Planning for Automated Vehicles with respect to Uncertainties and Incomplete Knowledge

Zur Erlangung des akademischen Grades eines
Doktors der Ingenieurwissenschaften

Dr.-Ing.

von der KIT-Fakultät für
Wirtschaftswissenschaften
des Karlsruher Instituts für Technologie (KIT)

genehmigte

DISSERTATION

von

M.Sc. Philip Benjamin Schörner
geb. in Reutlingen

Tag der mündlichen Prüfung:	09.04.2025
Hauptreferent:	Prof. Dr.-Ing. J. Marius Zöllner
Korreferent:	Prof. Dr.-Ing. Sören Hohmann

Kurzfassung

Die technologische Reife vollautomatisierter Fahrzeuge hat in den letzten Jahren stetig zugenommen und sie stehen kurz vor der Markteinführung. Die wiederholte Verzögerung ihres Einsatzes im Straßenverkehr zeigt jedoch, dass es noch viele Herausforderungen zu lösen gibt. Dabei steht der großflächige Einsatz von automatisierten Fahrzeugen im öffentlichen Raum vor den gleichen Herausforderungen wie der Mensch im alltäglichen Verkehr. Menschlich Fahrer*innen sind Expert*innen darin Situationen schnell zu erfassen. Dennoch hätten viele Unfälle durch ein besseres Bewusstsein für die aktuelle Situation und durch ein vorausschauenderes Verhalten vermieden werden können. Auch für den Einsatz vollautomatisierter Fahrzeuge, die sich sicher in den alltäglichen Verkehr einfügen sollen, ist ein tiefes Verständnis der Situation und ein entsprechendes vorausschauendes und sicheres Verhalten erforderlich.

In dieser Dissertation wird eine Lösung für dieses Problem vorgeschlagen. Die Lösung ist in drei Teile unterteilt. Zunächst wird gezeigt, wie ein umfassendes Situationsbewusstsein erreicht werden kann. Dann wird erforscht, wie Wissenslücken geschlossen oder quantifiziert werden können, indem ein Belief des aktuellen Zustands der Umgebung aufgebaut wird. Schließlich werden Strategien für vorausschauendes und risikobewusstes Verhalten und die Trajektorienplanung auf Basis des zuvor aufgebauten Beliefs vorgestellt. Der Belief spiegelt wider, wie sicher man sich über die Genauigkeit der verfügbaren Informationen ist und, was noch wichtiger ist, über die Bereiche, für die keine Informationen bereitgestellt werden.

Ein umfassendes Situationsbewusstsein wird durch die Analyse der Komponenten der automatisierten Fahrfunktionen und die Offenlegung von Unsicherheiten und potenziellen Wissenslücken erreicht. Anschließend wird ein Umgebungsmodell vorgestellt, das die Verkehrssituation vollständig abbildet, einschließlich sichtbarer und verdeckter Verkehrsteilnehmenden und der entsprechenden Unsicherheiten. Dabei wird der Fokus auf Unsicherheiten gelegt, die nicht direkt gemessen werden können. Ein hohes Maß an Szenenverständnis ist erforderlich, um hier Rückschlüsse zu ziehen. Als besonders relevant werden die unbekannten Intentionen der anderen Verkehrsteilnehmenden und verdeckte Verkehrsteilnehmende identifiziert.

Der Belief über den aktuellen Zustand der Umgebung wird dann erweitert und konkretisiert. Hierfür werden neuartige Methode zur Schätzung von Intentionen, ein zweigeteilter Ansatz für das Tracking verdeckter Bereiche, ein Ansatz zur Inferenz der Existenz von Verkehrsteilnehmenden in eben diesen verdeckten Bereichen sowie ein gelerntes Prädiktionsmodell, das für Verkehrsszenen mit verdeckten Bereichen geeignet ist, vorgestellt. Die vorgestellten Ansätze nutzen Kontextwissen, um die Situation genau einzuschätzen, wobei explizit Informationen über sichtbare und verdeckte Verkehrsteilnehmende einbezogen werden.

Der letzte Teil zeigt Strategien für eine sichere, risikobewusste und vorausschauende Planung von Manövern und Trajektorien auf der Grundlage des vertieften Beliefs. Dafür wird zunächst ein Manöverplaner auf der Basis von partiell beobachtbaren Markov-Entscheidungsprozessen (POMDP) präsentiert. Durch die Verwendung eines generischen Sichtfelds können vorausschauende Entscheidungen an Kreuzungen mit Verdeckungen getroffen werden, indem präzise Annahmen über zukünftige Beobachtungen gemacht werden können. Der anschließende Trajektorienplaner bestimmt risikobewusste Trajektorien. Das Risiko aus verschiedenen Quellen wie multimodalen Prädiktionen und verdeckten Bereichen wird über die Zeit zu sogenannten "risk maps over time" kombiniert. Diese quantifizieren die Risiken räumlich und zeitlich. Schließlich wird eine neuartige Methode für die sichere Anwendung von maschinellem Lernen in sicherheitskritischen Komponenten wie der Trajektorienplanung gezeigt, indem maschinelles Lernen zur Generierung von initialen Trajektorien (Samples) für einen Sampling-basierten Bewegungsplaner für dynamische Umgebungen eingesetzt wird.

Die Evaluierungen der Konzepte wurden mit mehreren Testfahrzeugen des FZI Forschungszentrum Informatik und Fahrzeugen von Projektpartnern oder der Simulation durchgeführt, um den Mehrwert der beschriebenen Methoden zu demonstrieren.

Abstract

The technological readiness of fully automated vehicles has steadily increased in recent years, and they are about to be launched on the market. However, the repeated delay in deploying fully automated vehicles in road traffic shows that there are still many challenges to solve. Thereby, the large-scale deployment of automated vehicles in public environments is confronted with the same challenges as humans in everyday traffic. Although human drivers are experts in understanding situations quickly and have a presentiment for dangerous situations, many accidents could have been avoided with better awareness of the current situation and more foresighted behavior. A deep understanding of the situation and the corresponding anticipatory and safe behavior is required to deploy fully automated vehicles where they safely flow with regular traffic.

This dissertation proposes a solution to the problem in three steps. First, it is shown how situational awareness can be achieved. Then, it is researched how gaps in knowledge can be closed or quantified by building a belief of the current state of the environment. Last, strategies for foresighted and risk-aware behavior and trajectory planning based on the afore-built belief are presented.

Thorough situational awareness is obtained by analyzing the components of the automated driving functions and revealing uncertainties and potential gaps in knowledge. Afterward, an environment model is proposed to fully represent the traffic situation, including visible and occluded traffic participants and the corresponding uncertainties. The focus is set on uncertainties that cannot be measured directly due to a lack of knowledge because these problematic aspects require a high level of scene understanding. The unknown intentions of other road users and occluded traffic participants are identified as particularly relevant.

The belief about the current state of the environment is then enriched using novel methods for estimating intentions, a dual approach for tracking occluded areas, an approach for inferring the existence of road users in precisely these occluded areas, and a learned prediction model suitable for traffic scenes with occluded areas. The belief reflects how certain one is about the accuracy of the available information and, even as important, of the parts where no information is provided. The presented approaches leverage context knowledge to accurately estimate the situation, explicitly including information about visible and occluded traffic participants.

The last section shows strategies for safe, risk-aware, and foresighted planning based on the enriched belief. Therefore, a maneuver planner based on Partially Observable Markov Decision Processes is proposed that anticipatory deals with occluded intersections through a generic field of view that allows reasoning about future observations. The subsequent trajectory planner determines risk-aware trajectories. The risk from various sources like multi-modal predictions and occluded areas is combined into risk maps over time

to quantify risks spatially and temporally. Finally, a novel method for the safe application of machine learning in safety-critical components like the trajectory planner is shown using machine learning to generate initial trajectories (samples) for a sampling-based motion planner for dynamic environments.

The evaluations of the concepts were performed with multiple test vehicles of the FZI Research Center for Information Technology and vehicles of project partners or the simulation to demonstrate the added value of the methods described.

Preface

The research leading to this work was conducted during my time as a scientific researcher in the Technical Cognitive Assistance Systems (TKS) department at the FZI Research Center for Information Technology (FZI) in Karlsruhe, Germany. I would like to thank everyone who supported me and made this work possible.

I sincerely thank Prof. Dr.-Ing. J. Marius Zöllner for the supervision of this thesis. The constructive discussions provided guidance, motivation, and freedom to pursue my ideas. The cooperation was always pleasant and productive, both as supervisor of the dissertation and as scientific director of TKS. I would like to thank Prof. Dr.-Ing. Sören Hohmann for acting as co-supervisor.

Many thanks to all my colleagues in the TKS and IDS departments at the FZI and the AIFB colleagues at the Karlsruher Institute for Technology (KIT) for the enjoyable working atmosphere and their support. Special thanks go to Jens Doll, Tobias Fleck, Karl Kurzer, Marc Zofka, Christian Hubschneider, Sven Ochs, and Daniel Grimm for the deep discussions, input, criticism, and impulses that contributed to this work.

Additionally, I thank all the students I supervised during their bachelor's or master's thesis or as a student assistant, particularly Lars Töttel, Mark Hüneberg, and Rupert Polley. I really enjoyed working with all of you and appreciate your dedication, effort, and constructive discussions.

Also, I would like to thank all project partners and, above all, the partners from the SafeADArchitect project, where a great part of this work was developed. The collaboration made it possible to evaluate and test the concepts in the overall system. The discussions also helped to open up other perspectives on the research questions.

The backbone that made this thesis possible is my family and all my friends who always supported me. Thereby, I especially thank my parents, my brother, and my sister. The largest thanks go to my wife and my children, who provided the necessary support, rest, distraction, and fun, especially in stressful times.

Grammarly was used to check spelling and grammar in this work.

Karlsruhe, April 2025 *Philip Schörner*

Contents

Acronyms and symbols

Acronyms

ABT	Adaptive Belief Tree
BNN	Bayesian Neural Network
CNN	Convolutional Neural Network
DBN	Dynamic Bayesian Network
FCN	Fully Convolutional Network
FOV	field of view
FZI	FZI Research Center for Information Technology
GNN	Graph Neural Network
GNSS	global navigation satellite system
HMM	Hidden Markov Model
IDM	Intelligent Driver Model proposed by [145]
IMU	inertial measurement unit
IoU	Intersection over Union
KIT	Karlsruher Institute for Technology
Lidar	light detection and ranging
MDP	Markov Decision Process
MLP	Multi-layer Perceptron
MSE	Mean Squared Error
NST	non-road-bound occluded area set tracking

OSM	Open Street Map
OST	oriented occluded area set tracking
POMDP	Partially Observable Markov Decision Process
PSO	Particle Swarm Optimization
Radar	radio detection and ranging
ReLU	Rectified Linear Unit
RMOT	risk maps over time
RNN	Recurrent Neural Network
ROI	region of interest
ROS	Robot Operating System
SLAM	Simultaneous Localization and Mapping
TAPIR	Toolkit for Approximating and Adapting POMDP Solutions in Real-Time

Variables and functions

P	probability
t	time
Δt	time interval between two discrete points in time
T_P	prediction and planning horizon
T_H	furthest point back in time, length of history
$\mathbb{S}$	state space
s	state of a single traffic participant
S	states of all traffic participants
s_r	state of regulatory element
$\mathcal{T}$	transition function
$\mathcal{Z}$	observation probabilities
$\mathbb{O}$	observation space
o	observation
$\mathbb{B}$	belief space
b	belief

$\mathbb{A}$	action space
a	action
π	policy
$\mathcal{R}$	reward
γ	discount factor
Π	initial probabilities
$\vec{v}$	velocity
v	speed
a	acceleration
θ	yaw angle around vertical axis, orientation
ψ	roll angle around longitudinal axis
ϕ	pitch angle around lateral axis
$\dot{\theta}$	yaw rate
I	intention
r	route
$\vec{\Gamma}$	trajectory
$\mathcal{D}$	development of a scene
R	risk
$\mathcal{U}$	set of non-road-bound occluded areas
u	element of $\mathcal{U}$
$\hat{\mathcal{U}}$	predicted set of non-road-bound occluded areas
$\hat{u}$	element of $\hat{\mathcal{U}}$
Q	set of oriented occluded areas
q	element of Q
$\hat{Q}$	predicted set of oriented occluded areas
$\hat{q}$	element of $\hat{Q}$
$\mathcal{L}$	loss

1 Introduction

Fully automated vehicles are about to be launched on the market. Technology readiness is steadily increasing and the first level 3 driving assistant systems are available for consumers for specific operational design domains [113]. These assistant systems promise to further increase the safety and efficiency of vehicles in everyday traffic. Driver assistance systems like emergency braking assistants or adaptive cruise control (ACC) significantly contribute to avoiding accidents. They are standard in many vehicles and are already mandatory in many countries. Although the first prototypes or robotaxis are tested in real-world traffic [152], the repeated delay of full-scale deployment of fully automated vehicles in road traffic shows [70] that fully automated driving still faces many challenges. Human traffic participants are experts in understanding situations quickly, including interpreting the current traffic rules, deriving the intentions of other road users, and having a presentiment for dangerous situations. From early childhood, humans learn to recognize connections to abstract situations and thereby establish links from unknown situations to known ones. This allows them to build up an awareness of the risks even in complicated and previously unknown situations. Additionally, and just as importantly, humans learn to know where they have no information and where they have to be extra careful. However, as statistics for traffic accidents in Germany show [138], many accidents still could have been avoided. The statistics show that even humans do not always fully understand the situation, do not react accordingly, or have a false risk awareness.

The large-scale deployment of automated vehicles in public environments faces the same challenges. It has to be shown that automated vehicles behave safely at all times for the homologation of automated driving functions and to obey the safety first principle [36]. Second, the vehicles have to be suitable for everyday use, meaning that they can flow with regular traffic and act in a way that humans can understand. This requires a deeper understanding of the situation followed by foresighted behavior planning and executing the behavior through safe and feasible trajectories.

This dissertation bridges the gap between understanding the scene and foresighted planning while showing how machine learning can be safely used in safety-critical components such as trajectory planning. First, it demonstrates how comprehensive situational awareness is achieved and how automated vehicles can extend their situational awareness to make informed and foresighted decisions. The focus is set on the unknown intentions of other traffic participants as well as occluded traffic participants. These aspects require a high level of scene understanding. In particular, the necessary information for making the correct decisions can not be measured directly and can only be gathered from context and background knowledge.

In return, the foresighted decisions based on a deeper understanding of the context will enable automated vehicles to blend in with everyday traffic and lead to safe and comprehensible behavior.

1.1 Problem Statement

Automated driving functions will not reach the performance level required for everyday traffic when relying on strict safety metrics that are designed too overcautious to make up for unquantified risks and uncertainties. These safety metrics like [135, 118] work fine for simple applications, where all information is available without errors. However, the metrics reach their limit when not all information is available and cannot properly handle uncertainties.

High-level behavior or intentions, for example, can not be measured but must be inferred from the traffic participants' motions [100, 64]. The estimation will always have a certain degree of uncertainty. Unlike humans, automated vehicles have no additional explicit communication, such as short gestures or head movements that unambiguously indicate the intention. There might be vehicle-to-vehicle communication between two automated vehicles, but the interaction with human drivers still needs to be solved.

The same applies to situations where occluded traffic participants can appear. Assuming that vehicles can approach from hidden lanes at any time or pedestrians crossing between two vehicles anywhere and anytime leads to overcautious behavior that might lead the automated vehicle not to move at all. These assumptions are, therefore, not applicable if not supported by a certain level of situational awareness. Current risk metrics and approaches often only rely on the information gathered at the current point in time [135].

Improved handling of these situations shall be achieved through a comprehensive situational awareness that results from accumulating all available online and offline context knowledge over a long time. The awareness shall also allow the assessment of the residual risks.

For this reason, the components of perception and the environment model must be investigated thoroughly and methods to accumulate context knowledge must be researched. This work focuses on the scenarios with uncertain intentions and occluded traffic participants that are mentioned above.

The gained awareness is to be used in the next step for foresighted behavior planning. Afterward, risk-aware trajectories that imminently determine the vehicle's motion must be found.

In total, this shall result in a comprehensive situational awareness based on which foresighted and anticipatory decisions are made and safe trajectories are planned. The following research questions are derived from the problem statement:

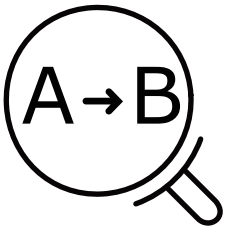
- How can a comprehensive awareness of traffic situations be achieved and missing knowledge about the situation be recognized?
- How can missing knowledge about the situation be enhanced or quantified by contextual information?
- How can the information about missing knowledge be considered in the planning process to drive foresighted and safely?

1.2 Dissertation Outline

The outline of this thesis is aligned with the research questions and follows the sequence of a decision-making process. Before presenting the first concept, the fundamentals and theoretical background are

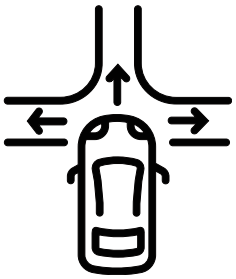
Comprehensive Situational Awareness (Chapter 3)

- Automated Driving (Sec. 3.1)
- Uncertainties (Sec. 3.2)
- Environment Model (Sec. 3.3)



Derivation of Knowledge and Building the Belief (Chapter 4)

- Intention Estimation through Situation Replay (Sec. 4.1)
- Tracking and Forecasting Occluded Areas (Sec. 4.2)
- Reasoning about Occluded Traffic Participants (Sec. 4.3)
- Learning Uncertainty-aware and Occlusion-aware Predictions (Sec. 4.4)



Anticipatory Maneuver and Trajectory Planning (Chapter 5)

- Foresighted Maneuver Planning (Sec. 5.1)
- Risk-aware Trajectory Planning (Sec. 5.2)
- Safe Application of Machine Learning for Trajectory Planning (Sec. 5.3)

Figure 1.1: Overview of the outline of the dissertation

given in Chapter 2. Related work is given for each contribution separately and represents the state of the art when the research was conducted. Fig. 1.1 depicts the chapter outline.

Chapter 3 lays the foundation for the latter decision-making process. It gives an overview of what information can be expected to be available. Additionally, the chapter shows what limitations of the perception capabilities need to be accounted for in behavior and trajectory planning. The goal is to obtain a comprehensive awareness of the situations that can occur in everyday traffic. The focus is set on uncertainties and gaps in the assessment of the environment. The gaps are, e.g., induced by incomplete knowledge due to sensor limitations and unavailable high-level information. The key insights of the chapter determined the choice of the approaches developed in this dissertation and presented in the following chapters.

Chapter 4 introduces approaches to obtain and improve the belief and assessment of the current state of the environment. This is achieved through the informed processing of available context knowledge and information. In particular, approaches for predicting the environment (Sec. 4.4), estimating the intentions of traffic participants (Sec. 4.1), inferring information about the presence of occluded road users (Sec. 4.3) and keeping track of areas where the potentially occluded traffic participants can be located in (Sec. 4.2).

Based on the derived knowledge and the belief created from it, Chapter 5 introduces planning and decision-making concepts for risk-aware and context-adjusted motion planning. Foresighted maneuvers are planned using a Partially Observable Markov Decision Process (Sec. 5.1). In Sec. 5.2, it is shown how to include multi-modal sources of risk in the trajectory planning process. Finally, Sec. 5.3 focuses on trajectory planning in dynamic environments using neural networks and gives a novel concept of checking the plausibility and validating the output of neural networks while combining the advantages of both, machine learning and model-based approaches.

The key takeaways of the corresponding chapter are briefly summarized at the end of each Chapter 3 to 5. Chapter 6 summarizes the dissertation and offers an outlook.

The content of this work has, in many cases, already been published. Therefore, this work may contain verbatim quotations from my previous work, especially [1, 2, 3, 4, 5, 6, 7, 8]. The relevant publications are referenced at the beginning of each chapter.

1.3 Contribution

The contributions of this thesis are listed in the following:

- Analysing sources of uncertainties, creating a holistic situational awareness, and providing an environmental model to represent and include arbitrary uncertainties, i.e., measurable uncertainties and not directly measurable uncertainties like occlusions or intentions (Chapter 3) [6, 7].
- Providing a novel approach to estimate the intentions of other traffic participants (Sec. 4.1) and to infer the existence of hidden traffic participants (Sec. 4.3) using the entire accumulated scene context, i.e., including the state of all traffic participants, static background knowledge, and regulatory elements [5]. It is achieved by generating multiple potential developments based on all possible intentions from a past time step. Afterward, the most likely development is successively assessed. Thus, not only a single road user's intention is determined, but the overall likelihood of the combinations of the intentions of all traffic participants is also determined.
- A dual approach for spatially and temporally tracking occlusions based on objects' movement patterns. The oriented occluded area set tracking tracks objects moving along a reference line like roads. Non-road-bound occluded area set tracking is designed for freely moving objects like pedestrians (Sec. 4.2)[7, 8].
- A novel concept for iterative prediction using convolutional neural networks improving accuracy and allowing the prediction of occluded traffic scenes. The network can be trained unsupervised, which is crucial when dealing with occlusions where no ground truth information is available (Sec. 4.4)[2].
- Incorporating a generic field of view in a probabilistic maneuver planner based on a Partially Observable Markov Decision Process to make assumptions about future observations in occlusion scenarios to achieve safer and more foresighted behavior (Sec. 5.1)[1].
- Introducing risk maps over time, a representation for accumulating arbitrary risks from different sources in a time series of two-dimensional grids and using them to achieve risk-aware trajectory planning (Sec. 5.2)[7, 8].
- Presenting the first approach to combine a machine learning-based and rule-based approach for trajectory planning in dynamic environments. Neural networks initialize a sampling-based motion planner based on particle swarm optimization (Sec. 5.3)[4].

2 Fundamentals

This chapter covers the necessary background information to comprehend the remaining thesis better and briefly describes the theory behind the methods used. For deeper explanations, the reader is referred to the cited literature.

To have a common understanding, first, the terms and definitions, including the description of the coordinate systems and states, are discussed in Sec. 2.1. In Sec. 2.2, the used map framework is briefly presented. Afterward, a short introduction to Hidden Markov Models, which are later used for inferring intentions, is given in Sec. 2.3. Sec. 2.4 and Sec. 2.5 present the concept of Markov Decision Processes and their extension, the Partially Observable Markov Decision Processes, including solving strategies and the adaptive belief tree algorithm. In Sec. 2.6, the evolutionary optimization algorithm of particle swarm optimization and subsequently its application to trajectory planning for automated vehicles (Sec. 2.7) is explained. The chapter concludes with a description of the field of view (Sec. 2.8) and a short introduction to neural networks (Sec. 2.9).

2.1 Terms and Definitions

Agent - The term agent is used to describe objects whose future trajectories are determined. There can be several agents in a traffic scene if multiple traffic participants' motions are planned simultaneously, e.g., if cooperative maneuvers are planned.

Ego vehicle - The own vehicle that is controlled by the automated driving functions is referred to as ego vehicle. It, therefore, is a specific agent. The term is mainly used if there is only the ego vehicle as an agent in the scene because only the motion of the ego vehicle is controlled.

Object - Object is a generic term for people, vehicles, signs, buildings, trees, or things in the scene.

Road user - A road user is an object or agent who arbitrarily participates in the traffic scene on and off the road. The term mainly comprises pedestrians, bicyclists, vehicles, or trucks. As a synonym for road user, the term traffic participant is used.

Vehicle-to-X - Vehicle-to-X comprises a vehicle's communication to its surroundings. The means of communication are diverse and can be achieved through ITS-G5 standards, consumer Wifi, or mobile communication networks.

2.1.1 Coordinate Systems

In order to speak a common language regarding positions, orientations, and velocities, the coordinate systems to which the information refers must first be defined. Automated driving has adopted the standard coordinate systems as a subdomain of robotics.

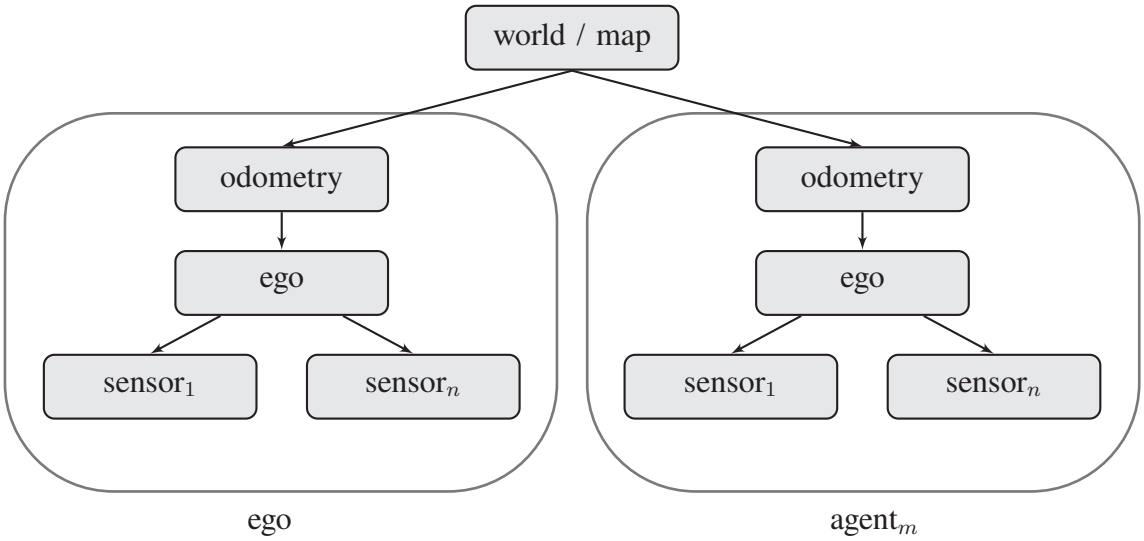


Figure 2.1: Kinematic chain of coordinate systems starting from the fixed *world* or *map* frame to the *sensor* frame of an object

The *world* frame is the central reference frame. This frame is also called *map* frame because all map information is given with respect to this global frame. The *map* frame is also the same for all agents in the scene. The transfer of information is therefore conducted in this frame. Global information, i.e., map information, is often processed in geo-referenced coordinates like GPS coordinates with latitude, longitude, altitude, and heading. In order to be able to perform calculations in a Cartesian coordinate system, the geo-referenced data is converted to metric coordinates with respect to a reference position using a suitable projection like the Transverse Mercator or the Universal Transverse Mercator (UTM). The East-North-Up (ENU) convention sets the orientation of the *map* frame so that the x-axis points to the east, the y-axis to the north, and the z-axis upwards.

The pose of an agent in its *odometry* frame is guaranteed to be continuous without discrete jumps. The pose changes smoothly and is, for example, determined by wheel or visual odometry. This is why this frame is often used for perception tasks. The pose in the *odometry* frame is accurate for short-term local localization but may drift globally over time. Through self-localization, the agent determines the offset between *odometry* and *map* frame to obtain its pose in a global frame. If the odometry and self-localization of the agent are perfect, there is a constant offset between the *odometry* and *map* frames.

The *ego* frame is the ego-centered frame. Like the *odometry* frame, each object or agent has a different *ego* frame. The position and orientation of sensors or the collision model are specified in the *ego* frame because they are stationary there. The x-axis of the *ego* frame points in the forward direction, the y-axis points to the left, and the z-axis points upwards.

The kinematic chain of frames is depicted in Fig. 2.1.

2.1.2 States

The geometry and location of an object are described by its position, orientation, and shape. The combination of position and orientation is called the *pose* $\vec{\xi}$ of the object. For a three-dimensional space, it results to

$$\vec{\xi}_{3D} = (x, y, z, \phi, \psi, \theta)^T \tag{2.1}$$

with

- ϕ as the roll angle for motions around the longitudinal axis,
- ψ as the pitch angle for motions around the lateral axis,
- θ as the yaw angle for motions around the vertical axis.

The *twist*, the differentiation of the pose with respect to time and thus the combination of linear and angular velocity, is subsequently defined as

$$\dot{\vec{\xi}}_{3D} = \left(\dot{x}, \dot{y}, \dot{z}, \dot{\phi}, \dot{\psi}, \dot{\theta}\right)^T. \tag{2.2}$$

Road users are only able to move along the ground. Furthermore, the so-called flat world assumption applies to most high-level automated driving functions like the prediction and planning components. This can be justified by the motion of the objects on a plane and the possibility of projecting all relevant information onto a two-dimensional plane. The description of poses and twists then simplifies to

$$\vec{\xi}_{2D} = (x, y, \theta)^T \tag{2.3}$$

and

$$\dot{\vec{\xi}}_{2D} = \left(\dot{x}, \dot{y}, \dot{\theta}\right)^T. \tag{2.4}$$

This work uses the flat world assumption if not stated differently. It is, therefore, referred to 2D information, where it is not explicitly mentioned that the information is three-dimensional. The suffixes $2D$ or $3D$ are neglected for better readability.

The *velocity* $\vec{v}$ of an object is the linear part of the twist and results in

$$\vec{v} = (\dot{x}, \dot{y})^T. \tag{2.5}$$

The *speed* v of an object is the absolute value of the velocity $|\vec{v}|$ and thus does not have a direction. The pose and twist information is often referred to as dynamic state information. It can also comprise higher-level derivatives like accelerations or also yerks.

In contrast to the dynamic state information, the shape of an object does not change. It is, therefore, referred to as static state information. The shape of an object is given by the polygon that defines its outlines. For some objects, e.g., vehicles, the shape can be approximated by a rectangle that is defined by its length and width. It should be mentioned that the shape of an object may change during object detection or tracking, for example, because it can only be detected gradually. However, in the components considered in this work, it is assumed that the shape is already known.

In addition to the previously mentioned information, objects can have further state information. The additional information used in the complementary approach in the further work is described in the corresponding chapter. This information is, in particular, *classifications* and *intentions*. The classification indicates the type of movements to be expected - whether along the road or on sidewalks - and whether the object is particularly vulnerable. Intentions I vary depending on the classification. For vehicles and

bicycles, the route intentions are of particular interest as they greatly determine the future motion of the object. The state s of an object results in

$$s = (\vec{\xi}, \dot{\vec{\xi}}, I, ...). \tag{2.6}$$

Uncertainties in the state s are represented by a covariance matrix for the position, orientation, and velocity components, assuming Gaussian distributed uncertainties. The uncertainties about the intentions are specified with a discrete probability distribution in which each intention is assigned a probability. Shapes and dimensions of objects are slightly overestimated for simplicity in this work to obtain values that are not subject to uncertainties. The overestimation can be conducted with the so-called beliefprints as proposed in [41]. The beliefprints represent the shape or area that is covered for a defined sigma uncertainty.

In this work, the time is discretized with the time interval Δt. The current point in time $t = 0$ is indexed with t_0. Previous times, meaning points in time before the current point in time, have negative indices, t_i with $i < 0$. Future points in time have positive indices, t_i with $i > 0$. The horizon to which a prediction or planning function is conducted is denoted with T_P. The state s of a traffic participant $n = 0 \dots N-1$ is described by s_n for a total of N traffic participants. The state s_n at time i is defined by $s_{n,i}$. The accumulated state information from a previous point in time t_i with $i < 0$ of the state of the traffic participant is defined as state history h

$$h_n = (s_{n,T_H}, s_{n,T_H+1}, \dots, s_{n,i}, \dots, s_{n,-1}), \text{ with } i = T_H, \dots, -1. \tag{2.7}$$

with T_H as the point furthest back in time.

The future evolution or prediction of the state s_n is defined as development $\mathcal{D}_n$.

$$\mathcal{D}_n = (s_{n,0}, \dots, s_{n,i}, \dots, s_{n,T_P}), \text{ with } i = 0, \dots, T_P. \tag{2.8}$$

The state of the environment S contains the states of all N traffic participants. It further contains the states of all regulatory elements $s_{r,i}$ that change over time, e.g., traffic lights. For N traffic participants and N_r regulatory elements, the state of the environment results to

$$\begin{aligned} S = \{&s_0, \dots, s_{N-1}, \\ &s_{r,0}, \dots, s_{r,N_r-1}\}. \end{aligned} \tag{2.9}$$

The history of the state of the environment is defined as

$$H = (S_{T_H}, \dots, S_i, \dots, S_{-1}), \text{ with } i = T_H, \dots, -1 \tag{2.10}$$

and the development is accordingly denoted as

$$\mathcal{D} = (S_0, \dots, S_i, \dots, S_{T_P}), \text{ with } i = 0, \dots, T_P. \tag{2.11}$$

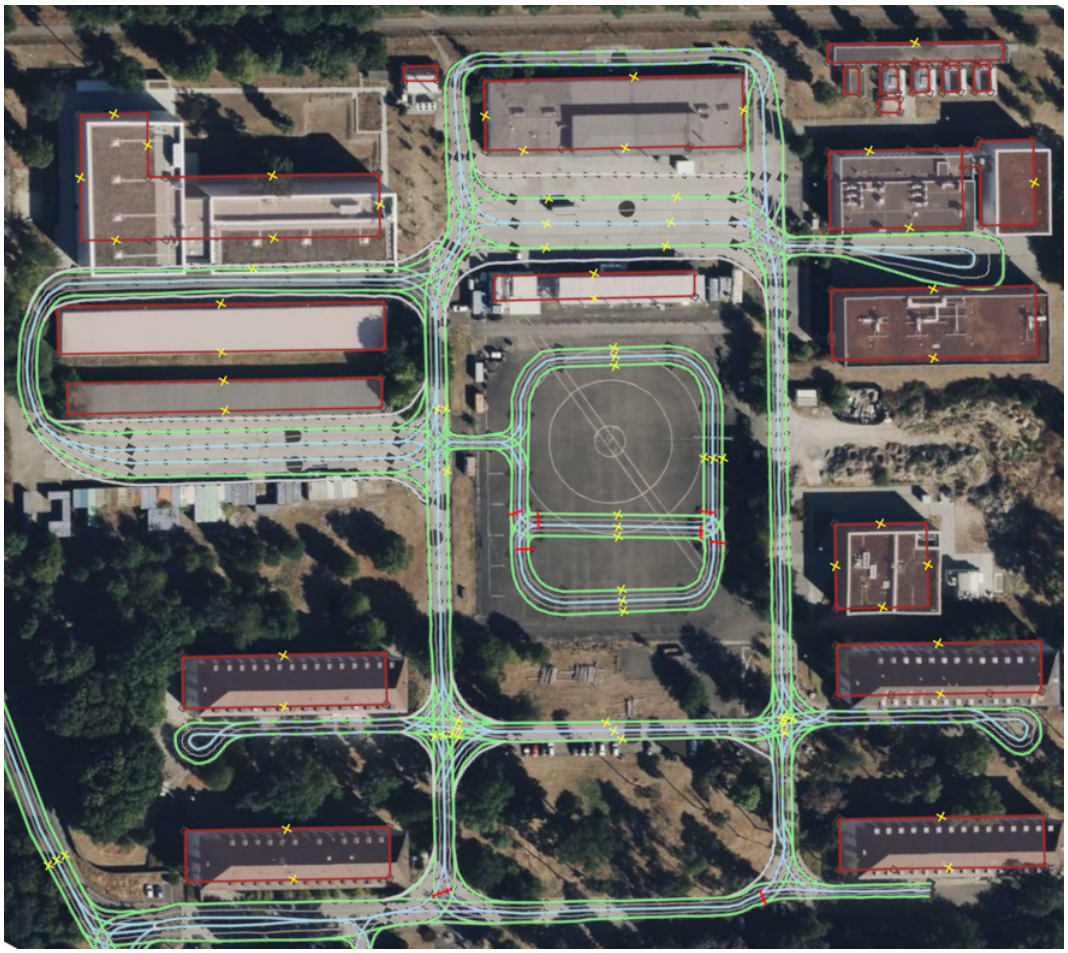


Figure 2.2: Lanelet map of the "KIT-Campus Ost" in Karlsruhe. The left boundaries of a lanelet are drawn in green, and the right boundaries are drawn in blue. Buildings are drawn with red bounds. Background image ©Google Maps, 2023 CNES / Airbus, Geobasis-DE/BKG, Geocontent, Maxar Technologies

2.1.3 Trajectory

The *trajectory* describes the change in a single object's dynamic state and motion over time. The trajectories can be expressed in two different ways. The typical way is to use absolute values, e.g., the absolute position for each point in time. The second way is to use incremental values with respect to the starting state, e.g., velocity values. This work uses time-discrete trajectories, which comprise the corresponding poses for discrete points in time. The trajectories are represented using absolute values unless it is explicitly specified otherwise. Derivatives can be calculated via differentiation.

A trajectory $\vec{\Gamma}$ from the current point in time to T_P has length $T_P + 1$ and is therefore given by $T_P + 1$ poses, including the start pose $\vec{\xi}_0$

$$\vec{\Gamma} = \begin{pmatrix} \vec{\xi}_0 \\ \vdots \\ \vec{\xi}_{T_P} \end{pmatrix}, \in \mathbb{R}^{3(T_P+1)}. \tag{2.12}$$

In contrast to a trajectory, a path is a sequence of positions or poses without temporal context. Besides the trajectory for each traffic participant, a development $\mathcal{D}$ additionally comprises further information like the change of intentions over time.

2.2 Map

Maps store static information about the environment and serve as an orientation for automated driving functions. Localization and navigation methods, in particular, make use of the stored information. Primarily, maps contain the road network with its lanes, speed limits, traffic signs, parking spaces, and traffic rules. Additionally, the maps often include information about buildings or characteristic landmarks for localization. Maps can be generated on the fly by the automated vehicle or generated offline. Certain automated driving functions rely on offline map data. Routing to a far away target location can only be achieved with prior knowledge of the road network. Furthermore, in this work, map information is used to track and reason about occluded road users and to infer the intentions of road users. The approaches rely on offline-generated map information because the vehicle cannot perceive the relevant areas beforehand.

This work uses the lanelet framework as proposed in [45]. The road network is divided into lanes, and lanes are further divided into small segments, the so-called lanelets. Lanelets have a left and a right boundary and can have arbitrary attributes like a type or speed limit. Relations are used to express the neighborhood relations between lanelets, like following or beside relations, and to describe traffic rules. Oncoming lanelets are also described with neighboring relations. Lanelets can also overlap. This primarily occurs in intersection areas. A lanelet with multiple following lanelets indicates different turning options. The following lanelets partially overlap because the boundaries have the same starting points but different end points. Also, crossing or merging lanelets may overlap. The challenge is to determine the correct mapping of traffic participants to the lanelet they are assumed to follow. The traffic rules are encoded using relations for prioritized roads and by adding information about traffic signs and lights. Lanelets can also be used to describe sidewalks or other areas of interest.

The framework is based on an XML-based data format derived from Open Street Map (OSM). It allows the encoding of arbitrary information on and off the road and allows three-dimensional positions.

2.3 Hidden Markov Models

A Hidden Markov Model (HMM) [125], as shown in Fig. 2.3, is a special form of a Dynamic Bayesian Network (DBN). The basic idea is that the underlying Markov chain can not be measured directly. The states are hidden and are estimated through observable emissions.

HMMs are applied in various fields like speech and text recognition, computer vision, or finance. In speech and text recognition tasks, the underlying system is modeled as a Markov process with a finite number of discrete hidden states $\mathbb{S}$ that cannot be observed. The HMM λ is formed by the tuple $\lambda = \{\mathbb{S}, \mathbb{O}, \mathcal{T}, \mathcal{Z}, \Pi\}$, with the discrete or continuous observed variables $\mathbb{O}$, the state transition function $\mathcal{T}$, observation probabilities $\mathcal{Z}$ and the initial probabilities Π. The symbols used here differ from the

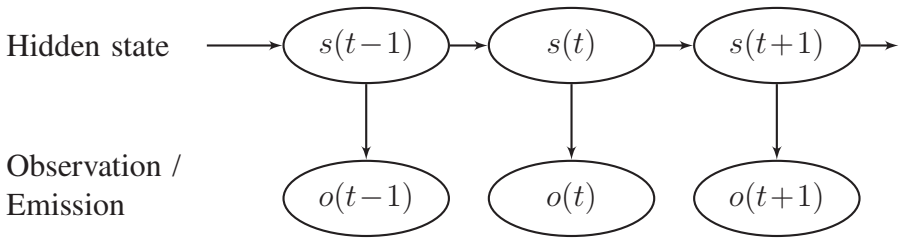


Figure 2.3: Depiction of a HMM with hidden states s_i and observations o_i

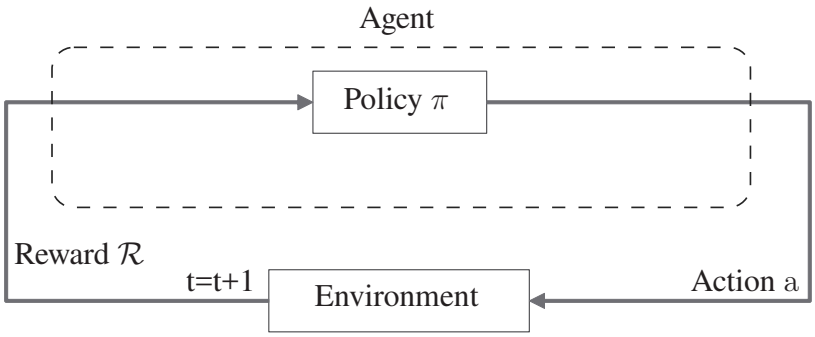


Figure 2.4: Depiction of the interaction of the agent with the environment for a MDP.

commonly used symbols to align with the terms used in this thesis. In literature, the standard notation is $\lambda = \{X, Y, A, B, \Pi\}$ with state space X, the discrete or continuous observed variables Y, the state transition function A, observation probabilities B and the initial probabilities Π.

The state transition function $\mathcal{T}$ and observation probabilities $\mathcal{Z}$ are typically trained to fit the model to the observed variables optimally. For the optimization, often the Baum-Welch algorithm [43] is used. It is an expectation-maximization algorithm that leverages the forward-backward algorithm to find the unknown parameters. A core advantage of HMMs is that assumptions about passed states are corrected with posterior observations. They can also be applied when the number of measurements is unknown. However, the estimation quality and confidence increase with an increasing number of measurements.

2.4 Markov Decision Processes

Richard Bellman introduced the Markov Decision Process (MDP) in 1957 as a general framework for modelling decision-making problems [44]. The problem is thereby completely described by the subsequent tuple

$$(\mathbb{S}, \mathbb{A}, \mathcal{T}, \mathcal{R}, T_P, \gamma)\,. \tag{2.13}$$

The state space $\mathbb{S}$ denotes the set of possible states s in which the agent or the world can be. It is assumed that the actual state is known at any time. Initially, the agent or the world starts in a state $s_0 \in \mathbb{S}$.

The agent interacts with its environment by executing an action $\mathrm{a}_t \in \mathbb{A}$. $\mathbb{A}$ is the action space, i.e., the set of actions the agent can take. It is the agent's only way to influence the environment. The executed action then generates a transition to another state s_{t+1}. This is expressed by the state transition function $\mathcal{T}(s_{t+1}, s_t, \mathrm{a}_t)$, which describes the transition to state s_{t+1} when executing action a_t in the current state s_t. It is a conditional probability function

$$\mathcal{T}(s_{t+1}, s_t, \mathrm{a}_t) = P(s_{t+1}|s_t, \mathrm{a}_t). \tag{2.14}$$

The stochastic state transition fulfills the *Markov Property* of order one. This means, that the next state s_{t+1} only depends on the current state s_t and action a_t, but not on previous states $s_{t-1}, ..., s_0$ or actions $\mathrm{a}_{t-1}, ..., \mathrm{a}_0$ leading to s_t [112]. The following equation can formally express this property

$$P\left(s_{t+1}|s_t, \mathrm{a}_t, s_{t-1}, \mathrm{a}_{t-1}, ..., s_0, \mathrm{a}_0\right) = P\left(s_{t+1}|s_t, \mathrm{a}_t\right). \tag{2.15}$$

The general sequence of a MDP is shown in Fig. 2.4. After taking an action, the agent receives a real-valued reward given by the reward function $\mathcal{R}(s_t, \mathrm{a}_t)$ for executing action a_t in state s_t. It expresses how desirable a particular action is for the agent.

Solving the MDP means finding the optimal sequence of actions for the agent to reach its goal with regard to uncertain future states. This mapping of actions to be taken in certain states is called policy $\pi : \mathbb{S} \rightarrow \mathbb{A}$.

The value V_π for a policy π is defined as the expected sum of rewards when following the policy starting from s_0

$$V_\pi(s_0) = E\left[\sum_{t=0}^{T_P} \gamma^t \mathcal{R}\left(s_t, \pi(s_t)\right)\right]. \tag{2.16}$$

The discount factor $\gamma \in [0, 1)$ emphasizes recent events and reduces the impact of future rewards. It assures a limited value for infinite time horizons, $T_P \rightarrow \infty$.

The optimal policy π^* is determined by maximizing each state's expected sum of rewards. As the reward function $\mathcal{R}$ depends on the corresponding action, the optimal policy π^* describes the action a to be taken in state s to maximize the expected reward

$$\pi^* = \arg\max_\pi V_\pi\left(s\right). \tag{2.17}$$

2.5 Partially Observable Markov Decision Processes

The assumption that the state of the environment and the agent's state are perfectly known is often not valid in practice. The state can often only be estimated from observations. The state estimation is subject to uncertainties that come with noisy sensor measurements. Additionally, measurements are only available in areas perceived by the sensors. To cope with the partially observable environment, the Partially Observable Markov Decision Process (POMDP) was invented as a generalization of MDPs. In POMDPs, the state of the agent and the environment is estimated through observations. It was first described by *Karl Johan Åström* in 1965 [37] and adapted by *Leslie P. Kaelbling* and *Michael L. Littman* in 1998 [87]. The estimated state is called belief and is represented by a probability distribution over all possible states.

A POMDP is defined by the tuple

$$\left(\mathbb{S}, \mathbb{A}, \mathcal{T}, \mathbb{O}, \mathcal{Z}, \mathcal{R}, b_0, \gamma\right), \tag{2.18}$$

with

- the state space $\mathbb{S}$ containing all possible states s,
- the action space $\mathbb{A}$ comprising the set of possible actions a,
- the state transition function $\mathcal{T}(s_{t+1}, s, \mathrm{a})$ for moving to the next state s_{t+1} when executing action a in the current state s,

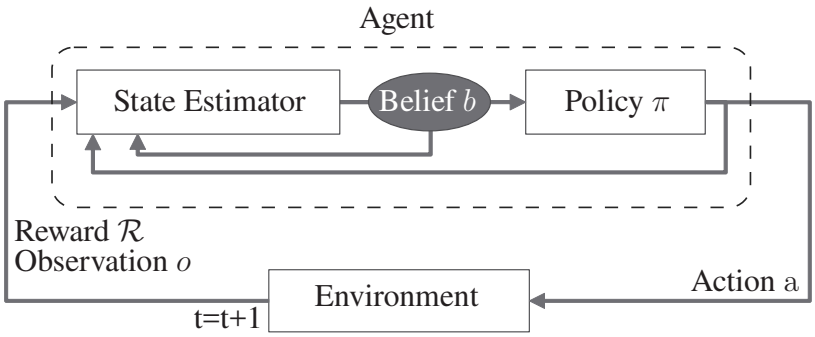


Figure 2.5: Depiction of the agent's interaction with the environment for a POMDP.

- the observation space $\mathbb{O}$, that contains the observations o,
- the conditional probability function $\mathcal{Z}(s_{t+1}, \mathrm{a}, o) = P(o|s_{t+1}, \mathrm{a})$ for the probability of making the observation o in dependence of the executed action a and the next state s_{t+1},
- the immediate reward $\mathcal{R}(s, \mathrm{a})$ after executing action a in state s,
- the initial belief b_0
- and the discount factor $\gamma \in [0, 1)$ for weighting rewards.

The agent and the environment start in state $s_0 \in \mathbb{S}$. As the state is not entirely known, an initial belief b_0 about the state is assumed. The agent's interaction with the environment is shown in Fig. 2.5. After the agent takes action a, the system transitions to the next state s_{t+1}. The state transition function $\mathcal{T}(s_{t+1}, s, \mathrm{a})$ describes the stochastic transition. The agent then receives an observation $o \in \mathbb{O}$ that is given by the conditional probability function $\mathcal{Z}(s_{t+1}, \mathrm{a}, o)$. $\mathcal{Z}$ describes the imperfect and noisy perception of the environment. The knowledge about the taken action a and the received observation o is used to update the belief b. Simultaneously, the agent receives the immediate reward $\mathcal{R}(s, \mathrm{a})$ for taking action a in state s. Similar to MDPs, the agent's goal is to find the optimal policy that maximizes the expected total reward for a specific planning horizon T_P. The discount factor γ guarantees that the reward converges to a finite value even for infinite planning horizons.

2.5.1 Belief

Before presenting strategies to solve POMDPs, the belief and the representation of the policy as a belief tree are described in the following to achieve a deeper understanding of the process.

As mentioned above, the state of the environment is not exactly known due to partial observability. Instead, a belief b about the state is estimated. The set of all beliefs is denoted by $\mathbb{B}$. The belief has to be inferred from the agent's previously taken actions and perceived observations. This is described by the history

$$h_t = (\mathrm{a}_0, o_0, \mathrm{a}_1, ..., o_{t-1}, \mathrm{a}_{t-1}), \tag{2.19}$$

with actions a_i taken at time $i = 0 \dots t-1$ and the corresponding gathered observations o_i with $i = 0 \dots t-1$ [129]. The current belief b_t can be calculated with the belief update function τ as the posterior probability distribution of being in state s given the complete history h_t and the initial belief b_0:

$$b_t(s) = P(s_t = s|h_t, b_0) = \tau\left(b_0, \mathrm{a}_0, o_0, \mathrm{a}_1, o_1, ..., \mathrm{a}_{t-1}, o_{t-1}\right), \tag{2.20}$$

POMDPs like MDPs satisfy the *Markov Property* of order one. Applied to beliefs, this means that the belief b_t only depends on the previous belief b_{t-1}, action a_{t-1} and the observation o_{t-1} [137]. Equation 2.20 accordingly simplifies to

$$b_t = \tau(b_{t-1}, \mathrm{a}_{t-1}, o_{t-1}) \tag{2.21}$$

and the belief update is performed with [129]

$$\begin{aligned} b_t(s_t) &= \tau(b_{t-1}, \mathrm{a}_{t-1}, o_{t-1})(s_t) \\ &= \frac{1}{P\left(o_{t-1}|b_{t-1}, \mathrm{a}_{t-1}\right)} \mathcal{Z}(s_t, \mathrm{a}_{t-1}, o_{t-1}) \sum_{s \in \mathbb{S}} \mathcal{T}(s_t, \mathrm{a}_{t-1}, s_{t-1}) b_{t-1}(s_{t-1}). \end{aligned} \tag{2.22}$$

The probability $P\left(o|b, \mathrm{a}\right)$ of observing o after executing action a in belief b normalizes $b_t(s_t)$ to remain a probability distribution. It is calculated with

$$P\left(o_{t-1}|b_{t-1}, \mathrm{a}_{t-1}\right) = \sum_{s_t \in \mathbb{S}} \mathcal{Z}(s_t, \mathrm{a}_{t-1}, o_{t-1}) \sum_{s_{t-1} \in \mathbb{S}} \mathcal{T}(s_t, \mathrm{a}_{t-1}, s_{t-1}) b(s_{t-1}). \tag{2.23}$$

2.5.2 Belief Tree

The rollout of taking an action and receiving an observation can be represented as a so-called belief tree. Belief trees vividly show the connection of beliefs through action-observation pairs and display the development over time. An example is shown in Fig. 2.6. In this example, the indices a and o serve as identifiers, not time indices. When starting at belief b_0 and taking action a_1, two possible observations o_1, o_2 can be made. With respect to the actually made observation, the environment finds itself in belief b_{a_1, o_1} or b_{a_1, o_2}.

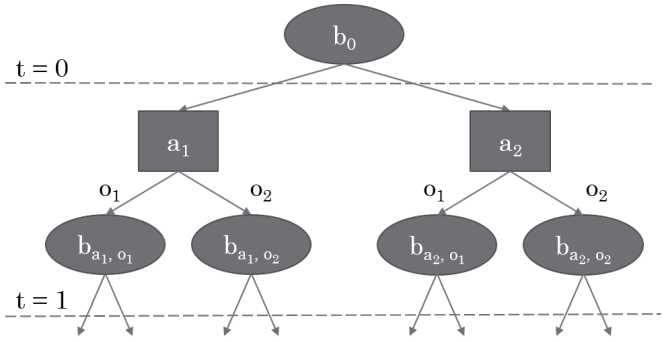


Figure 2.6: Belief tree with initial belief b_0, two actions a_1 and a_2, and two possible observations o_1 and o_2 resulting the successive child beliefs

The action space $\mathbb{A}$ and the observation space $\mathbb{O}$ determine the number of actions or observations. In the case of discrete spaces, the numbers are limited, and the development can be represented as a belief tree containing all possible combinations. For continuous spaces, only a subset can be represented in this way. The state, action, and observation space choices also influence finding the optimal policy.

2.5.3 Solving Partially Observable Markov Decision Processes

Solving POMDPs equals finding the optimal policy π^*. The policy $\pi : \mathbb{B} \mapsto \mathbb{A}$ assigns actions $\mathrm{a} \in \mathbb{A}$ to the beliefs $b \in \mathbb{B}$. The optimal policy maximizes the expected sum of discounted rewards

$$\pi^* = \arg\max_{\pi} E\left[\sum_{t=0}^{\infty} \gamma^t \sum_{s \in \mathbb{S}} b_t(s) \sum_{\mathrm{a} \in \mathbb{A}} \mathcal{R}(s, \mathrm{a}) \pi(b_t, \mathrm{a}) \mid b_0\right]. \tag{2.24}$$

$\pi(b, \mathrm{a})$ describes the probability that action a will be taken in belief b with respect to the policy π. The value of a policy π represents the expected total reward of following π from belief b. It is calculated using the value function V_π

$$V_\pi(b) = \sum_{\mathrm{a} \in \mathbb{A}} \pi(b, \mathrm{a}) \left[\mathcal{R}_B(b, \mathrm{a}) + \gamma \sum_{o \in \mathbb{O}} P(o \mid b, \mathrm{a}) V_\pi(\tau(b, \mathrm{a}, o))\right], \tag{2.25}$$

where $\mathcal{R}_B(b, \mathrm{a})$ is the immediate reward of executing action a in belief b

$$\mathcal{R}_B(b, \mathrm{a}) = \sum_{s \in \mathbb{S}} b(s) \mathcal{R}(s, \mathrm{a}). \tag{2.26}$$

Solving the POMDP thus means maximizing the expected cumulative future rewards by determining the best action to select for a given belief. When the POMDP is viewed as a belief state MDP, the expected cumulative reward for choosing action a in belief b is described by the Q-value with

$$Q(b, \mathrm{a}) = \mathcal{R}_B(b, \mathrm{a}) + \gamma \sum_{o \in O} P(o|b, \mathrm{a}) V^*\left(\tau(b, \mathrm{a}, o)\right). \tag{2.27}$$

As mentioned above, the discount factor γ ensures the value function converges, and thus, an optimal policy can be determined. The discount factor γ also balances the emphasis on immediate or future rewards. Lower values of γ discount future rewards more and give immediate rewards higher weights. For problems with finite horizon, γ can also be set to 1.

There are different strategies for solving POMDPs depending on the problem at hand and the formulation and choice of the POMDP components.

Online and Offline Policy Approximation

For problems with limited action, observation, and state spaces, the policy can be calculated for every given belief b. Examples are the tiger example or games like checkers. Checkers has discrete state, observation, and action spaces. The strategy of the opponent is not observable. For each possible belief

b, the optimal action can be calculated and stored offline. While playing, the optimal action can be looked up immediately at any time and for any belief.

However, this is not possible for complex problems. Here, only the current optimal action is determined and executed. Afterward, the process is repeated from the new belief. This is called online policy approximation. The convergence is achieved faster because only the immediate part of the policy is determined. On the downside, the calculation has to be performed in every iteration.

Point-based Solutions

The policy π, in general, is defined for every possible belief b. However, it often is not required to have the optimal policy for an arbitrary belief but for a specific current belief b_0. Then, the policy must only be determined for a specific belief, resulting in a point-based policy $\pi(b_0)$ or a point-based solution of the POMDP. Online applications often rely on point-based solutions. When executed online, the current belief for which the optimal action needs to be determined is known. Additionally, real-time constraints do not allow to find solutions for other beliefs unnecessarily.

Approximate Solutions

In general, [137] proved that the optimal value function V^* can be expressed using so-called alpha vectors α_i that span up a hyperplane in the belief space $\mathbb{B}$ for finite horizons. Each alpha vector α_i is associated with an action a_i and defines a linear value function. The value $V_t(b)$ then is the maximum of the alpha vectors evaluated at position b

$$V_t(b) = \max_{\alpha_i} \sum_{s \in \mathcal{S}} \alpha(s) b(s). \tag{2.28}$$

For problems with finite horizon, this representation can be used to find the exact solution. The α-vectors can be determined using value iteration. However, the complexity of finding the α vectors grows exponentially with the number of observations and quadratic in the number of states. This makes it an unsuitable strategy for complex problems [120] and infeasible for continuous state or observation spaces.

For this reason, approximate solving strategies became popular. Approximate solvers also allow to solve POMDPs with infinite horizon where no exact solution can be found [111]. These solvers seek the tradeoff between computational complexity, runtime, and convergence. Recent state-of-the-art solvers use Monte-Carlo methods, which are applicable for continuous state and observation spaces. Examples of approximate online solvers include Partially Observable Monte-Carlo Planning (POMCP) [136], Monte Carlo Value Iteration (MCVI) [39], Determinized Sparse Partially Observable Tree (DESPOT) [154], and Adaptive Belief Tree [102]. In this work, the Adaptive Belief Tree was used. It comes with a toolkit to use the solver for generic problems [97].

2.5.4 Adaptiv Belief Tree

This section only provides a brief overview of the Adaptive Belief Tree (ABT) algorithm. For more details, the reader is referred to [102].

The ABT leverages Monte-Carlo sampling. It can be used for problems with discrete action space and continuous or discrete state and observation spaces, making it a perfect match for decision-making for automated vehicles. Each belief thereby comprises a set of state particles. The policy is defined as action-belief pairs and explicitly maintains the relation between beliefs and states. By sampling episodes H_{ABT}, each belief b is associated with state trajectories from the initial belief b_0 to a state particle of b. Each episode $h_{ABT} = (s, \mathrm{a}, o, \mathcal{R}) \in H_{ABT}$ is a sequence of quadruples of states $s \in \mathbb{S}$, actions $\mathrm{a} \in \mathbb{A}$, observations $o \in \mathbb{O}$ and an immediate reward $\mathcal{R} = \mathcal{R}(s, \mathrm{a})$.

To create an episode, an initial state s_0 is sampled from the initial belief b_0. After selecting an action a_0 and an observation o_0, the observation probability, the successiv state s_1, and the corresponding reward $\mathcal{R}_0$ are generated forming the quadruple $h_{0,ABT} = (s_0, \mathrm{a}_0, o_0, \mathcal{R}_0)$. ABT models the transfer function $\mathcal{T}$ and observation function $\mathcal{Z}$ as generative functions to be designed by the user for the individual problem. The sampling process is repeated until a termination condition is reached. The episodes form a belief tree where all state particles pass through the associated belief. The actions are selected using the *Upper Confidence Bound* (UCB) strategy [38]. It represents the action selection as a multi-armed bandit problem [92] and aims to balance exploration and exploitation. Exploitation seeks to reuse actions with high Q-values, whereas exploration tries to expand branches that have yet to be searched thoroughly. The estimated Q-value is calculated as the mean of all episodes that pass through the belief node.

On changes, ABT identifies the affected episodes and regenerates them with respect to the change. This allows ABT to maintain the unaffected episodes, saving unnecessary recalculations of the complete policy. This adaption process gave the algorithm its name adaptive belief tree.

2.6 Particle Swarm Optimization

Kennedy and Eberhart proposed Particle Swarm Optimization (PSO) in 1995 [93]. The algorithm is a metaheuristic optimization algorithm inspired by fish or birds' flocking behavior. Especially for non-linear problems or problems where the calculation of gradients is impossible, PSO proved to find reasonable solutions. However, the algorithm is not complete, meaning that there is no guarantee that a globally optimal solution will be found. Each particle of the particle swarm represents a possible solution to the problem. On the one hand, this makes the algorithm anytime capable, so that the optimization can be stopped at any time. This is relevant for time-critical components or components with fixed round trip times and allows for balancing the solution's optimality against calculation time. On the other hand, for the initialization of the particle swarm, each particle has to be initialized with a valid solution. Depending on the optimization problem, this might pose a significant challenge.

Each particle is rated with the so-called fitness score, describing the quality of the solution according to a cost or rating function. Optimization aims to find the best fitness score by moving the particles in the solution space. Therefore, each particle of the set of particles $p \in \mathbb{P}$ has a position $\vec{x}_t$ in the solution space and a corresponding particle velocity $\vec{v}_t$ for every iteration t. Additionally, the personal best position $\vec{x}_{\mathrm{p},t}$ for each particle and the global best position $\vec{x}_{\mathrm{g},t}$ for all particles until iteration t is remembered. The global best position, i.e., the position and thus solution with the best fitness value, is, therefore, the best solution. The process of the PSO is divided into three steps.

The initial particle swarm is generated during the initialization step. Each particle represents a valid solution to the optimization problem at hand. The particle positions should thereby cover the search space thoroughly. Different strategies can obtain the initial distribution of the particles. For example, the

particles can be initialized entirely randomly, using available prior knowledge, or using a heuristic for a more informed initialization. The initial fitness of each particle is calculated using a given cost function. The particles can be initialized with and without initial velocities. It depends on the search space, the prior knowledge, and the strategies pursued during the optimization. After the initialization, the particle swarm consists of a set of valid particles, i.e., valid solutions to the initialization problem.

In the optimization itself, the new velocity of the particles is calculated. The velocity is determined using the personal best position $\vec{x}_{\mathrm{p},t}$ and the best position $\vec{x}_{\mathrm{g},t}$ within the entire swarm so far according to

$$\vec{v}_{t+1} = w_{\mathrm{v}} \cdot \vec{v}_t + w_{\mathrm{p}} \cdot u_1 \cdot (\vec{x}_{\mathrm{p},t} - \vec{x}_t) + w_{\mathrm{g}} \cdot u_2 \cdot (\vec{x}_{\mathrm{g},t} - \vec{x}_t)\,, \tag{2.29}$$

with w_{v}, w_{p} and w_{g} being constant weight factors and $u_i \sim U(0,1)$ uniformly distributed random numbers. Afterward, the position of the particles is updated as follows:

$$\vec{x}_{t+1} = \vec{x}_t + \vec{v}_{t+1}\,. \tag{2.30}$$

Finally, the new particle position is used to update the fitness values. Additionally, the personal best $\vec{x}_{\mathrm{p},t}$ and global best $\vec{x}_{\mathrm{g},t}$ are recalculated.

The operators for subtracting and adding two particles or particle velocities and the operator for multiplying a particle velocity with a scalar value from equations 2.29 and 2.30 must be defined. They must support the search space's and the optimization problem's characteristics.

The optimization step is repeated until a termination criterion is met. Termination criteria are manifold. The most used criteria are a given time limit, a maximum number of iterations, or reaching a specific fitness value. There are also process-related termination criteria, like losing diversity within the swarm. When the particles no longer change their position or their fitness values do not improve further, the optimization can be stopped.

The particles are checked for validity after the initialization and each optimization step. This step is essential to have valid solutions and to avoid drawing particles towards infeasible solutions.

2.7 Trajectory Planning based on Particle Swarm Optimization

In the last section, the PSO was described in general. The following briefly presents its application of trajectory planning for automated vehicles. A deeper introspection of PSO for trajectory planning is given in [3], [4], and [15].

When using PSO to optimize the initial trajectories, each particle represents a trajectory. The initialization of the particle swarm, i.e., finding a number of feasible initial trajectories, poses a huge challenge in the context of trajectory planning in dynamic environments. The approach presented in Sec. 5.3 provides a solution to the problem. The approach uses machine learning to replace hand-tuned initialization heuristics. It thereby shows how to safely apply machine learning in safety critical tasks like trajectory planning. The particle's position in the search space $\vec{x}$ is equivalent to the trajectory description. The dimension of the search space equals $\mathbb{R}^{\dim(\vec{x})\cdot T_P}$. However, due to the limitation of the road and constraints on the vehicle's motion, the area of the search space that is effectively used is strongly restricted.

The restrictions, in general, can be separated into inner and outer constraints. Inner constraints comprise the dynamic characteristics and restrictions of the vehicle. To obtain feasible trajectories, the accelerations, steering speeds, and steering accelerations are limited to respect the limitations of the engine, brakes, and steering unit. In order to achieve a certain comfort level, accelerations, yerk, and yaw rates, as well as decelerations, are also limited. A further restriction imposed by the speed limit concerns the vehicle's maximum speed. Outer constraints include the constraints inflicted by the environment. First of all, the trajectories have to be safe. The vehicle must keep a safe distance from other objects and must not cause any collisions. Finally, the vehicle has to stay on the road and should not cross any lane boundaries.

For a trajectory as given by 2.12, the optimization problem with respect to the previously mentioned constraints results to

$$\min_{\Gamma} \; J\left(\Gamma\right) = \sum_{k=0}^{T_P} C\left(t_k, \vec{\xi}_k, \dot{\vec{\xi}}_k, \ldots, \vec{\xi}_k^{(n)}\right) \; \Delta t\,, \tag{2.31}$$

where $\vec{\xi}^{(n)}$ denotes the n-th derivative with respect to time, and C is the cost functional to determine the fitness of the particles. It consists of M different cost terms that describe the desired behaviors of the vehicle. These are accumulated in a weighted sum

$$C = \sum_{j=1}^{M} w_j C_j. \tag{2.32}$$

The cost terms are adjusted to the specific problem. The following list provides examples that can be used for trajectory planning for automated vehicles:

- limitation of longitudinal accelerations for comfort and efficiency
- limitation of yaw rates for comfort and efficiency
- limitation of lateral accelerations for comfort
- limitation of yerk for comfort
- distance to obstacles
- distance to the road boundaries
- deviation from the speed limit

2.8 Field of View

The field of view (FOV) describes the area the automated vehicle can perceive with its sensors. In multi-modal sensor setups, the field of view can be built from the FOV of each sensor modality or each sensor.

The algorithms developed in this work are kept independent of the actual sensor setup by using a generic model for the sensor setup. Thus, the concepts are flexible to use cameras, light detection and ranging (Lidar) sensors, or any combination. The generic sensor setup determines the theoretic FOV that can be

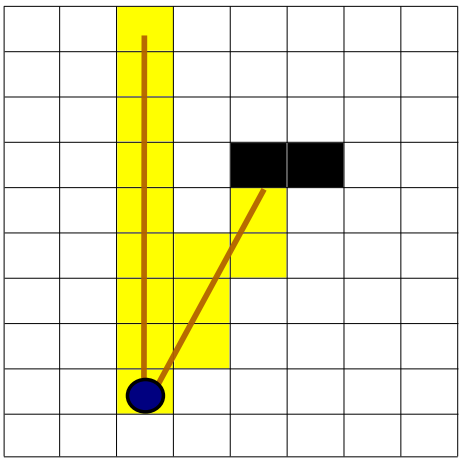


Figure 2.7: Illustration of raytracing algorithm on a grid. The rays are sent from the sensor in the bottom left. Cells traced as free space are marked in yellow.

obtained. It, therefore, also determines the areas where no observation can be expected. This becomes important when making assumptions about occluded areas and potential observations at future points in time.

In general, the FOV is calculated from three-dimensional sensor data. However, in accordance with the flat world assumption, a two-dimensional view of the result is sufficient. The FOV is calculated using ray casting [130]. Thereby, rays are sent out repeatedly in discretized steps for the sensor's opening angle. The sensor's mounting position and orientation determine the rays' starting point and direction. The endpoint can be marked as a hit if the ray collides with an obstacle or the ground. The endpoint does not mark a hit if the ray reaches the specified maximum sensor range. The area that the ray traveled through between the start and end point can be interpreted as free space. A simplified example is shown in Fig. 2.7.

The FOV is often visualized in a birds-eye view of the vehicle, the surrounding environment, and the visualization of the areas covered by the sensors. This work uses two ways to represent the FOV. These differ in calculation speed, memory requirements, precision, and the procedures that can be applied to them.

2.8.1 Polygon based Field of View

The polygon-based representation of the FOV uses a set of two-dimensional polygons to describe the areas that are perceived by the sensors. Without obstacles, the polygons approximate the circle segment that is captured by the sensor. The vehicle's FOV is thereby created as the union of the single sensors' FOV. If these all overlap, a single polygon is obtained. Otherwise, a set of non-overlapping polygons is the result. Areas outside the FOV that do not belong to an object are assumed to be not visible. The polygons of the FOV are obtained by performing ray casting as described above. Afterward, a single sensor's start and end points are connected, forming the outline of the resulting polygon. To accelerate the expensive ray tracing, this work made use of bounding volume hierarchies for efficient intersection tests.

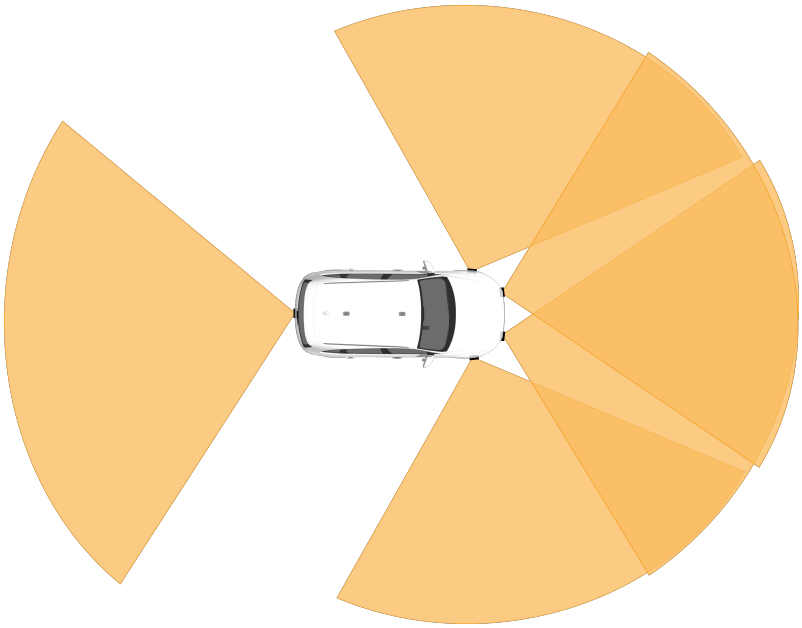


Figure 2.8: Lidar sensor setup for test vehicle CoCar. (Graphic from [1], ©IEEE 2019)

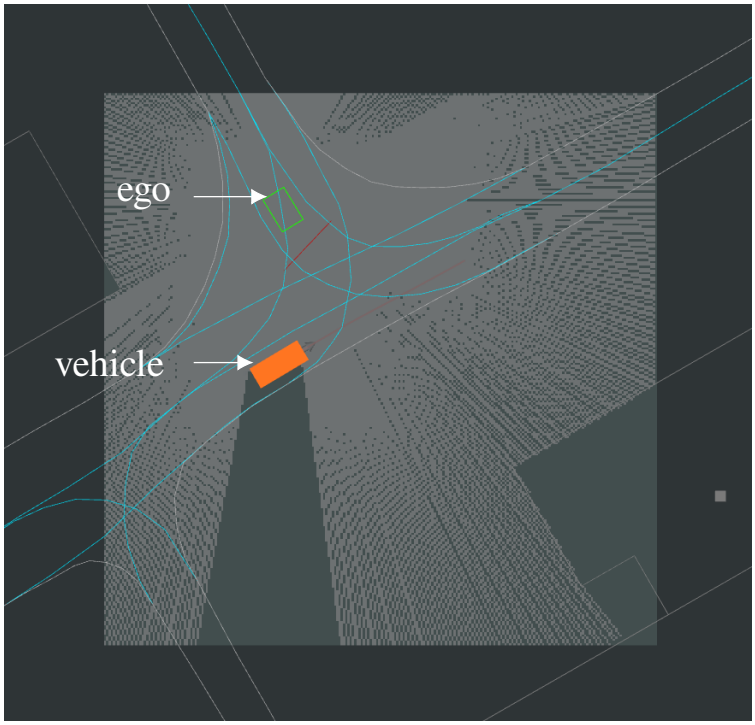


Figure 2.9: Grid-based FOV for an intersection situation. The contour of the ego vehicle is drawn as a green rectangle. A vehicle approaching the intersection is drawn in orange. The road network is drawn in blue.

Fig. 2.8 shows the exemplary sensor setup and resulting FOV for Cocar, one of the automated test vehicles of the FZI. The vehicle is equipped with five Ibeo Lux Lidar sensors, producing a two-dimensional point cloud around the vehicle. The sensor range is reduced for depiction purposes.

2.8.2 Grid-based Field of View

The grid-based representation uses two-dimensional grids with a fixed cell resolution to describe if the corresponding cells are occupied, free, or if no information is available. This means that they are not visible and, therefore, occluded.

The advantage of the grid-based representation is the direct use of machine learning techniques like convolutional neural networks, the faster calculation speed, and the union of multiple three-dimensional Lidar point clouds into one resulting grid. However, this comes with the downside of losing precision and higher memory usage.

The Bresenham algorithm [49] is used to determine the cells along the ray. This incremental error algorithm excels because of its speed and simplicity. It was developed in 1962 by Jack Elton Bresenham as one of the first algorithms developed in computer graphics and has been widely used ever since.

The confidence can be estimated since the number of rays passing through a single cell or hitting a single cell can be accumulated. This can be done by estimating the probability of the cell being occupied or by applying thresholds to, for example, eliminate outliers.

The FOV represented as a grid is depicted in Fig. 2.9. The scene shows the ego vehicle as a green box. It approaches an intersection from the top left corner. Another vehicle passes the intersection, which is drawn as an orange rectangle. The FOV is shown as a two-dimensional grid where each visible cell is marked in light grey. The other cells are kept dark. It can be seen that the orange vehicle and the static objects, where only the contours are drawn, obstruct the sight.

2.9 Neural Networks

Neural networks are used in almost all domains nowadays. The initial motivation of mimicking the biological architecture of actual brains resulted in various expressions and forms that are specialized for the corresponding task. The underlying principle is based on single neurons that are interconnected. A brief description of the types of neural networks used in this thesis is given below. For a detailed and extensive overview, the reader is referred to [67] or [33].

Activation Functions

An activation function mirrors the electrical potential and influences the forwarding of the information. The activation functions that are mainly used are non-linear to prevent the network from becoming a simple linear regressor. Prominent activation functions are the *sigmoid*, *Rectified Linear Unit (ReLU)*, and *spatial softmax*, which are explained in the following.

Sigmoid

The sigmoid activation function is defined as

$$f(x) = \frac{1}{1 + \mathrm{e}^{-x}} \tag{2.33}$$

for an input x. It maps the input value to the range $(0, 1)$. The gradients for x towards plus and minus infinity converge to zero. This leads to problems in the training process. It is known as the vanishing gradient problem.

Rectified Linear Units

The ReLU activation function [66] does not suffer from the vanishing gradient problem for positive inputs x. ReLU is defined as

$$f(x) = \begin{cases} 0, & x \leq 0, \\ x, & x > 0. \end{cases} \tag{2.34}$$

The gradient is zero for negative inputs. To overcome this problem, derivatives of ReLU like Leaky rectified linear unit (Leaky ReLU) or, more generally, the Parametric rectified linear unit (PReLU) [72] were proposed that use a linear slope for negative input values resulting in

$$f(x) = \begin{cases} \alpha x, & x \leq 0, \\ x, & x > 0. \end{cases} \tag{2.35}$$

Spatial Softmax

The spatial softmax considers input feature maps m of size $W \times H$ instead of single input values and effectively is a sigmoid activation normalized over the entirety of the corresponding feature map. At position (i, j), it is defined in the following way.

$$f(i, j) = \frac{\mathrm{e}^{m(i,j)}}{\sum_{x'=0}^{W-1} \sum_{y'=0}^{H-1} \mathrm{e}^{m(i',j')}} \tag{2.36}$$

In Convolutional Neural Networks (CNNs), the spatial softmax normalizes the values in the last layers so that they sum up to 1 to be regarded as a probability distribution.

Loss Functions

Loss functions are metrics that compare the network output with the target values of the training data and indicate how good the prediction is. The aim of the training process is to minimize the loss between target and network output. The loss function significantly influences the training performance and the network's ability to generalize on new input. Loss functions can be distinguished into two types.

Classification loss functions are used when the network is used for finding the most probable class from pre-set categories. The network predicts probabilities for each target class. An example is the Cross-Entropy with labels y and predictions x

$$\mathcal{L}_{CC} = -\sum_{j=0}^{C-1} y(i) \log\left(x(i)\right). \tag{2.37}$$

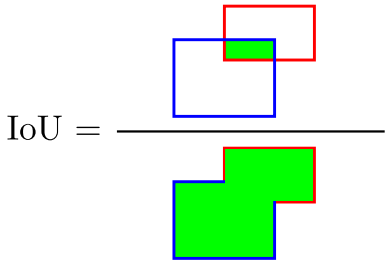


Figure 2.10: IoU for the example of two bounding boxes (red and blue) with the intersection and union drawn in green, respectively.

Regression loss functions rate the error of the predicted value to its corresponding input value. The Mean Squared Error (MSE) is a widely used example. It calculates the averaged squared error over the dataset between labels y and predictions x

$$\mathcal{L}_{MSE} = \frac{1}{N} \sum_{i=0}^{N-1} (y_i - x_i)^2 . \tag{2.38}$$

The Intersection over Union (IoU) is used for images and describes the ratio of the overlapping area of prediction x and target y to their combined area as displayed in Fig. 2.10. The IoU yields values in the interval $[0, 1]$

$$\mathcal{L}_{IOU} = \frac{x \cap y}{x \cup y}. \tag{2.39}$$

Network and Layer Types

Multi-layer Perceptron

A Multi-layer Perceptron (MLP) is one of the most basic neural networks. It consists of at least three layers of perceptrons that are fully connected in a feedforward style. The layers include an input, an output layer, and hidden layers in between. Each perceptron, i.e., each node, is fully connected to its predecessors and successors and has non-linear activation functions except for the nodes in the input layer.

Convolutional Neural Networks

CNNs are mainly used for image processing because of their ability to efficiently extract feature information from regions using translation-invariant convolutional filters, i.e., kernels while maintaining the spatial arrangement. The kernels have uneven sizes and are kept small to extract fine information. Typical kernel sizes are 3x3 or 5x5. A discussion of kernel sizes can be found in [140]. The weights of the kernel are learned during training. The information of different regions is successively associated on deeper levels while enriching the included features at the same time using more channels. Pooling layers are used in combination with the convolution layers to reduce the dimensions and finally obtain a latent representation of the input. Max pooling and average pooling are common pooling types. They take the

maximum or the average of the corresponding pooling window. The size of the pooling filter determines the factor of dimension reduction. A typical size is 2x2. This part of the network is the so-called encoder that condenses and encodes the information into the latent representation.

This latent representation can be used in fully connected networks like MLPs to, for example, get bounding box coordinates. When the output is also an image-like structure, e.g., for segmentation tasks where each pixel is to be classified, the output must have the input dimensions. The upscaling is achieved through transposed convolution layers in the decoder part [109]. The resulting network resembles a horizontal hourglass. Such networks are also called Fully Convolutional Networks (FCNs). To overcome deficits in recreating the correct spatial context, architectures like U-Net [128] have been proposed to keep connections between the downscaling and upscaling parts.

Graph Neural Networks

Graph Neural Networks (GNNs) are used for data that can be structured in graphs. Information is stored in nodes that are connected via edges. The edges contain information about the relations of the nodes. Homogeneous graphs have a fixed type of nodes and edges. Heterogeneous graphs, on the other hand, consist of different types of edges and nodes [151]. Message passing is used to exchange information and update the nodes.

3 Comprehensive Situational Awareness

The comprehensive situational awareness lays the foundation for informed decisions and the generation of safe maneuvers and trajectories. It is a necessary step towards successfully deploying automated vehicles in everyday traffic. This chapter gives an overview of the automated driving functions and their interconnections and dependencies in Sec. 3.1. Based on this, in Sec. 3.2, the uncertainties and potential blind spots in the context knowledge of a traffic scene are derived. The main difference between uncertainties and incomplete knowledge is that uncertainties can be quantified, while incomplete knowledge is difficult to identify and describe. Incomplete knowledge can be avoided and supplemented by experience or experts who gathered the experience. As there is a vast number of sources of uncertainties, only the most relevant are listed here. The section aims to balance scope, level of detail, and relevance. Sec. 3.3 then compares different environment representations. The section assesses the suitability of the environment representations for informed decision-making regarding incomplete knowledge about intentions and occluded traffic participants. The chapter's key points are given in Sec. 3.4.

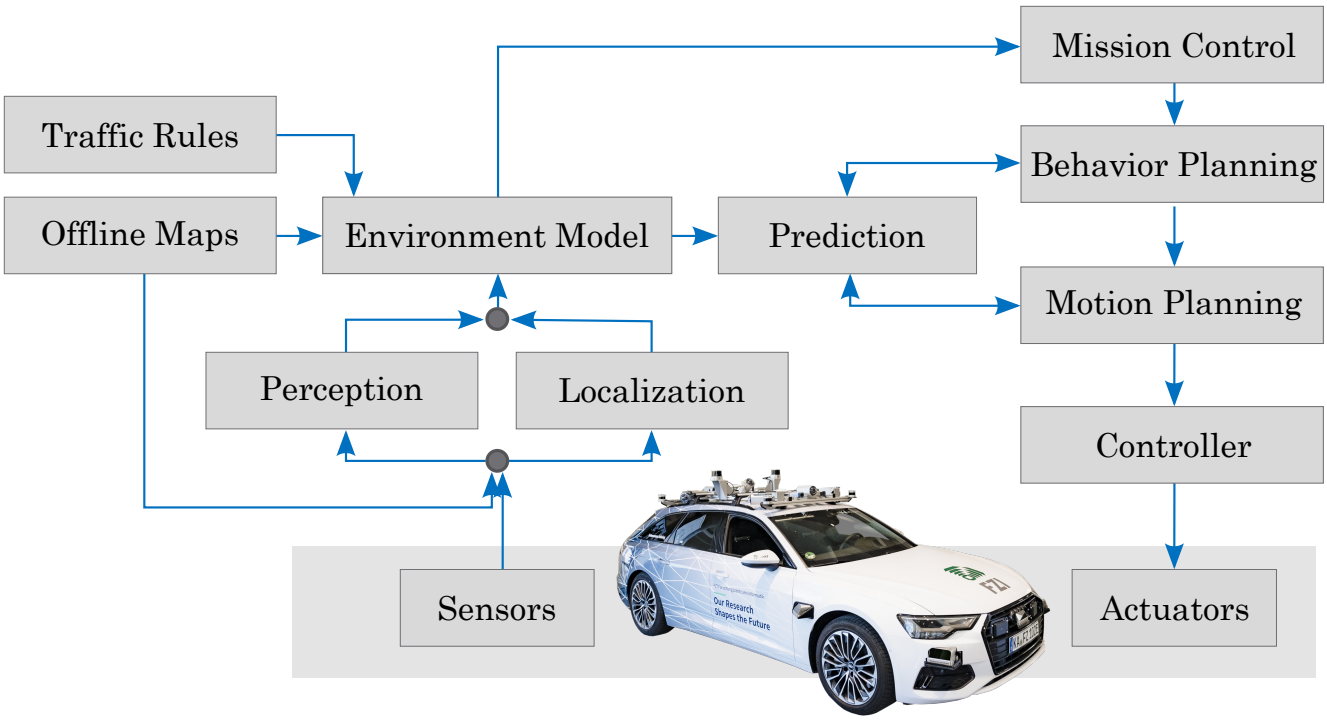


Figure 3.1: Architecture and components of an automated driving stack.

3.1 Automated Driving

Understanding sources of uncertainties and incomplete knowledge requires a deeper comprehension of the composition of automated driving functions and their interaction with the installed hardware, i.e., sensors and actuators. This section gives an overview, starting with the overall architecture shown in Fig. 3.1. In the following, the individual components are explained. Afterward, the flow of information and the interaction between the functions are addressed. The vehicle depicted in Fig. 3.1 is Cocar NextGen, an automated vehicle of the FZI [16].

Sensors - Sensors measure the vehicle's state (interoceptive) and perceive the environment around the vehicle (exteroceptive). Various sensors are available to observe the surroundings. Each sensor has its characteristics, limitations, advantages, and disadvantages. Automated vehicles are equipped with multiple modalities of sensors to overcome the shortcomings of one type of sensor.
Examples of exteroceptive sensors are Lidar, radio detection and ranging (Radar), cameras, stereo cameras, and ultrasonic sensors. Lidar sensors use millions of individual measurements of the surroundings to create 3D point clouds. Besides an accurate location of the points, the measurements also contain information about the intensity corresponding to the reflectivity of the hit material. While the majority of Lidar sensors are based on time of flight measurements to determine the distance of each point, the advances in manufacturing and computation capabilities enabled the production of Lidars based on frequency modulated continuous wave (FMCW) technology. These Lidars additionally determine the radial velocity of each point. Radars uses radio waves and the Doppler effect to also determine the distance and velocity of objects based on the radio echo. Radars are unaffected by weather that blocks visible light. However, the echo of radio waves strongly depends on the material. Thus, the Radars are often mounted hidden behind bumpers because the radio waves can pass through the plastic. Cameras capture colored images that provide much information and are the cheapest of the mentioned exteroceptive sensors. Moreover, as humans rely solely on vision while driving and the environment is designed accordingly, i.e., signs or road markings, cameras are a primary choice for assistant systems and automated driving.
The vehicle that has mainly been used to collect data and to perform evaluations and test drives for this thesis is Cocar (**Co**gnitive **Car**). Cocar is a test vehicle of the FZI. The vehicle is equipped with Lidars and cameras. Vehicle-to-X communication can also be interpreted as perceiving the environment by directly obtaining information from other vehicles or intelligent infrastructure.
The interoceptive sensors mainly include inertial measurement units (IMUs), global navigation satellite system (GNSS), and sensors for odometry. The odometry describes the vehicle's motion through its position and velocity relative to a starting pose. It is estimated through sensors of, in particular, wheel rotation speed, steering angle, and IMU. The IMU determines the linear and angular forces on the vehicle. By integrating the accelerations, the motion of the vehicle can be estimated. GNSS sensors like the navigational satellite timing and ranging - global positioning system (NAVSTAR GPS) provide global navigation information. Geo-referenced information is mainly used in combination with map data.

Map - As described in Sec. 2.2, maps are used to store static information. This comprises lanes, signs, road markings, or the information on whether the street is a one-way street. Traffic rules like speed limits or prioritized roads are also encoded on the map. Maps can be generated offline or online. This work uses offline-generated maps. They bring the advantage that information about non-visible areas is provided. Thus, offline maps strongly support foresighted decision-making. In online-generated maps, only the information that has already been observed is contained.

Traffic rules - The traffic rules greatly determine the behavior and motion of all traffic participants. They form a set of binding rules for safe travel and regulate the right-of-way at intersections. Traffic rules are

different for different countries. Because the rules are known, and the traffic participants are assumed to obey the rules, the range of potential actions is limited, and more accurate long-term predictions can be made. Examples of traffic rules are speed limits, the side on which the vehicles drive, or the right-of-way. They also regulate the width of the road and the distances to be kept between traffic participants. Hence, they provide an important and valuable source of knowledge.

Perception - In the perception module, data from the exteroceptive sensors is processed. The individual processing depends on the sensor and the desired information to be extracted. This includes filtering, segmentation, detection, tracking, and classification. A huge challenge is the association of single measurements or detections of an object to so-called object tracks. The objects are measured at multiple points in time from one or multiple sensors. In the tracking process, the single measurements are associated to form consistent tracks of an object. The classification can be achieved through direct image classification, attributes like dimensions, or how the object moves. The association of measurements to a specific object can be supported by feature recognition, like the vehicle color or type. The perception also determines the free space, where no object is assumed to be located. Furthermore, features like road markings or signs are also detected to support the scene understanding or localization. The quality and accuracy of the perception strongly depend on the sensor measurements and limitations. This is further discussed in Sec. 3.2. The information obtained in the perception is crucial for building a belief about the state of the environment. A higher accuracy of the perceived environment leads to more certain beliefs and, thus, to a better driving experience. Inaccuracies or unavailable information must be countered with cautious driving maneuvers.

Localization - The ego vehicle uses its sensors to localize itself. Localization is, therefore, often referred to as ego localization to differentiate it from the localization of other objects. In this work, the localization of the ego vehicle is simply referred to as localization. The determination of the position of other objects is assigned to the perception. There are different strategies to locate the ego vehicle. It is distinguished between local and global methods. Local localization is often achieved using odometry. Interoceptive sensors can provide odometry information, but there also is visual and Lidar based odometry. Two of the most prominent methods for global localization are GNSS and Simultaneous Localization and Mapping (SLAM) algorithms. Similar to the odometry, SLAM can be performed using visual or Lidar information. Global and local localization are often used for mutual support to increase the localization quality. Through the movement of the ego vehicle, sensor measurements do not have the same reference point for different points in time. Therefore, the sensor measurements or the perception information from different times are transferred into a common coordinate frame using the localization of the ego vehicle. In general, the odometry frame is used here. Because of this transformation, the localization quality also directly influences the perception quality and accuracy.

Environment Model - The environment model incorporates and accumulates the information gathered by the perception and enriches it with map information. As the environment model forms the core module for scene understanding, it is discussed in detail in Sec. 3.3. The environment model contains information about objects, like their positions, sizes, or velocities. It can also include semantic information like mappings of objects to their lanes. Additionally, the environment model is used by the scene understanding module to estimate the intentions of road users. Intentions are high-level assumptions that strongly influence the road users' future motion. Turn decisions are an example of intentions, i.e., what direction the road user will follow. The scene understanding module also explicitly tracks areas where it received no information. This is essential to account for potentially occluded traffic participants in these areas. The model is typically built over time, meaning that beliefs about the state of the environment are propagated and updated continuously with new observations.

Prediction - The prediction module determines potential developments in the present traffic situation based on the belief of the state of the environment. It, therefore, highly depends on the accuracy and quality of the information provided. Different kinematic models can be used to simulate the motion of objects. Their application depends on the assumptions made and on the classification and intention of the object. A basic assumption is the constant velocity, acceleration, or yaw rate assumption, which can be used if no further information is given. It represents an object that continues its current motion but is often only accurate for short time horizons. More sophisticated approaches, for example, use the so-called bicycle model for vehicles and bicycles to approximate their lateral motion better. The Intelligent Driver Model (IDM) as proposed in [145] is often used to predict the longitudinal motion. The car-following model is widely applied and uses the current and desired speed of a target vehicle as well as the speed and distance to a vehicle in front to calculate a realistic acceleration for the target vehicle. Map information and traffic rules are leveraged to increase the prediction accuracy further. In order to obtain accurate long-term predictions, the intentions and interactions of the objects are considered. As the intentions and behavior of other traffic participants are not perfectly known, the state of the environment can evolve differently. Thus, there will be multiple predictions for the same initial state of the environment. The predictions are assigned different likelihoods depending on the respective likelihood for the behavior in the belief about the state of the environment.

Mission control - The purpose of mission control is the overarching scheduling of operations and coordination of tasks. This ranges from specifying a target location by the user to complex dispatching for automated cargo or passenger transport. The mission control determines the start and end times and the route based on current traffic and weather conditions. The planning horizon lasts until all pending missions in the queue are completed.

Behavior planning - Behavior planning is responsible for high-level decision-making. The resulting behaviors are expressed through concrete maneuvers. For example, lane changes, overtaking, and right-of-way decisions are planned here. Maneuvers can be described on different levels, i.e., an overtaking maneuver can be seen as one or a sequence of lane-changing maneuvers. Since the behaviors have a long-term impact, behavior planning has a long planning horizon. The goal is to have the automated vehicle follow the route set by the mission control while complying with traffic rules by making foresighted and comprehensible maneuver decisions. This requires accurate predictions of the traffic scene. As the behavior of the ego vehicle and the other traffic participant mutually influence each other, the prediction and planning interact. This effect increases with longer planning and prediction horizons. For this reason, prediction and planning should be done simultaneously. Methods like POMDPs enable adaptive and uncertainty-aware behavior planning. However, these methods are computationally expensive. Therefore, in practice, planning and prediction are often conducted sequentially. In the behavior planning process, the motion of the traffic participants is represented more coarsely than in the motion planning step. So, higher planning horizons can be achieved with the same computational effort. The behavior might even be semantic, where relations between vehicles are not given in cartesian coordinates or distances but in terms like beside or in front. However, as more than these coarse descriptions are needed by the vehicle controller to be executed, the motion planning module has to calculate feasible trajectories. In order to align the high-level goals described by the maneuvers and the motion planning, specific longitudinal and lateral requirements are set with the maneuvers that need to be fulfilled by the subsequent motion planning module. The maneuvers must account for the ego vehicle's current state and allow feasible trajectories to be planned. When looking at the example of a right-of-way situation, where the ego vehicle has to stop for another vehicle on a prioritized road, the behavior of the ego vehicle would be to stop and give way to the other vehicle. The performed maneuver is stopping in front of the stopline. The boundary conditions for the motion planning are where and when the vehicle has to come to a halt.

Motion Planning - In the motion planning unit, trajectories for the automated vehicle are determined based on the previously planned maneuvers. Motion planning, therefore, often is also referred to as trajectory planning. The trajectories are planned using search or optimization algorithms. The algorithms determine an optimal trajectory that fulfills optimization criteria and constraints imposed through dynamic limitations, other traffic participants, and objects in the surroundings. Examples of the optimization criteria are given in Sec. 2.7. Here, further examples of the constraints or boundary conditions are also described.

Controller - The controller generates commands for the actuators with a high frequency based on the deviation from the vehicle to the planned trajectory.

Actuators - The actuators include all units that control the vehicle's motion. The steering angle determines the lateral motion. The longitudinal control is distributed over the powertrain and brake system components. Lights and horns are used to indicate turns and to inform about unusual or dangerous situations.

The components form a closed-loop system with the vehicle and the environment. The perception and the localization process the sensor data. The environment model is built in combination with additional knowledge sources of the map and traffic rules. Based on the belief of the state of the environment, predictions are made, and the behavior and motion of the vehicle are determined with respect to the overall goal set by the mission planning. The controller is responsible for keeping the vehicle following the planned trajectory. It can be seen that information that has not been extracted in the sensor processing and perception is missing in the following components. If this fact is not considered or compensated in the subsequent components, it may lead to incorrect or dangerous behavior. The comprehensive situational awareness described in this chapter allows to determine missing information. From this, it can be deduced that knowledge about missing information is as essential as the available information for making informed decisions. This fact directly influenced the choice and the focus of the approaches that are presented in the following chapters.

The level of abstraction in the information changes during the processing pipeline. Sensor data shows the lowest level of abstraction. It consists of a large amount of data, where much information does not contribute to the vehicle's behavior. The abstraction condenses the required information for efficient processing. For example, the single measurements of an object in a point cloud are combined to obtain an object description using only its position and dimensions. Thereby, the amount of total information is reduced. The challenge is to extract and know the correct information and to keep valuable data. This is discussed in the next section (Sec. 3.2). The environment model's representation and the environment's data abstraction levels are then elaborated on in Sec. 3.3.

[14] provides a brief overview of the stack for automated driving that was used and extended in this work.

3.2 Uncertainties

3.2.1 Distinguishing Uncertainties

The nature of uncertainties strongly varies with the tasks and how they are regarded. Knowing the type of uncertainty is salient to finding suitable coping strategies. In the decision-making domain, several methodologies were proposed in psychological studies when researching human decision-making. One of them is the methodology to distinguish between epistemic and aleatoric uncertainties. The following description is mainly derived from [58]. The reader is recommended to the article for more information and other methodologies used in researching human decision-making. This thesis shows how the methodology can also be applied to decision-making under uncertainties for automated vehicles.

Epistemic and aleatory uncertainties are differentiated by the attribution to the source, their coping strategies, the focus of the prediction, and the interpretation of the resulting probability. The authors of [58] emphasize that these forms are not mutually exclusive, meaning that characteristics of both forms can be present.

The term aleatoric is derived from the Latin word aleator, someone who rolls the dice. Aleatoric uncertainty results from stochastic processes that can not be predicted precisely, for example, flipping a coin. It is aleatoric uncertainty when predicting probability distributions of relative frequencies of several classes from repeated experiments. It, therefore, cannot be reduced due to the stochastic nature of the underlying processes.

Epistemology is the theory and study of knowledge. The epistemic uncertainty is attributed to model inaccuracies or insufficiencies due to missing knowledge or expertise. Epistemic uncertainty is the uncertainty in the prediction for single cases, whether they are true or not. It can be reduced by extending the model by, e.g., searching for patterns. For example, the likelihood of giving the correct answer to a specific item is epistemic uncertainty, and the proportion of times the correct answer was given is aleatory uncertainty.

The methodology is also used in other domains like machine learning. See, for example, [61]. Epistemic uncertainty analogously refers to the uncertainty of the trained model. The main reason is the need for more training data or underrepresented classes in a data set. It can be quantified via the corresponding variances. Hence, it is possible to add additional data to reduce epistemic uncertainty and to guide the training process in a targeted way. Aleatoric uncertainty is attributed to the randomness of the data. It can be decreased by increasing the quality of the data elements, for example, by increasing the resolution of images.

In the context of this thesis, the methodology and differentiation of uncertainty are essential to determine a suitable coping strategy. For aleatoric uncertainties, a suitable expression and quantification needs to be found. This includes, for example, uncertainty in estimating the position and speed of a vehicle. For epistemic uncertainties, the existing models need to be extended, or new models and approaches to describe the uncertainties need to be found. This is, for example, shown in Sec. 4.1 for intentions and in Sec. 4.2 for occluded areas. The uncertainties arising in the automated driving functions and tasks are presented below.

3.2.2 Measurements

Measurements are the first source of inaccuracies. They are starting the chain of uncertainty that then continues through all the following components. The measurement uncertainties can mainly be seen as aleatoric. Due to the underlying physical effects of the measurement processes, noise and inaccuracies cannot be avoided. This applies to all kinds of sensors. The upside is that most deviations can be represented conclusively and realistically using statistical models and probability distributions like Gaussian distributions [139]. The measurement accuracy of a Lidar sensor can, for example, be given using a Poisson distribution with its mean value and confidence.

Besides physical effects, many more factors influence the accuracy of measurements. Not only exteroceptive sensors are affected, but also interoceptive sensors such as odometry sensors. While physical effects are often predictable and, as mentioned, well represented, effects like distortion through water drops that are randomly spread on the lens of a camera are not. In general, weather conditions can severely influence measurement accuracy and add additional interferences. The influence can be reduced by using multiple modalities of sensors, surveilling the sensors themselves, and adjusting the vehicle's behavior. For example, if objects might not be detected, hidden objects have to be assumed in the higher level of decision-making.

Systematic errors are inflicted through miscalibrated sensors. Therefore, calibration must be carried out with the utmost precision to avoid these errors.

This work includes measurement uncertainties in the uncertainty-aware state information of objects by using a Gaussian distributed position and orientation uncertainty. The uncertainty in the shape and dimension of the object is eliminated by overestimation for simplicity as described in Sec. 2.1.2. However, it could be integrated straightforwardly. The uncertainty-aware state information is assumed to be provided by a perception module.

3.2.3 Sensor Limitations

Whereas the last section described errors in measurements, which can be seen as primarily aleatoric uncertainties, this section covers the limitations of sensors. It can be interpreted as an epistemic source of uncertainties. Sensor limitations result from the maximum range of sensors, objects in the FOV that occlude the area behind them, and the sensor setup in total. Further reasons are effects that limit the sensors' ability to perceive the environment inside its FOV. An example of this is a camera facing the sun. Due to overexposure, the resulting images cannot be used in the further process.

The maximum range is determined by the physical and mechanical principles that the sensor relies on. New sensor generations push the maximum usable range further and further. The improved resolution and the reduced noise in camera images allow to detect objects that are further away more accurately.

Occlusions can not be avoided in real-world conditions. Whereas occlusions through objects in close range can be compensated by a sensor setup that covers the areas around the vehicle from different angles, some occlusions can not be avoided. Examples of these are occlusions because of far distance, elevations and the slope of the road, or occlusions caused by big objects like houses. Therefore, it is crucial to be aware of occluded areas and the potential traffic participants located within them. A prominent example is an intersection in a rural area, where the prioritized lanes are occluded for the ego vehicle. In Sec. 2.8, it is shown how the FOV is described in the form of an occupancy grid or by employing a vector of

polygons and how the FOV can be obtained through sensor measurements. The example mentioned above of overexposure in camera images can also be expressed by restricting the FOV accordingly.

The epistemic nature of the uncertainties introduced by sensor limitations requires new models, algorithms, and concepts to be researched to cope with the uncertainties in a manageable, quantifiable way. Then, they can be taken into account in the components of scene understanding and behavior planning. Therefore, one emphasis of this work is set on dealing with occlusions and, in particular, with potentially occluded traffic participants.

3.2.4 Object Detection and Tracking

Object detection and tracking are complex tasks because single objects must be extracted from noisy measurements. First, there is the problem of track and data association. When the single measurements are temporally and spatially associated, assignment errors might occur by mapping a measurement to the wrong object. This especially happens when detections are close together. Algorithms like the Hungarian algorithm [99] try to find the most probable combination of detections to tracks. Re-identification methods are designed to circumvent the problem by finding unique features to reidentify an object over time. In case of problems in the data association, tracks are lost, or the identifier of the track and the correlated object changes. In the scene interpretation, this results in an object suddenly disappearing and another object that appears in return. This can lead to faulty assumptions as objects can not simply disappear and, therefore, have to be assumed to be occluded.

Even if the association problem is solved, there is still the problem that not all required information can be measured directly. While Radar or 4D-Lidar can measure the position and radial velocity, the turning rate has to be derived from other measurements. Bayesian filters like the Kalman filter perform very well at low computational overhead. Though, their performance also depends on the accuracy of the measurement. Errors in the sensor data segmentation and different viewing angles of objects as they move by can result in inaccurate size, shape, and pose estimations. This leads to inaccurate estimations of velocities and accelerations. In combination with the motion patterns, the shape can be used to infer the classification of the object. Based on the assumptions, misclassifications can appear. When cameras are used, the classification can be conducted directly on the images, e.g., using neural networks like YOLO [126]. However, the classificator still cannot provide absolute certainty even though machine learning approaches made massive progress in recent years. Machine learning based approaches are also used for segmentation and detection, e.g. [150]. No guarantee can be given for neural networks that no false positives or false negatives will occur.

As mentioned above, in this work it is assumed to obtain uncertainty-aware information from a perception module. Errors introduced through effects like wrong track association are included in the uncertainty values.

3.2.5 Localization

The uncertainty in the ego vehicle's position, orientation, and velocity has a significant impact on the overall system. It is, therefore, important to have accurate estimations of these values, including their confidence. Depending on the localization method, uncertainties arise from different sources. GPS-based localization performance suffers from poor GPS reception or multiple echos due to high-rise buildings, bridges, or tunnels. SLAM-based approaches deteriorate with a reduced number or poor quality of features

available [11]. The same accounts for Lidar or camera-based odometry estimations. Furthermore, the sensors are subject to the issues mentioned in Sec. 3.2.2. By combining different modalities, the overall localization quality can be made robust against single sources of uncertainties. As the ego vehicle's state is also part of the state of the environment, its state is also given with Gaussian distributed uncertainties.

3.2.6 Map

Maps are often regarded as a static storage of information. However, offline-generated maps might have changed between the recording and the current point in time. So, it has become outdated and needs to be updated. Changes can be temporary due to construction sites or permanent because of the reconstruction of the course of the road. An important task is to identify and update outdated information. The stored information is also subject to inaccuracies because of insufficient accuracy of global position during map generation or miscalibrated sensors. This is especially true for online-generated maps but also affects offline-generated ones. In this work, it is assumed that the information provided in the map is accurate. The identification of deprecated information is not part of this thesis.

3.2.7 Scene Understanding and Prediction

During the scene understanding task, the information gathered from the previous components is used to infer high-level information like lane mappings or intentions of traffic participants. The uncertainties of the previous components accumulate. This can lead to faulty interpretations of the context. If the position and orientation are inaccurate, the traffic participant can be associated with the wrong lane, leading to false route estimations. Errors in the classification result in false assumptions about the traffic participant's motion patterns and intentions.

Furthermore, the areas where no information is available must also be included to understand the scene context thoroughly. Potential occluded vehicles approaching on a not perceivable prioritized lane are crucial to be considered. As both the intentions and occluded traffic participants can not be measured directly, the second focus in this thesis was set on estimating road users' intentions. Thereby, each intention is given a likelihood.

A subpart of intention estimation is the detection of uncompliant behavior, i.e., intentions that do not comply with the traffic rules. This is not included explicitly in the latter presented approaches to deal with uncertainties arising from unknown intentions. Additionally, the concept presented in Sec. 4.1 shows the intention estimation on the example of route intentions. However, the concept is not limited to these. Other intentions, as well as uncompliant behaviors, could also be included.

3.2.8 Planning and Execution

In the planning process, the potential inaccuracies of the execution have to be taken into account. This means that it has to be expected that the vehicle controller cannot precisely follow the planned trajectory. Therefore, deviations need to be considered. This is achieved by propagating the current uncertainty-aware state accordingly, e.g., using an Extended Kalman Filter.

An extreme case is the failure of an actuator. If the function is degraded, the planning unit has to adapt to the new operating range. A complete failure results in a sudden emergency stop. The author evaluated

this in the publicly funded project SafeADArchitect. However, functional safety and degrading strategies are not the focus of this thesis.

Further uncertainties lay in the choice of the homotopy class, meaning that multiple subspaces of solutions are available. The point where the optimal class is revealed might lay outside of the scope of the planning unit. An example of the choice of the homotopy class is the decision on which side to pass the obstacle. Solutions for foresighted decision-making and trajectory planning are suggested in Sec. 5.

3.2.9 Vehicle-to-X

Vehicle-to-X communication allows automated vehicles to obtain information from other sources. This can be either from other traffic participants or local infrastructure. However, the received information could be invalid, deprecated, or sent from untrustworthy sources. Safety and security concepts for vehicle-to-X communication seek to prevent its misuse. It is mentioned here for completeness but is not further regarded in this thesis. Instead, the focus lies on the information the ego vehicle can gather itself. The concepts described, for example, in Sec. 4.3 could be extended to not reason about occluded objects but to verify if the incoming information is plausible.

3.2.10 Further Sources

There are further sources of errors and uncertainties that cannot be attributed to a single component. In the following, a non-exhaustive list is presented

- application of neural networks,
- human errors during development and validation,
- hardware errors and failure,
- rounding errors due to precision,
- race conditions, latencies, and timing issues,
- violation of real-time constraints,
- adversarial attacks.

The safe application of neural networks is described in Sec. 5.3. The other points constitute research questions that need to be regarded independently of the present work.

3.3 Environment Model

The environment model accumulates all information about the environment from different sources. The model has to contain all necessary information for predicting the environment, making informed maneuver decisions, and planning safe and feasible trajectories. Thus, the following requirements are posed for the environment model. The environment model

- is complete, meaning that all successive components can obtain the required information from the model and do not require additional sources.
- includes dynamic objects.
- represents free space and occluded areas to cope with occlusions.
- includes uncertainties in the state of objects.
- distinguishes between static and dynamic information for efficient processing and storage.
- accumulates information over time to build an accurate belief of the current situation.

An environment model consisting of three parts is proposed to fulfill these requirements. The first part is a map that stores static information. The second part comprises objects in an object list, representing each object by its state and attributes. Finally, occluded areas are represented by a set-based approach. Free space is implicitly given through areas without objects or occluded areas. The three parts allow the complete description of a traffic scene.

Fig. 3.2 shows the components of the environment model for an example situation. The ego vehicle (blue) approaches an intersection with limited visibility. From the right, a vehicle (red) that is not visible for the ego vehicle is approaching on a prioritized lane. The parking vehicles are depicted as grey boxes. The underlying road network is drawn with blue lines. In the following, the separate parts are discussed, and explanations for the choice of representation are given.

3.3.1 Map

As mentioned in Sec. 2.2, the map provides storage for information that is assumed to be static. For foresighted driving, the map is indispensable as the automated vehicle can use information from locations far away from its sensor range and from locations that are occluded for the vehicle. The following information is stored in the map of the proposed environment model:

- The **geometry** of the road includes the course of the single lanes, road boundaries, markings, and the position of stop lines.
- **Traffic rules** provide information about the right-of-way, one-way streets, no overtaking zones, turn restrictions, and speed limits. This information can be inferred from the corresponding traffic signs.
- **Traffic lights** in contrast to traffic signs can change their status. For the automated vehicle, it is essential to know which traffic light belongs to which signal group to determine what lanes have green lights simultaneously.
- **Sidewalks** and **pedestrian crossings** describe the areas where pedestrians are assumed to walk.

- **Static information**, e.g., buildings, monuments, parking spaces, and so on, are used for making assumptions about the ego agent's future FOV. For anticipatory maneuver planning on occluded intersections, it is vital to consider when to expect to have insights into occluded areas.

As the focus of this thesis is not localization, landmarks are left out for the sake of simplicity. However, this information would also have to be included for localization tasks to obtain an all-encompassing model. Fig. 3.2b shows the semantic separation of the scene in Fig. 3.2a and Fig. 3.2c depicts the corresponding road network. As an example, only the relations and traffic rules are added for the lane of the ego vehicle that is coming from the bottom to keep a readable illustration. Further information, like the average traffic density on the occluded prioritized lane, can be added to support behavior planning.

In [6], a comparison of map formats has been conducted. The open-source map formats OpenDRIVE[1], Road XML[2], MAP and Lanelets [45] were compared for distributed automated valet parking systems. The evaluation criteria were the suitability to represent different information in general, information used for parking, information used for on-road driving, and whether 2D or 3D information is stored. The Lanelet format was chosen because of the modeling freedom. Arbitrary attributes and relations can be defined, and the elements of the Lanelet map, namely points, line strings, Lanelets, areas, and regulatory elements, allow the representation of all necessary components. The results of the comparison can be transferred to driving in everyday situations. Therefore, Lanelets are used in this thesis. A short introduction is given in Sec. 2.2.

3.3.2 States of Objects and Regulatory Elements

When representing the perceivable part of the environment, a tradeoff between accuracy and abstraction has to be found. The human observer sees the environment and implicitly separates it into single objects, such as buildings or vehicles driving on the road, and filters irrelevant information. The abstract representation is sufficient for the driver to make a decision. For automated vehicles, the same applies.

Comparisons of representations were conducted in [6] for the context of a distributed automated valet parking system and in [132] for an advanced driver assistance system. The latter only considers areas reachable for the ego vehicle to describe the free space. The results in [6] can be generalized to a wider scope of automated driving. Low-level representations like stixel, point clouds, and images can be left out because a sensor-agnostic presentation is targeted. The so-called object list is the most compact and suitable representation of objects. Both aforementioned approaches use obstacle lists to represent objects. As in this thesis, they are prioritized over grid-based approaches for object representation because of the efficient summarization of information and the arbitrary extension with attributes.

In Fig. 3.2d, the scene's traffic participants are depicted as part of the object list. Although the grey vehicles are parking and can be considered static for the moment, they are included in the object list, too. The objects are described by their position, orientation, and the derivatives thereof. Additionally, the dimension, classification, and intentions are stored. The unclear route intention of the vehicle from the right is indicated with three red lines. If trees and buildings are not already part of the map, they could also be included in the object list. Also, the state of traffic lights is included in the current state of the environment. The resulting representation is described in detail in Sec. 2.1.2.

[1] www.asam.net/standards/detail/opendrive
[2] www.road-xml.org

3.3.3 Occluded Areas

The last component for a complete description of the environment is the representation of occluded areas. These are relevant for making assumptions about occluded traffic participants and making safe decisions.

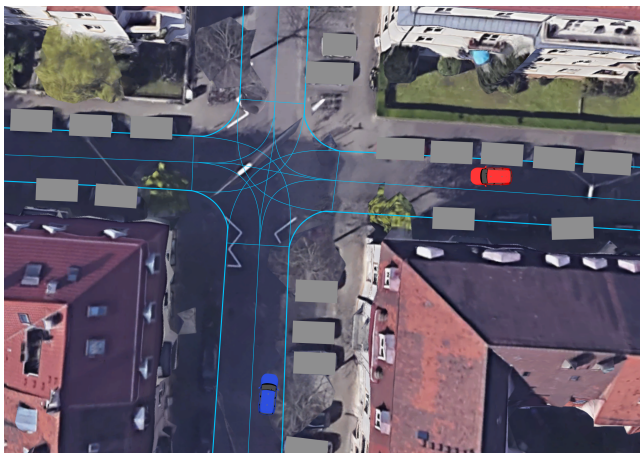
Occluded areas can be realized in various representations, i.e., set-based approaches, occupancy grids, and particles. Particles represent the areas by hypotheses about the objects in the area itself, like in [155] or [142]. The particles have individual states that allow inference of detailed information about the potentially occluded traffic participant. Sec. 4.3 shows how particles are used to infer the presence of occluded traffic participants.

Occupancy grids divide the environment into two-dimensional grids. Coué et al. [51] introduced Bayesian occupancy filter-based occupancy grids that track the probability of being occupied with the previously made observations. This results in a probabilistic description, where a value of zero indicates a cell is empty, and a value of one is seen as an occupied cell. Occlusions can be derived from occupation probabilities of around 0.5 because no measurement or ambiguous measurements are available that confirm or negate that the cell is free or occupied. This approach is, e.g., used by Hörmann et al. in [76] to predict occluded areas in an intersection scenario with limited visibility.

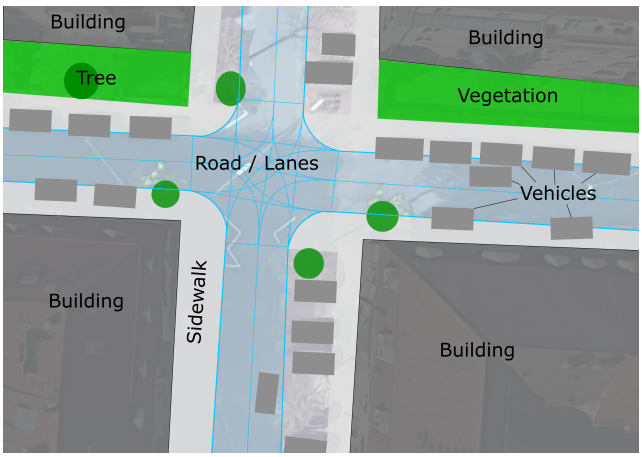
Set-based approaches summarize larger sections of the state space and thus reduce the computational requirements. Thereby, geometric sections, i.e., areas, as well as velocity ranges or both, can be combined in one set. [98], [116] and [149] use this representation. In Sec. 4.2, a set-based approach to track occluded areas is presented. The approach is characterized by using two different types of sets. They use different motion patterns to mimic the motion of the targeted potentially occluded traffic participant. The first is the oriented set with oriented motions along lanes, which accounts for the limitations of areas where vehicles are supposed to travel. The second type allows free motion at lower velocities, is not bound to roads, and aims to track occluded areas where pedestrians can be expected. Fig. 3.2e shows the current FOV in grey, non-road bound sets in blue, and oriented sets in orange.

Summary

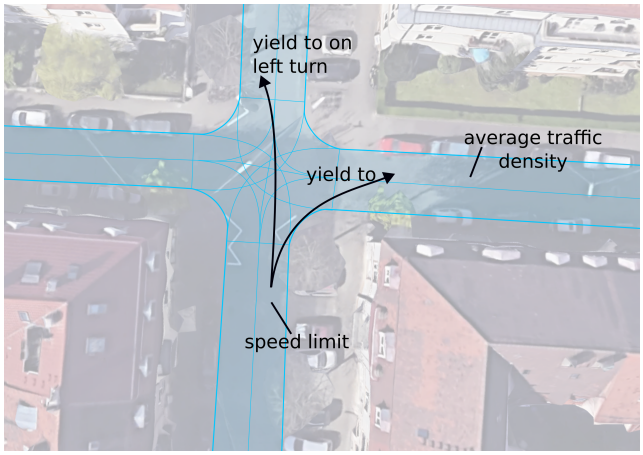
In total, the proposed environment model is complete by including all relevant information. Dynamic objects are included as object lists, and occluded areas are included through a dual set-based approach. Uncertainties are included in the state description of the objects themselves. The information can be accumulated over time by successively estimating the intentions of visible road users and by tracking areas where occluded traffic participants can be expected.



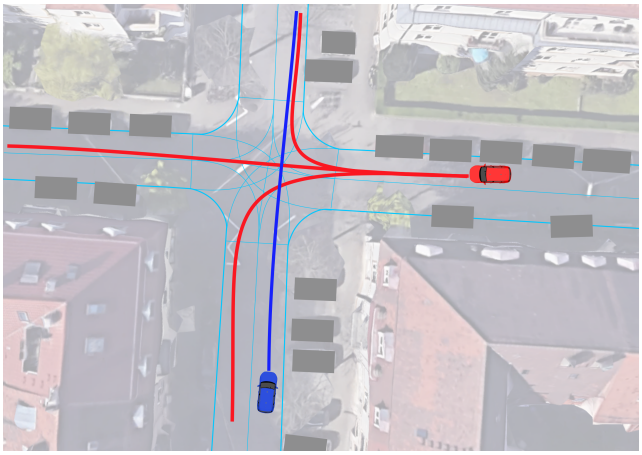
(a) Traffic scene



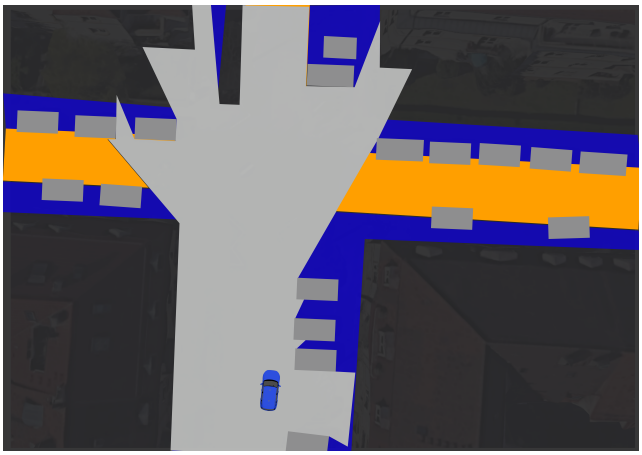
(b) Semantic map information



(c) Road network information



(d) Objects



(e) FOV, visible and not visible areas

Figure 3.2: Traffic scene from bird's eye perspective. The ego vehicle (blue) approaches from the bottom. A vehicle (red) is coming from the right on a prioritized lane. Other vehicles (grey) are parked along the road. (a) Traffic scene from an all-knowing perspective. (b) Semantic segmentation of map knowledge in the traffic scene. (c) Information about the road network is stored in a map with example relations of the lane from the bottom to other lanes and additional information that can be stored in the map. (d) Objects and a depiction of the unknown route intentions of the red vehicle. (e) FOV of the ego vehicle in grey. Road-bound occluded areas are depicted in orange and non-road-bound occluded areas in blue. [Background image: @ Google Maps, 2019, location: Karlsruhe, Germany, accessed: 24 January 2019]

3.4 Chapter Conclusion

To summarize the chapter, the addressed core points are listed in the following:

- For finding incomplete knowledge in automated driving functions, each component's functionality and tasks must be known. Therefore, the components and architecture of automated vehicles were analyzed.
- To classify and become aware of uncertainties that may arise, a methodology of uncertainties was presented, and an introspection of the sources of uncertainties in the components of the automated vehicle was given.
- The focus is laid on uncertainties that can not be measured directly but have to be inferred from context knowledge, namely unclear intentions and potentially occluded objects.
- Lastly, an environment model for comprehensive situational awareness was presented that consists of a map for static information, object lists for dynamic objects including dynamic regulatory elements like traffic lights, and a dual set-based approach for representing and tracking occluded areas.
- The chapter answers the first research question of how a comprehensive awareness of traffic situations can be achieved and how missing knowledge about the situation can be recognized.

4 Derivation of Knowledge and Building the Belief

The previous chapter outlined the uncertainties and the origins of incomplete knowledge in driving situations and how these can be described in the environment model. It was also mentioned that this work focuses on incomplete knowledge about other traffic participants' intentions and potentially occluded traffic participants. The state of the environment is thereby based on the proposed environment model. The uncertainty-aware assessment of the state of the environment is called belief. To obtain the most accurate belief, all available context information must be leveraged spatially and temporally. The resulting belief encodes uncertainties about all information of the environment model, including visible as well as occluded traffic participants.

The components of the automated driving stack, which are essentially considered here, are highlighted in color in Fig. 4.1. Building the belief takes the information from the perception and localization as input. The belief is enriched by inferred high-level information. Additionally, it uses map data and traffic rules. For a continuous situation assessment, a prediction module predicts the last estimation to the current point before it is updated with new measurements. A prediction module is also used to estimate future developments.

For building the belief of the state of the environment, this chapter first presents an approach for estimating the intentions of traffic participants (Sec. 4.1). Afterward, the tracking of areas where the potentially occluded traffic participants can be located is presented in Sec. 4.2. Based on the built belief about occluded areas, information about the presence of occluded road users is inferred in Sec. 4.3. Since prediction is essential for the propagation of the belief, two prediction modules are presented in this chapter. The prediction modules thereby focus on different use cases. On the one hand, a model-based prediction is presented in Sec. 4.1.3 that targets inner city scenarios like intersections. This prediction model is used in the model-based approaches presented in this thesis. On the other hand, a machine learning-based approach using CNNs that uses a stepwise prediction, allowing the network to be applicable in environments with occlusions, is introduced in Sec. 4.4. The network targets highway-like scenarios.

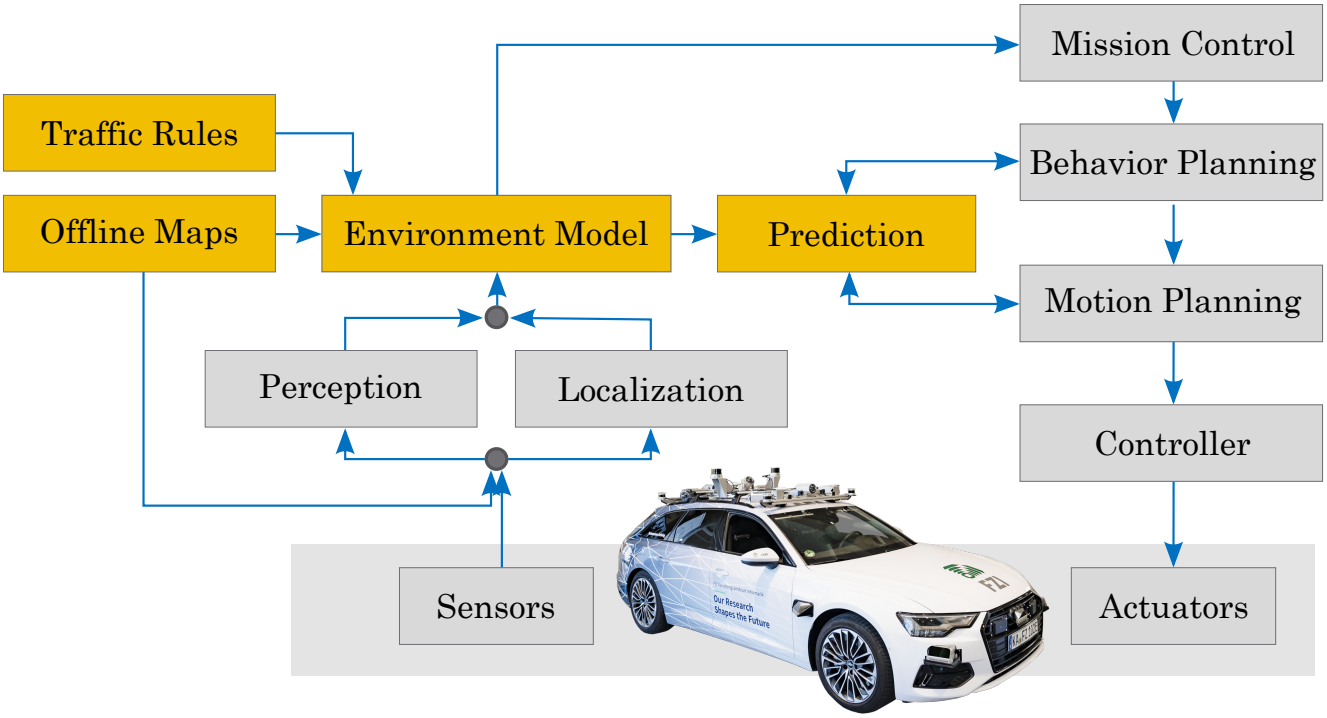


Figure 4.1: The components of the automated driving stack which are essentially involved in building the belief of the actual state of the environment.

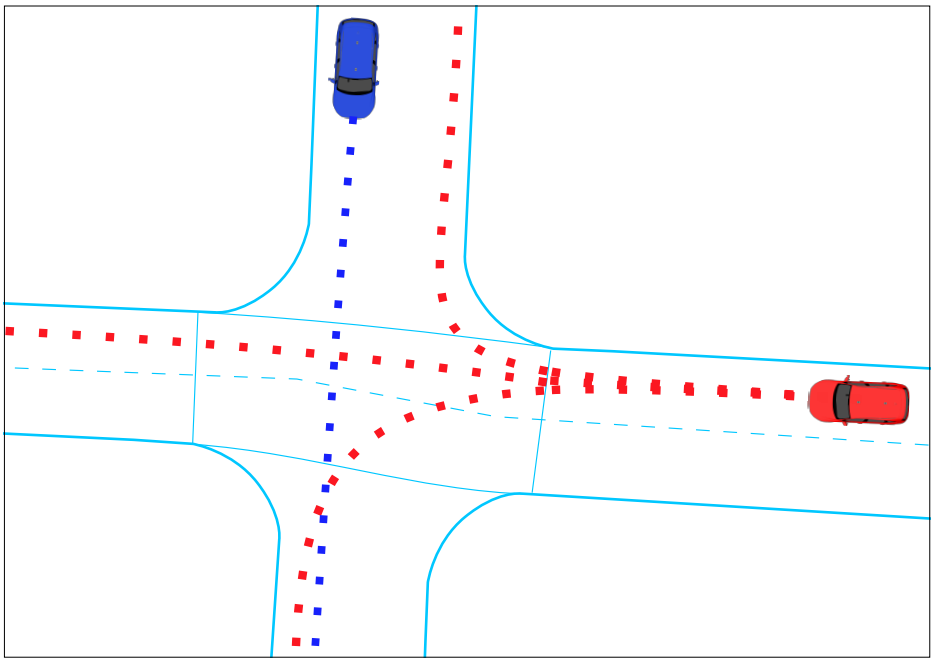


Figure 4.2: The ego vehicle in blue estimates the turn intention of the red vehicle on the prioritized road.

4.1 Intention Estimation through Situation Replay

Without relying on explicit communication, the intentions of other traffic participants must be estimated from noisy and partially inaccurate observations of, e.g., positions and velocities. Intentions are manifold and can be expressed on different levels of abstraction. For example, specific target locations can be seen as intentions. Also, the microscopic behavior to reach a goal, like stopping and turning, can be interpreted as intention. The concept presented in the following is applicable to any representation of intentions. This section shows the route intention estimation of traffic participants as an example application. It was presented in [5] and might contain verbatim quotes.

The concept is based on comparing a scene's expected development to its actual development. The traffic scene, therefore, is replayed from a past point in time. It is assumed that the actual intentions of the traffic participants are unknown at the beginning of the replayed sequence. Thus, the scene is predicted from the start of the replayed sequence for all intentions of the traffic participants. The better the predicted behavior for that intention matches the actual motion, the more likely the intention is. It is motivated by human reasoning that assumes a particular behavior. Deviations from the assumed intention lead to rejection of the intention hypothesis because the traffic participant should then have reacted differently. Here, an HMM is applied to assess the most likely intention.

To leverage all context information, all kinds of traffic participants, like vehicles, pedestrians, and bicyclists, are considered, as well as regulatory elements like traffic signals and rules. Additionally, the combination of intentions of all traffic participants is estimated. This contrasts with the frequently used approach of independently estimating the individual intentions of different traffic participants. With the approach presented in the following, the composition of intentions that best match the current situation in total is obtained.

It should be noted that the approach is independent of the prediction module as long as the prediction module produces a variety of different scene developments. This feature also means that different prediction modules can be used. It brings the advantage that intention or situation-specific prediction modules could be used so that the concept generalizes to arbitrary situations and intentions.

Further advantages of the proposed approach are that no discretization of the continuous spaces is needed to learn the HMM. By using a situation replay from a past point in time, the number of traffic participants is known during the estimation. Continuous tracking approaches often suffer from a changing number of traffic participants over time, as the dimensions of the state spaces change over time. Here, the number of traffic participants to be considered is known because the estimation uses only information from past and current points in time. The intention estimate thus also allows for retrospective corrections.

An example situation is shown in Fig. 4.2. The ego vehicle in blue approaches an intersection, where it has to yield to another vehicle (red). The other vehicle's turn intention determines the ego vehicle's further motion. In case the vehicle turns right, the ego vehicle does not have to stop. If the other vehicle goes straight or turns left, the ego vehicle will decelerate and wait until it can cross the intersection.

4.1.1 Related Work

As mentioned above, intentions can be represented on various abstraction levels. Intention estimation is a part of the situation interpretation. It further includes the perception of traffic participants and reasoning about their state and motion. After estimating the intentions, future behaviors can be predicted. Lefèvre

et al. [105] distinguish between three categories: *Physics-based*, *Maneuver-based* and *Interaction-aware* approaches in their overview over traffic participant prediction.

Physics-based approaches use kinematic models to predict the motion of traffic participants. Each traffic participant is predicted independently. The approaches achieve high accuracy for short-term predictions. However, when the prediction horizon rises, the accuracy decreases because interactions with other traffic participants or background information, like the course of the road, are not regarded. Kinematic models like those described in [127] or [133] can be applied in Bayesian filters like the Kalman Filter [89] and its extended and unscented variations [86]. Despite the disadvantages for long-term motion forecasting, these filters are widely used because of their simplicity, computational performance, and accuracy of consistent and uncertainty-aware state estimation and prediction.

Maneuver-based approaches leverage information like road geometry and topology to estimate the intention of traffic participants and to achieve more accurate predictions for longer time horizons. A variety of methods are applied for maneuver-based estimations. These comprise HMMs [35, 153], Gaussian processes [144] or other regression approaches like support vector machines (SVMs) [101] or artificial neural networks [110].

Interaction-based approaches use road information and interactions between all traffic participants. They explicitly consider dependencies between the motion and intentions of different traffic participants. Representative examples are right-of-way situations or car-following maneuvers. The vehicle following behavior is often described using models like the IDM [145], the Gazis-Herman-Rothery-Model [34], or the Nagel-Schreckenberg-Model [88] that relies on cellular automation theory.

More sophisticated approaches have been presented as the previously mentioned models are limited to certain behaviors. Kuhnt et al. [100] use object oriented probabilistic modeling language (OPRML) as a description language to model vehicle interactions. They thereby cope with the problem of uncertain dependencies between traffic participants and their motion. To achieve this, road features like geometry and topology are used while explicitly handling the cardinality uncertainty in the estimation. By describing the state of the environment using a DBN, Gindele et al. achieve a holistic probabilistic model of the traffic scene at hand that includes available context knowledge [64]. The authors use the representation to estimate driver interactions. The approach is limited to vehicles only. DBNs are also applied by Levèfre et al. [106] to compare reference trajectories to observations with a focus on risk estimation for unexpected behaviors and by Schulz et al. [134] to estimate the intentions of vehicles and subsequently predict traffic scenes with interacting vehicles. For inference, sequential Monte Carlo inference is applied.

Interactions and relations between entities like road users or road sections are represented in a tensor representation in [121]. The compressed representation is used to reason about reoccurring situations in multi-agent traffic situations. The tensor representation simplifies the use of neural networks for processing relational information. Recently GNNs [85] have been found helpful for prediction tasks in traffic environments. However, how all information can be encoded to be used with these models is currently being researched. In a later work by Grimm et al. [9], a holistic representation using different types of nodes to comprise all relevant information in one graph is proposed to predict the environment.

The interactions in pedestrian predictions are represented using social force models like in [74] and [91] or collision avoidance models as in [146]. For simulation, the MENGE framework [53] provides pedestrian interaction models for advanced pedestrian prediction.

In contrast to the approach introduced here, the approaches only consider vehicle interactions but not interactions with other traffic participants or regulatory elements like traffic lights. Additionally, the

approaches are often limited to a fixed number of traffic participants and intentions or do not generalize to other situations.

4.1.2 Concept

As mentioned above, the intention estimation process includes two steps. First, different potential developments of a traffic scene are created from a previous point in time. Then, these developments are rated using an HMM while taking the actual development as a measurement.

With intermediate steps, the process to estimate the intentions of all road users at time t_0 is described as follows. The corresponding nomenclature and state definition can be found in Sec. 2.1.2.

1) Start at the past time $t_{-\tau}$ with state of the environment $S_{-\tau}$.
2) Determine possible intentions for each traffic participant at $S_{-\tau}$, i.e., routes, maneuvers etc.
3) Generate permutations of intentions of different traffic participants to obtain M combinations of intentions that lead to different scene developments.
4) Predict the traffic scene from $t_{-\tau}$ to t_0 for all M combinations yielding the possible developments $\mathcal{D}_{\hat{S},k} = \{\hat{S}_{k,-\tau} \dots \hat{S}_{k,0}\}$ for $k = 1, \dots, M$.
5) Determine the transition matrix $\mathcal{T}$.
6) Determine observation probabilities $\mathcal{Z}$ by comparing the assumed developments $\mathcal{D}_{\hat{S},k}$ with the history of the actual development $H = \{S_{-\tau} \dots S_{-1}\}$ and the current state S_0 for $t = t_{-\tau}, \dots, t_0$ and $k = 1, \dots, M$.
7) Infer probabilities of all developments $\mathcal{D}_{\hat{S},k}$ with a forward pass in the resulting HMM and marginalize probabilities of intentions of individual road users.

Fig. 4.3 shows the concept for the situation from Fig. 4.2 for a later point in time to have more illustrative differences in the developments.

In step 1) the scene at time $t_{-\tau}$ is regarded. The state of the environment $S_{-\tau}$ hereby can be enriched with information about traffic participants that came into view later. In Fig. 4.3, the state of the vehicles is depicted as a transparent setback image of the vehicles.

Afterward, the intentions of each traffic participant are determined in step 2). For the example in Fig. 4.3, the intention of the ego vehicle is known. The red vehicle has three different intentions: going straight, turning left, or turning right. As mentioned above, the intentions can generally be arbitrary as long as there is a corresponding prediction module to predict the respective motion. Each traffic participant i for $i = 1, \dots, N$ has m_i different intentions. In step 3), to obtain different developments of the whole scene,

$$M = \prod_{i=1}^{N} m_i \tag{4.1}$$

permutations are determined. The advantage here is that considering all permutations provides all potential explanations for the movement of a traffic participant. However, this number increases exponentially. The number of potential developments of the scene can be reduced in some cases. For example, if the intention of a traffic participant is already determined with very high confidence. Additionally, if it can be

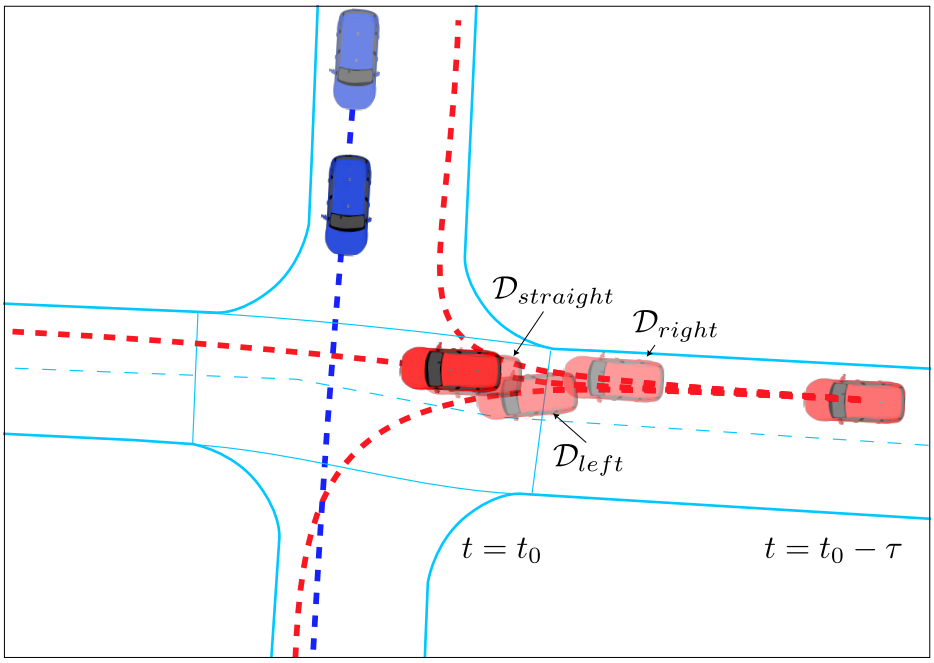


Figure 4.3: Depiction of the concept for the example situation. The ego vehicle in blue estimates the turn intention of the red vehicle on the prioritized road. The positions of the vehicles at the current point in time are drawn opaque. The positions of the vehicles at $t = t_0 - \tau$ are drawn at the edges of the image, slightly transparent. The expected position for the red vehicle, if it followed the intentions "turn left," "turn right," and "go straight" from the state at $t = t_0 - \tau$, is also shown as not fully opaque.

derived that a traffic participant moves independently of the others, the traffic participant's intention can be determined separately. A third case, where the number of permutations can be reduced, is when the behavior of a road user is solely determined by another traffic participant, like in car-following scenarios.

Nevertheless, in step 4) the state of the environment $S_{-\tau}$ is predicted form $t_{-\tau}$ to t_0 for all M combinations resulting in M possible developments $\mathcal{D}_{\hat{S},k}$. The choice of prediction module is not restricted. Different task-specific prediction modules could be used. The prediction module used to evaluate the presented approach is described in Sec. 4.1.3. The red vehicle's predicted positions in the example are shown transparently in Fig. 4.3. The positions correspond to where the vehicle would be expected to be for the corresponding turn intention.

The probability of the developments is determined using the HMM, where the hidden variables are the corresponding developments. Therefore, the transition matrix $\mathcal{T}$, the observation matrix $\mathcal{Z}$, and the initial probabilities must be set up in the steps 5) and 6).

The initial probabilities Π of the hidden states representing the developments can be determined using previous estimation results or set with fixed values. Prior knowledge can be included to guide the estimation. For route estimation, traffic data could be used to weigh specific routes differently to mirror the average expected traffic flow. In this work, they were assumed to be equally distributed. It follows that all routes are seen as equally likely.

The state transition matrix $\mathcal{T}$ specifies the probability of a development change from one point in time to the next. As the number of developments $\mathcal{D}_{\hat{S},k}$ is constant for the regarded time interval, the size of $\mathcal{T}$ is constant too. The assumption was made that the elements of $\mathcal{T}$ are also constant. It means that the

probability of changing between different developments and, therefore, different intention combinations does not change over time. This results in

$$\mathcal{T} = \begin{bmatrix} u_{1,1} & \dots & u_{1,M} \\ \vdots & \ddots & \\ u_{M,1} & & u_{M,M} \end{bmatrix} \text{ with } u_{i,j} = \begin{cases} p_{keep}, \text{ if } i = j, \\ \frac{1-p_{keep}}{M-1}, \text{ if } i \neq j. \end{cases} \tag{4.2}$$

In equation (4.2), p_{keep} indicates the probability of sticking with a development. It indicates that the traffic participants will stay with their intentions. A change of intention and, therefore, a change to a different development is considered to be equally likely to any other development.

To obtain the observation probabilities $\mathcal{Z}$, the history H provides measurements S_t for each time step. For every time step, the assumed development $\mathcal{D}_{\hat{S},k}$ with its states of the environment $\hat{S}_{k,t}$ are evaluated using the following features:

- f_1: The **offset** from the reference line. The reference line is built from the route intention of the traffic participant. In this work, the centerline of the road was used as reference line.
- f_2: The **orientation difference** to the orientation of the nearest point on the reference line normalized by π.
- f_3: The **speed difference** between the assumed speed $\hat{v}$ and the actual speed v at time t $\Delta v = \hat{v}_t - v_t$.

The features are considered to be independent. They are assumed to be normally distributed with zero mean in the following way

- $P(f_{1_{k,i}}|\hat{s}_{k,i,t}, s_{i,t}) \sim \mathcal{N}(0, \sigma_{f_1})$,
- $P(f_{2_{k,i}}|\hat{s}_{k,i,t}, s_{i,t}) \sim \mathcal{N}(0, \sigma_{f_2})$,
- $P(f_{3_{k,i}}|\hat{s}_{k,i,t}, s_{i,t}) \sim \mathcal{N}(0, \sigma_{f_3})$,

with the predicted state $\hat{s}_{k,i,t}$ and the actual state $s_{i,t}$ for development k of traffic participant i at time index t.

The transition function $\mathcal{T}$ and observation probabilities $\mathcal{Z}$ can be learned. The advantage in combination with the proposed approach here is that it generalizes to other situations as the approach is universally applicable and not bound to certain situations.

The unnormalized observation probability of development k subsequently results to

$$P(b_{k,t}|\hat{S}_{k,t}, S_t) \sim \prod_{j=1}^{N} P(f_{1_{k,i}}|\hat{s}_{k,i,t}, s_{k,t}) \cdot P(f_{2_{k,i}}|\hat{s}_{k,i,t}, s_{k,t}) \cdot P(f_{3_{k,i}}|\hat{s}_{k,i,t}, s_{k,t}). \tag{4.3}$$

In the final step 7), the likelihood of each potential development $\mathcal{D}_{\hat{S},k}$ is inferred in a forward pass through the resulting HMM. Each development $\mathcal{D}_{\hat{S},k}$ contains a combination of intentions of the traffic participants. A traffic participant can follow the same intention in multiple developments. To obtain the marginalized probability of an intention of a specific road user, the likelihood of each development, where the road user is assumed to have the corresponding intention, is summed up.

4.1.3 Prediction Module

The prediction module that is presented in the following is designed to iteratively predict a state of the environment S from a point in time $t = t_n$ to a certain time horizon $t = t_{n+T}$. The prediction is conducted stepwise so that $t_{n+1} = t_n + \Delta t$ while considering interactions between traffic participants at every point in time. Tracked objects are used as input for the prediction.

It is distinguished if a traffic participant follows a reference line or not. The reference line can, for example, be a lane or sidewalk. Objects that are not assumed to follow a certain reference line are predicted with constant velocity assumption.

$$x_{n+1} = x_n + \Delta t \cdot v_n \cdot sin(\theta_n) \tag{4.4a}$$
$$y_{n+1} = y_n + \Delta t \cdot v_n \cdot cos(\theta_n) \tag{4.4b}$$
$$\theta_{n+1} = \theta_t \tag{4.4c}$$
$$v_{n+1} = v_n \tag{4.4d}$$

This mainly affects pedestrians and unclassified objects that have a velocity. The motion of traffic participants that follow a reference line is predicted using a bicycle model

$$x_{n+1} = x_n + \Delta t \cdot v_n \cdot sin(\theta_n) \tag{4.5a}$$
$$y_{n+1} = y_n + \Delta t \cdot v_n \cdot cos(\theta_n) \tag{4.5b}$$
$$\theta_{n+1} = \theta_t + \Delta t \cdot v_n \cdot \frac{\tan(\delta_{steer})}{l} \tag{4.5c}$$
$$v_{n+1} = v_n + \Delta t \cdot a \tag{4.5d}$$

with steering angle δ_{steer} and acceleration a as input and l as the distance between front and rear axle. As l is typically unknown, it is estimated as a fraction of the total object length. Primarily, the bicycle model is used for vehicles, bicycles or similar. To account for state and motion uncertainties, an Extended Kalman Filter predictor is used to forecast the states of all traffic participants including the corresponding state uncertainty.

To determine the traffic participants that follow the road network, they are first mapped to the lanes in the road network. It is assumed that traffic participants can either still be on the same road segment, a neighboring segment, or a following segment in the following time step. This leads to higher accuracy in situations where a lot of road segments overlap, like, for example, in intersections. The mapping is done probabilistically. Therefore, the resulting mapping shows the likelihood of a traffic participant following a certain road segment or lane. The mapping to the road network lays the foundation of the route finding process. Starting from each road segment that the traffic participant is assumed to follow, potential routes are determined with respect to the intersections and turn options ahead. The potential routes are equal to the road users' intention, as the intention estimation is shown on the example of route intentions in this work.

The steering angle δ_{steer} is determined using a Stanley controller [77]. Therefore, a reference line is generated from the route. The reference line is a path that the traffic participant is supposed to follow without accounting for dynamic constraints. The simplest reference line for vehicles is the center of the road segments of the corresponding route. Unless there is an explicit path for bicycles, the reference

path for bicyclists is shifted to the side of the road. The lateral behavior is determined with regard to the orientation error $\Delta\theta$ and distance error d_{err} to the reference line

$$\delta_{steer} = \Delta\theta + \arctan\left(\frac{k_{s,1}\, d_{err}}{k_{s,2} + v}\right) \tag{4.6}$$

with the two constants $k_{s,1}$ and $k_{s,2}$. The second control input, the acceleration a of the traffic participant, determines the longitudinal behavior and is determined using the IDM

$$a_{A,IDM} = k_{a,IDM}\left(1 - \left(\frac{v_{front}}{v_{desired}}\right)^{\delta_{IDM}} - \left(\frac{d^*}{\Delta d}\right)^2\right), \tag{4.7}$$

$$d^* = d_0 + v_{front}\, T_{IDM} + \frac{v_{front}\,(v - v_{front})}{2\sqrt{k_{a,IDM}\, k_{b,IDM}}}, \tag{4.8}$$

with the components

- spacing Δd between the vehicles,
- desired speed of the ego vehicle $v_{desired}$,
- speed of the ego vehicle v in longitudinal direction,
- speed of vehicle in front v_{front} in longitudinal direction,
- acceleration exponent δ_{IDM},
- desired acceleration value $k_{a,IDM}$,
- comfortable braking deceleration $k_{b,IDM}$,
- time headway T_{IDM},
- and minimum bumper distance d_0.

Multiple factors influence the acceleration a. These are discussed in the following.

- The **desired speed** of the traffic participant corresponds to the speed limit. For bicycles, the desired speed is chosen as an average speed for bicyclists. The corresponding acceleration a_{free} is calculated with the IDM assuming a following vehicle far away with high speed so that the limiting factor of the IDM can be neglected ($\Delta d > 200\,\mathrm{m}$, $v_{front} = v_{speedlimit}$).
- The traffic participants do not exceed a **maximum lateral acceleration**. Therefore, the speed is adjusted accordingly to the curvature of the road ahead. Again, the acceleration a_{curve} is determined with the IDM with the limited speed as target speed ($v_{desired} = v_{max,curve}$).
- Traffic participants do not cause **collisions**. The distance to other traffic participants and their speed along the reference line is determined. This may be a crossing object or a traffic participant in front. Both ways, the acceleration a_{follow} is calculated with the IDM using the distance and speed of the object in front.
- Traffic participants obey the **rules of traffic**. Therefore, the distance to the next stopline or right-of-way situation is calculated. The related acceleration a_{yield} is again determined using the IDM

with an object with zero speed in front of the stopline and a minimal bumper distance so that the traffic participant stops in front of the stopline ($d_0 = 0.1\,\mathrm{m}$, $T_{IDM} = 0\,\mathrm{s}$, $v_{front} = 0\,\mathrm{m/s}$).

The accelerations are limited to the interval $[a_{min}, k_{a,IDM}]$ to stay within a feasible range. The minimum of the aforementioned accelerations is used as control input

$$a = \arg\min \left\{ a_{free}, a_{curve}, a_{follow}, a_{yield} \right\}. \tag{4.9}$$

4.1.4 Evaluation

The evaluation was conducted for several simulated and real-world scenarios. The scenarios include

- a simulated typical left turn situation,
- a simulated situation with a traffic light as a dynamic regulatory element,
- a simulated right turn situation with a pedestrian crossing the street,
- a simulated scenario with multiple vehicles approaching an intersection,
- and a real-world scenario from the InD Dataset [46].

A DBN with sequential Monte Carlo inference is used to compare the proposed HMM approach. Monte Carlo methods rely on repeated random sampling to find approximate solutions to complex problems. The use of Monte Carlo methods, in general, can be computationally expensive. Nevertheless, this work used Monte Carlo inference to solve the DBN because no closed analytical solution can be found that allows for efficient inference. This is because of the nonlinear components, the complexity, and arbitrary probability distributions. For such problems, Monte Carlo methods are a common choice. For the evaluation, particles, as known from particle filters, are used for the random samples. A particle represents the entire state of the environment with a concrete combination of the intentions of all traffic participants. The $N_{particles}$ particles are predicted using the same prediction module to avoid evaluating

Table 4.1: Parameters for the evaluation of the intention estimation

Parameter	Value
σ_{f_1}	$1.5\,\mathrm{m}$
σ_{f_2}	0.2
σ_{f_3}	$1.2\,\mathrm{m/s}$
$N_{particles}$	1000
N_{runs}	200
Δt_{sim}	$0.3\,\mathrm{s}$
Δt_{real}	$0.2\,\mathrm{s}$
τ	7

the prediction module instead of the intention estimation. Furthermore, the same observation features are used to update the particle weights. The initial sample distribution also corresponds to the prior probabilities Π. This ensures that the results are comparable. Additionally, the vehicle parameters, i.e., the IDM parameters used for calculating the accelerations, are varied with zero Gaussian mean to make the particles span a wider range of possible behaviors. In theory, this should lead to more accurate intention estimations.

The simulated scenarios were run $N_{runs} = 200$ times with varying simulation parameters to have the traffic participants act slightly differently and to avoid evaluating only with one type and position. All traffic participants' initial position and speed were varied. The vehicles' positions were randomized and did not exceed the road borders. The estimation feature parameters σ_{f_i} were determined empirically for both the HMM and the DBN. The time intervals for the scenarios are Δt_{sim} and Δt_{real}. The number of estimation steps is in all cases $\tau = 7$.

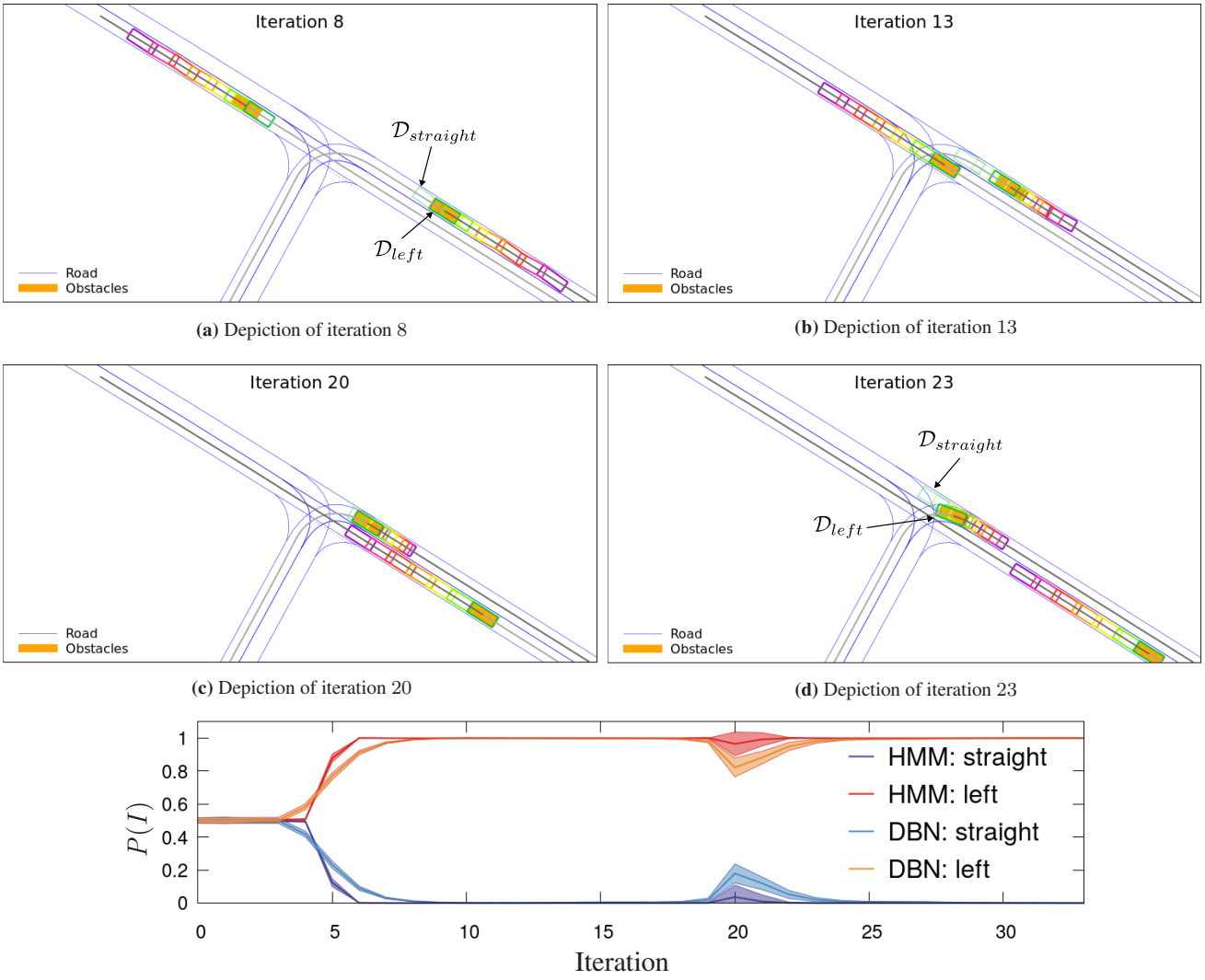
The color scheme for the depiction of the scenarios is the same for all scenarios, e.g. in Fig. 4.4a to Fig. 4.4d or Fig. 4.7a to Fig. 4.7c. The images only show one evaluation run for visualization purposes. The traffic participants or obstacles at $t = t_x$ are drawn in orange boxes. Only the contours are shown for the estimated developments $\mathcal{D}$ from $t = t_x - \tau$ to $t = t_x$, where t_x is the time of the estimation. The color of the contours encodes the point in time. Equal points in time have the same color. The colors range from purple for $t = t_x - \tau$ over red to green for $t = t_x$. The saturation encodes the corresponding likelihood, where higher saturations equal higher likelihoods. The figures 4.4e, 4.5b, 4.6b, and 4.8 show the estimated likelihoods of the intentions for the HMM and DBN with their mean value and standard deviation. The parameters used for the evaluation are shown in Table 4.1.

Scenario: Left turn with oncoming traffic

The first scenario includes two vehicles approaching an intersection from opposite directions. The turn intention of vehicle A from the right is to be estimated while vehicle B continues straight. Fig. 4.4a to Fig. 4.4d show the evolvement of the situation. The developments $\mathcal{D}_{left}$ and $\mathcal{D}_{straight}$ are color-coded as described above. $\mathcal{D}_{straight}$ is drawn in highly transparent colors because of its low likelihood.

Fig. 4.4e depicts the progression of likelihoods over time. When vehicle A is far from the turning point, the expected behavior for both possible developments $\mathcal{D}_{straight}$ and $\mathcal{D}_{left}$ are equal. This is because vehicle A has not yet started decelerating to wait for vehicle B to pass. Hence, the estimation predicts that both developments are equally likely.

When vehicle A starts to decelerate (t_4) and waits for vehicle B to pass ($t_7 - t_{21}$), the estimation confidence quickly increases because turning left is the only logical intention. Vehicle A would not have decelerated and waited in $\mathcal{D}_{straight}$. Both the HMM and the DBN correctly start to indicate a higher likelihood of a left turn. When vehicle A almost stops for the oncoming vehicle before turning left, the estimation uncertainty increases because vehicle A is very slow, and, therefore, the difference in the predicted developments $\mathcal{D}_{left}$ and $\mathcal{D}_{straight}$ from $t = t_{20} - \tau = t_{13}$ to t_{20} is marginal. This can be observed in Fig. 4.4c. The intention is correctly inferred.



(a) Depiction of iteration 8

(b) Depiction of iteration 13

(c) Depiction of iteration 20

(d) Depiction of iteration 23

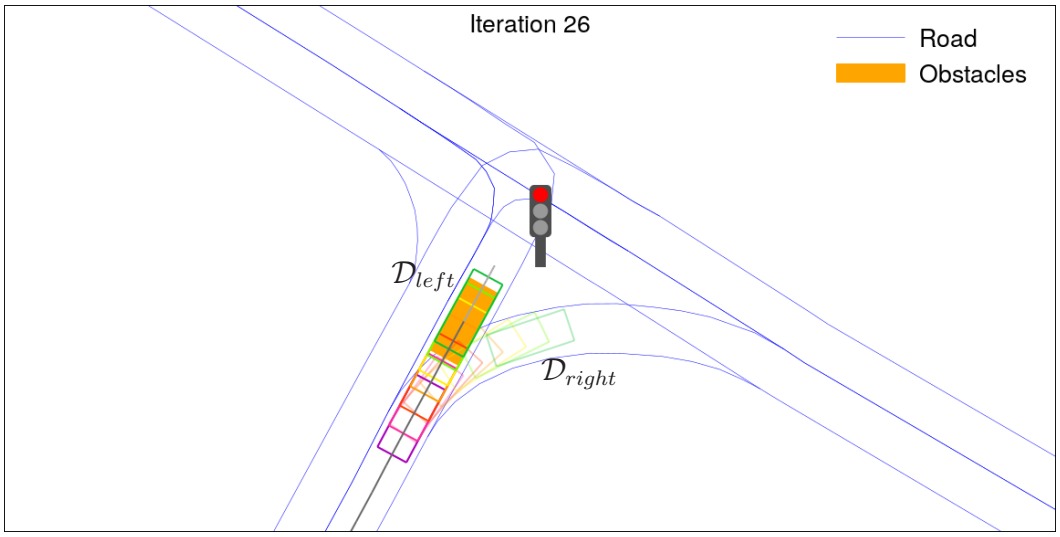
(e) Likelihood of turn intentions over time for HMM and DBN

Figure 4.4: Left turn scenario with oncoming traffic: (a)-(d) Vehicle A from the right turns left at an intersection with oncoming traffic. Developments $\mathcal{D}_{left}$ and $\mathcal{D}_{straight}$ are shown in color-coded contours. The color coding starts with purple for $t - \tau$ and changes over red and yellow to green for t. The path ahead is drawn in light grey, and the recent path is shown in dark grey. (e) The likelihoods of $\mathcal{D}_{left}$ and $\mathcal{D}_{straight}$ over time for the HMM and DBN for vehicle A. (Graphics based on [5], ©2021 IEEE)

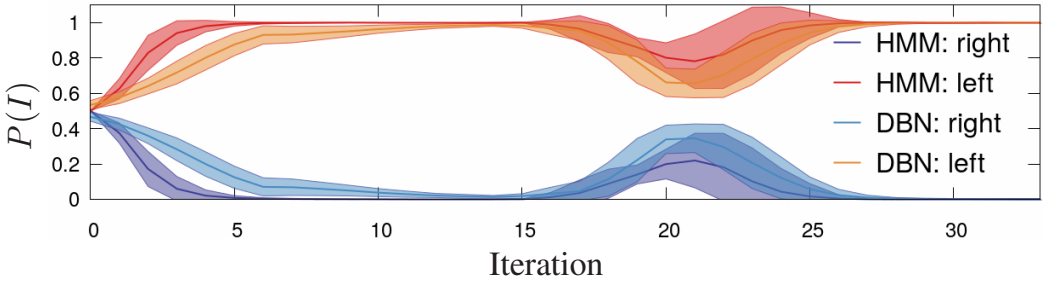
Scenario: Traffic light

The second scenario is used to examine two aspects. On the one hand, the scenario includes a traffic light. Regulatory elements like this are often neglected when estimating intentions. On the other hand, the influence of the velocity feature f_3 is investigated.

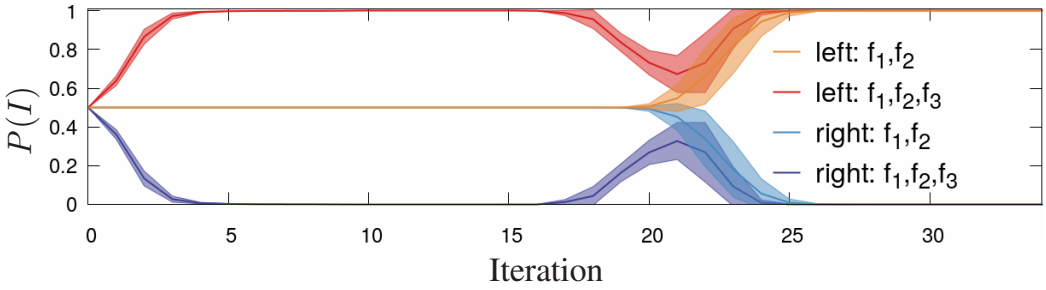
The vehicle drives towards the signalized intersection. The traffic light for turning left is red. There is an extra lane for turning right, where the vehicle does not have to pass the traffic light. The situation is shown in Fig. 4.5a. Initially, the vehicle follows the same path for both turn options. This fact can also be seen in Fig. 4.5c. When using the velocity feature f_3, the intention can be estimated earlier because the vehicle has to brake earlier for the traffic light than for the right turn. Like in the previous scenario, the confidence decreases when the vehicle slows down. The initial probabilities Π are to be determined



(a) Depiction of iteration 26



(b) Likelihood of turn intentions over time for HMM and DBN

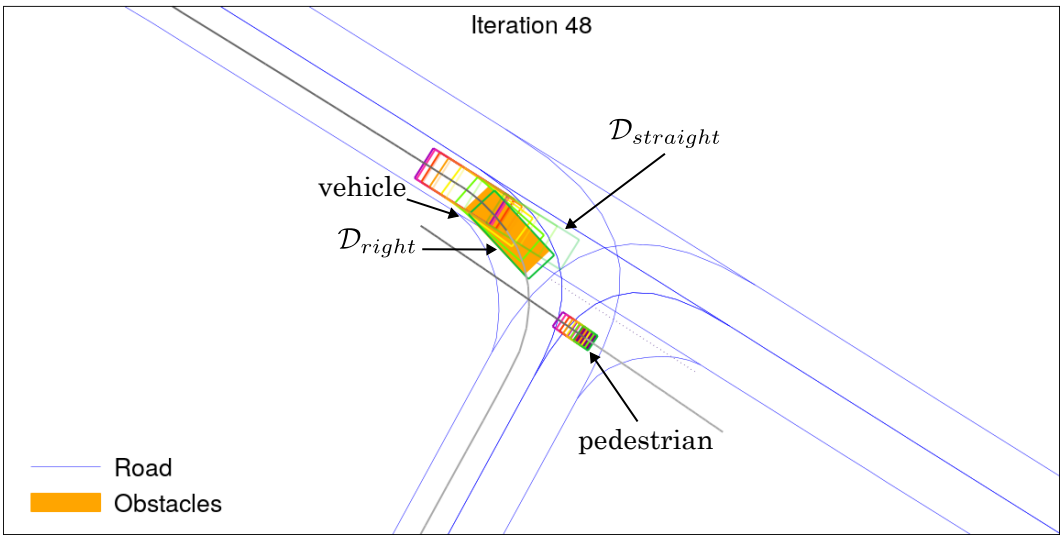


(c) Likelihood of turn intentions over time for HMM with and without velocity information

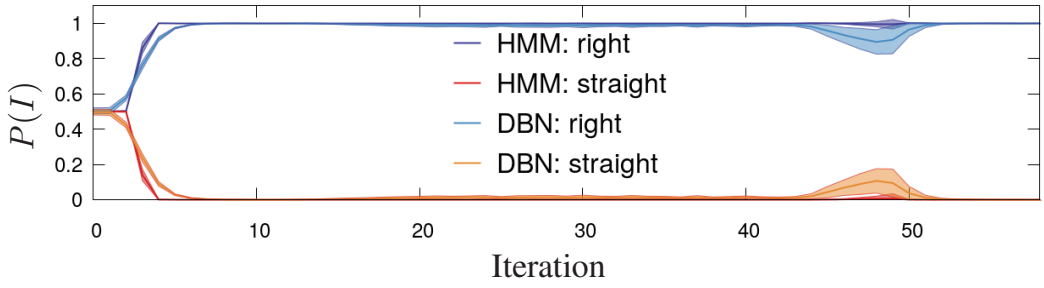
Figure 4.5: Scenario with traffic light: (a) Vehicle A (orange) approaches an intersection with a traffic light for left turns and a merging lane for right turns. Developments $\mathcal{D}_{left}$ and $\mathcal{D}_{straight}$ from t_{19} to t_{26} are shown in color-coded contours. The path ahead is drawn in light grey, and the recent path is shown in dark grey. (b) The likelihoods of $\mathcal{D}_{left}$ and $\mathcal{D}_{right}$ over time for the HMM and DBN. (c) Turn intention likelihoods over time for vehicle A in the traffic light scenario with and without using velocity information (i.e., feature f_3) for estimation using the HMM. (Graphics based on [5], ©2021 IEEE)

from previous estimation results to avoid this for online usage in the automated driving stack. Fig. 4.5b shows that the HMM again performs as well as the more expensive DBN. It should be noted that only 100 simulation runs were used to generate Fig. 4.5c. Hence, the confidence is higher.

The lessons learned from this scenario are that regulatory elements are as essential as traffic participants and that the velocities of traffic participants deliver valuable information for intention estimation.



(a) Depiction of iteration 48



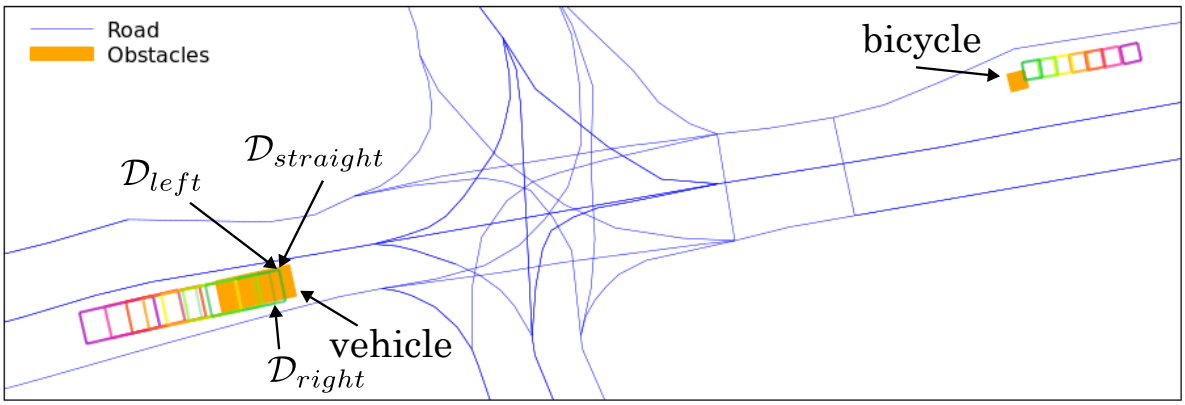
(b) Likelihood of turn intentions over time for HMM and DBN

Figure 4.6: Scenario with predestrian at intersection: (a) Vehicle A approaches an intersection and waits for a pedestrian to cross before turning right. Developments $\mathcal{D}_{straight}$ and $\mathcal{D}_{right}$ from t_{41} to t_{48} are shown in color-coded contours. The path ahead is drawn in light grey, and the recent path is shown in dark grey. (b) The likelihoods of $\mathcal{D}_{straight}$ and $\mathcal{D}_{right}$ over time for the HMM and DBN. (Graphics based on [5], ©2021 IEEE)

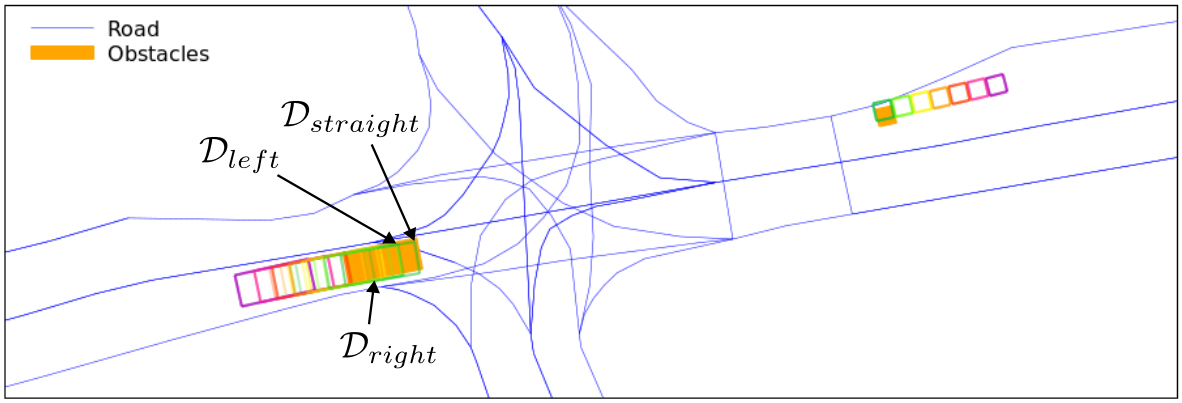
One behavior that frequently occurs in everyday situations with traffic lights is the passing or stopping of vehicles at yellow lights. This behavior can also be estimated using the presented approach. The route of the vehicle is the same, but once the vehicle is predicted to stop at the yellow light and once the vehicle is predicted to pass the intersection. These are two different intentions of the vehicle, which can be easily assessed using the concept described.

Scenario: Intersection with pedestrian

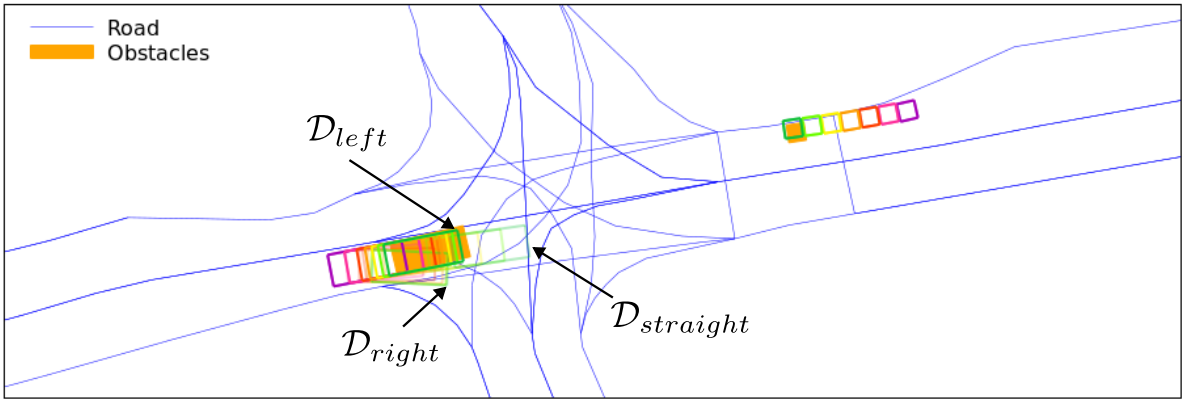
In the following scenario, a pedestrian walks parallel to vehicle A that wants to turn right at the intersection ahead (see Fig. 4.6a). It also demonstrates the ability of the approach to consider other traffic participants who do not drive on the road but are, nevertheless, an essential aspect of everyday driving situations. From Fig. 4.6b, it is early estimated that the vehicle wants to turn right. Again, the velocity feature f_3 significantly contributes as the vehicle decelerates instead of following straight with the same velocity.



(a) Scenario from Ind Dataset at iteration 13



(b) Scenario from Ind Dataset at iteration 20



(c) Scenario from Ind Dataset at iteration 25

Figure 4.7: Depiction of the situation for different times for the scenario from the InD Dataset. (a)-(c) The left-turning vehicle has to wait for an oncoming bicycle to pass. The arrows point to the foremost corner of the vehicle for the respective development.

Scenario: Left turn from InD dataset

Fig. 4.7a to Fig. 4.7c show a scene from the InD Dataset [46]. The vehicle from the left wants to turn left. It has to wait for the oncoming bicycle to pass. From the likelihoods over time in Fig. 4.8, it can be derived that the right turn intention is ruled out quite early as the vehicle would have had to decelerate sooner (iteration $0 - 16$). When the vehicle brakes (iteration $16 - 20$), the likelihood of driving straight initially increases slightly because the vehicle stops later than the prediction module expects. However,

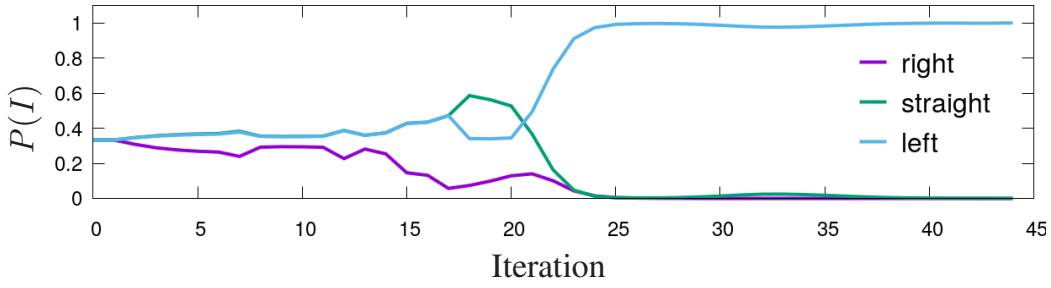


Figure 4.8: The likelihoods of $\mathcal{D}_{left}$, $\mathcal{D}_{straight}$, and $\mathcal{D}_{right}$ over time for the HMM for the scenario from the InD Dataset. (Graphic from [5], ©2021 IEEE)

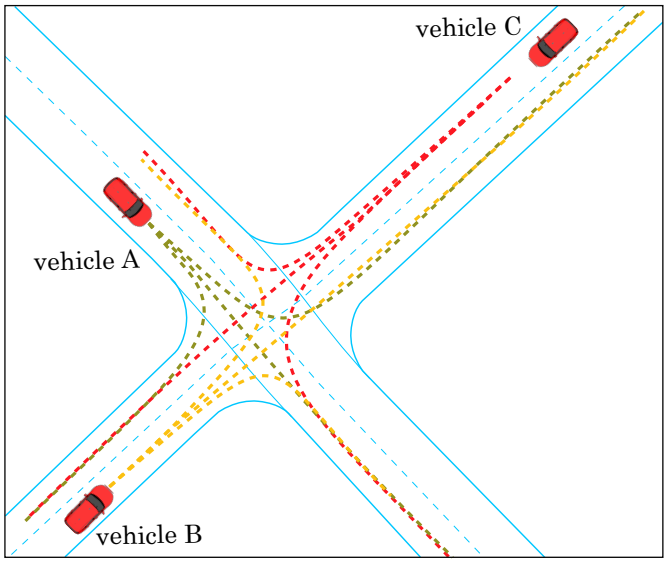
as the vehicle becomes slower (iteration $20 - 24$) to wait for the oncoming bicycle, the true intention is revealed and correctly assessed, see Fig. 4.7c.

In contrast to the previous scenarios, this scenario is taken from the real world. It can be noticed that there are more observable deviations in the predictions. The deviations result from inaccuracies in the prediction module that assumes the vehicle will stop further back. However, the approach still correctly infers the vehicle's intention, which could be proved in further samples from the dataset.

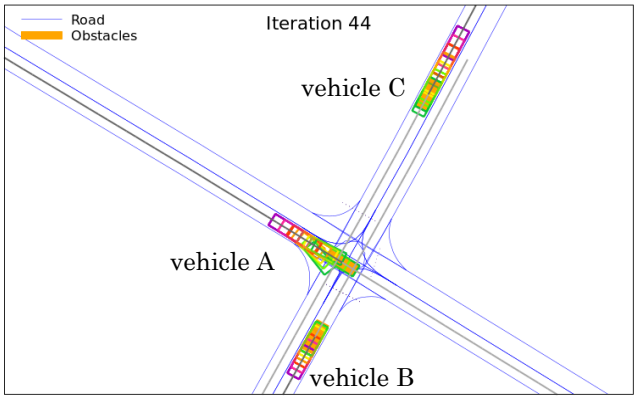
Scenario: All intentions unknown

In the previous scenarios, the intention of only one traffic participant was inferred to show single aspects like the inclusion of traffic lights. However, a primary advantage of creating various potential developments is that the likelihood of the combination of the agents' intentions is determined. Because the behavior of an agent always depends on its own intentions, the intentions of others, and the behavior of others, the agent can behave differently for the same intention, depending on the other agents. Thus, only estimating the intention of a single agent while neglecting the context can lead to false situation assessments. When inferring the intention in the proposed way, the marginal probability of an agent's intention can be extracted by summing up the probabilities of all developments with the corresponding intention.

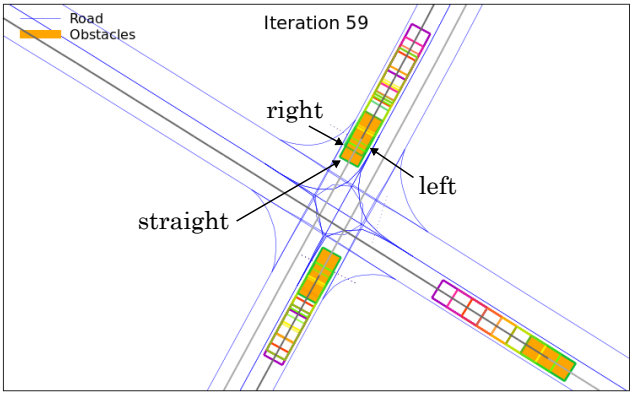
Fig. 4.9 shows an intersection with three vehicles approaching. Each vehicle can turn left, turn right, or go straight. This results in $M = 27$ different potential developments. The initial likelihood for each of those is $1/27$. The vehicles have different distances to the intersection and, therefore, arrive at the intersection at different times. When the vehicles approach the intersection around iterations 42, 52, and 58, approximately a third of the developments become unlikely as the actual intention reveals itself. Fig. 4.9c depicts the situation before vehicle C enters the intersection. Through the more significant deviations in the predictions for turning left or right, it can be concluded that vehicle C will go straight. The intentions for all vehicles are determined correctly early on, as the blue dashed line in Fig. 4.10 and the green lines in Fig. 4.11 show. Fig. 4.11 depicts the marginalized probabilities of the intentions of each of the vehicles. The estimation for vehicle B shows a higher likelihood of turning right during iteration 40 to 53. When vehicle B stops longer for vehicle A to pass than expected by the prediction module, the likelihood of vehicle B turning left increases (iteration 53 to 58). At that point, it cannot be differentiated if vehicle B just waited longer after vehicle A passed or if vehicle B waits for the oncoming vehicle C to pass in order to turn left. This can also be observed on the dashed lines in red and yellow in Fig. 4.10 for the probability of the corresponding combination of the intentions. Nevertheless, going straight is always estimated as the most probable and correct intention.



(a) Scenario with intention estimation for all vehicles



(b) Depiction of iteration 44



(c) Depiction of iteration 59

Figure 4.9: Scenario where all intentions are unknown at intersection: (a) Intersection with three vehicles coming from different directions. Each vehicle has three options (left, straight, right). (b)-(c) Evolvement of the scene when vehicle A and then vehicles B and C reach the intersection.

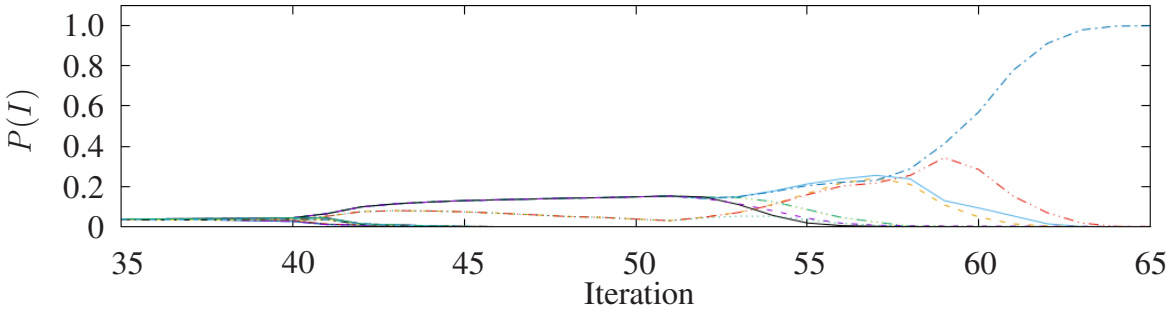


Figure 4.10: The likelihoods over time for the HMM for the developments $\mathcal{D}_i$ for $i = 0, \ldots, 26$. Colors and linestyles are varied to obtain 27 different lines. (Graphic from [5], ©2021 IEEE)

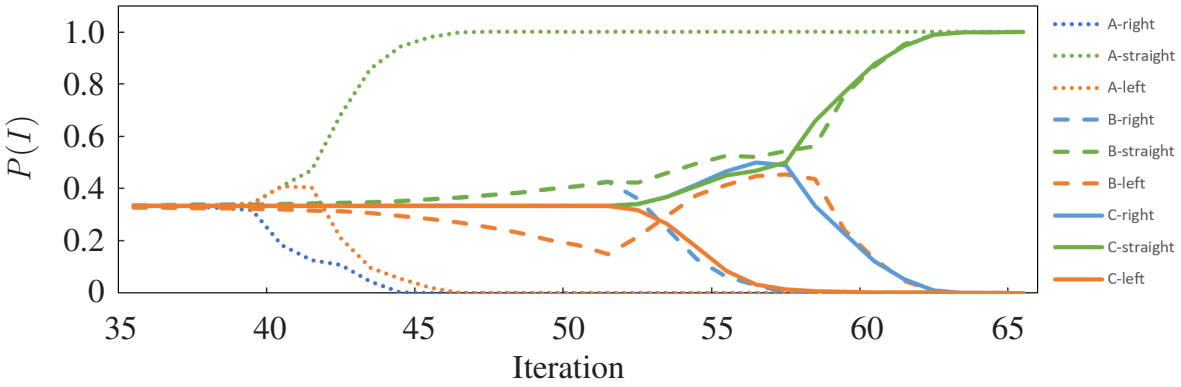


Figure 4.11: Marginalized probabilities for the intentions of vehicles A, B, and C.

Discussion

The presented approach for estimating intentions proved to be an efficient one-shot estimation approach. For the online application in an automated vehicle, the marginal probabilities of the single intentions of each traffic participant should be calculated and used for more informed initial probabilities Π in the next estimation cycle to iteratively improve the estimation result. By marginalizing the probability of single intentions, new road users that just have entered the scene can easily be integrated into the estimation process. The approach uses all relevant context information like regulatory elements or traffic participants on and off the road. The efficiency was shown in several simulated and real-world scenarios. The average runtime over all scenarios was $0.73\,\mathrm{ms}$ on an i7-10750H. The performance of the HMM was equal to that of the DBN. However, the computational effort of the DBN is $\sim$150-times higher for 1000 particles and $\sim$15-times for 100 particles used in the Monte Carlo inference. Consequently, the HMM should be preferred in the scenarios shown.

The estimation results almost reach the likelihood of 1.0, which looks unrealistic when thinking about everyday situations. Typically, other traffic participants' intentions are only known with a certain doubt. However, as mentioned above, the number of intentions included was limited. For vehicles, only route intentions were inferred. After the turn has been made, the actual route of the vehicle is the only valid option. Therefore, absolute certainty can be achieved in the evaluation scenarios. For the application in everyday traffic, more intentions should be included, like lane changes and stopping intentions for vehicles that simply stop to park at the side of the road. Additionally, it should be assumed that a traffic participant may not follow any of the predefined intentions. By doing so, absolute certainty is avoided, and the uncertainty in the estimation can be quantified better.

4.1.5 Conclusion

The main points of this chapter are summarized in the following

- The belief about the current state of the environment is enriched by estimating the intentions of traffic participants.
- The concept also uses all available context information, e.g., vehicles, pedestrians, or regulatory elements like traffic lights.

- The intentions are inferred by generating multiple potential developments of the scene from a past point in time $S_{t=-\tau}$ with successive classification of the most likely development based on the actual measurements.
- The concept is robust against traffic participants entering or leaving the scope because the number of traffic participants at inference time is known.
- A model-based prediction framework was presented. However, the intention estimation concept is agnostic to the applied prediction approach, so learned prediction modules can also be used.

4.2 Tracking and Forecasting Occluded Areas

After showing how intentions can be efficiently inferred from context information, this section focuses on occlusions as a second type of incomplete knowledge. As presented in Sec. 3.3, occluded areas are an essential part of the environment model. Occlusions are challenging for automated vehicles in two dimensions. On the one hand, many accidents or dangerous situations are caused by traffic participants' unawareness of occluded other road users. This leads to wrong decisions and late reactions. On the other hand, being aware of occlusions but assuming the worst case often obstructs fluent traffic and limits the suitability of automated driving for everyday use. More accurate estimations of occluded areas allow more precise prediction and decision-making. Accurately assessing occluded areas lays the foundation for foresighted maneuver decisions when dealing with occlusions.

For this reason, this section presents approaches to keep track of occluded areas based on previous sensor measurements and to forecast these occluded areas using context knowledge. Different traffic participants' motions differ, so two strategies are introduced for tracking occluded areas tailored to the corresponding object type. First, vehicles, trucks, and so on follow a road network or similar. This type of tracking is called oriented occluded area set tracking (OST). Second, some objects are not bound to a road network but move freely. This class mainly includes pedestrians but also animals or unclassified objects. The non-road-bound occluded area set tracking (NST) tracks the occluded areas where those objects can be expected. In both strategies, hidden areas are represented by a set of elements that define the hidden areas. The application of two different strategies is intentional. While one could argue that the NST could also be used for on-road tracking, many unlikely developments might occur, like an occluded vehicle moving onto the walkway. The separation leverages the particularities of both motion types and offers the best-suited description with regard to the effectiveness and transparency of the contained assumptions. They were partially published in [7] and [8], and the section can contain verbatim quotes from these works.

4.2.1 Related Work

Recent approaches for tracking occluded areas often focus on monitoring occluded areas or occluded objects that follow the road, e.g., [149, 106, 141, 155, 55]. In [149], Wang et al. estimate the longitudinal position and speed of potentially occluded road users with a set-based tracking approach. A particle-based approach that promises a more efficient risk assessment for occlusions at intersections was proposed in [155]. The authors, therefore, use the state and the particles' forward and backward reachability.

So-called reachable sets are used in [98] to determine where vehicles and pedestrians could be located in the future. The authors also apply it to occluded areas to account for areas where obstacles might appear. Nager et al. use a set-based approach for tracking potential occluded traffic participants on and off the road [116]. In [116], the FOV is used to update the potentially occluded areas. In this work, the information about tracked obstacles is additionally used to update the occluded areas. This ensures the usage of all available context information and further improves the accuracy of the occluded area tracking.

Further sensors could be installed at intersections with high-risk potential to reduce the extent of occluded areas. The support of automated vehicles with information from intelligent infrastructure at intersections with occlusions is, for example, shown in [115]. However, intelligent infrastructure (e.g. [18]) is only deployed in certain areas like the Test Area Autonomous Driving Baden-Württemberg [17] and is often

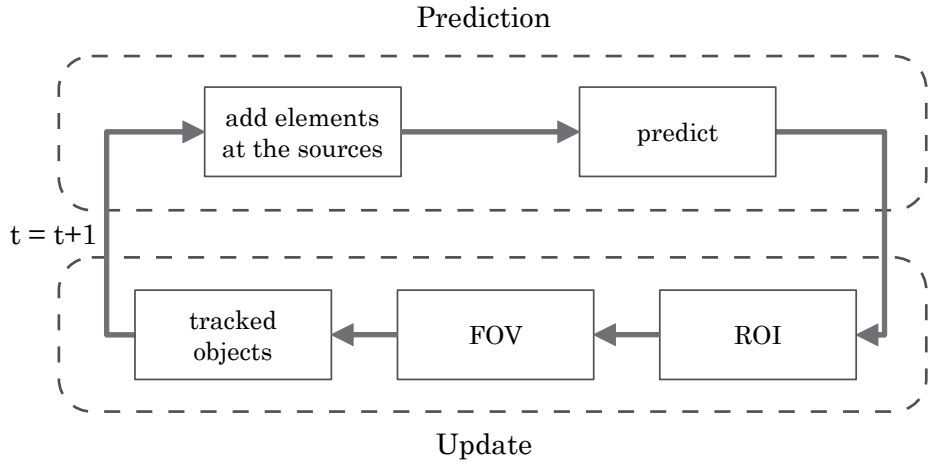


Figure 4.12: The tracking cycle consists of a prediction and an update. In the prediction step, new occluded areas can arise, and the occluded areas from the last point in time are predicted. During the update step, the predicted occluded areas are pruned by the region of interest, the field of view, and the tracked obstacles.

limited to specific use cases like infrastructure supported automated valet parking [6]. It does not scale to arbitrary situations, so the vehicle must build an awareness of occluded areas itself and react accordingly.

To the knowledge of the author, when [7] was published, no work included tracking and forecasting of occluded areas on and off the road that uses current sensor measurements and information about other visible objects to track and forecast occluded areas in a way that represents the particularities of the targeted object types.

4.2.2 Concept

Tracking and forecasting NST and OST follow the same principles. However, as the motions and representations differ, the following section is divided into general and strategy-specific parts.

The tracking process, in general, consists of a prediction and update cycle. In each time step, the occluded areas are first predicted from the previous estimation time to the current point in time with a time interval Δt. Then, the predicted areas are updated, meaning the assumption about how the areas have expanded is corrected. The process is depicted in Fig. 4.12.

In the prediction step, first new elements are added in source regions to the set that describes the occluded areas. These regions represent areas where new occluded objects could appear or enter the scene. These are mainly the edges of the region of interest (ROI). However, doors, garages, or parking cars can be considered source areas. If animals are also regarded, bushes and shrubbery might be included as source areas. The ROI limits the area where the tracking is performed to restrict computational resources. Furthermore, there is no benefit in tracking areas far beyond the sensor range as they can be considered occluded by default. In this work, the ROI is chosen to be a rectangle aligned with the x- and y-axis of the global Cartesian map frame. The ROI moves with the ego vehicle so that the ego vehicle is always in the ROI. The ego vehicle is not in the center of the ROI but shifted in a way that a larger part of the ROI is in front of the ego vehicle to account for the higher importance of the area in front of the ego vehicle and to account for the higher sensor range of the sensors to the front. The shift is called backtrack. When the ego vehicle and, therefore, the ROI is moved, the set elements forming the occluded areas can:

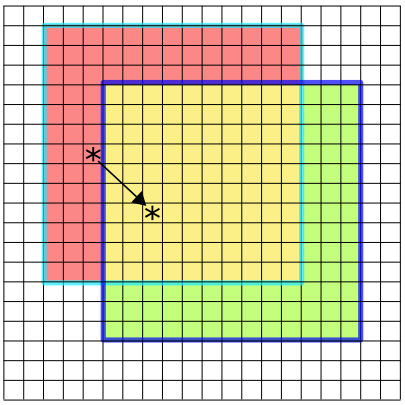


Figure 4.13: Depiction of a shift of the ROI. The boundaries of the ROI before are shown in light blue, and the shifted ROI in dark blue. The region that is no longer included in the tracking is shown in red. The yellow region is further tracked. The green area is newly included in the tracking. The green area is initially assumed to be fully occluded. The stars mark the position of the ego vehicle. The arrow indicates the shift of the position of the ego vehicle.

- still be in the ROI and can be continuously tracked.
- be completely outside of the ROI. The set element is dropped.
- be partially outside of the ROI. The parts outside of the ROI are truncated, and the parts inside the ROI are kept for tracking.
- be added in areas that are now in the ROI but have not been before.

For a grid-based world representation, this is depicted in Fig. 4.13.

In the update step, the predicted elements are corrected using the FOV and the information about visible and tracked objects. The result accurately assumes the currently occluded areas for the corresponding object type.

To get estimations about the areas that can be regarded as occluded in the future, they are forecasted to a time horizon T. These areas describe where a currently hidden road user could have moved to in the future. The corresponding process is shown in Fig. 4.14. It resembles the tracking process. However, the prediction of the tracked objects is used instead of their current state. Furthermore, the forecast process cannot use the FOV.

4.2.3 Non-Road-Bound Occluded Area Set Tracking

Non-road-bound occluded area set tracking (NST) is applied for occluded vulnerable road users like pedestrians or bicyclists who do not follow the road. The set of all non-road bound occluded areas $\mathcal{U}$ contains the elements u. The set and elements at a specific point in time t are represented by $\mathcal{U}_t$ and u_t. An element u is built from its position, shape, and speed. Here, $\mathcal{U}$ is modeled as a multi-layered two-dimensional grid, where each layer corresponds to a certain discretized speed value as shown in Fig. 4.15. Thus, the position and shape of u are determined through the cell and resolution of the grid. The speed is given through the layer to which the cell belongs. So, an element u only spans across one cell of one layer and, therefore, has a defined position, shape, and speed.

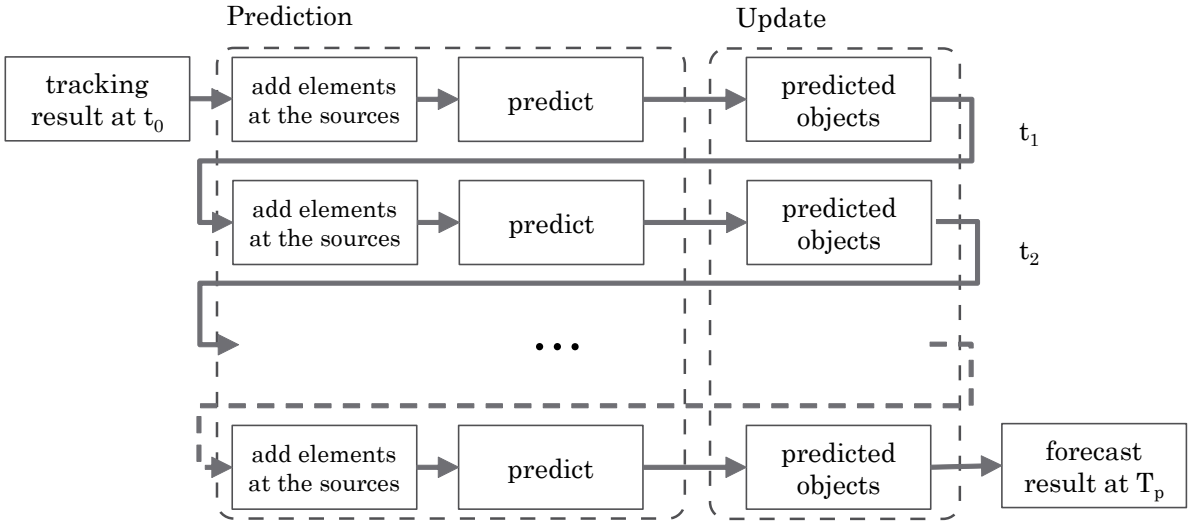


Figure 4.14: The forecasting process also uses prediction and update steps. The prediction step is similar to the prediction step of the tracking algorithm. In the update step, the predicted occluded areas are pruned based on the future positions of the predicted traffic participants.

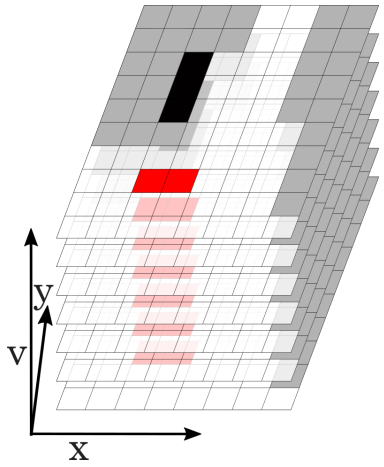


Figure 4.15: Depiction of the two-dimensional multi-layer representation used for NST. Every layer corresponds to a speed value. Each cell indicates if it is occupied, free, or if an occluded object of the corresponding speed can be located within the cell.

The discretization introduces errors in the estimation of occluded areas. Therefore, a cell is considered occluded if only a part of the cell is occluded in the FOV. This slightly overestimates the true extent of an occluded region but guarantees that all occluded areas are contained. The correct attunement of the resolution of the grid, the time interval of the tracking cycle, and the speed ranges can optimize the amount of overestimation. The grid has the exact dimensions as the ROI.

Due to the agility of pedestrians and their capability of sudden motions and changes in direction, the elements u are assumed to be able to move freely in every direction. Their speed determines the corresponding travel distance.

In the **initialization**, every cell of the multi-layered grid is assumed to be occluded. So potential obstacles with the entire speed range are expected everywhere in the ROI.

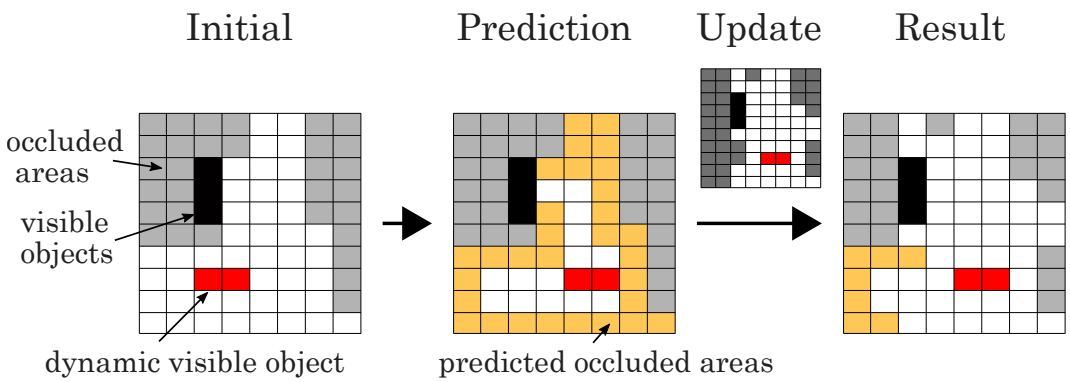


Figure 4.16: Depiction of the NST concept. The underlying representation is a multi-layer two-dimensional grid. Occluded cells are predicted in the prediction step based on the layers' speed values yielding the yellow pixels. The FOV and visible objects are used to update the prediction based on the current environmental information. (Graphic based on [7], ©2022 IEEE)

During the **prediction** step, the grid cells are dilated with respect to the determined travel distance of the contained elements. Cells that are occupied by static or visible objects block the movement so that the occluded areas can not pass these cells. The intermediate prediction result is shown in the center of Fig. 4.16. The figure shows the prediction and update step for a single speed value for simplicity. On the left, the grid before the prediction step is shown. Occluded cells are colored in grey, occupied in black, and cells considered free in white. In the center, the prediction result for a speed and time interval that corresponds to a dilation of one cell is shown. The cells that are considered to be occluded after the prediction step are colored in yellow. Cells that are occupied with visible, dynamic objects are drawn in red. These objects are not predicted like the occluded cells. However, the states and predictions of these objects given by the tracking and object prediction module provide important information for tracking occluded areas. New elements are coming in from the edges of the grid as these are source areas of new potentially occluded objects. Additionally, it can be seen that the area right of the occupied cells did not get occluded.

The **update** is performed with the latest FOV and state of the tracked objects. The FOV is directly determined from sensor measurements through ray casting. It is a two-dimensional grid with the same position and dimensions as the tracking grid containing free, occupied, and occluded cells, shown above the right arrow in Fig. 4.16. The grid-based FOV is further described in Sec. 2.8.2. The tracked objects are then entered into the grid, and the corresponding cells are marked as occupied. To update the elements of $\mathcal{U}$, a simple lookup is sufficient in the grid containing the FOV and the transformed objects. An element u is dropped if it is in a cell considered free space or occupied. The result is shown on the right in Fig. 4.16.

As mentioned above, in the **forecast** step, the prediction step is repeated with intermediate updates with the motion predictions of the tracked objects to obtain the forecast result $\hat{\mathcal{U}}$ to the desired time horizon.

4.2.4 Oriented Occluded Area Set Tracking

Unlike NST, oriented occluded area set tracking (OST) is applied for occluded road users that are assumed to follow a road or path. As mentioned above, separating both motion types allows target-specific tracking of occluded areas. OST assumes that the objects move in an area with a preferential direction, just like vehicles are following the road in the prescribed direction. The set of all oriented occluded areas Q is

built from its elements q. Sets and elements at a specific point in time t are described with Q_t and q_t. An element q is built from its position, shape, and speed range.

In the initial concept described in [7], an element of the oriented sets was represented as a polygon as shown in Fig. 4.17. The prediction was achieved by first dilating the polygon of each element q and combining overlapping polygons and elements afterward. The speed range was assumed to be between zero and a maximum speed. The update was performed using the FOV in polygon form traced from the surrounding objects as described in Sec. 2.8.1. This representation has the advantage that all areas can be covered and no discretization is needed, avoiding unwanted side effects imposed by it. However, the prediction (dilation and union) and the subsequent update (intersection) of multiple arbitrarily shaped polygons are extremely computationally expensive, and many edge cases must be considered.

Extensive application in real-world driving tasks and test runs in simulation revealed that information about speed ranges of potentially occluded objects is more important than their perfect outlines and that discretization of space can be neglected when keeping to a certain minimum resolution. For decision-making, it was found that it is more important to estimate the occluded object's speed to make foresighted decisions when deciding if and how to cross an intersection or occluded passage. The effects of the spatial discretization are leveled out by overapproximating occluded areas, which also increases safety. In the following, the updated concept is presented.

As noted above, the information about the speed of occluded objects is essential. Therefore, the focus is set to obtain this information as precisely as possible. The road is discretized in longitudinal and lateral directions, creating a grid that is oriented along the road with single cells or elements q. Road information is retrieved from an HD map that is represented using Lanelets. Each element q contains the set of speed ranges $\mathcal{V} = \nu_0, ...\nu_i..., \nu_n$ for $i = 0, \dots, N_v - 1$ that occluded objects might have with $\nu_i = [v_{i,min}, v_{i,max}]$. To give an example, an element q can keep track of slow objects with $\nu_0 = [3, 5]\mathrm{m/s}$ for bicycles and $\nu_1 = [9, 12]\mathrm{m/s}$ for vehicles giving the latter decision-making unit the freedom to determine if passing in between this two is possible. The discretization and the example are shown in Fig. 4.18a.

An element q is not occluded if it does not contain a speed range, i.e., if the set of speed ranges $\mathcal{V}$ is empty. There are no intersecting ranges in $\mathcal{V}$, i.e., $\nu_i \cap \nu_j = \emptyset$ for all $i, j = 0, \dots, N_v - 1$ and $i \neq j$. Intersecting speed ranges are combined to a single range (if $\nu_i \cap \nu_j \neq \emptyset$, then $\nu_{combined} =$

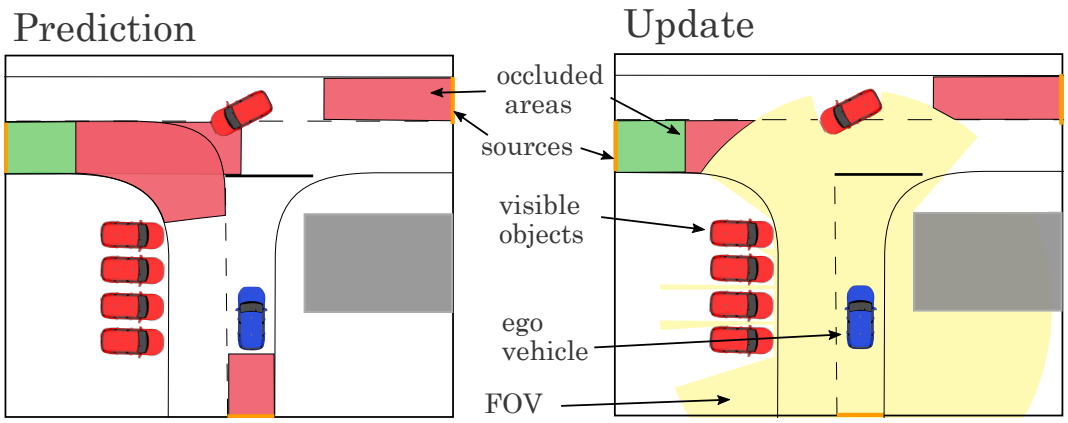
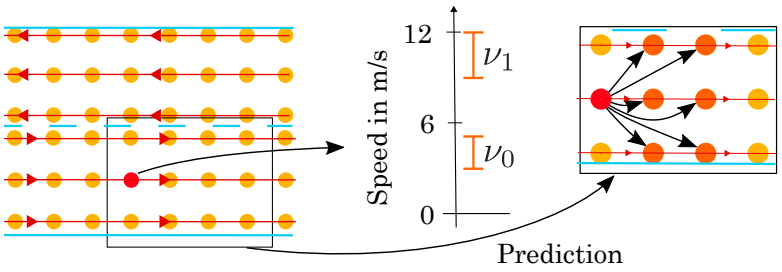
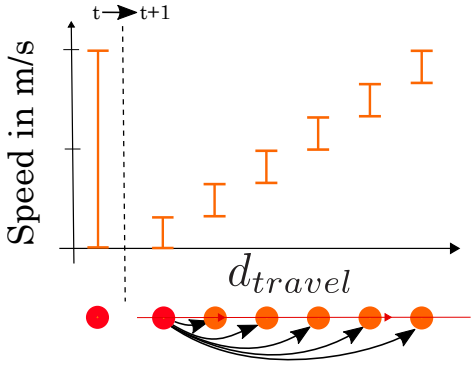


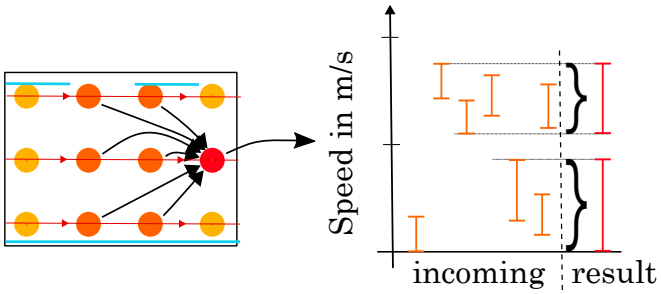
Figure 4.17: Depiction of the OST concept. The ego vehicle arrives from the bottom. Current elements (green) are predicted along the road network (red). New elements are spawned at the sources (orange). The ROI is shown as a black rectangle. In the update step, the predicted areas are truncated or invalidated using the FOV (yellow) and visible obstacles (red). (Graphic based on [7], ©2022 IEEE)



(a) Depiction of the lateral and longitudinal discretization of a road. The road markings are drawn in blue. The elements q_i are marked in yellow. Arrows indicate the direction of movement. Each element q contains speed ranges as depicted for the element highlighted in red. The right part shows how the speed ranges are propagated to the successors of the red element in longitudinal and lateral directions. The successors are drawn in orange.



(b) Distribution of a single speed range to successive elements during prediction. The speed range of the red element left of the dashed line is distributed to all successive elements and the element itself based on the speed, longitudinal resolution, and prediction time interval.



(c) Combination of all incoming speed ranges from the predecessors. Predecessors in the left part and incoming ranges in the right part are drawn in orange. The resulting speed ranges for the red element are also drawn in red.

Figure 4.18: Visualization of different details of the OST prediction steps.

$[min(v_{i,min}, v_{j,min}), max(v_{i,max}, v_{j,max})]$). This is shown in the right part of Fig. 4.18c for the incoming ranges after a prediction step. The speed ranges are sorted in ascending order so that $v_{i,max} < v_{i+1,min}$. The position and shape of q are determined through its position in the oriented grid and the grid resolution. The resolution consists of the lateral and longitudinal resolution. Since the arc lengths of the inner and outer sides are different in curves, the resolution is given for the centerline. The resolution also directly influences the calculation time as it determines the number of cells and, therefore, the number of calculations. Similar approaches like [149] are only capable of tracking in longitudinal direction.

All elements are assumed to be occluded in the **initialization**. The set of speed ranges $\mathcal{V}$ of every element consists of one speed range reaching from zero to the maximum allowed speed ($\nu_{init} = [0, v_{speedlimit}]\mathrm{m/s}$).

In the **prediction** step, the speed ranges of an element are propagated to its successors. The successors are determined in lateral and longitudinal directions as shown in the right part of Fig. 4.18a. All speed ranges of an element are propagated to its successive cells. Different speed ranges reaching the same target cell are combined (see Fig. 4.18c). This means that overlapping ranges are merged, and not intersecting ranges are added to the set of ranges for the target cell. New speed ranges are added at sources like the boundaries of the ROI. The prediction step is performed separately for every speed range of each element. First, the minimum and maximum travel distances d_{min} and d_{max} of an object with the corresponding range are

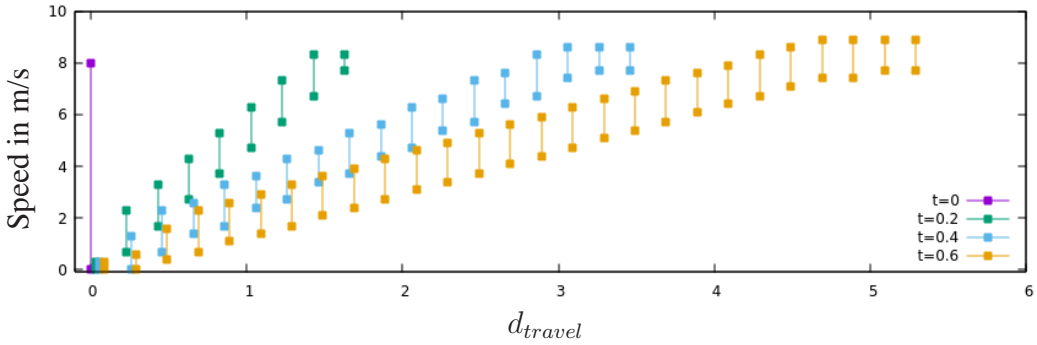


Figure 4.19: Speed ranges for a prediction example assuming only longitudinal motion and starting with a speed range $\nu = [0, 8]\mathrm{m/s}$ at $t = 0\,\mathrm{s}$ at a cell with $d_{travel} = 0\,\mathrm{m}$ (purple). The longitudinal resolution for the successive cells is $0.2\,\mathrm{m}$. The time interval is $0.2\,\mathrm{s}$. The speed ranges over the starting cell and its successors are drawn in different colors for each point in time. Bars for $t \neq 0$ are slightly shifted for better visualization by avoiding overlapping bars. d_{travel} corresponds to the distance in the longitudinal direction of subsequent cells. The initial speed range for $t = 0\,\mathrm{s}$ is drawn in purple. The prediction of the purple range yields the green speed ranges at the subsequent cells at $t = 0.2\,\mathrm{s}$. The prediction of the green speed ranges produces the blue ranges and so forth.

calculated first for propagating a speed range. Then, for each successor cell that is in the travel distance interval $[d_{min}, d_{max}]$, the speed that the object must have in order to reach the corresponding target cell is determined. This is shown in Fig. 4.18b where the distribution of the speed range of the origin cell over the successive cells during a prediction step is depicted. For successor cells at the minimum travel distance d_{min}, the object can only have the minimum of the speed range v_{min}. Otherwise, the object would have reached a target cell further away. Because of the discretization in the lateral and longitudinal direction with distance l between the cells, the speed ranges at the target cells ν_{target} are calculated as follows when starting at the origin cell with $\nu_{origin} = [v_{origin,min}, v_{origin,max}]$.

1. Determine $d_{min} = v_{origin,min} \cdot \Delta t$ and $d_{max} = v_{origin,max} \cdot \Delta t$
2. Calculate d_{travel} for target cell as distance from origin cell.
3. If $(d_{min} - l) < d_{travel} \leq d_{min}$ then
   $v'_{target,min} = d_{min}/\Delta t$ and
   $v'_{target,max} = l \cdot \lceil (d_{travel}/l) \rceil / \Delta t$
4. Else if $d_{min} < d_{travel} \leq d_{max}$ then
   $v'_{target,min} = \max(d_{min}, l \cdot \lfloor (d_{travel}/l) \rfloor)/\Delta t$ and
   $v'_{target,max} = \min(d_{max}, l \cdot \lceil (d_{travel}/l) \rceil)/\Delta t$
5. Else if $d_{max} < d_{travel} < (d_{max} + l)$ then
   $v'_{target,min} = l \cdot \lfloor (d_{max}/l) \rfloor / \Delta t$ and
   $v'_{target,max} = d_{max}/\Delta t$
6. Apply acceleration with
   $v_{target,min} = v'_{target,min} - \Delta t \cdot a$ and
   $v_{target,max} = v'_{target,max} + \Delta t \cdot a$
7. Obtain $\nu_{target} = [v_{target,min}, v_{target,max}]$.

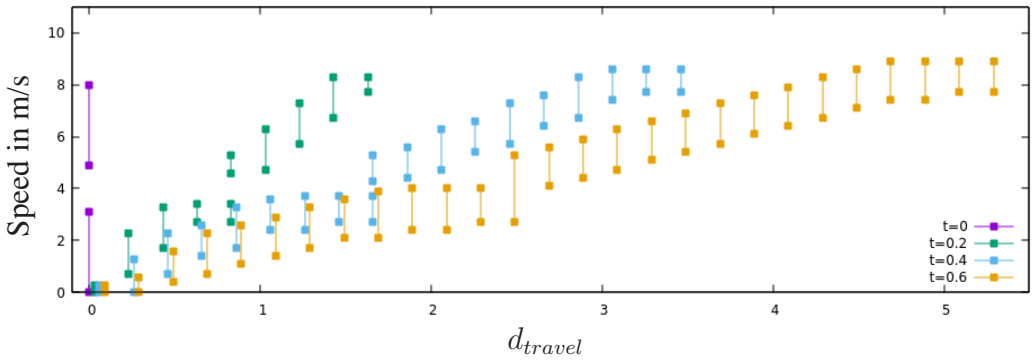


Figure 4.20: Speed ranges for a prediction example assuming only longitudinal motion and starting with two speed ranges $\nu_0 = [0, 3.1]\mathrm{m/s}$ and $\nu_1 = [4.9, 8]\mathrm{m/s}$. The rest is equal to Fig. 4.19.

While Fig. 4.18 depicts the concept, Fig. 4.19 shows an explicit example of the longitudinal propagation of a speed range to successive cells in the prediction step. The figure shows a speed range $\nu = [0, 8]\mathrm{m/s}$ being predicted with $\Delta t = 0.2\,\mathrm{s}$, what corresponds to an tracking cycle with $5\,\mathrm{Hz}$. The longitudinal resolution is $l = 0.2\,\mathrm{m}$ and the acceleration is $a = 1.5\mathrm{m/s}^2$. It can be seen how the ranges evolve and spread over time. The ranges reach cells that are further away with increasing time. Additionally, the individual speed ranges increase due to the applied acceleration and deceleration in each prediction step. It can also be observed that a speed range remains at the initial cell to account for very slow and standing objects and that, in total, the maximum speed is not exceeded.

In Fig. 4.20, the same parameters have been used. Two initial ranges are predicted with $\nu_0 = [0, 3.1]\mathrm{m/s}$ and $\nu_1 = [4.9, 8]\mathrm{m/s}$. It can be observed that for $t = 0.6\,\mathrm{s}$ (yellow bars), the predicted speed ranges intersect so that they are combined into one range. The intersection is a result of the maximum of the slower range accelerating and the minimum of the faster range decelerating.

The lateral motion is determined in relation to the corresponding longitudinal motion. It is assumed that there is a fixed ratio $\gamma_{lat,lon} = d_{lon}/d_{lat}$ for the upper bound on the shares of longitudinal and lateral motion. For $\gamma_{lat,lon} = 1$, objects move the same distance in the lateral and longitudinal direction. As the tracking is designed for objects that follow the road, $\gamma_{lat,lon}$ takes values ≥ 3. The assumption does not express the maximum allowed acceleration but is a design parameter for tracking the occluded areas. The oriented tracking is not designed for highly or purely lateral motions of objects. However, these are captured and tracked with the NST. The lateral motion is limited to both sides.

In order to account for objects that approach on the wrong side of the street, e.g., an overtaking vehicle, the lateral limits can be extended over the corresponding lane boundaries. This assumption is crucial because oncoming vehicles might be occluded due to a parking vehicle in their lane. As the ego vehicle is highly likely also occluded for the oncoming vehicle, it might try to overtake despite the approaching ego vehicle. This behavior would not be captured if the tracking area was fixed to the lane boundaries.

In comparison to Fig. 4.19 and Fig. 4.20 that focus on the longitudinal prediction of the speed ranges, Fig. 4.21 shows the longitudinal and lateral evolution in a 3D plot. As a geometric reference the road layout is shown in an underlying map. The visualization is generated in RViz. RViz is a visualization tool that is used in applications that are built with the Robot Operating System (ROS) as middleware. ROS is an open-source software framework for the development and operation of robotic platforms and autonomous vehicles. It is strongly supported by a large and active community. ROS allows the modularization and easy integration of different programs into a common software stack through nodes that contain different

functionalities and messages that exchange the information between them. The concepts in this work were also built using ROS.

In Fig. 4.21, the speed values are color-coded from yellow for low velocities to red for the maximum allowed speed. The initial state is a single speed range at the beginning of the lane. For all time steps, it can be observed that the occupied region never covers the area beside the starting range. This is because of the ratio $\gamma_{lat,lon}$. However, except for the beginning, the occluded area covers the complete wideness of the lane already at $t = 0.4\,\mathrm{s}$. With progressing time, the corresponding cells' speed ranges spread and grow, as was shown in Fig. 4.19 and Fig. 4.20. The darker red color for later time steps (Fig. 4.21e and Fig. 4.21f) indicates that the maximum speed of the speed ranges in the first row increases.

The **update** step is performed similarly to that explained above for NST. Each cell of the discretized lane grid is checked to see if it is occluded, occupied, or free using the FOV and the information about visible objects. If the cell is not occluded, the speed ranges are cleared. A grid-based FOV as described in Sec. 2.8.2 was used here.

A pruning step can be performed after the update using the FOV and other objects. The FOV often produces fragmentation of the cells due to the sparsity of Lidar measurements, meaning that there are lots of single occluded cells. Assuming a minimum length and width of expected occluded objects, these can be pruned by clearing all single cells or groups of neighboring cells that span a smaller area than the chosen threshold. This results in more accurate tracking results and better computational performance.

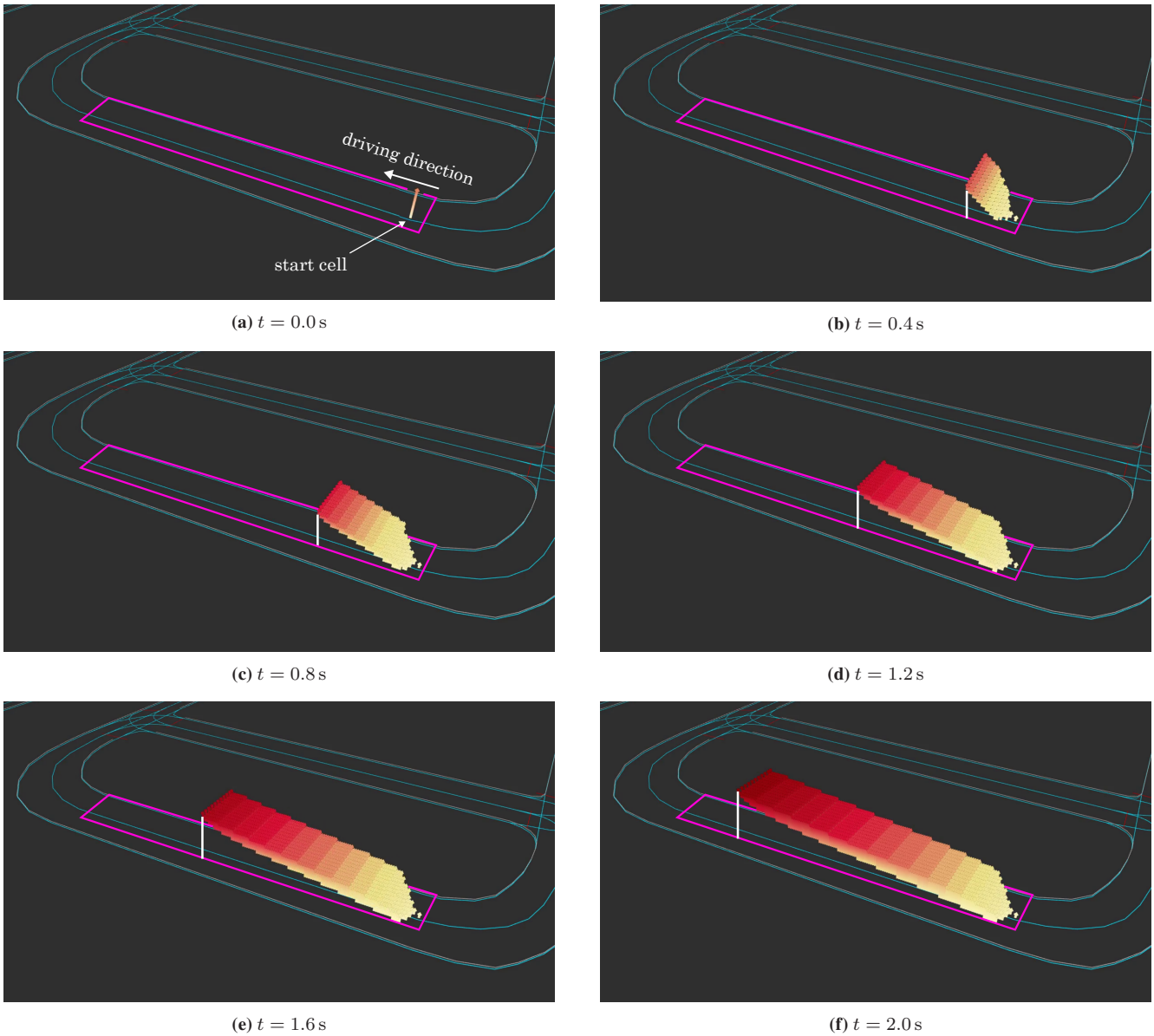
The **forecast** step repeats the prediction step until the desired time horizon is met. The forecast result is expressed with $\hat{\bar{Q}}$.

4.2.5 Evaluation

The feasibility of the tracking approaches was evaluated using a simulation. The evaluation aimed to ensure that hidden objects are always located in areas that are tracked as occluded as long as they are in the ROI. Therefore, the ground truth information is required, and thus, it was conducted in simulation. Visible or tracked objects can be located anywhere. The scenarios comprised blind intersections, pedestrians emerging from between parked vehicles, and occluded oncoming traffic through a vehicle in front of the ego vehicle.

Fig. 4.22 shows two of the scenarios in one example scene. The ego vehicle passes the parked vehicles, where the two occluded pedestrians appear. As it approaches the intersection, a previously hidden vehicle appears on the prioritized road. On the lane, that leaves the intersection in front of the ego vehicle to the right, it can be observed that the maximum speed increases with increasing distance from the visible area. Objects with higher speed at the first point that is not observable would have left the cell as shown in the examples in Fig. 4.19.

The evaluation in simulation proved the feasibility and functionality of both tracking approaches. Hidden objects were always in areas tracked by one of the occlusion tracking approaches, depending on their motion pattern. The evaluation also revealed that the chosen hyperparameters defining the assumptions on the motion of objects significantly influences the tracking accuracy. In this context, accuracy does not only refer to the probability of a hidden object being in an occluded area but to the size of the area that is regarded as occluded without any hidden objects inside. More accurate tracking of occluded areas allows for better decision-making based on it. On the other hand, underestimating the maximum speed of hidden objects leads to underestimating the occluded areas. This can lead to situations with higher risk.



(a) $t = 0.0\,\mathrm{s}$ **(b)** $t = 0.4\,\mathrm{s}$

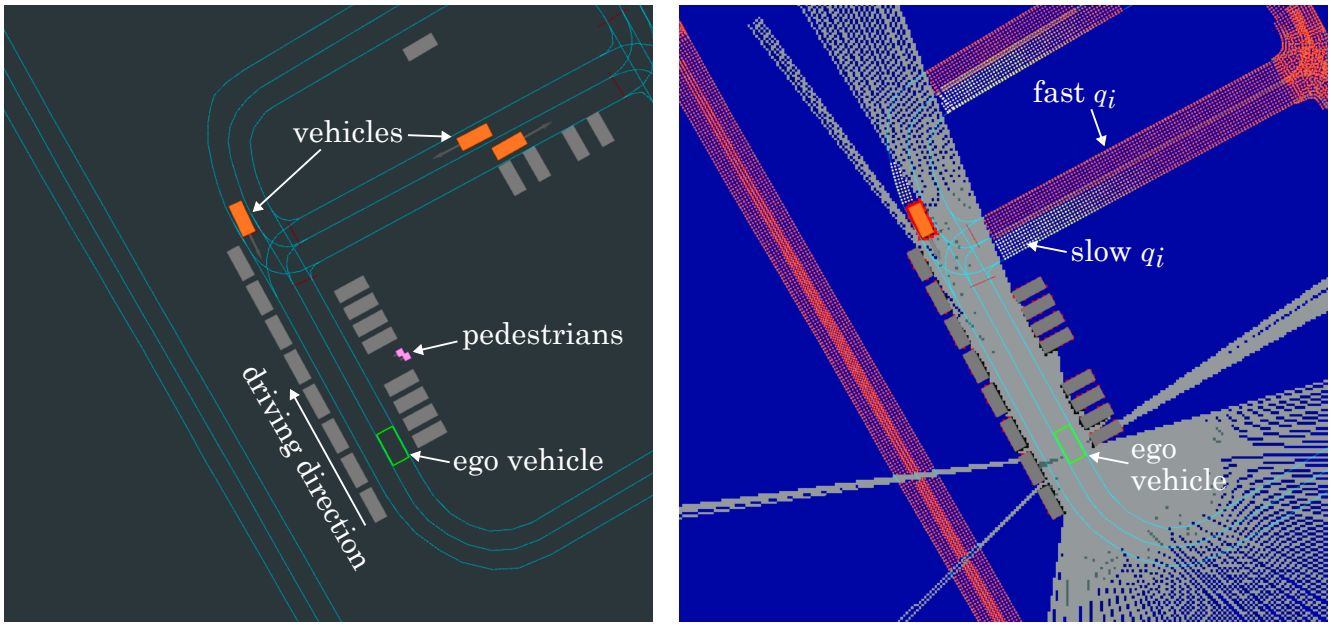
(c) $t = 0.8\,\mathrm{s}$ **(d)** $t = 1.2\,\mathrm{s}$

(e) $t = 1.6\,\mathrm{s}$ **(f)** $t = 2.0\,\mathrm{s}$

Figure 4.21: 3D visualization of the prediction of speed ranges using RViz. The initial range starts slightly shifted towards the center of the road and the beginning of a lane with $\nu = [0, 8]\mathrm{m/s}$. Speed values are color-coded from yellow for zero speed and red for the maximum allowed speed. A purple frame visualizes the boundaries of the area that is covered by the occlusion tracking for that lane. The left boundary is on the oncoming lane. So, the occlusions cover a part of the oncoming lane. White lines were added for the projection of the furthest cell, which contains speed ranges. The movement direction is from right to left.

Therefore, the hyperparameters should be chosen to slightly overestimate occluded areas. Guidelines are provided with information about speed limits and statistics about cycling speeds or speeds of pedestrians. It should also be considered to apply multiple parallel trackings to achieve accurate assumptions for different sets of parameters. Different parameter sets can represent different target groups with different probabilities of occurrence. For example, a parameter set could be defined for cyclists and vehicles obeying and exceeding the speed limit.

The application of tracking occluded areas in combination with the risk-aware trajectory planning counterpart is evaluated in Sec. 5.2.



(a) Ground truth view on the scene

(b) View of the ego vehicle and the tracked occluded areas

Figure 4.22: Example evaluation scene with occluded pedestrian and vehicle. The position of the ego vehicle is drawn as a green rectangle. The road layout is depicted in light blue. Unbound occluded areas $\mathcal{U}$ are shown in blue, and oriented occluded areas Q in yellow for a low maximum speed of the fastest speed range ($v_{N_v-1,max}$) to red for a high maximum speed of fastest speed range.

4.2.6 Conclusion

The core points of this section are listed in the following.

- Two approaches for tracking occluded areas relying on the same principles but different movement patterns were presented. OST tracks areas for hidden objects that move along roads, and NST keeps track of areas where objects move in arbitrary directions, which mainly occurs off the road.
- Accumulating available context information about occluded areas over time leads to more accurate assumptions about areas where hidden objects might be located. This extends the belief about the current state of the environment by occlusions.

Table 4.2: Parameters for the evaluation of the tracking of occluded areas

Parameter	**Value**
Frequency	$5\,\mathrm{Hz}$
ROI dimensions	$75\,\mathrm{m} \times 75\,\mathrm{m}$
NST resolution	$0.2\,\mathrm{m}$
speed values for NST	$0\,\mathrm{m/s}, 2\,\mathrm{m/s}, 4\,\mathrm{m/s}, 6\,\mathrm{m/s}$
OST resolution lateral	$0.4\,\mathrm{m}$
OST resolution longitudinal	$0.2\,\mathrm{m}$
min speed value OST	$0\,\mathrm{m/s}$
max speed value OST	$8.333\,\mathrm{m/s}$
min acceleration	$-2.0\,\mathrm{m/s^2}$
max acceleration	$2.0\,\mathrm{m/s^2}$
$\gamma_{lat,lon}$	3

4.3 Reasoning about Occluded Traffic Participants

The previous sections showed how intentions can be efficiently inferred from context information and how occluded areas can be tracked accurately. Based on the insights gained in these sections, this section presents an approach to infer the existence of a road user in an occluded area using context information and the motion of observable objects. The approach was proposed in [5] and the section may contain verbatim quotes.

The approach leverages the same concept as it was used to infer intentions in Sec. 4.1. However, this time the question is rephrased to: "How should the traffic scene have evolved if there was an occluded traffic participant?"

An example scene is shown in Fig. 4.23, where the ego vehicle approaches an intersection from the top. It is not on the prioritized road and it, therefore, has to react to the vehicles on the crossing street. The same applies to the oncoming vehicle. From the observations of the oncoming vehicle, the ego vehicle can infer if the oncoming vehicle sees a third vehicle that is not visible to the ego vehicle, as depicted in the figure.

4.3.1 Related Work

A lot of related work has already been presented in previous sections. Sec. 4.1.1 shows related work for prediction and intention estimation. Literature about tracking occlusions was presented in the related work in the last section 4.2.1. These works have in common that not all context is used to reason about

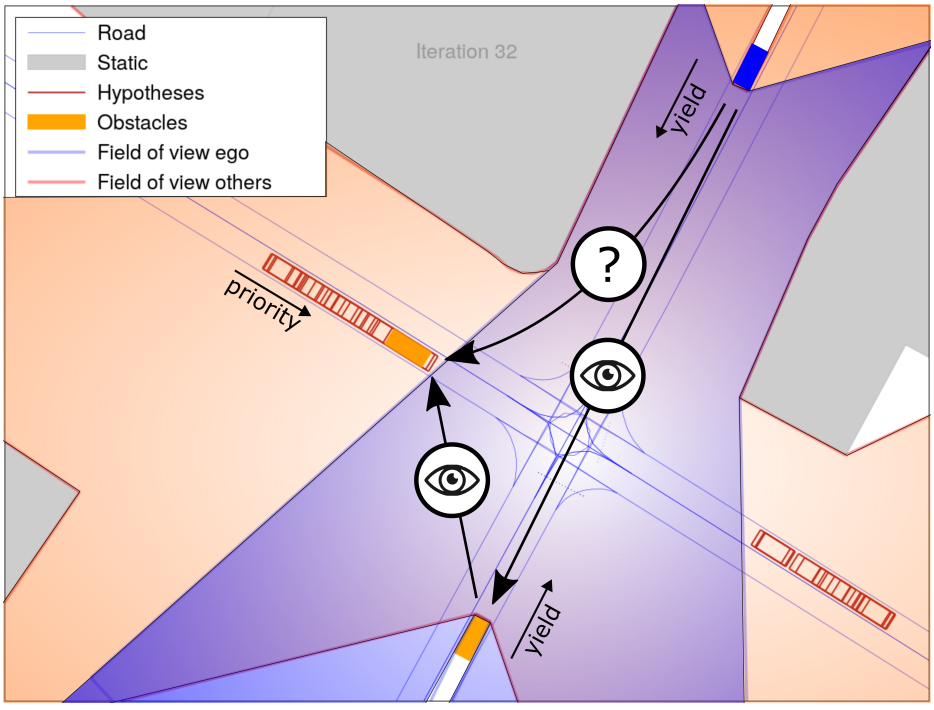


Figure 4.23: An intersection scenario is shown where the ego vehicle (blue) can infer the presence of the occluded vehicle from the left by observing the behavior of the oncoming vehicle from the bottom. Potential traffic participants are drawn as red contours on the prioritized road that goes from the top left to the right. (Graphic based on [5], ©2021 IEEE)

occluded traffic participants, especially the interaction with visible traffic participants. [62] shows that it is important to track already perceived vehicles through occluded areas. [155] and [142] use particle based assumptions about occluded traffic participants that are updated with the FOV. [149] estimates where potentially occluded road users can be located in the longitudinal direction and what velocity they can have with a set-based approach.

In contrast, this section presents an approach that explicitly leverages interactions of visible traffic participants with occluded traffic participants to infer their existence using the same approach as presented for infering the intention of traffic participants in Sec. 4.1.

4.3.2 Concept

The concept presented in Sec. 4.1.2 is extended to infer the presence of hidden traffic participants using the motion of visible ones as a measurement.

To achieve this, potential developments $\mathcal{D}_{\hat{S},i}$ need to be generated, where hidden traffic participants are included. In step 1), they are added to the state of the environment at $S_{-\tau}$ in occluded areas. These hypotheses about hidden road users are generated by the following process:

- Determine the FOV of the ego vehicle and the other traffic participants. Here, ray casting with a generic sensor setup is used. It can mimic the FOV of an automated vehicle or of a human driver.
- Determine the regions of interest, like prioritized roads or pedestrian crossings, where occluded traffic participants are supposed to influence the behavior of the corresponding visible traffic participants.
- Derive the areas in the region of interest that are occluded to the ego vehicle but visible to others.
- Make hypotheses about occluded traffic participants in that area by adding it to the corresponding state of the environment. Positions, velocities, and shapes are varied. The classification is also varied with the limitation that on pathways only pedestrians are assumed.

In step 2), alternative developments are generated - those with occluded traffic participants and those without. If a hidden traffic participant would have become visible for the ego vehicle during the prediction from $t = -\tau$ to $t = 0$ or if there would have been a collision with the assumed road user, the hypothesis and, therefore, the development is discarded. The prediction is performed using the prediction module described in Sec. 4.1.3.

The likelihood of each development is then inferred as described in steps 3) to 7) in Sec. 4.1.2.

As no measurements are available for the occluded traffic participants that were added in 1), they are consequently also neglected when determining the observation probabilities with respect to the actual development of the scene.

4.3.3 Evaluation

The approach to reason about the presence of hidden traffic participants is evaluated in two scenarios. In the first scenario, the presence of a vehicle on an oncoming road is inferred, and in the second scenario, the presence of a pedestrian is reasoned about.

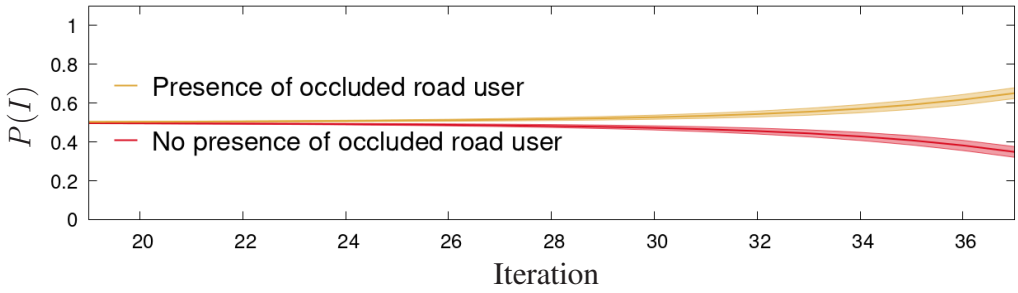


Figure 4.24: Likelihoods over time for the presence of an occluded vehicle on the prioritized road. (Graphic based on [5], ©2021 IEEE)

As for the evaluation of the intention estimation, 200 scenario runs were conducted with randomized simulation parameters, a number of seven estimation steps were used, and a time interval of $\Delta t = 0.3\,\mathrm{s}$.

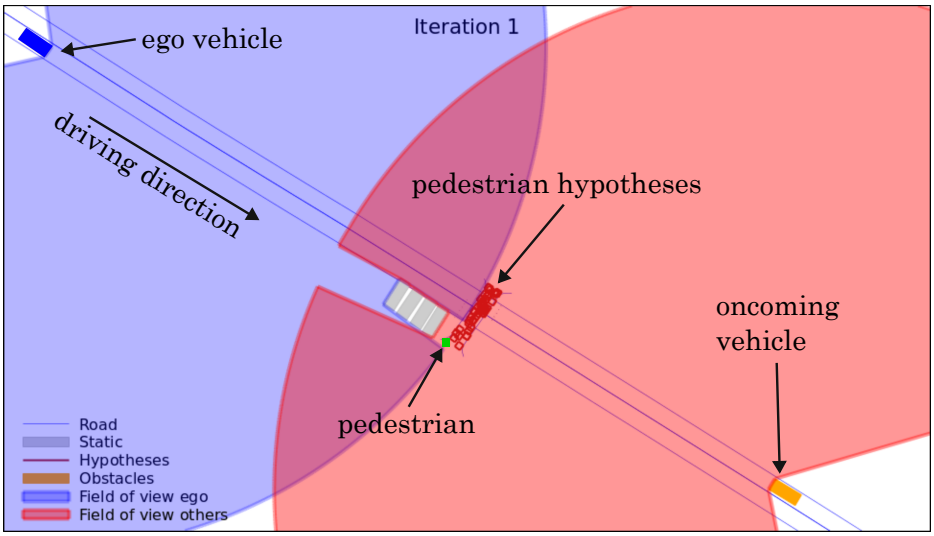
Scenario: Intersection with occluded vehicle

The first scenario is shown in Fig. 4.23, where the ego vehicle approaches an intersection where the prioritized lane is occluded. The corresponding estimation results are shown in Fig. 4.24.

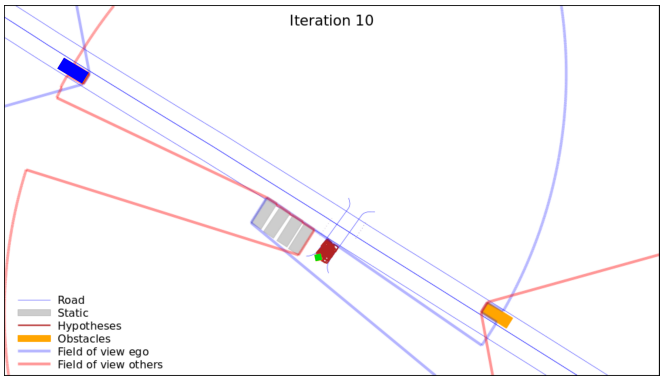
The ego vehicle can observe an oncoming vehicle with a clear sight of the prioritized lane. The FOVs are drawn in blue for the ego vehicle and in red for the oncoming visible vehicle. The ego vehicle generates hypotheses about hidden vehicles on the prioritized lane that are shown as red boxes. If any of the hypotheses are correct, the oncoming vehicle will have to yield. This is exactly what happens. Around iteration 20, the ego vehicle observes a deceleration of the oncoming vehicle. The oncoming vehicle decelerates to come to a stop at the intersection. Thus, it can be derived that there is a hidden vehicle on the prioritized lane. An additional but less likely development is that the oncoming vehicle wants to turn left and already waits for the ego vehicle that is still $55\,\mathrm{m}$ away from the intersection. At iteration 37, the ego vehicle can perceive the previously occluded vehicle in every simulation run. At that point, the likelihood of an occluded vehicle on the prioritized lane was estimated at 66%. The tendency already shows earlier, and that could lead the ego vehicle to release the throttle to safe energy. The fact that the oncoming vehicle also has to yield to the ego vehicle for a left turn also influences the estimation of the presence of the occluded vehicle at that point because the ego vehicle is only $25\,\mathrm{m}$ away from the intersection. In evaluation runs, where the oncoming vehicle started closer to the intersection, the estimation produced more confident results because the change in the behavior of the oncoming vehicle was more obvious. Additionally, the sensor range could be increased to allow an earlier observation of the oncoming vehicle and, thus, earlier changes in the ego behavior.

Scenario: Hidden pedestrian at pedestrian crossing

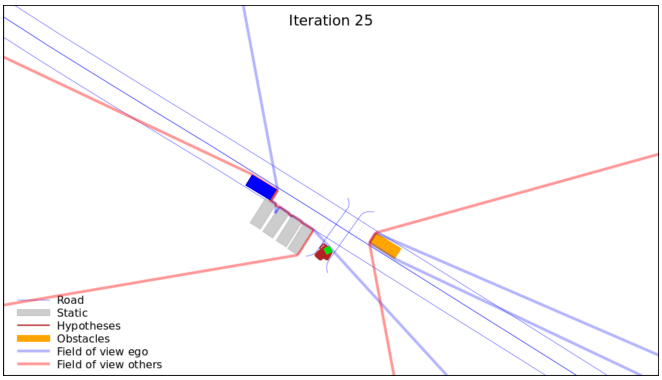
Fig. 4.25 depicts the second scenario with the corresponding likelihoods over time in Fig. 4.26. The ego vehicle and an oncoming vehicle approach a pedestrian crossing from opposite directions. A pedestrian is about to cross the pedestrian crossing, what can only be observed by the oncoming vehicle. The range of the FOV is reduced for drawing purposes. The ego vehicle can observe the oncoming vehicle from the beginning. Due to the observed braking behavior of the oncoming vehicle, the ego vehicle can infer the high likelihood of a pedestrian at the crosswalk early on. At around iteration 28, the pedestrian is



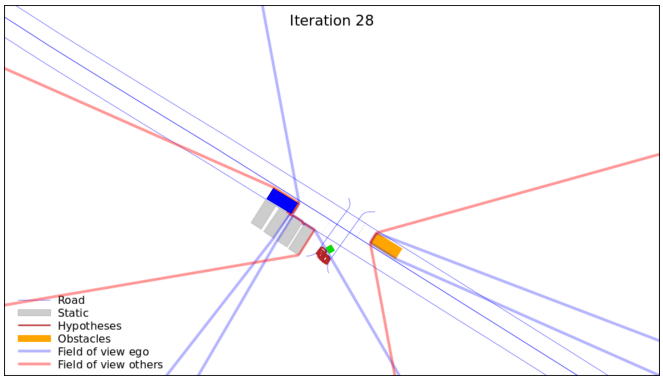
(a) Iteration 1



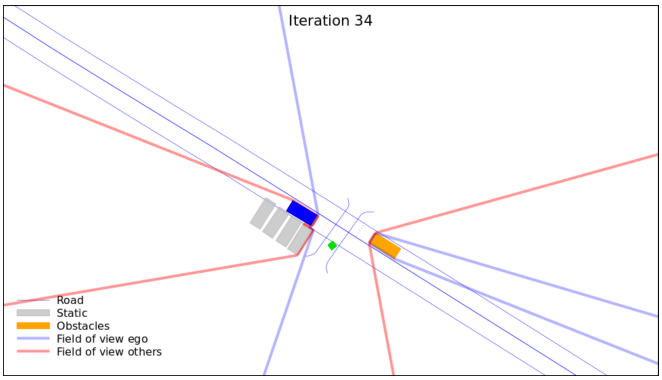
(b) Iteration 10



(c) Iteration 25



(d) Iteration 28



(e) Iteration 34

Figure 4.25: Evolution of the scenario with a hidden pedestrian at a pedestrian crossing. The pedestrian (green) is occluded for the ego vehicle (blue) but not for the oncoming vehicle (orange). Hypotheses about potential pedestrian shown in red contours. Different frames are shown (a)-(e), where both vehicle approach the pedestrian crossing. The FOV of the vehicles are only shown for the first frame for better visibility. In the other frames, the contours of the FOV are shown.

observed in every simulation run. At iteration 34, the ego vehicle stopped at the crossing, and it can fully observe the area around the crosswalk. The ego vehicle is quite sure that the oncoming vehicle stops for

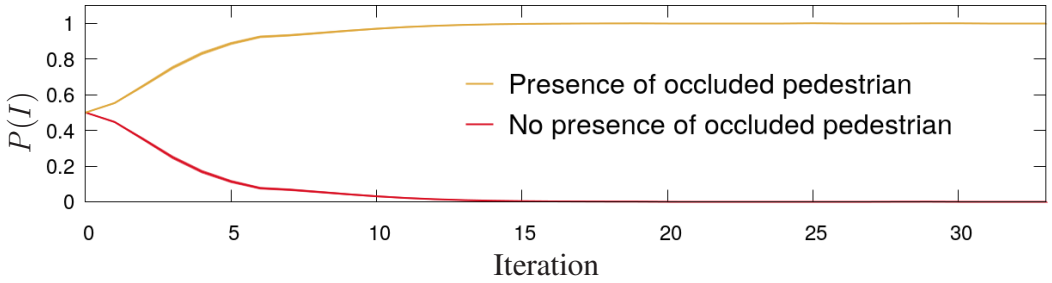


Figure 4.26: Likelihoods over time for the presence of an occluded pedestrian in the occluded pedestrian scenario from Fig. 4.25. (Graphic from [5], ©2021 IEEE)

an occluded pedestrian before iteration 5 when it is still $45\,\mathrm{m}$ away from the crosswalk with a speed of $10\,\mathrm{m/s}$. Thus, it can start to decelerate comfortably.

Discussion

The ego vehicle approaches the intersection and the pedestrian crossing carefully by default. Nevertheless, the information inferred from the stopping of the oncoming vehicle allows the ego vehicle to react foresightedly and to brake early and comfortably. If the likelihood for the occluded object is low, the ego vehicle could longer keep a higher speed and brake harder in case there is a hidden vehicle or pedestrian. In case there is no hidden road user, the ego vehicle would start accelerating for the desired speed from a higher start speed, which in total yields a higher efficiency. The inferred probability of the presence of the occluded traffic participant strongly influences the tradeoff between comfort, efficiency, and usability for everyday traffic. A comprehensible and understandable behavior of automated vehicles is required to achieve acceptance and a safe way to interact with automated vehicles. Hence, it is vital to abandon strict, overcautious rule-based strategies and use approaches like the one presented in this section to achieve behaviors that are safe and, at the same time, comprehensible and accepted by other road users. This is further discussed in Sec. 5.1, where a foresighted maneuver planner for occluded intersections is presented.

4.3.4 Conclusion

The key point of this section is that by considering how visible road users should have reacted in the presence or absence of an occluded road user, information about the existence of that road user can be inferred. This is achieved by applying a similar concept as in Sec. 4.1, where several possible developments from a past time are generated and then assessed by current measurements. The belief about objects in occluded areas can thus be further concretized.

4.4 Learning Uncertainty-aware and Occlusion-aware Predictions

In sections Sec. 4.1 and Sec. 4.3, approaches for enriching the belief about the current state of the environment have been presented by estimating the intentions of visible traffic participants and by reasoning about occluded traffic participants. Both approaches use a model-based prediction module to generate a variety of potential developments of a traffic scene and are independent of the specific prediction module. The model-based prediction of complex scenarios with many interacting traffic participants requires extensive modeling of the underlying assumptions and, as mentioned in Sec. 4.1, the number of developments may increase exponentially. In contrast, when learning from data, the intentions, motion models, and assumptions are learned implicitly and do not need to be defined explicitly. While the module presented in Sec. 4.1.3 is designed for intersection and urban scenarios, this section presents a model-free prediction of multi-lane traffic scenes using FCNs. A grid-based, sensor-agnostic input representation is used for a high level of generalization to arbitrary situations. The grid-based representation is built from multiple stacked input layers that encode the available environmental information about the road, traffic participants, and occluded areas. Thereby, all necessary information is gathered to fulfill the Markov Property to allow stepwise prediction in contrast to Recurrent Neural Networks (RNNs) that require a long history for accurate predictions. Accurate predictions for the near future are vital for foresighted decisions, especially in situations with limited sight due to occlusions. While machine learning techniques obtain very promising results for this task, the prediction's uncertainty is often neglected, and a perfectly known environment is assumed. Therefore, this section shows how uncertainties can be addressed in the learned prediction of traffic scenes. Additionally, a novel concept of iterative prediction is introduced to train and use neural networks to predict traffic scenes where no ground truth information is available due to occlusions. It is based on [2] and contains verbatim quotes from this work.

Learning predictions for occluded scenes brings several problems. In particular, supervised learning is impossible because no ground truth data is available in occluded areas. Furthermore, vehicles transitioning from visible to occluded areas can no longer be included in the loss function, leading the network to neglect vehicles in these areas. For this reason, a stepwise prediction is introduced to enable the network to predict traffic participants even through occluded areas. It is shown that this also enables unsupervised training.

This section is separated into two parts. First, the achieved accuracy for different loss functions and resolutions as well as the uncertainties in learned predictions, i.e., prediction inaccuracies and possible multi-modalities, are investigated in Sec. 4.4.4. Here, the epistemic uncertainty is determined using Monte Carlo dropout to approximate a Bayesian Neural Network (BNN). The second part presents the iterative prediction focusing on occlusions and unsupervised training (Sec. 4.4.5). In the end, an outlook is given on how occluded traffic participants that have not been observed so far can be anticipated using FCNs.

4.4.1 Related Work

Sec. 4.1.1 already presented an overview of model-based prediction approaches. Therefore, the focus here is mainly set on learned approaches.

In [51], Coué et al. present the Bayesian occupancy filter, a model-free approach to predict the environment using an occupancy grid representation. The grid is updated with observations that were previously made. Based on the work in [51], Bayesian occupancy filters were constantly extended and improved.

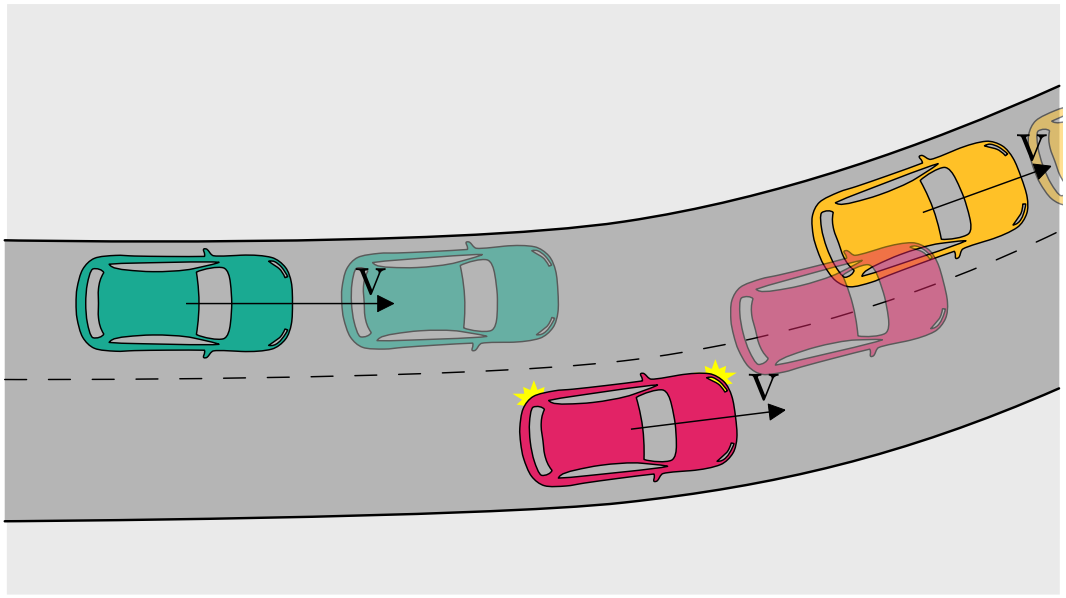


Figure 4.27: Predicting the movement of vehicles from time step t to $t + 1$ using a bird's eye view representation of a road segment. (Graphic from [2], ©2019 IEEE)

As examples, the explicit consideration of static cells to improve the prediction and applying particle filters to cope with the complexity can be named [117]. For the context of automated driving, Gindele et al. [65] guided the prediction by using map information to include knowledge about the main direction of movement. This resulted in a substantial improvement in the prediction accuracy for traffic situations. [131] gives an overview of further Bayesian occupancy filters.

The advantage of grid-based representations is their invariance against the number of traffic participants. The input size only depends on the size of the grid. Furthermore, free space is explicitly described by free cells. However, the previously mentioned grid-based approaches are not capable of properly modeling relations and interactions between traffic participants because the cells are hard to be associated with single agents like in the approach shown in Sec. 4.1. Instead of explicitly modeling the underlying assumptions and strategies to cluster cells to agents, learning-based approaches were proposed. Neural networks are used to understand and use the available context knowledge given in the input.

In [50], Caltagirone et al. apply a FCN to predict the path of a vehicle based on a grid representation, where additional information is stored in the grid-like multiple channels of an image. A further example is the work of Hoermann et al. [75] that predicts whole traffic scenes in an urban scenario using a CNN. The approach solely relies on fused sensor data from multiple sensors.

In [122], occupancy grids from Lidar measurements are used in combination with a RNN to track and predict the environment. The RNN learns an internal belief, accumulating all context information over time. The approach used simple shapes but was extended shortly after to urban scenarios in [56]. In contrast to the previously mentioned approaches, the ego vehicle's perspective was used. To compensate for the motion of the ego vehicle, the internal belief in the RNN was transformed using spatial transformers. Chauffeur Net [40] also uses RNNs. The network is used for trajectory planning for the ego vehicle. However, it requires a lot of information about previous time steps to achieve an inner encoding of the environment in the hidden layers of the RNN to produce good predictions reliably. While the inner belief can encode all necessary information with sufficient input data, the abstract inner states are hard or impossible for a human observer to interpret.

Neural Networks are also capable of modeling knowledge about uncertainties and the lack thereof. It has to be distinguished between the uncertainty in the networks' output and how confident the networks are about giving the correct result. For example, classification approaches yield the likelihood that an object belongs to a specific class. BNNs are able to directly represent uncertainties in each node by representing the state as a distribution instead of a single value. However, even though they can theoretically acquire and process probabilistic information, they are extremely hard to handle and are, therefore, not widely used. In [59], it was shown that epistemic uncertainty information can be extracted from arbitrary neural networks by averaging multiple forward passes using dropout. A description of epistemic uncertainty is given in Sec. 3.2.

To the author's knowledge, there was no approach to leveraging indicator information at that time. Furthermore, the approach presented in this section researched uncertainty-aware predictions and traffic scenes with occluded areas in particular using FCNs.

Since the research described in this chapter was conducted, GNNs became popular for motion prediction and outperformed other approaches on well-known leaderboards like the "Argoverse 2: Motion Forecasting Competition". Different types of GNN are used for the predictions, e.g., Graph Convolutional Networks (GCN) [96], Graph Attention Networks (GAT) [148], or Graph Recurrent Networks (GNN-RNN) [114]. Map information is often represented by polylines, i.e, VectorNet [63] or LaneGCN [107]. VectorNet and LaneGCN are one of the first approaches for motion prediction based on GNNs and the concept of polylines is used for the map representation in many works, e.g., [84]. As described in Sec. 4.1.1, [9] proposed a heterogenous graph to include relevant environment and temporal information in a single graph. In [68], the work has been extended by adding information through additional CNN inputs to include more details of the HD-Map, like walkways or pedestrian crossings.

Transformers also found their way into motion prediction [156, 103]. Approaches like Agentformer use the attention mechanism to accumulate attention over time and over other agents in a single transformer [156]. SEPT [103] uses a tensor that contains information about the n nearest traffic participants and VectorNet for the road information to predict multiple modes of trajectories and their probabilities.

4.4.2 Concept

It was mentioned above that grid-based representations are invariant against the number of traffic participants and directly encode free space as well as occluded areas. While the road network might be represented in a list of vectors or the traffic participants' states could be given explicitly with their position and velocity, the most suited representation for the occluded areas is the grid form. The grid explicitly provides information about every cell, so machine learning approaches can directly leverage information about occupied, occluded, and free areas. The approach presented here targets, in particular, traffic scenes with occlusion. Subsequently, a grid-based environment representation with multiple layers was chosen. The representation is sensor-agnostic, meaning that it is independent of the type of sensor as long as it provides the required (preprocessed) data. Sensors for automated driving were described in Sec. 3.1. The grid is discretized with a fixed resolution in the two-dimensional space. Unlike RNNs that build an inner belief, here, the Markov assumption is used. So, all required information for the prediction is given in the input. Like RGB channels for images, each property or feature of the input and output is encoded in a separate layer or channel. The chosen features are listed in the following:

- The **position** and shape of an object are described by the occupied area of the object. The occupancy value generally represents the probability of a cell being occupied.

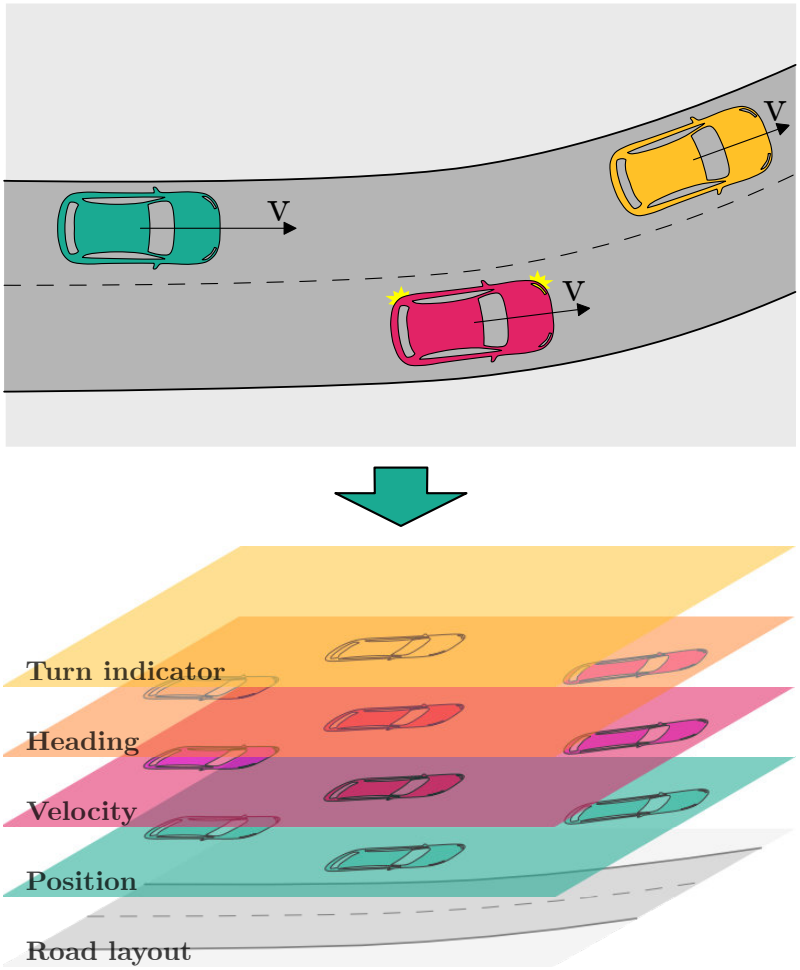


Figure 4.28: Representation of the input for the model with no occlusions. Road layout, vehicle position, speed and orientation as well as active turn indicators are stacked to form a multi-channel input, represented as occupancy grid. (Graphic from [2], ©2019 IEEE)

- The **speed** is the absolute value of the velocity of the corresponding object. The speed is also set for every corresponding cell. The value is normalized to the interval $[0, 1]$ with respect to a maximum speed value. Empty cells have a speed of zero.
- The **direction** of the velocity vector represents the movement direction of an object in the interval $[-\pi, \pi]$. The direction is also normalized to the interval $[-1, 1]$ by dividing by π.
- The **road layout** provides valuable information for the prediction. Traffic participants are assumed to follow the road. Thus, the road layout guides the prediction in feasible directions. The road layout is given by maps and is transferred into the grid.
- **Turn signals** are encoded using ± 1 for active right or left turn indicators, respectively, and 0 otherwise. All cells of the corresponding vehicle are marked accordingly.
- **Occluded areas** are explicitly included. Hence, the network knows where no information about the objects is available. These areas are marked in the position and road layout layers.

As mentioned above, in the next part, a base network is used to analyze the effect of different loss functions, uncertainty-aware predictions via BNN, the impact of the resolution, and deep residual connections. The network output is the traffic participants' position at the next time step. Hereby, a single prediction step

is used while not assuming any occlusions. A dilemma arises regarding occlusions. If the network is trained supervised with occlusions, the ground truth information about the objects in the occluded areas is included in the target. Thus, it cannot be guaranteed that the network ignores the occluded areas. When using unsupervised training, occluded areas are excluded from the loss calculation. The network does not learn to predict vehicles inside this area because there were never vehicles in the occluded areas in the target.

Therefore, a novel iterative prediction is proposed and discussed in the second part of the following section. The networks used for the iterative prediction predict the position and velocity at the next time step. This way, the Markov assumption holds and the output of a prediction step can be used as input for the next step. The iterative prediction solves the aforementioned dilemma by forcing the network to predict vehicles through occluded areas until they are visible again. Hence, the network learns to understand the meaning of occluded areas.

Next, the data generation process is described before the two concepts are presented and evaluated.

4.4.3 Data Generation

When the research was conducted, no dataset that contained all the required information, i.e., indicators, was available. So, the data was generated synthetically. As available open source traffic simulations either focussed on modeling the traffic flow or the vehicle dynamics but not on both, a simulation based on the IDM [145] in combination with the lane change extension MOBIL [94] was developed in [81] and extended in [29]. The generated data properly represents maneuvers such as lane changes to provide realistic vehicle movement data for highway scenarios. Furthermore, the simulation allowed access to ground truth data and augmentations with occlusions. As the input representation of the network is sensor-agnostic, the simulation data can be directly fed into the network. In the following, a short introduction to the simulation is given. For further details, the reader is referred to [81] and [29].

In the simulation, all agents are individually controlled. German traffic rules were applied, but other countries' rules could also be modeled. The agents consist of different types of vehicles, trucks, sports cars, and so on to obtain a realistic traffic composition. The types differ in their parametrization to mirror the specific behavior of the class. For example, a truck's maximum speed and acceleration are much lower than for a sports car, and the sports car is more likely to perform lane changes to make progress faster. The IDM is used for the longitudinal control. The vehicle in front is chosen with respect to the current intention. I.e., on lane changes, also the vehicle on the target lane is considered. MOBIL determines the lateral motion and, in particular, lane changes based on the criteria of safety and incentive. The safety criteria check if there is no collision and the vehicle behind the ego vehicle is able to decelerate with a certain maximum deceleration. However, the model does not contain a heuristic for using turn indicators before making a lane change. Therefore, MOBIL was modified to indicate a lane change as intended by the traffic rules. The lane change process was split into two parts. First, it is determined if an agent pursues a lane change. In case the agent wants to change the lane, the corresponding indicator is set for a certain waiting time. After the waiting time has passed, the lane change decision is rechecked to see if it is still desirable and safe to change lanes.

In each simulation step, the acceleration of each vehicle is calculated, and the velocity and position are updated accordingly. Afterward, the lane change decision is performed as described above. The order is determined randomly to prevent multiple vehicles from changing lanes to the same lane at the same time. Vehicles are assumed to not leave the road and indicate their lane change before execution.

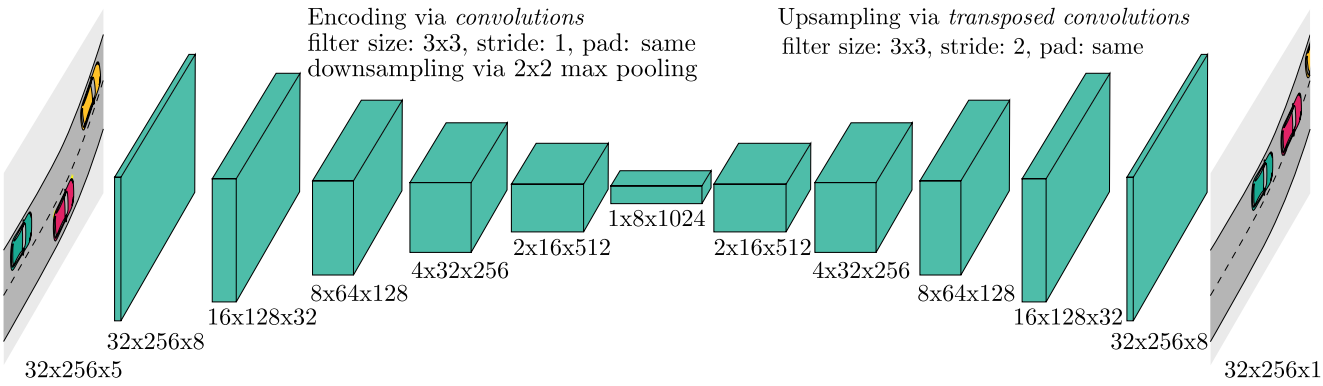


Figure 4.29: Fully convolutional layout of the base network. The feature extraction is performed using consecutive 3x3 convolutions with stride one that retains the input dimension (padding=same). The dimension is reduced using max pooling layers after each convolution layer. Upscaling is achieved through transposed convolutions with stride 2. (Graphic from [2], ©2019 IEEE)

Data is recorded from the simulation using a birds-eye perspective at varying angles at different locations throughout the test tracks of various road types (lane mergers, blocked and incoming lanes) over an area of $200\,\mathrm{m} \times 25\,\mathrm{m}$. A total of 46 hours of training data was used with a frame rate of $0.5\,\mathrm{s}$.

4.4.4 Single-Step Predictions

The input representation is depicted in Fig. 4.28. The grid-based representation is ideal for using convolutional layers for feature extraction while maintaining the spatial context. A FCN architecture was chosen to be independent of the input resolution and because the output also is an image-like representation of the environment at $t+1$ with the same resolution as the input. The network architecture is shown in Fig. 4.29.

Feature Extraction and Upscaling

For the feature extraction in the encoder, 3x3 convolution layers with ReLU activation function followed by 2x2 max pooling layers for dimension reduction are used until at least one dimension is reduced to one to obtain a fully concentrated latent representation of the traffic scene. With an input size of 32×256 cells, six couples of convolution and pooling layers are required because the dimension is halved after every pooling layer. The number of feature maps and, therefore, filters are increased after every pooling layer. A brief discussion on the kernel size can be found in Sec. 2.9 and a review of the advantages and disadvantages of smaller filter kernels over bigger ones was conducted in [140]. Max pooling was chosen because preevaluations in [81] showed that a higher performance could be achieved in comparison to convolution layers with stride $s > 1$. Max pooling layers pick the most important information first and imply better generalization.

In the decoder or upsampling part, the initial dimensions are restored using transposed convolutions as described in Sec. 2.9.

Loss Specification

The importance of the loss function for the training and the network performance was discussed in Sec. 2.9. Here, the loss function aims to penalize the network for deviations of the output from the target state of the environment at the next time step. This includes, in particular, predicting the vehicles inaccurately, adding new vehicles, and omitting vehicles. While the comparison on the entity level allows a straightforward understanding for the human observer, it is not feasible to calculate them for each iteration in the training process. The training process would take too long, and information like the size of the vehicles would be lost, leading to deteriorated gradients. Thus, suitable loss functions were developed. These allow an efficient training process, can be easily calculated in parallel on a GPU, and are directly applicable to the grid-based structure. The entity-level comparison will later be used in the evaluation for human interpretable results. For the applied loss functions, the MSE and IoU loss functions described in Sec. 2.9 were extended to a pixel-wise calculation. For the definition of the loss function, the following definitions are used:

- N is the number of samples in a mini-batch,
- x_i is the input sample i,
- y_i is the true label or target for input x_i,
- $\hat{y}_i$ is the output of the neural network for input x_i,
- x_{ijk}, $\hat{y}_{ijk}$, and y_{ijk} are the pixel values at position (j, k) for sample or target i.

Pixel-wise intersection over union

In Sec. 2.9, it was described how the IoU is used for image segmentation for the example of overlapping bounding boxes. To transfer it to a pixel-level, the IoU as proposed for regression tasks by Gygli et al. [69] is used. The element-wise operators for minimum and maximum are used to replace the set operators for intersection and union, respectively:

$$\hat{y}_i \cap y_i = \sum_j \sum_k min\left(\hat{y}_{ijk}, y_{ijk}\right), \tag{4.10}$$

$$\hat{y}_i \cup y_i = \sum_j \sum_k max\left(\hat{y}_{ijk}, y_{ijk}\right). \tag{4.11}$$

The pixel-wise IoU results to

$$IoU = \frac{\hat{y} \cap y}{\hat{y} \cup y}. \tag{4.12}$$

In this form, the IoU would penalize a high resemblance between the output $\hat{y}_i$ and the target y_i. When the intersection and union overlap perfectly, the IoU yields values close to 1. For no overlapping, the IoU is zero. Therefore, the IoU has to be reversed to obtain a loss function that can be minimized. In case there is no vehicle in the scene, the denominator is zero. Thus, this case is handled separately by

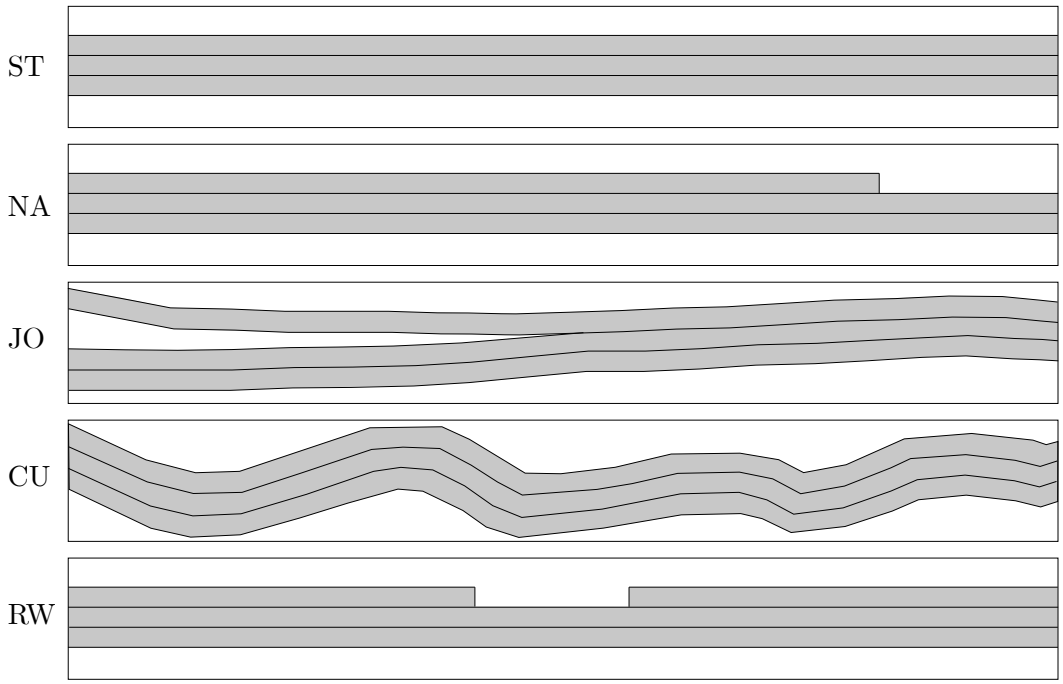


Figure 4.30: Evaluation road layouts. From top to bottom: straight (ST), narrow (NA), join (JO), curves (CU), roadworks (RW). (Graphic from [2], ©2019 IEEE)

assuming a loss value of zero. To stronger penalize larger deviations, the IoU term is squared. The loss function based on the pixel-wise IoU finally is defined as

$$\mathcal{L}_{IoU}\left(\hat{y}_i, y_i\right) = \begin{cases} 0 & \text{, for } \hat{y}_i \cup y_i = 0 \\ 1 - \left(\frac{\hat{y}_i \cap y_i}{\hat{y}_i \cup y_i}\right)^2 & \text{, else.} \end{cases} \tag{4.13}$$

Pixel-wise mean squared error

The pixel-wise MSE follows the principles of the regular MSE and calculates the mean squared error for each pixel

$$\mathcal{L}_{MSE} = \frac{1}{N} \sum_{i=0}^{N-1} \sum_{j=0}^{J-1} \sum_{k=0}^{K-1} \left(\hat{y}_{ijk} - y_{ijk}\right)^2, \tag{4.14}$$

with $J \times K$ as input dimension.

Post-processing

The network inputs and outputs the positions of the vehicles in a two-dimensional grid representation. The position channel of the input and output are post-processed to determine the prediction error on the vehicle entity level, which allows a direct interpretation of the prediction results. Thus, the occupied cells are clustered to reobtain the shape of the vehicles. Afterward, the vehicles from the input and output are matched to determine their positional error. Furthermore, the total number of vehicles, as well as omitted or added vehicles, are determined.

Evaluation of Single-Step Prediction

The evaluation of the single-step prediction focuses on the general performance and possible modifications to further improve the performance.

New road layouts were created to obtain new evaluation scenarios that the networks have not seen before. The layouts are shown in Fig. 4.30 and are ordered by their complexity, starting from straight roads (ST), ending (NA) or incoming lanes (JO), extremely curvy roads (CU), and a partially blocked lane (RW).

The performance results on the corresponding test layout for the single step prediction of $\Delta t = 1\,\mathrm{s}$ are listed in Table 4.3. The average prediction error of all vehicles per network configuration and road layout is given in meters. Additionally, the proportion of vehicles missed or falsely added is given. The area of the corresponding side, where vehicles might enter the scene between t and $t + 1$, is excluded from evaluation. The parameters used for training and evaluation are listed in Table 4.4. All models were implemented using TensorFlow [32].

Table 4.3: Results of different network configurations for single-step prediction on all test tracks. Errors are listed in meters. For missed or added vehicles, average values over all tracks are given in percentages. (Table from [2], ©2019 IEEE)

	ST	**NA**	**JO**	**CU**	**RW**	**miss**	**add**
FCN w/ IoU	0.76	0.87	1.19	1.32	0.99	2.4	2.6
FCN w/ MSE	0.87	0.91	1.12	1.44	1.08	2.2	14.0
deep residual	**0.45**	**0.48**	**0.73**	1.05	**0.59**	**0.4**	**0.2**
high resolution (512×64)	0.63	0.59	0.78	**0.99**	0.75	2.0	4.6
no turn signals	0.75	0.75	1.12	1.22	1.00	1.4	2.4
BNN	0.97	0.65	1.16	1.02	1.00	2.0	5.4

Table 4.4: Parameters used for training and evaluation

Parameter	**Value**
learning rate	0.00001
batch size	64
optimizer	Adam
activation function	ReLU
weight initialization	modified Xavier [72]
Δt	$1.0\,\mathrm{s}$
size of region of interest	$200\,\mathrm{m} \times 25\,\mathrm{m}$
resolution	256×32 pixel
Δt_S	$0.5\,\mathrm{s}$
Δt_I	$1.5\,\mathrm{s}$
number of prediction steps	3

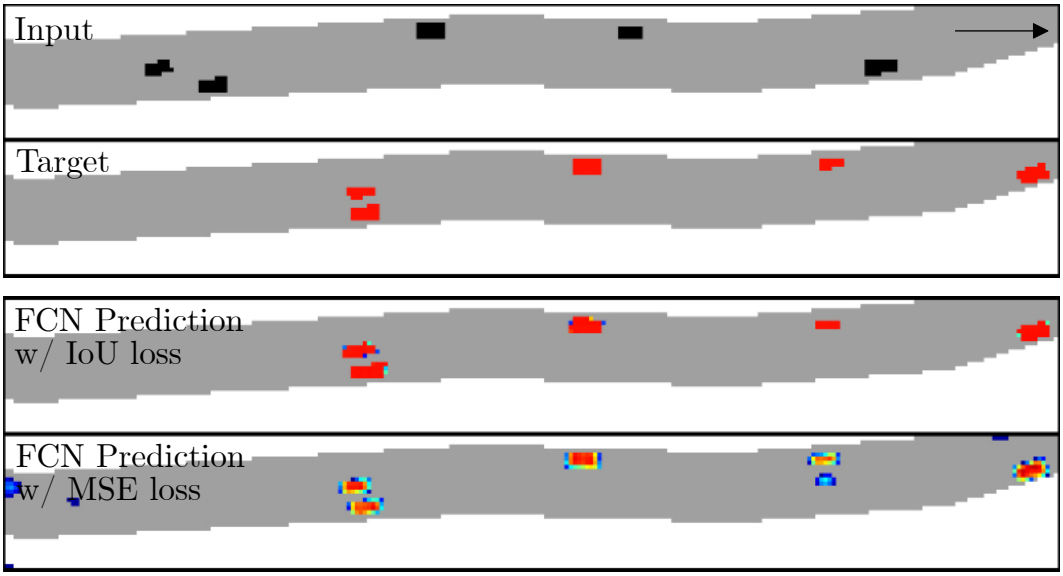


Figure 4.31: Prediction output using the FCN baseline model with different loss functions. From top to bottom: input positions, target positions, FCN prediction using IoU loss, FCN prediction using MSE loss. The driving direction is from left to right. (Graphic from [2], ©2019 IEEE)

An exemplary traffic scene is shown in Fig. 4.31. At the top, the input and target position channels are drawn. The road layout has only been added for a more straightforward interpretation. As mentioned above, the road layout is not part of the network output. Vehicles in the input are depicted in black with yellow indicators on the respective side. Vehicles in the target are drawn in red. In the network output, the vehicles are drawn in the color of the prediction probability of a cell being occupied from high probabilities in red over yellow to low probabilities colored in blue. In Fig. 4.31, the lower part shows the comparison of the FCN output between the pixel-wise IoU loss (top) and the pixel-wise MSE loss (bottom) respectively.

Loss evaluation

Both loss functions, the pixel-wise IoU and the pixel-wise MSE, are suitable for training. A comparison is shown in Fig. 4.31. The trained networks were able to predict the traffic scene accurately. However, the loss functions have different particularities. The networks trained with the pixel-wise IoU produced sharper edges. It is assumed that this results from the goal of the IoU to match the shapes optimally. On the other hand, the networks trained with pixel-wise MSE yielded more uncertain predictions. The network learned to expect new vehicles to enter the scene on the correct side even though the edges are excluded from the loss calculation. This can be seen on the left side of the lowest image in Fig. 4.31. The network also indicates that the first vehicle on the left-most lane might be about to change lanes.

Bayesian Neural Network

Fig. 4.32 shows the prediction of a traffic scene using Monte Carlo dropout as proposed by [61]. To compensate the effects of 50% dropout, the network was trained with a doubled filter size in each layer. Dropout was thereby used both during the training and during inference. To calculate the uncertainty in the prediction, 50 forward passes were used to calculate the mean value and variance for each pixel

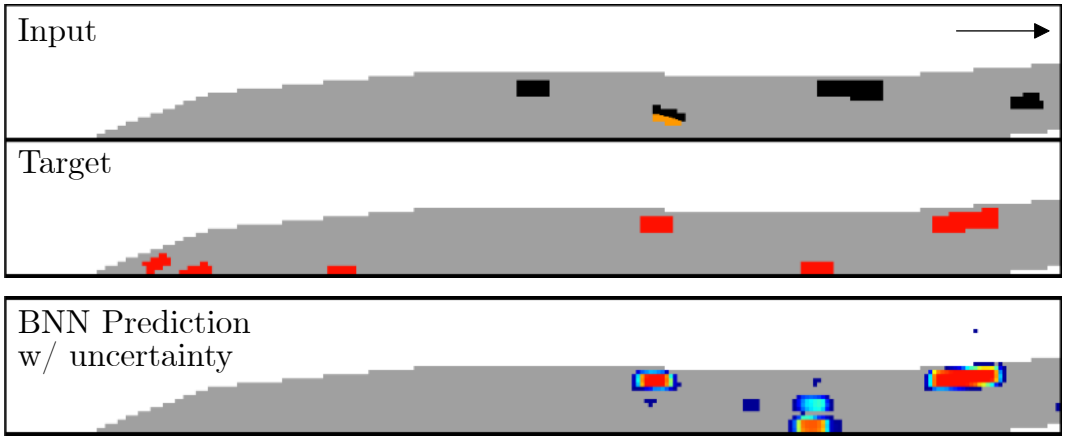


Figure 4.32: Bayesian Network epistemic uncertainty prediction output by forward pass sampling as described in [61]. The second car from the left has its right turn indicator activated, showing nicely that the resulting prediction is uncertain about the outcome, which results in multi-modalities in the output. The driving direction is from left to right. (Graphic from [2], ©2019 IEEE)

separately. The uncertainty in the prediction increases and the nominal performance is worse than the plain FCN. It has to be noted that the post-processing of the multi-modal outputs is a lot more challenging task than for single modalities.

However, the network is capable of predicting and anticipating multi-modalities and better generalizing the motion possibilities of the vehicles. For the second vehicle from the left, which indicates a lane change from the center to the right lane, the network predicts a non-negligible probability of the vehicle aborting the lane change. These uncertainty-aware forecasts are important, i.e., for safe maneuver decisions. Given this information, a following vehicle could anticipate the cut-in and extend the safety distance.

Deep residual connections

To examine the influence of the depth of the network, the baseline was compared to a deep residual network. It is known that the depth of the network has strong effects on its performance while making it more challenging and demanding to train because the number of weights increases. The base network was modified with 20 convolutional layers with max-pooling after every four layers in the encoder and ten transposed convolutional layers for upsampling accordingly. To overcome shortcomings due to vanishing gradients in the deep network, skip connections [71] were added to better transport gradients. Table 4.3 shows that overall, the deep network outperforms all other networks. Except for the curvy road, the deep network excels on every course. The error reaches values that are below the size of a single cell. It can be concluded that resolution is the limiting factor for deep networks to achieve higher precision.

Impact of input/output resolution

The impact of the input and output resolution was also evaluated. The pixel resolution was doubled in height and width to 512×64 while the size of the region of interest was kept the same. Thus, the number of pixels quadrupled. As was expected, the processing and training time increased. However, this came with the benefit of more accurate predictions as shown in Table 4.3. The network performed best on the curvy road layout. A possible explanation is the orientation of the vehicles. At a higher resolution,

the cells that are regarded as occupied by the vehicle can be chosen more finely. When the vehicles are oriented diagonally, the number of partially occupied cells is even higher than for horizontal or vertical orientations. The improvement through increasing the resolution is supposed to be limited. At some level, the accuracy will be limited by other factors like the depth of the network and, for real-world data also, the accuracy of the input data.

Turn signals

The evaluation of the influence of the turn signals in the input showed that it did not influence the network performance. The base network was compared to a network that was trained without the turn indicator input channel. The results in Table 4.3 suggest that the network trained without turn indicator information implicitly learned the behavior of the vehicles even without the extra cues. The network learned to interpret the available context information correctly. It predicted the lane change decisions not based on extra hints but on correctly drawn conclusions from the available data. It can be followed that it is more important to have accurate position and velocity information than indicator information, which is hard to get for real-world data and especially for data collected from a bird's-eye view.

4.4.5 Iterative Predictions

The iterative prediction profited from the insights gained in the single-step prediction. An evaluation of the influence of the direction layer in the single-step prediction showed only a marginal contribution. The network learned that the vehicles follow the road layout that is given as a separate channel. For this reason, the direction layer is dropped for the iterative prediction. Additionally, the indicator channel was not used because the network also learned the behavior of the vehicles, as was explained earlier. In initial experiments, it was found that the networks omit vehicles in occluded areas for a single-step prediction when the vehicles are not included in the target. Furthermore, occluded areas in the input were ignored when the target included ground truth data about objects in occluded areas. As described above, the primary idea behind the iterative prediction is to make the network learn to track vehicles through occluded areas. Therefore, occluded areas are encoded in the input representation. To keep to the Markov property for the iterative prediction cycle, the output was extended to also include the velocity of the objects. Static knowledge like the road network or unknown areas can be reused.

The iterative cycle is shown in Fig. 4.33. A fixed reference position is used for every prediction step to keep the spatial context. The feature extraction and upscaling are performed similarly to the single-step prediction described in Sec. 4.4.4. However, the number of filters is doubled to account for the increase in the output dimension. Additionally, the loss calculation must be modified to account for multiple prediction steps.

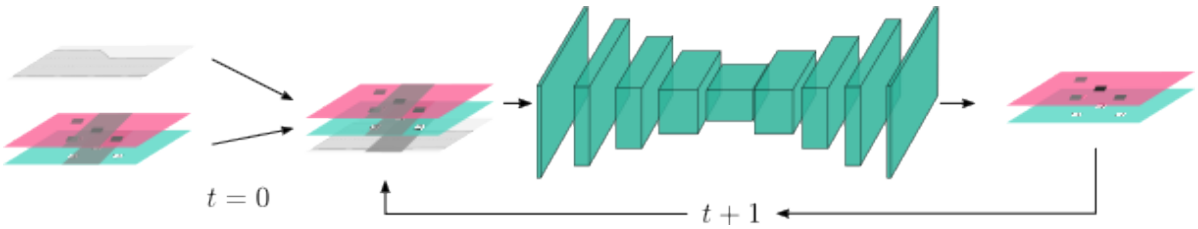


Figure 4.33: Layout of the iterative prediction. Output information about occupancy and velocity is combined with static knowledge and serves as input for the next prediction. (Graphic based on [2], ©2019 IEEE)

Loss Specification

Besides pixel-wise IoU and MSE like for the single-step prediction, a combination from both was evaluated for the iterative prediction process. The position loss, thereby, was calculated with the pixel-wise IoU and the velocity with the pixel-wise MSE as described above. The underlying idea is, that the pixel-wise IoU yields sharper edges of the vehicles, while the speed is a floating value that is better valued with the pixel-wise MSE. To combine the two loss functions, $\mathcal{L}_{MSE}$ had to be scaled as the $\mathcal{L}_{IoU}$ only produces values in the interval $[0, 1]$. With scaling factor γ, the combined loss function results to

$$\mathcal{L}_t = \mathcal{L}_{t,IoU} + \gamma \, \mathcal{L}_{t,MSE}, \tag{4.15}$$

where t denotes the corresponding time step. The total loss $\mathcal{L}_{it}$ over all time steps t is the sum over all partial losses $\mathcal{L}_t$

$$\mathcal{L}_{it} = \sum_{t=1}^{N_t} \mathcal{L}_t. \tag{4.16}$$

The loss was calculated only for non-occluded areas in the unsupervised training.

Evaluation of the Iterative Prediction

The post-processing was conducted the same way as described in Sec. 4.4.4 for the single-step prediction. The parameters used are listed in Table 4.4. The display of the predicted traffic scenes is also kept identical with occluded areas visualized in darker gray. The emphasis of the evaluation of the iterative prediction was set on the comparison between single-step and iterative prediction as well as handling occlusions. The results are listed in Table 4.5. For the evaluation, the complete scene, including occluded areas, is considered. Thus, the number of missed vehicles is higher. As the output of the iterative process also includes the velocity, it is also regarded in the evaluation. The clusters in the velocity channel are compared using different calculations of the average velocity of a cluster. The velocity errors are given in $\mathrm{m/s}$. As for the results above, the average prediction error per network configuration is given in meters, and the proportion of missed or mistakenly added vehicles is given in percentages.

Iterative training

The hypotheses for using an iterative prediction process were, that the network becomes aware of the meaning and importance of occluded areas and that the iterative prediction achieves better overall performance. To compare single-step and iterative predictions, a time interval of $\Delta t_S = 1.5\,\mathrm{s}$ and three consecutive steps of $\Delta t_I = 0.5\,\mathrm{s}$ were used accordingly. An example of an iteratively predicted scene is depicted in Fig. 4.34 for three steps. The network learned not to discard the vehicles moving through occluded areas but instead keeps to predict them. In Fig. 4.34, this can be observed for the truck in the center of the scene.

The prediction results also improved in comparison to a single-step prediction, as can be seen in Table 4.5. Both, the distance error and missed/added vehicles achieved better results. However, this comes at the cost of higher runtimes. As the forward passes can also only be executed iteratively and not in parallel in

Table 4.5: Average results of different network configurations on all test tracks. "S" indicates single-step and "I" indicates iterative predictions. Position errors are listed in meters. Missed or added vehicles average values are given in percent. Velocity errors are listed in $\mathrm{m/s}$. The prediction horizon is $1.5\,\mathrm{s}$ for a single-step and three times $0.5\,\mathrm{s}$ for iterative prediction. Velocity error calculations are as follows: Average Middle Velocity (AMV), Average Highest Velocity (AHV) (Table adapted from [2], ©2019 IEEE)

	Dist	**Miss**	**Add**	**AMV**	**AHV**
S IoU	1.48	0.52	0.03	0.46	0.95
S MSE	1.69	0.43	0.07	3.83	4.41
S IoU(pos), MSE(vel)	1.64	0.44	0.08	3.22	2.28
I IoU	1.37	**0.36**	**0.01**	1.20	0.98
I IoU(pos), MSE(vel)	1.47	0.38	0.04	2.89	1.66
I IoU unsupervised	1.40	0.37	0.02	1.01	1.32
I IoU deep residual	**1.29**	0.39	**0.01**	**0.87**	**0.97**

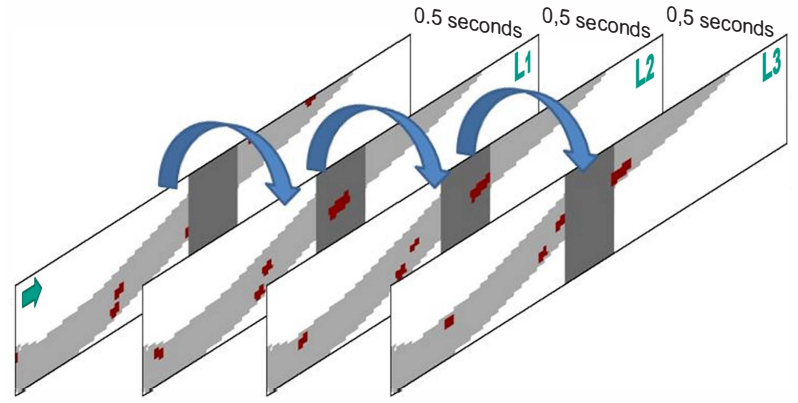


Figure 4.34: Iterative prediction for a traffic scene with occluded area. The big vehicle in the center is correctly predicted through the occluded area. The loss is calculated after each step and combined into a total loss. A prediction over $0.5\,\mathrm{s}$ is applied for three steps. (Graphic from [2], ©2019 IEEE)

a single batch, the runtime increases almost linearly with the number of prediction steps. A compromise between accuracy and runtime has to be found.

The awareness of occlusions brings a further major advantage. It enables the unsupervised training of the network so that no ground truth data needs to be available. As described above, occluded areas are excluded from the loss calculation anyway. This is crucial when using real-world data because there always are occlusions.

Deep residual connections

The network was adapted similarly as above with 20 convolutional layers with max-pooling after every four layers in the encoder, ten transposed convolutional layers for upsampling, and skip connections [71] for better gradient transport. Again, the deep network outperforms the other networks in the accuracy of position and velocity prediction.

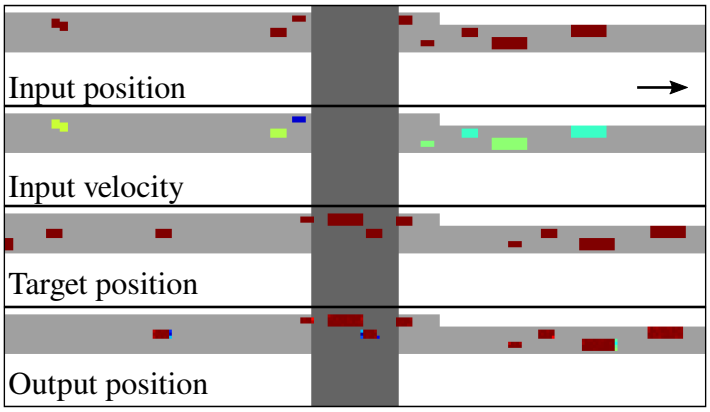


Figure 4.35: The network predicts a vehicle in the occluded area, that it has not seen before. The input for the positions and velocity of the vehicles are depicted as well as the target and output positions. The velocity input shows that the vehicle in front of the occluded area is almost at a stop, and the vehicle behind the area has already stopped. The network assumes a vehicle in between these two vehicles.

Infering presence of occluded traffic participants

Fig. 4.35 shows a traffic scene with an occluded area in the center. The network correctly assumes a vehicle inside this area because the vehicle about to enter the occluded area stops. From the context knowledge given by the road, the vehicles in front and after the occluded area, the network deduced the hidden vehicle in between. However, it must be noted that it is not possible to make these predictions with absolute confidence. The network predicted multiple small vehicles in the gap if the input was slightly modified. There are too many degrees of freedom, from the shape of vehicles to the personally preferred safety distance. This becomes clear when working with real-world data, where the position and dimensions of the vehicle can vary arbitrarily.

Therefore, the deduction of hidden vehicles from the given context was further researched in [28]. Based on real-world data from the INTERACTION dataset and synthetic data using simulation, convolutional long-short-term memories (convLSTMs) were used to infer information about occluded traffic participants. A sequence of ten frames was fed into the convLSTMs to internally build a hidden state in the recurrent networks and then predict the presence of objects in occluded areas, including their state. The convLSTMs were chosen to include more frames to gather deeper insights into the traffic scene over a longer time. The input consisted of environment tensors in a similar form as shown above, consisting of occupancy and occlusion information, velocities in x- and y-direction, the road layout, and other static information like buildings. The output was also a grid-based environment representation. Again, it was shown that, on the one hand, neural networks can make assumptions about occluded traffic participants. Nevertheless, on the other hand, the assumptions are not always accurate, especially with higher scene complexity, as depicted in Fig. 4.36. Compared to the set-based approaches for representing occluded traffic participants presented in Sec. 4.2, the learned approaches targeted the prediction of specific actors. Future research should focus on learning different representations to group assumptions about occluded traffic participants like the representations used for NST (see Sec. 4.2.3) or OST (see Sec. 4.2.4).

4.4.6 Discussion

The ablation study executed for the single-step prediction delivered important information for the iterative prediction and showed how uncertainties in the prediction could be included by mimicking BNNs

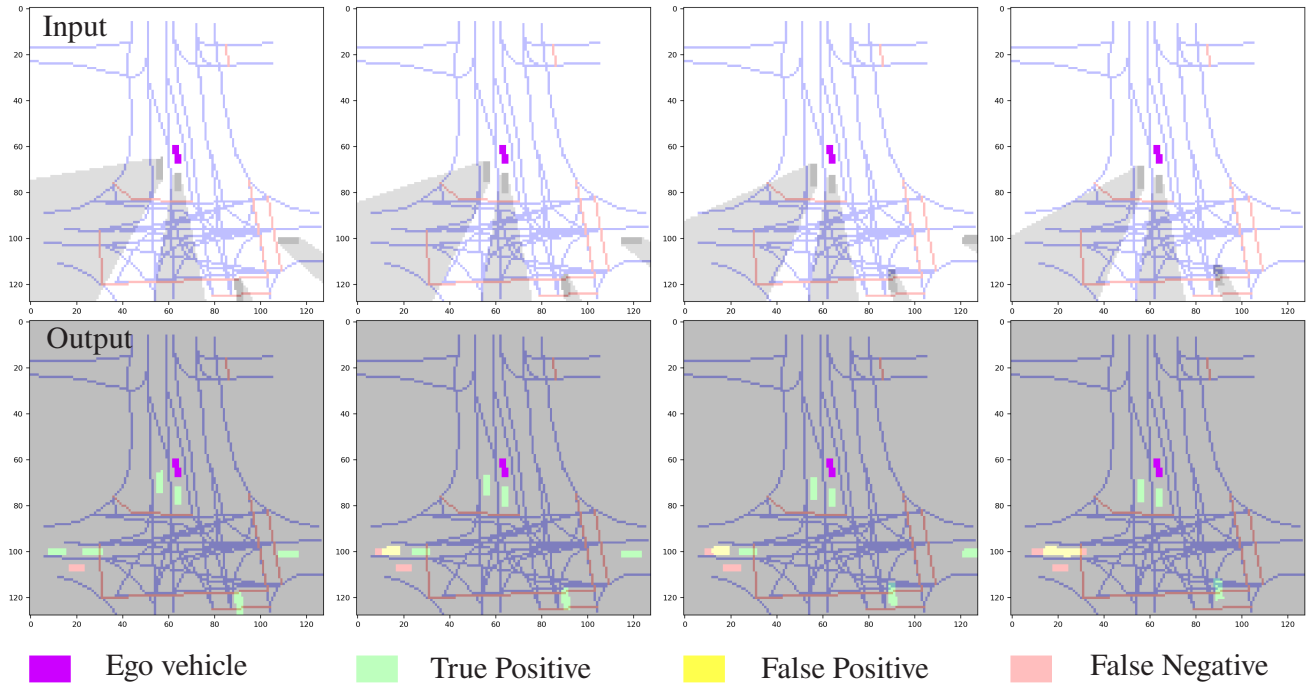


Figure 4.36: The excerpt from a figure from [28] shows an example sequence for the inference of hidden road users. The time sequence is from left to right. The first line shows the input and the second line the output of the network compared to the ground truth. (Graphic from [28])

through multiple forward passes with dropout. The concept of iterative predictions provided a method for the training and inference of the networks to predict occluded traffic scenes. The ability to train neural networks with real-world data while explicitly including occlusions is crucial for informed scene predictions.

However, the model-based prediction from Sec. 4.1.3 was used in the implementations of the concepts in Sec. 4.1, Sec. 4.3, or Sec. 5.1 instead of the learned prediction. First, the learned prediction focused on highway scenarios, while the other scenarios laid their emphasis on intersections. So, the learned and the model-based prediction modules must be seen as complementary and not mutually exclusive. Second, the networks yield single grid-based outputs that accumulate all multi-modal predictions. Single developments as defined in Sec. 2.1.2 cannot be derived from the output without further processing, i.e., extracting a prediction on entity level. This affects the single-step and the iterative prediction. A potential solution to overcome the problem might be combining the graph-based prediction as in [9] with a grid for information about occlusions, similar to the additional map information in [68]. Third, the neural networks need to be integrated efficiently in the processing pipeline, including inference and data transfer between CPU and GPU. This was not feasible for the foresighted maneuver planner based on POMDPs that will be presented in Sec. 5.1. However, the findings of this chapter contributed to the development of the concept in Sec. 5.3.

4.4.7 Conclusion

The key points of this section are:

- FCN generate promising results towards a model-free prediction framework.

- Using the Markov Property, the networks can be used to predict traffic scenes to arbitrary prediction horizons. These factors make the proposed network combinable with algorithms described in Sec. 4.1 or Sec. 5.1.
- Iterative training and prediction allows to predict occluded traffic scenes and to train the network in an unsupervised manner. This is vital when dealing with occlusions because occlusions imply incomplete data and the lack of ground truth information.
- The iterative training process increases the prediction quality.
- Using Monte Carlo dropout to approximate BNNs allows obtaining epistemic prediction uncertainty. The uncertainty can be used in successive components, i.e., to obtain safe maneuvers.

4.5 Chapter Conclusion

This chapter dealt with the second research question. A belief about the state of the environment was built and updated over time with a focus on the intentions of traffic participants and hidden traffic participants. Novel approaches to infer intentions and the existence of hidden road users from the accumulated information at the current point in time were presented. Two approaches for different targeted motion patterns were introduced to improve the inference of hidden road users and to track occluded areas in general. Two prediction modules, a model-based one and a machine learning-based one, were presented to predict traffic participants. Both are able to cover uncertainties and to predict to arbitrary prediction horizons through incremental prediction steps. The uncertainty in the state of the environment is already included in the road users' state that is assessed with uncertainty.

5 Anticipatory Maneuver and Trajectory Planning

The final step after gaining a complete situational awareness (Chapter 3) and building a belief about the current state of the environment (Chapter 4) is to make informed and foresighted behavior decisions and to plan risk-aware and safe trajectories.

Decision-making is a hierarchical process. First, the long-term and high-level maneuvers are determined that mirror the desired behavior. Afterward, a trajectory for the imminent seconds is planned according to the boundary conditions set by afore-determined maneuvers. This work uses a POMDP to probabilistically obtain the optimal maneuver decisions what is shown in Sec. 5.1 for occluded intersections. The trajectory planning process is made risk-aware by the concept of risk maps over time that combine various sources of risk to react to sudden and imminent dangers. The approach is presented in Sec. 5.2. Here, a particle swarm optimizer is used as a sampling-based nonlinear optimizer to solve the constraint optimization problem of trajectory planning. The convergence and optimality of the solution strongly depend on the initialization of the particles that represent single trajectory candidates. Therefore, in Sec. 5.3, an approach is presented to generate initial trajectories using neural networks. The approach shows how the advantages of both machine learning and model-based algorithms can be combined to improve the initialization process and how to deploy machine learning safely.

Fig. 5.1 shows the allocation of the planning modules in the overall automated driving stack. As can be seen, these components strongly depend on their input. In this work, the behavior and trajectory planning modules can rely on an accurate belief of the state of the environment due to a comprehensive situational awareness. The belief not only contains perceivable and measurable information. It also includes information where no data is available or where high-level information was deduced from context knowledge.

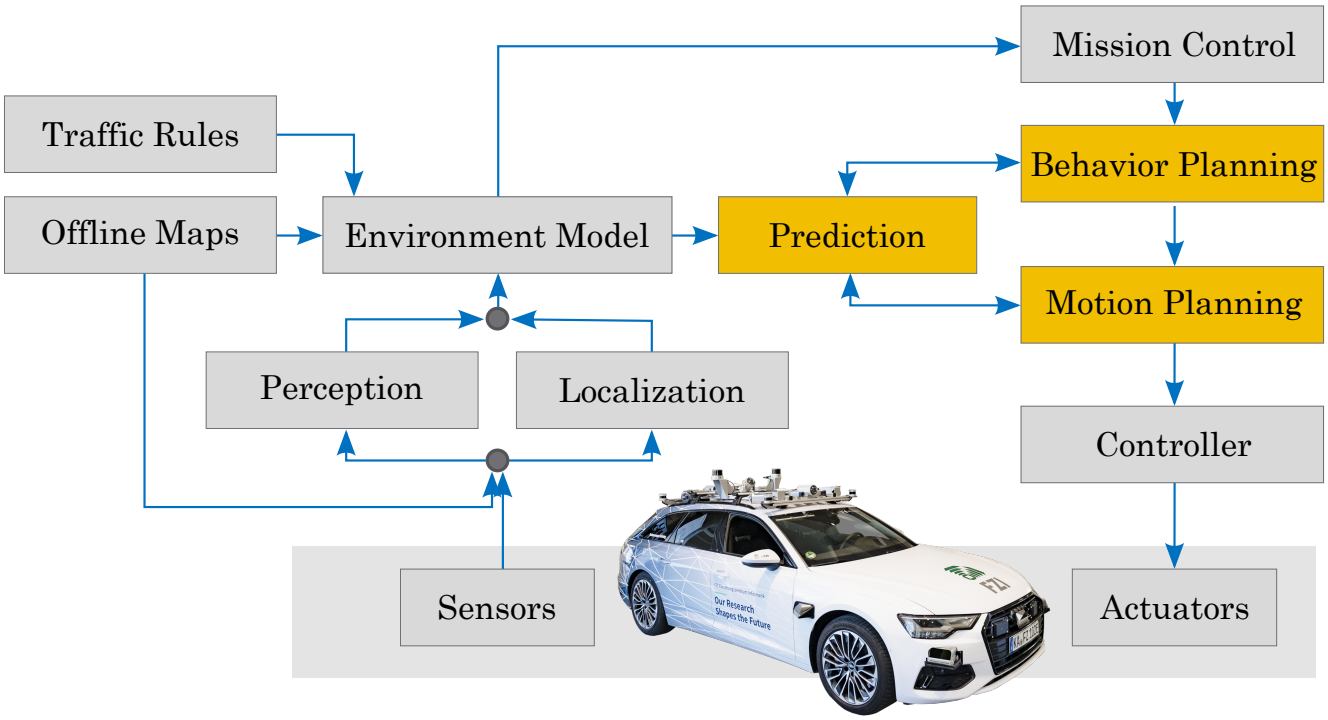


Figure 5.1: Allocation of the planning components in the overall architecture

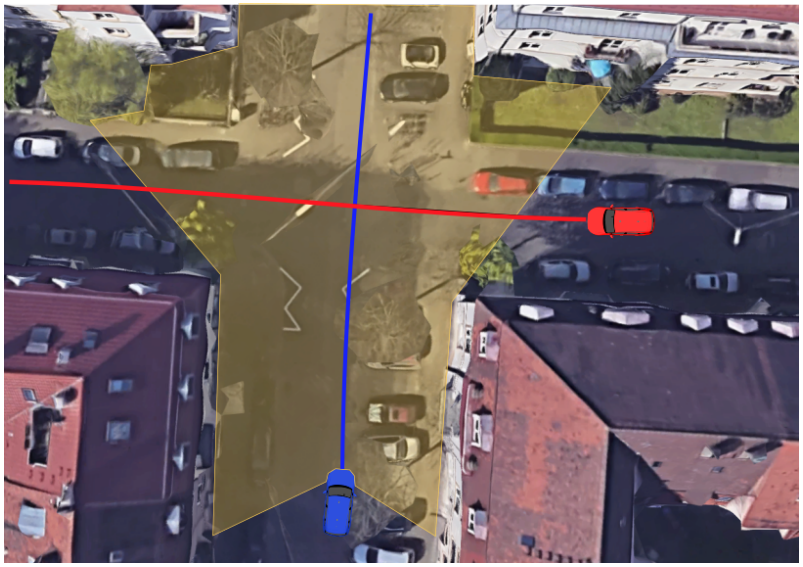


Figure 5.2: The ego vehicle (blue) approaches an unsignalized intersection where a red vehicle arrives from the right. It can not be seen by the ego vehicle because a building limits the ego vehicle's view (yellow). [image @ Google Maps, 2019, location: Karlsruhe, Germany, accessed: 24 January 2019] (Graphic from [1], ©2019 IEEE)

5.1 Foresighted Maneuver Planning

The following section is based on [1] and may contain verbatim quotes.

At the core of foresighted decision-making stands the question of how the situation will evolve. How will other traffic participants act? Is there an occluded traffic participant behind the next corner? And what will the ego vehicle be able to observe? These questions can not be answered with full certainty. Therefore, this work proposes to use a probabilistic maneuver planning module based on a POMDP and shows the application for tackling occluded intersections. POMDPs account for uncertain observations and developments after the agent chooses an action. They are described in Sec. 2.5. A generic description of the FOV is applied to obtain potential future observations, i.e., to determine if the ego vehicle will perceive a previously hidden traffic participant.

Statistics, e.g., [138], show that hidden traffic participants still cause many accidents. New driver assistance systems like emergency assistants constantly reduce the number of accidents, but they directly rely on sensor measurements. Automated vehicles, therefore, need more elaborate approaches like the one presented here.

There are alternative solutions to the questions above. However, these often rely on making assumptions that make the automated vehicle almost anxious, overcautious, and not comprehensible for human traffic participants. This can manifest itself, for example, in the vehicle keeping massive safety distances or even by coming to a standstill before a junction because it cannot enter the intersection due to the safety requirements implied by the assumptions made previously. The presented approach automatically balances comfort, efficiency, and cautiousness through probabilistic modeling. In situations of higher uncertainty, the automated vehicle reduces speed and keeps larger safety distances. Whereas in situations that the vehicle can assess with higher confidence, efficiency will be prioritized. This leads to an overall more comprehensible and safe behavior. It avoids dangerous situations in general and collisions in particular.

An example situation is shown in Fig. 5.2. The blue ego vehicle drives in an urban area with lots of occlusions due to parking vehicles, buildings, or vegetation. The red vehicle on the incoming lane from the right cannot be observed. Fig. 5.3 depicts the situation in a speed-over-distance diagram. The blue curve corresponds to a behavior where the deceleration process of the ego vehicle starts very early. The red curve instead represents a behavior where the braking process starts when the red vehicle is seen. Here, the ego vehicle is not able to brake on time. The green curve presents an example output of the maneuver planner. The speed curve for the latest possible moment to start the braking maneuver is shown in dashed green. The green curve runs between the blue and the dashed green curve, as the black arrows indicate. Safety is guaranteed when the ego vehicle follows a speed curve indicated by the green line. The course of the green curve depends on the existence of an occluded vehicle on the prioritized and the chosen emphasis on comfort. If no vehicle is perceived on the prioritized road when the road becomes observable, the ego vehicle may accelerate and traverse the intersection. This is shown by the dotted green and blue curves. The yellow area between the green and blue curves is proportional to the time and energy lost between the foresighted (green curve) and the defensive behavior (blue curve). The probabilistic planning approach promises to balance economy and comfort optimally while always maintaining safety.

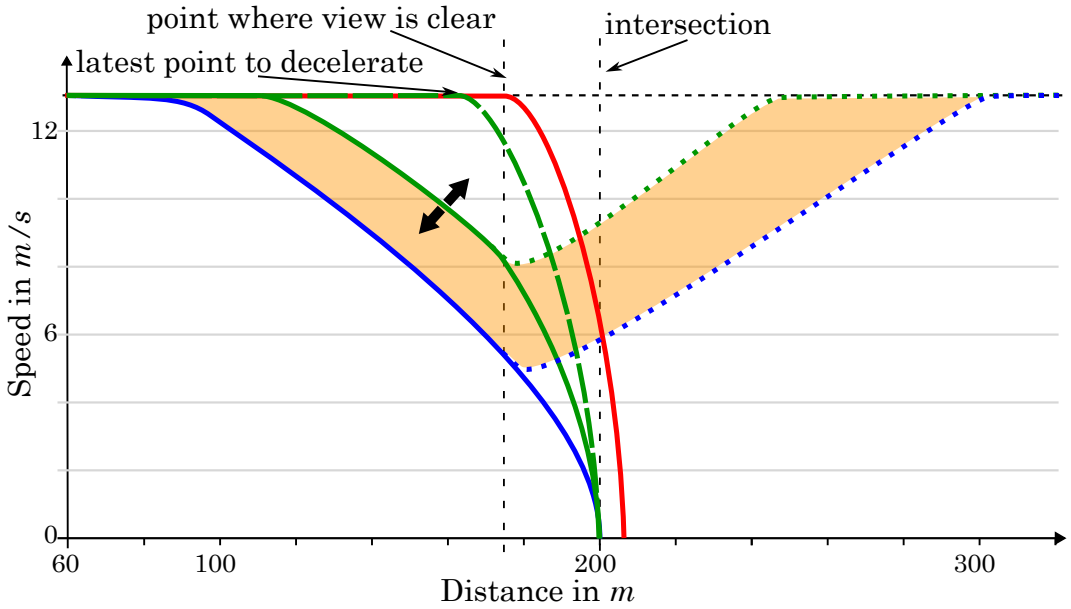


Figure 5.3: Speed over distance curves for defensive braking (blue), foresighted braking (green), and too late braking (red). The dotted lines present the speed curves when no occluded vehicle was present. The dashed green line is the latest possible moment to brake.

5.1.1 Related Work

Foresighted maneuvers have to consider uncertain developments of the traffic scene. When dealing with occlusions, this includes potential future observations of hidden traffic participants, i.e., the FOV. To circumvent this difficult question, approaches try to estimate the risk using metrics to measure the criticality, like the time to collision. An approach representing potentially hidden traffic participants using particles was proposed in [142]. The authors determine the time required for the ego vehicle and the potential other vehicle to reach the crossing point. This is used to calculate the maximum speed of the ego vehicle. The particles are predicted along the road network and are updated or discarded based on Lidar measurements. The safe passage across occluded intersections is also tackled in [76]. Visible and non-visible vehicles are regarded separately. Visible vehicles are predicted on the object level, while non-visible vehicles are used in an occupancy grid. The longitudinal planning for the vehicle is achieved by determining free path segments at the corresponding point in time. [143] uses an optimization-based approach for tackling occlusions at intersections. A downside of not considering future observations is situations that may lead to a standstill.

As mentioned above, POMDPs are designed to model uncertain scene development with respect to the actions taken by the agent and the resulting observations. Brechtel et al. [48] leverage this property and show the advantages in a lane merging scenario with limited sight for the longitudinal planning of a vehicle. The FOV is simplified only to check the direct line of sight between two road users to determine if they are occluded. The number of traffic participants must be known a priori as the policy is determined offline, and the representation of the scene is learned beforehand.

Solving POMDPs is computationally expensive. As explained in Sec. 2.5, it is distinguished between online and offline solving. Offline solvers are able to determine an optimal policy for an arbitrary initial belief, whereas online solvers try to seek the optimal immediate action for the current belief. Latest examples of state-of-the-art solvers include Randomized Belief-Space Replanning (*RBSR*, 2010) [73] or

Monte Carlo Simulation-based approaches like Partially Observable Monte-Carlo Planning (*POMCP*, 2010) [136], Monte Carlo Value Iteration (*MCVI*, 2011) [39], the Determinized Sparse Partially Observable Tree (*DESPOT*, 2013) [154], and the Adaptive Belief Tree (*ABT*, 2013) [97] algorithm. In [78], the *ABT* algorithm determines the correct behavior at intersections. Unknown intentions of other road users are considered, whereas non-visible vehicles are not taken into account. The POMDP yielded optimal reactions to the situation and considered the route intentions correctly. In [108], a similar scenario is investigated by Liu et al. using *DESPOT*, where the algorithm is used to determine the optimal longitudinal control of the ego vehicle using a small set of discrete actions. [47] proposes a scalable approach to solve the POMDP for a discretized environment and shows it in a scenario with occluded traffic participants.

Here, the Toolkit for Approximating and Adapting POMDP Solutions in Real-Time (TAPIR) [97], which is based on the *ABT* algorithm, is utilized for solving the POMDP. A belief is represented by a set of particles that form a probability distribution over the state of the environment. Correspondingly, beliefs are propagated and updated using particle filter techniques. The belief tree is constructed of sampled episodes that comprise a sequence of quadruples of encountered states, actions, observations, and rewards starting from an initial belief. A brief description can be found in Sec. 2.5.4 and details in [97] and [102].

5.1.2 Concept

The problem of determining the optimal behavior for the ego vehicle at occluded intersections is split into two parts. Leveraging the so-called "path-velocity decomposition" [90], first a valid path for the vehicle is planned. Afterward, the motion of the vehicle along the path is determined. The path has to be free of collisions with static obstacles and must meet kinematic constraints and outer constraints like road markings or lane boundaries. The path is generated according to the ego vehicle's route. In case there are no obstructions on the lane, the center of the route is used as a reference path. The route of the ego vehicle not only provides an orientation for the reference path. It is also used to derive which intersections the vehicle will pass, where it must give way, and where potentially dangerous situations can occur. The route also indirectly provides the speed limits. For the longitudinal behavior, a POMDP is utilized. The decision-making problem is reduced to a one-dimensional problem through the decomposition, decreasing the complexity.

The longitudinal behavior is expressed through accelerating or decelerating actions. It targets the behavior of being aware of dangers and safely navigating situations caused by hidden traffic participants. The separation between behavioral planning and trajectory planning becomes blurred in this context, as stopping at an intersection can be described both on behavioral and motion planning levels.

To account for potential future observations, a novel generic sensor setup is applied to generate the assumed FOV for the corresponding predicted state of the environment. The current state of the environment is obtained from the sequence of previous actions and observations. The predictions of the state of the environment also account for motion uncertainties. Here, the observations focus on traffic participants in hidden areas. Using the FOV, the position of those traffic participants can be determined. With a certain likelihood, the presence of a traffic participant is assumed.

The approach is independent of the number of traffic participants so that an arbitrary number of visible and occluded traffic participants can be considered. The concept can easily be extended to other kinds of incomplete knowledge, like uncertain routes of traffic participants, by predicting different modes for each agent in the prediction step and making the corresponding observations afterward.

State Space

The state propagation follows the Markov property. Hence, a state contains all the required information to predict future states. The concept is based on the description of the state of the environment given in Sec. 2.1.2. The intentions of the traffic participants are limited to routes r so that the state for a single agent results in

$$s = (\vec{\xi}, \dot{\vec{\xi}}, r). \tag{5.1}$$

A route consists of a sequence of Lanelets [45]. Except for discrete routes, the state space $\mathbb{S}$ is continuous. As described in Sec. 2.5.1, a belief is built from particles representing samples of states of the environment in the belief space. These beliefs can contain visible as well as occluded or previously occluded traffic participants. The ratio of particles with and without occluded traffic participants in the corresponding states of the environment reflects the likelihood of the presence of an occluded traffic participant.

Action Space

The action space $\mathbb{A}$ is given by a finite set of actions representing the desired stopping and accelerating behaviors. Solely the action of the ego vehicle is determined in the planning process and is then executed by the ego vehicle. In a subsequent maneuver and trajectory calculation, the acceleration and deceleration values in the maneuver have to be limited such that the trajectory planner is always able to find feasible solutions (see also Sec. 3.1). The lateral motion of the ego vehicle is fixed due to the path-velocity decomposition. So, the set of actions solely consists of several acceleration and deceleration values. The set has to contain at least one action to represent acceleration, another to represent deceleration, and one to maintain the velocity. The set of actions should be kept as small as needed to represent the desired behaviors because the number of actions significantly increases complexity and computing time. So semantic actions like "stop" or "go" could also be chosen. To avoid confusion, this thesis uses different symbols for accelerations a and actions a.

The other traffic participants are predicted according to the prediction module described in Sec. 4.1.3, and the belief is updated through observations.

Reward Function

The reward function mimics the agent's desired behavior. The reward is received after executing action a in state s. To include safety, comfort and economic aspects, the reward function is designed as follows

$$\mathcal{R}(s, \mathrm{a}) = \mathcal{R}_{goal}(s) + \mathcal{R}_{col}(s) + \mathcal{R}_{v}(s) + \mathcal{R}_{a}(\mathrm{a}). \tag{5.2}$$

The special reward $\mathcal{R}_{goal}$ is obtained when reaching the goal. In this case, it is after safely crossing the intersection. The collision reward $\mathcal{R}_{col}$ can also be used for violations of traffic rules or when leaving the road or lane. Comfort and efficiency are described with $\mathcal{R}_{\mathrm{a}}$ and $\mathcal{R}_{v}$. High accelerations and deviations from the desired speed are penalized. Decelerations are weighted stronger than accelerations to motivate the agent only to decelerate when required.

$$\mathcal{R}_{a} = \begin{cases} -k_{a,1} \cdot k_{a,2} \cdot a^2, & \text{if } a \geq 0 \\ -k_{a,2} \cdot a^2, & \text{else ,} \end{cases} \tag{5.3}$$

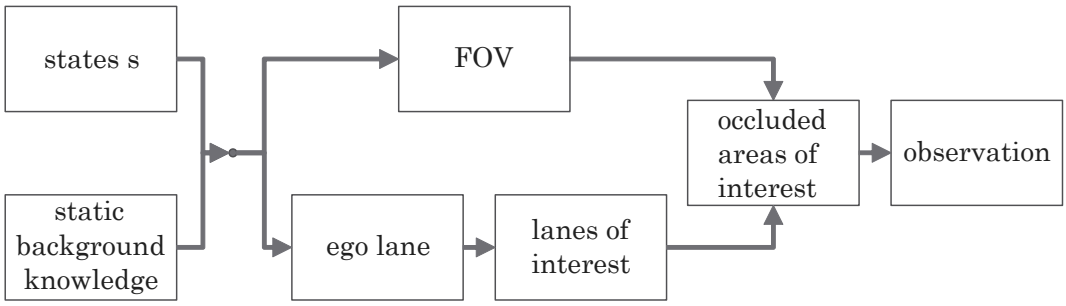


Figure 5.4: Concept of making an observation based on a state s and static background knowledge that contains maps and static objects. The FOV is generated and used to determine occluded areas on lanes of interest. These are used to make an observation. (Graphic based on [1], ©2019 IEEE)

To punish speed violations stronger than driving below the reference speed v_{ref}, the following cost term is used

$$\mathcal{R}_v = \begin{cases} -k_{v,1} \cdot (v_{ref} - v_0), & \text{if } v_0 \leq v_{ref} \\ -k_{v,2} \cdot (v_0 - v_{ref}), & \text{else} , \end{cases} \tag{5.4}$$

where v_0 is the speed of the ego vehicle and $0 < k_{v,1} < k_{v,2}$. As $k_{a,1}$, $k_{a,2}$, $k_{v,1}$ and $k_{v,2}$ are chosen to be positive, the rewards $\mathcal{R}_a$ and $\mathcal{R}_v$ are always negative. Thus, the ego vehicle will avoid collisions and limit accelerations while keeping to the speed limit to obtain the highest possible reward.

Observations

The observations space $\mathbb{O}$ is similar to the state space $\mathbb{S}$. An observation o includes measurements of the position and velocity of each vehicle i. The route is assumed to be known. However, as described above, the concept can easily be extended to deal with unknown routes. The observations result in

$$o = [o_0, o_1, ..., o_{n-1}]^\top \tag{5.5}$$

with

$$o_i = [\vec{\xi}_i, \dot{\vec{\xi}}_i, r_i, \beta_i]^\top, \; i \in \{0, ..., n-1\} \tag{5.6}$$

for n traffic participants. To distinguish if an object was measured or assumed, occluded traffic participants are assigned $\beta_i = occluded$ while visible objects are assigned $\beta_i = visible$.

In total, the set of possible observations results to

$$o_i \in \mathbb{S}_\beta, \tag{5.7}$$

where the observation space $\mathbb{O}$ consists of the state space from above with the state of each traffic participant extended by β resulting in $\mathbb{S}_\beta$.

The process of how an observation is retrieved after executing an action is depicted in Fig. 5.4. The state of the ego vehicle is perfectly observable. The state of other traffic participants is only partially observable. Thus, the current FOV needs to be taken into account. It is determined through ray tracing as described in Sec. 2.8 and Sec. 2.8.1.

The generic sensor setup used here consists of five Lidar sensors as depicted in Fig. 2.8. The sensor setup is based on a reduced configuration of our test vehicle, and the sensors are all the same. Using this generic sensor setup, the FOV is determined with ray tracing to obtain a set of polygons that describe the traced free space. In this step, all visible objects and static information like buildings are included. The generic model of the sensor setup could also be used to represent other types of occlusions, like cameras at night where only certain areas are illuminated. The resulting representation of the environment is a two-dimensional world, where obstacles are represented as polygons, visible areas are those areas that build the FOV, and all other areas are considered occluded.

The dimensions of hidden vehicles are assumed to be equal to the ego vehicle's. It is worth mentioning that the assumed vehicles themselves can also cause occlusions when they become visible in subsequent belief states.

To determine relevant areas for the ego vehicle, first, the current ego lane is determined. The lanes of interest are retrieved from the road network topology using map information. Lanes of interest can be manifold. They include prioritized and oncoming lanes or, when regarding non-compliant behavior of other traffic participants, all oncoming, merging, or crossing lanes. Additionally, hidden pedestrian crossings can be included. In the scope of the presented work, the lanes of interest are limited to prioritized lanes.

Combining the information about the FOV and the lanes of interest, occluded areas on lanes of interest are obtained. For the subsequent observation, road users in the FOV are measured as described above with their position and velocity with additive Gaussian distributed noise to account for sensor inaccuracies. If any part of the road user is in the FOV, it is considered visible ($\beta = visible$). Measurements of different vehicles are modeled to be independent. Hidden road users are assumed in occluded areas on the lanes of interest that are big enough ($\beta = occluded$) with a probability < 1. So, there is also the chance that no vehicle is located in the occluded area. The probability depends on various aspects, like the traffic density or the distance to other vehicles. The traffic density depends on the type of road, area, and time of day. A probability of 1 equals a worst-case assumption. In this work, a fixed value is assumed. The speed of a hidden vehicle can not be measured. Therefore, it is assumed that the vehicle will arrive at the intersection at the same time as the ego vehicle as long as the vehicle's speed is still plausible and does not exceed the speed limit.

For the example in Fig. 5.2, the lanes of interest are the crossing lanes from the right because they are prioritized. The area of the lane where the red vehicle is located is not included in the FOV. So, the position of the red vehicle in the image might be a possible observation of an occluded vehicle.

Transition Function

The transition function $\mathcal{T}$ models the dynamics of the system and the effect of the chosen actions. The state s_{t+1} at time step $t+1$ is predicted based on the chosen action a and the state s_t at time step t with the step size Δt. The step size Δt should be taken with care. The size of the belief tree grows exponentially with the number of planning steps. Smaller step sizes result in more planning steps for a given planning horizon, larger belief trees, and thus higher computational efforts. Larger step sizes may lead to inaccurate results. Intermediate collision checks were conducted to avoid tunneling effects, where road users pass each other in between one time step. All vehicles are predicted along their route. Only the motion of the ego vehicle is affected by an action a. The other traffic participants are predicted with constant velocity assumption. Additionally, zero mean Gaussian noise is added to their position and

velocity. For the longitudinal motion of the ego vehicle with progress d along route r, the transition can be described as

$$\begin{bmatrix} d \\ v \end{bmatrix}_{t+1} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} d \\ v \end{bmatrix}_t + \begin{bmatrix} \frac{1}{2}\Delta t^2 \\ \Delta t \end{bmatrix} a \ . \tag{5.8}$$

The state s is obtained by reconverting the progress d to a position with the geometry of route r.

5.1.3 Evaluation

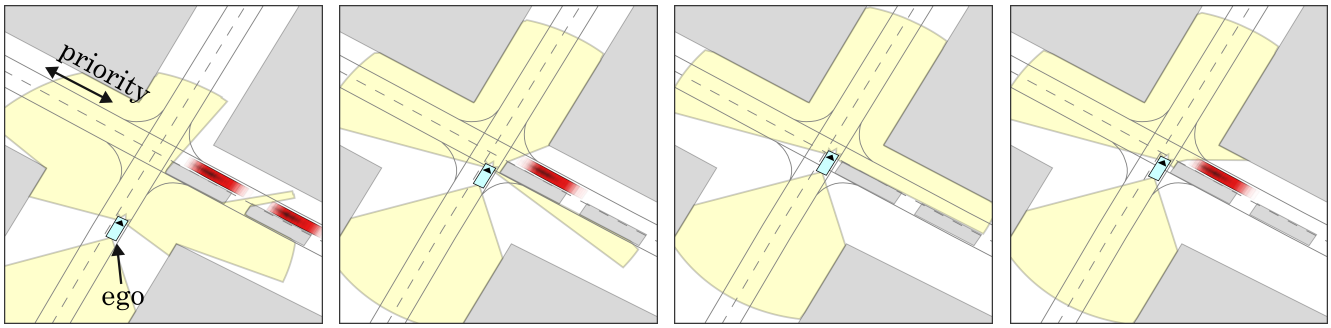
Due to the complexity and to be able to repeat scenarios multiple times under precisely the same conditions, the evaluation was performed in simulation. Additionally, using the simulation allows varying the ratio between simulated and actual elapsed time to perform closed-loop tests with converged policies.

In the following, three scenarios are presented. In two, the ego vehicle approaches an unsignalized intersection, once with heavy occlusion and once with occlusions that occur regularly in everyday traffic. The third scenario is a merging scenario with an occluded view of the lane to merge on, as it is often seen on highways. Buildings or other obstacles produce the occlusions. The ego vehicle does not have the right-of-way in all scenarios.

The current action for the agent is calculated online and executed afterward. The behavior of other traffic participants in the simulation is modeled by the IDM [95], and they try to keep to the center of their lane. They keep to the traffic rules and do not assume the incompliant behavior of other road users. The traffic participants' routes are randomly generated for each simulation run. Also, the traffic participants do not change their route halfway through the scenario. The parameters used in the evaluation are shown in Table 5.1.

Table 5.1: Parameters for the evaluation of the maneuver planner

Parameter	**Value**
Δt	$0.5\,\mathrm{s}$
γ	0.95
Horizon T_p	$5\,\mathrm{s}$
UCB Coefficient	700
min particles initial belief	350
sensor range	$60\,\mathrm{m}$
$\mathcal{R}_{goal}$	500
$\mathcal{R}_{col}$	-3000
$k_{v,1}$	20
$k_{v,2}$	200
$k_{a,1}$	5
$k_{a,2}$	8



(a) $t = 14\,\mathrm{s}$ **(b)** $t = 17.5\,\mathrm{s}$ **(c)** $t = 20\,\mathrm{s}$ **(d)** $t = 21.5\,\mathrm{s}$

Figure 5.5: The temporal development of a scenario with heavy occlusion through two parking vehicles. The ego vehicle is shown in blue, other road users in red, assumed vehicles in blurred red, static obstacles in grey, and the field of view in yellow. These colors are also used in Fig. 5.7. (a) $t = 14\,\mathrm{s}$: Approaching the intersection and assuming hidden vehicles. (b) $t = 17.5\,\mathrm{s}$: Decelerating and entering the intersection. (c) $t = 20\,\mathrm{s}$: Slowly feeling its way into the intersection, aware of potential approaching vehicles. (d) $t = 21.5\,\mathrm{s}$: Reaching the critical point where the ego vehicle must either make a full stop or accelerate. As the sight is clear, the ego vehicle crosses the intersection. (Graphics from [1], ©2019 IEEE)

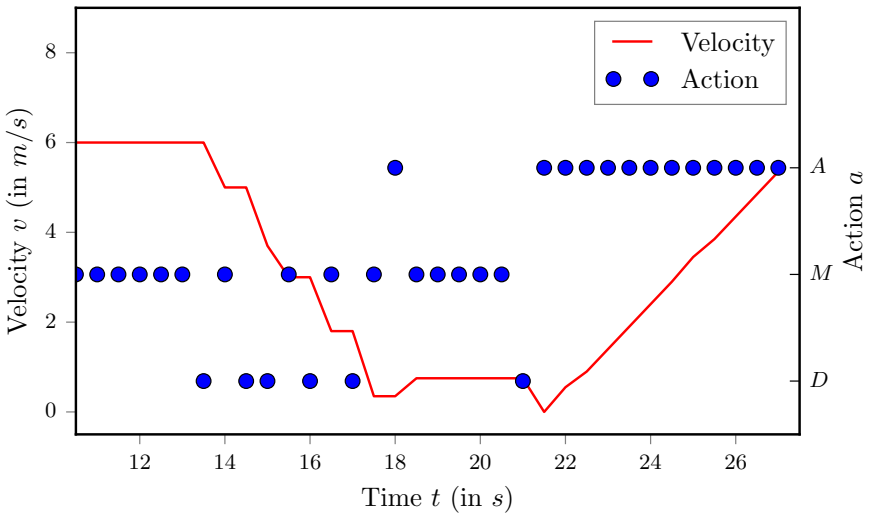
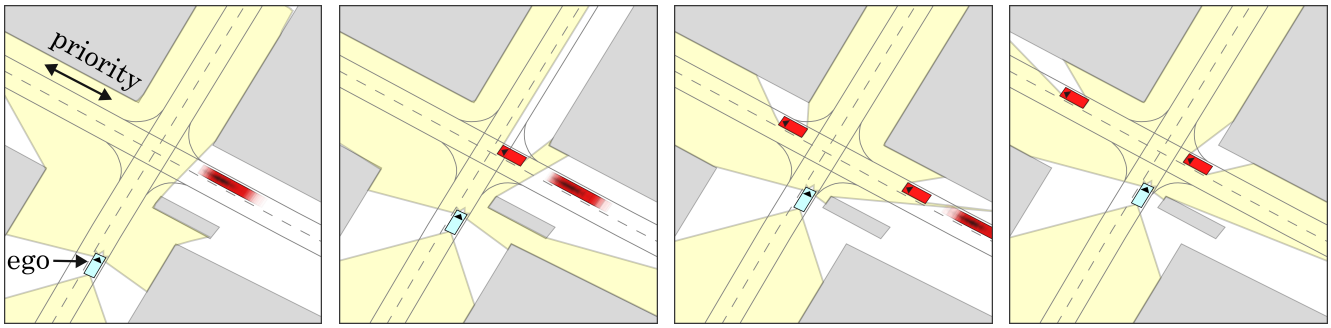


Figure 5.6: Velocity and action profile for a scenario with heavy occlusions due to two parked vehicles. Actions are drawn in blue and velocity in red. The agent decelerates first, then creeps slowly into the intersection and accelerates as it registers that the intersection is clear. (Graphic from [1], ©2019 IEEE)

Scenario: Heavy occlusion at intersection

Fig. 5.5 shows the scenario at different times. The ego vehicle approaches the intersection where two busses obstruct the view on the prioritized road. The agent can choose between a minimum set of actions for accelerating (A), decelerating (D), and maintaining the current speed (M), so that $\mathrm{a} \in \mathbb{A} = \{a_D = -2.0\,\mathrm{m/s^2}, a_M = 0.0\,\mathrm{m/s^2}, a_A = 1.0\,\mathrm{m/s^2}\}$. Fig. 5.6 shows the corresponding actions and speed profile. Initially, the ego vehicle keeps to the reference speed of $6.0\,\mathrm{m/s}$. It starts to decelerate because of the occluded prioritized road. At the intersection, the agent creeps forward little by little to safely enter the intersection until it perceives enough of the prioritized lane to pass it. In the depicted



(a) $t = 17\,\mathrm{s}$ **(b)** $t = 18.5\,\mathrm{s}$ **(c)** $t = 20\,\mathrm{s}$ **(d)** $t = 21.5\,\mathrm{s}$

Figure 5.7: The temporal development of a scenario with two hidden vehicles from the right. (a) $t = 17\,\mathrm{s}$: Approaching the intersection and assuming hidden vehicles. (b) $t = 18.5\,\mathrm{s}$: A hidden vehicle comes into view. Another vehicle is assumed in the hidden area behind it. (c) $t = 20\,\mathrm{s}$: The second vehicle is detected, and the ego vehicle has to stop. (d) $t = 21.5\,\mathrm{s}$: The ego vehicle waits for all road users to pass and has full sight of the intersection to cross it safely. (Graphics from [1], ©2019 IEEE)

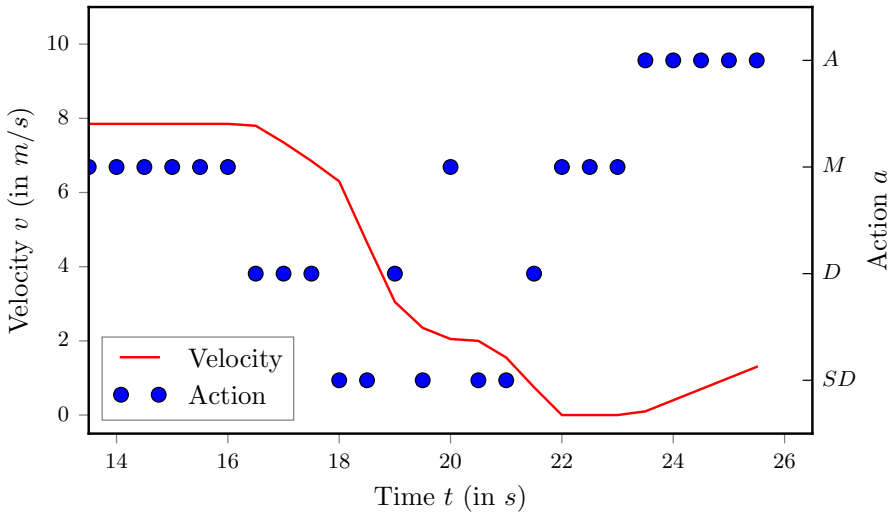


Figure 5.8: Velocity and action profile for the scenario with two hidden vehicles. Actions are shown with blue dots, and the velocity of the ego vehicle is shown with a red line. The ego vehicle solely decelerates as much as required to be able to yield an appearing vehicle at every time and decelerates strongly when it perceives another vehicle. (Graphic from [1], ©2019 IEEE)

scenario, no hidden vehicle appeared. However, due to the cautious behavior of the ego vehicle, it would have been able to stop on time.

When assuming the worst case, the agent would be stuck behind the parking car, blocking vehicles behind it and vehicles coming from the left side. This also accounts for approaches based on metrics like the time to collision. The minimum set of actions is already sufficient to overcome this unwanted behavior. The worst-case assumption was reproduced here by assuming an occluded traffic participant with a probability of 1.

Scenario: Occluded intersection

Fig. 5.7 shows the same intersection but with occlusions by a kiosk, shrubbery, or similar. The ego vehicle approaches the intersection with the desired speed of $8.0\,\mathrm{m/s}$. In this scenario, the agent can decide between a soft (D) and strong braking (SD) action as well as accelerating and maintaining the velocity as above $a \in \mathbb{A} = \{a_{SD} = -2.0\,\mathrm{m/s^2}, a_D = -1.0\,\mathrm{m/s^2}, a_M = 0.0\,\mathrm{m/s^2}, a_A = 1.0\,\mathrm{m/s^2}\}$. In anticipation that it might pass the intersection without stopping, it only decelerates slowly to a degree where it would still be able to come to a halt when it observes a vehicle on the prioritized lane. Unfortunately for the agent, a vehicle appears from the right, and the agent has to decelerate strongly. At this point, the ego vehicle does not have to stop entirely because it might pass behind the other vehicle. Therefore, it decelerates only as much as required. When the second vehicle approaches, the ego vehicle has to stop at the intersection to wait for the other road users to pass before crossing the intersection safely.

It can be observed that the ego vehicle is aware of the chance to avoid unnecessary decelerations. If no other vehicle appeared, the ego vehicle could accelerate again from a much higher speed, saving energy and providing more comfort. The behavior resembles human driving behavior at intersections with limited sight and shows the foresighted behavior of the automated vehicle.

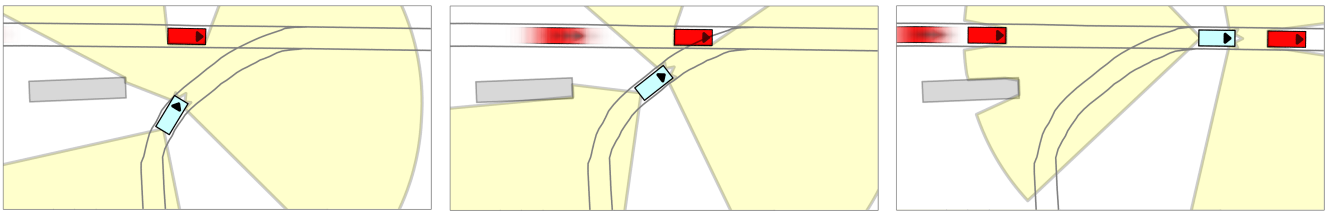
Scenario: Merging

The experiments of the merging scenario were conducted in [31]. Fig. 5.9 shows the temporal development of the scenario. The ego vehicle has a blind spot to its rear left and right. The minimal action set with $\mathrm{a} \in \mathbb{A} = \{a_D = -2.0\,\mathrm{m/s^2}, a_M = 0.0\,\mathrm{m/s^2}, a_A = 1.0\,\mathrm{m/s^2}\}$ is used. However, in the planning process, when the belief tree is built and the policy is determined, the ego vehicle can cope with the situation by having an internal belief that allows one to argue if merging is safe or not. The scenario shows that the agent sees the opportunity to merge right behind the first visible object.

Discussion

The presented approach showed that it is capable of making foresighted maneuver decisions in situations with occlusions. Additionally, as the intentions are already included in the state representation and the prediction algorithms, the approach can easily be extended to include unknown route intentions, for example.

The computation complexity is still a downside of the approach. On the one hand, the POMDP-solving process will converge at some point. On the other hand, convergence is not guaranteed to be reached within a defined time. To overcome this problem, an informed safety layer was developed. Based on the tracked occupied areas from Sec. 4.2, currently, the maximum speed for approaching the intersection safely is calculated. For prioritized lanes, the time for a potentially occluded obstacle, i.e., an element q of the occluded areas Q, to reach the intersection is calculated using the position and velocity information of q and their distance to the intersection. Thus, the informed safety layer uses the gathered context knowledge to provide an upper bound on the vehicle speed. In combination, the POMDP and the informed safety layer offer a safe and foresighted strategy to approach occluded intersections. Taking a look at Fig. 5.3, the informed safety layer determines the latest possible time to brake what is depicted in a dashed green line. The foresighted POMDP maneuver planner probabilistically determines the course of the solid green line based on the available information about the situation.



(a) $t = 1\,\mathrm{s}$ **(b)** $t = 2\,\mathrm{s}$ **(c)** $t = 4.5\,\mathrm{s}$

Figure 5.9: The temporal development of the merging scenario. (a) $t = 1\,\mathrm{s}$: Approaching the merging spot with $6\,\mathrm{m/s}$ with one visible vehicle on the target lane. (b) $t = 2\,\mathrm{s}$: Accelerating to merge behind the visible vehicle. (c) $t = 4.5\,\mathrm{s}$: Merging is complete. A second vehicle comes from behind at a safe distance. (Graphics from [31])

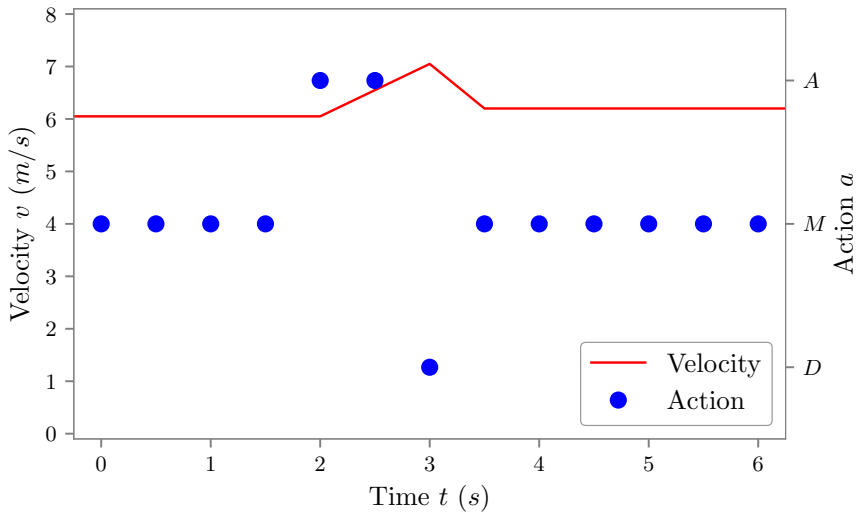


Figure 5.10: Velocity and action profile for merging scenario. Actions are shown with blue dots, and the velocity of the ego vehicle is shown with a red line. (Graphic from [31])

5.1.4 Conclusion

The main takeaway points of this chapter are

- The generic representation of the sensor setup allows making assumptions about future observations that are used in return in a POMDP maneuver planer for determining the optimal action for the ego vehicle.
- The POMDP explicitly propagates the initial belief into the future, representing possible future developments and their likelihood.
- The presented POMDP approach uses continuous state and observation spaces with a discrete action space to determine safe and optimal maneuvers in occluded intersections and overcomes problems of worst-case assumptions.

5.2 Risk-aware Trajectory Planning

The trajectory planner seeks to determine a safe and feasible trajectory with regard to the boundary conditions set by the maneuver planner. Maneuver planners often utilize a simplified motion model and an abstract representation of the environment to cope with the complexity. Furthermore, the maneuver decision is updated less frequently than the trajectory because it is planned for a longer time horizon. Therefore, the boundary conditions of the trajectory planner can not be described solely by the commands and conditions set by the maneuver planner. The trajectory planner itself needs to be aware of its surroundings, primarily of risks, e.g., uncertain predictions of others, particularly if the predictions are multi-modal. As discussed earlier, risks are manifold and often challenging to quantify. A graphic example is the risk emerging from occluded objects. As the environment along the road can be arbitrarily shaped, the number of locations where occluded objects can cross the street is countless, and the associated risk can not be expressed in a single metric.

It becomes clear that the temporal and spatial combination of various sources of risk poses many challenges. These challenges are tackled with the concept of risk maps over time (RMOT) proposed in this section. A single risk map spatially associates the risk of different sources in a two-dimensional grid for one point in time. A sequence of risk maps then encodes the risk over time.

This chapter is based on [7] and [8] and may contain verbatim quotes.

5.2.1 Related Work

Overviews about predicting traffic participants and risk assessment can be found in [105] or [54]. Both are active and fast-developing research areas, as ensuring safety is critical. The focus in the following is set on risk assessment. For the related work on prediction or tracking occlusions, the reader is referred to 4.1.1, 4.2.1, and 4.3.1.

Safety models like the Responsibility-Sensitive Safety (RSS) [135] approach or the Safety Force Field (SFF) [118] seek to guarantee absolute safety. These approaches are easy to understand and apply. However, they are limited to the current state of the environment. The result is often more conservative than approaches that continuously track the environment.

Risks from occlusions are tackled, e.g., in [155], [116], or [55]. Yu et al. determine the risk from occlusions at intersections using a particle-based approach [155]. Nager et al. use a set-based approach to include risk estimation for potential occluded traffic participants into a planning algorithm and evaluate it for a scenario with a pedestrian and distinguish between legal and illegal positions of occluded objects in [116]. Damerow et al. generate a two-dimensional map in [55], the so-called predictive risk maps that contain risk over the longitudinal progress and velocity. Based on these, the velocity planning along a given path determines the minimal risk trajectory in intersection scenarios with limited visibility.

In literature, it is often distinguished who is responsible for potential collisions or violations of traffic rules. The safety models RSS or SFF only guarantee safety if all participants follow the formulated principles. [147] proposes a not-at-fault approach for driving in traffic, and as mentioned, [116] distinguishes between legal and illegal areas where objects are allowed to stay.

[119] and [106] perform risk assessment for other visible traffic participants. In [119], a risk-aware RSS is proposed as an extension to RSS. By incorporating the risk of a driving situation, the aforementioned

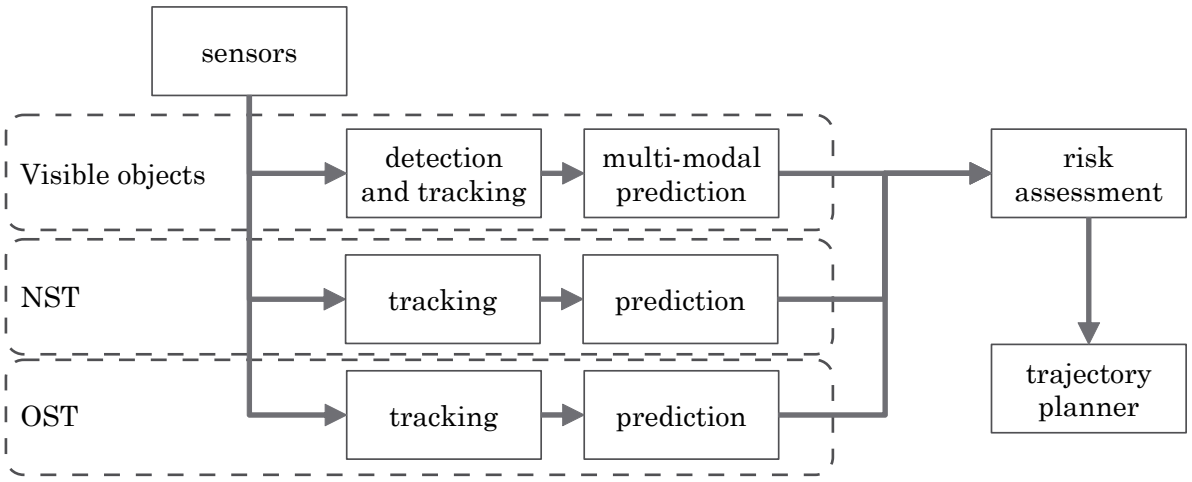


Figure 5.11: Concept overview with three sources of risk.

problem of too-conservative results could be improved, and a significant reduction in safety margins could be achieved for longitudinal driving maneuvers. Risk at intersections, in particular, is investigated in [106] by comparing the driver's intention to the expected behavior.

As shown in the previous section, approaches that adaptively account for future interactions or developments, like Partially Observable Markov Decision Processes, can also achieve foresighted behavior. These approaches can become computationally very expensive and are therefore more suited for maneuver planning than high-frequent trajectories.

In contrast to the listed literature, this section presents an approach that uses the tracking and forecasting of occluded areas both on and off the road from Sec. 4.2 and uses this information in combination with uncertainty-aware multi-modal motion predictions to determine the development of the risk potential over time and evaluates it in simulated and real-world environments.

5.2.2 Concept

The concept is depicted in Fig. 5.11. There are currently three pipelines shown that contribute to the risk assessment. However, the list is not exhaustive and can be extended for arbitrary sources of risks. The risk sources included here are the multi-modal predictions of visible objects, occluded areas of non-bound objects (see NST in Sec. 4.2.3), and occluded areas of objects moving oriented along a reference path (see OST in Sec. 4.2.4). These components were chosen because each can not be expressed suitably in a metric and requires accumulated context knowledge and an awareness of the current situation to be rated accurately. Including the accumulated context knowledge implies that risk-aware trajectories can be planned with the assurance that the situational context has been captured, understood, and assessed.

The risk assessment unit determines the risks of each input channel and generates RMOT, what is described in the next section. Finally, the trajectory planner plans risk-aware trajectories based on the RMOT.

Risk Assessment

The risk assessment unit takes in the different inputs, determines RMOT for each input channel, and accumulates the different RMOT afterward to output a combined representation of the risk over time. A single risk map contains the risk values in a two-dimensional grid in a Cartesian coordinate system. The RMOT is a series of risk maps for each point in time with a discrete time interval Δt, from the current time up to the planning horizon T_P. The size, origin, and resolution of all grids of the series are the same.

The risk values are calculated for each individual cell. The calculation is conducted for every point in time, filling the corresponding risk map. In this work, the risk values of each input source $k = 0 \dots N-1$ are combined using a discounted weighted sum per cell

$$R_t = \gamma^t \sum_{k=0}^{N-1} w_k R_{k,t}, \tag{5.9}$$

with discount factor γ to be able to emphasis immediate risks and weights w_k to balance risks from different sources. The cell indices are left out for better readability. However, it has to be noted that other operators should be evaluated in future works, i.e., taking the maximum of the risk values $R_{k,t}$ to determine R_t.

The cell-wise risk for not road bound sets $R_{0,t} = R_{\hat{\mathcal{U}},t}$ is determined from the forecasting result $\hat{\mathcal{U}}$ of the NST. $\hat{\mathcal{U}}$ contains the forecasted sets of occluded areas for all time steps. Each element $\hat{u}$ refers to a 2D cell with a certain velocity. As the approach presented in Sec. 4.2.3 tracks multiple velocity values, the risk for a cell and time step is obtained from the sum over N_v velocity values of the collision risk with a potentially occluded object with velocity v_i. It is obtained from the influence or severeness Υ_{v_i} of a collision and the probability P_{v_i} with

$$R_{\hat{\mathcal{U}}} = \sum_{i=0}^{N_v-1} \Upsilon_{v_i} P_{v_i}. \tag{5.10}$$

The severness Υ_{v_i} is influenced by the type of objects that generally move with the corresponding velocity and the impact of the velocity on a collision. Additionally, fault and not-at-fault behavior can be modeled by, e.g., using a small factor for very high velocities that can only be from an occluded vehicle driving with excessive speed.

The second input source is the risk from oriented occluded objects $R_{1,t} = R_{\hat{Q},t}$ from $\hat{Q}$ of the OST. $R_{\hat{Q}}$ is calculated from the maximum velocity of the velocity ranges of a predicted element $\hat{q}$ as follows

$$R_{\hat{Q}} = \Upsilon_{v_{max}(\hat{q})} P_{v_{max}(\hat{q})}. \tag{5.11}$$

Similar to above, the severness Υ and probability can be used to represent prioritized roads or the traffic density. The cell of the risk map needs to be correlated with the elements $\hat{q}$ using their positions and dimensions.

Finally, the risk from multi-modal uncertainty-aware predictions, i.e., the risk from different potential developments $\mathcal{D}$ of a scene, $R_{3,t} = R_{\mathcal{D},t}$ is determined by first calculating the belief-print of each object.

Table 5.2: Parameters for the evaluation of the risk-aware trajectory planner

Parameter	Value
Δt_{sim}	$0.3\,\mathrm{s}$
T_P	$6\,\mathrm{s}$
ROI dimensions	$100\,\mathrm{m} \times 100\,\mathrm{m}$
NST and risk map resolution	$0.2\,\mathrm{m}$
OST resolution lateral	$0.4\,\mathrm{m}$
OST resolution longitudinal	$0.4\,\mathrm{m}$
speed interval of NST	$[0, 7]\mathrm{m/s}$
speed interval of OST	$[0, 8.333]\mathrm{m/s}$
γ	0.95
Frequency of prediction, risk assessment, and trajectory planner	$20\,\mathrm{hz}$
Frequency of NST and OST	$10\,\mathrm{hz}$

The belief-print accounts for the state uncertainty as proposed in [41]. The belief-print is entered into the grid with its value scaled with the probability of the corresponding prediction mode P_p. This results in

$$R_{\mathcal{D}_t} = \sum_{i=0}^{N_{modes-1}} w_{\mathcal{D}} P_p \tag{5.12}$$

for N_{modes} predicted modes for every cell occupied by the corresponding belief-print weighted with $w_{\mathcal{D}}$. Using the sum of different modes mirrors the property that when a cell is occupied for each possible development, the maximum risk is assigned because the probability of a collision is also maximal.

Integration in the Trajectory Planner

The obtained RMOT need to be integrated into the trajectory planner to determine risk-aware trajectories. The planner used in this thesis is a sampling-based planner using PSO. The theory of PSO is described in Sec. 2.6 and the application to the problem of trajectory planning in Sec. 2.7. The cost functional from (2.31) contains multiple cost terms, optimizing aspects like safety, comfort, and efficiency. An additional cost term C_R is added for the risk. C_R is scaled with a corresponding weight to balance the costs against the other cost terms. The risk values from the RMOT are extracted from the two-dimensional grid for the corresponding time with a simple lookup based on the footprints of the contour of the ego vehicle. The risk awareness can be parametrized by setting the weights for the cost terms accordingly. Being aware in this context does not mean avoiding risks in any case. It means knowing and weighing the potential risks carefully against all other factors. In the end, the vehicle makes informed decisions. It neither ignores risks nor behaves entirely anxious. This leads to decisions that are transparent for the passengers. It also increases safety. If two trajectories seem identical, but one was planned without considering the risk,

and the other was planned with considering risk, the latter is much safer because the decision was made deliberately, considering all eventualities.

5.2.3 Evaluation

The evaluation took place in simulation and on different automated vehicles. The parameters used are shown in Table 5.2. All grids, i.e., the RMOT and the grid for NST, had the same resolution and had the same size as the ROI. The loop rate of the prediction, risk assessment, and trajectory planner was $20\,\mathrm{Hz}$ on the actual vehicle. OST and NST had a lowered loop rate of $10\,\mathrm{Hz}$ to reduce discretization effects.

Scenario: Intersection with occluded area on incoming road

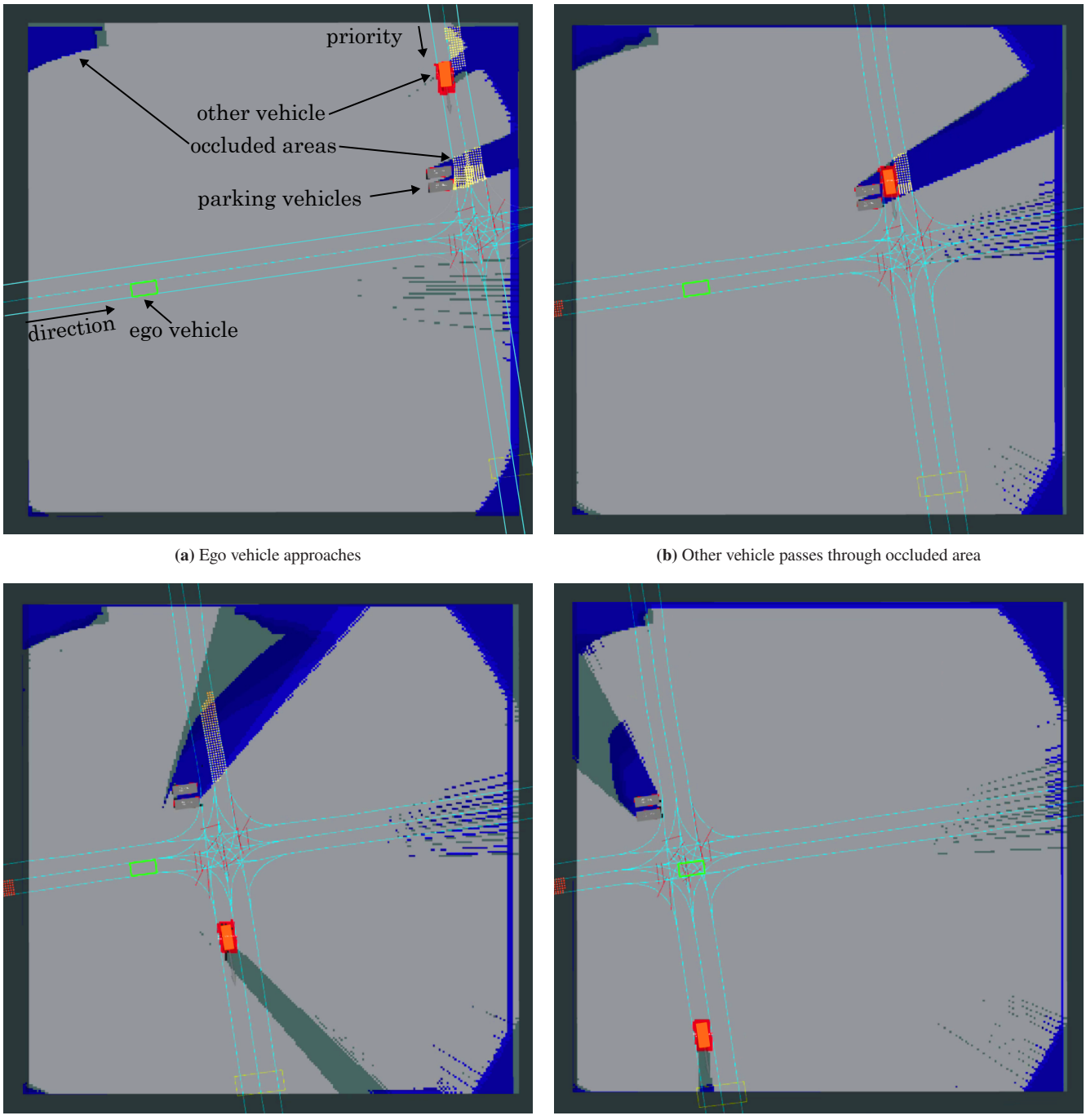
In the first scenario, the ego vehicle approaches an intersection where two parking vehicles block the view on the crossing prioritized road. It is shown in Fig. 5.12. Initially, hidden vehicles are assumed on the incoming road. As the ego vehicle approaches, one vehicle is perceived. When this vehicle passes through the area hidden by the parking vehicles, it becomes clear that no other road user can be there (Fig. 5.12b). In the following, the ego vehicle does not detect other incoming vehicles. Therefore, it reasons that it can cross the prioritized road safely without decelerating unnecessarily (Fig. 5.12d).

This scenario demonstrates that the OST leverages all context information to let the ego vehicle make informed decisions on how to handle the situation with regard to occlusions. It can also be observed that the areas tracked by OST that enter the scene behind the ego vehicle are drawn in red, indicating a high velocity. The areas that are occluded due to the parking vehicles are drawn in yellow because only very slow objects can be located there. Faster objects would have left the area in the meantime. For the occluded areas tracked by NST, the different velocity assumptions can also be observed at the edges of the tracked areas. When the ego vehicle moves and the FOV changes rapidly, e.g., around the parking vehicles, a darker blue of the NST areas indicates that only fast objects can be assumed in these areas.

Fig. 5.13a shows the related RMOT for the situation. In the RMOT, the ego vehicle is at the same position as in the corresponding view on the scene because the ROI and, therefore, the outer bounds of the area covered by the FOV and NST are aligned. For visualization purposes, only every second time step is depicted up to a horizon of $2.4\,\mathrm{s}$. In Fig. 5.13b and Fig. 5.13c the compositions of the RMOT are shown. From the left to the right, the combined risk, the risk from occluded areas tracked by NST, the risk from occluded areas tracked by OST, and the risk due to multi-modal predictions are shown. The corresponding risk values are color-coded. It has to be noted that the risk value is weighted again in the cost function of the trajectory planner. Thus, the values indicated here are mainly for introspection of areas with high and low risk. For example, before the other vehicle passes behind the parked vehicles, there is an area of high risk (Fig. 5.13b). With increasing time, the occluded areas become larger, and thus, the risk from occlusions also spreads. The discount for later times leads to a steady decrease in risk values over time.

Scenario: Overtaking obstacle with occlusions behind

In this scenario, the lane of the ego vehicle is blocked by an object, e.g., a parking vehicle. Behind the obstruction, a hidden pedestrian might appear. Additionally, the ego vehicle has to ensure that there is no oncoming traffic on the oncoming lane during the overtaking maneuver. The used vehicle had a different



(a) Ego vehicle approaches

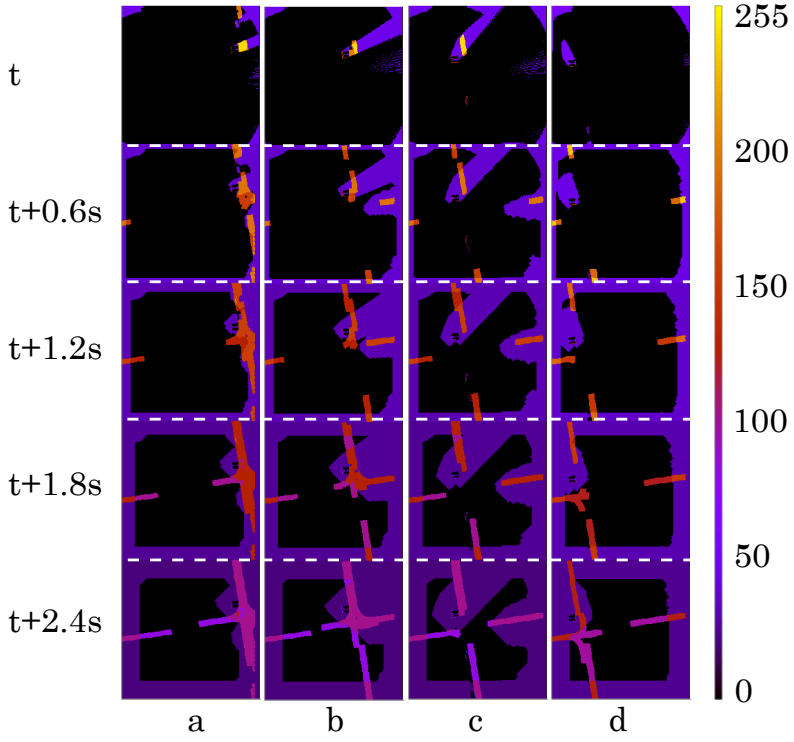
(b) Other vehicle passes through occluded area

(c) Passing of other vehicle invalidated assumptions about potential objects

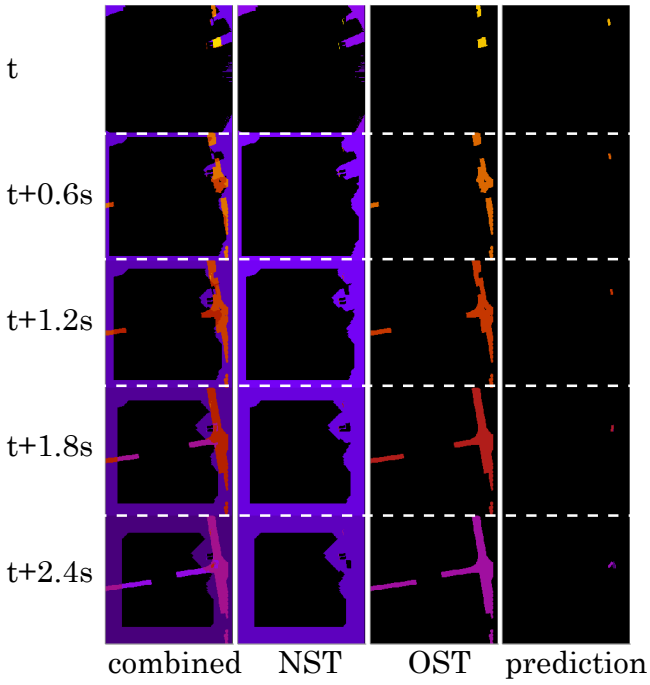
(d) Ego vehicle crosses intersection

Figure 5.12: The ego vehicle (green box) approaches an intersection from the left and has to cross a prioritized lane that goes from top to bottom. Another vehicle comes from the top (orange). Two parking vehicles (grey boxes) block the view. The areas of potential objects that follow the road network (blue lines) are depicted in yellow (OST). Areas where potential free-moving objects can be located are depicted in blue, and those that are occupied with dynamic objects are marked in red (NST). The grid-based FOV is shown as light grey in the background.

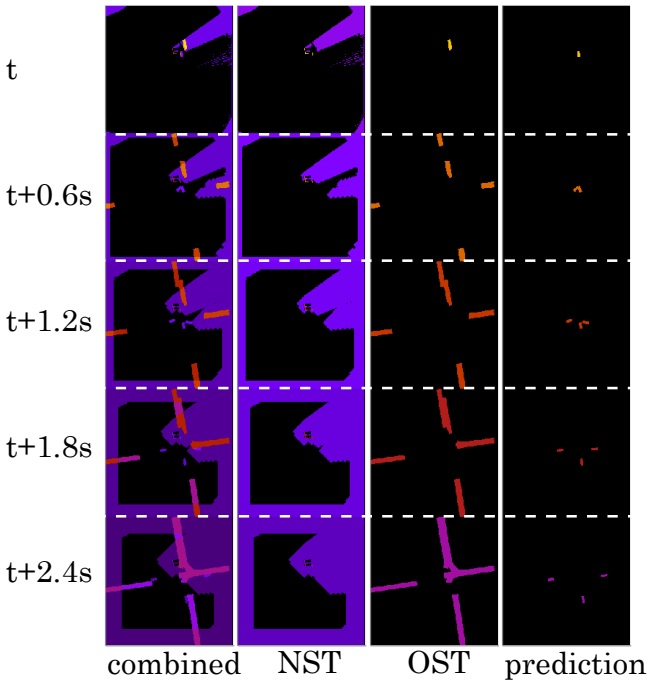
sensor setup focusing on forward sight and only minimal side and rear-facing coverage. Nevertheless, as the occlusion tracking approach is sensor agnostic, so is the risk assessment. The risk-aware approach is compared with the baseline, which consists of the unmodified automated driving functions, i.e., a trajectory planner without the proposed occlusion tracking and risk assessment. Fig. 5.14 shows a top



(a) Risk maps including predictions for Fig. 5.12a to Fig. 5.12d



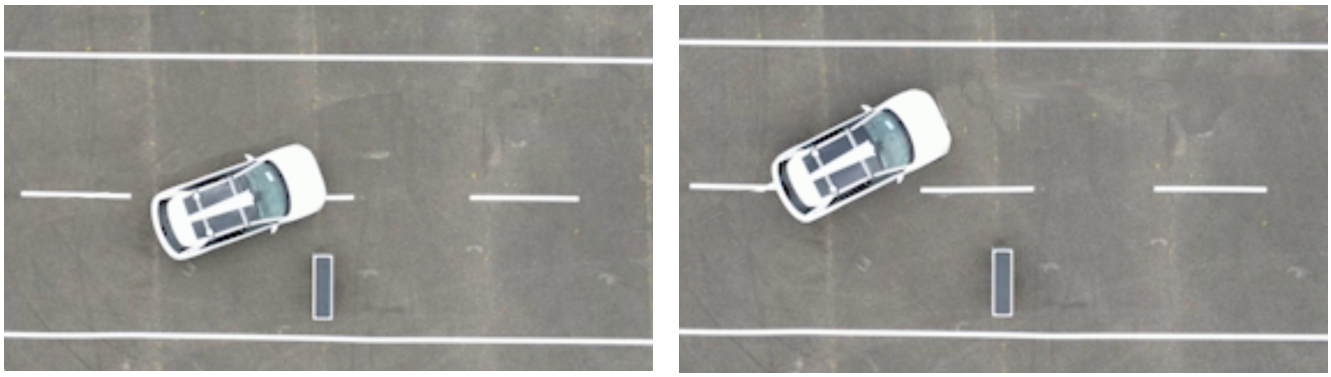
(b) Composition of the risk map for scene of Fig. 5.12a



(c) Composition of the risk map for scene of Fig. 5.12b

Figure 5.13: Depiction of the RMOT for the scenario shown in Fig. 5.12. (a) shows the RMOT for the situation at Fig. 5.12a to Fig. 5.12d. (b) and (c) show the compositions of the RMOT for Fig. 5.12a and Fig. 5.12b. From the left to the right, the resulting risk map, the risk from occluded areas tracked by NST, the risk from occluded areas tracked by OST, and the risk due to multi-modal predictions are shown. The corresponding risk value is color-coded.

view on the scenario. The obstacle obstructing the sight, in this case, was a box. On the right, the risk-aware approach is shown beside the baseline on the left. Evidently, the distance kept from the object



(a) Baseline

(b) Risk-aware

Figure 5.14: Comparison of risk-aware approach to baseline in top-down view on the scenario with potentially occluded pedestrian.

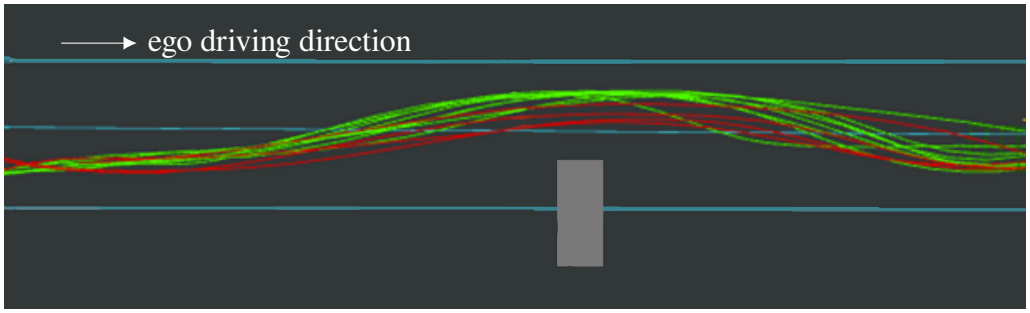


Figure 5.15: Comparison of the driven risk-aware and baseline trajectories of multiple runs in a top-down view on the scenario with potentially occluded pedestrian. The baseline is drawn in red and the risk-aware approach in green. The obstacle obstructing the view is drawn in grey, and road markings are in light blue. The ego vehicle drives from left to right.

is much more significant for the risk-aware approach ($\sim 0.5 - 1\,\mathrm{m}$). The risk-aware approach considers potentially occluded pedestrians appearing from behind the box. This can also be seen in Fig. 5.15. It can be seen that the behavior is repeatable, meaning that the situation is assessed similarly in each run. There was a bigger variation in the real-world experiments than when the experiments were repeated in simulation due to slightly varying outer conditions.

Fig. 5.16 shows the planned trajectories for passing the obstacle. The contours of the ego vehicle are drawn for each time step. It shows that the risk-aware trajectory passes the obstacle more slowly, as indicated by the shorter risk-aware trajectory. The planned trajectories again show a safer distance to the obstacle, allowing one to sooner observe the space behind the obstacle from a better angle.

The scenario of Fig. 5.14 was also part of demonstration at the final event of the project SafeADArchitect. The project's goals aligned with this thesis's goals to achieve risk-aware and comprehensible automated driving that can cope with everyday situations. At the final event, interested guests could experience the baseline and risk-aware behavior themselves. Therefore, the course was driven multiple times. First, the baseline version was shown. Afterward, the risk-aware trajectory was driven with and without a dummy pedestrian crossing the street from behind the obstruction. The general opinion was that the risk-aware approach was convincing, and the benefits in terms of safety were obvious. The parameterization in these runs was deliberately chosen so that a difference was visible and thus somewhat exaggerated. A challenge

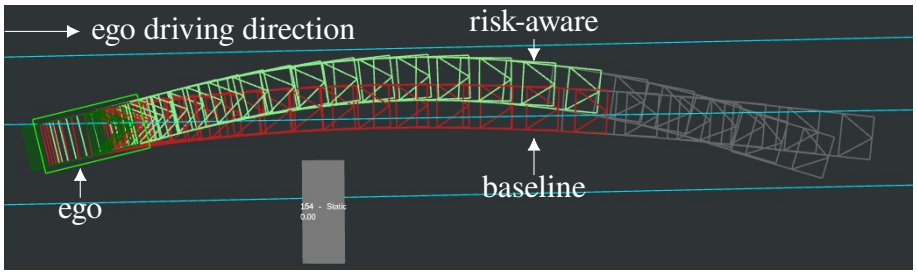


Figure 5.16: Comparison of the planned risk-aware (green) and baseline trajectories (red). The planned trajectories are depicted with their contours in RViz. The current ego position is shown in a light green frame. The previously passed states are drawn as dark green contours to visualize the seamless transition from the previous to the currently planned trajectory.

for subsequent works will be to parameterize in such a way that no difference can be seen, but the vehicle is always conscious and informed on the road.

In real-world test drives with Cocar, it could also be observed that the risk-aware planner keeps larger distances of around $0.5\,\mathrm{m}$ to the side of the road if there are moving objects or occluded areas behind bushes or parking vehicles.

Discussion

The presented approach showcased that arbitrary risk sources can be accumulated and integrated into the planning process to produce trajectories that consciously regard the surrounding risks. I.e., risks that cannot be expressed with simple metrics can be considered. Using the available context information leads to better assessments of the current situation. More accurate input allows for more informed and aware decisions. However, the approach comes with the cost of many parameters to determine, e.g., the weights of single risk sources, the weights of the planner's cost function, and the parametrization of the tracking and prediction algorithms. Additionally, there has yet to be a description of the maximum accepted risk level in the behavior of automated vehicles. This level also varies from one person to another and from one manufacturer to another. Learning techniques based on human-driven trajectories could be applied to find the optimal parametrization and balance between risk awareness and acceptance while maintaining a maximum level of safety.

5.2.4 Conclusion

The most important points of this chapter are the following:

- RMOT combine arbitrary risk sources spatially and temporally through a series of two-dimensional risk maps and they are also suitable for risks that cannot be expressed with metrics.
- Using all available context information for the risk assessment leads to safer and risk-aware trajectories of the vehicle.

5.3 Safe Application of Machine Learning for Trajectory Planning

Machine learning approaches excel in all kinds of domains. They are often employed because of their flexibility, generalization ability, and performance. Many applications use machine learning components, for example, for text completion. However, these are not safety-relevant functions, as they are required for fully automated driving. The use of machine learning poses risks and uncertainties regarding their output. Therefore, this section presents a strategy to combine the strengths of machine learning techniques and model-based algorithms for the example of trajectory planning for automated vehicles.

The previous section explained how trajectory planning is made aware of risks that are difficult to quantify. PSO was used to solve the optimization problem of finding the optimal solution while keeping to inner and outer constraints. The planning result of sampling-based algorithms, in general, strongly depends on the initialization quality of the samples. For PSO, the result depends on the initial particle swarm representing the initial trajectory candidates. The initial trajectories are often generated by rule-based heuristics and strategies. While these strategies yield good results for easy situations, the complexity quickly rises for difficult situations. This makes handcrafting corresponding heuristics an enormously tedious task. However, especially in complex situations, a good initialization is required for a proper response in no time due to a fast convergence to an optimal trajectory.

Therefore, this section presents an approach to use machine learning to generate initial trajectory candidates for automated driving in dynamic environments. These trajectory candidates are used to initialize the PSO, where they are checked for consistency and validity before the optimization. Due to the ability to generalize, the approach promises to make the time-consuming task of hand-tuning heuristics obsolete and a matter of the past. To the author's knowledge, the approach presented in [4] was the first to combine machine learning and sampling-based planning for trajectory planning in dynamic environments. The section may contain verbatim quotes from [4].

Two different initialization strategies based on deep learning are presented in the following. These are illustrated in Fig. 5.17 for an example scene. The ego vehicle has to overtake the parking vehicle. The first strategy generates probability distributions for each point in time, signaling the probability that the vehicle should be at the corresponding position at that time. Trajectories are then sampled from the respective distributions. The distributions are drawn in red. Two modes can be distinguished - either the ego vehicle overtakes the other vehicle, or it comes to a stop behind it. The second strategy directly generates various trajectories colored in blue in Fig. 5.17.

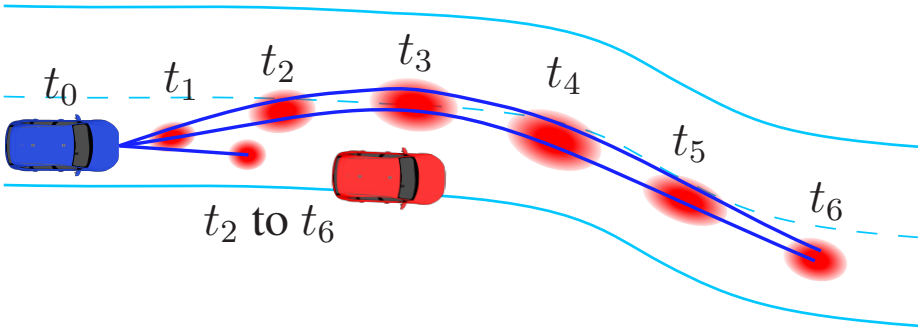


Figure 5.17: Depiction of the two strategies for an example scene. Distributions are shown in red, and trajectories in blue.

5.3.1 Related Work

Machine learning-based approaches produce very good results in environment perception tasks required for autonomous driving, i.e., object classification and data segmentation. These approaches also showed promising results for learned prediction and motion planning. However, machine learning approaches lack a guarantee for producing only feasible and safe results in motion planning. A possible solution to leverage both the data-driven intuition of learning techniques and the optimality and traceability of traditional algorithms is to validate the network output with a rule-based component. An example is to generate learned heuristics for the planning algorithms. In [82], Ichter et al. present an approach to improve the sampling efficiency of sampling-based algorithms like RRT* [104] by using conditional variational autoencoders (CVAE) and demonstrate it on robotic applications. They showed that the learned sampling distributions reached a faster algorithm convergence than uniform sampling because the samples are closer to the optimal path.

Improving the planning time and the convergence by guiding the sampling using neural networks was also shown by Qureshi and Yip [124]. A combination of a contractive autoencoder for the environment encoding and a fully connected network with the current and goal states is used to predict the next optimal state. Samples around the optimal path are generated using dropout in the forward passes. The process is repeated using the previously predicted state as a new input state to recursively obtain the next state until a total path to the goal is reached.

One of the key performance factors is the representation of the environment. It is a crucial design decision in all machine learning tasks and, therefore, when using neural networks to improve the planning or prediction process. The authors of ChauffeurNet [40] chose a representation based on multi-layered grids. The layers of the grids describe different aspects of the environment, e.g., the ego vehicle's past states, oriented bounding boxes of recognized obstacles, or the road layout. Using this representation, they are able to successfully demonstrate imitation learning based on CNNs in the context of autonomous driving. The vehicle controller executes the trajectories without further validation. In Sec. 4.4, also a grid-based sensor-agnostic environment representation was used to evaluate the prediction performance of different CNNs in highway-like traffic scenes. The approach is also suitable to predict scenes even when occlusions are present. This is achieved using an iterative process to predict multiple steps and propagate the loss back over all prediction steps.

In [52], the short-term prediction of vehicles in dynamic environments of traffic agents based on RGB raster images is shown. In contrast to the previously mentioned approaches, individual images for each agent from the corresponding perspectives are used, including information about road layouts and positions of other agents. The features extracted from these images using a CNN are then combined with a one-dimensional feature vector containing scalar information like velocities. They are fed into a fully connected network, yielding a fixed number of trajectories for predefined modes.

Improving the sampling performance of a sampling-based path planner in the context of autonomous driving in static environments was shown in [42]. In contrast, this chapter is focused on dynamic motions and environments. In [42], the environment is represented in a multi-layer grid similar to [40], containing information about free or occupied cells and the vehicle's current state. The input is fed into a CNN to generate grid-based probability distributions for the vehicle's positions and orientations. The authors showed a significant improvement in the convergence of the sampling-based path planner. Using CNNs to guide the sampling of an RRT* was also done in [123] to improve path planning for a robot by predicting its positions given the environment and the start, as well as the goal position.

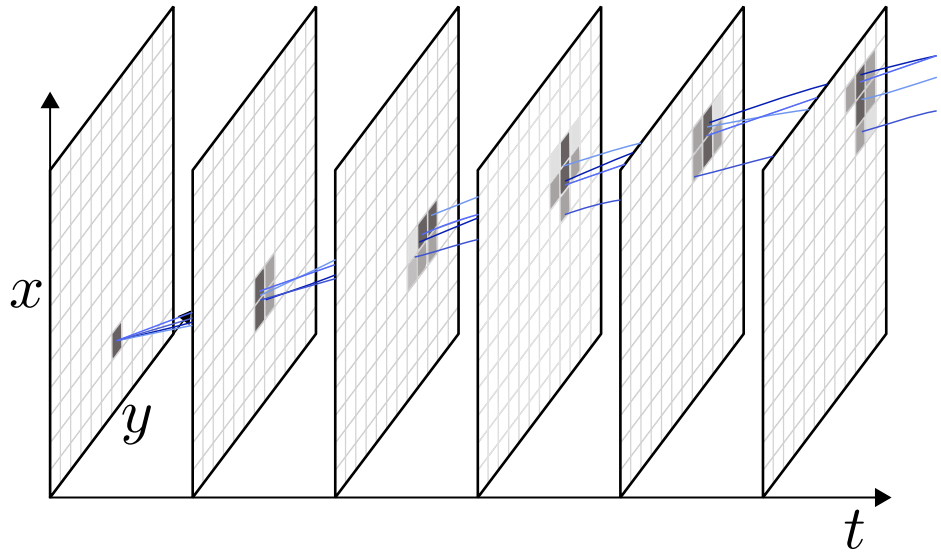


Figure 5.18: Visualization of the proposed approaches. The heat maps can be seen as discrete probability distributions of the vehicle's potential position at a certain time (distribution shown in grey colors). Further sampling from these distributions is required to generate trajectories. The approach based on dropout sampling directly generates trajectories (shown in blue). (Graphic from [4], ©2020 IEEE)

Guiding the initial sampling process for a motion planner based on PSO was pursued in [80]. Hubschneider et al. combined an end-to-end driving approach [79] and the sampling-based planner to predict steering angles for the vehicle and to improve the initial particle sampling process using camera images and CNNs. The amount of invalid sampled particles could be reduced. However, the strengths were mainly evident in short planning horizons. For longer planning horizons, more context information is required.

5.3.2 Concept

A heuristic for generating initial trajectory candidates for driving in dynamic environments can be shaped in different forms as long as the heuristic provides information about the state of the ego vehicle at the corresponding point in time. Two different representations are proposed here. The first approach provides a probability distribution for each time step for the discretized area around the ego vehicle in the form of a grid map. They are called heat maps. The trajectory candidates are then sampled from these distributions. Hereby, the network follows an encoder-decoder design. The second approach directly generates several trajectory candidates with a fully connected network using the previously extracted features. The variation in the trajectory candidates is generated through dropout as proposed in [124]. Fig. 5.18 shows the comparison of both strategies. The heat maps are drawn as grids with likelihoods marked in greyscale, whereas the dropout sampled complete trajectories are drawn in blue.

Both approaches use CNNs to extract information on the environment tensor, which is further described in the following section before presenting both approaches in detail and discussing their advantages and disadvantages.

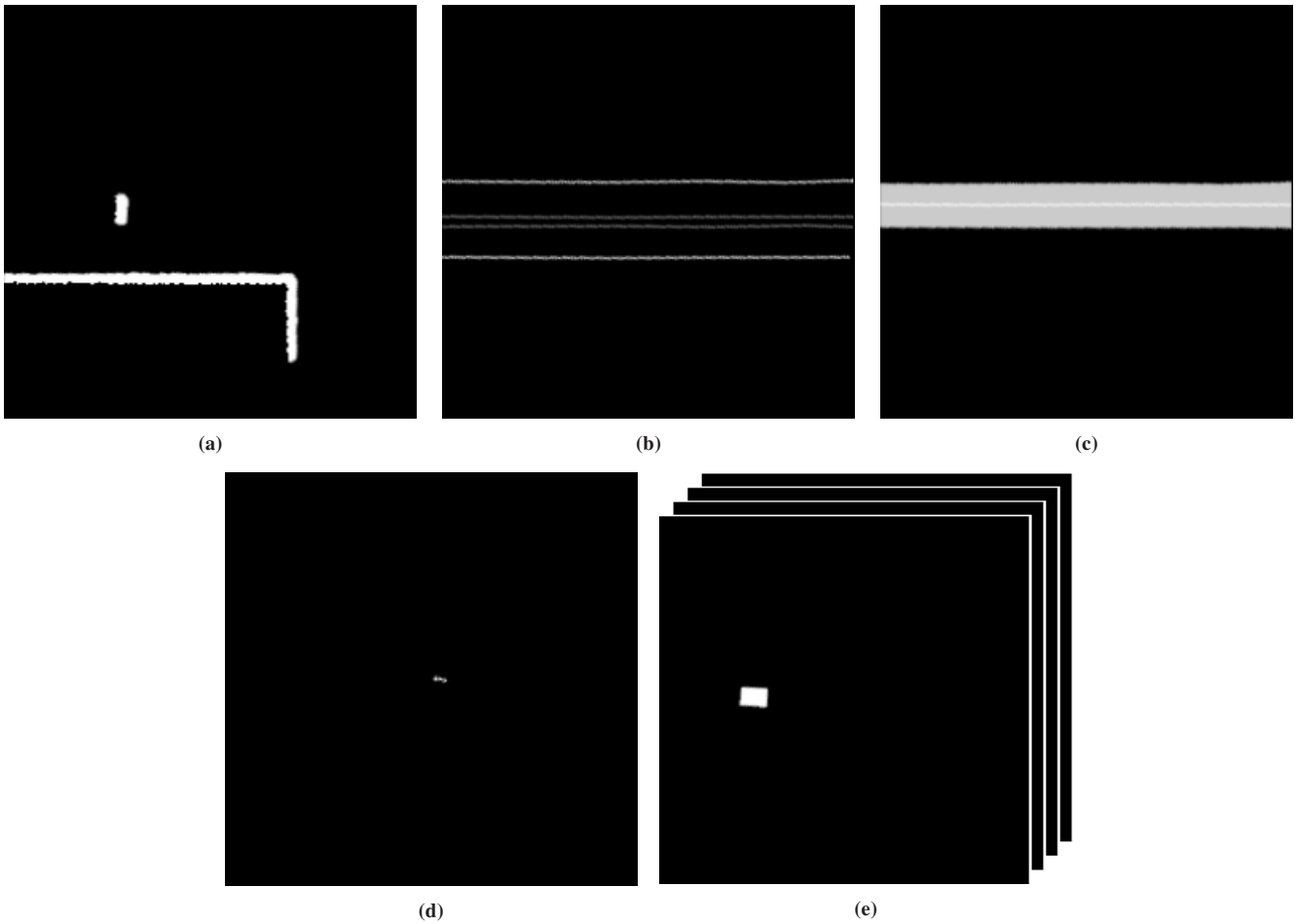


Figure 5.19: Example for the grid-based environment tensor. The ego vehicle drives from right to left with one vehicle in front and a building on the left side of the road. (a) Occupancy grid built from five Lidar sensors. (b) The road layout including boundaries and stop-lines at intersections. (c) The area where the vehicle is currently allowed to drive in. (d) The previous positions of the ego vehicle in order to generate consistent trajectories. (e) The previous and current bounding boxes of detected obstacles. (Graphics from [4], ©2020 IEEE)

Input and Environment Representation

The representation of the environment is depicted in Fig. 5.19. The components described in Sec. 3.3 are translated to a grid tensor for the learning task. As also explained in Sec. 4.4, the grid tensor representation is sensor agnostic. However, the representation differs from the one presented in Sec. 4.4.2 because the task is also different. The grid tensor is centered on the current position of the ego vehicle. It is also normalized to its heading so that the ego vehicle always drives to the left for the grid tensor at the current time. In detail, the channels are chosen as follows:

- The **occupancy** layer is in the form of an occupancy grid. It represents the static environment and can be used as a representation of the FOV, see Sec. 2.8.2. The grid is generated from Cocar's five Lidar sensors. Fig. 5.19a shows the contours of a building and the rear of a vehicle.
- The information from the **road layout** is transferred from a map into a grid by providing information about lane boundaries, see Fig. 5.19b. The type of boundary is encoded through the intensity.

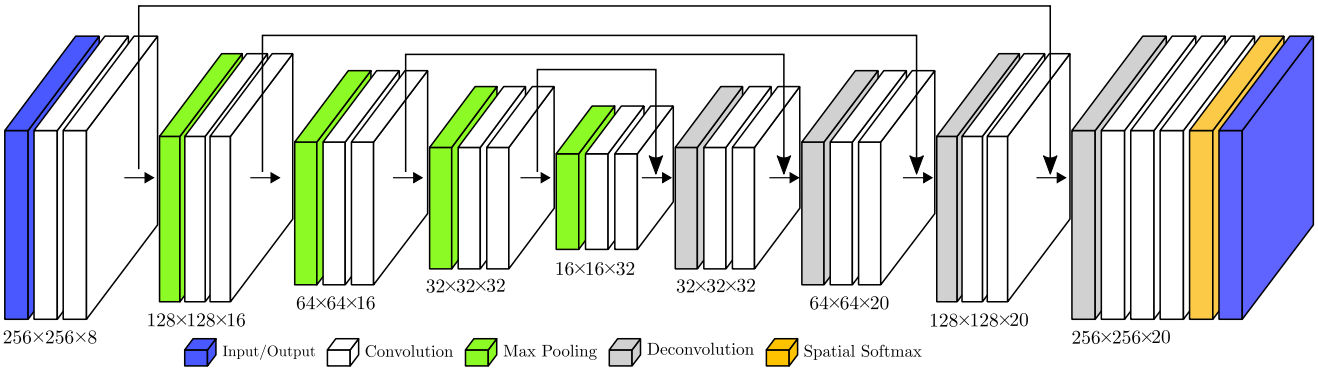


Figure 5.20: U-Net layout of the network to predict heat maps. The input is a tensor of dimension eight, and the output consists of 20 heat maps, one for each point in time of the trajectory. The trajectory candidates are then sampled from the resulting heat maps. (Graphic from [4], ©2020 IEEE)

- The **driving area** is where the ego vehicle is allowed to drive (Fig. 5.19c). It is built from the ego vehicle's route, the corresponding lanes, and potential extensions to the side, e.g., to overtake parking vehicles. The intensity is scaled with respect to the current desired velocity. The lateral bias of the ego vehicle is encoded with a different intensity than the rest of the driving area.
- The **trajectory prefix** provides previous positions of the ego vehicle. It allows seamlessly connected, consistent, and continuous trajectories. The intensity of the points is scaled with the respective velocity.
- The **history of obstacles** in the environment is depicted as oriented bounding boxes over time. It allows the networks to distinguish between static and dynamic objects. The information of several time steps is provided because the network has to predict the environment up to the planning horizon to deliver accurate heuristics for the trajectory initialization. The obstacle layers are given at constant intervals and have the same spatial reference point. Four layers with a time gap of $0.5\,\mathrm{s}$ were used. This results in a history of $1.5\,\mathrm{s}$, including the current time step.

All grid layers are centered on the current position of the ego vehicle with an area covered of $51.2\,\mathrm{m} \times 51.2\,\mathrm{m}$ with 256×256 cells at a resolution of $0.2\,\mathrm{m}$. The dimensions and resolution must be balanced between accuracy, available look-ahead distance, planning horizon, computational resources, covered space, and containing as few non-relevant areas as possible.

Initialization using Heat Maps

The heat map approach extends [42] and [123] for dynamic environments. Using CNNs, all trajectory positions in dynamic environments are predicted instead of only the next one.

The network output contains one heat map for each time frame $i = 0 \ldots N-1$. The i-th heat map contains the probability distribution of the i-th position. Thereby, each cell contains the likelihood of containing the optimal position of the trajectory at that time. The output dimensions equal the input area with $256 \times 256 \times N$ cells.

To better preserve the spatial context, a U-Net architecture as proposed in [128] was chosen. It is depicted in Fig. 5.20. Feature maps from the encoder are given to the corresponding upsampling layers in the decoder and are then leveraged in the transposed convolutions. Further architectures were investigated in [25]. Batch normalization [83] is applied in the convolutional layers. The chosen activation function is

ReLU activation [66]. To obtain a distribution that resembles a probability distribution, each heat map in the output is normalized using a spatial softmax.

Each cell in the output can be regarded as a separate class. During training, the network learns to classify the grid cell that most likely contains the position of the optimal trajectory for each time step, i.e., in each of the $N-1$ heat maps of the output. This results in $N-1$ heat maps containing the probability distribution mentioned above. To calculate the loss, the target trajectory is also encoded in a sequence of $N-1$ heat maps. In each target heat map, the cell that contains the position of the target trajectory obtains a value of 1. All other cells remain at 0. The loss is then calculated with categorical cross-entropy for each predicted heat map

$$\mathcal{L}_{CC,i} = -\sum_{c_x=0}^{W-1}\sum_{c_y=0}^{H-1} \hat{h}_i(c_x, c_y)\log\left(h_i(c_x, c_y)\right) \tag{5.13}$$

where $W \times H$ are the dimensions of the heat maps, $\hat{h}_i(c_x, c_y)$ is the intensity at grid position (c_x, c_y) of the i-th target heat map and $h_i(c_x, c_y)$ is the intensity at grid position (c_x, c_y) of the i-th predicted heat map. The overall loss is determined through the average loss over all time steps

$$\mathcal{L} = \frac{1}{N}\sum_{i=0}^{N-1}\mathcal{L}_{CC,i}. \tag{5.14}$$

The trajectory candidates of length N are then sampled from the time series of heat maps in the following way

1. Start either from the ego vehicle's current state or the prefix of a previous trajectory. The length is $i < N$.
2. Sample the cell $c_{i+1,sample}$ from the corresponding heat map $i+1$ using the encoded distribution. The values contained in the cells represent a discrete probability density. The values are accumulated in a vector to obtain a probability distribution. Afterward, a uniform random value in the interval $[0, 1]$ is generated to determine the index in the vector. The index, in return, can be used to determine the corresponding cell.
3. Sample a continuous position x_{i+1}, y_{i+1} by independently sampling uniformly inside the boundaries of cell $c_{i+1,sample}$.
4. Limit the position to the so-called circle of forces. It describes the total force a wheel can transfer to the road surface and thus correlates to the area where the next trajectory point will be located. This is important to obtain feasible trajectories only. Repeat 3 until a valid position is found. If no valid position can be found in the cell, repeat from 2.
5. Append position to current trajectory and continue with sampling the position for next step until trajectory length N is reached.
6. Check for validity of the trajectory. Repeat from 1 until a valid candidate is found. The validity constraints are shown in Sec. 2.7.
7. Add trajectory to the list of candidates.

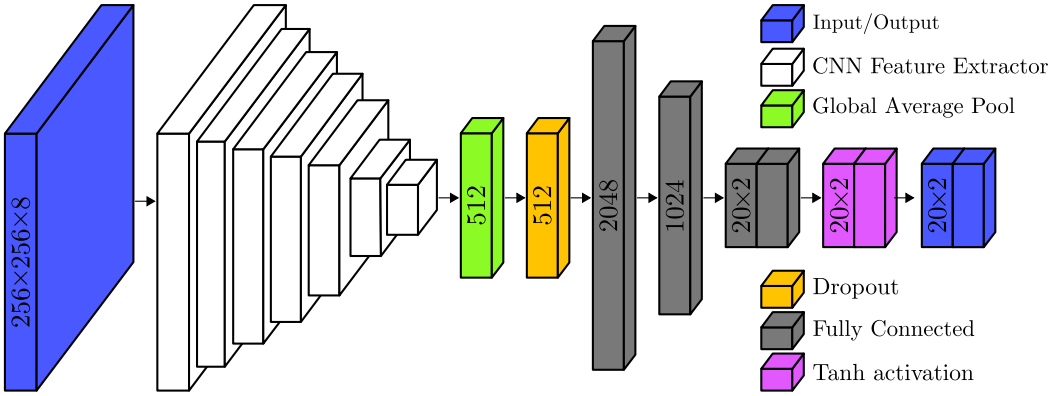


Figure 5.21: The input is first processed by a feature extractor based on CNNs. The resulting feature map is concatenated and fed into a fully connected part. The result is a vector of size $N \times 2$ comprising the x and y components of the positions of a trajectory. In the example, N is 20. Dropout is applied in a dropout layer to vary the resulting trajectory to obtain a variety of different trajectories to initialize the optimization process. (Graphic from [4], ©2020 IEEE)

The process is repeated until the desired number of trajectory candidates or particles is found. Due to the uncorrelated sampling of cells and positions, except for enforcing dynamic constraints, the trajectories often show a zig-zag pattern and, therefore, a lack of smoothness. This results in high initial particle costs. As the heat maps are predicted once initially, the encoded probabilities are not conditioned on the samples drawn in the previous steps. More elaborate sampling techniques could be developed to overcome this shortcoming in the future. Furthermore, generating conditioned heat maps for each candidate and sampling step is currently too computationally expensive.

The representation as a two-dimensional probability distribution is ideal for handling multi-modalities, for example, passing an obstacle on the left or the right side or deciding between an accelerating or decelerating movement. The architecture is suitable for retaining fine-grained spatial information in the deeper layers, making predictions potentially more exact in relation to inputs like the road layout.

Initialization using Dropout

A perpendicular approach to generating trajectory candidates is the direct generation of single trajectories over the total planning horizon. It is related to [124] and [57]. As mentioned above, the features are extracted with a CNN. The final layer of the CNN is concatenated and fed into a fully connected network. Using a dropout layer, the variety of the particle swarm is achieved. The feature extractor consists of max pooling layers followed by two convolution layers and a final output of $16 \times 16 \times 512$. The complete network outputs N positions with x and y coordinates relative to the center of the input grid. The positions, just like the heat maps, have constant time gaps. The values are scaled to $[-1, 1]$ with a tanh activation to ensure that the predictions are inside the defined area. The absolute positions can be obtained using the size of the grid. The architecture is shown in Fig. 5.21.

The trajectories for the initial particle swarm are efficiently generated by running the network multiple times on the same input with dropout activated in a single batch. Dropout of 50% was used during training and varied during inference. As shown in [60] and as discussed in Sec. 4.4.2, dropout can also be used to model uncertainties in neural networks. A more diverse set of trajectory candidates correlates to higher uncertainty about the optimum. A condensed set indicates that the network is sure about the latter optimum.

The applied loss determines the mean squared error of the displacement error between output and target based on the Euclidean distance

$$\mathcal{L}_{ED,i} = \sqrt{(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2}, \tag{5.15}$$

where (x_i, y_i) are the network predictions for time step i and $(\hat{x}_i, \hat{y}_i)$ are the positions of the ground truth or target trajectory at time step i. The total loss is defined as

$$\mathcal{L} = \frac{1}{N} \sum_{i=0}^{N-1} \mathcal{L}_{ED,i}. \tag{5.16}$$

In contrast to the heat map approach, the trajectory positions are timely correlated, leading to more consistent and smooth trajectories as the network considers the relationships between successive steps. Thus, the trajectory candidates are already relatively smooth and consistent, and the number of invalid trajectories due to dynamic constraints is lower. Nevertheless, the trajectories are again modified, and the positions are forced into the circle of forces.

The dropout approach does not explicitly provide different prediction modes. However, they might be determined after generating and clustering multiple trajectories. Additionally, multiple networks for specific modalities could be trained.

Data Generation and Training

Auto labeling through recording optimal trajectories and the corresponding environment data is used to obtain a vast amount of training data. The data was generated using simulation. The ego vehicle was driving in a digital twin of the target area with various scenarios and other traffic participants. To ensure converged trajectories close to the global optimum, the trajectory planner based on PSO was executed with a higher number of particles and iteration cycles. An increased number of particles better covers the search space. More iteration cycles ensure a converged solution at the end of the optimization. When the global optimum is found, the influence of the initial sampling heuristic is neutralized. The simulated vehicle is equipped with the same sensor setup as the real vehicle. Also, the same automated driving functions and especially the same trajectory planner are used.

The data is multiplied by augmentation. After being transformed into the ego vehicle's perspective, the data is rotated by a random angle in the interval $[-25°, +25°]$.

The networks are trained offline using the Keras framework in Python. The evaluation, on the other hand, is conducted in a closed loop. The closed-loop evaluation is required because situations can evolve differently if another trajectory is executed. The network is embedded into a ROS node using Keras in Python for the inference. All networks were trained using the Adam optimizer, an initial learning rate of 10^{-4}, and a batch size of 64 or 32, depending on the parameter count.

5.3.3 Evaluation

The learned initialization was evaluated on characteristic scenarios, namely a straight, curved, and narrow curved road, as well as braking, overtaking, and car following scenarios. An example of the overtaking scenario is shown in Fig. 5.22. The baseline is the unmodified planner using the current rule-based

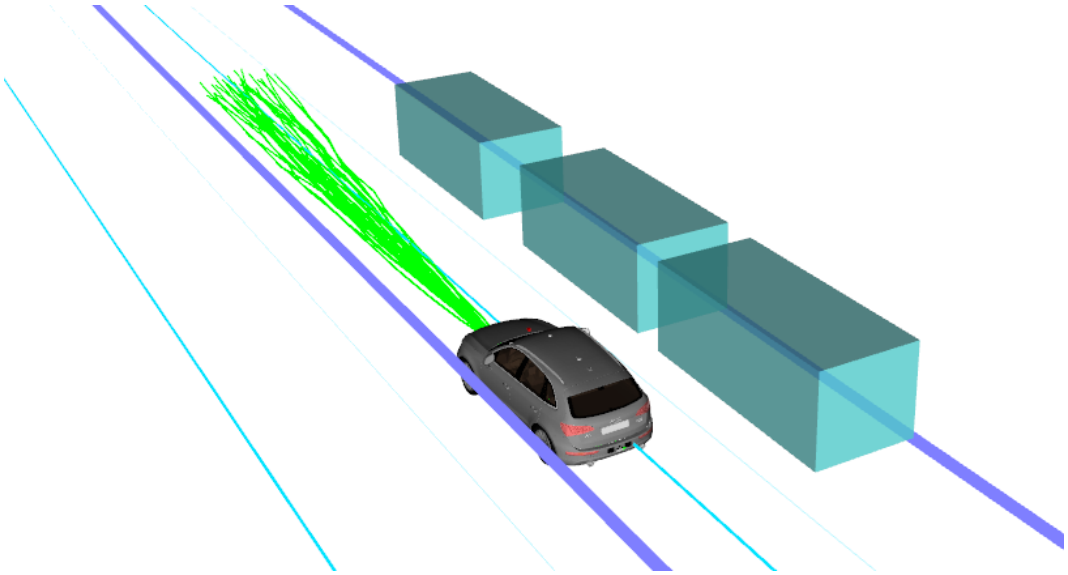


Figure 5.22: Exemplary situation for overtaking parked vehicles (shown as boxes) on the side of the road. Boundaries of the driveable area are marked in thick lines, and other road markings are marked in thin lines. The sampled trajectory candidates by the dropout network are shown in green. (Graphic from [4], ©2020 IEEE)

heuristics. The planner has been developed for several years, applied in everyday traffic, and is also used for automated passenger transport. The heuristics are designed and hand-tuned with a lot of expert knowledge. Hence, the baseline already meets very high-quality standards. The evaluation results are given in the two tables Table 5.3 and Table 5.4. The results are in percent with respect to the baseline performance because the absolute costs vary depending on the current situation. A percentage less than 100% indicates that the approach performs better than the baseline because the costs are smaller. The results are averaged over constant time intervals instead of over the total number of planning results. This avoids the influence of the duration of a planning cycle. In challenging parts of the scenarios, the optimization takes longer. Thus, fewer planning results are available than in simple parts.

Table 5.4 shows the results when the whole particle swarm is generated using the dynamic state of the ego vehicle or the recent trajectory prefix, i.e., the positions of the previous trajectory that were already passed by the vehicle. In contrast, Table 5.3 uses the previous planning result. This means that the recently planned and already optimized previous trajectory is extended. The extension is done by first removing the positions that the ego vehicle passed in the meantime. Afterward, the corresponding number of positions is added at the end of the trajectory. Alternatively, positions are deliberately removed from the end of the previous trajectory, and the trajectory is extended from there to enforce higher entropy in the swarm. Both initialization types regularly occur, and both types are important to evaluate. In the optimal case, the planner can reuse its previous optimized result to reach faster convergence. However, the previous solution can also become invalid due to unexpected changes in the environment.

The simulation was used to have accurate reproducibility for the closed-loop evaluation. This guarantees identical starting conditions for the evaluation scenarios. The cost function of the trajectory planner rates the particles or trajectories. To lay the focus of the evaluation on the initialization, the best and average initial costs and the resulting costs after the optimization are investigated. As mentioned in the data generation above, the initialization's influence disappears with an increasing number of optimization cycles. Therefore, the number was reduced to 10 for the evaluation as the influence of the initialization is examined.

Table 5.3: Average evaluation results for different scenarios compared to the baseline. The results are given in percent with respect to the baseline result. The resulting costs (Result), the best particle fitness in the initial swarm (Best initial), and the average fitness of all particles in the initial swarm (Average initial) are displayed. (Table from [4], ©2020 IEEE)

		Result	**Best initial**	**Average initial**
straight	Heat map	98.7%	98.7%	128.0%
	Dropout	103.7%	103.7%	272.5%
curve	Heat map	108.1%	116.4%	189.7%
	Dropout	90.6%	90.4%	85.8%
narrow curve	Heat map	101.9%	155.4%	178.9%
	Dropout	93.8%	93.7%	52.7%
braking	Heat map	105.5%	106.5%	126.3%
	Dropout	99.8%	99.7%	95.0%
overtaking	Heat map	96.5%	101.9%	117.0%
	Dropout	94.3%	94.2%	151.6%
car following	Heat map	92.7%	96.9%	102.2%
	Dropout	96.8%	96.8%	93.2%

The resulting costs are a measurement of how good the final solution is. It is the cost of the best particle, meaning the particle with the smallest costs in the swarm. If the baseline and the two approaches reached the optimum, the costs would be identical. A good initialization results in faster convergence so that the resulting cost rates how suitable the initial particle swarm was distributed for the latter optimization. The best and average initial costs are a reference for how well the particles are distributed around the optimum before the optimization. The best initial costs describe the costs of the best particle in the swarm after initialization. The average initial costs are the average of the costs of each particle in the initial swarm.

Both tables show that the heat map approach seems to provide a worse initialization than the baseline, i.e., produces costs >100%. This might result from the expected high costs for comfort and smoothness because of the sampling strategy in the heat maps and because the heat maps are not conditioned on the preceding heat map. The strong improvement of the average initial costs compared to the resulting costs in Table 5.3 supports this assumption. The optimization makes the trajectories smoother, leading to lower dynamic costs and, thus, to more significant improvements than the initial trajectories of the baseline. Additionally, through the discretization of the grid and the uniform sampling in the cells, the positions of the trajectories might be closer to the borders of the road or obstacles than desired by the network. Nevertheless, the heat map correctly predicts the possible positions of the future trajectory and provides a good heuristic for the initialization because the resulting costs are often better than the baseline. The heat map also proved to estimate the position in the presence of obstacles better, which can be seen in the scenarios "overtaking" and "car following" in Table 5.3.

When generating the initial trajectories without the previous planning result, the heat map performs better than the baseline for the driving straight scenario; see Table 5.4. The heat maps are very suitable for predicting the optimal longitudinal movements. However, the curve scenario shows the weaknesses of the heat maps as described above.

Table 5.4: Average evaluation results for different scenarios compared to the baseline without reusing the previous solution as a heuristic. For description see Table 5.3. (Table from [4], ©2020 IEEE)

		Result	**Best initial**	**Average initial**
straight	Heat map	93.33%	92.25%	74.25%
	Dropout	111.31%	110.38%	40.10%
curve	Heat map	176.82%	240.36%	195.19%
	Dropout	77.75%	76.08%	65.19%
braking	Heat map	82.50%	86.17%	106.87%
	Dropout	101.27%	101.12%	89.61%
overtaking	Heat map	144.17%	159.28%	131.74%
	Dropout	93.04%	91.99%	58.03%
car following	Heat map	103.42%	117.90%	135.66%
	Dropout	95.91%	95.84%	87.29%

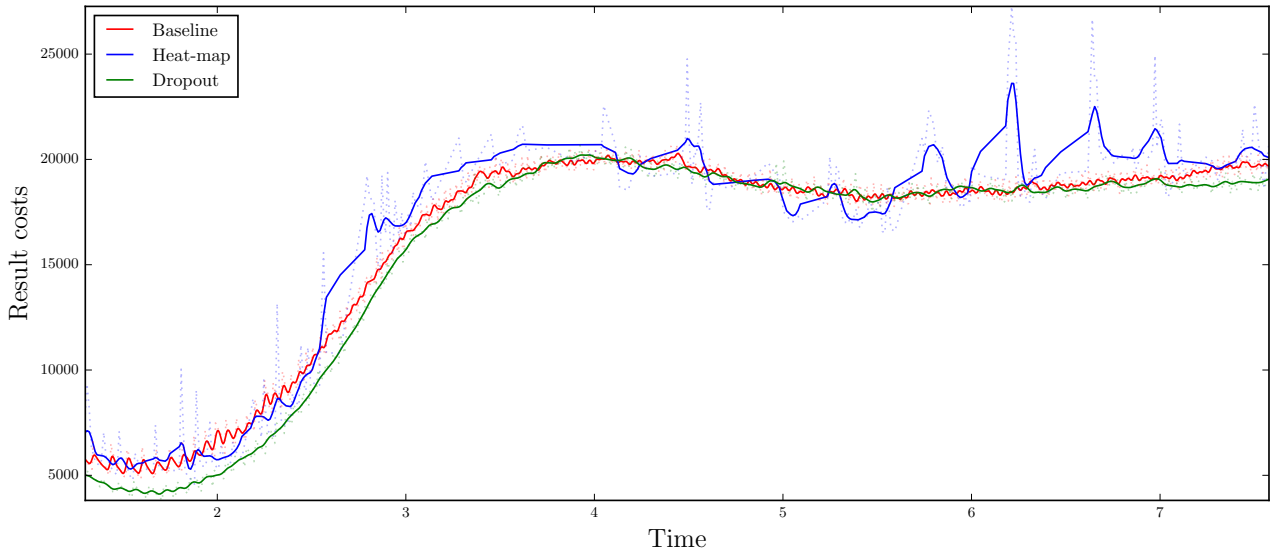


Figure 5.23: Resulting costs over time for the car following scenario without reusing the previous planning result. The data is smoothed by a Gaussian kernel. The unsmoothed data is drawn as transparent dashed lines. (Graphic from [4], ©2020 IEEE)

Fig. 5.23 depicts the resulting costs and Fig. 5.24 the average initial costs over time for the car following scenario. The costs increase as the ego vehicle comes close to the vehicle in front ($t = 0 - 4s$). The heat maps provide reasonable results, but the initialization quality could be better due to the weaknesses discussed above.

The trajectories generated using the dropout approach have lower costs in most scenarios, with and without using the previous planning result. The approach especially performs better in curves and generates very good approximations of the latter optimum, what can be observed in the best initial costs. The sampling of complete trajectories at once produces smooth trajectories and thus creates very low dynamic costs. The good performance of both approaches in the overtaking and car following scenarios

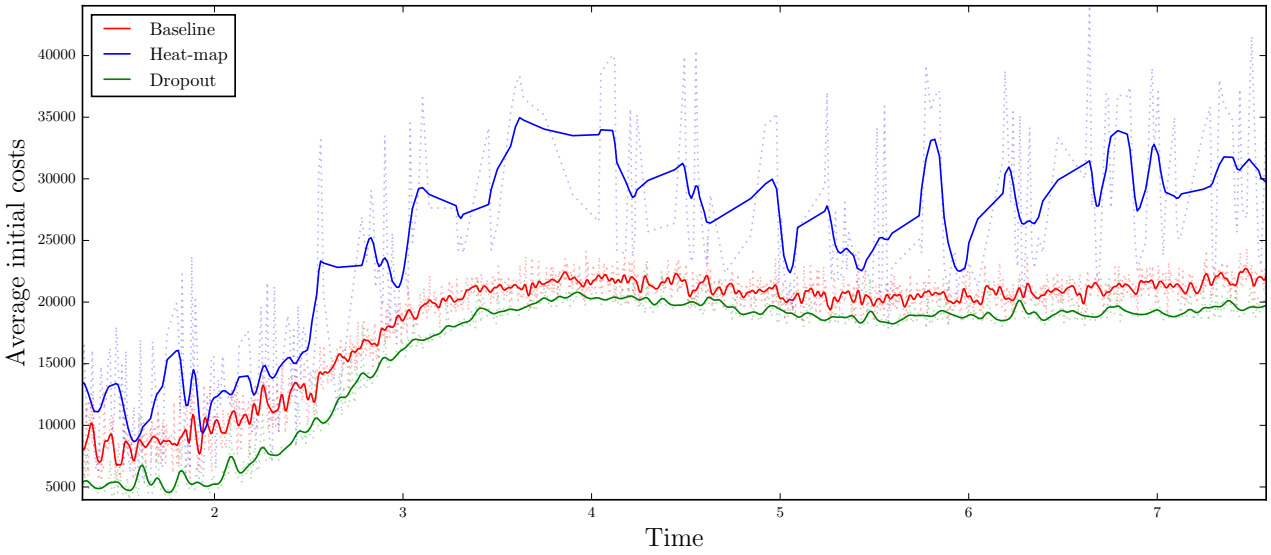


Figure 5.24: Average initial costs over time for the car following scenario without reusing the previous planning result. The data is smoothed by a Gaussian kernel. The unsmoothed data is drawn as transparent dashed lines. (Graphic from [4], ©2020 IEEE)

shows that the networks are both able to predict the obstacles correctly and differentiate between static and dynamic obstacles. In the straight street scenario, the dropout approach produces worse results than the baseline. Especially when the trajectories are initialized entirely new without using the previous result, see Table 5.4. However, the average initial costs show that the initial swarm was distributed closely around the optimal solution. As mentioned above, the best particle greatly influences the convergence speed, and the resulting costs and hand-tuned heuristics are almost perfect for this use case because they do not require a lot of context to be considered.

In the car following scenario (costs over time shown in Fig. 5.23 and Fig. 5.24), the dropout approach consistently beats the baseline.

In total, the different approaches have different strengths and weaknesses. None of the approaches dominates in every scenario. However, this is not absolutely necessary because the initial particles can be generated using different strategies. So, the strengths are combined, and the weaknesses are mutually compensated. This also accounts for the use of neural networks in safety-critical applications like trajectory planning. The output of the network is verified through the model-based optimization. When hand-tuned heuristics are used for basic behaviors and learned heuristics for complex ones, there is always a valid trajectory candidate in the improbable event that the network cannot produce a valid trajectory.

The evaluation was conducted with 20Hz on a workstation. The sampling, optimization, and preprocessing, which mainly consists of generating the environment grid tensor, was executed on the CPU. The inference of the neural networks was carried out on the GPU. The baseline planner can be executed with 50Hz. However, runtime optimization of the neural networks and the preprocessing have yet to be done. Thus, the concept can also be transferred to an actual test vehicle if a sufficiently powerful GPU is available and the networks are trained on real-world data.

5.3.4 Conclusion

This section included the following key points:

- Combining machine learning with model-based algorithms to verify outputs of neural networks for trajectory planning replaces hand-tuned heuristics to meet the complexity and generalization abilities of the planner.
- Two initialization strategies were proposed that differently generate the diversity in the initial trajectory samples for a trajectory planner based on PSO. As no single strategy is superior in every situation, combining strategies promises the best initialization quality.
- A better initialization of a sampling-based algorithm promises faster convergence to the optimal solution. For a trajectory planner, this results in faster response times.

5.4 Chapter Conclusion

Based on the belief of the current state of the environment, this chapter presented approaches for maneuver and trajectory planning that allow foresighted and risk-aware driving while answering the third initially posed research question. First, a maneuver planner using POMDPs for anticipatory decisions was introduced. The probabilistic modeling automatically balances comfort and efficiency while maintaining safety. Furthermore, the successive trajectory planner also considers the accumulated and expected risk using RMOT beside constraints from the maneuver planer. Finally, the safe application of machine learning for trajectory planning was shown. By combining the advantages of learning-based and model-based methods for a sampling-based algorithm, hand-tuned heuristics for initializing the sampling-based trajectory planner are made obsolete. At the same time, the neural network output is verified by the model-based components.

6 Conclusion

6.1 Summary

The entire chain of effects that leads to foresighted planning for automated vehicles was examined in this dissertation. Starting with the analysis of knowledge gaps in the situational awareness, to the closing or quantification of these gaps, to the planning of safe, situation- and risk-aware maneuvers and trajectories. The research questions posed at the beginning were answered in the corresponding order in chapters 3, 4, and 5.

First, the components for automated driving and the situations that occur were systematically analyzed. The uncertainties and gaps were identified where incomplete knowledge of the situation leads to decisions without sufficient situational awareness. For this, the single components of the automated driving functions were examined in detail (Sec. 3.2). The thesis focused on uncertainties that cannot be measured directly. Intentions and occluded areas were identified as particularly relevant. Here, it is necessary to draw conclusions from the existing context knowledge. An environment model was developed that contains all the necessary information to obtain a comprehensive situational awareness (Sec. 3.3), i.e., map information, uncertainty-aware information about visible objects, and occluded areas. The state of the environment is expressed based on the environment model. The state includes a measurable part that is subject to uncertainty, such as position uncertainties, and a partially observable part, such as occlusions and the intentions of road users. This results in a belief about the current state of the environment. The belief reflects what information is available and how confident the ego vehicle is about its accuracy.

In the following, this belief is concretized. A novel method for estimating intentions (Sec. 4.1), a dual approach for tracking occluded areas (Sec. 4.2), an approach for inferring the existence of road users in precisely these occluded areas (Sec. 4.3), and a model-based (Sec. 4.1.3) as well as a learned prediction model (Sec. 4.4) are presented. The prediction modules provide multi-modal predictions for the propagation of the belief about the state of the environment. The combination of the presented components leads to a precise understanding of the current situation. Incomplete knowledge about intentions and occluded areas have been made describable and are included in the belief. The main contributions in this chapter are a novel approach to infer information by generating and assessing multiple potential developments from past points in time, a dual approach to track occlusions spatially and temporally based on different movement patterns of objects, and a concept for unsupervised training of CNNs for situations where no ground truth data is available because of occlusions.

The last chapter showed how anticipatory driving can be achieved based on the enriched belief. The foresighted and anticipatory planning comprises a probabilistic maneuver planner based on POMDPs (Sec. 5.1) and a risk-aware trajectory planner (Sec. 5.2). Additionally, the safe application of machine learning methods for trajectory planning is shown (Sec. 5.3). The contributions here are the application of a generic FOV in the probabilistic maneuver planner for assumptions about future observations for more foresighted maneuvers. For the risk-aware trajectory planner, a representation was introduced that accumulates arbitrary risks from different sources in a series of two-dimensional grids, so-called RMOT.

Finally, the first approach to combine the strengths of machine learning-based and rule-based approaches for trajectory planning in dynamic environments was presented. A sampling-based trajectory planner is optimized by providing learned initial samples that represent trajectories to overcome hand-crafted heuristics.

The concepts were evaluated using appropriate scenarios. The evaluation was carried out in the simulation and with multiple test vehicles of the FZI and vehicles of project partners to demonstrate the added value of the methods described. During the work, new questions arose that shed further light on individual aspects but were not the focus of this thesis. These are explained in the following section.

6.2 Outlook

The FOV used in this work was assumed to be two-dimensional. While this is sufficient for most situations, a three-dimensional representation of the FOV can bring further improvements concerning the accuracy of the occluded areas and assumptions about hidden objects as well as visible objects. It was not considered in this thesis because of the imposed runtime complexity.

The thesis showed the successful application of machine learning for safe trajectory planning. The next step would be to also leverage the advantages of machine learning for maneuver planning. In [10], a first push in this direction was made by tackling the generalization of machine learning approaches. With the invariant environment representation, the reinforcement learning agent could learn how to cross occluded intersections safely. However, as shown in this work in Sec. 5.3, the output of neural networks must be verified for safe driving. This not only applies to learned maneuver decisions but also to learned prediction approaches. A model-based checker does not verify the network output in the approach presented in Sec. 4.4. Since the work in Sec. 4.4 was conducted, the insights flowed into other research like [20]. Including knowledge priors into the prediction ensures the validity of learned predictions with respect to the vehicles' kinematic constraints and the road's boundaries. In general, the optimal combination of machine learning, model-based algorithms, and expert knowledge is still a highly relevant research subject.

Another open research question regarding environment prediction is the learned prediction of different developments under the definition in Sec. 2.1.2. While current approaches generate multi-modal predictions for every agent, the predicted trajectories of the agents are not interdependent. However, it is necessary in behavior planning to know how other agents react to each other's behavior. Additionally, a graph-based prediction with CNNs to encode occlusions as suggested in the discussion in Sec. 4.4.6 could be researched for an entity-level prediction that also considers occluded areas.

Finally, while this thesis laid out the tools for situation-aware and risk-aware driving, the optimal parametrization and the maximum tolerated risk need to be determined in future works, e.g., using user studies or inverse reinforcement learning based on human-driven trajectories.

List of Figures

List of Tables

List of Publications

Publications by the author

This list includes all publications where the author of this thesis is either the main author or has contributed to the publication.

[1] Philip Schörner, Lars Töttel, Jens Doll, and J. Marius Zöllner. Predictive trajectory planning in situations with hidden road users using partially observable markov decision processes. In *IEEE Intelligent Vehicles Symposium (IV)*, 2019. ***Best paper award***.

[2] Philip Schörner, Christian Hubschneider, Jonathan Härtl, Rupert Polley, and J. Marius Zöllner. Grid-based micro traffic prediction using fully convolutional networks. In *IEEE International Intelligent Transportation Systems Conference (ITSC)*, 2019.

[3] Philip Schörner, Jens Doll, Maximilian Galm, and J. Marius Zöllner. Trajectory planning for a pseudo omnidirectional vehicle using particle swarm optimization. In *IEEE International Intelligent Transportation Systems Conference (ITSC)*, 2019.

[4] Philip Schörner, Mark Timon Hüneberg, and J. Marius Zöllner. Optimization of sampling-based motion planning in dynamic environments using neural networks. In *IEEE Intelligent Vehicles Symposium (IV)*, 2020.

[5] Philip Schörner, Tobias Fleck, and J. Marius Zöllner. What if? behavior estimation by predicting traffic scenes from state histories. In *IEEE International Intelligent Transportation Systems Conference (ITSC)*, 2021.

[6] Philip Schörner, Marcus Conzelmann, Tobias Fleck, Marc Zofka, and J. Marius Zöllner. Park my car! automated valet parking with different vehicle automation levels by v2x connected smart infrastructure. In *IEEE International Intelligent Transportation Systems Conference (ITSC)*, 2021.

[7] Philip Schörner, Daniel Grimm, and J. Marius Zöllner. Towards multi modal risk assessment. In *IEEE International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*, 2022.

[8] Fabian Oboril, Philip Schörner, Luka Sachße, Dominik Nees, Tolgahan Percin, Cornelius Bürkle, Bernhard Zeifang, Helen Gremmelmaier, Sven Ochs, Bernd Gassmann, Fabian Gottselig, Patrick Henkel, Tobias Schulz, Philipp Kautzmann, Michael Frey, Kay-Ulrich Scholl, Marius Zöllner, and Thomas Münsterer. Safeadarchitect: End-to-end architecture for safe and risk-aware automated driving in urban environments. In *IEEE International Intelligent Transportation Systems Conference (ITSC)*, 2024.

[9] Daniel Grimm, Philip Schörner, Moritz Dreßler, and J. Marius Zöllner. Holistic graph-based motion prediction. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[10] Karl Kurzer, Philip Schörner, Alexander Albers, Hauke Thomsen, Karam Daaboul, and J. Marius Zöllner. Generalizing decision making for automated driving with an invariant environment representation using deep reinforcement learning. In *IEEE Intelligent Vehicles Symposium (IV)*, 2021.

[11] Sven Ochs, Philip Schörner, Marc René Zofka, and J. Marius Zöllner. Lidar-slam using semantic information - how to deal with dynamic objects? In *IEEE International Conference on Ubiquitous Robots*, 2023.

[12] Sven Ochs, Philip Schörner, Marc René Zofka, Percin Tolgahan, and J. Marius Zöllner. What can we learn from virtual sensor models for self localization and mapping for autonomous mobile systems? In *IEEE International Intelligent Transportation Systems Conference (ITSC)*, 2023.

[13] Sven Ochs, Daniel Grimm, Jens Doll, Marc Heinrich, Stefan Orf, Tobias Fleck, Dennis Nienhüser, Miriam Schreiber, Artur Koch, Thomas Schamm, Ralf Kohlhaas, Steffen Knoop, Peter Biber, Dirk Fratzke, Jakob Kammerer, Ravi Shekhar Jethani, Christian Bäuerlein, Florian Kuhnt, Philip Schörner, Marc René Zofka, and J. Marius Zöllner. "Stepping Ahead with Electrified, Connected and Automated Shuttles in the Test Area Autonomous Driving BW". *Authorea Preprints*, 2023.

[14] Sven Ochs, Jens Doll, Daniel Grimm, Tobias Fleck, Marc Heinrich, Stefan Orf, Albert Schotschneider, Helen Gremmelmaier, Rupert Polley, Svetlana Pavlitska, Maximilian Zipfl, Helen Schneider, Ferdinand Mütsch, Daniel Bogdoll, Florian Kuhnt, Philip Schörner, Marc René Zofka, and J. Marius Zöllner. One stack to rule them all: To drive automated vehicles, and reach for the 4th level. *arXiv preprint arXiv:2404.02645*, 2024.

[15] Sven Ochs, Jens Doll, Marc Heinrich, Philip Schörner, Sebastian Klemm, Marc René Zofka, and J. Marius Zöllner. Leveraging swarm intelligence to drive autonomously: A particle swarm optimization based approach to motion planning. *arXiv preprint arXiv:2404.02644*, 2024.

[16] Marc Heinrich, Maximilian Zipfl, Marc Uecker, Sven Ochs, Martin Gontscharow, Tobias Fleck, Jens Doll, Philip Schörner, Christian Hubschneider, Marc René Zofka, Alexander Viehl, and J. Marius Zöllner. Cocar nextgen: a multi-purpose platform for connected autonomous driving research. In *IEEE International Intelligent Transportation Systems Conference (ITSC)*, 2024.

[17] Tobias Fleck, Karam Daaboul, Michael Weber, Philip Schörner, Marek Wehmer, Jens Doll, Stefan Orf, Nico Sußmann, Christian Hubschneider, Marc René Zofka, et al. Towards large scale urban traffic reference data: Smart infrastructure in the test area autonomous driving baden-württemberg. In *International Conference on Intelligent Autonomous Systems (IAS)*. Springer, 2018.

[18] Tobias Fleck, Lennart Jauernig, Rupert Polley, Philip Schörner, Marc René Zofka, and J. Marius Zöllner. Infra2go: A mobile development platform for connected, cooperative and autonomous driving. In *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2022.

[19] Marc René Zofka, Tobias Fleck, Philip Schörner, Marc Heinrich, and J. Marius Zöllner. Paving the ways for vehicles-in-the-loop: Wireless integration of autonomous vehicles into mixed reality proving grounds and test areas. In *IEEE International Intelligent Transportation Systems Conference (ITSC)*, 2023.

[20] Abhishek Vivekanandan, Ahmed Aboulazm, Philip Schörner, and J. Marius Zöllner. Ki-pmf: Knowledge integrated plausible motion forecasting. In *IEEE Intelligent Vehicles Symposium (IV) Workshop*, 2024.

Student works supervised by the author

This list includes student works that have been proposed and supervised by the author of this dissertation as part of his research.

[21] Ahmed Agha. Adapting the Trajectory Planning for an Omnidirectional Vehicle to Optimize the Sensor Usage Efficiency. Bachelors thesis, KIT Karlsruher Institut für Technologie, Karlsruhe, Germany, 2018.

[22] Sebastian Andrade Max. Dealing with Risks and Uncertainties in Autonomous Driving. Seminar, KIT Karlsruher Institut für Technologie, Karlsruhe, Germany, 2020.

[23] David Funke. Applied POMDP Planning in Partially Observable Environments. Bachelors thesis, KIT Karlsruher Institut für Technologie, Karlsruhe, Germany, 2020.

[24] Maximilian Galm. Trajektorienplanung für ein omnidirektionales Fahrzeug mittels Partikelschwarmoptimierung. Masters thesis, KIT Karlsruher Institut für Technologie, Karlsruhe, Germany, 2018.

[25] Mark Hüneberg. Optimierung Sampling-basierter Bewegungsplanung in dynamischen Umgebungen mittels CNNs. Masters thesis, KIT Karlsruher Institut für Technologie, Karlsruhe, Germany, 2019.

[26] Dániel Kiss-Illés. Predictive planning by partially observable markov decision processes using an area-based state space model. Masters thesis, KIT Karlsruher Institut für Technologie, Karlsruhe, Germany, 2020.

[27] Gloriya Miteva. Generalization Capability Estimation of Existing Methods for Predicting Surrounding Road Users' Intentions for an Autonomous Vehicle: A Survey. Seminar, KIT Karlsruher Institut für Technologie, Karlsruhe, Germany, 2022.

[28] Jonas Paczia. Prognostication of non-visible road users from the situational context using deep neural networks. Masters thesis, KIT Karlsruher Institut für Technologie, Karlsruhe, Germany, 2020.

[29] Rupert Polley. Prädiktion von Verkehrsteilnehmern bei partieller Sichtbarkeit mittels tiefer neuronaler Netze. Bachelors thesis, KIT Karlsruher Institut für Technologie, Karlsruhe, Germany, 2018.

[30] Florian Stelzer. Entwicklung von Konzepten und Methoden für intelligente Parkhausmanagementsysteme - Development of Concepts and Methods for intelligent Car Park Management Systems. Masters thesis, KIT Karlsruher Institut für Technologie, Karlsruhe, Germany, 2019.

[31] Lars Töttel. Predictive Trajectory Planning in Situations with Occluded Road Users Using POMDPs. Masters thesis, KIT Karlsruher Institut für Technologie, Karlsruhe, Germany, 2018.

Bibliography

[32] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[33] Charu C Aggarwal et al. *Neural networks and deep learning*, volume 10. Springer, 2018.

[34] Hamid Al-jameel. Examining and improving the limitations of gazis–herman–rothery car following model. 2009.

[35] Seifemichael B. Amsalu, Abdollah Homaifar, Ali Karimoddini, and Arda Kurt. Driver intention estimation via discrete hidden Markov model. *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 2017.

[36] Aptiv, Audi, Baidu, BMW, Continental, Daimler, FCA, Here, Infineon, Intel, and Volkswagen. Safety First for Automated Driving. Technical report, 2019. [Whitepaper, available online https://group.mercedes-benz.com/documents/innovation/other/safety-first-for-automated-driving.pdf; accessed December 28th, 2023].

[37] K. J. Åström. Optimal control of Markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications*, 10(1):174–205, 1965.

[38] Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2):235–256, May 2002.

[39] Haoyu Bai, David Hsu, Wee Sun Lee, and Vien A Ngo. Monte carlo value iteration for continuous-state pomdps. In *Algorithmic foundations of robotics IX*. Springer, 2010.

[40] Mayank Bansal, Alex Krizhevsky, and Abhijit S. Ogale. Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. *CoRR*, abs/1812.03079, 2018.

[41] Holger Banzhaf, Maxim Dolgov, Jan Stellet, and J. Marius Zollner. From Footprints to Beliefprints: Motion Planning under Uncertainty for Maneuvering Automated Vehicles in Dense Scenarios. *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2018.

[42] Holger Banzhaf, Paul Sanzenbacher, Ulrich Baumann, and J Marius Zöllner. Learning to predict ego-vehicle poses for sampling-based nonholonomic motion planning. *IEEE Robotics and Automation Letters*, 4(2):1053–1060, 2019.

[43] Leonard E Baum, Ted Petrie, George Soules, and Norman Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *The annals of mathematical statistics*, 41(1), 1970.

[44] Richard Bellman. A Markovian decision process, 1957.

[45] Philipp Bender, Julius Ziegler, and Christoph Stiller. Lanelets: Efficient map representation for autonomous driving. In *IEEE Intelligent Vehicles Symposium (IV)*, 2014.

[46] Julian Bock, Robert Krajewski, Tobias Moers, Steffen Runde, Lennart Vater, and Lutz Eckstein. The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections. *preprint arXiv:1911.07602*, 2019.

[47] Maxime Bouton, Alireza Nakhaei, Kikuo Fujimura, and Mykel J. Kochenderfer. Scalable decision making with sensor occlusions for autonomous driving. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[48] S. Brechtel, T. Gindele, and R. Dillmann. Probabilistic decision-making under uncertainty for autonomous driving using continuous pomdps. In *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2014.

[49] Jack E Bresenham. Algorithm for computer control of a digital plotter. *IBM Systems journal*, 4(1):25–30, 1965.

[50] Luca Caltagirone, Mauro Bellone, Lennart Svensson, and Mattias Wahde. Simultaneous perception and path generation using fully convolutional neural networks. *CoRR*, abs/1703.08987, 2017.

[51] Christophe Coué, Cédric Pradalier, Christian Laugier, Thierry Fraichard, and Pierre Bessière. Bayesian occupancy filtering for multitarget tracking: an automotive application. *The International Journal of Robotics Research*, 25(1), 2006.

[52] Henggang Cui, Vladan Radosavljevic, Fang-Chieh Chou, Tsung-Han Lin, Thi Nguyen, Tzu-Kuo Huang, Jeff Schneider, and Nemanja Djuric. Multimodal trajectory predictions for autonomous driving using deep convolutional networks. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[53] Sean Curtis, Andrew Best, and Dinesh Manocha. Menge: A modular framework for simulating crowd movement. *Collective Dynamics*, 1, 2016.

[54] John Dahl, Gabriel Rodrigues de Campos, Claes Olsson, and Jonas Fredriksson. Collision avoidance: A literature review on threat-assessment techniques. *IEEE Transactions on Intelligent Vehicles*, 4(1), 2018.

[55] Florian Damerow, Tim Puphal, Yuda Li, and Julian Eggert. Risk-based driver assistance for approaching intersections of limited visibility. *IEEE International Conference on Vehicular Electronics and Safety (ICVES)*, 2017.

[56] Julie Dequaire, Dushyant Rao, Peter Ondruska, Dominic Zeng Wang, and Ingmar Posner. Deep tracking on the move: Learning to track the world from a moving vehicle using recurrent neural networks. *CoRR*, abs/1609.09365, 2016.

[57] Nemanja Djuric, Vladan Radosavljevic, Henggang Cui, Thi Nguyen, Fang-Chieh Chou, Tsung-Han Lin, and Jeff Schneider. Short-term motion prediction of traffic actors for autonomous driving using deep convolutional networks. *arXiv preprint arXiv:1808.05819*, 1(2):6, 2018.

[58] Craig R Fox and Gülden Ülkümen. Distinguishing two dimensions of uncertainty. *Fox, Craig R. and Gülden Ülkümen (2011),"Distinguishing Two Dimensions of Uncertainty," in Essays in Judgment and Decision Making, Brun, W., Kirkebøen, G. and Montgomery, H., eds. Oslo: Universitetsforlaget*, 2011.

[59] Yarin Gal and Zoubin Ghahramani. Bayesian convolutional neural networks with bernoulli approximate variational inference. *arXiv 1506.02158*, 2015.

[60] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Insights and applications. In *Deep Learning Workshop, ICML*, volume 1, 2015.

[61] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, 2016.

[62] Enric Galceran, Edwin Olson, and Ryan M Eustice. Augmented vehicle tracking under occlusions for decision-making in autonomous driving. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015.

[63] Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. Vectornet: Encoding hd maps and agent dynamics from vectorized representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020.

[64] T. Gindele, S. Brechtel, and R. Dillmann. A probabilistic model for estimating driver behaviors and vehicle trajectories in traffic environments. In *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2010.

[65] Tobias Gindele, Sebastian Brechtel, Joachim Schroder, and Rudiger Dillmann. Bayesian occupancy grid filter for dynamic environments using prior map knowledge. In *IEEE Intelligent Vehicles Symposium (IV)*, 2009.

[66] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323, 2011.

[67] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. 2016.

[68] Daniel Grimm, Maximilian Zipfl, Felix Hertlein, Alexander Naumann, Juergen Luettin, Steffen Thoma, Stefan Schmid, Lavdim Halilaj, Achim Rettinger, and J Marius Zöllner. Heterogeneous graph-based trajectory prediction using local map context and social interactions. In *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2023.

[69] Michael Gygli, Mohammad Norouzi, and Anelia Angelova. Deep value networks learn to evaluate and iteratively refine structured outputs. *CoRR*, abs/1703.04363, 2017.

[70] Handelsblatt. Das vollkommen autonome Fahren wird vorerst nicht kommen. `https://www.handelsblatt.com/politik/deutschland/hohe-kosten-das-vollkommen-autonome-fahren-wird-vorerst-nicht-kommen/24597246.html?ticket=ST-3182964-bKdfGbHIbMZ0nWCLaaB6-ap5`, [Online; accessed July 22nd, 2019].

[71] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[72] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[73] Ruijie He, Emma Brunskill, and Nicholas Roy. Puma: Planning under uncertainty with macro-actions. In *Proceedings of the National Conference on Artificial Intelligence*, volume 2, 01 2010.

[74] Dirk Helbing and Péter Molnár. Social force model for pedestrian dynamics. *Phys. Rev. E*, 51, 1995.

[75] Stefan Hoermann, Martin Bach, and Klaus Dietmayer. Dynamic occupancy grid prediction for urban autonomous driving: A deep learning approach with fully automatic labeling. *CoRR*, abs/1705.08781, 2017.

[76] Stefan Hoermann, Felix Kunz, Dominik Nuss, Stephan Renter, and Klaus Dietmayer. Entering crossroads with blind corners. A safe strategy for autonomous vehicles. *IEEE Intelligent Vehicles Symposium (IV)*, 2017.

[77] Gabriel M Hoffmann, Claire J Tomlin, Michael Montemerlo, and Sebastian Thrun. Autonomous automobile trajectory tracking for off-road driving: Controller design, experimental validation and racing. In *American control conference*. IEEE, 2007.

[78] Constantin Hubmann, Marvin Becker, Daniel Althoff, David Lenz, and Christoph Stiller. Decision making for autonomous driving considering interaction and uncertain prediction of surrounding vehicles. *IEEE Intelligent Vehicles Symposium (IV)*, 2017.

[79] Christian Hubschneider, André Bauer, Michael Weber, and J Marius Zöllner. Adding navigation to the equation: Turning decisions for end-to-end vehicle control. In *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2017.

[80] Christian Hubschneider, André Bauer, Jens Doll, Michael Weber, Sebastian Klemm, Florian Kuhnt, and J. Marius Zöllner. Integrating end-to-end learned steering into probabilistic autonomous driving. In *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2017.

[81] Jonathan Härtl. Learned Micro-Scale Traffic Prediction using Neural Networks. Masters thesis, KIT Karlsruhe Institut of Technology, Karlsruhe, Germany, 2017.

[82] Brian Ichter, James Harrison, and Marco Pavone. Learning sampling distributions for robot motion planning. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[83] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *preprint arXiv:1502.03167*, 2015.

[84] Faris Janjoš, Maxim Dolgov, and J. Marius Zöllner. Starnet: Joint action-space prediction with star graphs and implicit global-frame self-attention. In *IEEE Intelligent Vehicles Symposium (IV)*, 2022.

[85] Weiwei Jiang and Jiayun Luo. Graph neural network for traffic forecasting: A survey. *prepring arXiv:2101.11174*, 2021.

[86] S. J. Julier and J. K. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 2004.

[87] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1-2):99–134, 1998.

[88] Kai Nagel and Michael Schreckenberg. A cellular automaton model for freeway traffic. *J. Phys. I France*, 2(12), 1992.

[89] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1), 1960.

[90] Kamal Kant and Steven W. Zucker. Toward Efficient Trajectory Planning: The Path-Velocity Decomposition. *The International Journal of Robotics Research*, 5(3):72–89, 1986.

[91] Ioannis Karamouzas, Peter Heil, Pascal Van Beek, and Mark H Overmars. A predictive collision avoidance model for pedestrian simulation. In *International workshop on motion in games*. Springer, 2009.

[92] Michael Katehakis and Veinott A F Jr. The multi- armed bandit problem: Decomposition and computation. *Math. Oper. Res.*, 12:262–268, 05 1987.

[93] James Kennedy. Particle swarm optimization. *Encyclopedia of machine learning*, pages 760–766, 2010.

[94] Arne Kesting, Martin Treiber, and Dirk Helbing. General lane-changing model mobil for car-following models. *Transportation Research Record*, 1999(1):86–94, 2007.

[95] Arne Kesting, Martin Treiber, and Dirk Helbing. Enhanced intelligent driver model to access the impact of driving strategies on traffic capacity. *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences*, 368(1928):4585–4605, 2010.

[96] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2017.

[97] Dimitri Klimenko, Joshua Song, and Hanna Kurniawati. TAPIR: A software Toolkit for approximating and adapting POMDP solutions online. *Australasian Conference on Robotics and Automation (ACRA)*, 02-04-Dece, 2014.

[98] Markus Koschi and Matthias Althoff. Set-Based Prediction of Traffic Participants Considering Occlusions and Traffic Rules. *IEEE Transactions on Intelligent Vehicles*, 6(2), 2021.

[99] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[100] Florian Kuhnt, Jens Schulz, Thomas Schamm, and J Marius Zöllner. Understanding interactions between traffic participants based on learned behaviors. In *IEEE Intelligent Vehicles Symposium (IV)*, 2016.

[101] P. Kumar, M. Perrollaz, S. Lefèvre, and C. Laugier. Learning-based approach for online lane change intention prediction. In *IEEE Intelligent Vehicles Symposium (IV)*, 2013.

[102] Hanna Kurniawati and Vinay Yadav. An online pomdp solver for uncertainty planning in dynamic environment. In *Robotics Research*, pages 611–629. Springer, 2016.

[103] Zhiqian Lan, Yuxuan Jiang, Yao Mu, Chen Chen, and Shengbo Eben Li. Sept: Towards efficient scene representation learning for motion prediction. In *The Twelfth International Conference on Learning Representations*, 2023.

[104] Steven M LaValle. Rapidly-exploring random trees: A new tool for path planning. 1998.

[105] S. Lefèvre, D. Vasquez, and C. Laugier. A survey on motion prediction and risk assessment for intelligent vehicles. *ROBOMECH Journal*, 1:1–14, 2014.

[106] S. Lefèvre, C. Laugier, and J. Ibañez-Guzmán. Risk assessment at road intersections: Comparing intention and expectation. In *IEEE Intelligent Vehicles Symposium (IV)*, 2012.

[107] Ming Liang, Bin Yang, Rui Hu, Yun Chen, Renjie Liao, Song Feng, and Raquel Urtasun. Learning lane graph representations for motion forecasting. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*. Springer, 2020.

[108] W. Liu, S. Kim, S. Pendleton, and M. H. Ang. Situation-aware decision making for autonomous driving on urban road using online pomdp. In *IEEE Intelligent Vehicles Symposium (IV)*, 2015.

[109] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[110] Xiaolei Ma, Zhuang Dai, Zhengbing He, Jihui Ma, Yong Wang, and Yunpeng Wang. Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction. *Sensors*, 17(4), 2017.

[111] Omid Madani, Steve Hanks, and Anne Condon. On the undecidability of probabilistic planning and infinite-horizon partially observable markov decision problems. In *AAAI/IAAI*, pages 541–548, 1999.

[112] A.A. Markov. *Theory of Algorithms*. TT 60-51085. Academy of Sciences of the USSR, 1954.

[113] Mercedes-Benz Group. Certification for SAE Level 3 system for U.S. market, 2023. [Online; accessed January 16th, 2024].

[114] Xiaoyu Mo, Yang Xing, and Chen Lv. Graph and recurrent neural network-based vehicle trajectory prediction for highway driving. In *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2021.

[115] Johannes Müller, Jan Strohbeck, Martin Herrmann, and Michael Buchholz. Motion planning for connected automated vehicles at occluded intersections with infrastructure sensors. *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[116] Yannik Nager, Andrea Censi, and Emilio Frazzoli. What lies in the shadows? Safe and computation-aware motion planning for autonomous vehicles using intent-aware dynamic shadow regions. *IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[117] Amaury Negre, Lukas Rummelhard, and Christian Laugier. Hybrid sampling Bayesian Occupancy Filter. In *IEEE Intelligent Vehicles Symposium (IV)*, 2014.

[118] David Nistér, Hon-Leung Lee, Julia Ng, and Yizhou Wang. The safety force field. *NVIDIA White Paper*, 2019.

[119] Fabian Oboril and Kay-Ulrich Scholl. Risk-aware safety layer for av behavior planning. In *IEEE Intelligent Vehicles Symposium (IV)*, 2020.

[120] Christos H. Papadimitriou and John N. Tsitsiklis. The complexity of markov decision processes. *Mathematics of Operations Research*, 12(3):441–450, 1987.

[121] D. Petrich, D. Azarfar, F. Kuhnt, and J. M. Zöllner. The fingerprint of a traffic situation: A semantic relationship tensor for situation description and awareness. In *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2018.

[122] Ingmar Posner and Peter Ondruska. Deep Tracking: Seeing Beyond Seeing Using Recurrent Neural Networks. *Proceedings of the 30th Conference on Artificial Intelligence (AAAI 2016)*, 2016.

[123] Noé Pérez Higueras, Fernando Caballero, and Luis Merino. Learning human-aware path planning with fully convolutional networks. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[124] Ahmed H Qureshi and Michael C Yip. Deeply informed neural sampling for robot motion planning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[125] Lawrence Rabiner and Biinghwang Juang. An introduction to hidden markov models. *IEEE ASSP Magazine*, 3(1), 1986.

[126] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. In *Computer Vision and Pattern Recognition*, 2016.

[127] X. Rong Li and V. P. Jilkov. Survey of maneuvering target tracking. part i. dynamic models. *IEEE Transactions on Aerospace and Electronic Systems*, 39(4), 2003.

[128] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image Computing and Computer-assisted Intervention*. Springer, 2015.

[129] Stéphane Ross, Joëlle Pineau, Sébastien Paquet, and Brahim Chaib-draa. Online planning algorithms for POMDPs. *Journal of Artificial Intelligence Research*, 32:663–704, 2008.

[130] Scott D Roth. Ray casting for modeling solids. *Computer Graphics and Image Processing*, 18(2):109–144, 1982.

[131] Marcelo Saval-Calvo, Luis Medina-Valdés, José María Castillo-Secilla, Sergio Cuenca-Asensi, Antonio Martínez-Álvarez, and Jorge Villagrá. A review of the bayesian occupancy filter. *Sensors (Switzerland)*, 2017.

[132] Matthias Schreier. *Bayesian Environment Representation, Prediction, and Criticality Assessment for Driver Assistance Systems*. PhD thesis, Technischen Universität Darmstadt, Darmstadt, 2016.

[133] R. Schubert, E. Richter, and G. Wanielik. Comparison and evaluation of advanced motion models for vehicle tracking. In *11th International Conference on Information Fusion*, 2008.

[134] J. Schulz, C. Hubmann, J. Löchner, and D. Burschka. Interaction-aware probabilistic behavior prediction in urban environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[135] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. On a formal model of safe and scalable self-driving cars. *arXiv preprint arXiv:1708.06374*, 2017.

[136] David Silver and Joel Veness. Monte-carlo planning in large pomdps. In *Advances in neural information processing systems*, pages 2164–2172, 2010.

[137] Richard D. Smallwood and Edward J. Sondik. The optimal control of partially observable markov processes over a finite horizon. *Operations Research*, 21:1071–1088, 1973.

[138] Statistisches Bundesamt Deutschland (Destatis). Traffic accidents - cause of accidents - driver mistakes and improper behaviour of pedestrians. `www.destatis.de`. [FactsFigures - Economic-Sectors - TransportTraffic - TrafficAccidents - Tables, Online; accessed January 28th, 2019].

[139] Jan Erik Stellet. Statistical modelling of algorithms for signal processing in systems based on environment perception. 2016.

[140] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[141] José Manuel Gaspar Sánchez, Truls Nyberg, Christian Pek, Jana Tumova, and Martin Törngren. Foresee the unseen: Sequential reasoning about hidden obstacles for safe driving. In *IEEE Intelligent Vehicles Symposium (IV)*, 2022.

[142] Eijiro Takeuchi, Yuki Yoshihara, and Ninomiya Yoshiki. Blind Area Traffic Prediction Using High Definition Maps and LiDAR for Safe Driving Assist. *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2015.

[143] Ömer Şahin Taş and Christoph Stiller. Limited visibility and uncertainty aware motion planning for automated driving. In *IEEE Intelligent Vehicles Symposium (IV)*, 2018.

[144] Q. Tran and J. Firl. Online maneuver recognition and multimodal trajectory prediction for intersection assistance using non-parametric regression. In *IEEE Intelligent Vehicles Symposium (IV)*, 2014.

[145] Martin Treiber, Ansgar Hennecke, and Dirk Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical review E*, 62(2):1805, 2000.

[146] Jur van den Berg, Stephen J. Guy, Ming C. Lin, and Dinesh Manocha. Reciprocal n-body collision avoidance. In *International Symposium on Robotics Research*, 2009.

[147] Sean Vaskov, Hannah Larson, Shreyas Kousik, Matthew Johnson-Roberson, and Ram Vasudevan. Not-at-fault driving in traffic: A reachability-based approach. In *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2019.

[148] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks, 2018.

[149] Lingguang Wang, Christoph Burger, and Christoph Stiller. Reasoning about Potential Hidden Traffic Participants by Tracking Occluded Areas. In *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2021.

[150] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. Fcos3d: Fully convolutional one-stage monocular 3d object detection, 2021.

[151] Xiao Wang, Deyu Bo, Chuan Shi, Shaohua Fan, Yanfang Ye, and Philip S. Yu. A survey on heterogeneous graph embedding: Methods, techniques, applications and sources. *IEEE Transactions on Big Data*, pages 1–1, 2022.

[152] Waymo. Waymo is opening its fully driverless service to the general public in Phoenix, 2020. [Online; accessed August 19th, 2022].

[153] J. H. Yang, D. Jung Kim, T. W. Kang, J. Sik Kim, and C. C. Chung. Decision of driver intention of a surrounding vehicle using hidden markov model with optimizing parameter estimation. In *20th International Conference on Control, Automation and Systems (ICCAS)*, 2020.

[154] Nan Ye, Adhiraj Somani, David Hsu, and Wee Sun Lee. DESPOT: Online POMDP planning with regularization. *Journal of Artificial Intelligence Research*, 58:231–266, 2013.

[155] Ming-Yuan Yu, Ram Vasudevan, and Matthew Johnson-Roberson. Risk assessment and planning with bidirectional reachability for autonomous driving. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[156] Ye Yuan, Xinshuo Weng, Yanglan Ou, and Kris M Kitani. Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.