Master Thesis 52

Optimization and Machine Learning in Dynamic Bayesian Networks for early and accurate Maneuver Recognition in Highway Traffic

Stevens RuiXi Wang

May 18, 2017

Prüfer: *Prof. Dr.-Ing. Frank Allgöwer*
Betreuer: *Dr. Galia Weidl*
M.Sc. Anne Romer

Institut für Systemtheorie und Regelungstechnik
Universität Stuttgart
Prof. Dr.-Ing. Frank Allgöwer

Acknowledgements

The present Master Thesis deals with the situation analysis for driver assistance systems and autonomous driving in highway traffic scenarios. The driving behavior of the surrounding traffic participants is analyzed in relation to the lane markings and to the own (EGO) vehicle. We use the theory of Bayesian Networks to classifier the traffic maneuvers. Supervised learning methods are implemented with regard to a big data set to determine the probability parameters of the developed Bayesian Networks models. According to the requirements of an accurate and early recognition, we extend the models using dynamic computation and modeling approaches to smooth the incoming measurement data and predict the further development. The work was performed in the department RD/FAF of Daimler AG (research and development) as well as at the Institute for Systems Theory and Automated Control (IST) of the University Stuttgart.

At this point, I would like to thank Prof. Dr. -Ing Frank Allgöwer for the supervision of the Master Thesis.
My special thanks go to my mentors Dr. Galia Weidl and Anne Romer, who took over the excellent support from Daimler AG and from the IST of the University Stuttgart. Their constant support contribute decisively to the success of this work.
I would like to thank my superiors at Daimler, Dr. Uwe Regensburger and Dr. Peter Hille, for supporting me in my work.
Furthermore, I want to thank my colleagues, Dr. Ralf Streiter, Dr. Dietmar Kasper, Wei Zhang, Markus Spoeri, Thomas Sherwin, Peter Hurt, Peter Zechel, Alexander Meisterknecht, Michael Sodamin, Ediz Herkert and Manuel Balisteri at Daimler AG. Their advices and willingness to help are always a great support to my work. I am grateful for the discussions and for all the time spent with them.
Finally, I would like to thank my parents and my sister Ida for their support over the years.

Stevens RuiXi Wang
Stuttgart, 2017

Abstract

Situation awareness and recognition of traffic maneuvers are key elements of advanced driver assistance and autonomous driving systems. This master thesis outlines the challenge of situation awareness with early and accurate recognition of traffic maneuvers. The thesis is a part of the EU-Research-Project AMIDST (Analysis of Massive Data STreams) [7], founded by the European Commission.
In particular, we investigate three different Bayesian Network models: the Static Bayesian Network, the Dynamic Bayesian Network and the Naive Bayesian Network, with regard to the sequential classification problem that the lane change and lane follow maneuvers can be correctly classified during the drive. On the other hand, we predict the situation features based on the motion trend in order to recognize the lane change maneuvers at an early stage such that the vehicle control system has enough time to perform necessary actions.
For the improvement of each model, we utilize supervised learning algorithms to determine the network parameters. By analyzing the situation features given in the collected data from the real highway traffic, we develop a suitable labeling concept to match the learning algorithms. Given the labeled ground-truth data set, we implement Expectation Maximization algorithm to learn the network parameters of the knowledge-based static model. We implement linear and logistic regression to predict the dynamic trend such that the network propagation is influenced in favor of the time gain. Furthermore, we extend the static model to a Dynamic Bayesian Network by adding temporal clones which represent certain variables at an earlier time point. The relationship between successive time steps are learned by use of the Adaptation approach. The resulted knowledge-based network models are compared to the Naive Bayesian Network which is a pure data-driven model.
The developed models are transferred to classifiers in the C-framework and deployed on the Linux platform for off-line evaluation as well as on the automotive platform of the experimental vehicle for on-line testing. Using the optimized and extended knowledge-based models, over 90% of the traffic maneuvers can be correctly recognized at an early stage, i.e. 1.1-1.5 seconds before the considered vehicle crosses the lane marking.

Contents

List of Figures

List of Tables

Nomenclature

Abbreviations

ABS	Anti-lock Braking System
ACC	Adaptive Cruise Control
ADAS	Advanced Driver Assistance Systems
AMIDST	Analysis of Massive Data STreams
BE	Bayesian Estimation
BN	Bayesian Network
BV	Image Processing (german: Bildverarbeitung)
CAN	Controller Area Network
CPD	Conditional Probability Distribution
CPT	Conditional Probability Table
DAG	Directed Acyclic Graph
DAS	Driver Assistance Systems
DBN	Dynamic Bayesian Network
D&C	Divide & Conquer computation strategy
EGO	EGO vehicle: The own vehicle
EM	Expectation Maximization (algorithm)
E-step	Expectation step
ESC	Electronic Stability Control
FOLLOW	Follow Maneuver
GPS	Global Positioning System
HDM	High Definition Digital Map
LC	Lane Change maneuver
L,R,G	Movement Direction (Left, Right, Straight)
LL,LR,LG,RL,RR,RG,GL,GR,GG	Relative Movement of a Vehicle Pair
LM	Lane Marking
LMC	Lane Marking Cross

LMT	Lane Marking Touch
MLE	Maximum Likelihood Estimation
M-step	Maximization step
NBN	Naive Bayesian Network
OOBN	Object-Oriented Bayesian Network
OBJ	Other (object) Vehicles in the Traffic
OONF	Object-Oriented Network Fragment
OEM	Original Equipment
ORIG	Original Bayesian Network Model
TCPD	Transitional Conditional Probability Distribution
TCPT	Transitional Conditional Probability Table
TCS	Traction Control System

Variables

a	Acceleration
$accuracy$	Variable representing accuracy
c_0	Curvature
c_1	Change of curvature
d	Data point
e	Evidence
$hypervolume$	Volume under the Pareto Frontier
i	count i
id	Identification index
n	count n
$param$	Variable representing parameter
$pa(A)$	Parent of a random variable
$\vec{r}(t)$	Curve vector
t	Variable representing time (step)
$timegain$	Variable representing timegain
$threshold$	Variable representing threshold
v	Velocity

x,y,z	Scalar (mathematical notation)
A	Network variable representing
$ALAT$	maximal exploitable lateral acceleration
$CRITCALITY$	Situation criticality (event variable)
D	Distance Constant
$\mathcal{D}$	Data set
$\mathbb{E}$	Expectation counts
$\mathcal{G}$	Bayesian Network structure
$HQMVT$	High Quality Movement (event variable)
$\mathcal{L}$	Links of a Bayesian Network
LE	Lateral Evidence (hypothesis variable)
LE_DBN	Dynamic extended fragment of Lateral Evidence (hypothesis variable)
$\mathcal{M}$	Network model including structure and probability parameters
$MESS$	Measurement Value
N	Counted number
$N(t)$	Tangent Unit
$OCCGRID$	Occupancy Grid (hypothesis variable)
$OLAT$	Lateral Offset
P	Probability distribution
$POSDESCR$	Position Description (event variable)
$QMVT$	Quality Movement (event variable)
$REAL$	Expected Measurement Value
REL_DYN	Longitudinal Relative Dynamic (hypothesis variable)
$SIGMA$	standard deviation
T	Prediction horizon (Time constant)
$T(t)$	Tangent Unit
$TRAJ$	Trajectory (hypothesis variable)
TTC	Time To Collision
TTD	Time To Disappear
TTE	Time To Enter
$TTLC$	Time To Lane Change
U	Variable denoting symmetric uncertainty

$\mathcal{U}$	Set of variables in a Bayesian Network
$VLAT$	Lateral Velocity
V_REL	Longitudinal relative velocity
X	State set X
X_REL	Longitudinal relative distance
Y	State set Y
Z	State set Z
β	Regression coefficient
Δs	Distance difference
Δt	Time difference
Δv	Velocity difference
ϵ	Deviance
ψ	Yaw angle
θ	Probability parameter
π	Parent configuration
Θ	Set of probability parameters

Unit

cm	centimeter
dm	decimeter
m	meter
m/s	meter per second
m/s^2	meter per second squared
mm	millimeter
ms	millisecond
s, sec	second
TB	Terabyte

Index

$_i$	numbering index i
$_\infty$	reference size

j	numbering index j
k	numbering index k
lat	lateral
m	numbering index m
max	maximum value
min	minimal value
n	numbering index n
pred	predicted value
ref	reference value
rel	relative value
real	expected value
s	stationary value

1 Introduction

1.1 Motivation

Situation awareness and prevision are the key elements of modern intelligent driving (according to [2]). Since more than 30 years automotive manufacturer consistently continues the development and improvement of driver assistance systems. Based on the idea that the best protection against accidents is to recognize and avoid them before they can occur, a series of assistance systems helping the driver to stabilize the vehicle in critical situations, such as the Anti-lock Braking System (ABS), Traction Control System (TCS) and the Electronic Stability Control (ESC), become the standard applications of the vehicles nowadays. The trend is further reinforced by the rapid progress in the microcomputer industry and by the use of environment sensor systems. The advanced driver assistance systems (ADAS) today are developed not only to improve the active safety, but also the driving comfort and the operational efficiency. Adaptive Cruise Control (ACC), for example, automatically ensures a safe distance from the object vehicle in the front by slowing down, if necessary, and accelerating, if the traffic situation permits. Lane Departure Warning, for example, warns the driver of unintentional lane changes, if the system detects that the own vehicle is drifting over the lane marking. For the future, the OEM manufacturers focus on the next development stage: the autonomous driving. Daimler AG and Bosch GmbH for example, announced a partnership this year (2017) in order to accelerate the development of self-driving cars. They aim for the target that fully-automated cars can be delivered "by the beginning of the next decade" [21].
Compared to the ADAS, a full autonomous driving system should be able to take over all the driving tasks, from the analysis of the traffic situation to the control of the lateral and longitudinal dynamics of the own vehicle, without intervention of a driver. The idea of autonomous driving exists since many years. However, there are several restrictions on the implementation. A crucial challenge is the adaptation of the cognitive abilities to the autonomous systems. For concentrated humans e.g., it is a simple task to

recognize the maneuver intentions of surrounding vehicles by observing their lateral and longitudinal motion trend. Similar to this human reasoning process, a self-driving car has to perceive the relevant information about the surroundings by environment sensing as well as accurately classify and interpret the situation at an early stage such that the control systems have enough time to perform the necessary actions. Finding and programming the most capable transfer functions involve one of the essential tasks of the modern autonomous engineering.
"Most tasks require a person or an automated system to reason: to take the available information and reach conclusions, both about what might be true in the world and about how to act" [15]. In this context, we investigate in this thesis a concept for situation analysis in the highway traffic combining the theory of Bayesian Networks and the methods of machine learning. Bayesian Networks belong to the probabilistic graphical models, which are conventional strategies representing the reasoning process. Based on the primary work prior and during the EU-project "Solution for maneuver recognition in highway traffic" (see [14], [24], [23] and the papers of the EU AMIDST project [27], [26], [17]), we aim in this master thesis both for accuracy and prevision in the developed models of maneuver recognition. The existing knowledge-based models are optimized by learning the particular network parameters. For this purpose, we use a labeled data set which is collected in real highway traffic on different highways in Europe. Furthermore, we take the trend analysis into account in order to obtain an even earlier recognition of lane change based on the dynamic progress of the driving process. We further investigate models which are solely constructed and adapted by the given observations in the data set, for the purpose, that the development process of the described conceptual approach for similar and also extended problems is general and systematically valid and independent from complex knowledge-based modeling. At the end, the most capable classifiers are implemented both on the Linux platform off-line and on-line in the experimental vehicle. The overall goal is the evaluation of the performance criteria: accuracy and timegain.

1.2 Outline

The master thesis starts with the theory of Bayesian Networks including the different modeling approaches and an investigation of suitable learning algorithms in Chapter 2.

For better understanding of the development and the usage of the network models, we introduce the technical background in Chapter 3 including the essential elements of modern intelligent vehicles as well as the essential aspects of situation awareness.
The most important aspects of the development environment are described in Chapter 4, which includes the computer systems in the experimental vehicle, the software architecture, the relevant input data, the concept of data labeling as well as the construction of the knowledge-based model. In this context, we especially clarify our definition of lane change maneuver which is an essential aspect for the formulation and the modeling of the problem domain as well as for the data labeling.
In Chapter 5, a procedure of learning static models is presented. We use multi-objective optimization to obtain an optimal initial (sigmoid) parameter set. Based on the optimal initial parameterization, we implement the EM algorithm to learn the conditional probability tables (CPTs) from to the collected data set.
Furthermore, we utilize trend analysis based on linear and logistic regression to improve the recognition timegain of the static models. The concept is explained in Chapter 6.
In Chapter 7, we use the dynamic extension of particular fragment of the static knowledge-based models, where temporal slices are added representing the different time steps. The relation between successive time steps is given by the transitional conditional probability distributions (TCPD) which is off-line adapted from the data set. The goal is primarily the smoothing of incoming data based on the data history in order to improve the accuracy of the developed classifiers.
In contrast to the model-based approaches we choose a data-based model, i.e. the Naive Bayesian Network (NBN). In Chapter 8 we explain the concept of constructing the Naive Bayesian Network.
In Chapter 9 the evaluation results of the best capable models are presented. Due to the performance criteria: accuracy and timegain, we compare the Static and Dynamic Bayesian Network models, where the parameters are learned using EM- or Adaptation-algorithm, to the original model which is constructed in the work of [14]. We find improvement in both the accuracy and timegain. We investigate the performance of the knowledge-based models by the use of linear and logistic regression which improve the timegain with decline in the accuracy. Furthermore, we evaluate the data-driven Naive Bayesian Network model which has better timegain than the knowledge-based models, but an unacceptable error rate in the recognition of FOLLOW

maneuvers.
In Chapter 10, we discuss two possible approaches for the transmission of the output of the network models to the control systems. Both concepts are based on the assessment of the situation criticality.
At the end, in Chapter 11, the work of this thesis is summarized according to the obtained results. Additionally, possible improvements and future development are discussed.

2 Theoretical Basis

For basic understanding of this thesis, relevant theories of the Bayesian Network including the Object-Oriented Bayesian Network, the Dynamic Bayesian Network. We investigate the learning algorithms to estimate the probability parameters (qualitative part) in the Bayesian Networks. The relevant learning algorithms are explained. Furthermore, basic concepts for the construction of the model structure (qualitative part) are described at the end of this chapter.

2.1 Bayesian Network

2.1.1 Construction

A Bayesian Network (BN) is a probabilistic graphical model $\mathcal{M} = (\mathcal{G}, \mathcal{P})$ consisting of network structure $\mathcal{G}$ and probability functions $\mathcal{P}$. Using both graph theory and probability theory it provides an effective and well founded framework for knowledge representation, prediction and reasoning under uncertainty. More details about the principles of probabilistic graphical models are given in the book [15]. In the following, the essential theories of the Bayesian Networks is introduced. The description follows the content of the books [15] and [12]. For better understanding, suitable examples and figures are presented.
The network structure of BN $\mathcal{G} = (\mathcal{U}, \mathcal{L})$ is a directed acyclic graph (DAG) which does not allow feedback cycles. It consists of a set of nodes representing the (discrete or continuous) random variables $\mathcal{U}$ and a set of directed edges (or links) $\mathcal{L}$ which denote the conditional dependencies between the variables.
Variables of a BN may represent observable quantities and features (evidence node), hidden relationships (latent node), event or hypothesis. A discrete variable contains a set of mutually exclusive states X which can be of different types (e.g. Boolean, numbered, labeled). Continuous variables are modeled using probability density function, e.g. Conditional Gaussian Distribution. They can be discretized and treated as discrete variables with

certain value range. A BN containing both discrete and continuous variables is called Hybrid Bayesian Network.
Variables are conditionally dependent if they are linked by a directed edge. If there is for example a link from variable A_1 to variable A_2, then A_1 is the parent of A_2, and A_2 is referred to as the child of A_1. The relationship between A_1 and A_2 is described by a conditional probability function $P(A_2|A_1)$. It returns for each value of its inputs the conditional probability distribution (CPD) as output. The CPD denotes the probabilities of the states of the child A_2 dependent to the probability distributions given in parent A_1. If the A_1 has no other parent nodes, it is specified by the prior probability distribution $P(A_1)$ representing the initial belief states of A_1 without any influences.

2.1.2 Conditional Probability Table

The CPD for discrete variables is expressed by the conditional probability table (CPT) which is the most frequently used conditional probability function for BN. CPT can be considered as parameter matrix. Each cell contains a probability parameter representing a state of the child variable given a specific configuration of the states of its parents. The number of cells can be determined by multiplying the number n of states of the child node and the number m of states of its parent nodes, i.e. $dim(CPT) = n \cdot m$. If the variable is continuous, the CPT contains a mean and a variance parameter for each discrete parent node and a regression coefficient for each continuous parent node [12].
The developed BN models in this thesis consist of sets of discrete hypothesis and event variables. We use learning algorithm to determine the CPTs, especially for the input hypothesis, in order to improve the performance of the developed BN models.

2.1.3 Chain Rule

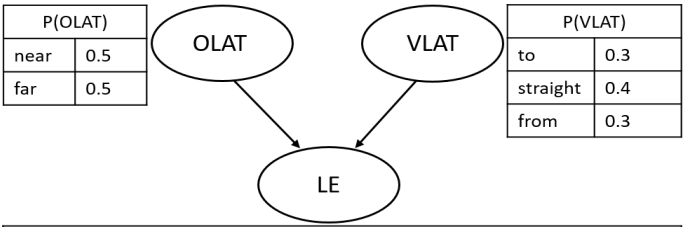
Given the local (conditional) probability distributions for each variable of a BN over $\mathcal{U} = (A_1, ..., A_n)$ the unique joint probabilty distribution representing the knowledge domain can be calculated using the Chain Rule [12]:

$$P(\mathcal{U}) = \prod_{i=1}^{n} P(A_i|pa(A_i)) \ , \tag{2.1}$$

where $pa(A_i)$ are the parents of the variable A_i in the BN structure $\mathcal{G}$. The computation of joint probability distribution is necessary step for further calculations.
A simple example, which is also relevant for this thesis, is given in Figure 2.1, where the output variable LE (denoting the lateral evidence) is conditionally dependent to the observations given in the variables $OLAT$ and $VLAT$ (denoting the relative distance and velocity to lane marking). Both parent nodes $OLAT$ and $VLAT$ are conditionally independent. The output variable LE is a Boolean variable with the states *true* and $false$. The variable $OLAT$ is modeled with the states *near* and far. The variable $VLAT$ contains the states *to*, *straight* and $from$ denoting the direction of the vehicle in relation to the lane marking. Furthermore, the prior probability distributions of the parent nodes $OLAT$, $VLAT$ and the CPT of the child node LE are given in Figure 2.1. We use the Chain Rule (Equation (2.1)) to compute the joint probability distribution:

$$P(OLAT, VLAT, LE) = P(OLAT) \cdot P(VLAT) \cdot P(LE|OLAT,VLAT) \,,$$

where every probability parameter given in the CPT of the child node LE is multiplied with the corresponding prior probabilities of the parent nodes. The computation is shown in Table 2.1. The computed joint probability distribution in Table 2.1 sums up to 1 ($\widehat{=}$100%).



P(OLAT)	
near	0.5
far	0.5

P(VLAT)	
to	0.3
straight	0.4
from	0.3

P(LE | OLAT,VLAT)						
OLAT	near			far		
VLAT	to	straight	from	to	straight	from
false	0	0.4	0.8	0.7	0.9	1
true	1	0.6	0.2	0.3	0.1	0

Figure 2.1: Conditional dependency between the information variables $OLAT$,$VLAT$ and the hypothesis variable LE

Table 2.1: Computation of the unique joint probability distribution $P(\mathcal{U}) = P(OLAT, VLAT, LE)$

$OLAT$	$near$		
$VLAT$	to	$straight$	$from$
$false$	$0.5 \cdot 0.3 \cdot 0 = 0$	$0.5 \cdot 0.4 \cdot 0.4 = 0.08$	$0.5 \cdot 0.3 \cdot 0.8 = 0.12$
$true$	$0.5 \cdot 0.3 \cdot 1 = 0.15$	$0.5 \cdot 0.4 \cdot 0.6 = 0.12$	$0.5 \cdot 0.3 \cdot 0.2 = 0.03$

$OLAT$	far		
$VLAT$	to	$straight$	$from$
$false$	$0.5 \cdot 0.3 \cdot 0.7 = 0.105$	$0.5 \cdot 0.4 \cdot 0.9 = 0.18$	$0.5 \cdot 0.3 \cdot 1 = 0.15$
$true$	$0.5 \cdot 0.3 \cdot 0.3 = 0.045$	$0.5 \cdot 0.4 \cdot 0.1 = 0.02$	$0.5 \cdot 0.3 \cdot 0 = 0$

2.1.4 Marginalization

The probability distribution $P(A_i)$ of a variable A_i is computed by summing out over irrelevant variables. This step is called marginalization (according to [12]) which is given by

$$P(A_i) = \sum_{\mathcal{U} \setminus |A_i|} \prod_{A_i} P(A_i | pa(A_i)) \; . \tag{2.2}$$

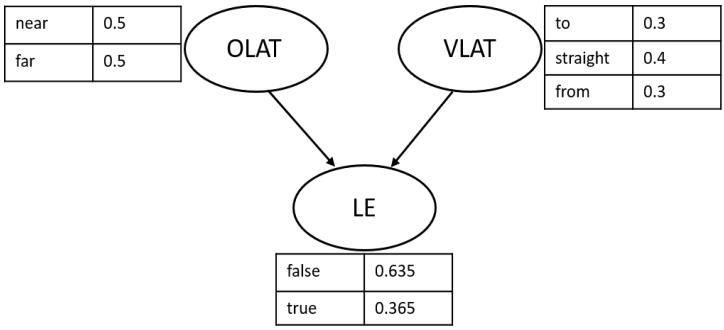


Figure 2.2: Probability distribution without inserting evidence

In the example given in Figure 2.1 we sum up the probability parameters in

the line denoting the states *true* and $false$ of the variable LE:

$$P(LE = false) = 0 + 0.08 + 0.12 + 0.105 + 0.18 + 0.15 = 0.635\ ,$$
$$P(LE = true) = 0 + 0.08 + 0.12 + 0.105 + 0.18 + 0.15 = 0.365\ ,$$

to compute the initial CPD (as shown in Figure 2.2) in the case that no evidence is given to the network.

2.1.5 Inserting Evidence

Assume a discrete hypothesis variable A (e.g. distance) with n states and the corresponding prior probability distribution $P(A) = (x_1,...,x_n)$. The evidence e (e.g. sensor measurement) is given as observation that A is only in the state i which is a so called hard evidence. The probability distribution is given as $P(A,e) = (0,...,x_i,...,0)$ which is the result of multiplying the prior probability distribution of $P(A)$ by $(0,...,1,...,0)$. In the case of an uncertain observation (e.g. uncertain measurement) we use the likelihood (soft) evidence, e.g. $(0, 0.2, 0.3, 0.5, 0,...)$ and we do the same multiplication. The prior probability distribution $P(e)$ is computed by marginalizing A out of $P(A,e)$ by (according to [12])

$$P(e) = \sum_A P(A,e) = P(x_1,e) + P(x_2,e) + ... + P(x_n,e)\ . \tag{2.3}$$

The posterior probability distribution of the variable A given the evidence e is calculated using the statement of the Bayes' theorem (according to [12]) which is given by

$$P(A|e) = \frac{P(e|A) \cdot P(A)}{P(e)} \tag{2.4}$$
$$= \frac{P(A,e)}{\sum_A P(A,e)}\ , \tag{2.5}$$

where $P(A|e)$ denotes posterior conditional probability for the variable A given the evidence e. $P(e|A)$ is the likelihood distribution denoting the probability distribution of the evidence e with fixed probability parameters of the variable A. $P(e)$ is the marginal likelihood which denotes the a-priori probability of observing the evidences given the prior beliefs [15].

In addition, if for example two variables A_1 and A_2 are independent given the evidence e, then the probability calculus must yield

$$P(A_1|e) = P(A_1|A_2,e) \ ,$$

according to the basic axioms of probability. More details are described in the books [12] and [15].
We again consider the example given in Figure 2.1: The node $VLAT$ (denoting the velocity vector) contains three states $(to, straight, from)$ with the corresponding prior probability distribution $(0.3, 0.4, 0.3)$. If an exact sensor measurement is given that the vehicle is moving in the direction of lane marking, we have the hard evidence $(1, 0, 0)$. In order to calculate the CPD $P(VLAT|e)$ for $VLAT$ given e we compute the joint probability distribution $P(VLAT,e)$ at first:

$$P(VLAT,e) = (0.3 \cdot 1, 0.4 \cdot 0, 0.3 \cdot 0) = (0.3, 0, 0) \ ,$$

then the prior probability $P(e)$:

$$P(e) = \textstyle\sum_{VLAT} P(VLAT,e) = 0.3 \cdot 1 + 0.4 \cdot 0 + 0.3 \cdot 0 = 0.3 \ ,$$

The CPD $P(VLAT|e)$ can be computed by normalization:

$$P(VLAT|e) = (\tfrac{0.3}{0.3}, \tfrac{0}{0.3}, \tfrac{0}{0.3}) = (1, 0, 0) \ .$$

In the case that the sensor measurement is not exact, we could get an evidence e as for example $(0.6, 0.3, 0.1)$. We again follow the presented computation steps:

$$\begin{aligned} P(VLAT,e) &= (0.3 \cdot 0.6, 0.4 \cdot 0.3, 0.3 \cdot 0.1) = (0.18, 0.12, 0.03) \ , \\ P(e) &= 0.3 \cdot 0.6 + 0.4 \cdot 0.3 + 0.3 \cdot 0.1 = 0.33 \ , \\ P(VLAT|e) &= (\frac{0.18}{0.33}, \frac{0.12}{0.33}, \frac{0.03}{0.33}) \approx (0.55, 0.36, 0.09) \ , \end{aligned}$$

to calculate $P(VLAT|e)$.

2.1.6 Bayesian Inference

Bayesian Inference is one of the fundamental methods for the interpretation of probability where probability is considered as a way to represent a degree of belief in a statement, or the degree of confidence given evidences. Based on this interpretation of probability there are different approaches within

Bayesian Inference. In the following the model-based Bayesian inference is represented which is derived from the Bayes' theorem.
By observing the statistically independent evidences $e_1,...,e_m$ for the variables of the model $\mathcal{U} = (A_1,...,A_n)$, we want to calculate the probability distribution of all dependent unobserved variables. Thus, we extend the Chain Rule in Equation (2.1) by multiplying the initial joint probability of the model $P(\mathcal{U})$ with the inserted evidence e (as described in [12]):

$$P(\mathcal{U},e) = P(\mathcal{U}) \cdot e = \prod_{i=1}^{n} P(A_i|pa(A_i)) \prod_{j=1}^{m} e_j \,. \tag{2.6}$$

The resulted joint probability distribution $P(\mathcal{U},e)$ is the updated summary of all probability parameters after inserting the evidences into the network. The posterior probability of a variable A_i can be computed by marginalizing it out of $P(\mathcal{U},e)$ [12]:

$$P(A_i|e) = \frac{\prod_{\mathcal{U}\setminus A_i} P(\mathcal{U},e)}{\sum_{\mathcal{U}} P(\mathcal{U},e)} \,, \tag{2.7}$$

which is also a general formulation of the Bayes' theorem.

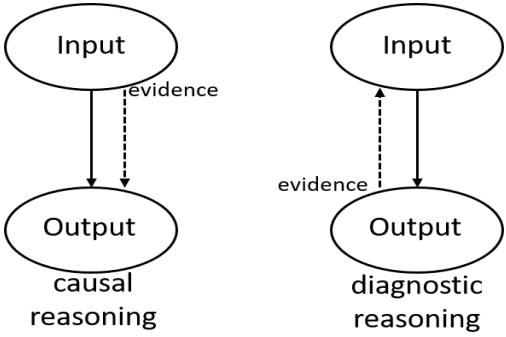


Figure 2.3: causal (left) and diagnostic (right) reasoning

Bayesian Inference can be implemented for two basic types of calculation: causal reasoning and diagnostic reasoning (as shown in Figure 2.3), which are relevant for this thesis. The first is top-down (from cause to effect) calculation

where the evidences are given to the parent node and the influences to its causal dependent child nodes should be calculated. The diagnostic reasoning denotes the bottom-up (from effect to cause) calculation where evidences are given to the child node and the probability distributions of related parent nodes are desired [12].
We consider again the example given in Figure 2.1. In this case the hard evidence $e = (1,0,0)$ is given for the states of $VLAT$. Using Equation (2.6) and Equation (2.7) we calculate the joint probability distribution $P(\mathcal{U},e)$ as shown in Table 2.2. We multiply the given evidence $e = (1,0,0)$ for $VLAT$ with corresponding probability parameters given in the Table 2.1 denoting the prior probability distribution $P(\mathcal{U})$.

Table 2.2: Joint probability distribution $P(\mathcal{U},e)$ after inserting the hard evidence $e = (1,0,0)$ to the states of $VLAT$

$OLAT$	$near$		
$VLAT$	to	$straight$	$from$
$false$	$(0) \cdot 1 = 0$	$(0.08) \cdot 0 = 0$	$(0.12) \cdot 0 = 0$
$true$	$(0.15) \cdot 1 = 0.15$	$(0.12) \cdot 0 = 0$	$(0.03) \cdot 0 = 0$

$OLAT$	far		
$VLAT$	to	$straight$	$from$
$false$	$(0.105) \cdot 1 = 0.105$	$(0.18) \cdot 0 = 0$	$(0.15) \cdot 0 = 0$
$true$	$(0.045) \cdot 1 = 0.045$	$(0.02) \cdot 0 = 0$	$(0) \cdot 0 = 0$

$P(\mathcal{U},e)$ can be utilized to (re)construct the probability distributions of the variables $OLAT$, $VLAT$ and LE by marginalizing every single of them. We first sum up the probability parameters in Table 2.2 denoting the joint probability distribution:

$$P(e) = \sum_{\mathcal{U}} P(\mathcal{U},e) = 0 + 0.15 + 0.105 + 0.045 + 0 + ... + 0 = 0.3 \,.$$

Then we determine the (conditional) probability distribution of every state by marginalizing it. We compute for the states of the node $OLAT$:

$$P(OLAT = near|e) = \sum_{OLAT=near} P(\mathcal{U},e) = \frac{0 + 0.15 + 0 + 0 + 0 + 0}{0.3} = 0.5 \,,$$

$$P(OLAT = far|e) = \sum_{OLAT=far} P(\mathcal{U},e) = \frac{0.105 + 0.045 + 0 + 0 + 0 + 0}{0.3} = 0.5 \,,$$

and for the states of the node $VLAT$:

$$P(VLAT = to|e) = \sum_{VLAT=to} P(\mathcal{U},e) = \frac{0+0.15+0.105+0.045}{0.3} = 1\,,$$

$$P(VLAT = straight|e) = \sum_{VLAT=straight} P(\mathcal{U},e) = \frac{0+0+0+0}{0.3} = 0\,,$$

$$P(VLAT = from|e) = \sum_{VLAT=from} P(\mathcal{U},e) = \frac{0+0+0+0}{0.3} = 0\,,$$

and for the states of the node LE:

$$P(LE = false|e) = \sum_{LE=false} P(\mathcal{U},e) = \frac{0+0+0+0.105+0+0+0}{0.3} = 0.35\,,$$

$$P(LE = true|e) = \sum_{LE=true} P(\mathcal{U},e) = \frac{0.15+0+0+0.045+0+0}{0.3} = 0.65\,.$$

The resulted conditional probability distribution is shown in Figure 2.4

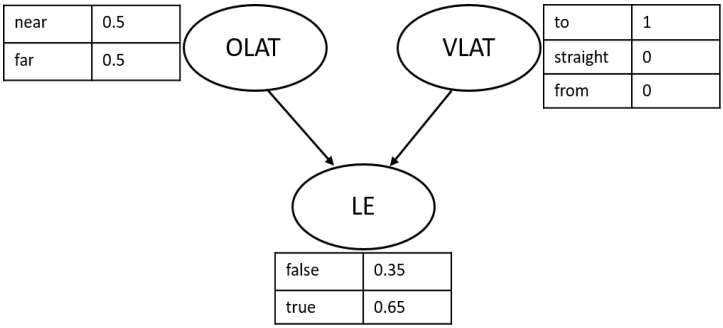


Figure 2.4: Probability distribution after inserting the hard evidence $e = (1,0,0)$ to the states of $VLAT$

In the case of diagnostic reasoning, we take for example the assumption that LE is true. Thus, we insert the hard evidence $e = (0,1)$ to the states of LE. For the calculation of the (conditional) probability distribution we follow the same computation steps before. We use the Chain Rule by inserting the evidence 2.6: $P(\mathcal{U},e) = P(\mathcal{U}) \cdot e$ (as shown in Table: 2.3).

Table 2.3: Joint probability distribution $P(\mathcal{U},e)$ in the case of diagnostic reasoning by inserting the hard evidence $e = (0,1)$ to the states of LE

$OLAT$	$near$		
$VLAT$	to	$straight$	$from$
$false$	$(0) \cdot 0 = 0$	$(0.08) \cdot 0 = 0$	$(0.12) \cdot 0.8 = 0$
$true$	$(0.15) \cdot 1 = 0.15$	$(0.12) \cdot 1 = 0.12$	$(0.03) \cdot 1 = 0.03$

$OLAT$	far		
$VLAT$	to	$straight$	$from$
$false$	$(0.105) \cdot 0 = 0$	$(0.18) \cdot 0 = 0$	$(0.15) \cdot 0 = 0$
$true$	$(0.045) \cdot 1 = 0.045$	$(0.02) \cdot 1 = 0.02$	$(0) \cdot 1 = 0$

Using $P(\mathcal{U},e)$ we compute the probability distribution of the parent nodes. We again sum up $P(\mathcal{U},e)$:

$$\sum_{\mathcal{U}} P(\mathcal{U},e) = 0.15 + 0.12 + 0.03 + 0.045 + 0.02 + 0 = 0.365 \,,$$

Then we marginalize the states of the node $OLAT$:

$$\sum_{OLAT=near} P(\mathcal{U},e) = \frac{(0 + 0.15 + 0 + 0.12 + 0 + 0.03)}{0.365} \approx 0.8219 \,,$$
$$\sum_{OLAT=far} P(\mathcal{U},e) = \frac{(0 + 0.045 + 0 + 0.02 + 0 + 0)}{0.365} \approx 0.1781 \,,$$

and the states of the node $VLAT$:

$$\sum_{VLAT=to} P(\mathcal{U},e) = \frac{(0 + 0.15 + 0 + 0.045)}{0.365} \approx 0.5342 \,,$$
$$\sum_{VLAT=straight} P(\mathcal{U},e) = \frac{(0 + 0.12 + 0 + 0.02)}{0.365} \approx 0.3836 \,,$$
$$\sum_{VLAT=from} P(\mathcal{U},e) = \frac{(0 + 0.03 + 0 + 0)}{0.365} \approx 0.0822 \,.$$

Also the states of LE can be reconstructed:

$$\sum_{LE=true} P(\mathcal{U},e) = \frac{(0.15+0.12+0.03+0.045+0.02+0)}{0.365} = 1\,,$$
$$\sum_{LE=false} P(\mathcal{U},e) = \frac{(0+0+0+0+0+0)}{0.365} = 0\,.$$

The resulted (conditional) probability distribution is illustrated in Figure 2.5.

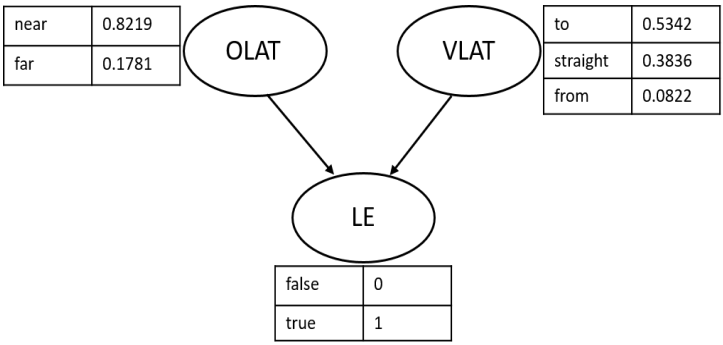


Figure 2.5: Probability distribution after inserting the hard evidence $e = (0,1)$ to the states of LE

2.2 Object-Oriented Bayesian Network

Object-Oriented Bayesian Network (OOBN) can be considered as a special modeling language combining Bayesian Network with logic-programming-like rules. Complex domains with identical or similar properties are modeled as inter-related and generic objects, also called Object-Oriented Network Fragments (OONFs). Each OONF is a separated instance which is reusable for different modelling approaches. Simple OONFs can be encapsulated into a more complex OONF within the same model. Using the OONFs and additional logic layers the BN itself can be built in hierarchical fashion. This structural organization makes the representation of complex knowledge using BN clear, efficient and flexible [16].

A simple example is given in Figure 2.6 where the BN model of Figure 2.1 is summarized as a single OONF which can be used as evidence for upper layers in a global logical framework.

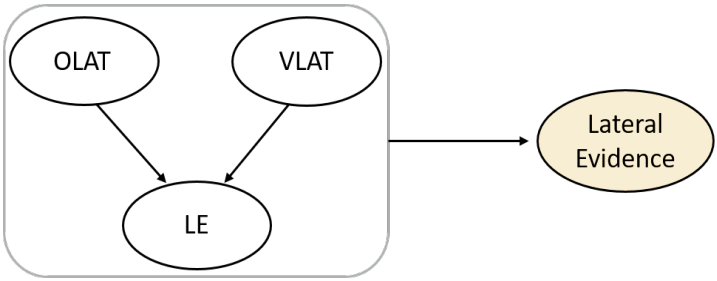


Figure 2.6: Object-Oriented Fragment denoting the Lateral Evidence which contains the information variables: $OLAT$ (Lateral Offset), $VLAT$ (Lateral Velocity), and the hypothesis variable: LE

The knowledge-based default model used in this thesis (original OOBN) is built up using three OONFs: LE, TRAJ and OCCGRID. The LE fragment is similar to the example given in Figure 2.1. It represents the lateral dynamics of vehicle using the lane marking as reference. TRAJ contains situation features which are derived from predicted lane change trajectory. OCCGRID is the free space description between vehicles in the target lane. These three basic OONFs are linked to the event variable LC (Lane Change) which indicates the lane change probability. In higher logic layers above the LC we formulate the vehicle-vehicle-relationship. We calculate the relative movement and the relative position of every considered vehicle pair in order to determine the probability distribution for the defined problem domain based on maneuver classes: OBJCUTIN, OBJCUTOUT, EGOCUTIN, EGOCUTOUT, OBJFOLLOW, LANEFOLLOW, DONTCARE. The modeling concept as well as the used OONFs are described in Chapter 4 in detail.

2.3 Dynamic Bayesian Network

Dynamic Bayesian Network (DBN) is extension of static BN by adding time slices. Each time slice represents an ordinary static BN at a single time instant. A DBN is characterized by linking a number of time slices using temporal nodes and links which denote the relationship of certain variable at different time points. It allows to follow the development of a process or event over time. The network structure of a DBN is fixed and doesn't change dynamically. Only the parameters change in the light of observations from different time points [15].
DBNs can be used to describe dynamic systems such as discrete-time stochastic processes with input, hidden and output variables. In the context of this thesis, we use temporal evidences which consist of time series where sensor measurements are recorded sequentially in time (e.g. camera data are recorded once every $40 - 60$ ms). We add a variable representing a certain sensor recording more than one time to the network to model the states of the variable at current, past and future. Past and future are represented by the temporal clones of the variable. Temporal clones can appear more than once for the same variable if the associated times are different, e.g. for $t-1, t-2, t-3, ...$ in the past. The output variable of a DBN can be prediction (future), filtering (current) or smoothing (past) of the input evidence. We also use hypothesis variable as output which represents the propagation result of different input evidences and their temporal clones. An example is given in Figure 2.7, where the example given in Figure 2.1 is extended to DBN by adding temporal nodes to both the distance $OLAT$ and velocity $VLAT$. The variables $T0.OLAT$ and $T0.VLAT$ denote the recorded data of $OLAT$ and $VLAT$ in the previous time step.
In order to specify the propagation between the time series, we consider a DBN that satisfies the first-order Markov assumption which denotes the conditional independence between future and past given the presence, i.e. $(A^{t+1} \perp A^{(0:t-1)} | A^t)$ where $(0 : t-1)$ denotes the time slices from time step 0 to time step $t-1$ and the time step t represents the presence. Thus the simplified representation of the joint distribution over $A^{(0:T)}$ can be formulated as [15]:

$$P(A^{(0:T)}) = \prod_{t=0}^{T-1} P(A^{t+1} | A^t) \, ,$$

where $P(A^{t+1}|A_t)$ denotes the transition model of the variable A from

the time step t to the next time step $t+1$. This transition model can be considered as the CPD between the states of A at time t conditioned to its parent which is the former time slice of A. Thus, we use the so-called transitional conditional probability table (TCPT) to specify the relation of the same variable between successive time steps. The resulted joint distribution $P(A^{(0:T)})$ contains information of the variable A of the entire time series from 0 to T.
A important task of this thesis is to learn the TCPTs in the developed DBN model. We create temporal clones of the input information variables in the LE fragment (similar to the example given in Figure 2.7). The relationship between the input information variable and their temporal clones should be learned from the collected data set using a suitable learning algorithm.

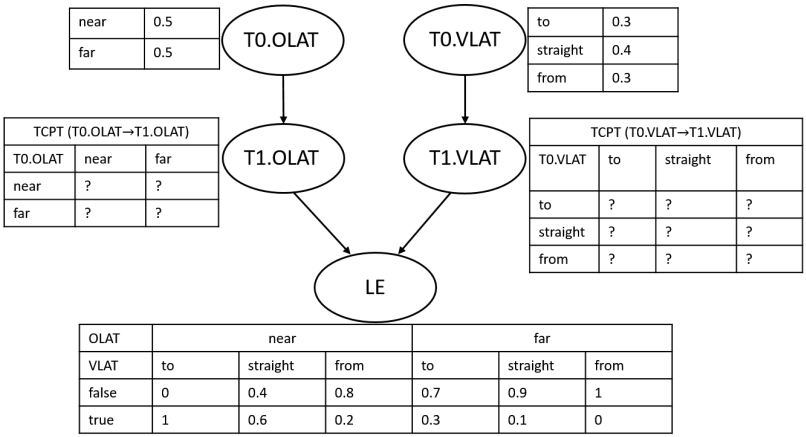


Figure 2.7: Extension of static BN to DBN using temporal cloned nodes of $T0.VLAT$, $T0.OLAT$, $T1.VLAT$, $T1.OLAT$ representing the same variables $OLAT$ and $VLAT$ of the previous time step $T0$ and the current time step $T1$

2.4 Learning Algorithms for Parameter Estimation in a Bayesian Networks

We investigate supervised learning algorithms for parameter estimation in the developed Bayesian Network models. Related literatures are e.g. [18], [22], [12], [15], [9], [11], [5] and also within the AMIDST project: [17]. In this chapter the essential learning algorithms are described.
In general, the methods for parameter estimation of a BN can be assigned to four situations [8]:

1. Network structure is known and the data set is complete.
2. Network structure is known and the data set is incomplete.
3. Network structure is unknown and the data set is complete.
4. Network structure is unknown and the data set is incomplete.

In situations one and two the graphical model $\mathcal{M} = (\mathcal{G}, \Theta)$ is given with structure $\mathcal{G}$ and probability parameters Θ. We use learning algorithms to estimate the network parameters from a complete or a incomplete data set $\mathcal{D}$. A complete data set usually denotes a matrix without missing entries, i.e. each variable is specified by a value for each case. In situations three and four the network structure is missing and needs to be specified at first. We need to select the set of network variables $\mathcal{U}$ from all available input variables as well as learn the strength of the causal relationship between them can be learned from the given complete or incomplete data set.
Common learning methods for Bayesian Networks are the Maximum Likelihood Estimation (MLE) and the Bayesian Estimation (BE). Both methods are based on the general formulation of the Bayes' Theorem [15]:

$$posterior = \frac{likelihood \cdot prior}{evidence} ,$$

which is

$$P(\Theta|\mathcal{D}) = \frac{P(\mathcal{D}|\Theta) \cdot P(\Theta)}{P(\mathcal{D})} . \tag{2.8}$$

where MLE maximizes the likelihood and BE fully calculates the posterior distribution. We assume in this case, that the network structure $\mathcal{G}$ is given and the prior probability $P(\Theta)$ for each probability parameter $\theta \in \Theta$ is

declared. Moreover, evidence $P(\mathcal{D})$ is the marginal likelihood of the data, i.e. the a-priori probability of seeing the data given the prior probability distribution (as described in Section 2.1.5) [15].
In this thesis, we use a particular labeling method such that the cases in the given data set is not completely specified. Thus, we need to take the missing the values into account. A common learning algorithm to cope with an incomplete data set is the Expectation Maximization algorithm, which fills the empty spaces with expectation values based on the current probability parameters and optimizes the outcomes using MLE or BE. Another possible approach is the Adaptation where the probability parameters are updated sequentially by each observed data. Both learning algorithms are suitable to cope with a massive data stream.

2.4.1 Maximum Likelihood Estimation

Maximum Likelihood Estimation (MLE) is a common algorithm for estimating the parameters in the case of given network structure and a complete data.
Assume given the model $\mathcal{M}$ and a data set of independent and identically distributed observations $\mathcal{D}$, then the likelihood of $\mathcal{M}$ given $\mathcal{D}$ is defined as [12]:

$$L(\mathcal{M}|\mathcal{D}) = \prod_{d \in \mathcal{D}} P(d|\mathcal{M}) \,, \tag{2.9}$$

where $P(d|\mathcal{M})$ is the likelihood of $\mathcal{M}$ given the single case $d \in \mathcal{D}$.
The principle of MLE is to choose the parameters set $\hat{\Theta}$ in the fixed model $\mathcal{M}$ such that the likelihood is maximized for the given data set $\mathcal{D}$. For the computation we usually maximize the log-likelihood by (according to [12])

$$\hat{\Theta} = \underset{\Theta}{\operatorname{argmax}} \log L(\Theta|\mathcal{D}) = \underset{\Theta}{\operatorname{argmax}} \sum_{d \in \mathcal{D}} \log P(d|\Theta) \,. \tag{2.10}$$

In a binomial case using binary states $X = (true, false)$ for example, we have the Bernoulli Distribution :

$$P(X|\theta) = \begin{cases} \theta & X = true \\ 1 - \theta & X = false \end{cases} .$$

For a data set $\mathcal{D} = (true, false, false, true, true, ...)$, where n cases are $true$ and m cases are $false$, the likelihood is given by

$$L(\theta|\mathcal{D}) = \theta^n (1 - \theta)^m \,.$$

We maximize the log-likelihood:

$$\log L(\theta|\mathcal{D}) = n \cdot \log(\theta) + m \cdot \log(1-\theta) \,,$$
$$\frac{d \log L(\theta|\mathcal{D})}{d\theta} = \frac{n}{\theta} - \frac{m}{1-\theta} = 0 \,,$$
$$\hat{\theta} = \frac{n}{n+m} \,.$$

In general case with K states where $X = (x_1, ..., x_k)$ and $\Theta = (\theta_1, ..., \theta_k)$ we compute for each parameter $\theta_k \in \Theta$

$$\hat{\theta}_k = \frac{N_k}{\sum_{k=1}^{K} N_k} \,, \tag{2.11}$$

using the counted number N_k of observations for the state k [12]. Thus, the maximum likelihood is computed in general as the fraction of counted observations of a particular state given for a parent configuration over the total number of the cases that this parent configuration is given in the data.

2.4.2 Bayesian Estimation

An alternative method for parameter estimation given a complete data set is Bayesian Estimation (BE) which is very similar to MLE and converges to the same values given a large number of cases. The difference is, that we maximize the posterior instead of the likelihood function. A frequently used approach is to give the prior as Dirichlet distribution which is specified by a set of hyperparameters $\alpha_1, ..., \alpha_k$. Thus, the posterior is also given as Dirichlet distribution [15]:

$$P(\Theta|\mathcal{D}) \propto \frac{1}{K} \prod_{k=1}^{K} \theta_k^{\alpha_k + n_k - 1} \,,$$

which is in this case very similar to the likelihood function (as shown in Equation (2.9)). The hyperparameter α_k is usually given as one more than the count of k_{th} outcome. We add them to the outcome Equation (2.11) [15]:

$$\hat{\theta} = \frac{N_k + \alpha_k}{\sum_{k=1}^{K} (N_k + \alpha_k)} \,. \tag{2.12}$$

An important advantage of BE is that the virtual counts (hyperparameters) prohibit the probable zero counts which cause zero probability parameters as outcome [12].

For the binary problem given in last subsection 2.4.1 e.g., we obtain

$$\hat{\theta} = \frac{n+1}{n+m+2},$$

as outcome if using Bayesian Estimation [12].

2.4.3 Expectation Maximization Algorithm

In the practice, the data set used for learning is often not complete. Some measurement data are missing or cannot be observed. Futhermore, values in the data matrix can be intentionally removed by use of labeling method to specify particular situations. We consider in this subsection the Expectation Maximization Algorithm (EM algorithm) which is a frequently used approach for parameter learning given a incomplete data set. We use for this thesis the "EM algorithm for graphical association models with missing data" which is developed by Steffen Lauritzen [18]. The algorithm is explained in the following based on example.
The basic concept of the EM algorithm is the iteration between the expectation (E-step) and maximization step (M-step). At first we define a stop criterion (maximum number of iterations or convergence threshold) and set the starting time $t := 0$. At each iteration step, expectation counts $\mathbb{E}[N(A_i, pa(A_i)|\mathcal{D}, \Theta^t)]$ of particular configurations are determined from the data $\mathcal{D}$ using the parameters Θ^t at current time t in the E-step and added to both the denominator and numerator of the maximization equation in the M-step. The new probability parameters are computed by maximizing either the likelihood function (MLE, see Subsection 2.4.1) or the a posterior distribution (BE, see Subsection 2.4.2). The iteration keeps running until the predefined stop criterion is reached. As precondition, the model structure $\mathcal{G}$ and the initial parameter set Θ^0 must be given over the variables $\mathcal{U} = A_1, ..., A_n$. We estimate the probability parameters $\theta_{ijk} \in \Theta$ corresponding to the conditional probability $P(A_i = k|pa(A_i) = j)$, i.e., the conditional probability for variable $A_i \in \mathcal{U}$ being in its kth state given the jth configuration of the its parents $pa(A_i) \in \mathcal{U}$. We find the parameters $\hat{\theta}_{ijk}$ which maximize the likelihood (MLE) or the posterior (BE) for the given data set $\mathcal{D} = d_1, ..., d_m$. A compact description of the EM algorithm used in this thesis is given as followed which is taken from the book [12] (Chapter 6, Parameter Estimation on page 224).

1. Choose a stopping criterion, e.g. the difference of log-likelihood between two iteration steps: $\epsilon > 0$

2. Let $\theta_{ijk} \in \Theta^0$, where $1 \leq i \leq n$, $1 \leq k \leq |\operatorname{sp}(A_i)| - 1$, and $1 \leq j \leq |\operatorname{sp}(pa(A_i))|$, be some initial estimates of the parameters (chosen arbitrarily).

3. Set t:=0.

4. Repeat:
   E-step: For each $1 \leq i \leq n$ calculate the table of expected counts:

$$\mathop{\mathbb{E}}_{\Theta^t}[N(A_i, pa(A_i)|\mathcal{D}, \Theta^t)] = \sum_{d \in \mathcal{D}} P(A_i, pa(A_i)|d, \Theta^t) \ .$$

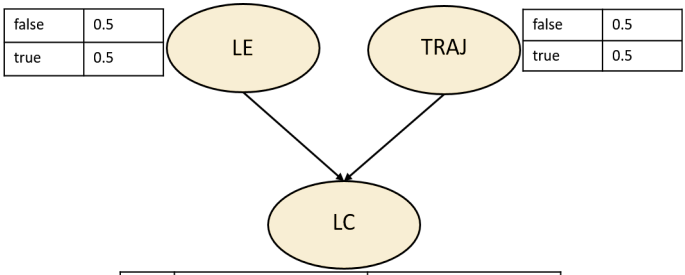
   M-step: Use the expected counts as if they were actual counts to calculate a new maximum likelihood estimate for all $\theta_{ijk} \in \Theta^t$:

$$\hat{\theta}_{ijk} = \frac{\mathbb{E}_{\Theta^t}[N(A_i = k, pa(A_i) = j)|\mathcal{D}]}{\sum_{h=1}^{|\operatorname{sp}(A_i)|} \mathbb{E}_{\Theta^t}[N(A_i = k, pa(A_i) = j|\mathcal{D})]} \ .$$

   Set $\Theta^{t+1} := \hat{\Theta}^t$ and $t := t + 1$.
   Until $|\log P(\mathcal{D}|\Theta^t) - \log P(\mathcal{D}|\Theta^{t-1})| \leq \epsilon$

Before running the algorithm we have to specify the set of variables for which conditional probability distributions and densities should be estimated. Thus, we add an experience table to each of those variables in order to store the experience counts, i.e. the number of times that a particular parent configuration given in the CPTs is observed or "expected". Furthermore, we need to optimize the initial parameter set since the EM algorithm does not guarantee the convergence to the global optimum.
For better understanding of the EM algorithm, we consider a simple example based on the lane change prediction model used in this thesis. The model $\mathcal{M}$ is shown in Figure 2.8. It contains two input variables: *LE* and *TR*, and one output variable: *LC*. *LE* and *TR* are the computed likelihoods of the corresponding BN-fragments of the input layer which represent a summary of the observed evidences. *LC* represents the lane change probability. A detailed description about this knowledge-based BN-model with all of its fragments is given in Chapter 4.
Using the training data set $\mathcal{D}$ (as shown in Table 2.4) with missing values and the initial default parameters given in the CPT of *LC*, we compute one iteration to update the probability parameters in the CPT. The training

data set of this example is labeled to determine the conditional probability parameter for the particular parent configuration that $LE = true$ and $TR = false$. We compute the conditional probability for $LC = true$ whereas the state $LC = false$ is the complementary probability. The prior probability distributions of LE and TR stay equally distributed and should not be changed.



LE	false		true	
TRAJ	false	true	false	true
false	0	0.3	0.7	1
true	1	0.7	0.3	0

Figure 2.8: Simplified LC-Model

LE	$TRAJ$	LC
$false$	$false$	$false$
$true$	$false$	?
$true$	$false$	?
$true$	$false$	?
$true$	$false$	$true$
$true$	$false$	$true$
$true$	$false$	$true$
$true$	$true$	$true$

Table 2.4: Training data set

In the E-step we calculate the expected counts $\mathbb{E}[N(LC = true, LE = true, TR = false)]$. Missing values in the labeling of LC are filled using

the current corresponding parameter, 0.7 in this case, for the configuration $LC = true, LE = true, TR = false$. Given a corresponding labeling ($LC = true$) in the data we count $+1$, otherwise we count $+0$. We count for the training data set (Table 2.4) given in the example:

$$\begin{aligned}
&\mathbb{E}[N(LC = true, LE = true, TR = false)|\mathcal{D},\theta^0] \\
&= \sum_{i=1}^{N} P_0(LC = true, LE = true, TR = false)|\mathcal{D},\theta^0) \\
&= 0 + 0.7 + 0.7 + 0.7 + 1 + 1 + 1 + 0 \\
&= 5.1 \ .
\end{aligned}$$

Furthermore, we determine the denominator by counting the frequency of seeing the parent configuration $LE = true$,$TR = false$:

$$\begin{aligned}
&\mathbb{E}[N(LE = true, TR = false|\mathcal{D}] \\
&= \sum_{i=1}^{N} P_0(LE = true, TR = false)|\mathcal{D}) \\
&= 0 + 1 + 1 + 1 + 1 + 1 + 1 + 0 \\
&= 6 \ .
\end{aligned}$$

In the M-step we add the counts above to the denominator and numerator if using the MLE:

$$\begin{aligned}
\hat{\theta} &= \frac{\mathbb{E}_{\theta^0}[N(LC = true, LE = true, TR = false)]}{\mathbb{E}_{\theta^0}[N(LE = true,TR = false)]} \\
&= \frac{5.1}{6} \\
&= 0.85 \ .
\end{aligned}$$

If we use the BE we additionally add virtual counts to the expectation counts, where 1 is added for the counting the state *true* and 2 is added for counting both *true* and *false* of the variable LC.

$$\begin{aligned}
\mathbb{E}[N(LC = true, LE = true, TR = false)|\mathcal{D},\theta^0] = 5.1 + 1 = 6.1 \ , \\
\mathbb{E}[N(LE = true, TR = false)|\mathcal{D},\theta^0] = 6 + 2 = 8 \ .
\end{aligned}$$

The probability parameter is updated in this case:

$$\begin{aligned}\hat{\theta} &= \frac{\mathbb{E}_{\theta^0}[N(LC = true, LE = true, TR = false)]}{\mathbb{E}_{\theta^0}[N(LE = true, TR = false)]} \\ &= \frac{6.1}{8} \\ &= 0.7625 \ .\end{aligned}$$

The resulted the CPT is shown in Table 2.5 (using MLE in the M-step) and 2.6 (using BE in the M-step).

LE	*true*		*false*	
TRAJ	*true*	*false*	*true*	*false*
true	1	0.85	0.3	0
false	0	0.15	0.7	1

Table 2.5: Resulted CPT after computing one iteration using EM algorithm where in the M-step the likelihood is maximized

LE	*true*		*false*	
TRAJ	*true*	*false*	*true*	*false*
true	1	0.7625	0.3	0
false	0	0.2375	0.7	1

Table 2.6: Resulted CPT after computing one iteration using EM algorithm where in the M-step the posterior is maximized

2.4.4 Adaptation

When a system is running we repeatedly receive at each new time cycle new observations which are inserted and propagated as evidences. Thus, we can immediately take the influence of the new observations into account to improve the (conditional) probabilities specified for our system. A suitable approach doing this is the Adaptation, also know as sequential updating, which can only be implemented to discrete chance nodes. [12]
For the task of parameter estimation there are different statistical methods

to adapt the new observation. A general approach is called fractional updating where a fictitious sample size s is introduced to express the certainty of a learned distribution. For a set of counts $(N_1, N_2, ..., N_m)$ of m states $X = \{x_1, x_2, ..., x_m\}$ of a child node A the sample size can be chosen as $s = N_1 + N_2 + ...N_i$. The larger the sample size, the smaller the second-order uncertainty. If e.g. a particular configuration is certainly observed one time in the state x_1, then we count $N_1 := N_1 + 1$ and $s := s + 1$ such that $P(A = x_1) := \frac{N_1+1}{s+1}$ and the rest are updated to$P(A = x_i) := \frac{N_i}{s+1}$ for $i = (2,3,...,m)$. Furthermore, we influence the counting by multiplying the sample size with a fading factor $q \in (0,1)$ before the adaptation takes place. This method is called fading which is used to ensure that the counts are not overestimated such that empty counts build up from the history and the parameters become hardly changeable. Fading factors are stored in a fading table, where each value in the table corresponds to a parent configuration. If a fading factor is not set between 1 and 0 for a particular parent configuration then the adaptation is disabled for this configuration. In the introduced example above for the incoming hard evidence to the state x_1 the counting and sample size are updated: $N_1 := N_1 \cdot q + 1$, $s := s \cdot q + 1$ and $N_i = N_i \cdot q$ for $i = (2,3...,m)$. If the same evidence is given more than one time, the counting is computed recursively: $N_1 := (((N_1 \cdot q + 1) \cdot q + 1) \cdot q + 1...)$. The second equation can be formulated as: $s^* = \frac{1}{1-q}$ to obtain the effctive sample size which represents the steady-state situation. On the other hand, we can adapt the fading factor by a declared effective sample size: $q := \frac{s^*-1}{s^*}$. [12]
For a general case, let assume as given a variable A with states $X = (x_1, x_2,...,x_n)$, the current counting: N_k for $k \in [1,n]$ and the fictitious sample size $s = \sum_1^n N_k$ denoting the present certainty of $P(A|pa(A) = \pi)$ for a particular parent configuration π, whenever a evidence $P(\pi|e)$ is inserted, the states of A are updated (sequentially) [12]:

$$P(A = x_k|pa(A) = \pi) := \frac{N_k \cdot q + P(X|\pi,e) \cdot P(\pi|e)}{s \cdot q + P(\pi|e)} , \qquad (2.13)$$

where $k \in [1,n]$. The fading factor $q \in (0,1)$ for the parent configuration π is adapted using a declared effective sample size s^* [12]:

$$q = \frac{s^* - P(\pi|e)}{s^*} . \qquad (2.14)$$

In this thesis we use the Adaptation to learn the probability parameters within a TCPT between different time-slices of the developed DBN fragment.

An additional explanation of the algorithm as example of this particular use case is given in the Section 7.2.

2.5 Construction of the Network Structure

In this thesis two different modeling approaches for the construction of the network structure are used and compared. The first is the knowledge-based modeling where the expert knowledge about each domain is explicitly incorporated into the network structure. The second approach is a pure data-driven model which is systematically determined by the given data without the use of knowledge-based modeling. This section provides a brief description about the modeling steps of both approaches.
To construct a knowledge-based BN model we need deep understanding of the problem domain. Similar to the human thinking process, a knowledge-based BN is constructed based on different logic layers. The goal is to provide estimates of certainties for events which are not directly given in the data set. First we need to identify these hypothesis events which are grouped into sets of mutually exclusive and exhaustive states of events to form the hypothesis variable(as described in [12]). In the next step we analyze the characteristics of the corresponding data set of the problem domain and extract all information variables which have high influence on the hypothesis variable. Related information variables are grouped into the same domain as a single network fragment (OONF). Causal related variables or fragments are aggregated by adding a corresponding child node to the next level. Each node within the BN model represents the knowledge of a physical or statistical relationship between the modeled random variables. The states of each variable must be explicitly modeled by considering the constraints and the causal relationship to the previous and next logic layer. We initialize the probability distributions of the modeled variables using a suitable expression based on expert knowledge or simple physics/kinematics relations. The probability parameters given in the CPTs can be estimated and optimized by suitable learning algorithms. The whole process is structured as follows:

1. Formulation of problem domain, defining hypothesis variables for the output layer
2. Feature selection, finding information variables in the input layer
3. Specify the states for all defined variables considering the constraints and requirements

4. Group and link all related information variables to form the information channels

5. Initialization of probability parameters

6. Optimization/learning of the probability parameters

The other approach of data-driven modeling used in this thesis is the Naive Bayesian Network. Under the assumption that each information variable is only dependent on the defined initial output event, we simply associate all the information variables in the input layer with the event variable in the output. In other words, there is no logic based structure since the relationship between the input and the output is only based on the given data set. Through a suitable feature selection algorithm (according to [4], [17], [12] and [19]), we systematically arrange all observable variables given in the data set by their influence on the output event. We form the input layer by selecting a certain number of variables which have high impact on the output event. The strength of causal relationship between the information variables and the initial output event (to be recognized) is determined systematically by a learning algorithm such as the EM algorithm as well as the Adaptation approach.

3 Technical background

This chapter gives an insight about the technical background according to the work of this thesis. At first, Section 3.1 provides a description of the term "intelligent vehicle" including its essential components and the development state. Section 3.2 briefly introduces the environment sensors which are the receptors of the essential information used for the situation analysis. An idea about how and where the developed models can be set up is given in Section 3.3 where the concept of situation awareness is explained.

3.1 Intelligent Vehicle

"Intelligent vehicle" represents a collective term of the connection and collaboration between computers, sensors and vehicle control systems. The sensor systems have the task of information perception. The collected information is processed and analyzed by the implemented algorithms. As a result particular instructions are send to the actuators. [29]
A common task of an intelligent vehicle is to assist the driver in vehicle operations. A variety of assistance systems are already available in the premium vehicles on the market today. Well known assistance systems in the serial cars nowadays are e.g. Anti Blocking System (ABS) and electronic stability control (ESP), which help the driver in the stabilization the vehicle in special situations. With the implementation of environment sensors, such as ultrasonic, radar and (stereo-) camera, the advanced driver assistance systems (ADAS) with defined use cases perceive input data from multiple sources and take necessary control actions according to the environment features. Different ADAS are already implemented in today's serial cars and have already achieved respectable improvements in terms of comfort, safety and efficiency. A representative system is the Adaptive cruise control (ACC) which is an extension of cruise control and can regulate the longitudinal distance relatively to the controller-object-vehicle in the front. Further development of ACC system is undertaken by many car manufactures. Mercedes Benz e.g. introduces in 2013 the DISTRONIC PLUS in combination with steering assist [1]. DISTRONIC PLUS is an adaptive control system

based on multiple radar sensors (long-, medium- and short-range radar) with integration of stereo camera. It monitors the longitudinal and lateral dynamics of EGO vehicle relatively to the front car and the lane marking up to a distance of 200 meters and a velocity of 200 kilometers per hour. If the driver activates the system and sets a desired velocity, the system takes over the driving task of braking and acceleration and holds the EGO vehicle in the center of the track. Moreover, sensors are installed within the vehicle to calculate the behavior of the driver. For the DISTRONIC PLUS system, the driver must have his or her hands on the steering wheel, otherwise the driver is getting warned by visual and acoustical signals and the system is deactivated (secured Human-Machine Interfaces (HMI)). [1] [29]

The described ADAS belong to the serial stage today. Driver is still responsible for the vehicle control. The automated functions can only be activated under special circumstances and the driver has to take all the remaining driving tasks such as the monitoring of driving environment and of the operational systems. These systems belong to the automation level 2. For the future, we are talking about autonomous driving which has become a very popular theme since 2009. The first stage of autonomous driving is the conditional automation (automation level 3) where the dynamic driving tasks such as braking, acceleration, steering and also overtaking are done by the systems with the expectation that the human driver will respond appropriately to a request of intervention. The automation level 4 is the high automation, where the systems must be able to handle all driving tasks, even if a human driver does not respond to the request of intervention. The final level 5 is the full automation, that the vehicle is able to handle driving tasks in all environment conditions that can be handled by a human driver. [6] [29]

One of the most important task in the development of autonomous driving system is the environment perception. The autonomous vehicle must be able to record and interpret the local environment in the required completeness and accuracy. Essential traffic environment features include e.g. the lane markings, the obstacles and the traffic participants. Furthermore, the state of the EGO vehicle is calculated using on-board sensors in order to compute the relative dynamics between the EGO vehicle and the detected environment features. The current development focuses on the use of High-Definition digital map (HDM) which should significantly improve the accuracy of satellite-based localization and compensate the weakness of environment perception. [29]

3.2 Sensor systems

There are two basic categories of sensor systems. The first are the on-board sensors which perceive information about the current state of EGO-Vehicle such as wheel speed sensors, steering wheel angle sensors, pedal sensors, gear sensors, pressure, temperature sensors and more. The second are the environment sensors such as stereo cameras, radar, laser scanners and ultrasonic sensors.
On a low level, different sensors independently provide measurement signals which are processed and associated by fusion algorithms. Sensor fusion is a very important task. Redundant data about the same object are reconciled. The improvement is achieved since the dependency of measurement errors can be taken into account by estimation algorithm. Incomplete or missing data are complemented mutually by different types of information. Acquisition rate of the entire system can be increased by this approach of data fusion. This can be achieved on the one hand by the parallel processing of information from the single sensors, on the other hand by a corresponding temporal design of the acquisition process. [29]

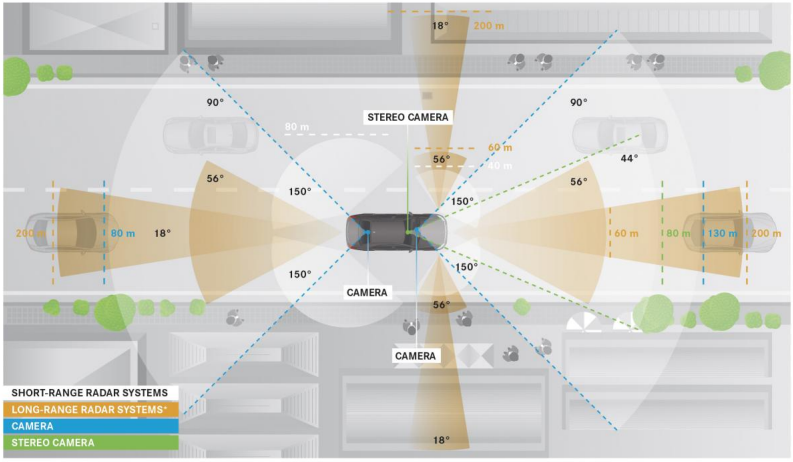


Figure 3.1: Environment Sensors [3]

The environment sensors radar (short, middle and long range) and camera

(mono and stereo) are specified in Figure 3.1. They are used to provide the measurement data for the purpose of this thesis. The measurements produced by stereo-camera and radar are fused by Kalman filter such that the strength of both sensors is combined and the weaknesses are reduced. Furthermore, a synchronization of measurements must be done to ensure data consistency. The stereo camera e.g. used for this thesis has the measurement frequency of 60 milliseconds and a detection distance up to 60 meters, while the radar systems have the measurement frequency of 66 milliseconds and the detection distance up to 200 meters.

3.3 Situation Awareness

Obtaining the data from environment sensing we want to determine the situation awareness which plays a fundamental role for ADAS and autonomous driving. In the human thinking process, situation awareness can be considered as a process of data perception and processing as well as the comprehension of the obtained information. Based on the given information as well as our knowledge and ability, we have to accurately interpret the situation features in very short time in order to perform the correct decisions and actions. The automotive situation awareness is not very different. It consists of the environment representation (situation model) and the situation analysis (Fig. 3.2). Environment representation can be considered as a dynamic data structure in which all internal and external representations of objects and infrastructure elements are integrated. The recording and tracking of the objects and elements are carried out using suitable, usually fused, sensor data, which are (mostly) obtained from (stereo) camera or radar. The detected and integrated objects and elements are mapped relative to a common reference system to extract the perceived features for situation analysis. In complex ADASs (e.g. avoidance assistance), comprehensive data about the lane marking, the vehicle dynamics as well as the free space analysis are recorded and processed for the tasks of situation analysis which gives a suitable interpretation of the environment situations as well as a probabilistic prediction of the future evolution of the traffic scene. Such a system for the situation assessment should not only be efficient in the sense of accuracy and short calculation time, but should also meet a number of functional requirements. [29]
There are two basic situation models (described in [29]) for the representation of the environment features: object-based representation and grid-based

representation. The grid-based approach represents the environment as a discrete cell-system where the vehicles move over the grid and the on-board sensor system provides information on whether specific cells are free or not. This type of modeling is primarily suited to the representation of a static scenario. It does not require any model hypothesis and is therefore very robust against model uncertainty. In the object-based approach all relevant traffic participants, the relevant infrastructure elements and also the EGO vehicle are described by their individual dynamic model such as a time-discrete state-space model. Information and states, such as positions, velocities and object expansion, are continually updated with the incoming sensor measurements. Since both the measurement data as well as the models themselves contain uncertainties, suitable filtering methods are necessary in order to take the uncertainties of the environment representation into account. Frequently used methods are e.g. Kalman Filter and recursive Bayes Filter. Since there are advantages of both grid-based and object-based approaches, the combination of both methods are utilized in the hybrid architecture for the environment representation [29]. In this thesis the knowledge-based BN models are based on the same strategy.

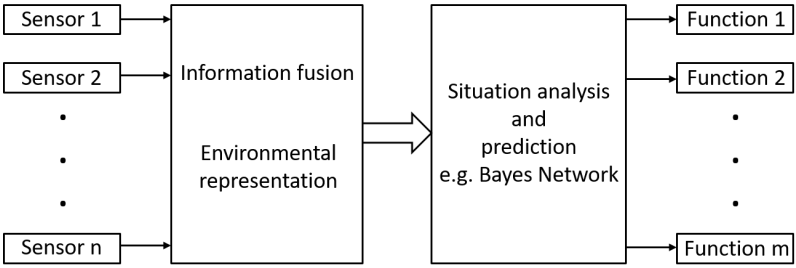


Figure 3.2: Data structure of a modular architecture for ADAS [29]

Situation analysis is used to establish associations among the entities in the situation model (according to Fig. 3.2). Due to the safety nature of the use case, necessary and accurate interpretations and prediction of the situation must be provided in very short time such that the vehicle control systems have enough time to react correctly to the given situation. The situation analysis uses a variety of different models and algorithms, among others the (Dynamic) Bayesian Network. The last is used in this thesis for

identification and interpretation of driving maneuvers of the EGO vehicle and the surrounding OBJ vehicles in the highway traffic. More details are given in the following chapters.

4 Development Environment

This chapter describes the essential aspects of the development environment including the experimental vehicle, the software architecture, the features of data set, the evaluation method and the construction of original knowledge-based model. The content is based on the work of [14] and [24]. The modified procedure for data labeling for the ground-truth data set, as well as the suitable data reorganization, reflected in the remodeling of logic layers for fast parallel computation has been part of this master thesis work, and its preliminary description is given in [28], which is under review.

4.1 Experimental Vehicle

A Mercedes E400 is available as an experimental vehicle which is used for the on-line evaluation of developed algorithms as well as for the collection of real traffic data for off-line learning and testing. The vehicle has a stereo camera behind the windshield and three radar systems for the short range from 0.25 to 60 meters, the mid range up to 130 meters and the long range up to 200 meters in front. The radar systems are from standard series production of Daimler AG and used in assistance systems such as DISTRONIC PLUS. The stereo camera is from development stage. It has a width of 0.2 meters and an aperture angle of 40°. The recording frequency is approx. 60 milliseconds. The experimental vehicle is equipped with three computer systems. The first computer is responsible for recording and processing radar data which are provided from the mess interface. Radar objects are formed and forwarded by the CAN-Bus. The second computer is image processing computer. It includes the algorithms for the fusion and analysis of the image processing objects, as well as provides processed data of the lane marking and object vehicles as well as for the free space analysis. These environment features form the basis for the developed maneuver recognition framework of this thesis. The third computer manages the communication between the computer systems and the sensors. It shows e.g. the status of the respective units.

4.2 Software Modules

The software modules developed for the project are grouped into so-called world model [14], which represents a unified framework for the organization of input data and the communication between the implemented algorithms. The world model is based on different logic modules, including the sensor data processing, the formation of BV-objects, the specification of object relationship as well as the recognition of driving maneuver. The procedure, from data processing until the forwarding of the environment features to the network classifiers, is visualized in the fig 4.1.

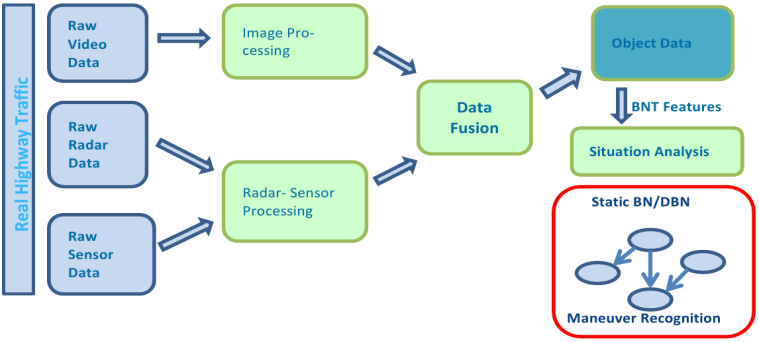


Figure 4.1: The procedure of processing and forwarding the environment features for the situation analysis using BN [27]

In data processing, all the necessary input data from radar and camera are encapsulated and organized for the fusion. A fusion algorithm based Kalman filter is implemented to extract the information from different environment sensors where the redundant data about the same object are improved and the incomplete data are complemented mutually by different types of information. Information e.g. about the lane marking and the outline of the OBJ vehicle is provided by the stereo camera. The relative position and velocity in the longitudinal direction (x-direction) between two OBJ vehicles are calculated by the fusion of the radar and camera data. The stereo camera can accurately determine the distance between the EGO vehicle and the OBJ vehicle in the front, but cannot observe the objects which are obscured by other OBJ vehicles or obstacles in the environment. The hidden

objects are usually detected only by the radar, based on electromagnetic waves. However, the radar data alone is not accurate enough for the exact specification of the information about each object in the surroundings. It can be only used for the recognition of possible driving intentions, like the need to perform a lane change due to a slower front vehicle.
The situation characteristics extracted from the data fusion are evaluated by means of a driving maneuver recognition module. Features such as object vehicle and lane markings are stored as input data in their own modules. They are passed on to the developed classifiers of Bayesian Networks.
The software modules have been transferred both to the Linux platform off-line for statistical evaluation and to the on-board computers in the experimental vehicle for the on-line testing in the real highway traffic. The algorithms are developed in *C++* and converted into *C* code in the computation systems of the experimental vehicle. The Bayesian networks models are developed within the software tool HUGIN, which allows the graphical modeling and parameterization of the BN, and provides functions such as feature selection, data-based learning of the network parameters and network structure, as well as the automatic C code generation. In addition, HUGIN has an interface for application programming with the corresponding *C* libraries for the simplified integration of the generated *C* code in different development environments. The integrated *C* library of HUGIN forms the basis for the exchange computation results and likelihood formation by the implemented BN models on-line in the experimental vehicle.

4.3 Data

The test data are collected on different European highways, in direction from Stuttgart-Singen(Germany) A81, in direction Stuttgart-Kalsruhe on A8, as well as in the vicinity of Luxembourg on *A*1, *A*3 and A6. When one of the defined lane change maneuvers to be recognized by the system is detected, the recording button is activated by the driver and a measurement sequence is saved for the purpose of analysis and learning. All sensor data before and after the fusion from the last 30 seconds are recorded. The resulting data occupies 4.3 GB of disk space for each driving sequence which might contain several vehicles in the traffic scene.
The recorded sequences are played offline as video sequences using the same image processing software as in the experimental vehicle. Each OBJ vehicle, which is documented in the video sequence, forms a pair relation

with the EGO vehicle. A part of the input data as well as the classification results of the Bayesian network for the vehicle pair are structured in Excel tables. We label the data for each vehicle side of the considered vehicle pair (OBJLEFT, OBJRIGHT, EGOLEFT, EGORIGHT) in a four-line block-wise representation for each time cycle. The resulted EXCEL sheet contains 77 variables including input data and the computed probability results. If all data per time cycle is combined as one data case, this results in $4 \cdot 77 = 308$ variables per each time step, and per each pair of vehicles.

4.3.1 Input Data

Input data can be assigned to different categories according to the modules in the software architecture. The first category is the lane detection containing information about the lane markings and lanes. The most important variables are the lane index id, the curvature and the change of the curvature. Lane index id is essential for the assignment of detected object vehicles. Since the image processing recognizes the lane markings to the left and right of EGO vehicle, the system can build up to 3 lanes with different identification index. Curvature and its change (derivation) are required to transform the lane coordinate systems into the Frenet-coordinate-system (as shown in Section 4.4) in order to make the relationship of the object vehicle independent from the curving segments of the road. This corresponds to computing relevant variables in the vehicles' lane coordinate system. The variables characterizing the lane and lane markings are presented in Table 4.1.

Table 4.1: Lane information (according to [14])

lane_id	lane index
c_0	curvature
c_1	derivation of the curvature
lanemarkingtype	types of the lane marking (solid or dashed)

The second category includes the radar objects which provide the essential information for the adaptive distance control of the ACC function. Radar objects mainly relate to the relative positions and velocities between the EGO vehicle and the detected OBJ vehicles in the longitudinal (x) and lateral (y) direction. Moreover, every detected OBJ vehicle maintains an identification index (id) and a position index ($posinf$). Since the radar can not detect the

lane markings, the position information of every OBJ is relative to the EGO vehicle. There are six possible positions specified by the radar of today's serial production. These are left, relevant, right, left hidden, front hidden and right hidden (see Fig. 4.2). In addition, the motion state *movstate* of a radar object is given as: undefined, stationary, moving, opposite. The essential variables of radar objects are listed in Table 4.2. [14]

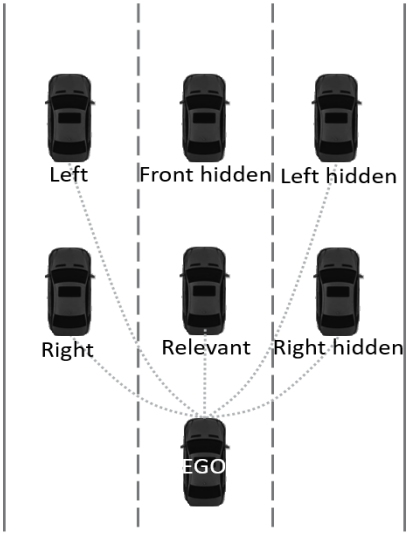


Figure 4.2: 6 possible positions ($posinf$) of radar objects [14]

Table 4.2: Variables for the specification of radar objects (according to [14])

id	identification index of detected object
x_{rel}	relative longitudinal distance $[m]$
y_{rel}	relative lateral distance $[m]$
$v_{x_{rel}}$	relative velocity in the longitudinal direction $[m/s]$
$v_{y_{rel}}$	relative velocity in the lateral direction $[m/s]$
$posinf$	position information of detected object
$movstate$	movement state of detected object

The third category denotes the BV-objects representing the results of the image processing algorithms which are developed by the BV-team of Daimler AG. BV-objects provide partly the same information about the relative distance and velocity between EGO and OBJ vehicle, as well as between two detected OBJ vehicles. These information are usually fused with information from the radar in order to obtain stable and robust estimates about the respective states. In addition, BV-objects include outline characteristics about the detected objects such as their length and width, and features with respect to the lane marking, e.g. the yaw angle (ψ) and its temporal derivative ($\dot{\psi}$) as well as the lateral distance ($OLAT$) and velocity ($VLAT$). The recognition distance of BV-objects is up to 60 m. The essential variables that characterize the BV-objects are listed in Table 4.3.

Table 4.3: Variables for the specification of BV-objects (according to [14])

id	identification index of detected object
x_{rel}	relative longitudinal distance $[m]$
y_{rel}	relative lateral distance $[m]$
$v_{x_{rel}}$	relative velocity in the longitudinal direction $[m/s]$
$v_{y_{rel}}$	relative velocity in the lateral direction $[m/s]$
$a_{x_{rel}}$	relative acceleration in the longitudinal direction $[m/s^2]$
$a_{y_{rel}}$	relative acceleration in the lateral direction $[m/s^2]$
ψ	yaw angle of OBJ vehicle according to the lane marking
$\dot{\psi}$	angular speed according to the lane marking
$OLAT$	lateral distance relative to the lane marking
$VLAT$	lateral velocity relative to the lane marking
L	length of detected object
B	width of detected object $[m/s^2]$

In addition to the categories described above, the free space analysis can be assigned to the last data category. The free space analysis is based on different algorithms provided by the BV-team and localization team of Daimler AG. The basic idea is to compute the longitudinal and lateral dynamics with respect to the detected obstacles, which can be e.g guardrail, people or vehicles. This category is not really relevant to the work of this thesis since the specification of the free space is modeled explicitly as BN-fragment using data from the three above described categories.

4.4 Coordinate System

As basis for the modeling of the situation characteristics (features) we use a symmetrical coordinate system where two Cartesian coordinates are transformed to Frenet coordinates (see Fig. 4.3) such that the lane is always rectilinear in the view. The transformation is done using the Frenet-Serret formulas, where each point on the lane marking, which can be considered as a curve $\vec{r}(t)$ =(cos(t),sin(t),t)$^\top$, is represented by the vector tangent unit $\vec{T}(t)$ to the curve, the vector normal unit $\vec{N}(t)$, and the binormal unit $\vec{B}(t)$ [14]:

$$\vec{T}(t) = \frac{\vec{r}'(t)}{\|\vec{r}'(t)\|} , \tag{4.1}$$

$$\vec{N}(t) = \frac{\vec{T}'(t)}{\|\vec{T}'(t)\|} , \tag{4.2}$$

$$\vec{B}(t) = \vec{T}(t)^\top \times \vec{N}(t)^\top . \tag{4.3}$$

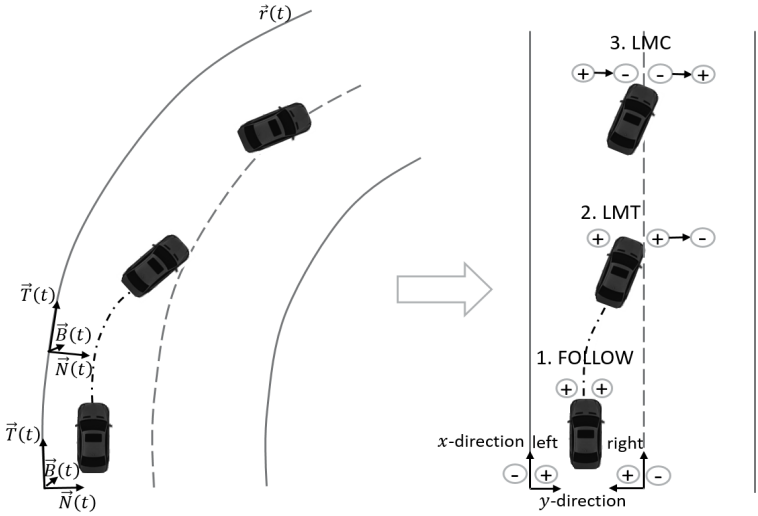


Figure 4.3: Symmetric coordinate system to model the Vehicle-Lane-Marking relation

The transformed coordinate system is implemented to each of the lane marking left and right of the vehicle, which allows a suitable description of the driving behavior within the lane between the two lane markings left and right, as well as for the time points when the vehicle is crossing the lane marking of the left or right side (Vehicle-Lane-Marking relation). An example is given in Fig. 4.3 where three situations are presented:

1. Vehicle is in the left lane. The coordinates for both vehicle sides are positive.

2. Vehicle touches as well as starts to cross the lane marking on the right side. The coordinates for the right side of the vehicle becomes negative.

3. Vehicle middle (mid point of front bumper) has crossed the lane marking which causes a coordinates transformation into the right target lane. (Definition of a lane change)

4.5 Definition of Lane Change

The coordinate system is the basis for the modeling of the vehicle-lane marking-relation which again leads to the definition of lane change. For the modeling, learning and evaluation, which are presented in the following sections and chapters, we clearly define the time points:

1. the vehicle is hurting the adjacent free space by touching the lane marking (Fig. 4.3: LMT).

2. the vehicle is entering the adjacent lane by crossing the lane marking (Fig. 4.3: LMC).

Based on the two time points of lane marking touching (called LMT) and lane marking crossing (called LMC), we can provide a logical definition of a lane change maneuver as well as develop methods for data labeling and model evaluation.

4.5.1 Ground-Truth Data Set

After the off-line simulation, the recorded measurements and calculated situation features are structured for each time step (data base) in EXCEL sheets where each file contains a pair relation between the EGO vehicle and one of the recognized OBJ vehicle within a certain period of time (< 30 s). In order to simplify the learning and evaluation procedure later we generate a ground-truth data set where the data are divided into clearly defined classes according to the driving maneuvers. If we consider the situations in the real traffic on the high way, we usually observe three general classes of driving maneuver: (fulfilled) Lane Change (LC), FOLLOW and relevant (for lane change) FOLLOW. There exist clear definitions over the first two classes. LC denotes the maneuver class that a vehicle starts approaching the lane marking of the current lane, crosses the lane marking and ends up the maneuver in the target lane. The definition of FOLLOW maneuver is that the vehicle is driving in its own current lane of motion over the entire time period without touching or crossing the lane marking. However, the third class "FOLLOW-relevant" does not have a clear definition. It includes canceled (or abandoned) lane change, intention of a lane change, unclean/destructed driving behavior or even sensor measurement error. We call this class "relevant follow" because all of the described situations end up without crossing into the target lane. Since the maneuver of this class cannot be explicitly specified and the data are often of low quality we exclude this class from the ground-truth data set.
The separation process of the data set is based on the measurements of lateral distance towards the lane marking ($OLAT$). The measurements of $OLAT$ are recorded by the stereo camera data within a distance of 60 meters. If a vehicle crosses the lane marking, its coordinates are transformed into the target lane. The transformation can be established in the data of $OLAT$. If e.g. a vehicle is crossing into the left lane, we can observe the time point of the coordinate transformation in the data when the $OLAT$ measurements denoting left side of the vehicle changes from minus to plus and of its right side from plus to minus. This change of sign means that the vehicle center (mid point of front bumper) just crossed the lane marking.
Based on the measurements of $OLAT$, two points: LMT (lane marking touch) and LMC (lane marking cross) are labeled in every data sequence. Each sequence of the LC class includes a piece of FOLLOW data, the time point of touching the lane marking (LMT) and the time point of crossing the lane marking (LMC). At the end of each LC sequence the vehicle's position

changes into the target lane. This new position is assigned to the coordinates of the target lane, which also leads to the change of the lane index ID. The data of a LC sequence are labeled and saved six seconds (100 cycles) before the time of the coordinate transformation (i.e. the actual LMC). The minimum length of the LC sequence is set to 2.4 seconds (40 cycles). We set those constraints on the dividing data in order to ensure the quality of each maneuver sequence. Moreover, the LC-sequences are specified by the vehicle type and the relevant side of the LC maneuver: EGOLEFT, EGORIGHT, OBJLEFT, OBJRIGHT. Only a single vehicle side is considered in each sequence under evaluation. If e.g. the OBJ vehicle changes to the lane on the right side, this maneuver is labeled using the name OBJRIGHT since the data of the right side of the OBJ vehicle is most relevant in this case.
The "Follow-real" class consists of tracking data of consistently FOLLOW maneuver without touching the lane marking. The sequences in this data class do not have any of the both defined labeling points (LMT and LMC). The true tracking data is at least 4.8 seconds (80 cycles) long and includes only the data where the vehicle follows the track without touching the lane marking.
The third class "Follow-relevant" is the tracking data, where the space violation occurs. In such scenarios, the observed vehicle exhibits an inconsistent driving behavior. The vehicle touches or crosses the lane marking several times, but returns to its own lane instead of changing into the target lane. The causes of such cases are e.g. canceled (or abandoned) lane change maneuvers, inattentive driving and measurement error. In such case, it is often not clear whether the classification of the algorithm should be evaluated as right or wrong. Therefore, we exclude this class from the ground-truth data set.
For the supervised learning (later in Chapter 5), the ground-truth data set (about 1500 maneuver sequences) is divided into learning data set (about 1000 sequences) and testing/validation data set (about 500 sequences). All data sequence (in form of EXCEL sheets) from both learning and testing data set are combined into a single EXCEL sheet. The data matrix is further processed by additional labeling method according to the learning algorithms: EM algorithm and Adaptation approach.

4.6 Modeling Approach

The basic idea for detecting driving maneuvers is based on the qualitative description of the movement of a vehicle relative to a reference system, such as the lane marking, a driving vehicle or existing free space. For the construction of a knowledge-based model, we first formulate the problem domain (as described in Section 2.5) using an approach based on the relationship between the EGO and one detected OBJ vehicle. In this context, there are seven maneuver classes:

1. OBJCUTIN: OBJ moves from the neighboring lane in front of EGO to the same lane.
2. OBJCUTOUT: OBJ drives away from the lane of EGO.
3. EGOCUTIN: EGO is driving into the same lane as the OBJ in front.
4. EGOCUTOUT: EGO is driving away from the lane of the OBJ in front.
5. OBJFOLLOW: EGO is driving in the same lane behind the OBJ.
6. LANEFOLLOW: EGO and OBJ are driving straight in two different neighbor lanes.
7. DONTCARE: This is a complementary class including all of remaining situations where EGO and OBJ are doing maneuver at the same time, but there is no any potential of collision, e.g. EGO and OBJ are driving from the same track into different neighboring lanes.

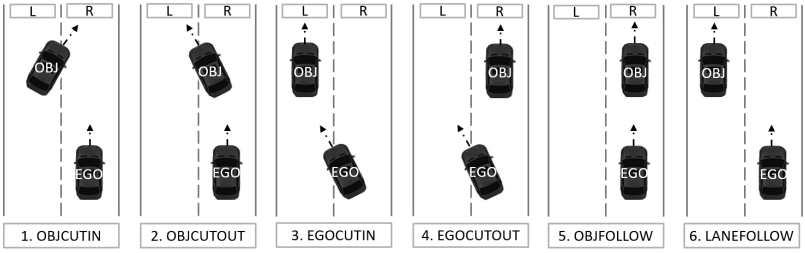


Figure 4.4: Driving maneuver classes in the output event (according to [14])

The six relevant maneuver classes are presented in Figure 4.4 where the red car represents the EGO vehicle and the blue car represents the OBJ vehicle. If this problem domain based on the described maneuver classes is used to classify the output event (as shown in Fig. 4.4), we have to specify the relation between the EGO and the OBJ vehicle. This Vehicle-Vehicle relation is given by the position of both vehicles and their movement at the moment of consideration. The situation 1 shown in Figure 4.4 f.e. is defined as OBJCUTIN, because the EGO vehicle is following the lane on the right side and the OBJ vehicle is driving from the right lane to the left, while cutting the EGO-trajectory. Thus, we can establish all of the described maneuver classes if the position description of both vehicles and their movement can be specified. The combined maneuver class DONTCARE is a complementary quantity denoting all other situations where the driving maneuvers of both vehicles do not carry any collision hazard for the motion of the vehicle pair.

4.7 Knowledge-based Original Model

In this section the ORIG-Model is described. It represents the prototype of the knowledge-based Object-Oriented Bayesian Network (OOBN). It is the basic model for the developed classifiers later. The ORIG model is also dynamically extended with regard to the trend features for early and accurate recognition. The performance of all developed classifiers are statistically evaluated in comparison to the ORIG model in off-line simulations using the recorded sensor data from the test drives.

4.7.1 Network Logic Layers

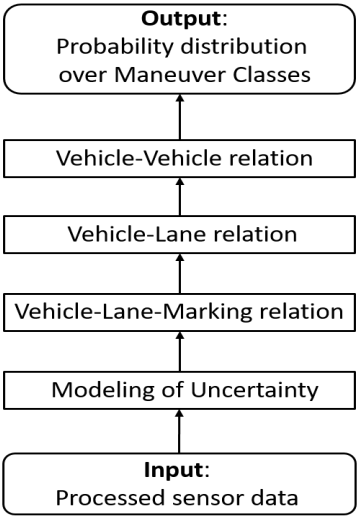


Figure 4.5: Description of the logic layers (according to [14])

The network structure is built up hierarchically in different logic layers representing the classes of object-object-information and their causal relationship. The logic layers are partly presented in Figure 4.6 where the movement of OBJ vehicle is considered. The logic layer contains BN fragments from the

original model and is introduced in this work specially for the purpose of parallelization of computation as described in [28]. The movement of EGO and OBJ vehicles is modeled by analogy in a similar manner. Both classes are combined in the layer Vehicle-Vehicle relation in order to specify the relative movement between the EGO and OBJ vehicle. In the following the logic layers are explained from top to bottom (see Fig. 4.5).

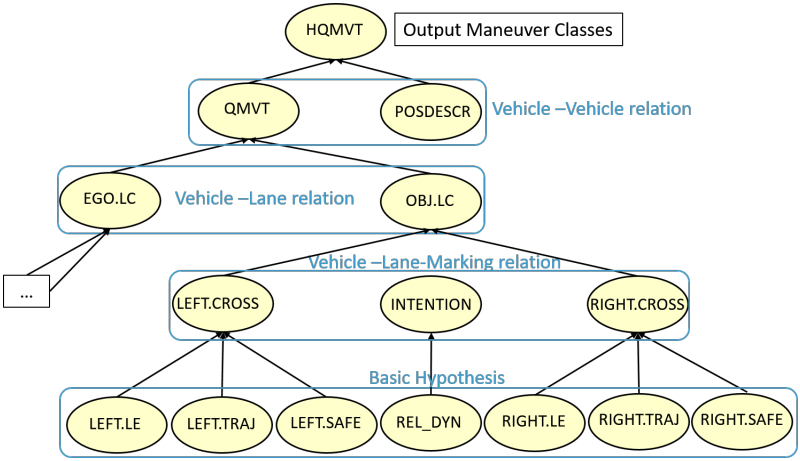


Figure 4.6: Network logic layers of the OBJ vehicle part

As described in Section 4.6, we formulate the output event using the defined maneuver classes which are grouped as states of the output variable *HQMVT* (High Quality Movement). *HQMVT* is conditioned to the parent nodes *QMVT* and *POSDESCR* denoting the relative movement and position between the EGO and a OBJ vehicle.
QMVT denotes the relative movement between EGO and OBJ vehicle. It is modeled using the state pair (mov_{EGO},mov_{OBJ}), where $mov_{EGO/OBJ} = \{L,R,G\}$. The complete set of movement state-pairs for the movement class *QMVT* is given as: *LL*, *LR*, *LG*, *RL*, *RR*, *RG*, *GL*, *GR* and *GG*. $L\widehat{=}$ left, $R\widehat{=}$ right and $G\widehat{=}$ straight (as show in Fig. 4.7) are states from the parent nodes *EGO.LC* and *OBJ.LC* denoting the Vehicle-Lane relation in the lower abstraction level. *LL* means e.g. that both EGO and OBJ are moving towards the lane on the left side, whereas *LR* means that EGO is moving towards

the lane on the left side and OBJ is moving towards the lane on the right side at the same time. Because of the knowledge-based modeling and the symmetrical states combinations, the initial CPT of $QMVT$ is approximately given as identity matrix (as shown in Fig. .2).

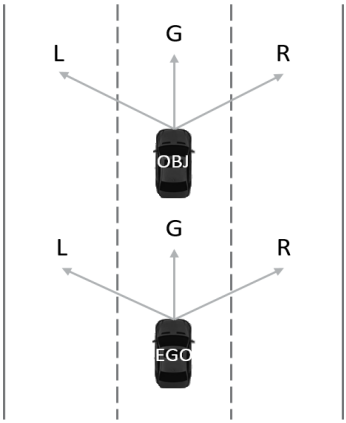


Figure 4.7: Relative Movement between EGO and OBJ (according to [14])

In addition to the movement class $QMVT$, the relative positioning of OBJ towards the EGO vehicle is modeled by the variable $POSDESCR$ which consists of three states: $POSLEFT$, $POSINFRONT$ and $POSRIGHT$. $POSLEFT$ ($POSRIGHT$) means that OBJ vehicle is located left (right) in front the EGO vehicle whereas $POSINFRONT$ indicates the positioning of OBJ in front of EGO on the same lane (as shown in Fig. 4.2). $POSDESCR$ has no parent node because the data of the position index are calculated by the fusion and provided as evidence to the BN. The prior probability distribution of $POSDESCR$ is initialized with uniform distribution. The occurrence frequency of the three relative positions should be given as equally probable and not weighted from the beginning which would influence the decision of the BN in the case that no observation of the relative position is given as input data.
As described before, the probability distribution of the defined maneuver classes in $HQMVT$ is conditioned to the states of $QMVT$ and $POSDESCR$. The conditional dependency is given as CPT (of $HQMVT$) which is modeled

based on expert knowledge. A part of the CPT is presented as example in Figure .1. First, we have the parent configuration that the OBJ vehicle is left in front the EGO vehicle ($POSLEFT$). Based on logical conclusion, we can determine the probability parameters in relation to the defined states of relative movement. Situations belonging to the complementary quantity $DONTCARE$ are e.g., if both vehicles are moving in the same direction (LL, RR) or in different unrelated directions (LR, RL, RG, GL). The situation, that EGO vehicle is moving left and the OBJ vehicle is driving straightly (LG), is defined as $EGOCUTIN$. If OBJ vehicle is moving right and EGO vehicle is driving straightly (GR), we assign the case to $OBJCUTIN$. If both vehicles are moving straightly (GG), it is defined as $LANEFOLLOW$. Furthermore, an observation can also be given as likelihood distribution, i.e. the situation can not be clearly allocated. If the relative movement $QMVT$ is given as likelihood distribution: $QMVT(LL = 0\%, LR = 0\%, LG = 70\%,..., GL = 30\%, ...)$ in the case that OBJ vehicle is left in front the EGO vehicle ($POSLEFT$), we obtain an uncertain statement in the output event as probability distribution: $HQMVT(DONTCARE \approx 30\%,...,EGOCUTIN \approx 70\%)$.
In the logic layer of Vehicle-Lane relation we have two separate variables with identical states for EGO and OBJ vehicle $EGO.LC$ and $OBJ.LC$. Both variables are modeled using the states: L, R, G, denoting the driving direction of left, right or straight with regard to the current lane of motion. These three states are conditioned to the Boolean states in the parent variables denoting the vehicle-lane marking-relation.
The variables $LEFT.CROSS$ and $RIGHT.CROSS$ in the layer of vehicle-lane marking-relation summarize the calculation results of the input data into Boolean statements. $LEFT.CROSS$ ($RIGHT.CROSS$) denotes the probability of crossing the lane marking on the left (right) side of the considered vehicle.

In the input layer we have the basic hypotheses (as shown in Fig. 4.8) evaluating the situational characteristics including their measurement uncertainties. It consists of three basic hypotheses: LE (lateral evidence), $TRAJ$ (trajectory), and $SAFE$. Each of them is modeled as an object-oriented network fragment (OONF). LE provides influence results on the lateral dynamics based on the situation features $OLAT$ and $VLAT$ from the current input data. $TRAJ$ is a model of lane change trajectory. Its influence is based on the combined evidence from the features: the yaw angle (PSI), the maximal exploitable acceleration ($ALAT$) and the time to lane cross ($TLCR$). $SAFE$ provides reasoning on the possibility of a maneuver by influence based on the free accessible space which results from a grid based modeling approach

(variable *OCCGRID*, see Section 4.7.5). Besides these original fragments of [14], an additional fragment *RelDyn* is developed for the modeling of the longitudinal dynamics (X_{rel}, V_{rel}). Considering the relative longitudinal distance and velocity between two vehicles, we calculate the intention of a vehicle in a particular situation. This model is based on the radar data up to a distance of 200 meters. Using this model allows conclusion even on the front vehicle's relative dynamic due to a front-hidden vehicle (see Fig. 4.2). Thus, LC intention can be recognized even before the vehicle starts to move towards the lane marking. However, the statement provided by *REL_DYN* is only about the driving intention of a vehicle, i.e. if a vehicle is driving steadily closer to its front man, a lane change would be required, which does not mean, that the maneuver is initialized. Thus, we separate this fragment from the upper layers and evaluate its event separately. In the following subsections, detailed description of the OONFs in the input layer is presented, which is necessary for the understanding of the further chapters.

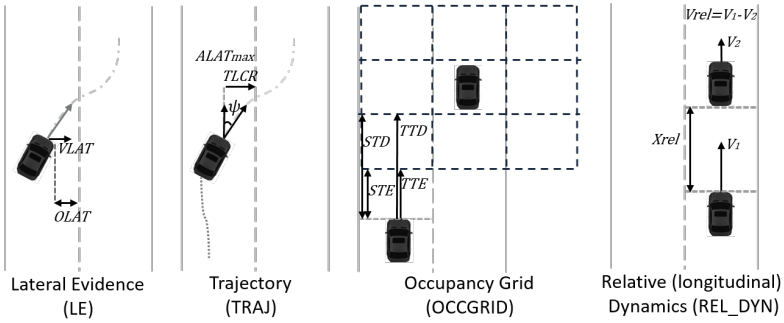


Figure 4.8: Basic hypothesis (according to [14])

4.7.2 Uncertainty

The maneuver recognition represents a task of the type reasoning under uncertainties with heterogeneous data which are acquired from fused multi-sensors measurements and thereof computed situation features by physical models. All fused and inferred data have naturally inherited uncertainties due to the sensor noise and the model uncertainties. Thus, the treatment of uncertainty of the input data is an important modeling task [14].

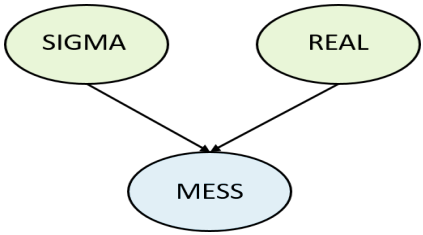
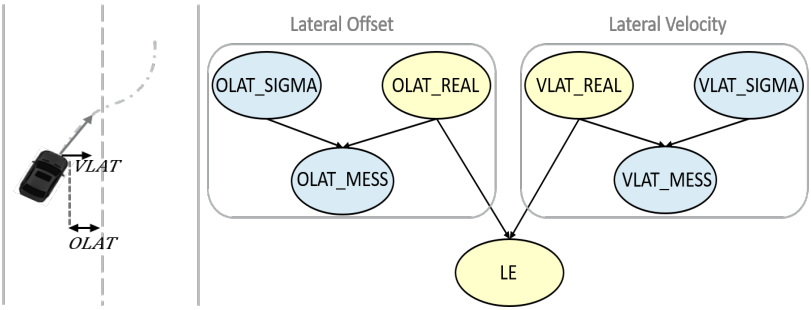


Figure 4.9: Modeling of uncertainty (according to [14])

Figure 4.9 shows the causal modeling taking care of the uncertainty in the input data. It consists of 3 variables: $SIGMA$, $MESS$ and $REAL$, where the sensor-reading $MESS$ of any measured variable is conditionally dependent on random changes in the expected input value $REAL$ and the precision of its measurement given by the variance $SIGMA$. Variance represents the sensor noise which is assumed as a zero-mean Gaussian random process. Thus, the CPT of $MESS$ is initialized as normal distribution of the expected value and its standard deviation:
$P(MESS|REAL,SIGMA) = \mathcal{N}(REAL,SIGMA)$
In principle the calculation of $P(REAL|MESS,SIGMA)$ is reversed due to the reversibility of the Bayesian Theorem. The probability distribution of the expected value ($REAL$) is inferred from the sensor-reading and the sensor disturbance which are obtained as evidences from the sensor fusion algorithm of a Kalman Filter.
The described principle of error estimation is applied in all modeled BN fragments while inferencing on the basic hypothesis (as mentioned in Section 4.7.1). They are described in the following subsections. The uncertainty treatment completes the set of situation features and improves the quality of input data used for the reasoning.

4.7.3 Lateral Evidences



(a) LE hypothesis [14] (b) LE model

Figure 4.10: LE Fragment, where the basic hypothesis (a) and the Bayesian model (b) are presented (according to [14])

The *LE* fragment (Fig. 4.10) is one of the basic motion hypothesis to model the Vehicle-Lane-Marking relation. There are two variables *V_LAT_MESS* and *O_LAT_MESS* denoting the situation features of lateral velocity and offset according to the transformed coordinates for the description of the course of lane marking. Both measurement quantities are computed by a street course model based on the approach of Euler spiral. The parameters of the clothoids are approximated using a polynom of 3*rd* degree in order to compute the course of the lane marking which is related to the data of the OBJ vehicle given in the coordinate system fixed to the EGO-vehicle. The derivation of the model equations is described in detail in Section 4.2.2.1 in [14]. The variables *V_LAT_SIGMA* and *O_LAT_SIGMA* denote the measurement uncertainty. Thus, the expected input values are given as normal distribution of the computed situation features (*OLAT_MESS* and *V_LAT_MESS*) and the standard deviation. Due to the range of the measured value and the standard deviation, there are 30 states in *OLAT_REAL* and *VLAT_REAL*, 32 states in *OLAT_MESS* and *VLAT_MESS*, and 24 states in *OLAT_SIGMA* and *V_LAT_SIGMA*. The number of states have been selected in order to provide high accuracy of inference results and smooth change of probability between the states. The smooth change has also an effect on the visualization of maneuver recognition (according to [14]).

The output hypothesis variable LE provides a Boolean statement about the probability of crossing the lane marking. The relation between the lateral dynamics of the vehicle (given as distributions in the expected values) and the lane change probability is modeled using sigmoid functions as initialization of the probability parameters given in the CPT of the variable LE:

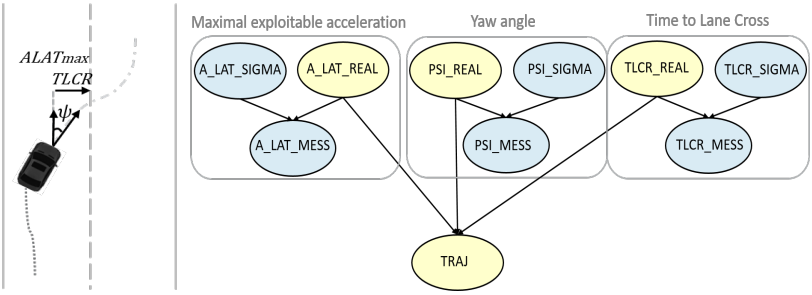
$$P(LE|OLAT\_REAL, V\_LAT\_REAL) = \frac{s_o}{s_o + e^{m_o \cdot OLAT\_REAL}} \cdot \frac{s_v}{s_o + e^{m_v \cdot VLAT\_REAL}} .$$

The conditional probability distribution in LE is given by the multiplication of the two sigmoid functions representing the probability function of lateral offset and velocity. The parameters s_o, m_o, s_v and m_v denote the shifts and slopes of the sigmoid curve and characterize the features of the sigmoid function. The sigmoid function is transferred into CPT by discretization (as shown in Fig. 4.11). The initialized probability parameters are learned from data using the Expectation Maximization algorithm.

VLAT_REAL	**-1- -0.9**			**-0.9- -0.8**			**...**	
OLAT_REAL	**-1- -0.9**	**-0.9- -0.8**	**...**	**-1- -0.9**	**-0.9- -0.8**	**...**	**...**	**...**
true	θ_{11}	θ_{12}	...	θ_{1n}	$\theta_{1(n+1)}$	...	...	...
false	θ_{21}	θ_{22}	...	θ_{2n}	$\theta_{2(n+1)}$	...	...	...

Figure 4.11: Representation of the CPT of the variable LE which is conditioned to the states (value ranges) of $OLAT\_REAL$ and $VLAT\_REAL$

4.7.4 Trajectory



(a) TRAJ hypothesis (b) TRAJ model

Figure 4.12: TRAJ Fragment, where the basic hypothesis (a) and the Bayesian model (b) are presented (according to [14])

In contrast to the clothoids-based concepts in the hypothesis LE, $TRAJ$ uses a trajectory-based lane change model where the temporal information is calculated and taken into account. The lane change trajectory is modeled as a polynomial of third degree [14]:

$$y = f_{lc}(x) = a_3 \cdot (x - x_s)^3 + a_2 \cdot (x - x_s)^2 + a_1 \cdot (x - x_s) + a_0 \ ,$$
$$x \in [x_s, x_s + D] \ ,$$

where x_s denotes the current position of the vehicle, D is a defined range in which the lane change trajectory is approximated by extrapolation, and the parameters a_0, a_1, a_2, a_3 are estimated on the basis of the last n detected positions of the vehicle. This model is described in more detail in Section 4.2.2.2 of [14].
As a result of the physical model, three measurement quantities: $TLCR$, $ALAT$ and PSI are derived as situation characteristics. $TLCR$ is the estimated time until the vehicle is predicted to cross the lane marking. $ALAT$ is the computed maximal exploitable acceleration. PSI is an angular parameter which denotes the orientation of the vehicle inside the lane according to the transformed course of the lane marking.
The network structure of $TRAJ$ is very similar to the LE fragment. The

given features are modeled as normal distribution of their measured values and standard deviations. The output is modeled as discrete Boolean variable which is conditioned to the computed distributions of the expectation variables named as "..._$REAL$". The CPT in the output variable $TRAJ$ is initialized by the multiplication of the sigmoid functions (similar to the variable LE as described in Section).

4.7.5 Occupancy Grid

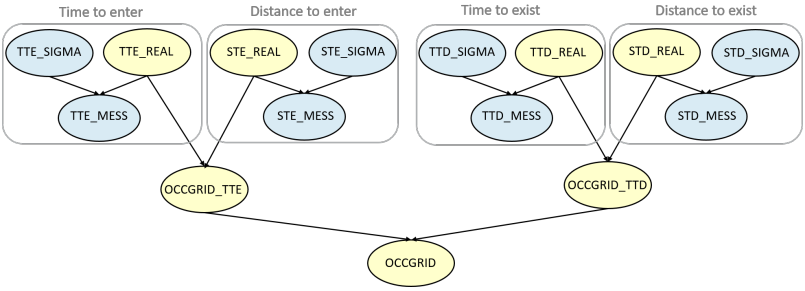


Figure 4.13: OCCGRID Fragment (according to [14])

$OCCGRID$ is a model for free space analysis which infers on the possibility of a lane change maneuver by evaluation of the drivable space. The model combines the temporal and distance analysis with occupancy grid. In this context, a 3×3 occupancy grid is constructed for each detected vehicle where the vehicle is in the middle of the grid. Each cell is built by the reference points (x_{ref}, y_{ref}) as well as the length and width of the cell. The status of a cell is defined as either free or occupied and its probability is inferred from the evidences on the time TTE and distance STE up to the entry into the cell as well as the time TTD and distance STD up to the exit from the cell (see Fig. 4.13). The entry and exit times are again determined from the fused measurement data with respect to the distance (STE and STD) as well as the velocity v_{rel} and the acceleration a_{rel} relative to the considered cell. The relation is computed using the functions representing

the kinematic motion of the vehicle [14]:

$$\frac{1}{2} \cdot a_{rel} \cdot TTE^2 + v_{rel} \cdot TTE - STE = 0 \,,$$
$$\frac{1}{2} \cdot a_{rel} \cdot TTD^2 + v_{rel} \cdot TTD - STD = 0 \,.$$

In order to categorize a cell as a free space to finalize a maneuver possibility, the entry and exit times as well as the corresponding distances must be greater than the predefined threshold values. A detailed description about the physical model of the hypothesis *OCCGRID* is given in [14].
The BN model of *OCCGRID* consists of four parts. Here, each of the time and distance of the entry and exit of a cell are calculated under consideration of the measurements uncertainties. The inference results are combined in two Boolean variables: *OCCGRID_TTE* and *OCCGRID_TTD*, which are again modeled using sigmoid functions. The output variable *OCCGRID* is simply a logic based statement, i.e. if both *OCCGRID_TTE* and *OCCGRID_TTD* are true, then the cell is defined as free (to drive in).

4.7.6 Relative Longitudinal Dynamics

In the hypothesis *REL_DYN* we use the radar data to characterize relative longitudinal dynamics between two successive vehicles driving on the same lane. The idea here is based on the human reasoning of driving maneuvers. When a vehicle is driving faster than its front vehicle, the probability of changing the lane becomes very high due to the intention of keeping the driving velocity by passing the front vehicle. This intention becomes clear if the vehicle approaches its front vehicle consistently without reducing its speed, otherwise it would drive on a collision course. In addition, the long range radar provides a view horizon up to 200 meters which allows the early recognition of a driving intention.
The model of *REL_DYN* (see Fig. 4.14) uses the same handling of uncertainties as the basic hypothesis described in the previous subsections. This model is constructed in the work of [24]. Instead of lateral features, the measurement data of relative longitudinal distance and velocity are propagated. The causal relation between the output variable *REL_DYN* and the input evidences is modeled using fuzzy logic instead of sigmoid functions. In this context two additional variables *REL_DYN_V_REL_OBJ* and *REL_DYN_X_REL_OBJ* are introduced for the data fuzzification using the linguistic variables "close", "comfort" and "far" as qualitative categories

denoting the statement of the relative distance *X_REL*, and "faster", "comparable" and "slower" representing the categories of relative velocity *V_REL*. The CPT in the output variable is given by the defuzzification where the qualitative categories are related to the driving direction *LEFT*, *STRAIGHT* and *RIGHT* using pseudo-trapezoidal membership functions. The parameters are chosen in regard to the driving praxis, that an overtaken maneuver is represented by a lane change to the left, whereas the vehicle leaves the faster moving (left) lane by slowing down and changing to the right.

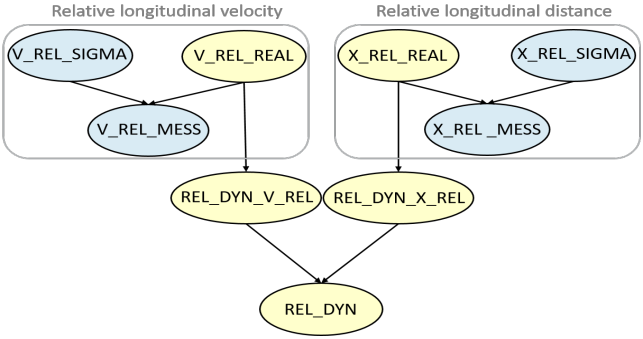


Figure 4.14: Bayesian network fragemtent, modeling of the relative longitudinal dynamics (according to [24])

In the logic layers, we combine *REL_DYN* with somewhat modified free space analysis to form the hypothesis *INTENTION* for the calculation of the lane change intention. The modified modeling for the free space analysis checks the suitability of a gab between two neighbor vehicles. This is performed by evaluating the safety features for longitudinal relative dynamics: the relative velocity (*V_REL* of FRONT and BACK) and time to collision (*T_REL* of FRONT and BACK), as a relation to the nearest vehicles in the front and behind on both the left and right lane. Thus, Maneuver Advice proves whether the lane change is desirable or not and if the necessary safety precautions, given by availability of a gap, are fulfilled. Both requirements are based on the longitudinal dynamics of the considered vehicle and its surrounding traffic participants. A detailed discription of the modeling approach using fuzzy logic for the calculation of the driving intention based on the variable *REL_DYN* is given in the thesis [24]. We use

this fragment in this thesis as an additional output of the BN to visualize the calculation of the relative longitudinal dynamics for the purpose. We do this for the purpose of evaluating the behavior of this model such that we could combine both modeling approaches of relative lateral and longitudinal dynamics for special use cases in the future.

5 Learning of Probability Parameters in Static Bayesian Network

Given the initial static BN model with clearly defined inputs and outputs we want to find a set of (most) suitable probability parameters and a standard probability threshold such that the requirements of accurate and early recognition of the defined maneuver classes are fulfilled. Given the ground-truth data set (see Section 4.5.1) we use methods of supervised learning which is based on two concepts: classification and regression. Classification denotes the data-based allocation of the most probable output (event states) values to the corresponding input evidences. Regression is used to obtain an accurate prediction of the development of the target variables over time. We use the software tool HUGIN as modeling, learning and test environment. The improved models are later implemented into the C-framework of the image processing program. The results are evaluated off-line by simulation and on-line by test driving in the real highway traffic.

To do the classification the "EM algorithm for graphical association models with missing data" [18] is used which is explained in Section 2.4.3. There are two different ways to perform the EM-learning. We can implement the algorithm to the complete BN-model including all evidence, hypothesis and event variables. In this way we are searching for an allocation between the input variables denoting the sensor measurements and the output probability distribution in the output event $HQMVT$ which infers on the maneuver classes to be recognized. The second way is only to classify the binary problems in the input hypothesis: LE, $TRAJ$ and REL_DYN.

In the case of learning knowledge-based models, we only learn the input hypothesis variables which have the task to evaluate the input data containing the situation features. The parameters in the input hypothesis variables are directly dependent to the frequency of observations in the data. The event variables are modeled using logical conclusion of expert knowledge where the parameters should not be determined by the frequency of observations. Therefore, we use the second method and carry out the EM-learning for the fragments: LE and $TRAJ$. Both hypothesis denote the real movement

and the orientation of a vehicle in relation to the lane marking which are consistently observable. The hypothesis *OCCGRID* and *REL_DYN* specify situation features which are not consequently observable. Either the free space criteria or the relative longitudinal dynamics (between front and front-hidden vehicles) are dependent on a third vehicle which is not always given in the driving scene. We do not perform the EM-learning on those two hypothesis variables since the corresponding data are not consistent.
In the following sections the procedure of supervised learning using EM algorithm is described. First we process the data such that the collected data are separated into LC- and Follow-classes and labeled for the counting issues of the EM algorithm. About 70% of the labeled data is used as learning data set and the rest is used for evaluation. Moreover, we use the sigmoid functions to initialize the CPTs. The sigmoid parameters are optimized due to the performance criteria: accuracy and timegain. We use the method of multi-objective optimization which is implemented in MATLAB. Given the knowledge-based model, the labeled learning data set and the optimized initial parameters, we perform the EM-learning in the software tool HUGIN to determine the probability parameters in the CPTs by counting the frequency of corresponding observations.

5.1 Data Labeling

As described in the previous section (Sec. 4.3), the collected data during the test drive are stored as individual driving sequences. Each sequence contains 30 seconds of measurements and can be simulated off-line in the image processing framework. The stored images are played as video sequences in the same frequency as the measurement frequency (60 ms) of the stereo camera. Simultaneously the sensor measurements and the inferred probabilities of the corresponding classifiers are recorded. For each formed vehicle-pair-relationship between an EGO vehicle and an OBJ vehicle with a specific ID number the recorded data is organized in an individual sequence.

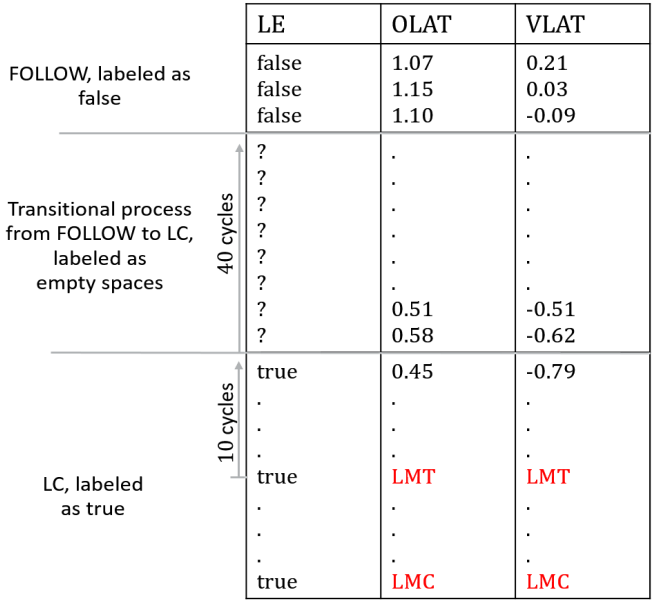


Figure 5.1: Labeling strategy as preparation for the EM-learning

We implement a labeling strategy to divide each data sequence into the set of LC data and the set FOLLOW-real-data. For this purpose we specify two fundamental issues to describe a LC maneuver based on our definition of

LC (as given in Section 4.5). The first is called LMC (Lane-Marking-Cross) which denotes the time point where the coordinate system is currently transformed into the target lane coordinates. The second is called LMT (Lane-Marking-Touch) where the time point is marked when the considered vehicle touches the Lane-Marking. At the end, we include the clearly separated data sequences into a single Excel table for learning purpose.
To support the EM algorithm, an additional column corresponding to each output (hypothesis) variable of the BN-fragment is created. We label explicitly the moment of LMT as the start of the ending phase of a LC maneuver. We follow the strategy that a LC sequence is labeled 10 cycles before the time point of LMT as true. The period ends up to the time point of LMC where the middle of the front bumper of the vehicle crosses the lane marking which is recognized by our system as transformation of lane coordinate. We can not specify the actual begin of the LC maneuver since we cannot find a generalization of the driving behavior of different OBJ-vehicles. Thus, we only label a time point where the LC maneuver has not started (in the most case). We define this time point as 40 cycles (about 2.4 seconds) before the LMT. We label the data as false. The gap between the two time points (40 cycles and 10 cycles before LMT) remains blank and should be filled automatically by the Expectation step using the current probability parameters in the corresponding CPTs. The labeling strategy is illustrated in Figure 5.1. As result, we labeled about 1000 maneuver sequences and use them as learning data. It is necessary that the data set is balanced according to the driving maneuver (LC or FOLLOW). Thus, about 500 sequences are a LC maneuvers and the other 500 sequences are FOLLOW maneuvers. The LC maneuver sequences contain processed and labeled data of a recording time between 2.4 and 6 seconds. The FOLLOW maneuvers are at least 4.8 seconds.

5.2 Multi-objective Optimization for finding an optimal Initial Guess

Due to the characteristics of the EM algorithm as described in Section 2.4.3, we need to find an optimal initial guess of the corresponding CPTs. The primary requirements of early and accurate recognition of driving maneuver lead to competitive criteria: accuracy and timegain. Thus, we formulate the problem of finding initial parameter guess of the relevant object-oriented Bayesian fragments for the EM-learning as a multi-objective optimization

problem. We use the strategy of Pareto Optimization and evaluate the sigmoid functions accordingly to the criteria. Related work is e.g. [13], where the Pareto-based approach is studied in terms of machine learning problems. In the following the approach of Pareto Optimization used for the LE-model is explained.
Wanted is the maximum of accuracy and timegain. The optimization problem is given as:

$$\max_{s_v,m_v,s_o,m_o} (accuracy, timegain)\ y$$

where both objectives are determined by evaluating the computed probability based on the labeled data set. In the design space, we compute the probability by the multiplication of the sigmoid functions which are used to approximate the probability distribution in the object-oriented BN-fragment LE to the measurement values of $OLAT$ and $VLAT$:

$$P(LE = true|VLAT, OLAT) = \frac{s_v}{s_v + e^{m_v \cdot VLAT}} \cdot \frac{s_o}{s_o + e^{m_o \cdot OLAT}} \ . \tag{5.1}$$

The parameters s_v, m_v, s_o, m_o denote the slopes and shifts of the sigmoid functions. We use an additional probability threshold to support the evaluation such that the computed probabilities can be mapped to a binary statement if the LC is either true or false. The threshold is implemented as $5th$ parameter which should be determined by the process of optimization in relation to the evaluation of the functional criteria.
Since the probability P cannot be negative or exceed 100% the constraints are set as: $-P \leq 0$ and $P \leq 1$ which simply represents the axiom of probability. In the case of the LE model, if we implement the search only in the positive parameter region, the constraints are automatically fulfilled according to the characteristics of sigmoid function.
The resulting optimization algorithm consists of two computation steps. In the first step we evaluate the variations of parameter sets from a large scaling step to a fine discretization. The results of the evaluation are stored into a table. In the second step we build the set of Pareto Frontier. In order to find the optimum, we maximize hypervolume in the criterion space and implement an appropriate weighting criteria. The implementation of the algorithm consists of the Evaluation and the Optimization-step, i.e.:
Evaluation-Step.

1. Read data sequences.
2. Variation of parameters.

3. Probability computation to the given parameter and data set.

4. Evaluation of the computed probabilities (mapping between design and criterion space).

5. Save the results of evaluation, e.g.:
   $M = [param_1, param_2, param_3, param_4, threshold, accuracy, timegain]$.

6. Go to 2 and repeat until the defined region is evaluated.

Optimization-Step:

1. Build the Pareto Frontier (strictly dominated).

2. Compute the hypervolume and implement the weighting criteria.

3. Find the maximum hypervolume.

A suitable weighting criteria is to define a minimum of acceptable accuracy: $accuracy_{min}$ which restricts the region of acceptance. On the one hand the criteria can be utilized to reduce the acceptable optimal Pareto set and on the other hand it can be implemented to the computation of hypervolume which is in this case the rectangular area under the Pareto Frontier:

$$hypervolume = (TP - TP \cdot accuracy_{min}) \cdot \frac{timegain}{1 - accuracy_{min}}.$$

In this context, we search for the maximum of the computed hypervolume by the use of the weighting criteria in order to find a particular parameter set which is most suitable according to the given trade-off problem.

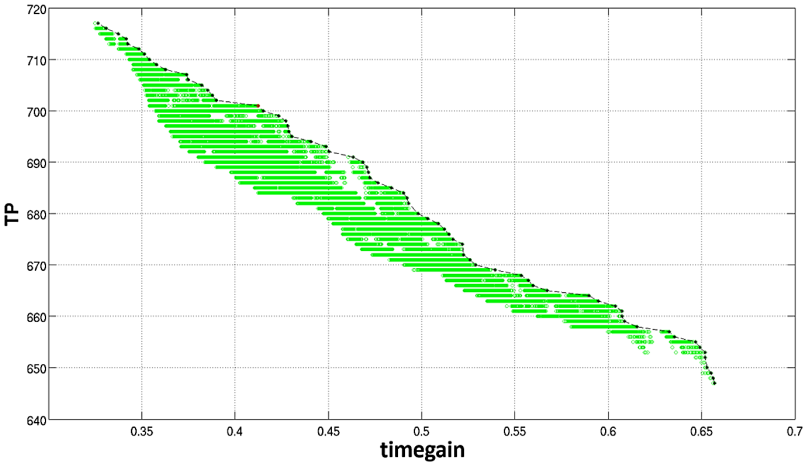


Figure 5.2: Plot of evaluations in the criterion space, using EGO+OBJ data of 740 maneuver sequences

Figure 5.2 shows the evaluation due to both objectives: accuracy and timegain in the criterion space where TP represents the number of correctly recognized maneuver sequences. We use a validation data set of 500 sequences containing 260 sequences of LC maneuver and 240 of FOLLOW maneuver. Moreover, every FOLLOW sequence contains each one of EGO and OBJ FOLLOW maneuver. Thus, the number of FOLLOW maneuver is doubled (480 FOLLOW maneuvers). The total number of maneuver is 740 which is the total number of LC and FOLLOW maneuvers (performed by a single vehicle).
The y-axis named TP represents the number of correctly recognized maneuver and x-axis *timegain* denotes the time difference between the recognition and the time point of LMC. Each plotted point represents the evaluation result of a unique parameter set. The Pareto-frontier (marked black) represents a set of potentially optimal (Pareto efficient) choices according to the trade-off problem, i.e there is no mathematical "better" solution in a particular configuration within the defined region of acceptance. It means literally, that you cannot obtain a better solution in the one optimization criterion without making worse the other criteria.
Furthermore, the plot includes both EGO and OBJ data in which the quality

of EGO data is better than the OBJ data. It means, we would obtain different optimal solutions, if we separately use the EGO or the OBJ data. Thus, different weighting criteria is implemented to support the algorithm for finding the optimal parameter set. The weighting criteria is an additional restriction to the acceptable region. It is set 90% for EGO data and 80% for OBJ data. Because the number of EGO and OBJ data sequences is exactly the same, we set the recognition minimum as mean of both weightings:

$$recognition_{min} = 0.85.$$

Implementing the weighting criteria explicitly to the computation of the rectangular area under the Pareto Frontier, we find the most suitable parameter set conditioned to all of the defined criteria of this optimization problem. It is marked as a red point in the plot which has a recognition accuracy of approx. 95% (701 TPs of 740 maneuver sequences), and a timegain of approx. 0.41 seconds to the time point of LMT. The corresponding optimal parameter set is for the BN-fragment LE is given as:

$$s_v = 0.07,\ m_v = 8,\ s_o = 109.5,\ m_o = 9.3,\ Threshold = 0.65$$

whereas the probability function used to approximate the CPT of LE is given as:

$$P(LE = true|VLAT, OLAT) = \frac{0.07}{0.07+e^{8 \cdot VLAT}} \cdot \frac{109.5}{109.5+e^{9.3 \cdot OLAT}}.$$

The resulting sigmoid functions is integrated into the C-code of the corresponding hypothesis in the C-framework for off-line simulation or for on-line testing in the test vehicle. The integration is necessary, because we need to perform the evaluation of network model using the optimized initial guess besides the MATLAB simulation in order to obtain a subjective feeling for the performance before we start the learning process.
Since the fragment $TRAJ$ also uses sigmoid functions as initial expressions, we follow the same optimization steps.

5.3 Expectation Maximization Learning

Given the labeled training data set (about 1000 maneuver sequences) of high quality and the optimal initial guess to approximate the probability parameters in the CPTs, we perform the EM algorithm to learn the conditional probabilities for the both essential hypothesis (LE and $TRAJ$) from the data. The characteristics of the EM algorithm are described in Section 2.4.3.

We use the optimized sigmoid function as expression to approximate the initial probability parameters of LE conditioned on the discrete states in $OLAT_REA$L and $VLAT\_REAL$. For the counting procedure of the EM algorithm the given expression has to be transferred into a CPT. A additional experience table is added to each of the state combinations, given in the CPT. The experience table contains the experience counts which reflect the number of times that a particular parent configuration is observed. Before the learning is started, experience tables have to be added and initialized in each variable where the probability distribution should be determined by the number of observations.
In the loaded data set we can also recognize the labeling (as given in Fig. 5.1) which is necessary to assign the states to either *true* for LC or $false$ for Follow. EM algorithm counts $+1$ for each labeled *true* and $+0$ for each labeled $false$. The empty spaces are counted using the current probability parameters which specify the occurrence of the respective parent configuration.
The process of EM-learning is controlled using two different abortion criteria. The first is to characterize a maximum number of iterations such that the iterative process is forced to terminate if the maximum number is reached. The second criterion is the definition of the convergence threshold of the relative difference ϵ between the log-likelihood for two successive iteration steps:

$$\epsilon = \frac{\log L(n) - \log L(n-1)}{\log L(n-1)} \ .$$

The log-likelihood is logarithm of the probability of all counted cases π:

$$\log L(n) = \sum_{i=1}^{N} \log P(\pi_i|\Theta) \ ,$$

where n is the number of the current iteration and the sum N of counted cases is the total number of counted cases given in the data. Log-likelihood denotes in this context the quantity of information given the current joint probability. The local maximum is reached if the log-likelihood between two successive iteration steps is sufficiently small.
For the learning procedure we use the convergence threshold as abortion criteria since we don't know how many iteration steps are necessary to converge to a local maximum. Furthermore, only the conditional probability parameters of the output variable LE should be learned from data. The reason is that the frequency of the adoption of a sensor measurement should

not be specified by the frequency of observations. For the knowledge-based models we consider the occurrence of a certain measurement value as equally probable. Thus, the probability distribution denoting the relation of the variables with the ending $REAL$ (expected value of measurements), $SIGMA$ (uncertainty of the measurements) and $MESS$ (sensor measurements, given as evidence) remain normally distributed as before.
In the case of learning the LE fragment, the algorithm converges after five iterations using the convergence threshold 0.0001. As a result, we obtain the learned CPT (for the variable LE) which is dependent on the frequency of the observed parent configurations and their labeling. The EM-learning procedure for $TRAJ$ fragment is similar as for the LE fragment since both models are used as basic hypothesis to calculate the vehicle lane-marking relation.

6 Trend Analysis for Static Bayesian Network

The approach of static BN naturally restricts the classifier performance due to the trade-off problem between the two objectives: accuracy and timegain, since the probability propagation is only based on the calculation of the currently incoming sensor recording. Consider the clustering plot of lateral dynamics given in the appendix (as shown in Fig. .7), if we e.g. push the parameterization of the *LE* into the blue area denoting the LC maneuvers, we obtain significant improvement in the timegain but an unacceptable accuracy in the recognition of FOLLOW maneuvers. Every movement tendency towards the lane marking introduces a high probability output denoting a LC maneuver. If we use instead a parameterization in favor of the accuracy, i.e. the lateral dynamics must exceed a high limitation for the classification of a LC maneuver. Then as consequence the LC maneuver is always recognized in the last moment.

In the reality, a LC maneuver typically starts with a phase driving towards the lane marking before the vehicle touches the lane marking. This so-called trend behavior must be recorded in the measurement data of lateral and longitudinal dynamics. It is therefore of great importance not only to use a snapshot of the recording, but also to keep and follow the data history in order to obtain a meaningful calculation of features showing a trend behavior.

In this chapter two trend algorithms: Linear- and logistic regression are presented which are used as supplements to the knowledge-based static BN in order to improve the timegain with no or little deterioration in the accuracy. linear regression is implemented to analyze the trend of the lateral dynamics. A prediction based on the history of the measurement data is given as evidence to the static network instead of the current measurement. Logistic regression is utilized to predict the probability in the LC-layer by following the development of the probability trend over time.

6.1 Linear Regression

Consider Figure 6.1, where the development of lateral offset of LC-sequences is plotted using a random chosen validation data set of over 150 LC-sequences. A (nearly) linear declining trend can be identified in the measurements of lateral distance towards the lane marking ($OLAT$). The measurement values of $OLAT$ keep decreasing until the coordinate system changes to the target lane. This phenomenon is also partly observable in the measurement of the lateral velocities ($VLAT$). However, the trend of $VLAT$ can be badly traced due to the unstable behavior of the velocity vector. In this context, we use linear regression to approximate trend history of $OLAT$ and compute a predicted value using the fitted regression coefficient. We repeat this at each time step when new measurement data are recorded.

A visual presentation of the idea of linear regression is given in Figure 6.2. A certain amount of data history of $OLAT$ can be used to fit the parameters of a linear function which can be extrapolated to predict the development of $OLAT$. There are two constraints for estimating the linear model. First, the range of data history used for the estimation must be strictly limited (<1 s) in order to ensure the piecewise linearity in the data development. Secondly, the computation time must be suitable for on-line evaluation.

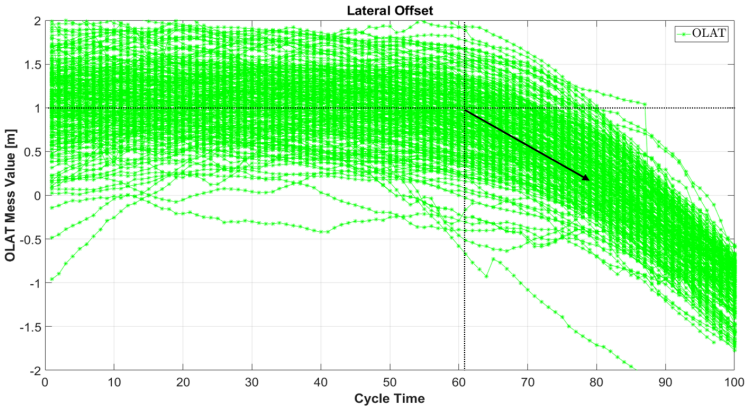


Figure 6.1: Development of lateral offset of LC data sequences (validation data set with over 150 LC-sequences)

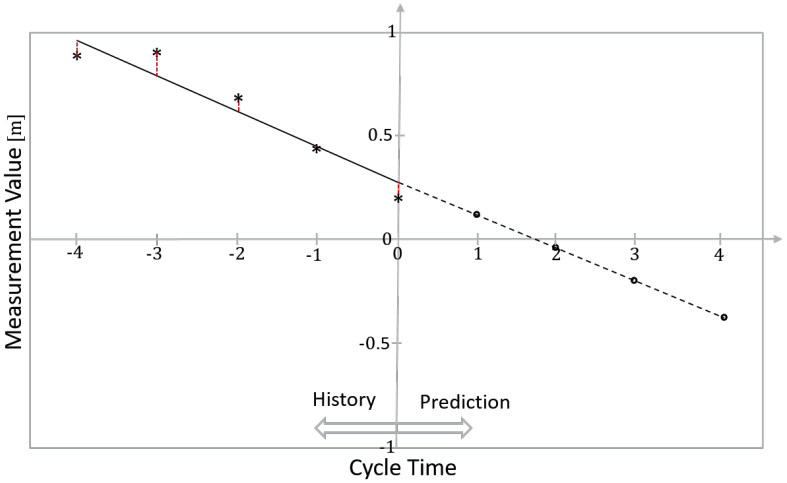


Figure 6.2: The idea of linear regression

To ensure the limitations of time and computation resources we use the simplified approach of least square curve fitting to determine the parameters of the discrete linear model denoting the development of $OLAT$ over time (according to [20]):

$$y_i = \beta_0 + \beta_1 \cdot t_i + \epsilon_i \ . \tag{6.1}$$

The scalar dependent variable y_i represents the observation of $OLAT$ measurement. We have only one explanatory variable t_i denoting the corresponding time point. The constant c denotes the axis section and the regression coefficient β_1 is the slope of the regression line. Error variable $\epsilon_i = y_i - \hat{y}_i$ is used to model the deviation between the observed value in the given data group and the computed value: $\hat{y}_i = \beta_0 + \beta_1 \cdot t_i$. Given a data group of the range n saved in a virtual buffer we can fit the parameters of the regression line by minimizing the quadratic error (according to [20]):

$$\min \sum_{i=1}^{n} \epsilon_i^2 = \min \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 \cdot t_i)^2 \ . \tag{6.2}$$

The algorithm is formulated as following:

1. Initialize the buffers of t_i and y_i.

2. Update the data set y_i by the current incoming measurement and actualize the time buffer for t_i.

3. Computation of the mean of the n scaled time points and the average of the corresponding independent $OLAT$ measurements:

$$\bar{t} = \frac{\sum_{i=1}^{n} t_i}{n} ,$$
$$\bar{y} = \frac{\sum_{i=1}^{n} y_i}{n} .$$

4. Computation of the regression coefficient β_1:

$$\beta_1 = \frac{\sum_{i=1}^{n} (t_i - \bar{t}) \cdot (y_i - \bar{y})}{\sum_{i=1}^{n} (t_i - \bar{t})^2} .$$

5. Compute the section constant β_0:

$$\beta_0 = \bar{y} - \beta_1 \cdot \bar{t} .$$

6. Extrapolation by defining a suitable prediction horizon T:

$$y_f = \beta_0 + \beta_1 \cdot T .$$

7. The predicted $OLAT$ value is given as input evidence to the BN.

8. In the next time step the process is repeated from the step two if the storage is updated by currently recorded data for the same object vehicle.

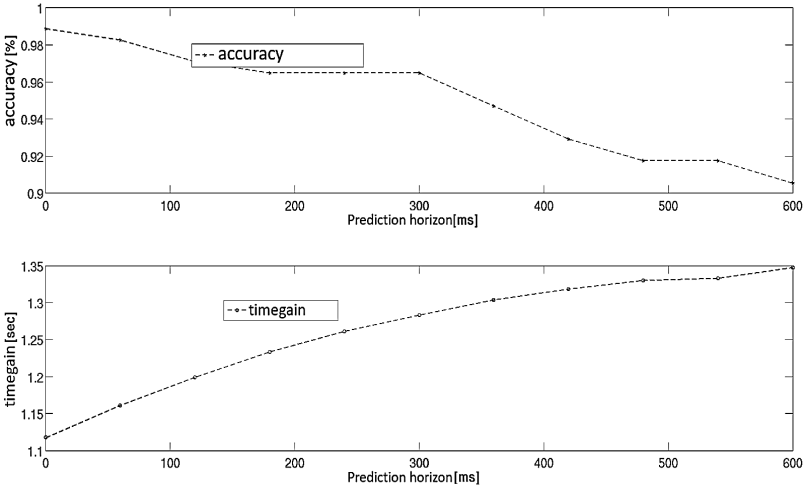


Figure 6.3: Accuracy vs. timegain in the relation with prediction horizon (0-600 ms) using linear regression

The range n of the storage buffer and the prediction horizon T have to be chosen appropriately. The prediction horizon T must be comparable to the range n of the data history, e.g. if we use data of last 5 cycles, the prediction horizon should be maximal 10 cycles. For this purpose the model is reproduced in MATLAB. Using the approach of multi-objective optimization (see Section 5.2) to the predicted data set by linear regression we can find a set of suitable parameterization of n and T. The buffer size n is optimally set between 5 and 10 where the data between 5 or 10 times the sampling frequency (60 ms) are stored into the buffer. The prediction horizon has significant impact on the performance objectives accuracy and timegain. In Figure 6.3 the evaluation of both competitive objectives using a validation data set is plotted in relation with the prediction Horizon. Here, we set a fixed buffer size $n = 5$ in the favor of computation resources. A significant deterioration in the accuracy is seen after 300 mini seconds while the timegain is growing nearly linearly with increased prediction horizon. Therefore, $n = 5$ and $T = 300ms$ would be a possible parameterization belonging to the optimal Pareto-set.

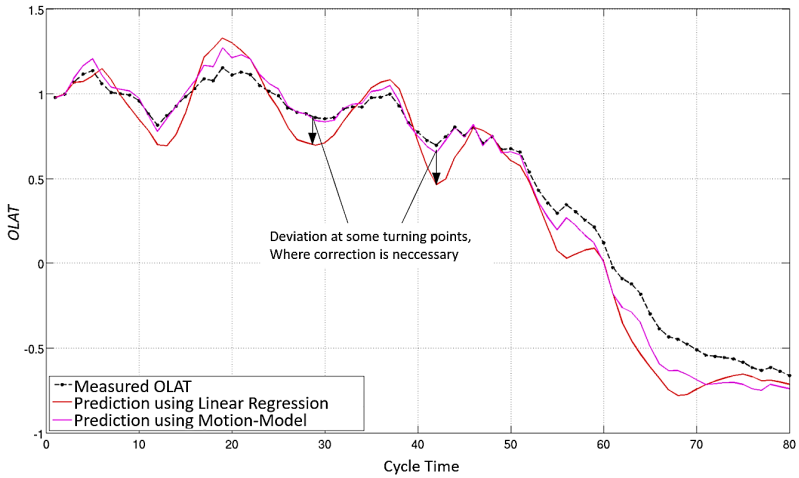


Figure 6.4: Deviation between prediction using linear regression (only based on $OLAT$) and Motion model (based on the kinematics using both $OLAT$ and $VLAT$)

In order to improve the correctness and robustness there are three different approaches which are developed for the trend analysis using linear regression. The first approach is to set the axis section c of the linear model to the current measurement value of $OLAT$. Thus, the least square estimation is reduced to a problem with a single parameter, namely the regression coefficient. The extrapolation (step 6) always starts by the current measurement value of $OLAT$:

$$y_f = OLAT(t_0) + m \cdot T \ .$$

The second approach is to define a set point for an additional evaluation of the predicted value. This can be done using the kinematic equation to describe the physics of the kinematic motion of the vehicle and is given as:

$$OLAT(T) = OLAT(t_0) + VLAT(t_0) \cdot T \ .$$

In figure 6.4 both prediction approaches are plotted to a LC sequence of an OBJ-vehicle recorded in the real highway traffic. We can recognize that

the linear regression keeps the prediction based on the data history even the curve is turning back in the next moment, i.e. the vehicle is moving back to lane middle and linear regression cannot follow this turning point immediately. Thus, we implement the rule: If the deviation between the predicted value y_f by the linear regression and the predicted value $OLAT(T)$ by the kinematic function exceeds a defined *threshold*, then the prediction is dropped, otherwise the predicted value y_f is given to the BN:

if $(y_f - OLAT(T)) < threshold$ **then**
  prediction $= y_f$;
else
  prediction $= OLAT(t_0)$;
end

The third approach is to divide the given set $\mathcal{D} = \{d_1,d_2,...,d_n\}$ of data history of length n in several sections $section_0 = \{d_n\}$, $section_1 = \{d_{n-1},d_n\}$, $section_2 = \{(d_{n-2},d_{n-1},d_n)\}$,..., $section_{n-1} = \{d_1,d_2,...,d_{n-2},d_{n-1},d_n\}$ with the length $n-k,...,n-1$, where $k = (1,...,n-1)$ is a specific time point in the past corresponding to a data value given in the data set. linear regression is implemented to each data section using the prediction horizon corresponding to the time difference between the first and the last data value of each section. The results are stored together with the current data value d_n in an additional output-buffer. The values in the output-buffer are again fitted to a linear function using the linear regression algorithm to obtain a smoothed predictive value. This approach has two advantages comparing to the classic linear regression. First, there is no need to find the prediction horizon T since it is automatically related to the buffer size n. Secondly, the current input value has more impact on the prediction so that the predicted value is corrected quickly in cases that the trend is canceled by a turning value. The algorithm is given as following.

for $k := 1$ *to* n **do**
  $result[k] \leftarrow LinearRegression((n-k):n), T = (n-k), x[(n-k):n])$
end
$prediction \leftarrow LinearRegression((1:n), T = 0, result)$;

6.2 Logistic Regression

Consider Figure 6.5 which shows a typical curve course of the output probability in the LC-layer describing the Vehicle-Lane relation (BN-fragments: *EGO.LC* and *OBJ.LC*). The red and blue curves denoting the lane change probability to the left and right sides show a typical sigmoidal behavior characterized by a growing trend and the constraints: $P(t) \in [0,1]$. This feature leads to the same consideration as given in Section 6.1 for the linear regression. Thus, we can implement the trend analysis to predict the output probability during its growing phase in order to reach the defined thresholds at a very early time point. In this context we approach the system outputs for a small time interval t by a logistic growth function given as:

$$P(t) = \frac{1}{1 + e^{\beta_0 + \beta_1 \cdot t}} \,, \tag{6.3}$$

where the parameters β_0 and β_1 denote the shift and the slope of the curve. The function value t represents a particular time point which is taken from the discrete time interval $t \in [t_1,...,t_n]$ of n sampling times.

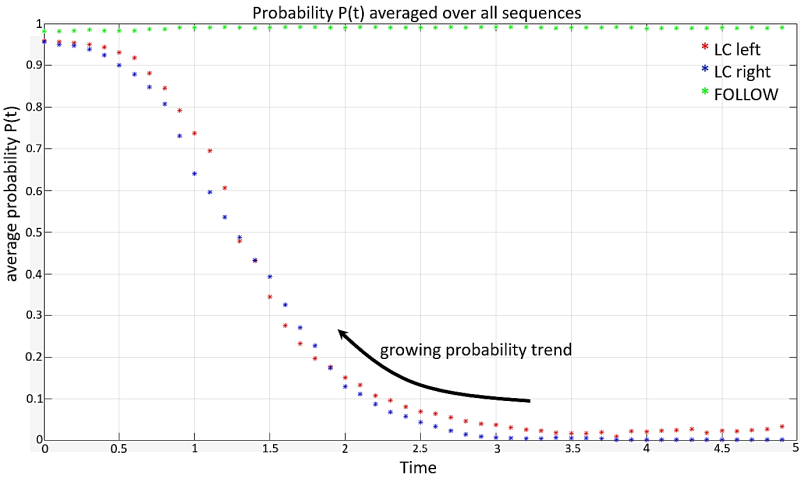


Figure 6.5: Typical curve course of the output probability [27]

Probability $P(t+T)$ of a certain future time point $t+T$ can be extrapolated

analytically by determining the two free parameters β_0 and β_1 from the recently obtained output probabilities $P(t)$. Because of the noise of the measured values as well as the fact that the function is only an approximation of the system output we can exclude an algebraic determination of the two free parameters β_0 and β_1. Instead, we use a regression approach to estimate the parameters.
First we transform the approximation function into a linear function such that the parameters β_0 and β_1 as well as the function variable t are linearly related to a function value $\tilde{P}(t)$:

$$\begin{aligned}
\frac{1}{P(t)} - 1 &= e^{\beta_0+\beta_1\cdot t}\,, \\
\frac{1-P(t)}{P(t)} &= e^{\beta_0+\beta_1\cdot t}\,, \\
\log(\frac{1-P(t)}{P(t)}) &= \beta_0+\beta_1\cdot t\,, \\
\tilde{P}(t) &= \beta_0+\beta_1\cdot t\,.
\end{aligned}$$

Then we use the simplified approach of linear regression to fit the independent regression coefficient β_1 and use it to compute the dependent parameter β_0:

$$\begin{aligned}
\tilde{P}(t) &= \beta_0+\beta_1\cdot t\,, \\
\bar{t} &= \frac{\sum_1^n t_i}{n}\,, \\
\bar{P}(t) &= \frac{\sum_{i=1}^n P(t_i)}{n}\,, \\
\beta_1 &= \frac{\sum_{i=1}^n (t_i-\bar{t})\cdot(P(t_i)-\bar{P}(t))}{\sum_{i=1}^n (t_i-\bar{t})^2}\,, \\
\beta_0 &= \bar{P}(t)-\beta_1\cdot\bar{t}\,.
\end{aligned}$$

After the parameter fitting, the function $\tilde{P}(t)$ is transformed back to the logistic function $P(t)$ which is then extrapolated using both fitted parameters β_0, β_1 and a defined prediction horizon T:

$$P(t+T) = \frac{1}{1+e^{\beta_0+\beta_1\cdot(t+T)}}\,.$$

The system architecture of logistic regression is similar as for the linear regression. We use a virtual buffer of size n to store the history of obtained probability values. The buffer is updated whenever a new probability value is computed by the BN in the same output. We use this approach to predict the outputs from the nodes $EGO.LC$ and $OBJ.LC$. The predicted probabilities are forwarded to the next logic layers to obtain an earlier recognition of the positive (LC) class: OBJCUTIN, EGOCUTIN, OBJCUTOUT, EGOCUTOUT, from the states modeled in the output event $HQMVT$.
Compared to the original output probability both regression approaches show a improvement in timegain with a deterioration in the recognition of FOLLOW maneuver. According to the chosen prediction horizon and buffer size we obtain a different result. We again use the Pareto-optimization approach to find out a suitable prediction horizon. The prediction horizon is set 10 time cycles (about 600 [ms]) for logistic regression. For linear regression we use the third approach where the prediction horizon is dependent on the buffer size. We simply follow in this case the logic that the more information we store the more we can set the prediction horizon. Thus, we optimize the buffer size n which is given as 5 values for the approach using linear regression.
Compared to the linear regression, logistic regression has a significant improvement in the timegain since we predict the probability in its growing phase. However, we observe bad robustness using logistic regression in the case of FOLLOW maneuver. Every small disturbance, where the probability values are growing for few cycles, is magnified by the prediction. We try to solve this problem using a output filter to smooth the prediction. We test different filter approaches such as PT1 filter, local regression filter and moving average filter. Since the difference between two time steps should be constantly given as the cycle time (60 ms) we implement the moving average filter which causes a delay in the prediction by relating the prediction to the history of the outcome probabilities. A suitable design is given as:

$$P(t) = \frac{1}{2} \cdot P(t_0) + \frac{1}{4} \cdot P(t_{-1}) + \frac{1}{8} \cdot P(t_{-2}) + \frac{1}{8} \cdot P(t_{-3}) \; .$$

The filter is designed smoothing the current prediction by the relation to three predicted values from the last three times steps in the past. Thus, every predicted value from the logistic regression is stored into a additional storage buffer of size four. The probability values from the storage buffer are weighted differently according to the time point of recording. We use the moving average filter because it has a comparable performance as the

other approaches. It is very simple to implement and does not require a lot computation and storage resources.

7 Extension to Dynamic Bayesian Network

The above sections 4.7, 5 and 6 described the approach of static BN using the current incoming data. The data are processed and predicted by linear regression to make a statement about the driving intention of the EGO and of the detected OBJ vehicles. In addition, the output probabilities are predicted using logistic regression to perform a even earlier recognition of a LC maneuver.
In this chapter, a extended approach using DBN is presented where the trend analysis is encapsulated into the network by extending particular fragments, showing trend features, to dynamic fragments. DBN model performs the inference based on information of past and presence. It takes into account the temporal relationships between successive system states, which are utilized for the trend analysis over consequent time steps in a big set of time series. The main advantage of DBN models is the compensation of missing measurement value. The current measurement is smoothed by the temporal reasoning. Thus, the recognition of traffic maneuvers using DBN has a better robustness compared to the static BN. In order to improve the timegain of recognition, it is possible to implement single-step-ahead or even multiple-step-ahead time series forecasting based on the temporal reasoning of multiple time steps in the past. The disadvantage of DBN is the high complexity by adding more than one time slice which requires more computation time and memory space compared to the static BN.

7.1 Dynamic Extension of Lateral Evidence

There are three hypothesis (*LE*, *TRAJ* and *REL_DYN*) in the developed knowledge-based BN model which naturally exhibit trend features and can be extended to dynamic fragments. Because of the computation resources and the quality of the data we only implement the dynamic extension to the BN-fragment *LE* which indicates trend features and a high data quality. The data of *REL_DYN* are not given consistently since the model considers the relative longitudinal dynamics between two vehicles following each other in the same track and this situation is not always present. The data consistency

problem is caused by the fact that the front-hidden vehicle can be lost from time to time. This occurs when it drives out of the detection field of view of the sensors. The physical model of $TRAJ$ is based on the extrapolation of the driving trajectory which already includes the tracking of the past positions of the vehicle for the calculation of the situation features $TLCR$, PSI and $ALAT$ (see 4.7.4).

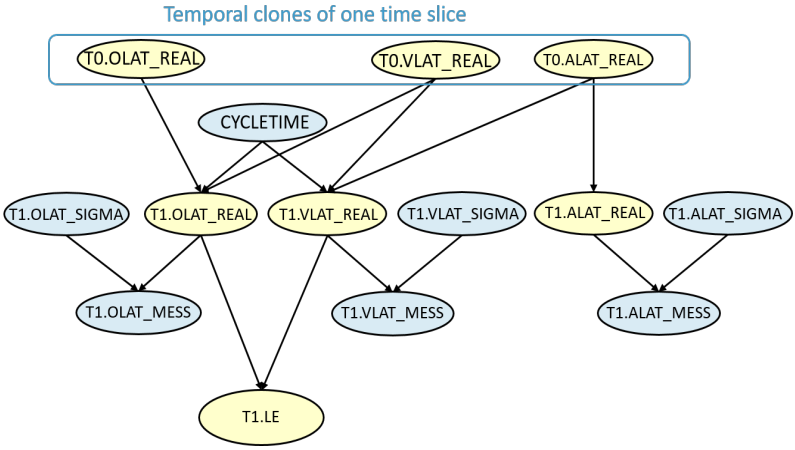


Figure 7.1: LE_DBN

Figure 7.1 shows the two steps time-sliced (2T-$DBNs$) LE_DBN fragment. It is characterized by the LE model over the feature variables $OLAT\_REAL$ and $VLAT\_REAL$. These features represent the expected values which are inferred from the measurement variables ($OLAT\_MESS$, $VLAT\_MESS$) and their uncertainties ($OLAT\_SIGMA$, $VLAT\_SIGMA$). In addition the state $ALAT$ denoting the currently maximal possible acceleration is added to complete the kinematic model of the movement process. Exactly one Temporal clone of each variable is created to model the successive process over time. The extended dynamic model satisfies both the first-order Markov assumption (see equation 2.3) that the next state is dependent only on the current state, and the stationary assumption that the transition probabilities are independent from the actual time step where the transitions is taking place. The relation between the time slices are represented by the transitional

probability table (TCPT) which is initialized for the LE_DBN fragment using normal distribution $\mathcal{N}(\mu, \sigma^2)$ (according to [27]):

$$T1.OLAT\_REAL \sim \mathcal{N}(T0.OLAT\_REAL + T0.VLAT\_REAL * \Delta T, 0.5^2) ,$$
$$T1.VLAT\_REAL \sim \mathcal{N}(T0.VLAT\_REAL + T0.ALAT\_REAL * \Delta T, 0.5^2) ,$$
$$T1.ALAT\_REAL \sim \mathcal{N}(T0.ALAT\_REAL, 0.5^2) .$$

The mean value μ is approximated by the expectation given by kinematic formula:

$$s(T_1) = s(T_0) + v(T_0) \cdot \Delta T + \frac{1}{2} \cdot a(T_0) \cdot \Delta T ,$$
$$v(T_1) = s(T_0) \cdot \Delta T + \frac{1}{2} \cdot a(T_0) \cdot \Delta T .$$

The (lateral) acceleration factor is neglected for $OLAT$ since its quadratic is near zero. The standard deviation σ is initialized with 0.5 which is resulted from based on knowledge-based data analysis. The distribution is implemented for each particular combination of states within the given ranges which are discretized by 10% variation and truncated on the interval. Truncation means to cut the ∞-tails of the normal distribution and to define the function values only on the predefined range:

- $O\_LAT\_REAL \in [-1, 2]$,
- $V\_LAT\_REAL \in [-1, 2]$,
- $A\_LAT\_REAL \in [0, 1.5]$.

For the initialization of the TCPT we follow in this context the basic idea that we use the most suitable knowledge-based relation given from the physical and logical modeling since. Based on this initialization we use machine learning to determine the relation between temporal nodes.
Furthermore, in the LE_DBN fragment (Fig. 7.1) we have modeled explicitly a variable called $CYCLETIME$ which represents the sampling frequency of the stereo camera. This additional variable is necessary for the modeling of time difference ΔT between two time slices. The currently recorded data must be correct synchronized between the time-slices in the DBN at every time step.

7.2 Learning of Parameters in a Dynamic Bayesian Network Fragment

For the learning procedure of DBN we need to preprocess the learning data set. In the case of learning LE_DBN we clone the relevant variables $OLAT$ and $VLAT$ in the data set where a new column for each variable with measurements values of the corresponding time step from the time series is created. After that the data set is transformed into the longitudinal representation such that each line contains an entire sequence (see Fig. 7.3). The columns represent the variables and their temporal clones, e.g. $OLAT$ is extended to $OLAT(t_1)$, $OLAT(t_2)$, $OLAT(t_3)$...$OLAT(t_n)$, where $n = \text{length}(Seq_i)$ (as shown in Fig. 7.3). Furthermore, the entire sequence must fit into the model as a single case, i.e the model has to be fitted with m time slices (=temporal clones) for each of the variables $OLAT$ and $VLAT$, where m is the maximal length of the given sequences.
We use two methods to learn a DBN fragment. The first is using the EM-learning (according to [18] and [12]). In this context the dynamic LE fragment is rolled out into a corresponding static BN model by adding the same number of time slices given in the data set. Thus, it can be treated accordingly by using the EM algorithm to determine the probability parameters in the TCPTs between time slices as well as in the CPT of the hypothesis node LE which is conditioned to the evidence nodes ($T1.OLAT\_REAL$ and $T1.VLAT\_REAL$).
An alternative method is using the Adaptation approach (according to [12] and [17]). For the learning of the $LE\text{-}DBN$ fragment, we define a fixed model of two time steps ($2T\text{-}DBN$), i.e a single temporal clone of each input variable ($OLAT$, $VLAT$, $ALAT$) is created. The process performed by the Adaptation is to update the parameters of all states X in a TCPT for a particular parent configuration π_i whenever an observation of this parent configuration is given as evidence. The sequential updating process is based on statistical counting. Consider the generalized representation of a TCPT given in Figure 7.2, where the states $(x_1,...,x_n)$ and the parent configurations $(\pi_1,...,\pi_m)$ are given. We add each a row to store the current sampling size $(s_1,...,s_m)$ and the fading factors $(q_1,...,q_m)$. The current sampling size are given as the sum of counts $s_i = N_1 + ... + N_n$ for a particular parent configuration π_i where a count N_k is the counted frequency of the state x_k given in the column. When for example a observation $P(e|\pi_i)$ of state x_k for the parent configuration π_i is given as evidence, the Adaptation-algorithm updates e.g. for the states

of the variable $T1.OLAT$ conditioned to a certain parent configuration π_i (given in this case by the states of $T0.OLAT$ and $T0.VLAT$):

$$P(T1.OLAT = x_k|\pi_i) := \frac{N_k \cdot q_i + P(e|\pi_i)}{s_i \cdot q_i + 1} , \tag{7.1}$$

for the state x_k in the column of the parent configuration π_i and for all other states $x_j, j \in [1,n] \setminus k$ in the same column:

$$P(T1.OLAT = x_j|\pi_i) := \frac{N_k \cdot q_i}{s_i \cdot q_i + 1} . \tag{7.2}$$

States\Config.	$\boldsymbol{\pi_1}$	$\boldsymbol{\pi_2}$	...	$\boldsymbol{\pi_m}$
$\boldsymbol{x_1}$	$P(x_1\|\pi_1)$	$P(x_1\|\pi_2)$	...	$P(x_1\|\pi_m)$
$\boldsymbol{x_2}$	$P(x_2\|\pi_1)$	$P(x_2\|\pi_2)$	...	$P(x_2\|\pi_m)$
. . .	. . .	. . .	... ...	. . .
$\boldsymbol{x_n}$	$P(x_n\|\pi_1)$	$P(x_n\|\pi_2)$	...	$P(x_n\|\pi_m)$
experience table	s_1	s_2	...	s_m
fading table	q_1	q_2	...	q_m

Figure 7.2: Generalized representation of a TCPT by inserting the experience and fading table for adaptation

The way of counting is done at each time step in and for all sequences in the transposed data set (see 7.3). As result we obtain a TCPT which is particularly configured for the given data set.

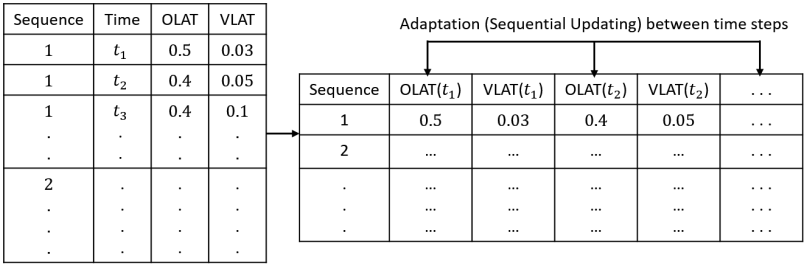


Figure 7.3: Data transposition for the Adaptation

In this work, we obtain better results by combining both methods to determine the quantitative part (parameters) of the DBN fragment. We use the EM algorithm to learn the causal relation between the information variables in the input and the hypothesis variable (*LE*) in the output. The learned CPT is exported and inserted into the adapted model. The reason of combining these two methods is that the EM algorithm fits better on the developed labeling strategy by using empty spaces to label the uncertain data. For the transposed data set, it is more efficient to update the parameters by adapting the measurement data at every single time step instead of rolling out the network to estimate the relation between the time steps using the EM algorithm.

Moreover, it is necessary that the TCPT is adapted for each vehicle side (EGOLEFT, EGORIGHT, OBJLEFT, OBJRIGHT) separately using the measurement data labeled for the vehicle side. The reason is, that the data measured for the successive driving process in highway traffic are different according to the type of vehicle (EGO or OBJ) and to the driving direction of the vehicle (LEFT or RIGHT). A maneuver of crossing the lane to the left is in the most cases (in our recorded data set) a overtaking maneuver where the velocity is increasing and the dynamic trend builds up very quickly compared to a maneuver of crossing the lane to the right side. A LC to the right side is usually accompanied by decreasing velocity as we observe in the recorded data sequences. On the highway a LC to the right is usually a movement from a faster lane to a slower lane. Moreover, the different data quality of EGO and OBJ vehicle must be also taken into account for the dynamic models. If we consider the clustering plots .6 and .7 in the appendix, the data denoting FOLLOW maneuver of the OBJ vehicle covers,

because of the measurement noisy, a much larger area compared to the data measured from the EGO vehicle.
If we learn the CPT between the information variables and the hypothesis variable, it is not necessary to divide the fragment for each vehicle side, since we consider in this case a static binary classification problem where the uncertainty in the data is specified in the modeling (see Section: 4.7.2), but not the dynamic trend. If we want to take the dynamic trend into account and adapt the TCPT for describing the relationship of successive time steps, we have to consider the difference in the development of the measurement data for different vehicle sides. Therefore, we divide the *LE_DBN* fragment into *LE_DBN_EGOLEFT*, *LE_DBN_EGORIGHT*, *LE_DBN_OBJLEFT*, *LE_DBN_OBJRIGHT* and adapt the TCPT for each vehicle side using the separated data set. The resulted classifier is called $DBN\_LE4fragm$ which contains four *LE*-hypothesis C-files for each vehicle side including the differently adapted TCPTs (corresponding data for each vehicle side) and the same CPT which is determined using the EM-learning (entire data including four vehicle sides).

8 Data-driven Model

Development of self-learning and -diagnostic systems becomes an essential subject in the latest trend in automotive processing of streaming data. Thus, in order to address these challenges the idea of running an alternative completely data-driven model in parallel with the knowledge-based models is considered in this chapter. A Naive Bayesian Network (NBN) is constructed under the assumption that each information variable is only dependent on the selected target variable (according to [27]). As typical for NBN, all information variables in the input layer are directly associated with the target variable. There is no logic based structure. The causal relationships between the input variables and the target variable are learned from the data. Through a suitable feature selection algorithm, we systematically rank all observable variables given in the data set by their influence on the target variable. We form the input layer by selecting a certain number of the variables which have high impact on the target variable (e.g. the first 30 variables in the ranking of influence). The causal relationship between the information variables and the target node is systematically learned by using the EM algorithm for the static model and by adaptation for the dynamic model. The procedure can be structured as follows:

1. Use a "rule-labeled" data set with the same features given for the knowledge-based models (e.g ORIG). The discretized states of all features are retained.

2. Feature selection and supervised discretization (IEM) where $HQMVT$ is used as target node.

3. Connection of all input variables to the target node and creation of temporal clones of all variables.

4. EM-learning of CPTs denoting the causal relation between the target variable $HQMVT$ and all input variables. Adaptation of TCPTs between two time-slices.

In general the data-driven models have faster inference (e.g. dynamic NBN 0.025 ms, compared to ORIG-DBN with 0.149 ms) and need less memory space compared to the knowledge-based models given in the previous sections [27]. However, the number of False Positives (FP), i.e. wrong recognized FOLLOW maneuver, is currently much higher than the evaluation of knowledge-based models. The recognition of LC maneuvers is comparable of both, the data-driven models and the knowledge-based models.

8.1 Feature Selection

Feature selection, also known as variable selection, is an important task in the process of creating a data-driven model. In the work of this thesis, we perform a maximum dependency selection to find the variables which have most impact on the hypothesis variable $HQMVT$.
We use a score based feature selection algorithm which is related to the cross entropy (mutual information). The algorithm belongs to the filter methods which is independent of the underlying network model (according to [17] and [8]). The entropy of a variable is the degree of chaos in the distribution of the variable which can be considered as a measure of randomness. The entropy of a random variable A is given by (according to [4])

$$H(A) = -\sum_{A} P(A) \log P(A) \geq 0 \,. \tag{8.1}$$

If the variable is normally distributed, we have the maximum of entropy. On the other hand, if the probability distribution of the variable is located in a single state, we have the minimum 0 of entropy [4].
We implement the event variable $HQMVT$ in the output and determine the influence of each information variable Y given in the data set on $HQMVT$. The cross entropy is given by (according to [4])

$$I(Y,HQMVT) = H(HQMVT) - H(HQMVT|Y) \,, \tag{8.2}$$

$$= \sum_{y \in \mathcal{Y}} \sum_{x \in HQMVT} P(x,y) \log \frac{P(x,y)}{P(x)P(y)} \,. \tag{8.3}$$

It denotes the reduction of entropy in $HQMVT$ by observing states of Y. Thus, we can compare the strength of dependency between different pairs of the output event $HQMVT$ and each information variable in the data

(event-features pairs). In order to do that, we use the normalized symmetric uncertainty [18]

$$U(HQMVT,Y) = \frac{2 \cdot I(HQMVT,Y)}{H(HQMVT) + H(Y)} \,. \tag{8.4}$$

At the end of the feature selection, the computation results of all variable pairs are stored into a ranking list.

8.2 Naive Bayesian Network

Giving the classification problem for the output event and the selected input variables after their influence on the event variable, we can construct the Naive Bayesian Network (NBN). It represents a particularly useful data-driven model for handling big data classification problems. The construction of a Naive Bayesian Network model (as shown in Fig. 8.1) follows two basic assumptions:

1. Input variables are independent each other

2. Output node is connected directly to each input node.

We select as input a certain number of information variables $Y_1, ..., Y_i$ in the ranking list according to their strength of dependency with the output event $HQMVT$. The input variables are directly connected to the output variable without hidden layers. Their causal relationships are described using a CPT as given in Table 8.2 where all states of the input variable are conditioned to the maneuver classes of $HQMVT$. The probability parameters in the CPTs are learned using EM algorithm (see Section 2.4.3).

The NBN can be dynamically extended by adding temporal clones to each node (Fig. 8.3). The output event $HQMVT$ is not only related to the input variables but also to their temporal slices. Thus, the calculation using dynamic NBN is generally more robust to the uncertainty in the measurement data as compared to the static NBN. We use the Adaptation approach (see Section 2.4.4) to learn the TCPTs.

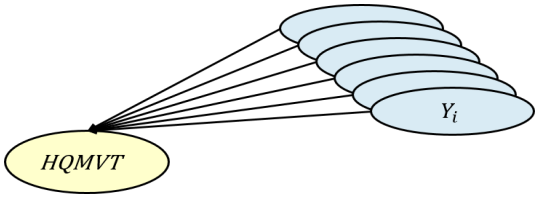


Figure 8.1: Naive Bayesian Network with selected input variables and the output event $HQMVT$

$Y_i \backslash HQMVT$	DONT-CARE	OBJ-CUTIN	OBJ-CUTOUT	EGO-CUTIN	EGO-CUTOUT	OBJ-FOLLOW	LANE-FOLLOW
x_1	θ_{10}	θ_{11}	θ_{12}	θ_{13}	θ_{14}	θ_{15}	θ_{16}
. . .		. . .	. . .	. . .	. . .	. . .	. . .
x_n	θ_{n0}	θ_{n1}	θ_{n2}	θ_{n3}	θ_{n4}	θ_{n5}	θ_{n6}

Figure 8.2: CPT for describing the causal relationship between each input information variable and the output event variable $HQMVT$

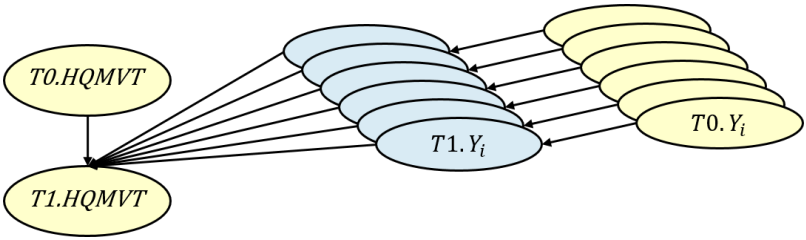


Figure 8.3: Dynamic extension of the Naive Bayesian Network Model

9 Statistical Evaluation of all developed classifiers

9.1 Evaluation Method

The developed evaluation method combines two modules. The module 1 is based on the recognition of the maneuver class in the highest logic layer ultilizing the vehicle-vehicle-relationship. This evaluation module is developed and used in [14]. The basic idea of scalability is that the developed model only considers a pair vehicle-vehicle-relationship between the EGO and one OBJ detected by the sensors. Therefore, if e.g. an OBJ vehicle performs a CUTIN maneuver and drives into the lane in front of EGO, the relation between both vehicles must be changed from LANEFOLLOW to OBJFOLLOW. We use this logic and check through the labeled data set in order to evaluate the performance of a classifier. The results are summarized as confusion matrix. The positive class represents the cases with LC maneuver. They can be either correctly recognized (TP: true positives) or not recognized (FN: false negatives). The negative class represents the FOLLOW maneuvers. They are correctly recognized (TN: true negatives) if the system does not send any signal of a LC, or can be wrongly recognized (FP: false positives) if the probability of LC exceeds the defined threshold of 65%. In this context, the confusion matrix can be established by considering both the change in the states and the overcoming of threshold (according to [14]):

- True Positive (TP): "correctly recognized", if LC probability is greater than 65%, and the state changes as followed: OBJFOLLOW → OBJCUTOUT or EGOCUTOUT → LANEFOLLOW,
  LANEFOLLOW → OBJCUTIN or EGOCUTIN → OBJFOLLOW.

- False Positive (FP): "wrong recognized", if LC probability is greater than 65%, and the state changes as followed:
  OBJFOLLOW → OBJCUTOUT or EGOCUTOUT → OBJFOLLOW,

LANEFOLLOW → OBJCUTIN or EGOCUTIN → LANEFOLLOW.

- False Negative (FN): "not recognized", if LC probability is less than 65%, and the state changes as followed:
  OBJFOLLOW → LANEFOLLOW,
  LANEFOLLOW → OBJFOLLOW.

- The rest is given as True Negative (TN), if there is no change in the states.

The second evaluation module closely examines the vehicle-lane marking-relationship. We consider two time points which are based on the actual measurement values (also used in the methods of data labeling in Section 5.1). The first time point is defined as lane marking touching (LMT) which is the starting time point where the vehicle enters the target lane. The second time point is named as lane marking crossing (LMC) where the mid bumper of the considered vehicle has penetrated the target lane so far such that the change of coordinate systems is caused. We use LMC as the definition of an accomplished LC maneuver. In this context, we again calculate the transgression of the defined threshold for LC probability and compare it to the both time points LMT and LMC in order to evaluate the classifier performance given as accuracy and timegain.

Accuracy is the percentage between the numbers of correctly recognized maneuver and the total number of the maneuver sequences given in the test data. A LC maneuver is correctly recognized if the corresponding probability reaches the threshold before the time point of LMC, whereas a FOLLOW maneuver is correctly recognized if the defined threshold of LC probability is never exceeded during the entire sequence.

The timegain of the developed BN classifier is calculated twice according to both time points of LMT and LMC. The time difference between the recognition and the LMC is established as the effective time win before a maneuver is accomplished. The time difference compared to the LMT is referred as the timegain to vehicle side, which provides information about the time win before the vehicle starts to enter the target lane. It is important for this thesis because we want to develop a classifier that can recognize the driving maneuver when the vehicle is still on its own lane.

9.2 Error Causes

This section gives a brief illustration about the error causes which mostly affect the classifier performance. They can be divided into two categories.

1. Incorrect input data:
   This category is independent from the modeling and parameterization of the BN since the errors are already established in the input data and cannot be compensated by the treatment of uncertainty or network logic. The most relevant input data are the relative distance and velocity between the vehicle and lane marking since we use the lane marking as the reference system of the developed models. However, the correct reconstruction of the virtual lane marking is not always fulfilled by the image processing in case of detection uncertainties and weakness of the stereo camera. Weather conditions, bad road conditions, as well as a large gradient and a strong curvature of the roadway can lead to the fact that the course of the lane markings cannot be clearly estimated, which in turn lead to incorrect computation of the relative lateral dynamics between the vehicle and the lane marking.

2. Incorrect parameterization of the Bayesian Network:
   The trade-off problem between the competitive requirement criteria of maximum accuracy and maximum timegain leads to difficulty in the parameterization of the network parameters, especially for the static BN models. The maneuvers that are taking place should be recognized not only correctly without any false positives, but also at an early stage. If the timegain is too much prioritized, the number of false positives will quickly exceed the limit of acceptable accuracy level. On the other hand, the model is only useful if a relevant maneuver is detected early enough, such that the vehicle control systems still have enough time to react at it.

9.3 Evaluation Results

For the evaluation we use a clean separated data set of 353 maneuvers. The balanced data set contains 178 lane change maneuver sequences and 175 follow sequences (including OBJFOLLOW and LANEFOLLOW). Data of low quality where errors occur in the sensor measurements as well as the follow-relevant data are excluded from the test data set. Follow-relevant data denote the situations where the considered vehicle touches or crosses the lane marking many times in the recording, but does not carry out the lane change at the end. They are excluded because there is no systematic approach to label them either to the positive class of LC or to the negative class of FOLLOW. In the following, the evaluation results are presented as table. We compare the performance of accuracy and timegain of the original OOBN and each of the best developed classifiers using static BN with trend analysis, DBN and NBN.
Figure 9.1 shows an overview over the evaluated classifiers: ORIG (with original and optimized parameters), static knowledge-based model with Linear (STAT_LinReg), DBN model (DBN_4fragm) where LE is divided into four fragments denoting each vehicle side (EGOLEFT, EGORIGHT, OBJLEFT, OBJRIGHT) and the dynamic NBN which is a data-driven model. STAT_LinReg and DBN_4fragm are extended using logistic regression to obtain even earlier recognition by predicting the probability of LC maneuver. Tables 9.1, 9.2, 9.3 and 9.4 provide detailed evaluation results of the knowledge-based classifiers where the number of TP and FN of the positive class denoting the LC maneuver as well as the number of TN and FP of the negative class denoting the FOLLOW maneuver are counted in the evaluation results. The time performance is evaluated due to starting point of a LC (timegain until LMT) and the end of an actual LC (timegain until LMC).
To evaluate the performance of static and dynamic knowledge-based models, we have transferred the classifiers to C-code and deployed them on Linux platform off-line and on the automotive platform of the experimental vehicle. Figures 9.2 and 9.3 show examples of the visual evaluations where a CUTIN sequence is recognized using implemented knowledge-based models STAT_LinReg and DBN_4fragm. As comparison, the same sequence recognized by the ORIG model is presented in the appendix .4. The results provide a basic understanding of the timegain performance. We can notice for example the different recognition (cycle) time using different classifiers with different threshold settings (35% and 65%, red arrow) as well as using the prediction of logistic regression (green arrow).

The NBN is only implemented in the graphical user interface of the software tool HUGIN and not integrated into the C-framework since we cannot stabilize the output for FOLLOW maneuver. Thus, the evaluation of the data-driven model is performed within the HUGIN tool using the test data set.

Comparing the evaluation results given in the tables 9.1 and 9.2, we notice, that the knowledge-based DBN classifier exhibits an higher accuracy (99.43% vs. 99.15%) and an improved timegain (1.13 vs. 1.05 s) compared to the ORIG model with optimized parameters. The knowledge-based static BN model with implementation of linear regression to the *LE* fragment exhibits a significant improvement in the timegain (1.4 s) with an acceptable deterioration (about 96%, as shown in Table 9.2) in accuracy. If we consider the situation given in Figures 9.2 and 9.3, we can notice the importance of the earlier recognition. The STAT_LinReg classifier recognizes the LC maneuver in this video sequence five time cycles (about 0.3 s) earlier than the DBN_4fragm classifier. In the earlier recognized time point, the OBJ vehicle with ID 31 is still in its own lane and starts the movement towards the lane marking (Fig. 9.3), where in the other scene (Fig. 9.2) the OBJ vehicle is immediately before entering our EGO lane.

The implementation of logistic regression improves as expected the timegain. The DBN_4frag classifier shows still a good accuracy with improvement in the timegain of three time cycles (about 0.16 s as shown in Fig. 9.1). In the case of the STAT_LinReg classifier, the timegain is improved from 1.40 to 1.54 s. The accuracy in the recognition of FOLLOW maneuver is reduced to 88.6% (Fig. 9.1).

				Performance with logistic regression		
Classifier\Performance	**Lane Change**	**Follow**	**timgain [s]**	**Lane Change**	**Follow**	**timegain [s]**
ORIG OOBN	96.1%	98.3%	0.77			
ORIG OOBN with optimized parameters	98%	100%	1.05			
DBN_4fragm with **logistic regression**	98.9%	100%	1.13	99.4%	98.4%	1.29
STAT_LinReg with **logistic regression**	99.4%	96.2%	1.40	99.4%	88.6%	1.54
Dynamic **NBN**	99.4%	55.2%	2.13			

Figure 9.1: Evaluation result overview

The dynamic NBN classifier exhibits a very good performance in the recognition of LC maneuver, but an unacceptable error rate in the recognition of FOLLOW maneuver. Only 55.2% of the FOLLOW sequences are correctly recognized (as shown in Fig. 9.1). That is also the reason, why we didn't implement the data-driven models in the experimental vehicle. We suspect that the model could be on the one hand "overfitted" because we didn't perform analysis according to the number and characteristics of the selected input variables. They are solely determined by the algorithms. On the other hand, the problem domain in the output including six maneuver classes and a complementary quantity is too complex for the case that the logic layers, which provides additional knowledge about the dependencies of the selected variables, are missing. It would be more compatible to reduce the complexity of the classification problem, e.g. to a binary classification problem.

Table 9.1: Classifier: **ORIG OOBN**

	MNVR all: 353			
	Positive Class all:178		Negative Class all:175	
	TP	**FN**	**TN**	**FP**
OBJCUTIN	47	4		
EGOCUTIN	42	0		
OBJCUTOUT	43	3		
EGOCUTOUT	39	0		
LANEFOLLOW			115	1
OBJFOLLOW			58	1
Recognition all	171		173	
Error Rate	3.93%		1.14%	
Accuracy	97.45%			
timegain (cross)	0.773 s			
timegain (touch)	−0.227 s			

Table 9.2: Classifier: **ORIG OOBN** with optimized parameters

	MNVR all: 353			
	Positive Class all:178		Negative Class all:175	
	TP	**FN**	**TN**	**FP**
OBJCUTIN	48	3		
EGOCUTIN	42	0		
OBJCUTOUT	46	0		
EGOCUTOUT	39	0		
LANEFOLLOW			116	0
OBJFOLLOW			59	0
Recognition all	175		175	
Error Rate	1.68%		0%	
Accuracy	99.15%			
timegain (cross)	1.05 s			
timegain (touch)	0.038 s			

Table 9.3: Classifier: **STAT_LinReg**

	MNVR all: 353			
	Positive Class all:178		Negative Class all:175	
	TP	**FN**	**TN**	**FP**
OBJCUTIN	50	1		
EGOCUTIN	42	0		
OBJCUTOUT	46	0		
EGOCUTOUT	39	0		
LANEFOLLOW			111	5
OBJFOLLOW			57	2
Recognition all	177		168	
Error Rate	0.56%		4%	
Accuracy	97.73%			
timegain (cross)	1.399 s			
timegain (touch)	0.351 s			

Table 9.4: Classifier: **DBN_4fragm**

	MNVR all: 353			
	Positive Class all:178		Negative Class all:175	
	TP	**FN**	**TN**	**FP**
OBJCUTIN	49	2		
EGOCUTIN	42	0		
OBJCUTOUT	46	0		
EGOCUTOUT	39	0		
LANEFOLLOW			116	0
OBJFOLLOW			59	0
Recognition all	176		175	
Error Rate	1.12%		0%	
Accuracy	99.43%			
timegain (cross)	1.126 s			
timegain (touch)	0.102 s			

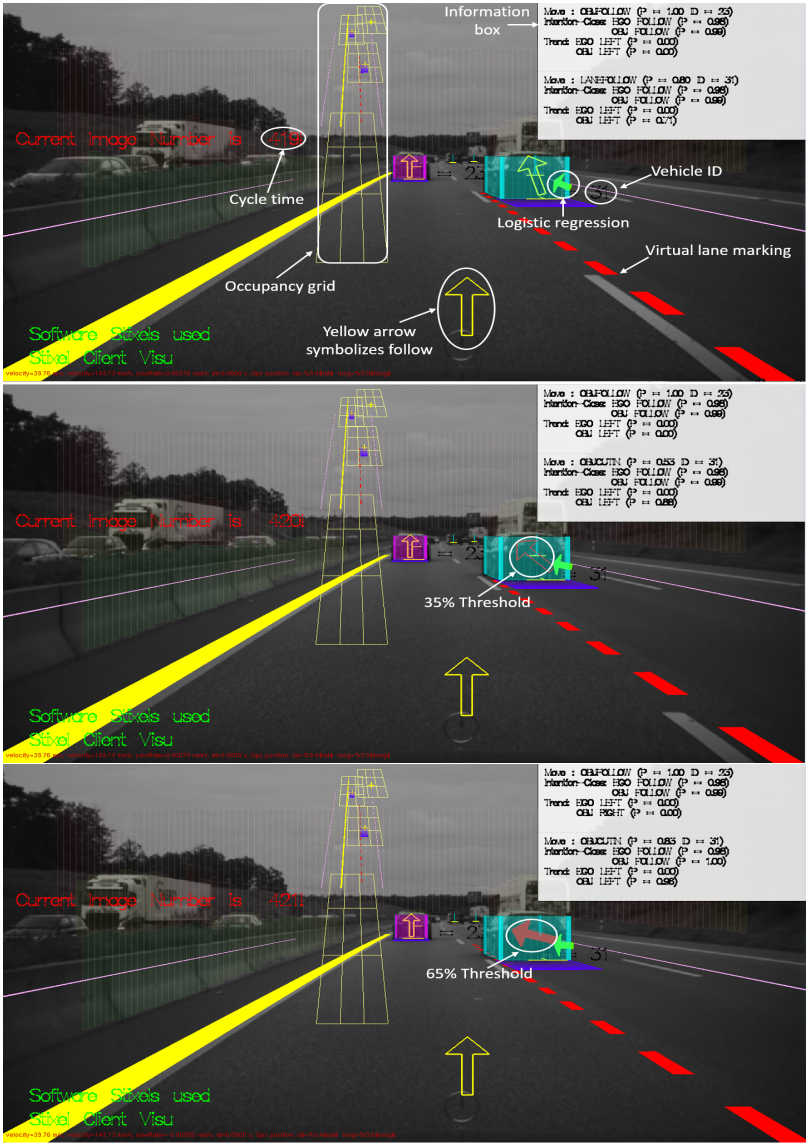


Figure 9.2: Visual Evaluation of DBN_4fragm

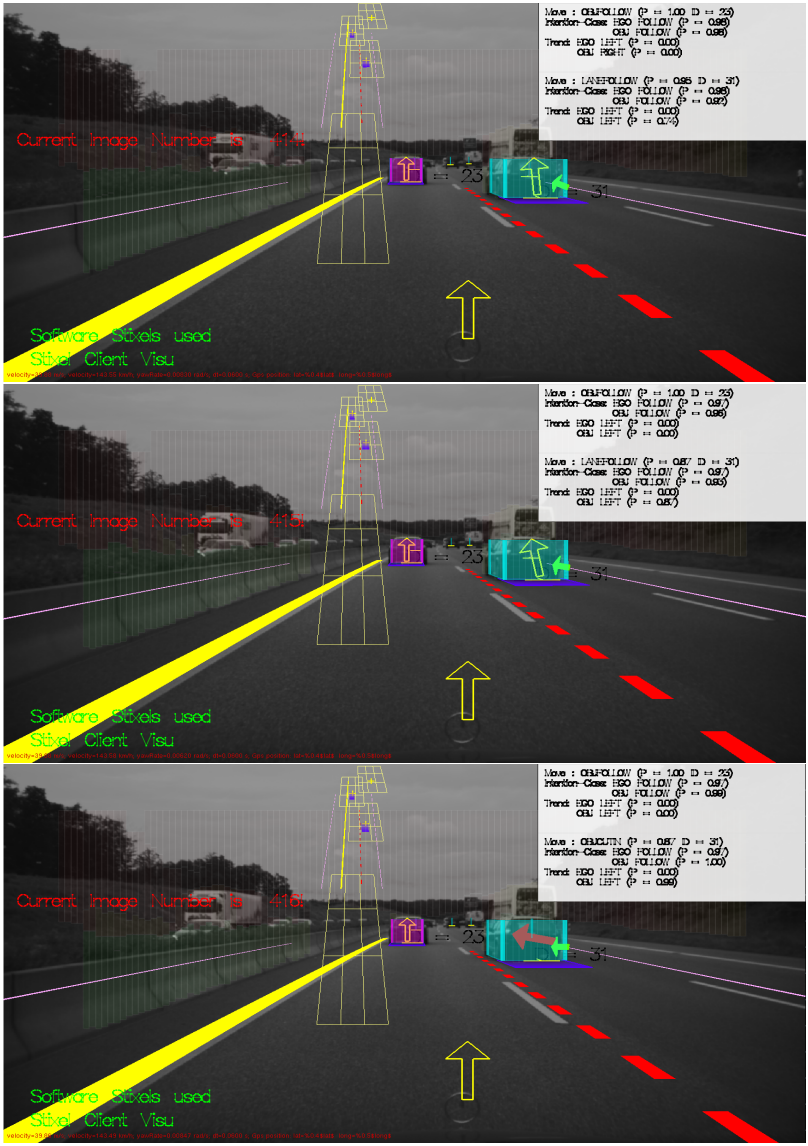


Figure 9.3: Visual Evaluation of STAT_LinReg

10 Concept to transfer the computed probabilities to Vehicle Control System

10.1 Integration into Adaptive Cruise Control

The first approach is the integration of the computed probability of maneuver classes into the ACC system with consideration of the relative lateral dynamics. The ACC system uses PID controller where the control parameters are affected by the calculation of the criticality. Up to six vehicles can be taken into consideration and the control object is the one with the highest criticality. The calculation of criticality is based on the relative longitudinal and lateral dynamics, where the parameters X_REL and V_REL in the longitudinal direction and the parameters $OLAT$ and $VLAT$ in the lateral direction are taken into account. In this context, we can replace the evaluation of the lateral dynamics with the BN models since we have shown in the previous chapters that the developed models have a very high accuracy and above all a good timegain.
A possible approach would be a combination of the output probability of the BN with the prediction of the time to collision (TTC_{pred}) between the EGO and each of the detected OBJ vehicle. Time to collision is a typical parameter based on the relative longitudinal dynamics between to successive driving vehicles. It is computed by the fraction of the measured relative distance Δs and relative velocity Δv [10]

$$TTC_{pred} = \frac{\Delta s}{\Delta v}$$

Based on TTC_{pred} and the output probability distribution of the predicted maneuver classes, we evaluate the criticality of each detected OBJ vehicle and take the necessary decisions for the different use cases. In the situation of a OBJCUTIN (Cutin Warning or active collision avoidance) e.g., the deceleration is activated if both thresholds (of TTC_{pred} and $P(OBJCUTIN)$) are exceeded. Another example would be the Lane Departure Warning. Instead of the calculation of the lane marking touching, we evaluate the

probabilities of EGO LC maneuver (EGOCUTIN and EGOCUTOUT) and simultaneously take the criticality of the OBJ vehicle in the target lane into account. By suitable parametrization, it would provide an improvement to the conventional Lane Departure Warning.

10.2 Extension of the Network Model

The second approach is to implement the criticality criterion in the BN model. A possible design is shown in Figure 10.1. We insert a new object-oriented fragment named *REL_DYN* which is a Boolean variable and represents the evaluation result of the relative longitudinal dynamics of the considered vehicle pair. In the existing network models, we already use a modeling of the longitudinal relative dynamics based on fuzzy logic (as described in Section 4.7.6). The relevance of a detected OBJ vehicle to the EGO vehicle, and the situation analysis, is given in the corresponding variable named *CRITICALITY* in a higher logic layer. Its conditional dependency to the evaluation results of longitudinal relative dynamics and the predicted maneuver is given by the CPT. In this case however, we can not determine the parameters using learning algorithms since we do not have collision data. Thus, the parameterization of the variable *CRITICALITY* is based on expert knowledge and simulation.

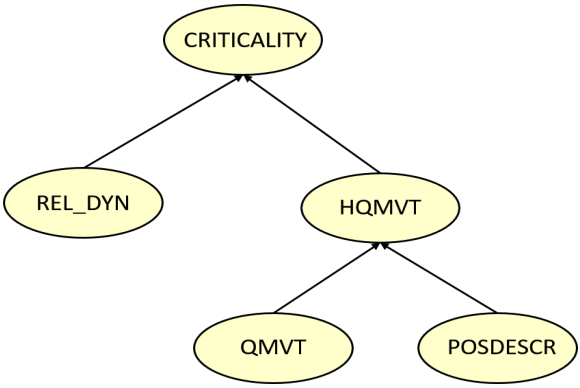


Figure 10.1: Modeling of situation criticality

11 Conclusion

11.1 Summarize

In this thesis we consider three modeling approaches for the prediction of relative driving maneuvers on the highway. The challenge is above all the consideration of the driving maneuver as a dynamic process and take the situation trend into account for earlier and accurate maneuver recognition. Moreover, we use supervised learning algorithm to determine the probability parameters. In this context, a big data set is collected from real highway traffic for the learning process.
At first we consider the knowledge-based static network model. The processed measurement data are evaluated in the hypothesis fragments in the input layer and the results are propagated as evidences to the higher logic layers. The intermediate logic layers representing different events are formulated by expert knowledge. In the output, a single event variable $HQMVT$ is defined including six maneuver classes: OBJCUTIN, EGOCUTIN, OBJCUTOUT, EGOCUTOUT, OBJFOLLOW, LANEFOLLOW and the complementary state DONTCARE. Using the Expectation Maximization algorithm, we systematically determine the probability parameters in the input hypothesis using a labeled learning data set (about 1000 maneuver sequences). In addition, we take the driving dynamics into account by predicting the lateral offset in the input layer and the lane change probability in the logic layer denoting the Vehicle-Lane relation. The prediction is based on the tracking of the trend by use of linear and logistic regression. The resulting Bayesian Network classifier shows a significant improvement in the timegain (1.1-1.5 s compared to 0.7-1.0 s at begin) either in the off-line simulation or in the on-line visual evaluation in the experimental vehicle. Furthermore, the regression approach solves the problem in the recognition of drifting maneuver where the vehicle drives very slowly over the lane marking. Due to the trade-off problem between recognition accuracy and timegain, there is a deterioration (about 3.5%) in the accuracy which is acceptable.
In the next stage, we extend the static model to a Dynamic Bayesian Network where the the trend analysis is considered within the network model. We

add temporal clones to the information variables of the hypothesis fragment Lateral Evidence (*LE*). The inserted temporal clones represent the same variable with the same states at one time step in the past. Thus, the temporal input information can be inherited over time steps. The causal relationship between two successive time steps is represented by the transitional probability table. We use the Adaptation approach to determine the probability parameters in the probability table. The Dynamic Bayesian Network model has a high accuracy (over 98%) and a better timegain (1.13-1.29 s) compared to the original static model. However, it requires more computational resource and storage space than the static model.
Besides the knowledge-based model, we also investigated the data-driven model. Using a feature selection algorithm based on the mutual information (cross entropy), the variable with most impact on the output event are selected and inserted into a network model, where the input information layer is directly connected to the output event without intermediate logic layers. The basic idea of doing this is to investigate a procedure where both the network structure and parameter are systematically learned from data such that stranger and complex classification problems can be modeled systematically. The resulted Naive Bayesian Network model has a good performance in recognizing the Lane Change maneuver, but its accuracy in the recognition of FOLLOW maneuver is not acceptable (only 55.2% accuracy). Because of different features and dependencies in the data, we need the logic knowledge-based layers such that the input data are grouped by their dependency, and only the necessary information is propagated through the network.
Summarized, we investigated in this thesis a procedure using probabilistic graphical models for the forecast and prediction of dynamically developed process, in this case the driving maneuvers on a highway. The qualitative part (network structure) is constructed using human knowledge and experience, whereas the quantitative part, namely the parameterization of conditional probabilities, is determined by the actual outcome for the corresponding parameter combination. Thus, we can use the same procedure to systematically construct systems for similar problems. The resulting model can be extended or divided for different use cases. We are able to retrace the plausibility of each solution.

11.2 Future Development

The future development would firstly focus on the extension of the existing modeling for special use cases such as CUTIN-Warning, Lane Departure Warning and Collision Avoidance.
The main disadvantage of the current model is the strong dependency on the existence of real or virtual lane markings which cannot be correctly recognized in particular situations. Therefore, it is of great importance that we use a robust reference system which is as little as possible influenced by the environment conditions. A possible system would be the EGO track which is formulated using the synchronization of sensor measurement and the High-Definition digital map.
Furthermore, the timegain of Dynamic Bayesian Network model can be improved using the strategy of multiple-step-ahead time series forecasting within the network based on the temporal reasoning of multiple time slices. For general purposes, it is necessary to find suitable learning algorithms to determine the network structure from data. As complement to the Naive Bayesian Network model which shows a high error rate in the recognition of FOLLOW maneuvers, we can associate and group the selected variables by their relative relationship. There are e.g. score-based and constraint-based methods for determining the Bayesian Network structure from data. Related explanations are e.g. given in [19], [17] and [15].

Bibliography

[1] Daimler AG. Mecerdes-Benz DISTRONIC PLUS. [Online; accessed 22-April-2017]: m.mercedes-benz.de/de_DE/distronic_plus/detail.html.

[2] Daimler AG. Mercedes-Benz Intelligent Drive. - Mercedes-Benz International. [Online; accessed 22-April-2017]: m.mercedes-benz.de/de_DE/distronic_plus/detail.html.

[3] Brett Davis. Intelligent Drive, Mercedes-Benz steps autonomous vehicle. [Online, accessed 22-April-2017]: http://performancedrive.com.au/mercedes-benz-steps-autonomous-vehicle-technology-1913/mercedes-benz-s-klasse-w-222-2013-intelligent-drive-assiste/.

[4] Thomas M. Cover and Joy A Thomas. Entropy, relative entropy and mutual information. *Elements of Information Theory*, pages:12–49, 1991.

[5] Peter J. Denning. Bayesian learning, *Research Institute for Advanced Computer Science NASA Arnes Research Center, American Scientist 77, No.3*. pages 125–174, 1989.

[6] VDA-Verband (der Automobilindustrie). Automation levels by VDA_BASt,SAE *International On-Road Automated Vehicle Standards Committee*.

[7] European Commission. EU-project, Analysis of Massive Data STreams. [Online, accessed 28-April-2017]: http://amidst.eu/.

[8] Nir Friedman and Daphne Koller. Learning Bayesian Networks from Data, *course slides, Probabilistic graphical models, Standford University*. pagees: 1-19, 2001.

[9] Z Ghahramani. Learning dynamic Bayesian networks. *Adaptive processing of sequences and data structures*, 1387:168–197, 1998.

[10] Andreas Hermann and Jörg Desel. Driving situation analysis in automotive environment. *Proceedings of the 2008 IEEE International Conference on Vehicular Electronics and Safety, pages 216-221, ICVES 2008*.

[11] Tommi Jaakkola. Machine Learning, course materials. *MIT OpenCourseWare, Massachusetts Institue of Technology*. 2016.

[12] Finn V. Jensen and Thomas D. Nielsen. *Bayesian Networks and Decision Graphs, Journal of Physics A: Mathematical and Theoretical, ISBN:978-0-387-68281-5*, volume 44. 2007.

[13] Yaochu Jin and Berhard Sendhoff. Pareto-based multiobjective machine learning: An overview and case studies. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, 38(3):397–415, 2008.

[14] Dietmar Kasper. Erkennung von Fahrmanövern mit objektorientierten Bayes-Netzen in Autobahnszenarien, *Dissertation der Mathematisch-Naturwissenschaftliche Fakultät der Eberhard Karls Universität Tübingen, Abteilung RD/FFA der Daimler AG Forschung und Vorentwicklung*. pages: 1-132, 2012.

[15] Daphne Koller and Nir Friedman. Probabilistic Graphical Models: Principles and Techniques, *The MIT Press, Cambridge, Massachusetts, London England, ISBN:0262013193*, 2009.

[16] Daphne Koller and Avi Pfeffer. Object-oriented Bayesian Networks. *In Proceedings of the Thirteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-97), Providence, Rhode Island, Standford University*, pages 302-313, 1997.

[17] Helge Langseth, Anders L Madsen, and Thomas D Nielsen. AMIDST Report, Analysis of Massive Data Streams: State of the art for learning in the context of the AMIDST framework; *Project no.:619209, Deliverable no.: D4.2,*. pages: 1-79, 29.02.2016.

[18] Steffen L. Lauritzen. The EM algorithm for graphical association models with missing data. *Computational Statistics and Data Analysis*, 19(2):191–201, 1995.

[19] Dimitris Margaritis, Sebastian Thrun, Christos Faloutsos, Andrew W Moore, Peter Spirtes, and Gregory F Cooper. Learning Bayesian Network Model Structure from Data, *School of Computer Science Carnegie*

Mellon University Pittsburgh, PA 15213, Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, . pages 1-104, 2003.

[20] John O Rawlings, Sastry G Pantula, and David a. Dickey. Applied Regression Analysis: A Research Tool, second Edition, *Springer, ISBN: 0387984542,.* pages: 1-40, 1998.

[21] Ulrich Schreyer. Daimler und Bosch: Autonomes Fahren kommt schneller - Wirtschaft - Stuttgarter Zeitung. [Online; accessed 22-April-2017]: http://www.stuttgarter-zeitung.de/inhalt.fahrzeugbau-daimler-autonomes-fahren-kommt-schneller.d1cbf9b2-270f-4029-9174-c4f3a96b21ac.html.

[22] David J. Spiegelhalter and Steffen L. Lauritzen. Techniques for Bayesian analysis in expert systems. *Annals of Mathematics and Artificial Intelligence*, 2(1-4):353–366, 1990.

[23] Viacheslav Tereshchenko. Parameteroptimierung von Bayes-Netzen für Fahrmanövererkennung in Autobahnszenarien Parameter optimization of Bayesian networks for maneuver recognition in highway scenarios, *University Stuttgart and Daimler AG, 2013*.

[24] Viacheslav Tereshchenko. Master Thesis, Relative object-object dynamics for earlier recognition of maneuvers in highway traffic, *Institute of Industrial Automation and Software Engineering, M. Weyrich, University Stuttgart,* pages: 1-69, 2014.

[25] Stevens Wang. Praktikumsbericht, *University Stuttgart and Daimler AG,* pages: 1-35, September, 2016.

[26] G. Weidl, A. L. Madsen, and S. Israelson. Applications of object-oriented Bayesian networks for condition monitoring, root cause analysis and decision support on operation of complex continuous processes. *Computers and Chemical Engineering*, pages:1997-2009, 2005.

[27] Galia Weidl and Anders L Madsen. AMIDST Report, Analysis of Massive Data Streams: Report on demonstration of prototype, Prject no:.619209, Deliverable no.:D6.4, pages: 1-41. 30.12.2016.

[28] Galia Weidl, Anders L Madsen, Stevens Wang, Dietmar Kasper, Viacheslav Tereshchenko, and Wei Zhang. Situation Awareness and Early

Recognition of Traffic Maneuvers. *9th EUROSIM Congress on Modeling and Simulation*, pages: 1-9, September, 2016.

[29] Hermann Winner, Stephan Hakuli, Felix Lotz, and Christina Singer. Handbuch Fahrerassistenzsysteme, Grundlagen, Komponenten und Systeme für aktive Sicherheit und Komfort, *3. Auflage, ISBN: 978-3-658-05733-6, Springer Vieweg*. 2015.

12 Appendix

.1 Work Procedure

1. Optimization of the sigmoid parameters of the logistic function for the knowledge-based Static Bayesian Network, representing the relative dynamics in highway traffic. Expectation Maximization Learning of Parameters of the Static Bayesian Network for early recognition of maneuvers.

2. Output Filtering, Probability-Trend analysis of Static Bayesian Network and Maneuver Prediction using logistic regression.

3. Early and precise recognition of vehicle maneuvers by use of Dynamic Bayesian Network. Expectation Maximization Algorithm for finding suitable parametrization.

4. Off-line Learning and adaptation of Transitional Conditional Probability Distribution Parameters from real traffic maneuver sequences.

5. Analyzing of Data-driven models as comparison to the knowledge-based models.

6. Statistical evaluation of the recognition performance in comparison between Static and Dynamic Bayesian Network, between learned and adapted models.

7. Concept for transfer of the computed probabilities of recognized maneuvers of surrounding vehicles to the control modules for autonomous driving.

The results will be integrated in a scalable framework for the analysis of streaming automotive data. The framework represents an independent software system, which allows analysis, simulation and extensive testing of the recognition capabilities, and will serve as a basis for the deployment on the experimental vehicle of the obtained results. The results of the final deployment in the vehicle will be statistically evaluated with data stream, representing real traffic situations. The master work will also perform a statistical evaluation of the initially set automotive requirements on recognition accuracy, time win of recognition as compared to actual lane marking crossing. This will be performed in comparison to other earlier developed statistical classifiers for maneuver recognition, based on Static and Dynamic Bayesian Networks.

.2 Conditional Probability Tables of the knowledge-based Variables

In this section the CPTs of the knowledge-based variables are given in the order from the higher logical layers to the lower logical layers. The probability parameters within the CPTs are defined on the basis of the logical relations between the modeled variables.

POSDCR	POSLEFT									POSLEFT	POSRIGHT
QMVT	LL	LR	LG	RL	RR	RG	GL	GR	GG	...	...
DONCARE	1	1	0	1	1	1	1	0	0	...	...
OBJCUTIN	0	0	0	0	0	0	0	1	0	...	...
EGOCUTIN	0	0	1	0	0	0	0	0	0	...	...
OBJCUTOUT	0	0	0	0	0	0	0	0	0	...	...
EGOCUTOUT	0	0	0	0	0	0	0	0	0	...	...
LANEFOLLOW	0	0	0	0	0	0	0	0	1	...	...
OBJFOLLOW	0	0	0	0	0	0	0	0	0	...	...

Figure .1: Partial representation of the CPT of $HQMVT$

EGO.LC	L			R			G		
OBJ.LC	L	R	G	L	R	G	L	R	G
LL	1	0	0	0	0	0	0	0	0
LR	0	1	0	0	0	0	0	0	0
LG	0	0	1	0	0	0	0	0	0
RL	0	0	0	1	0	0	0	0	0
RR	0	0	0	0	1	0	0	0	0
RG	0	0	0	0	0	1	0	0	0
GL	0	0	0	0	0	0	1	0	0
GR	0	0	0	0	0	0	0	1	0
GG	0	0	0	0	0	0	0	0	1

Figure .2: CPT of $QMVT$

POSLEFT	1/3
POSINFRONT	1/3
POSRIGHT	1/3

Figure .3: Prior probability distribution in the variable *POSDSCR* denoting the observation of relative position

LEFT.CROSS	false		true	
RIGHT.CORSS	false	true	false	true
L	0	1	0	0
R	0	0	1	0
G	1	0	0	1

Figure .4: CPT of *OBJ.LC* denoting the probability distributions of OBJ lane change conditioned to the Boolean states of the variables in the logic layer of (OBJ) Vehicle-Lane-Marking relation

OCCGRID	false				true			
TRAJ	false		true		false		true	
LE	false	true	false	true	false	true	false	true
false	1	1	1	0	1	0	1	0
true	0	0	1	1	0	1	0	1

Figure .5: CPT of *(OBJ.LEFT.)CROSS* denoting the probability distribution of crossing the lane marking conditioned to the basic motion hypothesis *LE*, *TRAJ* and *OCCGRID*, (Vehicle-Lane-Marking relation)

.3 Clustering of the Lateral Dynamics

The plots below specify the classification problem of lateral dynamics. Measurement values of $OLAT$ and $VLAT$ have high quality compared to other variables, e.g. PSI and $ALAT$. Since we processed the input data and divided them into clearly defined categories (LC, FOLLOW, FOLLOW_relevant), the boundaries are obvious in the clustering plots, especially in the case of EGO data, which have a very high quality. The challenge is however, that we not only classify the input features, but also accurately predict their evolution. This is the most important requirement, since we want recognize the driving maneuvers at a very early time point, such that the control systems have enough time to react to the critical situations.
The clustering is plotted in the $VLAT$-$OLAT$-axes. Each point in the plot represents a particular configuration. The LE fragment calculates those (parent) configurations and responses a binary (true or false) output denoting the probability of the crossing of the lane marking. The red points represent the labeled data of FOLLOW maneuvers whereas the blue points represent the labeled data of LC maneuvers. Since we consider a dynamic driving process, we want to find methods to predict the development of the lateral dynamics.

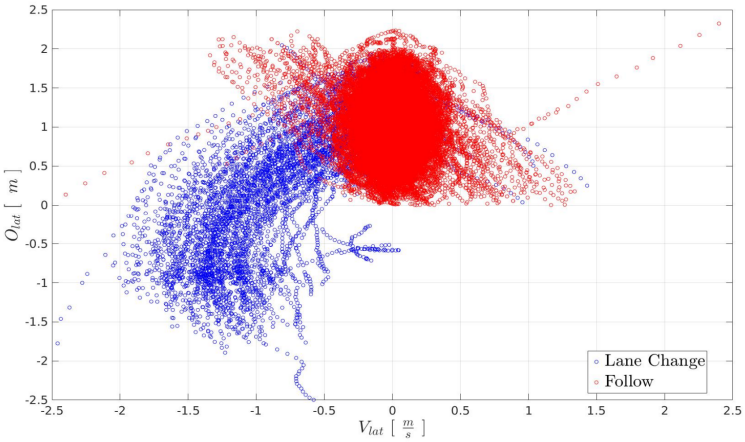


Figure .6: Clustering of lateral offset and velocity using EGO data

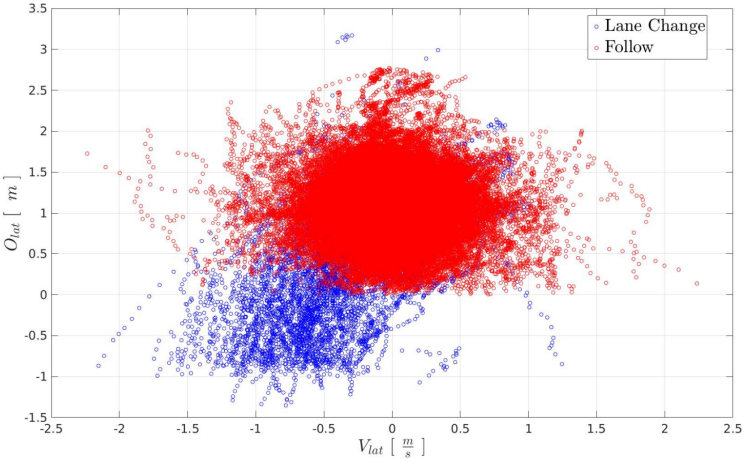
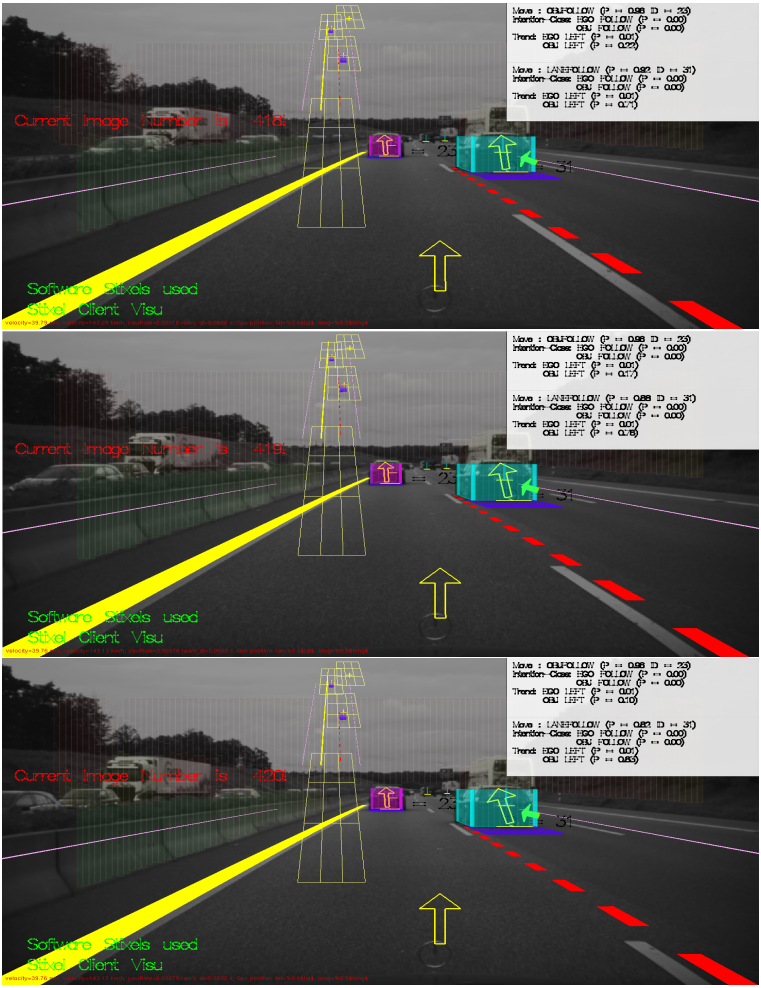
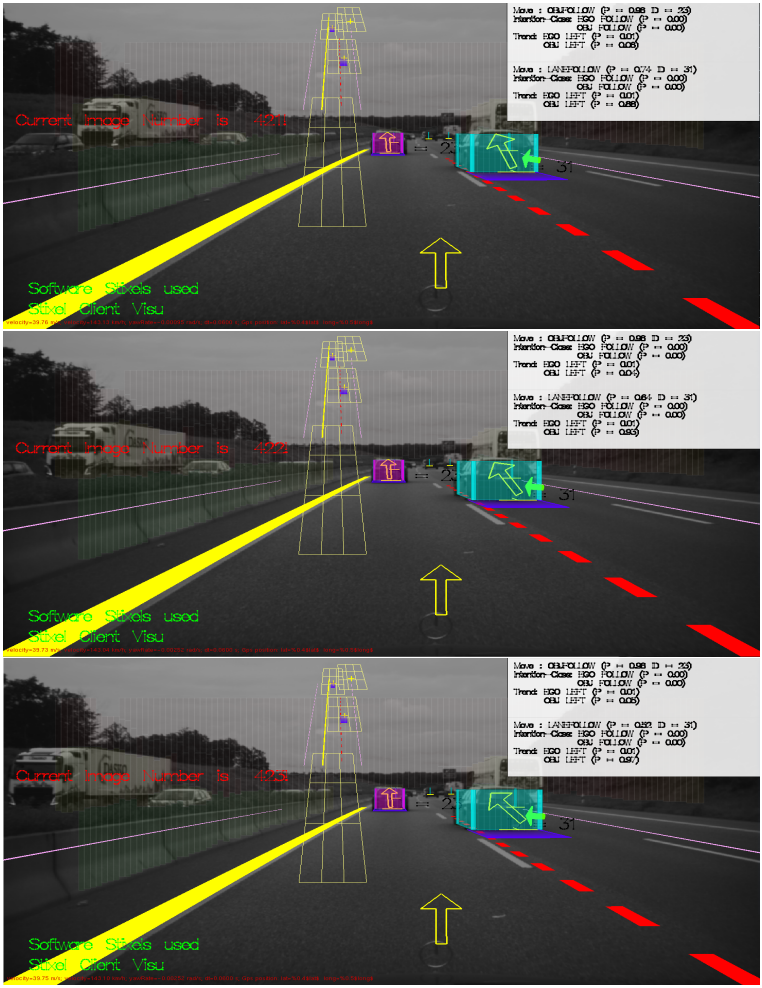


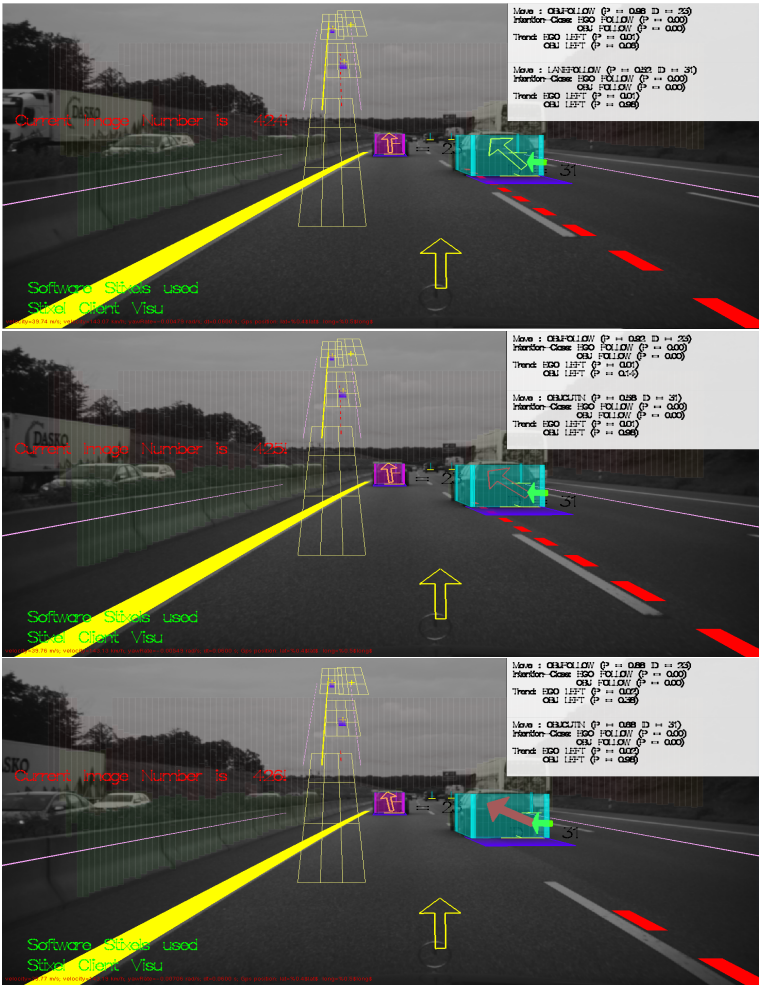
Figure .7: Clustering of lateral offset and velocity using OBJ data

.4 Visual evaluation of the original model

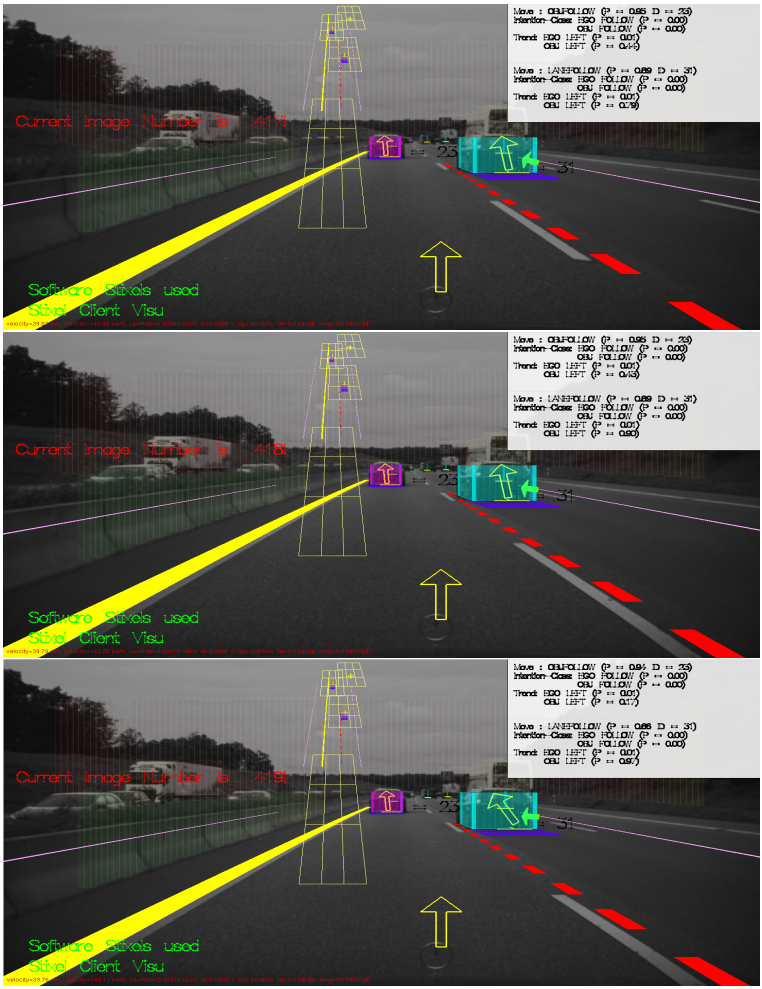
.4.1 ORIG OOBN with original parameters

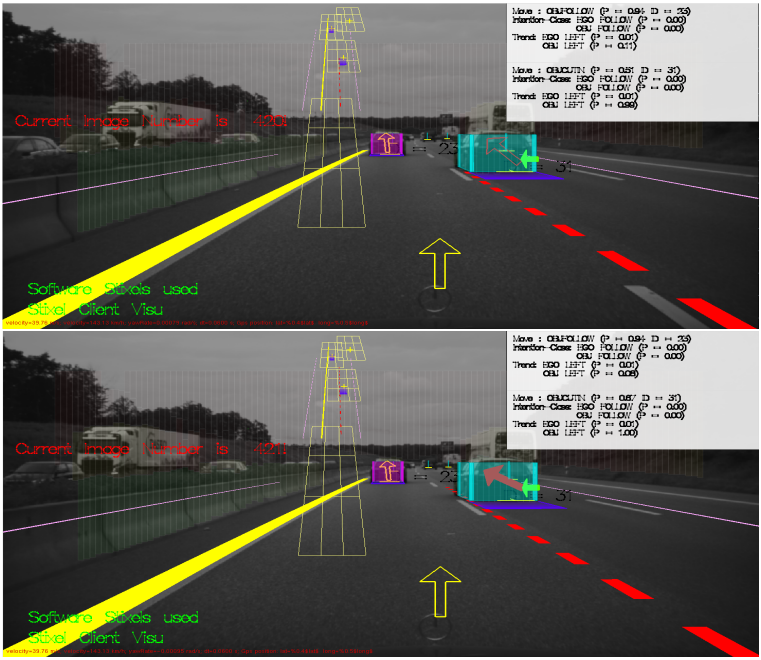



Current Image Number is 4221
Software Stixels used
Stixel Client Visu
Current Image Number is 4222
Software Stixels used
Stixel Client Visu
Current Image Number is 4223
Software Stixels used
Stixel Client Visu


Current Image Number is 492
Software Stixels used
Stixel Client Visu
Current Image Number is 495
Software Stixels used
Stixel Client Visu
Current Image Number is 498
Software Stixels used
Stixel Client Visu

.4.2 ORIG OOBN with optimized parameters




Current Image Number is 4200
Software Stixis used
Stixel Client Visu
Current Image Number is 4211
Software Stixis used
Stixel Client Visu

Eigenständigkeitserklärung

Ich versichere hiermit, dass ich, Stevens RuiXi Wang, die vorliegende Arbeit selbstständig angefertigt, keine anderen als die angegebenen Hilfsmittel benutzt und sowohl wörtliche, als auch sinngemäß entlehnte Stellen als solche kenntlich gemacht habe. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen. Weiterhin bestätige ich, dass das elektronische Exemplar mit den anderen Exemplaren übereinstimmt.

_____ _____

Ort, Datum Unterschrift