

Technische Universität München
TUM School of Computation, Information and Technology

Efficient and Anticipatory Behavior Planning under Uncertainty for Autonomous Driving in Urban Environments

Chi Zhang, M.Sc.

Vollständiger Abdruck der von der TUM School of Computation, Information and Technology der Technischen Universität München zur Erlangung eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitz: Prof. Dr. Sebastian Steinhorst

Prüfende der Dissertation:

1. Prof. Dr.-Ing. habil. Alois C. Knoll
2. Prof. Dr. Guang Chen

Die Dissertation wurde am 21. Dezember 2023 bei der Technischen Universität München eingereicht und durch die TUM School of Computation, Information and Technology am 02. September 2024 angenommen.

Abstract

Ensuring safe, rule-compliant, and anticipatory driving behavior is crucial for the widespread deployment and social acceptance of autonomous vehicles. However, planning driving behaviors for autonomous vehicles remains challenging due to the non-deterministic nature of the real world. Planning algorithms must account for uncertain information arising from noisy sensor data, occlusions, inaccurate motion predictions, and behavior estimations of other road users. Deterministic behavior planning approaches often address uncertainties by increasing safety margins, resulting in overly cautious driving behaviors. In contrast, probabilistic approaches balance risk and efficiency by accounting for diverse sources of uncertain information. Nevertheless, they struggle in dense urban environments due to their limited handling of specific occlusion scenarios, decreased performance in crowded areas, and insufficient guarantees for safety and rule compliance. This thesis proposes a novel probabilistic behavior planning algorithm for generating safe and rule-compliant driving behaviors in occluded, dense urban environments.

First, this thesis introduces a behavior planning algorithm based on a Partially Observable Markov Decision Process (POMDP), capable of handling arbitrary occluded scenarios by utilizing onboard sensor information. The algorithm also considers potentially occluded road users found on roads, crosswalks, or at bus stops by introducing phantom road users. This allows the planner to generate less conservative driving behaviors in a wider range of different traffic scenarios.

Second, this thesis incorporates Vehicle-to-Everything (V2X) communication to enhance the presented algorithm's ability to handle occlusions. The concept leverages the perception data from onboard sensors and V2X communications independently, eliminating the need to fuse them. To avoid using potentially unreliable V2X information, a communication module is introduced to select V2X messages based on safety-relevant criteria.

Next, applying the POMDP behavior planning algorithm in the real world requires significant computational effort. To accelerate the search for an optimal driving behavior, a Multi-step Occupancy Grid Map (MOGM) is combined with the POMDP formulation, condensing the state space and reducing the computation effort for collision checking.

Finally, to consider safety constraints and traffic rule constraints in the behavior planning algorithm, this thesis proposes a safe and rule-compliant framework with a safety checker and a traffic rule monitor. The planned driving behaviors are verified using the Responsibility Sensitive Safety (RSS) model, while the traffic rule monitor evaluates satisfaction of the right-before-left rule.

The presented algorithms are evaluated in simulations involving various challenging scenarios, such as occluded bus stops, crosswalks, intersections, and crowded areas. The experiments show that the presented approach utilizing onboard sensor information and V2X communication provides less conservative driving behaviors and is applicable to a greater range of occlusion scenarios when compared to state-of-the-art approaches. Furthermore, the combination of POMDP with MOGM enhances computational efficiency by a factor of ten. Finally, it is shown that the safety checker and traffic rule monitor allow the behavior planning algorithm to generate safe and rule-compliant driving behavior.

Zusammenfassung

Für den öffentlichen Einsatz von autonomen Fahrzeugen ist es essentiell ein sicheres, regelkonformes und vorausschauendes Fahrverhalten zu gewährleisten. Aufgrund der nicht-deterministischen Natur der realen Welt ist die Planung von Fahrverhalten für autonome Fahrzeuge eine Herausforderung. Ein Planungsalgorithmus muss mit Unsicherheiten behaftete Informationen berücksichtigen, die aus verrauschten Sensordaten, Verdeckungen, ungenauen Bewegungsvorhersagen und Verhaltenseinschätzungen anderer Verkehrsteilnehmer resultieren. Deterministische Ansätze zur Verhaltensplanung begegnen Unsicherheiten oft durch eine Vergrößerung von Sicherheitsspannen, was zu einem übermäßig vorsichtigem Fahrverhalten führen kann. Im Gegensatz dazu ermöglichen probabilistische Ansätze ein Gleichgewicht zwischen Risiken und Effizienz, indem sie diverse mit Unsicherheiten behaftete Informationen berücksichtigen. Dennoch haben sie in dichten städtischen Umgebungen Probleme, da sie nur begrenzt mit bestimmten Verdeckungsszenarien umgehen können, in überfüllten Gebieten weniger leistungsfähig sind und keine ausreichenden Garantien für die Sicherheit und die Einhaltung von Regeln bieten. In dieser Arbeit wird ein neuartiger probabilistischer Verhaltensplanungsalgorithmus zur Generierung sicheren und regelkonformen Fahrverhaltens in verdeckten, dichten urbanen Umgebungen vorgeschlagen.

Zunächst wird ein Verhaltensplanungsalgorithmus auf Basis eines Partially Observable Markov Decision Process (POMDP) vorgestellt, der in der Lage ist, beliebige Szenarien mit Verdeckungen unter Verwendung von bordeigener Sensorinformationen zu handhaben. Zudem werden potenziell verdeckte Verkehrsteilnehmer auf Straßen, Zebrastreifen oder an Bushaltestellen berücksichtigt, indem Phantom-Verkehrsteilnehmer eingeführt werden. Die Verwendung dieser Phantom-Verkehrsteilnehmer ermöglicht es dem Planer, ein weniger konservatives Fahrverhalten in einem breiteren Spektrum unterschiedlicher Verkehrsszenarien zu generieren.

Des Weiteren wird die Fähigkeit des vorgestellten Algorithmus mit Verdeckungen in Verkehrsszenarien umzugehen durch die Einbeziehung von Vehicle-to-Everything (V2X)-Kommunikation verbessert. Das Konzept nutzt die Umfeldinformationen von bordeigenen Sensoren und von V2X-Kommunikation unabhängig voneinander, so dass eine Datenfusion nicht mehr nötig ist. Um potenziell unzuverlässige V2X-Informationen zu vermeiden, wird ein Kommunikationsmodul eingeführt, das V2X-Nachrichten auf der Grundlage sicherheitsrelevanter Kriterien auswählt.

Darüber hinaus erfordert die Anwendung des POMDP-Verhaltensplanungsalgorithmus in der realen Welt erheblichen Rechenaufwand. Um die Suche nach einem optimalen Fahrverhalten zu beschleunigen, wird eine Multi-step Occupancy Grid Map (MOGM) mit der POMDP-Formulierung kombiniert, wodurch der Zustandsraum verdichtet und der Rechenaufwand für Kollisionsprüfungen reduziert wird.

Schließlich, um Sicherheitsbeschränkungen und Verkehrsregelbeschränkungen im Verhaltensplanungsalgorithmus zu berücksichtigen, schlägt diese Arbeit auch einen sicheren und regelkonformen Framework vor, der aus einem Safety Checker und einem Traffic Rule Monitor besteht. Die geplanten Fahrverhalten werden mit dem Responsibility Sensitive Safety (RSS)-Modell überprüft, und der Verkehrsregelmonitor bewertet die Erfüllung der Rechts-

vor-Links-Regel.

Die vorgestellten Algorithmen werden in Simulationen mit verschiedenen anspruchsvollen Szenarien wie verdeckten Bushaltestellen, Fußgängerüberwegen, Kreuzungen und belebten Gebieten evaluiert. Die Experimente zeigen, dass der vorgestellte Ansatz, der bordeigene Sensorinformationen und V2X-Kommunikation nutzt, ein weniger konservatives Fahrverhalten ermöglicht und im Vergleich zum Stand der Technik auf mehr Arten von Verdeckungsszenarien anwendbar ist. Darüber hinaus verbessert die Kombination von POMDP mit MOGM die Recheneffizienz um einen Faktor von zehn. Schließlich wird gezeigt, dass der Algorithmus zur Verhaltensplanung durch den Safety Checker und den Traffic Rule Monitor ein sicheres und regelkonformes Fahrverhalten generieren kann.

Contents

Abstract	iii
Zusammenfassung	v
1 Introduction	1
1.1 Motivation	1
1.1.1 Architecture of Autonomous Vehicles	1
1.1.2 Challenges for Behavior Planning Algorithms	3
1.2 Behavior Planning Approaches	5
1.3 Goals	6
1.4 Contributions	6
1.5 Publications	7
1.6 Outline	7
2 Background	11
2.1 Sequential Decision Making under Uncertainty	11
2.1.1 Environment and Agent	11
2.1.2 Uncertainties in Autonomous Driving	12
2.1.3 Markov Decision Process	14
2.1.4 Partially Observable Markov Decision Process	15
2.2 Solving Sequential Decision Making under Uncertainty	17
2.2.1 Solving Model using Offline Methods	17
2.2.2 Solving Model using Online Methods	19
2.2.3 Solving Model using Deep Reinforcement Learning	25
3 Behavior Planning under Spatial Occlusion with Onboard Sensors	27
3.1 Overview and Contributions	27
3.2 Related Work	30
3.3 Approach: Occlusion Handling with Onboard Sensors	31
3.3.1 Environment Representation	32
3.3.2 State and Observation Space	32
3.3.3 Action Space	34
3.3.4 Observation Model for Real Road Users	34
3.3.5 Observation Model for Phantom Road Users	35
3.3.6 Observation Model for Traffic Mirror	36
3.3.7 Transition Model	37
3.3.8 Reward Function	37
3.4 Experiments and Results	39
3.4.1 Evaluation of the Occlusion-aware POMDP Planner	39
3.4.2 Evaluation of the Traffic Mirror-Aware POMDP Planner	43
3.5 Summary	49

4	Behavior Planning under Spatial Occlusion with V2X Communication	51
4.1	Overview and Contributions	51
4.2	Related Work	54
4.3	Approach: Occlusion Handling With V2X Communication	54
4.3.1	Framework	55
4.3.2	State, Observation and Action Space	56
4.3.3	Observation Model of Real Objects	56
4.3.4	Onboard Occlusion Analysis	56
4.3.5	Observation Model of Phantom Objects	56
4.3.6	V2X Communication	58
4.3.7	Modification of Appearance Probability	62
4.3.8	Transition Model	62
4.3.9	Reward Model	63
4.4	Experiments and Results	64
4.4.1	Experiment Setup	64
4.4.2	Evaluation of the V2X-aware POMDP Planner	65
4.5	Summary	73
5	Efficient Behavior Planning in Dense Urban Environments	77
5.1	Overview and Contributions	77
5.2	Related Work	80
5.3	Approach: Efficient Environment Model Using Multi-step Occupancy Maps	81
5.3.1	Multi-step Occupancy Grid Maps	81
5.3.2	POMDP Behavior Planning	83
5.4	Experiments and Results	86
5.4.1	Experiment Setup	86
5.4.2	Evaluation of the Efficient POMDP Planner	87
5.5	Summary	90
6	Safe and Rule-Aware Behavior Planning	93
6.1	Overview and Contributions	93
6.2	Related Work	95
6.3	Approach: Incorporation of Safety and Traffic Rule Constraints	96
6.3.1	Framework	96
6.3.2	CARLA and SUMO Co-simulation	96
6.3.3	Traffic Rule Monitor	98
6.3.4	Safety Checker	99
6.3.5	DRL Model	99
6.4	Experiments and Results	101
6.4.1	Simulation Environment	101
6.4.2	Simulation Setup	101
6.4.3	Evaluation Setup	102
6.4.4	Evaluation of the Traffic Rule Monitor	102
6.4.5	Evaluation of the Safety Checker	104
6.5	Summary	107
7	Conclusion	109
7.1	Summary	109
7.2	Future Work	111
	Symbols	113

Acronyms	115
List of Figures	117
List of Tables	121
List of Algorithms	123
Bibliography	124

1

Introduction

1.1 Motivation

Autonomous vehicles may bring numerous social benefits that improve road safety and overall quality of life. According to the World Health Organization (WHO), road traffic collisions result in approximately 1.3 million fatalities each year [Org14]. The leading risk factors are speeding, driving under the influence of alcohol, and distracted driving, which result from human error and could be eliminated by autonomous vehicles. Autonomous vehicles also hold great promise for easing traffic congestion through their communication abilities. By communicating with other vehicles and infrastructures, autonomous vehicles could optimize routes, decrease congestion, and enhance transportation network efficiency without the need for traffic lights or stop signs in the future. Autonomous vehicles could also contribute towards environmental sustainability by using electric power or renewable energies that produce less carbon emissions; furthermore, through adopting efficient driving behaviors they help lower energy usage and emissions for an eco-friendlier future.

Recent years have seen advances in autonomous driving technology and many prototype vehicles are now allowed on roads. Nevertheless, most of these vehicles are supervised by test drivers, as complex urban scenarios, such as occluded and crowded city environments, are still challenging to navigate for autonomous vehicles. Therefore, this thesis aims to introduce novel behavior planning algorithms that generate safe and rule-compliant driving behaviors under such challenging scenarios.

1.1.1 Architecture of Autonomous Vehicles

Although the architecture and design of autonomous vehicles differ across research proposals and industry practices, they typically share a set of fundamental components. As shown in Fig. 1.1, the modular architecture for autonomous vehicles typically includes the following modules: perception and prediction, planning, and control. A planning module also can be divided into different levels, such as route planning, behavior planning, and trajectory planning.

The perception module acts as the sensory foundation of the vehicle and interprets raw data from onboard sensors, such as LiDAR, cameras, and radar. Its primary role is to detect and recognize essential elements within the environment, including obstacles, vehicles, pedestrians, road boundaries, etc. By continuously scanning its surroundings, this module

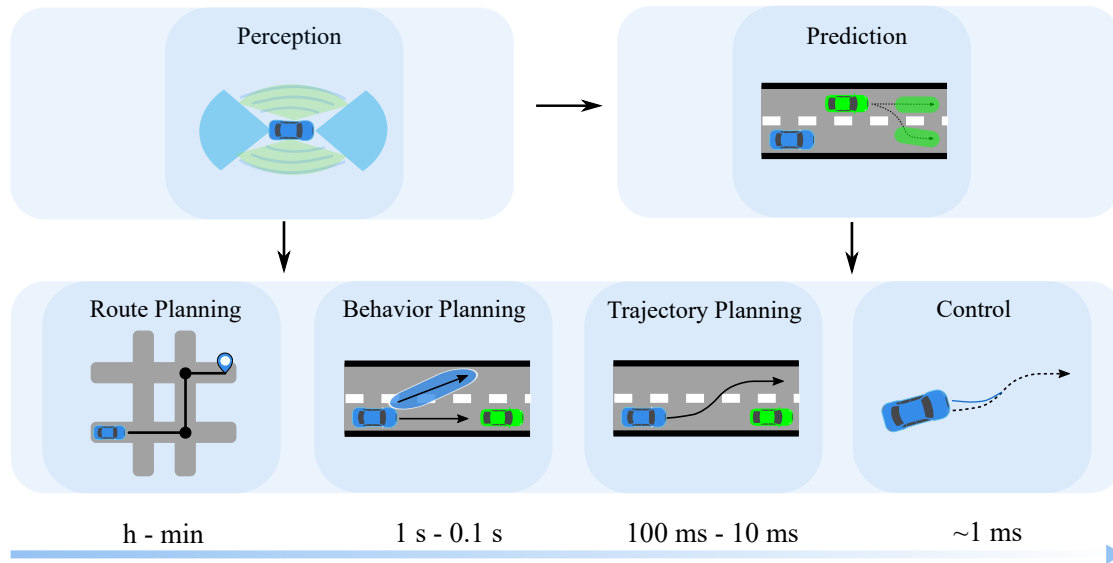


Figure 1.1: Diagram illustrating the modules of autonomous vehicles, from perception and prediction to planning and control, with respective time intervals for each step (Graphic adapted from [Bou20]).

generates a real-time representation of the environment which forms the input for subsequent decision-making processes in the autonomous vehicle.

Based on the data acquired from the perception stage, the prediction module is responsible for tracking nearby objects, forecasting their future movements and behaviors, and assessing associated risks. It utilizes history data and sensor inputs to predict behaviors and trajectories of vehicles, pedestrians, and other traffic participants. The prediction results are used in the planning module for supporting behavior and trajectory planning.

The route planning module ensures efficient navigation through complex road networks for autonomous vehicles. It determines an efficient route from the current location to the desired destination. The route planning integrates traffic data from various sources to account for congestion and traffic flow, adjusting the route accordingly. If unexpected changes occur during the journey, the route planning module can recalculate the route in real-time to adapt to new conditions.

The behavior planning module, also referred to as the decision-making module, determines the high-level actions that the vehicle should take based on its objectives as well as environmental information. It processes inputs from the perception and prediction module, which interprets the vehicle's surroundings, and then decides on actions such as lane changes, accelerations and decelerations, stopping for a pedestrian, yielding at an intersection, or overtaking other vehicles. The module plans safe driving behaviors by considering road conditions, traffic rules, and interactions with other vehicles, pedestrians, and cyclists.

The trajectory planning module takes the decisions from the behavior planner and converts them into trajectories the vehicle should follow. For instance, if the behavior planning module decides to change lanes, the trajectory planning module determines the precise trajectory the vehicle should take to execute this lane change smoothly and safely.

Once the trajectory planning module generates a desired trajectory, the control module is responsible for executing it. It sends commands to the vehicle's actuators, such as the throttle, brakes, and steering wheel, to ensure the vehicle follows the planned trajectory. For example, if the planned trajectory requires a certain turning radius at a specific speed, the control system actuates the steering wheel and adjusts the throttle or brakes to achieve this.

1.1.2 Challenges for Behavior Planning Algorithms

The complexity of urban scenarios for autonomous vehicles arises from the combination of dynamic traffic, diverse road users, unstructured environments, and the need to adhere to traffic regulations while ensuring safety and comfort for passengers and other road users. Meeting these challenges requires efficient and anticipatory behavior planning algorithms. The goal of this thesis is to develop novel behavior planning algorithms for addressing the following urban traffic situations.

Handling Spatial Occlusions

In complex urban environments, autonomous vehicles cannot fully observe their surroundings. This is due to both the limited detection range of their onboard sensors (such as cameras, radars, and LiDARs), and occlusion to the Field of View (FoV) caused by numerous dynamic road users as well as static obstacles like buildings, security fences, and dense vegetation. For autonomous vehicles, not considering situations with occlusions can pose safety risks. These scenarios require behavior planning algorithms to infer road users' potential positions and movements. An example is shown in Fig. 1.2 (A), in which the blue ego vehicle leaves the bus stop and a temporarily parked car occludes its FoV. Therefore, the ego vehicle cannot properly perceive the pedestrians in the occluded crosswalks.

Obeying Traffic Rules

A further challenge is to plan driving behaviors that obey traffic rules and consider interactions with other road users. Ensuring compliance with traffic rules is crucial for the safety of all road users and the social acceptance of autonomous vehicles. Fig. 1.2 (B) shows an intersection without traffic signs and traffic signals. The blue ego vehicle intends to drive through the intersection. The vehicles on the right-hand side have right of way over the ego vehicle. In order to make safe decisions, the ego vehicle has to understand the intentions of the other vehicles while adhering to the traffic rules so that other vehicles react to the ego vehicle properly.

Navigating Dense Traffic

Driving through dense urban environments is difficult for autonomous vehicles because they must reason about the unknown intentions of a large number of road users while dealing with a variety of uncertain information, such as sensor noise and inaccurate predictions. As shown in Fig. 1.2 (C), numerous cars and pedestrians surround the ego vehicle as it navigates through shopping areas. In such conditions, the autonomous vehicle must be capable of maneuvering through tight spaces and making real-time decisions. The behavior planning algorithm should have the ability to negotiate with others and efficiently adapt to sudden changes in the behavior of other road users.

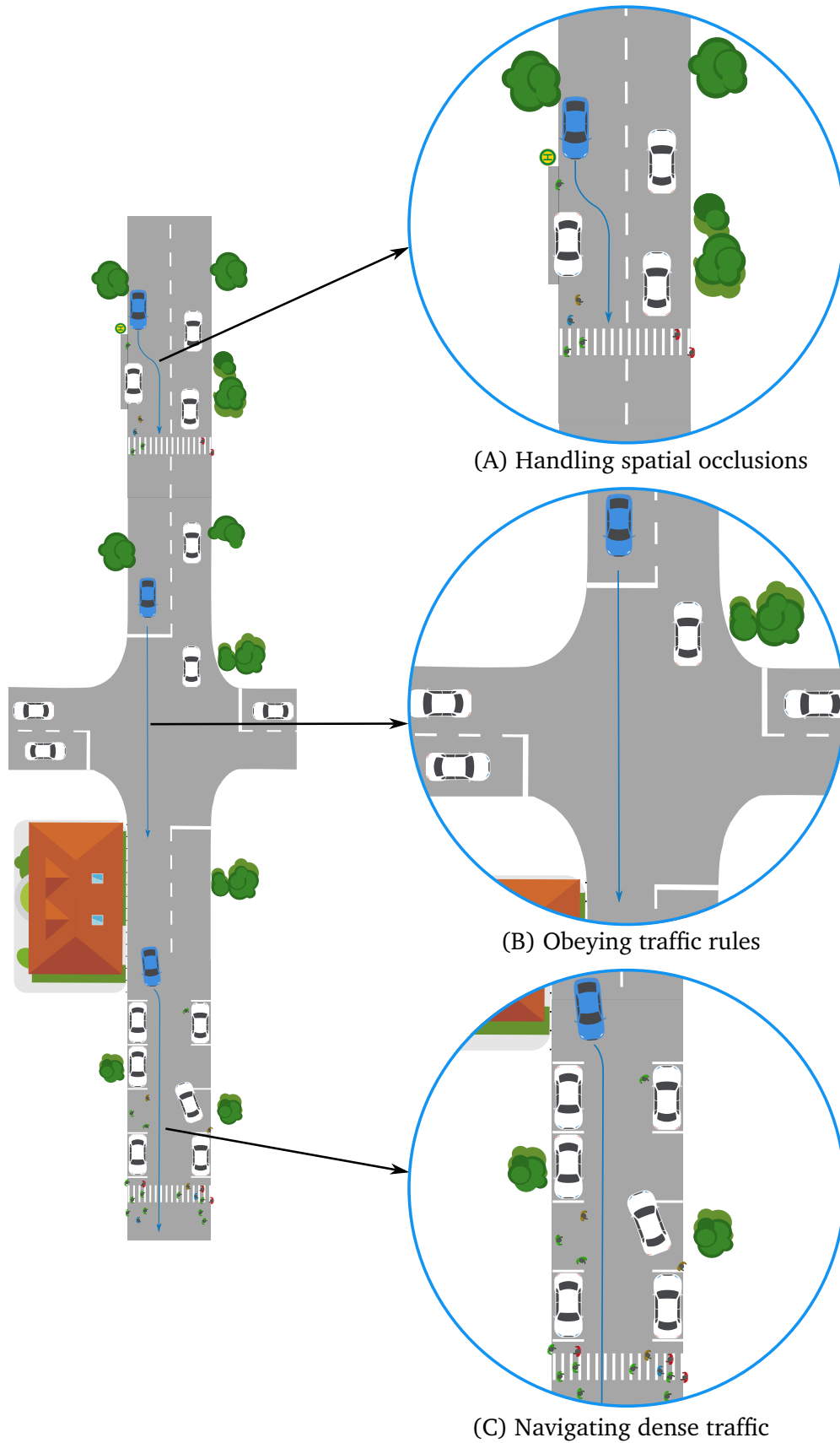


Figure 1.2: Illustration of challenging scenarios in urban environments.

1.2 Behavior Planning Approaches

Many approaches are presented in the literature for addressing sequential decision-making in autonomous driving. This section provides a general overview of various research directions in autonomous driving, categorized by the responsibilities of the designer and the tasks left to automation according to [KWW22].

Explicit Programming

Explicit programming is a rule-based method that considers all possible situations the agent may encounter and defines the agent's responses for each of them. While applying the explicit programming approach to simple driving problems is effective, it is hard to scale this approach to solve complex driving scenarios with various uncertainties.

Supervised Learning

Supervised learning is a machine-learning approach where an autonomous vehicle learns to generate driving behaviors by learning from labeled datasets. The algorithm is trained to provide actions based on input data, and relies on a dataset where each input (e.g., sensor data from cameras, radar, etc.) is paired with a corresponding label (e.g., steering commands, throttle, brake, or lane-following instructions). Supervised learning for autonomous vehicles applies the training data collected from human demonstration; therefore, it is also referred to as imitation learning or behavior cloning. Supervised learning may have limited performance when handling unforeseen or rare situations that may not be adequately represented in the training data.

Optimization

Optimization approaches require the designer to configure the space of the possible decisions and design metrics to be optimized. The algorithm optimizes the designed metrics by searching the space for optimal decision strategies. The dynamic model of the environment is normally used for the search of the optimal actions, but it is not typically used for guiding the search process, which may play an important role in addressing complex problems [KWW22].

Planning

Unlike optimization, which does not necessarily utilize the dynamic model for search, planning employs the environment model to guide the search for optimal decision strategies. A simplified deterministic model has scalability advantages for high dimensional problems and may be utilized depending on the tasks. However, for complex problems such as autonomous driving, planning must incorporate the various uncertainties inherent in the environment. Developing efficient planning methods that effectively handle these uncertainties remains a significant challenge.

Reinforcement Learning

While planning methods require a known environment model beforehand, reinforcement learning methods have no prior knowledge of the environment's dynamics, such as the transition probabilities between states. The agent learns optimal decision strategies by actively exploring and interacting with the environment. Each action the agent takes affects its immediate success in achieving objectives and its understanding of the environment.

1.3 Goals

Depending on the applied models, methods can be deterministic or probabilistic. Methods that only consider deterministic dynamic models tend to result in over-conservative driving behaviors as they commonly deal with uncertainties through expanded safety margins. In contrast, probabilistic approaches balance risks and efficiency by accounting for diverse, uncertain information. This thesis focuses on probabilistic behavior planning methods by modeling driving scenarios as Partially Observable Markov Decision Process (POMDP).

Existing probabilistic planning methods capable of considering various uncertainties have limitations when applied in dense urban environments due to their limited handling of specific occlusion scenarios, decreased performance in crowded areas, and insufficient guarantees for safety and rule compliance (for detailed discussions, see Sections 3.2, 4.2, 5.2, and 6.2). Probabilistic planning methods should not be constrained to handling only specific scenarios, but should be able to handle various types of occlusion scenarios. The computational efficiency of probabilistic planning methods should also be improved when applied in dense environments where the ego vehicle must infer the intentions of many road users. Furthermore, probabilistic methods should take into account traffic rules while ensuring safe driving behaviors in urban areas, where other road users follow the same rules. Therefore, the goal of this thesis is to develop probabilistic behavior planning algorithms for safe and rule-compliant driving in dense and occluded urban environments.

1.4 Contributions

This thesis applies POMDP formulations to model the driving scenarios. Solving POMDP is yet to be tractable [MHC99], therefore, this thesis relies on the approximate solving methods of online tree search and deep reinforcement learning. In summary, this thesis presents the following four major contributions to the field of planning under uncertainty for autonomous driving:

1. **Planning under spatial occlusions using onboard sensors:** A behavior planning algorithm based on a POMDP is presented, which can handle arbitrary types of occluded scenarios by using onboard sensor information. Potentially existing but unseen road users, such as vehicles on occluded roads, and pedestrians at bus stops and crosswalks are taken into account by introducing the phantom road users concept. By incorporating these phantom road users, the behavior planning algorithm can provide less conservative driving behaviors in a broader range of traffic scenarios. The safety and efficiency of the driving behaviors are further improved by introducing a traffic mirror observation module within the POMDP behavior planner.
2. **Planning under spatial occlusions with V2X communication:** The ability to handle occlusions of the introduced algorithm is advanced by incorporating Vehicle-to-Everything (V2X) communication into the POMDP observation model. The proposed algorithm can utilize perception data from both onboard sensors and V2X communications separately, thereby not requiring them to be fused. To avoid selecting potentially unreliable V2X messages, a V2X communication module is incorporated to choose V2X messages providing the most effective information for detecting occluded areas, considering factors like observation area coverage, communication latency, and sensor reliability. Utilizing V2X messages allows the behavior planner to better estimate the existence and movement of potentially occluded road users, thereby improving safety when driving in occluded scenarios.

3. **Efficient planning in dense urban environments:** Applying the proposed algorithms in dense urban environments necessitates significant computational effort. In order to improve the running efficiency, a MOGM-based POMDP planning algorithm is introduced. The MOGM is applied to represent driving environments with uncertain measurements, motion prediction, and intentions. The MOGM is combined with the POMDP formulation, condensing the state space and reducing the number of calculations used for collision checks. Due to the improved solving time of POMDP formulation with MOGM, the presented approach facilitates safe behavior planning in dense traffic scenarios.
4. **Safe and rule compliant planning:** A framework is introduced that ensures both safety and adherence to traffic regulations by using a safety checker and a traffic rule monitor. The safety checker employs the Responsibility Sensitive Safety (RSS) model to verify the safety of planned driving behaviors. The traffic rule monitor assesses the adherence to the right-before-left rule. To demonstrate the effect of the proposed approach, the formulated model is solved using deep reinforcement learning.

1.5 Publications

This thesis is based on the following publications:

- Zhang, C., Steinhauser, F., Hinz, G., and Knoll, A. “Improved Occlusion Scenario Coverage with a POMDP-based Behavior Planner for Autonomous Urban Driving”. In: *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems*. 2021, pp. 593–600
- Zhang, C., Steinhauser, F., Hinz, G., and Knoll, A. “Traffic Mirror-Aware POMDP Behavior Planning for Autonomous Urban Driving”. In: *Proc. of the IEEE Intelligent Vehicles Symposium*. 2022, pp. 323–330
- Zhang, C., Steinhauser, F., Hinz, G., and Knoll, A. “Occlusion-Aware Planning for Autonomous Driving With Vehicle-to-Everything Communication”. In: *IEEE Transactions on Intelligent Vehicles* (2023). Early Access
- Zhang, C., Ma, S., Wang, M., Hinz, G., and Knoll, A. “Efficient POMDP Behavior Planning for Autonomous Driving in Dense Urban Environments using Multi-Step Occupancy Grid Maps”. In: *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems*. 2022, pp. 2722–2729
- Zhang, C., Kacem, K., Hinz, G., and Knoll, A. “Safe and Rule-Aware Deep Reinforcement Learning for Autonomous Driving at Intersections”. In: *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems*. 2022, pp. 2708–2715

1.6 Outline

This section explains the outline of the thesis, as shown in Fig. 1.3. Chap. 2 gives the fundamentals of the Markov Decision Process (MDP) and POMDP for formulating autonomous driving behavior planning as a sequential decision-making problem. Next, it introduces in detail the solving methods applied in the thesis: the online tree search and deep reinforcement learning.

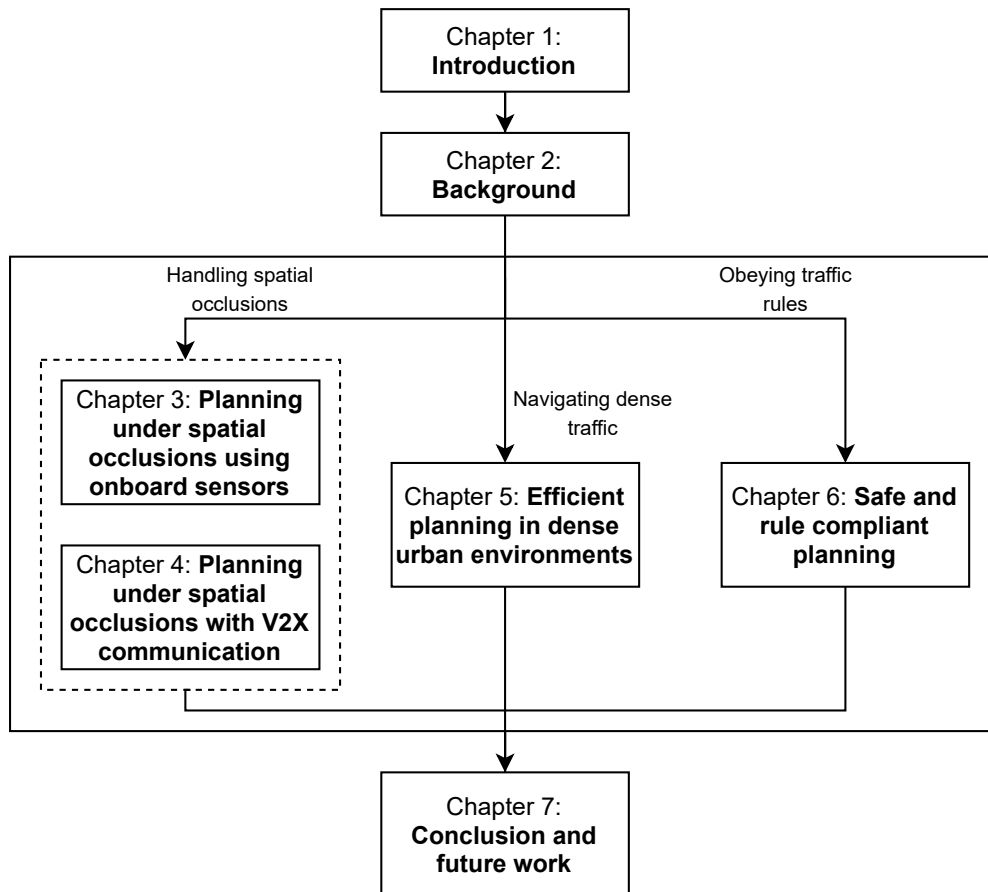


Figure 1.3: This diagram outlines the structure of the thesis, beginning with an introduction and followed by a background chapter. It details the probabilistic behavior planning algorithms for safe and rule-compliant driving in dense and occluded urban environments, divided into three parts: handling spatial occlusions, navigating dense traffic, and obeying traffic rules. The thesis ends with a chapter on conclusions and suggestions for future work.

Chap. 3 presents a POMDP behavior planner that can handle spatial occlusions under various driving scenarios using the information from the onboard sensors. It shows how different potentially occluded road users, such as vehicles and pedestrians, can be modeled and incorporated into the POMDP formulation. The reasoning for the further movement of potentially occluded road users allows the presented approach to reduce the collision risk with such unobservable road users. This chapter also takes advantage of perceived information from traffic mirrors and further extends the POMDP formulation with a traffic mirror observation module. It shows how the traffic mirror information can improve the estimation of potentially occluded road users and thus improve the safety of the planned driving behavior.

Chap. 4 presents a POMDP behavior planner that combines the benefits of methods using onboard sensors and methods using Vehicle-to-Everything (V2X) communication. It illustrates how potentially unreliable V2X messages are efficiently selected from multiple road infrastructures or V2X vehicles and are applied to improve the estimation of the occluded road users.

Chap. 5 focuses on improving the computational efficiency of the POMDP behavior planning algorithm. It introduces the efficient POMDP behavior planner to enable the application of the planner in dense urban environments. This chapter shows how a Multi-step Occupancy Grid Maps (MOGM) is applied for modeling the uncertain measurements, predictions, and

intentions of large numbers of road users, and how the POMDP formulation is extended to combine the MOGM to improve the planner's computational efficiency.

Chap. 6 addresses the problem of considering safety and traffic rule constraints in the POMDP-based algorithms. It presents a safe and rule-aware behavior planner for autonomous vehicles to handle intersection scenarios in urban environments. Instead of solving the formulation using an online tree search method, this chapter provides an alternative solving method using deep reinforcement learning. It demonstrates how the right of way rule is modeled within a rule monitor and how the RSS-based safety checker guides the training and planning phases of the behavior planning algorithm.

Finally, Chap. 7 summarizes the main outcomes of the thesis and discusses the future research direction in behavior planning under uncertainty for autonomous driving.

2

Background

This chapter gives the theoretical background and mathematical notions applied in this thesis. Section 2.1 summarizes the uncertainties that arise in autonomous driving and provides the formal definitions of the MDP and POMDP for modeling the sequential decision making problem. Section 2.2 introduces different solving methods, including offline, online methods, and Q-learning.

2.1 Sequential Decision Making under Uncertainty

Sequential decision making refers to the process of making a series of decisions over time in a specific context or environment [Lit96]. In sequential decision making, the result of each decision can affect the presently available options and future decisions. The objective is to choose actions that optimize a predetermined objective or utility function. To achieve this, the agent needs to consider the current state, possible actions, potential outcomes and their associated probabilities and rewards.

This section first summarizes various uncertain information sources an autonomous vehicle may face when making decisions for autonomous driving. Then, this section provides the mathematical definitions of the MDP and POMDP.

2.1.1 Environment and Agent

Agents are decision-making entities that can take various forms, such as humans, software programs, robots, or Artificial Intelligence (AI) systems, which perceive and interact with their environment to achieve specific objectives [KWW22]. For example, agents can be budget directors managing finances, or network monitoring equipment overseeing communication systems. They can also be software programs like chatbots and recommendation systems, autonomous robots performing tasks in the physical world, or AI systems playing games.

Although the agent is a decision-making entity, it is not sufficient for it to only make decisions. It must also select an action a_t to be executed to affect the environment's state. The agent's action selection depends on its perception of the environment. As shown in Fig. 2.1, the agent interacts with its environment following an observation and action loop. At time step t , the agent perceives its environment and transforms the state of the environment into an observation o_t . Perceptions can be made through biological sensors like the human eye or physical sensors like radar or LiDAR on autonomous vehicles. In some environments,

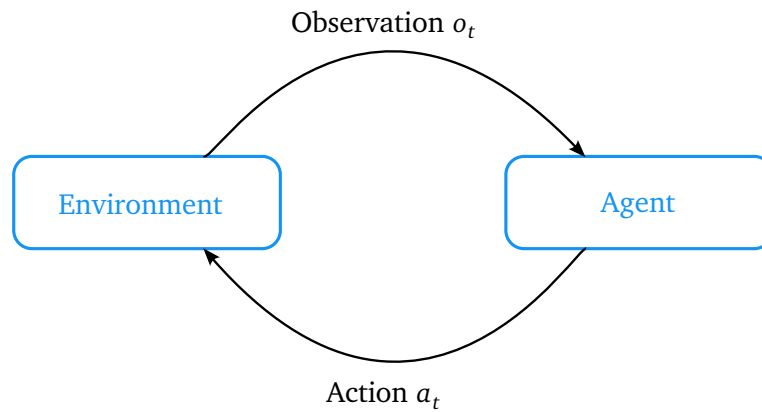


Figure 2.1: The process of interaction between the agent and its environment (graphic adapted from [KWW22]).

the agent has full access to the environment's true state. In other situations, the agent can only partially observe the environment.

The agent's goal is to choose the best action given its observations and knowledge of the environment while considering various sources of uncertainty: outcome uncertainty, model uncertainty, state uncertainty, and interaction uncertainty [KWW22]:

- state uncertainty: the underlying state of the system is not directly observable. There is uncertainty regarding the actual state of the environment,
- outcome uncertainty: even if the agent knows the current state and the action it wants to take, the outcome of the action may be uncertain. This exists particularly in stochastic environments,
- model uncertainty: complete knowledge of the environment's dynamics is not known to the agent. This can be due to incomplete or inaccurate models of the system, or to the fact that the system itself is changing over time.
- interaction uncertainty: the uncertainty arising from the interactions between the agent and other agents in the environment.

2.1.2 Uncertainties in Autonomous Driving

When making decisions for autonomous vehicles, an agent has to deal with multiple sources of uncertain information. As shown in Fig. 2.2, the ego vehicle has to consider the following uncertainties when driving through the intersection.

Measurement Uncertainty

An autonomous vehicle cannot obtain the precise states of surrounding objects because these states are inferred from different sensors with varying levels of noise. For example, when the autonomous vehicle detects another vehicle, there may be uncertainties in its measurements of the detected vehicle's localization, orientation, and velocity.

Prediction Uncertainty

Autonomous vehicles must predict the states of other road users to make informed decisions. As they do so, the uncertainty associated with these predictions increases over time. As illus-

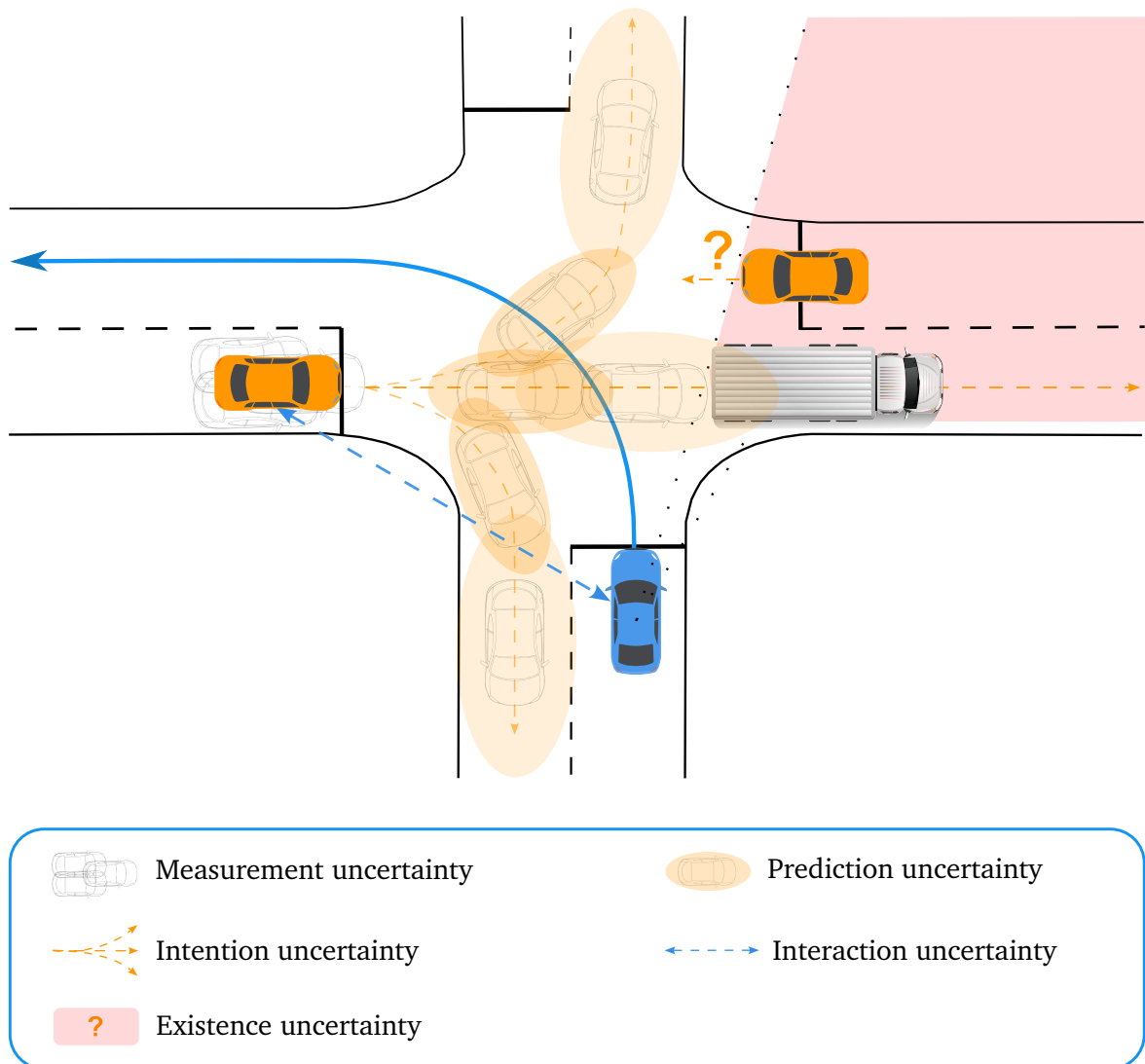


Figure 2.2: Illustration of the various sources of uncertainty for autonomous vehicles driving in urban environments.

trated in Fig. 2.2, shorter-term predictions are more precise, whereas longer-term predictions show a broader range, indicating increased uncertainty.

Intention Uncertainty

Predicting the intentions of dynamic road users also contains uncertainties. For example, consider orange vehicle at the intersection shown in Fig. 2.2. The autonomous vehicle faces the challenge of estimating the orange vehicle's intended behavior, which could be either to continue driving straight or to make a turn. Due to the uncertainty of these intentions, autonomous vehicles may find it challenging to make correct decisions and respond appropriately to other road users.

Interaction Uncertainty

When multiple vehicles interact, the actions of one vehicle, such as a sudden brake, can trigger a chain reaction among others. This increases the complexity of decision-making in

autonomous vehicles and requires them to anticipate and respond to these interactions to ensure safety on the road.

Existence Uncertainty

Occluded areas can result in uncertainty regarding the existence of other road users and objects. In urban environments, autonomous vehicles have a restricted FoV due to static or moving objects. Within the occluded areas, there could be vehicles or pedestrians hidden from view, which autonomous vehicles need to consider when making decisions.

2.1.3 Markov Decision Process

An MDP is a standard mathematical model in which an agent interacts with an environment in discrete time steps [KWW22]. As shown in Fig. 2.3, the agent observes the current state of the environment and chooses an action at each time step. Subsequently, the environment transitions to a new state according to its stochastic dynamics. After each state transition, the agent receives a reward signal that indicates how well or how poorly the agent's action in the previous state contributed to achieving its objectives in the environment. In general, a subscript t is used to denote the state or action at time step t . The state at time t is represented as x_t , and the action at time t is denoted as a_t . However, in some cases, only one time step is relevant. For simplicity and clarity, the subscript is dropped and superscript is applied to denote the next state x' of the system.

The MDP model assumes that the agent has complete knowledge of the environment. However, the agent's actions have uncertain effects on the environment. Thus, the MDP tackles the problem of selecting optimal actions in completely observable stochastic environments. This decision process is called Markov because it adheres to the Markov assumption; that is, the next state of the environment depends only on the current state and the actions [KWW22]. The current state contains all the relevant information about the environment for predicting the subsequent state.

Formally, a MDP is defined by the tuple $(\mathcal{X}, A, T, R, \gamma)$:

- The state space \mathcal{X} contains all possible states of the environment. A state is defined as $x \in \mathcal{X}$.
- The action space A includes all actions that the agent can choose. An action chosen by the agent is defined as $a \in A$.
- The transition model T describes the state transition of the environment over time. $T(x, a, x') := P(x' | x, a)$ models the probability of a system transition from the state $x \in \mathcal{X}$ to the next state $x' \in \mathcal{X}$ when action $a \in A$ is executed.
- The reward model $R(x, a)$ is the reward generated by performing the action $a \in A$ from the state $x \in \mathcal{X}$.
- The discount factor $\gamma \in [0, 1)$ prioritizes the current reward over future rewards.

The objective of the MDP is to obtain a policy $\pi : \mathcal{X} \rightarrow A$ that maximizes the value function $V(x)$, which is the expectation of accumulated reward over time:

$$V(x) := \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t R(x_t, a_t) \mid x_0 = x \right]. \quad (2.1)$$

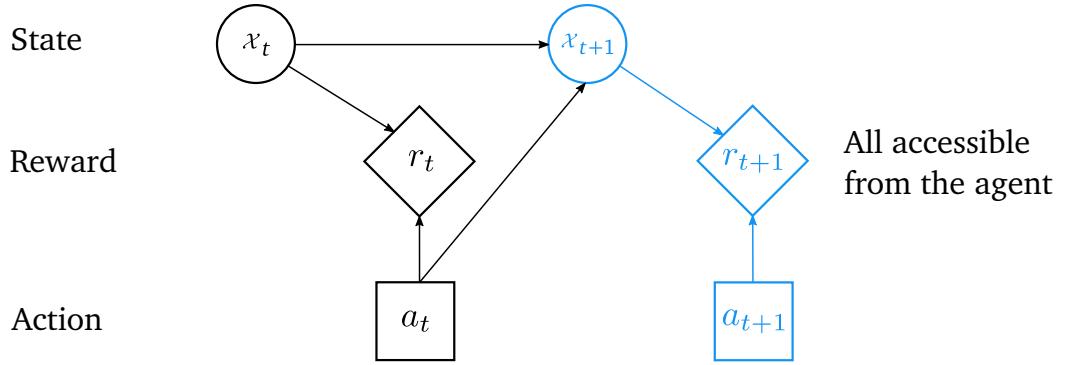


Figure 2.3: Illustration of the MDP depicting a state x_t at a given time step, the action a_t chosen by an agent, the subsequent state x_{t+1} following the action, and the reward r_{t+1} received as a result of the action.

An optimal policy π^* is a policy that maximizes expected value:

$$\pi^*(x) := \arg \max_{\pi} V(x). \quad (2.2)$$

The optimal value function, denoted by $V^*(x) : \mathcal{X} \rightarrow \mathbb{R}$, represents the cumulative expected reward when the agent starts in a state x and follows the optimal policy π^* . The objective of an MDP can also be specified as the optimal state action value function $Q^*(x, a)$, which is the expected cumulative reward when the agent starts in the state x , choose an action a , then follows the optimal policy π^* .

An optimal policy can be found by applying dynamic programming to recursively break down the problem into simpler subproblems. This idea is explained by Bellman's principle of optimality [Bel66]: an agent can achieve its optimal objective, regardless of its initial state and decisions, as long as it adheres to the optimal policy from the state resulting from the initial decision. As a result, the optimal value function $V^*(x)$ can be calculated recursively:

$$V^*(x) = \max_{a \in A} \left[R(x, a) + \gamma \sum_{x' \in \mathcal{X}} T(x, a, x') V^*(x') \right]. \quad (2.3)$$

2.1.4 Partially Observable Markov Decision Process

While MDP assumes that the agent can fully observe the environment state, in more complex scenarios, the agent may have limited or imperfect information about the environment. Such a problem can be modeled as a POMDP, which extends MDP by modeling the state uncertainty. In a POMDP, an agent makes sequential decisions in an environment where it receives partial and noisy information about the state of the environment (see Fig. 2.4).

A POMDP is defined by the tuple $(\mathcal{X}, A, O, T, Z, R, \gamma)$, where the state, action and observation spaces as well as the transition function, reward function and discount factor are the same as defined in MDP. A POMDP has two additional components for modeling the agent's observation of the environment state:

- An observation space O which includes all possible observations that the agent can receive from the environment. The observation is defined as $o \in O$.
- An observation model Z is the conditional probability function $Z(o, a, x') = P(o | x', a)$, describing the probability of receiving observation $o \in O$ after taking action $a \in A$ and transitioning to state $x' \in \mathcal{X}$.

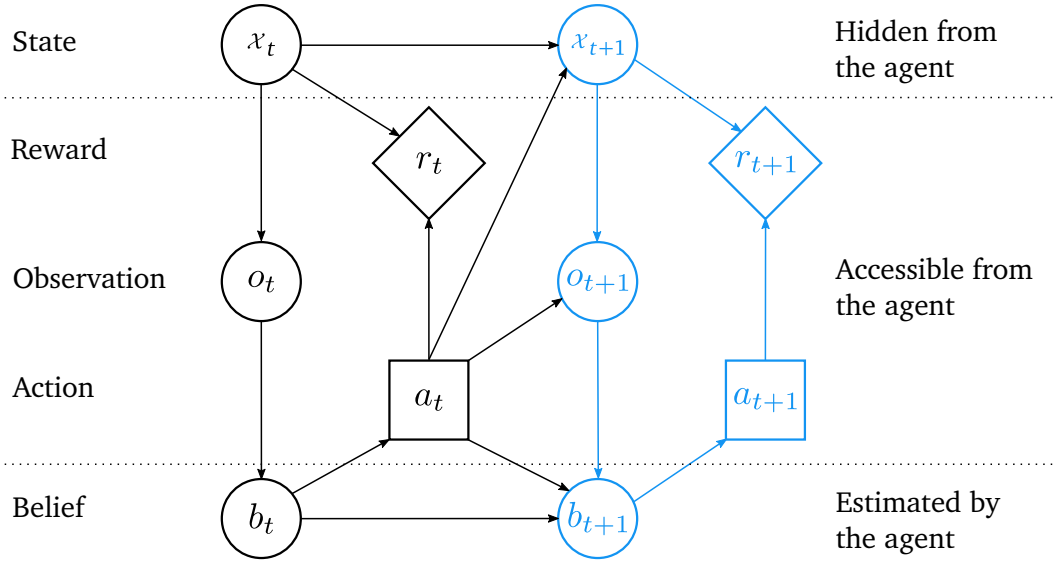


Figure 2.4: Illustration of the POMDP depicting the relationship between the hidden state x_t , the observation o_t available to the agent, the action a_t taken by the agent, and the belief b_t about the state which is updated based on the observation. The process then moves to the next time step, where the cycle repeats.

Belief and Belief Space

In a partially observable environment, the agent has only incomplete knowledge of its surroundings, and in contrast to MDPs, states cannot be directly observed but only estimated. Instead of mapping POMDP policies from state to action, the agent uses historical actions and observations to maintain a belief about the true state. A belief is a probability distribution over possible states $b \in B : \mathcal{X} \rightarrow [0, 1]$, where B is the set of all possible beliefs:

$$\sum_x b(x) = 1, \quad b(x) \geq 0, \forall x \in \mathcal{X}. \quad (2.4)$$

Value Function for Belief State

The policy $\pi : B \rightarrow A$ is a mapping from a belief $b \in B$ to an action $a \in A$. The objective of the POMDP is to obtain a policy π that maximizes the value function $V(b)$, which is the expectation of an accumulated reward over time:

$$V(b) := \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t R(x_t, \pi(b_t)) \mid b_0 = b, \pi \right], \quad (2.5)$$

where b_0 denotes the current belief and t represents the time steps.

The optimal policy for maximizing expected value is defined as:

$$\pi^*(b) := \arg \max_{\pi} V(b). \quad (2.6)$$

Similarly to MDP, the optimal value function for POMDP can be formulated using Bellman expectation equation:

$$V^*(b) := \max_{a \in A} \left[R(b, a) + \gamma \sum_{b' \in B} T(b, a, b') V^*(b') \right]. \quad (2.7)$$

The reward function $R(b, a)$ for the POMDP is the expected value over the belief and chosen action. The reward function for the discrete state space is formulated as:

$$R(b, a) = \sum_{x \in \mathcal{X}} R(x, a) b(x). \quad (2.8)$$

For continuous state space, the summation becomes an integral.

2.2 Solving Sequential Decision Making under Uncertainty

This section introduces the solving methods applied in this thesis. First, representatives of the offline method for solving POMDP are discussed. Next, detailed algorithms of the online method are presented. Finally, the concept of Deep Reinforcement Learning (DRL) as a learning-based method is presented.

2.2.1 Solving Model using Offline Methods

Offline methods compute the approximately optimal policy by calculating the value function over the entire belief space before execution. This section provides a brief review on offline methods for solving the MDP and the POMDP model.

Value Iteration

The value iteration algorithm is a standard method for finding the optimal policy π^* for MDPs [How60]. The Value Iteration algorithm iteratively updates the value function until it converges to the optimal values. It starts with an initial estimation of value function $V_0(x)$ at time step $t = 0$, then it iteratively updates the value function estimation using the Bellman Optimality Equation:

$$V_{t+1}(x) = \max_{a \in A} \left[R(x, a) + \gamma \sum_{x' \in \mathcal{X}} T(x, a, x') V_t(x') \right]. \quad (2.9)$$

The iteration is finished once $|V_{t+1}(x) - V_t(x)| < \epsilon$, where ϵ is the maximum difference between two successive value functions. Once the value function has converged, the optimal policy $\pi^*(x)$ can be extracted by selecting the action that maximizes expected value according to (2.2).

In a POMDP, value iteration is applied to the belief space instead of to the state space to find the optimal policy. The steps of value iteration for POMDPs are similar to those of value iteration for MDPs. It initializes $V_0(b), \forall b \in B$ at $t = 0$, then repeatedly computes $V_{t+1}(b), \forall b \in B$ and increments to the next time step:

$$V_{t+1}(b) = \max_{a \in A} \left[R(b, a) + \gamma \sum_{b' \in B} T(b, a, b') V_t(b') \right], \quad (2.10)$$

until the value function estimate converges with $|V_{t+1}(b) - V_t(b)| < \epsilon$.

Policy Iteration

Similar to value iteration, policy iteration is another method for resolving MDPs [How60]. Policy iteration begins with an initial policy π and then iteratively improves the policy until it converges into the optimal policy π^* .

Policy iteration consists of two steps: policy evaluation and policy improvement. It begins with a random policy $\pi_0(x)$, $\forall x \in \mathcal{X}$. In the policy evaluation step, the value of the policy π_t is computed:

$$V^{\pi_t}(x) = R(x, \pi_t(x)) + \gamma \sum_{x' \in \mathcal{X}} T(x, \pi_t(x), x') V^{\pi_t}(x'). \quad (2.11)$$

In the subsequent policy improvement step, the Q-value is first calculated using the policy π_t :

$$Q_{t+1}(x, a) = R(x, a) + \gamma \sum_{x' \in \mathcal{X}} T(x, a, x') V^{\pi_t}(x'). \quad (2.12)$$

Then the improved policy π_{t+1} is obtained according to:

$$\pi_{t+1}(x) = \arg \max_{a \in A} Q_{t+1}(x, a), \quad \forall x \in \mathcal{X}. \quad (2.13)$$

This process repeats until the policy converges, i.e., $\pi_{t+1}(x) = \pi_t(x)$.

The first policy iteration algorithm for POMDPs was introduced by Sondik [Son71]. However, its policy evaluation step is very complicated because the algorithm must translate different policy representations within this single step. Hansen [Han97] presents a simpler and more efficient policy iteration algorithm by using a single representation of a policy for POMDPs as a finite-state controller. The finite-state controller represents policies as a policy graph of internal states and their transitions, allowing the policy to be mapped from internal states to actions. Benefiting from this representation, the policy evaluation step becomes more straightforward. Similar to the policy iteration in MDPs, the policy represented by the finite-state controller is iteratively evaluated and improved until it converges to the optimal policy.

Exact solving methods for large POMDPs are highly intractable because of the curse of dimensionality [KLC98] and the curse of history [PGT06]. The curse of dimensionality refers to the exponential growth in computational complexity as the number of states increases, and the curse of history refers to the growth in complexity as the history of observations and actions increases. Therefore, many practical algorithms for POMDPs use approximations to make the problem tractable.

QMDP

One famous offline approximation method for POMDPs is the Fully Observable Value Approximation (QMDP) introduced by Littman, Cassandra, and Kaelbling [LCK95]. QMDP solves POMDPs approximately by assuming that the uncertainty in the agent's current belief state will be eliminated after the next action.

The QMDP approach extends the optimal value function for MDPs with regard to states, into a value function for POMDPs over beliefs. Value functions are more straightforward to calculate for MDPs than POMDPs, but MDPs depend on the assumption of the full observability of states.

The approximated optimal belief state action function for POMDPs over belief state is defined as:

$$Q_{\text{MDP}}(b, a) = \sum_{x \in \mathcal{X}} Q_{\text{MDP}}(x, a) b(x), \quad (2.14)$$

where $Q_{\text{MDP}}(x, a)$ is the optimal state action value function for the MDPs. Accordingly, the action selected at each step will be the one that maximizes the accumulated rewards from all states, weighted by the probability of being in each state.

Since the QMDP assumes all the uncertainty in the state will vanish in the next time step, it has no intention to reduce state uncertainty and QMDP therefore cannot choose the action

for information-gathering purposes, which is costly but can reduce uncertainty in the state. Nonetheless, QMDP is an efficient approximation method for solving POMDPs given that it solely relies on the solution to the fully observable MDP, which is typically computationally tractable through the efficient value iteration method.

Point-based Methods

While the QMDP approximation is computationally efficient, it may inadequately approximate the value function due to the suboptimal nature of the approach.

One known issue in solving POMDPs is the curse of dimensionality caused by the size of the belief space. As shown in [PGT+03], even a small amount of size $|\mathcal{X}|$ of discrete states results in a continuous belief space B with a dimensionality of $(|\mathcal{X}|-1)$. Hence, simple discretization of the belief space leads to a number of belief states that grows exponentially with the number of states. Another approach for solving POMDPs offline is point-based methods, which improve computational efficiency by only calculating a finite set of representative belief points for estimating the value function.

In point-based methods, the utility function for the belief state is depicted using a finite collection of alpha vectors, denoted by Γ . These alpha vectors are associated with designated belief points and are composed of $|\mathcal{X}|$ -dimensional vectors that establish a linear function across the belief space: $\Gamma = \{\alpha_1, \dots, \alpha_n\}$. The utility function can then be calculated as follows:

$$V(b) = \max_{\alpha \in \Gamma} \alpha^\top b. \quad (2.15)$$

Point-based Value Iteration (PBVI) is the first approximate POMDP solution that performed satisfactorily on problems involving hundreds of states [PGT+03]. Other point-based methods, such as Heuristic Search Value Iteration (HSVI) [SS04], and Successive Approximations of the Reachable Space under Optimal Policies (SARSOP) [KHL08] reduce computational complexity by only computing the solutions for reachable belief spaces using tree search. These methods guide the tree search by using upper and lower bounds on the value function.

2.2.2 Solving Model using Online Methods

Calculating whole solutions for a large belief state and action space may be intractable. Online methods compute approximate optimal policies for the current belief state, which are sometimes more easily applicable to large-scale problems [KWW22]. In contrast to offline methods, which determine the optimal policy for the entire belief state offline, online methods calculate the best action from the current belief state up to a limited horizon right before execution. An advantage of online methods is that the belief space reachable from the current state is usually smaller than the entire belief space. The belief state is updated after executing the policy, then the next iteration for computing and executing the optimal policy begins.

This section first introduces a widely applied online method for solving MDPs, then explains the algorithm used in this thesis: the Adaptive Belief Tree (ABT) algorithm for solving POMDPs with continuous state space.

Monte Carlo Tree Search

Monte Carlo Tree Search (MCTS) is a widely utilized algorithm for solving sequential decision-making problems, particularly in the context of perfect-information games [Bro+12]. MCTS combines elements of random simulation (Monte Carlo simulation) with the tree structure

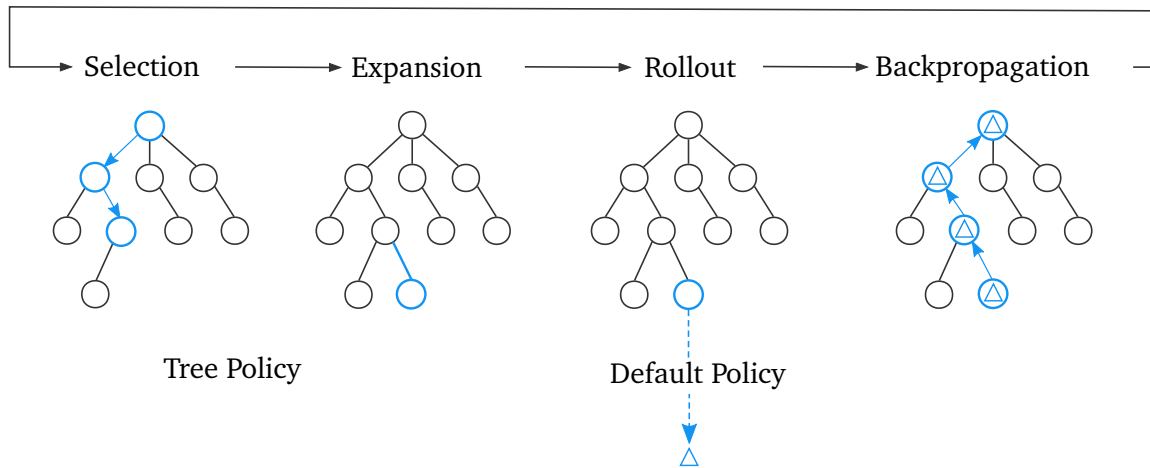


Figure 2.5: Four steps of each iteration in the Monte Carlo tree search algorithm (graphic adapted from [Bro+12]).

to efficiently explore the search space. MCTS incrementally constructs a policy tree based on Monte Carlo simulations, which are multiple simulations of the current state for different action sequences. The policy tree comprises state and action nodes. State nodes depict the state of the agent, while action nodes represent the available actions of the agent. Instead of generating an entire policy tree by exploring every possible action for each state, MCTS builds an asymmetric policy tree focusing more on extending the branches with higher values. Constructing the policy tree consists of iteratively performing the following four steps: Selection, Expansion, Simulation, and Backpropagation, as shown in Fig. 2.5.

1. Selection: The selection step starts from the root node. The algorithm traverses the policy tree recursively using a selection strategy until the expandable leaf node is reached. A node is expandable if it is in a non-terminal state and has at least one unvisited child.
2. Expansion: The expansion step adds a new child node corresponding to the selected node's action.
3. Simulation: The simulation step is also known as roll-out. In this step, simulations are performed to estimate the future reward of the newly created node. The simulation can follow a default policy or other sophisticated policies designed for more accurate value estimation of the new node.
4. Backpropagation: Finally, in the backpropagation step, the simulation results are backpropagated through the selected nodes to the root of the policy tree.

To obtain a promising policy in a limited time, the selection step in MCTS should use an appropriate strategy for guiding the tree search. The Upper Confidence Bounds for Trees (UCT) algorithm is a commonly applied selection policy [ACF02]. UCT provides a trade-off between two terms: the exploitation term, which represents the estimated value of the node based on current knowledge, and the exploration term, which encourages the algorithm to explore less-visited parts of the tree. UCT selects the action that maximizes the upper confidence bound:

$$U(x, a) = Q(x, a) + c \sqrt{\frac{\log N(x)}{N(x, a)}}, \quad (2.16)$$

where $Q(x, a)$ is the average reward of node x , $N(x)$ is the total number of visits to node x , $N(x, a)$ is the number of visits to its child node a , and c is the exploration constant that balances the trade-off between exploration and exploitation.

Adaptive Belief Tree

MCTS does not directly solve POMDPs involving unobservable states. Several MCTS extensions have been proposed to overcome this limitation, such as Partially Observable Monte Carlo Planning (POMCP) [SV10] and ABT [KY16]. These algorithms modify the core MCTS approach to handle belief states in POMDPs. The following section gives a detailed explanation of the ABT algorithm since it is an online method for solving POMDPs applied in this thesis.

1. Generative Model

ABT does not require an explicit definition of the transition, observation, and reward functions. Instead, it applies a generative model $G(x, a)$ to simulate the system dynamic. A generative model does not require an explicit model on control error, observation error, and uncertainty about the system dynamics. It is considered as a black box simulator, or a one-step dynamic function that encapsulates all the complexity, which are difficult to obtain for many problems [KY16]. The generative model is used to generate sequence of states, observations and rewards for estimation of the value function.

A generative model $G(x, a)$ is defined as follows:

$$(x', o', r') \sim G(x, a). \quad (2.17)$$

With a given state $x \in \mathcal{X}$ and the chosen action $a \in A$, the generative model stochastically samples a next state $x' \in \mathcal{X}$, an observation $o' \in O$ and assigns the immediate reward $r' = R(x, a)$. The probabilistic transition model T and observation model Z is implicitly defined in the generative model $G(x, a)$.

2. Belief Tree Representation

The ABT is represented as a belief tree \mathcal{T} with the association of a set of episodes \mathcal{H} . Each node in the belief tree \mathcal{T} represents a belief b . The root of the tree is the initial belief b_0 . An edge connects two successive belief nodes $b \in B$ and $b' \in B$. The edge indicates that when an agent at a belief $b \in B$ performs an action $a \in A$ and perceives an observation $o \in O$, its next belief is $b' \in B$.

ABT represents the policy as pairs of beliefs and actions. To explicitly represent the relation between beliefs, states, and their reachability information, it associates a path ϕ in the belief tree, which represents sequences of beliefs with an episode that collects sequential states, and their reachability information.

A path ϕ in the belief tree \mathcal{T} is a sequence of nodes and edges from the initial belief b_0 in the tree to belief b_{n+1} with $\phi = \langle b_0, a_0, o_0, \dots, a_n, o_n, b_{n+1} \rangle$, where $b_i, b_{n+1} \in B, a_i \in A$, and $o_i \in O$ for $i \in [0, n]$. The path ϕ is associated with all episodes \mathcal{H} , which contain the same path ϕ with the node sequence from b_0 to b_{n+1} .

An episode $h \in \mathcal{H}$ is one possible state trajectory starting from the current state to the maximum length of the planning horizon. An episode contains a sequence of quadruples (x, a, o, r) . The process of sampling an episode starts with selecting an initial state x_0 from the initial belief b_0 and action a_0 . Then, the ABT invokes the generative model $G(x, a)$ to sample an observation o_0 , an immediate reward r_0 , and a next state x_1 . The process is

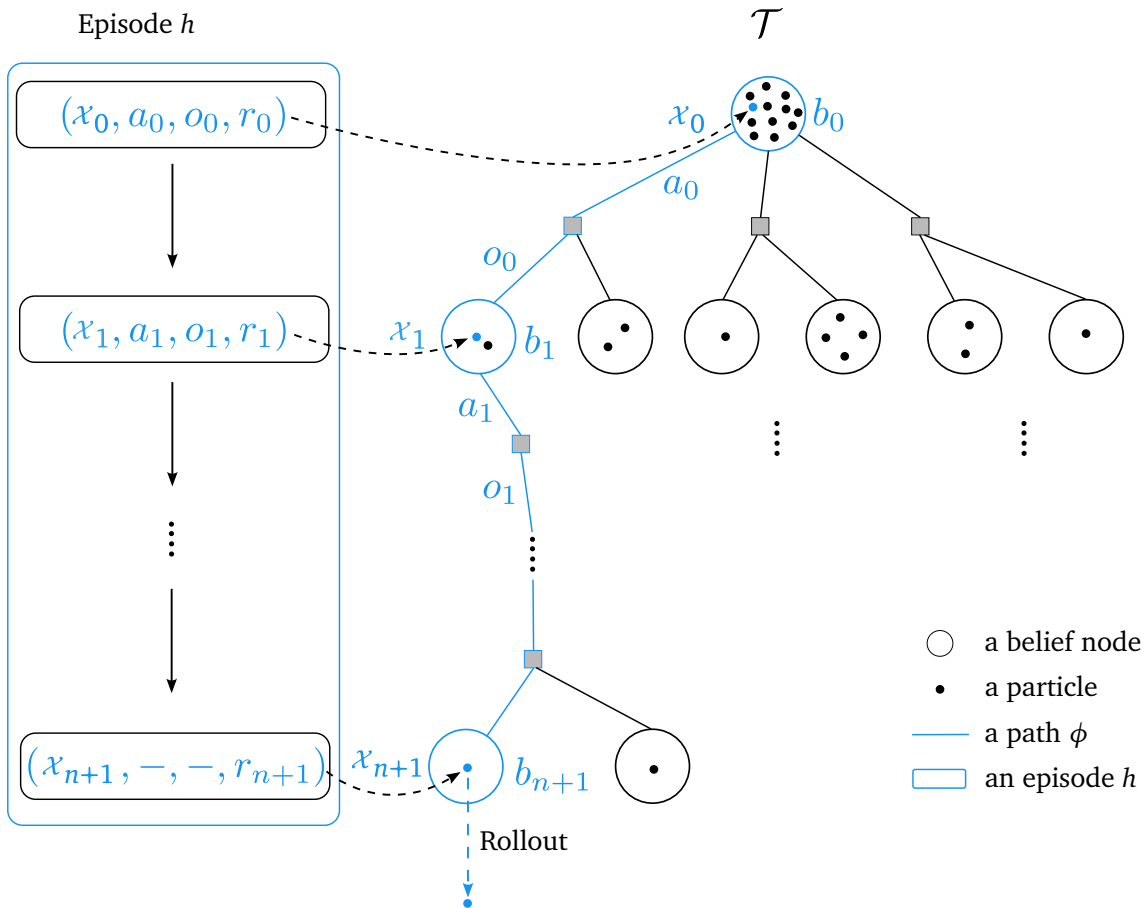


Figure 2.6: Illustration of a belief tree and the associated episodes. Each node in the tree represents a belief state, with edges representing actions leading to subsequent beliefs based on observations. Episodes trace the sequence of belief states and actions taken over time (graphic adapted from [KY16]).

repeated iteratively until either a terminal state is reached or the episode has reached a maximum length. The final element of the episode is a tuple $(x, -, -, r)$, where x is the next state sampled and r is the reward of being at that state. Fig. 2.6 shows the relation between an episode h and a path ϕ in the belief tree \mathcal{T} . Each episode h has been assigned to one path in the belief tree \mathcal{T} , while a path can be contained by multiple episodes.

3. Construction of the Belief Tree

ABT constructs the belief tree by performing many forward searches. It starts to construct the belief tree \mathcal{T} from the initial belief node b_0 , and randomly samples a state x_0 from the belief node b_0 . This sampled state is used to start a single search that generates an episode h . The single search contains multiple extension steps and one backup step. Given the sampled state, ABT selects an action using the UCT algorithm and invokes the generative model $G(x, a)$ to move one step forward and sample the next state, observation, and reward. It inserts this quadruple in the episode h and associates the generated state with the belief node. If this belief node has been visited before, the extension step repeats until it reaches the planning horizon. Otherwise, ABT generates a new belief node that contains only one state. ABT estimates the value of the new belief node $\hat{V}(b_n)$ by calculating the cumulative discounted reward following a predefined rollout policy. The last step of the single search

is to backpropagate the estimated value of the belief nodes along the path of the belief tree to the root. After the time for constructing the belief tree elapses, the approximate optimal action a can be obtained from the belief tree.

4. Approximate Optimal Solution

The ABT algorithm aims to find an approximately optimal policy π^* for a given belief state b by maximizing the Q-value $Q(b, a)$. The Q-value denotes the expected discounted reward when taking action a given the belief b :

$$Q(b, a) = R(b, a) + \gamma \sum_{o \in \mathcal{O}} T(b, a, o) V^*(T(b, a, o)), \quad (2.18)$$

where $T(b, a, o)$ represents the belief update function, i.e., $b' = T(b, a, o)$, given the initial belief state b , the chosen action a , and the perceived observation o . The optimal policy π^* that maps the belief to action can be obtained by:

$$\pi^*(b) := \arg \max_{a \in A} Q(b, a). \quad (2.19)$$

ABT estimates the Q-value using the generated episodes H associated with the belief tree \mathcal{T} . For estimation of the Q-value, the episode value is needed. The episode value $V(h, l)$ is the value of an episode h starting from the depth l of the belief tree \mathcal{T} :

$$V(h, l) = \sum_{i=l}^{|h|} \gamma^{i-l} R(h_{i:\cdot}, x, h_{i:\cdot}, a), \quad (2.20)$$

here, l indicates the depth of belief b in the belief tree, γ is the discount factor, and R is the reward function given the recorded state $h_{i:\cdot}, x$ and action $h_{i:\cdot}, a$ from the quadruples.

The approximated Q-value $\hat{Q}(b, a)$ given the belief b and action a is calculated using the subset of sampled episodes $H_{(b,a)}$ containing the sequence (b, a) :

$$\hat{Q}(b, a) = \frac{1}{|H_{(b,a)}|} \sum_{h \in H_{(b,a)}} V(h, l). \quad (2.21)$$

5. Action Selection

ABT applies two strategies for selecting an action given a state x corresponding to node b in the belief tree, depending on whether the belief node b has been fully explored. During the construction of the belief tree, if a belief node has not yet been fully explored, i.e., at least one action has not been visited before, the uniformly random selection strategy is applied:

$$a = \text{rand}(A'), \quad (2.22)$$

where $\text{rand}(\cdot)$ is a function that chooses an action $a \in A'$ based on uniform distribution. $A' \subseteq A$ contains actions which have not been selected to expand belief node b .

Otherwise, when all the actions in A for a given state x associated with the belief node b have been visited at least once, ABT employs the UCT strategy to select an action:

$$a = \arg \max_{a \in A} (\hat{Q}(b, a) + c \sqrt{\frac{\log |H_b|}{|H_{b,a}|}}), \quad (2.23)$$

where H_b represents the episodes starting from b , and $H_{b,a}$ is the set of episodes that begin from b and take action a . $(|\cdot|)$ indicates the size of the set, and $c \in \mathbb{R}$ is the exploration constant that balances the trade-off between exploration and exploitation. Here, exploration is

Algorithm 1: Belief Update using Unweighted Particle Filter

Input : Belief b , action a , observation o
Output: Belief b'

```

1  $b' \leftarrow \emptyset$ 
2 for  $i \leftarrow 1$  to  $|b|$  do
3   repeat
4      $x \sim b$ 
5      $(x', o', r') \sim G(x, a)$ 
6   until  $o' = o$ ;
7    $\text{addToList}(b', x')$ 
8 end for
9 return  $b'$ 

```

defined as searching in branches that have not been well-sampled, and exploitation as searching branches that appear to be promising. The UCT strategy avoids constructing the whole belief tree by only exploring the most promising parts of the tree. The Q-function $Q(b, a)$ is approximated using all the sampled episodes starting from belief node b . If the episodes are generated containing the optimal actions selected by the UCT, the approximated Q-function $\hat{Q}(b, a)$ does converge to the optimal Q-function $Q(b, a)$ [KY16].

6. Cyclically Executions and Belief Update

Once the planning time is over, the approximate optimal action is selected according to (2.19) from the constructed belief tree given the belief b . After execution of the action a , the agent receives an observation o from the environment. Then the agent updates its belief b' using a particle filter $b' = T(b, a, o)$. ABT resets the belief tree \mathcal{T} by setting the newly updated belief b' as the new root node. The process repeats cyclically.

In cases where the state space is small, it is possible to update the belief state exactly using Bayes' theorem, which is commonly employed by most POMDP planning methods [Ros+08]. However, in POMDPs with large state spaces, performing a single Bayes update for the belief state can become computationally infeasible. Moreover, obtaining a compact representation of the transition or observation probabilities in POMDPs with large state spaces may not be possible [SV10].

This thesis employs an unweighted particle filter introduced by [SV10] to approximate the belief state. The particles are updated using Monte Carlo simulation based on the generative model to sample observations, rewards, and states. While weighted particle filters are commonly used for representing belief states, applying an unweighted particle filter is advantageous when working with a generative model, as it does not necessitate an explicit model of the POMDP [SV10]. Moreover, the scalability of this approach makes it well-suited for tackling larger POMDPs.

Once the action a for the agent is chosen and an observation o is received, the belief is updated using Algo. 1. A state instance, i.e., a particle, is sampled from the current belief b by choosing a particle randomly from b . Then, the algorithm invokes the generative model $G(x, a)$ to compute a next state x' , an observation o' for the sampled particle x and selected action a . The algorithm adds the newly generated particle x' to b' when the sampled observation o' matches the real observation o .

7. Observation Matching

The ABT is applied in this thesis to the decision making of autonomous vehicles with continuous observation space. Constructing the belief tree in continuous space is difficult. Since the chance of obtaining the same real numbers from a continuous random variable is zero, the belief tree would only have one depth of the layer with infinite observations. Thus, building the belief tree deeper with continuous observations is impossible. To handle this problem, ABT approximates the continuous observation spaces by clustering observations into discrete numbers of possible observations.

The belief update, as well as the sampling of episodes in the belief tree, requires the binary comparison of two continuous observations. Each time a new observation is generated, ABT compares it with the existing observation clusters. Two observations are matched if their Euclidean distance is within the maximal distance:

$$o = o', \text{ iff } \|o - o'\|_2 \leq o_{\max}. \quad (2.24)$$

where o and o' represent two observations to be matched, and o_{\max} is the maximal Euclidean distance between observations. When the new observation can not be matched with existing observation clusters, ABT generates a new observation cluster using the new observation itself.

2.2.3 Solving Model using Deep Reinforcement Learning

Problem solving with online methods requires a generative model, which may be challenging to obtain, especially when applied to urban environments. In contrast, model-free reinforcement learning does not require knowledge of the environmental dynamics, but learns the policy by interacting within the environment. Therefore, apart from using the online POMDP solving methods, this thesis also investigates the application of model-free deep reinforcement learning for planning the driving behavior of autonomous vehicles.

DRL is a class of reinforcement learning techniques that use deep neural networks to represent the value function, policy, or environmental dynamics. The model-free approaches can be further categorized into value-based and policy-based methods. Value-based methods aim to estimate the value (or expected return) associated with each state and follow a specific action under a given policy. The value function indicates how advantageous it is for an agent to be in a particular state and encourages the agent to make decisions that result in higher cumulative rewards. Policy-based methods, on the other hand, directly learn the optimal policy without explicitly estimating value functions. The policy defines the agent's behavior and maps states to actions, with the goal of finding the best policy that maximizes the cumulative reward over time. Furthermore, there are hybrid methods that include both policy and value-based approaches, such as the actor-critic method [Haa+18].

Value-based RL methods use state value or state-action value functions to represent the policy. It represents the policy π by a state-action value function $Q(x, a)$ where the Q-function $Q(x, a)$ satisfies the Bellman equation:

$$Q(x, a) = R(x, a) + \gamma \sum_{x' \in \mathcal{X}} T(x, a, x') \max_a Q(x', a). \quad (2.25)$$

Q-learning is an off-policy value-based method that saves experiences in the form of tuples (x, a, r, x', a') in the replay buffer. Then, it uses the replay buffer to update the state-action value function:

$$Q(x, a) \leftarrow Q(x, a) + \alpha \left(r + \gamma \max_{a'} Q(x', a') - Q(x, a) \right). \quad (2.26)$$

For simple problems with small, discrete state spaces, the state-action value function $Q(x, a)$ of the Q-learning algorithm can be represented in a table. However, the table can not explicitly represent every state-action value when dealing with more complex problems that have a large number of states. In this case, the state-action value function $Q(x, a; \theta)$ can be approximated by a Deep Q-network (DQN) with weights θ [Mni+15]. An approximate solution for (2.25) can be obtained by minimizing the Temporal Difference (TD) error, which is the difference between the maximum possible value for the next state and the current prediction of the Q-value:

$$L(\theta) = \mathbb{E}_{(x,a,r,x') \sim \mathcal{D}} \left[\underbrace{\left(r + \gamma \max_{a'} Q(x', a'; \hat{\theta}) - Q(x, a; \theta) \right)^2}_{\text{TD error}} \right], \quad (2.27)$$

with \mathcal{D} denoting the replay buffer with collected experiences. For each update, mini-batches of experiences are selected from the replay buffer \mathcal{D} . $\hat{\theta}$ represents the neural network parameters of a target network. The parameters $\hat{\theta}$ of the target network are fixed for several training steps to stabilize the training process. The update to the weights is computed using gradient descent with respect to $L(\theta)$:

$$\theta \leftarrow \theta + \alpha \left(r + \gamma \max_{a'} Q(x', a'; \hat{\theta}) - Q(x, a; \theta) \right) \nabla_{\theta} Q(x, a; \theta). \quad (2.28)$$

where α denotes the learning rate.

Several methods have been introduced to further improve the performance of the deep Q-learning algorithm. Prioritized Experience Replay (PER) extends the Experience Replay mechanism by assigning priorities to experiences in the replay buffer based on their learning potential, allowing the agent to focus on more informative experiences. PER accelerates learning by frequently sampling experiences with higher TD errors, leading to better policies with fewer samples [Sch+15].

DQN tends to overestimate the value and the action-value functions due to the max operator. Double Deep Q-Networks (DDQN) address the overestimation of action values in DQN [VGS16]. It introduces a separate target network to evaluate the Q-value of the next state while the current network is used for action selection. The DDQN algorithm reduces the overestimation and improves the learning performance.

$$Q(x, a; \theta) = r + \gamma Q(x', \operatorname{argmax}_{a'} Q(x', a'; \theta); \theta'). \quad (2.29)$$

Dueling DQN separates the representation of the Q-values for each action into two components: the value function $V(x)$ and the advantage function $A(x, a)$ [Wan+16]. The total Q-value $Q(x, a)$ for a particular state-action pair (x, a) is then calculated as follows:

$$Q(x, a) = V(x) + A(x, a). \quad (2.30)$$

The state-value function estimates the value of being in a particular state regardless of the action taken. It represents how good it is for the agent to be in a given state. The advantage function estimates the additional value of taking a specific action in a given state compared to the expected value of the state. This separation allows the network to explicitly model the state's value and the relative advantages of different actions, which enables the agent to learn more effectively and perform better.

3

Behavior Planning under Spatial Occlusion with On-board Sensors

This chapter presents a behavior planning algorithm that can handle arbitrary occlusion scenarios based on the information provided by the onboard sensors of autonomous vehicles. The planner assesses collision risk by using a High-Definition map (HD map) and onboard sensors, such as cameras, radars, and LiDARs, to account for potentially occluded road users. Then, it plans driving behaviors which reduce collision risk with potentially hidden road users. A further improvement made to the behavior planner allows the planner to benefit from the information perceived by the autonomous vehicle's traffic mirror observation module. This chapter is based on the author's previously published works [Zha+21] and [Zha+22c].

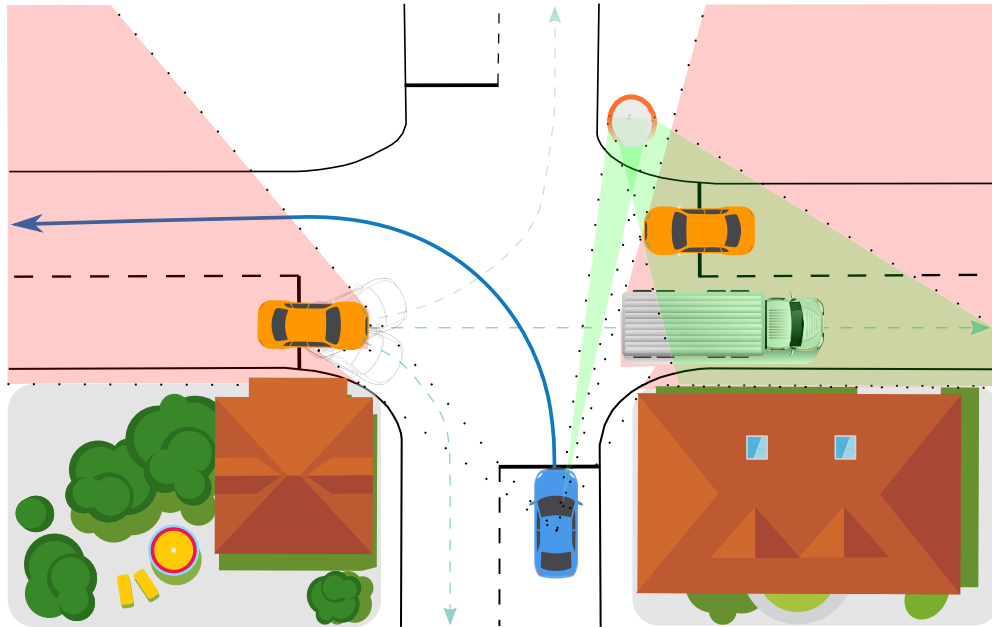
3.1 Overview and Contributions

Existing online behavior planning algorithms focusing on occlusion in intersections do not perform well in urban traffic scenarios where pedestrians are present, such as serving bus stops or approaching crosswalks (see Fig. 3.1). These scenarios are challenging because a decision-making system must consider traffic participants in the observable region and reason about the potential road users and their movement in invisible areas based on context information.

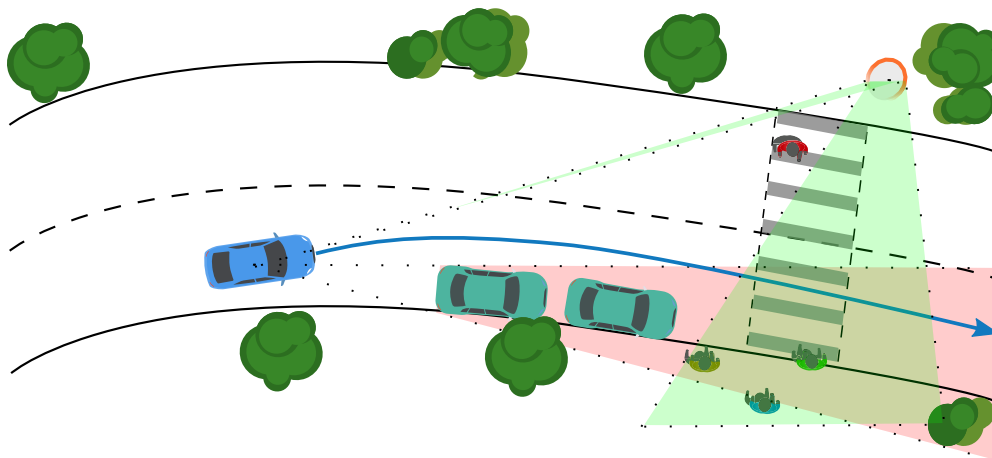
When human drivers pass through these occluded urban areas, they usually slow down and reason how likely it is that a road user will appear according to the amount of occlusion in the scenario. For example, in an occluded crosswalk shown in Fig. 3.1b, it is very likely to see pedestrians emerging and crossing the road. In this case, human drivers approach carefully until they have sufficient visibility. A similar driving behavior is also expected when leaving an occluded bus stop or facing an intersection with occluded oncoming lanes.

In complex urban areas with limited visibility, traffic mirrors are often placed by local authorities to reduce the risk of traffic accidents. A traffic mirror, also known as a security mirror or a road safety mirror allows drivers and pedestrians to better understand their surroundings and see around blind corners or other obstructions. They can be helpful at intersections, roads or parking areas where there are either natural or manufactured obstructions (see Fig. 3.1). Traffic mirrors are beneficial for humans. Utilizing traffic mirror information within autonomous vehicles could expand the FoV.

This chapter presents an occlusion-aware behavior planner by extending a POMDP planner with a phantom road users concept. The phantom road users are modeled virtual objects



(a) The ego vehicle intends to turn left at an occluded intersection.



(b) The ego vehicle drives through a partially occluded road section with a crosswalk.

Figure 3.1: Two driving scenarios in an urban environment in the presence of occlusions caused by buildings, parked cars or moving trucks. Traffic mirrors are found at the corner of an intersection (a) and on the other side of a road (b), and their information is used to increase the FoV. Red areas indicate the occluded area of the ego vehicle, whereas green areas show observed regions of the traffic mirror (graphic from [Zha+22c], ©2022 IEEE).

to represent the potential hidden road users in the occluded areas that the autonomous vehicles can not observe based on onboard sensors. First, the planner identifies risky occlusion areas along the ego navigation path based on map information. These areas can be intersections, bus stops, and crosswalks. Based on identified occlusion areas, the planner further generates potential hidden road users and infers their movements, such as vehicles in driving lanes or pedestrians at crosswalks whose intention is to cross the road. Next, the probability of phantom road users appearing outside of the occluded area is introduced. The appearance probability consists of an area-specific part that considers map information and a dynamic part that captures the change in the ego vehicle FoV in future time steps.

When a perception module for detecting traffic mirrors is available in the autonomous vehicle, a confidence modifier is modeled based on the detected traffic mirror and its observed lane or areas as well as the directions of tracked hidden road objects. The confidence modifier is then integrated into the POMDP observation model to increase or decrease the appearance probability of phantom road users in high-risk areas based on whether the tracked road users are close to or far away from the ego vehicle. The appearance probability of phantom road users is sampled for constructing the belief tree.

Furthermore, this chapter presents an active mirror perceiving method to encourage the ego vehicle to actively explore the environment and gain more information when the traffic mirror is temporarily occluded by dynamic obstacles. This method first searches the HD map for relevant traffic mirrors based on the ego mission. For every relevant traffic mirror, three-dimensional observability checks are performed between the ego vehicle position and the position of the relevant traffic mirror in the current and the future time steps while constructing the belief tree. The result of the checking is combined with the reward function to encourage the ego vehicle to keep relevant traffic mirrors visible. Finally, through the belief tree, the driving policies that maximize accumulated rewards are obtained (see Section 2.2.2).

In summary, the contributions of this chapter are as follows:

- the extension of the phantom road user concept to include pedestrians to improve the occlusion scenario coverage of a POMDP-based decision-making system,
- a context-based appearance probability method which easily incorporates context information and road topology in a POMDP to plan deadlock-free and comfortable driving behavior in the presence of heavy occlusions,
- a concept for using perceived traffic mirrors and uncertain objects tracking information as a confidence modifier,
- the combination of the confidence modifier and a phantom road user concept in a POMDP-based behavior planning to enable the autonomous vehicles to benefit from the uncertain traffic mirror information,
- an active mirror perceiving method combined with POMDP behavior planning for encouraging the autonomous vehicle to plan driving policies that maintain traffic mirror observability.
- the evaluation of the algorithm for handling multiple complex occlusion scenarios, including occluded crosswalks, bus stops (marginally studied but important for autonomous shuttle buses), and intersections in urban environments.

3.2 Related Work

The purpose of occlusion-aware behavior planning is to make the ego vehicle drive efficiently without overcautiousness while reacting safely if a road user suddenly appears from an occluded area. Recently, several techniques have been proposed to address sensor limitations and occlusions.

Reachability-based Analysis

Researchers perform reachability-based analysis with worst-case assumptions to consider the risks due to potential traffic participants in occluded areas. The general idea is to use a set of states to represent all possible configurations that a vehicle could reach. [PVN20a] compute the longitudinal speed profile for an autonomous vehicle with path-velocity decomposition. Occluded vehicles are incorporated as dynamic constraints generated with worst-case scenarios by modeling virtual vehicles with infinite length and maximum speed. Similarly, [Nau+19a] perform reachability analysis to analyze the safety of passing through a potential conflict area with occluded vehicles. The occurrence probability is used as a threshold. If it is sufficiently low, the uncomfortable emergency stop is also acceptable. [WLS20a] applies a set of particles to represent potential configurations of occluded vehicles in a region of interest and analyzes the collision risk with a predicted visibility range in future time steps. Then the predicted visibility risk is combined with a cost-based planner to plan the acceleration for the autonomous vehicle. Reachability analysis can prove safety but also results in conservative driving behavior in some special cases with very limited visibility [PVN20a]. In this case, carefully advancing into a conflict zone is necessary to gather more information instead of freezing.

Learning-based Methods

Contrary to worst-case assumptions, learning-based methods focus on automatically learning complex driving strategies from data without hand-coded rules. Reference [Ise+18] applies a DQN to learn the policy in an unsignaled intersection with static occlusion. The state space is described as an occupancy grid map to separate drivable regions and obstacles. [Bou+19] focuses on safe reinforcement learning using a model-checker to identify safe actions from the action space. Potential incoming road users are modeled via additional state variables. [Kam+20] handles occluded intersections using a risk-aware DQN, which incorporates risk evaluation within the reward function instead of only considering collisions. A vehicle with maximum allowed velocity is assumed if the intersection is occluded. These approaches give ideas for applying learning-based methods to generate driving behaviors. However, an approach providing both safe and robust driving behavior in the presence of unseen scenarios or corner cases when observed data varies slightly from training data still needs to be further investigated.

Probabilistic Models

Another planning category focuses on probabilistic models, which integrate uncertainty due to visibility limitation in the POMDP model. Previous works in this area can be classified into two groups depending on whether the policies are solved offline or online. As introduced in Chap. 2, the offline approaches calculate the approximated optimal policies over the entire state space for an arbitrary initial belief. [Sch+19b] combines a rule-based autonomous emergency braking system with POMDP to obtain less conservative driving behavior for a

pedestrian collision system under sensor occlusion. In [Bou+18a], static occlusion in a crosswalk and at a simple T-junction is considered in an offline POMDP. A scene decomposition method is introduced by treating each road user independently to improve the scalability of the POMDP with multiple road users. However, the authors do not consider occlusions due to dynamic moving objects. In addition, a state space model encompassing road users, map geometries, and traffic rules must be solved before using POMDP online, which makes the POMDP model more difficult to solve and restricts its application to a highly dynamic urban environment. By contrast, online POMDP solves the problem before execution by constructing belief trees with a reachable set of states for the current belief within a limited time horizon. In [Lin+19b] and [Sch+19a], phantom vehicles are placed on the edge of invisible areas at unsignalized intersections and treated as real vehicles in the planning phase. [Hub+19] considers traffic density and models the occurrence probability for phantom vehicles. The abovementioned approaches can handle occluded scenarios arising from static and dynamic objects but are limited to unsignalized intersections.

Traffic Mirror Perceiving Algorithms

In addition to occlusion-aware decision-making and motion planning methods, studies are focusing on improving occlusion-awareness in the prediction module of the autonomous vehicle. Formal set-based prediction is used to predict a set of occupancies for both detected and occluded road users [OML18; KA20]. Reference [Koç+21a] utilizes contextual information to estimate the emergence probabilities of the hidden pedestrians. Furthermore, some researchers have proposed methods for perceiving traffic mirrors and tracking dynamic objects based on detected traffic mirrors. Based on camera images, either traditional computer vision methods such as Gaussian filters [KYS13] or Convolutional Neural Network (CNN)-based approaches [Dha+19], [FEN+20] are used to detect traffic mirrors. The Kalman filter and optical flow are used for tracking objects moving direction, i.e., whether objects are approaching or receding from the observer [KYS13; FEN+20]. According to the research, no prior work investigates how to incorporate perceived uncertain traffic mirror information into the decision-making method for autonomous vehicles in the presence of occlusions.

3.3 Approach: Occlusion Handling with Onboard Sensors

The state-of-the-art solutions showed the progress of the decision-making system under visibility limitations. However, sophisticated occlusion scenarios, such as bus stops and crosswalks where lots of vulnerable road users could suddenly emerge, are less discussed but important for autonomous shuttle buses (see comparison listed in Table 3.1). This chapter aims to improve the occlusion scenarios coverage by proposing a POMDP-based behavior planner that considers phantom vehicles and pedestrians using onboard sensors. Furthermore, the proposed planner is extended to incorporate uncertain tracking information from traffic mirrors to improve its occlusion handling ability.

This section first introduces how to use the POMDP formulation to model a driving scenario by defining the state, observation, and action spaces. Following that, this section explains the observation model, which consists of observing real traffic road users as well as the context-based appearance probability of potentially occluded road users such as vehicles and pedestrians. Furthermore, an observation model for traffic mirrors is introduced. The model calculates a confidence modifier based on the object detection result from available detected traffic mirrors to influence the appearance probability of hidden road users. Then, the modification of context-based appearance probability with a confidence modifier is explained in

Table 3.1: Comparison of existing occlusion-aware planning methods with the proposed approach (table from [Zha+21], ©2021 IEEE).

Source	Static obstacles	Dynamic obstacles	Vehicles	Pedestrians	Intersections	Crosswalks	Bus stops
[PVN20a]	✓		✓		✓		
[Nau+19a]	✓	✓	✓		✓		
[WLS20a]	✓	✓	✓		✓		
[Ise+18]	✓		✓		✓		
[Bou+19]	✓		✓	✓	✓	✓	
[Kam+20]	✓		✓		✓		
[Sch+19b]	✓			✓		✓	
[Bou+18a]	✓		✓	✓	✓	✓	
[Lin+19b]	✓	✓	✓		✓		
[Sch+19a]	✓	✓	✓		✓		
[Hub+19]	✓	✓	✓		✓		
This chapter	✓	✓	✓	✓	✓	✓	✓

detail. Finally, with a further definition of the transition and reward model, the POMDP is solved using the algorithm explained in Section 2.2.2.

3.3.1 Environment Representation

Fig. 3.2 A and B illustrate how the urban driving environment is modeled using the state and observation space of the POMDP model. The definition of each space is explained in the following.

3.3.2 State and Observation Space

The environment contains the ego vehicle N_{ego} , other road users N_i , where $i \in [1, \dots, N]$, $N \in \mathbb{N}^+$, and the generated phantom road users N_k with $k \in [N + 1, \dots, K]$, $K \in \mathbb{N}^+$. The state of the ego vehicle is defined as: $x_{ego} = [x_{ego}, y_{ego}, \theta_{ego}, v_{ego}, r_{ego}]^T$, which includes the position (x_{ego}, y_{ego}) in Cartesian coordinates, orientation θ_{ego} , speed v_{ego} and intended driving path r_{ego} . The ego vehicle drives along its intended driving path r_{ego} , which is extracted from the lane network of the HD map. The state of other objects x_i is described similarly as $x_i = [x_i, y_i, \theta_i, v_i, r_i]^T$, and the state $x_k = [x_k, y_k, \theta_k, v_k, r_k]^T$ represents the state of phantom road users. The other road user N_i moves along its intended path $r_i \in \{r_1, \dots, r_I\}$ with $I \in \mathbb{N}$. The intended path represents the true intention for a set of possible behaviors. For example, before entering an intersection, a road user has one true intention from the possible intentions: going straight, turning left, or turning right. These intentions are represented as hypothetical paths. The intended path r_i is a hidden state that can only be estimated over time. The state is represented by combining the ego vehicle state, other road user states, and

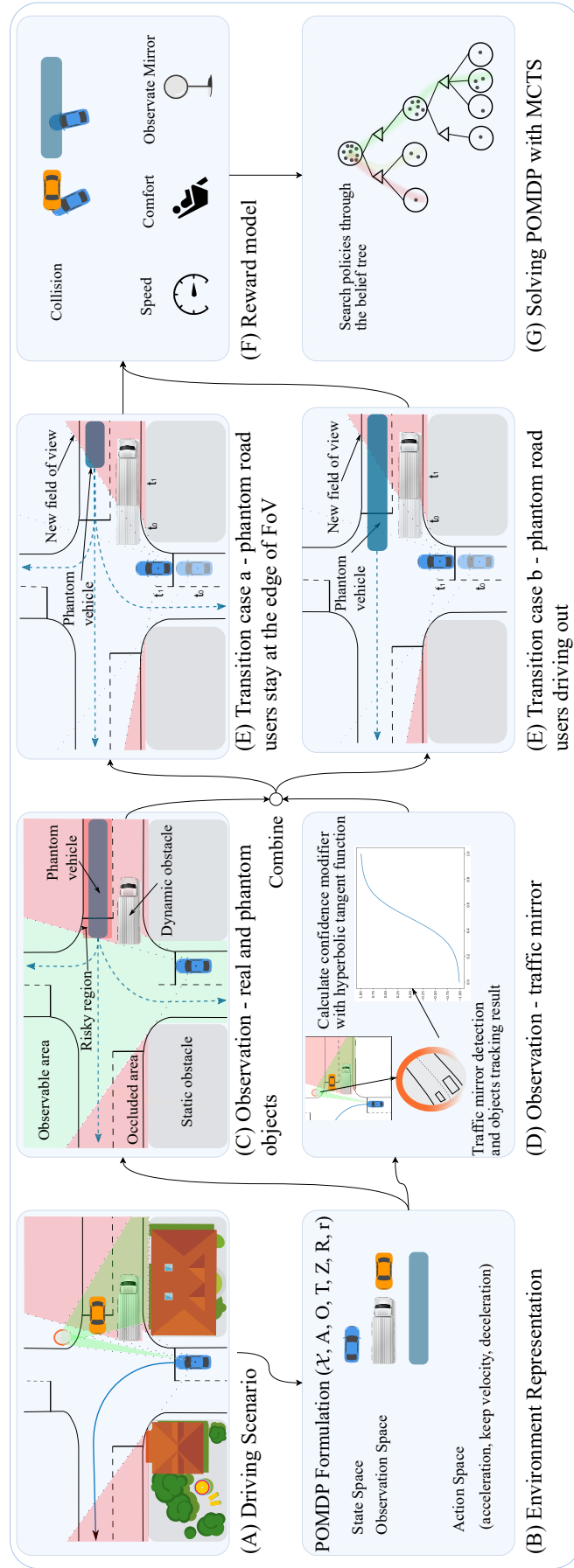


Figure 3.2: The process of the POMDP behavior planner using onboard sensors (graphic from [Zha + 22c], ©2022 IEEE).

phantom road user states:

$$\mathcal{X} := [\mathcal{x}_{ego}, \mathcal{x}_1, \dots, \mathcal{x}_N, \mathcal{x}_k, \dots, \mathcal{x}_K]^T. \quad (3.1)$$

3.3.3 Action Space

The design goal of the action space is to use as few actions as possible to represent a wide variety of behaviors, such as slowing down, stopping in front of a crosswalk, and accelerating to drive through a junction. To achieve that, a discrete set of acceleration values is applied to represent the maneuvers acceleration, keep velocity, deceleration, respectively: $A = \{+1.5 \text{ m/s}^2, 0 \text{ m/s}^2, -1.5 \text{ m/s}^2\}$:

3.3.4 Observation Model for Real Road Users

All the observable variables in the observation space can be directly updated from sensor measurements, including the position and velocity as well as the orientation of the ego vehicle and other real traffic participants. The noise of sensor measurements can also be considered during the update of observation. Unknown intentions are inferred by the autonomous vehicle's prediction module, which is updated each time after receiving new measurements.

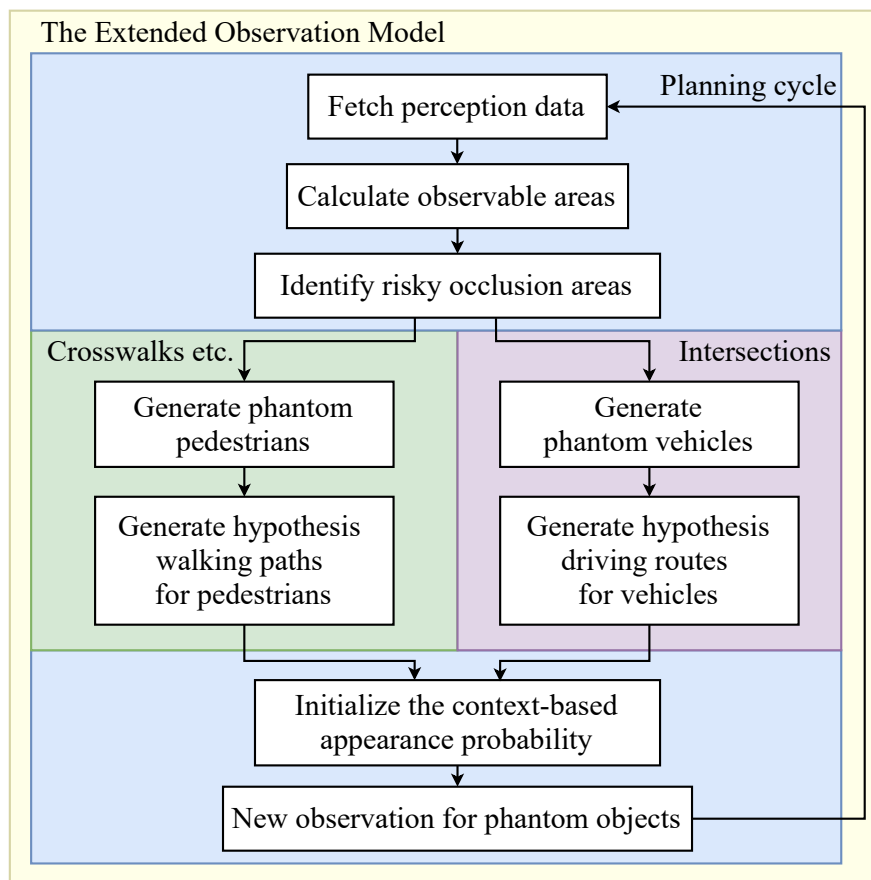


Figure 3.3: The extended observation model for belief update of phantom road users (graphic from [Zha+21], ©2021 IEEE).

3.3.5 Observation Model for Phantom Road Users

Phantom objects are defined as objects that cannot be observed. This chapter introduces an extended observation model for the generation of phantom vehicles and pedestrians (see Fig. 3.3). Since phantom objects can suddenly emerge and lead to potentially dangerous situations, their possible occurrence from the unobservable area needs to be inferred. To model the phantom road users, two steps are taken: the creation of phantom road users and the estimation of their appearance probability.

Generation of Phantom Road Users

The first step is to generate phantom road users. It begins with calculating observable areas based on static and dynamic obstacle information obtained from the perception module. The observable area is first initialized with the maximum range of the perception system. Then, the static and dynamic obstacles limit this observable area represented by a polygon. Next, potential risk areas in the map along the upcoming navigation path of the ego vehicle are sought from the occlusion catalog, which contains risk occlusion scenarios that require handling. For example, in a structured intersection without a traffic light, prioritized occluded lanes are selected over the lanes of the ego vehicle. Moreover, unobservable areas on or near crosswalks and bus stops, where pedestrians have a higher appearance probability, are taken into account.

After selecting the occluded risk areas, the phantom traffic participants are placed on each edge of the FoV (see Fig. 3.2 C). The lengths of the phantom objects are defined to be infinite, which enables us to represent a set of reachable states using only one configuration of phantom objects. Their potential paths also need to be determined. All following lanes for phantom vehicles in the intersection are extracted from the map as candidate paths. For the phantom pedestrians, pseudo priority walking paths are generated. The walking paths consist of waypoints starting at the occlusion edge and point to the other side of the road.

Context-based Appearance Probability

The next question that needs to be addressed is the probability of a phantom object entering the observable area. Assuming that the phantom object always appears at each planning cycle would cause the ego vehicle to drive overcautiously. In some cases, this assumption could even block the ego vehicle and lead to a “freezing state” without any further movement. The idea is to incorporate context information into the appearance probability:

$$p_a(d, \Delta s) := \min((p_{env}(d) + p_{FoV}(\Delta s)), 1). \quad (3.2)$$

The min operator is employed to guarantee an appearance probability $p_a(d, \Delta s) \leq 1$. The context-based appearance probability $p_a(d, \Delta s)$ consists of two parts. The first part $p_{env}(d)$ represents the environmental context (see Eq. 3.3). K_{env} is defined as an initial environmental probability to reflect the phenomenon that the appearance of road users depends on where the occlusion occurs. For example, the probability of pedestrians appearing at crosswalks is greater than that on ordinary roads. Similarly, pedestrians are more likely to appear around bus stops and school areas. If the occluded area is far away from a crosswalk, the appearance probability is small. Hence, a distance threshold D_s is introduced. The distance d from the start point of the phantom object to this risky region is included in a discount factor, which means that the area-specific appearance probability is no longer considered beyond a certain distance threshold D_s .

$$p_{env}(d) := \max\left(\left(K_{env} \frac{D_s - d}{D_s}\right), 0\right), \quad (3.3)$$

The second part of the probability $p_{FoV}(\Delta s)$ describes the probability of phantom objects' appearance due to a change in the FoV. The FoV can change during forward simulation in each time step. If the FoV is enlarged, the chance that the ego vehicle observes a phantom object previously hidden by obstacles increases. Inspired by [Hub+19], this thesis assumes that phantom vehicles and pedestrians are normally distributed in the occluded lane or hypothesis path. L is defined as the length within which the expectation is to observe exactly one phantom object. With the increase in the FoV of Δs meters, the probability for at least one phantom object to appear is $\frac{\Delta s}{L}$. In case the FoV decreases or remains constant, no additional probability of observing phantoms is assigned.

$$p_{FoV}(\Delta s) := \begin{cases} 0, & \text{if } \Delta s \leq 0, \\ \frac{\Delta s}{L}, & \text{if } \Delta s > 0 \wedge \Delta s < L, \\ 1, & \text{otherwise.} \end{cases} \quad (3.4)$$

3.3.6 Observation Model for Traffic Mirror

Confidence Modifier for Existing Object probability

The perception module of an autonomous vehicle system provides observation of traffic mirrors as well as object tracking information. Some research focuses on improving estimation and tracking results based on camera data [Dha+19], [FEN+20], [KYS13], whereas the primary focus of this chapter is on utilizing uncertain traffic mirror information for behavior planning. Ideally, a perception system should be able to observe the traffic mirror's location and match it within the map. Furthermore, it would provide perceived dynamic road users with detailed information such as position, velocity, directions etc. In this case, considering these tracked objects in the behavior planning module is straightforward since the objects can be modeled in the state space. However, tracking and providing dynamic road users' position, velocity, and direction through traffic mirrors based solely on camera images is difficult, because traffic mirrors are typically small in size, and minor uncertainties in mirror position and orientation have a large impact on the tracking result.

Instead of overloading the perception system to provide this information, this chapter proposes a concept that will reduce the difficulty of perceiving and tracking objects from traffic mirrors, as illustrated in Fig. 3.2 D. The concept is comprised of three steps. In the first step, the location of the traffic mirror and all its observing lanes or risk areas, such as crosswalks are stored in the HD map. The location of the traffic mirror and its associated observing lanes are then provided online to the perception and behavior planning module in the second step. The relevant traffic mirror regarding the ego vehicle's navigation path and the information of which lane it observes is identified. In the next step, the perception module is responsible for estimating whether objects move close to the ego vehicle or cross the risk area based on the camera data. The detection result is defined as the detection probability $0 \leq p_m \leq 1$, which denotes the confidence of an object that approaches the observation area of the traffic mirror.

Modification of Phantom Object's Appearance Probability with Traffic Mirror Observation

In this step, the presented planner calculates a confidence modifier $p_{cm} \in [-1, 1]$ according to the detected probability p_m using a hyperbolic tangent function,

$$p_{cm} = \tanh(5 \cdot (p_m - 0.5)). \quad (3.5)$$

The confidence modifier p_{cm} is applied to enlarge or reduce the context-based appearance probability p_a :

$$p_{a_modified} = \max(0, \min(1, p_a(d, u) + p_{cm})). \quad (3.6)$$

With this extension, the information additionally provided from the traffic mirror detection module can be taken into account. If the detection probability is high, the confidence modifier p_{cm} is close to 1. The modified appearance probability $p_{a_modified}$ can be increased up to 1. The opposite occurs if the detection module does not detect any road users in the hidden area. The confidence modifier calculated by Eq. (3.5) is around -1 , which will reduce the $p_{a_modified}$ to 0. Another benefit with this extension is that, as long as the detection is very uncertain, p_m is around 0.5. The confidence modifier p_{cm} results in 0, which does not influence the context-based appearance probability p_a .

3.3.7 Transition Model

The intended ego path remains unchanged such that $r'_{ego} = r_{ego}$. A point mass dynamic model is applied in Eq. 3.7 as the motion model of the ego vehicle to update the new state along with the ego mission path in Frenet coordinates [Wer+10], where s'_{ego} denotes the new location of the ego vehicle along the mission path r'_{ego} of the ego vehicle and v'_{ego} is the updated ego velocity.

$$\begin{bmatrix} s'_{ego} \\ v'_{ego} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} s_{ego} \\ v_{ego} \end{bmatrix} + \begin{bmatrix} \frac{1}{2}\Delta t^2 \\ \Delta t \end{bmatrix} a. \quad (3.7)$$

For the transition model of other real road users, motion predictions from the prediction model of the ego vehicle are employed based on the corresponding time step.

The state transition of phantom objects depends on the sampled result of the final appearance probability $p_{a_modified}$ after considering the traffic mirror detection. A sample is drawn according to $p_{a_modified}$ each time when the transition of phantom objects is needed. When the sample result is zero, the phantom road user does not appear from the occlusion area. It is updated at the edge of the new FoV in the next time step, as shown in Fig. 3.2 E case a. When the result is one, indicating that the phantom object comes out from the occluded region (see Fig. 3.2 E case b), a constant velocity model is applied:

$$s'_k = s_k + v_k \cdot \Delta t, \quad (3.8)$$

in this context, v_k represents the velocity of the phantom road user N_k . When N_k is a vehicle, v_k is set according to the speed limit of the respective driving lane, as defined in the HD map. For phantom road users that are pedestrians, v_k is assigned a value of 1.25 m/s, reflecting the normal walking speed of pedestrians. The phantom road user will only move forward along the path within a planning cycle as long as it is outside of the occlusion region.

3.3.8 Reward Function

The reward function of the approach includes the objectives safety, speed, comfort, and mirror observation:

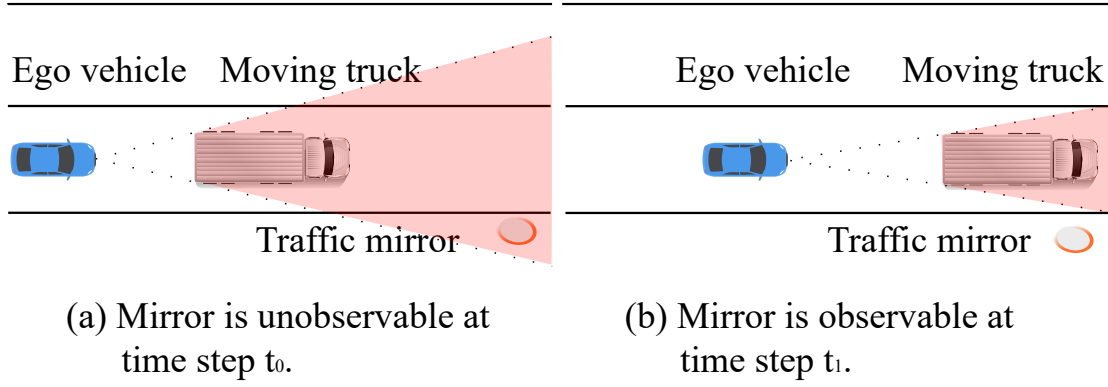


Figure 3.4: (a): Ego vehicle cannot observe the mirror, since the mirror is occluded by the moving truck. (b): Ego vehicle is able to observe the mirror (graphic from [Zha+22c], ©2022 IEEE).

$$R = R_{\text{collision_real}} + R_{\text{collision_phantom}} + R_{\text{speed}} + R_{\text{comfort}} + R_{\text{observation}}. \quad (3.9)$$

Several simulations are conducted to determine weights for the reward function.

Rewards Regarding Safety, Speed, and Comfort

To consider safety, a large negative reward $R_{\text{collision_real}} = -100000$ is assigned when the ego vehicle has collisions with other traffic participants. The collision with phantom road users is penalized with a different negative reward $R_{\text{collision_phantom}} = -10000$.

The ego vehicle is also encouraged to maintain the desired velocity v_{desired} following its mission:

$$R_{\text{speed}} = \begin{cases} -200 \cdot (v_{\text{desired}} - v_{\text{ego}}), & \text{if } v_{\text{desired}} \geq v_{\text{ego}} \\ -2000 \cdot |v_{\text{desired}} - v_{\text{ego}}|, & \text{otherwise.} \end{cases} \quad (3.10)$$

To obtain comfortable driving policies, changing acceleration is penalized with $R_{\text{comfort}} = -300 \cdot a^2$.

Reward for Active Traffic Mirror Perceiving

An active mirror perception method is introduced to encourage the ego vehicle to keep observing the traffic mirror. The idea is that the ego vehicle has access to the HD map and thus knows the position of all relevant traffic mirrors along the ego navigation path. Algo. 2 shows the process of traffic mirror observability check. A polygon between the ego vehicle and the traffic mirror (as shown in Fig. 3.2 A), is built for every relevant traffic mirror when the ego vehicle approaches it (line 1 to 2). Three-dimensional checks are performed using the polygon and bounding boxes from all static and dynamic objects (line 3 to 8). Finally, the checking result f_o indicates whether the traffic mirror is observable. Fig. 3.4 shows an example of the traffic mirror which is occluded by a dynamic truck in the first time step and is observable by the ego vehicle in the next time step. The check result f_o is considered in the active mirror perceiving reward $R_{\text{observation}}$:

$$R_{\text{observation}} = \begin{cases} -500 \cdot v_{\text{ego}}, & \text{if } f_o = \text{FALSE} \ \& \ v_{\text{ego}} > 5\text{m/s}, \\ 0, & \text{otherwise.} \end{cases} \quad (3.11)$$

Algorithm 2: Traffic Mirror Observability Check

```

Input : Current ego states  $x_{ego}$ , Next ego states  $x'_{ego}$ , Relevant traffic mirror list  $M_l$ ,
          Object state  $x = [x_1, \dots, x_N]^T$ 
Output: Traffic mirror observation flag  $f_o$ 
1 foreach relevant traffic mirror  $m_l \in M_l$  do
2    $bb_{ego} \leftarrow \text{egoToMirrorPolygon}(x'_{ego}, m_l)$ 
3   foreach object  $i \in \{1, \dots, N\}$  do
4      $bb_i \leftarrow \text{buildBoundingBox}(x_i)$ 
5     if  $\text{isIntersecting}(bb_{ego}, bb_i)$  then
6       return FALSE
7     end if
8   end foreach
9 end foreach
10 return TRUE

```

3.4 Experiments and Results

This section evaluates the presented approach in a proprietary simulator under various challenging occlusion scenarios, including crosswalks, bus stops, and intersections. To eliminate the influence of other road users' intelligent behavior in the evaluation, their behavior is controlled via predefined behaviors that do not consider collision avoidance. The velocity of all road users in the evaluation is chosen to compare different planning behaviors.

The evaluations are divided into two parts. The first part assesses the occlusion-aware POMDP planner, which takes into account phantom road users. The second part demonstrates the performance of the traffic mirror-aware POMDP planner, an extension of the occlusion-aware POMDP planner that incorporates an observation model for the traffic mirror (see Section 3.3.6).

3.4.1 Evaluation of the Occlusion-aware POMDP Planner

Experiment Setup

The performance of the occlusion-aware POMDP planner is compared against two other strategies. The presented generic occlusion-aware POMDP behavior planner is denoted as **GO-POMDP**. An **Omniscient** planner represents a ground truth planner, which has access to

Table 3.2: Parameters applied in evaluation (table from [Zha+21], ©2021 IEEE).

Parameter	Value	Parameter	Value
Planning Frequency	2 Hz	Planning Horizon	10 s
Discount Factor γ	0.95	Maximal Tree Depth	10
Scenario A: $V_{pedestrian1}$	1.5 m/s	$V_{pedestrian2}$	1.0 m/s
Scenario B: $V_{pedestrian1}$	0.8 m/s		
Scenario C: V_{truck}	5.0 m/s	$V_{vehicle}$	6.0 m/s

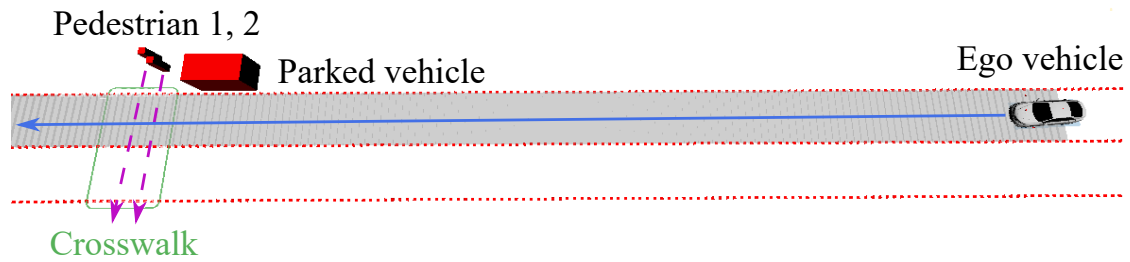


Figure 3.5: Ego vehicle driving through an occluded crosswalk. Two pedestrians are about to cross the crosswalk, but their view is blocked by the parked car (graphic from [Zha+21], ©2021 IEEE).

all available environmental information. Finally, a POMDP planner (**WO-POMDP**) is established with the worst-case assumption that phantom objects will always appear from occluded areas with maximal allowed velocity. The worst-case assumption has been widely applied in other studies [PVN20a], [Lin+19b]. Thus, it is considered a baseline approach for comparison with the presented approach. The parameters are listed in Table 3.2. Additional scenarios are presented in the supplementary video¹.

Occlusion in Crosswalk

The first scenario (see Fig. 3.5) is a crosswalk that is partially occluded by a parked vehicle. Two pedestrians in the occluded area will cross the road when the ego vehicle approaches near the crosswalk. The result in Fig. 3.6 shows that Omniscient planner adjusts its velocity when approaching the crosswalk because it fully observes the pedestrians. It can also be observed that Omniscient planner behaves aggressively as it starts to accelerate as soon as the pedestrians leave the conflict area, i.e., walk toward the other lane of the road.

Similarly, GO-POMDP reduces its velocity to approach the crosswalk carefully due to occlusion awareness. Because of limited sight in the crossing area, the ego vehicle moves forward at a very low speed. After the pedestrians appear, it first maintains its speed since it still has sufficient safe distance to the pedestrians. At time $t = 20$ s, the ego vehicle comes to a halt and lets the pedestrians pass first. The yellow and green lines of GO-POMDP in Fig. 3.6 indicate that the ego vehicle is waiting for pedestrians to leave the road completely. This is because the moving pedestrians may block the view of the ego vehicle to observe pedestrians coming from the other direction, which has been considered as a risk in GO-POMDP. WO-POMDP has similar driving behavior when approaching the crosswalk. However, due to the limited visibility, the ego vehicle will not continue to drive under the worst-case assumption that 100% of pedestrians will appear from the occluded area. This “freezing state” has also been reported in the previous study [PVN20a].

¹Video: <https://github.com/GitChiZhang/GO-POMDP>

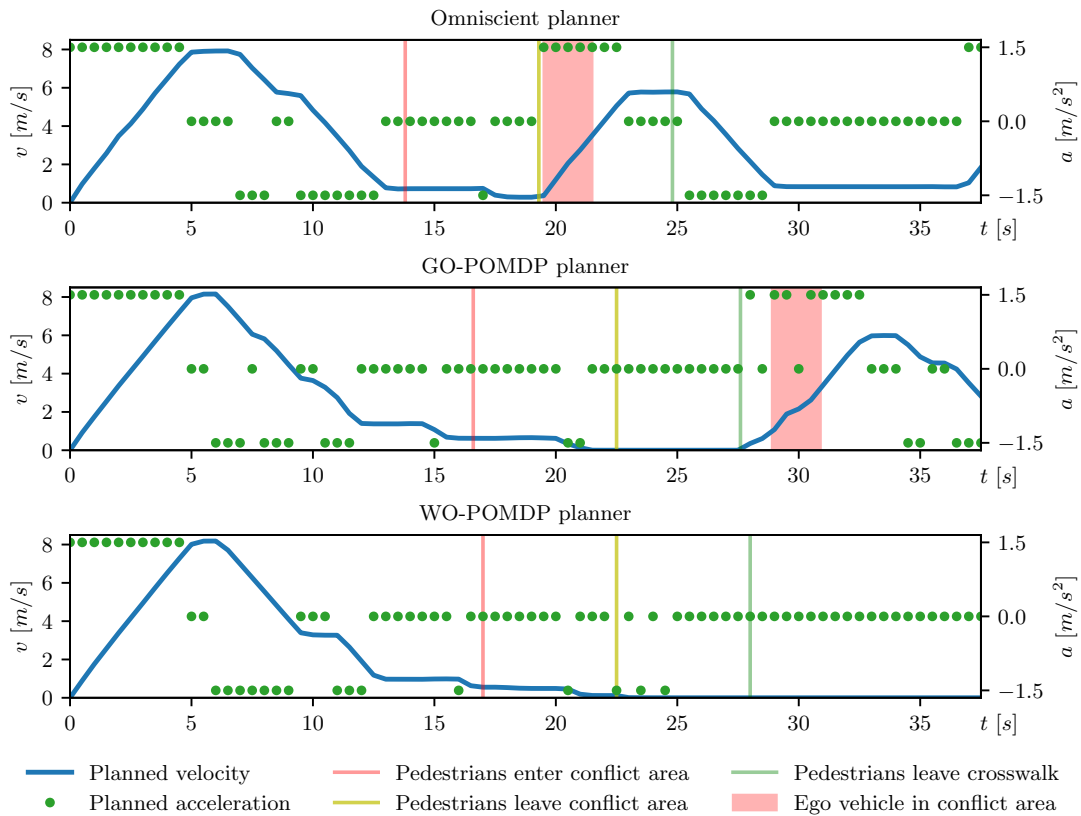


Figure 3.6: Planned velocity and acceleration profiles for crosswalk scenario with two occluded pedestrians (graphic from [Zha+21], ©2021 IEEE).

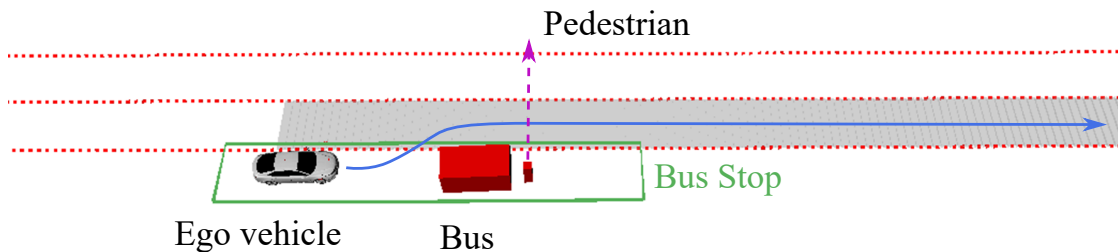


Figure 3.7: Ego vehicle is leaving the bus stop while a pedestrian is about to cross the road in front of a bus (graphic from [Zha+21], ©2021 IEEE).

Occlusion in Bus Stop

The second occluded scenario (Fig. 3.7) is at a bus stop. While the ego vehicle is about to leave the bus stop, a passenger wants to cross the road in front of the bus after exiting it. This is also a challenging scenario as the ego vehicle can't detect this pedestrian before it appears on the ego path.

The results in Fig. 3.8 show that Omniscient planner drives faster than the other two planners through the bus stop since it has full knowledge of the occluded road user. GO-POMDP performs very well in terms of avoiding a collision with the pedestrian. After the pedestrian leaves the high-risk area, GO-POMDP continues to drive with a low velocity to increase the visibility while driving through this area. In contrast, WO-POMDP once again leads to a deadlock situation due to overcautiousness.

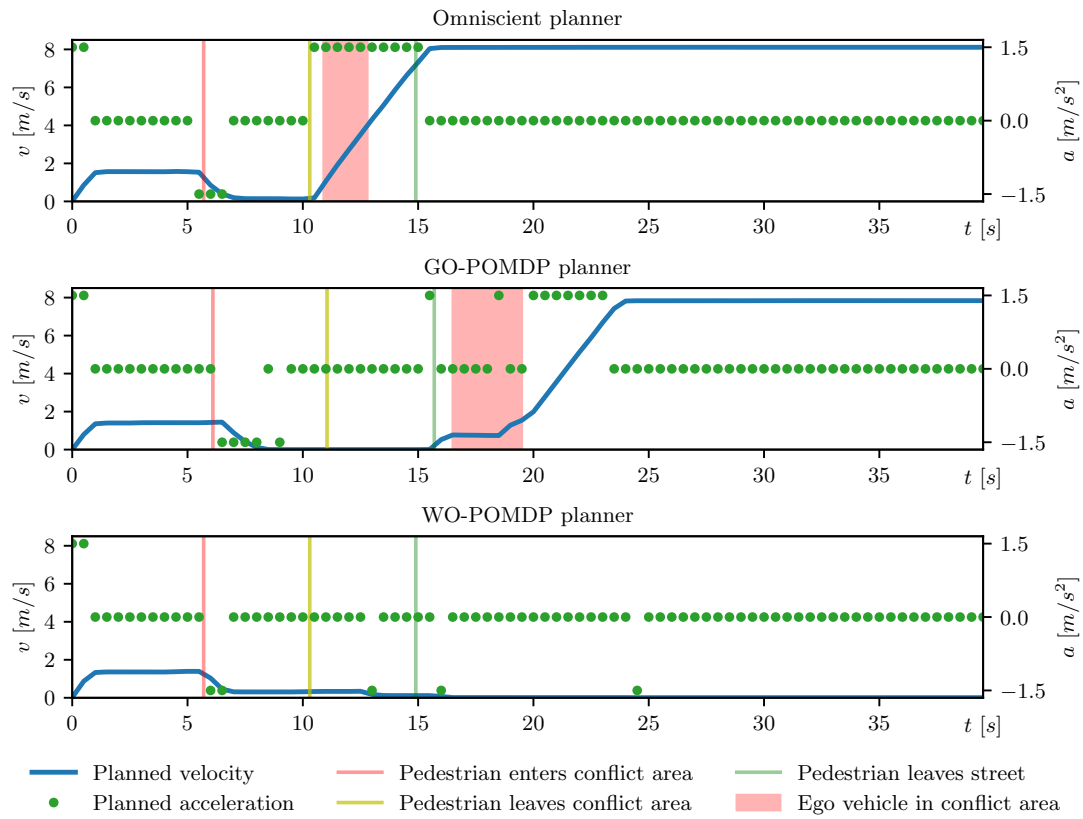


Figure 3.8: Planned velocity and acceleration profiles for bus stop scenario with an occluded pedestrian (graphic from [Zha+21], ©2021 IEEE).

Occlusion in Unsignalized Intersection

Finally, the occlusion-aware POMDP planner is demonstrated at an unsignalized intersection with a left-turn maneuver under dynamic occlusion (see Fig. 3.9). A moving truck arrives at the intersection before the ego vehicle and prevents the ego vehicle from observing another vehicle that has priority.

Fig. 3.10 shows that Omniscient planner waits for the vehicle on the prioritized lane after the moving truck has crossed the intersection. It can also be seen that GO-POMDP performed nearly as well as Omniscient planner. From $t = 10$ to 13 s, it slows down and performs creeping behavior to observe whether there is a vehicle on the priority lane occluded by the truck. Once the other vehicle appears and has left the conflict area, GO-POMDP accelerates slightly and enters the intersection after having sufficient visible area. WO-POMDP behaves more cautiously than the introduced approach.

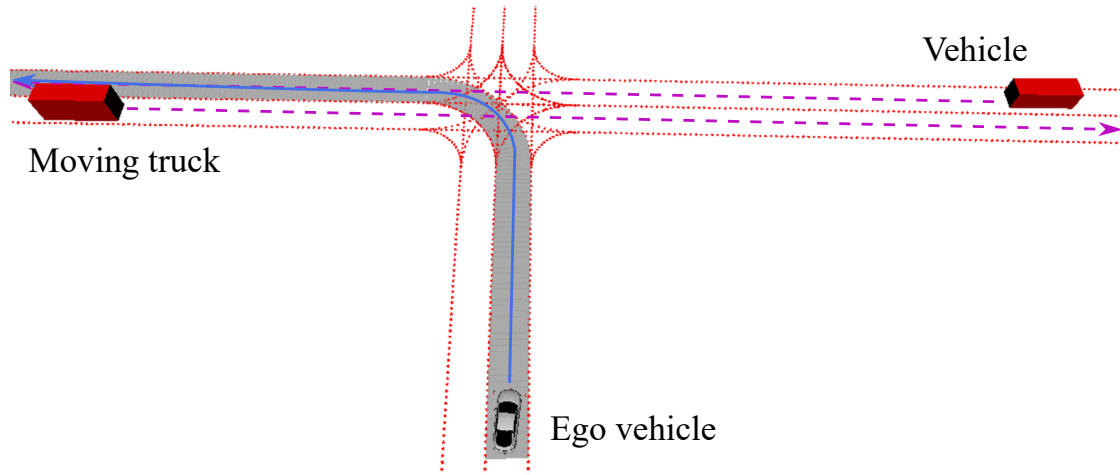


Figure 3.9: Left turn in an unsignalized intersection scenario with dynamic occlusion due to a moving truck (graphic from [Zha+21], ©2021 IEEE).

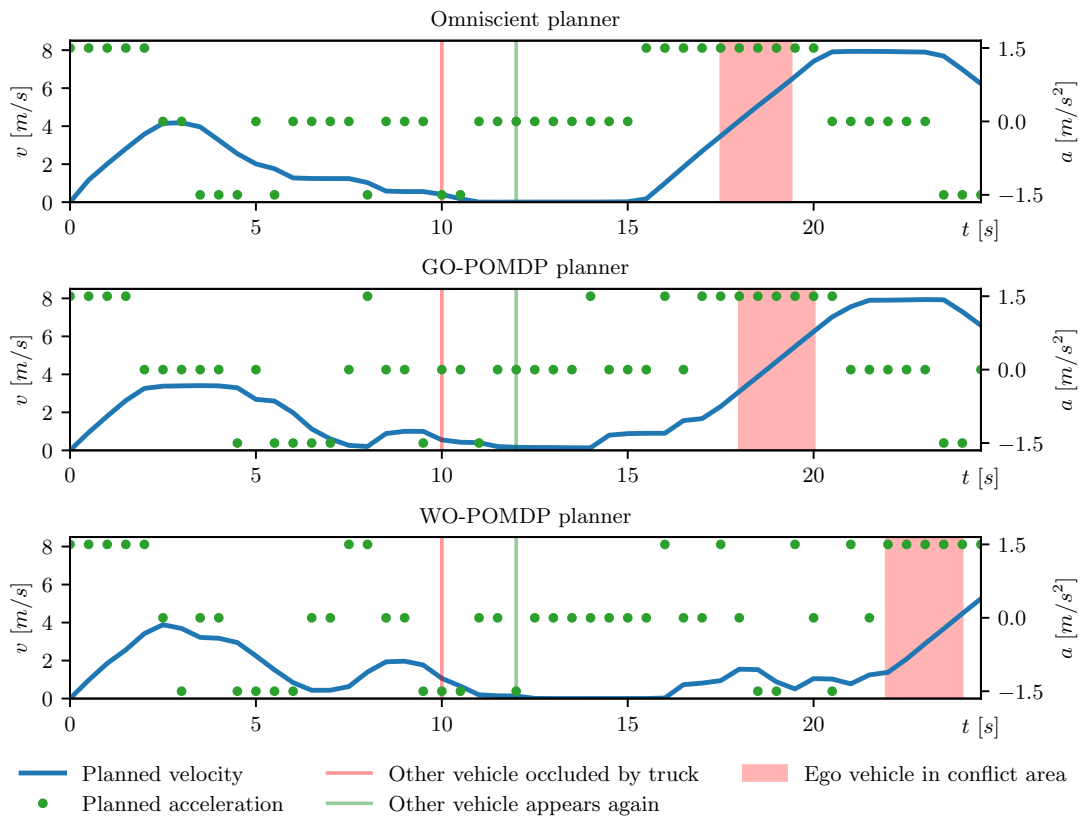


Figure 3.10: Planned velocity and acceleration profiles for unsignalized intersection scenario (graphic from [Zha+21], ©2021 IEEE).

3.4.2 Evaluation of the Traffic Mirror-Aware POMDP Planner

This section assesses the performance of the traffic mirror-aware POMDP planner. In the simulator, a simple perception module is established for observing simulated traffic mirrors, which provide information about observed lanes as well as the probability of oncoming objects on the observed lanes.

Table 3.3: Applied parameters in the simulation (table from [Zha+22c], ©2022 IEEE).

Parameter	Value	Parameter	Value
Scenario B: $V_{vehicle}$	6.5 m/s		
Scenario D: $V_{pedestrian1}$	2.0 m/s	$V_{pedestrian2}$	2.0 m/s
Scenario E: $V_{leading\ vehicle}$	5.0 m/s		

Experiment Setup

The introduced traffic mirror-aware POMDP behavior planner is denoted as **GT-POMDP**. Another version of the planner (**GTM-POMDP**) is configured that also takes into account whether the ego vehicle can observe the traffic mirror by including a reward for encouraging traffic mirror observation.

The presented approach is compared with two other approaches. The **GO-POMDP** from the previous section is a POMDP-based behavior planner that can handle intersections and crosswalks regarding potentially occluded vehicles and pedestrians but cannot use the information provided by the traffic mirror, which is considered as a baseline method in this section. As the ground truth, another strategy, an **Omniscient** planner is established, which has access to all environmental information, including all occluded traffic participants. The parameters applied in the simulation are chosen to compare different planning behaviors (see Table 3.3). Evaluation results are recorded in video².

Occlusion in Unsignalized Intersection

In scenario A shown in Fig. 3.11, the ego vehicle intends to turn left in an unsignalized intersection occluded by a building. A traffic mirror is placed at the street that point to the occluded lanes. In scenario B, a moving vehicle driving toward the intersection in the occluded area is set up to evaluate how the planners react to vehicles that suddenly appear.

²Video: <https://github.com/GitChiZhang/GT-POMDP>

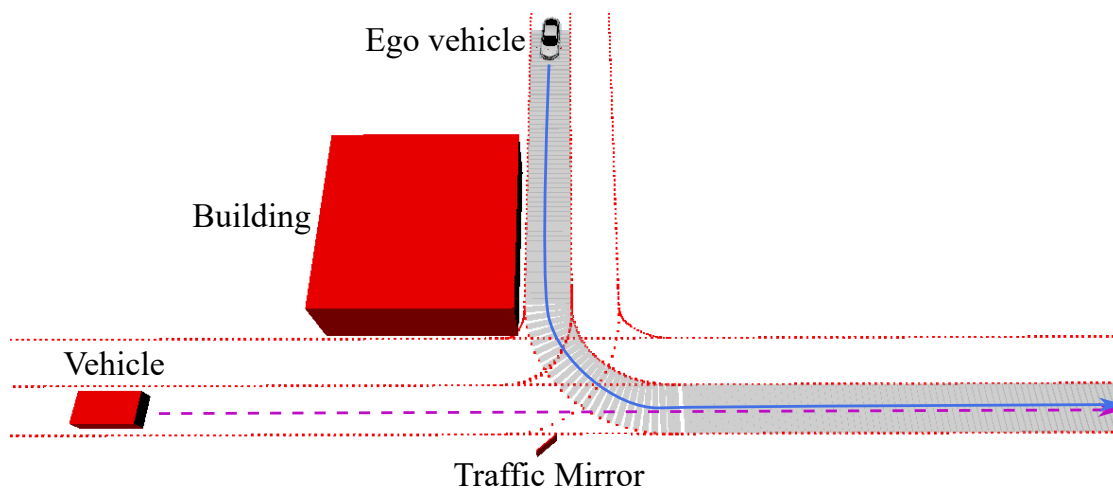


Figure 3.11: Scenario A: The ego vehicle intends to turn left in an empty unsignalized intersection with occlusion caused by a building. Scenario B: Additionally, a dynamic vehicle is approaching the intersection (graphic from [Zha+22c], ©2022 IEEE).

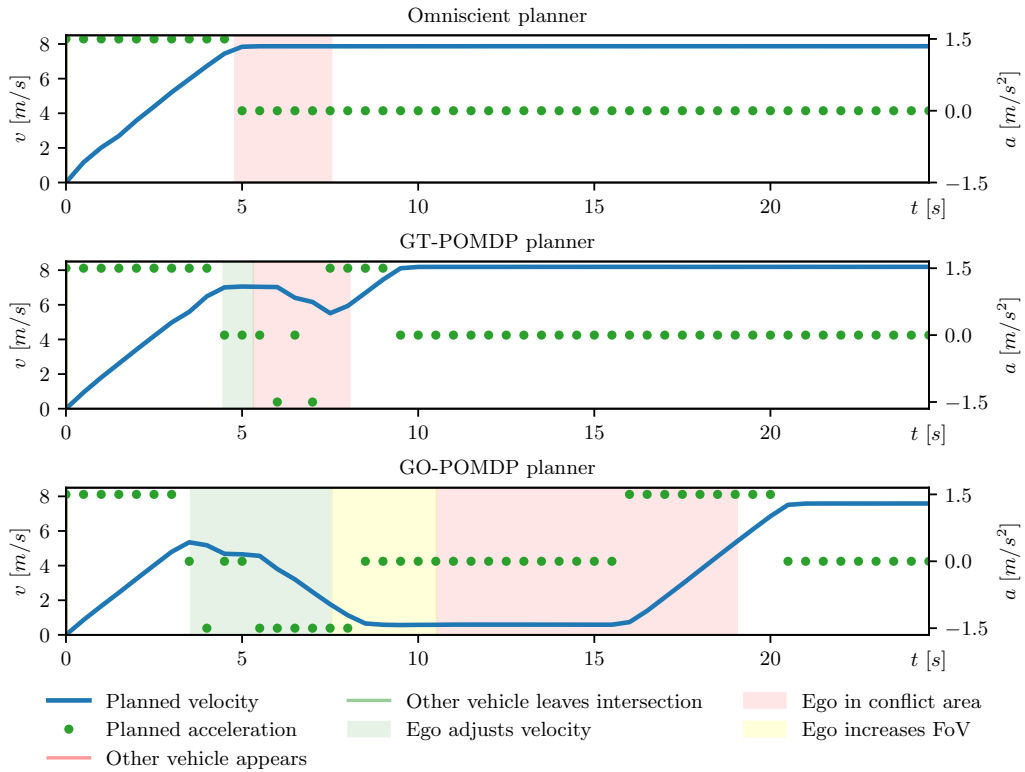


Figure 3.12: Comparison of planned driving strategies for handling the occluded intersection (scenario A), (graphic from [Zha+22c], ©2022 IEEE).

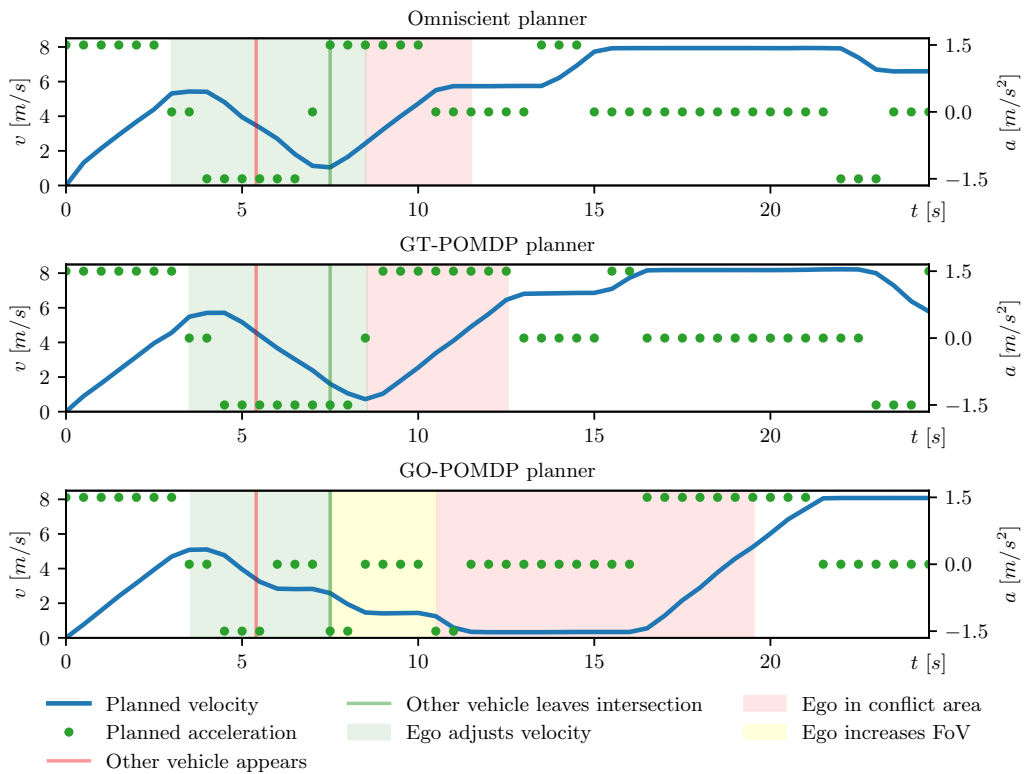


Figure 3.13: Comparison of planned driving strategies for handling the occluded intersection with a dynamic vehicle (scenario B), (graphic from [Zha+22c], ©2022 IEEE).

Because there are no other vehicles on the occluded lanes, Omniscient planner accelerates and drives through the intersection, as illustrated in Fig. 3.12. The GT-POMDP approach utilizes traffic mirror data to determine whether or not a vehicle is in an oncoming lane. So, it maintains a relatively high speed through the intersection. In contrast, GO-POMDP decelerates and creeps forward with low speed to increase the FoV so that it can safely drive through the intersection.

If there is a vehicle hidden in the occluded area that approaches the intersection in scenario B, GT-POMDP in Fig. 3.13 shows comparable performance to Omniscient planner. It slows down and waits for the vehicle that has higher priority. After the other vehicle leaves the conflict area at time $t = 7.52$ s, GT-POMDP enters the intersection at $t = 8.56$ s and accelerates as long as the mirror provides high confidence that no vehicles are in the blind spot. GO-POMDP again drives slowly to increase the FoV when handling the occluded scenario.

Occlusion in Crosswalk

A crosswalk scenario is depicted in Fig. 3.14 to evaluate the performance of the traffic mirror-aware planner in the presence of an occluded risk area where pedestrians may cross the street. Scenario C is without pedestrians, whereas in scenario D, two pedestrians will cross the street when the ego vehicle is near the crosswalk.

It can be seen in Fig. 3.15 that GT-POMDP stops decelerating at time $t = 7.47$ s and decides to cross the crosswalk due to the awareness of the low appearance risk of pedestrians in the occluded area. Without this knowledge provided by the traffic mirror, GO-POMDP needs to behave more cautiously.

The result for scenario D shown in Fig. 3.16 shows a similar reaction of GT-POMDP like Omniscient planner. When pedestrians cross the street, the ego vehicle has already slowed down to a reasonable speed and let them cross first. Compared to GO-POMDP, GT-POMDP drives through the occluded intersection faster.



Figure 3.14: Scenario C: Ego vehicle driving through a crosswalk with occlusion caused by a parked vehicle. Scenario D: Two pedestrians intend to cross the road, but their views are blocked (graphic from [Zha+22c], ©2022 IEEE).

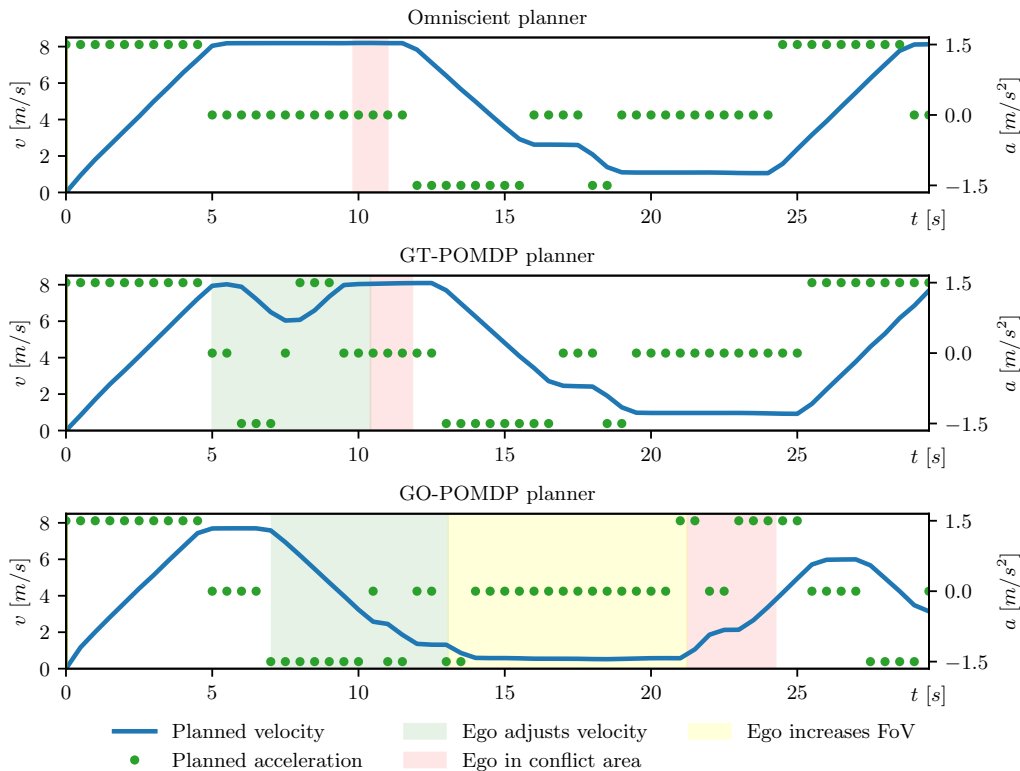


Figure 3.15: Comparison of planned driving strategies for handling the occluded crosswalk without pedestrians (scenario C), (graphic from [Zha+22c], ©2022 IEEE).

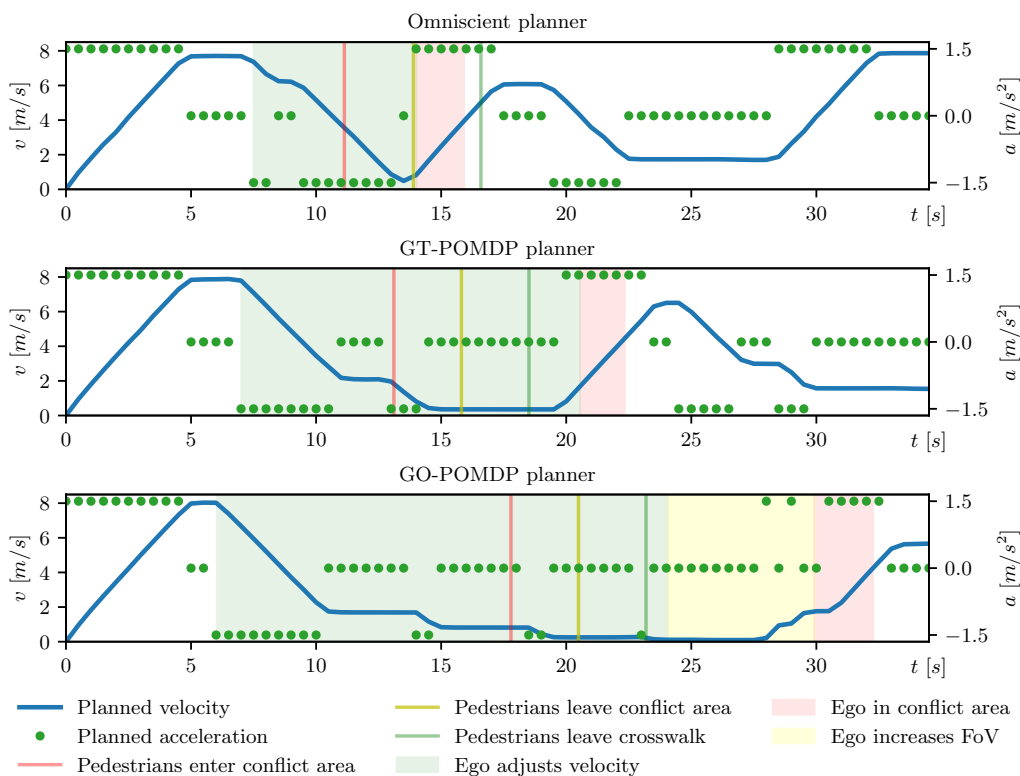


Figure 3.16: Comparison of planned driving strategies for handling the occluded crosswalk with crossing pedestrians (scenario D), (graphic from [Zha+22c], ©2022 IEEE).

Occluded Traffic Mirror

Finally, a challenging scenario is configured in Fig. 3.17 to demonstrate the capability of the traffic mirror-aware POMDP planner where a static building occludes the intersection, and a moving truck in the front of the ego vehicle limits more FoV of the ego vehicle. Even the traffic mirror is occluded by the truck when the ego vehicle approaches the occluded area.

The acceleration and velocity profiles are shown in Fig. 3.18. To keep observing the traffic mirror, the GTM-POMDP starts to slow down at time $t = 10.1$ s and tries to keep a large distance to the leading truck. The GTM-POMDP has more time to observe the traffic mirror before entering the conflict area of the intersection. The GT-POMDP is not aware of the traffic mirror, resulting in a late slow down before entering the intersection. Furthermore, GTM-POMDP has a shorter period that is obstructed from viewing the traffic mirror. GO-POMDP shows a more conservative driving style compared to both GT-POMDP and GTM-POMDP.

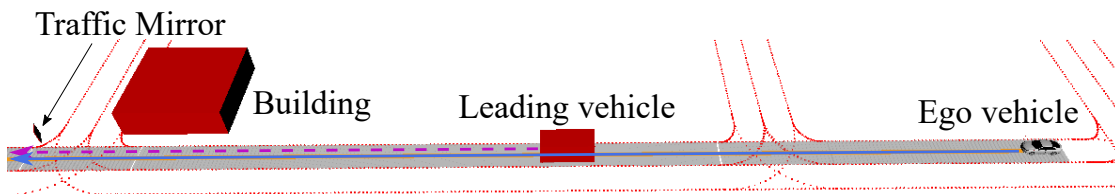


Figure 3.17: Scenario E: Ego vehicle is approaching a slow truck. The traffic mirror is blocked by the truck, (graphic from [Zha+22c], ©2022 IEEE).

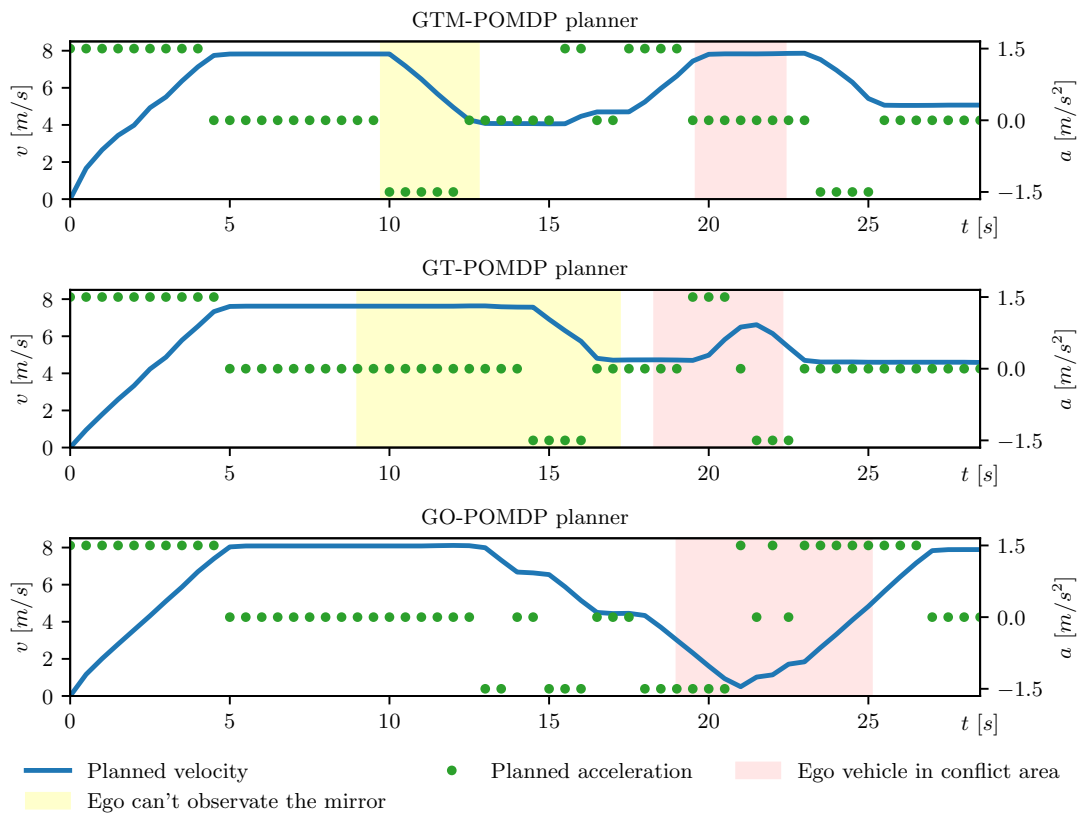


Figure 3.18: Comparison of planned driving strategies for handling the occluded intersection with occluded traffic mirror (scenario E), (graphic from [Zha+22c], ©2022 IEEE).

3.5 Summary

This chapter introduces an occlusion-aware POMDP behavior planner for driving in urban environments with improved coverage of occlusion scenarios. The occlusion-aware planner handles various occlusion situations caused by static or dynamic obstacles for various occluded road users. The main contribution is to combine the considerations for uncertainty made by the POMDP and context-aware phantom object modeling with appearance probability, which enables the effective use of map information. Evaluation results show that the ego vehicle can safely drive through challenging occlusion scenarios, such as crosswalks, bus stops, and intersections. The ego vehicle performs comfortable driving behavior under occlusions, i.e., creeps forward into the conflict area to increase visibility with the knowledge that other road users could suddenly emerge. Owing to the effective modeling of the context-based appearance probability, the presented approach does not cause deadlocks as the worst-case assumption approach would in heavy occlusion situations.

This chapter further extends the occlusion-aware POMDP behavior planner with an observation model for the traffic mirror. With this extension, the POMDP planner utilizes the information extracted from the traffic mirror to improve its estimation of the existence of the potential hidden road users. The evaluations show that using traffic mirror detection allows the planner to drive more safely and efficiently in the presence of occlusions in intersections and crosswalks. Because of the active traffic mirror perceiving method, the traffic mirror-aware POMDP planner can take into account the current and future observability of the traffic mirror and drive slower to better observe the traffic mirror and gain more information.

4

Behavior Planning under Spatial Occlusion with V2X Communication

The previous chapter introduces a POMDP-based behavior planning algorithm for handling occlusions using onboard sensor information. However, the behavior planner relying solely on onboard sensors suffers from a restricted FoV. Therefore, the onboard sensor-based approach needs to make assumptions for potential road users both within the occluded areas and beyond the sensor detection ranges. Integrating V2X communication with autonomous vehicles is beneficial since it provides information beyond the onboard sensors' FoVs. This chapter enhances the POMDP-based behavior planner presented in Chap. 3 by incorporating V2X communication to achieve safe driving behaviors in occlusion scenarios. This chapter is based on the author's previously published work [Zha+23].

4.1 Overview and Contributions

As discussed in Section 3.2, one solution to the occlusion problem is to estimate collision risk arising from limited visibility using the onboard sensors and incorporate the estimated risk into the motion planning module of autonomous vehicles. In occlusion-aware motion planning, the goal is to determine driving policies that minimize collision risks with potential hidden road users [DUG21; Koç+21b; Nar+21; YVJ19; ZF21; Li+21; Wan+22; PVN20b; Nau+19b; WLS20b; Sán+22; NS20; Kam+20; Bou+19; Ise+18; Sch+19b; Bou+18b; Hub+19; Lin+19a; Sch+19a; WGW21; Zha+21; Zha+22c]. However, such approaches have limitations in balancing risk reduction and efficiency improvement, as they rely solely on onboard sensors. Fig. 4.1 illustrates a potential collision situation between the ego vehicle and the orange vehicle, where the orange vehicle violates the right-of-way rule as it cannot observe the ego vehicle.

V2X communication provides the possibility to overcome this occlusion scenario. Previous works focus on fusing information from multiple sources and creating an environment model that transcends the ego vehicle's limitations [Che+19b; Arn+20; Wan+20; Guo+21; Guo+22; Gün+16; Amb+19; Yu+22; Qi+21; Dai+20; Xia+23; Dax+22; Müll+22]. However, V2X communication cannot guarantee comprehensive coverage of every occlusion area in mixed traffic situations, where human-driven vehicles, autonomous vehicles, and V2X vehicles share the road. This chapter addresses these issues by proposing a POMDP-based planning algorithm that utilizes but does not completely rely on V2X communication for autonomous vehicles in mixed traffic situations.

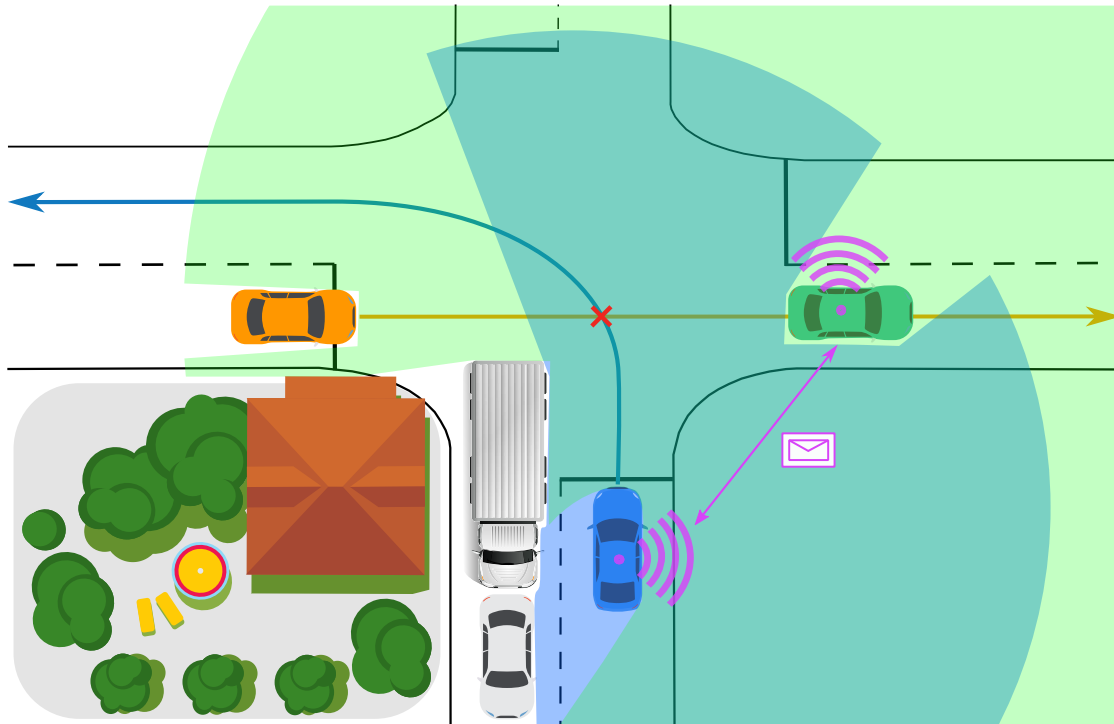


Figure 4.1: The ego vehicle (blue car) executes a left-turn maneuver at an unsignalized intersection, while another vehicle (orange car) on a low-priority lane also aims to traverse the intersection. Due to occlusion caused by a large truck (gray truck) stuck in a traffic jam, both the ego vehicle and the orange vehicle are unable to directly observe each other, creating a potentially hazardous situation. In this scenario, a V2X vehicle (green car) can detect the orange vehicle using its onboard sensors. It shares its observable area (green area) and detected object list with the ego vehicle through V2X communication, providing information about the orange vehicle beyond the ego vehicle's FoV (blue area), (graphic from [Zha+23], ©2023 IEEE).

Occlusion handling that relies solely on onboard sensors suffers from visibility limitations. Therefore, onboard sensor-based approaches need to make assumptions for potential road users both within the occluded areas and beyond the sensor detection ranges. In the scenario shown in Fig. 4.1, the ego vehicle would drive overcautiously if it assumes potential road users will always emerge from a low-priority occluded area. Yet algorithms solely based on V2X communication also have their challenges. Since autonomous vehicles need to operate in mixed traffic, the assumption may not be valid that the ego vehicle can receive V2X perception data for every risky occluded area along the navigation path, as some occluded areas may lack V2X agents that observe it.

This chapter addresses these challenges by proposing a POMDP behavior planner that combines the benefits of methods using onboard sensors and methods using V2X communication. The core idea is to rely on the information from the onboard sensors to identify occlusion areas and estimate the collision risk using the phantom road users concept while enhancing the estimation of the existence of the phantom road users using available V2X messages. This approach has two advantages. Firstly, it allows the ego vehicle to identify every occlusion area through its mission and map data, thus avoiding the problem of omitted occlusion areas that might occur when depending solely on V2X messages. Secondly, the presented approach does not require fusing the object lists from different V2X messages. Instead of directly using the potentially unreliable object lists provided by V2X for motion planning as proposed by [Dax+22] and [Mül+22], the approach proposed in this chapter selects the V2X messages to improve the estimation of potentially occluded objects, thus lowering the demands for perception accuracy.

The presented approach utilizes map information to analyze occluded areas along the ego navigation path. It generates phantom road users based on the types of risk areas, such as phantom vehicles in lanes and pedestrians on crosswalks. Next, it assigns their initial appearance probability by considering the environment and future changes to the ego's FoV. To utilize V2X messages, a V2X communication module is introduced to calculate and select the confidence modifier with the most promising detection result, considering factors like observation area coverage, communication latency, and sensor reliability. Then, the V2X-confidence modifier is applied to enhance or reduce the appearance probability of phantom road users in the current and future time steps. For example, if a vehicle is detected at the occluded lane, the appearance probability of a generated phantom vehicle is increased. Finally, the phantom road user with the V2X-modified appearance probability is integrated into the POMDP model, which is solved to provide safe driving policies.

This chapter extends Chap. 3 by incorporating V2X communication into the occlusion-aware POMDP behavior planner. In summary, the new contributions of this chapter are as follows:

- the introduction of an occlusion catalog with different types of occlusion areas, enabling the planner to address a greater variety of occlusion scenarios in urban environments,
- the introduction of a V2X communication module to calculate and select the confidence modifier containing the best detection result, which eliminates the need to fuse multiple V2X messages,
- the application of the confidence modifier to enhance occlusion risk estimation for the phantom road users concept. The confidence modifier enables the ego vehicle to benefit from V2X communication without solely relying on it, thus facilitating the operation of autonomous vehicles in mixed traffic environments, even when the information from V2X messages does not cover every risky occluded area,
- the demonstration of the performance of the presented approach both qualitatively and

quantitatively in challenging occlusion scenarios, such as occluded crosswalks, curved roads, and intersections in urban environments.

4.2 Related Work

An overview of the literature for occlusion handling using onboard sensors is provided in Section 3.2. This section summarizes related works utilizing V2X communication to handle occlusions.

Occlusion Handling using V2X Communication

By exchanging information with other vehicles or infrastructures, also denoted as V2X agents, occlusion can be handled using V2X communication. [Han+23] categorizes collaborative perception into three groups according to the stage of the collaboration: early collaboration, intermediate collaboration, and late collaboration. In early collaboration, other V2X agents send raw perception data such as camera images or LiDAR point clouds to the ego vehicle. The collaboration module of the ego vehicle fuses the received raw perception data with its onboard perception data [Che+19b; Arn+20]. Early collaboration provides the most comprehensive information about environments but also suffers from high data bandwidth and high coupling between the different data sources. Intermediate collaboration methods reduce this issue of high data bandwidth by fusing the features extracted from the raw perception data shared among agents [Wan+20; Guo+21; Guo+22]. In late collaboration, agents send their perception output to the ego vehicle in the abstracted form of an object list, which contains information about the location, size, shape, motion, and other relevant attributes of perceived objects [Gün+16; Amb+19; Yu+22]. The collaborative module of the ego vehicle has to aggregate object lists from different V2X agents. Late collaboration is bandwidth-efficient but sensitive to localization errors and transmission latency. To reduce computation workload on the collaborating vehicles, [Qi+21; Dai+20; Xia+23] propose to outsource collaborative computation to infrastructures. The European Telecommunications Standards Institute (ETSI) introduces the Collective Perception Message (CPM), defining the message format for sharing information using V2X communication for perceived objects, observable area, used sensors, and agent data.

As an alternative to improve perception ability with V2X technology, researchers have also investigated motion planning algorithms incorporating external information. [Dax+22] uses late collaboration in POMDP motion planning that integrates onboard and external object lists in the model's state space. [Mül+22] uses onboard and V2X object lists separately for motion planning. However, the algorithm requires fusion of multiple object lists from different V2X devices.

4.3 Approach: Occlusion Handling With V2X Communication

This section first presents the process of the POMDP behavior planner with V2X communication. Next, a detailed description of the V2X communication module and its integration within the POMDP behavior planner is presented.

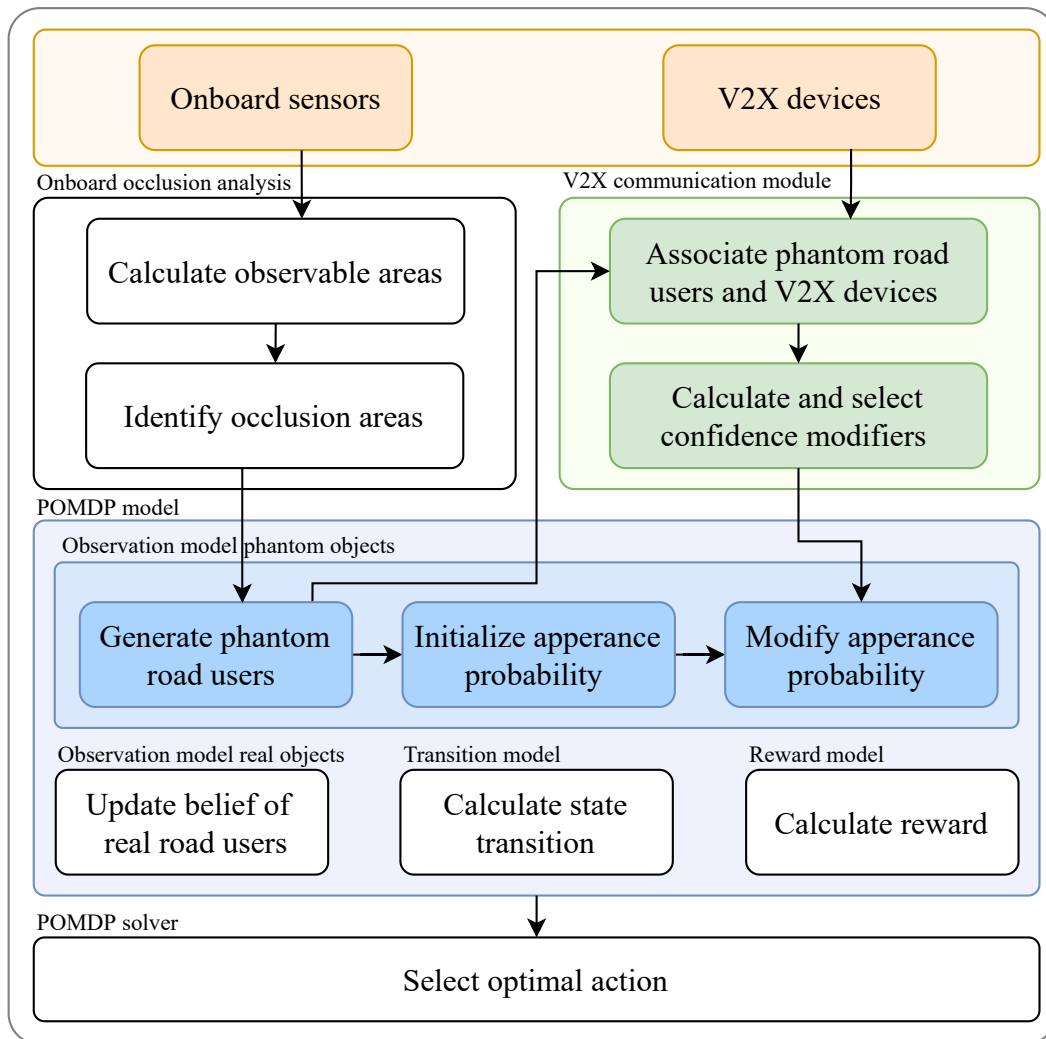


Figure 4.2: Illustration of the integration of the V2X communication module into the POMDP behavior planner (graphic from [Zha+23], ©2023 IEEE).

4.3.1 Framework

To effectively utilize both onboard perception and V2X communication information, this chapter presents an approach that processes each information source separately. First, onboard perception and map information are used to infer potential phantom objects and their appearance probabilities. Upon completion, the generated phantom objects can readily be incorporated into the POMDP state. If V2X messages are available, the approach does not directly use the potentially unreliable environment information provided by V2X for motion planning. Instead, it uses them to improve the estimation of the existence of road users in the occluded areas.

Fig. 4.2 shows the process of the POMDP behavior planner incorporating V2X communication for handling occlusion scenarios. The observation model for real objects utilizes environment data perceived by onboard sensors to update the belief of real road users. The observation model for phantom objects first generates phantom road users based on the identified occlusion areas provided by the onboard occlusion analysis module. Next, it initializes their appearance probabilities to represent the probability of phantom road users emerging from the occluded area. Both observed real road users and generated phantom road users

are included in the state space of the POMDP model.

To integrate V2X communication into the POMDP planner, this chapter introduces a V2X communication module to process the messages sent to the ego vehicle. The V2X communication module first associates each generated phantom object with the V2X devices that are able to observe it. In the following step, it calculates and selects a confidence modifier to quantify the confidence level regarding the presence of real road users at the location of a phantom road user. Finally, the confidence modifier is applied to adjust the appearance probability of each generated phantom object. By further defining the transition and the reward model, the POMDP model is solved to obtain driving policies online.

4.3.2 State, Observation and Action Space

The state and observation space described in the Section 3.3.2 is retained here. Similarly, the action space adheres to the description in Section 3.3.3.

4.3.3 Observation Model of Real Objects

The observation of the ego vehicle can be generated directly from measurements. For other real traffic participants N_i , their positions (x_i, y_i) , orientations θ_i , and speeds v_i can also be observed directly. Their intention of choosing a subsequent path r_i is inferred and updated by the ego vehicle's prediction module whenever new measurements are received.

4.3.4 Onboard Occlusion Analysis

The onboard occlusion analysis module is responsible for calculating the observable areas \mathcal{L}_{obs} and identifying occlusion areas \mathcal{L}_{occ} along the ego driving path r_{ego} using data from onboard perception. The observable areas \mathcal{L}_{obs} are the areas that the ego vehicle's onboard sensors can detect. The onboard occlusion analysis module first set up a full observable area centered at the ego vehicle and limited by the maximum range D_p of the perception system. Then, it uses perceived static and dynamic obstacles to shrink this full observable area to observable areas \mathcal{L}_{obs} . Next, the module search for occluded lanes and areas \mathcal{L}_{occ} in the map along the driving path r_{ego} . An occlusion catalog is introduced as shown in Table 4.1 to define the different occlusion types $\varphi_{occ,k}$ for assigning to \mathcal{L}_{occ} . The occlusion type $\varphi_{occ,k}$ describes the lane relations to the ego vehicle, such as opposite lanes or the ego navigation lane. Furthermore, the module extracts the relative priority of \mathcal{L}_{occ} over the ego vehicle in unsignalized intersections, i.e., high-priority and low-priority lanes. This is relevant in countries like Germany where drivers must follow the right-before-left rule when approaching intersections without traffic signs or lights. Moreover, the module considers unobservable areas around crosswalks and bus stops as occluded areas of interest.

4.3.5 Observation Model of Phantom Objects

The observation model of phantom objects is responsible for the generation of phantom road users N_k , including their attributes, states x_k , and estimated appearance probabilities $p_{app,k}$.

Table 4.1: Occlusion catalog for defining different types of phantom road users (table from [Zha + 23], ©2023 IEEE).

Occlusion type $\varphi_{occ,k}$	High-priority lane	Opposite lane	Low-priority lane	Ego navigation lane	Occluded area
Illustration					
Object type $\varphi_{obj,k}$	Phantom vehicle	Phantom vehicle	Phantom vehicle	Phantom vehicle	Phantom pedestrian
Priority $\varphi_{pri,k}$	Higher	Higher	Lower	Same	Higher
Location (x_k, y_k)	Edge of FoV	Edge of FoV	Edge of FoV	Ahead of leading vehicle	Edge of FoV
Orientation θ_k	Lane direction	Lane direction	Lane direction	Lane direction	Perpendicular to lane
Velocity v_k	Speed limit	Speed limit	Speed limit	0 m/s	1.25 m/s

Generation of Phantom Road Users

The observation model generates phantom road users depending on the types of the identified occlusion areas \mathcal{L}_{occ} . As shown in Table 4.1, it assigns attributes for each phantom road user N_k , including occlusion type $\varphi_{occ,k}$, object type $\varphi_{obj,k}$, and priority $\varphi_{pri,k}$. These attributes are not included in the states x_k of phantom road users N_k since they are static values. Object type $\varphi_{obj,k}$ represents a phantom vehicle or a phantom pedestrian. The priority $\varphi_{pri,k}$ represents the relative priority of a phantom object over the ego vehicle.

Table 4.1 further provides information regarding the location (x_k, y_k) , orientation θ_k and velocity v_k for generating the state x_k of the phantom road users. For example, a phantom vehicle N_k is generated at the edge of the FoV by arc-length s_k along a high-priority lane r_k with a velocity v_k of the speed limit. The Cartesian coordinates (x_k, y_k) and orientation θ_k are obtained from the intended path r_k according to the path geometry based on the position s_k . A static phantom vehicle is placed ahead of the leading vehicle at a safe distance D_f . This accommodates situations where the leading vehicle suddenly performs an evasive maneuver due to a static obstacle, and allows the ego vehicle to have sufficient reaction time. For a phantom pedestrian, pseudo-priority walking paths are generated. The walking paths consist of waypoints starting at the edge of the FoV and pointing to the other side of the road. The lengths of all phantom road users are defined as infinite in the occluded areas, enabling the representation of a set of reachable states using only one configuration of phantom road users.

Initialization of Appearance Probability

In the next step, the appearance probabilities of the generated phantom road users is initialized.

Assuming that the phantom road users always emerge from the occlusion area at each planning cycle would cause the ego vehicle to drive overcautiously. In some cases, this assumption could even block the ego vehicle and lead to a “freezing state” without further movement. To avoid this, different initial appearance probabilities depending on the occlusion type $\varphi_{occ,k}$ and the phantom object type $\varphi_{obj,k}$ are introduced.

The observation model of phantom objects assigns the appearance probability $p_{app,k} = 0$ to phantom vehicles on low-priority lanes, meaning that with only onboard perception, the observation model does not consider the risk of vehicles appearing and initiating an illegal behavior from these areas. This avoids overly cautious driving by the ego vehicle. However, with V2X communication, the appearance probability of low-priority phantom vehicles can be increased if a vehicle is detected that violates the right-of-way rule and crosses the intersection before the ego vehicle. Moreover, the observation model also assigns the appearance probability $p_{app,k} = 0$ for static phantom objects on the ego navigation lane. For the remaining group of phantom road users, the observation model initializes their appearance probability with $p_{app,k}$ introduced in Section 3.3.5.

4.3.6 V2X Communication

The V2X communication module first finds a subset of V2X devices $\mathcal{C}_{asc,k}$ that can observe the phantom road user N_k from all available V2X devices \mathcal{C} . Next, it calculates and selects the most suitable confidence modifier $p_{cm,k}^*$ to assess the level of confidence concerning the existence of road users at the location of the phantom road user N_k . The confidence modifier of a V2X device c for phantom object N_k considers the detection result $p_{d,k}^c$, the maximum number of trust steps $t_{t,k}^c$ for the device due to the observation area coverage for the phantom

Algorithm 3: V2X Device Association

Input : $K - N$ phantom objects, V2X devices \mathcal{C}
Output: Associated V2X devices \mathcal{C}_{asc}

```

1  $\mathcal{C}_{asc} \leftarrow \emptyset$ 
2 foreach  $k \in \{N + 1, \dots, K\}$  do
3    $\mathcal{C}_{asc,k} \leftarrow \emptyset$ 
4   foreach  $c \in \mathcal{C}$  do
5      $\mathcal{L}_{obs,c} \leftarrow c$ 
6     if observes ( $\mathcal{L}_{obs,c}, N_k$ ) then
7       addToList( $\mathcal{C}_{asc,k}, c$ )
8     end if
9   end foreach
10  addToList( $\mathcal{C}_{asc}, \mathcal{C}_{asc,k}$ )
11 end foreach
12 return  $\mathcal{C}_{asc}$ 

```

Algorithm 4: Confidence Modifier Calculation

Input : k -th phantom object N_k , associated V2X devices \mathcal{C}_{asc}
Output: Confidence modifier $p_{cm,k}^*$

```

1  $\mathcal{C}_{asc,k} \leftarrow \mathcal{C}_{asc}, P_{cm,k} \leftarrow \emptyset$ 
2 foreach  $c \in \mathcal{C}_{asc,k}$  do
3    $\mathcal{L}_{obs,c}, \mathcal{x}_c, T_c \leftarrow c$ 
4    $p_{d,k}^c \leftarrow \text{detectionResult}(N_k, \mathcal{x}_c)$ 
5    $t_{t,k}^c \leftarrow \text{maxTrustSteps}(N_k, \mathcal{L}_{obs,c})$ 
6    $f_{t,k}^c \leftarrow \text{getDelayFactor}(T_c)$ 
7    $f_{q,k}^c \leftarrow \text{getTrustFactor}(c)$ 
8    $P_{cm,k} \leftarrow \text{confidenceModifier}(p_{d,k}^c, t_{t,k}^c, f_{t,k}^c, f_{q,k}^c)$ 
9    $P_{cm,k} \leftarrow \text{addToList}(P_{cm,k}, P_{cm,k})$ 
10 end foreach
11  $p_{cm,k}^* \leftarrow \text{selectConfidenceModifier}(P_{cm,k})$ 
12 return  $p_{cm,k}^*$ 

```

road user N_k , delay factor $f_{t,k}^c$ for evaluating communication latency, and trust factor $f_{q,k}^c$ representing sensor reliability.

Algorithm 3 outlines the association step in the V2X communication module. Input to the algorithm are all the generated phantom road users N_k , and a list of V2X devices \mathcal{C} from which the ego vehicle receives messages. Each V2X device c provides a message containing the observable areas $\mathcal{L}_{obs,c}$, which represents the areas that the device c can observe, an object list \mathcal{x}_c containing the detected objects, and timestamp T_c indicating the time when the message is sent. For each phantom object N_k , it initializes an empty list of V2X devices $\mathcal{C}_{asc,k}$ (lines 2-3). Then, it determines whether the observable areas $\mathcal{L}_{obs,c}$ of each V2X device c encompass the location (x_k, y_k) of the phantom road user N_k (lines 4-6) and add it to the corresponding device list $\mathcal{C}_{asc,k}$ (lines 7). The output of the algorithm is the list of associated V2X devices \mathcal{C}_{asc} which contains subsets of V2X devices $\mathcal{C}_{asc,k}$ for each phantom road user N_k .

Algorithm 4 shows the calculation and selection of the confidence modifier $p_{cm,k}^*$. It first gets the observable areas $\mathcal{L}_{obs,c}$, observed objects list \mathcal{x}_c , and timestamp T_c from associated

V2X device c for phantom object N_k (line 3). It then computes the detection result $p_{d,k}^c$, maximum number of trust steps $t_{t,k}^c$, delay factor $f_{t,k}^c$, trust factor $f_{q,k}^c$, and a confidence modifier $p_{cm,k}^c$ representing the confidence of V2X device c for phantom object N_k (see lines 4-8). Then, it adds $p_{cm,k}^c$ to the list $P_{cm,k}$ for storing all confidence modifiers for phantom object N_k (line 9). Finally, it selects the most reliable confidence modifier $p_{cm,k}^*$ for the phantom object N_k when it is observed by multiple V2X devices (line 11). In the following, a detailed explanation of the algorithm is provided.

Detection Result

Algorithm 4 calculates the detection result $p_{d,k}^c$ by evaluating whether any detected object N_j from object list \mathcal{X}_c of V2X device c with phantom road users N_k holds the following conditions:

$$p_{d,k}^c := \begin{cases} 1, & \text{if } d_{jk} \leq D_c \wedge \Delta\theta_{jk} \leq \Theta_c, \\ -1, & \text{otherwise.} \end{cases} \quad (4.1)$$

with d_{jk} , D_c representing the Euclidean distance and distance threshold between phantom object N_k and detected object N_j , respectively. $\Delta\theta_{jk}$ and Θ_c denote the relative orientation and relative orientation threshold between them.

Maximum Number of Trust Steps

The presented algorithm calculates the maximum number of trust steps $t_{t,k}^c$ for V2X device c to determine the number of time steps for which it can apply the confidence modifier to adjust the appearance probability of the phantom object N_k :

$$t_{t,k}^c := \frac{d_{k,min}^c}{v_k}. \quad (4.2)$$

where $d_{k,min}^c$ is the minimum distance of the phantom object N_k to the boundary of the observable area $\mathcal{L}_{obs,c}$ of V2X device c , and v_k is the assumed velocity of the phantom object N_k . Beyond the maximum number of trust steps $t_{t,k}^c$, other undetected objects may enter the observable area and reach the position of the phantom object, as illustrated in Fig. 4.3.

Delay Factor

Delays must also be considered when the ego vehicle utilizes communication messages. During delays, detected objects move further, making the perception data unreliable. Therefore, a delay factor is introduced $f_{t,k}^c \in [0, 1]$ to indicate the degree of trust that the ego vehicle should assign to the message from V2X device c :

$$f_{t,k}^c := \begin{cases} 1 - \Delta t_c, & \text{if } 0 \leq \Delta t_c \leq 1, \\ 0, & \text{otherwise,} \end{cases} \quad (4.3)$$

here, $\Delta t_c = T_{cur} - T_c$ denotes the time interval between the moment T_{cur} , when the message is processed by the ego vehicle and T_c , when it is transmitted. When messages from V2X device c are older than 1 s, with $\Delta t_c \geq 1$ s, the delay factor becomes $f_{t,k}^c = 0$, indicating that such a message will not be taken into account.

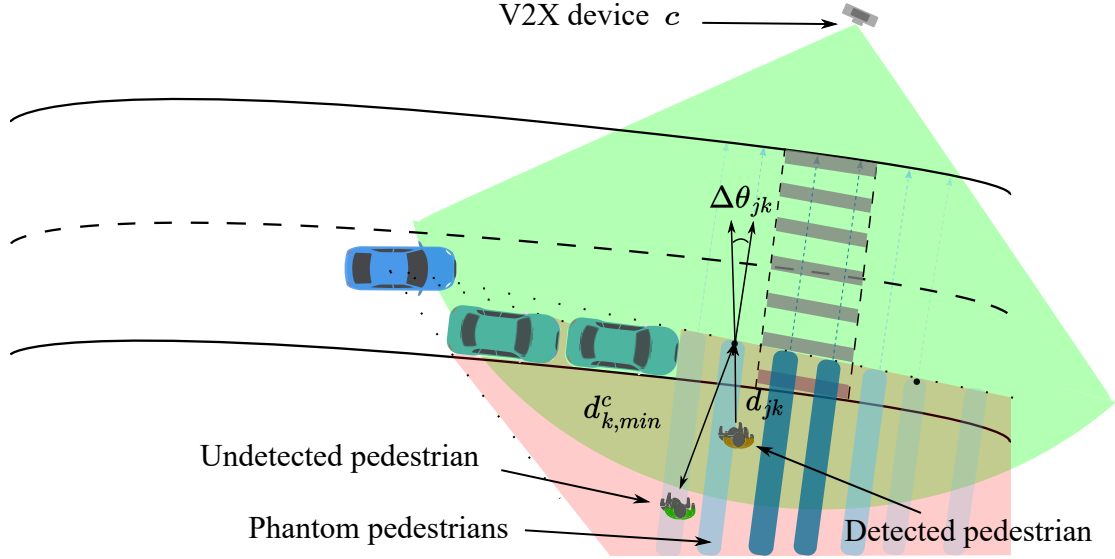


Figure 4.3: An occluded crosswalk scenario. A V2X infrastructure observes the crosswalk (green area) and sends communication messages to the ego vehicle (blue car). A pedestrian (yellow) is within the observable area of the infrastructure while the other pedestrian (green) is out of its detection range (graphic from [Zha+23], ©2023 IEEE).

Trust Factor

The purpose of introducing a trust factor $f_{q,k}^c$ is to prioritize more reliable devices when multiple V2X devices are observing the location of phantom object N_k . Table 4.2 shows the trust factors for selected types of V2X devices. For example, certified devices, such as the connectivity platform ZF ProConnect, are more trusted than uncertified devices, such as webcams. Moreover, a V2X vehicle from the same brand may provide more accurately calibrated data than those from other brands.

Confidence Modifier

The confidence modifier is computed by combining the detection result $p_{d,k}^c$, maximum number of trust steps $t_{t,k}^c$, delay factor $f_{t,k}^c$, and trust factor $f_{q,k}^c$:

$$p_{cm,k}^c := p_{d,k}^c \cdot f_{t,k}^c \cdot f_{q,k}^c, \text{ if } t_h < t_{t,k}^c, \quad (4.4)$$

with $t_h \in \{0, \dots, H\}$ being a time step within the planning horizon $H \in \mathbb{R}^+$. When multiple confidence modifiers $P_{cm,k}$ are available, the confidence modifier with the highest confidence

Table 4.2: Selected trust factors for V2X infrastructures and vehicles (table from [Zha+23], ©2023 IEEE).

V2X type	device	Trust factor	Description
Infrastructure		1.0	Government certified device
Infrastructure		0.8	Uncertified device, e.g. webcam
Vehicle		0.9	Vehicle from same/associated brand
Vehicle		0.7	Vehicle from another brand

value is selected:

$$P_{cm,k}^* := f(P_{cm,k}) := \begin{cases} \max(|P_{cm,k}|), & \text{if } \forall p_{cm,k}^c \in P_{cm,k} : p_{cm,k}^c < 0, \\ \max(P_{cm,k}), & \text{otherwise.} \end{cases} \quad (4.5)$$

If $p_{cm,k}^c > 0$, the V2X device detects an object that matches the phantom object N_k . The device with the highest confidence is chosen. (4.5) also implies a conservative preference when faced with conflicting detection results from multiple V2X devices. In such cases, the device that detects the object with $p_{cm,k}^c > 0$ is preferred over the one that does not ($p_{cm,k}^c < 0$). When all devices do not detect an object at the phantom object location ($p_{cm,k}^c < 0$), the device with the highest absolute confidence ($\max(|P_{cm,k}|)$) is chosen.

4.3.7 Modification of Appearance Probability

Finally, the presented approach applies the selected confidence modifier $p_{cm,k}^*$ to modify the initial appearance probability $p_{app,k}$ for the phantom object N_k when it is available:

$$p_{app,k}^* := \begin{cases} \max(0, \min(1, p_{app,k} + p_{cm,k}^*)), & \text{if } p_{cm,k}^* \text{ is available,} \\ p_{app,k}, & \text{otherwise.} \end{cases} \quad (4.6)$$

4.3.8 Transition Model

Ego Vehicle

The intended ego path remains unchanged such that $r'_{ego} = r_{ego}$. A point mass model is applied in (4.7) to predict the ego movement along the intended ego path r_{ego} . The position of the ego vehicle s_{ego} is the arc-length along r_{ego} . The new ego position s'_{ego} and velocity v'_{ego} are predicted by the chosen action a and the step size Δt :

$$\begin{bmatrix} s'_{ego} \\ v'_{ego} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} s_{ego} \\ v_{ego} \end{bmatrix} + \begin{bmatrix} \frac{1}{2}\Delta t^2 \\ \Delta t \end{bmatrix} a. \quad (4.7)$$

The x'_{ego} , y'_{ego} and θ'_{ego} in the updated ego state x'_{ego} are obtained by getting the Cartesian position from r'_{ego} according to the path geometry based on the position s'_{ego} .

Other Road Users

The state transition model for other objects is also referred to as the prediction model in the literature. The prediction model estimates other traffic participants' intentions and future states based on dynamics and map information. The prediction and uncertainty of the next state given the current state and the intention of other objects can be obtained through the interface with the prediction module. Thus, sophisticated prediction models can be used for different road users. Since this is not the focus of this thesis, a simple constant velocity model is applied to update the state of other objects along their path.

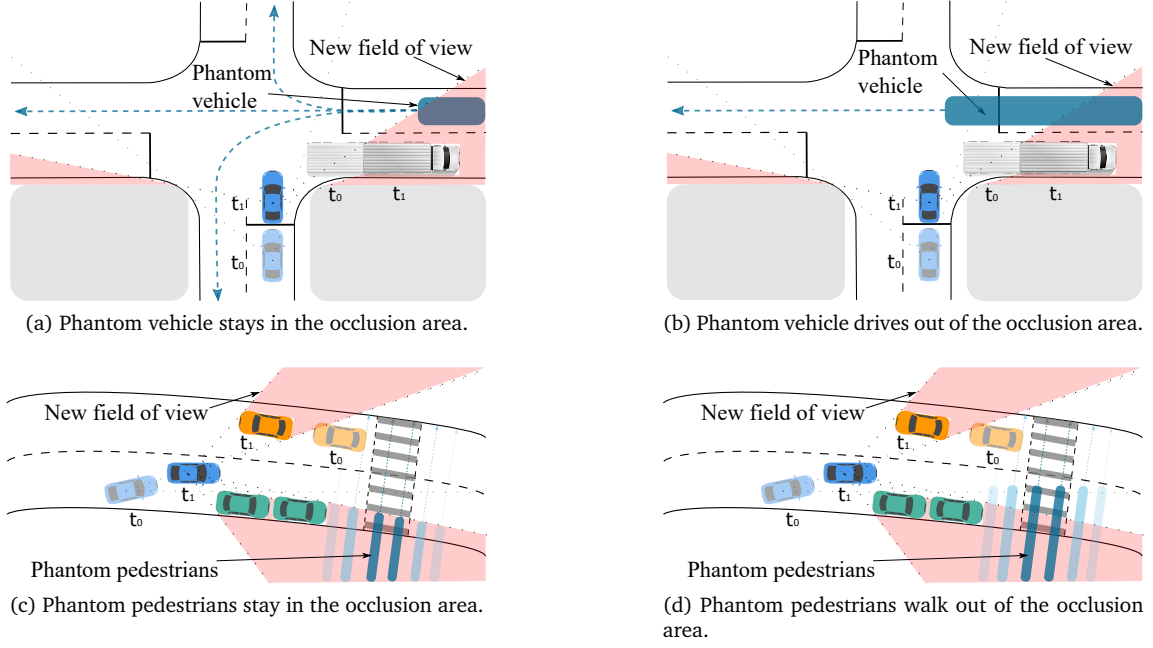


Figure 4.4: The transition model of phantom vehicles and pedestrians. The blue car is the ego vehicle. The yellow car and white truck are other moving traffic participants. Green cars represent parked cars. For phantom pedestrians, dark blue indicates high appearance probability (graphic from [Zha+23], ©2023 IEEE).

Phantom Road Users

When updating the state transition of phantom object N_k over time step Δt , sampling is initially performed based on the V2X modified appearance probability $p_{app,k}^*$. If the sample result is zero, the phantom object is placed at the updated edge of the FoV, as shown in Figs. 4.4a and 4.4c. Otherwise, when the sample result is one, its position s'_k is moved forward using a constant velocity model: $s'_k = s_k + v_k \cdot \Delta t$ (see Figs. 4.4b and 4.4d). Since static phantom objects do not move, the location of the static phantom objects is updated at the beginning of each planning cycle when new observations are available.

4.3.9 Reward Model

The reward function is a crucial factor in designing the ego vehicle's behavior which satisfies several objectives, such as safety, efficiency, and comfort. These objectives are encoded mathematically within the reward function $R(s, a)$:

$$R = R_{\text{collision_real}} + R_{\text{collision_phantom}} + R_{\text{speed}} + R_{\text{comfort}}. \quad (4.8)$$

A number of simulations were performed in order to find suitable weights for the reward function. Safety can be modeled by assigning a large negative reward $R_{\text{collision_real}} = -100000$ if the ego vehicle collides with other road users. A different penalty is assigned in the case of a collision with a phantom object $R_{\text{collision_phantom}} = -10000$.

The speed reward is used to encourage the ego vehicle to drive according to the desired velocity v_{desired} on the driving lane as far as possible while not exceeding it:

$$R_{\text{speed}} = \begin{cases} -200 \cdot (v_{\text{desired}} - v_{\text{ego}}), & \text{if } v_{\text{desired}} \geq v_{\text{ego}}, \\ -2000 \cdot |v_{\text{desired}} - v_{\text{ego}}|, & \text{otherwise.} \end{cases} \quad (4.9)$$

To obtain comfortable driving policies, acceleration is penalized with: $R_{\text{comfort}} = -300 \cdot a^2$.

4.4 Experiments and Results

4.4.1 Experiment Setup

The presented approach is evaluated on a system equipped with an Intel Core i7-6820HQ CPU running at 2.70 GHz. The evaluations are conducted within a proprietary simulation platform, supporting the configuration of static buildings, moving vehicles, and pedestrians on selected HD maps. To ensure a realistic comparison of the presented approach, control over other road users is established through predefined behaviors that do not consider collision avoidance. A module is developed for simulating V2X devices, including V2X vehicles and infrastructures which perceive environmental information within their observation area and transmit this data to the ego vehicle. Table 4.3 lists the parameters used in the simulation. A recorded video of the evaluations¹ is provided.

Planners Setup

The presented approach, denoted as the **V2X-POMDP** planner, is a POMDP-based behavior planner with V2X communication capabilities. As previously mentioned, the V2X-POMDP planner leverages V2X communication to improve presence estimations of phantom road users, rather than directly using object lists from other V2X devices. For comparison with the method of directly using object lists, another planner is established, named **V2X-Object-POMDP**, which incorporates the received object lists from other V2X devices into its perceived object list. As a baseline, the approach from Chap. 3 is utilized, the **GO-POMDP** planner, which infers the appearance probability of occluded road users based solely on map information without V2X capabilities. Lastly, an **Omniscient** planner is established as the ground truth, possessing accurate observations of all road users, including the occluded ones.

Evaluation Design and Metrics

To evaluate the presented approach, experiments are first conducted to examine the efficiency of the proposed V2X communication algorithm. The experiment first focuses on the association time of phantom objects and received messages from V2X devices, which is required once per planning cycle. Next, the accumulated total computation time needed in one planning cycle for the calculation and selection of the confidence modifier is compared. Furthermore, qualitative experiments are performed to compare the driving strategies of planners in four challenging scenarios. Finally, quantitative experiments are set up to evaluate the performance of the presented approach in comparison to other planners.

The following metrics are chosen to compare the performance of the planners:

- **Total reward:** The average cumulative reward obtained by the ego vehicle within an episode.
- **Success rate:** The percentage of episodes in which the ego vehicle successfully reaches the goal without collision within the maximum allowed episode length.
- **Collision rate:** The percentage of episodes in which the ego vehicle collides with an obstacle or another vehicle.

¹Video: <https://github.com/GitChiZhang/V2X-POMDP>

Table 4.3: Applied parameters in the simulation (table from [Zha+23], ©2023 IEEE).

Params.	Value	Description
F	2 Hz	Planning frequency
γ	0.95	Discount factor
D	10	Maximal tree depth
H	10 s	Planning horizon
D_p	100 m	Maximum range of the ego perception
D_f	20 m	Safe distance before the leading vehicle
K_{env}	0.2	Initial environmental probability
L	5/10 m	Length for detecting one pedestrian/vehicle
D_s	1 m	Distance threshold of risk area
D_c	10/20 m	Threshold for detecting an pedestrian/vehicle
Θ_c	$\frac{\pi}{2}$	Orientation threshold for detecting an object

- **Timeout rate:** The percentage of episodes in which the ego vehicle fails to reach the goal within the maximum number of time steps of an episode.
- **Average velocity:** The average speed of the ego vehicle over all episodes.
- **Average absolute acceleration:** The average magnitude of the acceleration of the ego vehicle over all episodes, for measuring comfort by considering abrupt speed changes.

4.4.2 Evaluation of the V2X-aware POMDP Planner

Computation Efficiency of Average Association Time

This evaluation examines the average association time while increasing the number of phantom objects and V2X devices. Fig. 4.5 demonstrates that as the number of phantom objects increases, the processing time also increases. Similarly, an increased number of V2X devices leads to a rise in processing time. However, even with 10 phantom objects and 5 V2X devices, the processing time required is less than 1 ms.

Computation Efficiency of Confidence Modifier

This experiment analyzes the total computation time required in one planning cycle for the calculation and selection of the confidence modifier. As shown in Fig. 4.6, the median total computation time linearly increases from 1.33 ms for one device to 11.11 ms for five devices as the number of V2X devices in the evaluation scenario increases. In the worst-case scenario with five devices, the total processing time requires 32.11 ms. For a V2X-POMDP planner with a computation cycle of 2 Hz, the computation time for the V2X communication module accounts for approximately 6% of the total computation time.

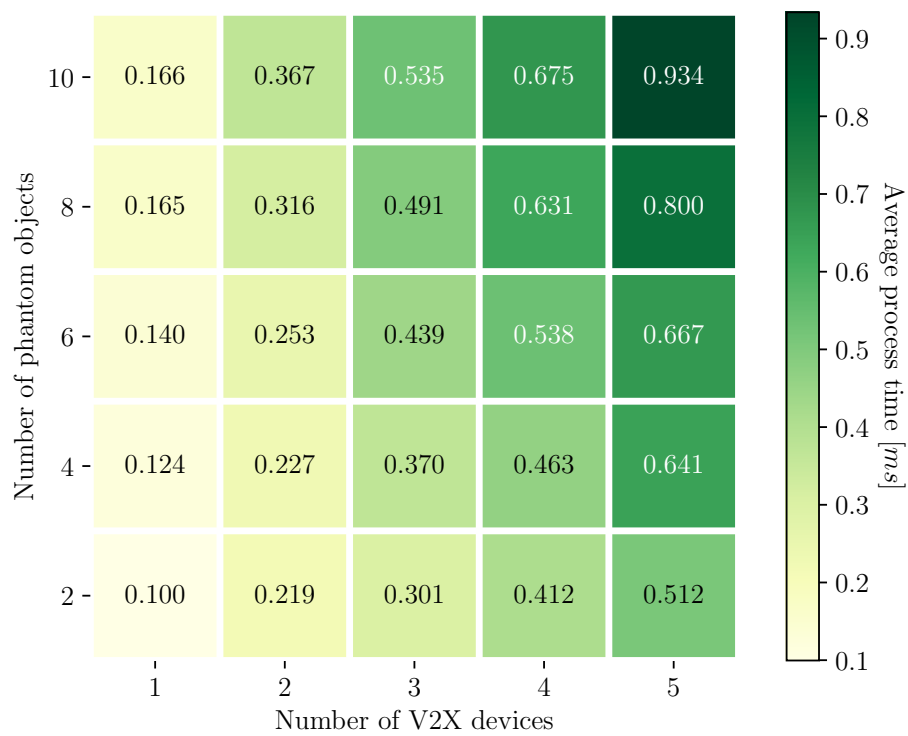


Figure 4.5: Comparison of average association time with different numbers of V2X devices and phantom objects (graphic from [Zha+23], ©2023 IEEE).

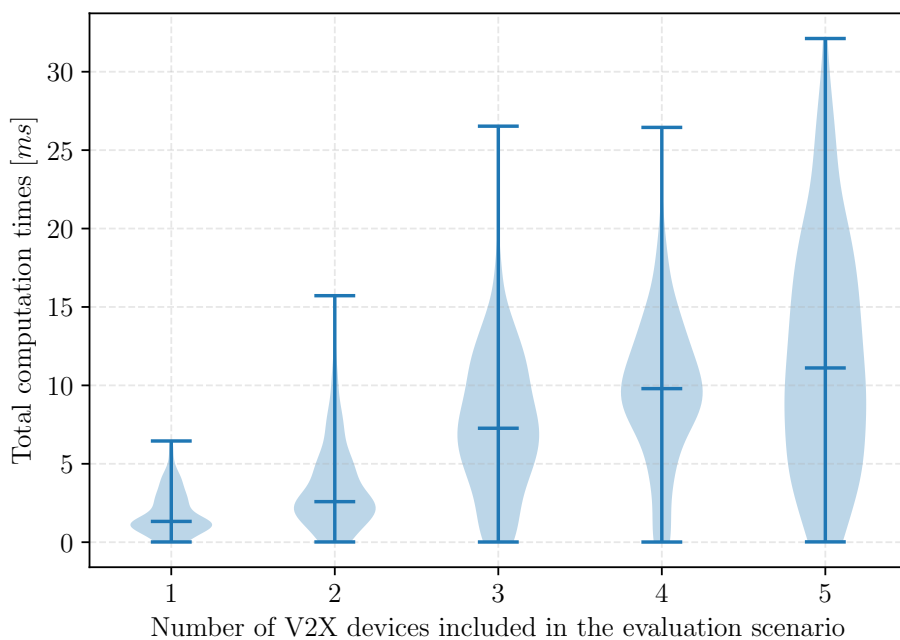


Figure 4.6: Comparison of average total computation times for calculating confidence modifiers during one planning cycle as the number of V2X devices included in the evaluation scenario increases (graphic from [Zha+23], ©2023 IEEE).

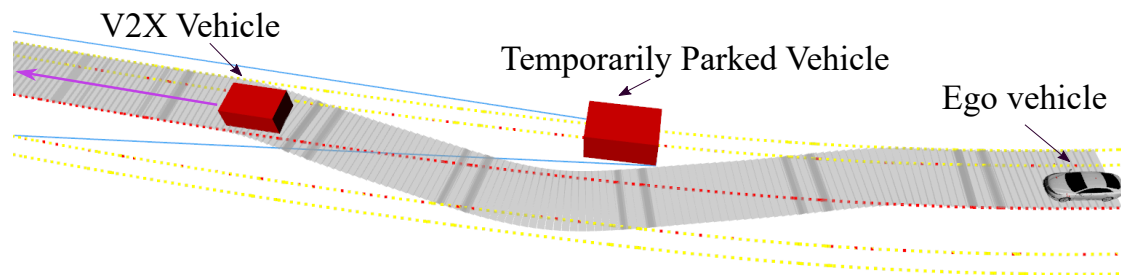


Figure 4.7: The ego vehicle must perform a lane change with limited FoV (graphic from [Zha+23], ©2023 IEEE).

Occluded Overtaking

Fig. 4.7 depicts a scenario where the ego vehicle must overtake a temporarily parked vehicle on a curved road. The occlusion from the parked vehicle and the road curvature limit the ego vehicle's FoV, increasing the difficulty of the overtaking maneuver. In this evaluation scenario, a V2X vehicle continuously transmits perception data to the ego vehicle.

Fig. 4.8 presents the evaluation results. The Omniscient planner drives directly into the opposite lane and completes the overtaking maneuver since there are no vehicles coming in the opposite direction. The V2X-POMDP planner behaves similarly, as it also directly accelerates and completes the overtaking maneuver. This is due to the information provided by the V2X vehicles driving ahead, which informs the V2X-POMDP planner that there are

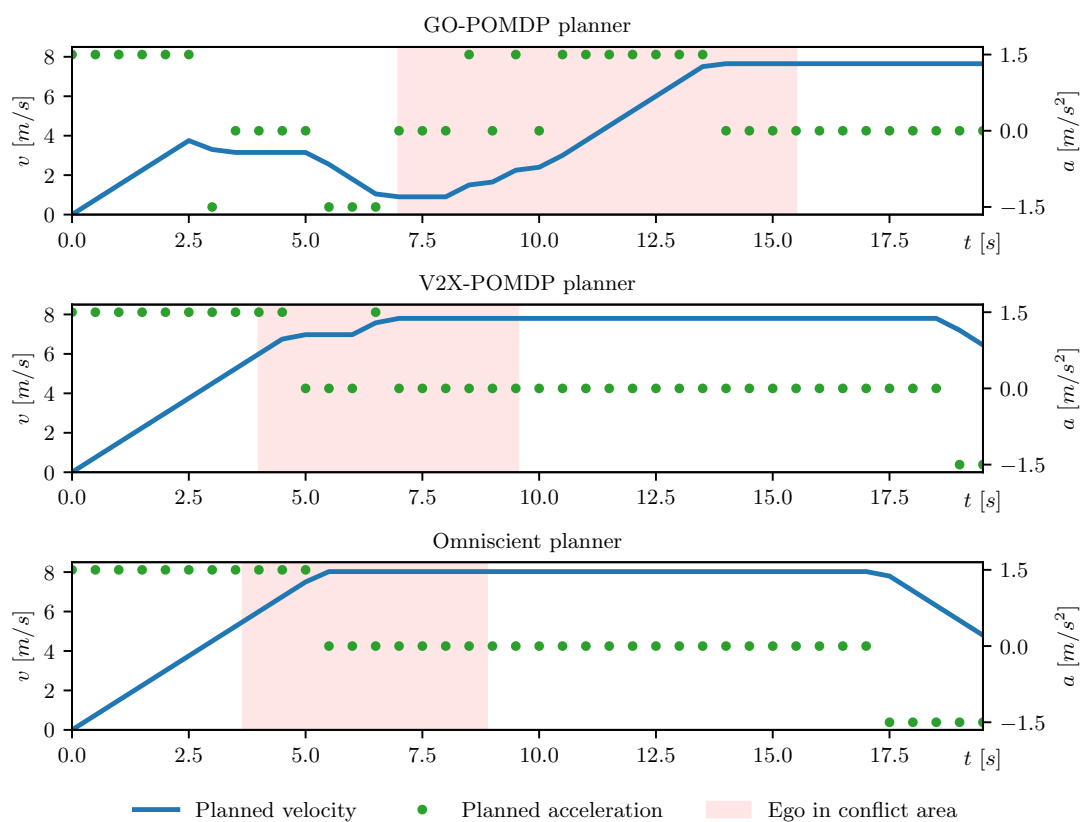


Figure 4.8: Comparison of planned driving strategies for handling the scenario where the ego vehicle must change lanes with limited observation of the oncoming lane (graphic from [Zha+23], ©2023 IEEE).

no vehicles in the oncoming lane. The V2X-POMDP planner can then reduce the probability of phantom vehicles appearing in the occluded area of the oncoming lane through the confidence modifier. In comparison, the GO-POMDP planner can only infer the appearance probabilities of vehicles on the oncoming lane through the map as it lacks V2X communication. Therefore, the GO-POMDP planner needs to decrease vehicle speed and increase its FoV before accelerating.

Confidence Modifier

Fig. 4.9 illustrates a scenario in which parked vehicles obstruct the ego vehicle's FoV as it approaches the crosswalk. Two infrastructures are configured to observe the crosswalk and send messages to the ego vehicle. Each infrastructure has an observation range of 30 meters and an angle of 160 degrees. The difference between the two infrastructures is that the observation range of infrastructure 1 covers the boundary of the crosswalk more closely than that of infrastructure 2. The purpose of this experiment is to compare how V2X-POMDP utilizes and trusts the information obtained through V2X communication.

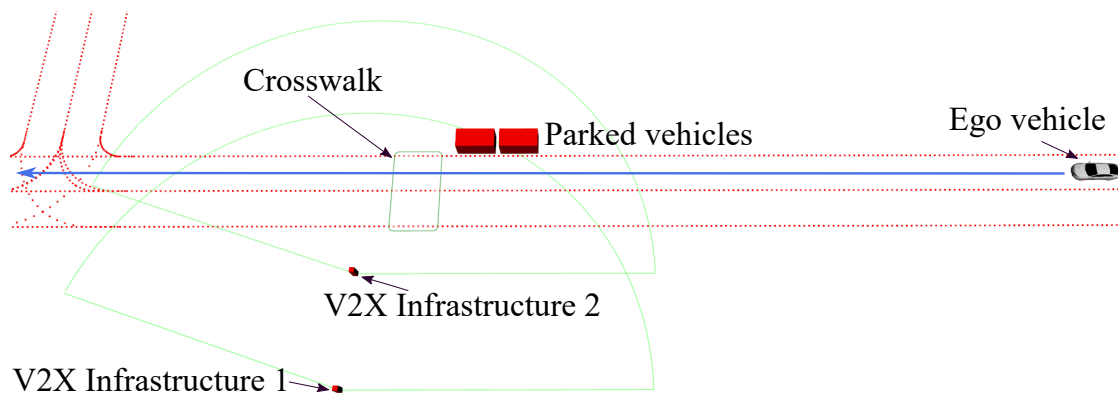


Figure 4.9: The ego vehicle is approaching an occluded crosswalk. Two V2X infrastructures are configured in this scenario (graphic from [Zha+23], ©2023 IEEE).

As demonstrated in Fig. 4.10, the GO-POMDP planner generates phantom pedestrians by assuming that there may be pedestrians standing in the occluded area who want to cross the road. Therefore, it reduces the speed of the ego vehicle such that the vehicle slowly passes through the crosswalk. The V2X-POMDP planner, which communicates with infrastructure 1, also reduces the driving speed when approaching the crosswalk. Although the observation area from infrastructure 1 is empty, the maximum number of trust steps for the V2X-POMDP regarding this V2X device is low. This is because the boundary of the observation area from infrastructure 1 is close to the crosswalk's boundary. In contrast, infrastructure 2 covers more area around the crosswalk, allowing the V2X-POMDP to reduce the appearance probability of phantom pedestrians over more time steps, as the observation area contains no objects. Therefore, the V2X-POMDP considering infrastructure 2 drives directly through the crosswalk without deceleration.

Illegal Behavior

Fig. 4.11 presents a scenario where the investigation focuses on how different planners respond to a cyclist violating traffic rules on a low-priority lane. The ego vehicle and a V2X truck are simultaneously driving towards an intersection, while a bicycle intends to cross the intersection before the truck. The challenge of this scenario is that, due to the truck's obstruction, the ego vehicle and the bicycle cannot see each other.

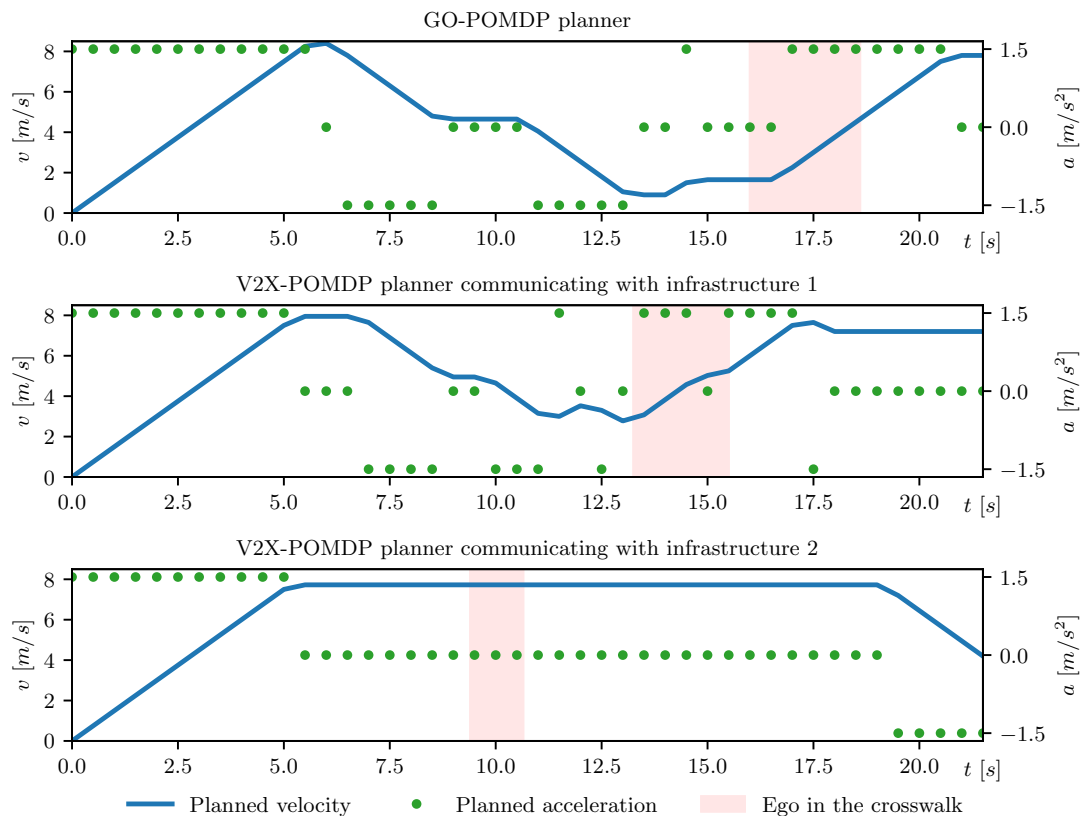


Figure 4.10: Comparison of planned driving strategies for handling an occluded crosswalk, with and without V2X communication (graphic from [Zha+23], ©2023 IEEE).

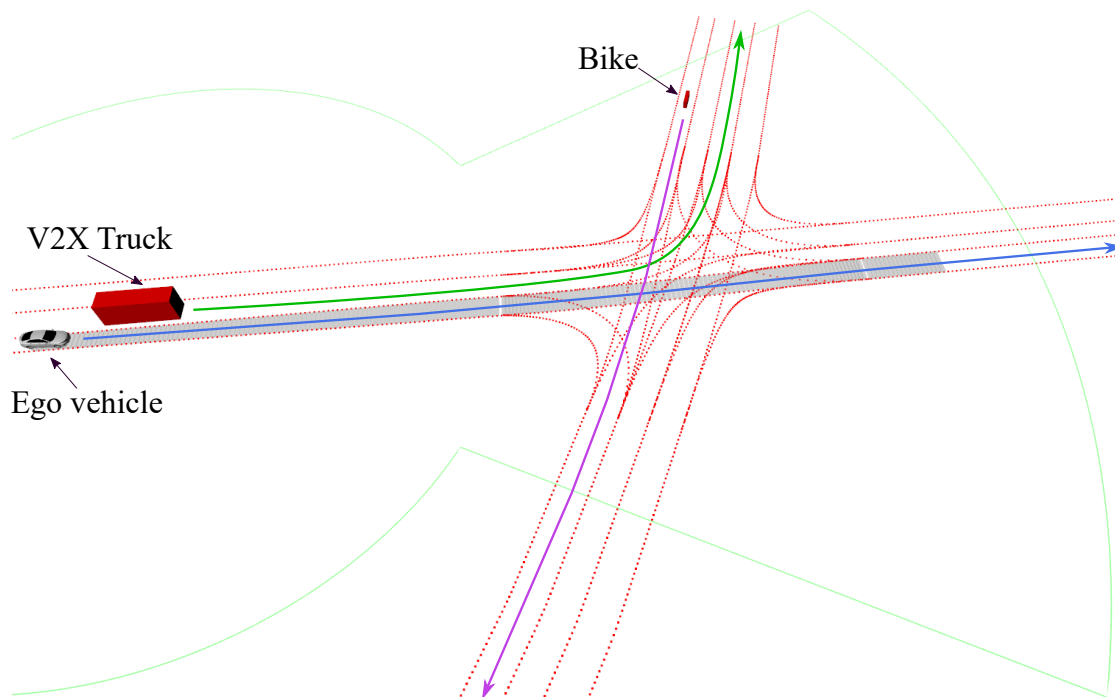


Figure 4.11: The occluded intersection involving a cyclist who behaves illegally by ignoring priority rules (graphic from [Zha+23], ©2023 IEEE).

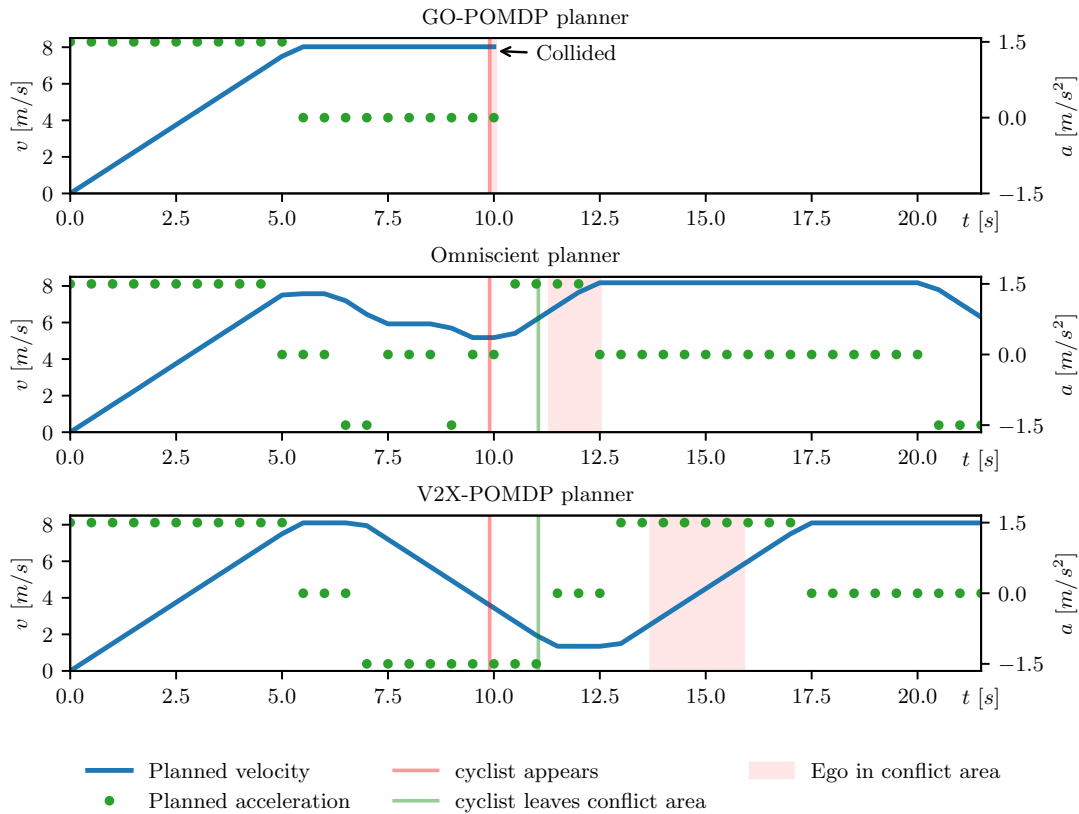


Figure 4.12: Comparison of planned driving strategies for handling the occluded intersection with a cyclist (graphic from [Zha+23], ©2023 IEEE).

Fig. 4.12 compares the driving strategies of different planners. The GO-POMDP planner assumes that the phantom vehicle on the occluded low-priority lane has a zero appearance probability. Therefore, it drives towards the intersection with the truck without reducing velocity. At time $t = 9.8$ s, the bicycle suddenly appears in front of the ego vehicle after crossing the conflict point before the truck. This results in a collision with the ego vehicle since the GO-POMDP planner does not have sufficient time to react. The Omniscient planner, which serves as the ground truth, knows the exact driving routes of the truck and bicycle, so it slows down and lets the bicycle cross first. Although the V2X-POMDP lacks the knowledge of the Omniscient planner, it executes the same behavior by slowing down before the conflict area, allowing the cyclist to cross safely. This is made possible due to the V2X information transmitted from the truck, which helps the V2X-POMDP increase the probability of phantom vehicles appearing on low-priority roads.

Communication Latency

In this scenario, the ego vehicle must traverse an intersection where a building obstructs a large part of the road (see Fig. 4.13). At the same time, another vehicle in the occluded lane approaches the conflict area of the intersection. A V2X vehicle can monitor the aforementioned vehicle and continuously transmit observation data regarding the intersection to the ego vehicle. A 1 s delay is intentionally introduced to the transmitted messages to evaluate the performance of planners under communication latency conditions.

Fig. 4.14 compares the driving strategies of the planners for handling this occluded intersection. The Omniscient planner decelerates in advance to let the approaching vehicle pass.

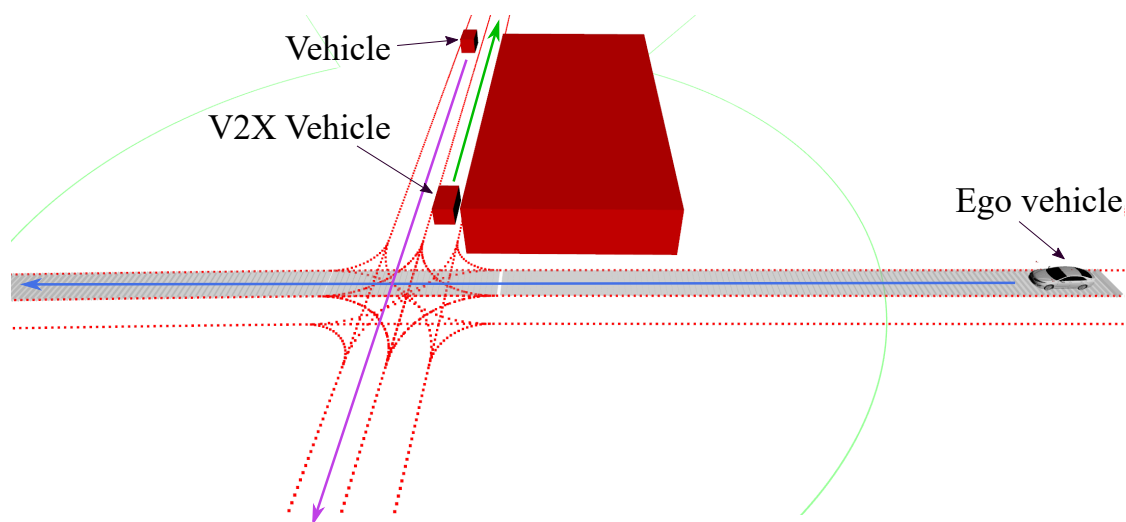


Figure 4.13: The ego vehicle approaches an intersection obstructed by a large building. A V2X vehicle transmits data with large latency (graphic from [Zha+23], ©2023 IEEE).

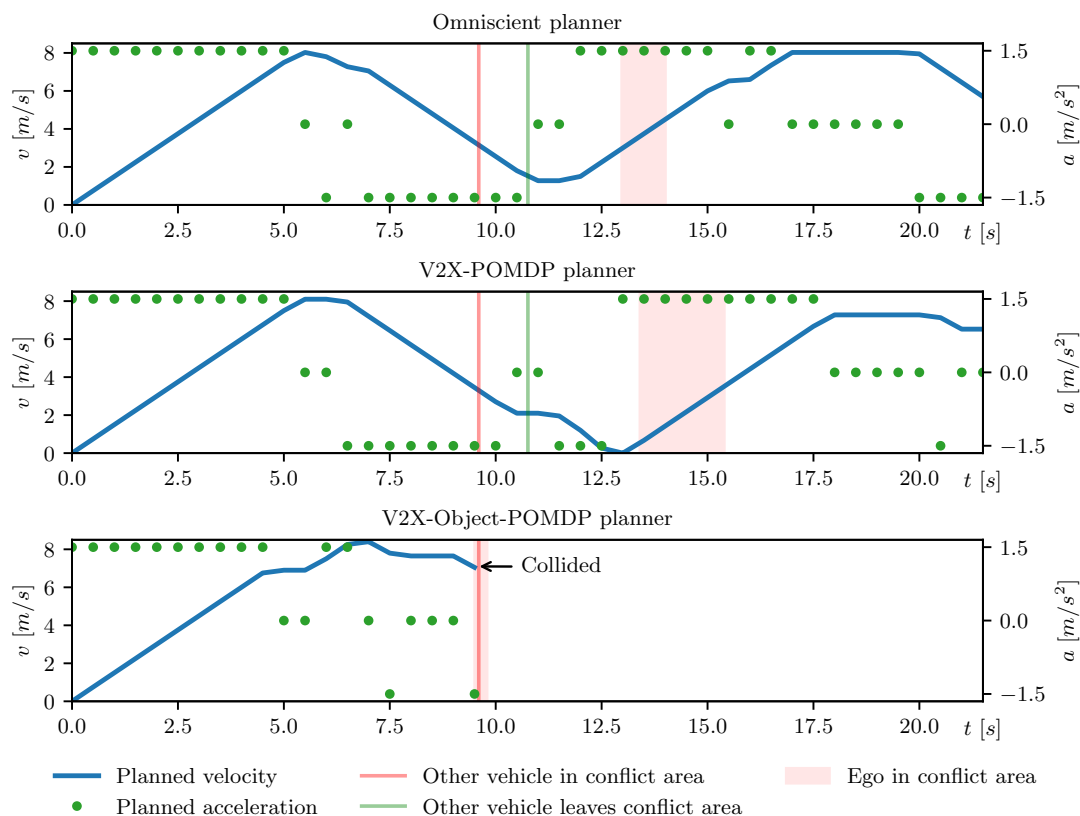


Figure 4.14: Comparison of planned driving strategies for handling the occluded intersection with communication latency (graphic from [Zha+23], ©2023 IEEE).

This is enabled by its accurate knowledge of the approaching vehicle's position and driving intention without any delay. In contrast, the V2X-Object-POMDP planner experiences a 1 s delay in receiving the object list via V2X communication and decides to let the ego vehicle pass through the intersection before the approaching vehicle. However, the actual position of the approaching vehicle is closer to the conflict zone than the position obtained by V2X

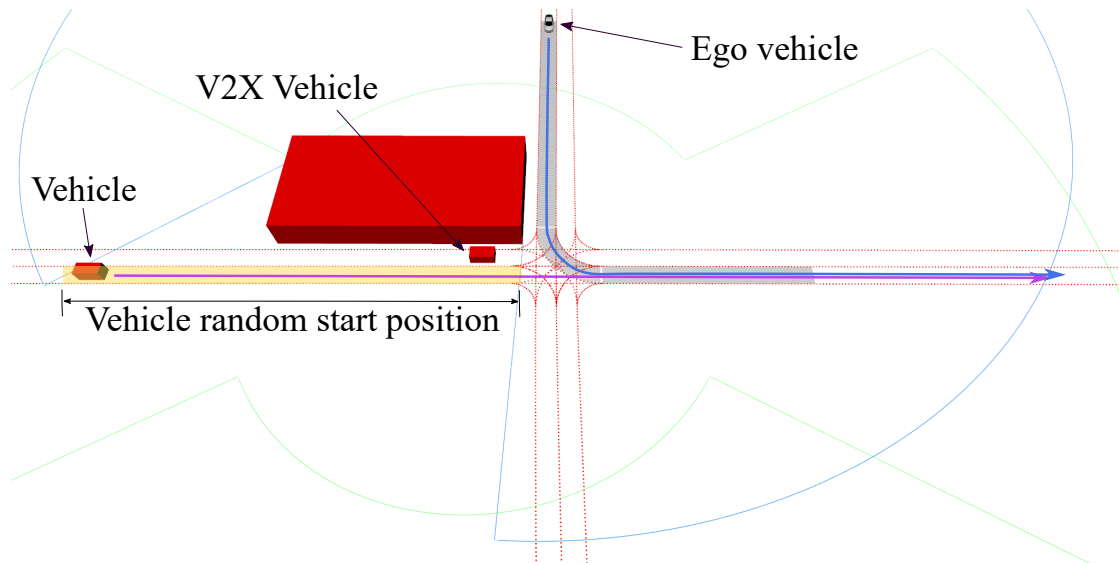


Figure 4.15: The ego vehicle performs an unprotected left turn at an occluded intersection (graphic from [Zha+23], ©2023 IEEE).

communication. As a result, the V2X-Object-POMDP planner cannot react in time when the vehicle suddenly emerges from the occluded area, leading to a collision. In comparison, the V2X-POMDP planner handles the delayed message by setting the confidence modifier value to zero. Thus, the initial appearance probability of the phantom vehicle does not change after applying the confidence modifier. As a result of this, the V2X-POMDP planner adopts a driving strategy of initially decelerating until it obtains sufficient FoV. Then, it accelerates away from the intersection.

Quantitative Evaluation at Intersections

Two quantitative evaluations are conducted on an unprotected left-turn maneuver at an intersection occluded by a large building. As illustrated in Fig. 4.15, the ego vehicle aims to traverse this intersection. A V2X vehicle provides observation messages from the occluded area. In the first quantitative evaluation, no potentially conflicting vehicles are placed in the occluded lane, while in the second evaluation, a vehicle is randomly positioned in the occluded lane, intending to drive through the intersection. 500 episodes are executed for each planner under the configured evaluation scenarios. The results are summarized in Table 4.4 and Table 4.5, respectively. Metrics are calculated as the average of all simulation results, encompassing total reward, success rate, collision rate, timeout rate, average velocity, and average absolute accelerations.

As shown in Table 4.4, the V2X-POMDP planner performs as well as the Omniscient planner, achieving a 100% success rate across all 500 episodes. In comparison, the GO-POMDP planner is more conservative, with a 3% timeout rate to reach the goal within the given time steps. The POMDP model is solved by a sampling-based solver. When drawing samples using the estimated appearance probability of phantom objects with current and predicted FoV, the planner might prefer to keep velocity over acceleration to increase FoV for some time steps. Although this is less risky, it results in a longer time to complete the mission. The average velocity indicates that the V2X-POMDP planner is 23.4% faster than the GO-POMDP planner. This suggests that the presented approach benefits from utilizing V2X communication. Furthermore, the average absolute acceleration demonstrates that the V2X-POMDP planner

chooses more comfortable actions with fewer unnecessary accelerations and decelerations compared to the GO-POMDP planner when no vehicles are present in the occluded lane.

Table 4.5 summarizes the evaluation results under the scenario with a randomly positioned vehicle. Similar to the previous evaluation, the V2X-POMDP planner performs as well as the Omniscient planner without any collisions and timeouts, while the GO-POMDP experiences 6 collisions and 10 timeouts throughout the 500 episodes. Benefiting from the V2X communication, the V2X-POMDP planner drives 13.4% faster than the GO-POMDP planner, even when the occluded area is not empty. Otherwise, both the V2X-POMDP and GO-POMDP planners behave similarly and plan less comfortable actions compared to the Omniscient planner.

4.5 Summary

This chapter extends the POMDP behavior planner for handling occlusion scenarios by incorporating V2X communication. The presented approach uses onboard sensors to generate phantom road users and infers their potential for collision with the ego vehicle in occlusion areas. The V2X communication is employed for calculating confidence modifiers to improve the estimation of the presence of phantom road users. Unlike existing approaches, the presented approach does not require fusing data from onboard sensors and V2X communication. The evaluation results show that the introduced planner can provide safer, more efficient, and more comfortable driving policies than the planner without V2X communication in various occlusion scenarios. Benefiting from evaluating and selecting V2X messages with the confidence modifier, the introduced planner is more robust against communication delays than the approach that fuses object lists from onboard sensors and V2X messages.

Table 4.4: Quantitative comparison of the performance of planners at an occluded intersection without potentially conflicting vehicles (table from [Zha+23], ©2023 IEEE).





































Planners	Total reward [-]	Success rate [%]	Collision rate [%]	Timeout rate [%]	Avg. v [$\frac{m}{s}$]	Avg. $ a $ [$\frac{m}{s^2}$]
Omniscient	-14957.7 	100 	0.0 	0.0 	6.94 	0.36 
V2X-POMDP	-24002.6 	100 	0.0 	0.0 	6.48 	0.58 
GO-POMDP	-52404.2 	97.0 	0.0 	3.0 	5.25 	0.76 

Table 4.5: Quantitative comparison of the performance of planners at an occluded intersection with a randomly appearing vehicle (table from [Zha+23], ©2023 IEEE).

Planners	Total reward [—]	Success rate [%]	Collision rate [%]	Timeout rate [%]	Avg. v [$\frac{m}{s}$]	Avg. $ a $ [$\frac{m}{s^2}$]
Omniscient	-27741.6 	100 	0.0 	0.0 	6.26 	0.47 
V2X-POMDP	-39988.7 	100 	0.0 	0.0 	5.73 	0.75 
GO-POMDP	-58000.3 	96.8 	1.2 	2.0 	5.05 	0.78 

5

Efficient Behavior Planning in Dense Urban Environments

The previous chapters introduce a POMDP-based behavior planner designed to address spatial occlusions in urban environments. The planner relies on data from onboard sensors (Chap. 3) and V2X communication (Chap. 4). In addition to addressing spatial occlusions, autonomous vehicles must effectively plan their driving behaviors within dense and interactive urban environments involving numerous road users. Navigating through dense urban areas poses a challenge for autonomous vehicles as they must anticipate the intentions of numerous road users with unclear intentions and manage various sources of uncertainty, including sensor noise and imprecise predictions. While the POMDP planner has been employed to plan driving behaviors considering occlusions, using POMDP in scenarios with a large number of road users necessitates significant computational effort. This chapter focuses on enhancing the computational efficiency of the POMDP-based behavior planner, enabling its effective use in dense urban environments. This chapter is based on the author's previously published work [Zha+22b].

5.1 Overview and Contributions

In comparison to an autonomous vehicle, a human can drive well in complex environments by better reasoning about other road users' hidden intentions and predicting stochastic future interactions with them. This human ability enables anticipative driving behavior, which is hard for a computer to achieve despite its shorter reaction time. One key challenge for behavior planning is to reason about other traffic participants' intentions.

The complexity of reasoning grows when considering noisy measurements and long-term prediction errors. As shown in Fig. 5.1, the ego vehicle must plan driving policies that take into account interactions influenced by the various future intentions of other vehicles.

As introduced in Section 2.1.4, the POMDP is a systematic method for planning optimal policies in a stochastic environment [KLC98]. However, due to the curses of dimensionality and history [PGT06], POMDP is difficult to apply in densely populated urban environments. Online POMDP planning algorithms construct a belief tree from a start state only to reachable states to approximate the optimal policy and thus reduce computational complexity [SV10; Som+13; KSK14]. Extensive works show the progress made in applying POMDP to autonomous vehicles to handle various driving scenarios in urban environments which require the reasoning of intentions and the consideration of different uncertainties [Wei+11;

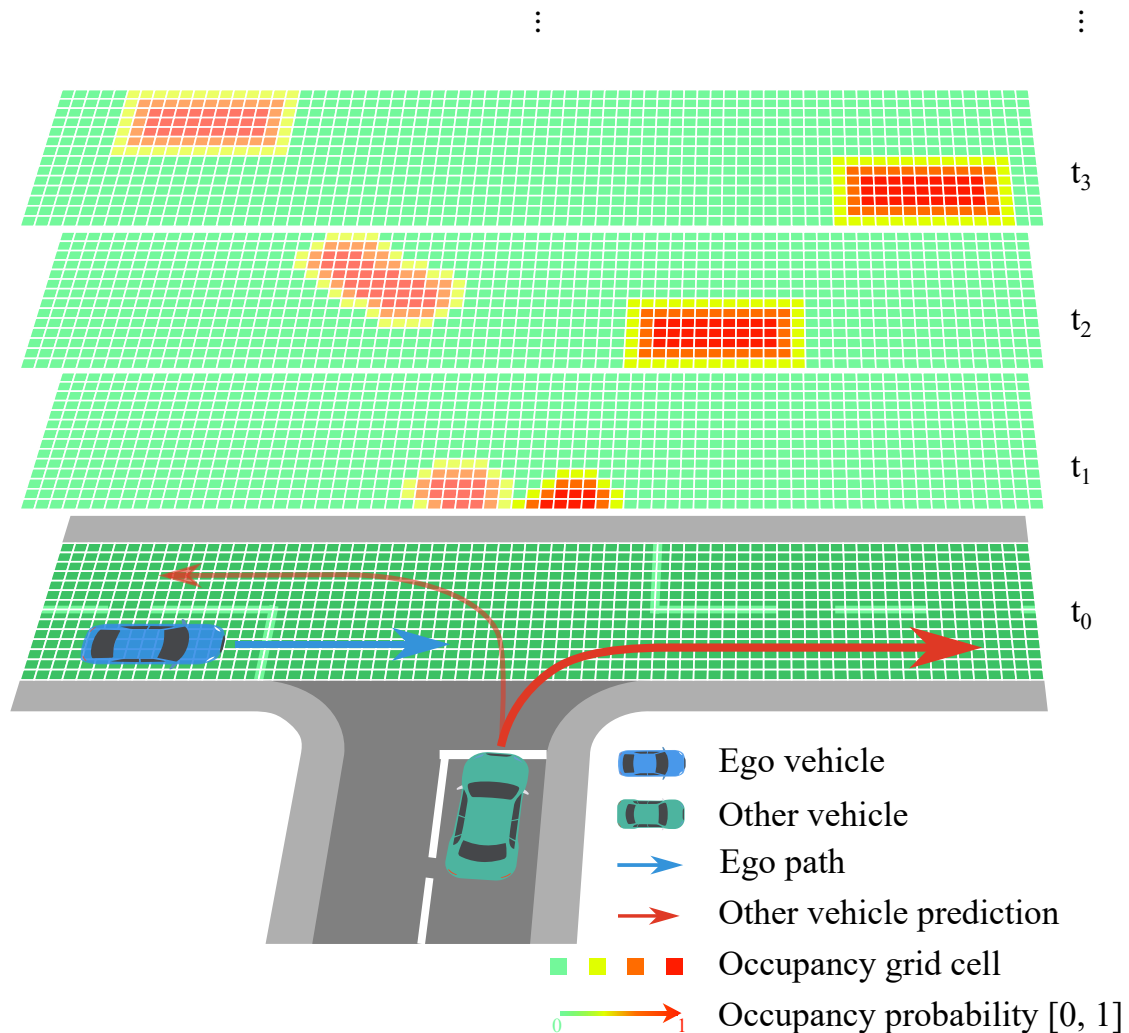


Figure 5.1: Example of an MOGM with uncertain measurements, predictions, and intentions. Each green layer is one grid map of the MOGM at each time step. The color of the grid cell indicates the risk of this grid cell being occupied by a road user at this time step. In this example, another vehicle (green car) intends to drive into the junction. This vehicle has two possible intentions: turning left or turning right. Both intentions and their corresponding predicted states are represented in the MOGM by setting the grid cells as occupied (red grids) or possibly occupied (orange and yellow grids), (graphic from [Zha+22b], ©2022 IEEE).

UM15; BHL14; Hub+17; Hub+18b; Son+18; BCK17; Sef+17; Hub+18a; Liu+15; Bey+21; Sch+19b; Tho+18; Hub+19; WGW21; Zha+21; Lin+19b; Sch+19a; Zha+20]. Most approaches so far have been demonstrated in environments with a few road users. To apply the POMDP planner in highly dynamic and dense urban environments with large numbers of road users, its scalability needs to be investigated.

This chapter focuses on improving the scalability of a POMDP behavior planner by introducing an efficient MOGM-based POMDP model. The online POMDP solver relies on Monte Carlo sampling to construct the belief tree from the current state to the reachable state (details see Section 2.2.2). A POMDP model is required as a black box simulator to generate a new episode, which includes transitioning from one state to the next and checking whether a state is a terminal state [KSK14]. Such a model needs to be called thousands to millions of times in a single planning cycle. Therefore, a more computationally efficient POMDP model can enhance the scalability of the POMDP planner.

Previous approaches typically build the POMDP state space and model using a feature-based representation of the environment, where the features of each road user, e.g., position, orientation and velocity, are set up directly in the state space. For each state transition of the ego vehicle and road users, a collision check is performed by simulating the interaction between each ego vehicle and road user pair, using the features defined in the state space. When the number of road users increases, the computational efforts of the collision checks also increase. Hence, feature-based approaches become increasingly inefficient for large numbers of road users. Increasing the efficiency of the collision check can reduce the computational effort for solving the POMDP model.

As mentioned in [Bey+21] and [Lin+19b], a large number of future states of road users can be calculated in advance and saved in a lookup table to improve the efficiency of the planner. This chapter extends this concept by using MOGM to represent all future states and probabilistic intentions of other road users. MOGM contains multiple Occupancy Grid Map (OGM) to represent the free and potential collision areas at each planning time step, up to the planning horizon. An OGM discretizes the road surface into grid cells indicating the state of a particular area, i.e., occupied or not occupied. The computational demands of the collision check are reduced by utilizing MOGM to determine whether the ego vehicle is located in areas occupied by other road users.

To enable the planning of driving policies in the uncertain environment, the uncertain measurements and predictions of road users are considered with extended occupancy area by using a two-dimensional Gaussian approximation. Their multiple intentions and the estimated probabilities are incorporated by extending the occupied grid cells to save the uncertain intention information. Finally, in the POMDP solver, the MOGM-based POMDP model is used as the black box simulator to build the belief tree and obtain optimized driving policies.

The main contributions of this chapter are:

- an extension of the MOGM that includes estimation of road users' uncertain intentions to enable intention-aware planning based on grid maps,
- the introduction of an efficient MOGM-based POMDP model for reducing computational effort, which improves the computational efficiency of the planner,
- the evaluation and analysis of computation time for the MOGM-based POMDP planner in scenarios with large numbers of road users.

5.2 Related Work

This section compares feature-based and grid-based approaches to representing the environment and predictions. Following that, the POMDP-based planners that consider various sources of uncertainty are summarized and discussed.

Environment Representation

The perception and prediction module of an autonomous driving function provides the current and predicted environment information to the planning module [Yur+20]. The environment can be represented in different forms.

1. Feature-based Method

The feature-based method applies a set of features to represent the environment. The features can be discrete, continuous, or a mixture of both. The discrete feature is used to classify detected objects into different categories such as vehicles, pedestrians or cyclists [ZSB04]. A list of bounding boxes over different time steps is frequently used to describe the current and future states of detected objects with continuous variables like position, orientation, and velocity [LYU18]. The prediction for one object can also contain multiple intentions. In such cases, the feature list includes both discrete features such as intentions and estimated probabilities to indicate whether a vehicle intends to turn right or left, as well as continuous variables to represent the trajectories associated with each intention [Cui+19].

2. Grid-based Method

Another common representation is OGM [Elf13]. An OGM discretizes the road surface into grid cells, where each grid cell shows whether an obstacle is present. An OGM grid cell can be expanded to include more information, such as object classification, to allow for more sophisticated decision-making [Sch18]. However, a single OGM is insufficient to represent moving objects in a dynamic environment. [AHM02] extends the OGM to temporal occupancy grids with several layers, where each layer shows which cells were occupied during a specific time interval. MOGM is used to represent predictions, with each OGM representing a time step of the prediction [MR19].

Decision-making Under Uncertainty

As introduced in the Chap. 2, a POMDP is a method for decision-making under uncertainty [KLC98]. Various sources of uncertain information can be taken into account when applying POMDP to autonomous driving for handling complex driving scenarios in urban environments.

1. Uncertain Measurements and Predictions

Uncertain measurements caused by sensor noise are the most commonly considered source of uncertainty when modeling the POMDP as a behavior planner. The state uncertainty is typically represented by a Gaussian distribution [Wei+11; BHL14]. The prediction uncertainty increases over time. Several studies explicitly include prediction uncertainty in the POMDP planning process [Liu+15; Hub+17].

2. Uncertain Intention and Interaction

Recent research focuses on optimizing driving strategies involving uncertain intentions and interactions with other road users, as well as providing driving strategies that account for possible future observations of scenarios. The ability of the POMDP formulation has been demonstrated for scenarios where the ego vehicle needs to negotiate and interact with other road users, such as on-ramp merging [Wei+11], lane changes [UM15; Hub+18b], unsignalized intersections [BCK17; Sef+17; Hub+18a], roundabouts [Liu+15; Bey+21], crosswalks and bus stops [Sch+19b; Tho+18].

3. Uncertain Appearance

Urban environments involve static and dynamic objects that limit the FoV of the ego vehicle. Previous works expand the POMDP model to account for the uncertain probability of potentially occluded vehicles and pedestrians appearing in such environments [Lin+19b; Sch+19a; Hub+19; WGW21; Zha+21]. As introduced in Chap. 3 and Chap. 4, the basic idea is to introduce phantom road users to represent the potentially existing pedestrians and vehicles which have certain probabilities of appearing from occluded areas. The phantom road users are incorporated into a POMDP formulation's state space and probabilistic transition model.

4. Scalability of the POMDP Planner

POMDP problems face both the curse of dimensionality and the curse of history [PGT06], making it very hard to scale up with a large POMDP model. However, recently developed open-source POMDP solvers based on Monte Carlo tree searches have been released to address this issue [SV10; Som+13; KSK14]. Parallel computing with CPU and GPU improves the POMDP solver efficiency even further [Cai+21]. The authors of [Cai+19] incorporate a learning-based heuristic to guide the tree construction process of the POMDP solver, enabling more efficient planning without searching too deep. Besides targeting solver, another direction for enhancing the scalability of POMDP planners is the model design. The authors of [Zha+20] reduce the computational cost of POMDP-based decision-making algorithms by utilizing domain knowledge in a policy tree. However, because the POMDP model and solver are coupled in one algorithm, their approach cannot benefit from further POMDP solver developments.

5.3 Approach: Efficient Environment Model Using Multi-step Occupancy Maps

This chapter introduces a scalable behavior planner approach using an efficient MOGM-based POMDP model. It first explains the structure and the generation of the MOGM. Secondly, the POMDP model based on the MOGM is introduced.

5.3.1 Multi-step Occupancy Grid Maps

The approach follows the assumption that the estimated intentions and their corresponding future states of each road user are fixed in one planning cycle [Lin+19b]. This information is delivered from the prediction module of the autonomous vehicle, where the future states

of other traffic participants are represented as bounding boxes. The corners of the bounding boxes are represented by the Frenet coordinates s and d , where s indicates the arc-length along the ego lane center, and d is the orthogonal distance to the ego lane center, as shown in Fig. 5.2.

Using the intentions and bounding boxes, the approach constructs an MOGM to represent all possible intentions and future states of road users. In order to account for uncertain measurements and predictions, the bounding boxes of road users are extended with risk areas approximated by a two-dimensional Gaussian function. The probabilities of the grid cells being occupied by the bounding boxes are stored in the MOGM.

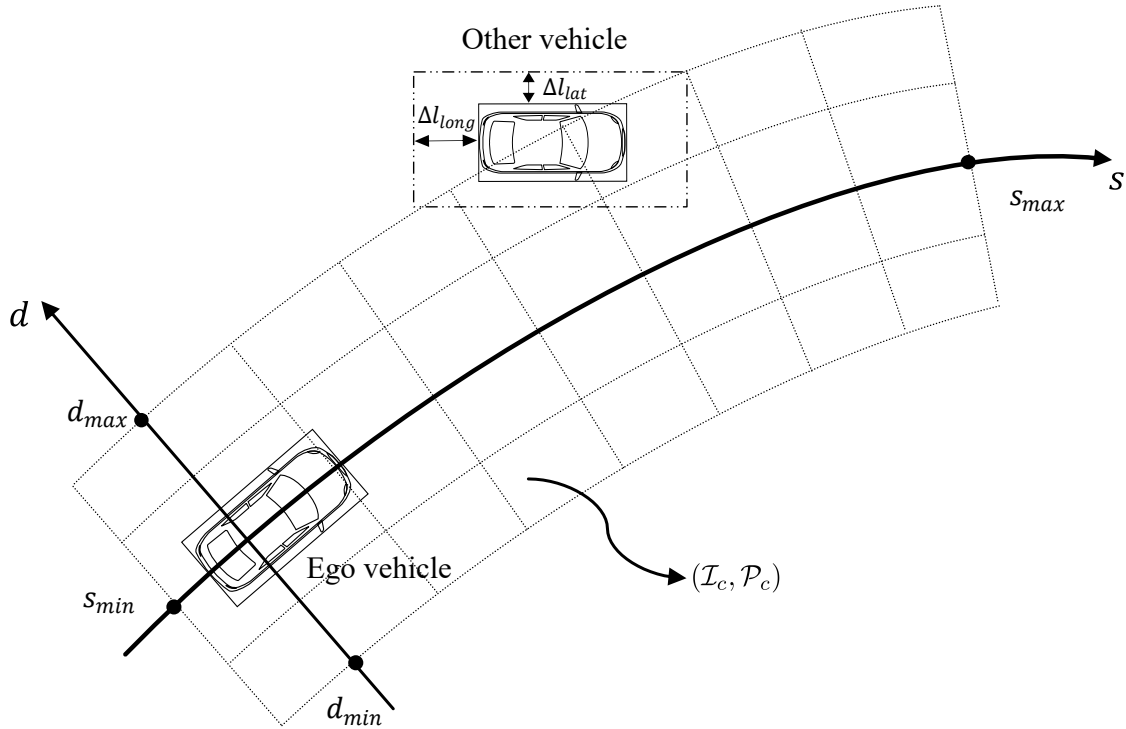


Figure 5.2: Construction of the MOGM. The ego vehicle and the predicted states of other vehicles are represented by bounding boxes. Δl_{long} and Δl_{lat} indicate the extended risk area (graphic from [Zha+22b], ©2022 IEEE).

Uncertainty approximation using Gaussian function

The occupancy probability p_c of an MOGM grid cell is set to one when it is occupied by bounding boxes. The approach considers the uncertainty of the measurements and predictions by extending the bounding box with risk areas (see Fig. 5.2). The occupancy probability p_c is approximated by a two-dimensional Gaussian function:

$$p_c = \exp\left(-\left(\frac{(\Delta l_{long})^2}{2\sigma_{long}^2} + \frac{(\Delta l_{lat})^2}{2\sigma_{lat}^2}\right)\right), \quad (5.1)$$

where $\Delta l_{long}, \Delta l_{lat} \in \mathbb{R}$ are the longitudinal and lateral distances to the bounding box boundary, and the variances $\sigma_{long}, \sigma_{lat} \in \mathbb{R}$ determine the risk distribution around the road users' bounding boxes in the longitudinal and lateral directions, respectively. In this thesis, the values of σ_{long} and σ_{lat} are selected as $\sigma_{long} = 1$ and $\sigma_{lat} = 0.5$.

In order to improve the representation of the risk distribution around the bounding boxes, the bounding box of a road user is extended with N_b longitudinal and lateral extensions $\Delta\mathcal{L}_{long}$, $\Delta\mathcal{L}_{lat}$, where $N_b \in \mathbb{N}^+$, $\Delta\mathcal{L}_{long}, \Delta\mathcal{L}_{lat} \in \mathbb{R}^{N_b}$.

Multiple Intentions

Every road user $i \in \{1, \dots, N\}$, $N \in \mathbb{N}^+$, has been assigned a set of intentions $\mathcal{I}^i = \{l_1^i, \dots, l_j^i\}$, where $J \in \mathbb{N}^+$ is the number of intentions of road user i (see Fig. 5.1). Each intention is associated with a set of predicted states over time. The approach stores $l_j^i \in \mathcal{I}^i$, $j \in \{1, \dots, J\}$ in the grid cells of the MOGM, if they geometrically overlap (i.e., are occupied) with the bounding boxes of predicted states from the intention l_j^i .

MOGM Generation

The planning time horizon $H \in \mathbb{R}^+$ is discretized into $M \in \mathbb{N}$ steps. As shown in Fig. 5.2, the OGM \mathcal{M}_m at time step $m \in \{0, \dots, M\}$ is constructed on the ego lane center. In the longitudinal direction, \mathcal{M}_m starts at s_{min} and ends at s_{max} along the ego lane center, while in the lateral direction, \mathcal{M}_m covers d_{max} to the left and d_{min} to the right of the ego lane center.

A cell in \mathcal{M}_m can be occupied, partly occupied, or unoccupied. A partly occupied cell is considered as occupied. Formally, \mathcal{M}_m can be represented as a matrix of tuples:

$$\mathcal{M}_m = \begin{bmatrix} c_m(1, 1) & \dots & c_m(1, Y) \\ \vdots & \ddots & \vdots \\ c_m(X, 1) & \dots & c_m(X, Y) \end{bmatrix}, \quad (5.2)$$

where $X, Y \in \mathbb{N}^+$ indicate the number of rows and columns of \mathcal{M}_m . Each grid cell $c_m \in \mathcal{M}_m$ stores a tuple $(\mathcal{I}_{c_m}, \mathcal{P}_{c_m})$, $\mathcal{I}_{c_m} \subseteq \bigcup_{i=0}^N \mathcal{I}^i$ is a set of intentions, from which the predicted states occupy the grid cell c_m , and \mathcal{P}_{c_m} is a set of occupancy probabilities calculated by the predicted states from \mathcal{I}_{c_m} .

The generation of \mathcal{M}_m is described in Alg. 5. Initially, an empty OGM is constructed such that intentions \mathcal{I}_{c_m} and occupancy probability \mathcal{P}_{c_m} of each grid cell are empty sets (line 1). The predicted states at time step m from all intentions of each road user is obtained (see lines 2 to 4). Each predicted state is represented by a bounding box bb_m (line 5). Next, the algorithm extends bb_m to a set of bounding boxes \mathcal{BB}_m with N_b longitudinal and lateral extensions $\Delta\mathcal{L}_{long}$, $\Delta\mathcal{L}_{lat}$ (line 6). For each grid cell c_m that is occupied by bb_m , the intention and occupancy probability $p_{c_m, t} = 1$ are inserted into the intention set \mathcal{I}_{c_m} and occupancy probability set \mathcal{P}_{c_m} , respectively (see lines 7 to 12). If c_m is not occupied by b_m , the algorithm checks whether any bounding boxes in \mathcal{BB}_m occupy the grid cell c_m and inserts the maximum occupancy probability $p_{c_m, t}$ calculated by (5.1) to \mathcal{P}_{c_m} and the intention to \mathcal{I}_{c_m} (see lines 13 to 16).

5.3.2 POMDP Behavior Planning

State and Observation Space

The state space is a representation of the driving scenario in the POMDP model. It contains the state of the ego vehicle x_{ego} and the state of other road users x_i . The state x is a vector:

$$x = [x_{ego}, x_1, x_2, \dots, x_N]^T, \quad (5.3)$$

where $x_{ego} = [x_{ego}, y_{ego}, \theta_{ego}, v_{ego}, r_{ego}]^T$ is the ego state which includes the position (x_{ego}, y_{ego}) , orientation θ_{ego} , speed v_{ego} and intended driving path r_{ego} . The states x_i with $i \in \{1, 2, \dots, N\}$, $N \in$

Algorithm 5: OGM Generation for Time Step m

Input : N road users, longitudinal extensions $\Delta\mathcal{L}_{long}$, lateral extensions $\Delta\mathcal{L}_{lat}$
Output: OGM \mathcal{M}_m

```

1  $\mathcal{M}_m \leftarrow \text{initializeOGM}(m)$ 
2 foreach road user  $i \in \{1, \dots, N\}$  do
3    $\mathcal{I}^i \leftarrow \text{getIntentions}(i)$ 
4   foreach  $\iota \in \mathcal{I}^i$  do
5      $bb_m \leftarrow \text{getPredictedState}(\iota, m)$ 
6      $\mathcal{BB}_m \leftarrow \text{extendBoxes}(bb_m, \Delta\mathcal{L}_{long}, \Delta\mathcal{L}_{lat})$ 
7     foreach  $c_m \in \mathcal{M}_m$  do
8        $\mathcal{I}_{c_m}, \mathcal{P}_{c_m} \leftarrow c_m$ 
9       if  $\text{isOccupied}(c_m, bb_m)$  then
10         $p_{c_m, \iota} \leftarrow 1$ 
11         $\text{addToList}(\mathcal{I}_{c_m}, \iota)$ 
12         $\text{addToList}(\mathcal{P}_{c_m}, p_{c_m, \iota})$ 
13       else if  $\text{isOccupied}(c_m, \mathcal{BB}'_m)$  then
14         $p_{c_m, \iota} \leftarrow \text{maxRisk}(\Delta\mathcal{L}_{long}, \Delta\mathcal{L}_{lat})$ 
15         $\text{addToList}(\mathcal{I}_{c_m}, \iota)$ 
16         $\text{addToList}(\mathcal{P}_{c_m}, p_{c_m, \iota})$ 
17       end if
18     end foreach
19   end foreach
20 end foreach
21 return  $\mathcal{M}_m$ 

```

\mathbb{N}^+ represent road users in the scenario. Optionally, phantom objects introduced from Chap. 3 and Chap. 4 can be added to the state space to allow the model to plan driving behavior while taking into account the uncertain probability that potentially occluded traffic participants will appear. To represent other road users, an MOGM-based state model is introduced. The state of a road user i is defined as $x_i = \iota_j^i$, $\iota_j^i \in \mathcal{I}^i$, $j \in \{1, \dots, J\}$, $J \in \mathbb{N}^+$, where ι_j^i is one of the intentions of the road user i , e.g., turn left or right at an intersection.

Action

The approach follows the path-velocity decomposition method [KZ86] for planning the longitudinal driving policies along the intended ego path r_{ego} . Possible driving behaviors of the ego vehicle are represented by longitudinal accelerations:

$$A = \{+1.5 \text{ m/s}^2, 0 \text{ m/s}^2, -1.5 \text{ m/s}^2\}. \quad (5.4)$$

State Transition with MOGM

The state transition from the current state x to the next state x' is determined by the transition models of the ego vehicle and other road users. The intended ego path remains unchanged such that $r'_{ego} = r_{ego}$. A point mass model is applied in (5.5) to predict the ego movement along the intended ego path r_{ego} . The position of the ego vehicle s_{ego} is the arc-length along r_{ego} . The new ego position s'_{ego} and velocity v'_{ego} are predicted by the chosen action a and the

step size Δt :

$$\begin{bmatrix} s'_{ego} \\ v'_{ego} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} s_{ego} \\ v_{ego} \end{bmatrix} + \begin{bmatrix} \frac{1}{2}\Delta t^2 \\ \Delta t \end{bmatrix} a. \quad (5.5)$$

The x'_{ego} , y'_{ego} and θ'_{ego} in the updated ego state x'_{ego} are obtained by getting the Cartesian position from r'_{ego} according to the path geometry based on the position s'_{ego} . With the assumption that a road user's intention does not change within a planning cycle, the states of road users x_i remain the same in all planning steps.

Algorithm 6: Calculate Collision Probability at Time Step m

Input : OGM \mathcal{M}_m , State $x = [x_{ego}, x_1, \dots, x_N]^T$
Output: Collision probability $p_{collision}$

- 1 $bb_{ego} \leftarrow \text{buildBoundingBox}(x_{ego})$
- 2 $\mathcal{C}_{m,ego} \leftarrow \text{findOccupiedCells}(bb_{ego}, \mathcal{M}_m)$
- 3 $\mathcal{P}_{collision} \leftarrow \emptyset$
- 4 **foreach** occupied cell $c_{m,ego} \in \mathcal{C}_{m,ego}$ **do**
- 5 $\mathcal{I}_{c_m}, \mathcal{P}_{c_m} \leftarrow c_{m,ego}$
- 6 **foreach** road user $i \in \{1, \dots, N\}$ **do**
- 7 $\iota \leftarrow x_i$
- 8 **if** $\iota \in \mathcal{I}_c$ **then**
- 9 $p_{c_m, \iota} \leftarrow \mathcal{P}_{c_m}$
- 10 $\text{addToList}(\mathcal{P}_{collision}, p_{c_m, \iota})$
- 11 **end if**
- 12 **end foreach**
- 13 **end foreach**
- 14 $p_{collision} \leftarrow \max(\mathcal{P}_{collision})$
- 15 **return** $p_{collision}$

Terminal Condition Check with MOGM

When a state transitions to the next state x' , a terminal condition check for x' is needed. A state is considered terminated when either the planning horizon is reached during the construction of the belief tree or a collision occurs between the ego vehicle and a road user. Alg. 6 performs the calculation of the collision probability. Firstly, the ego state is matched to the occupied cells $\mathcal{C}_{m,ego}$ in the OGM \mathcal{M}_m at time step m (see lines 1 and 2). Then, the collision probability is checked for every cell $c_{m,ego}$ that is occupied by the ego vehicle (line 4). For each road user i , the intention ι is obtained from x_i (see lines 6 and 7). If $c_{m,ego}$ contains the intention ι of road user i , the collision probability $p_{c_m, \iota}$ is saved in the list $\mathcal{P}_{collision}$ (see lines 8 to 10). Finally, the maximum collision probability $p_{collision}$ from the list $\mathcal{P}_{collision}$ is returned (see lines 14 and 15). If $p_{collision} = 1$, the state is denoted as the terminal state.

Reward

The reward function of the approach includes the objectives of safety, speed, and comfort:

$$R = R_{safety} + R_{speed} + R_{comfort}. \quad (5.6)$$

To emphasize safety, rewards are assigned depending on the collision probability $p_{collision}$:

$$R_{safety} = \begin{cases} -100000, & \text{if } p_{collision} = 1 \\ -10000 \cdot p_{collision}, & \text{otherwise.} \end{cases} \quad (5.7)$$

Table 5.1: Applied parameters in the simulation (table from [Zha+22b], ©2022 IEEE).

Parameter	Value	Description
d_{min}	-5 m	Min. lat. coordinate on the MOGM
d_{max}	5 m	Max. lat. coordinate on the MOGM
s_{min}	-20 m	Min. long. coordinate on the MOGM
s_{max}	80 m	Max. long. coordinate on the MOGM
X	200	Number of the MOGM rows
Y	20	Number of the MOGM columns
$\Delta\mathcal{L}_{long}$	{1.5, 3.0}	Long. extensions of bounding box
$\Delta\mathcal{L}_{lat}$	{0.5, 1.0}	Lat. extensions of bounding box
F	2 Hz	Planning frequency
γ	0.95	Discount factor
D	10	Maximal tree depth
H	10 s	Planning horizon
M	11	Number of planning time steps

The ego vehicle is also encouraged to maintain the desired velocity $v_{desired}$ following the ego path r_{ego} :

$$R_{speed} = \begin{cases} -200(v_{desired} - v_{ego}), & \text{if } v_{desired} \geq v_{ego} \\ -2000|v_{desired} - v_{ego}|, & \text{otherwise.} \end{cases} \quad (5.8)$$

To obtain comfortable driving policies, acceleration is penalized with $R_{comfort} = -300 \cdot a^2$.

5.4 Experiments and Results

5.4.1 Experiment Setup

This section evaluates the approach on a system with an Intel Core i7-6820HQ CPU running at 2.70 GHz. The evaluations are carried out in the same simulation platform used in previous chapters. The feature-based POMDP model (TAPIR-POMDP) without occlusion awareness is selected as a baseline planner. The approach from this chapter – a POMDP behavior planner with the MOGM-based POMDP model – is denoted as MOGM-POMDP. To demonstrate the planner’s ability to plan driving policies in a dense urban environment with occlusions, a MOGMOA-POMDP is created by extending the MOGM-based POMDP model with occlusion awareness, as proposed in Chap. 3. A supplementary video¹ of the evaluated scenarios is provided.

The parameters applied in the simulation are listed in Table 5.1. Firstly, the average generation time of MOGM with different numbers of objects is evaluated. Then, a dense traffic scenario is set up to conduct a qualitative evaluation for comparing driving strategies

¹Video: <https://github.com/GitChiZhang/MOGM-POMDP>

Table 5.2: MOGM generation times (table from [Zha+22b], ©2022 IEEE).

Num. object	10	20	30	40	50	60	70
Aver. time (ms)	6.4	11.4	18.2	25.0	29.7	36.4	43.2

using the aforementioned planners. To compare the computational efficiency of the MOGM-POMDP with that of the TAPIR-POMDP, the computation time for generating a new episode when constructing the belief tree with different numbers of road users is measured. The speedup is defined as the average computation time of the MOGM-POMDP compared to that of the TAPIR-POMDP. Furthermore, the overall performance is illustrated by comparing the number of active belief nodes in the belief tree within a planning cycle. The speedup is used to compare the average number of active nodes of the MOGM-POMDP to that of the TAPIR-POMDP.

5.4.2 Evaluation of the Efficient POMDP Planner

MOGM Generation Time

To evaluate the computation time of the MOGM, a setup with 10 to 70 objects located on the ego lane is used. In this evaluation, the simulation is configured in such a way that all objects are detected and provided to generate the MOGM.

Table 5.2 shows the average generation time of MOGM in one planning cycle. It can be seen that the MOGM generation time ranges from 6.4 *ms* to 43.2 *ms*, increasing linearly with the number of objects involved in the scenario.

Performance in a Dense Urban Environment

Fig. 5.3 illustrates a dense urban scenario with buildings alongside the street and lots of pedestrians walking around. The total number of static and dynamic objects in the scenario is 80. In this scenario, there are several pedestrians crossing a crosswalk. One pedestrian, not on the crosswalk, will abruptly cross the street from the middle of the road. The walking speed of all pedestrians ranges from 0.1 *m/s* to 1.5 *m/s* to reflect the diversity of the traffic situations. Apart from the pedestrians, there is a parked vehicle near the crosswalk that obscures the ego vehicle's FoV.

A qualitative evaluation of the planners is performed in the aforementioned scenario. Due to a large number of road users in the scenario, the TAPIR-POMDP is unable to sample enough belief nodes to provide safe policies for the given planning frequency of 2 Hz. In contrast to the TAPIR-POMDP, both the MOGM-POMDP and MOGMOA-POMDP can provide safe driving policies to navigate through the scenario. Fig. 5.4 shows the acceleration and velocity profile of the planned driving strategies. At time $t = 4.10$ s, the MOGM-POMDP stops accelerating and reacts to the abruptly crossing pedestrian by further reducing the speed. The MOGM-POMDP then waits for pedestrians crossing the road before driving through the crosswalk. The MOGMOA-POMDP demonstrates various driving strategies with occlusion awareness for dealing with the same scenario. At time $t = 17.50$ s, the MOGMOA-POMDP continues to drive cautiously to reduce the risk of pedestrians suddenly appearing from the blind spot of the ego vehicle. Once the ego vehicle has a sufficient FoV, it accelerates and drives through the crosswalk.

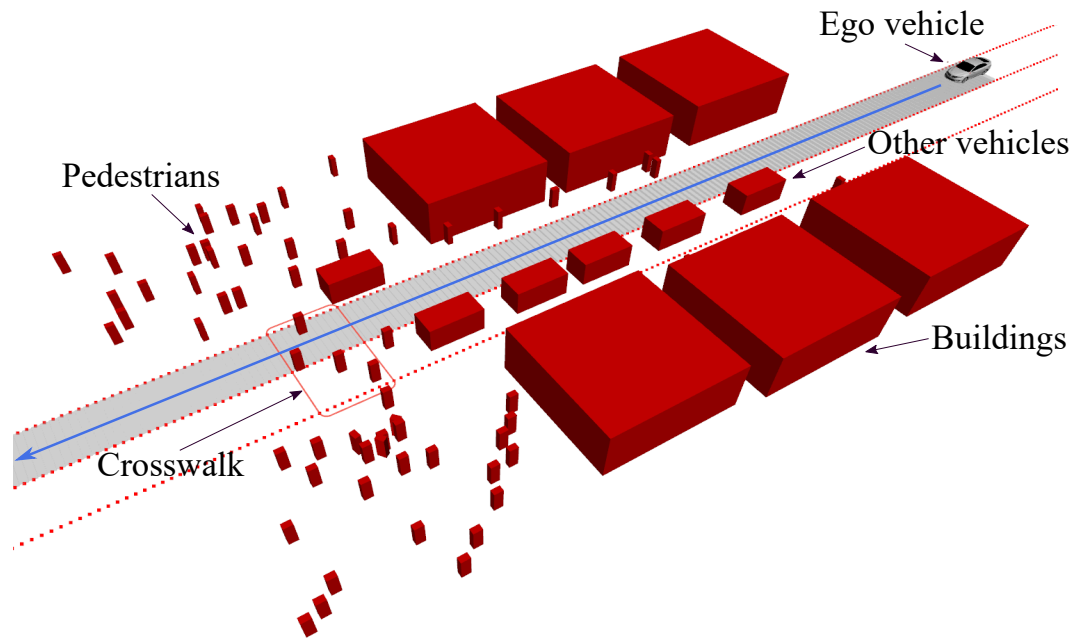


Figure 5.3: The ego vehicle intends to drive through the dense urban environment with buildings along the street and many pedestrians moving around. The total number of static and dynamic objects in the scenario is 80 (graphic from [Zha+22b], ©2022 IEEE).

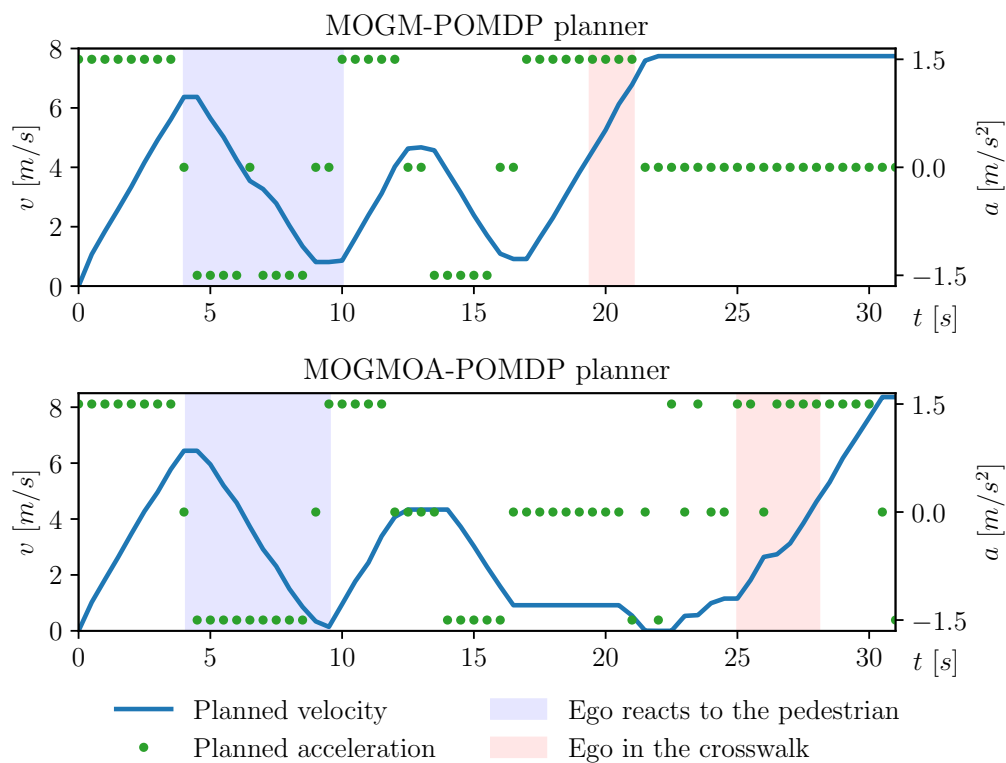


Figure 5.4: Comparison of planned driving strategies for handling the dense urban environment (graphic from [Zha+22b], ©2022 IEEE).

Effect on Episode Generation

In this evaluation, the computation time for generating a new episode in the tree construction is measured. As shown in Fig. 5.5, the MOGM-POMDP significantly reduces the average computation time for generating a new episode. The MOGM-POMDP's mean computation time varies from $2.5 \mu s$ to $3.5 \mu s$ depending on the number of objects in the state space, whereas the TAPIR-POMDP has a mean computation time of $50 \mu s$ to $600 \mu s$. It can also be seen that increasing the number of objects in the state space has a larger impact on the TAPIR-POMDP. Consequently, the MOGM-POMDP achieves a speedup of 50 to 200 times over the TAPIR-POMDP in calculating a new episode. There are two reasons for the improvement. First, the state space and observation space of the MOGM-POMDP are smaller than those of the TAPIR-POMDP, which reduces the matching time of a particle to the belief tree. Second, unlike the TAPIR-POMDP, which performs collision checks by simulating the interaction between each ego vehicle and object pair, the MOGM-POMDP only checks the new ego vehicle state within the corresponding occupancy map.

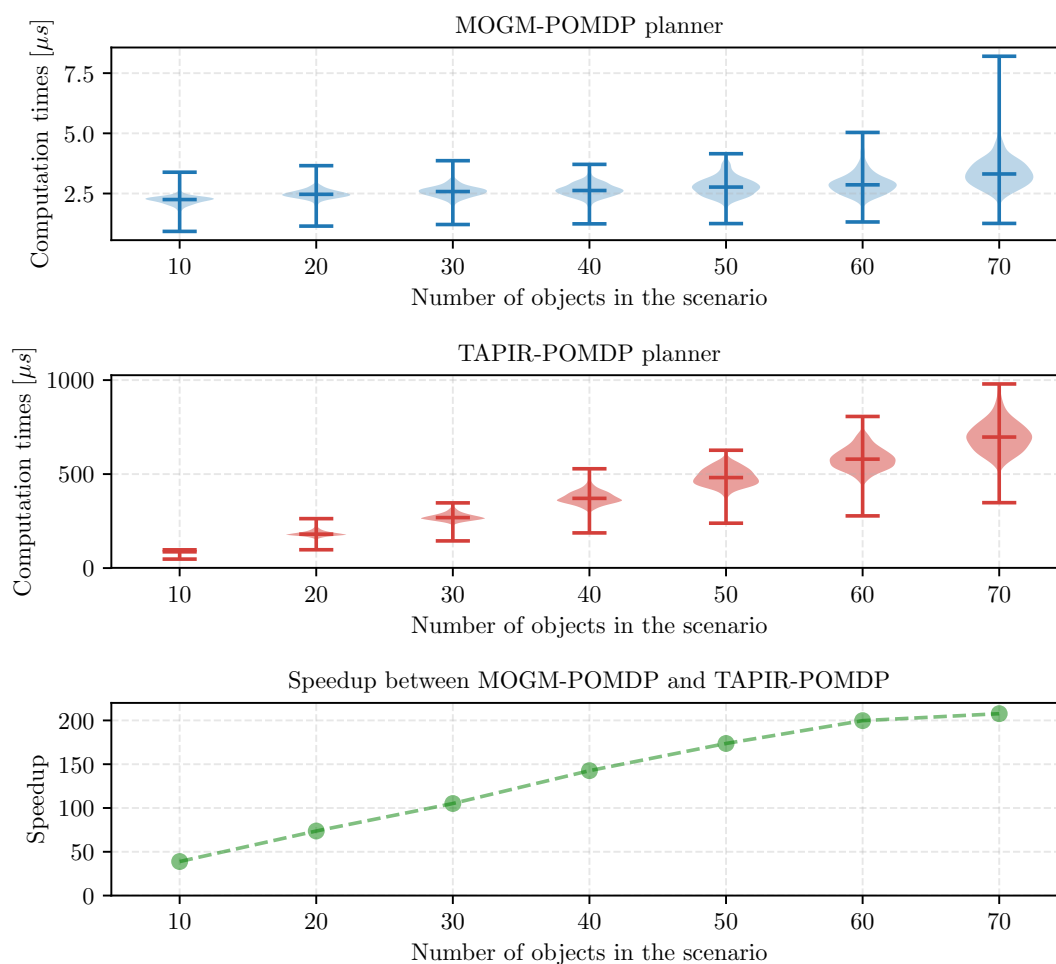


Figure 5.5: Comparison of the computation time of the MOGM-POMDP model and the TAPIR-POMDP model for generating a new episode. The speedup is calculated by comparing the average computation time of the MOGM-POMDP to that of the TAPIR-POMDP (graphic from [Zha+22b], ©2022 IEEE).

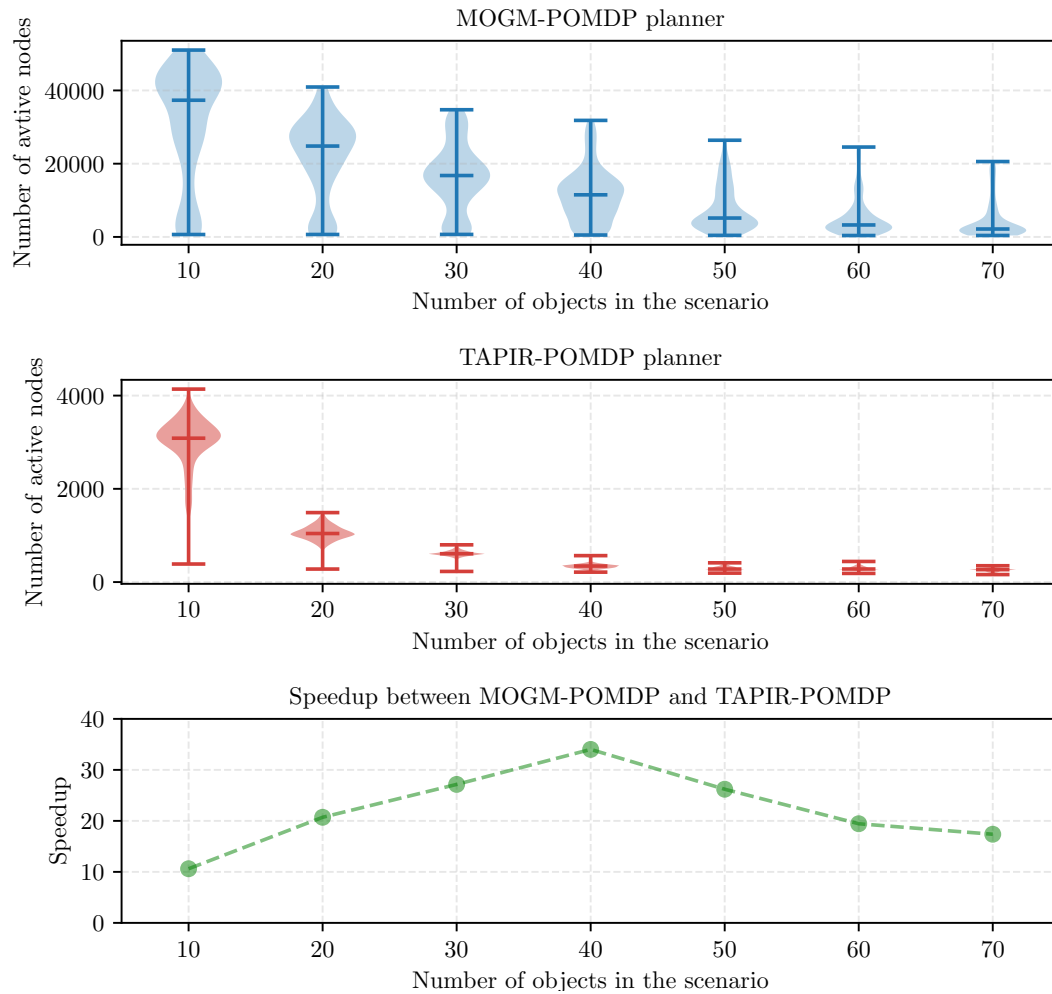


Figure 5.6: Comparison between the active nodes of the MOGM-POMDP and TAPIR-POMDP in the belief tree construction. The speedup is calculated by comparing the average number of active nodes of the MOGM-POMDP to that of the TAPIR-POMDP (graphic from [Zha+22b], ©2022 IEEE).

Effect on Belief Tree Construction

The evaluation result in Fig. 5.6 shows that for both the MOGM-POMDP and TAPIR-POMDP, the number of active belief nodes decreases as the number of road users involved in the state space increases. The performance of TAPIR-POMDP drops dramatically when more than ten objects are considered. The MOGM-POMDP outperforms the TAPIR-POMDP by a factor of 10 to 35. Major performance gains are seen when the number of objects is between 20 and 60 with a speedup of between 20 and 35.

5.5 Summary

In this chapter, an efficient POMDP planner that can plan safe driving behaviors in dense urban environments is presented. The uncertainties in the measurements, predictions and intentions of surrounding road users are incorporated into the MOGM to enable intention-aware planning with consideration for the risk associated with the uncertainties. Further-

more, an efficient POMDP model combining MOGM is introduced as a black box simulator for the POMDP online solver. According to the evaluation results, the MOGM-based POMDP model is approximately 50 times faster than the baseline POMDP model. The overall performance of the approach improves on the performance of the baseline POMDP planner by a factor of at least 10. Simulations show that the MOGM-POMDP planner can plan safe driving policies in an urban environment involving a large number of road users.

6

Safe and Rule-Aware Behavior Planning

Previous chapters address the behavior planning problem as a POMDP model and solve it using ABT online solving method. Another way to solve the model is using Deep Reinforcement Learning (DRL), which eliminates the need for implementing a mathematical model for the ego vehicle and the environment. When solving the planning behavior problem with DRL, the DRL agent learns from the experience of interacting with the environment without the need to model the environmental dynamics explicitly. However, planning driving behavior for autonomous vehicles using DRL in urban environments faces challenges in designing DRL models to obey the traffic rules and provide safe driving behaviors instead of only focusing on getting high rewards. Therefore, this chapter presents a DRL behavior planner for autonomous vehicles to handle intersection scenarios in urban environments while considering traffic rules. Furthermore, the safety of the DRL algorithm's decisions is enhanced by introducing a safety checker. This chapter is based on the author's previously published work [Zha+22a].

6.1 Overview and Contributions

In recent years, numerous studies have focused on applying DRL in the decision-making tasks of autonomous vehicles since it has successfully demonstrated superhuman level performance in the fields of robotics and video games [Ara22]. DRL facilitates learning of optimal long-term policies solving complex driving tasks, such as the handling of intersections, roundabouts, or lane changes on highways. Examples are presented by the authors of [Kam+20; QSD21; CBM20; Bou+20], who argue that DRL agents behave less conservatively compared to rule-based methods because of their negotiation and interaction capabilities.

This chapter focuses on applying DRL to learn driving policies for unsignalized intersections. In some countries, such as Germany, drivers must follow the right-before-left rule at intersections without traffic signs or traffic lights. For example, the ego vehicle must give way to the vehicles approaching from the right since they have higher priority (see Fig. 6.1). The DRL algorithm needs to safely drive through the intersection without harming other road users, while following the traffic rules. To train the DRL agent, a simulation environment is required that can provide high-fidelity vehicle dynamics for the ego vehicle, while simulating other vehicles that can comply with the right-before-left rule. The goal of this chapter is to address the aforementioned challenges to advance the safe and rule-compliant DRL algorithm toward real-world applications.

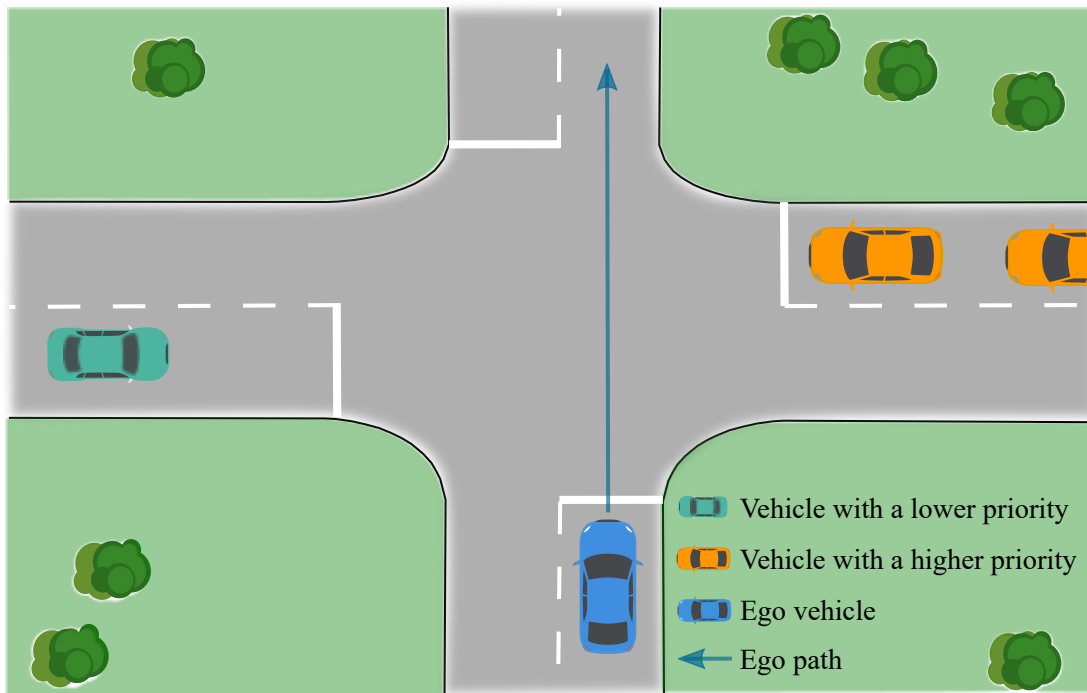


Figure 6.1: The figure shows an intersection without traffic signs and traffic signals. The ego vehicle (blue car) intends to drive through the intersection. The yellow vehicles have priority over the ego vehicle, whereas the ego vehicle has the right of way over the green vehicle (graphic from [Zha+22a], ©2022 IEEE).

Driving safely and obeying traffic rules are fundamental requirements for applying the DRL algorithm in the real world. This chapter proposes a safe and rule-aware DRL approach for high-level decision making in an intersection scenario for autonomous driving. First, a co-simulation environment is established using CAR Learning to Act (CARLA) and Simulation of Urban Mobility (SUMO), which provides high-fidelity vehicle dynamics for the ego vehicle while simulating other vehicles that adhere to traffic rules, such as the right-before-left rule at intersections. Furthermore, a traffic rule monitor is introduced that checks the priority of the DRL-based ego vehicle according to the right-before-left rule. In addition to this, a safety checker based on the Responsibility-Sensitive Safety (RSS) model [SSS17] is applied. The safety checker verifies the status of the ego vehicle and provides a fallback safe action for the DRL agent in unsafe situations during the training and inference phases.

The contributions of this chapter are summarized as follows:

- the introduction of a traffic rule monitor for detecting the compliance of the DRL agent with traffic rules,
- the application of an RSS-based safety checker to guarantee safety during the training and inference phases,
- the evaluation of different approaches for combining the traffic rule monitor and the safety checker to achieve safe and rule-compliant intersection driving with the DRL approach.

6.2 Related Work

Simulator

Several simulators have been designed specifically to develop autonomous driving systems and train DRL agents, as reviewed in [Ara22]. Among them, "Simulation of Urban MObility" (SUMO) [Lop+18] and "Car Learning to Act" (CARLA) [Dos+17] are representative examples that have active developer teams and communities. By utilizing the traffic-management and intersection model of SUMO, users are able to apply traffic rules when defining the behavior of simulated road users. However, SUMO does not support the simulation of road user's physical model. In comparison, CARLA can simulate vehicles using a high-fidelity vehicle dynamics model.

Traffic Rules for DRL

For autonomous driving tasks, traffic rules have been considered mostly to design reward functions for DRL agents. In highway scenarios, negative rewards are assigned if the DRL agent deviates from the center of its lane [Hue+19; Che+19a]. Compared to highway scenarios, unsignalized intersections in urban environments are more challenging to the incorporation of traffic rules during DRL agent training. Some works assume that the ego vehicle always has the lowest priority [Lia+18; Ise+18]. However, this simple assumption may cause issues such as overcautious driving behavior or deadlock situations. In [LM19], the DRL agent implicitly learns the right-of-way rule by interacting with other vehicles that comply with the traffic rules in the simulation environment. However, no explicit information about the road priority is represented in the state space for the agent. The deployment of the DRL agent is constrained to the same priority setup as that in which it was trained.

Instead of making simple assumptions, another way to consider traffic rules is to explicitly incorporate information into the state space or reward function of the DRL agent. In [Cap+21], stop lines and yield lines are represented in the state space using a grid map with different colors. The positions of other vehicles and their priority levels are also embedded within the state space. [LC18] uses the priority information in both the state space and reward function. The DRL agent is penalized when it does not wait for the vehicles in the prioritized lanes. In addition, whether other vehicles have right of way over the DRL agent is modeled into the state space. However, the priority definitions and violation conditions are not discussed. Moreover, the impact of the traffic rule rewards on the DRL agent is not investigated.

Safety for DRL

Because the DRL agent is employed in a safety-critical system, it should learn driving policies that both comply with traffic rules and ensure safety. Several techniques have been proposed to improve the safety of DRL agent decisions in autonomous driving tasks. One promising direction is to utilize a safety model checker to check the action space and only provide safe actions during the DRL agent's training. In [Liu+19] and [Mir+18], prior knowledge and constraints are used for the DRL agent to maintain a safe distance and avoid lane changes that result in close distances to other vehicles. In [KWA20], authors apply set-based predictions in a safety layer to remove the unsafe action candidates. [LKC18] introduces a safety DRL method that utilizes system dynamics and recursive feasibility techniques to construct a supervision module that guarantees safety during learning. In [Che+20], the authors apply regret theory to predict human drivers' lane-change decisions and use these predictions as safety constraints in the DRL training process. Instead of influencing the training process,

another type of approach combines traditional methods with DRL. In [Xio+16], the vehicle is first trained with DRL to learn the driving policy in a static environment without other vehicles. Afterwards, an artificial potential field is employed to avoid collision with other vehicles. [PK19] proposes a hybrid method that combines the approximate partially observable Markov decision process with DRL to enhance safety.

6.3 Approach: Incorporation of Safety and Traffic Rule Constraints

In this section, the safe and traffic rule-aware DRL framework is first briefly described. Next, the applied simulation environment is introduced. Following this, an explanation is provided for the traffic rule monitor responsible for checking compliance with the right-of-way rule and its integration into the state space and reward model of the DRL agent. Subsequently, the design of the safety checker, based on the RSS model, is discussed. Finally, the model of the DRL agent, including its state, action, and reward model, is explained.

6.3.1 Framework

This chapter proposes a DRL framework to make high-level decisions for autonomous vehicles to handle intersections in urban environments (see Fig. 6.2). The task of the DRL agent is to safely drive the ego vehicle across intersections in compliance with the right-before-left rule. To train the DRL agent, an intersection scenario is created using both CARLA and SUMO simulators. At each time step, the state of the ego vehicle is updated by CARLA, while the state of other road users is updated by SUMO. The environment is synchronized between both simulators and provides the current state to the traffic rule monitor and safety checker.

The DRL agent should understand its priority over other vehicles and obey the right-before-left rule at intersections. Therefore, a traffic rule monitor is introduced to provide the priority information $\varphi_{priority}$ containing the priority relationship between the ego vehicle and other vehicles. The traffic rule monitor further checks whether the ego vehicle violates this rule. The priority information $\varphi_{priority}$ and the result of monitoring the traffic rule violation $\varphi_{violation}$ are further incorporated into the state space $\mathcal{X}_{extend,t}$ and the reward function for the training of the DRL agent. The safety checker verifies whether the current state is safe according to the RSS model. Given the agent's action a_{agent} , the safety checker returns a tuple $(\varphi_{unsafe}, a_{safe})$, where φ_{unsafe} is a Boolean value to indicate whether the current state is unsafe and a_{safe} is a safe fallback action. If the DRL agent doesn't choose a brake action in an unsafe situation, the safety checker interferes with the DRL agent's decision and provides a safe fallback action a_{safe} as the final action a_{final} .

Finally, the control module converts the final action a_{final} to the control signal u for controlling the throttle or brake actuators in the autonomous vehicle. The reward function with traffic rule violation and unsafe information is used to guide the training process.

6.3.2 CARLA and SUMO Co-simulation

To train the DRL agent, a realistic simulation environment for testability and integration into real-world applications is required. The approach in this chapter applies SUMO to control road users so that they can comply with the right-of-way rule at the intersections. However, SUMO neither supports simulation of the vehicle's physical model nor includes sensor models. Therefore, the approach applies CARLA to simulate the environment and the ego vehicle with

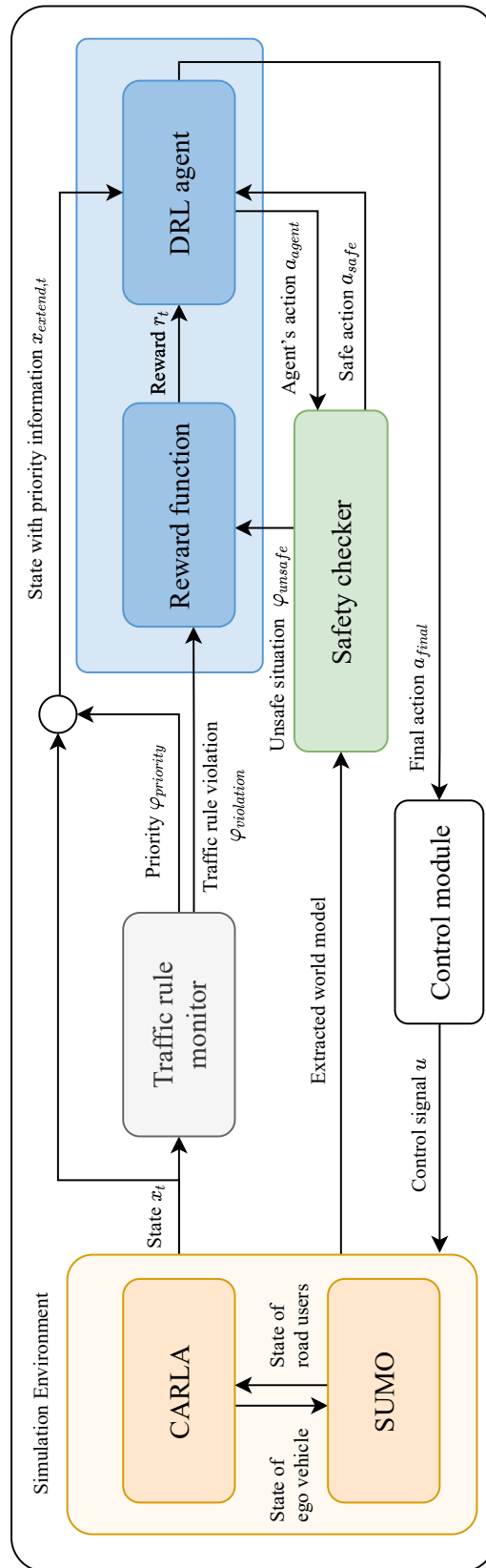


Figure 6.2: Safe and rule-aware framework for DRL decision-making module of the autonomous vehicle (graphic from [Zha+22a], ©2022 IEEE).



Figure 6.3: A scenario in the CARLA simulator (left) and SUMO simulator (right). The states of the ego vehicle (blue car) and other road users are synchronized between CARLA and SUMO (graphic from [Zha+22a], ©2022 IEEE).

its physical behavior model. As shown in Fig. 6.3, both CARLA and SUMO can interact with each other by synchronizing the states of the ego vehicle and the other road users.

6.3.3 Traffic Rule Monitor

The traffic rule monitor checks if other vehicles have priority at an intersection according to the right-of-way rule, and evaluates if the ego vehicle violates this rule. In the following, this evaluation is explained in more detail.

In order to determine if the ego vehicle violates the right-of-way rule, the vehicles driving on higher prioritized lanes with respect to the ego's current lane have to be identified. Only intersections are considered where the priority-relation between lanes does not change over time, i.e., intersections where no traffic lights are present. Thus, the priority-relation between lanes is obtained from a high-definition map.

For each intersection, a conflict area C is defined as the area enclosed by the stop lines (see Fig. 6.4). To determine if the ego vehicle violates the priority rule, only vehicles meeting the following priority condition are taken into account:

$$\varphi_{priority,i} = \{d_{c,i} \leq D_m\} \vee \left\{ \frac{d_{c,i}}{v_i} \leq T_a \right\}, \quad (6.1)$$

where $d_{c,i}$ is the Euclidean distance of vehicle i to the conflict area and v_i is its velocity. This chapter defines the arriving time threshold as $T_a = 3$ s and the monitoring range as $D_m = 30$ m.

The traffic rule monitor returns $\varphi_{violation} = True$ if the ego vehicle has a geometrical overlap with the conflict area and if there is at least one vehicle i , for which the condition $\varphi_{priority,i} = True$ holds.

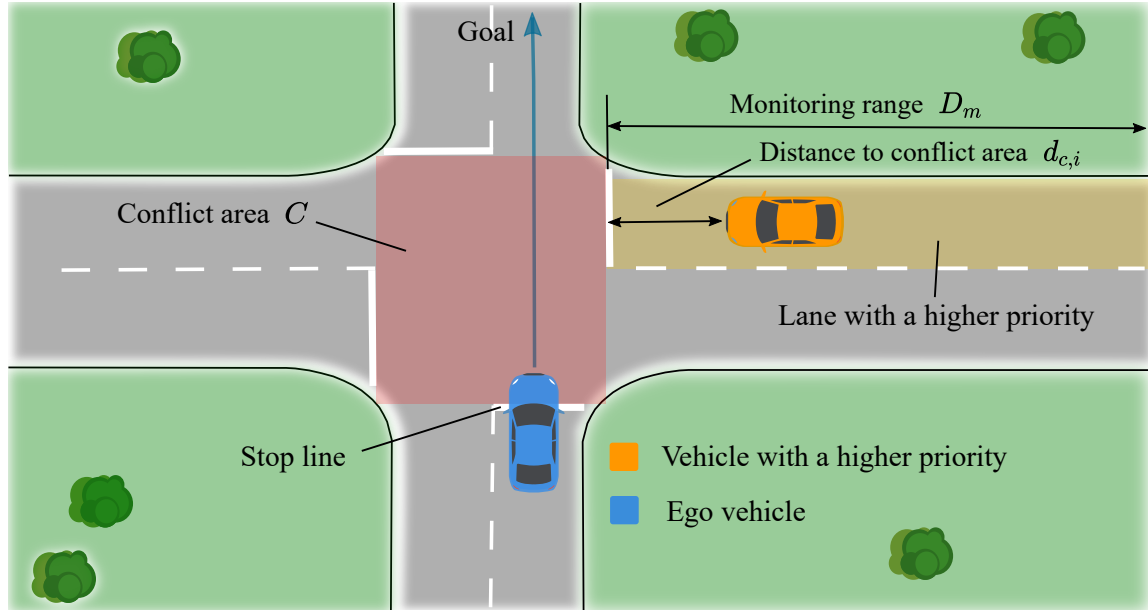


Figure 6.4: Example of the ego vehicle violating the traffic rule in a right-before-left intersection (graphic from [Zha+22a], ©2022 IEEE).

6.3.4 Safety Checker

The objective of the safety checker is to identify unsafe situations φ_{unsafe} and provide a proper reaction a_{safe} to ensure safety while guiding the DRL agent to learn how to behave properly in such situations. The open-source C++ library for RSS [Gas+19] is integrated within the safety checker in the proposed DRL framework. The RSS model formalizes safe driving by defining a set of logical rules and mathematical formulas such as safe distance to other vehicles (see [SSS17] for more details). Using these formalizations, the RSS model can identify dangerous situations, which are denoted as $\varphi_{dangerous}$. The states of surrounding objects and the ego vehicle are extracted based on the simulation environment. As shown in Fig. 6.5, the extracted world model is the input for the RSS model to identify whether the situation is dangerous, i.e., $\varphi_{dangerous} = True$.

The action selection module is applied to decide whether to interfere with the agent's decision. In cases where the situation is safe according to the RSS model ($\varphi_{dangerous} = False$), or the DRL agent chooses a brake action a_{brake} when $\varphi_{dangerous} = True$, the action selection module forwards the DRL agent's action without any interference, such that $a_{safe} = a_{agent}$, and the checking result φ_{unsafe} is set to *False*. Otherwise, if the agent does not choose a braking action in a dangerous situation $\varphi_{dangerous} = True$, the action selection module in the safety checker sets $\varphi_{unsafe} = True$ and overwrites the agent's action with the braking action $a_{safe} = a_{brake}$. As a result, the safety checker provides the identified unsafe situation φ_{unsafe} to the reward function and the action a_{safe} to the DRL agent.

6.3.5 DRL Model

State Space

The state representation of the DRL agent is a feature list containing information about the ego vehicle and other road users. Five vehicles with the closest Euclidean distance to the conflict area of the intersection are included. The state representation with priority information

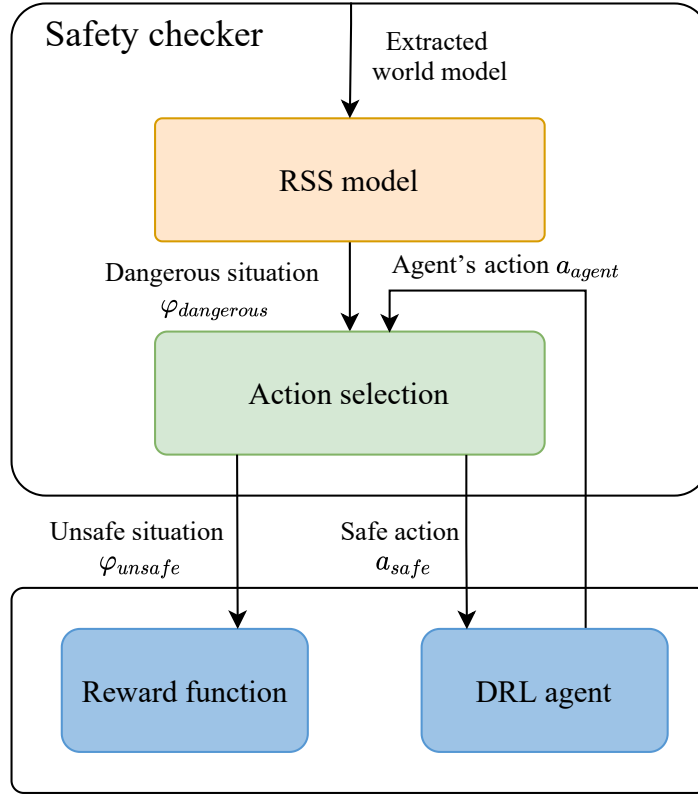


Figure 6.5: The functional architecture of the safety checker based on the RSS model (graphic from [Zha+22a], ©2022 IEEE).

at time t is: $x_{extend,t} = [x_{ego}, x_{veh,1}, x_{veh,2}, \dots, x_{veh,5}]$.

The state of the ego vehicle is defined as $x_{ego} = [v_{ego}, d_{goal}, d_{conflict}]$, which includes the ego vehicle's velocity v_{ego} , distance to the goal d_{goal} , and distance to the conflict area $d_{conflict}$.

The state of the other vehicles $x_{veh,i} = [v_i, d_{c,i}, \varphi_{priority,i}]$ includes the velocity of the vehicle v_i , distance to the conflict area $d_{c,i}$, and the priority information $\varphi_{priority,i}$. A negative sign to $d_{c,i}$ is added when a vehicle is driving away from the center of the intersection along its reference driving path. The priority information $\varphi_{priority,i}$ is set to *True* if a vehicle i has a higher priority than the ego vehicle.

Action Space

The task of the DRL agent is to make high-level decisions in order to drive the autonomous vehicle through the intersection. Therefore, this chapter applies high-level decisions represented by three discrete actions a_{drive} , $a_{cautious}$, and a_{brake} . The action a_{drive} indicates that the ego vehicle should drive with maximal velocity $v_{ego} = 5$ m/s. The action $a_{cautious}$ commands the ego vehicle to drive with velocity $v_{ego} = 1$ m/s. When a_{brake} is chosen, the ego vehicle should decelerate until $v_{ego} = 0$ m/s. In the DRL framework, a control module is employed to convert the high-level decisions from the DRL agent to actuator control signals. These control signals include the percentage of throttle and brake applied in the simulated ego vehicle in the co-simulation environment.

Reward Model

The total reward for the DRL agent in an episode is defined as follows:

$$R = R_{collision} + R_{unsafe} + R_{efficiency} + R_{rule}. \quad (6.2)$$

The training of each episode ends if a collision is detected, the maximum allowed episode length is reached (time-out), or the ego vehicle reaches the goal.

$R_{collision}$ is the collision penalty. If the ego vehicle collides with another vehicle, a penalty is assigned based on the ego's velocity:

$$R_{collision} = -2v_{ego} - 5. \quad (6.3)$$

$R_{efficiency}$ is used to encourage the ego vehicle to finish each episode as quickly as possible. Therefore, a negative reward $R_{efficiency} = -0.1$ is given for each time step in a training episode.

R_{rule} is the reward for compliance with the right-of-way rule based on the checking result $\varphi_{violation}$ from the traffic rule monitor. For comparison purposes, two methods for rewarding the rule awareness of the DRL agent are designed. The first method is a sparse rule violation reward $R_{rule,vi} = -5$, which is only assigned once in an episode when the ego vehicle violates the right-of-way rule given $\varphi_{violation} = True$. The second method is a dense rule-compliance reward $R_{rule,co}$, which encourages the ego vehicle to wait for other vehicles with a higher priority. For each time step when $\varphi_{violation} = False$, i.e., the ego vehicle waits behind the stop line to give way to vehicles with higher priority, it receives the positive reward $R_{rule,co} = 0.1$. In this case, the efficiency reward is reset to $R_{efficiency} = 0$, i.e., no penalty regarding efficiency is assigned when the ego vehicle complies with the traffic rule.

R_{unsafe} represents a penalty assigned at each time step $R_{unsafe} = -0.1$ if the ego vehicle is in an unsafe situation identified by the safety checker given $\varphi_{unsafe} = True$.

6.4 Experiments and Results

6.4.1 Simulation Environment

The implementation of the CARLA and SUMO co-simulation environment, as well as the training pipeline, is carried out on a computer with an Intel Xeon E5-2640 v4 processor running at 2.40 GHz and a Nvidia GeForce RTX 2080 Ti graphics processing unit. To investigate the effect of the proposed DRL approach, the DRL agent is based on the DDQN [VGS16] and prioritized experience replay [Sch+15]. The network structure and parameters are listed in Table 6.1.

6.4.2 Simulation Setup

The DRL agent is trained on an unsignalized four-way intersection. The other vehicles in the scenario comply with the right-before-left rule. At the beginning of the training phase, a random number of vehicles between one and ten are assigned to drive on one of the intersection lanes with different driving tasks: turning left, turning right, or going straight. At the beginning of each training episode, the ego vehicle is placed at its start position.

Table 6.1: Hyper-parameters of the Deep-Q Network (table from [Zha+22a], ©2022 IEEE).

Parameter	Value	Description
α	$2e-4$	Learning rate for Adam optimizer
B_u	5000	Buffer size of the replay buffer
B_a	64	Batched samples from replay buffer
$[l_1, l_2, l_3]$	$[64, 64, 32]$	Hidden layer size
ϵ_{decay}	0.998	Epsilon decay
ϵ_{final}	0.01	Epsilon final value
γ	0.99	Discount factor
τ	$1e-3$	Soft target network update rate
Δt	0.1 s	Time step

6.4.3 Evaluation Setup

The evaluation of the DRL framework involves two experiments. In the first experiment, three different agents are designed to evaluate the effect of the traffic rule monitor on the DRL agent. The second experiment investigates the advantage of further incorporating the safety checker into the DRL training pipeline. To train the DRL agents, a maximum allowed episode length of 60 s is established. The metrics listed below are used to compare the performance of the DRL agents.

- **Success rate:** The average number of successful episodes. An episode is a success if the DRL agent reaches the goal within the maximum allowed episode length without collision.
- **Collision rate:** The average number of collisions. A collision is counted if the DRL agent collides with another vehicle.
- **Infraction rate:** The average number of infractions. An infraction occurs when the DRL agent violates the right-of-way rule, as checked by the traffic rule monitor within an episode.

6.4.4 Evaluation of the Traffic Rule Monitor

Experimental Design

The goal of this experiment is to show the effect of the proposed traffic rule monitor on the DRL agent. Therefore, Three DRL agents are set up: a baseline DQN, and two DRL agents using the traffic rule monitor, namely RuleViDQN and RuleCoDQN.

DQN represents a standard DQN agent that implicitly explores and learns the right-of-way rule by interacting with other vehicles. Therefore, the DQN is trained only with the collision and efficiency rewards. Because this is the main configuration applied in previous works (e.g., [Ise+18; LM19]), the DQN is considered in this evaluation as the baseline approach.

RuleViDQN denotes a violation-aware DQN combined with the proposed traffic rule monitor. Compared to the standard DQN, the RuleViDQN has an additional reward $R_{rule,vi}$. The reward $R_{rule,vi}$ is a sparse reward that is only provided once in an episode when the DRL agent violates the right-of-way rule.

As suggested in [Kam+20], the sparse reward can be improved upon using dense reward by explicitly providing feedback for each time step. Therefore, another DRL agent is established using the traffic rule monitor: **RuleCoDQN**, a rule-compliant DQN. In contrast to RuleViDQN, the RuleCoDQN has a dense reward $R_{rule,co}$. RuleCoDQN receives this positive reward $R_{rule,co}$ at each time step when the DRL agent adheres to the right-of-way rule.

Training Performance

The moving average of success and infraction rates is calculated using a fixed subset size of 200 episodes, and each 200th average point is presented in Fig. 6.6 and Fig. 6.7. At the start of the training, it is observed that the agents do not comply with the traffic rule but still often reach the goal. As the training progresses, the agents reduce the number of collisions by learning conservative driving policies. However, this also results in an initial decrease in the success rate since the maximum allowed episode length is reached frequently.

After 20,000 episodes of training, the standard DQN reaches the goal with around 90% success and 30% infractions. The high infraction rate reflects that the DRL agent cannot implicitly learn the traffic rule solely by exploring the world with other vehicles.

RuleViDQN has a similar moving average of success and infraction rates to the standard DQN, which indicates that the sparse reward does not guide the RuleViDQN to learn to obey the traffic rule. RuleViDQN fails to learn the rule because of the low number of episodes containing rule violation rewards.

The positive effect of encouraging the rule compliance on the RuleCoDQN can be seen from the green curves of the success and infraction rates in Fig. 6.6 and Fig. 6.7. After 6,000 episodes, the RuleCoDQN starts to comprehend the traffic rule and learns to obey it to obtain higher rewards. The success rate also drops after 6,000 episodes because the RuleCoDQN starts to explore the benefit of obeying the traffic rule guided by the positive reward $R_{rule,co}$. However, it waits too long to explore this benefit, which causes frequent time-outs and reduces the success rate. The increase in the success rate after 14,000 episodes reveals that the RuleCoDQN complies with the rule while reaching the goal.

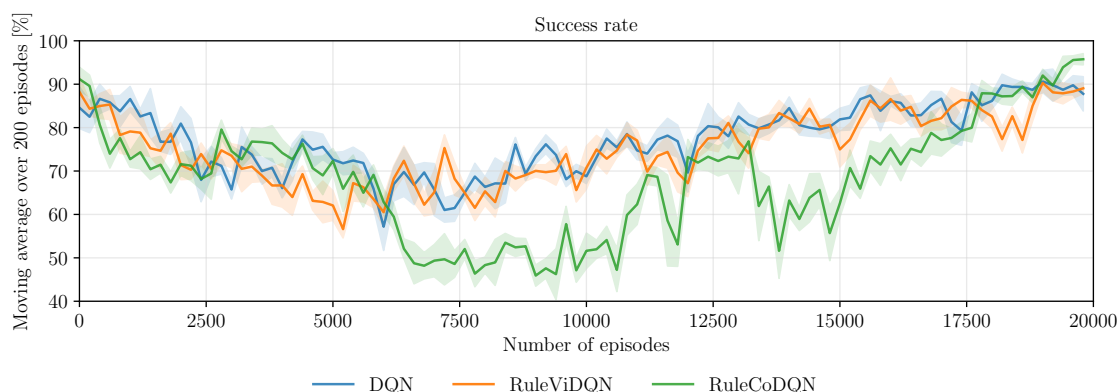


Figure 6.6: Success rate of the three agents combined with the traffic rule monitor during the training. Using a dense reward to encourage agent adherence to the right-of-way rule, RuleCoDQN takes more episodes to learn but then significantly outperforms both DQN and RuleViDQN (graphic from [Zha+22a], ©2022 IEEE).

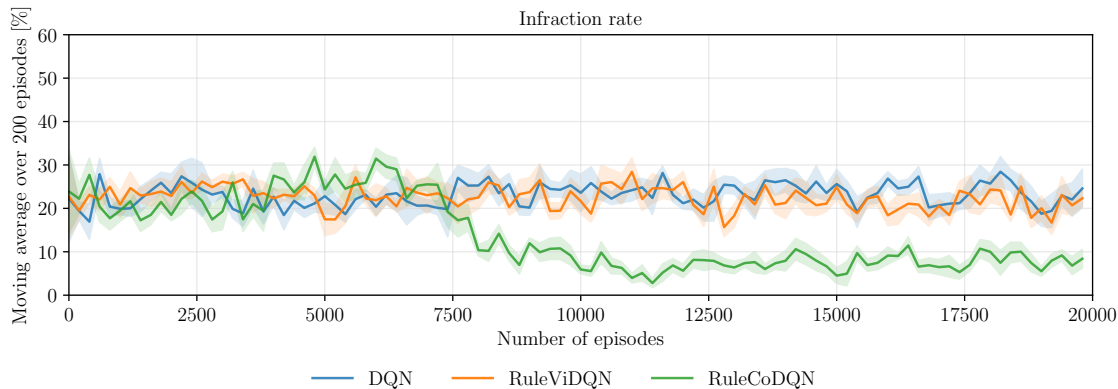


Figure 6.7: Infraction rate of the three agents combined with the traffic rule monitor during the training. (graphic from [Zha+22a], ©2022 IEEE).

Table 6.2: Evaluation results of rule-aware DRL agents (table from [Zha+22a], ©2022 IEEE).

DRL Agent	Success Rate	Collision Rate	Infraction Rate
DQN	88.6%	10.9%	33.3%
RuleViDQN	93.3%	6.7%	26.5%
RuleCoDQN	97.1%	2.9%	9.6%

Running Performance

Table 6.2 shows the success rate, collision rate, and infraction rate for the three agents during an evaluation with 2,000 episodes after training. The RuleCoDQN outperforms both the standard DQN and RuleViDQN, with the highest success rate of 97.1%, lowest collision rate of 2.9%, and lowest infraction rate of 9.6%. The evaluation results are similar to the training results, indicating that no overfitting occurred during training.

6.4.5 Evaluation of the Safety Checker

Experimental Design

The safety checker provides two results: unsafe situation φ_{unsafe} and safe action a_{safe} . The following three variants of the DRL agent are designed to compare their training performance. All three DRL variants are based on the RuleCoDQN, because it outperformed DQN and RuleViDQN in the previous experiment.

SafeReDQN uses the safety reward R_{unsafe} provided by the safety checker. It receives a penalty for each step when the ego is in an unsafe situation. **SafeAcDQN** uses the safe action provided by the safety checker instead of the original action, which may lead to an unsafe situation. **SafeReAcDQN** utilizes both the safety reward R_{unsafe} and the safe action during training. The goal is to guide the agent to identify unsafe situations and know how to react to them.

Training Performance

Fig. 6.8 and Fig. 6.9 shows the training results in the same way as previously. The success and infraction rates of SafeReDQN do not improve during training, which indicates that



Figure 6.8: Success rate of the three agents combined with the traffic rule monitor and safety checker during the training. SafeReAcDQN outperforms the other two agents with the best success rate at the end of training (graphic from [Zha+22a], ©2022 IEEE).

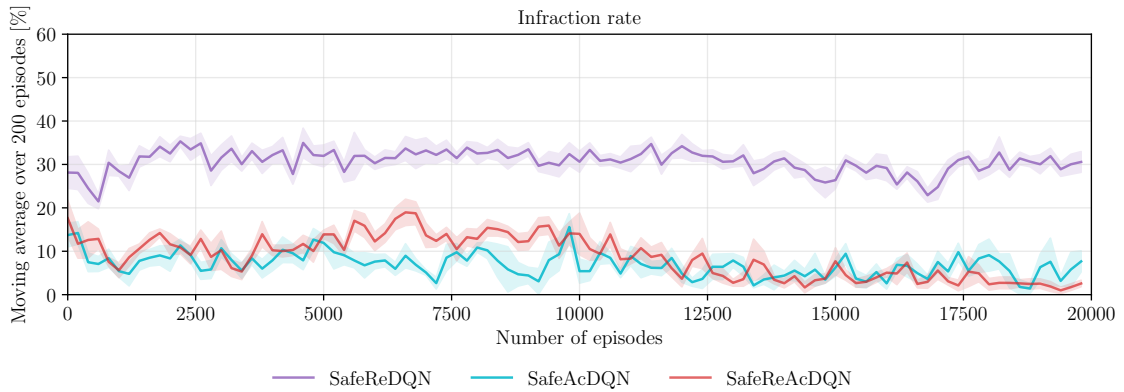


Figure 6.9: Infraction rate of the three agents combined with the traffic rule monitor and safety checker during the training. SafeReAcDQN outperforms the other two agents with the lowest infraction rate at the end of training (graphic from [Zha+22a], ©2022 IEEE).

the combination of the safety reward R_{unsafe} and the rule-compliance reward $R_{rule,co}$ cannot guide the SafeReDQN properly. A conflict between these reward combinations occurs in the situation shown in Fig. 6.4, where the ego vehicle has crossed the stop line. In this situation, SafeReDQN would drive the ego vehicle further to avoid accumulating the penalty $R_{unsafe} = -0.1$ at each time step, instead of waiting for the other vehicle, which is more likely to cause a collision. Without other methods of guidance, SafeReDQN cannot learn how to react correctly in such a situation, resulting in more collisions and rule infractions.

Compared to SafeReDQN, both SafeAcDQN and SafeReAcDQN are protected with safe action a_{safe} , which prevents it from driving into an unsafe situation. The learning curves of SafeAcDQN and SafeReAcDQN converge faster than those of the other DRL agents. Moreover, SafeReAcDQN achieves the best success rate and the lowest infraction rate with smaller standard deviations compared to the other agents at the end of training (see Fig. 6.8 and Fig. 6.9); thus, the combination of the safety reward R_{unsafe} and the safe action a_{safe} provides the best guidance for training the DRL agent.

Table 6.3: Evaluation results of safe and rule-aware DRL agents (table from [Zha+22a], ©2022 IEEE).

DRL Agent	Success Rate	Collision Rate	Infraction Rate
SafeReDQN	88.2%	11.7%	28.2%
SafeAcDQN	96.4%	0%	4.5%
SafeReAcDQN	98.6%	0%	2.6%

Running Performance

All trained agents are evaluated over 2,000 episodes, and the results are summarized in Table 6.3. Both SafeAcDQN and SafeReAcDQN benefit from the safe action and do not cause any collisions in the evaluation. Meanwhile, they drive more conservatively than the agents without safe action, with time-out rates of 3.6% and 1.4%, respectively. In summary, SafeReAcDQN performs best at reaching the mission goal successfully within the maximum allowed episode length, with the highest success rate of 98.6% and the lowest infraction rate of 2.6%. This result shows that SafeReAcDQN benefits the most from incorporating both a safety reward R_{unsafe} and a safe action a_{safe} from the safety checker. A supplementary video¹ of the evaluated scenarios is provided.

Finally, for a more intuitive comparison of the effect of the traffic rule monitor and safety checker, the training performance of the baseline DQN and the best agents from both evaluations, RuleCoDQN and SafeReAcDQN, is presented in Fig. 6.10 and Fig. 6.11.

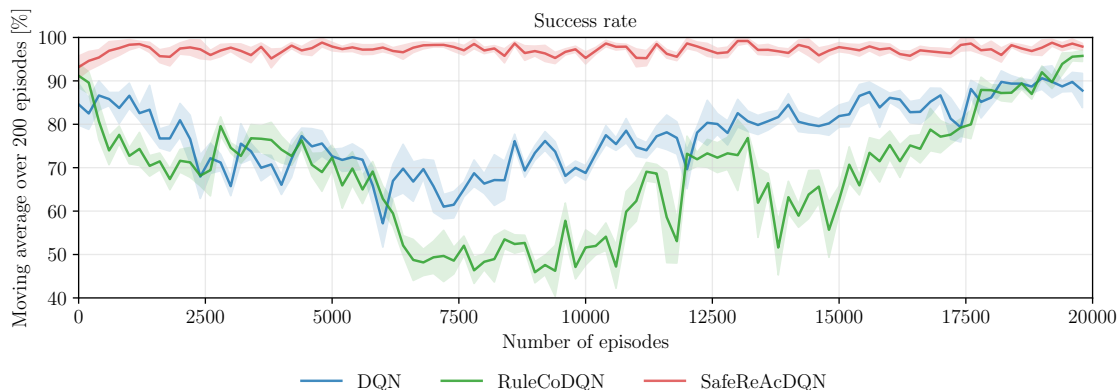


Figure 6.10: Comparison of the training performance of DQN, RuleCoDQN, and SafeReAcDQN from evaluations 6.4.4 and 6.4.5. By combining the traffic rule monitor and the safety checker, SafeReAcDQN achieves the best success rate compared to the baseline DQN and RuleCoDQN, which have only a traffic rule monitor (graphic from [Zha+22a], ©2022 IEEE).

¹Video: <https://github.com/GitChiZhang/SafeReAcDQN>

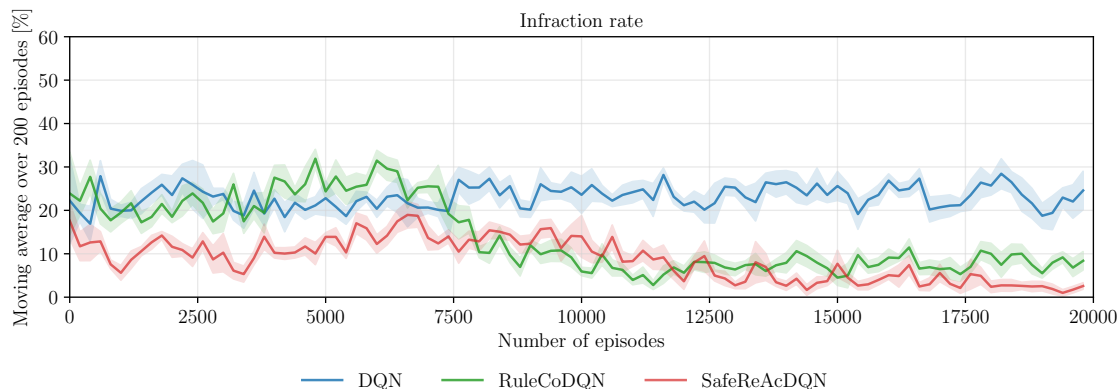


Figure 6.11: Comparison of the training performance of DQN, RuleCoDQN, and SafeReAcDQN from evaluations 6.4.4 and 6.4.5. By combining the traffic rule monitor and the safety checker, SafeReAcDQN achieves the lowest infraction rate compared to the baseline DQN and RuleCoDQN, which have only a traffic rule monitor (graphic from [Zha+22a], ©2022 IEEE).

6.5 Summary

In this chapter, a safe and rule-aware DRL planning algorithm for autonomous driving at urban intersections is presented. A co-simulation training environment combining CARLA and SUMO is applied to provide a realistic dynamic simulation of the ego vehicle and traffic participants complying with traffic rules. Different ways of incorporating rule awareness into the DRL agent are investigated using the proposed traffic rule monitor. The experiments show that the DRL agent achieves better rule-compliance results using a dense reward than a sparse reward. Finally, the rule-compliant agent is enhanced by applying a RSS-based safety checker to ensure the safety of the DRL agent's driving policy. The evaluations show that the DRL agent with a safety reward and safe action guidance achieves the best performance with no collisions.

7

Conclusion

This chapter first summarizes the contributions of the thesis in Section 7.1. Then, Section 7.2 provides future research directions to the field of behavior planning under uncertainty for autonomous vehicles.

7.1 Summary

Autonomous vehicles have been broadly investigated to improve safe and efficient transportation systems in society. Despite notable progress in the last decade, autonomous vehicles still encounter challenges in making decisions safely, comfortably and efficiently while navigating complex urban environments.

This thesis contributes to the field of planning under uncertainty for autonomous vehicles navigating complex urban environments. Explicitly encoding rules for each scenario is nearly impossible due to the range of possible scenarios for complex urban environments. Instead, planning under uncertainty formulates the driving scenarios as a POMDP problem, which allows the autonomous vehicle to make robust decisions when faced with various types of uncertain information, such as uncertainties regarding the position and velocity of the ego vehicle, the intended actions of other road users, i.e., turning right or driving straight, interactions between the ego vehicle and other vehicles, and the existence of potentially occluded road users.

Existing online probabilistic behavior planning algorithms are able to model different sources of uncertainties but have limitations when applied in complex urban environments. Probabilistic planning algorithms should be able to handle a variety of occlusion scenarios caused by static objects or dynamic road users. Furthermore, probabilistic planning algorithms are known to be computationally expensive. Therefore, such algorithms should be efficient when applied in highly interactive and dense environments. Moreover, to navigate safely through urban environments, the probabilistic planning algorithms should adhere to traffic rules while interacting with road users. This thesis presents several contributions to probabilistic behavior planning algorithms for addressing the above mentioned challenges for autonomous vehicles in complex urban environments.

Firstly, Chap. 3 presents POMDP-based planner that improves upon state-of-the-art online POMDP approaches to handle a greater range of occlusion scenarios in urban environments, such as crosswalks and bus stops. A phantom road user concept, which includes pedestrians and vehicles, is introduced to represent potentially existing occluded road users. The probability of their appearance and future movements are also modeled within the extended

POMDP formulation. The algorithm can implicitly discover an optimal policy for the autonomous vehicle by exploring various scenarios, considering the probability of occurrence, and ensuring it maintains a sufficient FoV to drive safely. The algorithm does not cause deadlocks as the worst-case assumption approach would in heavy occlusion situations. Chap. 3 further extends the algorithm to utilize information from the traffic mirror observation module, improving driving safety and efficiency by better estimating the potentially occluded road users. Furthermore, an active traffic mirror perceiving method is presented to encourage the ego vehicle to actively explore the environment and plan driving policies that support perception.

V2X communication provides additional information that can not be perceived from onboard sensors. Chap. 4 extends the algorithm to efficiently integrate information from different sources such as onboard sensors and external communication devices. Contrary to existing works, the algorithm does not require a fusion of environmental data from onboard sensors and V2X devices but rather considers them independently, thus reducing the overall computational complexity. Instead of using the potentially unreliable environmental information provided by V2X for behavior planning, the algorithm selects information based on safety-relevant criteria to improve the estimation of the existence of road users in occluded areas. By successfully integrating V2X communication within the POMDP framework, the presented approach can efficiently select messages sent from multiple infrastructures and vehicles, and generates safe and comfortable driving behaviors for challenging occlusion scenarios, especially when compared to state-of-the-art approaches that either only rely on onboard sensors or require a fusion of environmental information.

Applying the POMDP behavior planner in scenarios with a large number of road users requires significant computational effort since it needs to reason about the unknown intentions of these road users while also dealing with a variety of uncertain information, such as sensor noise and inaccurate predictions. Chap. 5 improves the efficiency of the online POMDP behavior planner to enable its application in dense urban environments. MOGM are constructed to incorporate environmental information such as the uncertain measurements, predictions, and intentions of surrounding road users. Then, MOGM is applied to create a more computationally efficient POMDP model by condensing the state space and reducing the number of calculations used for collision checks. The presented algorithm significantly improves the planner's scalability and efficiency, making it suitable for dense areas.

The behavior planning algorithm should adhere to traffic rules while providing safe driving behaviors in urban areas where other road users follow the same rules. Chap. 6 addresses the challenge of enforcing safety and traffic rule constraints for the behavior planning algorithms. Instead of solving the model using tree search, this chapter applies deep reinforcement learning, which enables the autonomous vehicle to learn to generate driving behaviors from the experience of interacting with the environment without explicitly modeling the environmental dynamics. A rule monitor is introduced to provide the priority relationship between the ego vehicle and other vehicles as well as the result of monitoring traffic rule violation. A RSS-based safety checker is presented to identify unsafe situations and provide a proper reaction that ensures safety while guiding the algorithm to learn how to behave properly in such situations. Benefiting from incorporating the traffic rule monitor and safety checker, the presented algorithm learns to navigate the intersection while obeying the right-before-left rule safely.

7.2 Future Work

This section outlines how the presented algorithms for the behavior planning of autonomous vehicles under uncertainties can be extended in future work.

Improving Solving Speed: Online planning under uncertainty requires generating driving policy in real-time. This thesis presents an efficient algorithm that incorporates MOGM as a black box simulator for constructing the belief tree more efficiently, thus providing optimized driving policies for dense environments. Another way to improve the running efficiency of the planner is to apply a better online POMDP solver. As described in Section 2.2.2, a belief tree needs to be constructed using Monte Carlo sampling. The sampling process can be accelerated using both CPU and GPU parallel calculations, as proposed by [Cai+21]. Furthermore, domain-specific heuristics can be applied to speed up the convergence to the optimal policy. For example, [CH23] suggests combining the learned heuristic functions within the belief tree search process to obtain better policies within the given solving time. As a next step, the efficient POMDP formulation from this thesis can be extended with those approaches to further improve the running efficiency.

Considering More Traffic Rules: This thesis presents a behavior planning algorithm incorporating safety and traffic rule constraints into the MDP model and then solves the model using deep reinforcement learning. The MDP model could be extended to a partially observable domain to capture the stochastic nature of the environment. Furthermore, this thesis shows how to incorporate the right-before-left rule in the decision-making framework for navigating urban intersections. As a next step, more traffic rules could be formulated in the model, such as zipper merges, and lane changes etc., so that the presented algorithm can be applied to more scenarios.

Incorporating Inverse Reinforcement Learning: One motivation for applying POMDP for behavior planning is eliminating the need to design driving policies manually. However, the reward functions for the behavior planners presented in this thesis are still constructed manually. This thesis performs a number of simulations to find an optimal combination of the weights for objectives like safety, efficiency, and comfort. One direction for improving the design of the reward function is to apply inverse reinforcement learning to automatically learn the reward function from human demonstration [You+19; HWL22]. The behavior planner could perform human-like driving behaviors with the combination of inverse deep reinforcement learning, which improves the social acceptance of autonomous vehicles.

Symbols

This list presents the basic meanings of the symbols, which will be further specified within the work through additional indexing as needed.

POMDP and Belief Tree:

a	Action
A	Action space
b	Belief state
B	Belief space
o	Observation
O	Observation space
γ	Discount factor
$T(\cdot)$	Transition model
$R(\cdot)$	Reward model
x	State
\mathcal{X}	State space
$V(\cdot)$	Value function
$Q(\cdot)$	Action-value function
π	Policy
$G(\cdot)$	Generative model
\mathcal{T}	Belief tree
$U(\cdot)$	Upper confidence bound
$N(\cdot)$	Total number of visits to a node
ϕ	Path in the belief tree
h	A episode
\mathcal{H}	Set of multiple episodes
$P(\cdot)$	Probability function
$L(\cdot)$	Prediction of the Q-value
\mathcal{D}	Replay buffer
θ	Parameters of the neural network
α	Learning rate
$P(\cdot \cdot)$	Conditional probability function
$\mathbb{E}[\cdot]$	Expectation function

Planning:

x, y	Positions of a road user in Cartesian coordinates
v	Velocity of a road user
θ	Orientation of a road user
s	Longitudinal distance in Frenet coordinates
l	Lateral distance in Frenet coordinates
r	Lane of a road user
D	Euclidean distance
N_{ego}	The ego vehicle
N_i	Other road user i
N_k	Phantom road user k
M_l	Traffic mirror list
\mathcal{L}_{obs}	List of observable areas
\mathcal{L}_{occ}	List of occlusion areas
\mathcal{C}	List of Vehicle-to-Everything devices
\mathcal{M}	Multi-step occupancy grid maps
F	Planning frequency
D	Maximal tree depth
H	Planning horizon
M	Number of planning time steps

Acronyms

ABT Adaptive Belief Tree 19, 21–25, 93

AI Artificial Intelligence 11

CARLA CAR Learning to Act 94, 96, 98, 101, 107

CNN Convolutional Neural Network 31

CPM Collective Perception Message 54

DDQN Double Deep Q-Networks 26, 101

DQN Deep Q-network 26, 30

DRL Deep Reinforcement Learning 17, 25, 93–97, 99–107, 120, 121

ETSI European Telecommunications Standards Institute 54

FoV Field of View 3, 14, 27–29, 37, 46, 48, 51–53, 57, 58, 63, 67, 68, 72, 81, 87, 110, 117–119

HD map High-Definition map 27, 29, 32, 36–38, 64

HSVI Heuristic Search Value Iteration 19

MCTS Monte Carlo Tree Search 19–21

MDP Markov Decision Process 7, 11, 14–19

MOGM Multi-step Occupancy Grid Maps 8, 9, 78–87, 90, 91, 119

OGM Occupancy Grid Map 79, 80, 83

PBVI Point-based Value Iteration 19

PER Prioritized Experience Replay 26

POMCP Partially Observable Monte Carlo Planning 21

POMDP Partially Observable Markov Decision Process 6, 7, 11, 15–19, 21, 24, 27, 29–33, 39, 40, 42–44, 48, 49, 51, 53–56, 64, 77, 79–81, 83, 86, 90, 91, 109–111, 117, 118

QMDP Fully Observable Value Approximation 18, 19

RSS Responsibility-Sensitive Safety 94, 96, 99, 107

SARSOP Successive Approximations of the Reachable Space under Optimal Policies 19

SUMO Simulation of Urban Mobility 94–96, 98, 101, 107

TD Temporal Difference 26

UCT Upper Confidence Bounds for Trees 20, 24

V2X Vehicle-to-Everything 8, 51–56, 58–65, 67, 68, 70–73, 110, 118, 119

List of Figures

1.1	Diagram illustrating the modules of autonomous vehicles, from perception and prediction to planning and control, with respective time intervals for each step (Graphic adapted from [Bou20]).	2
1.2	Illustration of challenging scenarios in urban environments.	4
1.3	This diagram outlines the structure of the thesis, beginning with an introduction and followed by a background chapter. It details the probabilistic behavior planning algorithms for safe and rule-compliant driving in dense and occluded urban environments, divided into three parts: handling spatial occlusions, navigating dense traffic, and obeying traffic rules. The thesis ends with a chapter on conclusions and suggestions for future work.	8
2.1	The process of interaction between the agent and its environment (graphic adapted from [KWW22]).	12
2.2	Illustration of the various sources of uncertainty for autonomous vehicles driving in urban environments.	13
2.3	Illustration of the MDP depicting a state x_t at a given time step, the action a_t chosen by an agent, the subsequent state x_{t+1} following the action, and the reward r_{t+1} received as a result of the action.	15
2.4	Illustration of the POMDP depicting the relationship between the hidden state x_t , the observation o_t available to the agent, the action a_t taken by the agent, and the belief b_t about the state which is updated based on the observation. The process then moves to the next time step, where the cycle repeats.	16
2.5	Four steps of each iteration in the Monte Carlo tree search algorithm (graphic adapted from [Bro+12]).	20
2.6	Illustration of a belief tree and the associated episodes. Each node in the tree represents a belief state, with edges representing actions leading to subsequent beliefs based on observations. Episodes trace the sequence of belief states and actions taken over time (graphic adapted from [KY16]).	22
3.1	Two driving scenarios in an urban environment in the presence of occlusions caused by buildings, parked cars or moving trucks. Traffic mirrors are found at the corner of an intersection (a) and on the other side of a road (b), and their information is used to increase the FoV. Red areas indicate the occluded area of the ego vehicle, whereas green areas show observed regions of the traffic mirror (graphic from [Zha+22c], ©2022 IEEE).	28
3.2	The process of the POMDP behavior planner using onboard sensors (graphic from [Zha+22c], ©2022 IEEE).	33
3.3	The extended observation model for belief update of phantom road users (graphic from [Zha+21], ©2021 IEEE).	34

3.4	(a): Ego vehicle cannot observe the mirror, since the mirror is occluded by the moving truck. (b): Ego vehicle is able to observe the mirror (graphic from [Zha+22c], ©2022 IEEE).	38
3.5	Ego vehicle driving through an occluded crosswalk. Two pedestrians are about to cross the crosswalk, but their view is blocked by the parked car (graphic from [Zha+21], ©2021 IEEE).	40
3.6	Planned velocity and acceleration profiles for crosswalk scenario with two occluded pedestrians (graphic from [Zha+21], ©2021 IEEE).	41
3.7	Ego vehicle is leaving the bus stop while a pedestrian is about to cross the road in front of a bus (graphic from [Zha+21], ©2021 IEEE).	41
3.8	Planned velocity and acceleration profiles for bus stop scenario with an occluded pedestrian (graphic from [Zha+21], ©2021 IEEE).	42
3.9	Left turn in an unsignalized intersection scenario with dynamic occlusion due to a moving truck (graphic from [Zha+21], ©2021 IEEE).	43
3.10	Planned velocity and acceleration profiles for unsignalized intersection scenario (graphic from [Zha+21], ©2021 IEEE).	43
3.11	Scenario A: The ego vehicle intends to turn left in an empty unsignalized intersection with occlusion caused by a building. Scenario B: Additionally, a dynamic vehicle is approaching the intersection (graphic from [Zha+22c], ©2022 IEEE).	44
3.12	Comparison of planned driving strategies for handling the occluded intersection (scenario A), (graphic from [Zha+22c], ©2022 IEEE).	45
3.13	Comparison of planned driving strategies for handling the occluded intersection with a dynamic vehicle (scenario B), (graphic from [Zha+22c], ©2022 IEEE).	45
3.14	Scenario C: Ego vehicle driving through a crosswalk with occlusion caused by a parked vehicle. Scenario D: Two pedestrians intend to cross the road, but their views are blocked (graphic from [Zha+22c], ©2022 IEEE).	46
3.15	Comparison of planned driving strategies for handling the occluded crosswalk without pedestrians (scenario C), (graphic from [Zha+22c], ©2022 IEEE).	47
3.16	Comparison of planned driving strategies for handling the occluded crosswalk with crossing pedestrians (scenario D), (graphic from [Zha+22c], ©2022 IEEE).	47
3.17	Scenario E: Ego vehicle is approaching a slow truck. The traffic mirror is blocked by the truck, (graphic from [Zha+22c], ©2022 IEEE).	48
3.18	Comparison of planned driving strategies for handling the occluded intersection with occluded traffic mirror (scenario E), (graphic from [Zha+22c], ©2022 IEEE).	48
4.1	The ego vehicle (blue car) executes a left-turn maneuver at an unsignalized intersection, while another vehicle (orange car) on a low-priority lane also aims to traverse the intersection. Due to occlusion caused by a large truck (gray truck) stuck in a traffic jam, both the ego vehicle and the orange vehicle are unable to directly observe each other, creating a potentially hazardous situation. In this scenario, a V2X vehicle (green car) can detect the orange vehicle using its onboard sensors. It shares its observable area (green area) and detected object list with the ego vehicle through V2X communication, providing information about the orange vehicle beyond the ego vehicle's FoV (blue area), (graphic from [Zha+23], ©2023 IEEE).	52
4.2	Illustration of the integration of the V2X communication module into the POMDP behavior planner (graphic from [Zha+23], ©2023 IEEE).	55

4.3	An occluded crosswalk scenario. A V2X infrastructure observes the crosswalk (green area) and sends communication messages to the ego vehicle (blue car). A pedestrian (yellow) is within the observable area of the infrastructure while the other pedestrian (green) is out of its detection range (graphic from [Zha+23], ©2023 IEEE).	61
4.4	The transition model of phantom vehicles and pedestrians. The blue car is the ego vehicle. The yellow car and white truck are other moving traffic participants. Green cars represent parked cars. For phantom pedestrians, dark blue indicates high appearance probability (graphic from [Zha+23], ©2023 IEEE).	63
4.5	Comparison of average association time with different numbers of V2X devices and phantom objects (graphic from [Zha+23], ©2023 IEEE).	66
4.6	Comparison of average total computation times for calculating confidence modifiers during one planning cycle as the number of V2X devices included in the evaluation scenario increases (graphic from [Zha+23], ©2023 IEEE).	66
4.7	The ego vehicle must perform a lane change with limited FoV (graphic from [Zha+23], ©2023 IEEE).	67
4.8	Comparison of planned driving strategies for handling the scenario where the ego vehicle must change lanes with limited observation of the oncoming lane (graphic from [Zha+23], ©2023 IEEE).	67
4.9	The ego vehicle is approaching an occluded crosswalk. Two V2X infrastructures are configured in this scenario (graphic from [Zha+23], ©2023 IEEE).	68
4.10	Comparison of planned driving strategies for handling an occluded crosswalk, with and without V2X communication (graphic from [Zha+23], ©2023 IEEE).	69
4.11	The occluded intersection involving a cyclist who behaves illegally by ignoring priority rules (graphic from [Zha+23], ©2023 IEEE).	69
4.12	Comparison of planned driving strategies for handling the occluded intersection with a cyclist (graphic from [Zha+23], ©2023 IEEE).	70
4.13	The ego vehicle approaches an intersection obstructed by a large building. A V2X vehicle transmits data with large latency (graphic from [Zha+23], ©2023 IEEE).	71
4.14	Comparison of planned driving strategies for handling the occluded intersection with communication latency (graphic from [Zha+23], ©2023 IEEE).	71
4.15	The ego vehicle performs an unprotected left turn at an occluded intersection (graphic from [Zha+23], ©2023 IEEE).	72
5.1	Example of an MOGM with uncertain measurements, predictions, and intentions. Each green layer is one grid map of the MOGM at each time step. The color of the grid cell indicates the risk of this grid cell being occupied by a road user at this time step. In this example, another vehicle (green car) intends to drive into the junction. This vehicle has two possible intentions: turning left or turning right. Both intentions and their corresponding predicted states are represented in the MOGM by setting the grid cells as occupied (red grids) or possibly occupied (orange and yellow grids), (graphic from [Zha+22b], ©2022 IEEE).	78
5.2	Construction of the MOGM. The ego vehicle and the predicted states of other vehicles are represented by bounding boxes. Δl_{long} and Δl_{lat} indicate the extended risk area (graphic from [Zha+22b], ©2022 IEEE).	82
5.3	The ego vehicle intends to drive through the dense urban environment with buildings along the street and many pedestrians moving around. The total number of static and dynamic objects in the scenario is 80 (graphic from [Zha+22b], ©2022 IEEE).	88

5.4	Comparison of planned driving strategies for handling the dense urban environment (graphic from [Zha+22b], ©2022 IEEE).	88
5.5	Comparison of the computation time of the MOGM-POMDP model and the TAPIR-POMDP model for generating a new episode. The speedup is calculated by comparing the average computation time of the MOGM-POMDP to that of the TAPIR-POMDP (graphic from [Zha+22b], ©2022 IEEE).	89
5.6	Comparison between the active nodes of the MOGM-POMDP and TAPIR-POMDP in the belief tree construction. The speedup is calculated by comparing the average number of active nodes of the MOGM-POMDP to that of the TAPIR-POMDP (graphic from [Zha+22b], ©2022 IEEE).	90
6.1	The figure shows an intersection without traffic signs and traffic signals. The ego vehicle (blue car) intends to drive through the intersection. The yellow vehicles have priority over the ego vehicle, whereas the ego vehicle has the right of way over the green vehicle (graphic from [Zha+22a], ©2022 IEEE).	94
6.2	Safe and rule-aware framework for DRL decision-making module of the autonomous vehicle (graphic from [Zha+22a], ©2022 IEEE).	97
6.3	A scenario in the CARLA simulator (left) and SUMO simulator (right). The states of the ego vehicle (blue car) and other road users are synchronized between CARLA and SUMO (graphic from [Zha+22a], ©2022 IEEE).	98
6.4	Example of the ego vehicle violating the traffic rule in a right-before-left intersection (graphic from [Zha+22a], ©2022 IEEE).	99
6.5	The functional architecture of the safety checker based on the RSS model (graphic from [Zha+22a], ©2022 IEEE).	100
6.6	Success rate of the three agents combined with the traffic rule monitor during the training. Using a dense reward to encourage agent adherence to the right-of-way rule, RuleCoDQN takes more episodes to learn but then significantly outperforms both DQN and RuleViDQN (graphic from [Zha+22a], ©2022 IEEE).	103
6.7	Infraction rate of the three agents combined with the traffic rule monitor during the training. (graphic from [Zha+22a], ©2022 IEEE).	104
6.8	Success rate of the three agents combined with the traffic rule monitor and safety checker during the training. SafeReAcDQN outperforms the other two agents with the best success rate at the end of training (graphic from [Zha+22a], ©2022 IEEE).	105
6.9	Infraction rate of the three agents combined with the traffic rule monitor and safety checker during the training. SafeReAcDQN outperforms the other two agents with the lowest infraction rate at the end of training (graphic from [Zha+22a], ©2022 IEEE).	105
6.10	Comparison of the training performance of DQN, RuleCoDQN, and SafeReAcDQN from evaluations 6.4.4 and 6.4.5. By combining the traffic rule monitor and the safety checker, SafeReAcDQN achieves the best success rate compared to the baseline DQN and RuleCoDQN, which have only a traffic rule monitor (graphic from [Zha+22a], ©2022 IEEE).	106
6.11	Comparison of the training performance of DQN, RuleCoDQN, and SafeReAcDQN from evaluations 6.4.4 and 6.4.5. By combining the traffic rule monitor and the safety checker, SafeReAcDQN achieves the lowest infraction rate compared to the baseline DQN and RuleCoDQN, which have only a traffic rule monitor (graphic from [Zha+22a], ©2022 IEEE).	107

List of Tables

3.1	Comparison of existing occlusion-aware planning methods with the proposed approach (table from [Zha+21], ©2021 IEEE).	32
3.2	Parameters applied in evaluation (table from [Zha+21], ©2021 IEEE).	39
3.3	Applied parameters in the simulation (table from [Zha+22c], ©2022 IEEE).	44
4.1	Occlusion catalog for defining different types of phantom road users (table from [Zha+23], ©2023 IEEE).	57
4.2	Selected trust factors for V2X infrastructures and vehicles (table from [Zha+23], ©2023 IEEE).	61
4.3	Applied parameters in the simulation (table from [Zha+23], ©2023 IEEE).	65
4.4	Quantitative comparison of the performance of planners at an occluded intersection without potentially conflicting vehicles (table from [Zha+23], ©2023 IEEE).	74
4.5	Quantitative comparison of the performance of planners at an occluded intersection with a randomly appearing vehicle (table from [Zha+23], ©2023 IEEE).	75
5.1	Applied parameters in the simulation (table from [Zha+22b], ©2022 IEEE).	86
5.2	MOGM generation times (table from [Zha+22b], ©2022 IEEE).	87
6.1	Hyper-parameters of the Deep-Q Network (table from [Zha+22a], ©2022 IEEE).	102
6.2	Evaluation results of rule-aware DRL agents (table from [Zha+22a], ©2022 IEEE).	104
6.3	Evaluation results of safe and rule-aware DRL agents (table from [Zha+22a], ©2022 IEEE).	106

List of Algorithms

1	Belief Update using Unweighted Particle Filter	24
2	Traffic Mirror Observability Check	39
3	V2X Device Association	59
4	Confidence Modifier Calculation	59
5	OGM Generation for Time Step m	84
6	Calculate Collision Probability at Time Step m	85

Bibliography

- [Amb+19] Ambrosin, M., Alvarez, I. J., Buerkle, C., Yang, L. L., Oboril, F., Sastry, M. R., and Sivanesan, K. “Object-level perception sharing among connected vehicles”. In: *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems*. 2019, pp. 1566–1573.
- [Ara22] Aradi, S. “Survey of Deep Reinforcement Learning for Motion Planning of Autonomous Vehicles”. In: *Journal of IEEE Transactions on Intelligent Transportation Systems* 23.2 (2022), pp. 740–759.
- [AHM02] Arbuckle, D., Howard, A., and Mataric, M. “Temporal occupancy grids: A method for classifying the spatio-temporal properties of the environment”. In: *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. Vol. 1. 2002, pp. 409–414.
- [Arn+20] Arnold, E., Dianati, M., Temple, R. de, and Fallah, S. “Cooperative perception for 3D object detection in driving scenarios using infrastructure sensors”. In: *IEEE Transactions on Intelligent Transportation Systems* 23.3 (2020), pp. 1852–1864.
- [ACF02] Auer, P., Cesa-Bianchi, N., and Fischer, P. “Finite-time analysis of the multi-armed bandit problem”. In: *Machine learning* 47 (2002), pp. 235–256.
- [BHL14] Bai, H., Hsu, D., and Lee, W. S. “Integrated perception and planning in the continuous space: A POMDP approach”. In: *The International Journal of Robotics Research* 33.9 (2014), pp. 1288–1302.
- [Bel66] Bellman, R. “Dynamic programming”. In: *Science* 153.3731 (1966), pp. 34–37.
- [Bey+21] Bey, H., Sackmann, M., Lange, A., and Thielecke, J. “POMDP Planning at Roundabouts”. In: *Proc. of the IEEE Intelligent Vehicles Symposium Workshops*. 2021, pp. 264–271.
- [Bou20] Bouton, M. *Safe and scalable planning under uncertainty for autonomous driving*. Stanford University, 2020.
- [BCK17] Bouton, M., Cosgun, A., and Kochenderfer, M. J. “Belief state planning for autonomously navigating urban intersections”. In: *Proc. of the IEEE Intelligent Vehicles Symposium*. 2017, pp. 825–830.
- [Bou+18a] Bouton, M., Nakhaei, A., Fujimura, K., and Kochenderfer, M. J. “Scalable decision making with sensor occlusions for autonomous driving”. In: *Proc. of the IEEE Int. Conf. on Robotics and Automation*. 2018, pp. 2076–2081.
- [Bou+18b] Bouton, M., Nakhaei, A., Fujimura, K., and Kochenderfer, M. J. “Scalable decision making with sensor occlusions for autonomous driving”. In: *Proc. of the IEEE Int. Conf. on Robotics and Automation*. 2018, pp. 2076–2081.

- [Bou+19] Bouton, M., Nakhaei, A., Fujimura, K., and Kochenderfer, M. J. “Safe reinforcement learning with scene decomposition for navigating complex urban environments”. In: *Proc. of the IEEE Intelligent Vehicles Symposium*. 2019, pp. 1469–1476.
- [Bou+20] Bouton, M., Nakhaei, A., Isele, D., Fujimura, K., and Kochenderfer, M. J. “Reinforcement learning with iterative reasoning for merging in dense traffic”. In: *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems (ITSC)*. 2020, pp. 1–6.
- [Bro+12] Browne, C. B., Powley, E., Whitehouse, D., Lucas, S. M., Cowling, P. I., Rohlfshagen, P., Tavener, S., Perez, D., Samothrakis, S., and Colton, S. “A survey of monte carlo tree search methods”. In: *IEEE Transactions on Computational Intelligence and AI in games* 4.1 (2012), pp. 1–43.
- [CH23] Cai, P. and Hsu, D. “Closing the Planning–Learning Loop With Application to Autonomous Driving”. In: *IEEE Transactions on Robotics* 39.2 (2023), pp. 998–1011.
- [Cai+21] Cai, P., Luo, Y., Hsu, D., and Lee, W. S. “HyP-DESPOT: A hybrid parallel algorithm for online planning under uncertainty”. In: *The International Journal of Robotics Research* 40 (2021), pp. 558–573.
- [Cai+19] Cai, P., Luo, Y., Saxena, A., Hsu, D., and Lee, W. S. “Lets-drive: Driving in a crowd by learning from tree search”. arXiv preprint arXiv:1905.12197. 2019.
- [CBM20] Capasso, A. P., Bacchiani, G., and Molinari, D. “Intelligent roundabout insertion using deep reinforcement learning”. arXiv preprint arXiv:2001.00786. 2020.
- [Cap+21] Capasso, A. P., Maramotti, P., Dell’Eva, A., and Broggi, A. “End-to-End Intersection Handling using Multi-Agent Deep Reinforcement Learning”. arXiv preprint arXiv:2104.13617. 2021.
- [Che+20] Chen, D., Jiang, L., Wang, Y., and Li, Z. “Autonomous driving using safe reinforcement learning by incorporating a regret-based human lane-changing decision model”. In: *Proc. of American Control Conference (ACC)*. 2020, pp. 4355–4361.
- [Che+19a] Chen, L., Chen, Y., Yao, X., Shan, Y., and Chen, L. “An adaptive path tracking controller based on reinforcement learning with urban driving application”. In: *Proc. of the IEEE Intelligent Vehicles Symposium (IV)*. 2019, pp. 2411–2416.
- [Che+19b] Chen, Q., Tang, S., Yang, Q., and Fu, S. “Cooper: Cooperative perception for connected autonomous vehicles based on 3D point clouds”. In: *Proc. of the IEEE Int. Conf. on Distributed Computing Systems*. 2019, pp. 514–524.
- [Cui+19] Cui, H., Radosavljevic, V., Chou, F.-C., Lin, T.-H., Nguyen, T., Huang, T.-K., Schneider, J., and Djuric, N. “Multimodal trajectory predictions for autonomous driving using deep convolutional networks”. In: *Proc. of the Int. Conf. on Robotics and Automation*. 2019, pp. 2090–2096.
- [Dai+20] Dai, B., Xu, F., Cao, Y., and Xu, Y. “Hybrid sensing data fusion of cooperative perception for autonomous driving with augmented vehicular reality”. In: *IEEE Systems Journal* 15.1 (2020), pp. 1413–1422.
- [Dax+22] Dax, V. M., Kochenderfer, M. J., Senanayake, R., and Ibrahim, U. “Infrastructure-enabled autonomy: An attention mechanism for occlusion handling”. In: *Proc. of the IEEE Int. Conf. on Robotics and Automation*. 2022, pp. 5939–5945.
- [DUG21] Debada, E., Ung, A., and Gillet, D. “Occlusion-Aware Motion Planning at Roundabouts”. In: *IEEE Transactions on Intelligent Vehicles* 6.2 (2021), pp. 276–287.

- [Dha+19] Dhalwar, S., Ruby, S., Salgar, S., and Padiri, B. “Image Processing based Traffic Convex Mirror Detection”. In: *2019 Fifth International Conference on Image Information Processing (ICIIP)*. IEEE. 2019, pp. 41–45.
- [Dos+17] Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., and Koltun, V. “CARLA: An open urban driving simulator”. In: *Proc. of conference on robot learning*. 2017, pp. 1–16.
- [Elf13] Elfes, A. “Occupancy grids: A stochastic spatial representation for active robot perception”. arXiv preprint arXiv:1304.1098. 2013.
- [FEN+20] FENG, Y., ONO, S., ITAGAKI, N., and SUDA, Y. “Detection of approaching objects reflected in a road safety mirror using on-vehicle camera”. In: *SEISAN KENKYU 72* (2020), pp. 201–206.
- [Gas+19] Gassmann, B., Oboril, F., Buerkle, C., Liu, S., Yan, S., Elli, M. S., Alvarez, I., Aerrabotu, N., Jaber, S., Beek, P. van, et al. “Towards standardization of AV safety: C++ library for responsibility sensitive safety”. In: *Proc. of the IEEE Intelligent Vehicles Symposium (IV)*. 2019, pp. 2265–2271.
- [Gün+16] Günther, H.-J., Mennenga, B., Trauer, O., Riebl, R., and Wolf, L. “Realizing collective perception in a vehicle”. In: *Proc. of the IEEE Vehicular Networking Conf.* 2016, pp. 1–8.
- [Guo+22] Guo, J., Carrillo, D., Chen, Q., Yang, Q., Fu, S., Lu, H., and Guo, R. “Slim-FCP: Lightweight-Feature-Based Cooperative Perception for Connected Automated Vehicles”. In: *IEEE Internet of Things Journal* 9.17 (2022), pp. 15630–15638.
- [Guo+21] Guo, J., Carrillo, D., Tang, S., Chen, Q., Yang, Q., Fu, S., Wang, X., Wang, N., and Palacharla, P. “CoFF: Cooperative Spatial Feature Fusion for 3-D Object Detection on Autonomous Vehicles”. In: *IEEE Internet of Things Journal* 8.14 (2021), pp. 11078–11087.
- [Haa+18] Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. “Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor”. In: *International conference on machine learning*. PMLR. 2018, pp. 1861–1870.
- [Han+23] Han, Y., Zhang, H., Li, H., Jin, Y., Lang, C., and Li, Y. “Collaborative Perception in Autonomous Driving: Methods, Datasets and Challenges”. arXiv preprint arXiv:2301.06262. 2023.
- [Han97] Hansen, E. “An improved policy iteration algorithm for partially observable MDPs”. In: *Advances in neural information processing systems* 10 (1997).
- [How60] Howard, R. A. “Dynamic programming and markov processes.” In: (1960).
- [HWL22] Huang, Z., Wu, J., and Lv, C. “Efficient deep reinforcement learning with imitative expert priors for autonomous driving”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2022).
- [Hub+19] Hubmann, C., Quetschlich, N., Schulz, J., Bernhard, J., Althoff, D., and Stiller, C. “A POMDP Maneuver Planner For Occlusions in Urban Scenarios”. In: *Proc. of the IEEE Intelligent Vehicles Symposium*. 2019, pp. 2172–2179.
- [Hub+17] Hubmann, C., Becker, M., Althoff, D., Lenz, D., and Stiller, C. “Decision making for autonomous driving considering interaction and uncertain prediction of surrounding vehicles”. In: *Proc. of the IEEE Intelligent Vehicles Symposium*. 2017, pp. 1671–1678.

- [Hub+18a] Hubmann, C., Schulz, J., Becker, M., Althoff, D., and Stiller, C. “Automated driving in uncertain environments: Planning with interaction and uncertain maneuver prediction”. In: *Journal of IEEE Transactions on Intelligent Vehicles* 3 (2018), pp. 5–17.
- [Hub+18b] Hubmann, C., Schulz, J., Xu, G., Althoff, D., and Stiller, C. “A belief state planner for interactive merge maneuvers in congested traffic”. In: *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems*. 2018, pp. 1617–1624.
- [Hue+19] Huegle, M., Kalweit, G., Mirchevska, B., Werling, M., and Boedecker, J. “Dynamic input for deep reinforcement learning in autonomous driving”. In: *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. 2019, pp. 7566–7573.
- [Ise+18] Isele, D., Rahimi, R., Cosgun, A., Subramanian, K., and Fujimura, K. “Navigating occluded intersections with autonomous vehicles using deep reinforcement learning”. In: *Proc. of the Int. Conf. on Robotics and Automation (ICRA)*. 2018, pp. 2034–2039.
- [KLC98] Kaelbling, L. P., Littman, M. L., and Cassandra, A. R. “Planning and acting in partially observable stochastic domains”. In: *Artificial intelligence* 101.1-2 (1998), pp. 99–134.
- [Kam+20] Kamran, D., Lopez, C. F., Lauer, M., and Stiller, C. “Risk-aware high-level decisions for automated driving at occluded intersections with reinforcement learning”. In: *Proc. of the IEEE Intelligent Vehicles Symposium*. 2020, pp. 1205–1212.
- [KZ86] Kant, K. and Zucker, S. W. “Toward efficient trajectory planning: The path-velocity decomposition”. In: *Sage Publications International Journal of Robotics Research* 5.3 (1986), pp. 72–89.
- [KSK14] Klimenko, D., Song, J., and Kurniawati, H. “TAPIR: A software toolkit for approximating and adapting POMDP solutions online”. In: *Proc. of the Australasian Conf. on Robotics and Automation*. Vol. 24. 2014.
- [KYS13] Kobayashi, A., Yamawaki, A., and Serikawa, S. “Recognition of Road Mirror with Vehicle Camera”. In: *Proceedings of the 1st IEEE/IIAE International Conference on Intelligent Systems and Image Processing*. 2013, pp. 175–178.
- [Koç+21a] Koç, M., Yurtsever, E., Redmill, K., and Özgüner, Ü. “Pedestrian Emergence Estimation and Occlusion-Aware Risk Assessment for Urban Autonomous Driving”. In: *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. IEEE. 2021, pp. 292–297.
- [Koç+21b] Koç, M., Yurtsever, E., Redmill, K., and Özgüner, Ü. “Pedestrian Emergence Estimation and Occlusion-Aware Risk Assessment for Urban Autonomous Driving”. In: *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems*. 2021, pp. 292–297.
- [KWW22] Kochenderfer, M. J., Wheeler, T. A., and Wray, K. H. *Algorithms for decision making*. MIT press, 2022.
- [KA20] Koschi, M. and Althoff, M. “Set-based prediction of traffic participants considering occlusions and traffic rules”. In: *IEEE Transactions on Intelligent Vehicles* 6 (2020), pp. 249–265.
- [KWA20] Krasowski, H., Wang, X., and Althoff, M. “Safe reinforcement learning for autonomous lane changing using set-based prediction”. In: *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems (ITSC)*. 2020, pp. 1–7.

- [KHL08] Kurniawati, H., Hsu, D., and Lee, W. S. “Sarsop: Efficient point-based pomdp planning by approximating optimally reachable belief spaces.” In: *Robotics: Science and systems*. Vol. 2008. Citeseer. 2008.
- [KY16] Kurniawati, H. and Yadav, V. “An Online POMDP Solver for Uncertainty Planning in Dynamic Environment”. In: *Robotics Research: The 16th International Symposium ISRR*. Springer International Publishing, 2016, pp. 611–629.
- [LM19] Leurent, E. and Mercat, J. “Social attention for autonomous decision-making in dense traffic”. arXiv preprint arXiv:1911.12250. 2019.
- [Li+21] Li, B., Acarman, T., Zhang, Y., and Kong, Q. “Occlusion-aware on-road autonomous driving: A path planning method in combination with honking decision making”. In: *Proc. of the Chinese Control and Decision Conf.* 2021, pp. 7403–7408.
- [LC18] Li, C. and Czarnecki, K. “Urban driving with multi-objective deep reinforcement learning”. arXiv preprint arXiv:1811.08586. 2018.
- [LKC18] Li, Z., Kalabić, U., and Chu, T. “Safe reinforcement learning: Learning with supervision using a constraint-admissible set”. In: *Proc. of American Control Conference (ACC)*. 2018, pp. 6390–6395.
- [Lia+18] Liang, X., Wang, T., Yang, L., and Xing, E. “Cirl: Controllable imitative reinforcement learning for vision-based self-driving”. In: *Proc. of the European Conference on Computer Vision (ECCV)*. 2018, pp. 584–599.
- [Lin+19a] Lin, X., Zhang, J., Shang, J., Wang, Y., Yu, H., and Zhang, X. “Decision Making through Occluded Intersections for Autonomous Driving”. In: *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems*. 2019, pp. 2449–2455.
- [Lin+19b] Lin, X., Zhang, J., Shang, J., Wang, Y., Yu, H., and Zhang, X. “Decision making through occluded intersections for autonomous driving”. In: *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems*. 2019, pp. 2449–2455.
- [LCK95] Littman, M. L., Cassandra, A. R., and Kaelbling, L. P. “Learning policies for partially observable environments: Scaling up”. In: *Elsevier Machine Learning Proceedings*. 1995, pp. 362–370.
- [Lit96] Littman, M. L. *Algorithms for sequential decision-making*. Brown University, 1996.
- [Liu+19] Liu, D., Brännstrom, M., Backhouse, A., and Svensson, L. “Learning faster to perform autonomous lane changes by constructing maneuvers from shielded semantic actions”. In: *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems (ITSC)*. 2019, pp. 1838–1844.
- [Liu+15] Liu, W., Kim, S.-W., Pendleton, S., and Ang, M. H. “Situation-aware decision making for autonomous driving on urban road using online POMDP”. In: *Proc. of the IEEE Intelligent Vehicles Symposium*. 2015, pp. 1126–1133.
- [Lop+18] Lopez, P. A., Behrisch, M., Bieker-Walz, L., Erdmann, J., Flötteröd, Y.-P., Hilbrich, R., Lücken, L., Rummel, J., Wagner, P., and Wießner, E. “Microscopic traffic simulation using sumo”. In: *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems (ITSC)*. 2018, pp. 2575–2582.
- [LYU18] Luo, W., Yang, B., and Urtasun, R. “Fast and furious: Real time end-to-end 3D detection, tracking and motion forecasting with a single convolutional net”. In: *Proc. of the IEEE conference on Computer Vision and Pattern Recognition*. 2018, pp. 3569–3577.

- [MHC99] Madani, O., Hanks, S., and Condon, A. “On the undecidability of probabilistic planning and infinite-horizon partially observable Markov decision problems”. In: *AAAI/IAAI*. 1999, pp. 541–548.
- [Mir+18] Mirchevska, B., Pek, C., Werling, M., Althoff, M., and Boedecker, J. “High-level decision making for safe and reasonable autonomous lane changing using reinforcement learning”. In: *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems (ITSC)*. 2018, pp. 2156–2162.
- [Mni+15] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. “Human-level control through deep reinforcement learning”. In: *nature* 518.7540 (2015), pp. 529–533.
- [MR19] Mohajerin, N. and Rohani, M. “Multi-step prediction of occupancy grid maps with recurrent neural networks”. In: *Proc. of the the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 10600–10608.
- [Mül+22] Müller, J., Strohbeck, J., Herrmann, M., and Buchholz, M. “Motion planning for connected automated vehicles at occluded intersections with infrastructure sensors”. In: *IEEE Transactions on Intelligent Transportation Systems* 23.10 (2022), pp. 17479–17490.
- [Nar+21] Narksri, P., Takeuchi, E., Ninomiya, Y., and Takeda, K. “Deadlock-free planner for occluded intersections using estimated visibility of hidden vehicles”. In: *MDPI Electronics* 10.4 (2021), p. 411.
- [Nau+19a] Naumann, M., Königshof, H., Lauer, M., and Stiller, C. “Safe but not overcautious motion planning under occlusions and limited sensor range”. In: *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2019, pp. 140–145.
- [Nau+19b] Naumann, M., Königshof, H., Lauer, M., and Stiller, C. “Safe but not overcautious motion planning under occlusions and limited sensor range”. In: *Proc. of the IEEE Intelligent Vehicles Symposium*. 2019, pp. 140–145.
- [NS20] Neel, G. and Saripalli, S. “Improving bounds on occluded vehicle states for use in safe motion planning”. In: *Proc. of the IEEE Int. Symposium on Safety, Security, and Rescue Robotics*. 2020, pp. 268–275.
- [Org14] Organization, W. H. *Road traffic injuries*. <https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries>. 2014.
- [OML18] Orzechowski, P. F., Meyer, A., and Lauer, M. “Tackling occlusions & limited sensor range with set-based safety verification”. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2018, pp. 1729–1736.
- [PGT+03] Pineau, J., Gordon, G., Thrun, S., et al. “Point-based value iteration: An anytime algorithm for POMDPs”. In: *Ijcai*. Vol. 3. 2003, pp. 1025–1032.
- [PGT06] Pineau, J., Gordon, G., and Thrun, S. “Anytime point-based approximations for large POMDPs”. In: *Journal of Artificial Intelligence Research* 27 (2006), pp. 335–380.
- [PVN20a] Poncelet, R., Verroust-Blondet, A., and Nashashibi, F. “Safe Geometric Speed Planning Approach for Autonomous Driving through Occluded Intersections”. In: *2020 16th International Conference on Control, Automation, Robotics and Vision (ICARCV)*. IEEE. 2020, pp. 393–399.

- [PVN20b] Poncelet, R., Verroust-Blondet, A., and Nashashibi, F. “Safe geometric speed planning approach for autonomous driving through occluded intersections”. In: *Proc. of the Int. Conf. on Control, Automation, Robotics and Vision*. 2020, pp. 393–399.
- [PK19] Pusse, F. and Klusch, M. “Hybrid online pomdp planning and deep reinforcement learning for safer self-driving cars”. In: *Proc. of the IEEE Intelligent Vehicles Symposium (IV)*. 2019, pp. 1013–1020.
- [Qi+21] Qi, Y., Zhou, Y., Liu, Y.-F., Liu, L., and Pan, Z. “Traffic-aware task offloading based on convergence of communication and sensing in vehicular edge computing”. In: *IEEE Internet of Things Journal* 8.24 (2021), pp. 17762–17777.
- [QSD21] Qiao, Z., Schneider, J., and Dolan, J. M. “Behavior planning at urban intersections through hierarchical reinforcement learning”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 2667–2673.
- [Ros+08] Ross, S., Pineau, J., Paquet, S., and Chaib-Draa, B. “Online planning algorithms for POMDPs”. In: *Journal of Artificial Intelligence Research* 32 (2008), pp. 663–704.
- [Sán+22] Sánchez, J. M. G., Nyberg, T., Pek, C., Tumova, J., and Törngren, M. “Foresee the unseen: Sequential reasoning about hidden obstacles for safe driving”. In: *Proc. of the IEEE Intelligent Vehicles Symposium*. 2022, pp. 255–264.
- [Sch+15] Schaul, T., Quan, J., Antonoglou, I., and Silver, D. “Prioritized experience replay”. arXiv preprint arXiv:1511.05952. 2015.
- [Sch+19a] Schörner, P., Töttel, L., Doll, J., and Zöllner, J. M. “Predictive trajectory planning in situations with hidden road users using partially observable Markov decision processes”. In: *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems*. 2019, pp. 2299–2306.
- [Sch+19b] Schratte, M., Bouton, M., Kochenderfer, M. J., and Watzenig, D. “Pedestrian collision avoidance system for scenarios with occlusions”. In: *Proc. of the IEEE Intelligent Vehicles Symposium*. 2019, pp. 1054–1060.
- [Sch18] Schreier, M. “Environment representations for automated on-road vehicles”. In: *at-Automatisierungstechnik* 66.2 (2018), pp. 107–118.
- [Sef+17] Sefati, M., Chandiramani, J., Kreisköther, K., Kampker, A., and Baldi, S. “Towards tactical behaviour planning under uncertainties for automated vehicles in urban scenarios”. In: *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems*. 2017, pp. 1–7.
- [SSS17] Shalev-Shwartz, S., Shammah, S., and Shashua, A. “On a formal model of safe and scalable self-driving cars”. arXiv preprint arXiv:1708.06374. 2017.
- [SV10] Silver, D. and Veness, J. “Monte-Carlo planning in large POMDPs”. In: *Proc. of the Advances in neural information processing systems*. Vol. 23. 2010.
- [SS04] Smith, T. and Simmons, R. “Heuristic Search Value Iteration for POMDPs”. In: *Proceedings of 20th Conference on Uncertainty in Artificial Intelligence (UAI '04)*. 2004, pp. 520–527.
- [Som+13] Somani, A., Ye, N., Hsu, D., and Lee, W. S. “DESPOT: Online POMDP planning with regularization”. In: *Proc. of advances in neural information processing systems* 26 (2013), pp. 1772–1780.
- [Son71] Sondik, E. J. *The optimal control of partially observable Markov processes*. Stanford University, 1971.

- [Son+18] Song, W., Su, B., Xiong, G., and Li, S. “Intention-aware Decision Making in Urban Lane Change Scenario for Autonomous Driving”. In: *Proc. of the IEEE International Conference on Vehicular Electronics and Safety*. 2018, pp. 1–8.
- [Tho+18] Thornton, S. M., Lewis, F. E., Zhang, V., Kochenderfer, M. J., and Gerdes, J. C. “Value sensitive design for autonomous vehicle motion planning”. In: *Proc. of the IEEE Intelligent Vehicles Symposium*. 2018, pp. 1157–1162.
- [UM15] Ulbrich, S. and Maurer, M. “Towards tactical lane change behavior planning for automated vehicles”. In: *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems*. 2015, pp. 989–995.
- [VGS16] Van Hasselt, H., Guez, A., and Silver, D. “Deep reinforcement learning with double q-learning”. In: *Proc. of the AAAI conference on artificial intelligence*. Vol. 30. 1. 2016.
- [Wan+22] Wang, D., Fu, W., Song, Q., and Zhou, J. “Potential risk assessment for safe driving of autonomous vehicles under occluded vision”. In: *Nature Portfolio Scientific Reports* 12.1 (2022), p. 4981.
- [WLS20a] Wang, L., Lopez, C. F., and Stiller, C. “Generating Efficient Behaviour with Predictive Visibility Risk for Scenarios with Occlusions”. In: *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2020, pp. 1–7.
- [WLS20b] Wang, L., Lopez, C. F., and Stiller, C. “Generating efficient behaviour with predictive visibility risk for scenarios with occlusions”. In: *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems*. 2020, pp. 1–7.
- [Wan+20] Wang, T.-H., Manivasagam, S., Liang, M., Yang, B., Zeng, W., and Urtasun, R. “V2VNet: Vehicle-to-vehicle communication for joint perception and prediction”. In: *Proc. of the Springer European Conf. on Computer Vision*. 2020, pp. 605–621.
- [WGW21] Wang, Y., Guo, Y., and Wang, J. “A hierarchical planning framework of the intersection with blind zone and uncertainty”. In: *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems*. 2021, pp. 687–692.
- [Wan+16] Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., and Freitas, N. “Dueling network architectures for deep reinforcement learning”. In: *International conference on machine learning*. PMLR. 2016, pp. 1995–2003.
- [Wei+11] Wei, J., Dolan, J. M., Snider, J. M., and Litkouhi, B. “A point-based MDP for robust single-lane autonomous driving behavior under uncertainties”. In: *Proc. of the IEEE Int. Conf. on Robotics and Automation*. 2011, pp. 2586–2592.
- [Wer+10] Werling, M., Ziegler, J., Kammel, S., and Thrun, S. “Optimal trajectory generation for dynamic street scenarios in a frenet frame”. In: *Proc. of the IEEE Int. Conf. on Robotics and Automation*. 2010, pp. 987–993.
- [Xia+23] Xiao, Z., Shu, J., Jiang, H., Min, G., Chen, H., and Han, Z. “Perception Task Offloading With Collaborative Computation for Autonomous Driving”. In: *IEEE Journal on Selected Areas in Communications* 41.2 (2023), pp. 457–473.
- [Xio+16] Xiong, X., Wang, J., Zhang, F., and Li, K. “Combining deep reinforcement learning and safety based control for autonomous driving”. arXiv:1612.00147. 2016.
- [You+19] You, C., Lu, J., Filev, D., and Tsiotras, P. “Advanced planning for autonomous vehicles using reinforcement learning and deep inverse reinforcement learning”. In: *Robotics and Autonomous Systems* 114 (2019), pp. 1–18.

- [Yu+22] Yu, H., Luo, Y., Shu, M., Huo, Y., Yang, Z., Shi, Y., Guo, Z., Li, H., Hu, X., Yuan, J., et al. “DAIR-V2X: A large-scale dataset for vehicle-infrastructure cooperative 3D object detection”. In: *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 21361–21370.
- [YVJ19] Yu, M.-Y., Vasudevan, R., and Johnson-Roberson, M. “Occlusion-Aware Risk Assessment for Autonomous Driving in Urban Environments”. In: *IEEE Robotics and Automation Letters* 4.2 (2019), pp. 2235–2241.
- [Yur+20] Yurtsever, E., Lambert, J., Carballo, A., and Takeda, K. “A survey of autonomous driving: Common practices and emerging technologies”. In: *IEEE access* 8 (2020), pp. 58443–58469.
- [Zha+20] Zhang, L., Ding, W., Chen, J., and Shen, S. “Efficient uncertainty-aware decision-making for automated driving using guided branching”. In: *Proc. of the Int. Conf. on Robotics and Automation*. 2020, pp. 3291–3297.
- [ZF21] Zhang, Z. and Fisac, J. “Safe Occlusion-aware Autonomous Driving via Game-Theoretic Active Perception”. In: *Proc. of the Robotics: Science and Systems*. 2021.
- [ZSB04] Zimmerman, N., Schlenoff, C., and Balakirsky, S. “Implementing a rule-based system to represent decision criteria for on-road autonomous navigation”. In: *Proc. of the AAAI Spring Symposium on Knowledge Representation and Ontologies for Autonomous Systems*. 2004.

Publications by the Author

- [Zha+22a] Zhang, C., Kacem, K., Hinz, G., and Knoll, A. “Safe and Rule-Aware Deep Reinforcement Learning for Autonomous Driving at Intersections”. In: *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems*. 2022, pp. 2708–2715.
- [Zha+22b] Zhang, C., Ma, S., Wang, M., Hinz, G., and Knoll, A. “Efficient POMDP Behavior Planning for Autonomous Driving in Dense Urban Environments using Multi-Step Occupancy Grid Maps”. In: *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems*. 2022, pp. 2722–2729.
- [Zha+21] Zhang, C., Steinhauser, F., Hinz, G., and Knoll, A. “Improved Occlusion Scenario Coverage with a POMDP-based Behavior Planner for Autonomous Urban Driving”. In: *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems*. 2021, pp. 593–600.
- [Zha+22c] Zhang, C., Steinhauser, F., Hinz, G., and Knoll, A. “Traffic Mirror-Aware POMDP Behavior Planning for Autonomous Urban Driving”. In: *Proc. of the IEEE Intelligent Vehicles Symposium*. 2022, pp. 323–330.
- [Zha+23] Zhang, C., Steinhauser, F., Hinz, G., and Knoll, A. “Occlusion-Aware Planning for Autonomous Driving With Vehicle-to-Everything Communication”. In: *IEEE Transactions on Intelligent Vehicles* (2023). Early Access.

Supervised Theses

- [Cha23] Chai, Q. “Risk-aware POMDP Behavior Planning for Autonomous Driving”. Master’s Thesis, ZF Friedrichshafen AG and Chair of Robotics, Artificial Intelligence, and Real-time Systems, Technical University of Munich. 2023.
- [Che22] Chen, Z. “Behavior Planning for Autonomous Driving using Monte Carlo Tree Search with Dynamic Search Strategies”. Semester’s Thesis, Chair of Robotics, Artificial Intelligence, and Real-time Systems, Technical University of Munich. 2022.
- [Don22] Dong, Y. “Performance Improvement for POMDP-based Behaviour Planner for Autonomous Driving using Context Modelling and Parallel Algorithms”. Master’s Thesis, ZF Friedrichshafen AG and Institut für Maschinenkonstruktion und Systemtechnik Fachgebiet Smart Mobility Systems, Technische Universität Berlin. 2022.
- [Kac21] Kacem, K. “Behavior Planning for Autonomous Driving using Deep Reinforcement Learning”. Master’s Thesis, ZF Friedrichshafen AG and Chair of Robotics, Artificial Intelligence, and Real-time Systems, Technical University of Munich. 2021.
- [Lu23] Lu, J. “Combining Monte Carlo Tree Search and Deep Reinforcement Learning in Behavior Planning for Autonomous Driving”. Semester’s Thesis, Chair of Robotics, Artificial Intelligence, and Real-time Systems, Technical University of Munich. 2023.
- [She23] Shen, S. “Safe Decision-making in Autonomous Driving under Uncertainty”. Master’s Thesis, Chair of Robotics, Artificial Intelligence, and Real-time Systems, Technical University of Munich. 2023.
- [Sun22] Sun, R. “Deep Q-Learning using Weighted Graph SAGE Networks for Autonomous Driving”. Master’s Thesis, Chair of Robotics, Artificial Intelligence, and Real-time Systems, Technical University of Munich. 2022.
- [Wan22] Wang, C. “Confidence Improvement Using Adversarial Reinforcement Learning in Autonomous Driving”. Semester’s Thesis, Chair of Robotics, Artificial Intelligence, and Real-time Systems, Technical University of Munich. 2022.
- [Wei22] Wei, X. “Traffic Rules Compliant Behavior Planning using Monte Carlo Tree Search for Autonomous Driving”. Semester’s Thesis, Chair of Robotics, Artificial Intelligence, and Real-time Systems, Technical University of Munich. 2022.
- [Wei23] Wei, X. “Rules Compliant Behavior Planning under Uncertainty for Autonomous Driving”. Master’s Thesis, Chair of Robotics, Artificial Intelligence, and Real-time Systems, Technical University of Munich. 2023.

- [Xie23] Xie, Q. “MCTS-Based Traffic Rules Compliant Behavior Planning Considering Interaction for Autonomous Driving”. Semester’s Thesis, Chair of Robotics, Artificial Intelligence, and Real-time Systems, Technical University of Munich. 2023.