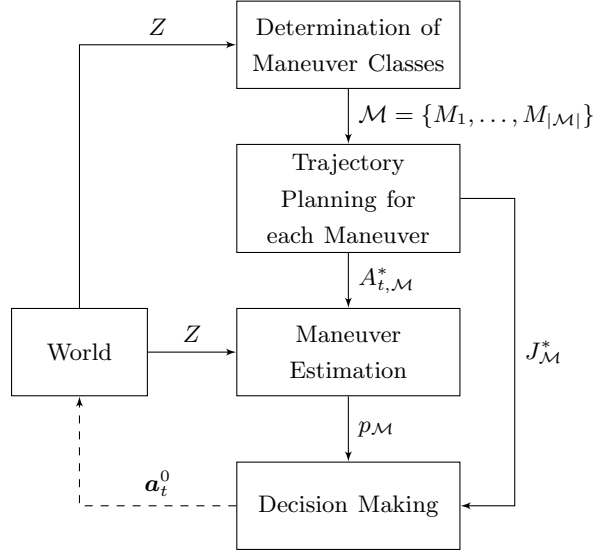


**Figure 5.5.** Two situations where the black and white vehicles’ motions are highly coupled. Multi-agent trajectories are shown for two collective maneuver hypotheses (blue and green). Both vehicles should intend to follow the same maneuver for collisions and misunderstandings to be avoided. Previously published in [165]. © IEEE 2017

In situations with very close interaction between agents, drivers tend to drive cooperatively and make space for one another, for example when merging in dense traffic or when overtaking with oncoming traffic. As exemplarily shown in Fig. 5.5 for two interacting agents, without anticipating the future behavior of the other driver, one can hardly decide for one of the two options. However, the own decision may influence the decision of the other driver as well. Modeling the behavior of multiple agents as a combined optimization problem with a shared cost function allows to represent such cooperative behavior and enables to include domain knowledge such as by defining constraints for collision avoidance and shaping the cost function according to typical behavior patterns.

As one cannot assume that humans will always decide for the actions with the lowest costs (according to the assumed cost function), we propose to plan *multiple* multi-agent trajectories that represent different local minima of the cost function and thus represent different possible outcomes of a scene. In order to derive these local minima, the space of possible solutions is subdivided into high-level multi-agent actions which we call *collective maneuvers* and which are based on the concept of trajectory homotopy. For each possible maneuver, one potential cooperative multi-agent trajectory is planned that should reflect common human behavior. To account for the corresponding homotopy constraints given by the derived maneuvers, the optimization problem is formulated using mixed-integer quadratic programming (MIQP). The resulting trajectories then serve as maneuver-dependent prediction models for all vehicles in a scene and thus allow for a comparison of abstract maneuver classes with observations of the actual motion of the surrounding agents. By utilizing a Bayesian filter similar to the DBN presented before, it is possible to estimate a probability distribution over these hypotheses given the observations of the behavior of the surrounding traffic participants. This distribution serves as a basis for the decision making of the autonomous vehicle: the ego-vehicle is able to either act according to the most likely hypothesis and thus to support the intentions of others or to try to convince them of the optimal maneuver that has the lowest costs for all agents (even if it might not be the most likely one).

Exactly as in the previously presented POMDP approach, both the ego-vehicle as well as all surrounding vehicles are represented within the state space. Thus, when



**Figure 5.6.** Maneuver estimation and decision making framework. Dashed arcs represent temporal dependencies whereas solid arcs represent causal dependencies. The variables are defined in the corresponding sections. Previously published in [165]. © IEEE 2017

solving the multi-agent planning problem, both tasks, prediction and motion planning, are solved at once. Thereby, the interdependency of prediction and planning is overcome, resulting in a cooperatively driving autonomous vehicle. Note that directly utilizing the trajectory of the joint optimization to control the ego-vehicle should not be considered safe, as it assumes others will behave cooperatively. We argue that one would have to add additional safety mechanisms that take potential uncooperative behaviors of others into account in order to deploy such an approach in real traffic. Fig. 5.6 provides an overview of the maneuver estimation and planning framework. The single components are explained in detail within the remaining sections.

### 5.2.1 Maneuver Determination

The multi-agent planning problem is approached by first defining the collective maneuvers on a multi-agent scale, describing the relative motion of multiple vehicles in a scene on a high level of abstraction. As the interrelation of decisions often arises from geometric collision avoidance constraints, the maneuver definition is based on the concept of trajectory homotopy.

This section proposes this collective maneuver representation and it shows how a set of possible maneuvers can be derived given a traffic scenario including multiple agents and the respective map with lane topology information.

### 5.2.1.1 Multi-Agent System

Assume a set of vehicles  $\mathcal{V} = \{V^0, \dots, V^K\}$  including the ego-vehicle  $V^0$  is considered within the multi-agent system. The state of vehicle  $V^i \in \mathcal{V}$  is given by  $\mathbf{x}^i = [s^i, v_s^i, d^i, v_d^i]^\top \in \mathbb{R}^4$ , consisting of the longitudinal and lateral Frenét-coordinates [152]  $s^i$  and  $d^i$  of the vehicle's center-position and the corresponding velocities  $v_s^i/d^i$ . To simplify the problem, changes in orientation, e.g., during lane changes, are neglected. The area occupied by this agent is approximated by a bounding box of length  $l^i$  and width  $w^i$  oriented along the tangent of the road's centerline:  $\mathbb{A}(s^i, d^i, l^i, w^i) := \mathbb{A}^i \subset \mathbb{R}^2$ . Furthermore, the area occupied by static obstacles is denoted as  $\mathbb{O} \subset \mathbb{R}^2$ . The multi-agent configuration is defined as

$$X = [\mathbf{x}^0, \dots, \mathbf{x}^K]^\top \in \mathcal{X} \subset \mathbb{R}^{4N} \quad (5.4)$$

within the collision-free configuration space

$$\mathcal{X} = \{X \mid \forall i, j \in \{0, \dots, K\}, i \neq j: \\ (\mathbb{A}^i \cap \mathbb{O} = \emptyset) \wedge (\mathbb{A}^i \cap \mathbb{A}^j = \emptyset)\}. \quad (5.5)$$

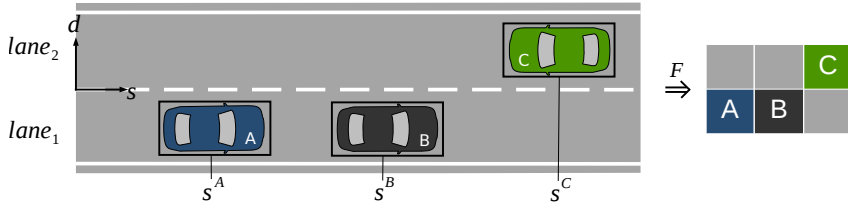
We define a multi-agent-trajectory to be a mapping from time span to configuration space:  $\gamma : [0, T] \rightarrow \mathcal{X}$ .

**Formation:** To describe the relative order of the different agents within a multi-agent configuration  $X$  in an abstract way, we define the so-called *formation*  $F(X)$ . Any formation  $F$  holds high-level information about the two dimensional relative position of objects along their lanes in longitudinal and lateral direction, neglecting the exact distances and lengths, only specifying a discrete relative ordering. This allows for a compact description of the maneuver relevant aspects of the current scene.

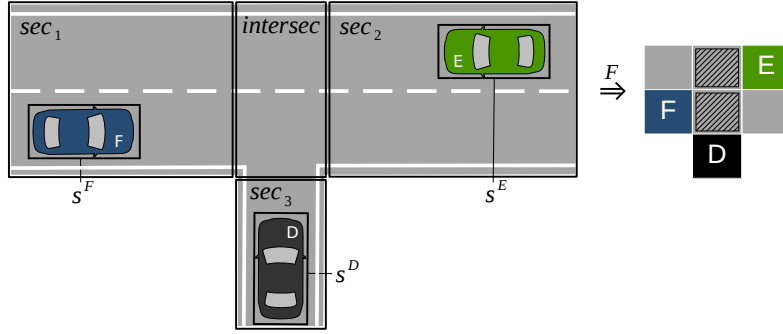
For every street section without intersections which can only contain multiple adjacent lanes (we denote those as *regular sections*), a local Frenét coordinate system is defined. The two dimensional positions of all vehicles on this regular section are first projected onto the longitudinal dimension. All vehicles are sorted in a list in ascending order according to their longitudinal position  $s$  of their reference point, storing the sequence of vehicles along the section. Although in reality, vehicles might overlap, it is not stored in the formation as it is not needed for the determination of the possible maneuvers. That means that, longitudinally, a vehicle can only be either in front or in the back of another vehicle. This longitudinal list is then extended to a second dimension using the discrete lane matching information. Therefore, in the formation representation, a vehicle can laterally only be on either the same lane or on any of the adjacent lanes to the left or to the right compared to another vehicle. An example of a scene and the resulting formation can be seen in Fig. 5.7(a). Generally, each cell within a formation is at most occupied by one vehicle. Furthermore, two vehicles cannot be on the same longitudinal column, i.e., laterally adjacent cells of an occupied cell are always unoccupied.

Using map data, it is possible to extract the different regular sections and how they are connected at intersections. The formations of the single sections are derived independently and then linked using intersection cells (see Fig. 5.7(b)). If two lanes of an





(a) Formation of a section using a local Frenét coordinate system and lanes.



(b) Formations of multiple sections are connected using intersection cells.

**Figure 5.7.** Derivation of the formation  $F(\mathbf{x})$  of a given configuration  $\mathbf{x}$ . Previously published in [165]. © IEEE 2017

intersection overlap, they share at least one common cell. If a vehicle is currently driving on an intersection, the formation will relocate the vehicle to the connecting lane of the subsequent regular section and queue it behind all other vehicles. Hence, intersection slots are never occupied in this formation representation, they only show how regular sections are connected and allow to set agents of different regular sections into relation.

### 5.2.1.2 Homotopy Classes

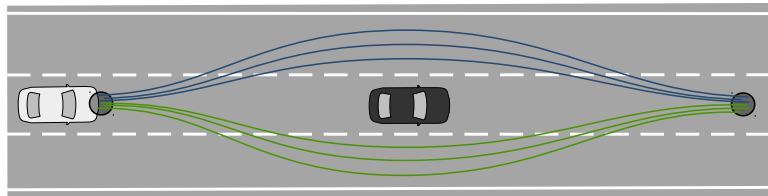
In the field of topology, two functions are said to be homotopic (or to be in the same homotopy class), if it is possible to *continuously deform* one into the other [21]. For that, let us first define a deformation variable  $\lambda \in [0, 1]$ . Then, mathematically speaking, two functions  $g_1, g_2 : Q \rightarrow R$  are said to be homotopic, if and only if a continuous homotopy  $H : Q \times [0, 1] \rightarrow R$  exists, such that

$$H(q, \lambda = 0) = g_1(q) \quad \wedge \quad H(q, \lambda = 1) = g_2(q) \quad \forall q \in Q. \quad (5.6)$$

The deformation variable  $\lambda$  thus describes “how close” the deformed function is to  $g_1$  and  $g_2$ .

In addition, for two functions  $g_1$  and  $g_2$  that share an identical mapping for a given subspace of the input space  $\tilde{Q} \subset Q$ , i.e.,

$$g_1(\tilde{q}) = g_2(\tilde{q}) \quad \forall \tilde{q} \in \tilde{Q}, \quad (5.7)$$



**Figure 5.8.** Single-agent path homotopy: All paths of same color are homotopic relative to the start and end position areas (gray circles). Previously published in [165]. © IEEE 2017

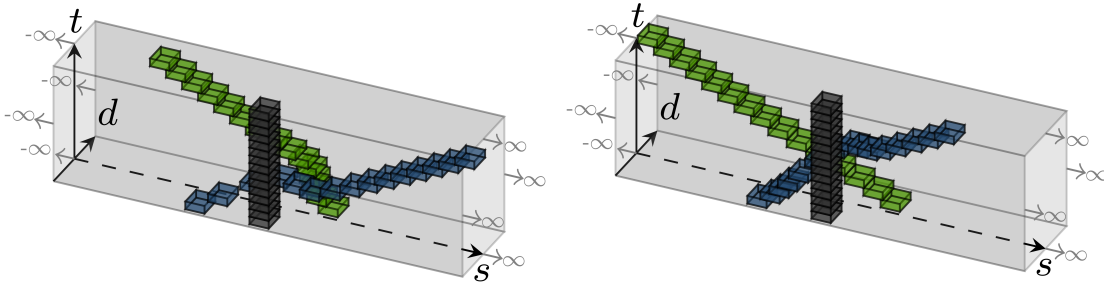
it is possible to define a homotopy *relative to this subspace*. A homotopy relative to a subspace forces the mapping of all elements of this subspace to be fixed during the complete continuous deformation process. Mathematically speaking, the functions  $g_1$  and  $g_2$  are called homotopic relative to the subspace  $\tilde{Q} \subset Q$ , if they are homotopic and

$$H(\tilde{q}, \lambda) = g_1(\tilde{q}) = g_2(\tilde{q}) \quad \forall \tilde{q} \in \tilde{Q} \wedge \forall \lambda \in [0, 1]. \quad (5.8)$$

**Single-Agent Path Homotopy:** The mathematical concept of homotopy is often utilized for analyzing paths in the field of robot motion planning. In this case, path homotopies are usually defined relative to the paths' start and end positions [19], [73]. Typically, the start position is known (given by the robot's current position) and it is desired to distinguish different end positions (e.g., goal regions) and whether obstacles are passed on the left or right side. To allow for inaccuracy in planning, this condition of exact start and end positions is often relaxed such that they only need to be in specific areas. This is illustrated in Fig. 5.8, where the white vehicle has a rough start position (gray circle on the left) and a desired goal area (gray circle on the right), and has to pass a static obstacle (black vehicle) on the way. Homotopically, it now has two options, either to pass that object on the left side or on the right side. While keeping the start and end positions in these fixed areas, there is no possibility to continuously deform a path that passes on the left (blue) to a path that passes on the right (green) without having a collision. Without keeping the start and end positions fixed, this would however be possible, and hence, no distinction between passing the obstacle on the left or right side could be made.

**Multi-Agent Trajectory Homotopy:** As we are not only interested in the paths the agents in a scene are going to take, but in their future trajectories, i.e., how their state is going to change over time, we extend the concept of path homotopy to the area of trajectories by adding the dimension of time, similarly to [20] and [49]. Furthermore, we introduce the following assumptions for structured environments: First, the constraint of keeping start and end states at fixed positions or within fixed areas is relaxed to keeping them within fixed formations. Secondly, road boundaries and the temporal planning horizon are interpreted as obstacles of infinite size, as shown in Fig. 5.9, such that it is not possible for a trajectory to run outside the road boundaries or beyond the planning horizon.

For two multi-agent trajectories to be homotopic, the trajectories of all vehicles need to allow a continuous transformation without colliding with obstacles or each other.



**Figure 5.9.** Multi-agent trajectory homotopy: Two non-homotopic multi-agent trajectories of the scene of the left part of Fig. 5.5 are shown. On the left, the blue agent passes the black agent first, then the green agent, on the right the other way around. Considering road boundaries and planning horizon to be obstacles of infinite size, different homotopy classes can be defined relative to the start and end formation. Previously published in [165]. © IEEE 2017

Two multi-agent trajectories  $\gamma_0, \gamma_1 : [0, T] \rightarrow \mathcal{X}$  with the same start and end formation  $F(\gamma_0(0)) = F(\gamma_1(0)) := F_0$  and  $F(\gamma_0(T)) = F(\gamma_1(T)) := F_T$  are said to be homotopic relative to  $\{F_0, F_T\}$ , if there exists a continuous homotopy  $H : [0, T] \times [0, 1] \rightarrow \mathcal{X}$ , such that

$$H(t, 0) = \gamma_0(t) \quad \wedge \quad H(t, 1) = \gamma_1(t) \quad \forall t \in [0, T] \quad (5.9)$$

and

$$F(H(0, \lambda)) = F_0 \quad \wedge \quad F(H(T, \lambda)) = F_T \quad \forall \lambda \in [0, 1]. \quad (5.10)$$

In addition to the lateral sides agents pass each other, trajectory homotopy also considers temporal aspects: Consider the scenario in Fig. 5.9 with a laterally bounded street such that at most two vehicles can be next to each other. Given the start and end formation, both the blue agent and the green agent have to pass the black static obstacle. As there is only space for one other vehicle next to the black obstacle, different *sequences* in which it is passed can be distinguished. As it is impossible to continuously transform the trajectories of the left part of Fig. 5.9 (blue agent passes first) to those of the right part (green agent passes first) without having a collision, they belong to different trajectory homotopy classes.

**Pseudo Multi-Agent Trajectory Homotopy:** The definition of multi-agent trajectory homotopy above allows to distinguish different maneuvers in many situations, but it also has some caveats. Except for the start and end formations, the definition above is solely based on the continuous multi-agent configuration including the areas of obstacles and agents, but it does not account for high-level information such as lanes. Consider the same scenario depicted in Fig. 5.9, but now assume that geometrically, there is enough space for all three vehicles to be on the same longitudinal position (even though there are only two lanes). With multi-agent trajectory homotopy, it is then not possible to distinguish these two maneuvers anymore, as it is possible to continuously

deform one into the other without a collision. The same is true for three or more adjacent lanes. However, as drivers tend to drive on lanes, typically there is still a discrete decision of a driver to either overtake before or after the oncoming vehicle. Thus, we want to be able to distinguish these maneuvers, even though they are homotopically indistinguishable in case the lanes are wide enough.

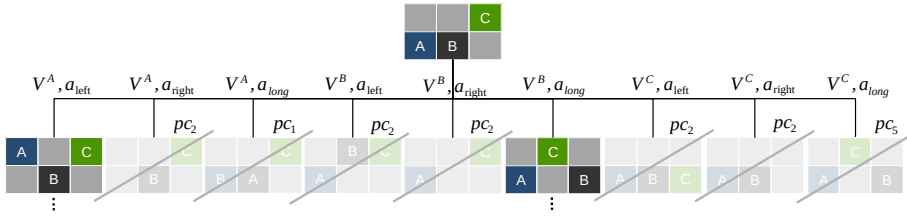
Therefore, we propose what we call a *pseudo-homotopy*, which is based on discrete formations and combines the advantages of homotopy and additional domain knowledge, explicitly considering the existence of lanes. Such a pseudo-homotopy comprises both spatial and temporal relations and represents a high-level description of the relative motion of multiple vehicles in a comprehensible way that can be used to distinguish various different maneuvers. Each pseudo-homotopy defines constraints that are utilized for the trajectory planning, which will be explained later in this chapter. First, if two longitudinally adjacent vehicles pass each other, the side on which they do is of interest. This is denoted as a *pairwise lateral relation* and can be derived by the discrete lane matchings. Secondly, in many situations the temporal order in which vehicles pass a specific area is also of importance. These areas which we call *critical areas* imply that they need to be passed sequentially by multiple agents, which might either be on an intersection where lanes overlap, or on a regular section in case there is a conflict on a lane such as in the overtake example above. This temporal order is denoted as *sequence of passing* of that critical area.

A maneuver  $M_i \in \mathcal{M}$  is given by a series of consecutive formations from  $F_0$  to  $F_{T_i}$ , including all *pairwise lateral relations* and all *sequences of passing* of the existing critical areas, and represents one pseudo trajectory homotopy class. The next section explains how these pseudo-homotopy classes are extracted from a given scene.

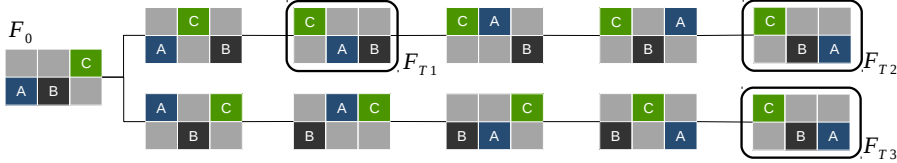
### 5.2.1.3 Formation Tree

As this thesis focuses on the uncertain prediction of human drivers, not only is the one optimal homotopy class (or maneuver) of interest, but all possible reasonable classes that human drivers might choose. To determine the set of possible maneuvers in every time step, first the current formation  $F_0$  of a given scene is identified using the positions of all considered vehicles and the available map data. By building a tree of formations with root node  $F_0$  which is expanded iteratively by selecting possible actions for all agents, different possible formation sequences can be found. Each such formation sequence  $[F_0, \dots, F_{T_i}]$  defines one possible future relative motion of all vehicles, i.e., a high-level abstraction of their future trajectories.

**Expansion:** The stepwise relative motion between agents is described on high abstraction level using a set of discrete formation actions  $\mathcal{A}_F = \{a_{\text{long}}, a_{\text{left}}, a_{\text{right}}\}$  from which vehicles can choose from. The actions do not represent continuous motion, but changes in the relative order. A vehicle-action-pair  $(V, a) \in \mathcal{V} \times \mathcal{A}_F$  transforms one formation into another. The vehicle that is in the longitudinal sequence of vehicles next in driving direction, independently of the lane, is denoted as the *vehicle ahead*. The lateral actions  $a_{\text{left}}$  and  $a_{\text{right}}$  represent lane changes whereas the longitudinal action  $a_{\text{long}}$  can represent two things: passing the vehicle ahead, resulting in a switch of their longitudinal



(a) First iteration of the maneuver expansion tree, showing how vehicle-action-pairs transform one formation to another. Formations that are pruned due to the pruning conditions  $pc_i$  are crossed out.



(b) Three possible maneuvers  $M_1, M_2, M_3$  consisting of the sequences from  $F_0$  to  $F_{T1}$  ( $V^A$  follows  $V^B$ ),  $F_{T2}$  ( $V^A$  overtakes  $V^B$  before  $V^C$  passes  $V^B$ ) and  $F_{T3}$  ( $V^A$  overtakes  $V^B$  after  $V^C$  has passed  $V^B$ ).

**Figure 5.10.** Formation tree expansion to identify reasonable maneuvers. Previously published in [165]. © IEEE 2017

order, or passing an intersection and queuing into the connecting lane on the subsequent regular section. Intersection cells only serve as a connection between regular cells and are never occupied within a formation.

For any vehicle-action-pair  $(V, a) \in \mathcal{V} \times \mathcal{A}_F$  and a given formation  $F_i$ , a new formation  $F'_i$  is derived. Fig. 5.10(a) shows an example of how the formation changes for the different possible vehicle-action-pairs given the initial situation from the same overtaking scenario as before. Different vehicle-action pairs can in fact result in the same new formation, as the same relative motion between two vehicles can be achieved by actions of either of both vehicles. For example, in the initial formation of Fig. 5.10(a),  $(V^B, a_{long})$  results in the same new formation as  $(V^C, a_{long})$ , as this just expresses that both vehicles pass each other. Through exhaustive expansion by checking all possible actions of all vehicles, the tree can be built up gradually.

**Pruning:** The following *pruning conditions* remove some of the possible actions during the tree building process, allowing to exclude the occurrence of loops, multiple identical transformations, and unfeasible or unlikely behavior:

- $pc_1$ : *Passing on same lane*: Trying to pass the vehicle ahead if it is on the same lane. We do not allow to call a longitudinal action in case the vehicle ahead is not on an adjacent lane.
- $pc_2$ : *Unsuitable lane change*: Changing to the same lane as the vehicle ahead is currently driving on, if the vehicle ahead is oncoming, as this is considered unreasonable. Furthermore, we forbid to perform a lateral action to a non existing lane, e.g., if the vehicle is already on the rightmost lane, action  $a_{right}$  is not allowed.

- $pc_3$ : *Reverse lateral action*: Changing to a lane the vehicle has already previously driven on within this maneuver, without having passed another vehicle. This is considered unreasonable behavior, as there was no effect in doing this lane change back and forth.
- $pc_4$ : *Re-overtaking*: Two vehicles passing each other that have previously passed each other within this maneuver. This is considered unreasonable behavior.
- $pc_5$ : *Existing new formation*: A new formation that already exists having the same parent formation. Removing these helps to avoid loops and duplicated formation sequences.

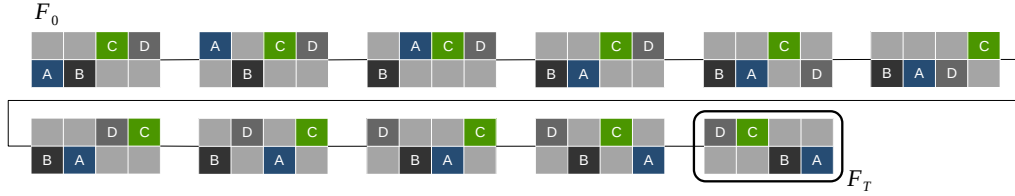
These heuristics do not consider reasonable vehicle dynamics as a prerequisite yet. Considering vehicle dynamics could help in removing impossible or unlikely maneuvers and thus further reduce complexity.

**Final formation:** A sequence of consecutive formations is only considered to be a valid maneuver if its final formation meets the *final formation conditions*, basically meaning that the scenario has been “resolved” by the agents:

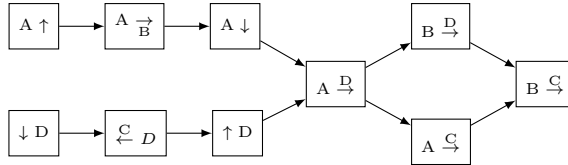
- $fc_1$ : All vehicles with different driving directions have passed each other.
- $fc_2$ : All vehicles are on a lane with correct driving direction.

The expansion process is continued even on possible final formations until no new formations can be found anymore. A graphical representation of a maneuver search tree including the possible final formations is shown in Fig. 5.10(b). For the situation *overtaking with oncoming traffic* (see left part of Fig. 5.5), three distinct maneuvers are extracted:  $M_1$  represents that  $V^A$  does not overtake but follows  $V^B$ ,  $M_2$  represents that  $V^A$  overtakes before the oncoming vehicle  $V^C$ , and  $M_3$  states that  $V^A$  overtakes after the oncoming vehicle  $V^C$ .

**Deriving Spatial and Temporal Constraints:** The formation sequences resulting from the exhaustive search given the initial formation describe the different possible developments of the scene. It is possible that multiple distinct formation sequences in the tree are similar in a way no distinction is desired. This happens when some actions in the formation sequence are not temporally dependent, i.e., it does not matter for the pseudo-homotopy class which action happens first. As an example, consider two vehicles  $V^A$  and  $V^B$  crossing an intersection using lanes that do not overlap (i.e., using different intersection cells). Then the temporal order in which they cross that intersection is not of importance, as they are not using the same critical area and are thus allowed to pass the intersection simultaneously. Therefore, trajectories of both options ( $V^A$  crossing first and  $V^B$  crossing first) can be continuously transformed into one another without a collision, meaning they belong to the same pseudo-homotopy class. Similarly, temporally independent actions can also happen on roads without intersections: Consider a vehicle  $V^A$  overtaking its preceding vehicle  $V^B$  while an oncoming vehicle  $V^D$  is overtaking another oncoming vehicle  $V^C$ . The order in which these two overtake actions happen is



(a) One possible formation sequence of the scene with initial formation  $F_0$ . Here,  $V^A$  overtakes  $V^B$  before  $V^D$  overtakes  $V^C$ , but this order does not influence the temporal dependency graph and thus changing it would not change the pseudo-homotopy class.



(b) Resulting temporal action dependency graph of the formation sequence above. Vertical arrows represent lane change actions whereas horizontal arrows represent passing another agent on the respective side. It can be seen that the overtaking actions are temporally independent and thus can happen in either order or even concurrently. However, they have to happen before any agent passes an oncoming agent in order to represent the same pseudo-homotopy class.

**Figure 5.11.** Possible formation sequence of a scene with four vehicles, two on each of the two oncoming lanes, and the resulting temporal action dependency graph.

not of interest, as both option can be continuously transformed into the other without a collision and thus belong to the same pseudo-homotopy class.

In order to unite formation sequences that represent the same pseudo-homotopy class, let us look at the actual spatio-temporal constraints they imply. For that, each action in the formation sequence (the transition from one formation to another) is first annotated with the involved entities: Lateral actions only include the agent performing the action, whereas longitudinal actions additionally include the entity that is being passed during this action (either another agent or an intersection cell). Then, each action is temporally dependent on other actions which contain at least one identical entity. Actions that do not have any agent or intersection cell in common do not define a temporal relation, even though they occur at different steps in the formation sequence. This information allows to build a temporal action dependency graph given a formation sequence, as exemplarily depicted in Fig. 5.11 for the above mentioned scene with the two overtaking actions. The temporal constraints for the trajectory optimization are defined by this action dependency graph, whereas the spatial constraints are directly given by the pairwise relations between two vehicles passing each other.

In case multiple formation sequences result in the same action dependency graph, they represent the same pseudo-homotopy class and are thus united to only represent one maneuver. Each maneuver then defines a set of spatio-temporal constraints which are utilized for the trajectory planning as explained in the next section.

### 5.2.2 Trajectory Planning

Given any of the derived high-level maneuvers, we desire to know how typical low-level behavior on trajectory level looks like and how the ego-vehicle can be controlled correspondingly. Thus, we generate a multi-agent sample trajectory for every maneuver, by formulating an optimization problem that aims to minimize a collective cost function while satisfying the hard constraints given by the specific maneuver class. Similar to [106], mixed-integer quadratic programming (MIQP) with logical constraints is used to enforce specific maneuvers. Assuming this trajectory reflects common driver behavior for that maneuver, it is possible to infer the probability distribution over the possible maneuvers by comparing the sample trajectories to the observed motion, which will be explained in the subsequent section.

#### 5.2.2.1 Kinematics Model

The kinematics model of each vehicle is represented by a double integrator assuming the longitudinal and lateral acceleration  $\mathbf{a}^i = [a_s^i, a_d^i]^\top$  of vehicle  $V^i$  can be controlled directly:

$$\mathbf{x}_{t+1} = \begin{pmatrix} s_{t+1} \\ v_{s,t+1} \\ d_{t+1} \\ v_{d,t+1} \end{pmatrix} = \begin{pmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} s_t \\ v_{s,t} \\ d_t \\ v_{d,t} \end{pmatrix} + \begin{pmatrix} 0.5\Delta t^2 & 0 \\ \Delta t & 0 \\ 0 & 0.5\Delta t^2 \\ 0 & \Delta t \end{pmatrix} \begin{pmatrix} a_{s,t} \\ a_{d,t} \end{pmatrix} \quad (5.11)$$

For such a separated motion model, the side slip angle  $\beta = \arctan(v_d/v_s)$  can be limited to allow the trajectories to be followed by non-holonomic vehicles [96].

The multi-agent trajectory is defined with the initial state of all agents  $X_0$  and the sequence of their control inputs  $A(t) = [\mathbf{a}^0(t), \dots, \mathbf{a}^K(t)]^\top$  with  $t \in \{0, \dots, T-1\}$ . The motion of the single vehicles are interdependent, as their control inputs are coupled through the multi-agent optimization which accounts for the mutual influence between vehicles. Although this motion model is simplistic, we argue that it is sufficient for prediction purposes and that the trajectory to actually control the ego-vehicle can be smoothed in a post-processing step.

#### 5.2.2.2 Hard Constraints

We define a couple of hard constraints for the trajectory optimization problem that allow to limit the solution space of the continuous trajectories to reasonable values and to enforce a specific homotopy class of a maneuver. At first, to ensure reasonable vehicle dynamics independently of the maneuver, the state and control variables of all vehicles are bounded. Furthermore, vehicles are constrained to stay within the road boundaries. These maneuver-independent constraints are denoted as  $\mathcal{K}$ . To enforce that the planned trajectory lies within the range of a given maneuver  $M_i$ , i.e., restricting the planning space to a specific pseudo-homotopy class, maneuver-dependent constraints  $\mathcal{K}_{M_i}$  are formulated. As explained in Sec. 5.2.1, maneuvers distinguish between spatial



conditions (i.e., on which side two vehicles pass each other) and temporal conditions (i.e., in which order vehicles pass a critical area), which are explained in the following.

**Spatial Collision Avoidance:** Consider any pair of two vehicles  $V^A$  and  $V^B$  within a multi-agent environment, approximated by rectangles with length  $l$  and width  $w$  oriented along the centerline of a two-lane road. The spatial collision avoidance constraints of  $V^A$  and  $V^B$ , i.e., interdicting any overlapping of their shapes, but not stating whether they pass each other or if they do on which side, are given by:

$$\begin{aligned} & (s^A \leq s^B - l) \vee (s^A \geq s^B + l) \\ & \vee (d^A \leq d^B - w) \vee (d^A \geq d^B + w) \end{aligned} \quad (5.12)$$

Additional safety distances can easily be incorporated by adding or subtracting the desired margins respectively.

Constraints of this nature are called logical constraints, as they combine linear constraints using logical operators. By using the so-called *Big-M-method* [123] [97, p. 473], we reformulate those logical constraints into a set of linear inequality constraints. Therefore, we introduce an application-specific big number  $M^{\text{big}}$ , allowing for single conditions of these logical constraints to be rendered inactive by the addition or subtraction of  $M^{\text{big}}$  depending on the value of a binary variable  $\delta_i$ . This binary variable states whether the condition is active ( $\delta_i = 1$ ) or not ( $\delta_i = 0$ ). For each of the four single linear constraints above, we introduce one binary variable. Then, the logical disjunction of the single binary variables is equivalent to at least one of the corresponding conditions to be active:

$$s^A \leq s^B - l + (1 - \delta_1)M^{\text{big}} \quad (5.13)$$

$$s^A \geq s^B + l - (1 - \delta_2)M^{\text{big}} \quad (5.14)$$

$$d^A \leq d^B - w + (1 - \delta_3)M^{\text{big}} \quad (5.15)$$

$$d^A \geq d^B + w - (1 - \delta_4)M^{\text{big}} \quad (5.16)$$

$$\delta_1 + \delta_2 + \delta_3 + \delta_4 \geq 1 \quad (5.17)$$

This effectively transforms the logical constraint into a set of linear constraints and thus the optimization problem into a mixed-integer problem. We need to ensure that  $M^{\text{big}}$  is big enough such that for  $\delta = 0$  the respective linear constraint will always be fulfilled given the search space of the optimization problem. The *pairwise lateral relations* (i.e., whether two agents pass on the left or on the right) specified by a maneuver can now be enforced using these spatial conditions by setting the corresponding binary variable (either  $\delta_3$  or  $\delta_4$ ) to 0.

**Temporal Collision Avoidance:** Now let's consider the overtaking scenario from before again (see Fig. 5.9) with the three vehicles  $V^A$ ,  $V^B$ ,  $V^C$ . In addition to the conditions for the *pairwise lateral relations*, a desired temporal *sequence of passing* (e.g.,  $V^A$  before  $V^C$ ) can be enforced by the logical condition

$$(s^C \leq s^B) \Rightarrow (s^A > s^B). \quad (5.18)$$

This can be reformulated in Big-M notation as

$$s^C > s^B - \delta_k M^{\text{big}} \quad (5.19)$$

$$s^A > s^B - (1 - \delta_k) M^{\text{big}}, \quad (5.20)$$

only using a single binary variable to describe this implicative condition (or logical consequence). Thus, only one additional binary optimization variable is needed to enforce the desired sequence for the scenario with the three interacting vehicles.

**Separation of Longitudinal and Lateral Optimization:** To reduce the number of needed binary variables and hence reduce complexity, the problem is divided into two subproblems: Instead of optimizing the longitudinal and lateral control inputs jointly, we first optimize the longitudinal ones given the desired homotopy class and then optimize the lateral ones given the homotopy class and the results of the longitudinal components of the trajectory. In the field of single-agent planning, separating the longitudinal and lateral components of a trajectory has already been suggested by Gutjahr et al. [52]. In our case, the longitudinal optimization ensures the correct *sequences of passing*, whereas the lateral optimization ensures the correct *pairwise lateral relations*. By first solving the longitudinal optimization, its results can be used as an input for the lateral part. Hence, for the lateral optimization, the longitudinal positions of all vehicles are known in advance. This allows for the spatial collision avoidance constraints to be expressed without the need of binary variables, as it is known apriori in which time step which condition must hold. For the aforementioned overtaking scene, the number of binary variables is thus reduced from ten per time step (three per *pairwise lateral relation* and one for the *sequence of passing*) to one per time step (specifying the sequence of passing). Furthermore, as the values of the binary variables are known for  $t = 0$ , motion is only allowed within driving direction, and re-overtaking is forbidden, the binary variables describing the sequences of passing can change their value at most once during one maneuver. Hence the number of possible combinations per binary variable reduces from  $2^T$  to  $(T + 1)$ .

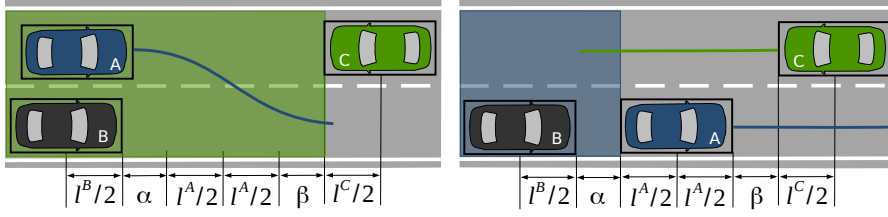
The longitudinal constraints to enforce a desired *sequence of passing* are adapted by the typical length needed for a lane change  $\alpha$  and a sufficient safety distance  $\beta$ . These distances are illustrated exemplarily in Fig. 5.12 for the maneuver  $M_2$  where  $V^A$  passes before  $V^C$  (recall the possible maneuvers for the *overtaking with oncoming traffic* scenario from Fig. 5.10(b)). The longitudinal constraints Eq. (5.19) and Eq. (5.20) of  $M_2$  change accordingly:

$$s^C > s^B + l^B/2 + \alpha + l^A + \beta + l^C/2 - \delta_1 M^{\text{big}} \quad (5.21)$$

$$s^A > s^B + l^B/2 + \alpha + l^A/2 - (1 - \delta_1) M^{\text{big}} \quad (5.22)$$

The constraints of  $M_3$  ( $V^A$  passes after  $V^C$ ) are defined analogously, the ones for  $M_1$  ( $V^A$  follows  $V^B$  and does not pass at all) only contain a safety distance between  $V^A$  and  $V^B$ , as no critical area exists. For intersection scenarios, the longitudinal constraints are defined relative to the overlapping areas.

The state and the control inputs as well as the optimization constraints, the cost function and its weights can be divided into a longitudinal and a lateral part, which are from



**Figure 5.12.** Longitudinal constraints for maneuver  $M_2$  to enforce a desired *sequence of passing*. On the left, Eq. (5.21) is active ( $\delta_1 = 0$ ) and  $V^C$  is not allowed to enter the green area. As soon as  $V^A$  has overtaken  $V^B$ , this restriction is removed and Eq. (5.22) is activated ( $\delta_1 = 1$ ). Previously published in [165]. © IEEE 2017

now on denoted by subscript  $(\cdot)_s$  and  $(\cdot)_d$ , respectively. The above defined maneuver-dependent constraints of a given maneuver  $M_i$  are denoted as  $\mathcal{K}_{M_i,s}$  and  $\mathcal{K}_{M_i,d}$  for the longitudinal and lateral aspects, respectively. The maneuver-independent constraints (e.g., vehicle dynamics, road boundaries) are denoted as  $\mathcal{K}_s$  and  $\mathcal{K}_d$ .

### 5.2.2.3 Cost Function

For the trajectories to reflect common human behavior, a quadratic cost function is chosen that penalizes acceleration, lateral velocity, as well as the deviation from the desired longitudinal velocity  $v_{\text{des}}$  and the deviation from the centerline of the current lane at  $d_{\text{center}}$ . Thus, we define a reference state  $\mathbf{x}_{\text{ref}}^i = [s_{\text{ref}}^i, v_{\text{des}}^i, d_{\text{center}}^i, 0]^\top$  with the deviation of state and reference state denoted as  $\Delta \mathbf{x}^i = \mathbf{x}^i - \mathbf{x}_{\text{ref}}^i$ . As we do not want to define a specific desired longitudinal position for each time step but rather specify a reference velocity, we set the weight of  $s_{\text{ref}}^i$  to zero, thus making this variable irrelevant.

The collective cost function of the multi-agent optimization problem is defined quadratic w.r.t. the control inputs as well as the deviation of state and reference state of all agents, divided in the longitudinal and lateral part

$$J = J_s + J_d = \sum_{i=0}^K J_s^i + \sum_{i=0}^K J_d^i, \quad (5.23)$$

with the components of a single vehicle  $V^i$

$$J_s^i = \sum_{t=1}^T (\Delta \mathbf{x}_{s,t}^i)^\top \mathbf{Q}_s^i \Delta \mathbf{x}_{s,t}^i + \sum_{t=0}^{T-1} a_{s,t}^i R_s a_{s,t}^i, \quad (5.24)$$

$$J_d^i = \sum_{t=1}^T (\Delta \mathbf{x}_{d,t}^i)^\top \mathbf{Q}_d^i \Delta \mathbf{x}_{d,t}^i + \sum_{t=0}^{T-1} a_{d,t}^i R_d a_{d,t}^i. \quad (5.25)$$

By taking the costs of all vehicles into account, drivers are assumed to behave cooperatively. As we want to penalize deviation from the reference state stronger in case an agent has right of way compared to agents that have to yield, we define a right of way dependent weight  $\omega = 1 + \psi\chi$ , with  $\chi = 1$  in case an agent has the right of way and

$\chi = 0$  otherwise.  $\psi \in \mathbb{R}^+$  is a factor stating how strong drivers weigh these traffic rules. For example, in our evaluation, we set  $\psi = 1$  to penalize it double if an agent with precedence has to deviate from its state. This encourages traffic rule compliance in a soft manner, still allowing drivers to slightly deviate if necessary. The state weighting matrices  $\mathbf{Q}_s^i = \text{diag}(0, \omega)$  and  $\mathbf{Q}_d^i = \text{diag}(\omega, \omega/2)$  and the control input weights  $R_s^i = \omega$  and  $R_d^i = \omega/2$  are then chosen according to domain knowledge.

The longitudinal optimization for maneuver  $M_i$  can be formulated as

$$[A_{s,M_i}^*, \boldsymbol{\delta}^*]^\top = \arg \min_{[A_s, \boldsymbol{\delta}]^\top} J_s \quad \text{subject to} \quad \mathcal{K}_s \cup \mathcal{K}_{M_i,s}, \quad (5.26)$$

resulting in the optimal longitudinal costs  $J_{M_i,s}^*$ . After determining  $A_{s,M_i}^*$ , the quadratic program for the lateral optimization can be defined only using linear inequality constraints as

$$A_{d,M_i}^* = \arg \min_{A_d} J_d \quad \text{subject to} \quad \mathcal{K}_d \cup \mathcal{K}_{M_i,d}, \quad (5.27)$$

resulting in the optimal lateral costs  $J_{M_i,d}^*$ . Note that only the hard constraints depend on the specific maneuver  $M_i$ , whereas the cost function is maneuver-independent. The overall optimal costs of  $M_i$  are determined by the sum of the optimal costs of both the longitudinal and lateral component:

$$J_{M_i}^* = J_{M_i,s}^* + J_{M_i,d}^* \quad (5.28)$$

For a visualization of the trajectory planning, see Fig. 6.23 in Sec. 6.6.2.

### 5.2.3 Maneuver Estimation

Given the set of possible maneuvers (or pseudo-homotopy classes) and the corresponding reference trajectories derived using the previously defined optimization routine, we now desire to estimate the maneuver intention probabilities, i.e., how likely it is that each of these maneuvers is going to happen. The multi-agent system is assumed to follow exactly one maneuver at a given time step. Even though different agents might intend to follow conflicting maneuvers at some points, typically they will converge to the same maneuver to avoid collisions. In case of two vehicles intending to follow two different maneuvers, our estimation will show both maneuvers as possible, but with decreased probability. We model a maneuver to be able to switch between time steps, which satisfies the fact that drivers continuously re-evaluate situations and may change their intentions. Thus, the motion of vehicles is modeled as a stochastic process that consists of  $|\mathcal{M}|$  different models, exactly one being active per time step. Following [80], the maneuver switching probability is set to  $\mu = 0.1$  and distributes uniformly among all other maneuvers.

The prediction task is to estimate the active model from which the drivers choose their actions. This is achieved with an interacting multiple model (IMM) Kalman filter. The prediction steps of maneuver  $M_i$  are performed according to the controls  $A_{M_i}^*$  of the optimal multi-agent trajectory. Through Bayesian statistics, the posterior mode

probabilities  $\mathbf{p}_{\mathcal{M}} = [p_{M_1}, \dots, p_{M_{|\mathcal{M}|}}]^\top$  are obtained in every time step given the predicted states, the measurement of the positions  $Z = [s_z^0, d_z^0, \dots, s_z^K, d_z^K]^\top$ , the last posterior and the switching probabilities. Initially, a uniform prior is assumed. For information about how the single distributions are determined, the reader is referred to [14, pp. 24-29].

Although drivers may deviate from the optimal trajectory of their intended maneuver used for prediction (as the cost function does not model human behavior perfectly), it is still possible to estimate the maneuver correctly, as long as the deviation is smaller compared to the optimal trajectories of the other maneuvers.

#### 5.2.4 Decision Making

To close the loop of the prediction and planning framework (see Fig. 5.6), a simple decision making layer is applied, that can either decide for the most likely maneuver according to  $\mathbf{p}_{\mathcal{M}}$ , or for the cost-optimal maneuver. For instance, if the situation is not yet critical as the next upcoming critical area is still far away, the ego-vehicle can still aim for the cost-optimal maneuver and try to convince other drivers. However, if the vehicles are closer to a critical area and the estimated distribution has a strong indication for the intended maneuver of others, the ego-vehicle can decide to accommodate. Given the chosen maneuver, the ego-vehicle is then controlled according to the accelerations  $(\mathbf{a}^0)^*$  of the optimal trajectory of that maneuver.

Note that this simplistic decision making strategy should not be considered safe, as one cannot assume that others will behave cooperatively all the time. We argue that one would have to add additional safety mechanisms that take potential uncooperative behaviors of others into account.

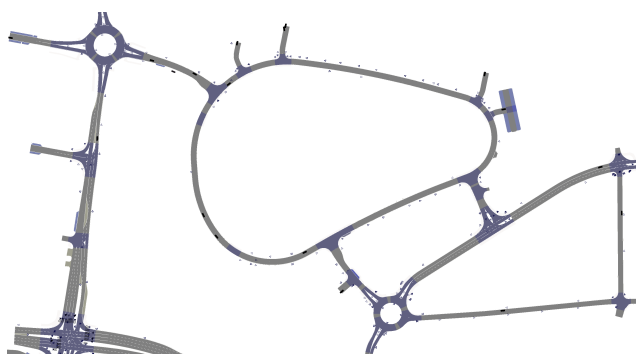


## 6 Experiments

This chapter presents different experiments demonstrating the general purpose of the presented DBN for intention estimation and trajectory prediction and highlights the benefits of accounting for context-dependency and interaction-awareness. The upsides and downsides of the alternative behavior models (rule-based and deep learning-based) and inference methods (SMC and MM-UKF) are investigated. Furthermore, we compare the interacting single-agent DBN variant to the full-combinatorial variant and highlight its runtime improvements. Lastly, it is evaluated how the presented prediction approach can be utilized for interactive ego-vehicle motion planning for autonomous vehicles using the presented multi-agent planning scheme and the POMDP formalism with the sampling-based tree search solution method. The reported runtimes in this chapter are based on an Intel Core i7-5820K CPU @ 3.30GHz with non-optimized C++ code, if not specified otherwise. This chapter is based on the author’s previous publications [162], [165]–[168].

### 6.1 Environment Model

The input to the presented approaches is given by an environment model which consists of a topological map and all agents in a scene. The map, as depicted in Fig. 6.1, contains the lane topology (how lanes are connected), geometric information (such as road curvatures and lane widths), road infrastructure (traffic lights, traffic signs, lane markings, etc.) as well as the prevailing traffic rules such as the right of way. It is used to determine the possible routes, conflict areas and features for the action models and is based on the *OpenDRIVE* [149] format. The agents are represented with their dimensions, poses, and velocities and are either simulated using a proprietary simulator,



**Figure 6.1.** Environment model showing a digital map and simulated agents (black rectangles).

## 6 Experiments



**Figure 6.2.** Autonomous test vehicle used to record data of surrounding traffic participants and the ego-vehicle. The vehicle is equipped with GPS/INS-based localization, as well as lidar- and radar-based object detection. © BMW Group

or are recorded with a measurement vehicle in real traffic. The proprietary simulator is using the driver model presented in [56] to control its agents.

Desirably, this environment model should be ideal, meaning no occlusions were limiting our field of view, all relevant agents were detected without uncertainty, and the map was containing all information necessary for predicting the scene. However, due to theoretical and practical limitations, this is generally not the case in the real world and thus there always will be inaccuracies in the environment model. This section explains how the data used in this evaluation is recorded and gives an overview of the limitations this data comes with.

Real driving data is recorded with an autonomous test vehicle (see Fig. 6.2) with GPS/INS-based localization and dynamic occupancy grid-based object tracking [128]–[130] using multiple lidar and radar sensors. These sensors allow for a localization of the measurement vehicle relative to a given map and for the detection and tracking of static and dynamic obstacles in its surroundings. This measurement vehicle is a BMW 7 series equipped with both a variety of different sensors as well as electronically controllable actuators. The data used for evaluation and the training of the behavior models was however recorded with a human test driver (with disabled autonomous driving mode) to ensure realistic human-human interactions.

There are multiple aspects that contribute to the recorded data not being ideal in the sense described above. First, the surrounding objects in the scene are only detected in the vicinity of the ego-vehicle, due to lidar and radar range limitations of about 200 m. In addition, occlusions may limit the field of view, resulting in missing detections of traffic participants that are either behind static obstacles such as buildings (e.g., at intersections) or behind other dynamic objects such as cars or trucks. Another important aspect is the inherent noise in sensor measurements. The sensors used for localization of the ego-vehicle as well as the ambient environment sensors used to detect other traffic participants are afflicted with noise. This results in an imperfect state estimation of the available agents. However, as we are utilizing data from both lidar and radar, it can be argued that the distance and velocity estimation of objects is rather accurate (e.g., compared to when using cameras), whereas there is a higher difficulty in clas-



sifying the type of an object (e.g., distinguishing a car from a truck). Future work could benefit from additionally fusing camera data, allowing to utilize more fine-grain object features. In addition to the noise of the raw sensor readings, the software used to detect and track objects comes with limitations. Generally, there is a compromise between detecting agents early and keeping a low false positive rate, i.e., not falsely detecting an object when there is none. Additionally, it is difficult to estimate the dimensions of objects, as only a part of their surface is reached by the lidar and radar beams (typical L-shape detections). For more details about the utilized detection and tracking module, we refer to [128]–[130]. To reflect these inaccuracies at least to some degree also in simulation, instead of directly using the ground truth simulation state of the agents, we disturb the observations with zero mean Gaussian noise with variances  $[\sigma_{x_{\text{sim}}}^2, \sigma_{y_{\text{sim}}}^2, \sigma_{\theta_{\text{sim}}}^2, \sigma_{v_{\text{sim}}}^2] = [0.01 \text{ m}, 0.01 \text{ m}, 0.0001, 0.01 \text{ m/s}]$ . Another aspect is that the map cannot be seen as a perfect ground truth, as it was generated with a mapping vehicle that was also using sensors to map its environment. These sensors come with the same limitations as mentioned above, thus the map is not perfectly accurate either. In addition to low level position inaccuracies, there might be lanes that are not properly extracted, such as driveways or entrances to parking lots.

The presented approach in this thesis does not explicitly model map inaccuracies, the highly complex measurement uncertainty in the agents' states, nor the possibility of undetected agents due to occlusions. However, the received measurements are assumed to be afflicted with strong zero mean Gaussian noise (see Tab. 6.1), allowing the DBN to cope with bad observations of different sorts and additionally to reduce the problem of sample impoverishment [51]. Although this is a strong simplification, results show that this assumption works well in practice. Explicitly handling missing features such as non-extracted lanes or undetected agents is left for future work.

## 6.2 Rule-based Behavior Models

In order to assess the necessity of interaction-aware prediction, the presented rule-based behavior model is compared to simpler, non-interactive and physics-based models in simulated and real driving scenarios that contain multiple interacting agents. For this analysis, the SMC-based inference method is used in order to prevent the approximation error of the MM-UKF Gaussian assumption to interfere. To reduce sample impoverishment due to resampling, a small number of particles is directly sampled from the current measurement distribution with probability 0.001. The computation time of one time step of a scene with three vehicles, each having three route options, is approximately 0.3s on the above mentioned hardware. The evaluation parameters defined in Chapter 2 and Chapter 3 can be seen in Tab. 6.1. The IDM model parameters as well as the noise terms were tuned by hand using different urban scenarios that are not part of the evaluation.

## 6 Experiments

**Table 6.1**  
EVALUATION PARAMETERS OF DYNAMIC BAYESIAN NETWORK. ADAPTED FROM [166].

General		IDM		Noise	
step size $\Delta T$	0.2 s	desired distance $d_d$	2 m	$\sigma_{x/y}^2$	0.5 m
particles $N$	1000	desired time headway $T_d$	0.1 s	$\sigma_{\theta}^2$	0.05
route horizon $l_H$	30 m	comfortable acceleration $a_d$	0.7 m/s <sup>2</sup>	$\sigma_v^2$	1.5 m/s
		comfortable braking $b_d$	-0.5 m/s <sup>2</sup>	$\sigma_a^2$	1.5 m/s <sup>2</sup>
		acceleration exponent $\delta$	4	$\sigma_{\theta}^2$	0.05/s
		max. lateral acceleration $a_{\text{lat}}^{\text{max}}$	2 m/s <sup>2</sup>	$\sigma_{x_z/y_z}^2$	15 m
				$\sigma_{\theta_z}^2$	3.14
				$\sigma_{v_z}^2$	15 m/s

### 6.2.1 Intention Estimation

The imprecision of the route (and analogously maneuver) estimate  $p(r^i) = [p(r_1^i), \dots, p(r_{|\mathcal{R}^i}^i)]$  of agent  $V^i$  is measured using the Kullback-Leibler divergence (KLD)

$$D_{\text{KL}}(p(r_{\text{GT}}^i) \| p(r^i)) = \sum_{j=1}^{|\mathcal{R}^i|} p(r_{\text{GT},j}^i) \log \frac{p(r_{\text{GT},j}^i)}{p(r_j^i)} \quad (6.1)$$

from estimate to the ground truth distribution  $p(r_{\text{GT}}^i) = [p(r_{\text{GT},1}^i), \dots, p(r_{\text{GT},|\mathcal{R}^i}^i)]$ , with

$$p(r_{\text{GT},j}^i) = \begin{cases} 1 & \text{if } V^i \text{ follows } r_j^i \\ 0 & \text{else} \end{cases} \quad (6.2)$$

The intention estimation of the presented rule-based model from Chapter 3, here called *interactive* model, and a solely *map-based* model are evaluated. The *map-based* model uses all of the features given by the map but ignores surrounding vehicles and, therefore, is interaction-unaware. Thus, agents are predicted as if there were no other vehicles around. In order to highlight the differences between both models, we specifically identify scenarios with a high potential for interaction between agents. We evaluate situations in which multiple vehicles cross an intersection, and hence, contain strong interdependencies between vehicles. Though intersection crossings might be statistically rare when looking at the overall number of time steps on intersections versus on other parts of the road, these situations tend to be most critical and therefore require explicit evaluation. The intention estimation is evaluated in detail for multiple scenes:

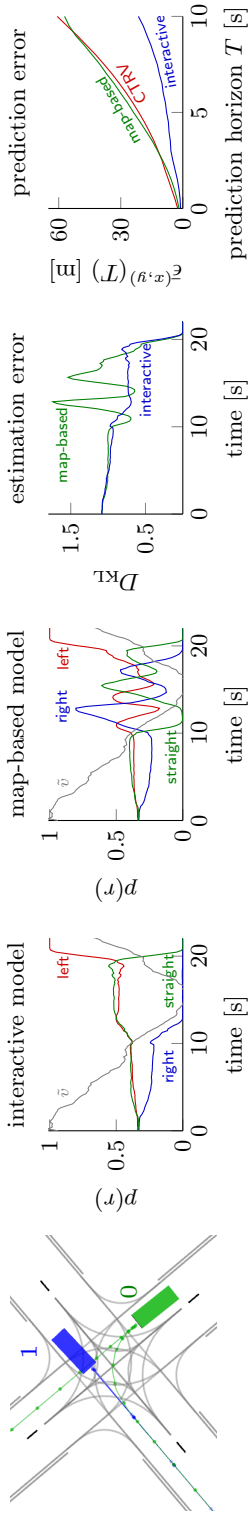
*Scene 1 - Yielding vehicle* (depicted in Fig. 6.3(a)): In this simulated scene,  $V^1$  has right of way and goes straight, whereas  $V^0$  has to yield and wants to turn left. To improve readability, at first it is assumed that  $V^0$  is actually yielding and therefore only has one possible maneuver ( $V^1 \prec V^0$ ), but multiple possible routes whose probabilities we want to estimate. While  $V^0$ 's routes for going straight and turning left demand yielding, the route for turning right is free and does not imply any waiting for other traffic. As  $V^0$  waits for  $V^1$  ( $t=10-18$ s), it is inferred by the interactive model that turning right is

unlikely (as waiting would not be necessary) and turning left and going straight is equally likely (as both routes are blocked and thus imply a similar deceleration). As soon as  $V^1$  has left the conflict area,  $V^0$  accelerates again and starts turning, whereby the left route is inferred correctly. The map-based model, however, infers incorrectly that  $V^0$  wants to turn right ( $t=13$  s), as this route has the highest curvature and thus implies the lowest velocity, matching the actual behavior best. For  $t>13$  s, as  $V^0$  even becomes too slow for turning right, none of the map-based models can explain the actual behavior anymore. Thus, only the particles sampled newly from the measurement survive, resulting in a random oscillation and a momentary improvement of the KLD.

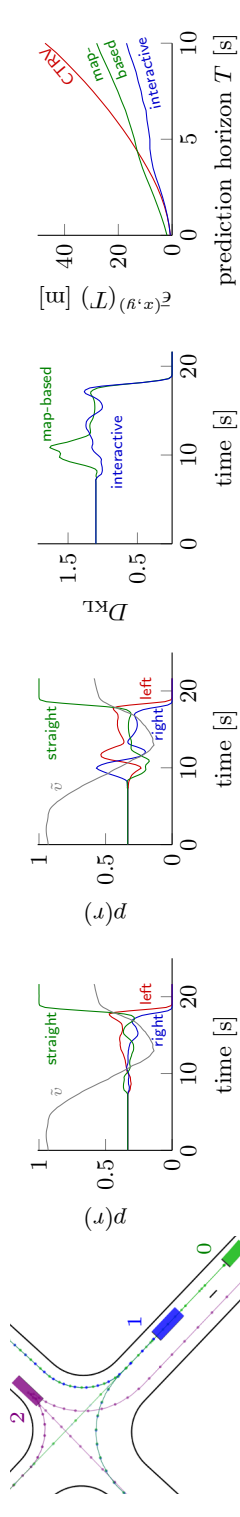
*Scene 2 - Preceding vehicle* (depicted in Fig. 6.3(b)): In this real driving scene,  $V^0$  follows  $V^1$  which is approaching an intersection. As  $V^1$  has to yield and therefore starts to decelerate,  $V^0$  decelerates as well in order to keep the desired headway distance to  $V^1$ . All three possible routes of  $V^0$  are blocked by the preceding vehicle, hence, it is not possible to infer the route until the preceding agent has passed the intersection ( $t=17$  s). A uniform distribution is the desired result, as all three routes imply the same behavior, which is correctly inferred by the interactive method. The map-based method incorrectly determines that  $V^0$  wants to turn right ( $t=10$  s), as it is slowing down (actually caused by the preceding vehicle and not by the high curvature of the route that turns right). For  $t>10$  s, none of the map-based models can explain the observations anymore, also resulting in a random oscillation.

*Scene 3 - Preceding vehicle* (depicted in Fig. 6.3(c)): In this simulated scene,  $V^0$  again follows  $V^1$  approaching an intersection. As  $V^1$  this time wants to turn right and thus slows down, it again forces  $V^0$  to also decelerate. A uniform distribution is thus the desired estimate again, as the behavior for all three possible routes is dominated by the preceding agent. Only the interactive model is able to infer the influence of the preceding agent, whereas the map-based model assumes  $V^0$  wants to turn as well.

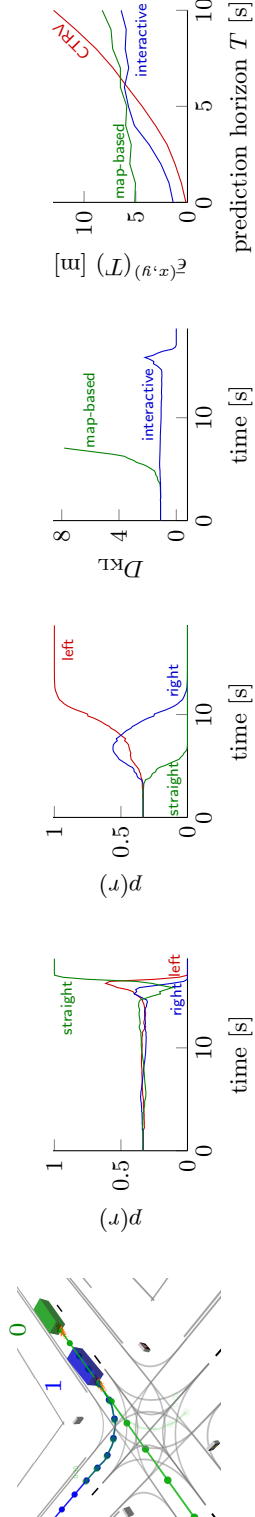
*Scene 4 - Maneuver distinction* (depicted in Fig. 6.4): The combined maneuver and route estimation is analyzed in a scene that is similar to scene 1, but which is modified such that  $V^0$  is crossing before the other agent that has right of way. The interactive model with maneuver distinction is compared to the interactive model without maneuver distinction (which assumes  $V^0$  will yield): At  $t=0$  s, all routes are equally likely, but as  $V^0$  does not decelerate strongly ( $t=2-10$  s), the probability to yield decreases, whereas the probability to either turn right (no conflict) or merge or cross before  $V^1$  increases. As  $V^0$  slows down to respect the upcoming curvature ( $t=9-11$  s), the straight route becomes unlikely. Finally ( $t=12-20$  s), as the velocity is still too high for turning right (because the high curvature would require driving slower), it is correctly inferred that  $V^0$  will turn left and merge before  $V^1$ . Without the distinction of the two possible maneuvers, assuming  $V^0$  is going to yield, it is incorrectly inferred that  $V^0$  wants to turn right (as this lane has no conflict), resulting in a higher estimation and trajectory prediction error (see the following section).



(a) Scene 1 (simulated data):  $V^0$  approaches the intersection and yields to the blue agent  $V^1$  to its right which has right of way.

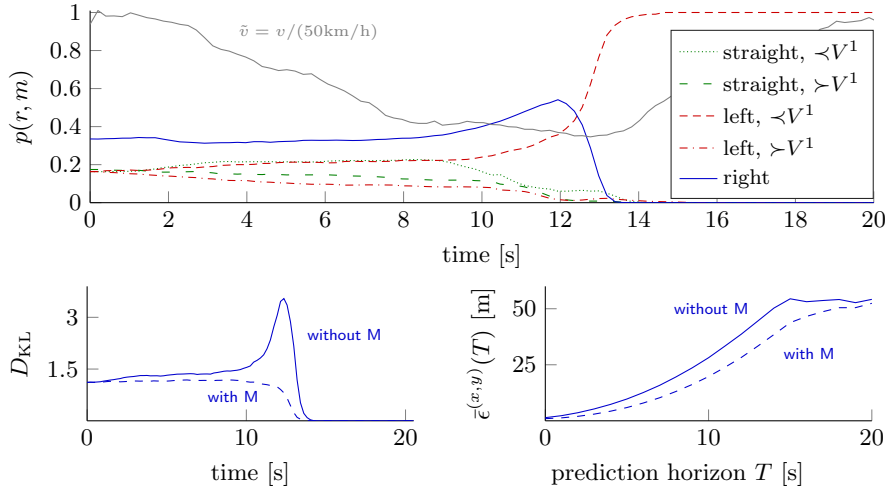


(b) Scene 2 (real data):  $V^0$  slows down in order to keep a safe distance to its preceding agent  $V^1$  which decelerates to yield in order to  $V^2$ .



(c) Scene 3 (simulated data):  $V^0$  slows down in order to keep a safe distance to its preceding agent  $V^1$  which decelerates before turning right.

**Figure 6.3.** Evaluation of multiple scenes: Comparison of route estimation and trajectory prediction for agent  $V^0$  for different tracking methods (CTRV, map-based, interactive), also showing the velocity profile  $\tilde{v} = v/(50\text{km/h})$ . Previously published in [166]. © IEEE 2018



**Figure 6.4.** Route and maneuver estimation of agent  $V^0$  in scene 4, comparing interaction-aware model with and without maneuver distinction. Previously published in [166]. © IEEE 2018

### 6.2.2 Trajectory Prediction

For each possible combination of routes and maneuvers of all agents, one forward simulation resulting in a multi-agent trajectory is executed. Each forward simulation is initialized to the mean kinematic state of the given mode. The accuracy of the trajectory prediction of all agents at time  $t$  for the future time step  $(t + T)$  is quantified using the position components (denoted by superscript  $(x,y)$ ) of the root weighted square error between predicted mean position  $\hat{X}_{t+T|t,R_t,M_t}^{(x,y)}$  and measurement  $Z_{t+T}^{(x,y)}$

$$\epsilon_{t+T|t}^{(x,y)} = \sqrt{\sum_{R_t, M_t} p(R_t, M_t) \left\| \hat{X}_{t+T|t,R_t,M_t}^{(x,y)} - Z_{t+T}^{(x,y)} \right\|^2}, \quad (6.3)$$

and the corresponding measurement likelihood

$$\mathcal{L}_{t+T|t}^{(x,y)} = \sum_{R_t, M_t} p(R_t, M_t) p(Z_{t+T}^{(x,y)} | \hat{X}_{t+T|t,R_t,M_t}^{(x,y)}). \quad (6.4)$$

For evaluating the prediction of a single agent on its own, one can replace the capital letter variables with the ones representing the respective agent, i.e.,  $r$ ,  $m$ ,  $x$  and  $z$ . The errors and likelihoods depending on the prediction horizon  $T$  are calculated and averaged over the complete scene (from  $t = 0$  until  $t = t_{\max}$ )

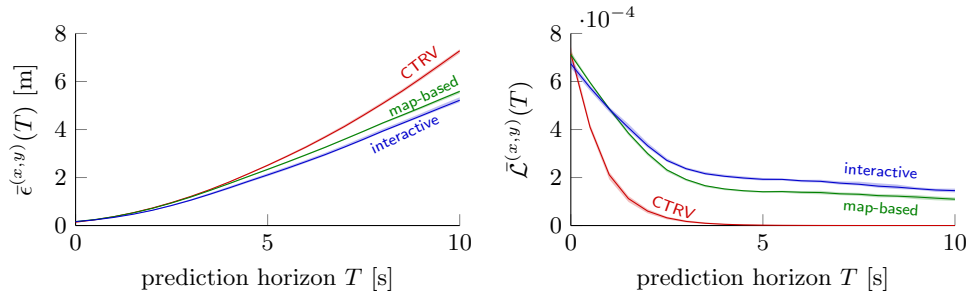
$$\bar{\epsilon}^{(x,y)}(T) = \frac{1}{1 + t_{\max}} \sum_{t=0}^{t_{\max}} \epsilon_{t+T|t}^{(x,y)} \quad \text{and} \quad \bar{\mathcal{L}}^{(x,y)}(T) = \frac{1}{1 + t_{\max}} \sum_{t=0}^{t_{\max}} \mathcal{L}_{t+T|t}^{(x,y)}. \quad (6.5)$$

A visualization of the trajectory prediction can be seen in Fig. 6.5, showing the view of the front facing camera of the measurement vehicle together with the detected objects

## 6 Experiments



**Figure 6.5.** Camera view of measurement vehicle with detected objects and predicted trajectories while yielding to oncoming traffic in order to turn left into a parking lot. Previously published in [166]. © IEEE 2018

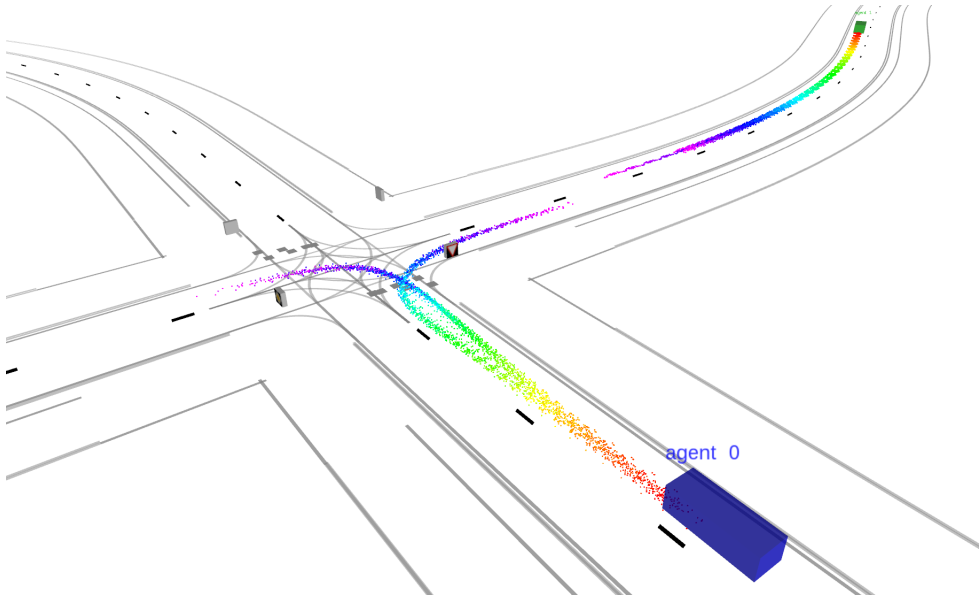


**Figure 6.6.** Prediction error and likelihood in five different driving scenarios. Previously published in [166]. © IEEE 2018

and corresponding predicted trajectories. As the ego-vehicle is driven by a human, its trajectory is also predicted by including it within the DBN’s state space.

The *interactive* model is compared to the *map-based* model and a *constant turn rate and velocity (CTRV)* model [119], which serves as a simple baseline algorithm. It is independent of both the map and surrounding vehicles and is thus context-unaware. The error of the trajectory prediction of  $V^0$  for scenes 1 to 3 are depicted in the right most column of Fig. 6.3. The CTRV model performs worse in scene 1, as  $V^0$  changes its velocity and orientation more intensely. For the map-based model, the first scene is also more challenging, as  $V^0$  stops for a long time, which cannot be explained by the model at all. Its high route estimation error negatively affects its prediction accuracy. The interactive model outperforms the other two approaches in all three scenes.

In order to compare the three different models in a more quantitative manner, five different real driving scenes have been recorded on a test track and on real roads. These scenes altogether consist of 15 vehicles, two four-way intersections, two T-junctions, and a roundabout. The statistical results showing the mean prediction error and measurement likelihood over all scenes and vehicles are depicted in Fig. 6.6. It can be seen



**Figure 6.7.** Probabilistic forward simulation using the deep learning-based behavior model (linear feed-forward). It can be seen how the model picks up subtleties such as sheering out to the opposite direction before a turn.

that the interaction-aware model outperforms both CTRV and map-based models. Although the differences between the interactive and the map-based model might seem rather small, it has to be noted that the time steps in which traffic participants actually interact with each other (i.e., the behavior of an agent is significantly influenced by the existence of the other agent) do not predominate. As shown in Fig. 6.3, however, in scenes where the behaviors of drivers are highly interdependent, interaction-aware prediction becomes essential. A video of the approach using the interactive rule-based model in exemplary scenes can be found at <https://mediatum.ub.tum.de/1449806>.

### 6.3 Deep Learning-based Behavior Models

This section evaluates the capabilities of the deep learning-based behavior models as presented in Sec. 3.2 and compares different model architectures with each other and to the rule-based model from Sec. 3.1. Fig. 6.7 exemplarily depicts the probabilistic forward simulation using a linear feed-forward network and shows how the model is able to learn that agents typically sheer out to the opposite direction before an upcoming curve. The evaluation comprises the trajectory prediction errors, the importance of the single features, and a qualitative result on the route intention estimation, which is obtained by embedding the learned model into the DBN for inference. First, the different model architectures and the used training data are specified.

**Table 6.2**  
 HYPERPARAMETERS OF NEURAL NETWORK ACTION MODELS. PREVIOUSLY PUBLISHED IN [168].  
 ©IEEE 2019

Parameter	Linear	LSTM	GRU	Lenz
activation function	ReLU	–	–	ELU
number of layers	4	2	2	5
neurons per layer	274	274	274	400
dropout probability	0.06	0.2	0.3	0.5
batchnorm momentum	0.3	–	–	–
batchnorm eps	1e-5	–	–	–

### 6.3.1 Models and Hyperparamters

The following types of neural networks are compared to show how they are suited for the task of one-step motion prediction: Linear fully connected (Linear), long short-term memory (LSTM), gated recurrent unit (GRU), and the architecture presented by Lenz et al. [82] which is also a fully connected linear model, and which was originally introduced as a driver behavior model for highway scenarios. Each layer of the linear models consists of a fully connected layer followed by batch-norm (not present in the Lenz model), activation function, and dropout. The last layer of each of the network types is a fully connected linear layer that outputs the action distribution parameters  $\mu_a, \mu_\delta, \sigma_a^2, \sigma_\delta^2$ . The outputs for the variances are transformed by an exponential function before calculation of the loss to ensure positive values.

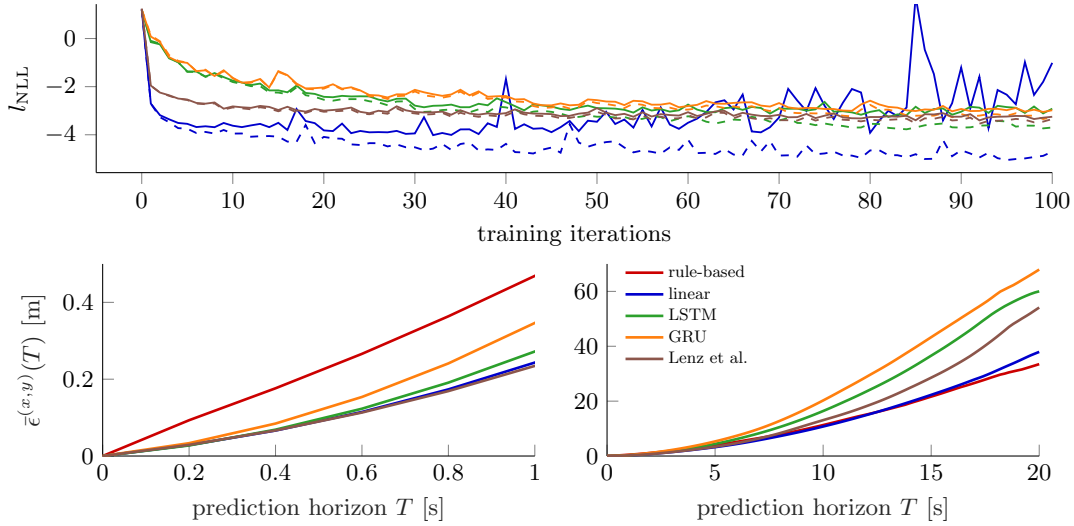
The best found hyperparameters for the presented models as well as the parameters used by Lenz et al. are presented in Tab. 6.2. We use the deep learning framework PyTorch [99] for model definition, for training, and for inference. Furthermore, the Adam optimizer [65] is used with a learning rate of 0.001, running average coefficients  $\beta_1 = 0.9$  and  $\beta_2 = 0.999$ , term added for numerical stability  $\text{eps} = 1\text{e-}8$ , no weight decay, no gradient clipping, and without AMSGrad. The batch size is set to 1024 and the sequence length for the recurrent architectures is chosen to 3s with a sampling rate of 0.2s.

### 6.3.2 Training and Validation Data

For training and validating our neural network-based behavior models on real data, we recorded 40 minutes of urban scenarios including both roundabouts and unsignalized intersections with our autonomous test vehicle. This dataset is denoted as  $\mathcal{D}^{\text{real}}$  and is split into 30 minutes for training and 10 minutes for validation.

As the real driving data is very limited and not as diverse (e.g., does not contain traffic lights), additional recorded simulations from our proprietary traffic simulator are utilized as well. This dataset, denoted as  $\mathcal{D}^{\text{sim}}$ , consists of 0.465 hours of naturalistic driving data including 40 agents randomly traversing a different, and more diverse urban environment. It includes traversing signalized as well as unsignalized intersections (both





**Figure 6.8.** Comparison of different network architectures using  $\mathcal{D}^{\text{sim}}$ : Losses over training iterations (validation solid, training dashed) depicted in the upper image and corresponding position prediction errors on validation data in the lower images with different zoom factors. Previously published in [168]. © IEEE 2019

T-junction and 4-way junctions) and some lane changes on multi-lane intersections. This results in 18.6 hours of data, split into 10 hours for training and 8.6 hours for validation.

As the training and validation split is defined by a given time step in the whole dataset, no partial data from any validation trajectory is represented within the training dataset. The data is resampled to a step size of  $\Delta T = 0.2$  s, which is used for target reconstruction (using the inverse bicycle model), for the training procedure, and for inference and the iterative forward simulation. Both features and targets are normalized to zero mean and unit variance according to the training dataset. The training data for the non-recurrent networks is shuffled across all available data samples and for the recurrent networks, a sequence length of 3 s is used.

### 6.3.3 Comparison of Model Architectures

The different model architectures are compared using the simulation dataset  $\mathcal{D}^{\text{sim}}$ , as it contains more diverse scenes (e.g., including traffic lights or lane changes, not present in the real driving dataset), thus allowing for a better comparison. In order to prevent overfitting, so-called early-stopping is applied, such that the model that has performed best on the validation dataset over the complete training procedure is saved and used for evaluation.

The validation and training losses of the different models are depicted in the upper part of Fig. 6.8. It can be seen that the linear model trains the fastest and achieves the lowest overall loss ( $l_{\text{NLL}} = -4.02$ , see also Tab. 6.3). The lower part of Fig. 6.8 shows the corresponding position prediction errors  $\bar{\epsilon}^{(x,y)}$  on the validation dataset depending on the prediction horizon  $T$ . For that purpose, the models have been applied iteratively

## 6 Experiments

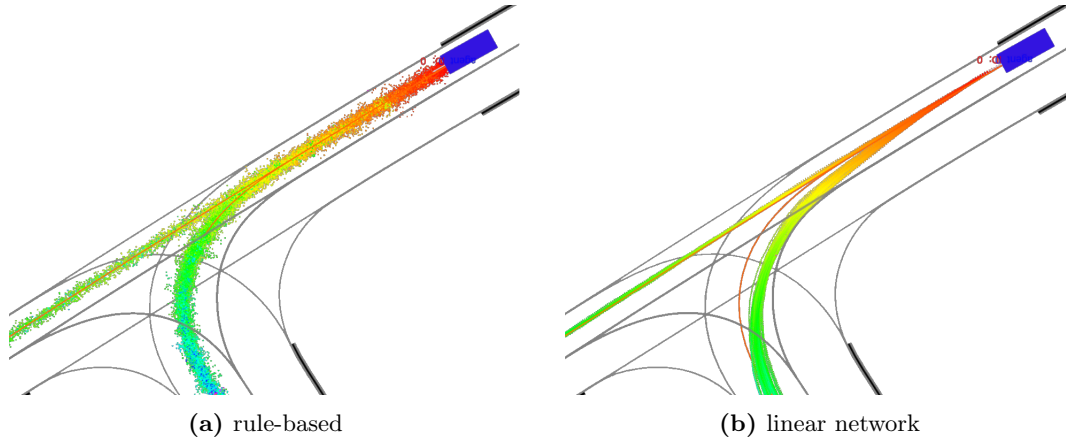
**Table 6.3**  
 RUNTIMES  $\tau$  [s], LOSSES  $l_{\text{NLL}}$  AND POSITION PREDICTION ERRORS  $\bar{\epsilon}^{(x,y)}$  [m] FOR DIFFERENT NETWORK ARCHITECTURES USING VALIDATION SET OF DATASET  $\mathcal{D}^{\text{sim}}$ . PREVIOUSLY PUBLISHED IN [168]. © IEEE 2019

Method	$\tau$	$l_{\text{NLL}}$	$\bar{\epsilon}(0.2\text{ s})$	$\bar{\epsilon}(1\text{ s})$	$\bar{\epsilon}(5\text{ s})$	$\bar{\epsilon}(10\text{ s})$	$\bar{\epsilon}(20\text{ s})$
Rule-Based	2.2e-6	–	0.093	0.469	3.959	11.28	<b>33.48</b>
Linear	7.1e-7	<b>-4.02</b>	0.028	0.243	<b>3.237</b>	<b>10.75</b>	37.95
LSTM	6.8e-7	-3.26	<b>0.027</b>	0.272	4.414	16.40	60.02
GRU	6.4e-7	-3.05	0.033	0.346	5.326	20.25	67.96
Lenz et al.	<b>6.3e-7</b>	-3.40	0.029	<b>0.235</b>	3.393	13.07	54.11

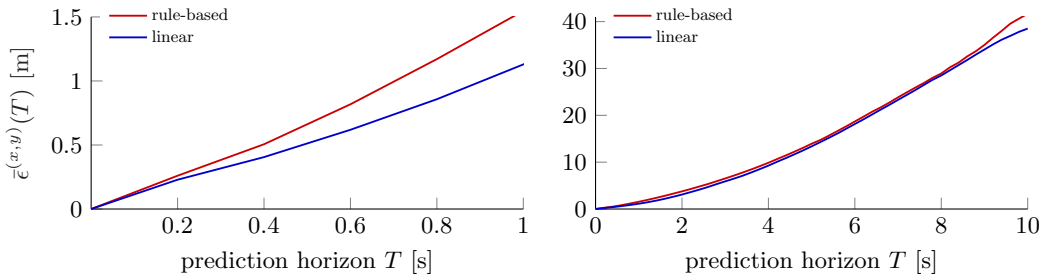
with step size  $\Delta T = 0.2\text{ s}$  to predict up to the full prediction horizon. For short term predictions of up to around 2 s, all of the learned models outperform the rule-based model. Low one-step prediction errors are especially important for inference in a DBN. For long-term predictions (for more than around 5 s), the linear model achieves the best results among all neural network models. However, for very long horizons exceeding 15 s, the rule-based model becomes more accurate than any of the learned models. This can be explained by the prediction error that compounds over time, for which the learned models cannot counteract. If this error is too high, the features determined during forward simulation are not represented within the training data anymore. This problem is commonly known in machine learning as compounding error problem. In supervised learning, it is typically assumed that the training and testing data are independent and identically distributed (i.i.d.). For an iterative forward simulation, this is not true, as the future inputs to the model are influenced by the learned policy itself. The same problem exists in imitation learning (IL), as the learned policy also influences the future states upon which the policy has to act [110]. Ross and Bagnell [110] also show that this leads to compounding errors and a regret bound that grows quadratically with prediction time. This error could be reduced either with a training distribution of more variety, or with techniques such as *data as demonstrator* [148]. Another possibility is to switch from supervised model learning to IL approaches such as inverse reinforcement learning (IRL) or generative adversarial imitation learning (GAIL). The rule-based system, on the contrary, is designed by hand to be attracted by the lane center, such that a vehicle will always stay close to the centerline.

The runtime  $\tau$  of any of the learned models (averaged over a batch of 1000 samples) is less than a third of the one of the rule-based model (see Tab. 6.3). This is foremost the case due to efficient batch processing, allowing for fast execution when multiple samples can be processed independently (such as in particle filtering within our DBN).

A qualitative result on the real data  $\mathcal{D}^{\text{real}}$  can be seen in Fig. 6.9, comparing the forward simulation of the rule-based model to the learned linear model. It can be seen that the learned model is able to pick up subtleties such as cutting curves and curvature dependent lateral uncertainty. The linear neural network model also outperforms the



**Figure 6.9.** Qualitative comparison of rule-based and learned linear model on real driving data. The learned model is able to reproduce subtleties such as cutting curves and different lateral uncertainties for curved and straight roads. Previously published in [168]. © IEEE 2019



**Figure 6.10.** Position prediction errors on validation data of the real driving dataset with different zoom factors. Previously published in [168]. © IEEE 2019

rule-based model, as shown in Fig. 6.10. However, it has to be noted that this dataset is quite small and does not contain as diverse situations as the simulated one.

### 6.3.4 Feature Importance

To determine the importance of the single features, the linear network is trained on  $\mathcal{D}^{\text{sim}}$  starting with using only a single feature and iteratively adding features which result in the highest reduction of validation loss. For that, we train the neural network multiple times from scratch, always adding one feature to the current list of most relevant features and analyze the resulting validation losses. Thus, redundant features can be pruned and the complexity of the model reduced. The five most important features and the corresponding validation losses by adding each of these features are:

	$[v^i]$	$[v^i, \phi_{15}]$	$[v^i, \phi_{15}, d_{\text{inters.}}]$	$[v^i, \phi_{15}, d_{\text{inters.}}, \phi_7]$	$[v^i, \phi_{15}, d_{\text{inters.}}, \phi_7, c_{70}]$	all
$l_{\text{NLL}}$	-0.714	-2.093	-2.779	-3.249	-3.346	-4.016

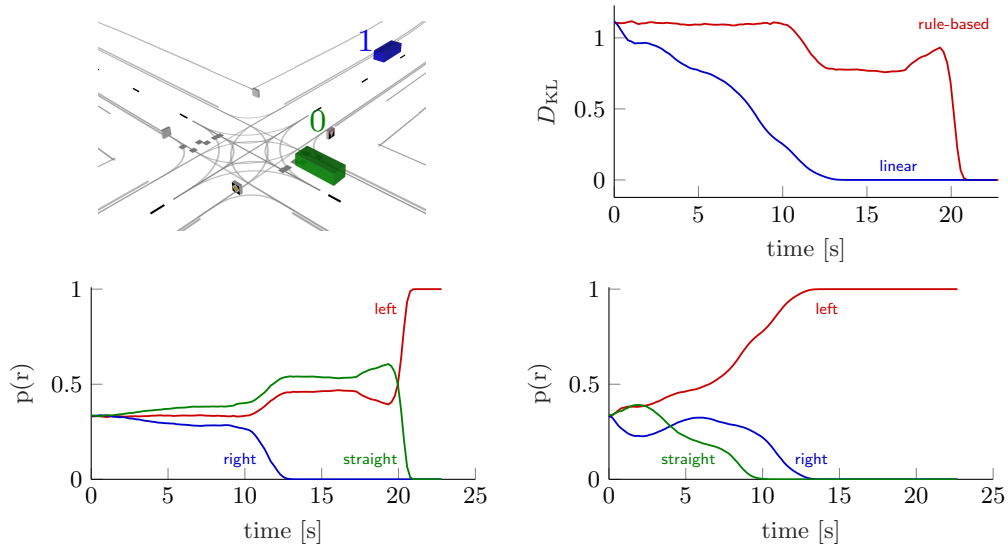
## 6 Experiments

It can be seen that the velocity of the predicted vehicle  $v^i$  together with important map information that describe the upcoming route represent the most important features. The angles  $\phi_{15}$  and  $\phi_7$  hold information about how to steer the vehicle in the near future. The distance to the next intersection  $d_{\text{inters.}}$  and the upcoming curvature  $c_{70}$  tell the agent when it needs to reduce its speed. Similar to the rule-based model, the interaction features (e.g., distances to other vehicles) have a statistically lower impact. Nevertheless, they are important in situations with strong interaction which are, however, statistically rare. See Sec. 3.2.3.2 for a list and description of the utilized features.

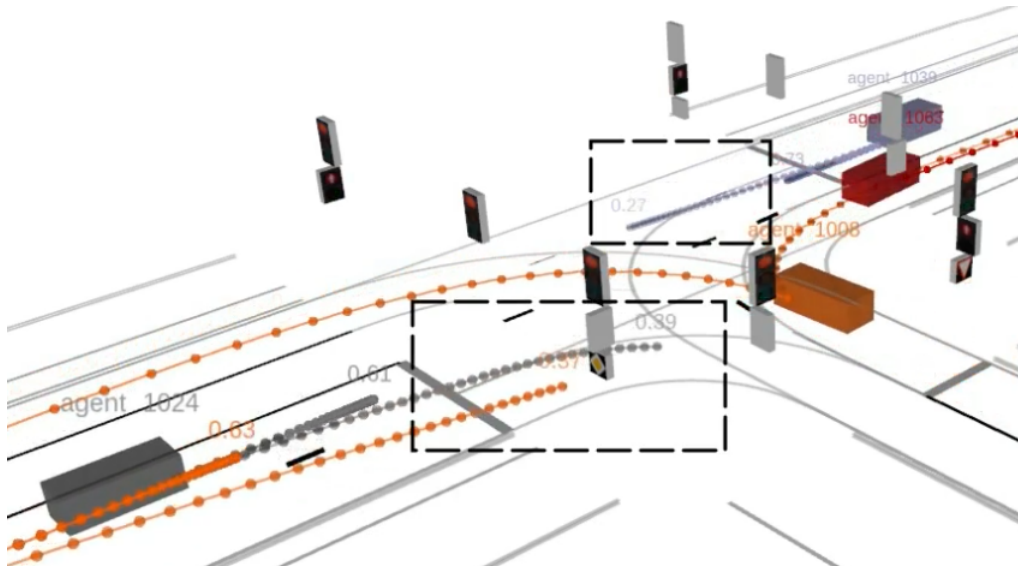
### 6.3.5 Route Intention Estimation

By conditioning the action on a driver’s route intention, one can query the neural network for the action distribution for each of the different route hypothesis and then compare them with the observations of the actual motion to estimate the route probabilities. We achieve this by integrating the learned model into the DBN from Chapter 2. A qualitative comparison of route estimation and the corresponding Kullback-Leibler divergence  $D_{\text{KL}}$  to the ground truth is shown in Fig. 6.11. The learned model achieves a lower divergence and is able to tell the correct route faster than the rule-based model.

An interesting disadvantage of conditioning the model on a driver’s route intention is depicted in Fig. 6.12: When enumerating all possible routes and running a forward simulation for each of the conditioned models, there might exist route candidates that are so unlikely that they have never (or only very rarely) been followed in the training data. Thus, their features may result in unreasonable actions during inference, as the network has never seen any similar values during training. To put it another way, the network only learns what actions are reasonable given a route, but not which routes are reasonable given a situation. Thus, if we query for unreasonable routes, the network is likely to produce unreasonable actions. In the depicted scenario, agents  $V^{1024}$  and  $V^{1039}$  are predicted to run a red light given the unlikely route hypotheses of doing a lane change this close to the intersection (which has not happened during training). As an improvement, one could additionally learn the route priors and prune very unlikely hypotheses before forward simulation.

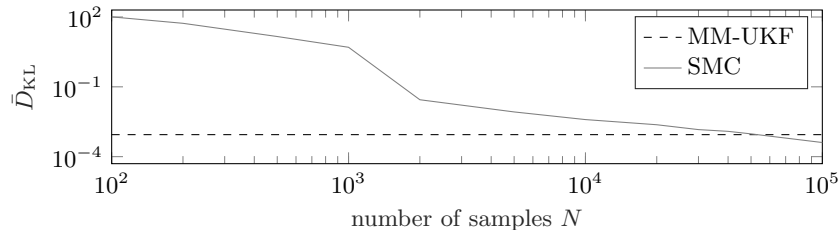


**Figure 6.11.** Qualitative comparison of route estimation of the green agent  $V^0$  wanting to turn left, with rule-based (left plot) and learned linear neural network (right plot) action models and corresponding KLD to ground truth. Previously published in [168]. © IEEE 2019



**Figure 6.12.** A disadvantage of conditioning a machine learning model on a driver's route intention: Routes that are so unlikely that they are not present in training data may result in unreasonable actions, such as red light violations. Previously published in [168]. © IEEE 2019

## 6 Experiments

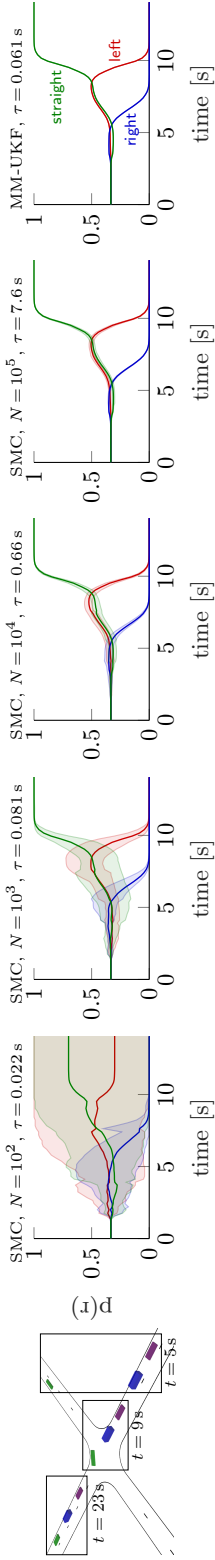


**Figure 6.13.** Comparison of KLD of MM-UKF and SMC with different number of samples over multiple scenes (15 agents in total). Performances meet at around 50000 samples, with corresponding runtimes of  $\tau_{\text{SMC}} = 7.9$  s and  $\tau_{\text{MM-UKF}} = 0.57$  s. Previously published in [167]. © IEEE 2018

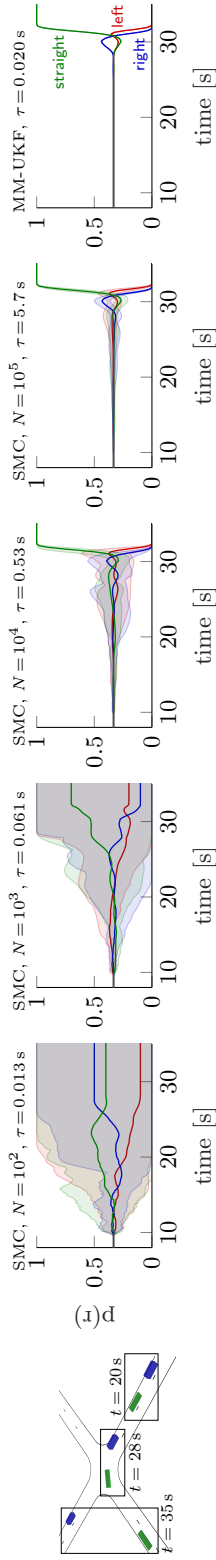
### 6.4 Comparison of Inference Methods

To highlight the differences of SMC-based and MM-UKF-based inference, their route and maneuver intention estimates are compared in simulation and real world scenarios. As SMC inference converges to the optimal estimate for  $N \rightarrow \infty$ , the result of each estimator is compared to the mean estimate over ten runs of SMC with a high number of samples ( $N = 10^5$ ), which is abbreviated as BE for *best estimator*. The imprecision of each estimator is thus measured using the Kullback-Leibler divergence between the estimate and the best estimator’s estimate, instead of comparing to the ground truth distribution. This allows to evaluate estimation capabilities in isolation, without an interference of the model imprecisions. Generally, it is often the case that even a perfect estimator is not able to infer the ground truth, either due to imperfect behavior models or because the different models might result in (close to) identical one-step prediction hypotheses. On the other hand, a bad estimator might accidentally result in a probability distribution that is closer to the ground truth. The Kullback-Leiber divergence is thus defined as in Eq. (6.1), but with the ground truth distribution being replaced by the best estimator’s distribution  $p(r_{\text{BE}}^i)$ .

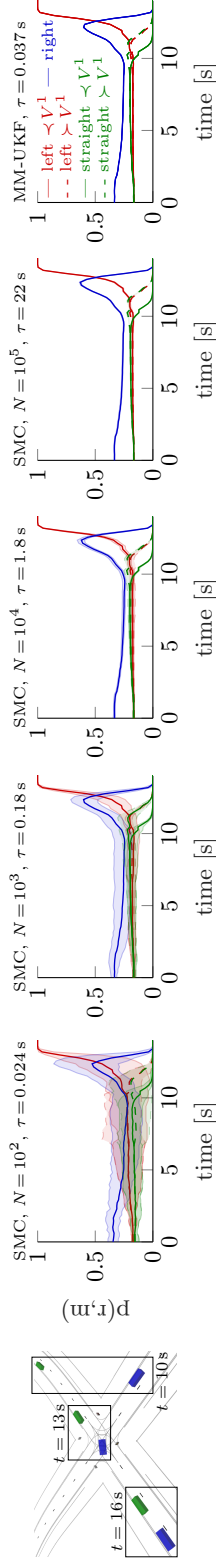
A statistical evaluation of the mean Kullback-Leibler divergence  $\bar{D}_{KL}$  of all agents’ route intentions over six different scenes (real and simulated) with 15 vehicles in total is depicted in Fig. 6.13. On average, the MM-UKF outperforms SMC with up to  $N = 50000$  particles, while having a maximum runtime per time step of  $\tau = 0.57$  s compared to  $\tau = 7.9$  s of SMC. As the estimation results are strongly dependent on the scenario, three different scenes are evaluated in more detail in Fig. 6.14. For all scenes, it can be seen that the SMC inference method can result in high variances in the estimates depending on the number of samples. The more samples are used, the less variance the estimation results have, as the randomness in the sampling process is averaged out. However, while the variance decreases, the inference runtime increases with the number of samples. On the other hand, as MM-UKF chooses its sigma points deterministically, multiple runs of the same scene always result in the same estimates. Thus, MM-UKF inference does not show any variance. It can be seen that it achieves similar results to SMC with about  $N = 10^5$  samples while having a significantly decreased runtime. The corresponding KLD plots for the three scenarios can be seen in Fig. 6.15.



(a) Scene 1 (real data): The blue agent has to yield to the green agent and then goes straight as soon as the intersection is cleared.



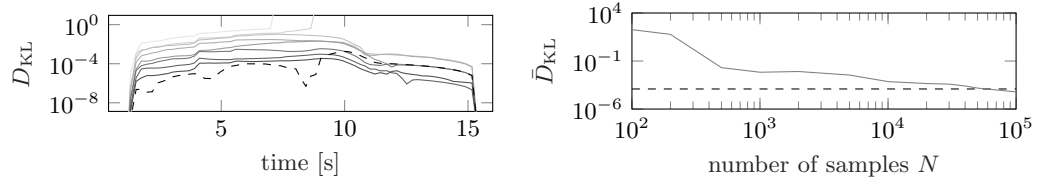
(b) Scene 2 (real data): The blue agent follows the green agent and has to slow down to keep a desired distance. As soon as the green agent has turned left, the blue agent accelerates and goes straight.



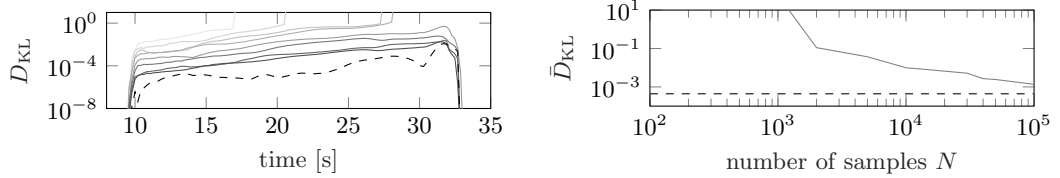
(c) Scene 3 (simulated data): The blue agent turns left at the intersection and merges just before the green agent. The estimates depict the combined route and maneuver probabilities.

**Figure 6.14.** Comparison of route/maneuver estimation accuracy and variance of MM-UKF and SMC with different number of samples  $N$  for the blue agent. Shown is the mean (thick line) and the min/max values (filled areas) over ten runs with different random seeds. As MM-UKF has deterministic output, there is no variance in the estimate. The MM-UKF estimates are almost identical to the ones of SMC with  $N = 10^5$  samples, but come with a greatly reduced runtime. Previously published in [167]. © IEEE 2018

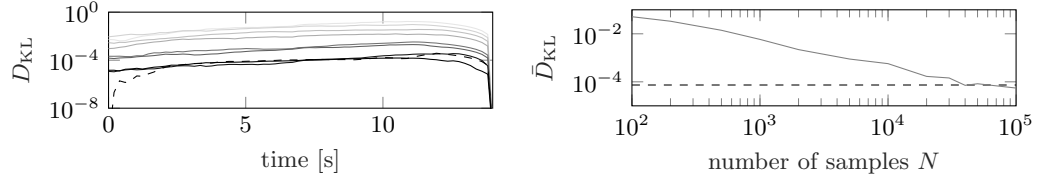
## 6 Experiments



(a) Scene 1 (real data): The MM-UKF has a maximum runtime  $\tau = 0.061$  s, which is comparable to SMC with  $N = 600$ , but performs as good as SMC with  $N = 50000$ .



(b) Scene 2 (real data): The MM-UKF has a maximum runtime  $\tau = 0.020$  s, which is comparable to SMC with  $N = 300$ , but performs even better than SMC with  $N = 100000$ .

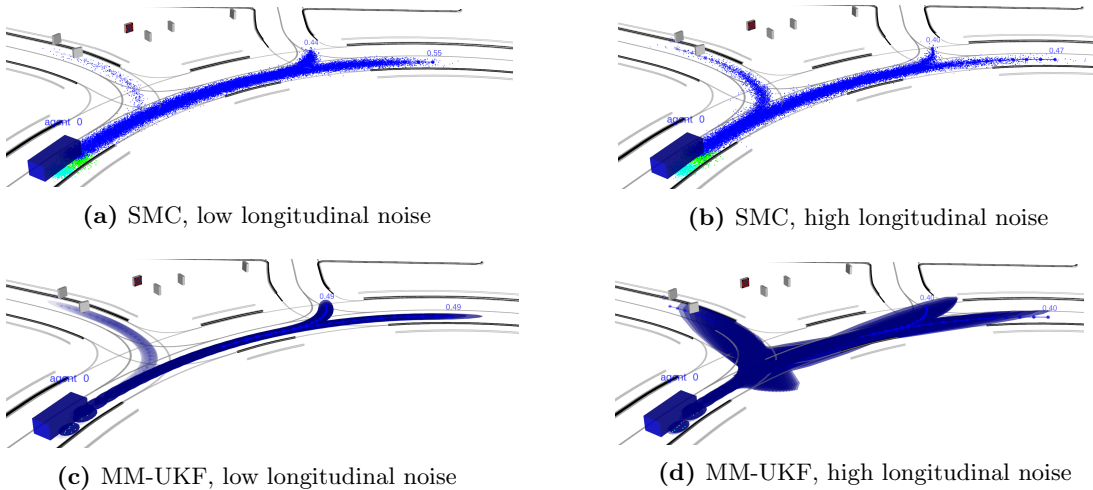


(c) Scene 3 (simulated data): The MM-UKF has a maximum runtime  $\tau = 0.037$  s, which is comparable to SMC with  $N = 200$ , but performs as good as SMC with  $N = 40000$ .

**Figure 6.15.** KLD over time and mean KLD over number of SMC samples for scenarios from Fig. 6.14. Dashed lines represent MM-UKF, solid lines SMC with intensity showing the number of samples (100, 200, 500, 1000, 5000, 10000, 50000, 100000; the darker, the more samples). Previously published in [167]. © IEEE 2018

The first scene consists of three vehicles with human drivers approaching an intersection. The blue vehicle stops to yield to the green vehicle. As the blue vehicle would not have to yield in case it wanted to turn right (as this route does not overlap with the green vehicle's possible routes), it can be inferred that it either wants to turn left or go straight ahead ( $t \approx 8$  s). As soon as the green vehicle has crossed the intersection, the blue vehicle starts to drive again and goes straight ( $t \approx 9$  s). The MM-UKF has a maximum runtime comparable to SMC with  $N = 600$  samples, but outperforms SMC with up to  $N = 50000$ . In the second scene which is also based on real data, the blue vehicle is driving behind the green vehicle, which approaches the intersection. As the green vehicle slows down before turning left, the blue vehicle also slows down to maintain the desired headway distance, irrespective of its desired route ( $t \approx 20$  s). As soon as the green vehicle has left the intersection, the blue vehicle accelerates again and continues straight ( $t \approx 30$  s). The runtime of MM-UKF is similar to SMC with  $N = 300$ , but it performs even better than SMC with  $N = 10^5$ . The third scene shows how maneuvers and routes are estimated simultaneously. Initially, all three routes of the simulated blue vehicle have equal probability. As it only slows down slightly because of the upcoming





**Figure 6.16.** Comparison of forward simulation based on SMC (first row) and MM-UKF (second row) depending on longitudinal noise. For high road curvatures and high longitudinal uncertainty, the Gaussian approximation of the UKF is not a good approximation anymore, resulting in sigma points being outside of lanes and thus an overestimated variance.

curvature, but does not stop in order to yield to the green vehicle, it is inferred that neither going straight, nor yielding is likely. As soon as it starts turning, it is correctly inferred that it is turning left and merges before the green vehicle.

Although SMC is naturally able to achieve higher accuracy as it does not make any (wrong) assumptions about the distribution, this usually comes at the cost of a higher runtime. In all of the evaluated examples, MM-UKF outperformed SMC in terms of accuracy per runtime. However, it has to be noted that the number of needed sigma points and thus the runtime of MM-UKF is dependent on the number of modes and vehicles in a scene. The results indicate that the presented MM-UKF-based inference provides a reasonable compromise between accuracy and runtime. Furthermore, they show that given the discrete intentions of all agents, the state can reasonably be approximated by a single multivariate Gaussian. MM-UKF achieves lower variance and higher accuracy compared to SMC inference with a similar runtime.

A comparison of the forward simulation-based trajectory prediction is depicted in Fig. 6.16. It can be seen that the approximation quality highly depends on the modeled transition noise. For reasonably low longitudinal noise or shorter prediction horizons, the Gaussian approximation is reasonable (cf. left part of figure). However, for longer horizons, high transition noise, and high road curvatures, the Gaussians may approximate the curved road badly, at some point even resulting in the sigma points being located outside of lanes (cf. right part of figure). This may lead to *exploding variances* as a result of the highly non-linear behavior models. However, it was found that with reasonable scaling of the transition noise, forward simulation of up to around 10 s still approximates the belief similar to the one of SMC. One way to counteract the problematic of bad approximation in high curvatures could be to utilize lane-centered coordinates such as the Frenet-Serret formulation (e.g., as in [152]), instead of the Cartesian representation.

## 6.5 Interacting Single-Agent DBNs

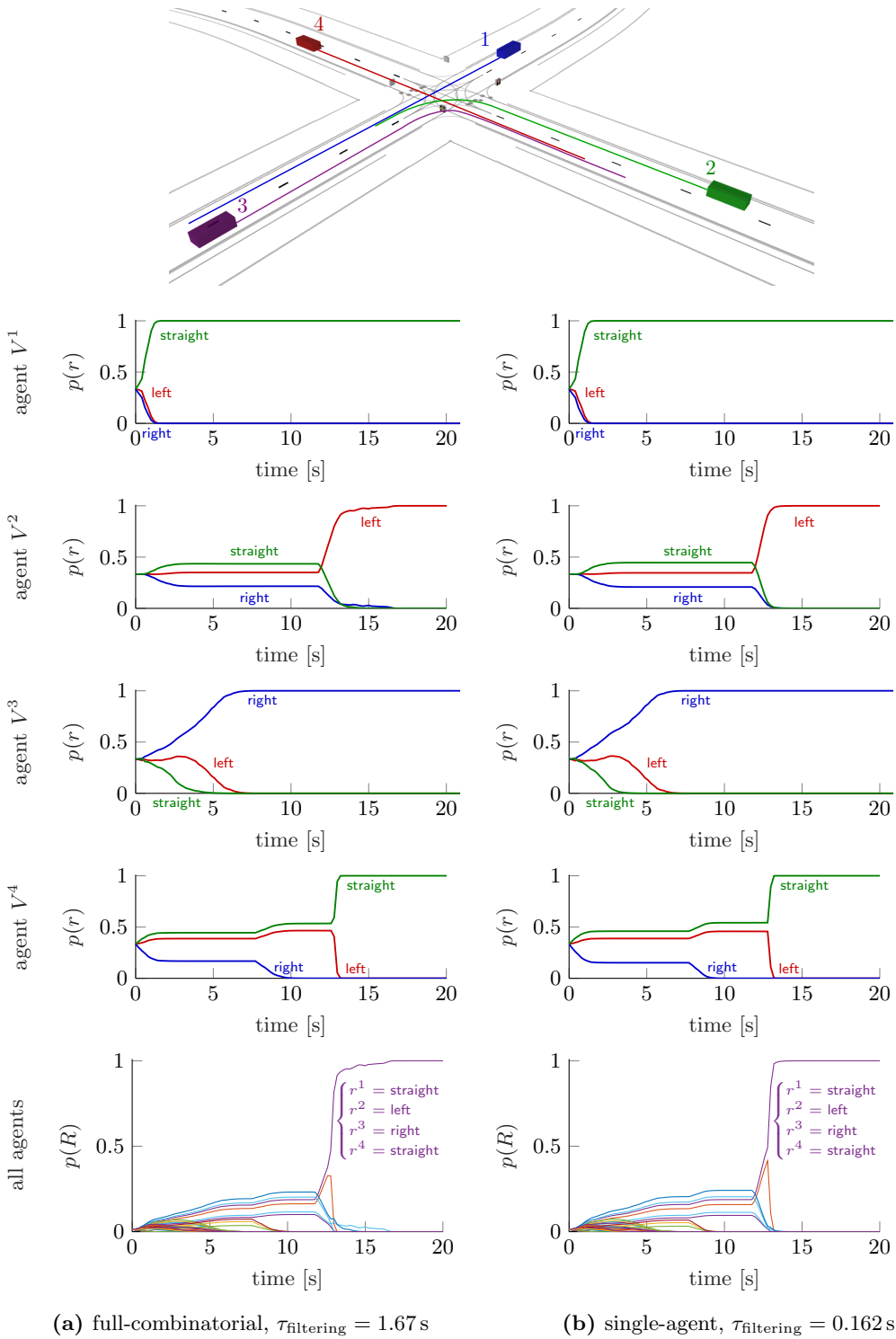
In this section, we evaluate the *interacting single-agent DBNs* as proposed in Sec. 4.4 and compare them to the full-combinatorial DBN. Dividing the problem into interacting single-agent models serves the purpose of reducing complexity and making it possible to handle more complex scenarios with a high number of possible combinations of intentions of all agents. This section utilizes the MM-UKF-based inference method. In contrast to the previously reported runtimes, the runtimes reported in this section are based on an Intel Core i7-8550U CPU @ 1.80GHz.

The route estimation capabilities of both DBN types are shown for a scene with four vehicles traversing an intersection in Fig. 6.17. To enable a comparison between both types, the *route combination probabilities*  $p(R)$  of the full-combinatorial DBN are marginalized to retrieve the route probabilities  $p(r)$  for all single agents, and the *single agents' route probabilities*  $p(r)$  of the interacting single-agent DBNs are combined to retrieve the route combination probabilities  $p(R)$ . It can be seen that the estimated probabilities (both  $p(r)$  and  $p(R)$ ) do only differ slightly while the runtime of filtering with the interacting single-agent DBNs is about ten times lower. As there are three routes each (neglecting different maneuvers for readability, assuming all agents obey the right of way), the single-agent DBNs only have to track  $|\mathcal{R}|K = 3 \cdot 4 = 12$  combinations, whereas the full-combinatorial one has to track  $|\mathcal{R}|^K = 3^4 = 81$  combinations. The more complex the scenario, the higher the runtime improvement will be, as the complexity only scales linearly with the number of agents, compared to the exponential growth in the full-combinatorial version.

Although the single-agent DBNs do not *track* all possible combinations of high-level intentions, for the trajectory prediction (i.e., the forward simulation of the DBN) various different multi-agent combinations are generated to account for possible future interaction. The probabilities of these intention combinations are calculated based on the estimated probabilities of the single agents' intentions, such that the most likely hypotheses can be utilized for prediction.

For assessing how good such a multi-modal prediction is, we introduce a metric which we call the *likelihood miss rate* given some likelihood threshold  $\mathcal{L}_{\text{threshold}}$ . As the name suggests, this metric reports the relative number of times where the prediction model has a likelihood given the observed future which falls below the defined threshold. This metric is specifically suited for analyzing multi-hypotheses predictions, as it shows whether the true future was in the vicinity of *any* of the given prediction hypotheses. Other typical metrics like the mean likelihood or the mean RMSE over a statistical evaluation of a large number of agents can yield unintuitive results for multi-modal predictions. The following exemplary calculation showcases this undesired effect:

Let's consider a single agent traversing the same intersection  $N$  times, always randomly picking one of the three available routes [left, right, straight], i.e.,  $p(r) = [\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$ . Given are three prediction models  $A, B, C$ , where  $A$  always predicts  $p(r|\text{model}=A) = [1, 0, 0]$ ,  $B$  always predicts  $p(r|\text{model}=B) = [\frac{1}{2}, \frac{1}{2}, 0]$ , and  $C$  always predicts  $p(r|\text{model}=C) = [\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$ . For each intersection crossing  $n$ , we denote the actual route being taken as  $r_n \in \{[1, 0, 0], [0, 1, 0], [0, 0, 1]\}$  in a one-hot encoding which is sampled according to the



**Figure 6.17.** Route probabilities of full-combinatorial and of interacting single-agent DBNs for the depicted scene (actual routes are visualized in the scene at the top with the colored lines). It can be seen that the probability estimates do not differ noticeably while the maximum runtime per time step of the filtering stage  $\tau_{\text{filtering}}$  is about ten times lower for the single-agent filter than for the full-combinatorial one.

## 6 Experiments

uniformly distributed probability  $p(r)$ . The routes that have not been taken are defined using a one-cold encoding as  $\neg r_n \in \{[0, 1, 1], [1, 0, 1], [1, 1, 0]\}$ . For simplicity, let us assume that in case the route prediction is correct, the likelihood for the continuous state prediction will constantly be  $\mathcal{L}_{\text{correct}}$  and in case the route prediction was wrong, the likelihood will constantly be  $\mathcal{L}_{\text{wrong}}$ . The mean likelihood given a model can then be calculated as

$$\bar{\mathcal{L}}_{\text{model}} = \frac{1}{N} \sum_{n=1}^N r_n^\top \cdot p(r|\text{model}) \mathcal{L}_{\text{correct}} + \neg r_n^\top \cdot p(r|\text{model}) \mathcal{L}_{\text{wrong}}. \quad (6.6)$$

Although model  $C$  is optimal in our example and model  $A$  and  $B$  are not representing the multi-modality appropriately, for a large number of  $N$  and given exemplary values of  $\mathcal{L}_{\text{correct}} = 1$  and  $\mathcal{L}_{\text{wrong}} = 0$ , the likelihoods of all prediction models average out at the same value of  $\frac{1}{3}$ :

$$\bar{\mathcal{L}}_A = \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{n=1}^N r_n^\top \cdot [1, 0, 0] = \frac{1}{3} \quad (6.7)$$

$$\bar{\mathcal{L}}_B = \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{n=1}^N r_n^\top \cdot \left[ \frac{1}{2}, \frac{1}{2}, 0 \right] = \frac{1}{3} \quad (6.8)$$

$$\bar{\mathcal{L}}_C = \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{n=1}^N r_n^\top \cdot \left[ \frac{1}{3}, \frac{1}{3}, \frac{1}{3} \right] = \frac{1}{3} \quad (6.9)$$

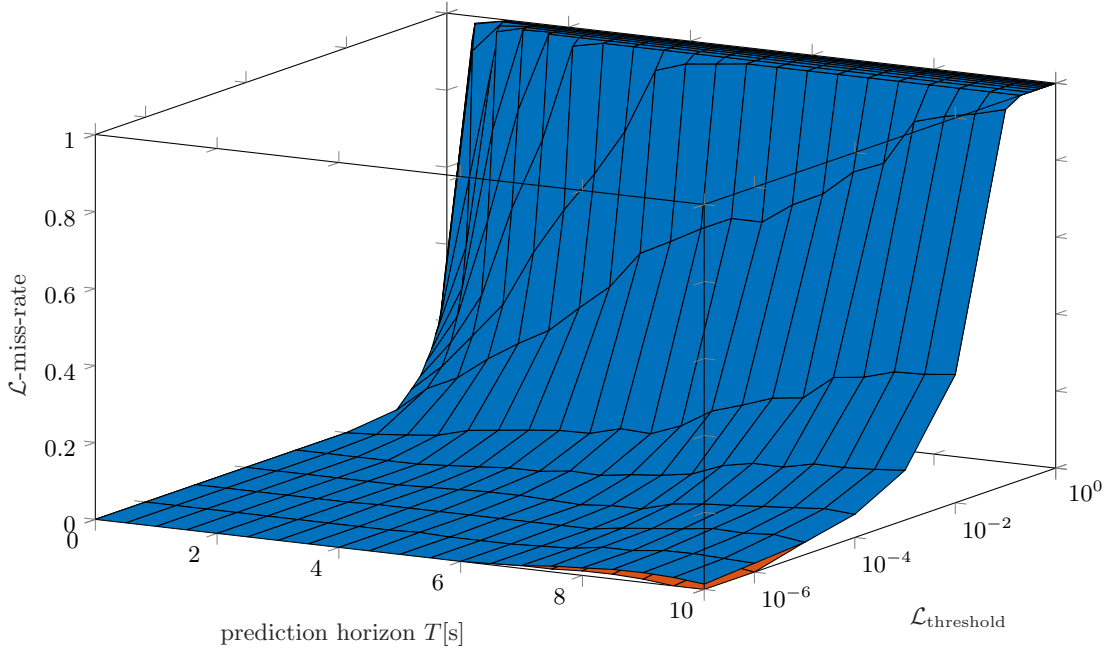
On the contrary, the likelihood miss rate

$$\mathcal{L}\text{-miss-rate}_{\text{model}} = \frac{\sum_{n=1}^N \mathbb{1}_{\text{miss}}}{N}, \quad \text{with } \mathbb{1}_{\text{miss}} := \begin{cases} 1, & \text{if } \mathcal{L}_{\text{model}} < \mathcal{L}_{\text{threshold}} \\ 0, & \text{if } \mathcal{L}_{\text{model}} \geq \mathcal{L}_{\text{threshold}} \end{cases} \quad (6.10)$$

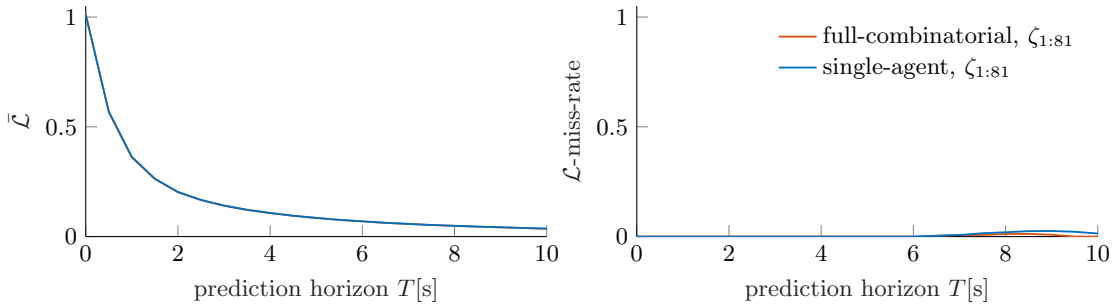
does not result in the same value for all prediction models but, depending on the threshold, gives more informative results. As it is hard to say what a good threshold is in general, one can plot the likelihood miss rate over a range of thresholds for a better analysis.

For the scene from Fig. 6.17, the mean likelihoods as well as the likelihood miss rates are depicted in Fig. 6.18 for the full-combinatorial and the single-agent DBNs, utilizing all available 81 hypotheses. As expected, both the likelihoods and the likelihood miss rates are similar for both approaches (with the full-combinatorial performing slightly better), as the route combination probability estimates are also similar and the forward simulation of both models are identical given the same route combination probabilities. Intuitively, if the likelihood threshold is set too high (e.g.,  $\mathcal{L}_{\text{threshold}} = 1$ ), none of the models achieve any hits anymore.

In case it is computationally prohibitive to actually simulate *all* combinations forward, it is possible to only predict a subset of these hypotheses. One way to choose this subset is to utilize the most likely hypotheses in order not to waste computational power on low probability outcomes. Other things like the criticality of a hypothesis or how



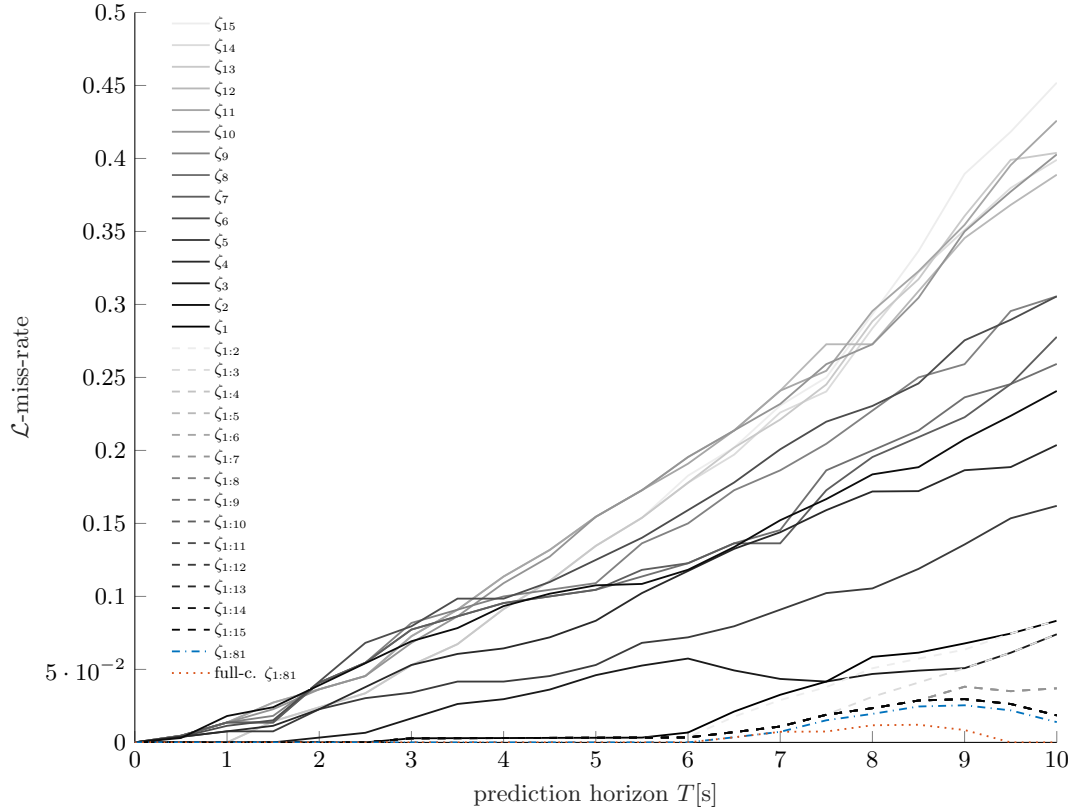
(a) Mean likelihood miss rate for various possible likelihood thresholds.



(b) Mean likelihood for all variants of  $n$ , all resulting in the same values. (c) Mean likelihood miss rate for the given threshold  $\mathcal{L}_{\text{threshold}} = 10^{-7}$ .

**Figure 6.18.** Comparison of likelihood and likelihood miss rate for full-combinatorial DBN and interacting single-agent DBNs using all available 81 hypotheses for the scene in Fig. 6.17. The 2D plot of the likelihood miss rate has a threshold of  $\mathcal{L}_{\text{threshold}} = 10^{-7}$ . It can be seen that both models achieve very similar results.

## 6 Experiments



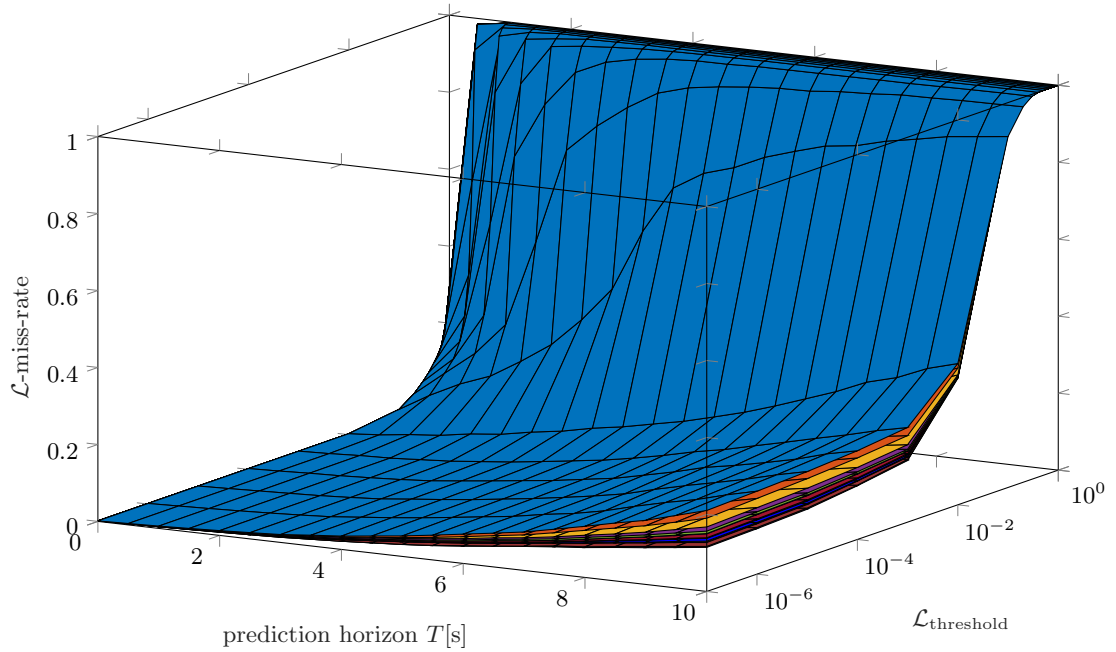
**Figure 6.19.** Comparison of likelihood miss rates of different hypotheses of the single-agent DBNs for the scene in Fig. 6.17 with  $\mathcal{L}_{\text{threshold}} = 10^{-7}$ . Solid lines represent only predicting a single hypotheses, ranging from the 15th-best hypothesis  $\zeta_{15}$  to the best hypothesis  $\zeta_1$ . Dashed lines represent predicting the  $n$  best hypotheses  $\zeta_{1:n}$  together, with  $n$  ranging from 2 to 15 (note that some of those lines are overlapping as they achieve identical likelihood miss rates). The blue dash-dotted line depicts the case when using all 81 hypotheses, the orange dotted line depicts the full-combinatorial DBN also with 81 hypotheses.

much a hypothesis might influence the autonomous vehicle’s future behavior can of course be very relevant selection criteria as well, but are left as future work. Given the assumptions from Sec. 4.4 of the single-agent DBNs, it is possible to determine the  $n$  most likely hypotheses  $\zeta_{1:n} = [\zeta_1, \dots, \zeta_n]$  without the need to track all  $|\mathcal{C}|$  possible hypotheses, which might be prohibitive in complex situations. Fig. 6.19 depicts the likelihood miss rates when predicting different hypotheses forward, ranging from the 15th-most-likely hypothesis  $\zeta_{15}$  over the most-likely hypothesis  $\zeta_1$  to the combination of the best  $n$  hypotheses  $\zeta_{1:n}$  weighted with their estimated probabilities. It can be seen that the hypotheses that have been estimated to have higher probability also perform better in the forward simulation. Furthermore, it can be seen that combining multiple hypotheses can further decrease the miss rate, as it allows to represent multi-modal distributions.

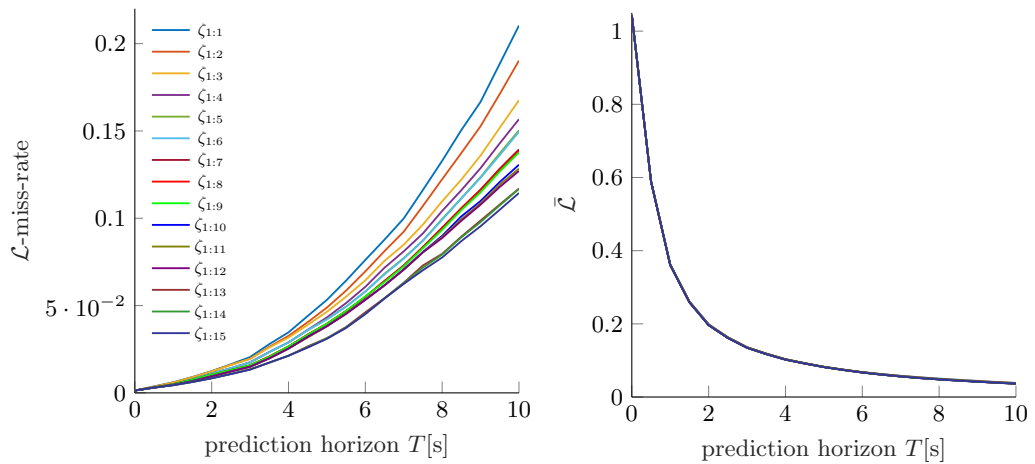
To analyze this effect in a more quantitative manner, we simulate a scene with multiple agents being spawned randomly around a four way intersection and picking their routes at random. This scene consists of 32 agents crossing the intersection. Fig. 6.20 depicts the average likelihood miss rate and the average likelihood depending on the number of hypotheses being considered during forward simulation (always using the  $n$  most likely hypotheses). It can be seen that more hypotheses in the forward simulation result in less misses of a given likelihood threshold.

Using the interacting single-agent DBNs, an important reduction of tracking complexity can be achieved while maintaining reasonably good intention estimates. The number of hypotheses to be considered during forward simulation can easily be adjusted according to runtime requirements, allowing to only predict the most-likely intention combinations without the need to enumerate and track the probabilities of all different hypotheses.

## 6 Experiments



(a) Mean likelihood miss rate for various possible likelihood thresholds.



(b) Mean likelihood miss rate for the given threshold  $\mathcal{L}_{\text{threshold}} = 10^{-7}$ . (c) Mean likelihood for all variants of  $n$ , all resulting in the same values.

**Figure 6.20.** Likelihood miss rate and likelihood depending on the prediction horizon for scene with multiple agents randomly traversing an intersection, averaged over all agents and time steps. As the complexity of the scene is too high to do inference in the full-combinatorial DBN, only the combinations created from the single-agent DBNs are shown.



## 6.6 Interactive Motion Planning

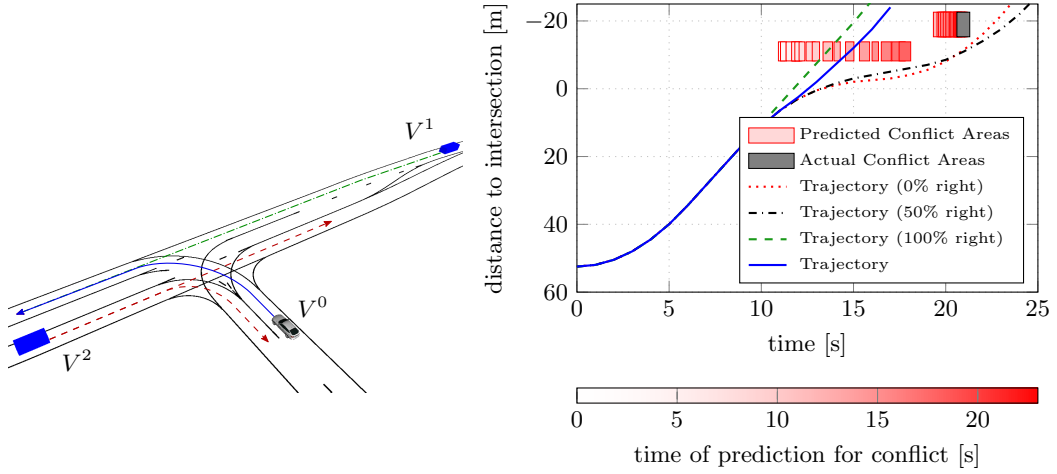
This section aims to show the benefits of the presented prediction approach for interaction-aware motion planning algorithms. As the task of ego-vehicle motion planning is not the focus of this thesis, the interested reader is referred to the following publications for a more detailed evaluation [161]–[163], [165].

### 6.6.1 Partially Observable Markov Decision Process

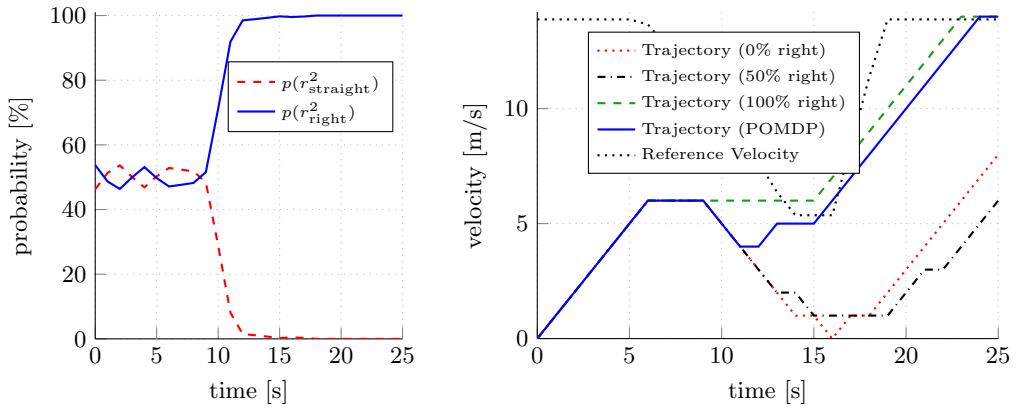
A major advantage of POMDPs is that they can deal with uncertainties and can find an optimal policy given that it is not known exactly what is going to happen in the future. One part of this uncertainty is the unknown intention of other agents and their unknown future behavior. The presented DBN allows to determine estimates about these quantities which can be utilized by the POMDP.

An example scenario that shows the POMDP’s planning routine at an intersection is shown in Fig. 6.21. Vehicle  $V^2$  has two route options, it can either turn right (thus not conflicting with the ego-vehicle) or go straight (such that the ego-vehicle has to yield). Ego wants to turn left and also has to account for vehicle  $V^1$  that has right of way. In the hypothetical case of ego knowing that  $V^2$  will turn right (green trajectory in Fig. 6.21(d)), it can merge before  $V^1$  and does not have to slow down. In the hypothetical case of ego knowing that  $V^2$  will go straight, it has to yield and needs to slow down and let both vehicles pass (red trajectory). If we assume a uniform distribution (50 % right, 50 % straight) which is constant over time, ego also has to slow down and yield to both vehicles as well (black trajectory), as it cannot risk a potential collision. In the case of the POMDP as presented in Chapter 5 (blue trajectory), the existence of future observations is explicitly modeled, allowing for less conservative behavior. Continuously updating our belief with new measurements and furthermore knowing that the estimates about the other agents’ intentions will improve over time enables the ego-vehicle to postpone the decision on whether to stop or not and keep both options available. As soon as it is confident enough of  $V^2$ ’s route, it decides that it does not have to yield and that it can still merge before  $V^1$ . One important aspect is that solely knowing that one will get additional observations in the future can help finding a better policy, even though one does not know what the observations will be.

The POMDP is solved using the *Toolkit for approximating and Adapting POMDP solutions In Real time (TAPIR)* [66], which is a C++ implementation of the Adaptive Belief Tree (ABT) [76] algorithm. The planning horizon is 8 s with a corresponding step size of 1 s. Further results of utilizing POMDPs to solve the problem of interactive motion planning for autonomous vehicles (including merging in dense traffic and handling occlusions) can be found in our previous publications [161]–[163].

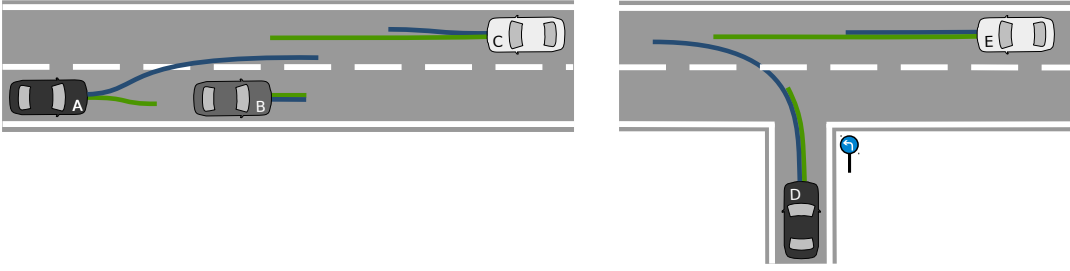


(a) T-junction scenario with two other vehicles. (b) Planned positions over time with different assumed probabilities of  $V^2$ 's route and with online inference.



(c) Route probability of  $V^2$  inferred over time by the POMDP inference. (d) Planned velocities over time with different assumed probabilities of  $V^2$ 's route and with online inference. Reference velocity is calculated based on road curvature.

**Figure 6.21.** Evaluation of a T-junction scenario with the ego-vehicle  $V^0$  wanting to turn left but potentially having to yield to two vehicles.  $V^2$ 's route is not known to ego, but it can be estimated over time. The POMDP solver keeps both options, yielding and not yielding, open until it is confident enough about  $V^2$ 's route. In the end,  $V^2$  is turning right and ego can merge before  $V^1$ . Previously published in [162]. © IEEE 2018



**Figure 6.22.** Two scenarios which are evaluated. The left image shows the *overtaking with oncoming traffic* scenario with three possible maneuvers:  $M_1$  represents  $V^A$  staying behind  $V^B$  (green).  $M_2$  represents  $V^A$  overtaking before  $V^C$  (blue).  $M_3$  represents  $V^A$  overtaking after  $V^C$  (identical to green trajectory in the beginning and then  $V^A$  overtaking after  $V^C$  having fully passed). The right image shows two vehicles *merging at an intersection*.  $M_4$  represents  $V^D$  merging first (blue) and  $M_5$  represents  $V^D$  merging second (green). Previously published in [165]. © IEEE 2017

### 6.6.2 Multi-Agent Planning

Regarding our prediction and planning approach that is based on cooperative multi-agent planning, we want to evaluate whether the Bayesian multiple model filter can outperform two other typical estimation methods that are commonly used in planning-based prediction: cost-based estimation and cost-gradient-based estimation. The cost-based estimation uses the softmax function to map the different planning costs of the optimal trajectories of each maneuver  $J_{M_i,t}^*$  to a probability distribution (such that low costs result in high probability). The cost-gradient-based estimation uses the difference of planning costs of two subsequent time steps  $\Delta J_{M_i,t}^* = J_{M_i,t}^* - J_{M_i,t-1}^*$  and maps this change of costs to a probability distribution (high decrease in costs result in high probability). Such a cost-gradient-based estimation has already been utilized in the context of autonomous vehicles in [38].

The following results are based on another closed-loop simulator in which the desired maneuvers of the non-ego-vehicles can be set a priori. All vehicles are controlled with model predictive control (MPC) according to the optimal trajectories of the given maneuver. Gaussian noise is added to the transition model and measurements. For analyzing the closed-loop estimation capabilities, the ego-vehicle is simply controlled with the optimal trajectory of the most likely maneuver (according to the selected estimation method).

The two scenarios from Fig. 6.22, *overtaking with oncoming traffic* and *merging at an intersection*, are statistically evaluated. For the overtaking scenario, we assume  $V^C$  to be the ego-vehicle,  $V^A$  to be predicted and  $V^B$  having constant velocity. Three possible maneuvers have been identified,  $M_1$  ( $V^A$  not overtaking but following  $V^B$ ),  $M_2$  ( $V^A$  overtaking before the oncoming ego-vehicle), and  $M_3$  ( $V^A$  overtaking after the oncoming ego-vehicle). For the *merging* scenario,  $V_E$  is assumed to be the ego-vehicle whereas  $V_D$  is to be predicted. Two maneuvers are distinguished:  $M_4$  ( $V_D$  merges before ego) and  $M_5$  ( $V_D$  merges after ego). All parameters used for the evaluation are

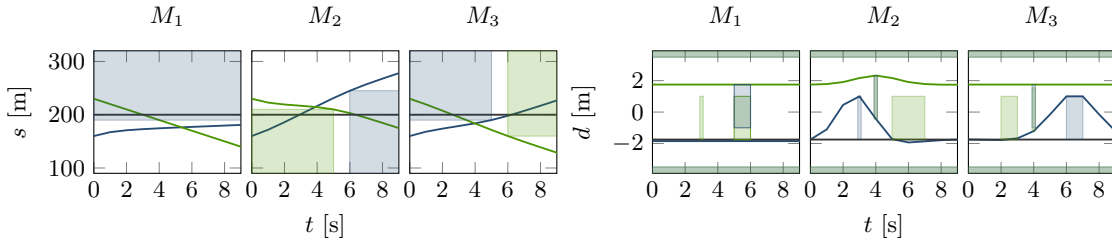
## 6 Experiments

**Table 6.4**  
EVALUATION PARAMETERS OF MULTI-AGENT PLANNING. ADAPTED FROM [165].

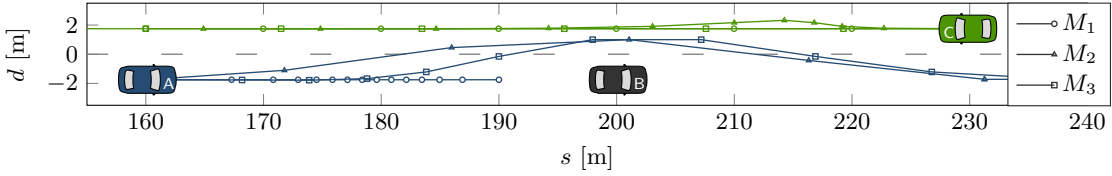
Planning and Estimation		Noise	
planning horizon $T$	14 s	$\sigma_s^2$	1 m <sup>2</sup>
step size $\Delta T$	1 s	$\sigma_d^2$	0.1 m <sup>2</sup> /s <sup>2</sup>
long. acceleration range $a_s$	$[-9, 5]$ m/s <sup>2</sup>	$\sigma_{v_s}^2$	0.25 m <sup>2</sup>
lat. acceleration range $a_d$	$[-2, 2]$ m/s <sup>2</sup>	$\sigma_{v_d}^2$	0.01 m <sup>2</sup> /s <sup>2</sup>
long. velocity range $v_s$	$[0, 20]$ m/s	$\sigma_{s_z}^2$	5 m <sup>2</sup>
lat. velocity range $v_d$	$[-5, 5]$ m/s	$\sigma_{d_z}^2$	5 m <sup>2</sup>
desired velocity $v_{\text{des}}$	10 m/s		
car length $l$	5 m		
car width $w$	1.75 m		
lane change length $\alpha$	2.5 m		
safety distance $\beta$	30 m		
right of way weight $\psi$	1		
maneuver switching probability $\mu$	0.1		

chosen by domain knowledge and are shown in Tab. 6.4 (see Sec. 5.2 for definitions). To reflect different situations, the initial conditions consisting of desired velocities and initial positions of all vehicles are altered iteratively. Each run consists of 14 seconds and is repeated five times with the same initial conditions but different random seeds for each possible maneuver intention. The execution of one time step of the overtaking scenario including maneuver determination ( $C++$ ), multi-agent trajectory planning and IMM update (*MATLAB*) takes approximately 0.26 s on an *Intel Core i7-4910MQ* with 2.9 GHz without code optimization or parallelization.

The trajectory planning results are exemplarily depicted for the overtaking scenario in Fig. 6.23, showing the cost-optimal trajectories for each maneuver and the corresponding hard constraints. Within this example,  $V^B$  is considered to be static (e.g., a parked car). For maneuver  $M_2$  where the blue agent is overtaking the parked car before ego has passed it, the optimal trajectory for ego creates a little more lateral space for the oncoming car by driving further to the side, thus supporting the other agent’s decision and being cooperative. Fig. 6.24 depicts the corresponding IMM maneuver probabilities depending on the tracking time steps with the maneuver intention set to  $M_1$  ( $V^A$  following  $V^B$ ). Although maneuver  $M_1$  and  $M_3$  have similar optimal trajectories in the beginning, they can still be distinguished after about three time steps. Maneuver  $M_2$  results in  $V^A$  accelerating already in the very beginning, making it easier to distinguish from the other two maneuvers. The confusion matrices given in Tab. 6.5 show a quantitative comparison of the actual maneuver and the estimates of the different estimators for time steps  $t_{1:14}$  for the cost-based estimator and  $t_{2:14}$  for the other estimators (as they need at least two time steps for estimation). The results show for both overtaking (left part) and merging (right part) that only relying on the costs often yields wrong maneuver estimates, as the simulated vehicles randomly decide for a maneuver which is not necessarily the one with lowest costs. However, the costs of the intended maneuver tend to get smaller the longer the predicted vehicles follow their intentions, slightly



(a) Longitudinal components of trajectories and (b) Lateral components of trajectories and respective hard constraints.

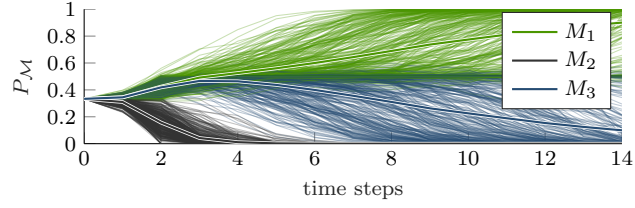


(c) Optimal trajectories of all considered maneuvers.

**Figure 6.23.** Overtaking scenario with oncoming traffic: Multi-agent trajectories for all maneuvers in  $\mathcal{M}$ . The colors of the trajectories (lines) and the hard constraints (areas) correspond to the vehicles' colors. The hard constraints constrain the trajectories due to the road boundaries and other vehicles: the trajectories are not allowed to be inside the areas of same color. Previously published in [165]. © IEEE 2017

improving the accuracy over time. One advantage of the cost-based estimation is the instant assessment without being dependent on the availability of any history, i.e., it does not have to wait for the second time step for its initial classification. Also, the cost-based classifier could potentially perform better in non-simulated traffic, as humans might choose their maneuver more often according to its expected cost and not purely at random as in our simulation. The cost-gradient estimator achieves better results in our environment and faster convergence, as it uses two consecutive measurements and therefore is able to analyze the actual behavior between two time steps instead of only utilizing a single frame of the situation. Thus, even if the agent chooses a high-cost maneuver, by following the respective trajectory, the cost-gradient for that maneuver might still be the highest. However, the cost-gradient completely neglects the absolute costs, sporadically resulting in even worse accuracies. In our experiment, the IMM estimator, using all available past measurements and focusing on the actual trajectory similarities between observations and prediction instead of solely relying on the costs, outperforms the cost-based as well as the cost-gradient-based classification.

## 6 Experiments



**Figure 6.24.** IMM maneuver probabilities for the overtaking scenario with maneuver intention  $M_1$  ( $V^C$  follows  $V^B$ ). The mean values of all runs with different initializations are drawn as lines with white borders. Previously published in [165]. © IEEE 2017

**Table 6.5**

CONFUSION MATRICES FOR ESTIMATORS BASED ON  $J_{\mathcal{M}}^*$ ,  $\Delta J_{\mathcal{M}}^*$  AND  $p_{\mathcal{M}}$  FOR OVERTAKING SCENARIO (LEFT) AND MERGING SCENARIO (RIGHT). PREVIOUSLY PUBLISHED IN [165]. © IEEE 2017

Actual	based on $J_{\mathcal{M}}^*$			Acc	Actual	based on $J_{\mathcal{M}}^*$		Acc
	$M_1$	$M_2$	$M_3$			$M_4$	$M_5$	
$M_1$	<b>746</b>	811	3393	0.15	$M_4$	<b>11112</b>	5763	0.66
$M_2$	312	<b>3833</b>	805	0.77	$M_5$	5660	<b>11215</b>	0.66
$M_3$	693	836	<b>3421</b>	0.69				<b>0.66</b>
				<b>0.54</b>				

Actual	based on $\Delta J_{\mathcal{M}}^*$			Acc	Actual	based on $\Delta J_{\mathcal{M}}^*$		Acc
	$M_1$	$M_2$	$M_3$			$M_4$	$M_5$	
$M_1$	<b>505</b>	517	3598	0.11	$M_4$	<b>13690</b>	2060	0.87
$M_2$	19	<b>4458</b>	143	0.96	$M_5$	2352	<b>13398</b>	0.85
$M_3$	546	526	<b>3548</b>	0.77				<b>0.86</b>
				<b>0.61</b>				

Actual	based on $p_{\mathcal{M}}$			Acc	Actual	based on $p_{\mathcal{M}}$		Acc
	$M_1$	$M_2$	$M_3$			$M_4$	$M_5$	
$M_1$	<b>3476</b>	60	1084	0.75	$M_4$	<b>15060</b>	690	0.96
$M_2$	20	<b>4539</b>	61	0.98	$M_5$	530	<b>15220</b>	0.97
$M_3$	825	84	<b>3711</b>	0.80				<b>0.96</b>
				<b>0.85</b>				

## 7 Discussion

Self-driving cars are considered to be a game-changing technology for future mobility. They are expected to drastically reduce the number of accidents and thus human injuries and fatalities, improve traffic flow and efficiency, and to allow passengers to be productive while traveling. Furthermore, they open up possibilities of efficient and affordable ride-hailing and car-sharing services, reducing the need for privately owned cars.

A major difficulty of self-driving cars is to achieve behavior that is safe while not being over-cautious [29]. One component of achieving this behavior is the anticipation of the motion of surrounding agents. Accounting for what other agents intend to do within the ego-vehicle motion planning allows to drive in a cooperative and foresighted manner. Modeling the influence of the ego-vehicle’s future motion on surrounding agents enables to drive less conservatively, allowing for interactive maneuvers such as merging into narrow gaps. The prediction of multiple interacting agents is combinatorial in nature, highly depends on the situational context, and is inherently uncertain and multi-modal, thus still representing a great challenge today [81].

### 7.1 Conclusion

This thesis has presented a context-dependent, interaction-aware and probabilistic prediction framework that is able to estimate high-level driver intentions and to predict complete scene developments considering the interdependencies between multiple agents’ trajectories. The set of possible intentions is generated during runtime given a topological map and the current belief state, removing the need to predefine a discrete set of classes and allowing for arbitrary road layouts. The proposed approach is capable of dealing with an arbitrary number of agents, with the uncertainty in measurements and human behavior, and with the inherent multi-modality of possible future trajectories. Additionally, we presented two interactive ego-vehicle motion planning algorithms for autonomous vehicles that build on this prediction framework and account for the interrelation of ego-vehicle planning and the prediction of surrounding agents, resulting in more interactive and less conservative behavior.

**Combined Estimation and Prediction:** The decision making processes of multiple interacting agents in a scene are modeled in a dynamic Bayesian network (DBN). By including all agents within the state space, it can be accounted for interactions between multiple agents. As this DBN is a generative models, it allows to both infer the hidden intentions of drivers with recursive Bayesian estimation and to predict how a traffic scene will evolve in a probabilistic manner using forward simulation of the current belief. Intention estimation and trajectory prediction are very related problems, as they both describe how an agent is going to act in the future, but on different levels of abstraction.

We handled these problems in a combined way by defining a single generative model, in contrast to having two distinct models as commonly done in existing literature. This ensures consistency of the results of both levels of abstraction and reduces design effort as only one model is needed. An improvement in the state prediction model does not only benefit the trajectory prediction, but also directly benefits the intention estimation capabilities. Fine-tuning of the parameters of this model by hand is considerably less cumbersome than the ones of a discriminative classifier, as the forward simulation can be easily compared to the ground truth trajectories. Another advantage of this Bayesian estimation framework is that the kinematic states of all agents are filtered in a more informed way by utilizing more sophisticated prediction models compared to classical tracking methods that only rely on physics-based state prediction.

**High-Level Intentions:** We introduced high level intentions into the decision making process of an agent to subdivide the tracking and prediction problem into multiple subproblems. Conditioning the behavior model on these intentions reduces modeling complexity and improves interpretability. Furthermore, it enables to estimate these intentions which might in turn be needed within subsequent software modules of an autonomous vehicle such as planning algorithms for making high-level maneuver decisions. In this work, we introduced two types of high-level intentions, the route intention (i.e., the path an agent desires to follow) and the maneuver intention given a specific route (i.e., whether an agent wants to pass a conflict area before or after other agents).

Our evaluation has shown that the presented framework is able to estimate these intentions even in interactive scenarios in which non-interactive models do result in wrong findings. By implicitly reasoning about what might influence an agent’s behavior depending on its intention, it can come up with conclusions such as that the deceleration before an intersection might be rather due to a slow preceding agent and not due to an upcoming turn (resulting in a uniform route distribution, as all options are equally likely), or that when stopped at an intersection it is unlikely for an agent to follow a route for which it would not even have to yield.

**Route Intention:** In contrast to many existing works, the different possible route intentions have not been predefined by hand (such as “turn right”, “turn left”, and “go straight”), but are determined at runtime according to a topological map. Thus, arbitrary road networks and an arbitrary number of route hypotheses can be represented. We conditioned the action of an agent on its route intention, allowing to extract relevant features more easily such as the upcoming curvature of that route or the prevailing traffic rules. The action thus does not depend on a single discrete label of the route such as “turn left”, but on the *characteristics* that describe the route by a multitude of continuous and discrete features. We utilized the same set of features for each route, resulting in a *fixed number of inputs* to the action model. This enables the use of various fixed input size learning models such as typical neural network architectures. For each possible route, the action model can then be queried, allowing for an arbitrary number of hypotheses. The evaluation of this thesis has shown that the route intention is one of the main causes of multi-modality of the future trajectory of an agent. Estimating the route intention thus allows to draw important conclusions about what modes of the prediction are more or less likely and thereby improve overall prediction.



An assumption that has been made by conditioning the behavior model only on the characteristics of a single route is the disregard of the existence of other possible routes and their respective appearances, or mathematically speaking  $p(\mathbf{a}|r_i, \mathcal{R}) = p(\mathbf{a}|r_i), r_i \in \mathcal{R}$ . If the sole existence of other routes did influence an agent’s actions, the input would not be of fixed size anymore. A possibility to improve on this assumption of independence of other routes is to use a behavior model that is able to handle variable input sizes, such as recurrent [91] or recursive models [124], or utilizing fixed input sized featuremaps such as top-down images. So-called sequence-to-sequence models [135] could even be used to directly learn the actions of all possible routes as they allow for a variable input and variable output size.

**Maneuver Intention:** The second high-level intention introduced within this thesis is the so-called maneuver intention, which describes the sequence in which agents are going to pass an upcoming conflict area. This maneuver definition follows the concept of trajectory homotopy and that agents engage in mutual collision avoidance. It is able to distinguish both spatial as well as temporal aspects. The set of possible maneuvers of an agent given a specific route hypothesis is determined at runtime depending on the existence of other agents and the topological map. Thus, an arbitrary number of maneuver hypotheses can be generated. The maneuver intention represents another strong cause for multi-modality, as drivers may change their underlying actions drastically depending on their maneuver decisions. Thus, the maneuver estimation also yields interesting insights on the decision making process of an agent and thereby in the prediction of their continuous motion.

**Inference Methods:** This thesis proposed two different inference methods for the aforementioned DBN, a sequential Monte Carlo (SMC)-based and a multiple model unscented Kalman filter (MM-UKF)-based method. Given a specific combination of route and maneuver intentions of all agents, the evaluation of this thesis has shown that the belief and predictions of the kinematic states stay mostly unimodal, such that they can reasonably be approximated by a single multivariate Gaussian. This enabled us to utilize the proposed MM-UKF-based inference method, where each mode represents a combination of route and maneuver intentions of all agents. Compared to the baseline SMC method that does not come with a Gaussian assumption, the MM-UKF method enhances inference efficiency drastically: Our evaluation has shown that it has a generally lower runtime than SMC with comparable accuracy or achieves improved accuracy given a similar runtime.

The computational efficiency is especially pronounced when the belief has high uncertainty, as SMC requires many particles to represent a widespread distribution. Especially when the measurement noise is low compared to the system noise, many particles end up being eliminated during resampling because of their low probability. This amplifies the typical problems of sample degeneracy and impoverishment. One possible improvement could be to sample from a more informed proposal distribution that takes into account the already received measurement, as suggested in [137, p. 262].

The comparison of multiple runs with different SMC random initialization highlighted the high variance in the estimation results, which is naturally not the case for MM-UKF. MM-UKF ensures reproducible results allowing for an easier validation of the

overall framework. However, SMC has the advantage that the number of samples can be adjusted according to runtime requirements, whereas the number of sigma points for the MM-UKF is defined by the situation. Furthermore, discrete variables and non-additive noise terms can be added straightforwardly for SMC, whereas for MM-UKF the modes need to account for each discrete variable and the state of the sigma points needs to be augmented for each non-additive noise term, making SMC more flexible.

Both proposed inference methods are able to represent the multi-modal and hybrid belief state and thus are well suited for the problem domain of behavior prediction. Another significant benefit of these inference methods lies in their sample-based nature: The samples can be populated through highly non-linear transition functions while no derivatives have to be calculated. The employed non-differentiable models in this work (such as the lane-matching model) forbid the usage of inference methods that rely on derivatives of the transition models such as the extended Kalman filter (EKF).

**Rule-Based Behavior Model:** In this thesis, we proposed two distinct action models that represent a probabilistic mapping from a feature-based representation of a traffic scene from the point of view of a driver to a distribution over their continuous actions: one rule-based and one deep learning-based model. Both models take the available information from a digital map (topology, geometry, traffic infrastructure, and traffic rules) as well as the dependency on surrounding agents into account. Our proposed rule-based model partly builds on the established intelligent driver model (IDM) used for highway car-following, but in contrast is suitable for complex urban environments and able to account for uncertainty in human behavior. We have achieved this by introducing new submodels for curvature approach, intersection crossing, and stopping at red lights or stop signs, allowing to predict on various road layouts and account for typical influences in urban environments. It is based on the assumptions that agents tend (with some uncertainty) to drive as fast as possible within the speed limits while remaining safety distances to other agents and that they tend to drive close to the center of the lane. The interaction-awareness is modeled as mutual dependencies between multiple agents: Using small time steps of independent actions allows to break down these interdependencies and to predict the action of each agent for one time step without knowing the actions of others. This is a small assumption, as one can argue that human drivers also do not know what other drivers are going to do with complete certainty. Thus, the presented behavior models only depend on the current context and not on the prediction of others, but still act in an anticipatory way (i.e., make implicit predictions on how the scene is likely going to evolve, e.g., by assuming constant velocity of others).

The hand-tuned model was successfully employed for long-term motion prediction and the estimation of both route and maneuver intentions of multiple interacting agents in simulation and in real world scenarios. The evaluation in this work has shown that the map-dependency as well as the interaction-awareness are very important aspects, allowing the rule-based model to outperform simpler interaction-unaware or solely physics-based models. An interesting aspect is that despite this importance, statistically, this context-dependency might seem to be of minor relevance: Most of the time, predicting using only physics-based models such as CTRV results in satisfactory performance, as drivers tend to change their behavior slowly enough. However, in situations that re-

quire a fast change of behavior, such as on curvy roads or at intersections, physics-based models come to a limit. Furthermore, with physics-based models, intentions cannot be estimated (as they do not depend on a driver’s intention). A similar statement can be made for interaction-awareness: Although in many situations, predicting an agent on its own while neglecting the existence of other agents may perform adequately, there exists a variety of situations in which a high interdependency between agents is present, resulting in interaction-unaware models to fail drastically. As it is hard to foresee when a situation might become interactive, forward simulation should be done in a combinatorial and interaction-aware fashion.

As we added Gaussian noise to the actions (and potentially also to the states) to account for behavior uncertainty, the resulting trajectories of the rule-based model follow a somewhat “zigzag motion”. More consistent trajectories could be achieved by sampling from uncertain behavior model parameters that represent the driving style (such as comfortable breaking deceleration or desired time gaps) and not having any uncertainty in a driver’s actions given these parameters. These parameters could in turn also be estimated as part of the DBN during the runtime.

**Deep Learning-Based Behavior Model:** The deep learning-based action models embedded into the presented DBN have shown how novel deep learning approaches can be combined with classical Bayesian estimation approaches and expert knowledge. Conditioning the model on the route intention, which is the main cause of multi-modality in predictions, reduces learning complexity and allows to handle a variable number of hypotheses while still having a fixed input size model. With the presented network architectures, we were able to learn lane following, distance keeping, yielding to other traffic participants at intersections, stopping at red lights as well as to even capture subtleties such as cutting curves. By also learning the variance in a driver’s actions, we accounted for the context-dependent magnitude of uncertainty in the transition model, which better captures the reality of human driving and which is difficult to model by hand. Sampling from this learned distribution also resulted in more consistent trajectories compared to the “zig-zag motion” of the rule-based model. Due to the simplicity of the models (2-4 layers), they come with low runtimes, enabling their application to sampling-based frameworks such as Monte Carlo tree search (MCTS) or sequential Monte Carlo (SMC). The evaluation in this thesis has shown that the proposed deep learning models can outperform the hand-tuned rule-based model, especially performing better for short prediction horizons of up to around 10s. As a result of the significant decrease of the one-step prediction error, the intention estimation quality improved considerably. Having models with higher accuracy in turn reduces the need for a high number of particles.

Achieving thorough generalization with deep learning-based prediction models is still a great challenge. In supervised learning, it is typically assumed that the training and testing data are independent and identically distributed (i.i.d.). For an iterative forward simulation, this is however not the case, as the future inputs to the model are influenced by the learned policy itself. Long-term predictions can cause the compounding error to grow beyond a point at which the input features are too different from the training distribution and thus result in poor predictions. This could be diminished with more di-

verse training data and with data augmentation techniques such as data as demonstrator [148] and dataset aggregation (DAGger) [111], allowing the models to learn to approach the ground truth trajectories again after slight deviation. A more fundamental alternative, as explained in [110] in the context of imitation learning, is the use of inverse reinforcement learning or general adversarial imitation learning instead of supervised learning.

A related challenge can be found in the prediction conditioned on very unlikely route hypotheses: as the model was trained with data from another distribution in which this hypothesis never or only very rarely occurred, the forward simulation may result in unreasonable behavior (as shown in Fig. 6.12). Two possibilities to improve this are to either increase the diversity and the amount of training data such that it is accounted for such outliers, or to introduce learned priors that allow to pre-prune very unlikely hypotheses before they are even tracked or predicted.

A requirement for learning the behavior model in a supervised fashion is the availability of ground truth data, including both the actions to be predicted as well as the intentions on which the model is conditioned on. We assumed a driver does not change its route intention and thus were able to automatically label it after having observed a trajectory that completely traversed an intersection. It is apparent that this is a mild assumption, as drivers decide for a route mostly based on their navigation goal which is typically not influenced by the situational context. However, this is not the case for the intended maneuver, which an agent might change in the course of getting closer to an intersection. This makes it difficult to determine the actual ground truth labels, as it would require a survey of humans disclosing their intentions while driving, which is impractical. Instead, this thesis has shown that the maneuver intentions can be learned implicitly within the action model, such that the maneuvers are represented by the trajectories generated from the learned actions. Thus, the deep learning-based behavior model is not conditioned on an explicit maneuver intention but decides on the maneuver implicitly on its own. Conditioning on uncertain intentions during training requires so-called *learning from incomplete data* which could be achieved with the expectation maximization (EM) algorithm [42], [43].

**Combinatorial Complexity:** The prediction of multiple interacting agents under consideration of uncertainty is combinatorial in nature. As agents may not only interact with each other at the current point in time, but also at any point in the future, the predictions of all agents potentially become interdependent. The combinatorial complexity is thus inherent to the problem of interaction-aware prediction and the number of possible combinations grows exponentially with the number of agents. Existing literature often either neglects this interaction altogether or introduces strong assumptions, resulting in lower complexities on the one hand, but also in potentially inaccurate predictions on the other hand. In cases with close interaction between traffic participants, the results of this thesis have shown that their interdependencies should not be neglected.

In this thesis, we proposed a full combinatorial solution method that allows to track and predict all possible combinations of route and maneuver intentions of all agents. However, if the number of combinations is too large to be computationally feasible, the question on which combinations to neglect has to be answered. Limiting the number of

particles in the SMC-based inference method represents one possibility to limit computation time, effectively pruning hypotheses based on their likelihood. Analogously, pruning by likelihood can also be achieved with the MM-UKF inference method. However, if the number of possible combinations is too large to even be tracked for a short period, there remain untracked hypotheses for which the likelihoods cannot be estimated. Thus, new combinations to replace unlikely already tracked ones have to be selected at random or solely based on priors.

To this end, we presented so-called *interacting single-agent DBNs* as another possibility to get this complexity under control. They track each of the agents in a scene on its own utilizing statistics about the other agents' poses, resulting in a tracking complexity reduction from exponential to linear in the number of considered agents. Using the estimated single agent intention probabilities, the most likely combinations (assuming independence of intentions) can then be determined in order to focus computational efforts on the most promising hypotheses for the combinatorial forward simulation. Our results have shown that the single-agent filters come with heavily reduced runtime while maintaining tracking capabilities comparable to the ones of the full-combinatorial method and thus represent a reasonable approximation. Further, we have shown that picking the most likely hypotheses based on this assumption improves prediction accuracy compared to selecting hypotheses that have lower probability estimates.

**Interactive Ego-Vehicle Motion Planning:** Considering interaction is not only important for the prediction of other agents, but also for interactive ego-vehicle motion planning. Although the prediction of surrounding agents and the planning of the ego motion are highly coupled problems (prediction influences planning and vice versa), these problems are often tackled separately in existing literature. A very common approach is to solve prediction first (neglecting the influence of the ego-vehicle) and then utilizing this prediction as an input for the motion planning algorithm to plan trajectories around these fixed predictions to avoid collisions [63], [94], [145]. Although this may work in less interactive scenarios, other scenarios like merging in dense traffic require to model the influence of the ego-vehicle on other agents in order to be successful. Not modeling this influence typically results in too conservative behavior, as can be seen in recent complaints of residents about too conservative self-driving cars at intersections [29]. To this end, we proposed two methods to solve these interdependent prediction and planning problems in a coupled manner, utilizing the presented Bayesian prediction framework.

In the first method, we formulated the problem as a partially observable Markov decision process (POMDP) that includes the presented DBN as part of the state space. Utilizing a point-based solver transformed the problem into a Monte Carlo tree search (MCTS) problem, allowing for an efficient search within the solution space. Multiple episodes are simulated forward in order to evaluate the expected costs of possible ego-vehicle trajectories according to the corresponding likely reactions of the surrounding agents, taking the ego-vehicle's influence on its surrounding agents into account. Reasoning about future observations allows to anticipate knowledge that will be gained in the future, thus allowing to drive less conservatively. Because of the coarse action discretization, this method should be combined with a continuous trajectory optimization that smoothes the optimized action sequence. However, with the expected increase in

computational power in the future, continuous action spaces, smaller step sizes, and longer planning horizons will become feasible. Although the complexity of solving the POMDP for complex situations can be very high, the anytime capability of the point-based solver allows to still handle scenarios with many agents and find reasonable but potentially suboptimal solutions.

The second method utilizes cooperative multi-agent planning for predicting other agents and deriving ego-vehicle trajectories. We defined so-called *collective maneuvers* for the multi-agent case, describing the relative motion of all considered agents including the ego-vehicle (in contrast to the other parts of this thesis where a maneuver represents pairwise relations from a single agent’s perspective). For all possible maneuvers, multi-agent trajectories based on mixed-integer quadratic programming (MIQP) are generated. The planned trajectories of the surrounding agents represent their predictions and are used as transition models for the Bayesian maneuver estimation. The ego-vehicle can be controlled with the corresponding trajectories of the most likely maneuver, thus ensuring consistency of planning and prediction. By defining a global cost function that is composed of cost-terms for all vehicles, this method generally assumes cooperative behavior of others. Nevertheless, our evaluation has shown that due to the Bayesian estimation, we were still able to detect when agents decide for a suboptimal maneuver, enabling the ego-vehicle to act accordingly. In contrast to the POMDP approach, this method explicitly analyzes all available homotopy classes and tries to estimate which option is intended by other traffic participants. Thus it allows to also decide for suboptimal maneuvers (according to the cost function) in case other agents seem to follow a suboptimal option and we want to accommodate. For complex situations, reasonable pre-pruning becomes necessary, as the number of possible homotopy classes grows exponentially with the number of agents and thus enumerating all of them becomes infeasible. To further improve the approach, recovering a cost function which describes real human behavior more accurately could be achieved by utilizing inverse reinforcement learning (IRL) in the future.

Our results have shown that both of these methods allow for dense and interactive scenarios to be succeeded by an autonomous vehicle in a less conservative way, as it is accounted for the influence of the ego-vehicle on the actions of surrounding agents. Future work should investigate how safety specifications can be determined for these approaches, either based on statistical evaluation or by adding an additional safety layer that allows for formal verification.

## 7.2 Future Research Directions

The prediction of human drivers’ motion and the estimation of their intentions as well as how to include this knowledge within the motion planning of an autonomous vehicle are still part of active research. This thesis cannot fully cover all aspects of these topics and thus we present some interesting open points in the following that should be investigated in the future. Besides the already mentioned assumptions and possible improvements

of the presented approach in the previous section, there are also more general research directions that are worth looking into.

**Anomaly Detection:** An important aspect of conditioning an agent’s behavior on its intentions is the necessity of completeness of hypotheses: If the actual route or maneuver an agent desires to follow is not represented within the set of possible intentions, no meaningful estimation and prediction will be possible. For example, at an intersection we might ask the question “Will the agent go straight or will it turn left?”, when it actually is going to make a U-turn, but we were not able to detect that this possibility even existed. This might happen if the topological map is outdated (e.g., due to construction zones) or not detailed enough (e.g., missing driveways) or if a misdetection of another agent occurs, resulting in only a subset of possible intentions being extracted. Another issue lies in the possibility of wrong features for the action model being extracted, e.g., due to the misdetection of a traffic light’s state. As it is not possible to guarantee that all intentions can always be determined and all features are extracted correctly, a prediction and estimation framework should be able to handle such cases. One possibility to deal with this problem is to introduce an anomaly detection of an agent’s behavior (e.g., based on the measurement likelihoods of all existing hypotheses) and a fallback prediction model that might be solely physics-based. When an agent does not behave as expected given the existing models, the fallback model is activated and used for the forward simulation.

Another benefit of such a fallback solution is the actual case of abnormal behavior of a driver: If the used models (despite correct hypotheses and features) do not describe the actual behavior of a driver well, a physics-based model might still yield sufficient predictions. A similar difficulty can be found in areas where no lanes are present such as on parking lots, in which a physics-based fallback solution could also be employed.

As proposed by Fisac et al. [39], it is also possible to reason about the confidence of the prediction model at runtime and to increase model uncertainty when the model performs poorly.

**Occlusions:** Another major direction is the field of occlusions: In contrast to standard measurement noise or misdetections, it is possible to be aware of occluded areas and thus know that it is impossible to get detections within this area. Therefore, occlusions can be handled differently than other missing detections due to sensor fault. Most existing literature, including this thesis, does not explicitly model occlusions and assumes that all objects can be detected. Explicit modeling of the existence probability of objects in occluded areas could improve both prediction as well as ego-vehicle planning. It is even possible to infer the existence of objects given the behavior of other agents that might have a different field of view [132]. How occlusions can be modeled within the ego-vehicle motion planning using POMDPs can be found in our work [161].

**Prior Distributions:** In this work, we have set the priors over the possible intentions according to a simple uniform distribution. Although promising results could already be achieved, having more informed priors will further improve prediction and estimation quality. One way of achieving this is to utilize simple heuristics such as it being more likely to drive straight than to turn, or that it is more likely to merge in a gap that can be reached without strong acceleration or deceleration. Another possibility is to derive

the priors based on real driving data. The route prior could for example be learned given features such as lane types, turn directions, vehicle types, and velocities or simply be queried from a lookup table given a specific intersection id which is based on traffic statistics. The maneuver priors could for example be learned based on features such as distances to the conflict areas, velocities and road information such as curvature and traffic rules.

**Pruning of Hypotheses:** The number of possible combinations for predicting a traffic scene in a full-combinatorial fashion (in order to account for possible future interaction) grows exponentially with the number of considered agents. Thus, pruning becomes unavoidable for more complex scenes. In this thesis, we proposed to get a grip on this combinatorial explosion by pruning according to probability. This allows to account for the most likely future trajectories while enabling real time performance.

However, it is up for debate whether the probability alone is a sufficient measure for neglecting possible future trajectories. It stands to reason that events of low likelihood but high impact should also be considered, as over the course of the high number of miles driven, these events might actually occur. Thus, we suggest to extend the presented framework in the future to additionally estimate the impact of the considered hypotheses on the future behavior of the ego-vehicle. Appropriate utility functions that assess a hypothesis’ relevance and criticality could then be taken into consideration for pruning, instead of solely relying on the estimated probabilities.

**Evaluation Metrics:** The evaluation of a prediction algorithm is not as straightforward as it may seem at first. One problem lies in the multi-modality of prediction, potentially rendering typical metrics like the root mean square error (RMSE) or the likelihood to be counterintuitive, as even a perfect model might result in lower scores as obviously false models. To this end, we proposed an additional metric, the so-called *likelihood miss rate*, that measures how often the prediction has a likelihood below a certain threshold given the corresponding observed trajectory, i.e., the trajectory is “too far outside” of any significant mode of the prediction distribution, or informally speaking, has not been predicted correctly. Standardized and adequate metrics should be developed further and adopted widely in order to improve comparability between different works.

If a prediction approach is to be utilized by an autonomous agent, its predictive performance should ultimately be assessed in a closed-loop setting as well. Due to the potentially different actions of an autonomous vehicle (compared to human drivers) and the corresponding reactions of surrounding agents, one might end up in new situations that have not been observed before. When the prediction model is trained with human-only data but tested on data that includes an autonomous agent, the i.i.d. assumption of training and testing distributions is violated. This so-called *covariate shift* might result in poor prediction performance. In an extreme case, the motion planning algorithm might even exploit an insufficiently trained predictor to achieve low costs in planning that are based on false predictions and thus result in unsafe behavior. Although open-loop metrics allow for a first assessment and a comparison between different prediction approaches, evaluating how an autonomous vehicle acts given a predictive model is an important aspect that should be investigated in the future.



**Evaluation Data:** Another crucial component of evaluation is the existence of high quality traffic datasets that contain important contextual information such as the road geometry, road topology, lane markings, traffic infrastructure, and the states of surrounding agents. Unfortunately, there have been very few public datasets, that typically do not contain urban scenarios, sufficient road and infrastructure information, already detected and tracked objects or scenarios with extensive interaction between traffic participants. The so-called NGSIM highway datasets [26], [53] are among the most popular datasets in the prediction literature. Recently, as autonomous driving gains more research focus and interest of large companies, new datasets have become available such as the highD dataset [70], the INTERACTION dataset [157] and the datasets by Lyft [64] and Waymo [134], [151]. Thus, we can expect that future work will be based on publicly available datasets more frequently, resulting in more comparable research.

On top of that, we want to highlight the potential benefits of creating a public prediction challenge (as often done in the field of computer vision) that includes diverse and interactive scenarios in both urban and highway environments and utilizes standardized and adequate evaluation metrics. This would allow authors to benchmark and compare their approaches more fairly and thus be a great contribution to the motion prediction community.



# Bibliography

## Literature

- [1] P. Abbeel and A. Y. Ng, “Apprenticeship learning via inverse reinforcement learning,” in *International Conference on Machine Learning (ICML)*, ACM Press, 2004, p. 1.
- [2] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, “Social LSTM: Human trajectory prediction in crowded spaces,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2016, pp. 961–971.
- [3] M. Althoff and S. Magdici, “Set-based prediction of traffic participants on arbitrary road networks,” *Transactions on Intelligent Vehicles (TIV)*, vol. 1, no. 2, pp. 187–202, 2016.
- [4] M. N. Andersen, R. O. Andersen, and K. Wheeler, “Filtering in hybrid dynamic bayesian networks,” in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 5, IEEE, 2004, pp. V–773.
- [5] G. S. Aoude, V. R. Desaraju, L. H. Stephens, and J. P. How, “Behavior classification algorithms at intersections and validation using naturalistic data,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2011, pp. 601–606.
- [6] G. S. Aoude, V. R. Desaraju, L. H. Stephens, and J. P. How, “Driver behavior classification at intersections and validation on large naturalistic data set,” *Transactions on Intelligent Transportation Systems (TITS)*, vol. 13, no. 2, pp. 724–736, 2012.
- [7] A. Armand, D. Filliat, and J. Ibanez-Guzman, “Modelling stop intersection approaches using gaussian processes,” in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2013, pp. 1650–1655.
- [8] K. Asadi, D. Misra, S. Kim, and M. L. Littman, “Combating the compounding-error problem with a multi-step model,” *arXiv:1905.13320*, 2019.
- [9] J. A. Bagnell, “An invitation to imitation,” Technical Report, Carnegie Mellon University, 2015.
- [10] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” *arXiv:1409.0473*, 2014.
- [11] M. Bahram, C. Hubmann, A. Lawitzky, M. Aeberhard, and D. Wollherr, “A combined model-and learning-based framework for interaction-aware maneuver prediction,” *Transactions on Intelligent Transportation Systems (TITS)*, vol. 17, no. 6, pp. 1538–1550, 2016.

- [12] Y. Bai, Z. J. Chong, M. H. Ang, and X. Gao, “An online approach for intersection navigation of autonomous vehicle,” in *International Conference on Robotics and Biomimetics (ROBIO)*, IEEE, 2014, pp. 2127–2132.
- [13] B. Banerjee and B. Chandrasekaran, “A framework of voronoi diagram for planning multiple paths in free space,” *Journal of Experimental and Theoretical Artificial Intelligence*, vol. 25, no. 4, pp. 457–475, 2013.
- [14] Y. Bar-Shalom, P. K. Willett, and X. Tian, *Tracking and Data Fusion*. YBS publishing, 2011.
- [15] M. Barbier, C. Laugier, O. Simonin, and J. Ibanez-Guzman, “Classification of drivers manoeuvre for road intersection crossing with synthetic and real data,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2017, pp. 224–230.
- [16] J. Barraquand, B. Langlois, and J.-C. Latombe, “Numerical potential field techniques for robot path planning,” *Transactions on Systems, Man, and Cybernetics*, vol. 22, no. 2, pp. 224–241, 1992.
- [17] P. Bender, O. S. Tas, J. Ziegler, and C. Stiller, “The combinatorial aspect of motion planning: Maneuver variants in structured environments,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2015, pp. 1386–1392.
- [18] H. Berndt and K. Dietmayer, “Driver intention inference with vehicle onboard sensors,” in *International Conference on Vehicular Electronics and Safety*, IEEE, 2009, pp. 102–107.
- [19] S. Bhattacharya, V. Kumar, and M. Likhachev, “Search-based path planning with homotopy class constraints,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2010.
- [20] S. Bhattacharya, M. Likhachev, and V. Kumar, “Identification and representation of homotopy classes of trajectories for search-based path planning in 3d,” in *Robotics: Science and Systems (RSS)*, 2011, pp. 9–16.
- [21] S. Bhattacharya, M. Likhachev, and V. Kumar, “Topological constraints in search-based robot path planning,” *Autonomous Robots*, vol. 33, no. 3, pp. 273–290, 2012.
- [22] R. P. Bhattacharyya, D. J. Phillips, B. Wulfe, J. Morton, A. Kuefler, and M. J. Kochenderfer, “Multi-agent imitation learning for driving simulation,” *arXiv:1803.01044*, 2018.
- [23] S. Bitzer, “The UKF exposed: How it works, when it works and when it’s better to sample,” doi:10.5281/zenodo.44386, 2016.
- [24] S. Brechtel, T. Gindele, and R. Dillmann, “Probabilistic decision-making under uncertainty for autonomous driving using continuous POMDPs,” in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2014, pp. 392–399.
- [25] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using RNN encoder-decoder for statistical machine translation,” *arXiv:1406.1078*, 2014.

- [26] J. Colyar and J. Halkias, “US highway 101 dataset,” Federal Highway Administration, Technical Report FHWA-HRT-07-030, 2007.
- [27] H. Cui, V. Radosavljevic, F.-C. Chou, T.-H. Lin, T. Nguyen, T.-K. Huang, J. Schneider, and N. Djuric, “Multimodal trajectory predictions for autonomous driving using deep convolutional networks,” *arXiv:1809.10732*, 2018.
- [28] A. G. Cunningham, E. Galceran, R. M. Eustice, and E. Olson, “MPDM: Multi-policy decision-making in dynamic, uncertain environments for autonomous driving,” in *International Conference on Robotics and Automation (ICRA)*, IEEE, 2015, pp. 1670–1677.
- [29] J. D’Onfro. ”I hate them”: Locals reportedly are frustrated with Alphabet’s self-driving cars, [Online]. Available: <https://www.cnbc.com/2018/08/28/locals-reportedly-frustrated-with-alphabets-waymo-self-driving-cars.html> (visited on 10/09/2019).
- [30] A. Darwiche, *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press, 2009.
- [31] D. J. Demyen and M. Buro, “Efficient triangulation-based pathfinding,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 6, 2006, pp. 942–947.
- [32] N. Deo and M. M. Trivedi, “Convolutional social pooling for vehicle trajectory prediction,” in *Conference on Computer Vision and Pattern Recognition Workshops*, IEEE, 2018, pp. 1549–15498.
- [33] N. Deo and M. M. Trivedi, “Multi-modal trajectory prediction of surrounding vehicles with maneuver based LSTMs,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2018, pp. 1179–1184.
- [34] F. Diehl, T. Brunner, M. T. Le, and A. Knoll, “Graph neural networks for modelling traffic participant interaction,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2019, pp. 695–701.
- [35] W. Ding and S. Shen, “Online vehicle trajectory prediction using policy anticipation network and optimization-based context reasoning,” in *International Conference on Robotics and Automation (ICRA)*, IEEE, 2019, pp. 9610–9616.
- [36] N. Djuric, V. Radosavljevic, H. Cui, T. Nguyen, F.-C. Chou, T.-H. Lin, and J. Schneider, “Short-term motion prediction of traffic actors for autonomous driving using deep convolutional networks,” *arXiv:1808.05819*, 2018.
- [37] A. Doucet and A. M. Johansen, “A tutorial on particle filtering and smoothing: Fifteen years later,” *Handbook of Nonlinear Filtering*, vol. 12, no. 656, p. 3, 2009.
- [38] A. von Eichhorn, M. Werling, P. Zahn, and D. Schramm, “Maneuver prediction at intersections using cost-to-go gradients,” in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2013, pp. 112–117.

- [39] J. F. Fisac, A. Bajcsy, S. L. Herbert, D. Fridovich-Keil, S. Wang, C. J. Tomlin, and A. D. Dragan, “Probabilistically safe robot planning with confidence-based human predictions,” *arXiv:1806.00109*, 2018.
- [40] J. F. Fisac, E. Bronstein, E. Stefansson, D. Sadigh, S. S. Sastry, and A. D. Dragan, “Hierarchical game-theoretic planning for autonomous vehicles,” *arXiv:1810.05766*, 2018.
- [41] E. Galceran, A. G. Cunningham, R. M. Eustice, and E. Olson, “Multipolicy decision-making for autonomous driving via changepoint-based behavior prediction,” in *Robotics: Science and Systems (RSS)*, 2015, p. 2.
- [42] Z. Ghahramani and M. I. Jordan, “Supervised learning from incomplete data via an EM approach,” in *Advances in Neural Information Processing Systems (NIPS)*, 1994, pp. 120–127.
- [43] Z. Ghahramani and M. I. Jordan, “Learning from incomplete data,” *Technical Report AI Lab Memo No. 1509, CBCL Paper No. 108, MIT AI Lab*, 1995.
- [44] T. Gindele, S. Brechtel, and R. Dillmann, “Learning context sensitive behavior models from observations for predicting traffic situations,” in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2013, pp. 1764–1771.
- [45] T. Gindele, S. Brechtel, and R. Dillmann, “A probabilistic model for estimating driver behaviors and vehicle trajectories in traffic environments,” in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2010, pp. 1625–1631.
- [46] D. S. Gonzalez, J. S. Dibangoye, and C. Laugier, “High-speed highway scene prediction based on driver models learned from demonstrations,” in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2016, pp. 149–155.
- [47] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in Neural Information Processing Systems (NIPS)*, 2014, pp. 2672–2680.
- [48] A. Gopnik and H. M. Wellman, “The theory theory,” in *Mapping the mind: Domain specificity in cognition and culture*, 1994, p. 257.
- [49] T. Gu, J. M. Dolan, and J. Lee, “Automated tactical maneuver discovery, reasoning and trajectory planning for autonomous driving,” in *International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2016, pp. 5474–5480.
- [50] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, “Social GAN: Socially acceptable trajectories with generative adversarial networks,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2018, pp. 2255–2264.
- [51] F. Gustafsson, “Particle filter theory and practice with positioning applications,” *Aerospace and Electronic Systems Magazine*, vol. 25, no. 7, pp. 53–82, 2010.

- [52] B. Gutjahr, C. Pek, L. Gröll, and M. Werling, “Efficient trajectory optimization for vehicles using quadratic programming,” *Automatisierungstechnik*, 2016.
- [53] J. Halkias and J. Colyar, “Interstate 80 freeway dataset,” Federal Highway Administration, US Department of Transportation, 2006.
- [54] J. Ho and S. Ermon, “Generative adversarial imitation learning,” *arXiv:1606.03476*, 2016.
- [55] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [56] A. Hochstädter, P. Zahn, and K. Breuer, “A comprehensive driver model with application to traffic simulation and driving simulators,” in *Human-Centered Transportation Simulation Conference*, 2001.
- [57] S. Hoermann, M. Bach, and K. Dietmayer, “Dynamic occupancy grid prediction for urban autonomous driving: A deep learning approach with fully automatic labeling,” *arXiv:1705.08781*, 2017.
- [58] A. Houenou, P. Bonnifait, V. Cherfaoui, and W. Yao, “Vehicle trajectory prediction based on motion model and maneuver recognition,” in *International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2013, pp. 4363–4369.
- [59] D. Howard and D. Dai, “Public perceptions of self-driving cars: The case of Berkeley, California,” *Transportation Research Board 93rd Annual Meeting*, vol. 14, no. 4502, pp. 1–16, 2014.
- [60] Y. Hu, W. Zhan, L. Sun, and M. Tomizuka, “Multi-modal probabilistic prediction of interactive behavior via an interpretable model,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2019, pp. 557–563.
- [61] C. Hubmann, “Belief state planning for autonomous driving: Planning with interaction, uncertain prediction and uncertain perception,” PhD thesis, Karlsruhe Institute of Technology, 2019.
- [62] S. Julier and J. Uhlmann, “Unscented filtering and nonlinear estimation,” *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, 2004.
- [63] S. Kammel, J. Ziegler, B. Pitzer, M. Werling, T. Gindele, D. Jagzent, J. Schröder, M. Thuy, M. Goebel, F. v. Hundelshausen, O. Pink, C. Frese, and C. Stiller, “Team AnnieWAY’s autonomous system for the 2007 DARPA urban challenge,” *Journal of Field Robotics*, vol. 25, no. 9, pp. 615–639, 2008.
- [64] R. Kesten, M. Usman, J. Houston, T. Pandya, K. Nadhamuni, A. Ferreira, M. Yuan, B. Low, A. Jain, P. Ondruska, S. Omari, S. Shah, A. Kulkarni, A. Kazakova, C. Tao, L. Platinsky, W. Jiang, and V. Shet. Lyft Level 5 AV Dataset, [Online]. Available: <https://level5.lyft.com/dataset/> (visited on 07/23/2020).
- [65] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv:1412.6980*, 2014.

- [66] D. Klimenko, J. Song, and H. Kurniawati, “TAPIR: A software toolkit for approximating and adapting POMDP solutions online,” in *Australasian Conference on Robotics and Automation*, vol. 24, 2014.
- [67] S. Klingelschmitt, M. Platho, H.-M. Gross, V. Willert, and J. Eggert, “Combining behavior and situation information for reliably estimating multiple intentions,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2014, pp. 388–393.
- [68] J. Kong, M. Pfeiffer, G. Schildbach, and F. Borrelli, “Kinematic and dynamic vehicle models for autonomous driving control design,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2015, pp. 1094–1099.
- [69] M. Koschi and M. Althoff, “SPOT: A tool for set-based prediction of traffic participants,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2017, pp. 1686–1693.
- [70] R. Krajewski, J. Bock, L. Kloeker, and L. Eckstein, “The highD dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems,” in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2018, pp. 2118–2125.
- [71] D. Krajzewicz, “Traffic simulation with SUMO – simulation of urban mobility,” in *Fundamentals of Traffic Simulation*, Springer, 2010, pp. 269–293.
- [72] S. Krauß, P. Wagner, and C. Gawron, “Metastable states in a microscopic model of traffic flow,” *Physical Review E*, vol. 55, no. 5, p. 5597, 1997.
- [73] M. Kuderer, C. Sprunk, H. Kretschmar, and W. Burgard, “Online generation of homotopically distinct navigation paths,” in *International Conference on Robotics and Automation (ICRA)*, IEEE, 2014, pp. 6462–6467.
- [74] A. Kuefler, J. Morton, T. Wheeler, and M. Kochenderfer, “Imitating driver behavior with generative adversarial networks,” *arXiv:1701.06699*, 2017.
- [75] P. Kumar, M. Perrollaz, S. Lefevre, and C. Laugier, “Learning-based approach for online lane change intention prediction,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2013, pp. 797–802.
- [76] H. Kurniawati and V. Yadav, “An online POMDP solver for uncertainty planning in dynamic environment,” in *International Symposium of Robotics Research*, Springer, 2013, pp. 611–629.
- [77] J.-C. Latombe, *Robot Motion Planning*. Springer Science & Business Media, 2012, vol. 124.
- [78] S. M. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.
- [79] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. S. Torr, and M. Chandraker, “DESIRE: Distant future prediction in dynamic scenes with interacting agents,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2017, pp. 2165–2174.



- [80] S. Lefèvre, C. Laugier, and J. Ibañez-Guzmán, “Risk assessment at road intersections: Comparing intention and expectation,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2012, pp. 165–171.
- [81] S. Lefèvre, D. Vasquez, and C. Laugier, “A survey on motion prediction and risk assessment for intelligent vehicles,” *Robomech Journal*, vol. 1, no. 1, pp. 1–14, 2014.
- [82] D. Lenz, F. Diehl, M. T. Le, and A. Knoll, “Deep neural networks for markovian interactive scene prediction in highway scenarios,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2017, pp. 685–692.
- [83] D. Lenz, T. Kessler, and A. Knoll, “Tactical cooperative planning for autonomous highway driving using monte-carlo tree search,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2016, pp. 447–453.
- [84] T. Li, S. Sun, T. P. Sattar, and J. M. Corchado, “Fight sample degeneracy and impoverishment in particle filters: A review of intelligent approaches,” *Expert Systems with Applications*, vol. 41, no. 8, pp. 3944–3954, 2014.
- [85] X.-R. Li and Y. Bar-Shalom, “Multiple-model estimation with variable structure,” *Transactions on Automatic Control*, vol. 41, no. 4, p. 16, 1996.
- [86] M. Liebner, M. Baumann, F. Klanner, and C. Stiller, “Driver intent inference at urban intersections using the intelligent driver model,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2012, pp. 1162–1167.
- [87] W. Liu, S.-W. Kim, S. Pendleton, and M. H. Ang, “Situation-aware decision making for autonomous driving on urban road using online POMDP,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2015, pp. 1126–1133.
- [88] M.-T. Luong, H. Pham, and C. D. Manning, “Effective approaches to attention-based neural machine translation,” *arXiv:1508.04025*, 2015.
- [89] W.-C. Ma, D.-A. Huang, N. Lee, and K. M. Kitani, “Forecasting interactive dynamics of pedestrians with fictitious play,” in *Conference on Computer Vision and Pattern Recognition*, IEEE, 2017, pp. 4636–4644.
- [90] J. Mänttari and J. Folkesson, “Incorporating uncertainty in predicting vehicle maneuvers at intersections with complex interactions,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2019, pp. 2156–2163.
- [91] G. Mesnil, X. He, L. Deng, and Y. Bengio, “Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding,” *Interspeech*, pp. 3771–3775, 2013.
- [92] K. Messaoud, I. Yahiaoui, A. Verroust-Blondet, and F. Nashashibi, “Non-local social pooling for vehicle trajectory prediction,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2019, pp. 975–980.
- [93] V. Mnih, N. Heess, and A. Graves, “Recurrent models of visual attention,” in *Advances in Neural Information Processing Systems (NIPS)*, 2014, pp. 2204–2212.

- [94] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Haehnel, T. Hilden, G. Hoffmann, B. Huhnke, D. Johnston, S. Klumpp, D. Langer, A. Levandowski, J. Levinson, J. Marcil, D. Orenstein, J. Paefgen, I. Penny, A. Petrovskaya, M. Pflueger, G. Stanek, D. Stavens, A. Vogt, and S. Thrun, “Junior: The stanford entry in the urban challenge,” *Journal of Field Robotics*, vol. 25, no. 9, pp. 569–597, 2008.
- [95] K. P. Murphy, “Dynamic bayesian networks: Representation, inference and learning,” PhD thesis, University of California, Berkeley, 2002.
- [96] J. Nilsson, M. Brannstrom, J. Fredriksson, and E. Coelingh, “Longitudinal and lateral control for automated yielding maneuvers,” *Transactions on Intelligent Transportation Systems (TITS)*, vol. 17, no. 5, pp. 1404–1414, 2016.
- [97] J. Nocedal and S. J. Wright, *Numerical Optimization*. Springer, 2006.
- [98] S. H. Park, B. Kim, C. M. Kang, C. C. Chung, and J. W. Choi, “Sequence-to-sequence prediction of vehicle trajectory via LSTM encoder-decoder architecture,” *arXiv:1802.06338*, 2018.
- [99] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “PyTorch: An imperative style, high-performance deep learning library,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2019, pp. 8024–8035.
- [100] C. Paxton, V. Raman, G. D. Hager, and M. Kobilarov, “Combining neural networks and tree search for task and motion planning in challenging environments,” in *International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2017, pp. 6059–6066.
- [101] D. Petrich, T. Dang, G. Breuel, and C. Stiller, “Assessing map-based maneuver hypotheses using probabilistic methods and evidence theory,” in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2014, pp. 995–1002.
- [102] D. J. Phillips, T. A. Wheeler, and M. J. Kochenderfer, “Generalizable intention prediction of human drivers at intersections,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2017, pp. 1665–1670.
- [103] M. Platho, H.-M. Gros, and J. Eggert, “Predicting velocity profiles of road users at intersections using configurations,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2013, pp. 945–951.
- [104] M. Platho, H.-M. Gros, and J. Eggert, “Learning driving situations and behavior models from data,” in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2013, pp. 276–281.
- [105] E. A. I. Pool, J. F. P. Kooij, and D. M. Gavrila, “Context-based cyclist path prediction using recurrent neural networks,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2019, pp. 824–830.

- [106] X. Qian, F. Althé, P. Bender, C. Stiller, and A. De La Fortelle, “Optimal trajectory planning for autonomous driving integrating logical constraints: A MIQP perspective,” in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2016, pp. 205–210.
- [107] S. Quinlan and O. Khatib, “Elastic bands: Connecting path planning and control,” in *International Conference on Robotics and Automation*, IEEE, 1993, pp. 802–807.
- [108] E. Rehder and H. Kloeden, “Goal-directed pedestrian prediction,” in *International Conference on Computer Vision Workshops*, 2015, pp. 50–58.
- [109] E. Rehder, F. Wirth, M. Lauer, and C. Stiller, “Pedestrian prediction by planning using deep neural networks,” *arXiv:1706.05904*, 2017.
- [110] S. Ross and J. A. Bagnell, “Efficient reductions for imitation learning,” in *International Conference on Artificial Intelligence and Statistics*, 2010, pp. 661–668.
- [111] S. Ross, G. J. Gordon, and J. A. Bagnell, “A reduction of imitation learning and structured prediction to no-regret online learning,” in *International Conference on Artificial Intelligence and Statistics*, 2011, pp. 627–635.
- [112] A. Sadeghian, V. Kosaraju, A. Sadeghian, N. Hirose, H. Rezaatofghi, and S. Savarese, “SoPhie: An attentive GAN for predicting paths compliant to social and physical constraints,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2019, pp. 1349–1358.
- [113] A. Sadeghian, F. Legros, M. Voisin, R. Vesel, A. Alahi, and S. Savarese, “Car-net: Clairvoyant attentive recurrent network,” in *European Conference on Computer Vision (ECCV)*, 2018, pp. 151–167.
- [114] D. Sadigh, S. S. Sastry, S. A. Seshia, and A. Dragan, “Information gathering actions over human internal state,” in *International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2016, pp. 66–73.
- [115] D. Sadigh, S. Sastry, S. A. Seshia, and A. D. Dragan, “Planning for autonomous cars that leverage effects on human actions,” in *Robotics: Science and Systems (RSS)*, vol. 2, Robotics: Science and Systems Foundation, 2016.
- [116] A. Sarkar, K. Czarnecki, M. Angus, C. Li, and S. Waslander, “Trajectory prediction of traffic agents at urban intersections through learned interactions,” in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2017, pp. 1–8.
- [117] J. Schlechtriemen, A. Wedel, J. Hillenbrand, G. Breuel, and K.-D. Kuhnert, “A lane change detection approach using feature ranking with maximized predictive power,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2014, pp. 108–114.
- [118] J. Schmidhuber. Highlights of Robot Car History, [Online]. Available: <http://people.idsia.ch/~juergen/robotcars.html> (visited on 09/12/2019).

- [119] R. Schubert, E. Richter, and G. Wanielik, “Comparison and evaluation of advanced motion models for vehicle tracking,” in *International Conference on Information Fusion*, IEEE, 2008, pp. 1–6.
- [120] V. Sezer, T. Bandyopadhyay, D. Rus, E. Frazzoli, and D. Hsu, “Towards autonomous navigation of unsignalized intersections under uncertainty of human driver intent,” in *International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2015, pp. 3578–3585.
- [121] W. Si, T. Wei, and C. Liu, “AGen: Adaptable generative prediction networks for autonomous driving,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2019, pp. 281–286.
- [122] S. Singh, “Critical reasons for crashes investigated in the national motor vehicle crash causation survey,” National Motor Vehicle Crash Causation Survey DOT HS 812 115, 2015.
- [123] J. C. Smith and Z. C. Taskin, “A tutorial guide to mixed-integer programming models and solution techniques,” *Optimization in Medicine and Biology*, pp. 521–548, 2008.
- [124] R. Socher, “Recursive deep learning for natural language processing and computer vision,” PhD thesis, Stanford University, 2014.
- [125] K. Sohn, H. Lee, and X. Yan, “Learning structured output representation using deep conditional generative models,” in *Advances in Neural Information Processing Systems (NIPS)*, 2015, pp. 3483–3491.
- [126] J. Song, H. Ren, D. Sadigh, and S. Ermon, “Multi-agent generative adversarial imitation learning,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018, pp. 7461–7472.
- [127] W. Song, G. Xiong, and H. Chen, “Intention-aware autonomous driving decision-making in an uncontrolled intersection,” *Mathematical Problems in Engineering*, vol. 2016, pp. 1–15, 2016.
- [128] S. Steyer, G. Tanzmeister, and D. Wollherr, “Object tracking based on evidential dynamic occupancy grids in urban environments,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2017, pp. 1064–1070.
- [129] S. Steyer, C. Lenk, D. Kellner, G. Tanzmeister, and D. Wollherr, “Grid-based object tracking with nonlinear dynamic state and shape estimation,” *Transactions on Intelligent Transportation Systems (TITS)*, vol. 21, no. 7, pp. 2874–2893, 2019.
- [130] S. Steyer, G. Tanzmeister, and D. Wollherr, “Grid-based environment estimation using evidential mapping and particle tracking,” *Transactions on Intelligent Vehicles (TIV)*, vol. 3, no. 3, pp. 384–396, 2018.
- [131] T. Streubel and K. H. Hoffmann, “Prediction of driver intended path at intersections,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2014, pp. 134–139.

- [132] L. Sun, W. Zhan, C.-Y. Chan, and M. Tomizuka, “Behavior planning of autonomous cars with social perception,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2019, pp. 207–213.
- [133] L. Sun, W. Zhan, M. Tomizuka, and A. D. Dragan, “Courteous autonomous cars,” *arXiv:1808.02633*, 2018.
- [134] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, *et al.*, “Scalability in perception for autonomous driving: Waymo open dataset,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 2446–2454.
- [135] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” in *Advances in Neural Information Processing Systems (NIPS)*, 2014, pp. 3104–3112.
- [136] The Robotics Institute at Carnegie Mellon University. Navlab: The Carnegie Mellon University Navigation Laboratory, [Online]. Available: <https://www.cs.cmu.edu/afs/cs/project/alv/www/index.html> (visited on 09/12/2019).
- [137] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT press, 2005.
- [138] R. S. Tomar, S. Verma, and G. S. Tomar, “Prediction of lane change trajectories through neural network,” in *International Conference on Computational Intelligence and Communication Networks (CICN)*, IEEE, 2010, pp. 249–253.
- [139] Q. Tran and J. Firl, “A probabilistic discriminative approach for situation recognition in traffic scenarios,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2012, pp. 147–152.
- [140] Q. Tran and J. Firl, “Online maneuver recognition and multimodal trajectory prediction for intersection assistance using non-parametric regression,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2014, pp. 918–923.
- [141] P. Trautman and A. Krause, “Unfreezing the robot: Navigation in dense, interacting crowds,” in *International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2010, pp. 797–803.
- [142] M. Treiber, A. Hennecke, and D. Helbing, “Congested traffic states in empirical observations and microscopic simulations,” *Physical Review E*, vol. 62, no. 2, p. 1805, 2000.
- [143] M. Tsogas, X. Dai, G. Thomaidis, P. Lytrivis, and A. Amditis, “Detection of maneuvers using evidence theory,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2008, pp. 126–131.
- [144] C. Urmson. How a driverless car sees the road, [Online]. Available: <https://www.youtube.com/watch?v=tiwVMrTLUWg&t=8m29s> (visited on 10/09/2019).

- [145] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. N. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. M. Howard, S. Kolski, A. Kelly, M. Likhachev, M. McNaughton, N. Miller, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski, B. Salesky, Y.-W. Seo, S. Singh, J. Snider, A. Stentz, W. “. Whittaker, Z. Wolkowicki, J. Zigar, H. Bae, T. Brown, D. Demitrish, B. Litkouhi, J. Nickolaou, V. Sadekar, W. Zhang, J. Struble, M. Taylor, M. Darms, and D. Ferguson, “Autonomous driving in urban environments: Boss and the urban challenge,” *Journal of Field Robotics*, vol. 25, no. 8, pp. 425–466, 2008.
- [146] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in Neural Information Processing Systems (NIPS)*, 2017, pp. 5998–6008.
- [147] A. Vemula, K. Muelling, and J. Oh, “Social attention: Modeling attention in human crowds,” *arXiv:1710.04689*, 2017.
- [148] A. Venkatraman, B. Boots, M. Hebert, and J. A. Bagnell, “Data as demonstrator with applications to system identification,” in *Advances in Neural Information Processing Systems (NIPS) Workshop: Autonomously Learning Robots*, 2014.
- [149] VIRESS Simulationstechnologie GmbH. OpenDRIVE, [Online]. Available: <http://opendrive.org/> (visited on 09/13/2018).
- [150] E. A. Wan and R. Van Der Merwe, “The unscented kalman filter for nonlinear estimation,” in *Adaptive Systems for Signal Processing, Communications, and Control Symposium*, IEEE, 2000, pp. 153–158.
- [151] Waymo LLC. Waymo Open Dataset, [Online]. Available: <https://waymo.com/open/> (visited on 10/04/2019).
- [152] M. Werling, J. Ziegler, S. Kammel, and S. Thrun, “Optimal trajectory generation for dynamic street scenarios in a frenet frame,” in *International Conference on Robotics and Automation (ICRA)*, IEEE, 2010, pp. 987–993.
- [153] T. A. Wheeler, P. Robbel, and M. J. Kochenderfer, “Analysis of microscopic behavior models for probabilistic modeling of driver behavior,” in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2016, pp. 1604–1609.
- [154] J. Wiest, M. Höffken, U. Kreßel, and K. Dietmayer, “Probabilistic trajectory prediction with gaussian mixture models,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2012, pp. 141–146.
- [155] H. Williams. The Man Who Gave Cars Eyes: Ernst Dickmanns, [Online]. Available: <https://www.lifehacker.com.au/2016/02/creator-of-the-worlds-first-self-driving-cars-ernst-dickmanns/> (visited on 09/12/2019).
- [156] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, “A comprehensive survey on graph neural networks,” *arXiv:1901.00596*, 2019.

- [157] W. Zhan, L. Sun, D. Wang, H. Shi, A. Clausse, M. Naumann, J. Kümmerle, H. Königshof, C. Stiller, A. de La Fortelle, and M. Tomizuka, “INTERACTION dataset: An INTERnational, Adversarial and Cooperative moTION dataset in interactive driving scenarios with semantic maps,” *arXiv:1910.03088*, 2019.
- [158] T. Zhao, Y. Xu, M. Monfort, W. Choi, C. Baker, Y. Zhao, Y. Wang, and Y. N. Wu, “Multi-agent tensor fusion for contextual trajectory prediction,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2019, pp. 12 126–12 134.
- [159] B. D. Ziebart, N. Ratliff, G. Gallagher, C. Mertz, K. Peterson, J. A. Bagnell, M. Hebert, A. K. Dey, and S. Srinivasa, “Planning-based prediction for pedestrians,” in *International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2009, pp. 3931–3936.
- [160] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, “Maximum entropy inverse reinforcement learning.,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 8, 2008, pp. 1433–1438.

## Own Publications

- [161] C. Hubmann, N. Quetschlich, J. Schulz, J. Bernhard, D. Althoff, and C. Stiller, “A POMDP maneuver planner for occlusions in urban scenarios,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2019, pp. 2172–2179.
- [162] C. Hubmann, J. Schulz, M. Becker, D. Althoff, and C. Stiller, “Automated driving in uncertain environments: Planning with interaction and uncertain maneuver prediction,” *Transactions on Intelligent Vehicles (TIV)*, vol. 3, no. 1, pp. 5–17, 2018.
- [163] C. Hubmann, J. Schulz, G. Xu, D. Althoff, and C. Stiller, “A belief state planner for interactive merge maneuvers in congested traffic,” in *International Conference On Intelligent Transportation Systems (ITSC)*, IEEE, 2018, pp. 1617–1624.
- [164] F. Kuhnt, J. Schulz, T. Schamm, and J. M. Zöllner, “Understanding interactions between traffic participants based on learned behaviors,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2016, pp. 1271–1278.
- [165] J. Schulz, K. Hirsenkorn, J. Löchner, M. Werling, and D. Burschka, “Estimation of collective maneuvers through cooperative multi-agent planning,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2017, pp. 624–631.
- [166] J. Schulz, C. Hubmann, J. Löchner, and D. Burschka, “Interaction-aware probabilistic behavior prediction in urban environments,” in *International Conference On Intelligent Robots and Systems (IROS)*, IEEE, 2018, pp. 3999–4006.
- [167] J. Schulz, C. Hubmann, J. Löchner, and D. Burschka, “Multiple model unscented kalman filtering in dynamic bayesian networks for intention estimation and trajectory prediction,” in *International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2018, pp. 1467–1474.
- [168] J. Schulz, C. Hubmann, N. Morin, J. Löchner, and D. Burschka, “Learning interaction-aware probabilistic driver behavior models from urban scenarios,” in *Intelligent Vehicles Symposium (IV)*, IEEE, 2019, pp. 1326–1333.
- [169] G. Swamy, J. Schulz, R. Choudhury, D. Hadfield-Menell, and A. Dragan, “On the utility of model learning in HRI,” *arXiv:1901.01291*, 2020.