Situation Assessment in Urban Intersection Scenarios

Dissertation zur Erlangung des akademischen Grades
Doktoringenieur (Dr.-Ing.)

vorgelegt der Fakultät für Informatik und Automatisierung
der Technischen Universität Ilmenau

von Dipl.-Inf. Matthias Platho
geboren am 02.09.1983 in München

vorgelegt am: 31.03.2014

Gutachter:

1.) Univ.-Prof. Dr.-Ing. Horst-Michael Groß

2.) Univ.-Prof. Dr. rer. nat. Christian Wöhler

3.) Dr. rer. nat. Julian Eggert

urn:nbn:de:gbv:ilm1-2014000411

Acknowledgments

The submission of this thesis concludes an exciting PhD project that started four years ago. During my PhD I received support by many people and I am very grateful for their assistance. These people did not only enable a successful completion of my project but made the four years more instructive, more rewarding and more fun than I could have hoped for.

First of all, I would like to thank my two supervisors: Julian Eggert and Professor Horst-Michael Groß. Julian provided the initial vision for the project and has been a great mentor from the very start. Although (or better: because) we are almost completely different minded concerning scientific approaches, Julian's input complemented my ideas well. It happened more than once that we started a discussion with totally different methods in mind just to arrive at a solution that we both favored over our initial ones. Professor Groß complemented my research in a similar way: his background and expertise in the field of robotics provided a new and valuable view on my research problems and helped me to overcome obstacles in ways I would not have considered otherwise.

Furthermore, I would like to thank all my colleagues at the *Honda Research Institute Europe*. Michael, for being a both fun and supportive scientific advisor. Sven, Nils and Jörg, with whom I shared my office, for aiding me in countless situations. The car group, consisting of Jannik, Tobias, Sarah, Bram and most notably Thomas, for helping me greatly to finally evaluate my work on their test vehicle. Thomas Guthier from TU Darmstadt for many enjoyable discussions. Stefan Klingelschmitt for his contribution in implementing my demonstration system.

I would also like to thank my friends Christian and Christine for reviewing my thesis.

Special thanks goes to my family for their ongoing support. My parents sparked my interest in computer science with always giving me access to top notch hardware as if they had foreseen the digital revolution. Nevertheless they never pushed me in any direction but supported me on the way that I chose, even if it was to study Social Sciences...I am very grateful for everything they have done for me.

Last, but not least, I would like to thank my wife Susanne. Every single day she has been a source of motivation; her dedication and her persistent confidence in me enabled me to reach my personal and professional goals.

Dedicated to my parents

Kurzfassung

Ein Großteil der Verkehrsunfälle auf Europas Straßen findet in städtischen Kreuzungsbereichen statt. Die Ursache liegt in den meisten Fällen in einem vorangegangen Fehler seitens eines Fahrers. Eine Möglichkeit, um solche Unfälle zu verhinden, wäre der Einsatz eines Fahrassistenzsystems, welches im Fahrzeug verbaut ist und den Fahrer in Kreuzungssituationen unterstützt und auf mögliche Gefahren hinweist. Das Fahrassistenzsystem müsste dafür in der Lage sein eine komplexe Verkehrssituation ganzheitlich zu erfassen und basierend darauf abzuschätzen wie sich die Situation in naher Zukunft weiter entwickelt. Derzeit gibt es aber noch kein System, dass solch eine Funktionalität bietet, und auch im akademischen Bereich fehlt es an Methoden für eine ganzheitliche Situationserfassung.

Die vorliegende Arbeit präsentiert eine neuartige Methode um innerstädtische Kreuzungssituationen zu erfassen. Sie basiert auf der Erkenntnis, dass derartige Verkehrssituationen zu komplex und zu variabel sind, um sie im Ganzen zu analysieren. Deshalb wird die Verkehrssituation in kleinere, leichter handhabbare Teile zerlegt, wobei jeder Teil aus zwei in Beziehung stehenden Entitäten besteht. Dabei beeinflusst eine Entität das Verhalten der anderen, zum Beispiel ein Fahrzeug, welches ein folgendes Fahrzeug zum Abbremsen veranlasst. Mögliche Konstellationen für in Beziehung stehende Entitäten werden von einem Experten in Modellen spezifiziert, die als *Konfigurationen* bezeichnet werden.

Im Zuge der Arbeit wurde eine Reihe von Methoden entwickelt, die von dem Konfigurationen-Konzept Gebrauch machen. Neben der eigentlichen Erkennung einer Konfiguration wird das Konzept auch für Methoden zur Verhaltensprädiktion verwendet, unter anderem, um ausgehend von der Konfiguration eines Fahrzeugs das longitudinale Verhalten oder nächste Manöver vorherzusagen. In einer umfassenden Evaluation kann gezeigt werden, dass sich Konfigurationen zuverlässig erkennen lassen und sich zur Situationseinschätzung eignen. Zusätzlich übertreffen die vorgeschlagenen Prädiktionsmethoden die zum Vergleich herangezogenen bisherigen Methoden. Als Grund dafür wird die Berücksichtigung der Konfigurationsinformation ausgemacht.

Abstract

In Europe, the major share of traffic accidents takes place at urban intersections. In most of the cases, these accidents are the result of a preceding driver error. One possibility for avoiding these accidents would be to employ a vehicle-mounted Advanced Driver Assistance System (ADAS) that supports the driver during negotiating an intersection and warns of possible hazards. For this purpose, the ADAS would be required to assess complex traffic situations comprehensively in order to anticipate the future evolution of the current situation. At the time being, there is no system available offering this functionality, and also in academia there are no suitable methods for performing a comprehensive situation assessment.

The work at hand presents a novel method for assessing urban intersection situations. It is based on the insight that these traffic situations are too complex and too variable to assess them as a whole. Therefore, a decomposition of the traffic situation into smaller, more manageable parts is proposed. Each part consists of a pair of interrelated entities, where one entity affects the behavior of the other, for example a vehicle forcing a trailing vehicle to slow down. Possible constellations of interrelated entities are defined by a human expert in models which are tagged *configurations*.

In the course of this work, a set of methods was developed that employ the concept of configurations. Besides fundamental methods aiming at the recognition of a road user's configuration also novel approaches for predicting the behavior or the upcoming maneuver of a vehicle are presented, which take the configuration of the vehicle explicitly into account. In an extensive evaluation, it is shown that configurations can be robustly recognized and are suited for a comprehensive situation assessment. In addition, the proposed prediction methods excel the state-of-the-art methods used for reference which can be traced back to the consideration of configuration-information.

Contents

1. Introduction

Road safety has been an important topic since the first days of the automobile. Public authorities have enacted laws to enforce safer driving and invested in infrastructure for a hazard-free traffic flow. Additionally, car manufacturers have used technological advancements in order to improve control over a vehicle under difficult conditions and protect passengers during crashes. In recent times the advancement has been propelled by the rise of information technology.

Modern cars have an ever-increasing amount of sensors and computing power at their disposal. Besides realizing an efficient engine management and providing real-time information about the state of a vehicle a large amount of electronics is dedicated to active and passive safety systems. Systems that warn the driver of critical situations or even take corrective action autonomously, are called Advanced Driver Assistance Systems (ADAS). The work at hand is concerned with fundamental research towards an ADAS for inner-city intersections.

The remainder of this chapter is structured as follows. At first, the context of the work is given in Section 1.1. It shows that intersections are the most crash-prone spots encountered in every-day driving and that currently sold ADAS do not address this issue sufficiently. Section 1.2 presents the problem that this thesis addresses: the development of a method that allows for an scalable situation assessment even in highly unstructured environments. Section 1.3 enumerates the contributions made towards solving the stated problem. In Section 1.4 an outline of the structure of this work and the contents of the chapters to come are provided.

1.1. Context: Driver Assistance Systems for inner-city driving

While in the last decades the number of fatalities has steadily declined, still 3600 people lost their lives on German roads in 2012 [sta, 2012]. Altogether 299,600 road injuries accidents occurred across Germany, of which 17,800 occurred on highways, 75,100 occurred on rural roads and the vast majority of them, 68% or 206,700, occurred in inner-city locations (see Figure 1.1). The European research project TRACE (Traffic Accident Causation in Europe) investigated on pre-accidental driving situations and found that in 2004 43% of road injury accidents in the EU27 had taken place on intersections [Molinero Martinez et al., 2008]. According to

TRACE, intersections accounted for 32% of accidents resulting in serious injuries and fatalities.

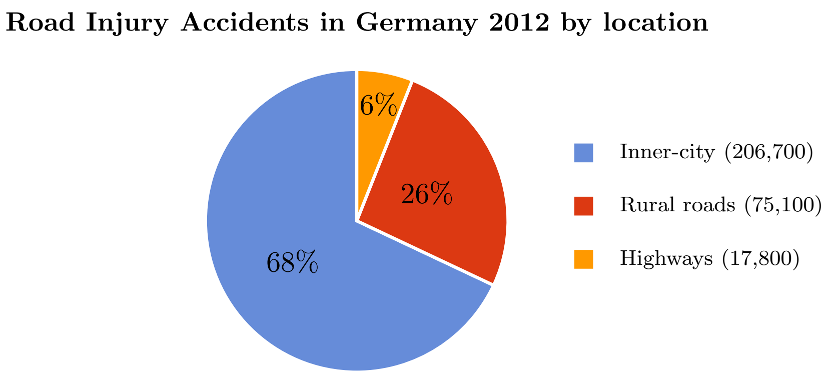


Figure 1.1.: *According to the Federal Statistical Office, 299,600 road injuries accidents occurred 2012 in Germany. The major share took place in inner-city areas. Data source: Destatis, Unfallentwicklung auf deutschen Straßen, 2012.*

The numbers demonstrate that intersections are accident-prone locations in the roadway system. A recent study issued by the U.S. National Highway Traffic Safety Administration, named *Crash Factors in Intersection-Related Crashes*, investigates the causes for such a high share in accidents [Choi, 2010]. The authors state that intersections require driving activities like crossing over or turning into other roads that have a high potential for conflicts resulting in crashes. In the National Motor Vehicle Crash Causation Survey (NMVCCS) accident data has been collected over a two year period. In this dataset, the *critical reason* for each of the reported crashes was either attributed to a driver error, a vehicle malfunction or the environmental circumstances. NMVCCS defines a *critical reason* as the immediate reason for the event that puts a vehicle on the course that makes a collision unavoidable. The results of the survey are given in Figure 1.2.

Interestingly, the critical reason for more than 96% of intersection-related crashes can be attributed to the driver. Preventing human error is thus a major issue for increasing road safety.

1.1.1. History of Driver Assistance Systems

Car manufacturers identified long ago the potential for Advanced Driver Assistance Systems (ADAS) that support the driver in his task. The first marketed ADAS was an Adaptive Cruise Control (ACC) system offered by Mitsubishi in 1995. An ACC keeps a motor vehicle at the speed set by the driver like a regular Cruise Control system does, but it additionally comprises a forward looking sensor monitoring the

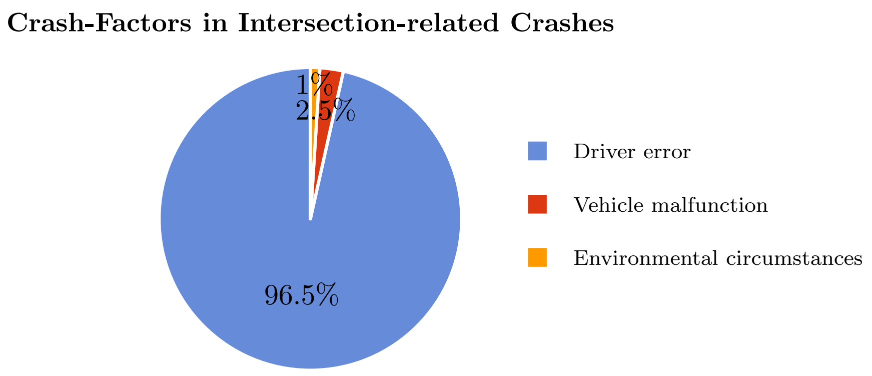


Figure 1.2.: *Results of the National Motor Vehicle Crash Causation Survey. Data source: NMVCCS, 2007.*

area ahead. If the sensor detects a vehicle in front, the ACC adapts the speed in order to keep the vehicle at a safe distance.

A more recent type of collision avoidance systems, also relying on forward looking sensors like LIDAR and RADAR, are *Precrash systems*. The first system of its kind was offered by Honda in 2003 [Shaout et al., 2011]. Contrary to an ACC, a Precrash system is permanently active during driving and constantly observes the area ahead of the vehicle. As soon as the system detects a possible collision with an obstacle in front it issues a warning to the driver and in most cases also prepares seatbelts and brakes for an upcoming emergency break. If the driver still does not react, the system triggers an emergency braking that mitigates or even prevents a crash.

Blind Spot Information Systems (BLIS) are even more tailored to accident prevention, as they notify the driver of vehicles situated in the uncovered (=blind) area of side- and rear-view mirrors. A vehicle equipped with BLIS has on each side an additional sensor like a laser scanner, radar sensor or a camera. These sensors inform the driver visually of other vehicles being currently in its blind spot by an illuminated LED close to the corresponding side-view mirror. A BLIS was first presented by Volvo in their 2004 model update of the S80 sedan [Shaout et al., 2011]. Newer systems also take corrective action into steering to prevent anticipated accidents.

With sensors and onboard computers becoming increasingly powerful the use cases and capabilities of driver assistance systems have accordingly increased. Currently commercially offered ADAS feature Lane Keeping Assistants, automatic parking and traffic sign recognition [Bosch GmbH, 2013] to name a few.

1.1.2. Assistance Systems for intersections

Although intersection-related crashes account for an overproportional share in accidents there are currently no commercially available ADAS that assist the driver in negotiating an intersection. Certainly, a Precrash system can help to prevent rearending on intersection entrances and a BLIS assists when merging into a desired lane for turning. But both of these systems support the driver only during approaching instead of during the whole crossing maneuver. One reason for the lack of suitable ADAS is the high specialization of nowadays systems: they are all tailored to a very specific use case in a constrained situation. Precrash systems for example base their decision to intervene on the detection of an obstacle in the vehicles path without requiring to identify it as a certain entity [Shaout et al., 2011]. The information that it lies on a collision course with an extrapolated trajectory of the own vehicle is sufficient.

At intersections it is often not possible to anticipate conflicts based on extrapolated trajectories. One example where this holds true is given in Figure 1.3.

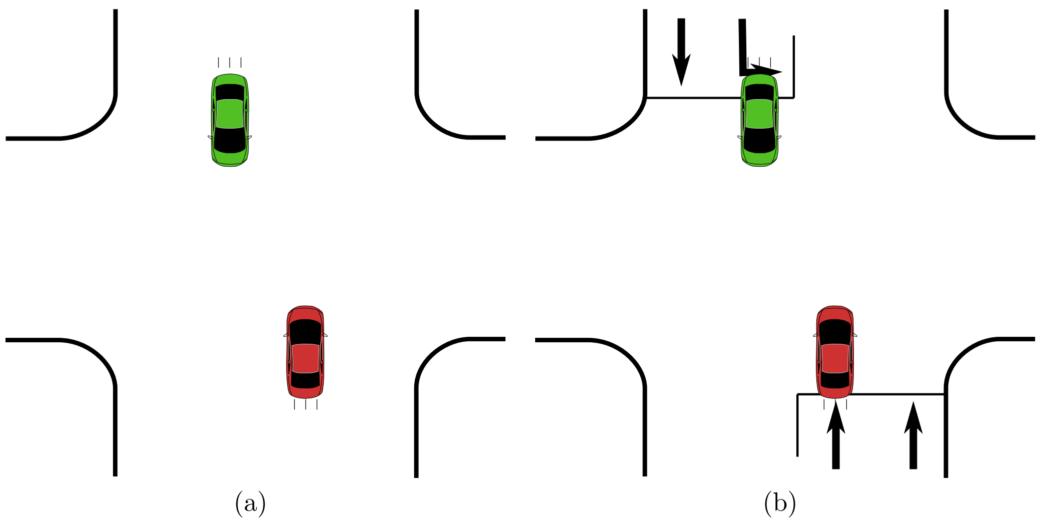


Figure 1.3.: *Physically extrapolated trajectories do not provide any hint of an upcoming conflict (a), while considering the structure of the intersection does (b).*

In Figure 1.3(a) two vehicles entering an intersection from opposite sides are depicted. Judging from their current movement they are expected to pass each other. This assessment changes completely when taking the additional lane information as provided in Figure 1.3(b) into account: The red vehicle can now be expected to follow a route that crosses the route of the green vehicle and thus a potential conflict can be assumed.

The example above demonstrates that in inner-city scenarios a comprehensive situation assessment can not be obtained from vehicle kinematics alone. Instead, a multitude of entities, concepts and rules need to be considered, each of which might be crucial to obtain a correct interpretation of a situation. For the situation depicted in Figure 1.3 the consideration or neglect of a single aspect – the lane markings – changes the interpretation completely.

When talking about situations, it is important to clarify what is actually meant by this term. In this work two types of situations are discerned: *traffic situations* and *driving situations*. The definitions for these types are loosely along the lines of [Reichart, 2001]. Both of these types are defined around an acting road user, in this case the driver. A traffic situation is the objectively given spatial and temporal constellation of traffic related entities, including applicable traffic rules in which a road user is acting. In contrast, a driving situation is the section of a traffic situation comprising only these aspects that are currently relevant for a driver's behavior. An aspect is relevant if it is perceivable by the driver and can possibly play a role in its decision making. To give an example: At an intersection the traffic situation is constituted by all vehicles in the surrounding including the driver itself, traffic lights, routing of streets, right-of-way rules and all other nearby road users like pedestrians and bicyclists. The driving situation is an excerpt of the traffic situation, with only those road users that are noticeable by the driver and that can eventually cross or join his path.

Whenever the term situation is used without further specification, a traffic situation is meant. Situation assessment as described here is always concerned with the assessment of a *traffic* situation as a whole.

Two major obstacles hinder the realization of a driver assistance system capable of performing comprehensive situation assessment: lack of adequate sensor technology and lack of methods for situational inference. For inner-city driving adequate sensors are required for detecting nearby road users, read lane markings, recognize traffic signs and provide accurate localization. While for all of these tasks sensor systems have been developed their performance is still far from perfect. Additionally, many sensor systems are still far too costly to be reasonably used for an ADAS system. For example, a 360 degree laser scanner like the Velodyne HDL-64e has a list price of 75,000 $. Nevertheless, one can expect that as more and more cars are equipped with ADAS the prices of sensor systems will fall while their robustness will increase.

The second obstacle towards driver assistance systems for inner-city intersections is the lack of methods for situational inference. The following section will present the challenges in developing such methods and detail in which way the thesis at hand is expected to contribute to mastering them.

1.2. Problem formulation

There are two properties of inner-city scenarios that make situation assessment especially difficult compared to highway scenarios: High complexity and high variability. The high complexity arises from the fact that a significantly *higher number* of entities can be present at the same time. Besides vehicles also pedestrians, bicyclists, traffic lights and right-of-way rules have to be considered. Additionally, road users cross the path of others at intersections or zebra-crossings and thus require coordination. Some entities are even so important that neglecting them can have serious consequences, e.g. a red traffic light. It is essential to judge the *relevance* of one entity to another correctly.

The high complexity of inner-city scenarios is accompanied by a high variability. When driving in dense inner-city traffic one will rarely encounter the same situation twice. The number, location and behavior of nearby road users will vary from time to time and thus the number of possible situations is infinite. That is why it is crucial to identify in which way a variation changes the correct interpretation. It is not constructive to consider two situations as completely different just because the position of a single entity differs by one millimeter. Therefore a mechanism for abstraction is needed that extracts the gist of a situation and is *robust* to irrelevant changes.

The requirements identified in the preceding paragraphs serve as problem to be addressed by this thesis and can be stated as follows:
Provide an approach to situation assessment:

1. Based on the concept of relevance between entities
2. Scaling to high numbers of road users
3. Showing robustness by abstraction

Situation assessment does not provide a value per se. Stating that a certain situation belongs to type X and another situation belongs to type Y offers no additional information unless it serves as a basis for further usage. In a descriptive usage identifying elementary types helps to understand how inner-city traffic behaves. In a predictive usage the behavior of road users can be projected into the future to anticipate their maneuvers. Both usage types should be enabled by this work as well.

Though this work is focused rather on methodical aspects of situation assessment than on sensory issues it is unrealistic to assume that in the near future sensors will provide perfect measurements. Until then sensors are prone to inaccuracies, errors or even complete failure. Based on the preceding considerations the problem formulation is extended.
Anticipate the future behavior of other road users:

1. Foresee their intended maneuvers
2. Predict their future positions
3. ...while being capable of handling uncertainties of nowadays sensor systems

The aforesaid requirements determined the goals of the PhD project that culminated in the thesis at hand.

1.3. Contributions

The work presented here is a novel, comprehensive approach towards driver assistance systems for urban intersections. It was developed along the problem formulation given above and comprises the following contributions:

1. Providing a novel approach to situation assessment:
   - **Based on the concept of affecting and affected entity:** The interactions between multiple road users are modeled in a relevance view, allowing to identify for each road user which entity does currently affect his behavior.
   - **Scaling to high numbers of road users:** When modeling the interactions between multiple road users by determining pairs of affecting and affected entity the difficulty of the assessment is significantly lowered: Instead of all-to-all relations only bilateral dependencies are considered.
   - **Showing robustness towards the challenges of intersection scenarios:** Allows to incorporate expert knowledge for distinguishing relevant from irrelevant variations between situations.
2. Providing a framework for situation assessment:
   - **Recognizing interacting entities robustly** The framework provides a consistent method to learn and to recognize the relations between road users. It identifies which entity affects whom based on probabilistic reasoning. The probabilistic implementation allows to take uncertainties of real-world sensors into account.
   - **Predicting Behavior** Based on the recognized dependencies of a road user his future behavior is estimated. This comprises estimating his maneuver intention as well as his longitudinal maneuver execution.

1.4. Thesis Outline

The remainder of this thesis is structured as follows:

Chapter 2: Related work presents a review on approaches addressing similar problems as formulated in this thesis. Relevant works come from various areas which are categorized into three fields according to their focus, namely *Situation Assessment*, *Trajectory Prediction* and *Intention Estimation*. After a discussion of the advantages and downsides of each method the need for a novel approach is motivated.

Chapter 3: Configurations introduces a novel concept for decomposing complex traffic situations into manageable parts. A *Configuration* can be used to model the relationship between interrelated entities and offers insight on the determinants of individual behavior. Configurations describe how and by what a vehicle's behavior is affected and are a central building block for all methods developed in the course of this work.

Chapter 4: Situation Assessment using Configurations details all these methods. A special emphasis is put on Situation Assessment itself and some of the presented methods have been developed for exactly this purpose. Other methods in turn use Situation Assessment as an additional cue for e.g. improving the accuracy of a behavior prediction system.

Chapter 5: CarD Simulation Framework describes all aspects of the microscopic traffic simulator *CarD* that was developed in the course of this thesis. Besides its overall architecture also the design of its independently acting road users is presented and discussed. *CarD* has been used for creating large datasets on which newly developed methods have been evaluated.

Chapter 6: Evaluation and Results presents the results obtained from evaluating the methods of Chapter 4. The evaluation is performed on traffic situations both simulated by *CarD* and recorded by a test vehicle and the benefit of the newly developed methods is investigated.

Chapter 7: Future Work is concerned with all the areas that need additional research. It discusses insights requiring further validation and issues that have been raised and might be worth looking into. The chapter also identifies ways for extending the framework presented herein.

Chapter 8: Conclusion provides a summary of the work and gives an outline of all previous chapters. It also relates to the introduction by comparing the initially set goals and the final achievements.

2. Related Work

The issues identified in the problem formulation of Chapter 1 are subject to vital research. This chapter reviews the state of the art in related work and identifies shortcomings of current approaches for the problem at hand.

Section 2.1 starts by matching the goals of this thesis with research areas that pursue similar goals. Three areas are identified as highly related, namely *Situation Assessment*, *Intention Estimation* and *Trajectory Prediction*. The corresponding works are reviewed in Subsections *1-4*. The subsequent discussion in Section 2.2 reveals that current methods suffer from various shortcomings: e.g. they are tailored to highway-scenarios or do not regard intersection-specific aspects like traffic lights and right of way.

2.1. Literature Review

In the preceding chapter two major goals were defined that a sought-after method should reach. These goals are:

1. Assess complex traffic situations in inner-city driving. Determine which road users interact and in which way.
2. Anticipate the future behavior of other road users

The first goal is usually tackled by a field called *Situation Assessment*. Due to the fact that *situation* is a very general term, an assessment can be performed wherever spatial and temporal constellations of multiple acting entities are analyzed. That is why approaches for Situation Assessment range from robotics [Wendler and Lenz, 1998] across medicine [Zahlmann et al., 2000] to warfare [Das et al., 2002]. As the overlap of problems between the various applications is rather small this review will only focus on works concerned with traffic situations. These works approach the problem in one of two ways: Either by classifying a situation as one out of a set of previously specified ones, such that the known interpretation of a pre-specified situation can be reused for the currently observed one. Or by interpreting the spatial and temporal constellations of entities based on previously learned patterns.

The second goal matches the field of *Intention Estimation*, which is concerned with anticipating the next maneuver of an observed road user. The set of considered maneuvers is defined beforehand and ranges in most works between two and four; examples are *Going straight* vs. *Turning Left/Right* or *Following* vs. *Overtaking*.

Intention Estimation provides a coarse, discrete description of a road user's future behavior. Methods from *Trajectory Prediction* allow for a more fine-grained estimation that also comprises a route with anticipated positions and velocities. Works in this field could serve also as a basis for the second goal.

In the following subsections works of all mentioned fields are presented. Some approaches do not belong exclusively to one field or another, but combine multiple fields e.g. when simultaneously estimating intention and trajectory of an observed vehicle. These works are presented in Subsection 2.1.4.

2.1.1. Situation Assessment

A recent approach for classification-based situation assessment is described in [Reichel et al., 2010]. The authors propose to decompose complex situations into subsets, which are termed *situation aspects* and are based on a concept first introduced by Schaaf in [Schaaf, 1997]. A situation aspect is defined as a "...relevant hypothesis that must be answered in order to select and parameterize the correct behavior" [Reichel et al., 2010]. Reichel et al. use a *Convoy Merging Situation Aspect* (CMSA) that is designed to answer the question "Does the EGO participate at a convoy merging on the absorbing convoy lane?" Based on a set of features describing state and constellation of ego and nearby vehicles a classifier is trained that answers the question posed by the CMSA. An extension of *Random Forests* [Breiman, 2001], a *Scenario-Based Random Forest* is employed for the classification task, which differs from the original algorithm in that it is oversampling training cases based on their associated risk. Using this method the presence of a convoy merging situation is recognized with an accuracy of about 91 %. Downsides of the approach is its limitation to a single type of situation and its focus on highway scenarios.

Another work on situation classification is presented in [Vacek et al., 2007], describing a method for a more general situation assessment. The goal is to interprete arbitrary situations by comparing the observed one with already encountered ones stored in a memory. At the same time the experience gained from already encountered and successfuly mastered situations is used to deduce the appropriate action to take. To arrive at this goal *case-based reasoning* is employed, a framework from the field of Artificial Intelligence. In this paper, cases represent situations. An initial set is manually designed and stored in a memory, the so-called case-base. This case-base is structured in a hierarchical manner, as it is shown in Figure 2.1(a).

The hierarchy orders situations depending on how general they are, with the most general situations being on top. An additional temporal interconnection is used to store the evolution of situations as a result of maneuvers performed by drivers being in that situation. Figure 2.1(b) displays the evolution of an intersection situation depending on the turning behavior chosen by the present vehicles, where one behavior leads to a crash.

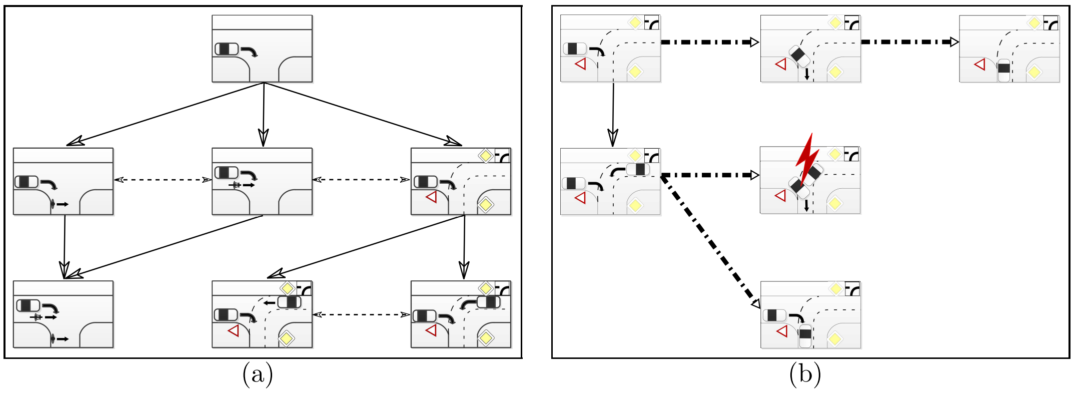


Figure 2.1.: *Case-bases for representing different situations and their evolution; as proposed in [Vacek et al., 2007]. The case-base is structured in a hierarchical manner (a). Temporal links between cases represent the consequences resulting from alternative actions (b).*

Using a case-base as in Figures 2.1, newly encountered situations are matched to it and the closest case is retrieved. For the retrieved case the possible behaviors and their experienced outcomes are checked and the behavior with the most desired outcome is chosen and executed. The new situation is added to the case-base along with the outcome of the behavior. The overall approach is appealing in theory, but it requires a significant amount of handcrafting to set up the case base and it might not be desired that a vehicular system learns from experience, as this means that critical situations need to be encountered to obtain a learning signal.

The framework proposed by [Hülsen et al., 2011] is also capable of reasoning about situations based on given knowledge. Huelsen et al. employ Description Logic [Baader, 2003], a subset of first-order predicate logic that is limited to unary and binary relations, in order to specify an ontology. This ontology consists of concepts and relations. Concepts are entities like lanes, traffic signs or cars. The taxonomy of these objects is realized in a hierarchical manner, e.g. *YieldSign* is a sub-concept of TrafficSignAtCrossing. Relations between entities describe dependencies between concepts and provide the basis for any reasoning. Relations considered are *isPart*, *approachesTo*, and *hasToYield* to new a few. The way concepts and relations are used is illustrated in Figure 2.2.

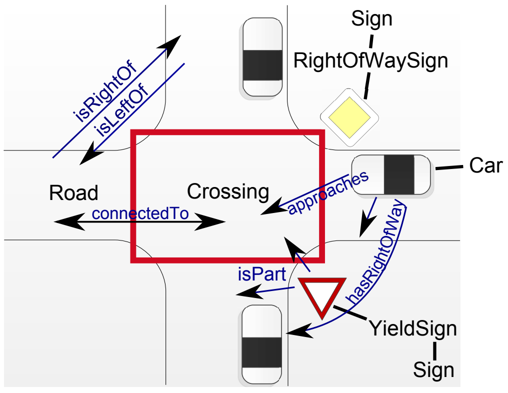
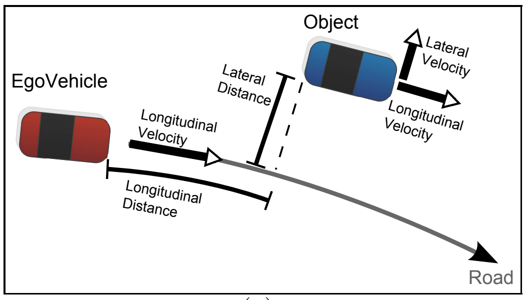


Figure 2.2.: *An ontology for describing intersection situations as proposed in [Hülsen et al., 2011]. It consists of both concepts like* Car *and* YieldSign *as well as relations like* hasToYield*.*

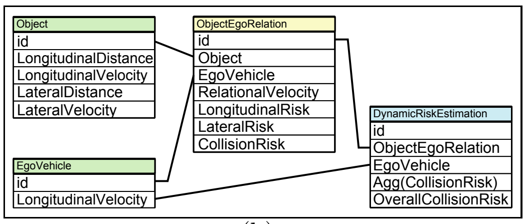
The goal of the reasoning framework is to create a system that fully comprehends a traffic situation. The authors demonstrate its capabilities by querying various aspects of a complex intersection scenario, for example *'Retrieve all instances with the relation "hasToYield" coming from car 1'*. They can show that their system arrives at the right conclusions, but a single query takes 3 seconds on a modern quad-core processor. Though description logic circumvents the problems arising from a closed world assumption as given in other logic formalisms, it is still not capable to cope with noisy sensor information.

While logic formalisms have the advantage of providing a well-defined mechanism for reasoning, they also suffer from their inability to handle inexact information. In [Schamm and Zöllner, 2011] a method is proposed that aims at combining first-order logic with probabilistic networks, as the latter is a viable method to incorporate noisy sensor data . The authors employ Object-Oriented Probabilistic Relational Language (OPRL), an entity-relationship based formal description, in order to assess situations and judge the level of risk associated with them. In Figure 2.3(a) an exemplary situation for risk assessment is given. Using OPRL, this situation can be transferred into a relational model as it is shown in Figure 2.3(b).

The relational model serves as a basis for constructing an object-oriented Bayesian Network, where each entity or relation is mapped to an individual network fragment that is connected to other fragments according to the OPRL description. The resulting Bayesian Network is a polytree for which its conditional probabilities can be determined exactly. Schamm and Zöllner claim that their computations take



(a)



(b)

Figure 2.3.: *Approach to situation assessment using Object-Oriented Probabilistic Relational Language (OPRL) as presented in [Schamm and Zöllner, 2011]. Sketch of a typical driving situation and features for describing it (a). The corresponding OPRL description models the relations between the vehicles in a class-based notation (b).*

less than 0.02 seconds on a single CPU core. A downside of their approach is the high effort for the manual model specification, as it requires to specify both the logic description as well as its realization as Bayesian Network.

The method for situation assessment presented in [Schubert et al., 2010] is also based on Bayesian Networks while setting logic formalism aside. The goal of the assessment is to select an appropriate and safe maneuver while driving on highway, i.e. changing or keeping the current lane. For this purpose the measurements of an upstream lane and occupancy detection system are converted into probabilities and discretized into a small number of states. These states are part of the Bayesian Network used for taking a maneuver decision, which is depicted in Figure 2.4.

Schubert et al. use the Bayesian Network to turn measurements of an observed situation into a utility for performing a lane change versus keeping the lane. The utility value depends directly on the situation assessment provided by the Bayesian Network, which judges the individual safety of driving on each of the nearby lanes. The assumed safety of a lane serves as basis for the expected utility of driving on that lane in that the overall system recommends the maneuver that results in driving on the safest lane. The overall approach is very suitable to handle uncertain measurements and requires only a moderate amount of handcrafting, however, it is limited to a very constrained highway scenario.

2.1.2. Intention Estimation

A situation assessment returns a *descriptive* model answering such aspects about a situation as e.g. which lane can be considered safe, which road user is interacting

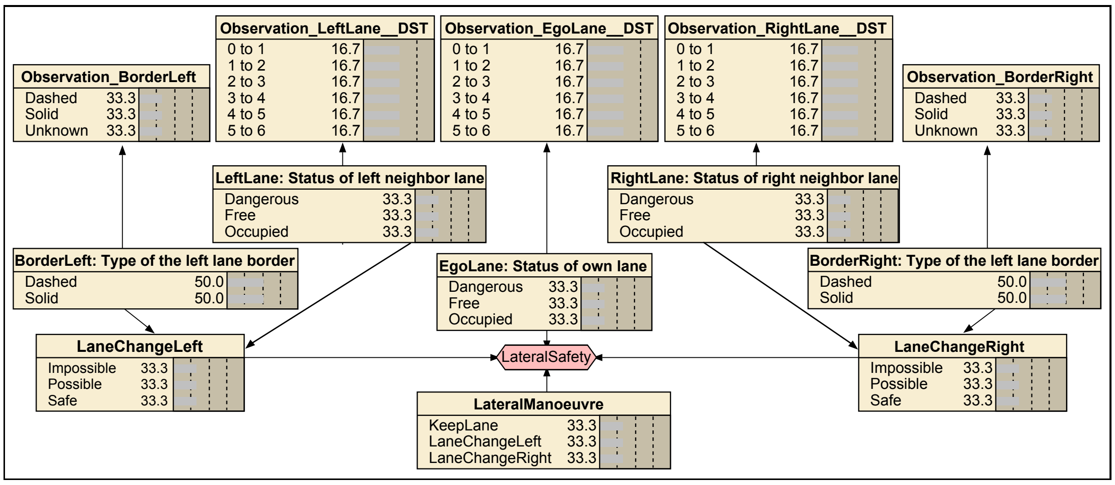


Figure 2.4.: *The Bayesian Network for deriving lane-change decisions as proposed in [Schubert et al., 2010]. Based on the conditional probabilities in* OwnLane, LaneChangeLeft *and* LaneChangeRight *the utility of a lane change is determined in the* LateralSafetyNode.

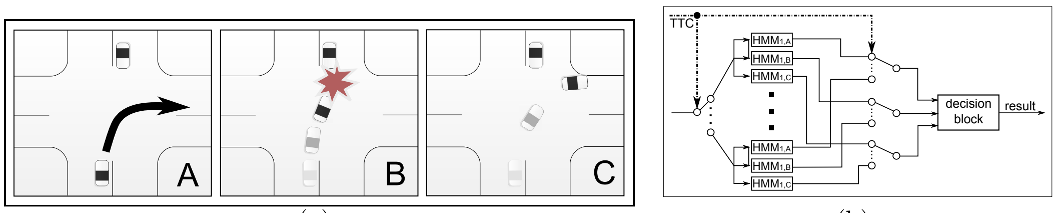
with whom and which type of driving situation a driver is currently in. This information might suffice already for various applications, though in many cases a *predictive* model for the future behavior of a road user is more helpful. Especially for collision avoidance systems an accurate prediction capability for the maneuvers of nearby road users is crucial. The field of intention estimation is concerned with the development of methods for anticipating maneuvers.

Case-based reasoning is not only used for situation assessment as described in the previous subsection, but can also be applied to intention estimation. In [Graf et al., 2013] a learning concept for maneuver prediction which relies on case-based reasoning is presented. The reasoning system is tailored to highway scenarios, where it estimates whether a leading vehicle is going to perform an overtaking maneuver or stays in its lane. Cases are created by coding situations as sequence of characters that represents the constellation of all nearby vehicles. The case *'rsf'* for example stands for a situation in which the intent for a car driving on the **r**ight lane relative to the ego vehicle is estimated, which in turn has a car passing by its **s**ide and another car driving in **f**ront of it. A case is further augmented by information describing a situation's dynamic aspects, here the relative velocity and the distance between observed and its leading vehicle, which are also discretized and character coded. The case-base thus consists of cases that can be retrieved and compared by their character code and for which the resulting maneuver is known. The complete system is evaluated on real-world data and it is shown that the system improves with experience, though the low number of samples used impedes statements about the long-term stability of the system. Another problem is that constellations are discretized into very coarse categories like *right* or *faster*. This discretization might sacrifice accuracy for tractability.

Incorporating sensor uncertainty into prediction methods is a major topic in the research community, which is why probabilistic models are becoming increasingly popular also in the field of intention estimation. In [Lidstrom and Larsson, 2008] a probabilistic approach for predicting the turning intention of vehicles approaching an intersection is presented. Lidström and Larsson design a state space model that captures the dynamics of a vehicle during an intersection approach. The corresponding velocity evolution model consists of two components. The first component implements a car-following behavior based on the Gipps model [Gipps, 1981] and the second component a decelerating behavior for turning maneuvers. A particle filter takes the observed velocity profile of an approaching vehicle as input and estimates how likely it belongs to either of these components. As soon as the likelihood of one component is significantly higher than the likelihood of the other component, the system returns whether the observed vehicle intends to go straight or turn at the intersection. The accuracy of the estimation is about 85 %, but for almost every fifth vehicle no decision could be made as the likelihoods were not discriminative enough. As 400 particles need to be evolved for each vehicle the approach is also computationally costly.

A common framework for state space models are hidden Markov Models (HMM). They are called 'hidden' because the internal state of the modelled entity can not be measured directly. In the works discussed here the hidden internal state is the intended maneuver. In [Hayashi and Yamada, 2009] HMM's are used to predict unusual and potentially dangerous right-turn behavior where the driver leaves the correct driving corridor and thus provokes conflicts with other road users. The three considered behaviors are depicted in Figure 2.5(a).

The situation consists of the observed vehicle arriving from below and an additional vehicle arriving from above. Note that left-hand traffic is assumed. Hayashi and Yamada train an individual HMM for each of the three behaviors and for each of twelve Time-To-Collision (TTC) intervals. The TTC is obtained by extrapolating the kinematic movement of the two vehicles. The resulting 36 HMM's are trained individually on data obtained by a driving simulator. In order to arrive at a single decision two mechanisms are employed. At first, only those HMM's are considered which match the currently estimated TTC. Second, out of these models the one with the highest likelihood that also surpasses a given threshold is selected. If no model surpasses the threshold, the intention is considered unpredictable (see Figure 2.5(b). An evaluation on driving simulator data shows an almost perfect estimation accuracy for TTC's below 1.5 seconds. Still, it is limited to a very specific situation.



(a) (b)

Figure 2.5.: *A HMM-based approach for predicting potentially dangerous right-turn behaviors as presented in [Hayashi and Yamada, 2009]. The three considered right-turn maneuvers (a). Maneuver A is correctly executed while the others lead either to a critical situation (B) or end in the wrong lane (C). In (b) the method for predicting unusual right-turn behavior is depicted. Based on the current TTC the results of corresponding HMM's are gated to the decision block.*

Another solution for anticipating risky intentions is described in [Aoude et al., 2012]. The work is concerned with early detection of red light runners such that vehicles intending to violate a red traffic light are detected before they enter the critical intersection area. This information can then be used to either remind the driver of the violating car of braking or to warn other drivers nearby. Besides a support vector machine the authors also use HMM's for recognizing a drivers intention to violate a red traffic light. One HMM, denoted as λ_c, is trained solely on instances where the observed vehicle stops correctly at the designated line. A second HMM, λ_v is trained solely on instances where the observed vehicle does not stop correctly but violates the traffic light by crossing the intersection. In order to decide which of the two possible maneuvers an approaching vehicle intends to perform the likelihood ratio of λ_c and λ_v is computed based on the behavior of the observed vehicle so far. If the ratio surpasses a given threshold the system outputs a warning of an upcoming violation. In a first evaluation the authors achieve a true positive rate of almost 98%, but at the expense of an false positive rate of about 17 %.

In [Meyer-Delius et al., 2009] HMM's are used to anticipate one of three different maneuvers on highways: 'Following', 'Passing' and 'Aborted Passing'. Here, HMM's constitute the top layer of a hierarchical, two-layer model and Dynamic Bayesian Networks [Murphy, 2002] (DBN) serve as bottom layer. The bottom layer implements a state space model that tracks a vehicle's behavior on a physical level while the more abstract top layer recognizes the intended maneuvers. Separating state space model and maneuver recognition into two separate layers has the purpose of lowering the complexity of the overall system. Meyer-Delius et al. evaluate their method on both simulated and real data and show that their hierarchical model is able to identify maneuvers with a good accuracy. Unfortunately, their approach requires that the observed vehicle is already executing the first part of a maneuver before it can be recognized. Using this approach, there is only little time to react to an intended maneuver after it has been recognized.

A longer prediction horizon is pursued in the work presented in [Dagli et al., 2003] which is concerned with an early detection of overtaking maneuvers on highways. Dagli et al. use a Dynamic Bayesian Network that captures both the current state and the driving situation of an observed vehicle and combines this information for anticipating its next maneuver which can be either changing or keeping the current lane. This is depicted in Figure 2.6.

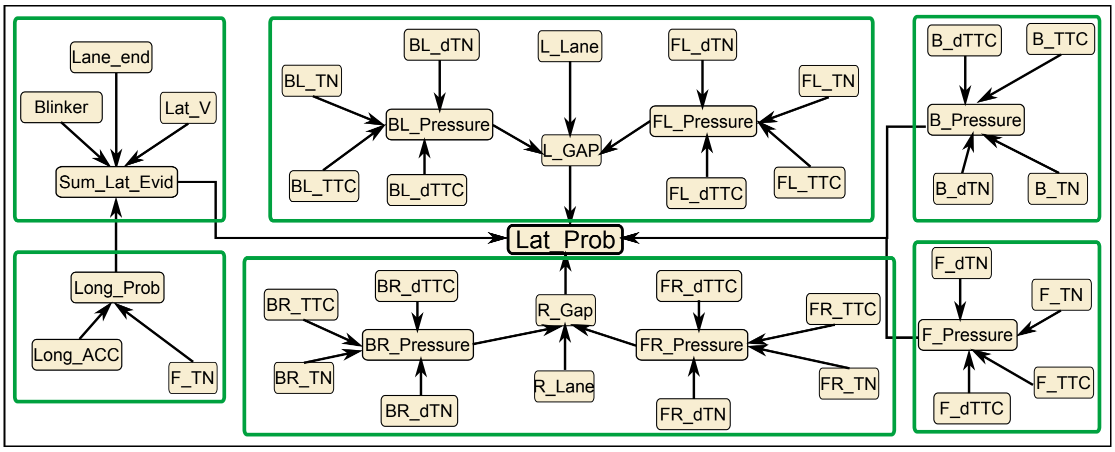


Figure 2.6.: *The static part of a DBN for predicting overtaking maneuvers as proposed in [Dagli et al., 2003]. Green borders denote individual subnets. The necessity for a lane change, called* Pressure*, is derived from the TTC and Net-Time-Gap (TN) to vehicles in* **F***ront,* **B***ack, to the* **L***eft or* **R***ight.*

The DBN is separated into five subnets (framed green), where each subnet captures a different aspect of a situation, for example the possible gain of performing a lane change or the behavior of the observed vehicle. Nodes of the DBN represent situational features like the Time-To-Collision to nearby vehicles and the observed vehicle's lateral position in the current lane. The TTC and the Net-Time-Gap is transformed into a probabilistic necessity to perform a lane change, called *pressure*. In a qualitative evaluation on simulated data the authors demonstrate that their approach can predict an intended lane change 1.5 seconds in advance. At the same time they have to admit that due to the complexity of both network and its features a bayesian learning is intractable.

A much leaner and more tractable method for intention estimation using Dynamic Bayesian Networks is presented in the works by Lefèvre [Lefèvre et al., 2011, Lefèvre et al., 2012]. Her approach aims for identifying risky situations at intersections by detecting conflicts between intention and expectation. This means that the proposed algorithm compares a driver's intended behavior with the behavior expected by him and if it finds a significant difference it defines the situation as risky. The intuition behind this is that each driver selects his own behavior based on the anticipated maneuvers of others and an erroneous anticipation may result in a crash. For example, a situation where a vehicle approaching an intersection would be expected to yield to vehicles with right-of-way but behaves as if it intends to cross, is considered risky. The Dynamic Baysian Network used by Lefèvre et al. consists of only three nodes per timestep; its structure for three consecutive timesteps is given in Figure 2.7.

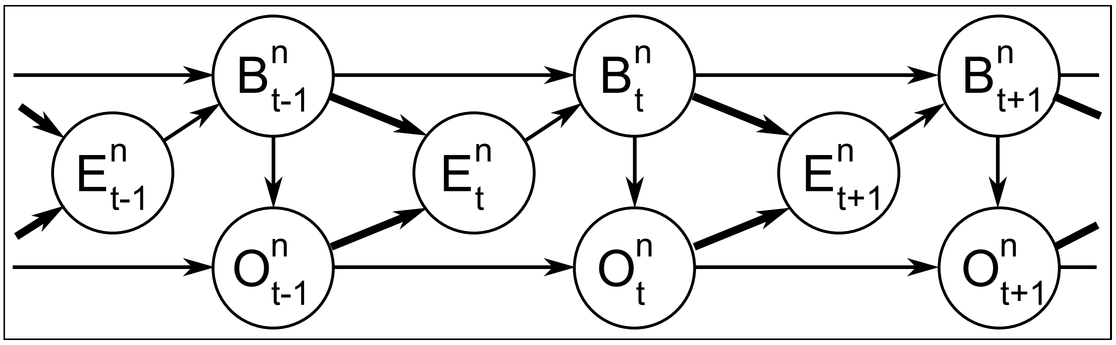


Figure 2.7.: *Risk estimation based on a detected conflict between the intended and expected behavior of a vehicle as proposed in [Lefèvre et al., 2012]. The Dynamic Bayesian Network for a single vehicle and three consecutive time steps.* E *models the expected behavior and* B *models the intended behavior. The latter is hidden and is therefore derived from the observed behavior* O. *Bold arrows represent multi-vehicle dependencies.*

The node with the letter E denotes the expected behavior, node B captures the hidden intention and node O the physical behavior for an observed vehicle. Physical behavior is obtained by measuring a vehicles position, speed and heading. The expected behavior is obtained by modeling the driver as compliant road user obeying traffic rules and avoiding unsafe maneuvers. The intention is derived from the similarity of expected and physical behavior. The risk level is obtained by computing the probability of a mismatch between intention and expectation. If this probability surpasses a certain threshold a situation is considered dangerous. In a large scale evaluation on real-world data Lefèvre et al. can show that their system detects more than 90 % of risky situations at a TTC of at least 1 second. The only downside of the approach is its limitation to only two vehicles.

One major benefit of Dynamic Bayesian Networks is their ability to model ongoing processes in a probabilistically consistent manner. This ability comes with significant computationally costs which can be either met with shrinking a network to the bare minimum like Lefèvre did or by representing the dynamic part outside the network. This results in a standard Bayesian Network plus a dynamic behavior model and was proposed in [Liebner et al., 2012]. The goal of the approach is to estimate whether the driver of the ego vehicle is intending to turn right or to drive straight at the next intersection. A Bayesian Network is used to relate a driver's hidden intention to observable behavior. The dynamic aspect of the behavior is captured using the Intelligent Driver Model (IDM) [Treiber and Helbing, 2002], which is one of the most widely used driver models for car-following behavior. The IDM reproduces the way a driver keeps a gap to a leading vehicle depending on his velocity, the leading vehicle's velocity and his driving style. The main feature for the intention estimation by Liebner et al. is the velocity profile of the ego vehicle while approaching an intersection. This is motivated by the observation that a vehicle with the intention to turn will significantly decelerate already long before reaching the intersection while a vehicle crossing straight will not. In order to exploit this property the IDM is extended such that it also considers turning behavior. The match between an observed velocity profile and the expected behaviors as computed by the IDM returns a likelihood for turning versus driving straight. This likelihood is provided to the Bayesian Network which returns its estimation of the intention. An evaluation on real data confirms that the approach achieves highly accurate estimations, however, this accuracy drops significantly when the driver's chosen velocity is dominated by a preceding vehicle.

The work presented in [Kasper et al., 2011] is another example where a static Bayesian Network is used for modeling dynamic behavior. It is concerned with an early recognition of intended maneuvers of vehicles driving on highway. The Bayesian Network used for recognition captures all dynamic aspects of the observed situation in discrete states of its nodes. As the goal is to identify 27 different maneuvers the complexity of the network is considerably high. This is adressed in two ways. Firstly, the Bayesian Network is modeled in an object-oriented manner (OOBN) [Koller and Pfeffer, 1997] which allows for modularization and reuse of subnets. Secondly, not all conditional probabilities in the network are trained but they are parameterized by hand. For example, the node *LaneChange* is set to the state *right* if the probability of node *CrossingLaneMarkingLeft* is 0 and the probability of node *CrossingLaneMarkingRight* is 1. Unfortunately, the authors do not provide a quantitative evaluation of their approach for demonstrating its feasibility.

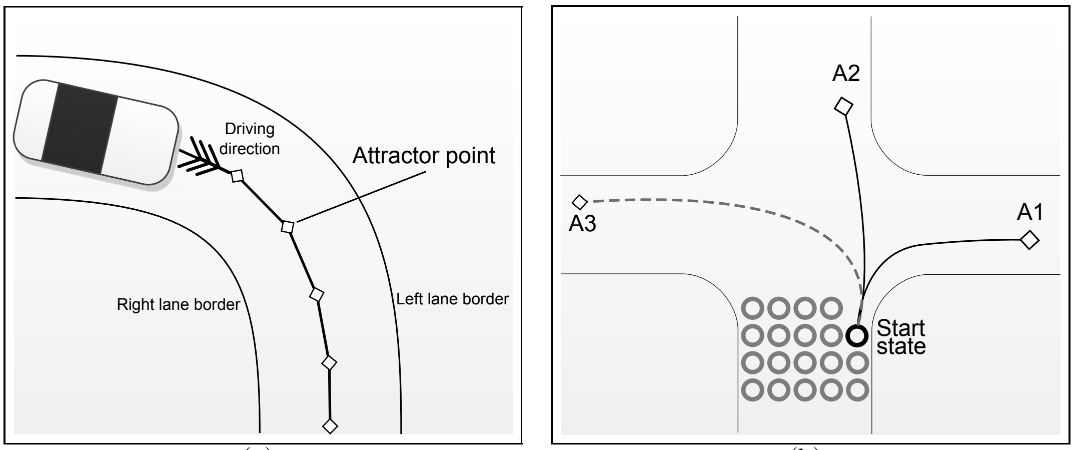
2.1.3. Trajectory Prediction

A method for intention estimation outputs the upcoming maneuver of an observed vehicle. In some cases, however, it might not only be of interest *which* maneuver a vehicle will perform but also *how* it will execute it. Knowing when a vehicle will be at a certain position is an important requirement for an accurate collision avoidance system. Therefore a prediction of a vehicle's path over time, namely its trajectory, is needed. The following approaches are concerned with this problem.

In [Yao et al., 2013] a method for predicting trajectories during lane change maneuvers is presented. The method relies on a large database of previously recorded lane change maneuvers. When an observed vehicle initiates a lane change, its current trajectory is compared to the ones already in the database. The distance metric for this comparison takes besides kinematic properties also the distances to nearby, surrounding vehicles into account. By means of a nearest-neighbor algorithm, the k most similar trajectories are retrieved from the database and combined into a single one using an inverse distance weighting. At the moment the approach works only on straight highway sections and does consider possible changes in a vehicles velocity during the maneuver only via heuristics.

A parametric approach to trajectory prediction is undertaken in [Hermes et al., 2009]. In this work the goal is to predict a vehicle's motion for intervals of up to three seconds. The proposed system applies a two-step procedure: In the first step a coarse path is predicted, which is then refined in the second step. The path is predicted by using an RBF network classifier [Schürmann, 1996] with adapted radial basis functions. Instead of computing the Euclidean distance between training samples the radial basis functions employ a variant of a string matching method that is known for its suitability for trajectories. In the second step, particle filters are initialized with the path given by the classifier. The mean-shift algorithm [Comaniciu and Meer, 2002] condenses the trajectories predicted by the individual particle filters into a single one. Based on recorded vehicle odometry data the authors can show that their method achieves an accurate motion prediction even up to three seconds in the future. Nevertheless, for this result the number of considered path alternatives are limited to two very distinct ones.

A more general take on long term motion prediction is presented in [Alin et al., 2012]. It is based on the intuition that a vehicle's path is not only determined by its current dynamics but also by its surrounding, for example by the course of its current lane or the behavior of other traffic participants. This environmental knowledge is incorporated as attractor functions into a Bayesian filtering framework. The basis of the framework is provided by a grid-based Bayesian filter, that distributes the state estimate of a vehicle's position, velocity and direction over a uniformly arranged grid to handle multi-modal probability distributions. The probabilities for the individual states in the grid are determined by both the kinematic behavior of the observed vehicle and the influence exercised by the attractor points. In Figure 2.8(a) attractor points that model lane-following are depicted.



(a) (b)

Figure 2.8.: *In [Alin et al., 2012] attractor functions are used for a trajectory prediction method that takes lane information directly into account. A curved lane segment with attractor points along its center that model lane-following behavior (a). Splines represent a path from a grid node to all attractor locations (b). If a spline's curvature is too high such that a vehicle could not traverse it then the corresponding attractor is considered unreachable and not used anymore.*

In order to determine whether a grid node is influenced by a certain attractor splines are fitted between the current vehicle position, this node and the attractor points as shown in Figure 2.8(b). If the curvature of the spline is above a predetermined threshold thus leading to an unrealistic vehicle movement then the attractor is not considered for that node. In an evaluation on simulated data Alin et al. can show that using environmental knowledge improves both tracking and prediction accuracy. One downside of the approach is that no method for the parameterization and weighting of attractor points is provided.

The authors of [Petrich et al., 2013] are also incorporating lane-following into their models in order to obtain more accurately estimated trajectories for long prediction horizons. Petrich et al. employ a stochastic filter framework based on Extended Kalman Filters (EKF) for predicting the lateral dynamics of vehicles for up to 4.8 seconds in advance. A key element of their approach is the use of Active Lane Points (ALP) for encoding a vehicle's lateral position with respect to nearby lanes. The ALP of a lane is the perpendicular projection of a vehicle's x and y position to the center of this lane. Under consideration of measurement noise and the typical deviation exercised by drivers during lane-following, the distance to nearby lanes can be used to determine which lane a driver is currently driving on. This aspect makes the approach suitable for multilane roads, as opposed to the approach presented by Alin et al. Additionally, Petrich et al. do not assume that the observed vehicle stays strictly in the same lane but they are able to detect lane-changing maneuvers based on the change of the distance to the nearest ADP over time. The trajectory prediction is in all cases obtained by initializing the EKF with the current dynamics and using the nearest ADP as pseudo-measurement for its update step. In a qualitative evaluation, the authors show the general feasibility of their method. Unfortunately, it does not yet consider longitudinal dynamics, which is necessary for anticipating critical situations.

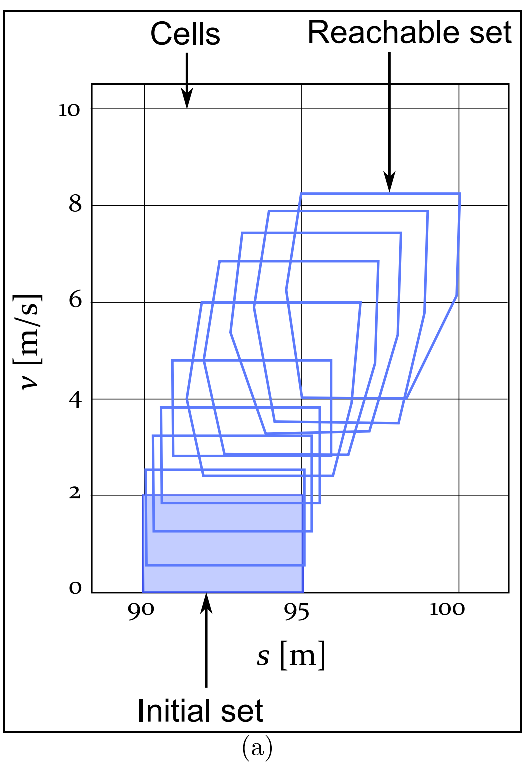
In [Althoff et al., 2009] lane-following is only one of the aspects that are taken into account in their prediction system. Additionally they also incorporate other traffic participants along with their interactions and maneuvers. In order to make such a comprehensive state space tractable, Althoff et al. rely on Markov chains [Norris, 1998] and a coarse discretization of the state space. Markov chains model the transition probabilities from one discrete state to the next and are used here to compute stochastic reachable sets for the longitudinal and lateral behavior of all nearby road users. An illustrating comparison of a continuous reachability set and a discrete, stochastic reachability set based on Markov chains is given in Figure 2.9.

The probability distribution in the sets is adjusted by models that implement car-following behavior and lane change behavior by increasing the probabilities in cells that are in line with these behaviors. The final result is a set of probabilistic trajectories of all considered vehicles. A drawback of this approach is its complexity: it has a high number of free parameters that need to be carefully set to reasonable values.

2.1.4. Combined Approaches

As it was shown above, the accuracy of a trajectory prediction can be significantly improved by incorporating information about the course of roads and individual lanes, since especially in single-lane roads it is viable to assume that a driver will stay within the borders of its current lane. But this assumption does not hold anymore when the lane splits up as it is the case at intersections. In order to still make use of lane information a prediction method needs to know which of the alternatives a driver plans to take, which is addressed by intention estimation. Due to this dependency, there are some works that combine intention estimation and trajectory prediction in a common framework for obtaining accurate predictions on both maneuver level and physical level. Two of the most relevant approaches are discussed in the following.

The approach presented in [Gindele et al., 2013] employs a single Dynamic



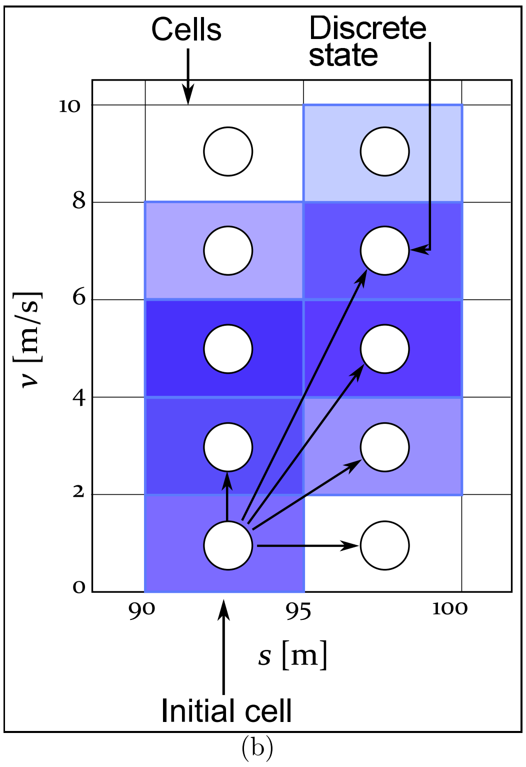


Figure 2.9.: *In [Althoff et al., 2009] discrete reachable sets are proposed for a trajectory prediction method that takes lane information as well as other road users into account. Reachable set for a given time interval, where blue polygons describe the future development of position and velocity (a). The corresponding stochastic reachable set of a Markov-chain is discretized and provides probabilities for individual cells in the state space, encoded here in saturation levels of blue (b).*

Bayesian Network. The nodes of the DBN encode context knowledge with their states determined by a set of separate models, each of them capturing a different aspect of the environment. For example, one model is tagged as *Lane Matching Model* and uses a vehicle's position relative to nearby lanes to determine the probability of the vehicle following the respective lane. Another model, the *Traffic Participants Relations Model* consists of multiple submodels that turns interrelation between road users like right of way or TTC into probabilities. All of the context models are combined by a *Policy Model* that takes evidence about the environment of a vehicle as input and returns a probabilistic estimate of its future behavior, e.g. the future trajectory. This combining model is trained on recorded driving data, while the other models are parameterized by experts in order to make better use of limited training data. While the authors claim that the approach can handle arbitrary numbers of road users the evaluation is restricted to only two vehicles approaching an intersection.

In [Tran and Firl, 2013] a prediction method is presented that combines a linear, probabilistic regression algorithm named Gaussian Processes [Doob, 1944] with a nonlinear filtering method, the Unscented Kalman Filters (UKF) [Julier and Uhlmann, 1997]. The goal of the work is to determine which route a vehicle approaching an urban intersection will take and how it will execute its maneuver. In the first step, a dataset of approaching maneuvers was recorded using a 360° laser scanner stationed at an intersection. For each of the three possible maneuvers, *turn-left, turn-right, go-straight* a pair of two-dimensional Gaussian Processes is trained. In a coordinate system that is stationary for the considered intersection, one Gaussian Process learns the horizontal velocity and the other one the vertical velocity for a given position. A pair of Gaussian Processes can be seen as describing a motion flow field describing the expected vehicle movement for a given position. An illustrative example of such a motion flow field is given in Figure 2.10.

For a vehicle approaching the intersection, the intention estimation is realized by computing the likelihood of all intentions given the vehicles position and velocity. This likelihood can be directly obtained from the trained pairs of Gaussian Processes. The maneuver that is associated with the pair having the clearly highest likelihood is considered the intention, where 'clearly' means that its likelihood is at least twice as high as for any other model. Once the intention has been determined, the trajectory prediction is accomplished by an Unscented Kalman Filter, which obtains its measurement updates from the Gaussian Processes as well, by sampling from the motion flow field. The advantage of coupling Gaussian Processes and UKF is that for each part of the trajectory the uncertainty of the prediction can be determined. The authors do not provide a quantitative evaluation of their approach but one downside of it is its inability to handle cases where multiple cars arrive at an intersection and interact.

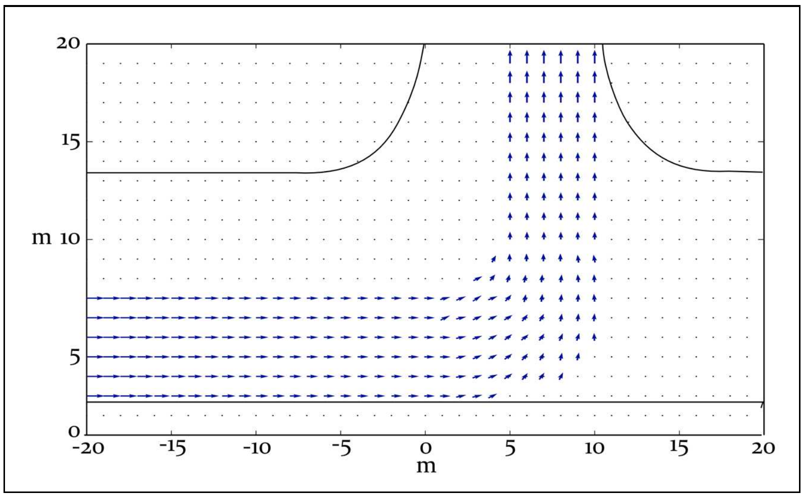


Figure 2.10.: *Schematic Motion flow field as taken from a pair of Gaussian Processes for jointly predicting the intention and the trajectory of a vehicle. It shows the most probable velocity vector for a vehicle performing a left turn. From the length of the arrows one can tell that a vehicle slows down before turning and accelerates afterwards. Adapted from [Tran and Firl, 2013].*

2.2. Discussion

As already stated in the beginning of this chapter, there are two problems for which suitable methods are needed. The methods should be able to:

1. Assess complex traffic situations in inner-city driving. Determine which road users interact and in which way.

2. Anticipate the future behavior of other road users

The first problem is investigated in the field of Situation Assessment; related methods were discussed in Subsection 2.1.1. It turned out that none of these methods provide an adequate solution. The approaches presented in [Reichel et al., 2010] and [Schubert et al., 2010] consider only a single, specific situation and are restricted to highway scenarios. There is no obvious way how these algorithms can be extended to multiple, distinct situations or adapted to an urban setting. More versatile methods have been proposed in [Vacek et al., 2007] and [Schamm and Zöllner, 2011]. Their frameworks can handle various types of situations but this versatility is achieved by a labor-intensive modeling of the considered situations by human experts. The framework presented in [Hülsen et al., 2011] requires less handcrafting and stands out with its ability to infer a large number of relations that also take traffic signs and rules into account. Unfortunately, the logic formalism providing this capability is computationally costly and unable to handle noisy sensors. But especially the fact that nowadays sensors provide imperfect and uncertain measurements is a problem that is recognized and increasingly addressed by the research community. This is why most of the discussed works rely on some kind of probabilistic modeling to incorporate uncertain sensor measurements, and the method developed in this thesis should possess this ability as well.

The second problem is addressed by research in the areas of Intention Estimation and Trajectory Prediction. Works from the former area were discussed in Subsection 2.1.2. They contain useful ideas but all of them miss at least one aspect import for this work. The approaches proposed in [Graf et al., 2013] and [Dagli et al., 2003] struggle with their complexity and either require a coarse discretization or guessed parameters in order to be tractable. The works presented in [Aoude et al., 2012] and [Hayashi and Yamada, 2009] are tailored to a very specific situation that is only rarely encountered in everyday driving. A more common situation is considered in [Lefèvre et al., 2012], however, the method fails as soon as a vehicle driving in front of the observed vehicle influences the observed vehicle's velocity. Car-following behavior is more directly addressed in the works of [Lidstrom and Larsson, 2008] and [Liebner et al., 2012] but their methods also experience a sharp drop in accuracy whenever a vehicle's velocity profile is not only determined by its intention but also by a preceding vehicle. Still, the velocity profile appears to be an import feature for estimating a drivers intention. At the same time it is important to take interactions between vehicles into account.

In Subsection 2.1.3 works from the area of Trajectory Prediction were discussed and, again, none of these allow a direct application to the problem at hand. For example, one method is limited to lane change maneuvers [Yao et al., 2013] while another method requires predefined path alternatives [Hermes et al., 2009]. The approach taken in [Alin et al., 2012] is tailored to single lane roads but disregards interactions with other vehicles. In [Petrich et al., 2013] multilane roads are explicitly modeled, but other vehicles are also not considered. The more comprehensive framework proposed in [Althoff et al., 2009] takes other vehicles into account but requires a large number of parameters whose values are not trivial to determine.

Two works combining Intention Estimation and Trajectory Prediction in a single framework were discussed in Subsection 2.1.4. In [Gindele et al., 2013] the authors also struggle with the parameterization of their complex model while the method proposed in [Tran and Firl, 2013] neglects other vehicles.

Two important insights can be drawn from this literature review. The first is, that probabilistic methods are the state-of-the-art in all of the three presented fields. Their ability to model sensor uncertainties as well as hidden states like a driver's intention make them very attractive for a use in this work. The second insight is that even for predicting trajectories not only kinematic but also situational aspects are gaining increasing attention. Interactions between vehicles, especially during car-following situations, are modeled by many approaches. However, in these cases the preceding vehicle is simply assumed to be currently relevant for the following driver instead of determining this relevance methodically. A framework that would be able to determine first which entities in a given situation interact could then exploit this knowledge to improve the behavior prediction of individual behaviors. This means, that a method for solving problem 1.) is needed that is versatile enough to provide the basis for solving problem 2.). The development of such a method will be detailed in Chapter 3.

3. Configurations

The literature review in Chapter 2 demonstrated the need for a scalable method for situation assessment. An elemental building block for the method proposed in the work at hand is the concept of *configurations*, that will be detailed in this chapter. It is a novel method for modeling and understanding the interactions between road users that was also developed in the course of this work.

This chapter is structured as follows. In Section 3.1 an explanation is given why standard approaches fail for inner-city traffic situations. Based on the insights a model tagged *configuration* is proposed that avoids the shortcomings of standard approaches. Its specification is given in Section 3.2.

3.1. Motivation

The central goal of this work is to develop a method for assessing traffic situations as they are typically encountered in urban driving, especially when negotiating intersections. Assessing a situation means in this work to obtain an understanding that is sufficient to either, from a driver's point of view, take correct actions, or, from an observer's point of view, anticipate the actions of others. The thesis at hand puts its emphasis on the latter part: to anticipate the actions of other road users based on the interpretation of their current state and individual driving situation.

Independent from the specific goal, a situation assessment consists of two steps.

1. **Perception** In the first step, one or multiple sensors take measurements from the surroundings in order to obtain a preferably complete registration of all relevant entities and the environment. These sensors can be video cameras, laser scanners or radar systems to name a few. Raw measurements need to be processed with specific algorithms, e.g. for detecting lanes in camera images, recognizing pedestrians in laser scans or classifying radar reflections as vehicles.

2. **Interpretation** The environment and the entities perceived in the first step provide the basis for the second step. In this step questions like 'Which rules apply for a certain entity?', 'What determines the current behavior of

a vehicle?' or 'Which entities are interrelated?' are posed and answered given the provided evidence.

Put simply, the first step tackles the 'What?' and 'Where?' aspect of a given traffic situation, whereas the second step tackles the 'Why?' and 'How?' aspect. Whereas the first step returns a physical representation of a traffic situation, the second step aims for a logical, interpretative representation. This thesis takes the imperfections of nowadays sensor systems explicitly into account, but apart from that considers the physical representation as given and puts its focus on the logical representation. This representation explains a situation by identifying those relations, patterns and interactions that have caused the observed situation. The gain of such an explanation is twofold: Knowing what determines a vehicle's *current* behavior is an important prerequisite for anticipating its *future* behavior. This can be illustrated by the scenario given in Figure 3.1: Vehicle 'A' overtakes a slow-driving, preceding truck and changes to the leftmost lane. But as on this lane vehicle 'B' is already arriving at a much higher speed, 'B' is forced to brake in order to avoid a collision with 'A'. The interpretation *'B' has to slow down because of the maneuver of 'A'* serves, a short time before 'A' initiates its lane change, as a prediction: *'B'* **will** *slow down because of the maneuver of 'A'.*

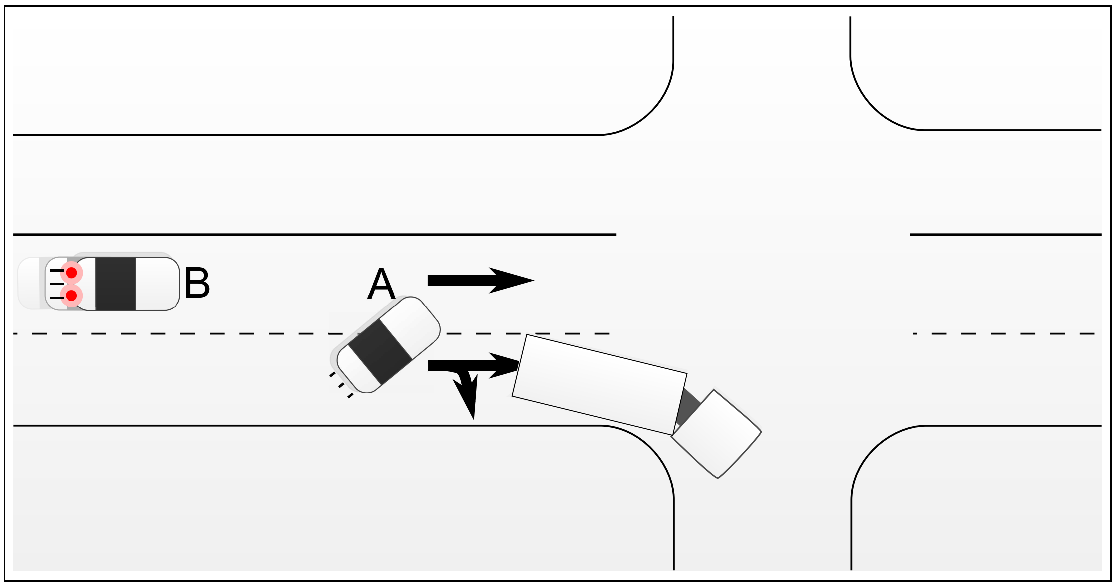


Figure 3.1.: *A critical situation where the ability to interpret the relations between vehicles A and B is useful. For example, anticipating that the maneuver of A forces B to brake allows successive vehicles to keep a sufficient headway to B.*

The second advantage of an interpretation is that it can be used to evaluate the consequences of maneuvers which is important for choosing the most appropriate maneuver from a set of different alternatives. In the example above, 'A' can choose between waiting behind the truck and overtaking it. If 'A' was operated by a system that considers that overtaking results in a critical situation with 'B', it would have selected to wait.

3.1.1. Challenges: Complexity and Variability of Intersection Situations

In order to interpret a specific situation a general model is needed that describes how traffic-related entities behave and interact. It turns out that the development of a general model is a challenging endeavor, as a straightforward solution is hindered by two obstacles which are particularly prominent in inner-city driving: the high variability and the high complexity of intersection scenarios.

'High variability' describes the insight that a driver will rarely encounter a situation twice whenever one ore more other road users are present, as each time their positions and dynamics will be different. The examples given in Figure 3.2 illustrates that this property also holds on a simple intersection with only two road users. Of course, also highway scenarios are subject to a certain extent of variability, but as opposed to inner-city scenarios they are significantly more constrained as there is no crossing traffic. At the same time they possess a lower diversity of road users as bicyclists and pedestrians do usually not take part in this traffic.

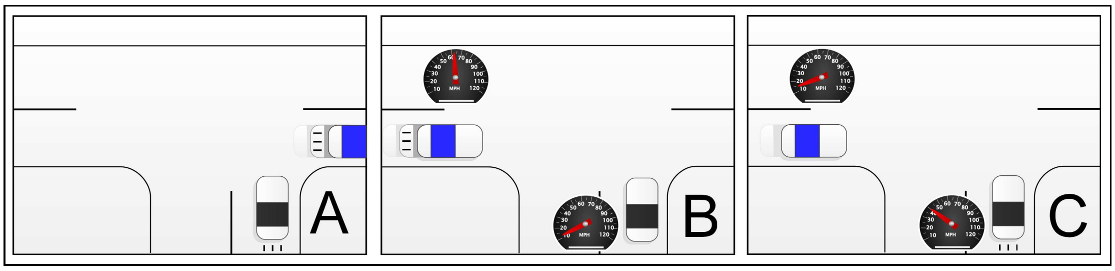


Figure 3.2.: *Three different situations at the same intersection. Even though only two vehicles are participating, all three situations are inherently different. Situations B and C differ only by the dynamics, such that in B the black vehicle could not enter the intersection before the blue one, but in C it could.*

The challenge of a high variability comes from the fact that it is not possible to build a system that is trained for all possible situations that can be encountered - simply because the number of possible situations is infinite. Accordingly, a solution based on a set of prototypical situations would be too inflexible to fit every situation.

The second obstacle, the high complexity, is visualized by the image shown in Figure 3.3.

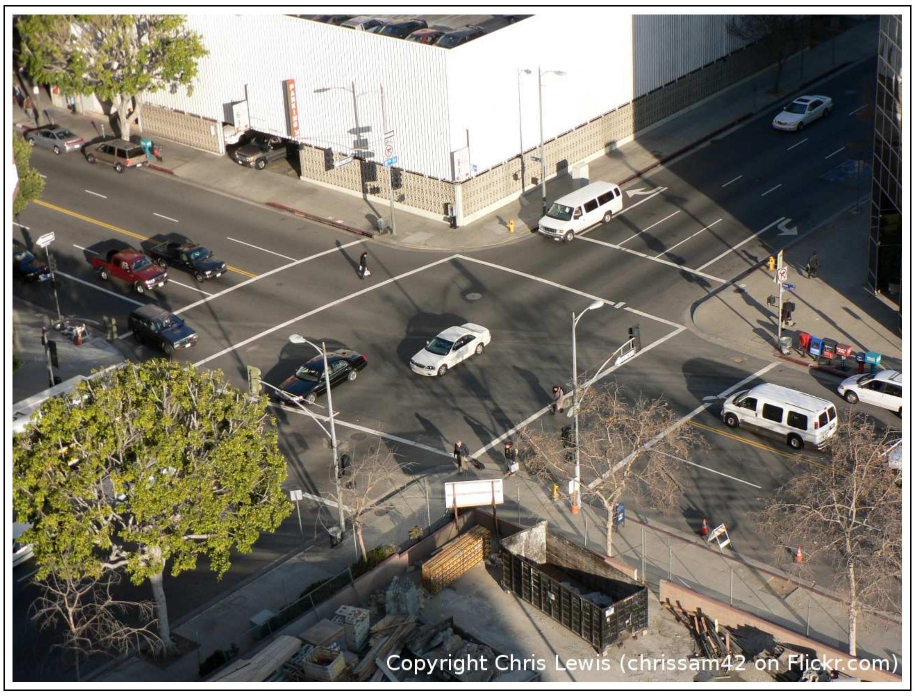


Figure 3.3.: *Although the intersection shown above is only of a medium size, interpreting applicable traffic rules, interactions and possible maneuvers takes a significant amount of time even for a human observer.*

It depicts an urban intersection of medium size. Though the number of road users present is rather moderate, it takes even for a human observer some time to grasp applicable rules and admissible maneuvers for all vehicles close to the intersection. The example shows that even for a basic intersection a situation assessment is non-trivial. And its complexity rises significantly when more and a higher variety of road users need to be considered. The more entities are present, the more potential dependencies and interactions need to be checked and evaluated by an algorithm. A straightforward method that relates each entity with any other entity is therefore intractable for all but the smallest intersections.

3.1.2. Solution: Decomposing a Situation into parts

In Computer Science, a common approach for dealing with overly complex problems is to decompose the problem into smaller subproblems that are easier to handle. Prominent examples are algorithms for sorting which can be reduced from a polynomial complexity down to quasilinear complexity, by sorting small subsets of the data and merging these intermediate results later on. Transferred to the problem of situation assessment, this means to decompose a complex traffic situation into parts that can be analyzed separately.

A decomposition can also be used to tackle the high variability of traffic situations. Traffic situations are not concerted by a single, controlling instance, but arise from the actions performed by multiple, independent agents, such as drivers or pedestrians. Because of that, one can argue that a complex traffic situation is the aggregation of much simpler situations, like individual interactions between two entities. These basic situations are then the parts into which a complex situation can be decomposed.

The preceding considerations suggest a parts-based approach, by which all present entities are sorted into small groups. The challenge here is to ensure that the decomposition process does not loose valuable information about the situation. A part needs to comprise all the information currently relevant to the entities contained, which is why the notion of relevance has to be a key element of the decomposition scheme. It is often the case that for a given road user not all of the present entities are equally relevant, as it is illustrated in Figure 3.4.

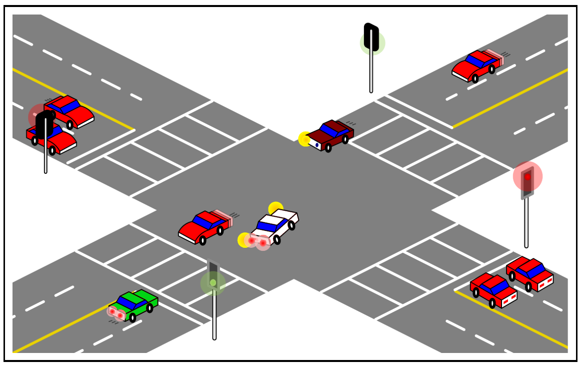


Figure 3.4.: *For a road user not all present entities are equally relevant. For example, the green vehicle can neglect the cars waiting at the red traffic lights.*

From the green car's view, the vehicles currently waiting at red traffic lights are of no direct relevance and will thus not affect its immediate behavior. Instead, the green car is mainly affected by the white car ahead in the center of the intersection. The white car itself has stopped in order to yield to oncoming traffic thus blocking the green car's way, which is thereby forced to slow down.

This shows that for a given road user the relevance of other entities is based on their effect on his behavior. According to that, a possible decomposition scheme is to create overlapping sets of each road user and all of its affecting entities. In this thesis, such a set is termed *Configuration*. It is a novel concept proposed and extensively used in the work at hand.

3.2. Specification

A configuration is a model for describing the relation between a road user and the entities that are affecting the road user. There might be multiple possibilities how the influence of one entity on another can be quantified, but in this work it is based on acceleration. An entity is affecting another entity, for example a vehicle, when it is causing the vehicle to slow down or stop. A vehicle that slows down for a crossing pedestrian or stops in front of a red traffic light is considered to be affected by the pedestrian or the traffic light, respectively. Analog to that, a vehicle accelerating for reaching a desired velocity or keeping its current one is considered to be currently unaffected, as the longitudinal behavior of the driver is solely determined by his own goals. Using deceleration and stopping as a measure of influence is motivated by two aspects. First, these behaviors are more interesting for safety concerns, as vehicles are usually able to decelerate far more suddenly than they can accelerate, and rear-ending is a typical crash scenario in urban environments. Second, the causes for a decelerating behavior can be generally obtained from cues in the near surroundings, like red traffic lights or obstacles. Nevertheless, the restriction to this definition of influence is not a limitation by the concept of configurations itself, but a deliberate choice in the work at hand.

This is also the case for the decision to constrain the number of affecting entities in a configuration to one, the most influential one, reducing the complexity of individual configurations. Nevertheless, if it is so desired, multiple affecting entities can be still modeled by using multiple configurations.

The use of configurations aims at decomposing a traffic situation into sets of related entities, in order to obtain an understanding which road user is affected by what. Furthermore, this understanding can be useful in additional ways, for example as a preprocessing step for attention control, such that attention can be focused on entities that were found relevant. Another use case is behavior prediction, where the information how an observed vehicle is affected by others can be directly incorporated in the prediction process. These and further areas of application will be explored in Chapter 4.

The formal specification of a configuration is obtained by describing the respective entities and their relations in a graph. Using graphs as specification method was primarily motivated by their suitability for modeling relationships in a comprehensible representation. The graph for a configuration C is an ordered tuple

$$C = \{L, R, A, F, E\} \tag{3.1}$$

where:

L	= root node, holding the configuration label
R	= node that represents the reference entity
A	= node that represents the affecting entity
$\boldsymbol{F}$	= nodes that describe the relations of entities when being in that particular configuration
$\boldsymbol{E}$	= edges between nodes

Herein, the set of nodes $\boldsymbol{F}$ plays an important role for deciding whether a certain configuration is present or not. This is accomplished by inspecting the relations of and between reference and affecting entity. Relations can be either unary when involving only a single entity or binary when involving both entities. A unary relation of an entity is simply its state, like its position, velocity or orientation. Binary relations result from combining states of both entities into a single value. For example, the binary relation $distance$ is the difference in $position$ between the two entities and $relativeVelocity$ is their difference in $velocity$.

Relations are specified as child nodes of their corresponding entity and can be used for recognizing the configuration of a vehicle. For this purpose a classifier is learned that decides based on the values of the relations whether the corresponding configuration is present. Accordingly, the values of the relations serve as features for the classifier. It is important to note that a human expert needs only to specify which relations are necessary for a correct recognition but not the values themselves.

In the following, the specification of a configuration is explained by means of an example. The configuration used for it is labeled *StoppedByRedTrafficLight* which is present whenever a vehicle has to slow down or stop in front of a red traffic light. Its graph is plotted in Figure 3.5.

The root node L situated on top of the graph holds the label. It has two child nodes: the reference entity node R stating that the affected entity is a vehicle and the affecting entity node A defining that the corresponding entity is a traffic light. The remaining nodes are feature nodes $\boldsymbol{F}$ that define in which relations the involved entities need to be so that the configuration is present. Slowing down and stopping is captured by the features $acceleration$ and $velocity$, respectively, which are unary relations. As they belong to the reference entity they are modeled as its child nodes. Likewise, the property 'red' is a state of the affecting entity, the traffic light, that is implemented by the feature *trafficLightState*, also modeled as child node. Additionally, the specifying expert incorporated the knowledge that a vehicle will not be affected by a red traffic light when it is arbitrarily far away, but only when it is in a certain range. That is why a binary relation for the feature $distance$ is added as common child node of both involved entities.

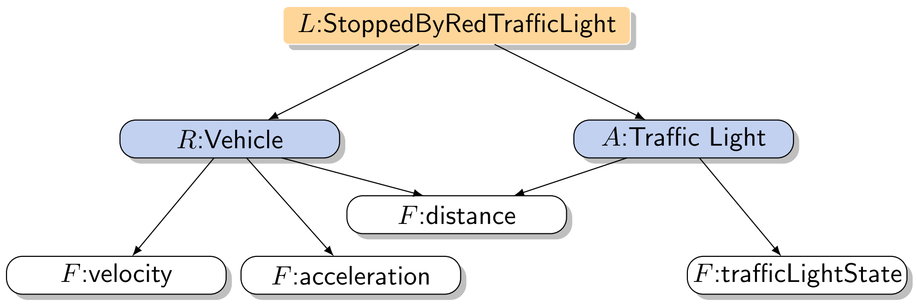


Figure 3.5.: *A graph-based specification of the* StoppedByRedTrafficLight-*configuration. The root node L holds the label, the reference entity R is a vehicle and the affecting entity A is a traffic light. F describes which relations of the entities are relevant for recognizing the configuration.*

At this point the specification process is complete. The example demonstrates that the amount of hand-crafting is limited to a minimum – the involved entities and possibly relevant features – while still providing the opportunity to incorporate domain knowledge.

3.3. Classification Methods

As discussed above, the motivation behind the concept of configurations is to decompose complex traffic situations into parts, which are sets of interrelated entities. The specification of a set of interrelated entities belonging to a configuration was given in the preceding section. But as the specification is reduced to the bare minimum it only states which relations are cues for a configuration to be present. The actual values, for example the maximum distance in meters to the considered traffic light in the *StoppedByRedTrafficLight*-configuration (see Figure 3.5), are not specified. Likewise, a vehicle will only be in a *StoppedByRedTrafficLight*-configuration when its *acceleration* is negative or its *velocity* is zero, but this information is not given either. Instead, the values that a feature or a combination thereof takes when a configuration is present need to be learned by a dedicated learning algorithm. The algorithm requires a batch of labeled training data that consists of feature combinations along with the information whether the configuration is present for that combination. Given the training data, the algorithm learns a decision function for recognizing configurations, i.e. that can decide for a given pair of entities whether they are in a configuration or not, based on their features. Such a method is called a classification algorithm.

A classification algorithm takes feature values as input and returns a label, in this case whether for a pair of entities a configuration is present or not.

$$h(\mathcal{F}) \rightarrow \{0, 1\} \tag{3.2}$$

Here, h denotes the classification function, $\mathcal{F}$ denotes the feature values of the considered configuration and a label of 1 denotes that the configuration is present and a label of 0 that it is not present.

If multiple configurations are considered, say N different ones, Equation 3.2 becomes

$$h(\mathcal{F}_1, \mathcal{F}_2, ..., \mathcal{F}_N) \rightarrow \{0, 1, 2, ..., N\} \tag{3.3}$$

with $\mathcal{F}_n$ denoting the features of the n'th configuration and $n \in \{0, 1, ..., N\}$.

There is a large variety of state-of-the-art classification algorithms, most notably Support Vector Machine [Cortes and Vapnik, 1995], Neural Networks [Bishop, 1995] and Tree Ensembles [Breiman, 2001] for nonlinear classifiers and Logistic Regression and Probabilistic Models for linear classifiers. Nonlinear classifiers are capable of learning more complex decision functions than linear classifiers but they lack an important ability that was found to be crucial for this work: Probabilistic treatment. In the problem formulation of Section 1.2 it was demanded that the method for situation assessment should be able to cope with inaccurate sensor measurements. The significance of this property was confirmed in the literature review of Chapter 2 where almost all state-of-the-art methods addressed the issue of unreliable sensor data. All of those who did, use probabilistic methods for explicitly taking the uncertainty of a sensor reading into account. While there are extensions to nonlinear classification algorithms that aim at incorporating probabilities, for example as the level of confidence for the returned labels, only linear methods allow a consistent treatment.

This reduces the set of possible algorithms to Logistic Regression and Probabilistic Models. Hereof, the latter has another useful property: It can also handle the case when a sensor measurement is not available at all, be it a fault or a lack of computational resources, both of which are possible scenarios in automotive applications. Probabilistic Models allow a thorough probabilistic treatment of all the information obtained (or lacking) from sensors. Accordingly, their output is not a single label like in Equation 3.3. Instead, the models compute the probability of each individual configuration C_n to be present, given all features $\mathcal{F}_n$:

$$h(\mathcal{F}_1, \mathcal{F}_2, ..., \mathcal{F}_N) \rightarrow P(C|\mathcal{F}_1, \mathcal{F}_2, ..., \mathcal{F}_N) \tag{3.4}$$

C is a random variable with the states $\{C_0, C_1, C_2, ..., C_N\}$.

Using a fully probabilistic method for situation recognition means to consider besides C also all individual feature values of $\mathcal{F}_n \in \boldsymbol{\mathcal{F}}$ as random variables. A straightforward probability assessment, that conditions each variable on each other, can turn out prohibitively complex. For example, even in the simpler case of recognizing only a single configuration C_n, with K individual feature values $\{f_1, ..., f_k, ...f_K\}$ of $\mathcal{F}_n$, the joint probability distribution is

$$\begin{aligned} P(C_n, f_1, ..., f_k, ..., f_K) = P(C_n) \times P(f_1|C_n) \times P(f_2|C_n, f_1) \times ... \\ \times P(f_K|C_n, f_1, ..., f_k, ..., f_{K-1}) \end{aligned} \tag{3.5}$$

It becomes clear that conditioning each random variable on each other is impractical as it requires to compute a large number of probabilities. That is why conditional dependency is usually only assumed for certain sets of random variables and most variables are considered conditionally independent of each other. To specify dependency relations, graphical models are used that are termed Bayesian Networks [Pearl, 1988]. In the following, their application to recognizing configurations will be outlined in order to explain the process of mapping configuration graphs to Bayesian Networks. However, a comprehensive explanation of the recognition method will be given in 4.1. Furthermore, a brief introduction to Bayesian Networks can be found in Appendix A.

Besides the ability to specify conditional dependency, Bayesian Networks also provide a consistent calculus that defines how probabilities are computed based on the given representation. One downside of Bayesian Networks is that they are still computationally demanding and require like all probabilistic methods substantial amounts of training data for working properly. Nevertheless, these shortcomings are overcompensated by two major advantages. A useful property of Bayesian Networks is that they are white-box classifiers, which means that they allow for introspection on how they arrive at their results. The parameters of a Bayesian Network give a clear statement on the contribution of individual features and how their values affect the final outcome. Especially in safety-critical applications this information is valuable to check for unwanted system behavior.

The second major advantage is the close relationship to the graphical representation used for configurations. It allows to map configurations specified as graphs directly to Bayesian Networks, using a simple procedure. The procedure removes both reference and affecting entity nodes R and A, as they do not represent a random variable, and attaches all feature nodes as children of the label node as described in the following procedure:

1. **Root node:** The label node becomes the only parent of the Bayesian Network and provides the classification result. It holds two states, with *True* denoting the configuration is present and *False* denoting it is not.

2. **Entity nodes:** Both entity nodes are removed.

3. **Feature nodes:** All feature nodes become direct child nodes of the root node.

An illustration of the mapping result is given in Figure 3.6.

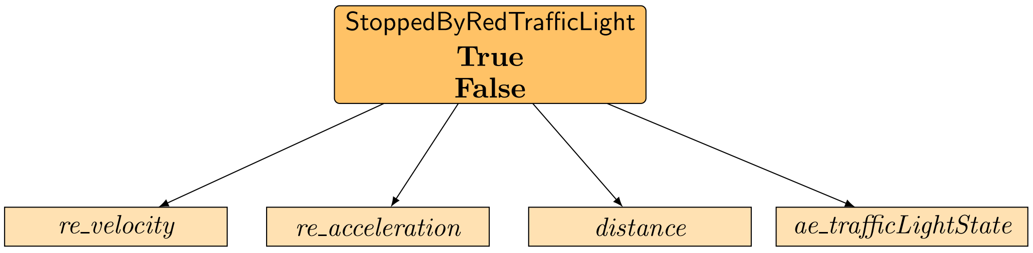


Figure 3.6.: *The Bayesian Network resulting from the* StoppedByRedTrafficLight-*configuration. The prefixes* 're' *and* 'ae' *indicate that the nodes represent features from reference entity and affecting entity, respectively.*

The resulting Bayesian Network is a so called Naive Bayes Classifier, which assumes conditional independence between all features, and is therefore computationally cheap and requires only small amounts of training data. Due to the conditional independence, the joint probability distribution from Equation 3.5 becomes, apart from a constant scaling factor Z:

$$P(C_n, f_1, ..., f_k, ..., f_K) \propto \frac{1}{Z} P(C_n) \prod_k P(C_n|f_k) \tag{3.6}$$

While conditional independence seems to be an assumption that severely restricts the applicability of the algorithm, it has been shown that Naive Bayes Classifiers perform well even in cases where the assumption does not fully hold [Zhang, 2004].

3.4. Discussion

In this chapter the concept of *configurations* was presented which was developed in the course of this work. It is the result from the insight that inner-city traffic situations are too complex and too diverse to be assessed in a straightforward manner. Thus, neither an assessment based on a manageable set of predefined, prototypical situations nor a brute-force interpretation relating every entity to another will be feasible approaches. The solution proposed here is to tackle both complexity and variety of traffic situations by decomposing them into smaller parts which are easier to handle. Influence is the main criterion for the decomposition in that each part consists of a pair of interrelated entities where the behavior of one entity is affected by the other. Possible constellations of interrelated entities are defined by a human expert in a model that is tagged *configuration*.

A configuration consists of a label and two entities: the affected or reference entity and the affecting entity. Furthermore, a set of features determines which states and relations need to be considered to judge the presence of this configuration. All this information is specified in a graph notation which was chosen because of its suitability for encoding relational aspects.

An advantage of a graph notation is that it can be directly mapped to the classifier used for recognizing configurations: Bayesian Networks. A Bayesian Network is a probabilistic model that computes the probability of an entity being in a given configuration based on the corresponding feature values. The advantage of recognizing configurations using a probabilistic model is that it can naturally cope with inaccurate or missing sensor data as it can be expected in an automotive setting. The Bayesian Network used for the recognition is designed as Naive Bayes Classifier, because they work even when the number of training samples is low and because they are computationally cheap.

Configurations are a fundamental concept in this work as they offer a basis for scalable situation assessment. All methods presented in Chapter 4 build on top of this approach. The amenities of this concept will be shown in Chapter 6.

4. Situation Assessment Using Configurations

In Chapter 3 a method for modeling complex traffic situations has been introduced that is based on the concept of configurations. Using configurations a traffic situation is assessed according to the notion of relevance: Inter-dependencies between road users are identified in order to make a situation interpretable. In this chapter methods that make use of configurations, especially in order to overcome problems of current approaches, are presented.

Before a method can profit from configuration information, it is necessary to recognize a road user's configuration in the first place. In Section 4.1 the probabilistic recognition method using Bayesian Networks is described in detail, that was only briefly outlined in the previous chapter.

Using configurations allows to reduce the complexity of a situation assessment. Nevertheless, it is still beneficial if the computational effort for recognizing configurations can be further decreased. Another method for configuration recognition that reduces the number of necessary sensor readings during the recognition process is presented in Section 4.2.

By employing these methods for recognizing configurations, a descriptive situation assessment can be performed. It provides an interpretation of a situation in that individual behavior is explained on the basis of the identified dependencies. While a descriptive assessment is already of use, it is in many cases more interesting to obtain a predictive assessment. In this case a prediction on the future situation is made which allows to make decisions in time and to foresee upcoming conflicts. In Section 4.3 a method for predicting the future velocity profiles of vehicles is described. Knowledge about a vehicles velocity is especially useful for avoiding rear-end crashes. In Section 4.4 another method for behavior prediction is presented, but in this case the manual specification of configurations is replaced by a learning algorithm.

A central claim in this thesis is that configurations, which are a way of recognizing a vehicle's driving situation, can be used to improve methods that currently neglect situation information. For evaluating this claim a demonstrator system has been built, that estimates the intention of a driver when it approaches an intersection, for example whether the driver plans to cross straight or to turn. The goal is to reliably distinguish between more intentions than the state-of-the-art by incorporating the information obtained from a configuration recognition. This approach is detailed in Section 4.5. A summary of this chapter is given in Section 4.6.

4.1. Recognition of Configurations

In this section a method for recognizing configurations is presented. This method serves as basis for all other methods in this chapter that take configuration information into account. The method was published in [Platho et al., 2012].

4.1.1. Considered Configurations

Before a recognition method can be developed it is first necessary to decide which configurations should be considered. The selection can be guided by various goals, but in this case three criteria appeared to be particularly reasonable:

1. **Criticality** The considered configurations should cover situations in a way that critical settings can be detected.

2. **Recognizability** A configuration is only useful when it can be recognized by nowadays sensor technology.

3. **Commonness** Each of the considered configurations should occur frequently in every-day driving. At the same time the considered configurations should be sufficient to fully model all of the traffic situations encountered.

Criteria 1 and 2 lead to the decision to focus on considerations that result in slowing down or stopping of the reference vehicle. Decelerating behaviors are critical because a vehicle can usually change its velocity more suddenly by braking than by accelerating. Furthermore, unexpected stopping maneuvers are a common cause for rear-end crashes. In terms of recognizability, the advantage of focusing on these configurations is that the cause for a deceleration can usually be explained from the situation, e.g. in case of a blocking obstacle, a red traffic light or a crossing road user.

Criterion 3, commonness, is harder to fulfill. It is difficult to find a set of configurations that fits for explaining all and every situation possibly encountered. But as this work focuses on intersection scenarios, it is sufficient to be able to model typical traffic situations in the vicinity of urban intersections. Still, in urban areas there is a high variety in road users, namely pedestrians or bicyclists, which could all require an individual set of possible configurations. As the methods proposed here are targeted for a vehicle-based assistance system and as other vehicles are the most commonly encountered road users, it was decided to focus on configurations where entities can be vehicles or part of the infrastructure. With these parameters, it was found that three general configurations can cover most of the typical intersection situations. An illustration of these configurations is given in Figure 4.1.

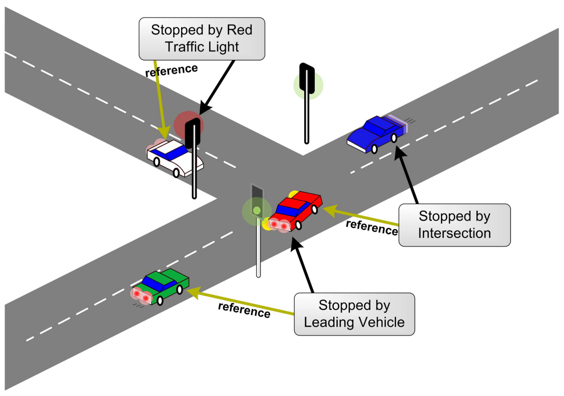


Figure 4.1.: *A traffic situation with various configurations.*

In this illustration, the green vehicle brakes because of the stopped red vehicle ahead. Cases, in which of two consecutive vehicles the leading vehicle forces the following vehicle to slow down, are modeled by a *StoppedByLeadingVehicle*-configuration. Its representation as Bayesian Network is shown in Figure 4.2.

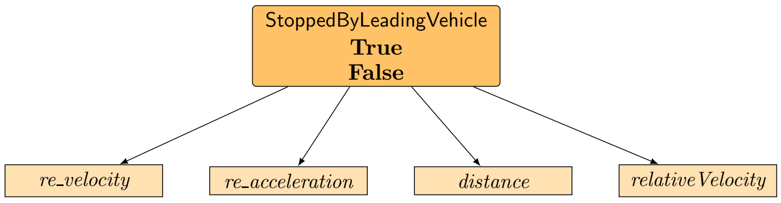


Figure 4.2.: *The Bayesian Network resulting from the* StoppedByLeadingVehicle-*configuration. For recognizing this configuration the features* velocity, acceleration, distance *and* relativeVelocity *are used. The leading* 're' *denotes features that are taken from the reference entity. Features without such a prefix stem from binary relations taking both entities into account.*

The features *re_velocity* and *re_acceleration* are required to check whether the reference entity is currently decelerating or even stopped. This is a necessary condition for a configuration to be present, as defined above. In addition, the distance between reference and affecting entity is taken as feature as well as their relative velocity. The intuition behind these features is that closer vehicles will to a higher probability be in a configuration, especially if the leading vehicle is significantly slower than the following one.

Another typical scenario at intersections is that one vehicle on a minor road yields to another vehicle on a major road, as it is the case for the red and blue vehicle in Figure 4.1. This configuration is tagged *StoppedByIntersection*. It is depicted in Figure 4.3.

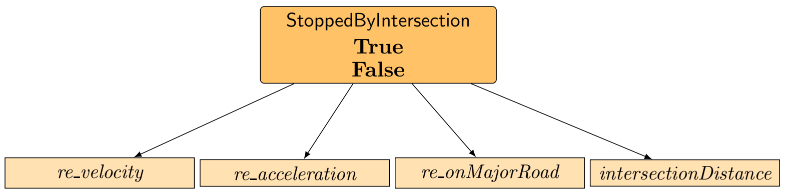


Figure 4.3.: *The Bayesian Network resulting from the* StoppedByIntersection-*configuration.* Velocity, acceleration *and* onMajorRoad *are features of the reference entity, denoted by the prefix* 're'. *The feature* intersectionDistance *is given by the maximum of the distances of both vehicles to the intersection.*

The feature *re_onMajorRoad* captures whether the reference entity is on a major or a minor road and the feature *intersectionDistance* covers the distance of both vehicles to the intersection. A single value is obtained by taking the maximum of both distances. Another option would be to incorporate the distances of both vehicles individually, but besides increasing the state space the Bayesian Network could hardly learn their nonlinear relationship.

The *StoppedByRedTrafficLight*-configuration, in which a vehicle is forced to stop at a red traffic light, has been already discussed in Chapter 3. It is given in Figure 4.4.

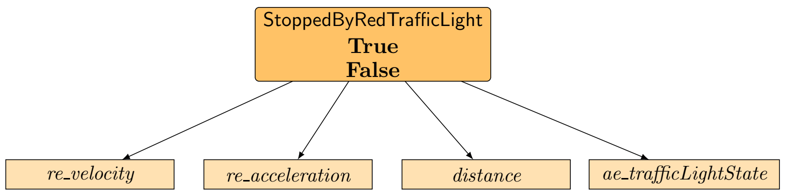


Figure 4.4.: *The Bayesian Network resulting from the* StoppedByRedTrafficLight-*configuration. The prefixes* 're' *and* 'ae' *indicate that the nodes represent features from reference entity and affecting entity, respectively.*

4.1.2. Discretization

When creating a Bayesian Network with continuous features one has the choice between incorporating them as continuous or as discrete nodes. Continuous nodes avoid a loss in precision caused by discretization errors in exchange for a higher computational complexity during learning and inference. Another downside is that continuous nodes can not have discrete child nodes. In order to circumvent these issues and to keep learning and inference tractable, continuous feature are discretized.

Discretization means to define how many states a formerly continuous node should have and then to define the intervals for each state. For example, the continuous feature *re_velocity* could be discretized into three states: *Stopped, Low Velocity*, and *High Velocity.* The interval for the state *Stopped* could comprise all velocities below $2\frac{m}{s}$, the interval for the state *Low Velocity* could comprise all velocities above that and below $7\frac{m}{s}$ and the state *High Velocity* all velocities beyond $7\frac{m}{s}$. It shows that both the number of states for a discretization and the exact intervals provide many parameters that need to be determined. This task can be either performed by an expert or by a dedicated learning method.

A well-known method for performing a discretization via learning was presented in [Friedman and Goldszmidt, 1996]. The learning method guides the discretization process by the principle of the minimum description length for trading the number of discretization levels against the classification accuracy on the training set. However, parameterizing this trade-off is non-trivial. Additionally, there are discretization levels that make more sense than others for individual features. Reasonable values depend to a large extent on the feature itself. For example, a straightforward approach to separating the feature *re_acceleration* into three states for positive, negative and zero acceleration would come up with the levels $> 0\frac{m}{s^2}$, $< 0\frac{m}{s^2}$ and $= 0\frac{m}{s^2}$, respectively. But this neglects the fact that the acceleration of a vehicle is difficult to measure and therefore it is a very noise feature, which oscillates constantly around its true value. Due to this property, levels should be set according to the characteristics of the feature, as it was done for the Bayesian Network used here.

The number of intervals influences the complexity of the resulting network and the effort necessary during learning and inference. It is therefore advisable to keep the number of intervals as small as possible.

In a study a discretization using the method by Friedman & Goldszmidt was evaluated. It showed that the resulting Bayesian Networks fitted the training data much better, but at the same time their generalization abilities were impaired. Due to this the discretization was performed by an expert with the goal to limit the number of intervals to the bare minimum.

4.1.3. Network Structure

For recognizing the configuration of a vehicle it is necessary to combine the Bayesian Networks of the individual configurations into a single network. The beliefs of individual configurations need to be gathered in a single node that provides the classification result – the recognized configuration. The classification node has four states, one for each configuration and a forth termed *NoConfiguration*. A vehicle is in no configuration when it is currently unaffected by the considered entities.

Assembling all configurations into a single network is necessary in order to normalize their beliefs. Otherwise the probabilities of the individual configurations could not be compared against each other. The overall network is given in Figure 4.5.

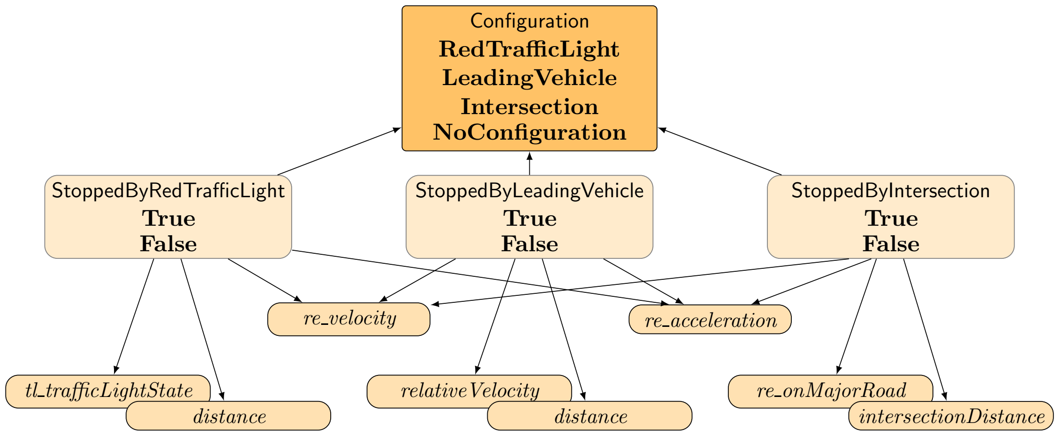


Figure 4.5.: *The Bayesian Network used for recognizing configurations. The beliefs of the individual configurations are combined by a single classification node (top).*

Since *re_velocity* and *re_acceleration* are used by all configurations they also share the corresponding nodes. This introduces additional conditional dependencies between the configurations, which turns out to be beneficial for the recognition accuracy.

Given sufficient training data, the network can learn which feature combinations lead to which configuration. An evaluation of the recognition performance of this network will be given in Section 6.1.

4.2. Incremental Situation Assessment

Decomposing complex traffic situations into configurations lowers already the computational costs of a situation assessment as compared to standard approaches. Nevertheless, recognizing the configuration of a single road user requires to measure all of the features considered in the classifying Bayesian Network. A single feature is measured by using a suitable sensor to perceive a certain part of the environment and turn the resulting sensor reading into a feature value. For example, the feature *relativeVelocity* can be obtained by identifying a leading vehicle in the point cloud provided by a laser scanner and integrating the distance over multiple steps. The example shows that a measurement results in costs like the computational costs from running an object recognition algorithm over multiple point clouds. The more configurations are considered, the higher will be the effort to compute the corresponding features up to the point where the costs become prohibitively large. To counter this, it would be advantageous if the number of considered features during a recognition, and thus the number of measurements, could be reduced to a minimum.

This reduction can be achieved by changing the recognition method from a batch process taking all features at once to a sequential method. In this case the recognition becomes an active process: Measurements are triggered according to the current hypothesis about which configurations are likely. Features that could separate between the currently most likely ones would be measured first while features of already very unlikely configurations would not be measured at all. A method for such an active measuring process is detailed in the following. It was also published in [Platho and Eggert, 2012].

4.2.1. Sensor Level and High Level

For understanding the intuition behind an active measuring process it is necessary to consider a configuration recognition system as a whole. An actual system consists of two parts as it is depicted in Figure 4.6.

In the upper part there is the Bayesian Network that was presented in the previous section. Based on a set of features the probability of each individual configuration to be present is determined. The individual probabilities are then aggregated in a single classification node on top of the network. It has to be noted that in this network the classification node is termed *hypothesis*. In the sequential recognition process described in the following the interesting probabilities in this node are not only the highest one, as it would be necessary for a pure classification task. Instead, the probabilities of all configurations are considered as they are regarded as confidence in the hypothesis that the corresponding configuration is present.

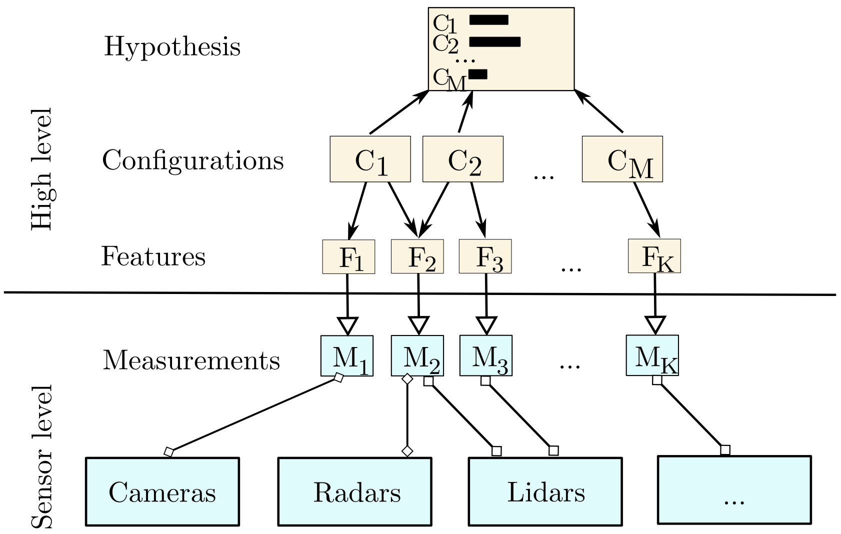


Figure 4.6.: *Schematic representation for the relation between high level and sensor level in a complete system for configuration recognition. The Bayesian Network in the high level relies on measurements taken in the sensor level.*

The lower part of Figure 4.6 depicts the sensor level. It is capable of performing different measurements for perceiving the environment. Each measurement is obtained by using one or multiple sensors like cameras, radar or laser scanners. As discussed above, each measurement is associated with some cost like blocking an exclusive resource, consuming energy or requiring computational resources for processing sensor data. Even in cases where no direct cost can be attributed to a measurement, it still takes time to wait for the measurement's data.

The connection between the Bayesian Network in the high level and the sensor level is limited to the point where sensor measurements are turned into feature values. Each feature obtains a dedicated, unique measurement. Apart from this connection both levels are completely independent. The independence becomes relevant when considering the fact that inference in the high level is comparably cheap to obtaining measurements in the sensor level. According to this, if additional computations in the Bayesian Network allow to reduce the number of sensor measurements, it will also reduce computational costs. This insight is the starting point for an active measuring approach.

4.2.2. Active Measuring

An active measuring approach takes sensor measurements sequentially, one at a time. In each step it selects that measurement for which the corresponding feature offers the highest gain. In a recognition task the highest gain is provided by that feature that is expected to maximize the probability of a single, preferably the correct, hypothesis and minimize the probabilities of all other hypotheses. In a Bayesian Network the gain of measuring an individual feature can be directly computed by means of the *expected mutual information* [Shannon and Weaver, 1949].

Given a set of hypotheses $\mathbf{H}$ comprising N hypotheses H_n, one for each considered configuration $C_n \in \{C_1, ..., C_n, ..., C_N\}$ and K features F_k, the mutual information $I(H; F_k)$ is defined as

$$I(\mathbf{H}; F_k) = \sum_n \sum_k p(H_n, f_k) log \frac{p(H_n, f_k)}{p(H_n)p(f_k)} \tag{4.1}$$

where $p(H_n)$ denotes the probability of hypothesis H_n and $p(f_k)$ denotes the probability of feature F_k having value f_k. Via inference, the probabilities can be obtained from the Bayesian Network used for recognition.

Mutual information measures how much knowing about one variable reduces the uncertainty of the other. The higher the mutual information of a feature is, the more its measurement will contribute to the beliefs of the hypotheses.

The active measuring method presented here selects one measurement after the other until the probability of a single hypothesis surpasses a predetermined threshold τ. The method aims at decreasing the set of probable configurations quickly to a single, confident one. The goal is to terminate the costly measurement process as soon as it becomes improbable that further measurements will change the most likely hypothesis anymore. The working principle can be separated in four steps:

1. *Measurement Selection*: In the Bayesian Network, compute the expected mutual information of each yet unobserved feature.

2. *Observation*: In the sensor level, trigger the measurement of the feature with the highest expected mutual information.

3. *Inference*: Perform inference incorporating the newly obtained feature. If the belief in the most probable hypothesis is below the threshold τ and not all features have been measured already, continue with step 1, otherwise continue with step 4.

4. *Result*: Return the most probable configuration as recognition result.

It is important to note that the sequence of measurements depends on the evidence obtained so far. This is why the active measuring process can not define the sequence in advance but needs to compute it online.

The threshold τ serves as parameter for trading accuracy against computational speed. For values of τ close to 1, more measurements will be triggered as the method will stop only if the system is perfectly sure about a hypothesis. In this case the system will still provide the same accuracy as the method presented in Section 4.1.

By considering only the single, most promising feature at a time, the proposed method would neglect features that contribute only slightly when measured alone but are strong in combination with each other. However, if these features belong to the same configuration this would indicate that they are strongly dependent on each other, which is not the case for the features used in the network. Nevertheless, a solution is to consider the expected joint mutual information of multiple features, although this would significantly increase the complexity of the selection process.

The degree to which computational costs can be reduced, while maintaining a high recognition accuracy, will be evaluated in Section 6.2.

4.3. Prediction of Velocity Profiles

The methods that were presented in Sections 4.1 and 4.2 can be employed for assessing traffic situations. They identify the configurations of present road users and thus provide a descriptive model of the situation. Whereas understanding the *current* situation has a value per se, in many cases it is even more important to consider the *future* situation. Predicting how the situation will change in the next few seconds is particularly helpful for planning own maneuvers or anticipating upcoming conflicts. The method that will be presented in the following takes a configuration-based situation assessment as starting point for predicting the behavior of individual vehicles. It was published in [Platho et al., 2013a].

4.3.1. Overall Prediction System

At urban intersections upcoming conflicts can in many cases only be detected by predicting the evolution of the current situation comprehensively, which means to take all possibly relevant entities into account. But for a prediction the same issues apply as for situation assessment: An all-encompassing prediction model that incorporates all entities at once will grow overly complex. Thus, a more feasible approach is to perform the prediction individually for each road user. In this case it is important to ensure that the prediction takes situational aspects into account instead of considering the regarded road user isolated from its context. Accordingly, situation assessment is an essential part of a prediction process and the proposed approach accounts for this by employing a two-staged method.

In the first stage, the situation is decomposed into configurations using one of the recognition methods that were presented in Sections 4.1 and 4.2. For each vehicle its current driving situation is determined by recognizing its configuration. The considered configurations are besides *StoppedByRedTrafficLight*, *StoppedByLeadingVehicle* and *StoppedByIntersection* also *NoConfiguration*, which denotes the case when a vehicle is currently unaffected by other entities. The first stage is depicted in step 1 of Figure 4.7.

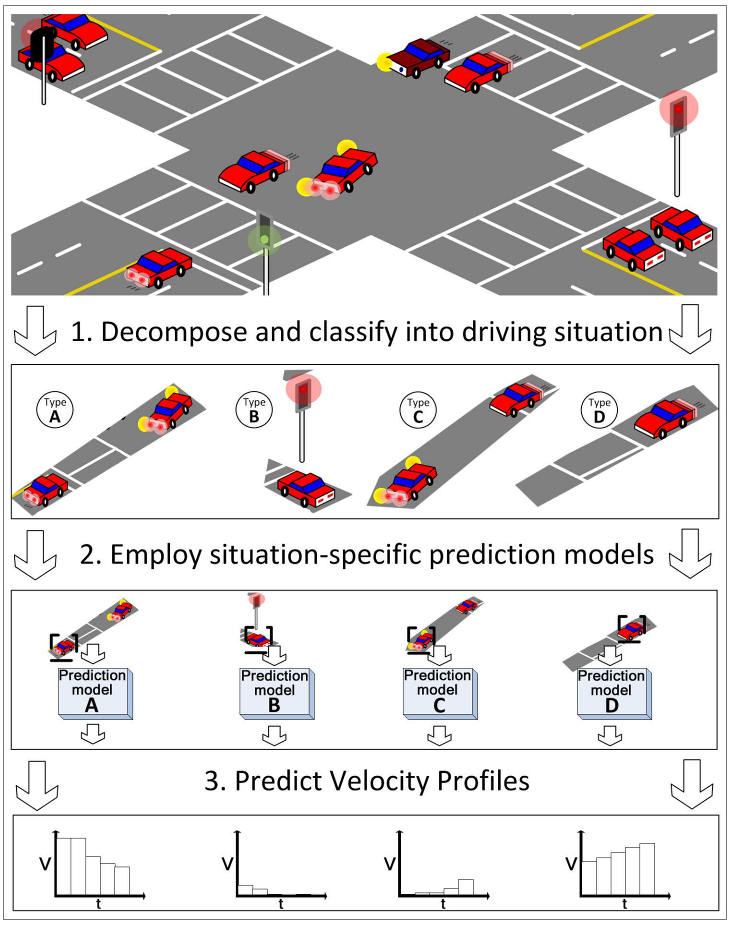

1. Decompose and classify into driving situation
Type A
Type B
Type C
Type D
2. Employ situation-specific prediction models
Prediction model A
Prediction model B
Prediction model C
Prediction model D
3. Predict Velocity Profiles
V
t

Figure 4.7.: *System overview for a situation-aware behavior prediction. After determining a vehicle's configuration (step 1), a configuration specific behavior model (step 2) is employed to predict the velocity profile of the vehicle (step 3).*

In the second stage, for each vehicle the longitudinal behavior in form of its velocity profile is predicted. For this purpose, there are 4 situation-specific prediction models used, each of them trained exclusively for one configuration. Based on a vehicle's recognized configuration in the first stage, the appropriate model is used to predict its behavior up to three seconds into the future. Using situation-specific prediction models has two advantages: First, it allows to naturally incorporate a feature selection that discards features that are irrelevant for the driving situation. This reduces the feature dimensionality for the prediction algorithm and thus reduces training effort while increasing the models robustness. The second advantage is that the prediction model is tailored to a single driving situation for which it can be assumed that vehicles show a similar behavior. The working principle of the second stage is illustrated in steps 2 and 3 of Figure 4.7.

4.3.2. Prediction Process

The system returns the predicted velocity profile of a vehicle for the next three seconds. A velocity profile is represented by a vector with 30 individual velocity values, which is obtained by sampling the velocity at 10 Hz for 3 seconds. For each of the considered configurations an individual prediction model is trained. The model takes, besides the velocity and acceleration, also features specific to the driving situation as input. Only the prediction model for *NoConfiguration* utilizes no additional features.

Altogether, there are 7 features used as independent variables for the prediction models. In addition to the velocity, later denoted as *VEL*, and the acceleration (*ACC*) of the target car, for which the prediction is performed, there are 5 more features.

- *Traffic light distance (TLD)*: Distance to the stopping line of the next, relevant traffic light in m
- *Car ahead relative speed (CAS)*: Relative velocity between target car and its leading car in m/s
- *Car ahead distance (CAD)*: Distance between target car and its leading car in m
- *Intersection distance (ID)*: Distance to the entry point of the next intersection in m
- *Time (TIME)*: Time instance for which the velocity is predicted in s. Values are $0.1, 0.2, 0.3 ... 3.0$

Each of the four prediction models take a proper subset of these features as input variables. One reason for using only subsets is that in certain configurations some features may not be specified at all. For example, when a vehicle approaches an unsignalized intersection there is no traffic light and thus no value for the distance to the traffic light *TLD*. Another, more important reason for using proper subsets is that features that are not relevant in the current driving situation only increase the dimensionality of the regression task and distract the regression algorithm from learning the effect to relevant features. Table 4.1 lists which features are used by the individual models.

Prediction model	VEL	ACC	TLD	CAS	CAD	ID	TIME
$P_{TrafficLight}$	x	x	x				x
$P_{LeadingCar}$	x	x		x	x		x
$P_{InterSection}$	x	x				x	x
$P_{NoConfiguration}$	x	x					x

Table 4.1.: *Features used by the four prediction models.*

All models are realized by a *Random Forest Regressor* (RFR) [Breiman, 2001]. An RFR is a nonlinear, state-of-the-art regression method which is known for its robustness against noise and over-fitting. Its name originates from its underlying working principle, as it aggregates a set of individually learned *Classification-and-Regression-Trees* (CART) [Breiman et al., 1984] into an ensemble, a so-called forest. In a pre-study, also *Multiple Linear Regression* was evaluated as regression method, but it was significantly excelled by RFR.

A Random Forest Regression has, depending on the implementation, multiple parameters, with two of them having the greatest impact: the depth of each tree and the size of the forest. The maximal depth can be interpreted as the degree of the dependencies among features. The number of trees is set according to computational restrictions. The larger the forest, the better is the regularization ability of the regression method, up to the limit that the given data imposes. In turn both learning and prediction time increase linearly with the number of trees. Based on the results of a preliminary evaluation, the maximal depth is set to 4 and the number of trees is set to 400.

The optimization criterion for the RFR is the sum of squared distances. It minimizes the prediction error e between the actual velocity profile V and the estimated velocity profile $\hat{V}$ for each of the 30 individual values:

$$e = \sum_{i=1}^{30} (\hat{V}_i - V_i)^2 \tag{4.2}$$

To summarize, given a traffic situation, its evolution is predicted by recognizing the current configuration of each vehicle and applying the corresponding prediction model to foresee its future velocity profile. The feasibility of this approach and the results obtained are presented in Chapter 6, Section 6.3.

4.4. Learning Configurations from Observations

The system presented in the previous section predicts a vehicles longitudinal behavior in a two-staged approach: in the first stage the configuration of the vehicle is determined in order to employ in the second stage the corresponding, configuration-specific behavior model for predicting the future velocity profile. This method requires manual work by an expert in two areas: For specifying the considered configurations and for manually annotating training data with the information which configuration is currently active. Regarding the specification of the considered configurations there are multiple advantages of letting an expert perform this task. For example, it allows to directly incorporate expert knowledge and to make configurations interpretable.

Today, however, collecting data has become so ubiquitous that even comprehensive driving data is readily available [Zecha and Rasshofer, 2009]. In cases, where large amounts of driving data can be obtained easily but manually annotating the data is intractable, it would be advantageous if the manual work could be circumvented. In the following a method is presented for learning a complete prediction system as in Section 4.3 from unlabeled data. The method was published in [Platho et al., 2013b].

4.4.1. System Overview

The goal of the learning algorithm described in the following is to learn all parameters of a system for predicting the velocity profiles of individual road users. The prediction system has the same working principle as the two-staged approach presented in Section 4.3: In the first stage, a vehicle's driving situation in terms of its current configuration is determined. In the second stage a configuration-specific prediction model is used to estimate the vehicle's future velocity profile. Both stages can be further subdivided into two steps each. Thus there are altogether four steps that the system performs in order to arrive at a prediction. The steps are sketched in Figure 4.8 and are as follows.

I. **Feature Extraction:** By taking sensory measurements, a set of features $\boldsymbol{F}$ is obtained. These features are designed to capture the current driving situation and the state of the target vehicle, for which the prediction will be performed, adequately.

II. **Configuration Recognition:** Given the measured features, the current driving situation of the vehicle is determined. Formally, the second step can be interpreted as a classifier that takes the features as inputs and returns a configuration label, i.e. $h(\boldsymbol{F}) \to C_n$ with N being the number of considered driving situations and $\boldsymbol{C} = \{C_1, .., C_n, ..., C_N\}$.

III. **Feature Selection:** Depending on the estimated driving situation a proper subset $\boldsymbol{F}'$ of the measured features $\boldsymbol{F}$ is selected. The subset selection function $s_n(\boldsymbol{F})$ leaves only these features that are found to be relevant for the

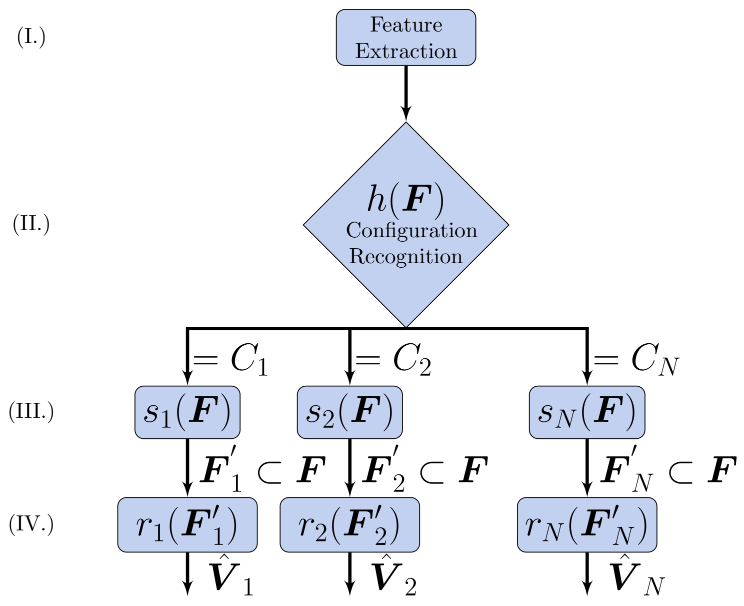


Figure 4.8.: *Architecture of a behavior prediction. Based on a set of extracted features of the target vehicle (I), its driving situation is determined, here by recognizing its configuration (II). Then a subset of the extracted features (III) is used as input for a configuration-specific prediction model, which returns a predicted velocity profile $\hat{\boldsymbol{V}}$ (IV) (see text).*

subsequent velocity prediction. This step has the goal to increase the chance of learning diverse prediction models that are highly specialized to their configuration and that work with as few features as possible. Additionally, it can be assumed that some features will not contribute to a prediction in all possible configurations.

IV. **Prediction:** A regression model r_n takes the subset $\boldsymbol{F}'_n$ as input and returns the predicted velocity profile $\hat{\boldsymbol{V}}$ for the target car.

The challenge in learning this prediction system lies in the fact that three components need to be learned which are dependent from each other. The components are the configuration recognition function h, the subset selection functions s_n and the prediction models r_n. An iterative learning algorithm is employed because for this problem no closed-form solution exists.

4.4.2. Learning Method

As starting point for the algorithm an unlabeled data set containing driving data is given. The data consists of many observations, where each observation comprises a set of features $\boldsymbol{F}$ and a velocity profile $\boldsymbol{V}$. The K features of $\boldsymbol{F}$ have been measured by sensors for a given target vehicle and its surrounding at a time t_0. The velocity profile $\boldsymbol{V}$ consists of a vector of 30 individual values that were obtained by sampling the velocity of the target vehicle between $t_0 + 0.1s$ and $t_0 + 3s$ at 10 Hz.

Target Function

The goal of the overall system is to predict the behavior of individual vehicles as accurately as possible. More formally, it aims at minimizing the error e between the actual velocity profile $\boldsymbol{V}$ and the estimated velocity profile $\hat{\boldsymbol{V}}$ for all observations i:

$$e = \sum_i ||\boldsymbol{V}_i - \hat{\boldsymbol{V}}_i||_2 \tag{4.3}$$

Incorporating the fact that each velocity profile consists of 30 individual values and representing time as $t \in \{1, .., 30\}$ this equation becomes

$$e = \sum_i \sum_t ||V_{it} - \hat{V}_i(t)||_2 \tag{4.4}$$

The estimation $\hat{V}_i(t)$ is written as a function of time to account for the use in a regression model in which time is an independent variable. Given the features, the first step for obtaining $\hat{\boldsymbol{V}}$ is to recognize the configuration using a classifier function h:

$$h(F_i) : F \to C_n \in \{C_1, ..., C_N\} \tag{4.5}$$

Unfortunately, this classifier can not be trained because only unlabeled data is available and the configuration themselves are part of the learning procedure. Therefore an approach similar to the Expectation-Maximization [Dempster et al., 1977] algorithm is taken, where observations are directly assigned to configurations using a probabilistic assignment matrix H. The assignments can be gradually adapted by the learning method and upon convergence a classifier can be trained using the assignment information as label. The assignment matrix $\boldsymbol{H}$ is of order $I \times N$ with

$$H_{in} = p(C_n|i) \text{ with } \sum_n H_{in} = 1 \tag{4.6}$$

denoting the probability of the i-th observation to belong to configuration C_n.

A predicted velocity profile $\hat{\boldsymbol{V}}$ is obtained by summing over the predictions of all N regression models r_n, each of which takes the subset of features returned by $s_n(\boldsymbol{F})$ and the time as input. The assignment matrix $\boldsymbol{H}$ weights and normalizes the individual contributions of the regression models.

$$\hat{V}_i(t) = \sum_n H_{in} r_n(s_n(F_i), t) \tag{4.7}$$

However, using a mixture of regression models is not intended. Instead each observation should be exclusively assigned to a single configuration. Achieving this goal is left to the learning algorithm which ensures that for each observation the weight of a single regression model becomes close to '1' and '0' elsewhere.

Based on the previous considerations, the final target function is

$$\min_{H,r,s} \sum_i \sum_t ||V_{it} - \sum_n H_{in} r_n(s_n(F_i), t)||_2 \tag{4.8}$$

Regression model

Each velocity profile comprises 30 individual values, one for each time step in the prediction horizon. One option for a suitable regression model is to take a non-parametric approach and predict the velocity for each time step individually. While this approach provides a high flexibility and allows parameterizing each time step separately, such a model requires a large number of parameters which are costly to learn. Furthermore, real velocity profiles favor smooth parametric models: Since vehicles are physical systems and thus subject to inertia, their velocity follows a smooth function as the change in velocity between consecutive time steps is rather moderate. It is therefore reasonable to model a profile by a smooth function. An example of such a function is

$$v_i(t) = v_{i0} + a_{i1}t + a_{i2}t^2 \tag{4.9}$$

which is determined by two parameters, a_1 and a_2 for linear and quadratic term, respectively. In order to make the representation invariant towards the initial velocity v_0, it can be rewritten as

$$v_i(t) - v_{i0} = a_{i1}t + a_{i2}t^2 \tag{4.10}$$

The parameters a_1 and a_2 are learned independently from each other via Multiple Linear Regression (MLR)

$$a_{ij} = \beta_{j0} + \beta_{j1} f_1 + ... + \beta_{jK} f_K \tag{4.11}$$

with $j \in \{1, 2\}$ and β_f denoting the regression coefficients for individual feature values $f_k \in \{f_1, ..., f_k, ..., f_K\}$. By inserting Equation 4.11 into Equation 4.10 the regression model for the n-th configuration is

$$b_n(F_i, t) = (\sum_{k=0} \beta_{1nk} f_{ik})t + (\sum_{k=0} \beta_{2nk} f_{ik})t^2 \tag{4.12}$$

with f_{i0} defined as 1 in order to serve as intercept variable for the regression. The model is uniquely defined by the vectors of regression coefficients $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$.

Although there are more powerful regression methods than MLR, it was chosen due to its computational speed and its ability to handle weights for observations. This property is important as H can be seen as a weight matrix for the observations which an MLR can directly incorporate into the estimation of its coefficients.

The feature selection function s for selecting a configuration-specific subset of features $\boldsymbol{F}'$ from the given Features $\boldsymbol{F}$ are realized by a matrix $\boldsymbol{S}$ of size $N \times K$ with

$$s_{nk} = \begin{cases} 1 & \text{if } f_{ik} \in F'_i \text{ for } C_n, \\ 0 & \text{otherwise.} \end{cases} \tag{4.13}$$

Each configuration-specific feature can only be exclusively assigned to a single configuration.

The estimated velocity profile from Equation 4.7 can thus be rewritten as

$$\hat{V}_i(t) = \sum_n H_{in} \left((\sum_{k=0} s_{nk} \beta_{1nk} f_{ik})t + (\sum_{k=0} s_{nk} \beta_{2nk} f_{ik})t^2 \right) \tag{4.14}$$

Minimization algorithm

In order to minimize the total error e between actual and estimated velocity profiles, a simultaneous adaptation of assessment matrix $\boldsymbol{H}$, feature selection matrix $\boldsymbol{S}$ and regression coefficients $\boldsymbol{\beta}$ is needed, as postulated in Equation 4.8. Because there is no analytical solution to this optimization problem, an iterative method which was inspired by the EM-algorithm is employed. It starts with a randomly initialized $\boldsymbol{H}$ and then repeats to sequentially update $\boldsymbol{\beta}$, $\boldsymbol{S}$ and $\boldsymbol{H}$, in that order.

The coefficients $\boldsymbol{\beta}$ are updated by a weighted linear regression

$$\beta_{jn} = (\mathsf{F}^T \boldsymbol{H_n} \mathsf{F})^{-1} \mathsf{F}^T \boldsymbol{H_n} \boldsymbol{a}_j \tag{4.15}$$

where $\boldsymbol{H}_n$ denotes the n-th column of matrix $\boldsymbol{H}$. F represents the matrix that results from concatenating the features $\boldsymbol{F}_i$ of all observations into a single matrix. The use of $\boldsymbol{H}$ as weight matrix ensures that the coefficients of a regression model are determined only by observations that are sufficiently probable to belong to the corresponding configuration.

With $\boldsymbol{S}$ being a binary matrix no gradient can be computed for it. The solution that was chosen here is an exhaustive search on all pairwise swaps of entries. A swap is performed by assigning a feature from the current configuration to another configuration which is realized by exchanging a '1' and a '0' in the same column. For each possible swap the change in e, Δe, is computed. The swap with the lowest Δe is selected as long as $\Delta e < 0$, that is the swap improves the overall estimation.

The entries of the assignment matrix $\boldsymbol{H}$ are updated by computing for each observation i the model error e_{in} for each of the N configurations. The lower e_{in} is, the stronger is the increase in probability that observation i belongs to configuration n. More formally:

$$p_{in} = \frac{H_{in}}{e_{in}} \tag{4.16}$$

$$w_{in} = \frac{p_{in}}{\sum_n H'_{in}} \tag{4.17}$$

$$H'_{in} = l \times w_{in} + (1 - l) \times H_{in} \tag{4.18}$$

The current probability of the i-th observation in the assignment matrix is divided by its error using the n-th regression model (Equation 4.16) and normalized (Equation 4.17). The parameter l acts as the learning rate and controls the impact of an update (Equation 4.18). It has to be noted that due to the multiplication in Equation 4.16 the rows of $\boldsymbol{H}$ converge to a vector with all zeros and a single 1, which is a desired property.

The error e is decreased in every iteration because the updates of β and $\boldsymbol{H}$ ensure that the total error decreases and the update of $\boldsymbol{S}$ can at least not increase it.

The minimization is stopped when the relative improvement between successive iterations drops below a predefined threshold τ.

Application to unseen data

When the minimization algorithm terminates all components of a prediction system have been learned, except for the classifier for recognizing configurations. As stated before, the use of unlabeled data prohibits a training of the classifier which is dependent on label information. The solution to this was to replace the classifier by an assignment matrix $\boldsymbol{H}$ that could be directly incorporated into the minimization algorithm, but looses its classification abilities. Nevertheless, after termination this assignment matrix provides the required label information: The configuration that is the most probable for an observation is assumed to be the correct one. The label l_i for the correct configuration of the i-th observation is obtained by

$$l_i = n |\ H_{in} = \max_{o} H_{io} \tag{4.19}$$

The labels are used for training a classifier. The classification algorithm can be chosen based on the number of observations and the availability of computational resources. Here a Random Forest Classifier [Breiman, 2001] is employed instead of a Bayesian Network as used so far. This is due to the fact that generally the learned configurations can not be as well separated by a linear classifier as the manually specified configurations.

The method presented in this section trades interpretability and generalizability for avoiding a time-consuming labeling. An evaluation on the degree to which learned configurations match specified configurations and the accuracy of the resulting prediction system is given in Section 6.4.

4.5. Intention Estimation

The systems presented in Sections 4.3 and 4.4 provide a prediction *how* a road user will execute its next maneuver by anticipating the longitudinal behavior. In many cases it is more interesting to predict *which* maneuver the driver will execute next. As it was already discussed in Section 2.1.2, this task is termed *intention estimation* and aims at predicting the next maneuver of a road user out of multiple possibilities. Research in this area has attracted increasing interest in recent years and considerable progress has been made. However, in an extensive review of related work (see Subsection 2.1.2) no method was found that considered more than two intentions for an estimation. Furthermore, the methods make only limited use of situational cues but focus mainly on cues from a vehicle's behavior. But taking behavior as sole cue neglects the fact that a driver selects its behavior not solely according to its intentions but also according to its driving situation. The preceding considerations motivated the development of a method for intention estimation that takes situational cues explicitly into account in order to distinguish between more than two intentions. One central goal in the development process was to build a system that implements the new method and demonstrates the feasibility of the approach. This intention estimation system is presented in the following. It aims at predicting the intended maneuver of the ego-vehicle when it approaches a signaled intersection. A paper about the system was submitted to [IV2014].

4.5.1. System Overview

The goal of the system is to anticipate the intended maneuver of the ego-vehicle when it approaches a signaled intersection. Four different intentions are considered:

I. Go straight

II. Turn right

III. Stop at red traffic light

IV. Car following

Current literature focuses on intentions *I* and *II* as a correct estimation allows to anticipate trajectories that conflict with other road users. Additionally, these intentions can be generally well distinguished based on a vehicle's velocity profile, as a driver with the intent to turn will slow down the vehicle early and a driver with the intent to go straight will maintain the current velocity. Significantly fewer approaches consider intention *III*, although it is highly relevant e.g. to anticipate red-light running. The intention is harder to detect as its corresponding decelerating behavior can be mistaken for a right turn intention which is why it profits from situational cues. Intention *IV*, *'Car following'*, considers the case when the behavior of the considered vehicle is dominated by the behavior of the leading vehicle. In this case the following vehicle is forced to slow down or stop in order to keep a sufficient distance to the vehicle ahead. This intention is equivalent to a *StoppedByLeadingVehicle*-configuration and its consideration is especially beneficial: it has been shown in [Liebner et al., 2012] that a close preceding vehicle impairs a behavior-based intention estimation considerably. Making *'Car following'* an explicit intention allows to detect it and pass this information to subsequent systems. The intention can hardly be detected by a vehicle's behavior itself, instead situational cues need to be taken into account.

A system capable of anticipating the four intentions correctly, one to two seconds before the target vehicle reaches the intersection, can be used for multiple purposes. An already mentioned purpose is its use as warning and prevention system for red-light running. Additionally, subsequent assistance systems can utilize the intention estimation as input, as Liebner pointed out: a system for preventing collisions with bicyclists during turning maneuvers could then only issue a warning if the driver actually intends to turn right. Otherwise the system would warn the driver at every intersection whenever a bicyclist is nearby, even if the driver plans to go straight. Eventually, the driver would switch off the warning system and thus loose its benefit.

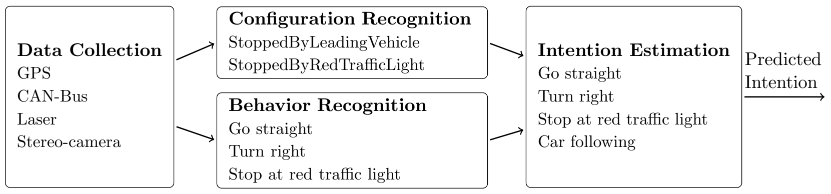


Figure 4.9.: *Schematic representation of the system for intention estimation. In order to distinguish between four intentions, both the behavior and its configuration are considered.*

In order to distinguish between all four intentions a system needs to consider both a vehicle's behavior and its driving situation. This insight lead to a system architecture as sketched in Figure 4.9.

In this system two separate models are learned, one for a behavior-based estimation and another for a situation-based estimation. The output of both models is then combined in a single one to compute the final estimation. Before detailing the individual system components the sensor equipment of the test vehicle and the data acquisition procedure are presented in order to make certain design decisions better understandable.

4.5.2. Data Acquisition

Data generated by a traffic simulator is naturally biased as compared to data acquired from real-world sensors. Since the system should work on real-world data, the characteristics of real measurements were regarded throughout the development process. In order to understand all of the design decisions made, the sensor equipment and the acquisition procedure are detailed in this subsection.

A dedicated test vehicle was used to record approaches to urban intersections. The vehicle was equipped with an IBEO laser scanner operating at 100 Hz. It faces forwards and is able to detect objects directly in front and to determine their distance as well as their relative velocity. The filtering and object detection is therefore performed by the scanner. A leading vehicle is detected by taking the closest object in a 1 m corridor in front of the vehicle, if any. A consumer-grade GPS provides the absolute position of the vehicle at 10 Hz. The velocity of the ego-vehicle is obtained directly from its CAN-bus. A forward-facing stereo camera filmed the drives.

In order to increase the variability of the data, the test drives were performed by two different drivers approaching seven different urban intersections. During the test drives 37 intersection approaches were recorded.

The recorded data was further processed. The acceleration of the ego-vehicle is computed by deriving the recorded velocity. Since the derivation increases any measurement noise present in the velocity, the acceleration oscillates heavily. This is countered by smoothing the acceleration using a moving average over the last 20 measurements.

The GPS positions of all intersections' stop lines are obtained from the vehicle's GPS position when it waits as first vehicle at a red traffic light. Due to the lack of a traffic light recognition system the state of a traffic light is annotated manually.

Images from the dataset can be found in Appendix B.

4.5.3. Configuration Recognition

The configuration recognition is realized similar to the method presented in Section 4.1, but without a separate classification node as the classification takes place in the final, combining component of the system.

Since only intentions *III* and *IV* require situational cues, there are only the corresponding two configurations considered:*StoppedByRedTrafficLight* and *StoppedByLeadingVehicle*, where the former is specified as described in Section 4.1 while the latter is augmented by two additional features. The feature *carAheadTTC* represents the Time-To-Collision to the leading vehicle and the feature *netTimeGap* represents the time that passes until the following vehicle reaches the position of the leading vehicle. These features turned out to improve the recognition accuracy of the *StoppedByLeadingVehicle*-configuration. The Bayesian Network including all states is given in Figure 4.10.

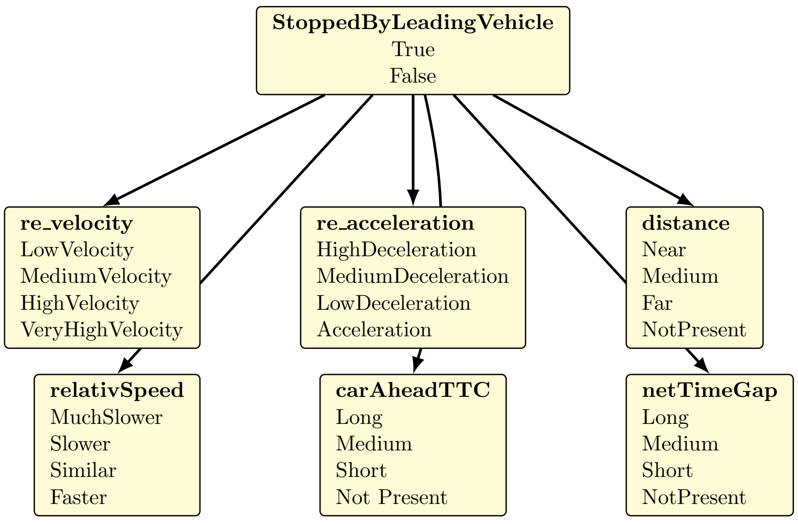


Figure 4.10.: *The Bayesian Network used for recognizing a* StoppedByLeadingVehicle-*intention including all states.*

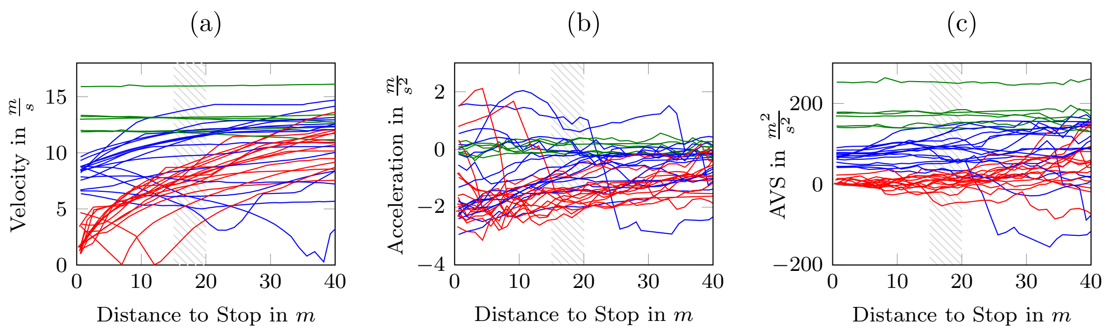


Figure 4.11.: *Velocity profiles (a), acceleration (b) profiles and AVS feature (c) for 37 intersection approaches. The red lines show the velocity profiles for a red traffic light, green corresponds to straight intersection crossings and blue to right turn intersection crossings. The shaded gray area indicates the interesting range of* 15 *to* 20 *meters, where an estimation leaves sufficient time for a reaction if necessary.*

4.5.4. Feature Selection

Since Intention *IV* can not be recognized based on a vehicle's behavior, only intentions *I-III* are considered for the behavior-based estimation. Before an estimation model can be built, appropriate features need to be selected or engineered.

One feature that is widely accepted in the state of the art [Liebner et al., 2012, von Eichhorn et al., 2013] is the velocity. It is clear that a driver will slow down the vehicle for an intended turn or a red traffic light whereas the driver will maintain the current velocity for a straight crossing. These characteristics can be found in our data as well, as it is shown in Figure 4.11a. However, in the shaded region at a point 15 to 20 meters away from the stop line, a clear separation is not possible.

Intuitively, the acceleration is also a useful feature. Bringing a vehicle to a complete stop at a red traffic light requires a stronger deceleration than for a turning maneuver and even less so for a straight crossing. But the acceleration is a very noisy feature that needs to be heavily filtered. The filtering stabilizes the values but still the acceleration is a weaker feature than the velocity as it can be seen in Figure 4.11b.

Even when combining both features a satisfactory separation is not possible. A solution is to engineer a stronger feature out of the existing. A close look on the velocity profiles shows that the intentions are linearly separable when the vehicle is close to the stop line. Another observation is that the last 20 meters of a velocity profile can be approximated by a straight line. Combining both observations leads to the assumption that the velocity at the stop line can be predicted already

20 meters before and that this prediction can be used to distinguish between the individual intentions. The corresponding feature will be tagged *Anticipated Velocity at Stop Line*(AVS).

The AVS feature results from predicting a vehicle's velocity for the moment it reaches the stop line. The prediction is made from the current distance d, velocity v and acceleration a of the considered vehicle; it thus combines all kinematic information in a single value. AVS is obtained by a straightforward extrapolation of the vehicle dynamics:

$$v_s = v + at_s, \tag{4.20}$$

with v_s representing the velocity at the stop line and t_s the expected time for reaching the stop line. The value of t_s can be obtained from the equation of motion

$$x_s = \frac{1}{2}at_s^2 + vt_s + x, \tag{4.21}$$

for $x_s = 0$ and $x = -d$. When solving in Equation 4.21 for t_s, two possible solutions for t_s are obtained:

$$t_{s,1} = \frac{-v + \sqrt{v^2 + 2da}}{a} \tag{4.22}$$

$$t_{s,2} = \frac{-v - \sqrt{v^2 + 2da}}{a}. \tag{4.23}$$

The next step is to decide which solution to take. For realistic intersection approaches both v and d are positive. If $a > 0$, then $t_{s,2}$ becomes negative whereas $t_{s,1}$ is positive. For $a < 0$ also $t_{s,1}$ is the appropriate choice, but this is less obvious so it will be explained in the following paragraph.

For $a < 0$ both $t_{s,1}$ and $t_{s,2}$ are positive as the stop line is passed twice. A universal decision whether $t_{s,1}$ or $t_{s,2}$ provide the first pass can be made by inspecting the equations more closely. For real-numbered solutions and $v > 0$, $d > 0$ and $a < 0$ the inequality

$$\sqrt{v^2 + 2da} < v.$$

holds true. In combination with Equations 4.22 and 4.23 it can be inferred that $t_{s,1} < t_{s,2}$. Thus, $t_{s,2}$ represents the case that a vehicle continues to decelerate after passing the stop line until it eventually reverses and crosses the line again. Since this point in time is not of interest it shows that only $t_{s,1}$ needs to be considered. Inserting Equation 4.22 into 4.20 results in

$$v_s = \sqrt{v^2 + 2da}. \tag{4.24}$$

In cases where $2da < -v^2$ a vehicles deceleration is high enough that the vehicle will never reach the stop line. In these cases v_s is not real-numbered and also $t_{s,1}, t_{s,2} \in \mathbb{C}$. In order to still obtain a real-numbered result and a fully continuous function for all possible input values $d, v \in \mathbb{R}_+$ and $a \in \mathbb{R}$, v_s is squared. The novel AVS feature is thus defined as

$$AVS = v^2 + 2da.$$

The values of the AVS feature for the considered intersection approaches are shown in Figure 4.11c. Obviously, the individual behaviors can be separated significantly better than for the other features.

4.5.5. Behavior-based Intention Estimation

One method that was recently proposed for behavior-based intention estimation are *Gaussian Processes* [Tran and Firl, 2013, Armand et al., 2013]. One advantage is that they provide a probabilistic output; assigning each intention a likelihood of its presence. This way the estimation can be naturally combined with the Bayesian Networks for the configuration recognition. However, a first evaluation using Gaussian Processes revealed that they are not suitable for the given data. The problem is that for each intention the individual approaches, regardless of the considered feature, show a high variability and thus a high variance. But Gaussian Processes assume that variance results only from measurement noise and thus struggle with representing the variability adequately.

An alternative method that also provides a probabilistic output is *Logistic Regression*. Despite its name, Logistic Regression is a linear classifier. It is known for its computational efficiency for both learning and classification and its high accuracy. For a set of K features $f_1, ..., f_K$ and $K+1$ regression coefficients $\beta_0, ...\beta_K$, it arrives at a prediction hypothesis h for a binary classification task by

$$h(\boldsymbol{f}) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 f_1 + ... + \beta_K f_K)}} \tag{4.25}$$

Since h returns values in the interval $[0, 1]$, its output can be interpreted as probability. For a classification, values above 0.5 are interpreted as class 1 and values below or equal to 0.5 are interpreted as class 0. For tasks with more than two classes multiple Logistic Regressions are trained and combined into a single output per class.

During training, the regression coefficients are learned using gradient descent with a parameter C controlling the strength of the regularization. The regularization term is used to prevent overfitting [Tsuruoka et al., 2009]. In the system, C is set to its typical value of 1.

While a first evaluation using only the distance and AVS as features already provided accurate estimations, adding the velocity increased the accuracy further. The acceleration is not used as feature, as its incorporation reduced the performance of the estimation slightly.

Given the features, the Logistic Regression returns three values P_I, P_{II} and P_{III}, denoting the probability of intentions *I*, *II* and *III*, respectively.

4.5.6. Overall System

The overall system is a Bayesian Network related to the one used for configuration recognition in Section 4.1. It is depicted in Figure 4.12.

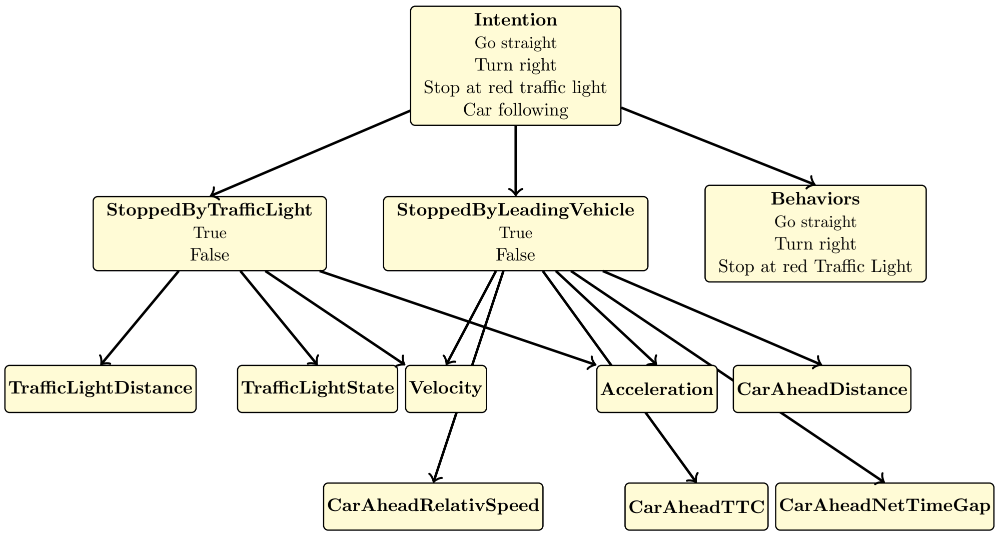


Figure 4.12.: *The Bayesian Network aggregates the beliefs of the behavior-based intention estimation and of the configurations. It provides the probability of each of the four considered intentions in its node* Intention.

A classification node, tagged *Intention*, combines the evidence of both considered configurations. Additionally, the probabilities P_I, P_{II} and P_{III} from the behavior-based intention estimation are fed into a dedicated node on the same level as the configurations; it is tagged *Behaviors*.

The system is designed to run continuously during the time the ego-vehicle approaches an intersection. It steadily computes the vehicle's Time-To-Intersection (TTI) by dividing the current distance by the current velocity. As soon as the

TTI falls below a given threshold, in this case 1.5 seconds, all features are entered into the Bayesian Network given in Figure 4.12. The intention with the highest probability is returned as estimation.

An evaluation of the system and the newly engineered AVS-feature will be given in Section 6.5.

4.6. Discussion

In this chapter methods that make use of the concept of configurations were presented. In Section 4.1 a method for recognizing configurations using Bayesian Networks was detailed that can be used to assess complex situations by identifying the current configuration of each individual road user. The method was extended in Section 4.2 with the goal to reduce the computational costs for a recognition by serializing the recognition process and limiting the number of sensor measurements for a recognition to a minimum. Both recognition algorithms serve as a basis for the methods presented in the remaining parts of the chapter.

In Section 4.3 an approach for predicting the longitudinal behavior of vehicles was discussed that uses configuration-specific prediction models which can be, due to their specialization, simpler and more tailored to the task than a single, general prediction model. A similar system was presented in Section 4.4, but with the goal to avoid any manual annotation of data or configurations. The system trades the interpretability and generalizability of expert-crafted configurations for the ability to realize a constrained application on unlabeled data.

Section 4.5 detailed a targeted real-world application of configurations in a demonstrator system for predicting a driver's intended maneuver. It aims at confirming the suitability of configurations for Advanced Driver Assistance Systems. Additionally, it was chosen to test the claim that configurations can provide useful situation information and that this information can be used to bring the state-of-the-art forward. This is why it tackles a more difficult problem in intention estimation than the published work by considering more than two intentions.

The methods that were presented herein will be extensively evaluated in Chapter 6. Since for many evaluations also driving data obtained by a self-developed traffic simulator is used, this simulator will be presented before, in Chapter 5.

5. CarD Simulation Framework

In this chapter a traffic simulation framework tagged *CarD* is presented that was developed in the course of the PhD project. The motivation was to create a tool for generating sufficient and suitable driving data that can be used to benchmark newly developed methods. While it is is undoubted that data obtained from recordings in the real-world enables more resilient results, its acquisition is a costly and time-consuming process. Furthermore, as long as the goal is not to parameterize an actual system but to evaluate the feasibility of methods and algorithms, a simulator's major advantage is that exactly these types of situations can be generated for which a method should be tested.

CarD is able to simulate urban intersection scenarios with vehicles that interact, adhere to traffic lights and respect right-of-way rules. The vehicles are not controlled by a central instance but each driver decides individually according to its own goals. *CarD* was used to generate data labeled with ground truth information, e.g. the actual configuration of vehicles, and this data is used for some of the evaluations presented in Chapter 6.

The remainder of this chapter is structured as follows. In Section 5.1 the motivation for creating a traffic simulator is given. A review of available simulation frameworks regarding their suitability for data generation reveals that none of these tools provides the required capabilities. Section 5.2 outlines the design goals of *CarD* and presents the software architecture chosen for reaching these goals. One central design goal is to create traffic by the aggregated behavior of autonomously acting drivers. Each driver behaves according to a model which is the subject of Section 5.3. The implementation of *CarD* is sketched in Section 5.4. This chapter concludes with a summary and a discussion of *CarD*'s capabilities for research on ADAS in 5.5.

5.1. Motivation

After a new method for a given problem has been developed, its suitability for solving the problem and its benefit over existing methods needs to be determined. In some cases this benefit can be assessed analytically, for example by considering convergence properties or proving optimality. In other cases, and especially for a complex real-world application like an ADAS, the benefit can most adequately be assessed by evaluating a method online during a test drive, as part of the system for which it was developed. For comparing multiple methods against each other, however, an online evaluation is unsuitable, as each test drive will differ considerably. The solution is to record test drives by logging all incurring data and then to play this data back for offline evaluations.

A major downside of real-world recordings is the associated cost for obtaining them. A data set that is large and variable enough to make valid statements about the benefit of the evaluated methods can require tenths or hundreds of hours of recording. Furthermore, there is no guarantee that during the recordings all the situations are encountered on which a method should be tested. While it is theoretically possible to reenact situations of interest this approach becomes extremely costly for urban intersection scenarios with dozens of road users.

A traffic simulator being able to generate and record arbitrary situations is especially for urban scenarios a more affordable and quicker alternative. The downside of a traffic simulator is in turn that the data recorded by it is less realistic. The data will be subject to bias resulting from simplifying assumptions made in the simulator's models for generating sensor measurements and the behavior of road users. Nevertheless, as long as the results are not used to parameterize actual systems but to compare a set of given methods for a specific purpose, a simulator's data can provide resilient results. The preceding considerations motivated the use of a traffic simulator.

Traffic simulators can be coarsely divided into two groups: *macroscopic* traffic simulators and *microscopic traffic simulators*. Macroscopic traffic simulators model traffic on an aggregated level, measuring traffic flow and traffic density in the road network in order to identify the potential of jams or study the effects of congestion. Opposed to that, microscopic traffic simulators model traffic on entity level. Each road user is controlled by a more complex behavior model that reacts appropriately to other road users. For example, cars stop for pedestrians or yield to prioritized public transport vehicles. As a tool for generating suitable data for an evaluation only microscopic traffic simulators come into consideration.

Commercially available microscopic traffic simulators like *VISSIM* [PTV, 2014] or *Paramics* [Quadstone, 2014] claim that they implement realistic, extensively validated models for their simulated road users. However, in none of these programs it is possible to access these behavior models in order to obtain the reasons for a specific behavior. But this information becomes relevant for evaluating configuration-based methods: Does the driver stop because of a red traffic light? Does the driver slow down because of a vehicle ahead? Another shortcoming of the available traffic simulators is that they lack the freedom to generate arbitrary traffic situations but are limited to rudimentary setting options. These insights lead to the development of *CarD*.

5.2. Design Concept

The name *CarD* is a short form of *Car Director* and as such a design goal was to create a framework for setting up arbitrary situations. And comparable to a movie director a user should only set the infrastructure and conditions while any actions are in the responsibility of the simulation.

Traffic emerges from the behavior of individual road users, each of them acting autonomously for reaching a desired destination. In order to account for this property, the simulator models each road user as intelligent agent [Russell and Norvig, 2003]. An intelligent agent is an autonomous entity which perceives its environment using sensors, plans its actions for reaching given goals and acts accordingly. But even when each road user plans and acts autonomously, a coordinating instance is needed for the perception part. Whenever an agent senses its environment it requests information about the world it is in. The state of the world needs to be in a central place, especially for allowing agents to sense each other. The coordinating instance of *CarD* is tagged *SceneManager* and its relation to an agent is depicted in the component diagram given in Figure 5.1

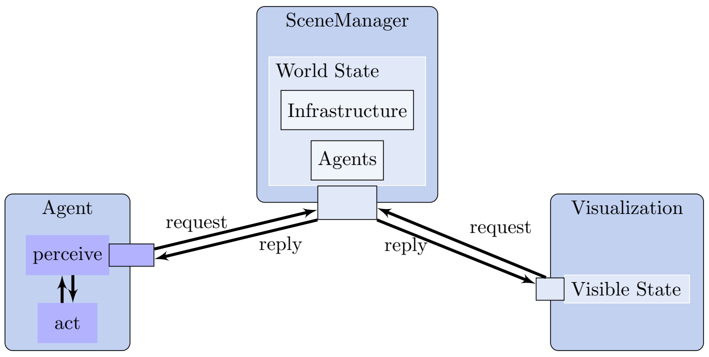


Figure 5.1.: *The basic architecture of* CarD*, the simulation framework created for data generation. A central managing instance, the* SceneManager*, provides information about the simulated world for both the visualization module and individual agents.*

The world state comprises information about both infrastructural conditions and agents. Infrastructural conditions are information about the layout of the road network, the positions of traffic lights including their assignment to individual lanes or whether a road is a minor or major road.

The perception of an agent is realized by a message exchange pattern, in which the perception module requests information about the environment. The *SceneManager*, having all information about the world at its disposal, provides the requested information in its reply. A central design decision was to make the *SceneManager* agnostic to the sensory capabilities of road users. It provides any information that was requested even if that includes entities that are beyond the line-of-sight and by no means detectable for the requester. Perceptional particularities need to be implemented by the agent itself. The advantage of this method is that the simulator framework can be extended more easily as adding an agent with a modified sensor equipment does not require any changes in the *SceneManager*.

Expandability played also an important role for the decision to realize the visualization in a separate module, as it is shown in Figure 5.1. The module requests the world state from the *SceneManager* which replies by delivering mainly position and orientation information about all agents and infrastructural entities in the simulation. Since the reply is limited to basic physical properties and no visual information about the agents is given, all of the visualization logic as well as images, 3D models or textures need to be implemented in the visualization model. This allows for modifying or switching the visualization independently from the *SceneManager*. A screenshot of a 3D visualization is given in Figure 5.2.

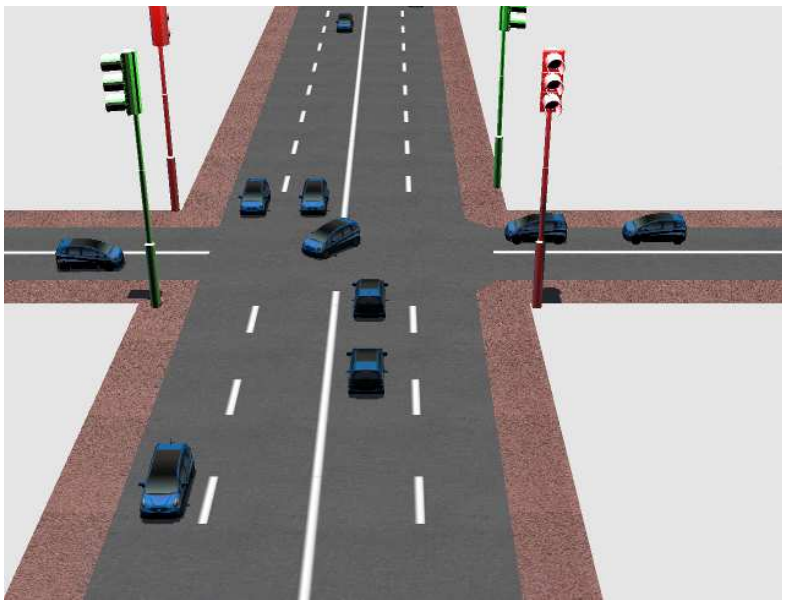


Figure 5.2.: *A screenshot of a* CarD *simulation run using a 3D visualization.*

5.3. Behavior Models

Each vehicle is controlled by an autonomously acting agent. At creation time, the agent obtains a route and a destination, which the agent tries to follow. The agent's lateral behavior is realized by a controller for ensuring that a vehicle stays inside the bounds of its current lane. The longitudinal behavior is determined by four dedicated behavior models. The agent selects the appropriate model based on the current driving situation. These four models are

1. Car following
2. Adhere to traffic light
3. Negotiate an intersection
4. Free driving

and their working principle is given below.

5.3.1. Car Following

Car following is a behavior that enables an agent to react to a vehicle ahead; otherwise vehicles in the simulation would be rear-ending all the time. The car following model implemented in *CarD* is the linear model that was proposed by Helly [Helly, 1961]. It was also used for the *SITRA-B+* microscopic traffic simulator [Gabard and Breheret, 1999] and is known for matching natural driving behavior well [Panwai and Dia, 2005].

The model adapts the acceleration of a vehicle in order to maintain a safe distance to a leading vehicle. Formally, taken from [Panwai and Dia, 2005]:

$$a_c[k+1] = C_1 \Delta v[k-T] + C_2 \Delta x[k-T] - D[k+1] \tag{5.1}$$

$$D[k+1] = \alpha + \beta v[k-T] + \gamma a[k-T] \tag{5.2}$$

where

$a_c[k+1]$	is the acceleration of the regarded vehicle at time step $k+1$
$D[k+1]$	is the desired following distance to the nearest leading vehicle in front of it
v	is the speed of the vehicle
Δx	is the relative distance between the regarded vehicle and the leading vehicle
Δv	is the relative speed between the regarded vehicle and the leading vehicle
T	is the driver reaction time
$\alpha, \beta, \gamma, C_1, C_2$	are (vehicle-specific) calibration constants

An advantage of the model is that its calibration constants are interpretable which simplifies setting them correctly. For example, α denotes the desired minimum distance a driver wants to keep to a leading vehicle. Additionally, the constants can be used to implement various driving styles from aggressive to cautious. *CarD* sets these values by drawing randomly from reasonable intervals such that each agent is different from the other, thus adding variability into the simulation to make it more realistic.

5.3.2. Adhere To Traffic Lights

This behavior makes an agent aware of traffic lights. When a traffic light associated to an agent's lane and within perception range ρ_t turns red, the vehicle is steadily slowed down until it is brought to a stop at the stopping line.

$$a_t[k+1] = -\frac{1}{2}\frac{v[k]^2}{\Delta x_s[k]} \text{ if } \Delta x_s[k] < \rho_t \tag{5.3}$$

where Δx_s is the relative distance between the regarded vehicle and the nearest stopping line on its lane and direction.

Since in the real world a car has a limited braking force, the same is implemented in *CarD*: When a traffic light turns red but the required deceleration surpasses the maximum possible deceleration of a car it runs the red light.

5.3.3. Negotiate an Intersection

This behavior is the most complex behavior of an agent. It controls how an agent crosses an intersection. Assuming there is no vehicle blocking the way – which would be handled by the car following behavior – the agent crosses the intersection if it is driving on a road with right of way or if it is able to traverse conflict-free.

Determining whether the agent will conflict with other road users during the traversal is a complex process. It starts with identifying areas at intersections where two lanes intersect each other, which are tagged *conflict zones* in the following. An example of a conflict zone for a left turn on a T-intersection is depicted in Figure 5.3.

The size of a conflict zone is given by the dimensions of the agent's vehicle in a way that only when entering the conflict zone traffic on the crossing lane will be affected. For deciding when a vehicle is able to safely pass a conflict zone the agent has to compute two aspects: the duration l_a that passing through the conflict area will take and a prediction of the intervals during which the conflict zone is unoccupied. The prediction is obtained by extrapolating the current kinematic properties of oncoming vehicles, namely acceleration and velocity, for determining

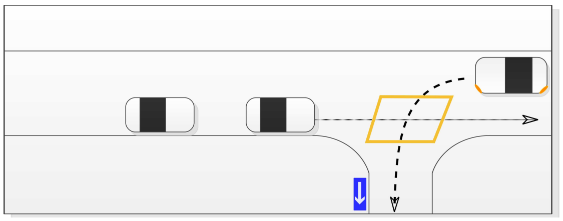


Figure 5.3.: *The conflict zone for a left turn maneuver. It is centered around the intersecting point and has the dimensions of the area covered by the crossing vehicle.*

when the conflict zone is entered and left. For the situation given in Figure 5.3 the time gap profile could look like it is illustrated in Figure 5.4.

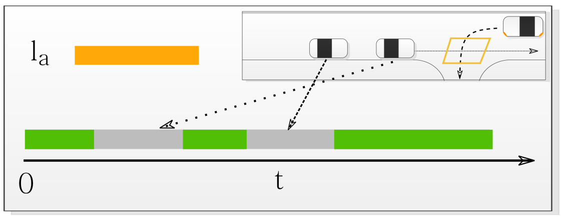


Figure 5.4.: *Intervals in which the conflict zone is free are drawn in green on the time bar. The duration l_a that the turning vehicle takes to pass the conflict zone is so long that it has to wait until the vehicles on the crossing lane have passed by.*

In this case the agent of the turning vehicle will wait until both vehicles have passed by in order to have sufficient time for the turning. In the implementation the agents also add a safety margin to l_a, which varies between agents to create both risky and defensive drivers. The deceleration for reaching the beginning of the conflict zone x_c is therefore computed as in Equation 5.3:

$$a_i[k+1] = -\frac{1}{2}\frac{v[k]^2}{\Delta x_c[k]} \tag{5.4}$$

Here, Δx_c denotes the distance to the beginning of the conflict zone. As soon as the agent enters the conflict zone, it switches to a car-following or free driving behavior, depending on whether there is a vehicle in the lane at the end of the conflict zone.

5.3.4. Free Driving

Free driving is the most simple behavior. An agent accelerates according to its desired maximum acceleration a_{max} until it reaches the speed limit for the road it is driving on.

5.3.5. Behavior selection

With four different behaviors at its disposal, an agent needs to decide which one to choose when. This is achieved by computing the deceleration proposed by each of the four behaviors. In cases where there is no vehicle,traffic light or intersection in perception range the corresponding behaviors are not considered. The behavior with the highest deceleration is selected. This ensures that the agent always reacts to the most critical entity.

The behavior selection is also useful for creating labeled data. Each of the four behaviors can be mapped to one of the configurations presented in Chapter 3. For example, whenever an agent chooses the car following behavior, in order to react to a slow driving vehicle ahead, the simulator logs this as a *StoppedByLeadingVehicle*-configuration. Labeled driving data created in this fashion will be used in the evaluation presented in Chapter 6.

5.3.6. Variability

For a realistic simulation it is desired that various traffic situations emerge from the joint behavior of individual agents. Such a variability in the generated driving data can be best achieved by making the agents variable themselves. The intuition is that in real traffic each road user has its own driving style. This is why the driving style is varied between agents by a set of parameters. For each agent the parameters are chosen by sampling uniformly from the intervals given in Table 5.1.

The values were selected either according to values recommended in the literature or are the result of adjustments when the behavior of agents seemed unrealistic. The parameters allow to model driving styles ranging from risky and aggressive to cautious and defensive.

5.4. Implementation

CarD is implemented in the scripting language *Python*, version 2.66. The visualization is realized using external libraries; for the 2D visualization a Python-binding to wxWidgets named *wxPython* [WxPython, 2014] is used and for the 3D visualization the open-source library *Panda3D* [Panda3D, 2014] is employed. The 3D visualization allows to take arbitrary view points in the simulation, for example to take the view of a driver. The visualization module runs in its own, separate process in order to keep its impact on the simulation speed minimal.

Agents inherit from a base class named ***SceneElement*** which provides them with two methods: ***update(delta_time)*** and ***draw()***. For a simulation the *SceneManager* runs a loop in which in each iteration for all registered agents first their ***update()*** and then their ***draw()*** method is called. The parameter ***delta_time***

Parameter	Description	Interval
alpha	Desired minimum distance to a leading vehicle (m)	[1,2]
beta	Factor for the distance to a leading vehicle depending on the velocity	[0.8,1.2]
C_1	Weight of the relative velocity during car-following	[2.5,3.5]
C_2	Weight of the desired distance during car-following	[1.7,2.3]
rho_t	Perception range for traffic lights (m)	[50,60]
rho_i	Perception range for intersections (m)	[40,50]
a_{max}	Maximum acceleration during free driving (m/s^2)	[2.5,3.5]
t_s	Safety margin for computing l_a during crossing a conflict zone (s)	[0.5,2]

Table 5.1.: *Parameters for specifying an agent's driving style.*

gives the milliseconds that have passed since the last iteration. This information allows to run a simulation in real time and is also relevant to determine how far a vehicle can be moved since the last iteration.

When an agent's ***update()*** method is called, it starts with perceiving the environment by querying the *SceneManager*. The agent requests within its perception range:

- distance to a leading vehicle Δx
- relative speed to a leading vehicle Δv
- distance to the stop line for the next relevant traffic light Δx_s
- state of the next relevant traffic light
- distance to the next intersection
- positions of vehicles at the next intersection
- kinematic state of vehicles at the next intersection
- the right-of-way for the next intersection

Based on the obtained information the agent selects a behavior as described in Section 5.3. This gives the longitudinal behavior of the agent, i.e. the desired acceleration, and is fed into the lateral control loop. This control loop uses a linearized bicycle model for keeping the vehicle in its lane by giving steering commands.

In order to guarantee a stable simulation the ***update*** method is called at least at 20 Hz. That means, if between two iterations of the *SceneManager*'s loop pass more than 50 milliseconds, then the ***update*** method is called multiple times in a row with a ***delta_time*** of at most 50 milliseconds. Otherwise the lateral control loop could exhibit unwanted oscillations.

The ***draw()*** method returns a representation of the current state of an agent to the caller. The representation is limited to those aspects that are relevant to a visualization. This includes world position and orientation as well as states, like the one of a turn-indicator. The *SceneManager* sends all representations via a pipe to the visualization module.

After the calls to ***update()*** and ***draw()*** the *SceneManager* writes the new world state to a log file. This log file can then be used to extract features and configuration information for each vehicle in the simulation at any given time.

5.5. Discussion

In this chapter the microscopic traffic simulation framework *CarD* was presented. Its purpose is to create driving data that can be used for evaluating some of the methods presented in Chapter 4. One major benefit of using *CarD* over commercial simulators is that it provides information about the current configuration of a vehicle.

In order to make the traffic simulated by *CarD* more realistic, vehicles are not centrally controlled by a single instance, but each vehicle is controlled by an intelligent agent. The agent tries to reach a given destination and on its way it constantly evaluates which entity it should react to and chooses an appropriate behavior. Based on the selected behavior the agent's configuration can be determined and logged.

Using a simulator for an evaluation of course evokes the question whether the results obtained can be generalized to the real world. It is obvious that this generalizability is hard to prove and the answer will particularly depend on the use case. Nevertheless, throughout the design of *CarD* achieving a sufficient level of realism has been targeted, for example by decentralizing traffic and by using behavior models that have proven to mimic driver behavior. Additionally, in Chapter 6 it will be shown that some methods that build on insights gained from using the simulator, work well on data obtained from real test drives. This may be still not a proof of *CarD*'s realism, but it is at least an indication of its usefulness.

6. Evaluation and Results

In the following the methods described in Chapter 4 are evaluated. For this purpose both simulated data obtained from *CarD* as well as driving data obtained from test drives is used.

Sections are named identically to those in Chapter 4 in order to make their relatedness explicit. In Section 6.1 the proposed method for recognizing configurations is tested and in Section 6.2 its extension to a resource saving recognition process is evaluated. Section 6.3 benchmarks the accuracy of the presented prediction system for velocity profiles. In Section 6.4 it is investigated whether such a prediction system can be learned from unlabeled data. Section 6.5 presents the results obtained on the configuration-based intention estimation system. A summary of this chapter and a discussion of the found benefit of using configurations is given in Section 6.6.

6.1. Recognizing Configurations in Complex Traffic Scenes

The method that is evaluated here was detailed in Section 4.1. It uses a Bayesian Network as shown in Figure 6.1 to recognize whether a given vehicle is in one of three possible configurations or in no configuration at all. The considered configurations are *StoppedByRedTrafficLight* (TL), *StoppedByLeadingVehicle* (LV) and *StoppedByIntersection* (IS).

6.1.1. Evaluation Method

For data generation the simulation framework *CarD* is used. An intersection scenario is set up consisting of a major road with two lanes in each direction and a crossing minor road with a single lane in each direction. The intersection is signalized with traffic lights. Cars approach from all incoming lanes; on average about 15 vehicles are nearby the intersection. A top view on the intersection is given in Figure 6.2.

The simulation was run for about 20 minutes. Every 0.1 seconds the state, behavior, features and configuration of each present vehicle is recorded and logged as a separate case. In total, there are thus 142030 cases.

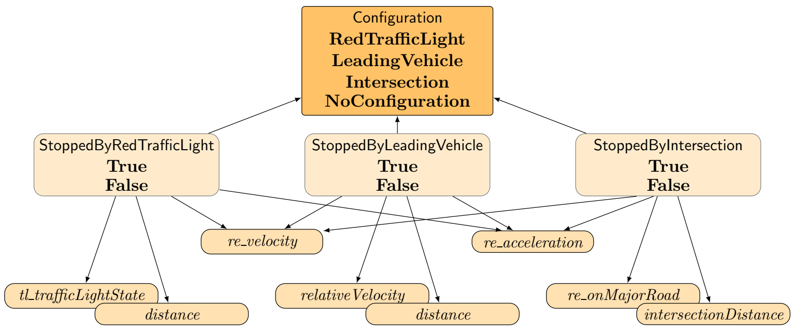


Figure 6.1.: *The Bayesian Network used for recognizing configurations.*

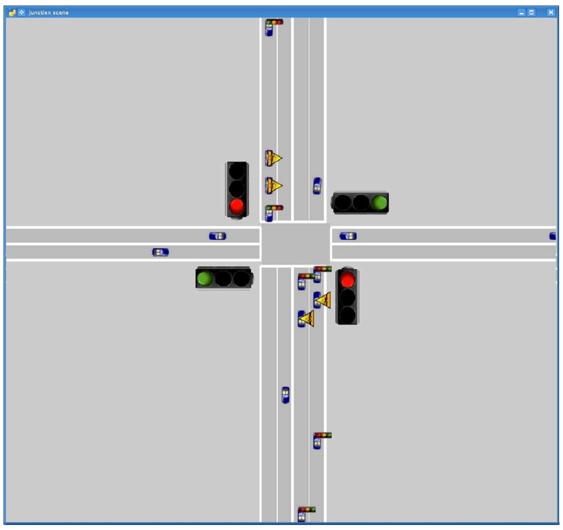


Figure 6.2.: *A screenshot of the top view of the intersection scenario used for the CarD simulation. Icons on the cars indicate their corresponding situation.*

For the evaluation the cases are randomly split into 10 partitions in order to employ a 10-fold cross-validation. The Bayesian Network is trained using Expectation-Maximization.

6.1.2. Results

The results obtained are given in the confusion matrix shown below:

The matrix has high values on its main diagonal, indicating a high accuracy for recognizing each of the four possibilities. The overall recognition accuracy, averaged over the 10 folds, is 97.9%. Accordingly, 2.1% or about 2900 cases were misclassified. The major share of these misclassifications occurs in cases where a

TL	LV	IS	None	← Measured/Actual ↓
30530	792	17	241	TL
293	**25481**	1185	0	LV
0	30	**1184**	0	IS
0	355	138	**81784**	None

Table 6.1.: *Confusion matrix for configuration recognition.*

vehicle is forced to brake by more than one entity and it is hard to distinguish which of them is the most relevant one. One example is when a vehicle approaches a red traffic light at which another vehicle is already waiting. Then, for a certain range during the approach, it is hard to tell whether the considered vehicle is more affected by the traffic light or the leading vehicle. This is exacerbated by the property that individual agents differ in their driving style and thus react slightly different to other entities (see 5.3.6). In view of the fact that at a busy intersection as the simulated cases with more than one potential affecting entity occur frequently, a misclassification rate of 2.1% can be considered low.

The configuration that is least accurately recognized is *StoppedByLeadingVehicle* with an accuracy of 94.1%. The highest accuracy, 99.3% was achieved for the case where a vehicle is in no configuration.

The results show that the proposed method is able to robustly recognize the configuration of a vehicle. The method can therefore be used to assess complex traffic situations according to the concept of configurations.

6.2. Incremental Situation Assessment

The method that is evaluated in the following was described in Section 4.2. It is concerned with tackling the problem that for recognizing a single vehicle's driving situation all considered features of all considered configurations need to be measured, which is computationally costly. The proposed solution employs an active measurement process where features are obtained one after the other until the belief in one configuration is sufficiently high. At this point the measurement process is terminated and the most probable configuration is returned as result. The order of measurements is obtained by selecting in each step that feature with the highest expected information gain as computed in the Bayesian Network for configuration recognition.

6.2.1. Evaluation Method

The active measurement process pursues two goals:

1. Achieve a high belief in the correct configuration quickly
2. When terminating the measurement process early, the most probable configuration should be the correct one

Goals 1 and 2 are interrelated because both of them contribute to a correct recognition after as few measurements as possible. Nevertheless, a separate evaluation allows to determine which aspects work and which do not work in the proposed method. If the first goal is not met, the proposed method will – for a given, conservative threshold – still measure most of the features and the saving by using the method will be negligible. If the second goal is not met, the most probable configuration at the time of termination will often be a wrong one.

The benefit of using information gain for determining the order of measurements is quantified by comparing the results with a baseline algorithm. The baseline algorithm selects in each step randomly one of the yet unmeasured features.

In this evaluation the same data set is used as in the previous section, because the active measurement process is an extension of the therein evaluated method. By using the same data the results can be compared.

6.2.2. Results

The degree to which the first goal, achieving a high belief in the correct configuration quickly, is met can be seen in Figure 6.3. It shows the average belief in the correct configuration after a given number of measurements performed. Using information gain for selecting the order of measurements, the average belief surpasses 80% after only 3 measurements and it surpasses 90% after 4 measurements. In contrast, the baseline algorithm needs 7 out of 8 measurements for reaching 90% confidence. Given these results it can be stated that the first goal of the active measurement process is met.

In Figure 6.4 the recognition accuracy achieved after a given number of measurements is plotted. Using information gain as selection criterion results in a steep increase of the recognition rate as compared to the gradual increase of a random selection. The proposed method is able to correctly recognize the configuration in more than 96% of the cases after performing only 3 measurements. This is less than two percentage points away from the recognition accuracy that is obtained after 8 measurements, which is 97.9%, as reported in Section 6.1. Opposed to that, the baseline method requires 6 measurements to reach an accuracy above 90%.

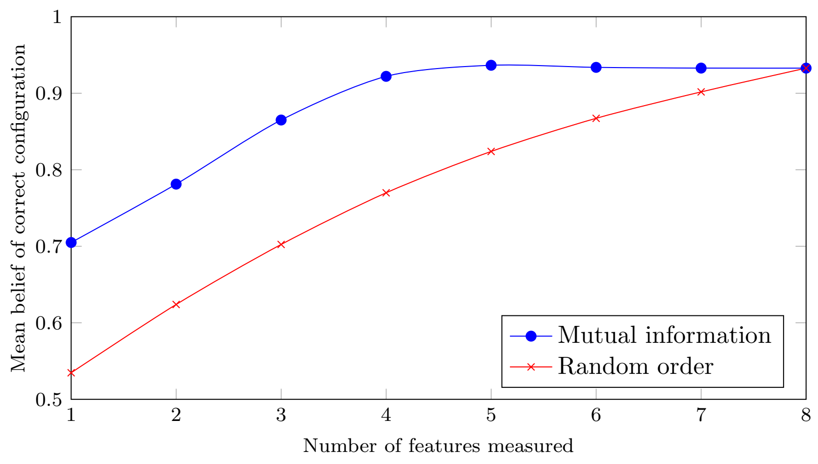


Figure 6.3.: *Average belief in the correct configuration after conducting a certain number of measurements. After only four measurements the belief surpasses 90% when using mutual information.*

The evaluation shows that both goals have been met. The proposed method is able to quickly maximize the belief in a single, confident configuration that is in the vast majority of cases also the correct one. Using the method allows to save more than half of the measurements while impairing the recognition accuracy by less than 2 percentage points. Nevertheless it has to be noted that the measurements are assumed to be free of noise. It is possible that erroneous measurements could impact the recognition accuracy more severely using the active measurement process than the recognition method presented in the previous section, because the latter takes always all available evidence into account. Investigating this topic is left to future work.

6.3. Predicting Velocity Profiles

The method evaluated in the following utilizes recognized configurations for estimating the future evolution of a situation. The future evolution is anticipated by predicting the future velocity profile of each individual vehicle present. The overall system is realized as a two-staged approach which was presented in Section 4.3. In the first stage the configuration of a vehicle is determined in order to select in the second stage a configuration-specific prediction model. Each prediction model is tailored to its corresponding configuration by regarding only those features that are considered relevant for the velocity profile in that configuration.

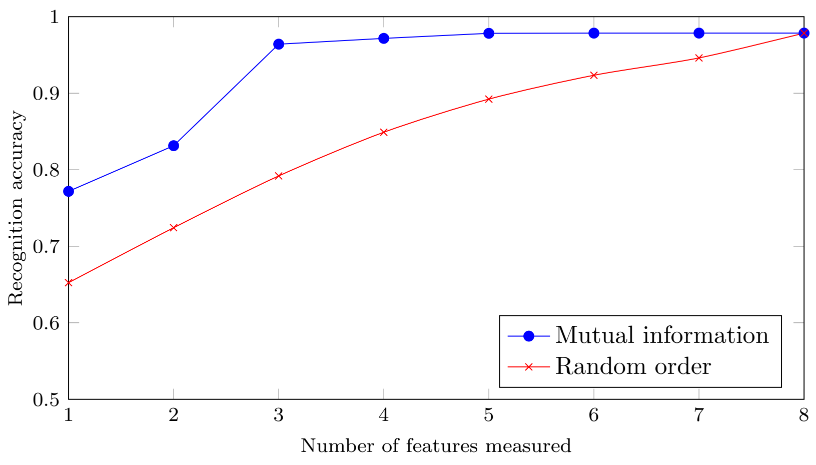


Figure 6.4.: *Recognition accuracy when terminating the measurement process after a certain number of measurements. After only 3 measurements a recognition accuracy of 96% is achieved when using mutual information as selection criterion.*

6.3.1. Evaluation Method

The data for the evaluation is again obtained by a simulation run of *CarD*. In an urban intersection scenario a total of 30 minutes was logged, of which the first 20 minutes are used as training data and the rest is used as testing data. This results in 15361 cases for training and 7084 cases for testing. Here, a case consists of all considered features as measured for a single vehicle at a certain point in time t_0 as well as the velocity profile for the next three seconds. The velocity is sampled at 10 Hz, thus the velocity profile consists of 30 individual values for $t_0 + 0.1s$ up to $t_0 + 3s$. The error measure used for the evaluation is the sum of squared distances. The prediction error e between the actual velocity profile $\boldsymbol{V}$ and the predicted velocity profile $\hat{\boldsymbol{V}}$ is therefore:

$$e = \sum_{t=1}^{30} (\hat{V}_t - V_t)^2 \tag{6.1}$$

In order to put the results obtained into perspective, two alternative methods for profile prediction are also evaluated which are tagged *KINEMATIC* and *PREDONLY*. *KINEMATIC* employs a straightforward prediction method and serves as a baseline such that the gain of more sophisticated methods can be assessed. The prediction method simply extrapolates the current kinematic properties of a vehicle into the future. The predicted velocity $\hat{V}_t$ for time step t given the current velocity v_0 and acceleration a_0 is thus:

$$\hat{V}_t = max(0, v_0 + t * a_0) \tag{6.2}$$

The *max* operator prevents a calculation of negative velocities.

The *PREDONLY* method takes also solely current velocity and acceleration as well as time as input features. As opposed to *KINEMATIC*, *PREDONLY* learns a prediction model from training data. This allows the model to capture certain characteristics of the training data, like the speed limit or typical acceleration and deceleration behaviors. A Random Forest Regression is employed as prediction model, the same regression method that is used by the proposed two-staged approach. The proposed method will be later on referred to as *TWO-STAGED*.

If the proposed *TWO-STAGED* method excels the other methods used for reference this could be attributed to the fact that the reference methods are limited to three input features while *TWO-STAGED* has eight features at its disposal. In order to exclude this possibility and to show the benefit of a two-staged approach a variant of *TWO-STAGED*, tagged *TS-BASIC*, is also evaluated. The prediction models of *TS-BASIC* are limited to the same three features as *PREDONLY* is, with all other aspects being equal.

6.3.2. Results

The results of the evaluation are given in Table 6.2. In order to ease the comparison the total error on the test set for each of the methods was divided by the total error of the proposed *TWO-STAGED* method, thus giving the relative error. The table shows that *KINEMATIC* yields an error more than twice as high than the proposed method and also *PREDONLY* is 26% less accurate. Even the stripped-down variant *TS-BASIC* performs better than both reference methods.

Method	Relative error
TWO-STAGED	1
TS BASIC	1.17
PREDONLY	1.26
KINEMATIC	2.27

Table 6.2.: *Relative velocity prediction error*

An interesting aspect of a prediction model is its accuracy depending on the time. In Figure 6.5 the relative velocity error over time for all methods is plotted. The error of *KINEMATIC* rises steeply for a prediction horizon beyond 0.5 seconds. The result demonstrates that a velocity profile prediction based on a simple extrapolation of a vehicle's current behavior is not sufficiently accurate. *PREDONLY* and *TWO-STAGED* achieve almost the same accuracy for the first

1.3 seconds, but then the error of *PREDONLY* rises significantly faster than for *TWO-STAGED*.

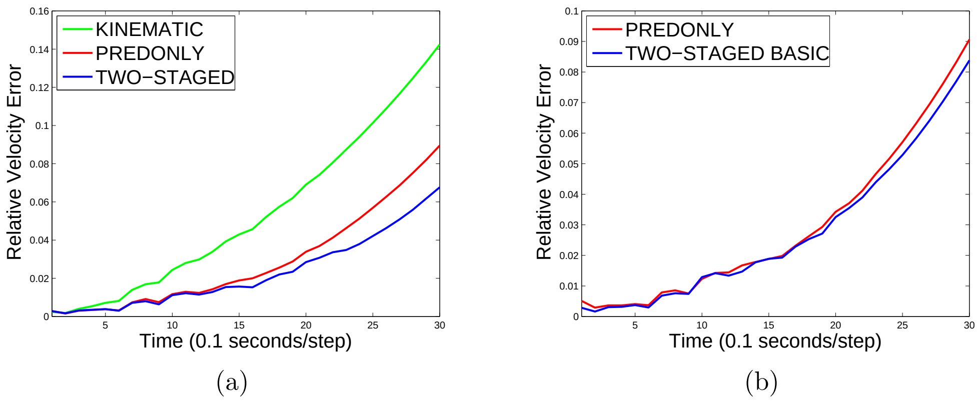


Figure 6.5.: *The relative prediction error over time is lowest when employing the proposed two-staged method (a), even if no configuration-specific features are used (b).*

A similar observation can be made for the comparison in Figure 6.5(b): For longer prediction horizons the improved accuracy of the *TS-BASIC* method as compared to *PREDONLY* becomes increasingly visible.

The results show that the proposed method is superior to the methods that neglect configuration information for their prediction models. The impact that the 21% higher accuracy of *TWO-STAGED* has can be best understood when considering two findings on the data set. First, about a fifth of the cases in the test set contain a stopped car for which the velocity prediction is trivial so that the proposed method can not set itself apart. Secondly, *TWO-STAGED* performs less severe mispredictions than the reference methods. This can be quantified by integrating the predicted velocity profile to obtain a future position. *KINEMATIC* is off by more than an average car length (4 meters) in about 19% of the cases in the test set and *PREDONLY* is off in 10% whereas this happens for *TWO-STAGED* only in 5% of the considered cases.

The results obtained for *TS-BASIC* also confirm that a two-staged approach is beneficial for the accuracy. The specialization of the prediction models to their corresponding configuration improves the prediction accuracy notably, even without taking additional features into account. But it is also important to note that the prediction models of the proposed *TWO-STAGED* method incorporate at most two additional features and the thereby improved accuracy shows that these configuration-specific features are indeed relevant for the prediction.

6.4. Learning Configurations from Observations

In Section 4.4 a method was presented that is designed for the case where a large amount of unlabeled real-world driving data is available. Using the method a time-consuming, manual labeling with configuration information is circumvented by learning the configurations from the data itself. The proposed method learns a parameterization of a complete prediction system for velocity profiles, as it was presented in Section 4.3. The system employs a two-staged approach where in the first stage a vehicle's configuration is estimated and then in the second stage a dedicated, configuration-specific prediction model is used for estimating the vehicle's future velocity profile. In the following the learning method will be referred to as *TS-LEARNED.*

6.4.1. Evaluation Method

The evaluation is split into two parts. In the first part the degree to which the learned configurations match actual configurations is quantified. For this purpose, *TS-LEARNED* is applied to driving data obtained from *CarD*, where the actual configuration of a vehicle is known. The data set stems from a 40 minute long simulation of urban traffic and comprises 35506 cases. Each case consists of a vehicle's velocity profile for the next three seconds and nine features indicating its current driving situation:

- *Velocity (VEL)*: Velocity of target car in $^m/_s$
- *Acceleration (ACC)*: Acceleration of target car in $^m/_{s^2}$
- *Traffic light distance (TLD)*: Distance to the stopping line of the next, relevant traffic light in m
- *Traffic light state (TLS)* : State of next, relevant traffic light. 1 if green, 0 otherwise
- *Car ahead relative speed (CAS)*: Relative velocity between target car and its leading car in $^m/_s$
- *Car ahead distance (CAD)*: Distance between target car and its leading car in m
- *Car ahead TTC (TTC)*: Time to contact between target car and in ts leading car in s
- *Intersection distance (ID)*: Distance to the entry point of the next intersection in m

- *Major Road (MJ)*: Whether target car is driving on a major road (1) or minor road (0).

TS-LEARNED uses this data to identify configurations; after termination it is determined whether the cases belonging to an actual configuration are also assigned to the same learned configuration. This is accomplished by using the *purity*-measure known from the field of clustering [Zhao and Karypis, 2001]. Using this measure, each learned configuration $\hat{C}_n$ is assigned to the configuration C_n which occurs most frequent in the cases assigned to $\hat{C}_n$. Formally,

$$purity(\hat{\boldsymbol{C}}, \boldsymbol{C}) = \frac{1}{I} \sum_n \max_n |o_k \cap \hat{o}_k| \tag{6.3}$$

where I denotes the number of cases, o_n and $\hat{o}_n$ denote the cases belonging to C_n and $\hat{C}_n$, respectively. A value of 1 indicates a perfect match whereas a purity of 0 denotes no match. Because the optimization problem solved by *TS-LEARNED* is not convex its solution varies depending on the random initialization of the assignment matrix $\boldsymbol{H}$. In order to account for this, the proposed method is run 64 times and the obtained results are aggregated to give a reliable value.

In the second part of the evaluation the prediction accuracy of *TS-LEARNED* is compared to two other methods. A baseline algorithm, tagged *KINEMATIC*, performs its prediction by extrapolating the current velocity and acceleration like it was specified in Equation 6.2. The other method is a state-of-the-art regression model based on Random Forests. The Random Forest Regression (*RFR*) directly takes all features as input and returns a velocity value.

For the comparison of *TS-LEARNED* with the reference methods data obtained from real-world test drives is used. The data set was recorded by a test vehicle equipped with multiple sensors. A laser scanner provides the distance to and the relative velocity of a vehicle ahead, yielding features CAD and CAS. The CAN-bus is tapped for obtaining the kinematic state of the ego-vehicle (VEL, ACC). Camera data is used to determine stop line positions and the states of traffic lights (TLD, TLS). Additionally the time step is added as variable which is utilized by the reference methods to specify the time instance for which a velocity value is predicted. The time step ranges from $0.1s$ to $3s$. The data set comprises a total of 29 minutes of recording of which the first 15 minutes were used for training and the remaining 14 minutes were used for testing.

For all experiments the learning parameter l of *TS-LEARNED* is set to 0.1 and the algorithm terminates if the relative improvement between two consecutive iterations is less than 1%.

Figure 6.6.: *Camera images taken from the data set used for training and testing.*

6.4.2. Results

The purity between the configurations learned by *TS-LEARNED* and the actual configurations as given by *CarD* is shown in Table 6.3. The average purity is 0.63 and thus the learned configurations have only a limited similarity with the actual ones. The reasons for this can be either that the method is unable to capture the underlying domain knowledge or that by coupling configuration recognition with velocity prediction the latter dominated the former part during the minimization.

Purity		
Min	Mean	Max
0.38	0.63	0.80

Table 6.3.: *Match between learned configurations and actual configurations.*

An indication that the minimization method of *TS-LEARNED* emphasizes the prediction part can be found when considering the results obtained in the second part of the evaluation. In Figure 6.7 the prediction accuracy of *TS-LEARNED* and the methods used for reference is given. It shows that the proposed method is significantly better than *KINEMATIC* and *RFR*, especially for predictions beyond 1 second in the future. The unexpected bad performance of *RFR* was in a separate evaluation traced to the fact that the statistics of training and test set differ considerably. The results indicate that the two-staged approach of *TS-LEARNED* adds a sufficient degree of robustness to this variation.

The first part of the evaluation shows that the configurations learned are not comparable to the configurations as specified in Chapter 3. They are not necessarily bilateral, that is restricted to a single affecting entity, for example when both traffic light distance and car ahead distance are features of the same configuration. Furthermore, the learned configurations do not enable any conclusions on affecting

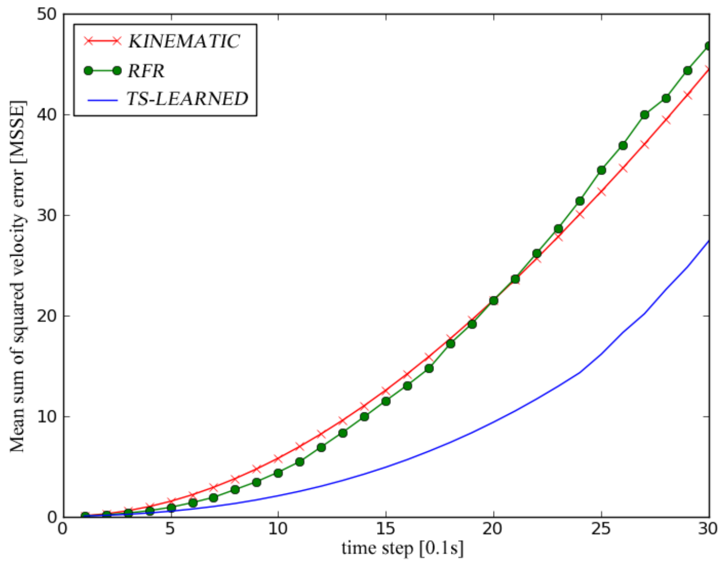


Figure 6.7.: *TS-LEARNED excels both KINEMATIC and RFR, especially for longer prediction horizons.*

and affected entity. Nevertheless, the proposed method is particularly suited for a specific application if large amounts of unlabeled driving data are available: The results show that the proposed method allows to parameterize a prediction system that is capable of beating a state-of-the-art regression method.

6.5. Intention Estimation

A central claim in this thesis is that configurations are a useful concept for assessing traffic situations and that a configuration captures the driving situation of a vehicle adequately. Furthermore, the concept of configurations is developed to extend the state-of-the-art.

To put these claims to the test, a demonstration system for intention estimation was developed as described in Section 4.5. The system distinguishes between four instead of the usually considered two intentions, which are

I. Go straight

II. Turn right

III. Stop at red traffic light

IV. Car following

The increased number of intentions is enabled by combining the typically considered behavioral cues with situational cues, where the latter are captured by two configurations. For the behavioral cues also a novel feature has been engineered, the *Anticipated Velocity at Stop Line* (AVS), which is expected to improve the accuracy of the behavioral part of the intention estimation.

During an intersection approach, the overall system continuously computes the expected *Time-To-Intersection* (TTI). If the TTI drops below 1.5 seconds, both behavioral cues and situational cues are measured and combined and the system returns the estimated intention.

6.5.1. Evaluation Method

The evaluation consists of three parts. In the first part the accuracy of the TTI estimation is benchmarked. This is necessary because if the actual TTI is overestimated such that the remaining time to the stop line is in fact significantly less than 1.5 seconds, then the estimated intention will be available too late for subsequent systems to react properly.

In the second part of the evaluation the benefit from using the newly developed AVS feature is quantified. For this purpose the accuracy of the behavior-based intention estimation when using the traditional features velocity and acceleration is compared with the accuracy obtained when using solely AVS. The distance is provided as feature in both cases. For the comparison only approaches with the intentions *I* or *II* are used, because for these intentions no situational cues are required.

The third part of the evaluation is concerned with the overall system performance. Since among the 37 intersection approaches used for the evaluation there are only four approaches for 'Car following', a stratified four-fold cross validation is employed, such that in each fold such an approach is present. In order to provide sufficient training data all measurements from each approach in the training set, where the vehicle is less than 25 meters away, are incorporated. This yields 600 cases per fold for training.

6.5.2. Results

The relation between estimated and actual TTI is depicted in Figure 6.8. The histogram shows that the computation method for the TTI used here is a conservative estimation such that the actual TTI is rather under- than over estimated. For all approaches more than one second remains until the vehicle reaches the stop line and for only six approaches the remaining time is less than the anticipated 1.5 seconds.

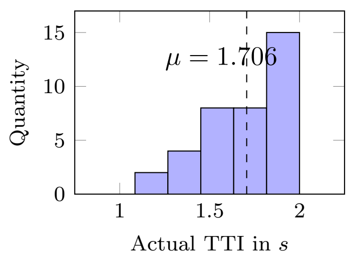


Figure 6.8.: *Histogram of the actual TTI at a predicted TTI of 1.5s*

The results of the comparison between kinematic features and the proposed AVS feature are given in Figure 6.9. The ROC curves plotted therein illustrate the superiority of the AVS feature. The behavior-based estimation is significantly more accurate when relying on the AVS feature than when employing kinematic features directly. Nevertheless, it has to be noted that this evaluation takes only 20 approaches into account and before far-reaching generalizations can be derived more data will be needed.

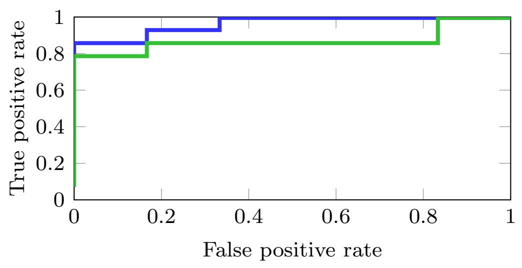


Figure 6.9.: *Behavior recognition of 'Go straight' and 'Turn Right' using Logistic Regression. The AVS feature (blue) is significantly more accurate than velocity and acceleration (green).*

The performance of the overall intention estimation system is presented in Table 6.4. The confusion matrix shows that only three approaches have been misclassified, yielding an overall classification accuracy of 91.9%. The system is able to identify only 50% of the approaches labeled 'Car following' correctly, though this is satisfactory given the low number of training examples.

The overall system performs also well for longer prediction horizons. In Figure 6.10 the estimation accuracy is plotted against the estimated TTI. The plot shows that accuracy decreases slowly for longer horizons and even for a TTI of 3 seconds the accuracy is above 80%.

I	II	III	IV	← Prediction / Actual ↓
6	0	0	0	I
0	13	0	1	II
0	0	13	0	III
0	2	0	2	IV

Table 6.4.

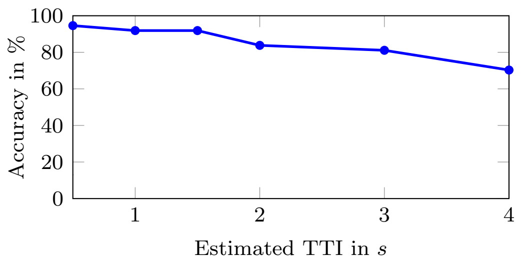


Figure 6.10.: *The accuracy of the overall system, dependent on the prediction horizon*

The results document that the goals set in the forefront have been attained by the final system. The estimated TTI leaves sufficient time for subsequent systems to react to an estimated intention. The proposed AVS feature is superior to a direct incorporation of kinematic features. Furthermore, the overall system achieves an accuracy of nearly 92% even though the increased number of considered intentions complicates the estimation task.

6.6. Discussion

In the preceding evaluation the feasibility of the methods proposed in Chapter 4 has been confirmed. At first, in Section 6.1 it was shown that multiple configurations can be reliably recognized using a Bayesian Network. The network achieves a recognition accuracy of 97.9%.

In Section 6.2 it was furthermore shown that the recognition process itself can be streamlined in order to save sensory and computational resources without deteriorating the recognition rate considerably. The proposed reduction method takes advantage of the fact that the recognition is realized as a Bayesian Network and allows to save more than 50% of the measurements normally taken while maintaining an accuracy above 96%.

Section 6.3 details how the explicit consideration of a vehicle's configuration can be used to improve a system for behavior prediction. Training and employing configuration-specific prediction models leads to a significantly more accurate estimation of a vehicle's future velocity profile than when configuration information is neglected.

For cases, where sufficient unlabeled driving data is available but the capacity for its annotation is lacking, the method evaluated in Section 6.4 can be employed. The proposed method is capable of parameterizing a complete prediction system using solely unlabeled data. Additionally, the resulting system is superior to state-of-the-art regression methods. However, it turns out that the configurations as defined in Chapter 3 can not be learned from unlabeled data, especially because the learned configurations miss their bilateral character and any interpretability.

The demonstration system evaluated in Section 6.5 puts configurations to the test in a real-world application. The results confirm that by using configurations the state-of-the-art in the area of intention estimation can be advanced. The system is able to distinguish reliably between four different driver intentions whereas nowadays approaches consider usually only two.

The first three methods were solely evaluated on simulated data. It is possible that they profited from using this data, as simulated data can be assumed to be much cleaner and more accurate than data taken from the real world. Additionally, the simulator provides an abundance of data, which allows for parameterizing also complex models. However, the remaining two methods are based on the first three methods and turned out to work well on real data. This indicates that the findings on simulated data have a sufficient validity for real-world data.

Although the evaluation has shown that the methods proposed in Chapter 4 work the way as it was hoped for, the evaluation also revealed many aspects that require further investigation. For example, why are the learned configurations so far off from the actual configurations? Is this a property of the minimization algorithm or is the domain knowledge that is encoded during a manual specification simply not present in driving data? These and more starting points for future work are given in Chapter 7.

7. Future Work

Developing the foundation of an urban driver assistance system is an open-ended research project. The preceding evaluation showed that the methods proposed in the course of the work accomplish the goals they were developed for, however, there is plenty of room for improvement. This chapter discusses possible starting points for future work.

In Section 7.1 possible extensions are discussed which are expected to enhance the capabilities of the proposed methods and tackle their shortcomings. Section 7.2 is concerned with the question how the concept of configurations can be further augmented to increase its applicability for future driver assistance systems. It names areas in which further research seem promising.

7.1. Extensions

The active measurement process proposed in Section 4.2 can be extended in multiple ways. One option is to assign different costs to individual measurements. These costs can be based on the computational effort required for a sensor processing. Additionally, the process should consider the case where the same sensor measurement can be used for computing multiple features, for example when a camera image is acquired for both detecting cars ahead and on nearby lanes. Insights from an approach, where a related problem has been tackled for a computer vision system, can be found in [Rebhan et al., 2009].

One downside of recognizing configurations using an active measurement process is the high latency resulting from measuring features sequentially. If this high latency becomes an issue further research on a cost function that considers parallel measurements will be needed.

The behavior prediction system that was detailed in Section 4.3 offers also multiple starting points for improvement. So far, only the longitudinal behavior in form of velocity profiles is considered, whereas an extension towards lateral behavior would be interesting. Lateral behavior can be used to anticipate the next maneuver of a vehicle; the resulting system would be thus related to the intention estimation system proposed in Section 4.5 and could adopt the behavior estimation used therein.

Because nowadays the availability of driving data is constantly increasing, methods that are able to learn directly from data will become more and more important. One study for example, the SHRP2 naturalistic driving study [Campbell, 2012], expects to record one petabyte of driving data. The data will in its first version only consist of unlabeled, raw sensor measurements. The method proposed in Section 4.4 was developed to work with unlabeled data. The evaluation showed that it is possible to parameterize a competitive prediction system but it also revealed that the algorithm employed is unable to identify the underlying configurations. This could be countered by introducing additional terms into the target function, that punishes when a learned configuration comprises more than two entities. Furthermore, the currently implemented feature selection mechanism which ensures that each feature is only exclusively assigned to a single configuration, hinders the computation of a gradient. A solution for this problem is to replace the currently used binary feature selection matrix by a continuous weight matrix, as it was done for the assignment matrix $\boldsymbol{H}$. At the same time, another term is added to the target function that punishes when a feature is utilized by multiple configurations.

The intention estimation system detailed in Section 4.5 could be further extended to anticipate also left turn maneuvers. This requires a lane-level accurate localization, like it was proposed in [Vu et al., 2012], and sufficiently accurate map data. The currently needed manual annotation regarding the traffic light state can be replaced by a state-of-the-art traffic light detector as it was presented in [deCharette and Nashashibi, 2009].

7.2. Potential Research Directions

In order to make the increased volume of available driving data usable for all presented methods rather than only the learning algorithm, a convenient solution for labeling needs to be found. A possibility, which was not further pursued due to time constraints, is to employ a driver behavior model. The intuition behind this is that if a driver's reaction to leading vehicles or red traffic lights is captured by a driver model, it will be possible to parameterize this model using the data. For example, when in the data the driver approaches a signalized intersection many times and in some cases the traffic light shows green and in other cases it shows red then differences in the driver's behavior can be attributed to the traffic light state – given that everything else remains unchanged. Even though every two approaches will differ in more than one aspect, if the number of recorded approaches is sufficiently high these changes can be neglected. A driver model that takes multiple driving situations into account is for example the CAIDM [Sridharan et al., 2012], which is an extension of the Intelligent Driver Model [Treiber and Helbing, 2002]. A driver model, that is learned from data, was lately presented in [D'Agostino et al., 2013].

For test drives where the need for configuration information is known in advance the labeling can take place during the recording itself. A method for this purpose is the think-aloud-protocol as proposed in [Lewis, 1982]. It was recently employed for the evaluation of an ADAS for traffic jam mitigation [Risto and Martens, 2013]. Using the think-aloud-protocol the driver would comment on the decisions made during driving, especially which road users are currently affecting him. If the protocol is sufficiently reliable and the additional workload remains manageable for the driver such a protocol could speed up the later annotation by an expert.

Another promising extension to this work is to increase the number of considered configurations. Though many situations can be covered by the four configurations used so far, the regarded entities are currently only vehicles and traffic lights. For urban driver assistance systems the consideration of pedestrians, bicyclists and zebra crossings would open up additional areas of use. A higher number of entities challenges the current recognition method, but an advantage of configurations with a single affecting entity as used here is that each possible affected entity can be treated by a separate Bayesian Network.

Future work can also be directed to the development of methods that are concerned with anticipating critical situations. By now, the future evolution of a situation is only predicted without any assessment of potentially upcoming hazards. Inspiration for research in the direction of risk estimation is provided by the works of Lefèvre [Lefèvre et al., 2011, Lefèvre et al., 2012]. Here, a mismatch between the anticipated and the usually expected behavior of a driver is taken as a measure for quantifying risk. This approach can be transferred to the concept of configurations: a vehicle for which situational cues hint to a *StoppedByRedTrafficLight* configuration while its behavior is more close to *NoConfiguration* might be about to run a red light. A method that identifies this mismatch and draws appropriate conclusions from it would be able to identify risky behavior and warn the driver.

8. Conclusion

In the preceding chapters fundamental research has been presented towards an Advanced Driver Assistance System for inner-city intersections. In this chapter a summary of the work is given.

Chapter 1 started with an analysis of traffic accident reports. The statistics reveal that the most accident-prone spots in the road network are urban intersections. Furthermore, an in-depth study found that intersection-related crashes can almost exclusively be attributed to errors committed by the driver. The findings suggest that an intelligent system for assisting the driver in negotiating intersections could greatly reduce the number of accidents. However, for the time being such Advanced Driver Assistance Systems are neither commercially available nor have they been presented in academia; current ADAS are only helpful before an intersection is entered. But in order to support the driver throughout a crossing maneuver a comprehensive assessment of the current traffic situation is needed. Such an assessment turns the traffic situation into a descriptive model that captures dependencies and interactions of all road users present. A descriptive model provides a basis for performing inference and gaining an understanding of the situation. While understanding a situation is a value per se, using this understanding to predict upcoming conflicts promises further benefit. Based on these considerations the goal of the PhD project was specified as finding a method for situation assessment that can also be employed to improve the state-of-the-art in predicting future situations.

In order to find the desired methods a literature review was conducted in Chapter 2. Three research areas were identified as closely related: Situation Assessment, Intention Estimation and Trajectory Prediction. The review revealed that in the field of Situation Assessment none of the current works provide an adequate solution. Some works consider only a single, specific situation or are restricted to highway scenarios. More versatile methods require an extensive modeling by designated experts which becomes intractable the more complex the covered situations are. Similar findings were made for works presented in the fields of Intention Estimation and Trajectory Prediction each of which misses at least one aspect important for this work. Some approaches are highly complex which results in difficulties concerning their parameterization as well as their application to a real-world system. Other methods are unable to deal with situations – or are at least severely impaired – if more than the considered one or two road users are present. Two pivotal conclusions were drawn from the literature review: probabilistic methods are widely accepted as state-of-the-art due to their ability to model sensor uncertainties. Furthermore, dependencies and interactions between road users are currently rarely taken into account.

Based on the insights gained from the preceding review the concept of configurations was devised in Chapter 3. It is based on the fact that urban traffic situations are too complex and too diverse for a straightforward assessment. Neither an assessment on a manageable set of predefined, prototypical situations nor a brute-force interpretation relating every entity to each other are feasible. To overcome the problems caused by the high complexity and variability of urban traffic situations the decomposition of situations into smaller parts, which are easier to handle, is proposed. Each part consists of a pair of interrelated entities, with one entity affecting the behavior of the other. Possible constellations of interrelated entities are defined by a human expert in a model that is tagged 'configuration'. Configurations are specified as graphs such that they can be directly mapped to a Bayesian Network (a probabilistic model). The Bayesian Network can then be used for recognizing the presence of the corresponding configuration.

In Chapter 4 several methods were presented, all of them related to the concept of configurations. A method for recognizing configurations is proposed which aims at identifying the correct configuration of a road user out of multiple possibilities. The method is then extended to a streamlined version for improving the efficiency of the recognition process. Both of these methods are concerned with the descriptive part, the assessment of a traffic situation. In addition, approaches for situation prediction which employ these recognition methods were presented. One approach is concerned with predicting the velocity profiles of other vehicles by utilizing configuration-specific prediction models. Another approach is tailored to the case where large amounts of unlabeled data are available. For this case a method is proposed that learns not only the parameters of a prediction system but also tries to identify configurations from the data. In order to show the applicability of configurations to real-world systems an intention estimation system was also developed. It aims at distinguishing between four considered intentions instead of the usually two in state-of-the-art methods. The claim is that this made possible not only by an also newly engineered feature but mainly by the explicit incorporation of configuration information.

Instead of a subsequent evaluation of the proposed methods, the next chapter, Chapter 5, presented the traffic simulation framework *CarD*. This order was chosen because the evaluation relies to a large extent on driving data that were generated by the simulator. The simulation framework *CarD* was developed in the course of this work after it became clear that no currently available traffic simulator grants access to a driver's behavior model. In order to obtain a sufficiently realistic simulation, traffic is not governed by a central instance but each vehicle is controlled by an individual intelligent agent. Agents perceive the world via virtual sensors and choose their driving behavior according to the gathered information and their goals. Based on the selected behavior of an agent its configuration can be determined and logged.

In Chapter 6 both simulated driving data and data obtained during test drives were used to evaluate the proposed methods. On a large data set it was shown that configurations were reliably recognized by the designated method; an accuracy of nearly 98% was achieved. The proposed extension for saving computational resources turned out to be able to halve the required effort for an individual recognition without impairing the overall recognition accuracy significantly. Furthermore it was shown that the presented behavior prediction system profits from its explicit consideration of configurations and thereby outperforms state-of-the-art methods that neglect this information. Also the proposed learning algorithm for parameterizing prediction systems from unlabeled data demonstrated its advantages over state-of-the-art methods. However, the configurations identified by the learning algorithm did not match the specification as given in Chapter 3, because a learning algorithm can hardly extract the domain knowledge that a human expert introduces in the course of specifying a configuration. At last, the intention estimation system showed also the desired results. The newly engineered feature excels typically used kinematic features and the proposed method distinguishes between the considered intentions with an accuracy of nearly 92%. This is especially remarkable as the system has to distinguish between four instead of the usually considered two intentions in the state-of-the-art. The evaluation indicates that the higher number of considered intentions is made possible by the explicit incorporation of configuration information.

Possible starting points for future work were discussed in Chapter 7. The streamlined method for recognizing configurations could be further improved by parallelization and the prediction systems could also incorporate a vehicle's lateral behavior. In order to annotate driving data already during test drives the use of a think-aloud-protocol was proposed.

The work at hand was motivated by the finding that the major share of road injury accidents takes place at urban intersections. A pivotal contribution of this work is the newly introduced concept of configurations, which understands traffic situations as composed of parts. An additionally proposed method for recognizing configurations enables for the first time a situation assessment that scales also to complex situations. The related work focused so far either only on highway scenarios, was limited to at most two vehicles or struggled with the complexity of a comprehensive assessment.

Building on top of the new approach for situation assessment, multiple methods were presented for predicting the evolution of traffic situations. It is shown that, by taking configuration information explicitly into account, the current state of the art in prediction methods can be advanced. The research contribution made by this thesis is also demonstrated by an intention estimation system which is able to distinguish reliably between more intentions than related systems do.

The configurations used throughout the thesis did not consider entities such as bicyclists or pedestrians. Since these entities are usually encountered during urban driving, it is necessary to extend the proposed methods accordingly, but this has to be left to future work.

The research presented in this thesis provides a foundation for the development of an Advanced Driver Assistance System that assists the driver in negotiating urban intersections. Nevertheless, still a lot of work has to be done and it is hoped that this thesis provides a relevant part of which in the near future such a system will be composed.

A. Bayesian Networks

A Bayesian Network (BN) is a directed acyclic graph in which nodes represent random variables and edges represent conditional dependencies. Bayesian Networks describe the joint probability distribution of all considered random variables $X_i \in \boldsymbol{X}$. An edge between two random variables denotes their conditional dependence, whereas the lack thereof denotes their independence.

An exemplary Bayesian Network is given in Figure A.1. It models the relation between a vehicle's gas level, the turning of the ignition key and the start of the engine.

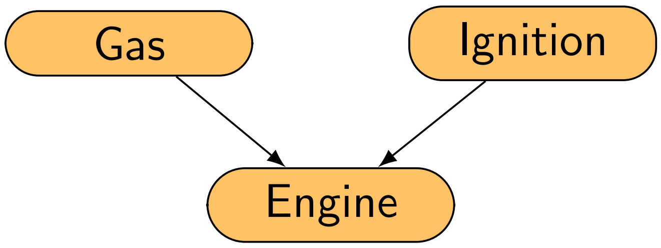


Figure A.1.: *An exemplary Bayesian Network. The random variables Gas and Ignition are conditionally independent.*

The Bayesian Network states that *Gas* (G) and *Ignition* (I) are conditionally independent, as their nodes are not connected by an edge. This means that

$$P(G|I) = P(G) \text{ and } P(I|G) = P(I) \tag{A.1}$$

At the same time *Engine* (E) is conditionally dependent from *Gas* and *Ignition*. The joint probability distribution of E,G and I is thus

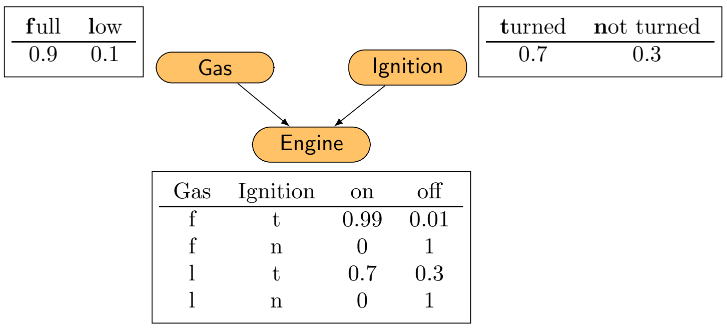
$$P(E, G, I) = P(E|I, G)P(I)P(G) \tag{A.2}$$

In general, the joint probability distribution of a Bayesian Network is given by

$$P(X_1, .., X_i, .., X_N) = \prod_i P(X_i|parents(X_i)) \tag{A.3}$$

where '$parents(X_i)$' returns the nodes of which X_i is conditionally dependent.

For a complete Bayesian Network the conditional probability distribution of each node given its parents needs to be specified. For nodes without parents a prior probability is needed. In case the considered probability distributions are discrete, the distributions are given by a conditional probability table. The example given above could for example have distributions as depicted in Figure A.2.



full	low
0.9	0.1

turned	not turned
0.7	0.3

Gas	Ignition	on	off
f	t	0.99	0.01
f	n	0	1
l	t	0.7	0.3
l	n	0	1

Figure A.2.: *Bayesian Network including priors and conditional probability tables*

A.1. Inference in Bayesian Networks

Bayesian Networks can be used to answer probabilistic queries about yet unobserved variables, which is termed as *inference*. The complexity of this inference is NP-hard, thus in real-world applications often approximate inference methods are used that are more tractable. In the work at hand inference is used to employ a Bayesian Network as classifier: given the features, the state of the unobserved node for the class label is inferred.

Since a Bayesian Network specifies a complete joint probability distribution over its variables, an inference query can be answered by *marginalization*, that is summing out any irrelevant variable. For example, one query to the network presented above could be to infer the probability that the engine will start when the ignition is turned, $P(E = on|I = turned)$. In this case, E is the query variable, I is the evidence variable and G is a hidden, irrelevant variable. G is thus marginalized:

$$\begin{aligned}
P(E = on|I = t) &= \sum_g P(E = on|I = t, G = g)P(I = t)P(G = g) \\
&= P(E = on|I = t, G = full)P(I = t)P(G = full) + \\
&\quad P(E = on|I = t, G = low)P(I = t)P(G = low) \\
&= 0.99 \cdot 1 \cdot 0.9 + 0.7 \cdot 1 \cdot 0.1 \\
&= 0.961
\end{aligned}$$

The probability that the engine will start is therefore 0.961.

In practice, a straightforward marginalization will be inefficient for networks with a higher number of nodes because many intermediate computations will be performed multiple times. Therefore, approaches to *variable elimination* were developed, in which by reusing intermediate results a significant share of computations can be saved [Kschischang et al., 2001].

A.2. Learning in Bayesian Networks

The values in a network's conditional probability tables are usually obtained by learning from training data. A fictive data set is given in Table A.1.

Observations		
Gas	Ignition	Engine
full	turned	on
low	turned	on
full	not turned	off
full	turned	on
full	not turned	off
full	turned	on
full	turned	on
full	turned	on
full	turned	on
full	not turned	off

Table A.1.: *Data set of observations.*

In cases where the structure of the Bayesian Network itself is already specified and the training data is fully observed (no missing entries), the individual probabilities can be determined by *counting learning.* As the name implies, the probabilities are obtained by simply counting the occurrences of each combination. For example, to compute the priors in the conditional probability table of *Gas* the frequency of both events - 'full' and 'low' - is divided by the total number of observations:

$$P(G = full) = \frac{\#full}{\#observations} = \frac{9}{10} = 0.9$$

$$P(G = low) = \frac{\#low}{\#observations} = \frac{1}{10} = 0.1$$

Like other machine learning algorithms, Bayesian Networks perform best when trained with sufficiently large and balanced data sets. The data set in Table A.1 has only ten observations and thus some combinations are not covered by a single observation, e.g. for determining $P(E|G = low, I = notturned)$. In this case either a default value can be taken or the data set has to be enlarged.

It is also possible that the available data set is incomplete in that one or more variables are unknown for some observations. In these cases *counting-learning* can no longer be used for learning. Instead, gradient descent [Russell et al., 1995] or expectation maximization [Dempster et al., 1977] algorithms need to be employed.

B. Test Drive Data

B.1. Test Drives

The data was obtained in the course of three test drives, conducted by two different drivers in the period between August and December 2013. In Figure B.1 a map of the downtown area of Offenbach is given, in which the test drives have taken place. A total of 37 approaches to 7 different intersections have been recorded.

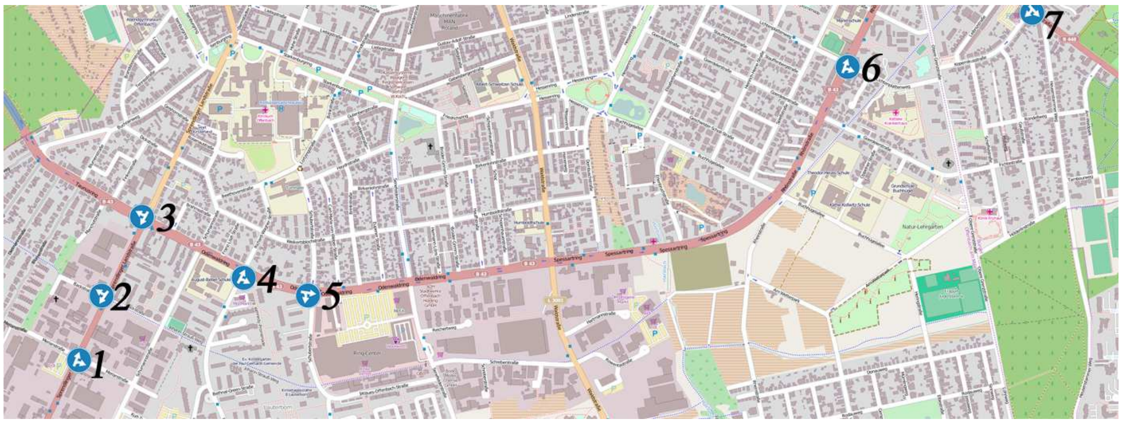


Figure B.1.: *Map of the downtown area of Offenbach, in which the test drives have taken place. The turning symbols mark the intersections at which approaches have been recorded.*

B.2. Distribution

The 37 approaches are distributed among the four intentions as given in Table B.1.

I.	'Go straight'	6
II.	'Turn right '	14
III.	'Stop at red traffic light'	13
IV.	'Car Following'	4.

Table B.1.: *Number of approaches per intention*

B.3. Intersections

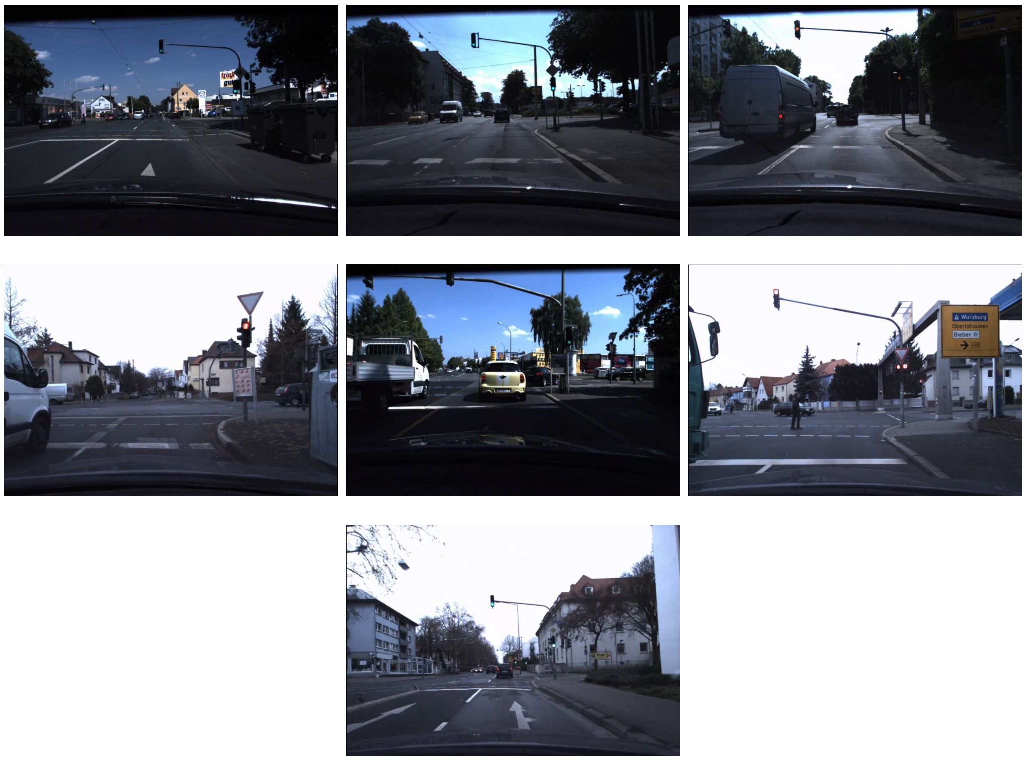


Figure B.2.: *Camera images of intersections* 1 *to* 7 *contained in the data set. Images are ordered from left to right, top to bottom.*

Bibliography

[sta, 2012] (2012). Unfallentwicklung auf deutschen straßen 2012.

[Alin et al., 2012] Alin, A., Butz, M. V., and Fritsch, J. (2012). Incorporating environmental knowledge into bayesian filtering using attractor functions. In *Intelligent Vehicles Symposium (IV), 2012 IEEE*, pages 476–481. IEEE.

[Althoff et al., 2009] Althoff, M., Stursberg, O., and Buss, M. (2009). Safety assessment of driving behavior in multi-lane traffic for autonomous vehicles. In *Intelligent Vehicles Symposium, 2009 IEEE*, pages 893–900. IEEE.

[Aoude et al., 2012] Aoude, G. S., Desaraju, V. R., Stephens, L. H., and How, J. P. (2012). Driver behavior classification at intersections and validation on large naturalistic data set. *Intelligent Transportation Systems, IEEE Transactions on*, 13(2):724–736.

[Armand et al., 2013] Armand, A., Filliat, D., Ibañez-Guzmán, J., et al. (2013). Modelling stop intersection approaches using gaussian processes. In *Proceedings of the 16th International IEEE Conference on Intelligent Transportation Systems-ITSC*.

[Baader, 2003] Baader, F. (2003). *The description logic handbook: theory, implementation, and applications*. Cambridge university press.

[Bishop, 1995] Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford university press.

[Bosch GmbH, 2013] Bosch GmbH, R. (2013). Driver assistance systems.

[Breiman, 2001] Breiman, L. (2001). Random forests. *Machine Learning*, 45.

[Breiman et al., 1984] Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A. (1984). *Classification and regression trees*. Chapman & Hall/CRC.

[Campbell, 2012] Campbell, K. L. (2012). The shrp 2 naturalistic driving study: Addressing driver performance and behavior in traffic safety. *TR News*, (282).

[Choi, 2010] Choi, E.-H. (2010). Crash factors in intersection-related crashes: An on-scene perspective.

[Comaniciu and Meer, 2002] Comaniciu, D. and Meer, P. (2002). Mean shift: A robust approach toward feature space analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(5):603–619.

[Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3):273–297.

[Dagli et al., 2003] Dagli, I., Brost, M., and Breuel, G. (2003). Action recognition and prediction for driver assistance systems using dynamic belief networks. In *Proceedings of the NODe 2002 agent-related conference on Agent technologies, infrastructures, tools, and applications for E-services*, pages 179–194.

[D'Agostino et al., 2013] D'Agostino, C., Saidi, A., Scouarnec, G., and Chen, L. (2013). Volvo group, features, verification & validation, 69800 st-priest, france. In *Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on*, pages 1778–1783. IEEE.

[Das et al., 2002] Das, S., Grey, R., and Gonsalves, P. (2002). Situation assessment via bayesian belief networks. In *Information Fusion, 2002. Proceedings of the Fifth International Conference on*, volume 1, pages 664–671. IEEE.

[deCharette and Nashashibi, 2009] deCharette, R. and Nashashibi, F. (2009). Real time visual traffic lights recognition based on spot light detection and adaptive traffic lights templates. In *Intelligent Vehicles Symposium, 2009 IEEE*, pages 358–363. IEEE.

[Dempster et al., 1977] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *J. o. t. Royal Statistical Society. Series B*, 39(1):1–38.

[Doob, 1944] Doob, J. L. (1944). The elementary gaussian processes. *The Annals of Mathematical Statistics*, 15(3):229–282.

[Friedman and Goldszmidt, 1996] Friedman, N. and Goldszmidt, M. (1996). Discretizing continuous attributes while learning bayesian networks. In *ICML*, pages 157–165.

[Gabard and Breheret, 1999] Gabard, J. and Breheret, L. (1999). The sitra-b+ microscopic traffic simulation model - examples of use and future developments. Technical report, ONERA/CERT.

[Gindele et al., 2013] Gindele, T., Brechtel, S., and Dillmann, R. (2013). Learning context sensitive behavior models from observations for predicting traffic situations. In *Proc. 16th Int Intelligent Transportation Systems (ITSC) IEEE Conf.*

[Gipps, 1981] Gipps, P. G. (1981). A behavioural car-following model for computer simulation. *Transportation Research Part B: Methodological*, 15(2):105–111.

[Graf et al., 2013] Graf, R., Deusch, H., Fritzsche, M., and Dietmayer, K. (2013). A learning concept for behavior prediction in traffic situations. In *Intelligent Vehicles Symposium (IV), 2013 IEEE*, pages 672–677. IEEE.

[Hayashi and Yamada, 2009] Hayashi, T. and Yamada, K. (2009). Predicting unusual right-turn driving behavior at intersection. In *Intelligent Vehicles Symposium, 2009 IEEE*, pages 869–874. IEEE.

[Helly, 1961] Helly, W. (1961). Simulation of bottlenecks in single-lane traffic flow. *Theory of Traffic Flow*, pages 207–238.

[Hermes et al., 2009] Hermes, C., Wöhler, C., Schenk, K., and Kummert, F. (2009). Long-term vehicle motion prediction. In *Intelligent Vehicles Symposium, 2009 IEEE*, pages 652–657. IEEE.

[Hülsen et al., 2011] Hülsen, M., Zöllner, J. M., and Weiss, C. (2011). Traffic intersection situation description ontology for advanced driver assistance. In *Proc. IEEE Intelligent Vehicles Symp. (IV)*, pages 993–999.

[Julier and Uhlmann, 1997] Julier, S. J. and Uhlmann, J. K. (1997). New extension of the kalman filter to nonlinear systems. In *AeroSense'97*, pages 182–193. International Society for Optics and Photonics.

[Kasper et al., 2011] Kasper, D., Weidl, G., Dang, T., Breuel, G., Tamke, A., and Rosenstiel, W. (2011). Object-oriented bayesian networks for detection of lane change maneuvers. In *Proc. IEEE Intelligent Vehicles Symp. (IV)*, pages 673–678.

[Koller and Pfeffer, 1997] Koller, D. and Pfeffer, A. (1997). Object-oriented bayesian networks. In *Proceedings of the Thirteenth conference on Uncertainty in artificial intelligence*, pages 302–313. Morgan Kaufmann Publishers Inc.

[Kschischang et al., 2001] Kschischang, F. R., Frey, B. J., and Loeliger, H.-A. (2001). Factor graphs and the sum-product algorithm. *Information Theory, IEEE Transactions on*, 47(2):498–519.

[Lefèvre et al., 2011] Lefèvre, S., Ibañez-Guzmán, J., and Laugier, C. (2011). Context-based estimation of driver intent at road intersections. In *Computational Intelligence in Vehicles and Transportation Systems (CIVTS), 2011 IEEE Symposium on*, pages 67–72. IEEE.

[Lefèvre et al., 2012] Lefèvre, S., Laugier, C., and Ibañez-Guzmán, J. (2012). Risk assessment at road intersections: comparing intention and expectation. In *Intelligent Vehicles Symposium (IV), 2012 IEEE*, pages 165–171. IEEE.

[Lewis, 1982] Lewis, C. (1982). *Using the" thinking-aloud" method in cognitive interface design*. IBM TJ Watson Research Center.

[Lidstrom and Larsson, 2008] Lidstrom, K. and Larsson, T. (2008). Model-based estimation of driver intentions using particle filtering. In *Intelligent Transportation Systems, 2008. ITSC 2008. 11th International IEEE Conference on*, pages 1177–1182. IEEE.

[Liebner et al., 2012] Liebner, M., Baumann, M., Klanner, F., and Stiller, C. (2012). Driver intent inference at urban intersections using the intelligent driver model. In *Proc. IEEE Intelligent Vehicles Symp. (IV)*, pages 1162 – 1167.

[Meyer-Delius et al., 2009] Meyer-Delius, D., Plagemann, C., and Burgard, W. (2009). Probabilistic situation recognition for vehicular traffic scenarios. In *Proc. IEEE Int. Conf. Robotics and Automation ICRA '09*, pages 459–464.

[Molinero Martinez et al., 2008] Molinero Martinez, A., Carter, E., Naing, C., Simon, M., and Hermitte, T. (2008). Accident causation and pre-accidental driving situations. part 1: Overview and general statistics. Technical report, EU-Project TRACE.

[Murphy, 2002] Murphy, K. P. (2002). *Dynamic bayesian networks: representation, inference and learning*. PhD thesis, University of California.

[Norris, 1998] Norris, J. R. (1998). *Markov chains*. Number 2008. Cambridge university press.

[Panda3D, 2014] Panda3D (2014). https://www.panda3d.org/. Accessed at 9 January 2014.

[Panwai and Dia, 2005] Panwai, S. and Dia, H. (2005). Comparative evaluation of microscopic car-following behavior. *IEEE Transactions on Intelligent Transportation Systems*, 6(3):314–325.

[Pearl, 1988] Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann.

[Petrich et al., 2013] Petrich, D., Dang, T., Kasper, D., Breuel, G., and Stiller, C. (2013). Map-based long term motion prediction for vehicles in traffic environments. In *Proc. 16th Int Intelligent Transportation Systems (ITSC) IEEE Conf.*

[Platho and Eggert, 2012] Platho, M. and Eggert, J. (2012). Deciding what to inspect first: Incremental situation assessment based on information gain. In *Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on*, pages 888–893. IEEE.

[Platho et al., 2012] Platho, M., Groß, H.-M., and Eggert, J. (2012). Traffic situation assessment by recognizing interrelated road users. In *Proc. IEEE Intelligent Transportation Systems (ITSC).*

[Platho et al., 2013a] Platho, M., Gros, H.-M., and Eggert, J. (2013a). Predicting velocity profiles of road users at intersections using configurations. In *Intelligent Vehicles Symposium (IV), 2013 IEEE*, pages 945–951. IEEE.

[Platho et al., 2013b] Platho, M., Gross, H.-M., and Eggert, J. (2013b). Learning driving situations and behavior models from data. In *Intelligent Transportation Systems - (ITSC), 2013 16th International IEEE Conference on*, pages 276–281.

[PTV, 2014] PTV (2014). Ptv vissim. Accessed at 16 January 2014.

[Quadstone, 2014] Quadstone (2014). Quadstoneparamics. Accessed at 16 January 2014.

[Rebhan et al., 2009] Rebhan, S., Richter, A., and Eggert, J. (2009). Demand-driven visual information acquisition. In *Computer Vision Systems*, pages 124–133. Springer.

[Reichart, 2001] Reichart, G. (2001). *Menschliche Zuverlässigkeit beim Führen von Kraftfahrzeugen.* Fortschritt-Berichte: Mensch-Maschine-Systeme. VDI-Verlag.

[Reichel et al., 2010] Reichel, M., Botsch, M., Rauschecker, R., Siedersberger, K., and Maurer, M. (2010). Situation aspect modelling and classification using the scenario based random forest algorithm for convoy merging situations. In *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*, pages 360–366.

[Risto and Martens, 2013] Risto, M. and Martens, M. H. (2013). Centre for transport studies, university of twente po box 217, 7500 ae, enschede, the netherlands. In *Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on*, pages 1923–1928. IEEE.

[Russell et al., 1995] Russell, S., Binder, J., Koller, D., and Kanazawa, K. (1995). Local learning in probabilistic networks with hidden variables. In *IJCAI*, volume 95, pages 1146–1152. Citeseer.

[Russell and Norvig, 2003] Russell, S. and Norvig, P. (2003). *Artificial Intelligence: A Modern Approach (2nd Edition)*. Prentice Hall, 3 edition.

[Schaaf, 1997] Schaaf, J. W. (1997). *Über die Suche nach situationsgerechten Fällen im fallbasierten Schließen*. Infix.

[Schamm and Zöllner, 2011] Schamm, T. and Zöllner, J. M. (2011). A model-based approach to probabilistic situation assessment for driver assistance systems. In *Proc. 14th Int Intelligent Transportation Systems (ITSC) IEEE Conf*.

[Schubert et al., 2010] Schubert, R., Schulze, K., and Wanielik, G. (2010). Situation assessment for automatic lane-change maneuvers. *IEEE Transactions on Intelligent Transportation Systems*, 11(3):607–616.

[Schürmann, 1996] Schürmann, J. (1996). *Pattern classification: a unified view of statistical and neural approaches*. John Wiley & Sons, Inc.

[Shannon and Weaver, 1949] Shannon, C. and Weaver, W. (1949). *The Mathematical Theory of Information*. University of Illinois Press.

[Shaout et al., 2011] Shaout, A., Colella, D., and Awad, S. (2011). Advanced driver assistance systems - past, present and future. In *Computer Engineering Conference (ICENCO), 2011 Seventh International*, pages 72–82.

[Sridharan et al., 2012] Sridharan, S., Shweta, V., and Manglik, A. (2012). Caidm: Context aware intelligent driver model. *International Journal of Soft Computing*, 7(3):113–119.

[Tran and Firl, 2013] Tran, Q. and Firl, J. (2013). Modelling of traffic situations at urban intersections with probabilistic non-parametric regression. In *Intelligent Vehicles Symposium (IV), 2013 IEEE*, pages 334–339. IEEE.

[Treiber and Helbing, 2002] Treiber, M. and Helbing, D. (2002). Realistische mikrosimulation von strassenverkehr mit einem einfachen modell. In *16th Symposium Simulationstechnik ASIM*, volume 2002, page 80.

[Tsuruoka et al., 2009] Tsuruoka, Y., Tsujii, J., and Ananiadou, S. (2009). Stochastic gradient descent training for l1-regularized log-linear models with cumulative penalty. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 477–485. Association for Computational Linguistics.

[Vacek et al., 2007] Vacek, S., Gindele, T., Zöllner, J. M., and Dillmann, R. (2007). Situation classification for cognitive automobiles using case-based reasoning. In *Proc. IEEE Intelligent Vehicles Symp*, pages 704–709.

[von Eichhorn et al., 2013] von Eichhorn, A., Werling, M., Zahn, P., and Schramm, D. (2013). Maneuver prediction at intersections using cost-to-go gradients. In *Intelligent Transportation Systems - (ITSC), 2013 16th International IEEE Conference on*, pages 112–117.

[Vu et al., 2012] Vu, A., Ramanandan, A., Chen, A., Farrell, J. A., and Barth, M. (2012). Real-time computer vision/dgps-aided inertial navigation system for lane-level vehicle navigation. *Intelligent Transportation Systems, IEEE Transactions on*, 13(2):899–913.

[Wendler and Lenz, 1998] Wendler, J. and Lenz, M. (1998). Cbr for dynamic situation assessment in an agent-oriented setting. In *Proc. AAAI-98 Workshop on CBR Integrations. Madison (USA)*, pages 172–186.

[WxPython, 2014] WxPython (2014). http://wxpython.org/index.php. Accessed at 9 January 2014.

[Yao et al., 2013] Yao, W., Zhao, H., Bonnifait, P., and Zha, H. (2013). Lane change trajectory prediction by using recorded human driving data. In *Intelligent Vehicles Symposium (IV), 2013 IEEE*, pages 430–436. IEEE.

[Zahlmann et al., 2000] Zahlmann, G., Kochner, B., Ugi, I., Schuhmann, D., Liesenfeld, B., Wegner, A., Obermaier, M., and Mertz, M. (2000). Hybrid fuzzy image processing for situation assessment [diabetic retinopathy]. *Engineering in Medicine and Biology Magazine, IEEE*, 19(1):76–83.

[Zecha and Rasshofer, 2009] Zecha, S. and Rasshofer, R. (2009). Forschungsinitiative ko-fas: Neue perspektiven für die fahrzeugsicherheit. Technical report, VDI.

[Zhang, 2004] Zhang, H. (2004). The optimality of naive bayes. *A A*, 1(2):3.

[Zhao and Karypis, 2001] Zhao, Y. and Karypis, G. (2001). Criterion functions for document clustering: Experiments and analysis. *Machine Learning*.